



Universidade de Évora - Escola de Ciências e Tecnologia

Mestrado em Engenharia Mecatrónica

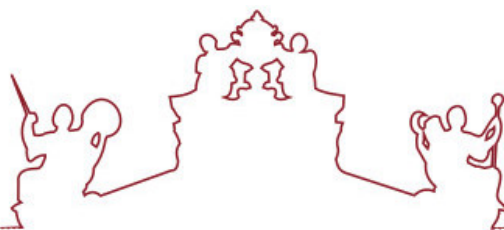
Dissertação

**3D Object Detection and Tracking System Using Distributed
Cameras: Development, Simulation, and Implementation**

Nuno Miguel Pereira Tonim Santos

Orientador(es) | Mouhaydine Tlemcani

Évora 2025



Universidade de Évora - Escola de Ciências e Tecnologia

Mestrado em Engenharia Mecatrónica

Dissertação

3D Object Detection and Tracking System Using Distributed Cameras: Development, Simulation, and Implementation

Nuno Miguel Pereira Tonim Santos

Orientador(es) | Mouhaydine Tlemcani

Évora 2025



A dissertação foi objeto de apreciação e discussão pública pelo seguinte júri nomeado pelo Diretor da Escola de Ciências e Tecnologia:

Presidente | Frederico José Grilo (Universidade de Évora)

Vogais | José Jasnau Caeiro (Escola Superior de Tecnologia e Gestão (ESTIG / IP Beja))
(Arguente)
Mouhaydine Tlemcani (Universidade de Évora) (Orientador)

*Para a minha filha Catarina.
Que sejas sempre a estrela que ilumina o meu caminho.*

*To my daughter Catarina.
May you always be the star that lights my path.*

3D Object Detection and Tracking System Using Distributed Cameras: Development, Simulation, and Implementation

Abstract

This Master Degree in Mechatronics Engineering Dissertation presents a study of a Distributed Tracking Camera System, based on low cost cameras and microprocessor, to achieve 3D location of an IR beacon over a space of interest where there is camera coverage. In order to study the proposed system, mathematical models of camera transformations were inverted to obtain direction vectors defined in the world reference system (to which are the cameras poses themselves referred to) pointing from each camera to the object, and an optimisation algorithm based on Newton Conjugate Gradient to find the location of the object that best matches those vectors. To assess the error of the system, simulations have been developed using simulated cameras to generate the images of the beacon from each camera respective position, orientation, focal length, resolution and sensor size. A prototype of the system and respective software has been built to test the system.

keywords: Artificial Vision, Navigation Systems, Robotics, Image Processing, Remote Visual Localisation

Sistema de detecção e seguimento de objectos em 3D com Recurso a Câmaras Distribuídas: Desenvolvimento, Simulação e Implementação

Resumo

Esta Dissertação de Mestrado em Engenharia Mecatrónica apresenta um estudo de um Sistema de Tracking Distribuído por Câmaras, baseado em câmaras de baixo custo e microprocessador, para localizar em 3D um emissor de infravermelhos (IR beacon) numa área de interesse com cobertura por câmaras. Para estudar o sistema proposto, foram invertidos modelos matemáticos das transformações de câmara, de forma a obter vetores de direcção definidos no sistema de referência do mundo (ao qual se referem as próprias poses das câmaras), apontando de cada câmara para o objeto. Posteriormente, foi utilizado um algoritmo de otimização baseado no Gradiente Conjugado de Newton para encontrar a posição do objeto que melhor se ajusta a esses vetores. Para avaliar o erro do sistema, foram simuladas câmaras, capazes de gerar as imagens do emissor a partir da respetiva posição, orientação, distância focal, resolução e dimensão do sensor de cada câmara. Foi ainda construído um protótipo do sistema, juntamente com o respetivo software, de forma a testar o funcionamento do sistema.

keywords: Visão Artificial, Sistemas de Navegação, Robótica, Processamento de imagem, Localização Visual Remota

Acknowledgments

I would like to express my deepest gratitude to all those who have supported and guided me throughout the development of this work. First and foremost, I am profoundly grateful to my supervisor, Mouhaydine Tlemçani, for his invaluable guidance, insightful feedback, and continuous encouragement. His expertise, patience, and knowledge were instrumental in shaping both the direction and quality of this research.

Second, but not least important, I also address my gratitude to Professor José Lourenço da Saúde, CEiiA Chair in Aerospace Science and Technology Holder, for his continuous support, motivation, and encouragement. Our office discussions about how a robotic system works, how it senses the world, among others, have always, in one way or another, been insightful for me and transpired in this work.

I would also like to acknowledge the contributions of my colleagues, collaborators, and peers, particularly Oumaima Mesbahi, André Albino, and Leonardo Andrade, whose stimulating discussions, constructive suggestions, and moral support greatly enriched this work. Their encouragement and shared experiences have been a constant source of motivation. To Souhila Chabane, Adriano Lino, and Miguel Martins, I also express my gratitude for their support and patience during this time.

I am grateful to the University of Évora, Department of Mechatronics, for providing the facilities, resources, and administrative support that enabled the successful completion of this dissertation and project. I would also like to thank CEiiA Chair in Aerospace Science and Technology for the financial support from PRR Aero.Next Portugal program (projects ILAN and ARL/I&D), which made it possible to pursue this research.

I also appreciate the developers of the software tools and libraries used in this work, including, e.g., Python, MATLAB, OpenCV, etc., whose contributions greatly facilitated the research process.

Finally, I wish to express my heartfelt appreciation to my family and friends for their unwavering support, understanding, and encouragement throughout this troubled journey. Their patience and belief in my abilities were essential in overcoming the challenges encountered along the way.

Contents

List of Figures	xxi
List of Tables	xxiv
1 Introduction	1
1.1 General Aspects	1
1.2 Problematic and motivation behind this work	2
1.3 State of Art in Robotic Tracking Systems	2
1.4 Objectives	3
1.5 Dissertation Structure	3
2 Mathematical Modelling	5
2.1 Camera Transformations	5
2.1.1 Pinhole camera model	5
2.1.2 Forward projection	6
2.1.2.1 World to camera coordinates transformation	6
2.1.2.2 Camera to pixel coordinates transformation	6
2.1.3 Backprojection	7
2.2 Position estimation from an ensemble of cameras	7
2.2.1 Position Estimation	7
2.2.2 Blob detection	10
3 Simulations	15
3.1 Simulation Methodology	15
3.1.1 Test 01 Blender configuration	16
3.1.2 Test 02 Blender configuration	17
3.1.3 Test 03 Blender configuration	17
3.1.4 Test 04 Blender configuration	18
3.1.5 Test 05 Blender configuration	19
3.2 Systematic position estimation tests	20
3.3 Trajectory 1 position estimation tests	22
3.4 Trajectory 2 position estimation tests	23
3.5 Systematic Position Simulation results	24
3.5.1 Analysis of Test 01	24
3.5.2 Analysis of Test 02	31
3.5.3 Analysis of Test 03	37
3.5.4 Analysis of Test 04	43
3.5.5 Analysis of Test 05	49
3.6 Trajectory 1 Simulation results	55
3.6.1 Analysis of Test 01	55
3.6.2 Analysis of Test 02	59
3.6.3 Analysis of Test 03	63
3.6.4 Analysis of Test 04	67
3.7 Trajectory 2 simulation results	72
3.7.1 Analysis of Test 01	72
3.7.2 Analysis of Test 02	76
3.7.3 Analysis of Test 03	80

3.7.4	Analysis of Test 04	84
4	Implementation	89
4.1	Hardware	89
4.1.1	Cameras	89
4.1.2	Beacon	91
4.2	Communication	93
4.3	Distributed Tracking Camera Software	95
4.3.1	Description of the implemented software	95
4.3.2	Communication with Robotic Systems	98
4.4	Real tests of the system	99
4.4.1	Tests setup	99
4.4.2	Pattern for ground truth assessment	101
4.4.3	Accuracy and precision of the system	102
4.4.4	Sampling frequency of the system	109
5	Conclusion	111
5.1	System analysis and applicability	111
5.2	System Current Limitations	111
5.3	Future Work	112
	Bibliography	113
A	Systematic Simulations Datasets	A1
A.1	Test 01	A1
A.2	Test 02	A52
A.3	Test 03	A103
A.4	Test 04	A134
A.5	Test 05	A185

List of Figures

2.1	Pinhole camera model. Source: OpenCV website	5
2.2	Cameras and vectors used in the beacon position finding algorithm, illustrated here in two dimensions.	8
2.3	zoom close to object position, to show the deviation of blob-derived direction vectors around the true object position.	9
2.4	Relation between vectors \mathbf{r}_i , $\hat{\mathbf{v}}_i$ and distance d	9
2.5	Relation between the blob coordinates in image plane and the \mathbf{v} vector used for estimation of camera to object direction.	11
2.6	Example of an image used in the simulation showing a blob (2.6(a), at left), and the corresponding detection marked as a red circular patch, from the blob detection algorithm (2.6(b), at right), for resolution 640×480. In the lower right corner is a zoomed view of the blob and the corresponding detection.	11
2.7	Another example of an image used in the simulation showing a blob (2.7(a), at left), and the corresponding detection marked as a red circular patch, from the blob detection algorithm (2.7(b), at right), now for resolution 1024×768. As in figure 2.6(b), a zoomed view of the blob and corresponding detection is shown at the lower right corner.	11
2.8	Error between blob detection measured coordinates and camera model predicted coordinates in pixels, for x and y and resolution 320×240 (top plots) and corrected pixel coordinates error after equation (2.23) correction to pixel values (bottom).	12
2.9	Error between blob detection measured coordinates and camera model predicted coordinates in pixels, for x and y and resolution 1600×1200 (top plots) and corrected pixel coordinates error after equation (2.23) correction to pixel values (bottom).	13
3.1	Camera sensor and lens data in Blender software.	16
3.2	Camera image resolution defined in Blender.	16
3.3	Camera distribution for test 01, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.1.	16
3.4	Camera distribution for test 02, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.2.	17
3.5	Camera distribution for test 03, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.3.	18
3.6	Camera distribution for test 04, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.4.	19
3.7	Camera distribution for test 05, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.5.	20
3.8	Test positions simulated in the systematic case. The figure was captured from test 01. . .	21
3.9	Absolute error e_{abs} per iteration for resolution 320×240 and test 01.	26
3.10	Absolute error e_{abs} per iteration for resolution 1600×1200 and test 01.	27
3.11	Histogram of error per camera in resolution 320×240 and test 01.	27
3.12	Histogram of error per camera in resolution 1600×1200 and test 01.	28
3.13	Variation of mean error \bar{e}_{abs} in estimated position as a function of number of cameras and resolution used, for test 01.	29
3.14	Estimation error contour map for test 01 with 8 cameras using a resolution of 320×240 pixels, at level $z = 2.00$ m.	29
3.15	Camera coverage for test 01 at level $z = 2.00$ m and resolution 320×240.	29

3.16	Estimation error contour map for test 01 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.	30
3.17	Camera coverage for test 01 at level $z = 2.00$ m and resolution 1600×1200.	30
3.18	Absolute error e_{abs} per iteration for resolution 320×240 and test 02.	31
3.19	Absolute error e_{abs} per iteration for resolution 1600×1200 and test 02.	32
3.20	Histogram of error per camera in resolution 320×240 and test 02.	33
3.21	Histogram of error per camera in resolution 1600×1200 and test 02.	34
3.22	Variation of mean error \bar{e}_{abs} in estimated position as a function of number of cameras and resolution used, for test 02.	35
3.23	Estimation error contour map for test 02 with 8 cameras using a resolution of 320×240 pixels, at level $z = 2.00$ m.	35
3.24	Camera coverage for test 02 at level $z = 2.00$ m.	35
3.25	Estimation error contour map for test 02 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.	36
3.26	Camera coverage for test 02 at level $z = 2.00$ m.	36
3.27	Absolute error e_{abs} per iteration for resolution 320×240 and test 03.	37
3.28	Absolute error e_{abs} per iteration for resolution 1600×1200 and test 03.	38
3.29	Histogram of error per camera in resolution 320×240 and test 03.	39
3.30	Histogram of error per camera in resolution 1600×1200 and test 03.	40
3.31	Variation of mean error \bar{e}_{abs} in estimated position as a function of number of cameras and resolution used, for test 03.	41
3.32	Estimation error contour map for test 03 with 8 cameras using a resolution of 320×240 pixels, at level $z = 2.00$ m.	41
3.33	Camera coverage for test 03 at level $z = 2.00$ m.	41
3.34	Estimation error contour map for test 03 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.	42
3.35	Camera coverage for test 03 at level $z = 2.00$ m.	42
3.36	Absolute error e_{abs} per iteration for resolution 320×240 and test 04.	43
3.37	Absolute error e_{abs} per iteration for resolution 1600×1200 and test 04.	44
3.38	Histogram of error per camera in resolution 320×240 and test 04.	45
3.39	Histogram of error per camera in resolution 1600×1200 and test 04.	46
3.40	Variation of mean absolute error \bar{e}_{abs} in estimated position as a function of number of cameras and resolution used, for test 04.	47
3.41	Estimation error contour map for test 04 with 8 cameras using a resolution of 320×240 pixels, at level $z = 2.00$ m.	47
3.42	Camera coverage for test 04 at level $z = 2.00$ m.	47
3.43	Estimation error contour map for test 04 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.	48
3.44	Camera coverage for test 04 at level $z = 2.00$ m.	48
3.45	Absolute error e_{abs} per iteration for resolution 320×240 and test 05.	49
3.46	Absolute error e_{abs} per iteration for resolution 1600×1200 and test 05.	50
3.47	Histogram of error per camera in resolution 320×240 and test 05.	51
3.48	Histogram of error per camera in resolution 1600×1200 and test 05.	52
3.49	Variation of mean error \bar{e}_{abs} in estimated position as a function of number of cameras detecting the beacon and resolution used, for test 05.	53
3.50	Estimation error contour map for test 05 with 8 cameras using a resolution of 320×240 pixels, at level $z = 2.00$ m.	53
3.51	Camera coverage for test 05 at level $z = 2.00$ m.	53
3.52	Estimation error contour map for test 05 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.	54
3.53	Camera coverage for test 05 at level $z = 2.00$ m.	54
3.54	Trajectory 1 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 01	55
3.55	Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 01.	56
3.56	Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 01.	57
3.57	Real error in position estimation for trajectory 1 at resolution 320×240 and test 01, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	57

3.58	Real error in position estimation for trajectory 1 at resolution 1600×1200 and test 01, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	58
3.59	Error distribution histogram for trajectory 1 and test 01 at resolution 320×240.	58
3.60	Error distribution histogram for trajectory 1 and test 01 at resolution 1600×1200.	58
3.61	Trajectory 1 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 02	59
3.62	Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 02.	60
3.63	Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 02.	61
3.64	Real error in position estimation for trajectory 1 at resolution 320×240 and test 02, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	61
3.65	Real error in position estimation for trajectory 2 at resolution 1600×1200 and test 02, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	62
3.66	Error distribution histogram for trajectory 1 and test 02 at resolution 320×240.	62
3.67	Error distribution histogram for trajectory 1 and test 02 at resolution 1600×1200.	62
3.68	Trajectory 1 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 03	63
3.69	Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 03.	64
3.70	Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 03.	65
3.71	Real error in position estimation for trajectory 1 at resolution 320×240 and test 03, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	65
3.72	Real error in position estimation for trajectory 1 at resolution 1600×1200 and test 03, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	66
3.73	Error distribution histogram for trajectory 1 and test 03 at resolution 320×240.	66
3.74	Error distribution histogram for trajectory 1 and test 03 at resolution 1600×1200.	66
3.75	Trajectory 1 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 04	67
3.76	Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 04.	68
3.77	Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 04.	69
3.78	Real error in position estimation for trajectory 1 at resolution 320×240 and test 04, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	70
3.79	Real error in position estimation for trajectory 1 at resolution 1600×1200 and test 04, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	70
3.80	Error distribution histogram for trajectory 1 and test 04 at resolution 320×240.	70
3.81	Error distribution histogram for trajectory 1 and test 04 at resolution 1600×1200.	71
3.82	Trajectory 2 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 01	72
3.83	Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 01.	73
3.84	Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 01.	74
3.85	Real error in position estimation for trajectory 2 at resolution 320×240 and test 01, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	74
3.86	Real error in position estimation for trajectory 2 at resolution 1600×1200 and test 01, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	75
3.87	Error distribution histogram for trajectory 2 and test 01 at resolution 320×240.	75
3.88	Error distribution histogram for trajectory 2 and test 01 at resolution 1600×1200.	75
3.89	Trajectory 2 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 02	76
3.90	Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 02.	77
3.91	Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 02.	78

3.92	Real error in position estimation for trajectory 2 at resolution 320×240 and test 02, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	78
3.93	Real error in position estimation for trajectory 2 at resolution 1600×1200 and test 02, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	79
3.94	Error distribution histogram for trajectory 2 and test 02 at resolution 320×240.	79
3.95	Error distribution histogram for trajectory 2 and test 02 at resolution 1600×1200.	79
3.96	Trajectory 2 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 03	80
3.97	Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 03.	81
3.98	Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 03.	82
3.99	Real error in position estimation for trajectory 2 at resolution 320×240 and test 03, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	82
3.100	Real error in position estimation for trajectory 2 at resolution 1600×1200 and test 03, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	83
3.101	Error distribution histogram for trajectory 2 and test 03 at resolution 320×240.	83
3.102	Error distribution histogram for trajectory 2 and test 03 at resolution 1600×1200.	83
3.103	Trajectory 2 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 04	84
3.104	Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 04.	85
3.105	Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 04.	86
3.106	Real error in position estimation for trajectory 2 at resolution 320×240 and test 04, for x coordinate (top), y coordinate (centre) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	86
3.107	Real error in position estimation for trajectory 2 at resolution 1600×1200 and test 04, for x coordinate (top), y coordinate (centre) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.	87
3.108	Error distribution histogram for trajectory 2 and test 04 at resolution 320×240.	87
3.109	Error distribution histogram for trajectory 2 and test 04 at resolution 1600×1200.	87
4.1	ESP32-CAM used in this work (a), with sensor OV2640 attached. Also in the image are the expansion board (b) and the external antenna (c).	90
4.2	3D printed camera casing (a, b) and support (c). Also shown in the image are the support base (d), the filter holder (e), and the IR filter (f).	90
4.3	Spectral response of image sensor OV2640, adapted from the sensor data sheet.	90
4.4	IR Beacon schematic circuit used to develop the PCB	92
4.5	Top layer of the IR Beacon PCB	92
4.6	Bottom layer of the IR Beacon PCB	92
4.7	IR Beacon built for this project - top side	92
4.8	IR Beacon built for this project - bottom side	92
4.9	Camera access point screen, where the user can insert the details of the Wi-Fi network desired to be used, the Wi-Fi password and the name to be given to the camera, as viewed in a mobile device.	93
4.10	Camera access point screen as in figure 4.9, but from a computer.	93
4.11	Camera control screen after successful connection of camera to Wi-Fi network.	94
4.12	Main window of the Distributed Camera System software developed in Python, using PyQt5.	95
4.13	DTCS “Add New Camera” menu.’	96
4.14	DTCS “Preferences” menu.	96
4.15	Zoom into the camera controls tab, displaying some of its interfaces to the remote camera	96
4.16	Main window of the Distributed Camera System software with cameras connected and tracking a blob, estimating its position.	97
4.17	Zoom into the Main window of the Distributed Camera System software, showing the “Status” bar with cameras connected and tracking a blob, estimating its position. Cameras are also displaying their predicted vectors.	97

4.18	Output of the socket client when connected to the DTCS embedded server. It shows the beacon position being transmitted over the socket connection.	98
4.19	Laboratory set-up for the Distributed Camera System testing.	100
4.20	One of the cameras set up for live tests.	100
4.21	A0-sized checkerboard used as a reference for beacon coordinate measurement, to provide a ground truth for comparison with the output of the system.	100
4.22	Measurements calibration paper sheet created to assist in the definition of a ground truth for the system.	101
4.23	Calibration sheet zoomed in to show the indication of the axis defined and the scale (in cm).	101
4.24	Calibration sheet zoomed in to show the indication of the coordinate system origin and some alignment markers, to help identify coordinate directions at a distance.	101
4.25	Positions used for accuracy and precision assessment of the system over the space defined by the pattern described above (see 4.4.2). Also shown with different marks and colours are the mean estimated positions by the system, with its radius scaled proportionally to 2 times its standard deviation.	103
4.26	Top row: Position 1 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	104
4.27	Top row: Position 2 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	104
4.28	Top row: Position 3 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	105
4.29	Top row: Position 4 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	105
4.30	Top row: Position 5 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	106
4.31	Top row: Position 6 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	106
4.32	Top row: Position 7 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	107
4.33	Top row: Position 8 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	107
4.34	Top row: Position 9 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	108
4.35	Top row: Position 10 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.	108
4.36	Framerate of the system during live test. From top to bottom are presented the framerate for all tests and zoomed-in plots of framerate for test positions 1, 2, 5 and 10, respectively.	109

List of Tables

1.1	Compact comparison between Tracking Systems	3
1.2	Advantages and limitations of different Tracking Systems	3
2.1	Blob detection linear correction coefficients	13
3.1	Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 01.	17
3.2	Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 02.	18
3.3	Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 03.	18
3.4	Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 04.	19
3.5	Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 05.	20
3.6	Combinations of camera positions (tests), resolutions, field of view, number of cameras and number of beacon positions simulated in the systematic case.	21
3.7	Combinations of camera positions (tests), resolutions, field of view, number of cameras and number of beacon positions simulated in the trajectory 1 case.	22
3.8	Combinations of camera positions (tests), resolutions, field of view, number of cameras and number of beacon positions simulated in the trajectory 2 case.	23
3.9	Global statistics and root mean square error for test 01 with resolution 320×240.	25
3.10	Global statistics and root mean square error for test 01 with resolution 1600×1200.	25
3.11	Statistics per number of cameras observing the beacon for test 01 with resolution 320×240.	25
3.12	Statistics per number of cameras observing the beacon for test 01 with resolution 1600×1200.	25
3.13	Weibull PDF parameters and bin count per camera used in figure 3.11 for test 01 and resolution 320×240.	28
3.14	Weibull PDF parameters and bin count per camera used in figure 3.12 for test 01 and resolution 1600×1200.	28
3.15	Global statistics and root mean square error for test 02 with resolution 320×240.	32
3.16	Global statistics and root mean square error for test 02 with resolution 1600×1200.	32
3.17	Statistics per number of cameras observing the beacon for test 02 with resolution 320×240.	33
3.18	Statistics per number of cameras observing the beacon for test 02 with resolution 1600×1200.	33
3.19	Weibull PDF parameters and bin count per camera used in figure 3.20 for test 02 and resolution 320×240.	34
3.20	Weibull PDF parameters and bin count per camera used in figure 3.21 for test 02 and resolution 1600×1200.	34
3.21	Global statistics and root mean square error for test 03 with resolution 320×240.	38
3.22	Global statistics and root mean square error for test 03 with resolution 1600×1200.	38
3.23	Statistics per number of cameras observing the beacon for test 03 with resolution 320×240.	39
3.24	Statistics per number of cameras observing the beacon for test 03 with resolution 1600×1200.	39
3.25	Weibull PDF parameters and bin count per camera used in figure 3.29 for test 03 and resolution 320×240.	40
3.26	Weibull PDF parameters and bin count per camera used in figure 3.30 for test 03 and resolution 1600×1200.	40
3.27	Global statistics and root mean square error for test 04 with resolution 320×240.	44

3.28	Global statistics and root mean square error for test 04 with resolution 1600×1200.	44
3.29	Statistics per number of cameras observing the beacon for test 04 with resolution 320×240.	45
3.30	Statistics per number of cameras observing the beacon for test 04 with resolution 1600×1200.	45
3.31	Weibull PDF parameters and bin count per camera used in figure 3.38 for test 04 and resolution 320×240.	46
3.32	Weibull PDF parameters and bin count per camera used in figure 3.39 for test 04 and resolution 1600×1200.	46
3.33	Global statistics and root mean square error for test 05 with resolution 320×240.	50
3.34	Global statistics and root mean square error for test 05 with resolution 1600×1200.	50
3.35	Statistics per number of cameras observing the beacon for test 05 with resolution 320×240.	51
3.36	Statistics per number of cameras observing the beacon for test 05 with resolution 1600×1200.	51
3.37	Weibull PDF parameters and bin count per camera used in figure 3.47 for test 05 and resolution 320×240.	52
3.38	Weibull PDF parameters and bin count per camera used in figure 3.48 for test 05 and resolution 1600×1200.	52
3.39	Global statistics and root mean square error for test 01 with resolution 320×240.	56
3.40	Global statistics and root mean square error for test 01 with resolution 1600×1200.	56
3.41	Weibull PDF parameters and bin count per camera used in figure 3.59 for test 01 and resolution 320×240.	56
3.42	Weibull PDF parameters and bin count per camera used in figure 3.60 for test 01 and resolution 1600×1200.	57
3.43	Global statistics and root mean square error for test 02 with resolution 320×240.	60
3.44	Global statistics and root mean square error for test 02 with resolution 1600×1200.	60
3.45	Weibull PDF parameters and bin count per camera used in figure 3.66 for test 02 and resolution 320×240.	60
3.46	Weibull PDF parameters and bin count per camera used in figure 3.67 for test 02 and resolution 1600×1200.	60
3.47	Global statistics and root mean square error for test 03 with resolution 320×240.	64
3.48	Global statistics and root mean square error for test 03 with resolution 1600×1200.	64
3.49	Weibull PDF parameters and bin count per camera used in figure 3.73 for test 03 and resolution 320×240.	64
3.50	Weibull PDF parameters and bin count per camera used in figure 3.74 for test 03 and resolution 1600×1200.	64
3.51	Global statistics and root mean square error for test 04 with resolution 320×240.	68
3.52	Global statistics and root mean square error for test 04 with resolution 1600×1200.	68
3.53	Weibull PDF parameters and bin count per camera used in figure 3.80 for test 04 and resolution 320×240.	69
3.54	Weibull PDF parameters and bin count per camera used in figure 3.81 for test 04 and resolution 1600×1200.	69
3.55	Global statistics and root mean square error for test 01 with resolution 320×240.	73
3.56	Global statistics and root mean square error for test 01 with resolution 1600×1200.	73
3.57	Weibull PDF parameters and bin count per camera used in figure 3.87 for test 01 and resolution 320×240.	73
3.58	Weibull PDF parameters and bin count per camera used in figure 3.88 for test 01 and resolution 1600×1200.	74
3.59	Global statistics and root mean square error for test 02 with resolution 320×240.	77
3.60	Global statistics and root mean square error for test 02 with resolution 1600×1200.	77
3.61	Weibull PDF parameters and bin count per camera used in figure 3.94 for test 02 and resolution 320×240.	77
3.62	Weibull PDF parameters and bin count per camera used in figure 3.95 for test 02 and resolution 1600×1200.	78
3.63	Global statistics and root mean square error for test 03 with resolution 320×240.	81
3.64	Global statistics and root mean square error for test 03 with resolution 1600×1200.	81
3.65	Weibull PDF parameters and bin count per camera used in figure 3.101 for test 03 and resolution 320×240.	81
3.66	Weibull PDF parameters and bin count per camera used in figure 3.102 for test 03 and resolution 1600×1200.	82
3.67	Global statistics and root mean square error for test 04 with resolution 320×240.	85

3.68	Global statistics and root mean square error for test 04 with resolution 1600×1200.	85
3.69	Weibull PDF parameters and bin count per camera used in figure 3.108 for test 04 and resolution 320×240.	85
3.70	Weibull PDF parameters and bin count per camera used in figure 3.109 for test 04 and resolution 1600×1200.	86
4.1	Technical specifications of the ESP32-CAM	90
4.2	Technical specifications of the OV2640 image sensor	91
4.3	Camera measured positions used in the laboratory testing of the system.	99
4.4	Test positions used for accuracy and precision testing of the system.	102
4.5	Test positions tested in the live tests for accuracy and precision assessment. Each line contains the real position of the beacon, the estimated mean and standard deviation of the beacon position estimate, the number of points used (counts) and the estimated accuracy for each test	103

Chapter 1

Introduction

1.1 General Aspects

In today's world, the use of robotic systems is expanding from the industrial realm into everyday life aspects. From robotic vacuum cleaners to autonomous driving vehicles, regardless of their applications, robotic systems must rely on ways of sensing the world to navigate it. They are, for example, especially important in remote robotic systems, like for example in robotic systems exploring other planets, like Mars [1].

To accomplish the task of navigating around an environment, autonomous robotic systems must be able to determine their position, orientation, and the existence of walls, objects, or other obstacles in their path. The path to follow can be either a predefined path, planned for some task, or an adaptive path, meaning there is no predefined plan; the system has to be able to generate its own path, based on algorithms that use data from the robotic system's integrated sensors. To achieve autonomous navigation, several techniques have been developed over the last few years. They go from traditional graph-based methods like A* path finding or Dijkstra, or like cell decomposition and Voronoi diagrams in the case of geometric techniques. Besides traditional methods, there is also the category of meta-heuristic algorithms, which can be, for example, the Genetic algorithm or the Particle Swarm optimisation. More advanced methods can integrate machine learning or neural networks into these traditional techniques, giving rise to the field of Hybrid models. A general review about these subjects can be found in [2, 3].

These allow us to find a trajectory that the robot should follow, given that its position and orientation (pose) are known in relation to the reference system that defines the world around it. In the simplest form, this position and orientation can be found by using Inertial sensors as a way to achieve it, but given the nature of Inertial sensors, both the accelerometer and gyroscope will accumulate errors, and their values will drift over time, providing inaccurate positions to the system, and requiring an existing map of the space around.

Other navigation methods focus on vision, using different sensors and SLAM (Simultaneous Localisation and Mapping) algorithms [4, 5], to autonomously adapt to the environment around [6]. Vision in this sense can be achieved by using either typical cameras as a sensor or a LiDAR as a way of mapping space. More advanced techniques incorporate the fusion of Visual-LiDAR-inertial sensors to provide a more detailed and complementary view of the space around the system and its position in it. Adding AI into the fusion models [7] has also become more and more common in recent years, providing a way of minimising the errors in the system mapping and also generating more accurate maps.

However, all the aforementioned methods run embedded into the robotic system itself, and all of them require a high processing power to solve the pose of the system in relation to the world. Due to this issue, many times it is useful to have an external system, able to provide a more accurate position and orientation to the system, like, for example, what GPS does to an aircraft autopilot. The autopilot relies on its sensors to know its velocity and altitude, and ground VOR, but is assisted with GPS tracking its position over the planet. But in case of indoor tracking or in locations where a GPS signal is not available for some reason, other tracking methods are necessary.

Some of the external tracing systems will be reviewed in more detail in the next section 1.3, but they

usually require exploring the existing Wifi networks' signal, the existence of ultrasonic beacons, or cameras able to track the system.

This work presents a visual tracking system based on a distribution of cameras over a region of interest, with the main focus of being used indoors, providing location and navigation data to a robotic system that does not rely on GPS or equivalent systems. The proposed system in this work should be able to assist autonomous robotic systems in a given space, providing position estimates of high precision.

1.2 Problematic and motivation behind this work

This work focuses on the external tracking of a robotic system using remote cameras, in calibrated positions, to estimate its position over time. This is a part of a series of systems being studied in a project from University of Évora's CEiiA Chair in Aerospace Science and Technology currently ongoing, namely ILAN-VR (Automated & Integrated Large aircraft NDI Assisted by Virtual Reality), which aims to conduct automated aircraft inspections using autonomous drones, using indoor navigation given that the system should not rely on GPS signals. There are two main problems to solve: for one, to find the pose of the aircraft inside the hangar, and for the second, the position of the drone in relation to the same hangar, to obtain the relative position of the drone in relation to the aircraft.

These problems are currently being treated as separate problems, and it can be deduced that this work is related to the second problem, finding the drone position in relation to a hangar-defined reference frame. To assist with that task, the CEiiA Chair team has been conducting tests using systems as LiDAR to map the space around the drone (as an internal system to the drone) and MarvelMind beacons [8] as external reference frames and to find the position of the drone. The first has proven to be, for now, computationally expensive, which made the team focus on the Marvelmind beacons as a way of tracking the drone. More on this system will be presented in the next section 1.3, but it has also been sometimes unreliable, mainly because it seems to be susceptible to interference. Still, most of the time it gives positions with good precision. There are systems based on visual methods using cameras, which provide position estimates with high precision, but these systems are expensive, which would increase the overall costs of the project. Examples of such systems are Vicon [9] and OptiTrack [10]. More on this system is also on section 1.3.

So, in this work, a possible solution using cameras to do visual tracking is studied, focusing in the use of regular consumer cameras, sensitive in the infrared part of the electromagnetic spectrum, as an alternative solution to those higher-end camera solutions. This study involved developing our own position estimation mathematical model, algorithm and software for the cameras, simulating it, studying the errors in the system and finally implementing a prototype version for testing.

1.3 State of Art in Robotic Tracking Systems

Here, some of the possible solutions to solve the external position referencing of the robotic system are presented. Several methods can be used or even combined to provide reliable position estimates [11].

Among these solutions, there is Ultra Wide Band (UWB) [12, 13, 14, 15] which uses long wavelength radio waves to estimate Time of Flight (ToF) or Time of Arrival (ToA) of a signal between two anchors, enabling the estimation of positions up to a accuracy of 10 30 cm if there is Line Of Sight (LOS) between the anchors. Depending on the accuracy required, this may be a good solution given its low cost and low latency. However, its signal may be degraded by the presence of large obstacles or even large walls where it is operating, meaning, if there is Non Line Of Sight (NLOS).

Also measuring ToF, there is the Marvelmind tracking system [8], created for indoor use, which makes use of ultrasounds to estimate distances between the anchors and one or more beacons installed on the robotic system. Its precision can reach typical values between ± 2 cm in the horizontal plane and ± 5 cm in the vertical direction. The manufacturer specifies a range from 30 m to 50 m if beacons are in line-of-sight, with an update frequency up to 100 Hz.

On the Visual products, there are Vicon and OptiTrack, as examples. Vicon [9] uses high-speed infrared (IR) cameras to track a passive or an active IR (reflective or emissive, respectively) beacon installed on the system to be tracked. OptiTrack [10] works in a similar fashion to Vicon, and both systems can reach a precision of sub-millimetres, but require a controlled environment to work at their full capabilities.

A study of the performance of Vicon camera system has been presented in [16], where it was found that Vicon can provide static position estimates with a mean absolute error of 0.15 mm and a variability within 0.025 mm. Depending on the moving velocity of the beacon being tracked, it can reach an absolute error smaller than 0.3 mm, provided that the sampling frequency is correctly chosen.

Table 1.1 presents the working principle of each system and its main properties. On Table 1.2, the advantages and limitations of each system.

Table 1.1: Compact comparison between Tracking Systems

System	Technology	Accuracy	Update Rate	Range
Vicon	Optical (IR)	< 1 mm	up to 1000 Hz	Room with cameras
OptiTrack	Optical (IR)	0.1–1 mm	up to 360 Hz	Room with cameras
Marvelmind	Ultrasonic + RF	2–5 cm	8–100 Hz	up to 50 m
UWB	Radio (Ultra-Wideband)	10–30 cm (LOS); up to 0.5 m (NLOS)	10–100 Hz	up to 100 m (indoor)

Table 1.2: Advantages and limitations of different Tracking Systems

System	Advantages	Limitations
Vicon	Very high accuracy; industrial and academic reference standard	Very expensive; sensitive to occlusions
OptiTrack	High accuracy, lower cost than Vicon	Requires line of sight; careful calibration needed
Marvelmind	Low cost; does not depend on vision	Affected by ultrasonic noise; degraded in NLOS
UWB	Moderate cost; works with moderate obstacles; robust to light/noise	Lower accuracy than optical systems; degraded in NLOS; requires multiple anchors

1.4 Objectives

The main objectives of this study are to develop a mathematical model of the position recovery based on the perspective of many cameras of a given object to be tracked remotely, to simulate the mathematical model to assess the feasibility, precision and accuracy of this proposal and to implement a demonstration version of the system working.

As secondary objectives, it is planned that the system should be simple and easy to implement physically, and its use should be fairly intuitive. It should also be affordable in terms of equipment used, scalable, and capable of providing the required precision for tracking a robotic system accurately.

1.5 Dissertation Structure

This dissertation is divided into 5 chapters. The first chapter (this one), contains an introduction to the idea behind this work, the problems associated with navigation and tracking of robotic systems and a review of the state of the art in these fields. Other systems of this kind, even if their functioning principles are based on other physical properties (i.e., not based on visual methods), are compared in that section. In this chapter, there is also a presentation of the main objectives to be developed or implemented on the system, namely the mathematical development of the position estimation model, the simulation of cameras and of the system and also the physical implementation of a prototype. Finally, chapter 1 ends with this section presenting the dissertation structure.

Chapter 2 contains the mathematical model of this work, starting in section 2.1 with the camera transformations that model the projection of an object in 3D space onto the image plane of a camera and

the inverse process of recovering a view direction to the object from its projection in the image plane. section 2.2 presents the mathematics behind the position estimation optimisation process and also gives details on how the blob detection works in order to detect the object in the image.

Chapter 3 presents all the simulated tests developed to study the response of the system and to assess the effect of resolution and spatial distribution of cameras on the position estimation process. This is the longest of all Chapters, because after presenting all the simulations developed, it continues with the analysis of the simulated results for each scenario tested.

Chapter 4 describes the development of a prototype system to test in real-life scenarios the proposed methodology. It describes the cameras chosen to implement the system, the IR beacon developed and the software that computes the position of the beacon, based on the cameras' images. It also describes the communication between the cameras and the computer and between the tracking software and robotic systems.

Finally, chapter 5 presents the main conclusions of this dissertation, an analysis of the applicability of the system and its current limitations. There are also references to future work to be developed that can improve the system or provide new functionality to it.

Chapter 2

Mathematical Modelling

2.1 Camera Transformations

2.1.1 Pinhole camera model

The camera model first explored in this work is the simplest one, the pinhole camera model, which describes the image formation on the image plane when light goes through a small opening, the pinhole, hence the name. Pinhole cameras are known since ancient times, with historical mentions from the Greeks, in the Aristotelian *Problems* dating from around 300-600 BC. This camera model is the simplest one, mainly because it does not need a lens to work, with the model using only the fact that light travels in a straight line and the similarity of triangles to generate an image. In the next pages, the model is mathematically described, presenting how the forward projection (subsection 2.1.2) transforms the coordinates of an object in the real world into image coordinates in the camera system of reference. In section 2.1.3, it is also presented how to recover the real-world direction vector from the camera to an object in 3D space.

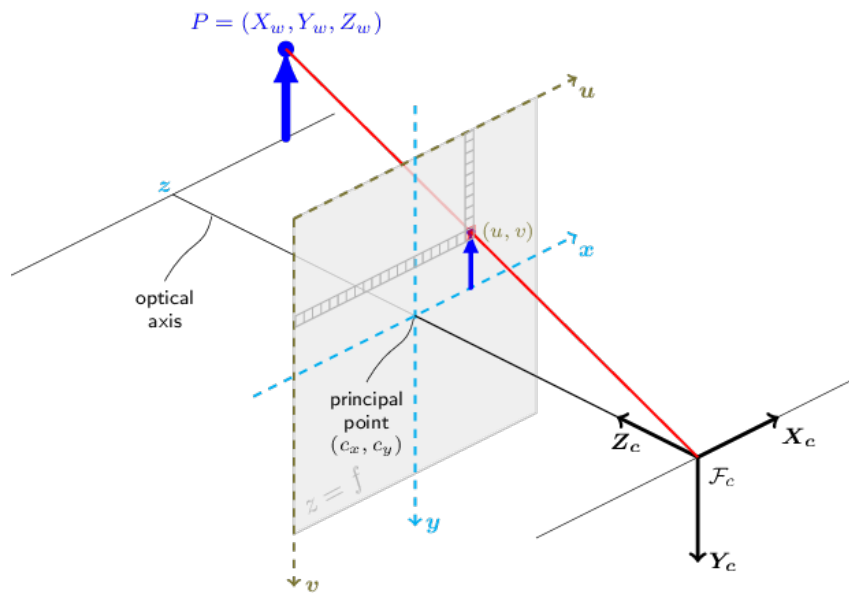


Figure 2.1: Pinhole camera model. Source: OpenCV website

Figure 2.1 [17] shows the relations between an object world position relative to the camera, and its projection onto the image plane of the camera (either a film or a sensor). These relations will be further explored in the next section 2.1.2.

2.1.2 Forward projection

The process of modelling the projection of an object in 3D space onto the image plane of a camera is referred to in this work as forward projection, following the same approach as presented in [18]. To introduce the subject, the first process is to obtain camera coordinates from an object's world coordinates. That process is described below.

2.1.2.1 World to camera coordinates transformation

Let $\mathbf{P}_w = [X_w, Y_w, Z_w, 1]^T$ denote the homogeneous coordinates of an object in a world coordinate system \mathbf{W} with an arbitrary origin, and the position of a camera \mathbf{C}_w to be given by $\mathbf{C}_w = [X_w^c, Y_w^c, Z_w^c, 1]^T$ in the same reference system.

From these two positions, we can derive the object position $\mathbf{P}_c = [X_c, Y_c, Z_c]^T$ from the camera reference frame, which is coincident with the camera optical axis. The camera generates an image of the object in the camera focal plane, given by its "film coordinates" $\mathbf{P}_f = [x, y, 1]$, usually in mm. The process of modelling the camera projection of an object from world coordinates to film coordinates, and later to image coordinates (pixels) is described by the following transformations.

Transforming the position of the object from world coordinates \mathbf{P}_w to camera coordinates \mathbf{P}_c is accomplished using equation (2.1):

$$\mathbf{P}_c = \mathbf{R}(\mathbf{P}_w - \mathbf{C}_w) \quad (2.1)$$

where \mathbf{R} is the camera rotation matrix, which can be described as the set of rotations of the camera optical centre axis in relation to \mathbf{W} :

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma) \quad (2.2)$$

where α is the angle of rotation around the world z axis, β is the rotation angle of the camera around the world y axis and γ is the rotation angle around the world x axis. These three angles describe the orientation of the camera in the world coordinate system.

2.1.2.2 Camera to pixel coordinates transformation

Transforming from camera coordinates \mathbf{P}_c to film coordinates $\mathbf{P}_f = [x, y, 1]^T$ requires a projection of the scene onto the camera image plane. To achieve this, a projection matrix \mathbf{K} is used:

$$\mathbf{P}_f = \frac{1}{Z_c} \mathbf{K} \mathbf{P}_c \quad (2.3)$$

where \mathbf{K} is given by:

$$\mathbf{K} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (2.4)$$

and f is the focal length of the camera lens in mm.

The film coordinates \mathbf{P}_f given above are related to film photography, and would locate the projection of the real-world object onto the image plane in mm. But currently most images are taken using digital cameras, and positions in images are usually given in pixel coordinates $\mathbf{P}_i = [u, v, 1]^T$. To achieve the transformation from film coordinates to pixel coordinates, an affine transformation matrix \mathbf{A} is used in equation (2.5).

$$\mathbf{P}_i = \mathbf{A} \mathbf{P}_f \quad (2.5)$$

where the affine matrix \mathbf{A} is given by:

$$\mathbf{A} = \begin{bmatrix} -\frac{s_w}{w} & 0 & o_x \\ 0 & \frac{s_h}{h} & o_y \\ 0 & 0 & 1 \end{bmatrix} \quad (2.6)$$

where s_w and s_h are the camera sensor width and height, respectively, in mm, w and h the horizontal and vertical pixel dimensions of the sensor and o_x and o_y the image horizontal and vertical centre coordinates in pixels.

2.1.3 Backprojection

This section presents the process of recovering the direction vector of a point in the image plane, from the camera position, referred to the world coordinate system \mathbf{W} . The direction vector recovered this way is the basis for the proposed tracking system described on Section 2.2.

Let $\mathbf{v} = [v_x, v_y, v_z]^T$ be the direction vector of the point $\mathbf{P} = [u, v, 1]^T$ in the image plane, measured from the camera position, defined to be coincident with the focal plane of the camera. To find \mathbf{v} , the following transformations are applied, starting with the estimation of film coordinates \mathbf{P}_f^* from pixel coordinates \mathbf{P} , using equation (2.7):

$$\mathbf{P}_f^* = [x^*, y^*, 1]^T = \mathbf{A}^{-1}\mathbf{P} \quad (2.7)$$

and from the estimated film coordinates \mathbf{P}_f^* , \mathbf{v} can be estimated by Equation (2.8):

$$\mathbf{v} = [v_x, v_y, v_z]^T = -\mathbf{R}(\mathbf{P}_f^* - [0, 0, 1 - f]^T) \quad (2.8)$$

However, the normalised $\hat{\mathbf{v}}$ is much more useful for the calculations, which is easily achieved by dividing the \mathbf{v} vector by its norm:

$$\hat{\mathbf{v}} = \frac{\mathbf{v}}{\|\mathbf{v}\|} \quad (2.9)$$

2.2 Position estimation from an ensemble of cameras

This section presents the algorithm developed to estimate the 3-D position of an object in world coordinates, based on the images from an ensemble of cameras, distributed in the space of interest.

The main idea is to use n cameras (with $n \geq 2$) to track an object as it moves in space. The object (which can be a robotic system) should be fitted with an infrared beacon, which is going to be detected by the cameras. Blob detection is then applied to the n cameras' images, and image positions of the beacon are extracted for each image in parallel. The subsection 2.2.2 gives the details on this process. After estimating the beacon position in each image, data from all the cameras can be combined to estimate the intersection in space of the direction vectors \mathbf{v}_i of each camera. Because all the cameras are "seeing" the object, their direction vector must converge somewhere in space. In the following subsection 2.2.1, this process is described in higher detail.

2.2.1 Position Estimation

Figure 2.2 shows an example of a distribution of cameras, covering an area where an object equipped with an IR beacon. Let \mathbf{r}_i be the estimated position vector of the object O at coordinates \mathbf{W} in relation to camera C_i (where i goes from 2 to n , with n the number of cameras available) located at coordinates \mathbf{C}_i (as described in section 2.1.2, referred to a "world" coordinate system arbitrarily defined, as shown in section 2.1, equation (2.1), such that, where i is an integer from 1 to n cameras, where \mathbf{W} is the unknown position of the object:

$$\mathbf{r}_i = \mathbf{W} - \mathbf{C}_i, \quad i \in \{2, 3, \dots, n\} \quad (2.10)$$

Vector \mathbf{v}_i is the direction from camera C_i to O as estimated from the respective image and camera pose and can be calculated by backprojecting the pixel coordinates and focal distance of camera lens f , as

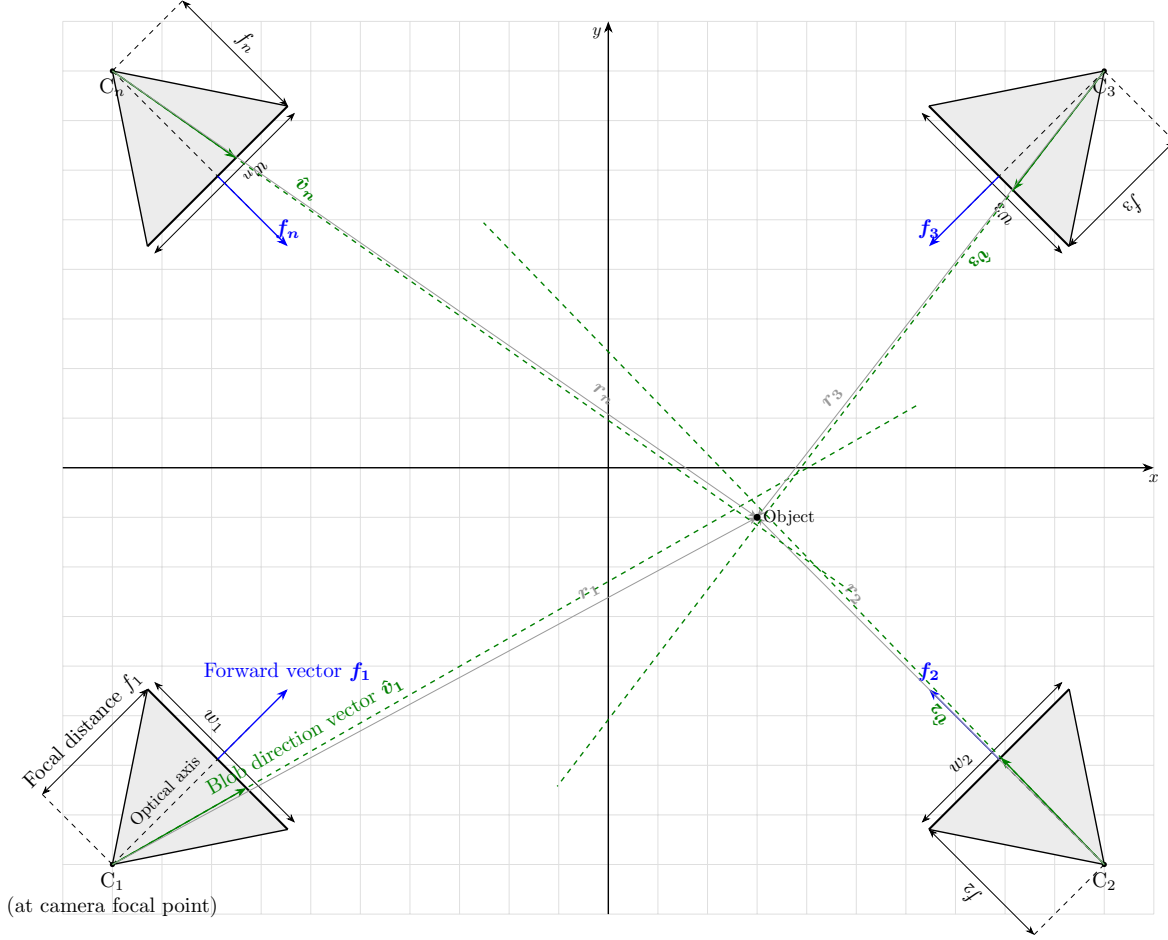


Figure 2.2: Cameras and vectors used in the beacon position finding algorithm, illustrated here in two dimensions.

given by Equation (2.8). The components of \mathbf{v}_i are then referred to in the same “world” coordinate system \mathbf{W} used to define camera poses.

From linear algebra it is known that the magnitude of the cross product of two vectors \mathbf{a} and \mathbf{b} gives the area of the parallelogram formed by these two vectors, as long as the vectors are not parallel, or zero magnitude, in which case the cross product gives a zero vector, and trivially its magnitude is also zero. In the case that the vectors aren’t parallel or zero magnitude, the following general relationship applies:

$$\|\mathbf{a} \times \mathbf{b}\| = \|\mathbf{a}\| \|\mathbf{b}\| \sin \theta \quad (2.11)$$

Replacing in our case \mathbf{a} with \mathbf{r}_i and \mathbf{b} with $\hat{\mathbf{v}}_i$, it becomes:

$$\|\mathbf{r}_i \times \hat{\mathbf{v}}_i\| = \|\mathbf{r}_i\| \|\hat{\mathbf{v}}_i\| \sin \theta \quad (2.12)$$

And noting that $\hat{\mathbf{v}}_i$ is normalised and substituting

$$\|\mathbf{r}_i\| \sin \theta = d_i \quad (2.13)$$

where θ is the angle between \mathbf{a} and \mathbf{b} , and d_i is the projected distance between \mathbf{r}_i and $\hat{\mathbf{v}}_i$.

Substituting equation (2.13) and $\|\hat{\mathbf{v}}_i\| = 1$ into equation (2.12), it is obtained the following relation:

$$\|\mathbf{r}_i \times \hat{\mathbf{v}}_i\| = d_i \quad (2.14)$$

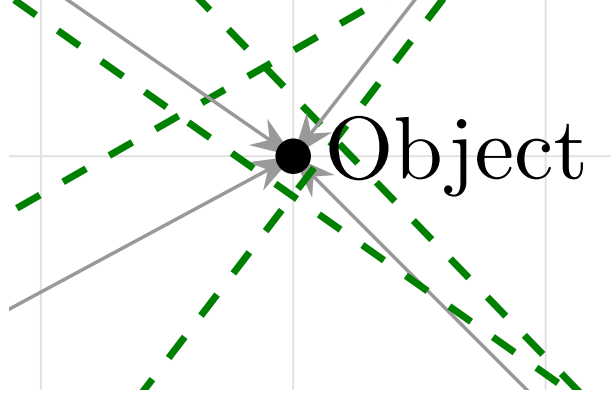


Figure 2.3: zoom close to object position, to show the deviation of blob-derived direction vectors around the true object position.

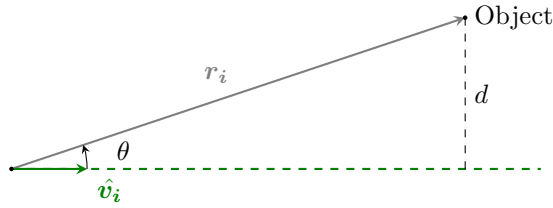


Figure 2.4: Relation between vectors \mathbf{r}_i , $\hat{\mathbf{v}}_i$ and distance d .

$$\|\mathbf{r}_i \times \hat{\mathbf{v}}_i\|^2 = d_i^2 \quad (2.15)$$

Equation (2.14) is illustrated in figure 2.4. To estimate the position of the object using the n cameras available, we proceed by computing the sum of all the distances for all the cameras using equation (2.16).

$$S = \sum_{i=2}^n d_i^2, \quad n \geq 2 \quad (2.16)$$

In the case of an ideal alignment of the estimated direction vector $\hat{\mathbf{v}}_i$ and the estimated position vector \mathbf{r}_i , the corresponding $d_i = 0$, because the vectors are coincident. Therefore, in the ideal case, the sum S should be zero ($S = 0$). But due to measurement and systematic errors in the process, in real applications, the sum S will always be greater than zero, reaching a minimum value as the alignment error between \mathbf{r}_i and $\hat{\mathbf{v}}_i$ approaches 0. So, the process of estimating the position of the object based on the direction $\hat{\mathbf{v}}_i$ each camera observes it, is accomplished by finding the vector \mathbf{r}_i that minimises the equation (2.16), which is easily done by setting :

$$\min S : \vec{\nabla} S = 0 \quad (2.17)$$

$$\vec{\nabla} S = \begin{cases} \frac{\partial S}{\partial x} &= \sum_{i=0}^n 2v_y^i [v_y^i (x - x_c^i) - v_x^i (y - y_c^i)] + 2v_z^i [v_z^i (x - x_c^i) - v_x^i (z - z_c^i)] \\ \frac{\partial S}{\partial y} &= \sum_{i=0}^n 2v_z^i [v_z^i (y - y_c^i) - v_y^i (z - z_c^i)] - 2v_x^i [v_y^i (x - x_c^i) - v_x^i (y - y_c^i)] \\ \frac{\partial S}{\partial z} &= \sum_{i=0}^n -2v_x^i [v_z^i (x - x_c^i) - v_x^i (z - z_c^i)] - 2v_y^i [v_z^i (y - y_c^i) - v_y^i (z - z_c^i)] \end{cases} \quad (2.18)$$

The gradient $\vec{\nabla} S$ above is minimised numerically using the Newton Conjugate Gradient method, implemented in the SciPy Python library [19].

2.2.2 Blob detection

The tracking algorithm developed in this work requires the estimation of the direction vector \mathbf{v}_i based on the projection of the beacon onto the image plane. To achieve this, the beacon is detected in the image using a blob detection algorithm, based on the functions implemented in the OpenCV Python library. Multiple blobs are detected on this process, but as a first approach to the problem, only the one with maximum intensity is measured, and its image coordinates and size are output by the algorithm.

There are several ways of detecting blobs, but in this work, we used the SimpleBlobDetector from OpenCV, which is based on geometric and intensity filtering. More sophisticated approaches, like Differences of Gaussians (DoG), used in SIFT - Scale invariant feature transform, another computer vision algorithm to detect and match local features of images [20], were not tested in this work. In the following lines, the SimpleBlobDetector algorithm is described.

A blob is, in mathematical terms, a connected region where neighbouring pixels have similar values of intensity (or colour, depending on the application) and that is distinguishable from the local background. An example can simply be a dot in a black background or even a shape (which can be geometrical, like a circle or an ellipsoid or on the other hand, arbitrary in shape).

To detect the blob, the first step is to convert the image to grey-scale and then to binarise it, accordingly to equation (2.19)

$$I_{bin}(x, y) = \begin{cases} 1, & \text{if } I(x, y) > T \\ 0, & \text{if } I(x, y) \leq T \end{cases} \quad (2.19)$$

where $I(x, y)$ is the pixel intensity in the original image and T is a limit intensity threshold, usually defined by the user but can also be computed automatically, depending on the application. In our case, we opted for manually defining a threshold based on detection tests done on some of the simulation images.

Then, the algorithm searches for connected components, grouping adjacent binarised pixels. This is achieved by using connected components algorithms, from which a set of components \mathbf{C} is generated:

$$\mathbf{C} = \{C_1, C_2, \dots, C_n\} \quad (2.20)$$

From each potential blob on this list of components \mathbf{C} , several properties are calculated, such as area, circularity, convexity and inertia (also known as ellipticity ratio). For each of these properties, the blob is accepted if it falls into thresholds defined by the user; otherwise, it is rejected. In our case, we chose to use Area and Circularity filtering.

After having the filtered blobs, its centre (x_c, y_c) , can be determined using equation (2.21):

$$p_{C_i} = (x_c, y_c) = \left(\frac{1}{|C_i|} \sum_{(x,y) \in C_i} x, \frac{1}{|C_i|} \sum_{(x,y) \in C_i} y \right) \quad (2.21)$$

The centre coordinates of the blob are then used in this work as the blob coordinates p . From the set of blobs in the components \mathbf{C} , the one with the highest intensity (which in our case is basically area) is chosen as the blob being tracked.

From the blob coordinates, the vector \mathbf{v}_i can then be determined following the procedure already discussed when presenting the Backprojection, using equation (2.7). The relation between the blob pixel coordinates and the \mathbf{v} vector can also be seen in figure 2.5. A few images from the simulation and respective blob detection are presented in figures 2.6 for resolution 640×480 and in figures 2.7 for resolution 1024×768.

It was noted that the blob detection implemented suffered from a systematic error in the determination of the image coordinates when compared to the expected coordinates using the camera's mathematical model. It was also noted that this systematic deviation showed a linear trend, which enabled the fitting of a linear correction to the measured blob coordinates. This correction is also dependent on the resolution used, with no correlation with the camera's configuration on each test.

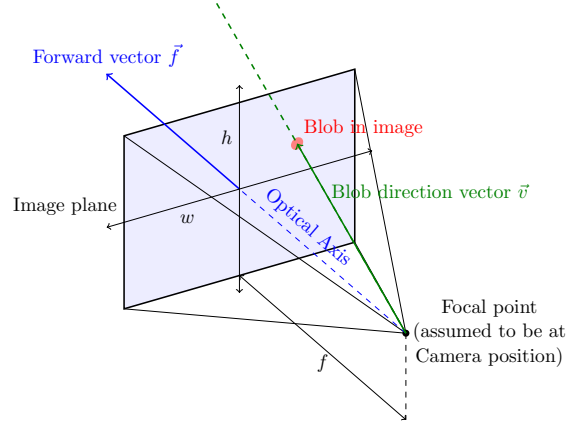


Figure 2.5: Relation between the blob coordinates in image plane and the \mathbf{v} vector used for estimation of camera to object direction.

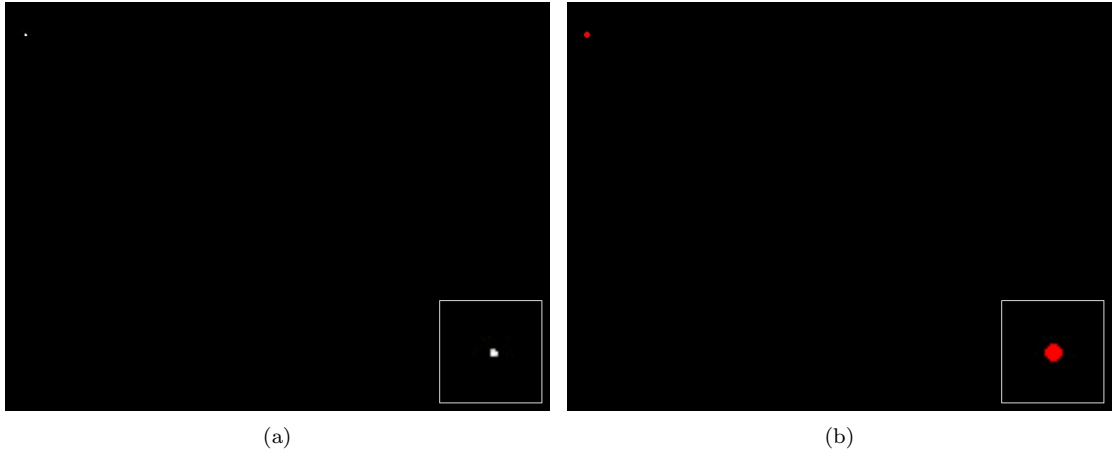


Figure 2.6: Example of an image used in the simulation showing a blob (2.6(a), at left), and the corresponding detection marked as a red circular patch, from the blob detection algorithm (2.6(b), at right), for resolution 640×480. In the lower right corner is a zoomed view of the blob and the corresponding detection.

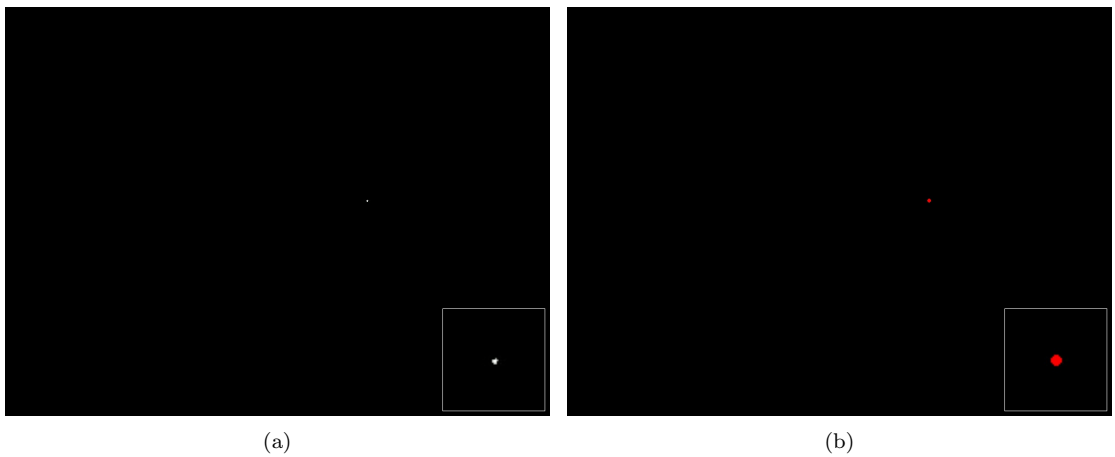


Figure 2.7: Another example of an image used in the simulation showing a blob (2.7(a), at left), and the corresponding detection marked as a red circular patch, from the blob detection algorithm (2.7(b), at right), now for resolution 1024×768. As in figure 2.6(b), a zoomed view of the blob and corresponding detection is shown at the lower right corner.

The plots below (figures 2.8 and 2.9) show the systematic deviation on the blob detection for resolution 320×240 and for resolution 1600×1200, as an example. Table 2.1 presents the linear coefficients fitted to the blob position distribution that was then used to correct the blob estimated coordinates for each resolution.

The regression of these values was achieved by fitting a linear relation of the form:

$$e(u) = e(u_0) + s_u u \quad (2.22)$$

where u is the coordinate being evaluated (in our case u can be either x or y pixel coordinates), $e(u)$ is the error at coordinate u , $e(u_0)$ is the error at origin ($u_0 = 0$) in our case, and s_u is the slope of the fitted linear relation for coordinate u .

From equation (2.22), it is possible to correct the values of measured blob positions, using the following equation (2.23):

$$p_{C_i}^* = (x_c^*, y_c^*)_i = \begin{cases} x_c^* = x_{c_i} - e(x) \\ y_c^* = y_{c_i} - e(y) \end{cases} \quad (2.23)$$

where $p_{C_i}^* = (x_c^*, y_c^*)$ is the pixel corrected coordinates.

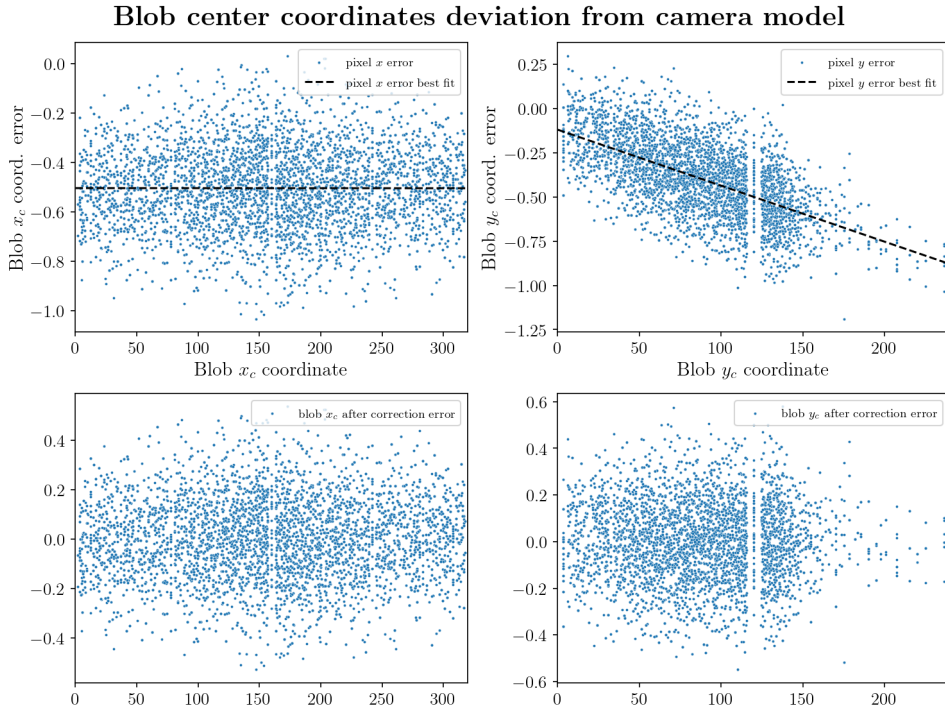


Figure 2.8: Error between blob detection measured coordinates and camera model predicted coordinates in pixels, for x and y and resolution 320×240 (top plots) and corrected pixel coordinates error after equation (2.23) correction to pixel values (bottom).

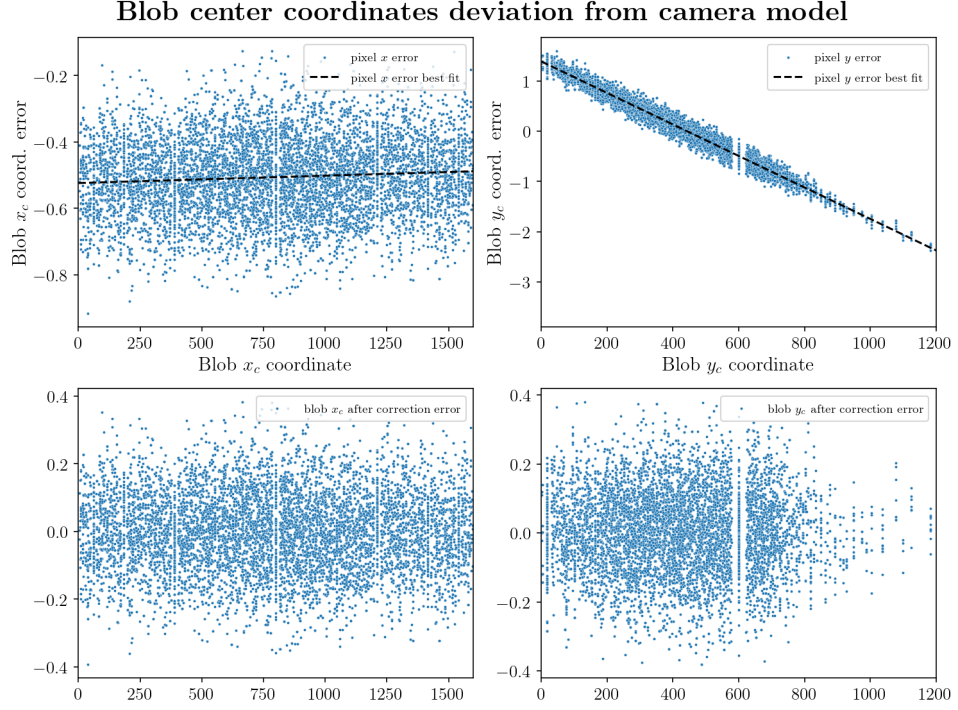


Figure 2.9: Error between blob detection measured coordinates and camera model predicted coordinates in pixels, for x and y and resolution 1600×1200 (top plots) and corrected pixel coordinates error after equation (2.23) correction to pixel values (bottom).

Table 2.1: Blob detection linear correction coefficients

Resolution	e_{x_0}	s_x	e_{y_0}	s_y
320×240	-0.5037	-1.6962e-06	-0.1190	-3.1697e-03
640×480	-0.5176	4.0765e-05	0.2604	-3.1325e-03
800×600	-0.5072	7.9498e-06	0.4378	-3.0950e-03
1024×768	-0.5078	6.5107e-06	0.7166	-3.1486e-03
1600×1200	-0.5236	2.2132e-05	1.3879	-3.1339e-03

Chapter 3

Simulations

This chapter describes the simulations of cameras developed to study the response of the proposed system. To achieve this goal, simulations of different distributions of cameras were simulated using Blender software [21] (see section 3.1), from which the scenes were rendered into frames, for each camera. These frames are then used as a simulated feed to the algorithm implemented in this work (described in section 2.2). The results of these simulations are analysed in higher detail in subsections 3.5, 3.6 and 3.7. Using this methodology, tests were done using a systematic scanning of the space of interest, and two trajectories (tilted circular path - trajectory 1, figure eight - trajectory 2) were also studied and analysed.

3.1 Simulation Methodology

For the purpose of simulating the camera images, Blender [21] was used. Blender is a freeware and open-source 3D modelling and animation software, capable of simulating visual effects, such as lighting, shadows, textures, among other effects, and can simulate the views of cameras of the rendered scenes, which can be used as a single image or combined into a sequence of frames to generate a movie. Due to the ease of use, combined with support for Python [22] scripts, Blender proved to be a very useful tool in generating the simulations.

In this study, the camera chosen to be studied and simulated was a cheap consumer camera, OV2640, usually available to integrate with ESP-32 micro-controllers and Raspberry Pi microcomputers. The OV2640 sensor and lens data (based on the available data-sheet [23]) were set up in Blender's camera (fig. 3.1) and lens data (fig. 3.2).

The camera's position and orientation are defined in relation to an arbitrary reference frame. The position is defined by a position vector $\mathbf{r} = [x, y, z]$ and the orientation by three Euler angles, namely α , the rotation around the reference frame x axis, β , the rotation around the y axis and γ , the rotation around the z axis.

Then, for each distribution of cameras, a beacon was simulated. For that purpose, a sphere model with a 5 cm diameter, with an emissive material applied, was used. The position of the beacon was then modified via a Python script, and images from the perspective of each simulated camera were captured. The frames for each camera/position were then saved and used to generate movies of the moving beacon, and also stored in a zip file for each corresponding camera, for further analysis.

In the systematic simulation, the beacon position was modified systematically, scanning the entire study space. Depending on the simulation, the beacon was moved by 0,5 m (Tests 01, 02, 03 and 04) or by 1 m (Test 05), with each case detailed in the respective test description.

In the trajectory simulations, the beacon followed a predefined trajectory, being a circular path tilted by an angle in relation to the horizontal plane of the first cases (trajectory 1) and an eight-figure in the other case (trajectory 2).

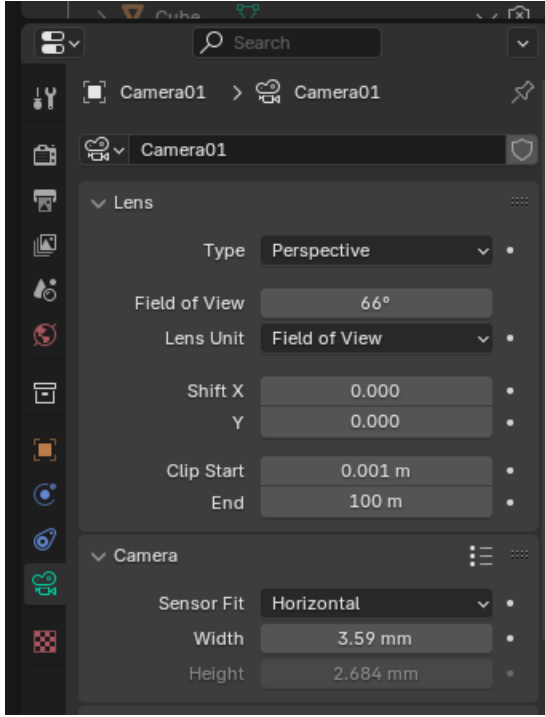


Figure 3.1: Camera sensor and lens data in Blender software.

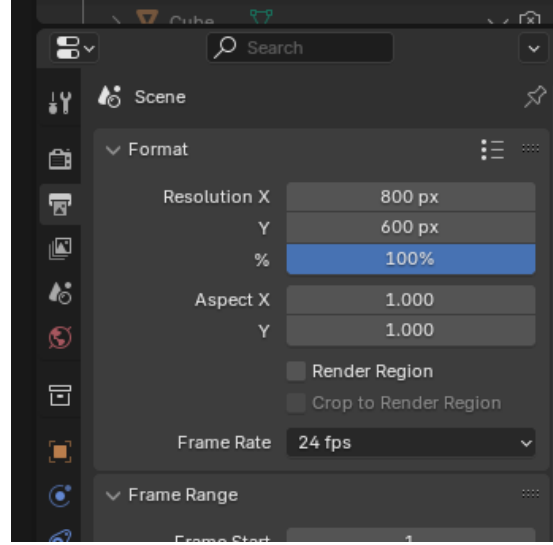


Figure 3.2: Camera image resolution defined in Blender.

3.1.1 Test 01 Blender configuration

In test 01, cameras are distributed in a square pattern, with a 10 m side length. 4 cameras are set up along the vertices of the square, and the other 4 are in the middle of the edges. All cameras are looking at the centre of the space. The coordinate axis is defined at the centre of the space, and the positions of the cameras in relation to this coordinate axis are presented in table 3.1. In the same table, the angles that define the orientation of the camera are also presented. Figure 3.3 shows a screenshot of the Blender scene with the camera setup.

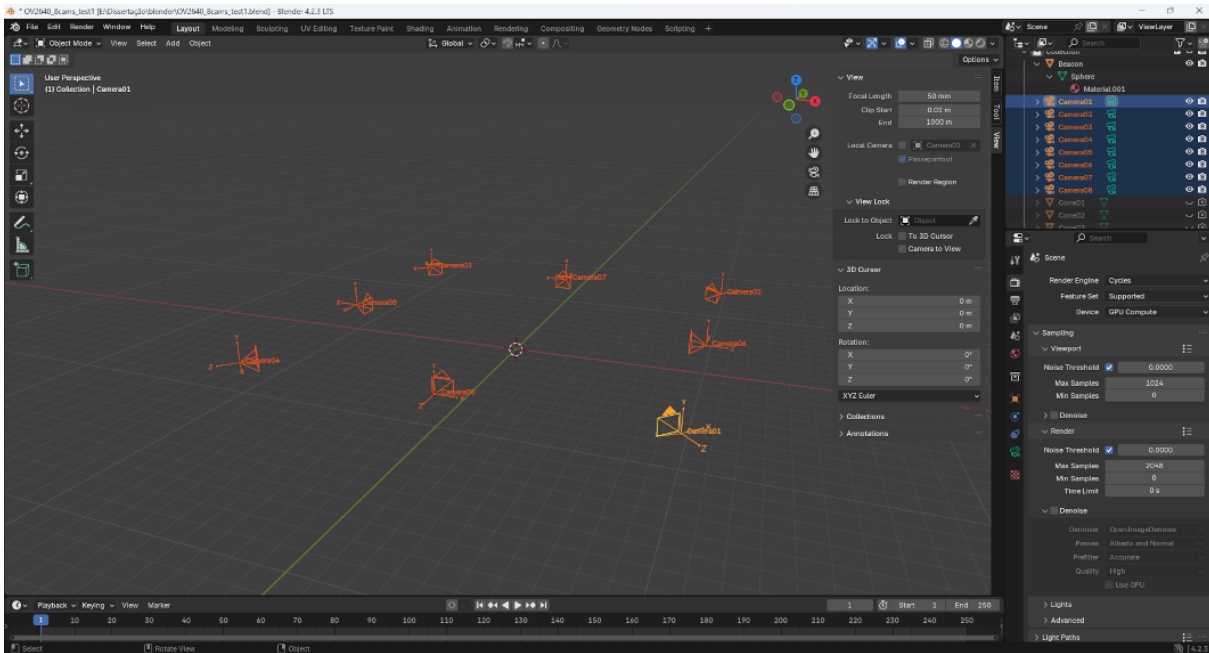


Figure 3.3: Camera distribution for test 01, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.1.

Table 3.1: Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 01.

Camera	x	y	z	α	β	γ	f	FOV
01	10.0000	-10.0000	1.5000	90.000	0.000	45.000	2.764	66.0
02	10.0000	10.0000	1.5000	90.000	0.000	135.000	2.764	66.0
03	-10.0000	10.0000	1.5000	90.000	0.000	225.000	2.764	66.0
04	-10.0000	-10.0000	1.5000	90.000	0.000	315.000	2.764	66.0
05	0.0000	-10.0000	1.5000	90.000	0.000	0.000	2.764	66.0
06	10.0000	0.0000	1.5000	90.000	0.000	90.000	2.764	66.0
07	0.0000	10.0000	1.5000	90.000	0.000	180.000	2.764	66.0
08	-10.0000	0.0000	1.5000	90.000	0.000	270.000	2.764	66.0

3.1.2 Test 02 Blender configuration

As in test 01, cameras are distributed in test 02 in a square pattern. However, in this case, all cameras are located at the vertices of the square pattern, only differing in their heights and orientations. 4 cameras are in a lower position compared to the other 4. Their orientations are such that one vertical face (side) of the field of view frustum is aligned (parallel) with the edge of the square pattern along which the cameras are positioned, as can be seen in the screenshot presented in Figure 3.4. Table 3.2 shows in more detail the positions and orientations of the cameras.

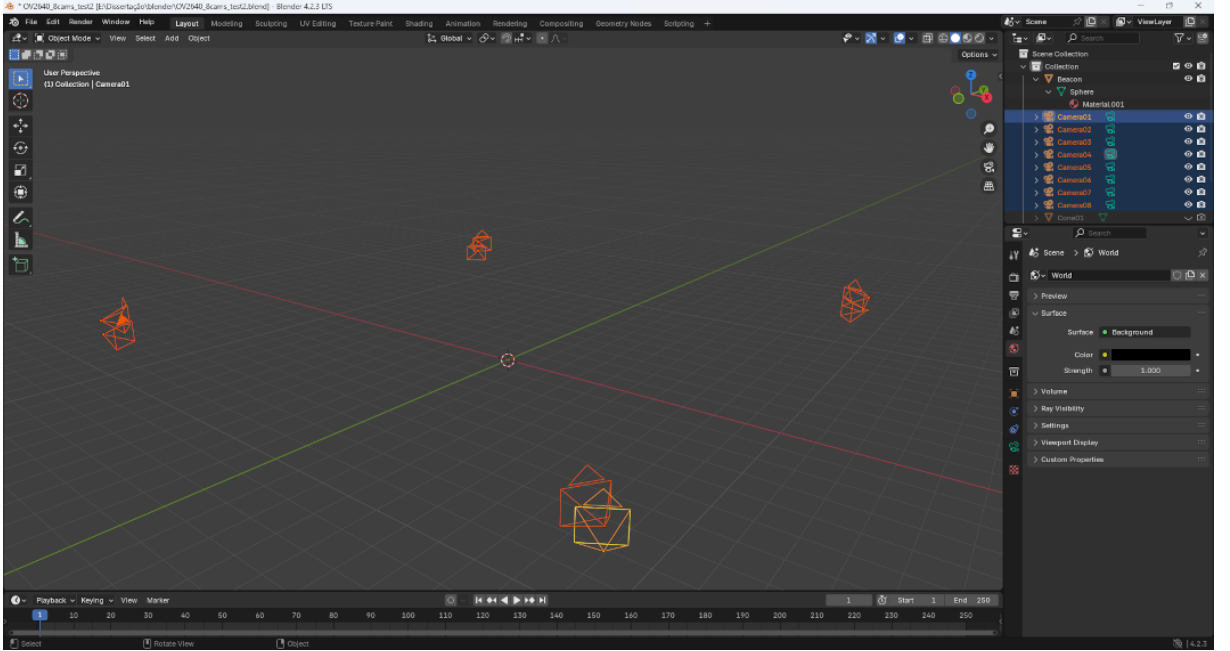


Figure 3.4: Camera distribution for test 02, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.2.

3.1.3 Test 03 Blender configuration

In test 03, the cameras follow the same positional pattern as test 01, with the major difference being their vertical position and their orientation. In this test, cameras are located at a height of 5 m and tilted in such a way that their top edge of the field of view is parallel to the plane with $z = 5$ m. This test provides a way of assessing the tracking algorithm when the cameras are located in a high position, looking down onto the workspace. Table 3.3 gives the cameras' location and orientations on this test. Fig. 3.5 shows a screenshot from Blender with the camera setup for this test.

Table 3.2: Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 02.

Camera	x	y	z	α	β	γ	f	FOV
01	10.0000	-10.0000	1.5000	90.000	0.000	33.000	2.764	66.0
02	10.0000	10.0000	1.5000	90.000	0.000	123.000	2.764	66.0
03	-10.0000	10.0000	1.5000	90.000	0.000	213.000	2.764	66.0
04	-10.0000	-10.0000	1.5000	90.000	0.000	303.000	2.764	66.0
05	10.0000	-10.0000	2.0000	90.000	0.000	57.000	2.764	66.0
06	10.0000	10.0000	2.0000	90.000	0.000	147.000	2.764	66.0
07	-10.0000	10.0000	2.0000	90.000	0.000	237.000	2.764	66.0
08	-10.0000	-10.0000	2.0000	90.000	0.000	327.000	2.764	66.0

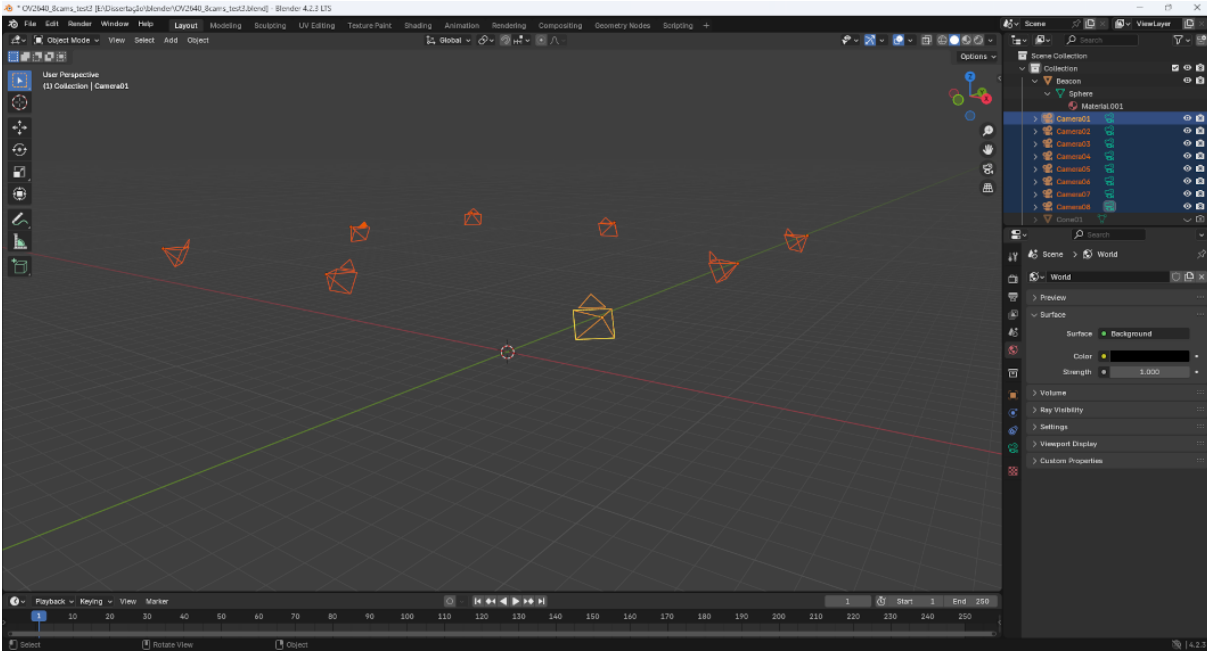


Figure 3.5: Camera distribution for test 03, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.3.

Table 3.3: Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 03.

Camera	x	y	z	α	β	γ	f	FOV
01	10.0000	-10.0000	5.0000	64.000	0.000	45.000	2.764	66.0
02	10.0000	10.0000	5.0000	64.000	0.000	135.000	2.764	66.0
03	-10.0000	10.0000	5.0000	64.000	0.000	225.000	2.764	66.0
04	-10.0000	-10.0000	5.0000	64.000	0.000	315.000	2.764	66.0
05	0.0000	-10.0000	5.0000	64.000	0.000	0.000	2.764	66.0
06	10.0000	0.0000	5.0000	64.000	0.000	90.000	2.764	66.0
07	0.0000	10.0000	5.0000	64.000	0.000	180.000	2.764	66.0
08	-10.0000	0.0000	5.0000	64.000	0.000	270.000	2.764	66.0

3.1.4 Test 04 Blender configuration

In test 04, the cameras follow the same positional pattern of test 03, but are now located at a height of 10 m and tilted in such a way that their top edge of the field of view is parallel to the plane with $z = 10$ m. This test provides a way of assessing the tracking algorithm when the cameras are located at an even higher position compared to test 03, and also looking down onto the workspace. Table 3.4 gives

the camera's location and orientations on this test. Fig. 3.6 shows a screenshot from Blender with the camera setup for this test.

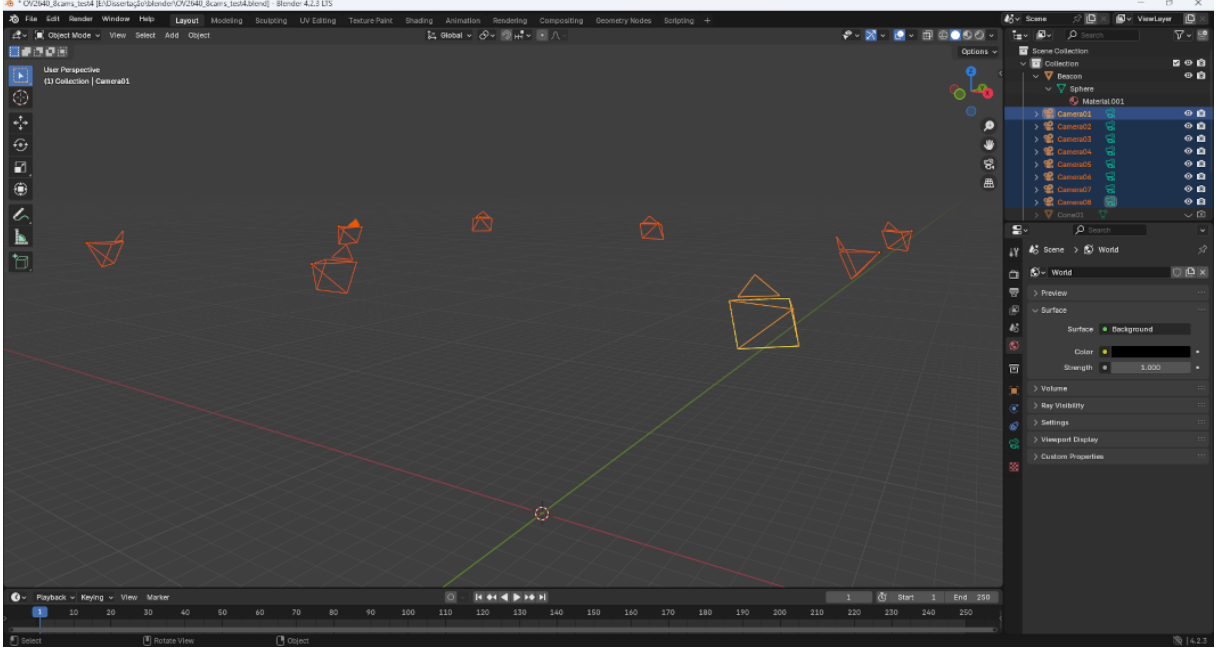


Figure 3.6: Camera distribution for test 04, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.4.

Table 3.4: Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 04.

Camera	x	y	z	α	β	γ	f	FOV
01	10.0000	-10.0000	10.0000	64.000	0.000	45.000	2.764	66.0
02	10.0000	10.0000	10.0000	64.000	0.000	135.000	2.764	66.0
03	-10.0000	10.0000	10.0000	64.000	0.000	225.000	2.764	66.0
04	-10.0000	-10.0000	10.0000	64.000	0.000	315.000	2.764	66.0
05	0.0000	-10.0000	10.0000	64.000	0.000	0.000	2.764	66.0
06	10.0000	0.0000	10.0000	64.000	0.000	90.000	2.764	66.0
07	0.0000	10.0000	10.0000	64.000	0.000	180.000	2.764	66.0
08	-10.0000	0.0000	10.0000	64.000	0.000	270.000	2.764	66.0

3.1.5 Test 05 Blender configuration

Test 05 is the most different scenario in the systematic case. In this simulation, cameras are distributed along a larger square pattern. In the centre of this pattern is located a 3D model of an aircraft (C-130J model from HJMediaStudio released under Creative Commons Attribution 3.0 (CC BY 3.0) [24], available from Blender Swap website [25]), which is used to assess the effect of a vision occluding object when cameras are tracking the beacon. The positions and orientations of cameras are compiled in Table 3.5 as in previous tests. One of the main differences in this test is that cameras are covering a larger working space, with 50 m side length of. The cameras are, however, not uniformly distributed in space, with the location of cameras in front and behind the aircraft. Fig. 3.7 shows the aforementioned camera distribution.

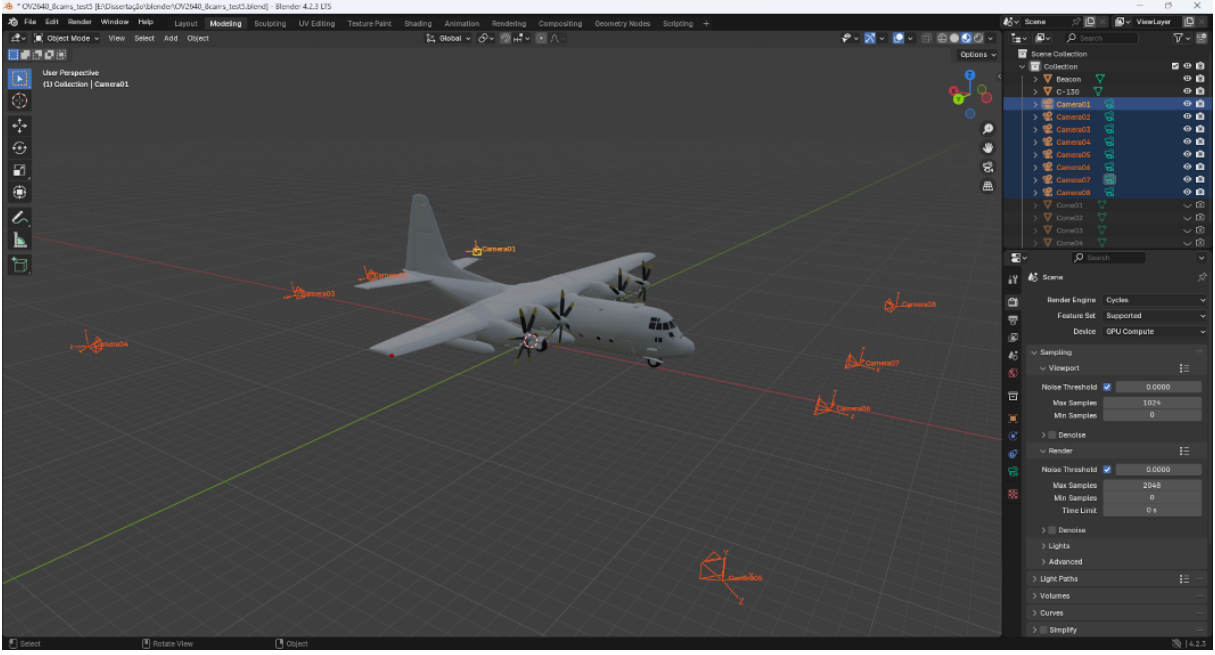


Figure 3.7: Camera distribution for test 05, captured from Blender interface. The positions and orientations of cameras are compiled in table 3.5.

Table 3.5: Camera positions (x, y, z) in meters and angles (α, β, γ) in degrees, and camera parameters used in test 05.

Camera	x	y	z	α	β	γ	f	FOV
01	25.0000	-25.0000	1.5000	90.000	0.000	50.000	2.764	66.0
02	25.0000	-5.0000	1.5000	107.000	0.000	90.000	2.764	66.0
03	25.0000	5.0000	1.5000	107.000	0.000	90.000	2.764	66.0
04	25.0000	25.0000	1.5000	90.000	0.000	130.000	2.764	66.0
05	-25.0000	25.0000	1.5000	90.000	0.000	230.000	2.764	66.0
06	-25.0000	5.0000	1.5000	95.000	0.000	273.000	2.764	66.0
07	-25.0000	-5.0000	1.5000	95.000	0.000	267.000	2.764	66.0
08	-25.0000	-25.0000	1.5000	90.000	0.000	310.000	2.764	66.0

3.2 Systematic position estimation tests

For each one of the scenarios described previously, a Python script in Blender was used to change the position of the beacon in the scene and record images from the perspective of each camera. The scenes are rendered without any lighting to simulate the appearance of the beacon as seen by the cameras. As explained in section 4.1.1, the cameras implemented in the system are equipped with an IR-pass filter, which blocks most of the visible light. So in Blender, the use of an emissive sphere 3D object in a dark scenario simulates the appearance of the beacon on the real cameras with good approximation. In table 3.6 are presented all the scenarios and resolutions tested for the systematic study.

Figure 3.8 shows the distribution of test positions simulated for the systematic case.

Distribution of simulated test positions in 3 dimensional space

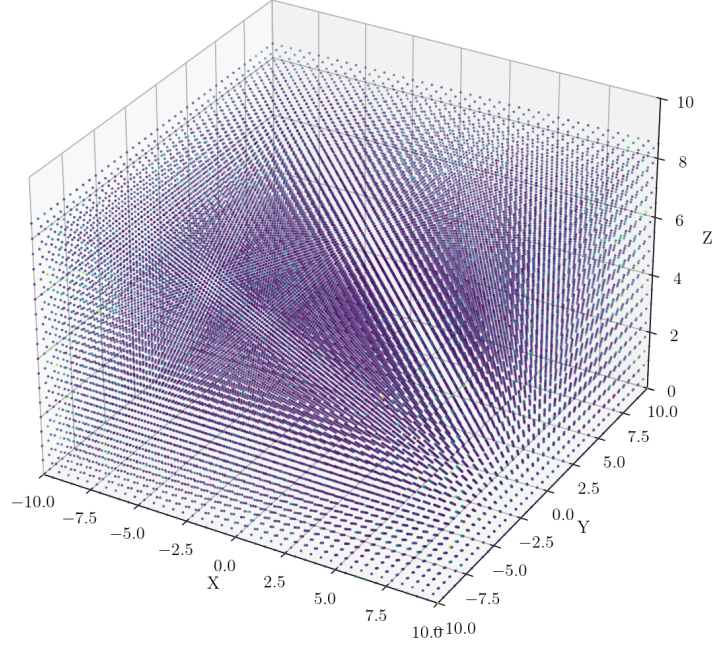


Figure 3.8: Test positions simulated in the systematic case. The figure was captured from test 01.

Table 3.6: Combinations of camera positions (tests), resolutions, field of view, number of cameras and number of beacon positions simulated in the systematic case.

Test	Resolution	FOV	N.er Cameras	Positions
1	320×240	66.0	8	30258
1	640×480	66.0	8	30258
1	800×600	66.0	8	30258
1	1024×768	66.0	8	30258
1	1600×1200	66.0	8	30258
2	320×240	66.0	8	30258
2	640×480	66.0	8	30258
2	800×600	66.0	8	30258
2	1024×768	66.0	8	30258
2	1600×1200	66.0	8	30258
3	320×240	66.0	8	30258
3	640×480	66.0	8	30258
3	800×600	66.0	8	30258
3	1024×768	66.0	8	30258
3	1600×1200	66.0	8	30258
4	320×240	66.0	8	30258
4	640×480	66.0	8	30258
4	800×600	66.0	8	30258
4	1024×768	66.0	8	30258
4	1600×1200	66.0	8	30258
5	320×240	66.0	8	30258
5	640×480	66.0	8	30258
5	800×600	66.0	8	30258
5	1024×768	66.0	8	30258
5	1600×1200	66.0	8	30258

3.3 Trajectory 1 position estimation tests

This simulation along with the trajectory 2 simulation, were generated with the aim of studying the capacity of the system to accurately recover a pre-planned trajectory followed by a robot. If the system is able to recover the trajectory, then it is possible to use the system to track and guide a robotic system along a predefined trajectory. This simulation implemented a beacon following a circular trajectory tilted by an angle $\phi = 15^\circ$ in relation to x axis. The parametric equation that describes the trajectory is given by Equation (3.1). The beacon was moved along the trajectory with a simulated speed of 0.5 m/s, which led to a total number of 1884 frames for a complete revolution along the trajectory. Table 3.7 shows all the resolutions and test scenarios studied for this case.

$$\begin{cases} x = r \cos \frac{k\omega}{f_r} \\ y = r \sin \frac{k\omega}{f_r} \cos \phi \\ z = 2.5 + r \sin \frac{k\omega}{f_r} \sin \phi \end{cases} \quad (3.1)$$

where $\omega = \frac{v}{r}$ is the angular speed of the beacon, with v the tangential speed of the beacon along the trajectory, r the radius of the circular trajectory, k the frame number (with $k \in \{0, 1, \dots, n\}$, n being the total number of frames and f_r the frame rate of the simulated video feed.

Table 3.7: Combinations of camera positions (tests), resolutions, field of view, number of cameras and number of beacon positions simulated in the trajectory 1 case.

Test	Resolution	FOV	N.er Cameras	Positions
1	320×240	66.0	8	1884
1	640×480	66.0	8	1884
1	800×600	66.0	8	1884
1	1024×768	66.0	8	1884
1	1600×1200	66.0	8	1884
2	320×240	66.0	8	1884
2	640×480	66.0	8	1884
2	800×600	66.0	8	1884
2	1024×768	66.0	8	1884
2	1600×1200	66.0	8	1884
3	320×240	66.0	8	1884
3	640×480	66.0	8	1884
3	800×600	66.0	8	1884
3	1024×768	66.0	8	1884
3	1600×1200	66.0	8	1884
4	320×240	66.0	8	1884
4	640×480	66.0	8	1884
4	800×600	66.0	8	1884
4	1024×768	66.0	8	1884
4	1600×1200	66.0	8	1884

3.4 Trajectory 2 position estimation tests

This simulation aimed to continue the study of the recovery of a trajectory, like the trajectory 1 simulation. The difference is that it implements a figure eight trajectory, also known as Bernoulli Lemniscate [26], with the same speed of the previous case, $v = 0.5$ m/s. The parametric equation that describes the trajectory is given by eq. (3.2). Table 3.8 compiles all the simulations tested for this trajectory.

$$\begin{cases} x = \frac{a \cos(\frac{\omega k}{f_r})}{1 + \sin^2(\frac{\omega k}{f_r})} \\ y = \frac{a \sin(\frac{\omega k}{f_r}) \cos(\frac{\omega k}{f_r})}{1 + \sin^2(\frac{\omega k}{f_r})} \\ z = 1.5 \end{cases} \quad (3.2)$$

where k is the frame number as defined in eq. 3.1, $\omega = 2\pi/T$, with T the period of revolution along the trajectory, a the semi-major axis of the trajectory and f_r the frame rate used on the simulation.

Table 3.8: Combinations of camera positions (tests), resolutions, field of view, number of cameras and number of beacon positions simulated in the trajectory 2 case.

Test	Resolution	FOV	N.er Cameras	Positions
1	320×240	66.0	8	1884
1	640×480	66.0	8	1884
1	800×600	66.0	8	1884
1	1024×768	66.0	8	1884
1	1600×1200	66.0	8	1884
2	320×240	66.0	8	1884
2	640×480	66.0	8	1884
2	800×600	66.0	8	1884
2	1024×768	66.0	8	1884
2	1600×1200	66.0	8	1884
3	320×240	66.0	8	1884
3	640×480	66.0	8	1884
3	800×600	66.0	8	1884
3	1024×768	66.0	8	1884
3	1600×1200	66.0	8	1884
4	320×240	66.0	8	1884
4	640×480	66.0	8	1884
4	800×600	66.0	8	1884
4	1024×768	66.0	8	1884
4	1600×1200	66.0	8	1884

3.5 Systematic Position Simulation results

This section describes the results from the simulations of the systematic study. For each scenario (test), the estimation error is computed and analysed. By absolute error e_{abs} , it is meant the difference between the estimated value from the position estimation algorithm and the ground-truth (the real position simulated) for each iteration of beacon position as given by eq. (3.3).

For the purpose of analysing the simulated data, a camera “TrackingCamera” Class was implemented in Python, which contains methods to read the images (simulating frames from a real camera), to measure the blob position (identifying the beacon), to project a 3D point to the image plane, to compute the beacon direction vector from the image position and camera orientation, among others auxiliary methods. From this base class, a “SimTrackingCamera” was derived to integrate into the simulation data. This class is part of a Python routine that analyses all the data from all the cameras in a simulation, outputting the measured data as CSV (Comma Separated Values) and JSON (JavaScript Object Notation) files. Also generates a text file with the simulation statistics and several plots, showing the error per iteration, the error mapping and camera mapping over space at several z levels, comparisons between different resolutions for each test and histograms of error distribution per number of cameras observing the beacon.

$$e_{abs} = \|\mathbf{X}_{est} - \mathbf{X}_{gt}\| \quad (3.3)$$

where \mathbf{X}_{est} is the estimated position and \mathbf{X}_{gt} is the ground-truth position.

Below, the results of each test are described and analysed in greater detail.

3.5.1 Analysis of Test 01

For the analysis of the camera distribution featured on test 01, described in subsection 3.1.1, each resolution simulated was input into the “simulation_analysis” python script. After processing the data, figures and tables were used to better understand the results. Figure 3.9 presents, in the top, the absolute error of the tracking algorithm for test 01 of camera distribution using a resolution of 320×240, and in the bottom, only values in the 90-percentile.

Each point corresponds to one iteration along the positions simulated, and is colour coded accordingly to the number of cameras detecting the beacon. By the colours alone, it is noticeable that higher errors match mostly the cases where the number of cameras detecting the beacon is low. It is also noticeable that when the beacon is detected by 3 cameras, there is a repetitive pattern. This is related to the position of the beacon on those iterations, which is close to the edges of the simulated space, and coincidentally, close to one of the cameras. Figure 3.10 is similar, but for the case of 1600×1200 of resolution.

The error data distribution per camera is presented in figure 3.11 in the form of histograms for the case of 320×240. It starts from 2 cameras and goes up to 8 cameras, detecting the object. The same plot shows a curve which represents the Weibull distribution that best fits the data and is further described below. Figure 3.12 is again similar to figure 3.11 but for the higher resolution of 1600×1200. In both cases, the error axis of the plot has the same range, with the largest error of all the cases being the maximum limit of the range, enabling direct visual comparison between the different sets of cameras.

Figure 3.13 compares the error as a function of the number of cameras (horizontal axis) and resolution (colour coded), allowing the observation that error decreases as the number of cameras also increases, in particular above 4 cameras, and that it also decreases with higher resolution.

Table 3.9 compiles all the global statistics, regardless of the number of cameras detecting for all the iterations, for resolution 320×240 and statistics per number of cameras detecting the object are presented in table 3.11. Similarly, for the maximum resolution simulated, 1600×1200, the same statistics are compiled in table 3.10 and 3.12. These are the global statistics, not taking into account the number of cameras observing the beacon at each position. For that case, statistics per camera observing the beacon, Tables 3.9 and 3.10 are presented for cases of resolution 320×240 and 1600×1200 respectively.

It can be observed that 90% of the values show an absolute error e_{abs} lower than 0.0285 m, in the case of resolution 320×240. When using a resolution of 1600×1200, it can be seen that 90% of the values are now lower than 0.0035 m.

The mean absolute error is $\bar{e}_{abs} = 0.0135$ m with a standard deviation $\sigma = 0.0108$ m at resolution 320×240 while at resolution 1600×1200 both the mean and standard deviation decrease down to $\bar{e}_{abs} = 0.0018$ m and $\sigma = 0.0013$ m.

The full set of error per iteration plots for all the resolutions simulated in test 01 is presented in the appendix section A.1.

Table 3.9: Global statistics and root mean square error for test 01 with resolution 320×240.

	Abs. error	Est. error
count	30153	30153
mean	0.0135	0.0235
std	0.0108	0.0093
min	0.0002	0.0000
25%	0.0060	0.0171
50%	0.0099	0.0230
75%	0.0175	0.0293
90%	0.0285	0.0357
max	0.0824	0.0629
RMSE	0.0173	

Table 3.10: Global statistics and root mean square error for test 01 with resolution 1600×1200.

	Abs. error	Est. error
count	30225	30225
mean	0.0018	0.0032
std	0.0013	0.0012
min	0.0000	0.0000
25%	0.0009	0.0024
50%	0.0014	0.0032
75%	0.0023	0.0040
90%	0.0035	0.0048
max	0.0112	0.0095
RMSE	0.0022	

Table 3.11: Statistics per number of cameras observing the beacon for test 01 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	420	11400	4340	4440	3808	1848	3897
mean	0.0154	0.0220	0.0122	0.0083	0.0072	0.0062	0.0050
std	0.0079	0.0121	0.0070	0.0038	0.0032	0.0026	0.0023
min	0.0038	0.0004	0.0002	0.0006	0.0005	0.0012	0.0002
25%	0.0095	0.0131	0.0076	0.0055	0.0048	0.0044	0.0033
50%	0.0131	0.0193	0.0110	0.0079	0.0068	0.0059	0.0048
75%	0.0208	0.0285	0.0152	0.0107	0.0094	0.0077	0.0064
max	0.0427	0.0824	0.0649	0.0256	0.0178	0.0168	0.0146

Table 3.12: Statistics per number of cameras observing the beacon for test 01 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	320	10888	4820	4552	3892	1784	3969
mean	0.0018	0.0028	0.0018	0.0012	0.0010	0.0008	0.0007
std	0.0008	0.0016	0.0009	0.0005	0.0005	0.0003	0.0003
min	0.0003	0.0002	0.0002	0.0001	0.0000	0.0001	0.0001
25%	0.0012	0.0017	0.0012	0.0009	0.0007	0.0005	0.0005
50%	0.0016	0.0025	0.0016	0.0012	0.0010	0.0008	0.0007
75%	0.0022	0.0036	0.0023	0.0015	0.0013	0.0010	0.0009
max	0.0041	0.0112	0.0105	0.0042	0.0034	0.0020	0.0021

From these figures, it is clearly noted that the improvement in the error of the system is achieved by the higher resolution. It makes sense, given that this increase in resolution results in an increased spatial resolution of the direction vector \mathbf{v} , which is used to find the location of the object.

The histograms of figures 3.11 and 3.12 show the distribution of error for the cases of 320×240 and 1600×1200, which in both cases, show a higher concentration of error towards zero, with values of higher error occurring less frequently, which is also probably related to the position of the estimated point in relation to the fields of view of cameras and their location. The data has been separated accordingly to the number of cameras observing the blob at each iteration, to make it possible to compare visually the effect of the number of cameras on the error of the system. The red curve on each histogram is the fitted

Distance absolute error e_{abs} per test position
Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

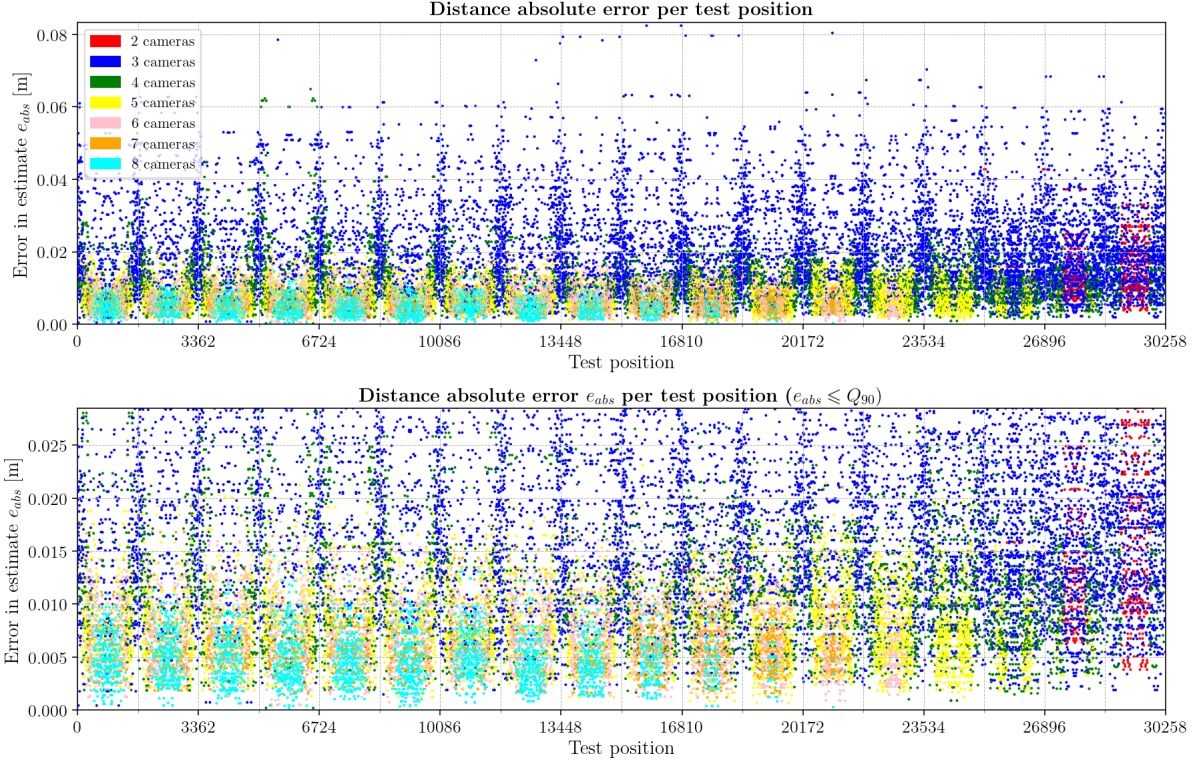


Figure 3.9: Absolute error e_{abs} per iteration for resolution 320×240 and test 01.

Weibull PDF for each case, and also, as a vertical line, the median of the error \tilde{e} and the mean error \bar{e} as a vertical dashed line.

However, it can be noted by observing the histograms presented in figures 3.11 and 3.12 that the error distribution is right-skewed, which implies that the use of mean error and standard deviation may not be the best way of describing the typical error of the system. This typical error is probably better described using the median of the data, which by tables 3.9 and 3.10 are $\tilde{e}_{abs} = 0.0099$ m for case 320×240 and $\tilde{e}_{abs} = 0.0014$ m for case 1600×1200.

Because of the skew presented in the data, this work also evaluated the fitting of a Weibull probability density function (PDF) [27, 28] to the error data, which would enable a better understanding of each simulation and determine the most probable values of error to arise for a given distribution of cameras.

The Weibull Probability Distribution Function is defined by three parameters, the shape c , the scale λ and the location parameter usually denoted by loc , and for a given value x , the probability of that value occurring is given by equation (3.4)

$$f(x; c, loc, \lambda) = \frac{c}{\lambda} \left(\frac{x - loc}{\lambda} \right)^{c-1} e^{-\left(\frac{x - loc}{\lambda} \right)^c}, \quad x \geq loc \quad (3.4)$$

Referring to tables 3.13 and 3.14 (for resolutions 320×240 and 1600×1200 respectively), it can be observed that the curve shape parameter c is always greater than 1, which indicates that larger errors are less frequent, but still likely to happen. This can be due to the position of the beacon in relation to cameras or to the number of cameras observing the beacon at a given frame, for example. But other causes can not be excluded, as this work did not focus on these effects.

The loc parameter is $loc = 0$ for all the fitted distributions, regardless of resolution used, which makes sense, given the error being a Euclidean distance, which is always positive with a minimum value of 0.

More important for this study is the scale parameter λ , which gives the most likely value of error to show up in a given position estimate. Again, using table 3.13 it can be seen that at 320×240 resolution

Distance absolute error e_{abs} per test position
 Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

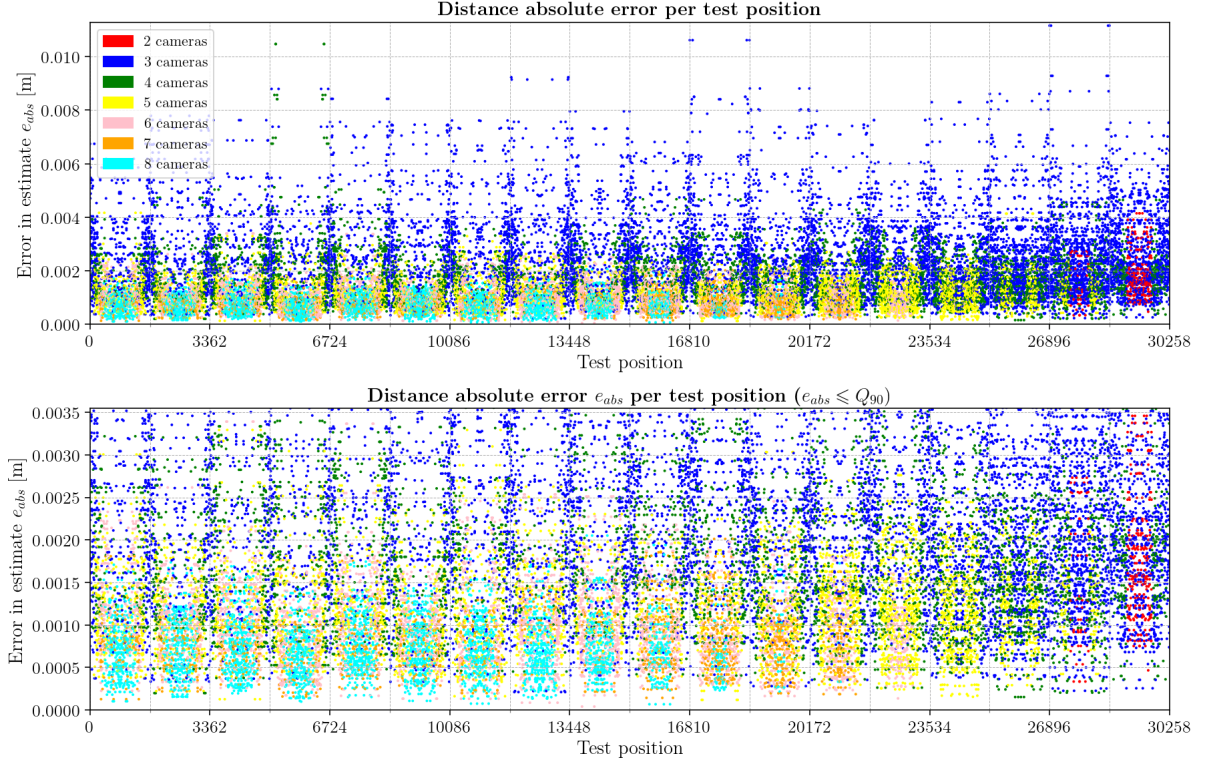


Figure 3.10: Absolute error e_{abs} per iteration for resolution 1600×1200 and test 01.

the smallest “most probable” error is $\lambda = 0.0056$ m when using 8 cameras and similarly for resolution 1600×1200 the smallest most probable error decreases to $\lambda = 2.3422$ m, again for the case of 8 cameras detecting the beacon. The worst cases in both resolutions happen when using a combination of 3 cameras.

Absolute error distribution in position estimation

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

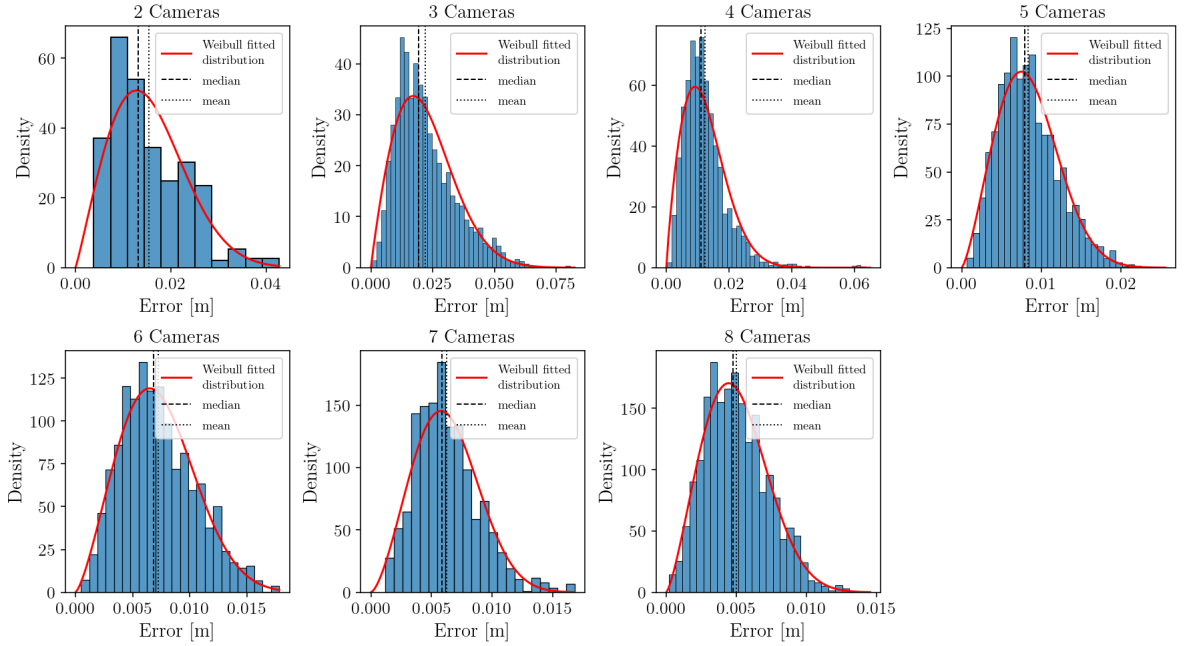


Figure 3.11: Histogram of error per camera in resolution 320×240 and test 01.

Absolute error distribution in position estimation

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

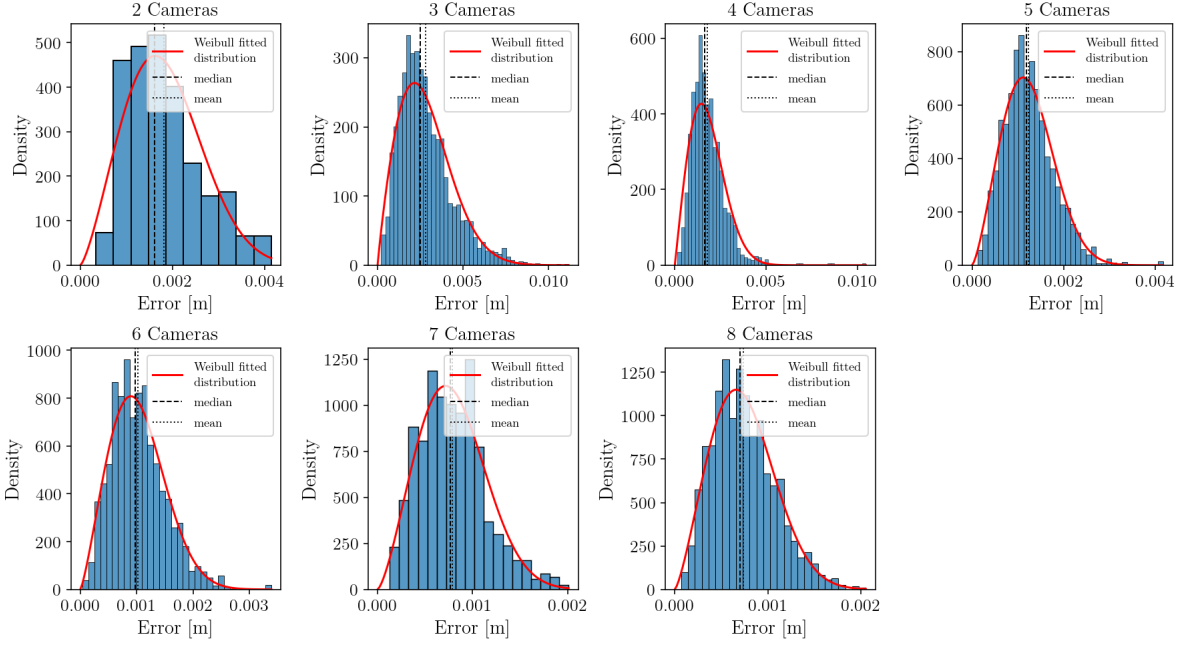


Figure 3.12: Histogram of error per camera in resolution 1600×1200 and test 01.

Table 3.13: Weibull PDF parameters and bin count per camera used in figure 3.11 for test 01 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	2.0963	0.0000	0.0175	420	11
3	1.9331	0.0000	0.0249	11400	44
4	1.8632	0.0000	0.0138	4340	44
5	2.3617	0.0000	0.0094	4440	32
6	2.3785	0.0000	0.0082	3808	24
7	2.5380	0.0000	0.0070	1848	22
8	2.3444	0.0000	0.0056	3897	29

Table 3.14: Weibull PDF parameters and bin count per camera used in figure 3.12 for test 01 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	2.3583	0.0000	0.0021	320	10
3	1.9150	0.0000	0.0032	10888	45
4	2.0339	0.0000	0.0020	4820	54
5	2.4069	0.0000	0.0014	4552	36
6	2.2673	0.0000	0.0012	3892	32
7	2.4249	0.0000	0.0009	1784	19
8	2.3422	0.0000	0.0008	3969	27

Figure 3.14 maps the error of the system with the beacon at a fixed height, or in other words, maps the error at a given level. The error is colour-coded, according to the bar at the right of the map, with dark blue colour meaning lower errors and yellow colour meaning higher errors. The colour scale maps errors from 0.0000 m to 0.040 m, and errors greater than 0.040 m are truncated to the same yellow as 0.040 m. Error maps were produced for levels from $z = 0.000$ m to $z = 8.500$ m, with a step of 0.500 m. On this

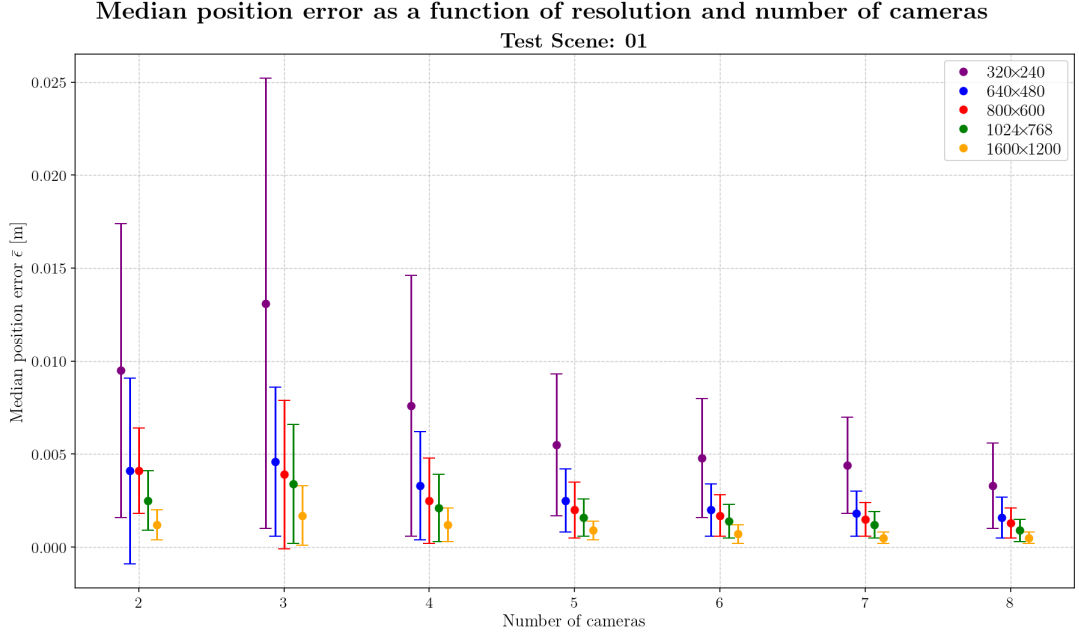


Figure 3.13: Variation of mean error $\bar{\epsilon}_{abs}$ in estimated position as a function of number of cameras and resolution used, for test 01.

section only the error maps for level $z = 2.000$ m are presented for resolutions 320×240 (fig. 3.14) and 1600×1200 (fig. 3.16).

At the right of these figures, the corresponding camera coverage over the simulated study space is presented in figures 3.15 and 3.17, with the number of cameras covering each location encoded with a colour scale. Due to the spacial resolution used for exploring the simulated space, there aren't noticeable differences between the resolutions used in the printed version of the maps; however, in higher resolution images on a screen, it can be noticed that with higher resolution, the borders of camera coverage levels become better defined.

The full set of error maps for test 1 is available in appendix A, on section A.1, page A1.

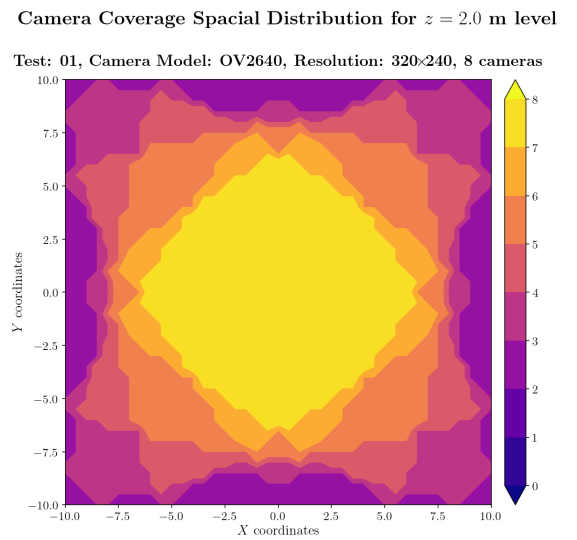
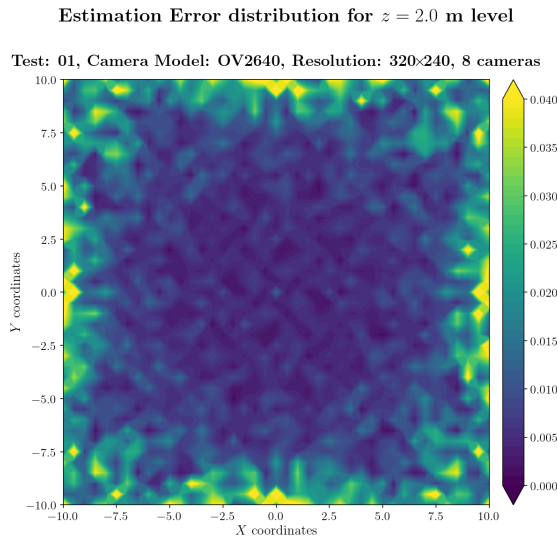
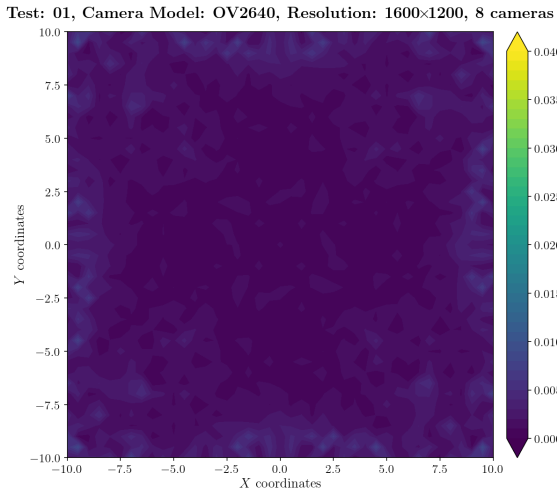


Figure 3.14: Estimation error contour map for test 01 with 8 cameras using a resolution of 320×240 pixels, at level $z = 2.00$ m.

Figure 3.15: Camera coverage for test 01 at level $z = 2.00$ m and resolution 320×240.

Estimation Error distribution for $z = 2.0$ m level



Camera Coverage Spacial Distribution for $z = 2.0$ m level

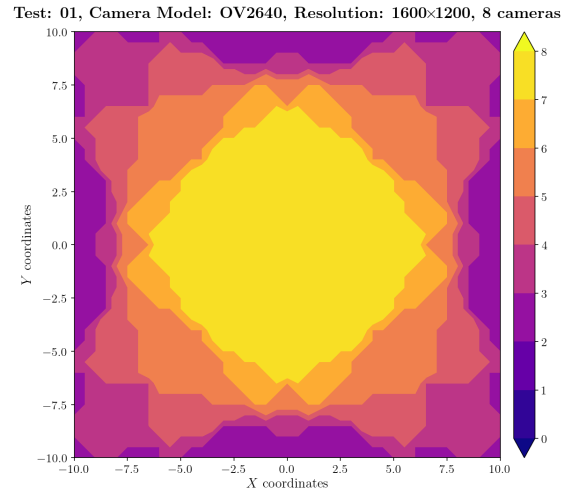


Figure 3.16: Estimation error contour map for test 01 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.

Figure 3.17: Camera coverage for test 01 at level $z = 2.00$ m and resolution 1600×1200.

3.5.2 Analysis of Test 02

For the analysis of the test of camera configuration number 2 (described in subsection 3.1.2), the procedure was adopted as in the previous test. Figures 3.18 and 3.19 show the error per iteration for resolution 320×240 and 1600×1200 respectively, encoding again the number of cameras observing the beacon by colours.

From these two figures, it is clearly noticeable that the existence of a few outlier cases where the error reached abnormally high values. But all these cases happened when only two cameras were detecting the beacon, and also when the beacon was close to one of them. Still, there is an overall reduction in estimation error of the algorithm, when comparing to the similar figures of test 01 (figs. 3.9 and 3.10). In this case, referring to the bottom plot of the figures and to table 3.15, it can be seen that 90% of the values fall below $e_{abs} \leq 0.0203$ m for resolution 320×240, and at the highest resolution of 1600×1200, the error in estimation drops even further, with 90% of values under $e_{abs} \leq 0.0028$ m, as shown on table 3.16.

Statistical data per camera number observing the beacon is presented in Tables 3.17 and 3.18 for resolutions 320×240 and 1600×1200 respectively.

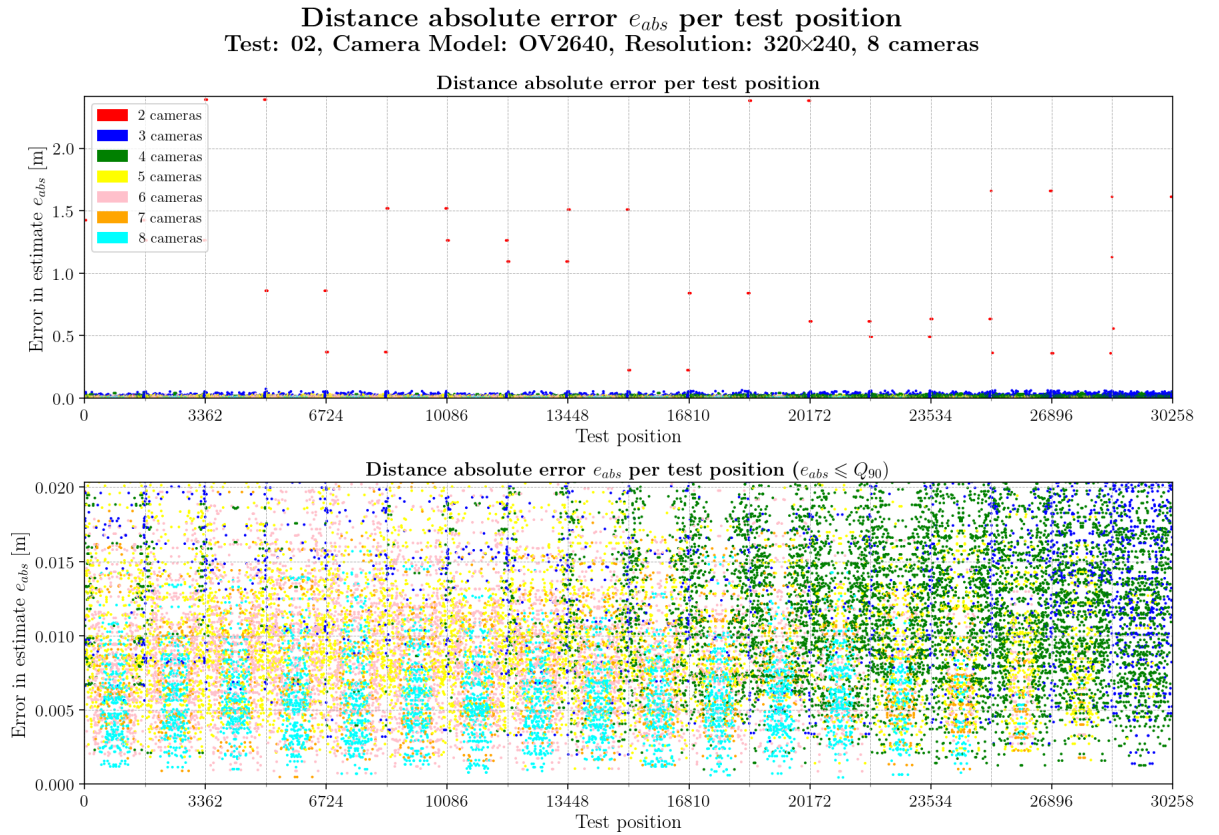


Figure 3.18: Absolute error e_{abs} per iteration for resolution 320×240 and test 02.

Distance absolute error e_{abs} per test position
Test: 02, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Figure 3.19: Absolute error e_{abs} per iteration for resolution 1600×1200 and test 02.

Table 3.15: Global statistics and root mean square error for test 02 with resolution 320×240.

	Abs. error	Est. error
count	30258	30258
mean	0.0140	0.0290
std	0.0624	0.0096
min	0.0004	0.0017
25%	0.0064	0.0224
50%	0.0096	0.0283
75%	0.0144	0.0349
90%	0.0203	0.0414
max	2.3882	0.0792
RMSE	0.0640	

Table 3.16: Global statistics and root mean square error for test 02 with resolution 1600×1200.

	Abs. error	Est. error
count	30258	30258
mean	0.0020	0.0040
std	0.0123	0.0013
min	0.0000	0.0004
25%	0.0008	0.0031
50%	0.0013	0.0039
75%	0.0019	0.0047
90%	0.0028	0.0056
max	0.5734	0.0149
RMSE	0.0124	

As in test 01, figures 3.20 and 3.21 show the histograms of error data for 320×240 and 1600×1200. Like before, a Weibull distribution was fitted to the data histograms, with the results of the fitted Weibull parameters per number of cameras condensed into table 3.19, for the lowest resolution tested (320×240) and in table 3.20 for the highest resolution (1600×1200).

Again, it can be seen that the scale parameter λ of Weibull PDF gives the most frequent error presented in the system, with $\lambda = 0.0066$ m, for the combination of 8 cameras and resolution of 320×240 and with $\lambda = 0.0009$ m for the combination of 8 cameras at resolution of 1600×1200. These values are close to the medians calculated, which according to table 3.17 is $\tilde{e}_{abs} = 0.0096$ m and from table 3.18 is $\tilde{e}_{abs} = 0.0013$ m, again, for resolutions 320×240 and 1600×1200 respectively.

Table 3.17: Statistics per number of cameras observing the beacon for test 02 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	80	3628	8061	5444	6548	1900	4597
mean	1.0074	0.0222	0.0132	0.0103	0.0090	0.0079	0.0058
std	0.6843	0.0119	0.0061	0.0048	0.0043	0.0038	0.0026
min	0.0197	0.0014	0.0010	0.0010	0.0007	0.0005	0.0004
25%	0.4612	0.0135	0.0089	0.0070	0.0059	0.0049	0.0039
50%	0.8594	0.0200	0.0127	0.0095	0.0084	0.0074	0.0056
75%	1.5088	0.0289	0.0167	0.0129	0.0114	0.0102	0.0074
max	2.3882	0.0739	0.0427	0.0391	0.0344	0.0215	0.0157

Table 3.18: Statistics per number of cameras observing the beacon for test 02 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	72	3360	8145	5448	6532	2020	4681
mean	0.2094	0.0031	0.0018	0.0014	0.0012	0.0010	0.0008
std	0.1409	0.0015	0.0009	0.0006	0.0005	0.0004	0.0004
min	0.0118	0.0003	0.0001	0.0001	0.0000	0.0001	0.0000
25%	0.1306	0.0020	0.0012	0.0009	0.0008	0.0006	0.0005
50%	0.1519	0.0029	0.0017	0.0013	0.0011	0.0009	0.0008
75%	0.3384	0.0040	0.0023	0.0018	0.0015	0.0012	0.0011
max	0.5734	0.0089	0.0066	0.0042	0.0047	0.0029	0.0031

Absolute error distribution in position estimation

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras

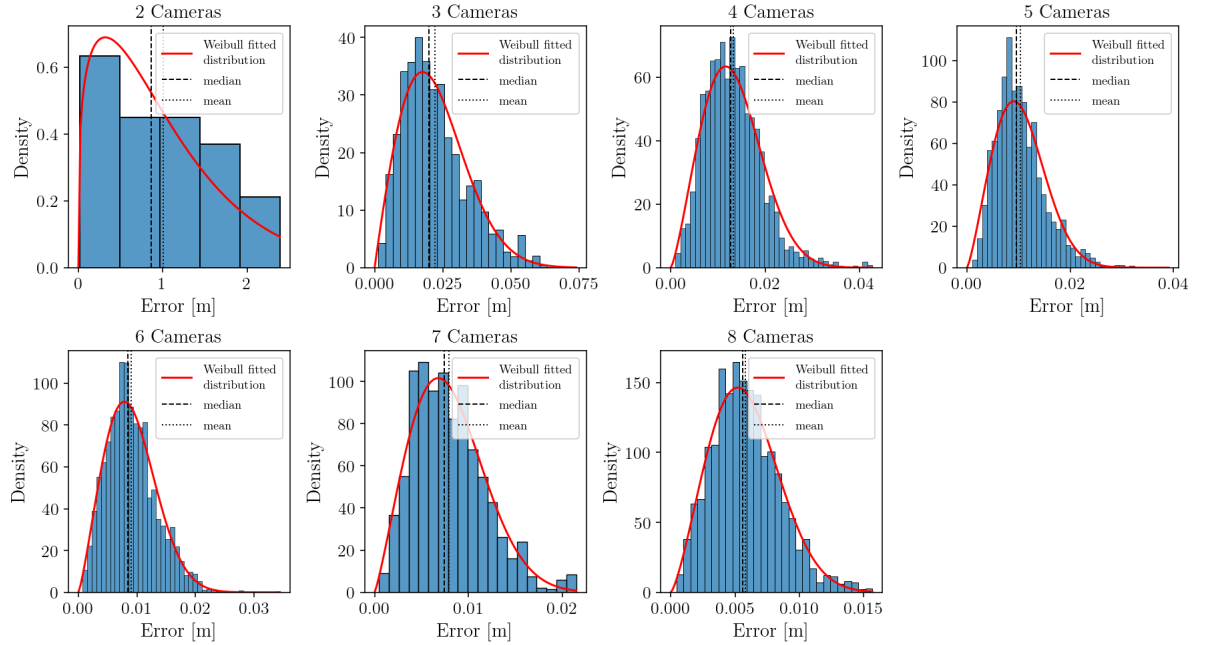


Figure 3.20: Histogram of error per camera in resolution 320×240 and test 02.

As in the previous test analysis, figure 3.22 relates the absolute error of the system with the number of cameras (horizontal axis) and the resolution used (colour coded). In this particular case, due mainly to the configuration of cameras used, there are many outliers in the data, caused by beacon positions only detected by two cameras, which probably were the ones located at the same horizontal coordinates, which caused them to behave as stereo cameras, reducing the ability to accurately recover the position of the beacon, as the baseline between cameras is non-existent in the horizontal plane. In the same figure,

Absolute error distribution in position estimation

Test: 02, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

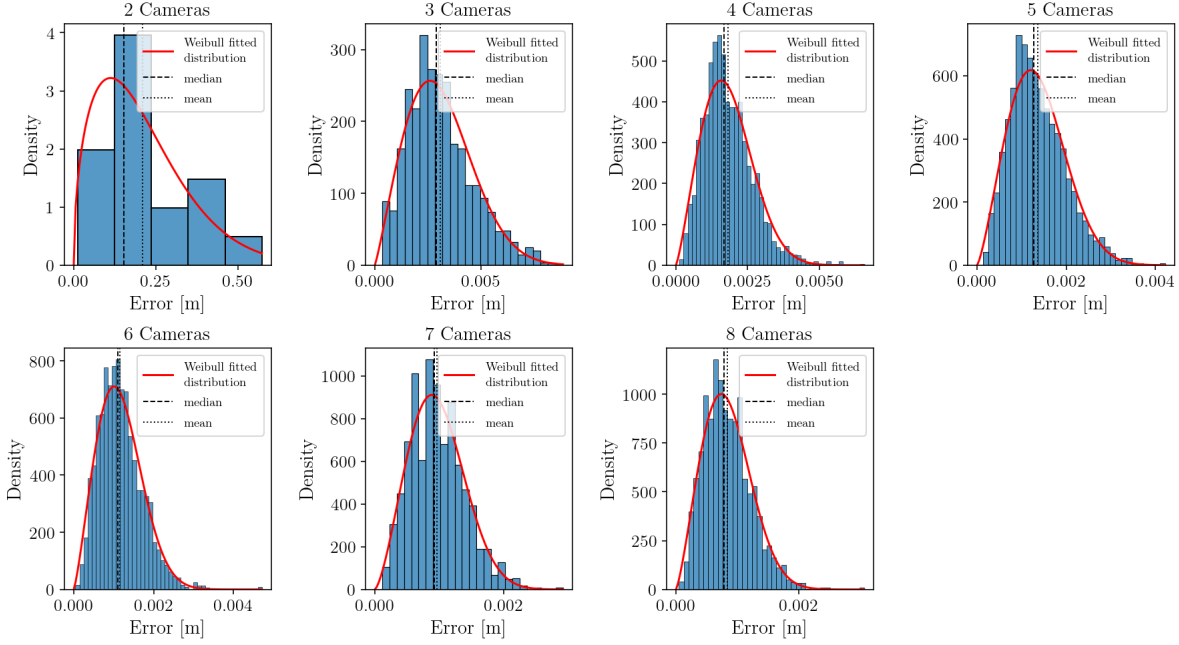


Figure 3.21: Histogram of error per camera in resolution 1600×1200 and test 02.

Table 3.19: Weibull PDF parameters and bin count per camera used in figure 3.20 for test 02 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	1.2710	0.0000	1.0725	80	5
3	1.9801	0.0000	0.0251	3628	27
4	2.2977	0.0000	0.0149	8061	40
5	2.2724	0.0000	0.0117	5444	40
6	2.2472	0.0000	0.0102	6548	43
7	2.1837	0.0000	0.0090	1900	20
8	2.3529	0.0000	0.0066	4597	28

Table 3.20: Weibull PDF parameters and bin count per camera used in figure 3.21 for test 02 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	1.5056	0.0000	0.2316	72	5
3	2.1348	0.0000	0.0035	3360	24
4	2.2418	0.0000	0.0021	8145	44
5	2.3103	0.0000	0.0015	5448	33
6	2.2562	0.0000	0.0013	6532	46
7	2.4571	0.0000	0.0011	2020	25
8	2.3024	0.0000	0.0009	4681	38

there is a zoomed plot in the region with 3 to 8 cameras, where it can be seen that when more than 2 cameras are detecting the beacon, the system behaves similarly to test 01, with a reduction of error with increased number of cameras and increased resolution.

Figures 3.23 and 3.25 map the error of the system with the beacon at a fixed height, as presented in

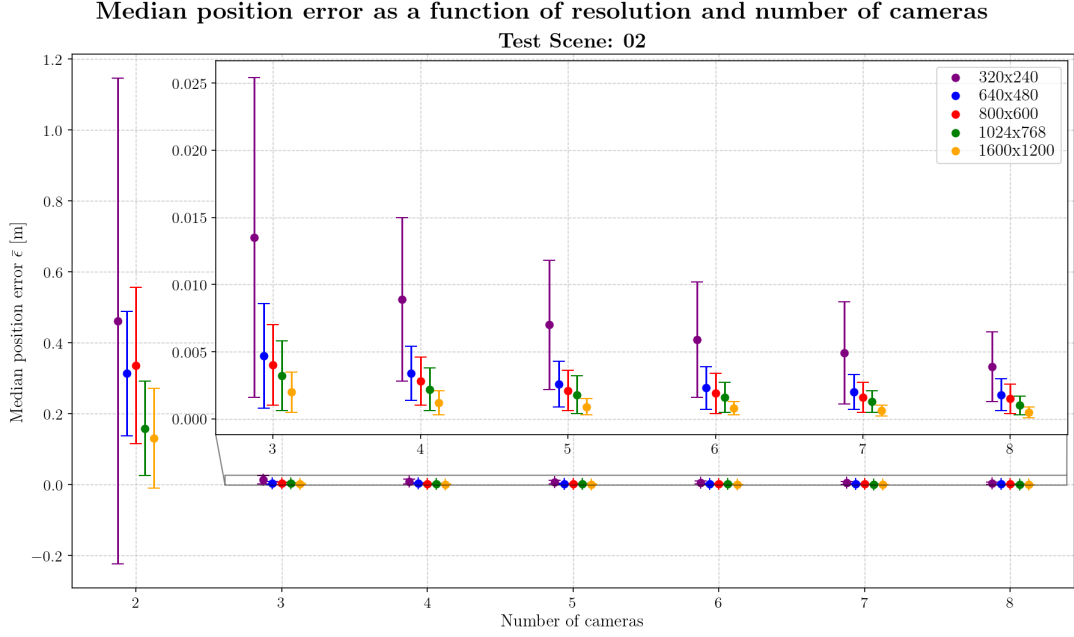


Figure 3.22: Variation of mean error \bar{e}_{abs} in estimated position as a function of number of cameras and resolution used, for test 02.

section 3.5.1. Error maps were again produced for levels from $z = 0.000$ m to $z = 8.500$ m, with a step of 0.500 m. On this section only the error maps for level $z = 2.000$ m are presented for resolutions 320×240 (fig. 3.23) and 1600×1200 (fig. 3.25). The full set of error maps for test 2 is found in appendix A, on section A.2, page A52. Also, is presented in figures 3.24 and 3.26, and it is shown with the number of cameras covering each location graded by colours.

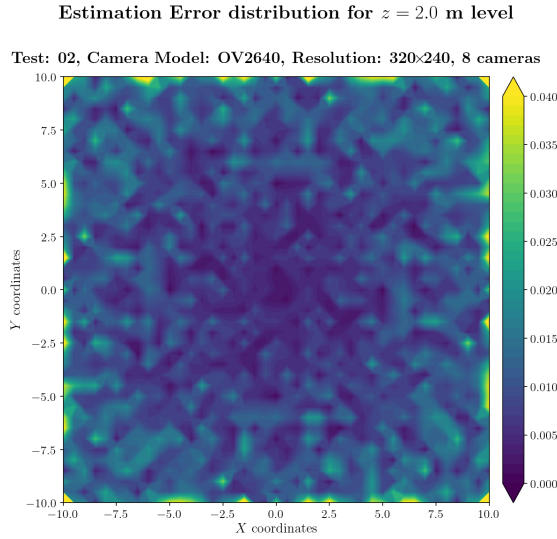


Figure 3.23: Estimation error contour map for test 02 with 8 cameras using a resolution of 320×240 pixels, at level $z = 2.00$ m.

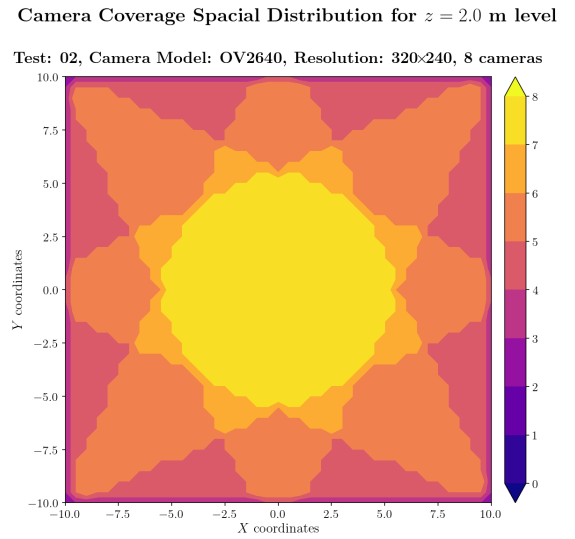
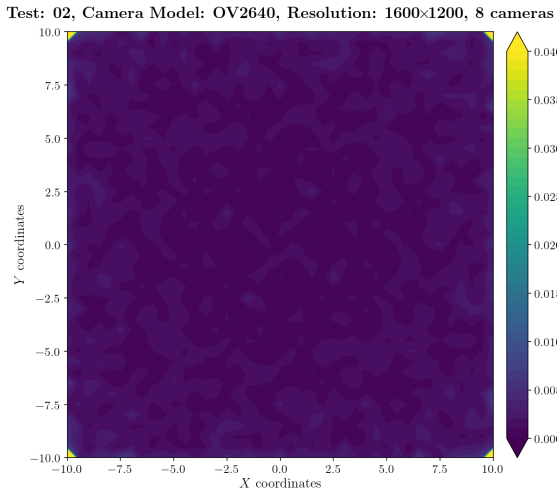


Figure 3.24: Camera coverage for test 02 at level $z = 2.00$ m.

Estimation Error distribution for $z = 2.0$ m level



Camera Coverage Spacial Distribution for $z = 2.0$ m level

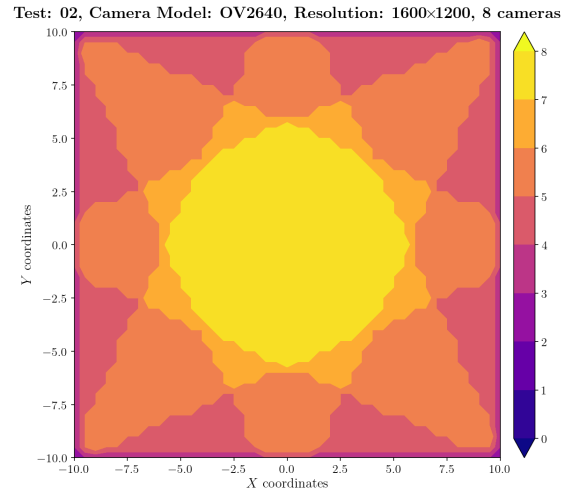


Figure 3.25: Estimation error contour map for test 02 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.

Figure 3.26: Camera coverage for test 02 at level $z = 2.00$ m.

3.5.3 Analysis of Test 03

In this subsection 3.5.3, the test camera configuration 3 is analysed. In this case, cameras are now set up in a higher position than in the previous test (see subsection 3.1.3) and Figure 3.27 presents the error per iterated test position, for 320×240. Same results are presented for resolution 1600×1200 in Figure 3.28. It is noticeable that there is an absence of data after test position 16810. That is, since this position corresponds to level $z = 5$ m, precisely the height at which the cameras are set up, and their top faces of fields of view aligned with this plane. Therefore, above this height, no points are visible by any of the cameras in the system. Colour codes encode again the number of cameras observing the beacon.

From these two figures, it is also visible that the existence of a few outlier cases where the error reached high values. But like in test 02, most of these cases happened when only two cameras were detecting the beacon and also with the beacon at a height close to camera level. Ignoring these outliers, the pattern of error resembles the one obtained in test 01, most likely due to a similar configuration of cameras, just with a different orientation. Using resolution 320×240, we can observe that 90% of values fall below $e_{abs} \leq 0.0227$ m, with a mean value of $\bar{e}_{abs} = 0.0110$ m, standard deviation of $\sigma = 0.0105$ m and a median value of $\sigma = 0.0078$ m, as can be seen in Table 3.21. For resolution 1600×1200, 90% of error values become smaller than $e_{abs} \leq 0.0030$ m, with an overall mean value of $\bar{e}_{abs} = 0.0015$ m, standard deviation of $\sigma = 0.0013$ m and a median value of $\bar{e}_{abs} = 0.0011$ m. Results for this resolution are compiled in Table 3.22.

The statistical data per camera number observing the beacon is presented in Table 3.23 for resolution 320×240 and in Table 3.24 for resolution 1600×1200.

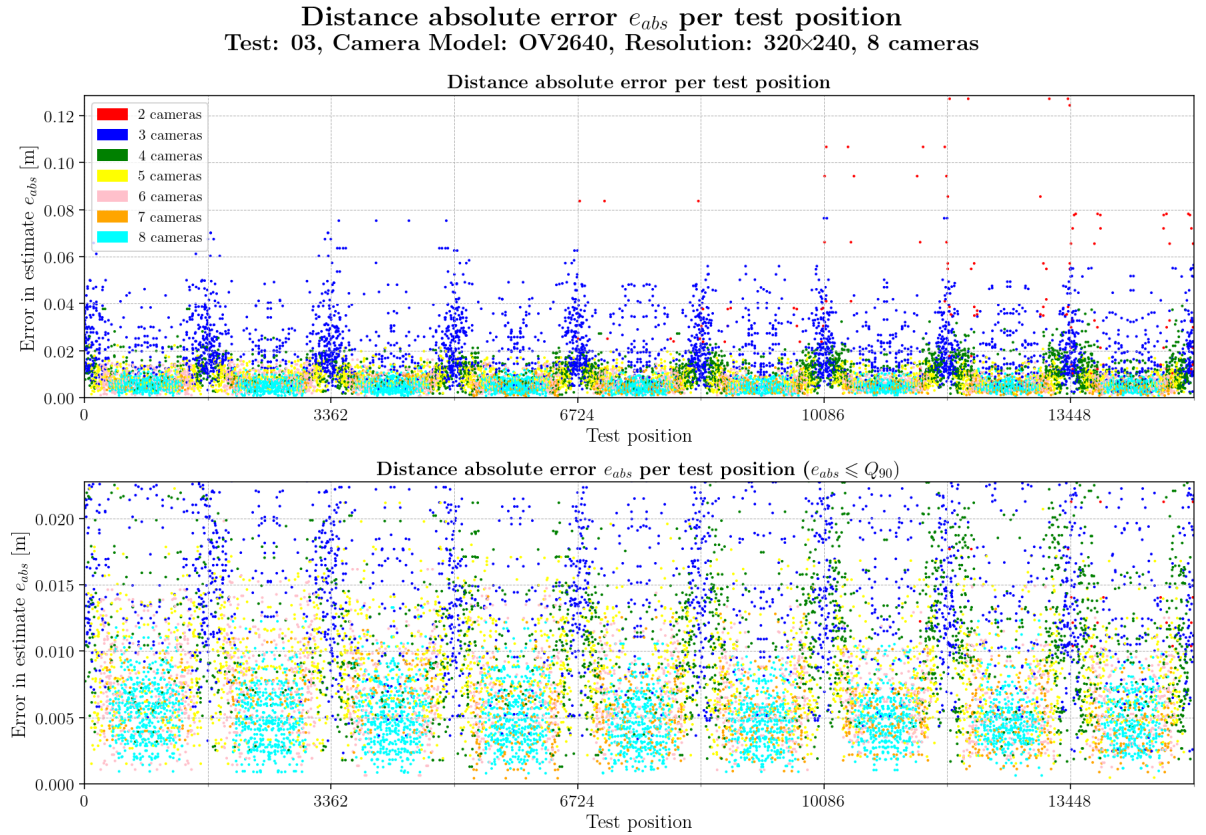


Figure 3.27: Absolute error e_{abs} per iteration for resolution 320×240 and test 03.

Distance absolute error e_{abs} per test position
Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

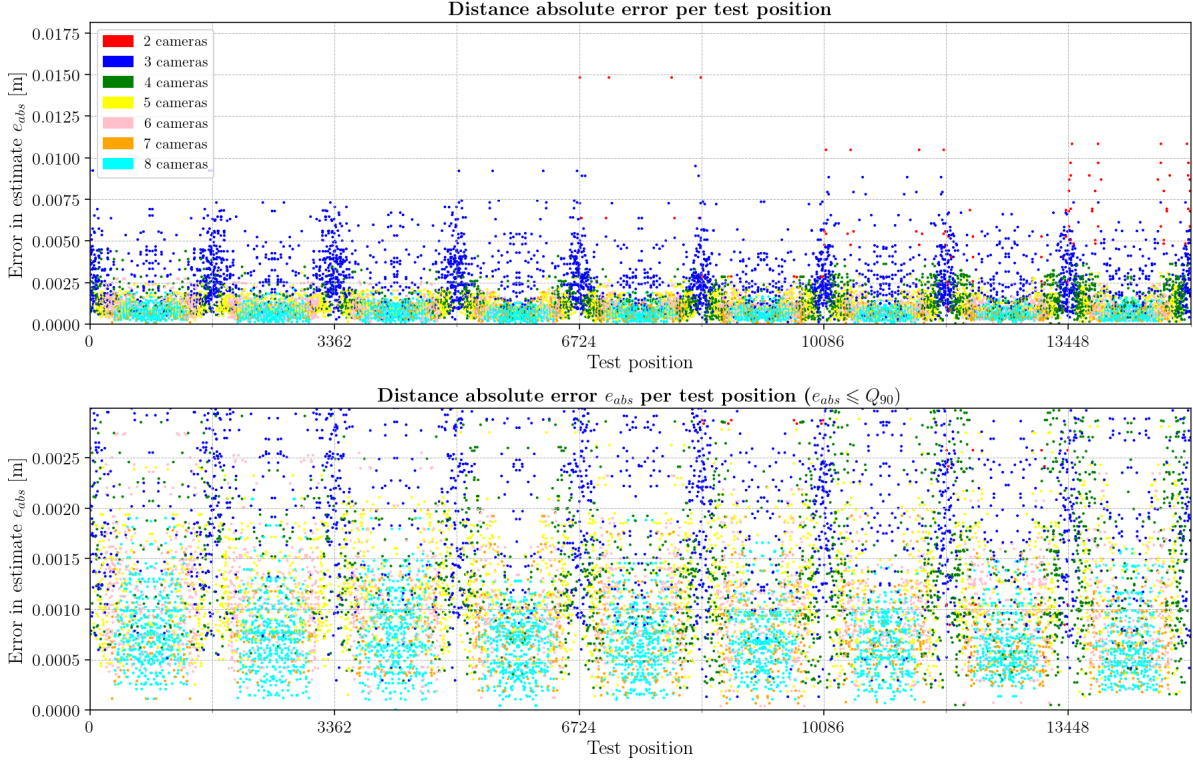


Figure 3.28: Absolute error e_{abs} per iteration for resolution 1600×1200 and test 03.

Table 3.21: Global statistics and root mean square error for test 03 with resolution 320×240.

	Abs. error	Est. error
count	16810	16810
mean	0.0110	0.0238
std	0.0105	0.0084
min	0.0004	0.0001
25%	0.0049	0.0180
50%	0.0078	0.0233
75%	0.0131	0.0294
90%	0.0227	0.0345
max	0.1272	0.0700
RMSE	0.0152	

Table 3.22: Global statistics and root mean square error for test 03 with resolution 1600×1200.

	Abs. error	Est. error
count	16810	16810
mean	0.0015	0.0033
std	0.0013	0.0011
min	0.0000	0.0000
25%	0.0007	0.0025
50%	0.0011	0.0032
75%	0.0018	0.0040
90%	0.0030	0.0047
max	0.0179	0.0076
RMSE	0.0020	

As in previous tests, Figure 3.29 and Figure 3.30 present the histograms of error data for 320×240 and 1600×1200 resolutions, respectively. Table 3.25 contains the Weibull distribution fitted parameters to the data histograms, for the lowest resolution tested of 320×240. Similarly, Table 3.26 contains the Weibull distribution fitted parameters for resolution 1600×1200.

From these tables, the scale parameter λ of Weibull PDF gives the most frequent error presented in the system on this configuration, with $\lambda = 0.0055$ m, for the combination of 8 cameras and resolution of 320×240 and with $\lambda = 0.0008$ m for the combination of 8 cameras at resolution of 1600×1200. These values are close to the medians presented above and also in Table 3.23 and Table 3.24 for each presented resolution.

Figure 3.31 compares again the absolute error of the system with the number of cameras (horizontal axis) and the resolution used (colour coded). This plot demonstrates again the overall reduction in absolute

Table 3.23: Statistics per number of cameras observing the beacon for test 03 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	176	3772	2434	2454	2380	1264	4330
mean	0.0494	0.0218	0.0119	0.0086	0.0069	0.0054	0.0049
std	0.0305	0.0123	0.0060	0.0036	0.0031	0.0024	0.0021
min	0.0104	0.0014	0.0013	0.0005	0.0005	0.0004	0.0004
25%	0.0239	0.0126	0.0077	0.0059	0.0046	0.0037	0.0033
50%	0.0386	0.0189	0.0110	0.0084	0.0067	0.0052	0.0047
75%	0.0655	0.0286	0.0150	0.0110	0.0090	0.0070	0.0062
max	0.1272	0.0763	0.0390	0.0222	0.0198	0.0141	0.0133

Table 3.24: Statistics per number of cameras observing the beacon for test 03 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	152	3680	2432	2460	2440	1284	4362
mean	0.0060	0.0029	0.0015	0.0012	0.0010	0.0008	0.0007
std	0.0037	0.0016	0.0008	0.0005	0.0005	0.0004	0.0003
min	0.0008	0.0001	0.0001	0.0001	0.0000	0.0001	0.0000
25%	0.0030	0.0017	0.0009	0.0009	0.0007	0.0005	0.0005
50%	0.0054	0.0025	0.0014	0.0012	0.0010	0.0007	0.0007
75%	0.0080	0.0039	0.0020	0.0016	0.0013	0.0010	0.0009
max	0.0179	0.0095	0.0047	0.0031	0.0027	0.0022	0.0021

Absolute error distribution in position estimation

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras

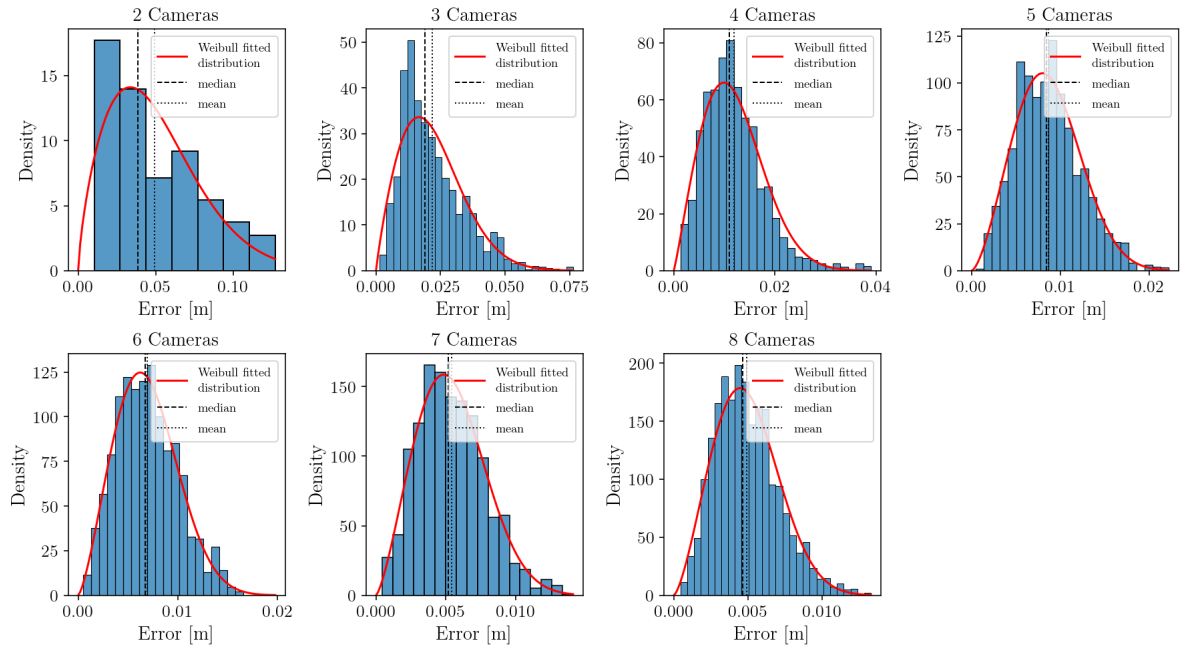


Figure 3.29: Histogram of error per camera in resolution 320×240 and test 03.

error of the system as the number of cameras increases, and with an increase in image resolution.

Error maps over the study space for this configuration of cameras test are presented below with the beacon at a fixed height like before, or in other words, only the plane XY at level $z = 2.000$ m. Figures 3.32 and 3.34 plot the error of the system as presented in section 3.5.1. Error maps were again produced for levels from $z = 0.000$ m to $z = 8.500$ m, with a step of 0.500 m. Results presented are only for resolutions

Absolute error distribution in position estimation

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

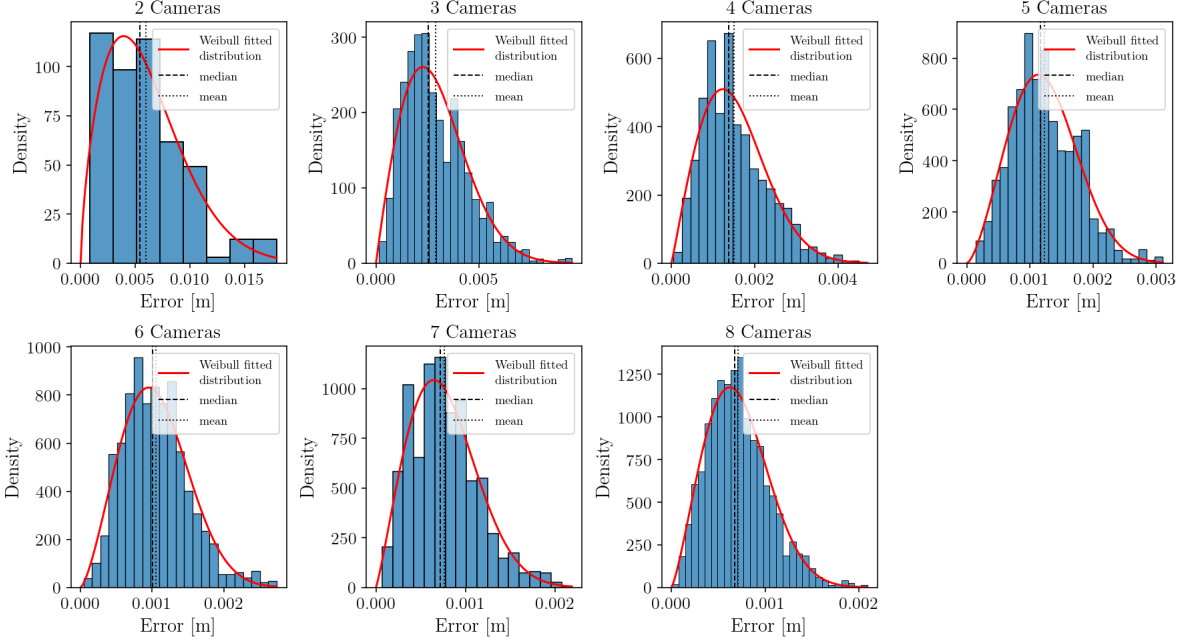


Figure 3.30: Histogram of error per camera in resolution 1600×1200 and test 03.

Table 3.25: Weibull PDF parameters and bin count per camera used in figure 3.29 for test 03 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	1.7214	0.0000	0.0557	176	7
3	1.8992	0.0000	0.0247	3772	28
4	2.1053	0.0000	0.0134	2434	25
5	2.5427	0.0000	0.0097	2454	24
6	2.3902	0.0000	0.0078	2380	24
7	2.3742	0.0000	0.0061	1264	18
8	2.4404	0.0000	0.0055	4330	28

Table 3.26: Weibull PDF parameters and bin count per camera used in figure 3.30 for test 03 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	1.6818	0.0000	0.0067	152	8
3	1.9685	0.0000	0.0033	3680	27
4	2.0507	0.0000	0.0017	2432	23
5	2.5233	0.0000	0.0014	2460	23
6	2.4210	0.0000	0.0012	2440	23
7	2.1482	0.0000	0.0009	1284	18
8	2.2815	0.0000	0.0008	4362	30

320×240 (fig. 3.32) and 1600×1200 (fig. 3.34). As in other tests, the full set of error maps for test 3 is found in appendix A, on section A.3, page A103.

The camera coverage of the simulated study space, with the number of cameras covering each location for the corresponding resolutions, is shown in figures 3.33 and 3.35.

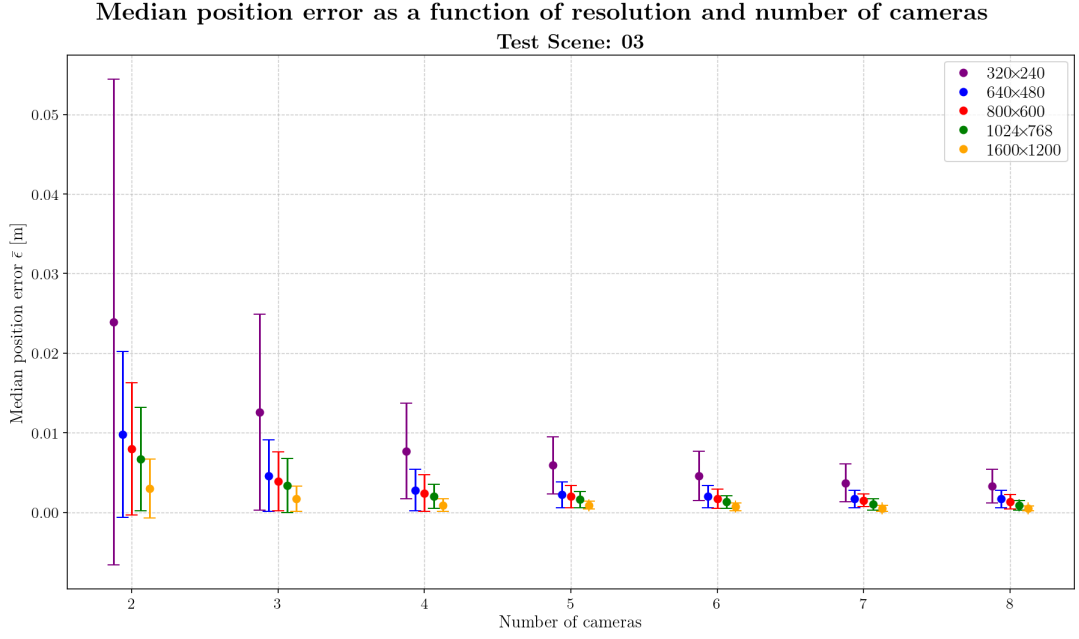


Figure 3.31: Variation of mean error $\bar{\epsilon}_{abs}$ in estimated position as a function of number of cameras and resolution used, for test 03.

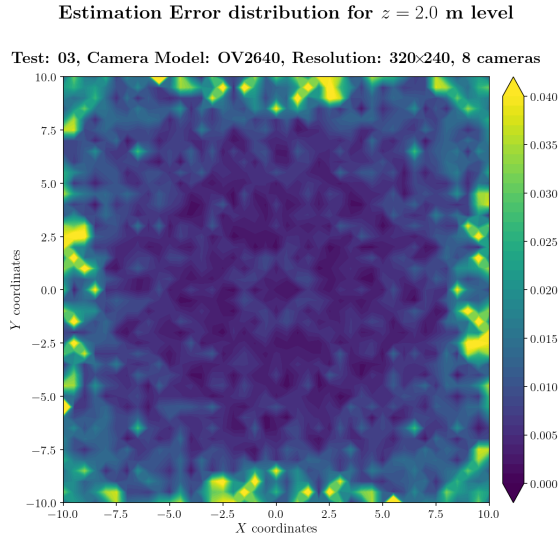


Figure 3.32: Estimation error contour map for test 03 with 8 cameras using a resolution of 320x240 pixels, at level $z = 2.00$ m.

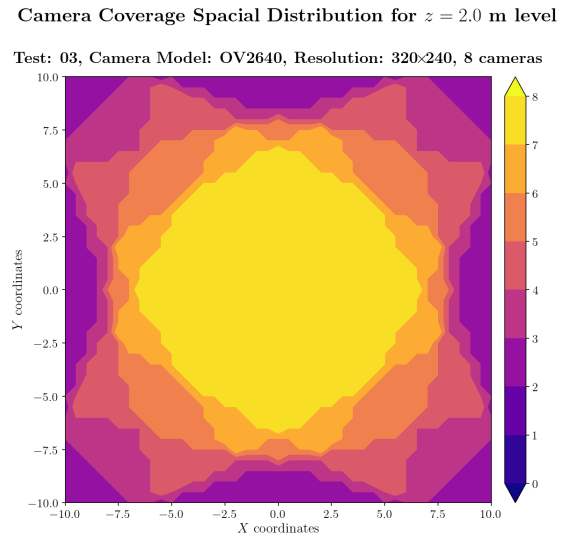
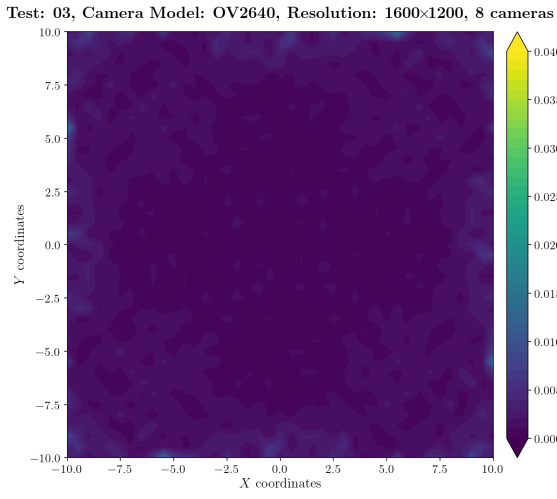


Figure 3.33: Camera coverage for test 03 at level $z = 2.00$ m.

Estimation Error distribution for $z = 2.0$ m level



Camera Coverage Spacial Distribution for $z = 2.0$ m level

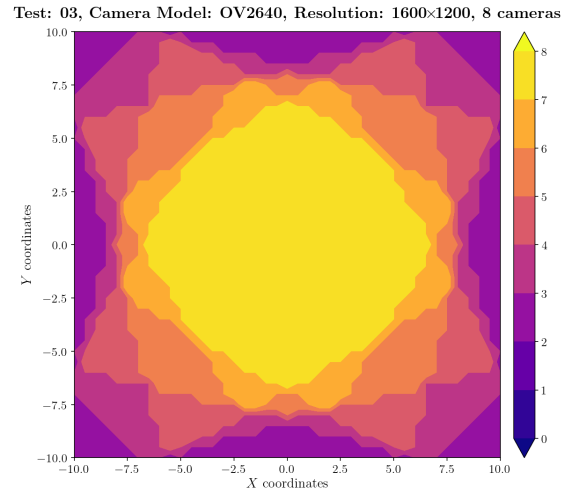


Figure 3.34: Estimation error contour map for test 03 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.

Figure 3.35: Camera coverage for test 03 at level $z = 2.00$ m.

3.5.4 Analysis of Test 04

Results for camera configuration 4 (see subsection 3.1.3) are presented in this subsection. Like in the previous configuration (test 03), cameras are also set up in a higher position, now positioned at level $z = 10$ m, but with the same orientations as test 03, with the positions and orientations for this test available at Table 3.4.

Figure 3.36 presents the error per iterated test position for this configuration and for 320×240. Results for resolution 1600×1200 are presented in Figure 3.37. Because cameras are placed at level $z = 10$ m, unlike the previous case, the plot now has points to the end of the test positions range. Colour codes encode again the number of cameras observing the beacon.

Analysing these two figures, the pattern of error resembles the one obtained in test 01, again most likely due to the similar configuration of cameras, up to a different orientation. It also resembles Figure 3.27 and Figure 3.28, because the configuration of cameras is almost identical, only differing in camera height. Using resolution 320×240, we can observe that 90% of values fall below $e_{abs} \leq 0.0233$ m, with a mean value of $\bar{e}_{abs} = 0.0118$ m, standard deviation of $\sigma = 0.0103$ m and a median value of $\sigma = 0.0088$ m, as can be seen in Table 3.27. For resolution 1600×1200, 90% of error values become smaller than $e_{abs} \leq 0.0032$ m, with an overall mean value of $\bar{e}_{abs} = 0.0016$ m, standard deviation of $\sigma = 0.0013$ m and a median value of $\bar{e}_{abs} = 0.0012$ m. Results for this resolution are compiled in Table 3.28.

The statistical data per camera number observing the beacon is presented in Table 3.29 for resolution 320×240 and in Table 3.30 for resolution 1600×1200.

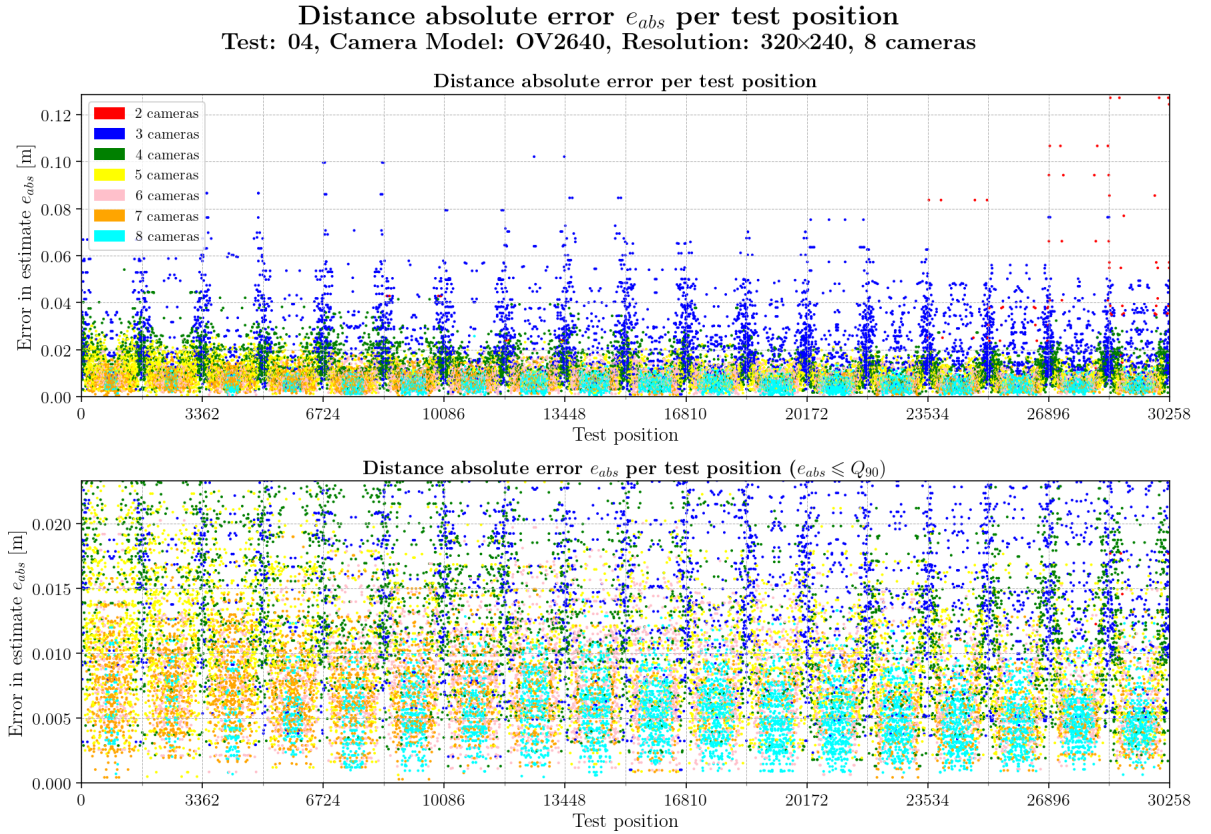


Figure 3.36: Absolute error e_{abs} per iteration for resolution 320×240 and test 04.

Distance absolute error e_{abs} per test position
Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

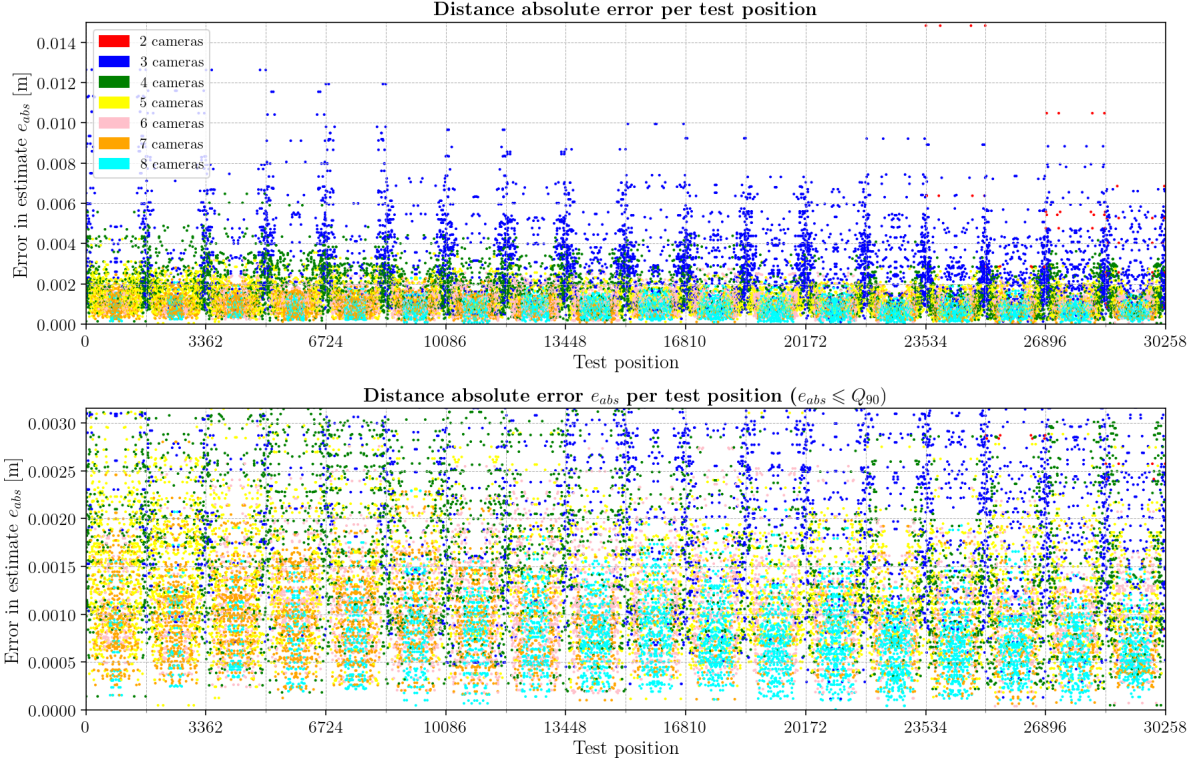


Figure 3.37: Absolute error e_{abs} per iteration for resolution 1600×1200 and test 04.

Table 3.27: Global statistics and root mean square error for test 04 with resolution 320×240.

	Abs. error	Est. error
count	30258	30258
mean	0.0118	0.0280
std	0.0103	0.0095
min	0.0003	0.0001
25%	0.0056	0.0214
50%	0.0088	0.0275
75%	0.0142	0.0341
90%	0.0233	0.0402
max	0.1272	0.0701
RMSE	0.0156	

Table 3.28: Global statistics and root mean square error for test 04 with resolution 1600×1200.

	Abs. error	Est. error
count	30258	30258
mean	0.0016	0.0038
std	0.0013	0.0013
min	0.0000	0.0000
25%	0.0008	0.0030
50%	0.0012	0.0038
75%	0.0019	0.0046
90%	0.0032	0.0055
max	0.0148	0.0119
RMSE	0.0021	

Figure 3.38 and Figure 3.39 display the histograms of error data for 320×240 and 1600×1200 resolutions, respectively, for this test. Following the approach used in all other tests, Table 3.31 contains the Weibull distribution fitted parameters to the data histograms, for the lowest resolution tested of 320×240. Similarly, Table 3.32 contains the Weibull distribution fitted parameters for resolution 1600×1200.

Looking at aforementioned tables, the most frequent error presented in the system on this configuration given by the scale parameter λ of Weibull PDF is $\lambda = 0.0060$ m, for the combination of 8 cameras and resolution of 320×240 and with $\lambda = 0.0009$ m for the combination of 8 cameras at resolution of 1600×1200. These values are close, like in previous tests, to the medians presented above and also in Table 3.29 and Table 3.30 for each resolution simulated.

As before, Figure 3.40 compares again the absolute error of the system with the number of cameras

Table 3.29: Statistics per number of cameras observing the beacon for test 04 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	80	5728	4593	4859	4772	4096	6130
mean	0.0530	0.0244	0.0141	0.0103	0.0078	0.0069	0.0053
std	0.0313	0.0141	0.0068	0.0046	0.0035	0.0031	0.0023
min	0.0122	0.0010	0.0013	0.0005	0.0006	0.0003	0.0005
25%	0.0305	0.0139	0.0093	0.0068	0.0052	0.0046	0.0035
50%	0.0386	0.0209	0.0130	0.0098	0.0075	0.0067	0.0051
75%	0.0688	0.0322	0.0180	0.0131	0.0100	0.0088	0.0067
max	0.1272	0.1021	0.0541	0.0301	0.0245	0.0190	0.0257

Table 3.30: Statistics per number of cameras observing the beacon for test 04 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	56	5464	4664	4800	4888	4184	6202
mean	0.0051	0.0033	0.0018	0.0014	0.0011	0.0009	0.0008
std	0.0036	0.0019	0.0010	0.0006	0.0005	0.0004	0.0004
min	0.0008	0.0001	0.0001	0.0000	0.0000	0.0001	0.0000
25%	0.0028	0.0019	0.0011	0.0009	0.0008	0.0006	0.0005
50%	0.0048	0.0029	0.0017	0.0013	0.0011	0.0009	0.0007
75%	0.0058	0.0043	0.0024	0.0017	0.0014	0.0012	0.0010
max	0.0148	0.0126	0.0065	0.0042	0.0032	0.0028	0.0023

Absolute error distribution in position estimation

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

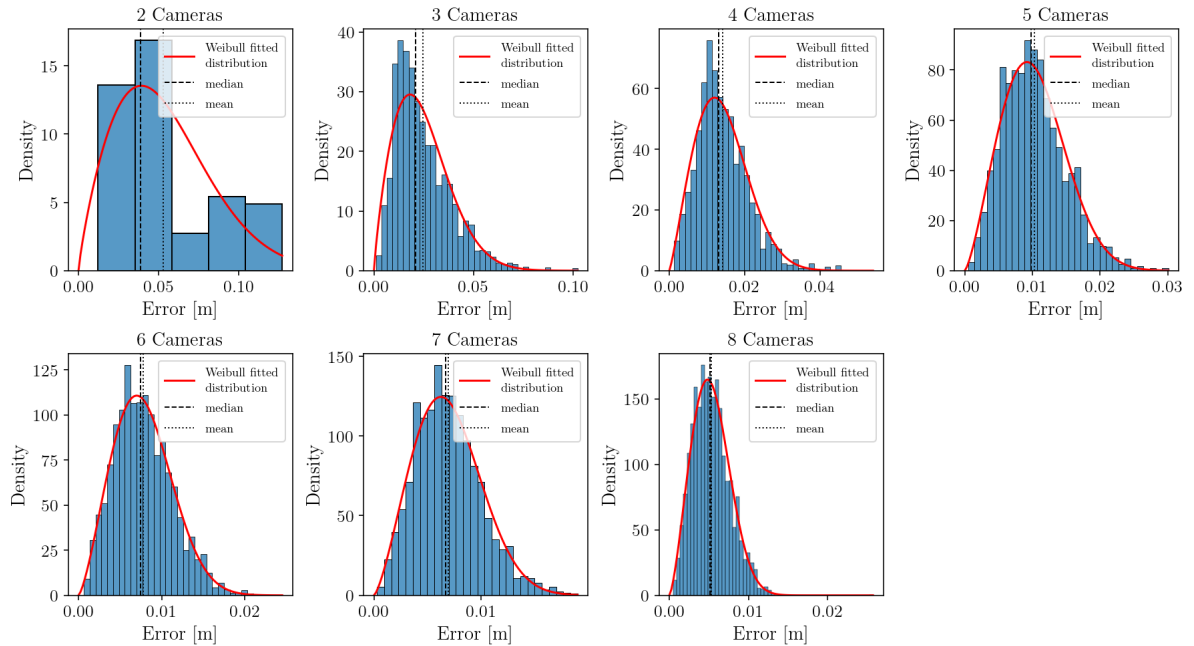


Figure 3.38: Histogram of error per camera in resolution 320×240 and test 04.

(horizontal axis) and resolution used (colour coded). This plot demonstrates again the overall reduction in absolute error of the system as the number of cameras increases, and with an increase in image resolution.

Error maps were again produced for levels from $z = 0.000\text{m}$ to $z = 8.500\text{m}$, with a step of 0.500m . Results presented are only for resolutions 320×240 (fig. 3.41) and 1600×1200 (fig. 3.43). Error maps for

Absolute error distribution in position estimation

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

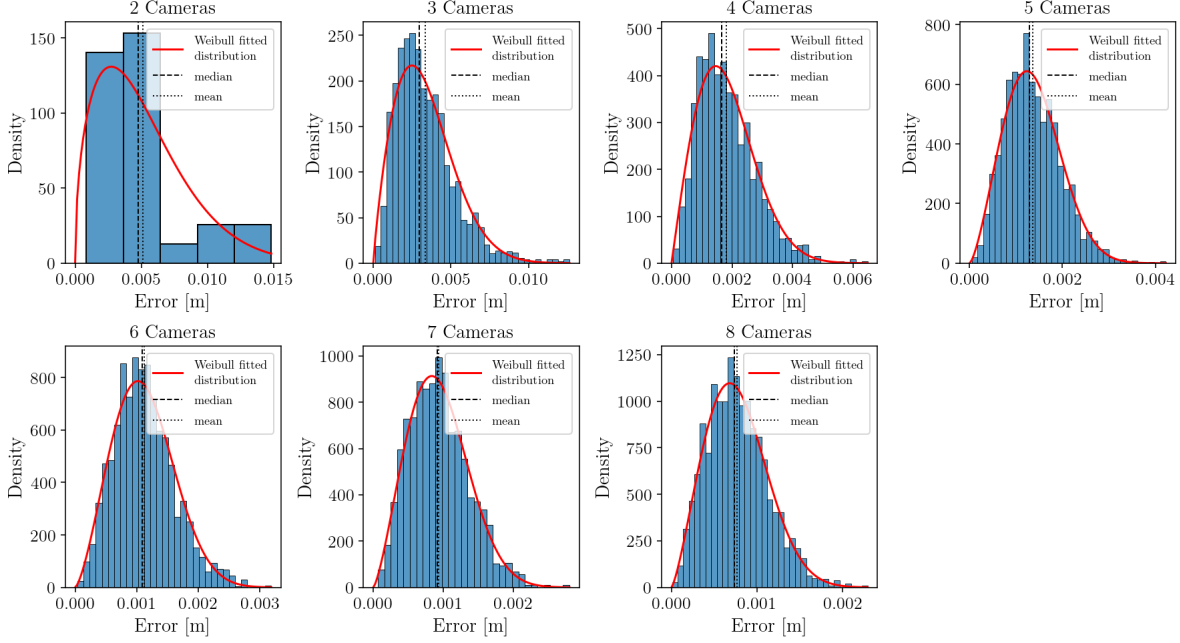


Figure 3.39: Histogram of error per camera in resolution 1600×1200 and test 04.

Table 3.31: Weibull PDF parameters and bin count per camera used in figure 3.38 for test 04 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	1.8374	0.0000	0.0601	80	5
3	1.8445	0.0000	0.0276	5728	37
4	2.1733	0.0000	0.0159	4593	37
5	2.3606	0.0000	0.0116	4859	32
6	2.3832	0.0000	0.0088	4772	34
7	2.3874	0.0000	0.0078	4096	28
8	2.4315	0.0000	0.0060	6130	57

Table 3.32: Weibull PDF parameters and bin count per camera used in figure 3.39 for test 04 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	1.4993	0.0000	0.0057	56	5
3	1.8576	0.0000	0.0038	5464	34
4	2.0151	0.0000	0.0021	4664	33
5	2.4285	0.0000	0.0015	4800	34
6	2.4470	0.0000	0.0013	4888	32
7	2.3568	0.0000	0.0011	4184	30
8	2.3150	0.0000	0.0009	6202	35

the plane XY at level $z = 2.000$ m are presented below. Figures 3.41 and 3.43 plot the error of the system as presented in section 3.5.1. Error maps for this test configuration were again produced for levels from $z = 0.000$ m to $z = 8.500$ m, with a step of 0.500 m. Results presented are only for resolutions 320×240 (fig. 3.41) and 1600×1200 (fig. 3.43). As in other tests, the full set of error maps for test 4 is found in appendix A, on section A.4, page A134.

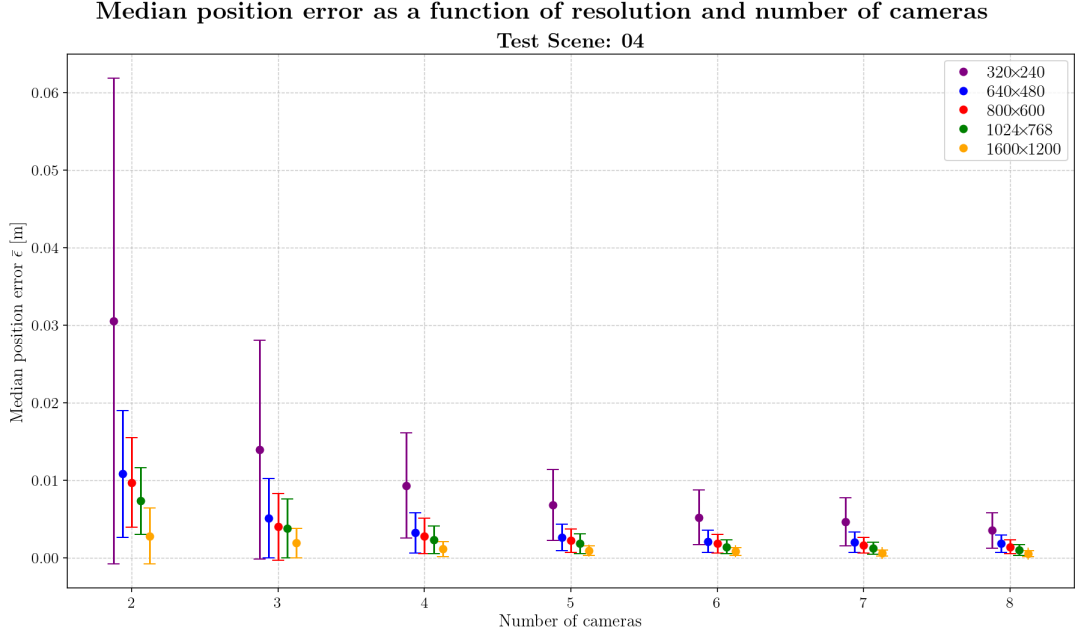


Figure 3.40: Variation of mean absolute error $\bar{\epsilon}_{abs}$ in estimated position as a function of number of cameras and resolution used, for test 04.

To the right of these error maps, the camera coverage over the simulated study space, with the number of cameras covering each location for the corresponding resolutions, is shown in figures 3.42 and 3.44.

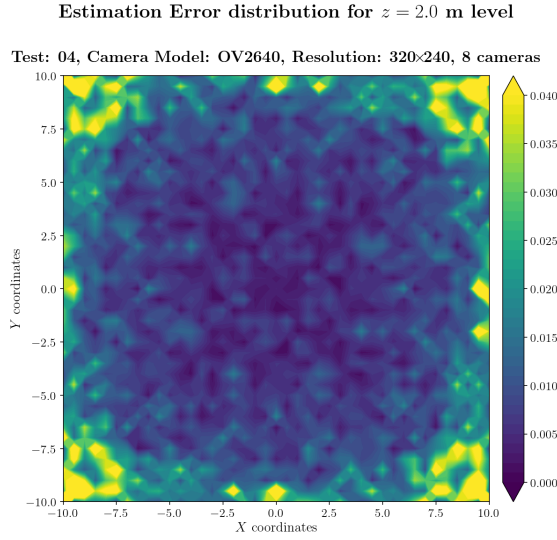


Figure 3.41: Estimation error contour map for test 04 with 8 cameras using a resolution of 320x240 pixels, at level $z = 2.00$ m.

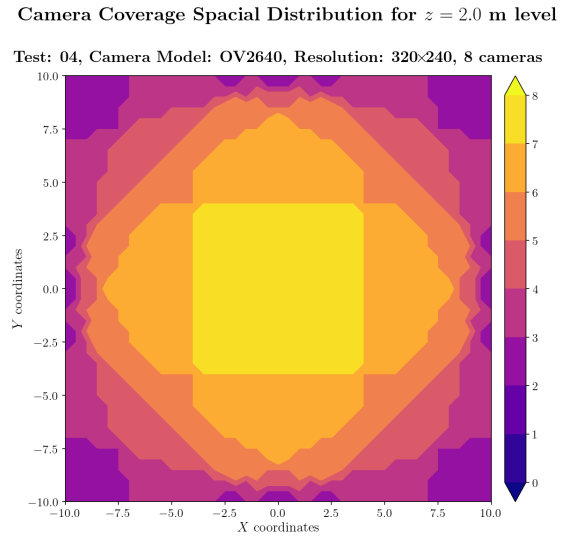
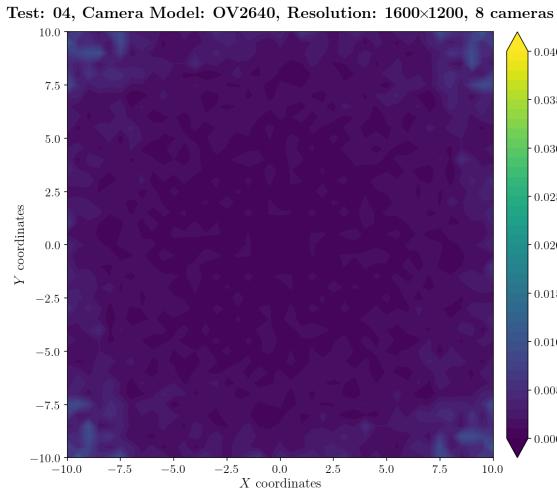


Figure 3.42: Camera coverage for test 04 at level $z = 2.00$ m.

Estimation Error distribution for $z = 2.0$ m level



Camera Coverage Spacial Distribution for $z = 2.0$ m level

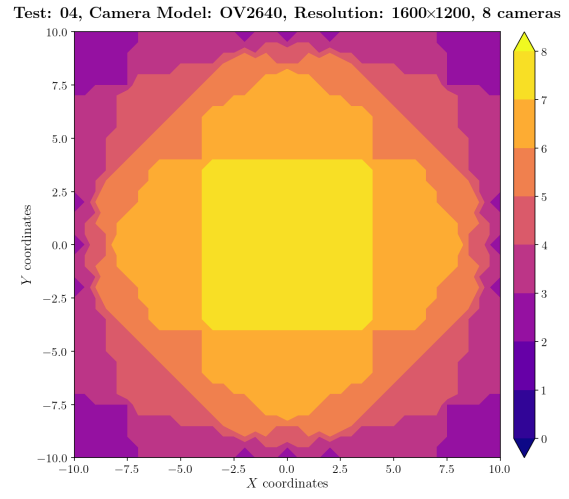


Figure 3.43: Estimation error contour map for test 04 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.

Figure 3.44: Camera coverage for test 04 at level $z = 2.00$ m.

3.5.5 Analysis of Test 05

The simulated test configuration 5 was created to test the use of the system in a larger environment and with an object that may cause the beacon occlusion from the point of view of some of the cameras. Cameras are set up in a different configuration from all other tests, with the positions and orientations for this test available at Table 3.5. Figure 3.45 presents the error per iterated test position for this configuration and for 320×240.

Results for resolution 1600×1200 are presented in Figure 3.46, with colour codes encoding as before the number of cameras observing the beacon. The occluding object chosen to simulate the obstruction of the view of some cameras was an aircraft model (C-130). Because of the presence of this object, a large concentration of red points from test positions 6724 to 13448. This clustering of red points, associated with only two cameras observing the beacon, is caused by the proximity of the beacon position to the aircraft model. Some of the simulated points even happened to be inside the aircraft, and therefore, weren't detected by any cameras.

Analysing these two figures, the pattern of error became very different from the previous test and also shows an increase in error in the estimation of positions. This can be associated with the larger space covered by the cameras and also their very different orientation, compared to previous tests. Starting by looking at resolution 320×240, it is observable that 90% of values fall below $e_{abs} \leq 0.0554$ m, with a mean value of $\bar{e}_{abs} = 0.0305$ m, standard deviation of $\sigma = 0.0271$ m and a median value of $\sigma = 0.0238$ m, as can be seen in Table 3.33. For resolution 1600×1200, 90% of error values are less than $e_{abs} \leq 0.0078$ m, with an overall mean value of $\bar{e}_{abs} = 0.0045$ m, standard deviation of $\sigma = 0.0060$ m and a median value of $\bar{e}_{abs} = 0.0034$ m. Results for this resolution are compiled in Table 3.34.

Like in other tests, the statistical data per camera number observing the beacon is presented in Table 3.35 for resolution 320×240 and in Table 3.36 for resolution 1600×1200.

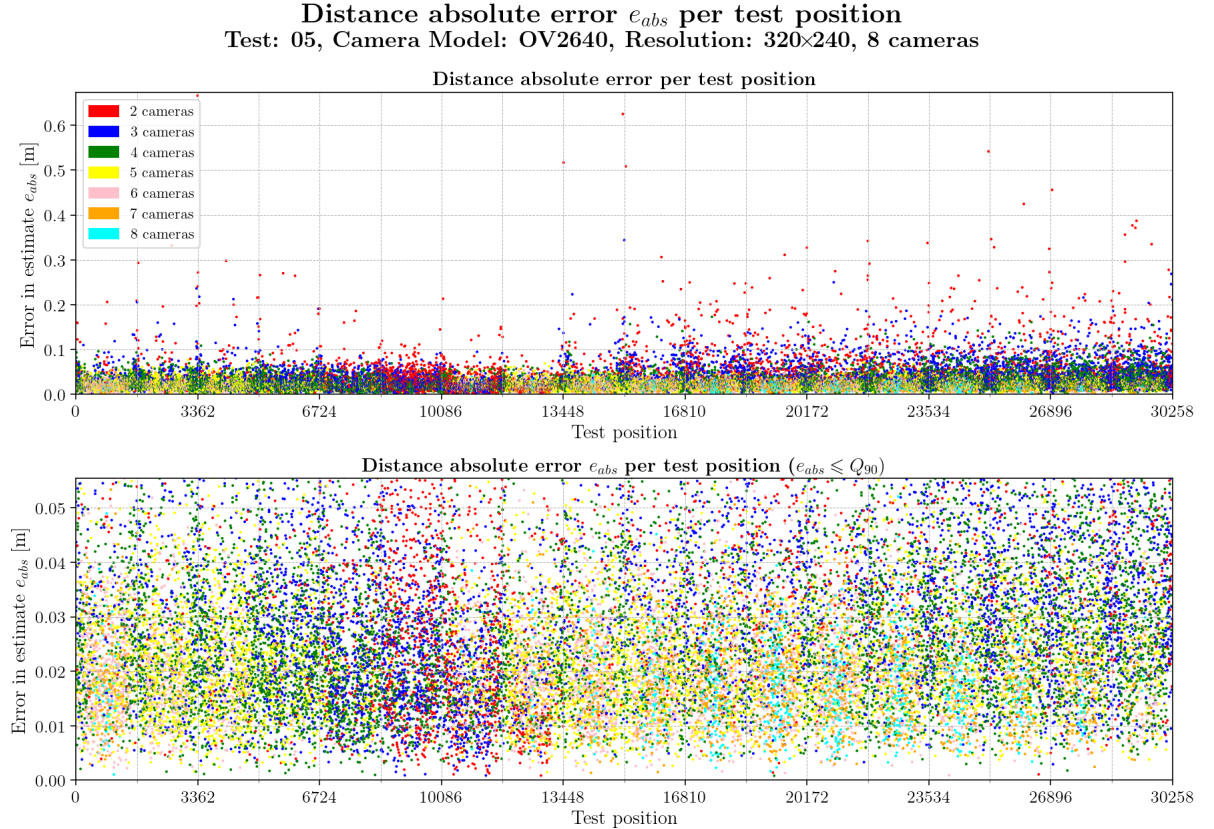


Figure 3.45: Absolute error e_{abs} per iteration for resolution 320×240 and test 05.

Distance absolute error e_{abs} per test position
Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

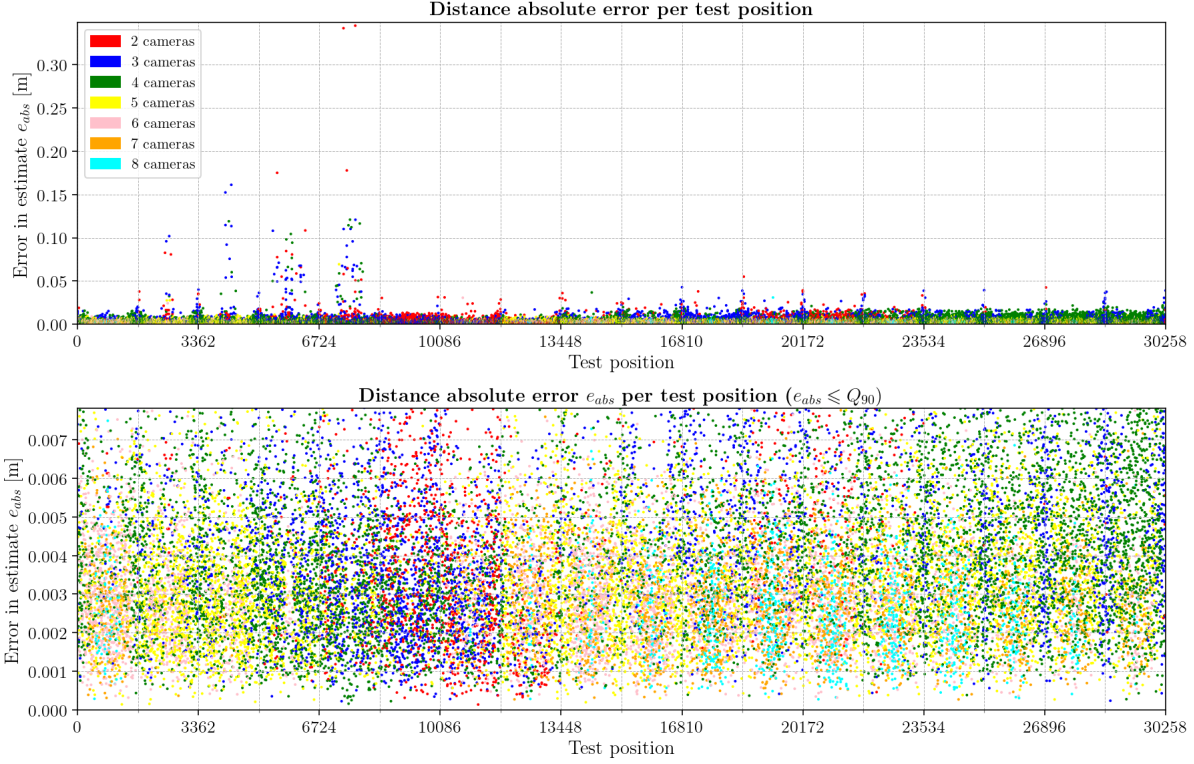


Figure 3.46: Absolute error e_{abs} per iteration for resolution 1600×1200 and test 05.

Table 3.33: Global statistics and root mean square error for test 05 with resolution 320×240.

	Abs. error	Est. error
count	28632	28632
mean	0.0305	0.0528
std	0.0271	0.0253
min	0.0008	0.0000
25%	0.0158	0.0349
50%	0.0238	0.0523
75%	0.0360	0.0698
90%	0.0554	0.0855
max	0.6663	0.1880
RMSE	0.0408	

Table 3.34: Global statistics and root mean square error for test 05 with resolution 1600×1200.

	Abs. error	Est. error
count	29228	29228
mean	0.0045	0.0084
std	0.0060	0.0051
min	0.0001	0.0000
25%	0.0023	0.0058
50%	0.0034	0.0082
75%	0.0051	0.0107
90%	0.0078	0.0128
max	0.3452	0.1647
RMSE	0.0075	

Histograms of error data for this test are shown in Figure 3.47 and Figure 3.48 for 320×240 and 1600×1200 resolutions, respectively. Recurring to the approach used in all other tests, Table 3.37 contains the Weibull distribution fitted parameters to the data histograms, for the lowest resolution tested of 320×240, and Table 3.38 contains the Weibull distribution fitted parameters for resolution 1600×1200.

In these tables, it can be seen that the scale parameter λ of the Weibull PDF is $\lambda = 0.0195$ m, for the combination of 8 cameras and resolution of 320×240 and is $\lambda = 0.0029$ m for the combination of 8 cameras at resolution of 1600×1200.

As before, Figure 3.49 compares again the absolute error of the system with the number of cameras (horizontal axis) and the resolution used (colour coded). This plot demonstrates again the overall reduction in absolute error of the system as the number of cameras increases, and with an increase in image resolution.

Table 3.35: Statistics per number of cameras observing the beacon for test 05 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	3224	6007	7347	5349	3640	2018	1047
mean	0.0549	0.0387	0.0290	0.0228	0.0201	0.0188	0.0173
std	0.0541	0.0277	0.0178	0.0111	0.0093	0.0087	0.0076
min	0.0008	0.0008	0.0008	0.0014	0.0009	0.0013	0.0010
25%	0.0222	0.0202	0.0168	0.0149	0.0135	0.0122	0.0117
50%	0.0400	0.0320	0.0251	0.0212	0.0186	0.0177	0.0163
75%	0.0668	0.0485	0.0364	0.0289	0.0256	0.0237	0.0218
max	0.6663	0.3438	0.1727	0.0838	0.0746	0.0578	0.0601

Table 3.36: Statistics per number of cameras observing the beacon for test 05 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	1882	4590	8759	5939	4199	2473	1386
mean	0.0072	0.0065	0.0050	0.0034	0.0030	0.0028	0.0026
std	0.0145	0.0086	0.0050	0.0019	0.0015	0.0012	0.0014
min	0.0001	0.0002	0.0002	0.0001	0.0003	0.0003	0.0003
25%	0.0027	0.0029	0.0027	0.0022	0.0020	0.0019	0.0017
50%	0.0047	0.0046	0.0040	0.0031	0.0028	0.0027	0.0024
75%	0.0081	0.0073	0.0061	0.0043	0.0038	0.0035	0.0033
max	0.3452	0.1612	0.1209	0.0693	0.0306	0.0081	0.0309

Absolute error distribution in position estimation

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

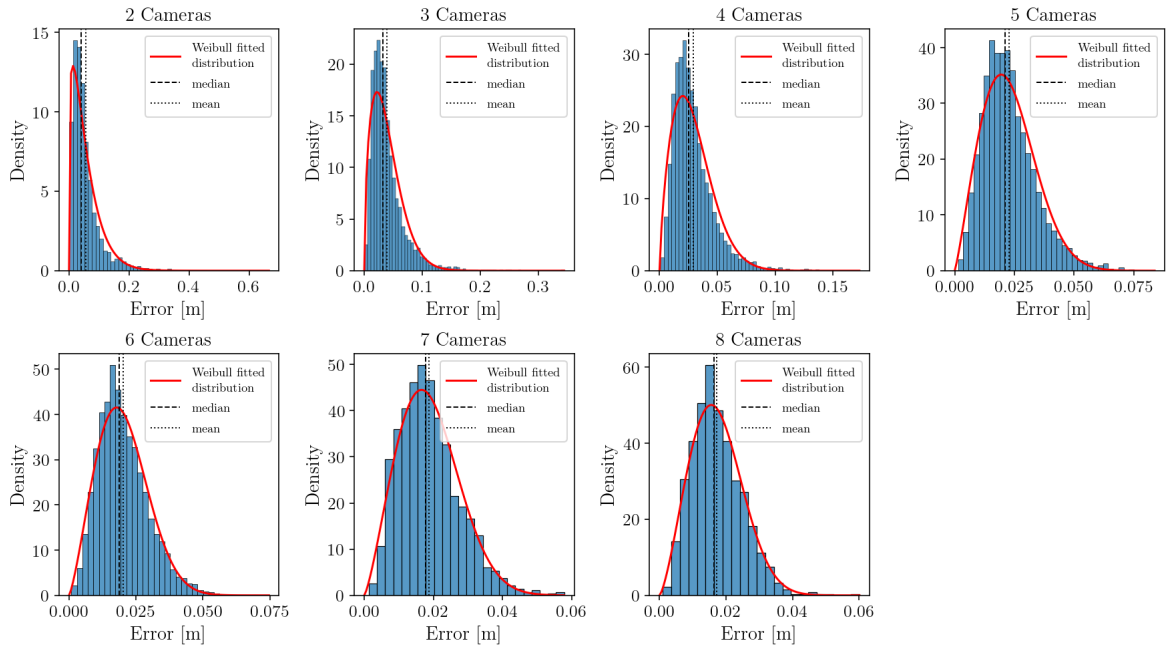


Figure 3.47: Histogram of error per camera in resolution 320×240 and test 05.

Error maps were also produced, but for this test, levels ranged from $z = 0.000$ m to $z = 17.000$ m, with a step of 1.000 m. Results presented are only for resolutions 320×240 (fig. 3.50) and 1600×1200 (fig. 3.52). As in other tests, the full set of error maps for test 5 is found in appendix A, on section A.5, page A185. Error maps for the plane XY at level $z = 2.000$ m are presented below. Figures 3.50 and 3.52 plot the error of the system (see section 3.5.1). Error maps for this test configuration were again produced for levels from $z = 0.000$ m to $z = 17.000$ m, with a step of 1.000 m. Results presented are only for resolutions

Absolute error distribution in position estimation

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

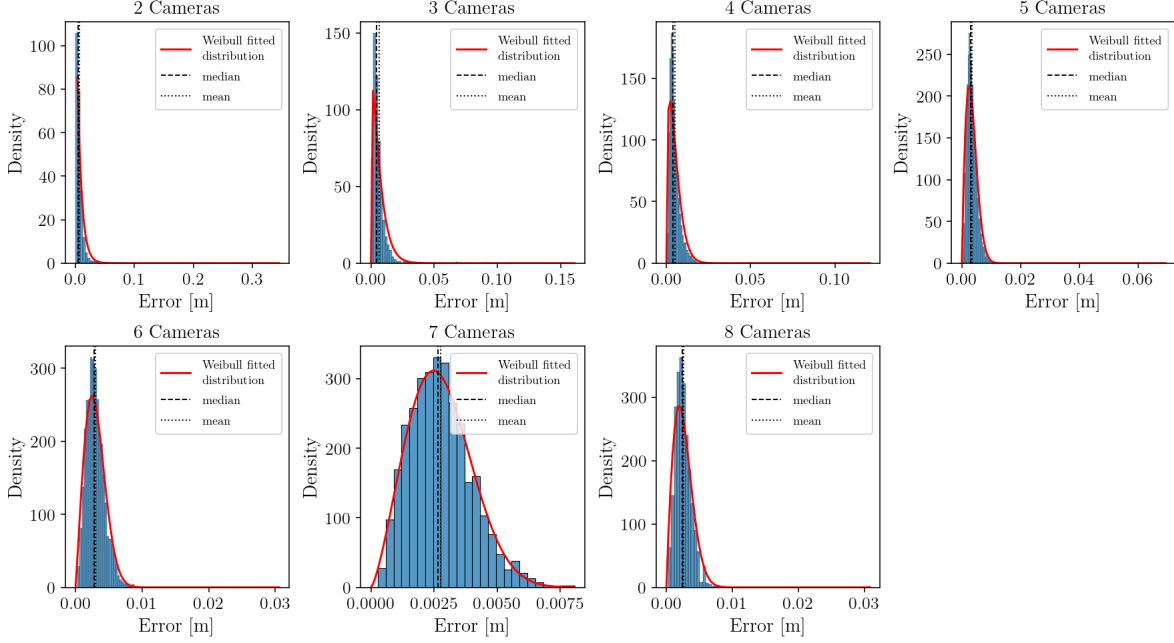


Figure 3.48: Histogram of error per camera in resolution 1600×1200 and test 05.

Table 3.37: Weibull PDF parameters and bin count per camera used in figure 3.47 for test 05 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	1.1928	0.0000	0.0586	3224	53
3	1.5395	0.0000	0.0434	6007	65
4	1.7489	0.0000	0.0327	7347	54
5	2.1761	0.0000	0.0258	5349	38
6	2.2998	0.0000	0.0228	3640	36
7	2.2969	0.0000	0.0212	2018	24
8	2.3986	0.0000	0.0195	1047	23

Table 3.38: Weibull PDF parameters and bin count per camera used in figure 3.48 for test 05 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
2	0.9987	0.0000	0.0072	1882	84
3	1.1391	0.0000	0.0068	4590	89
4	1.3607	0.0000	0.0056	8759	143
5	1.8534	0.0000	0.0038	5939	186
6	2.1039	0.0000	0.0034	4199	96
7	2.3785	0.0000	0.0031	2473	25
8	1.9066	0.0000	0.0029	1386	70

320×240 (fig. 3.50) and 1600×1200 (fig. 3.52). As in other tests, the full set of error maps for test 5 is found in appendix A, on section A.5, page A185.

On Figure 3.50, a white coloured patch indicates where there are no error measurements available. That can be seen on the camera coverage plot to the right, namely Figure 3.51, where the corresponding patch

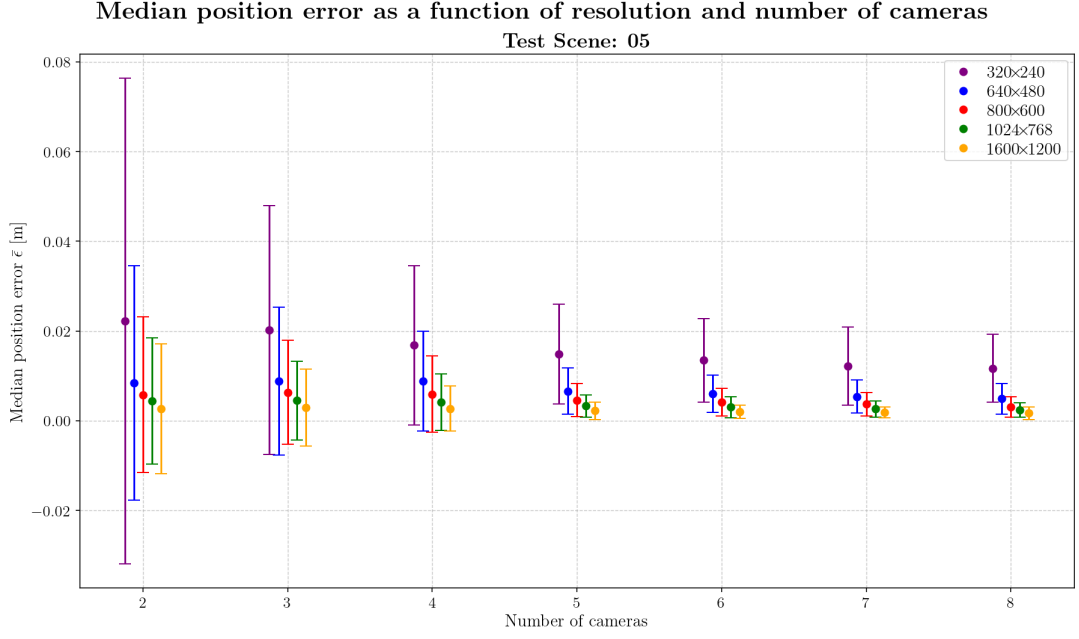


Figure 3.49: Variation of mean error $\bar{\epsilon}_{abs}$ in estimated position as a function of number of cameras detecting the beacon and resolution used, for test 05.

is coloured dark purple, which means 0 cameras observing the beacon. This corresponds to the situation of beacon total occlusion, because it was located inside the aircraft structure. and therefore not visible.

Camera coverage over the simulated study space, with the number of cameras covering each location for the corresponding resolutions, is shown in figures 3.51 and 3.53, for their corresponding resolutions.

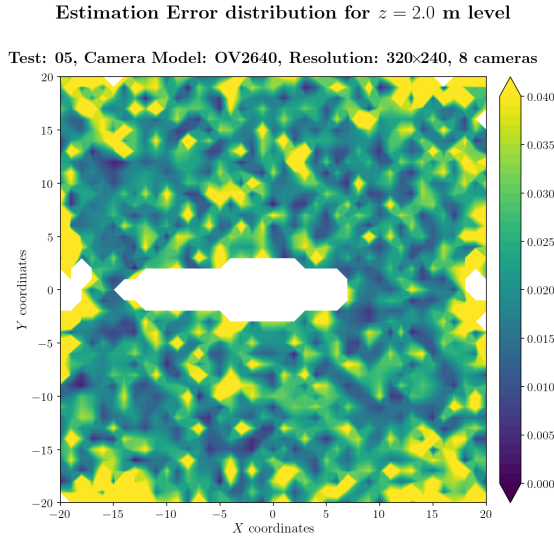


Figure 3.50: Estimation error contour map for test 05 with 8 cameras using a resolution of 320x240 pixels, at level $z = 2.00$ m.

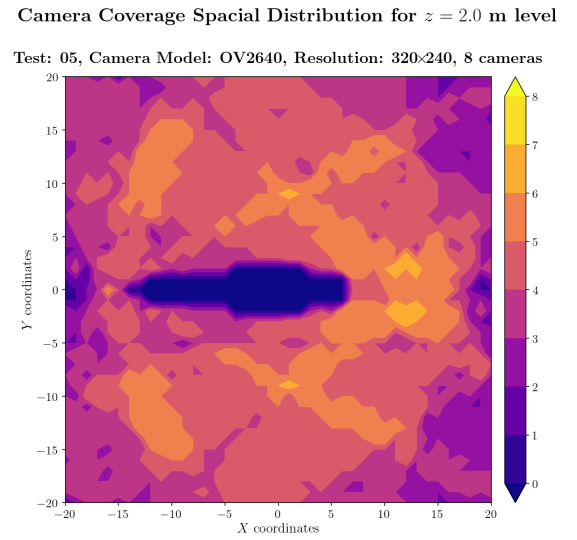
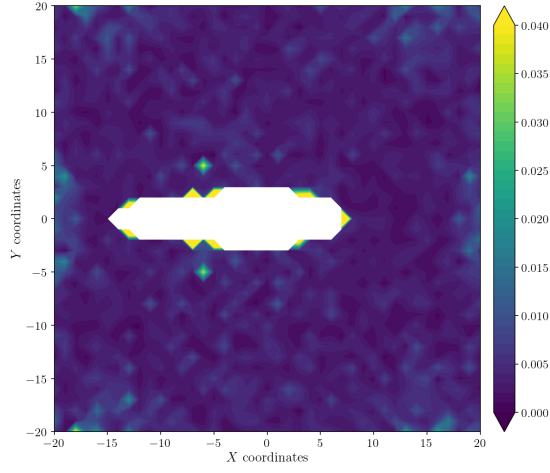


Figure 3.51: Camera coverage for test 05 at level $z = 2.00$ m.

Estimation Error distribution for $z = 2.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

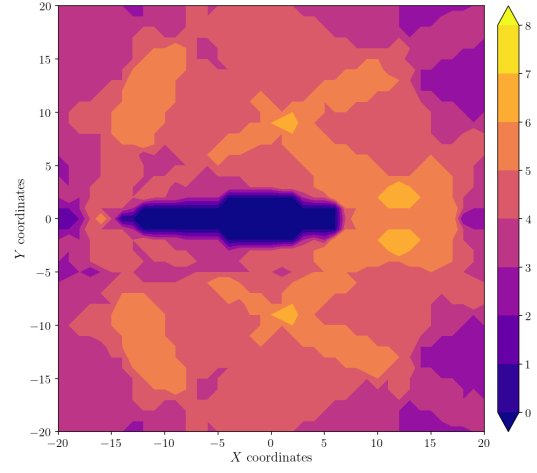


Figure 3.52: Estimation error contour map for test 05 with 8 cameras using a resolution of 1600×1200 pixels, at level $z = 2.00$ m.

Figure 3.53: Camera coverage for test 05 at level $z = 2.00$ m.

3.6 Trajectory 1 Simulation results

3.6.1 Analysis of Test 01

In this section, the response of the system tracking a beacon following a given trajectory, which in this scenario is a basic circular trajectory inclined at an angle $\phi = 15$ deg around the x axis of the coordinate system (see section 3.3, Equation 3.1).

Figure 3.54 shows the simulated trajectory in a dashed black line and the estimated points in red (which, due to the scale used in the plots, seem to form a connected curve) using this work's visual tracking algorithm. The left plot is the trajectory projected onto the XY plane, and the right plot is its projection onto the XZ plane. It can be observed by analysing Table 3.39 and Figure 3.55 that the algorithm can recover the beacon position 90% of the time with an accuracy better than $e_{abs} \leq 0.0084$ m when using a low resolution of 320×240 . Also, in the aforementioned table, it can be seen that the mean error is $\bar{e}_{abs} = 0.0053$ m, standard deviation $\sigma = 0.0023$ m and median $\tilde{e}_{abs} = 0.0050$ m.

In the case of using a high resolution, namely 1600×1200 , Table 3.40 and Figure 3.56 show that the error in position determination decreases to 90% of values with $e_{abs} \leq 0.0012$ m, with mean error $\bar{e}_{abs} = 0.0008$ m, standard deviation $\sigma = 0.0003$ m and median $\tilde{e}_{abs} = 0.0007$ m.

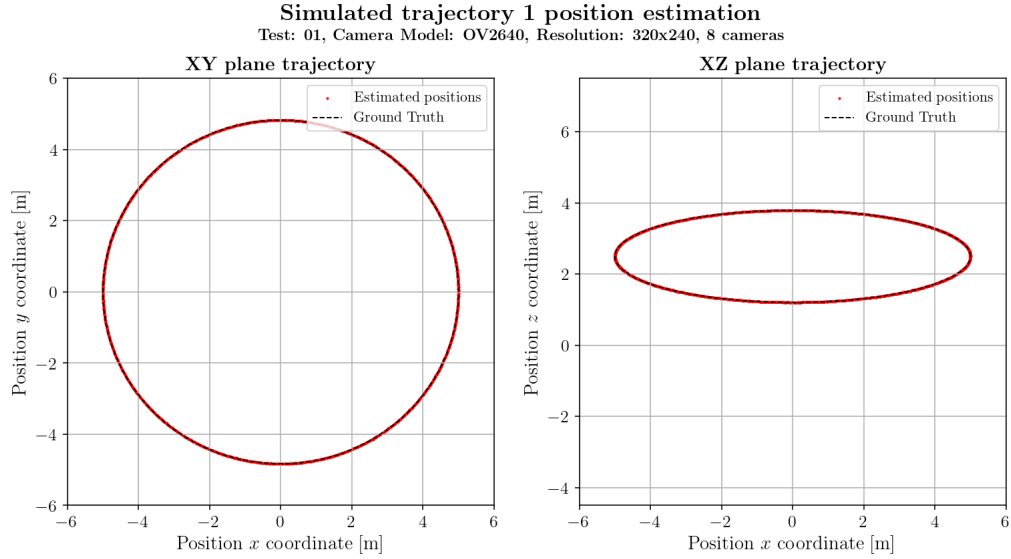


Figure 3.54: Trajectory 1 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 01

Figures 3.57 and 3.58 show the error per coordinate component for resolutions 320×240 and 1600×1200 respectively, for every point along the trajectory.

In the case of lower resolution, the maximum error in x, y and z coordinates is of the order $e_x \sim e_y \sim e_z \sim 0.0010$ m. The analysis of the error per coordinate is better evaluated by looking at the root mean square error (RMSE) between the simulated trajectory points and the estimated points, which resulted in $RMSE_x = 0.0036$ m, $RMSE_y = 0.0037$ m and $RMSE_z = 0.0026$ m.

Referring to resolution 1600×1200 , the maximum error in x, y decreases to approximately $e_x \sim e_y \sim 0.002$ m and z coordinates also decreases to approximately $e_z \sim 0.001$ m. The analysis of the error per coordinate is better evaluated by looking at the root mean square error (RMSE) between the simulated trajectory points and the estimated points, which resulted in $RMSE_x = 0.0005$ m, $RMSE_y = 0.0005$ m and $RMSE_z = 0.0004$ m.

Like in the systematic cases presented in previous sections, the absolute error distribution has been plotted as a histogram. This trajectory was always followed by the 8 cameras available in the simulation, so Figure 3.59 shows the histogram for resolution 320×240 and Figure 3.60 for resolution 1600×1200 . A Weibull PDF was again fitted to these data distributions, yielding the parameters presented in Tables 3.41 and 3.42 for the respective resolutions. In the case of resolution 320×240 , the Weibull PDF has a

Distance absolute error e_{abs} per test position
Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

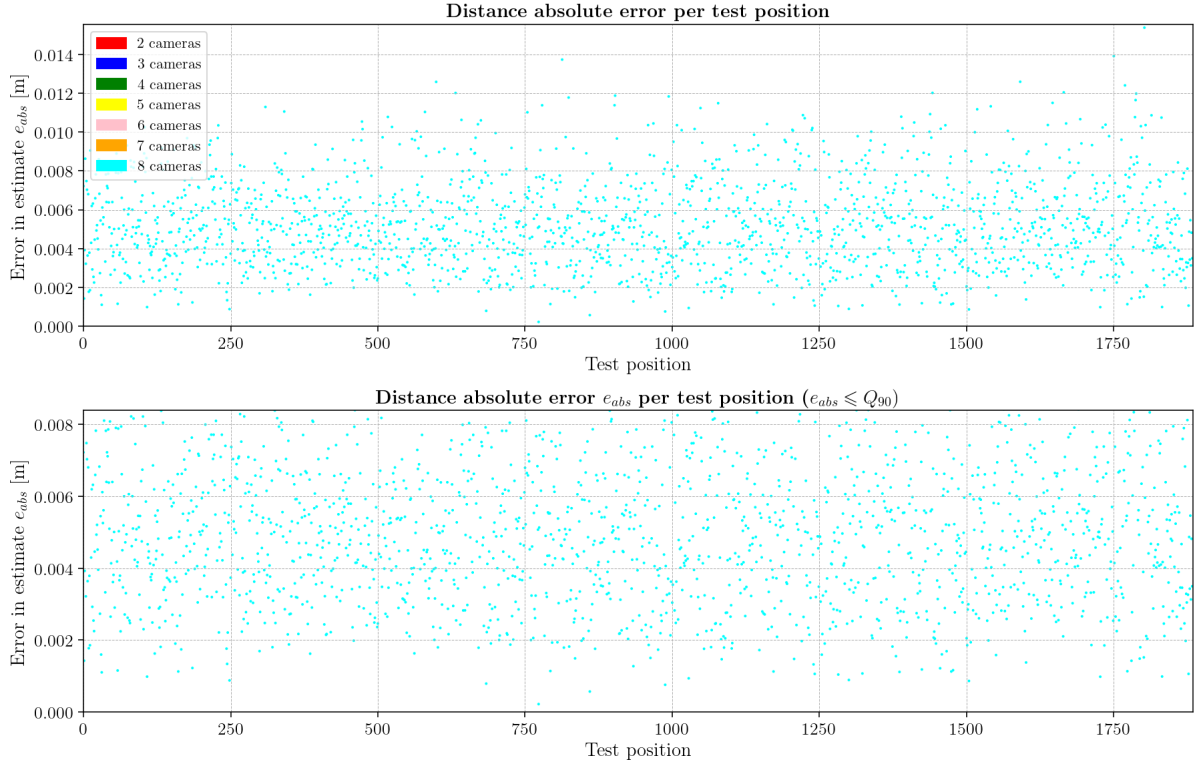


Figure 3.55: Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 01.

Table 3.39: Global statistics and root mean square error for test 01 with resolution 320×240.

	Abs. error	Est. error
count	1884	1884
mean	0.0053	0.0256
std	0.0023	0.0064
min	0.0002	0.0084
25%	0.0036	0.0209
50%	0.0050	0.0253
75%	0.0066	0.0299
90%	0.0084	0.0341
max	0.0154	0.0581
RMSE	0.0057	

Table 3.40: Global statistics and root mean square error for test 01 with resolution 1600×1200.

	Abs. error	Est. error
count	1884	1884
mean	0.0008	0.0038
std	0.0003	0.0009
min	0.0001	0.0014
25%	0.0005	0.0031
50%	0.0007	0.0037
75%	0.0010	0.0043
90%	0.0012	0.0050
max	0.0029	0.0158
RMSE	0.0008	

scale parameter $\lambda = 0.0059$ m, which like in the systematic cases, approaches the median value of absolute error along the trajectory. In case of resolution 1600×1200, the scale parameter drops to $\lambda = 0.0009$ m.

Table 3.41: Weibull PDF parameters and bin count per camera used in figure 3.59 for test 01 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.4758	0.0000	0.0059	1884	24

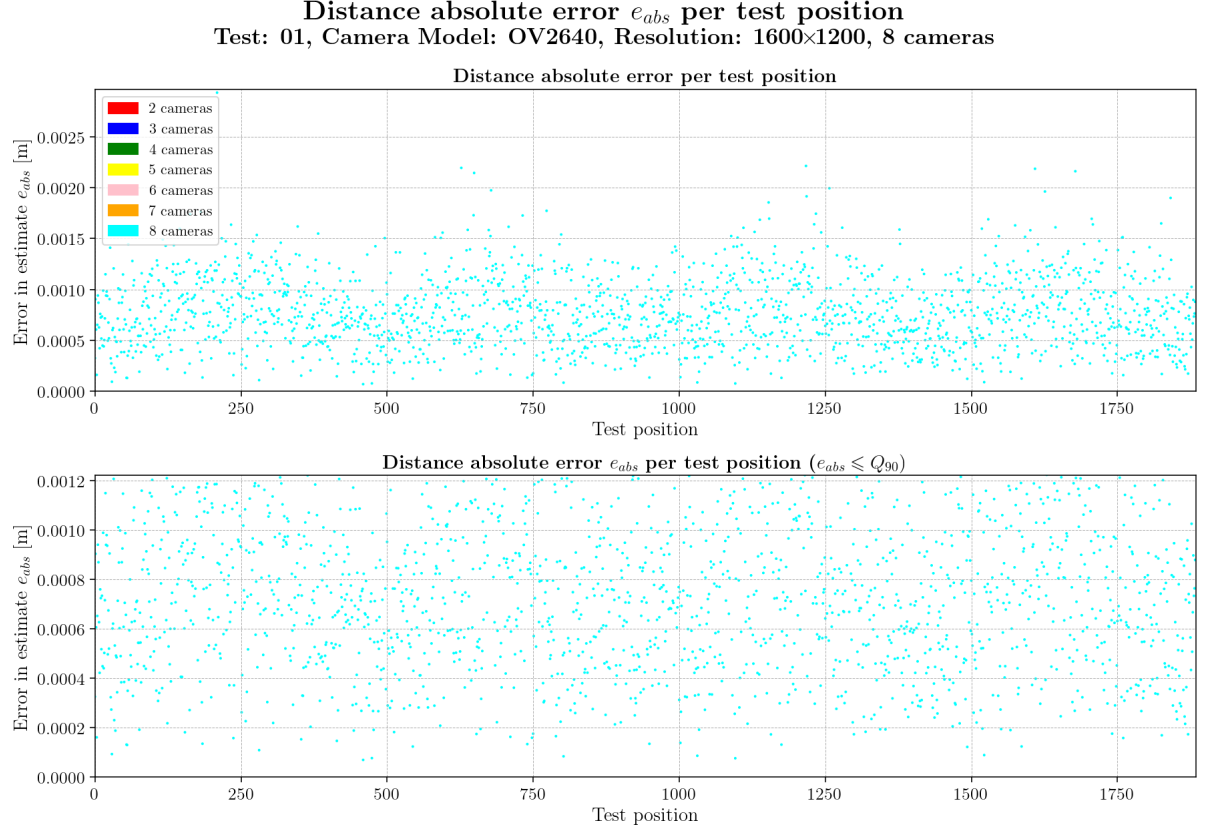


Figure 3.56: Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 01.

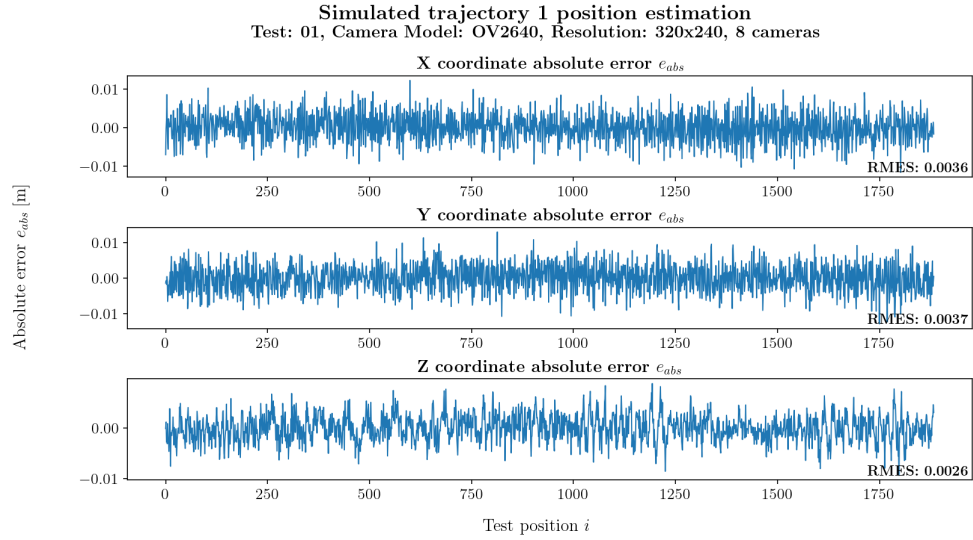


Figure 3.57: Real error in position estimation for trajectory 1 at resolution 320×240 and test 01, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

Table 3.42: Weibull PDF parameters and bin count per camera used in figure 3.60 for test 01 and resolution 1600×1200.

N ^r Cameras	Shape	Loc	Scale	Count	N ^r bins
8	2.3664	0.0000	0.0009	1884	30

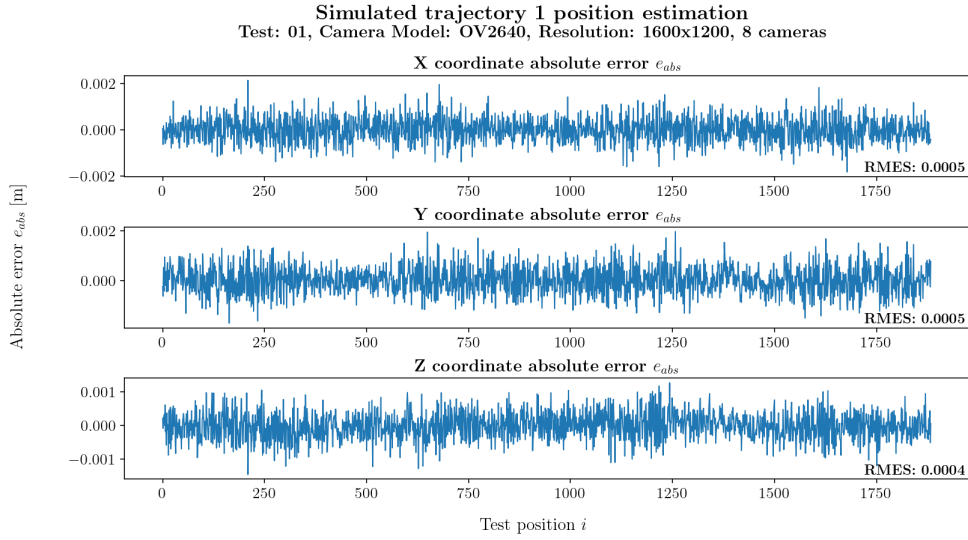


Figure 3.58: Real error in position estimation for trajectory 1 at resolution 1600×1200 and test 01, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

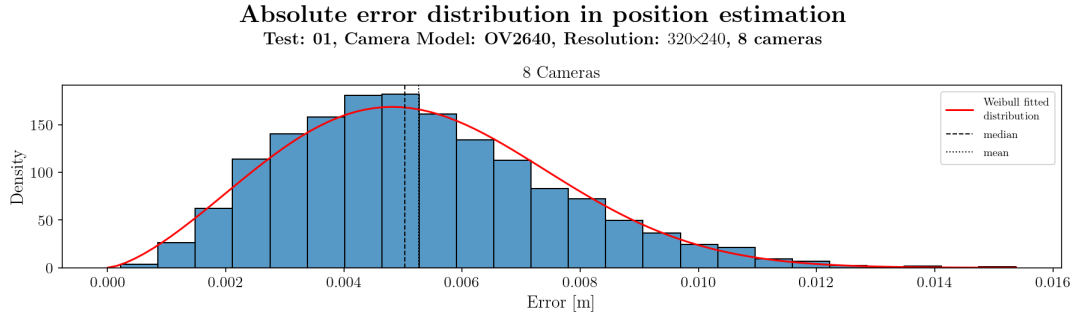


Figure 3.59: Error distribution histogram for trajectory 1 and test 01 at resolution 320×240.

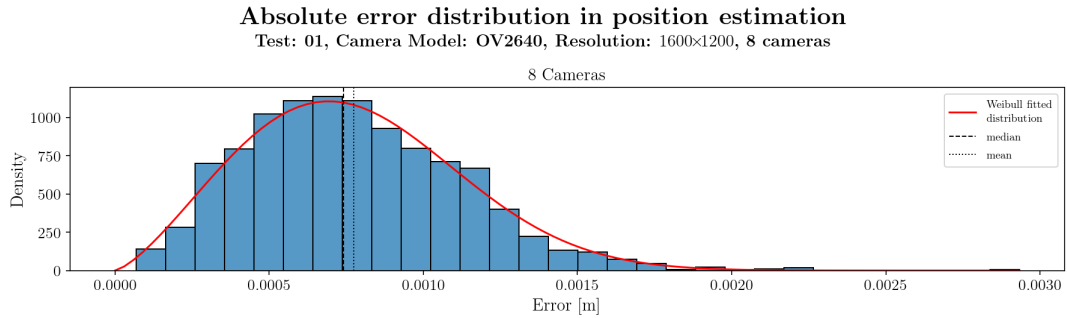


Figure 3.60: Error distribution histogram for trajectory 1 and test 01 at resolution 1600×1200.

3.6.2 Analysis of Test 02

Here, the recovery of trajectory 1 under camera test configuration 2 is evaluated. Like in the previous case, Figure 3.61 shows the simulated trajectory in a dashed black line and the estimated points in red (which, again due to the scale used in the plots, becomes apparently a connected curve) using this work's visual tracking algorithm. As before, the left plot is the trajectory projected onto the XY plane and the right plot is its projection onto the XZ plane. It can be observed by analysing Table 3.43 and Figure 3.62 that the algorithm recovers the beacon position 90% of the time with an error smaller than $e_{abs} \leq 0.0098$ m when using a low resolution of 320×240 . Also, in the aforementioned table, it can be seen that the mean error is $\bar{e}_{abs} = 0.0062$ m, standard deviation $\sigma = 0.0027$ m and median $\tilde{e}_{abs} = 0.0058$ m.

When using high resolution of 1600×1200 , Table 3.44 and Figure 3.63 show that the error in position determination decreases to 90% of values with $e_{abs} \leq 0.0013$ m, with mean error $\bar{e}_{abs} = 0.0008$ m, standard deviation $\sigma = 0.0004$ m and median $\tilde{e}_{abs} = 0.0008$ m.

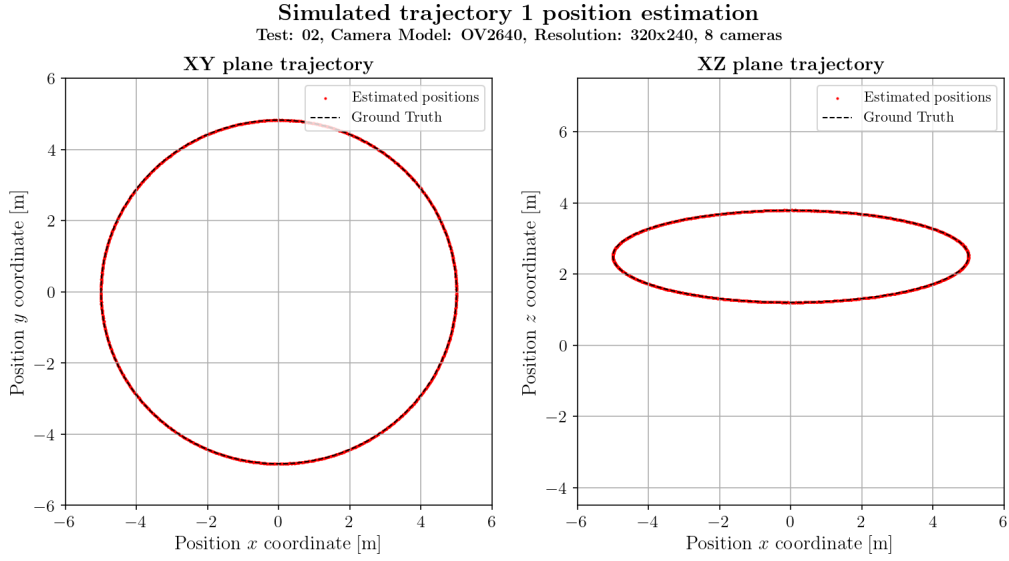


Figure 3.61: Trajectory 1 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 02

Figures 3.64 and 3.65 present the error of the system per coordinate component for resolutions 320×240 and 1600×1200 , respectively, for every point recovered along the trajectory.

When using a resolution of 320×240 , the maximum error in x, y and z coordinates is of the order $e_x \sim e_y \sim e_z \sim 0.0010$ m, like in previous test analysis of trajectory 1. Again, the analysis of the error per coordinate is better evaluated by looking at the root mean square error (RMSE) between the simulated trajectory points and the estimated points, which resulted in $RMSE_x = 0.0043$ m, $RMSE_y = 0.0043$ m and $RMSE_z = 0.0029$ m. The RMSE of all coordinates are degraded when compared to test 01, which is probably related to the configuration of cameras on test 02. The same degradation of results happened in the systematic study of test 02.

In case of resolution 1600×1200 , the maximum error in x, y decreases to approximately $e_x \sim e_y \sim 0.002$ m and z coordinates also decreases to approximately $e_z \sim 0.001$ m. The root mean square error (RMSE) between the simulated trajectory points and the estimated points resulted in $RMSE_x = 0.0006$ m, $RMSE_y = 0.0006$ m and $RMSE_z = 0.0004$ m.

Like in the systematic cases presented in previous sections, the absolute error distribution has been plotted as a histogram. This trajectory was always followed by the 8 cameras available in the simulation, so Figure 3.66 shows the histogram for resolution 320×240 and Figure 3.67 for resolution 1600×1200 . A Weibull PDF was again fitted to these data distributions, yielding the parameters presented in Tables 3.45 and 3.46 for the respective resolutions. In the case of resolution 320×240 , the Weibull PDF has a scale parameter $\lambda = 0.0070$ m, which, like in the systematic cases, approaches the median value of absolute error along the trajectory. In case of resolution 1600×1200 , the scale parameter drops to $\lambda = 0.0009$ m.

Distance absolute error e_{abs} per test position
Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras

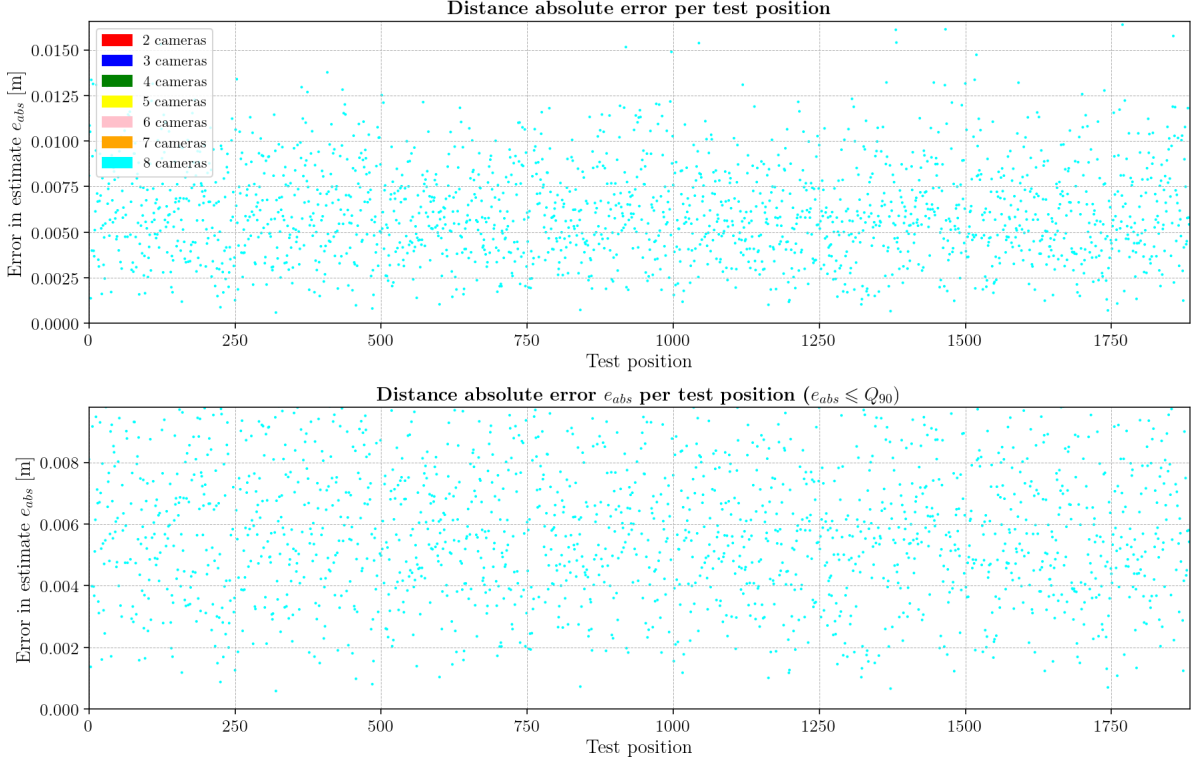


Figure 3.62: Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 02.

Table 3.43: Global statistics and root mean square error for test 02 with resolution 320×240.

	Abs. error	Est. error
count	1884	1884
mean	0.0062	0.0298
std	0.0027	0.0069
min	0.0006	0.0090
25%	0.0042	0.0248
50%	0.0058	0.0296
75%	0.0079	0.0346
90%	0.0098	0.0390
max	0.0164	0.0549
RMSE	0.0067	

Table 3.44: Global statistics and root mean square error for test 02 with resolution 1600×1200.

	Abs. error	Est. error
count	1884	1884
mean	0.0008	0.0041
std	0.0004	0.0008
min	0.0001	0.0015
25%	0.0006	0.0035
50%	0.0008	0.0041
75%	0.0011	0.0046
90%	0.0013	0.0052
max	0.0023	0.0069
RMSE	0.0009	

Table 3.45: Weibull PDF parameters and bin count per camera used in figure 3.66 for test 02 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.4682	0.0000	0.0070	1884	21

Table 3.46: Weibull PDF parameters and bin count per camera used in figure 3.67 for test 02 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.5209	0.0000	0.0009	1884	22

Distance absolute error e_{abs} per test position
Test: 02, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

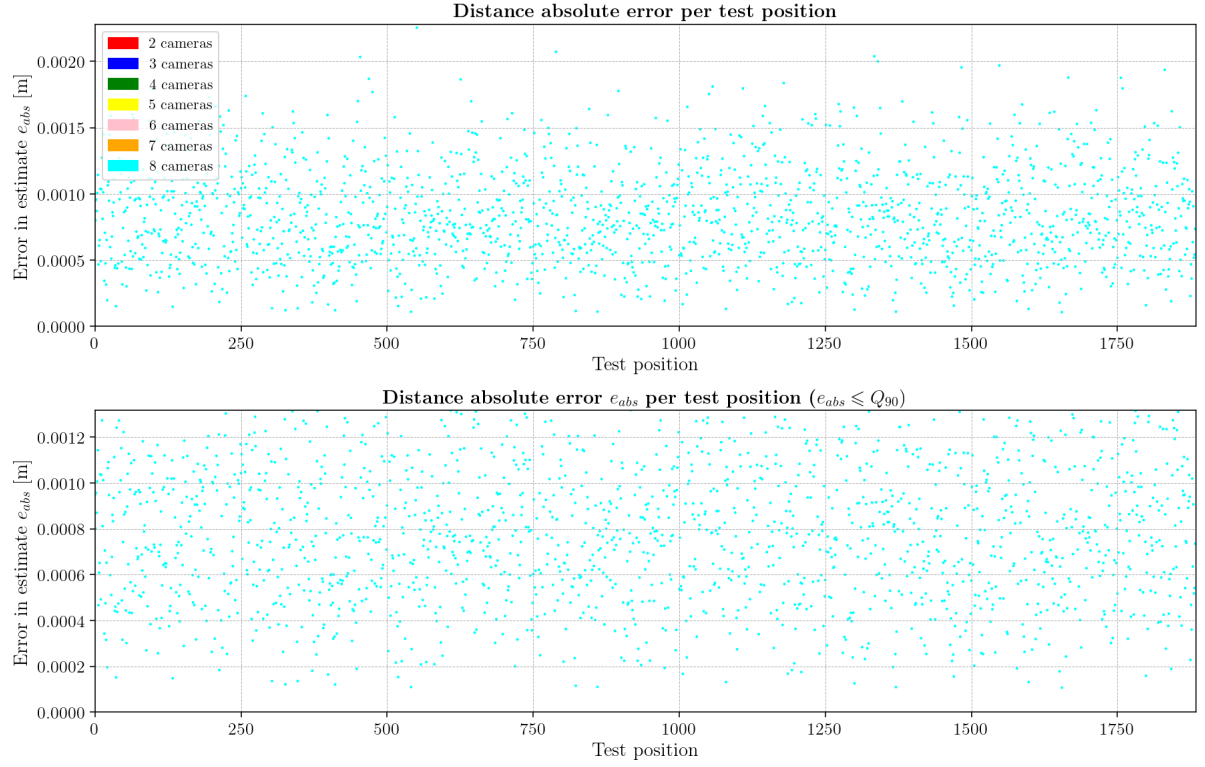


Figure 3.63: Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 02.

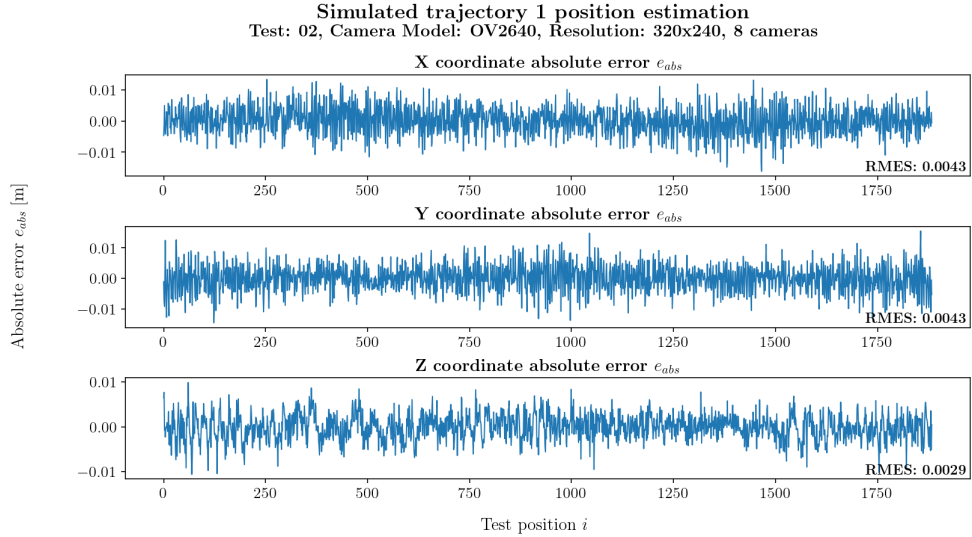


Figure 3.64: Real error in position estimation for trajectory 1 at resolution 320×240 and test 02, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

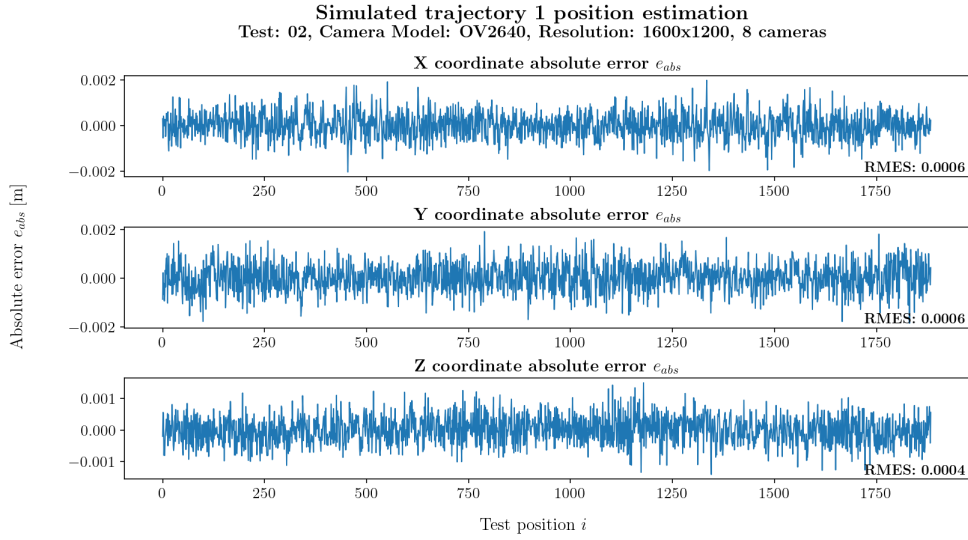


Figure 3.65: Real error in position estimation for trajectory 2 at resolution 1600×1200 and test 02, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

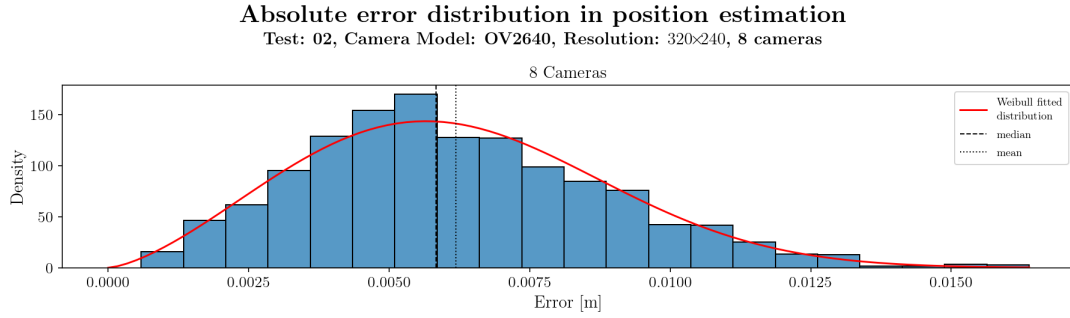


Figure 3.66: Error distribution histogram for trajectory 1 and test 02 at resolution 320×240.

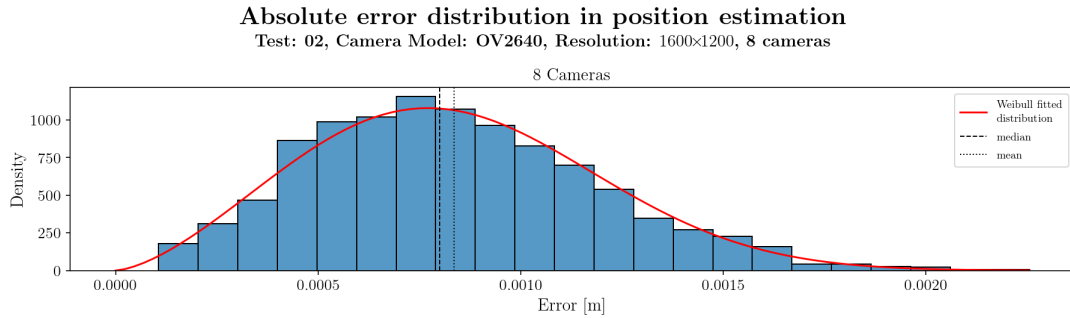


Figure 3.67: Error distribution histogram for trajectory 1 and test 02 at resolution 1600×1200.

3.6.3 Analysis of Test 03

Trajectory 1 captured by cameras in test configuration 2 is evaluated in this section. Like in the previous case, Figure 3.68 presents the simulated trajectory in a dashed black line and the estimated points in red (which, again, due to the scale used in the plots, becomes apparently a connected curve). As in previous cases, the left plot is the trajectory projected onto the XY plane and the right plot is its projection onto the XZ plane. By analysing Table 3.47 and Figure 3.69, it can be observed that the algorithm recovers the beacon position 90% of the time with an error smaller than $e_{abs} \leq 0.0079$ m when using a low resolution of 320×240 . Also, in that same table, it can be seen that the mean error is $\bar{e}_{abs} = 0.0049$ m, standard deviation $\sigma = 0.0022$ m and median $\tilde{e}_{abs} = 0.0046$ m.

When using high resolution of 1600×1200 , Table 3.48 and Figure 3.70 show that the error in position determination decreases to 90% of values with $e_{abs} \leq 0.0011$ m, with mean error $\bar{e}_{abs} = 0.0007$ m, standard deviation $\sigma = 0.0003$ m and median $\tilde{e}_{abs} = 0.0007$ m.

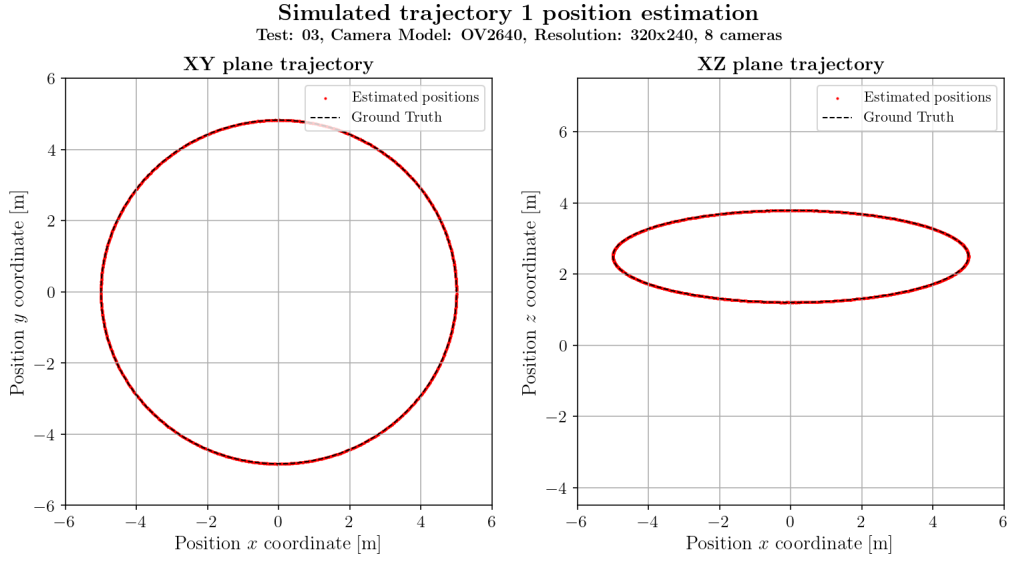


Figure 3.68: Trajectory 1 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 03

Figures 3.71 and 3.72 present, as before, the error of the system per coordinate component for resolutions 320×240 and 1600×1200 , respectively, for every point recovered along the trajectory.

With a resolution of 320×240 , the maximum error in x, y and z coordinates is of the order $e_x \sim e_y \sim e_z \sim 0.0010$ m, like in previous tests analysis of trajectory 1. The error per coordinate is better described by the root mean square error (RMSE) between the simulated trajectory points and the estimated points, where $\text{RMSE}_x = 0.0034$ m, $\text{RMSE}_y = 0.0033$ m and $\text{RMSE}_z = 0.0024$ m.

In case of resolution 1600×1200 , the maximum error in x decreases to approximately $e_x \sim 0.002$ m and y, z coordinates decreases to approximately $e_y \sim e_z \sim 0.001$ m. The root mean square error (RMSE) between the simulated trajectory points and the estimated points resulted in $\text{RMSE}_x = 0.0005$ m, $\text{RMSE}_y = 0.0005$ m and $\text{RMSE}_z = 0.0004$ m.

Like in the systematic cases presented in previous sections, the absolute error distribution has been plotted in the form of a histogram. This trajectory was always followed by the 8 cameras available in the simulation, so Figure 3.73 shows the histogram for resolution 320×240 and Figure 3.74 for resolution 1600×1200 . A Weibull PDF was again fitted to these data distributions, yielding the parameters presented in Tables 3.49 and 3.50 for the respective resolutions. In the case of resolution 320×240 , the Weibull PDF has a scale parameter $\lambda = 0.0055$ m, which, like in the systematic cases, approaches the median value of absolute error along the trajectory. In case of resolution 1600×1200 , the scale parameter goes down to $\lambda = 0.0008$ m.

Distance absolute error e_{abs} per test position
Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras

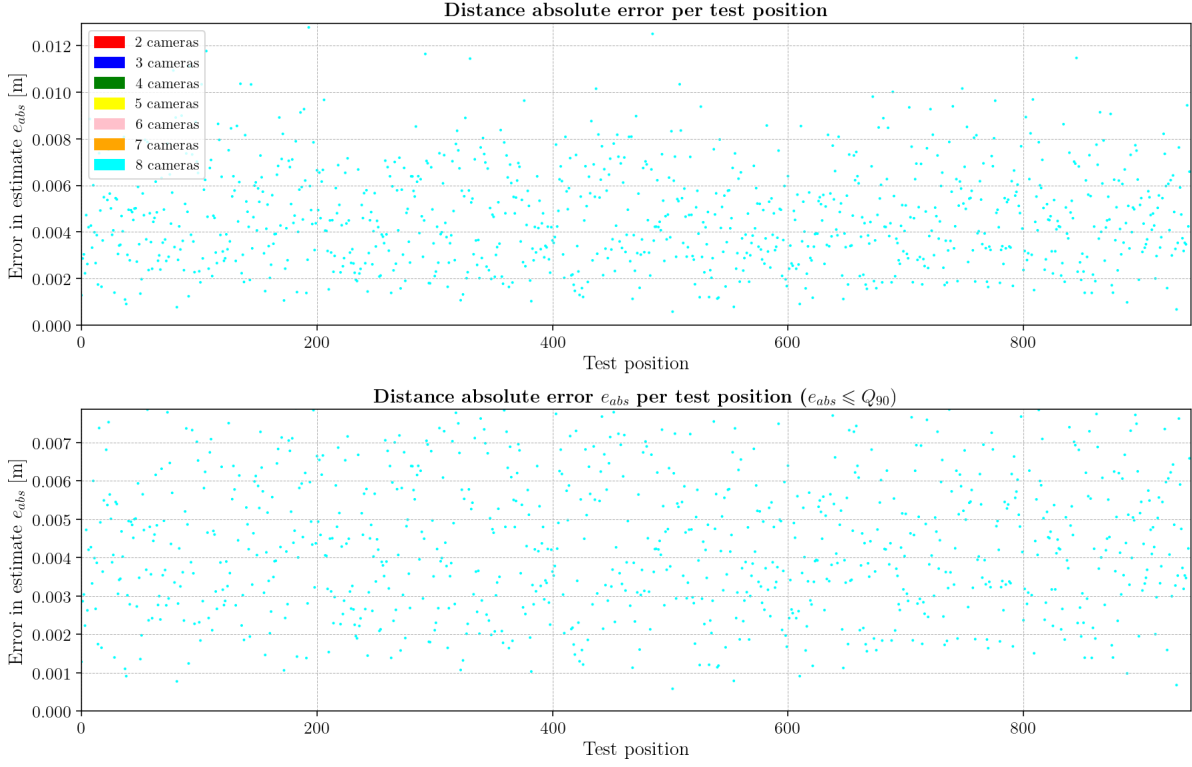


Figure 3.69: Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 03.

Table 3.47: Global statistics and root mean square error for test 03 with resolution 320×240.

	Abs. error	Est. error
count	1884	1884
mean	0.0049	0.0239
std	0.0022	0.0056
min	0.0006	0.0097
25%	0.0033	0.0200
50%	0.0046	0.0236
75%	0.0062	0.0276
90%	0.0079	0.0312
max	0.0128	0.0424
RMSE	0.0053	

Table 3.48: Global statistics and root mean square error for test 03 with resolution 1600×1200.

	Abs. error	Est. error
count	1884	1884
mean	0.0007	0.0036
std	0.0003	0.0008
min	0.0001	0.0015
25%	0.0005	0.0030
50%	0.0007	0.0036
75%	0.0009	0.0041
90%	0.0011	0.0046
max	0.0020	0.0069
RMSE	0.0008	

Table 3.49: Weibull PDF parameters and bin count per camera used in figure 3.73 for test 03 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.3640	0.0000	0.0055	1884	20

Table 3.50: Weibull PDF parameters and bin count per camera used in figure 3.74 for test 03 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.4459	0.0000	0.0008	1884	23

Distance absolute error e_{abs} per test position
Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

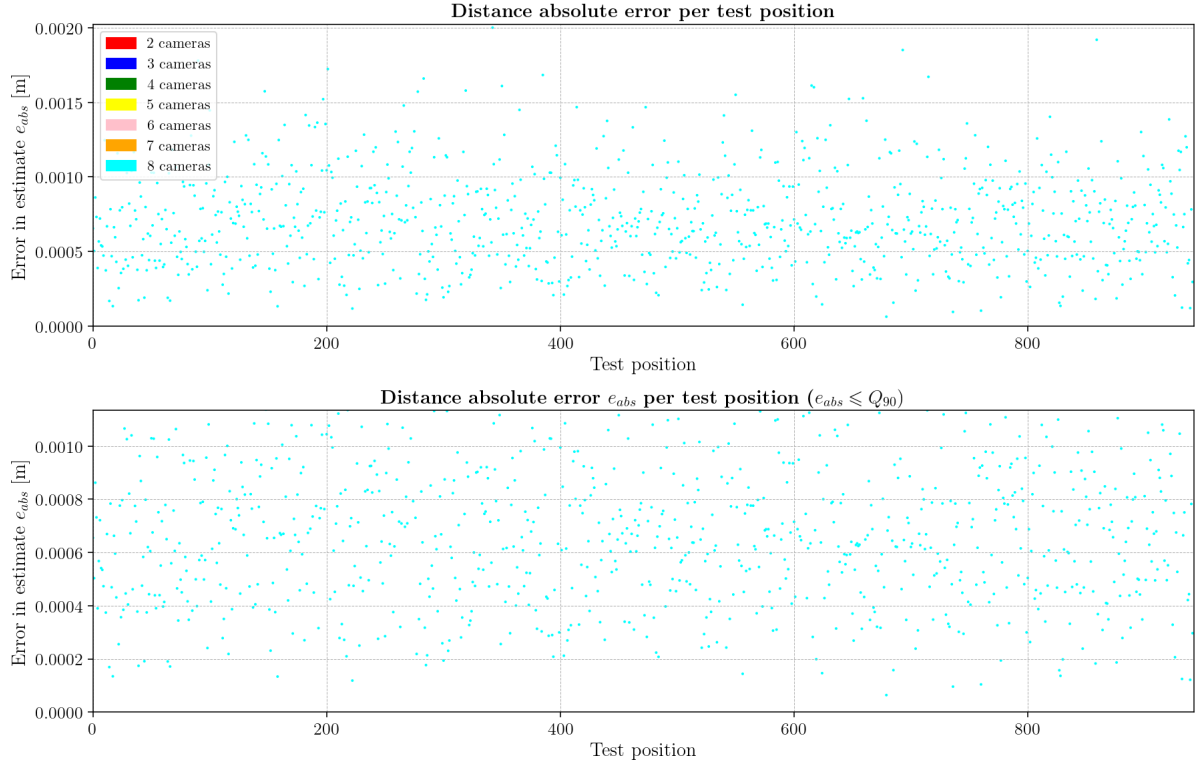


Figure 3.70: Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 03.

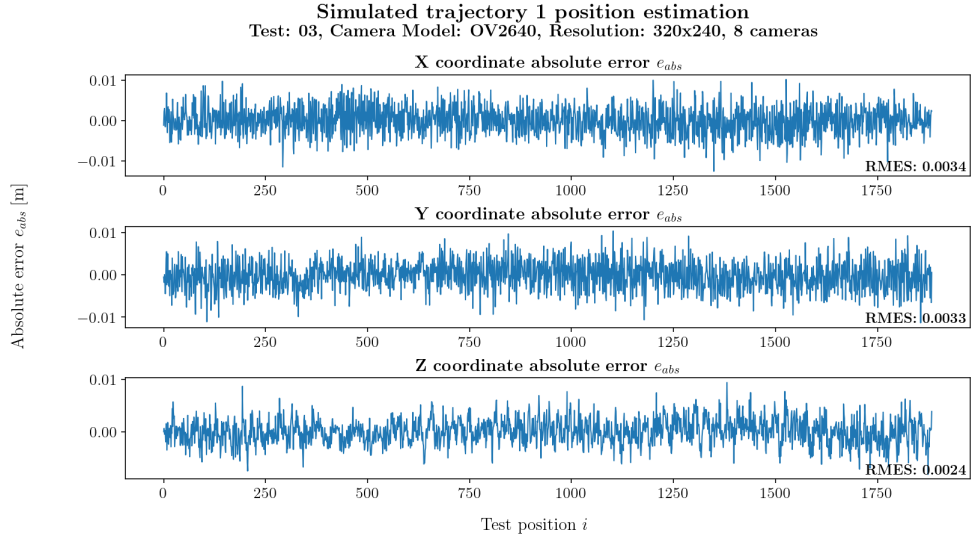


Figure 3.71: Real error in position estimation for trajectory 1 at resolution 320×240 and test 03, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

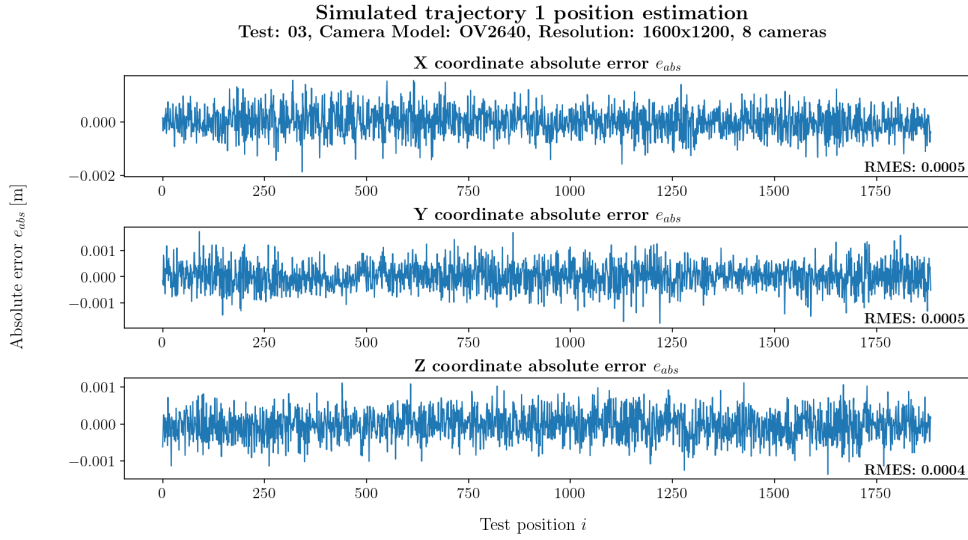


Figure 3.72: Real error in position estimation for trajectory 1 at resolution 1600×1200 and test 03, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

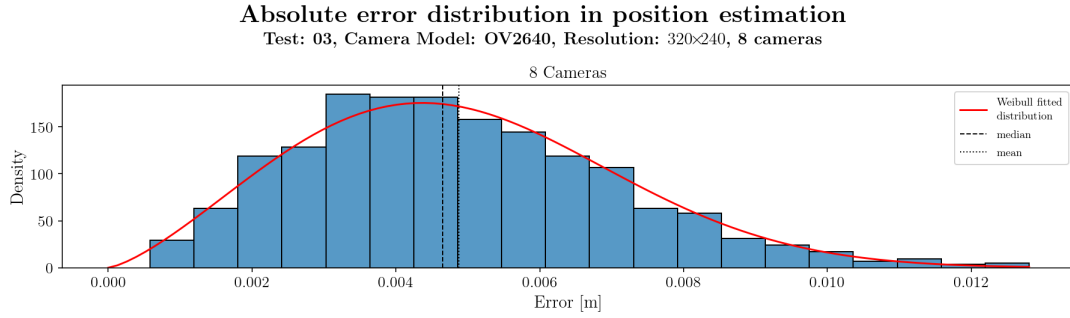


Figure 3.73: Error distribution histogram for trajectory 1 and test 03 at resolution 320×240.

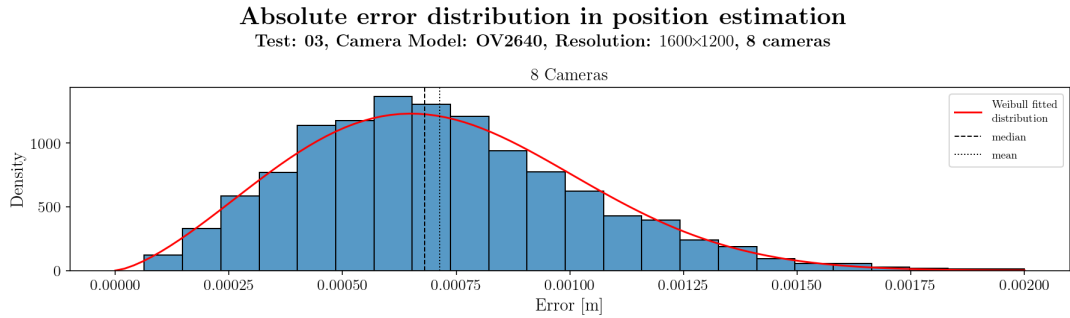


Figure 3.74: Error distribution histogram for trajectory 1 and test 03 at resolution 1600×1200.

3.6.4 Analysis of Test 04

The evaluation of trajectory 1 under camera test configuration 4 is presented on this subsection. Like in previous cases, Figure 3.75 presents the simulated trajectory in a dashed black line and the estimated points in red. As in other cases, the left plot is the trajectory projected onto the XY plane and the right plot is its projection onto the XZ plane. It can be observed by analysing Table 3.51 and Figure 3.76 that the algorithm recovers the beacon position 90% of the time with an error smaller than $e_{abs} \leq 0.0107$ m when using a low resolution of 320×240 . Also, in that same table, it can be seen that the mean error is $\bar{e}_{abs} = 0.0066$ m, standard deviation $\sigma = 0.0029$ m and median $\tilde{e}_{abs} = 0.0062$ m.

In the case of higher resolution of 1600×1200 , Table 3.52 and Figure 3.77 show that the error in position determination decreases to 90% of values with $e_{abs} \leq 0.0015$ m, with mean error $\bar{e}_{abs} = 0.0009$ m, standard deviation $\sigma = 0.0004$ m and median $\tilde{e}_{abs} = 0.0009$ m.

Figures 3.76 and 3.77 also show that only in this camera's configuration, there are points that aren't always detected by all the cameras in the system. Some points have been detected only by 6 or 7 cameras. This may be due to the higher location of the cameras in the system, when compared to the previous test 03, where the horizontal distribution of cameras is similar, but at a lower height compared to this test.

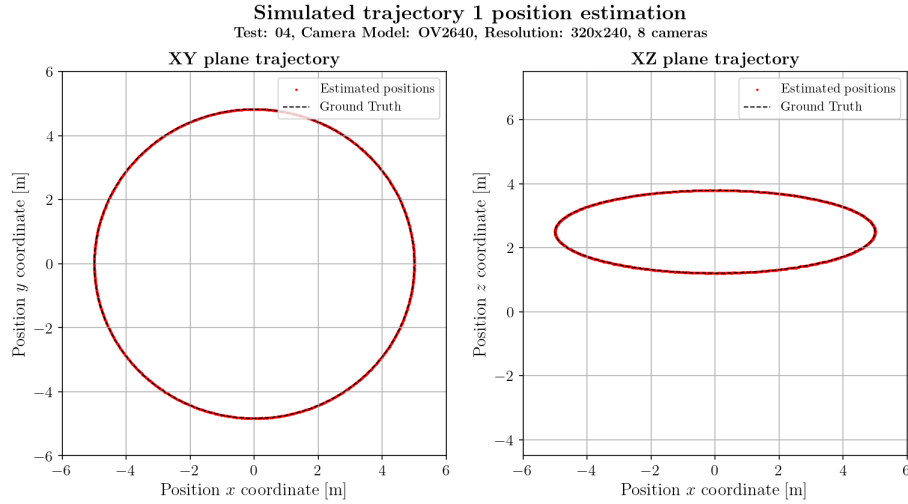


Figure 3.75: Trajectory 1 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 04

Figures 3.78 and 3.79 show the error per coordinate component for resolutions 320×240 and 1600×1200 respectively, for every point along the trajectory.

In the case of lower resolution, the maximum error in x, y and z coordinates is of the order $e_x \sim e_y \sim e_z \sim 0.0010$ m and the root mean square errors (RMSE) between the simulated trajectory points and the estimated points are $RMSE_x = 0.0041$ m, $RMSE_y = 0.0042$ m and $RMSE_z = 0.0043$ m.

In case of resolution 1600×1200 , the maximum error in x, y and z decreases to approximately $e_x \sim e_y \sim e_z \sim 0.002$ m. The root mean square errors (RMSE) between the simulated trajectory points and the estimated points are $RMSE_x = 0.0006$ m, $RMSE_y = 0.0006$ m and $RMSE_z = 0.0006$ m.

The RMSE of all the coordinates increased when compared with the previous test configuration. This is likely caused by the higher position of the camera, together with the fact that not all positions are being detected by all cameras available in the system.

Like in the previous cases, the absolute error distribution has been plotted in the form of a histogram. This trajectory was always followed by the 6, 7 or 8 cameras available in the simulation, so Figure 3.80 shows the histogram for resolution 320×240 and Figure 3.81 for resolution 1600×1200 . A Weibull PDF was fitted as before to these data distributions, yielding the parameters presented in Tables 3.53 and 3.54 for the respective resolutions. In the case of resolution 320×240 the Weibull PDF has a scale parameter $\lambda = 0.0098$ m, for 6 cameras, $\lambda = 0.0079$ m, for 7 cameras and $\lambda = 0.0064$ m, for 8 cameras. In case of

Distance absolute error e_{abs} per test position
Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

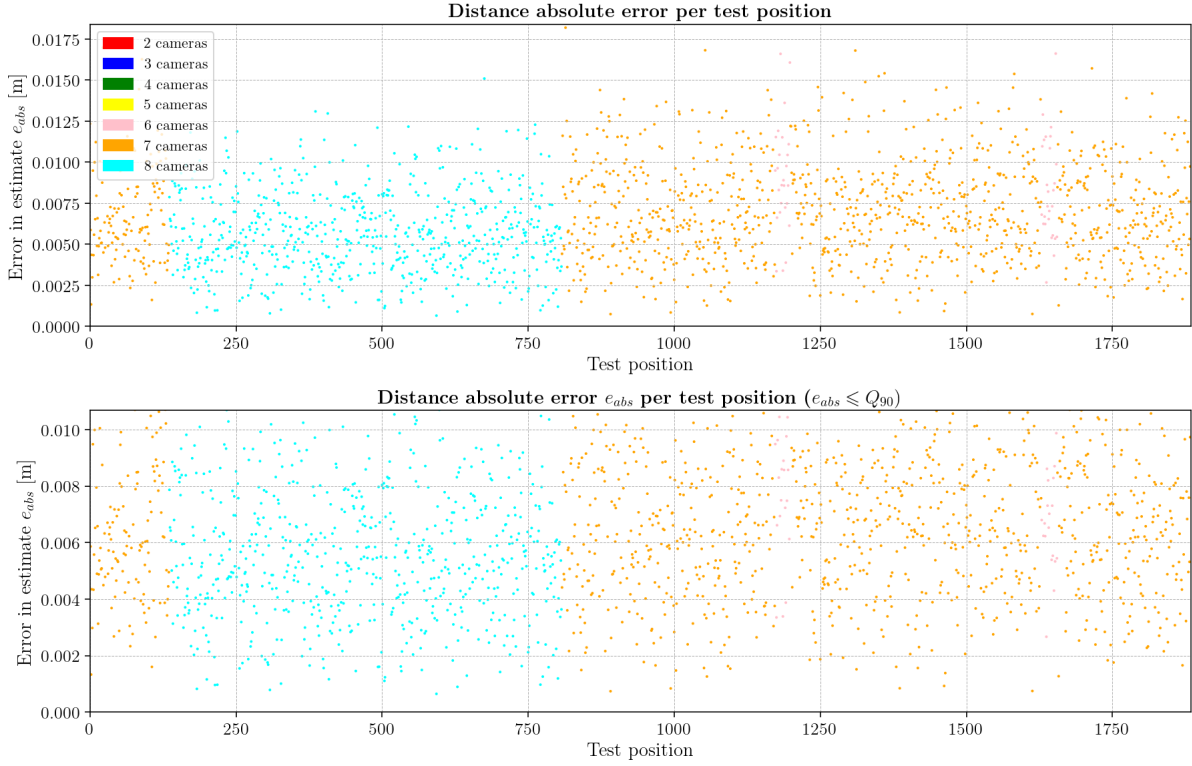


Figure 3.76: Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 04.

Table 3.51: Global statistics and root mean square error for test 04 with resolution 320×240.

	Abs. error	Est. error
count	1884	1884
mean	0.0066	0.0302
std	0.0029	0.0072
min	0.0006	0.0103
25%	0.0045	0.0251
50%	0.0062	0.0301
75%	0.0085	0.0352
90%	0.0107	0.0396
max	0.0182	0.0536
RMSE	0.0072	

Table 3.52: Global statistics and root mean square error for test 04 with resolution 1600×1200.

	Abs. error	Est. error
count	1884	1884
mean	0.0009	0.0044
std	0.0004	0.0009
min	0.0000	0.0017
25%	0.0006	0.0037
50%	0.0009	0.0043
75%	0.0012	0.0049
90%	0.0015	0.0056
max	0.0024	0.0080
RMSE	0.0010	

resolution 1600×1200, the scale parameter goes down to $\lambda = 0.0013$ m for 6 cameras, $\lambda = 0.0011$ m for 7 cameras and $\lambda = 0.0009$ m for 8 cameras.

Like in the systematic cases, the scale parameter λ approaches the median value of absolute error along the trajectory, for the corresponding number of cameras.

Distance absolute error e_{abs} per test position
Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

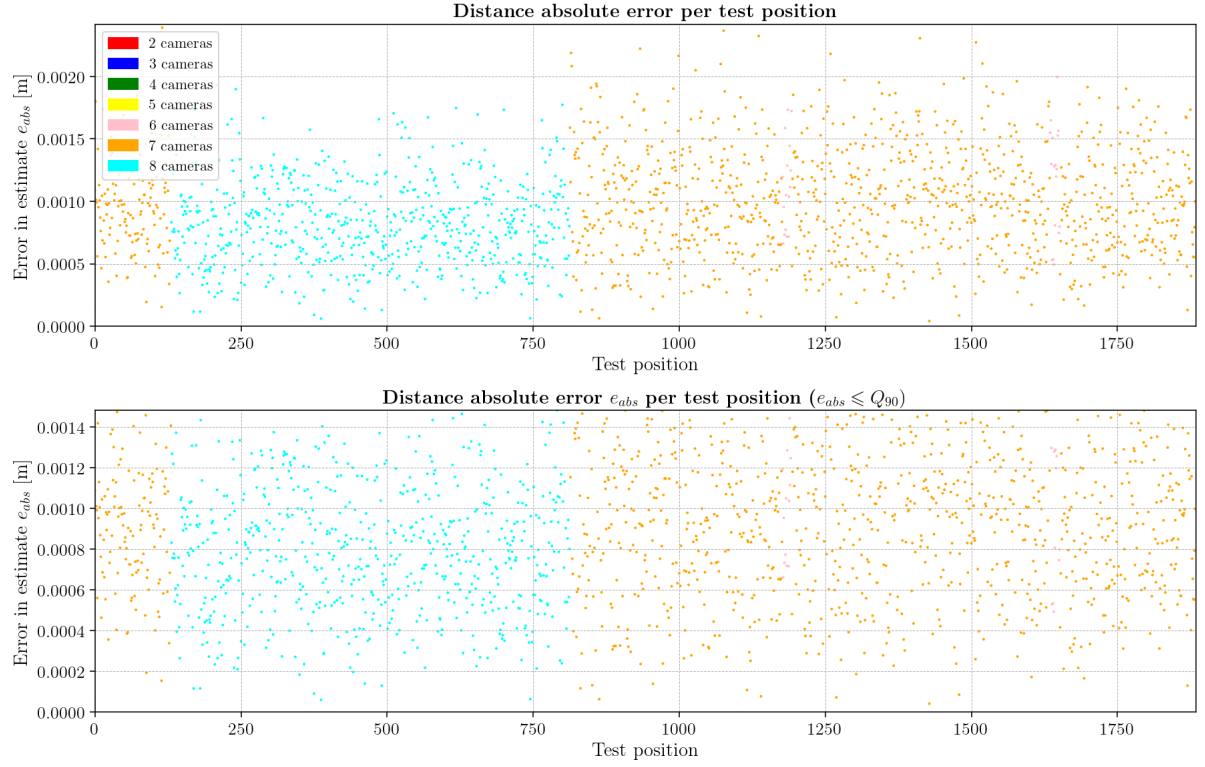


Figure 3.77: Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 04.

Table 3.53: Weibull PDF parameters and bin count per camera used in figure 3.80 for test 04 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
6	2.9345	0.0000	0.0098	53	5
7	2.5323	0.0000	0.0079	1159	18
8	2.4582	0.0000	0.0064	672	15

Table 3.54: Weibull PDF parameters and bin count per camera used in figure 3.81 for test 04 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
6	3.2474	0.0000	0.0013	33	4
7	2.5442	0.0000	0.0011	1169	17
8	2.6045	0.0000	0.0009	682	15

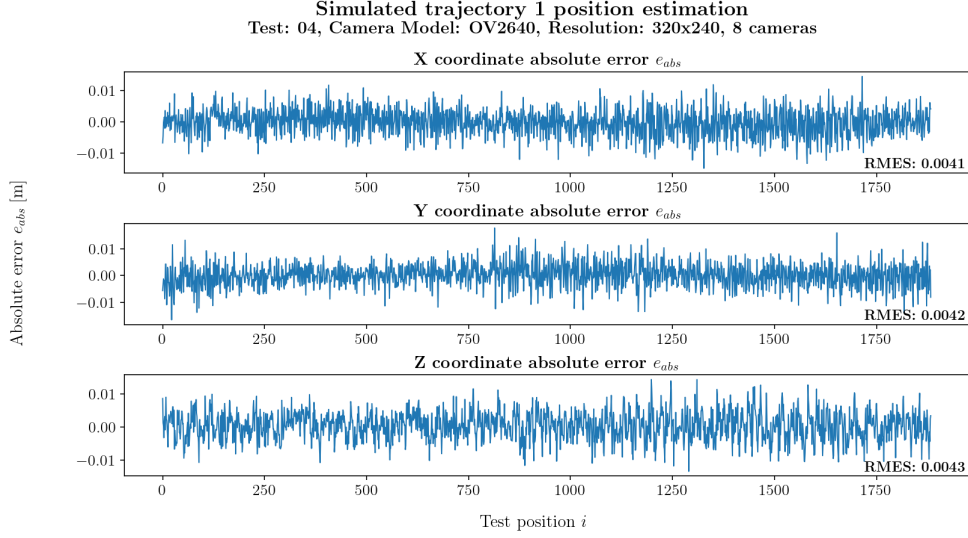


Figure 3.78: Real error in position estimation for trajectory 1 at resolution 320×240 and test 04, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

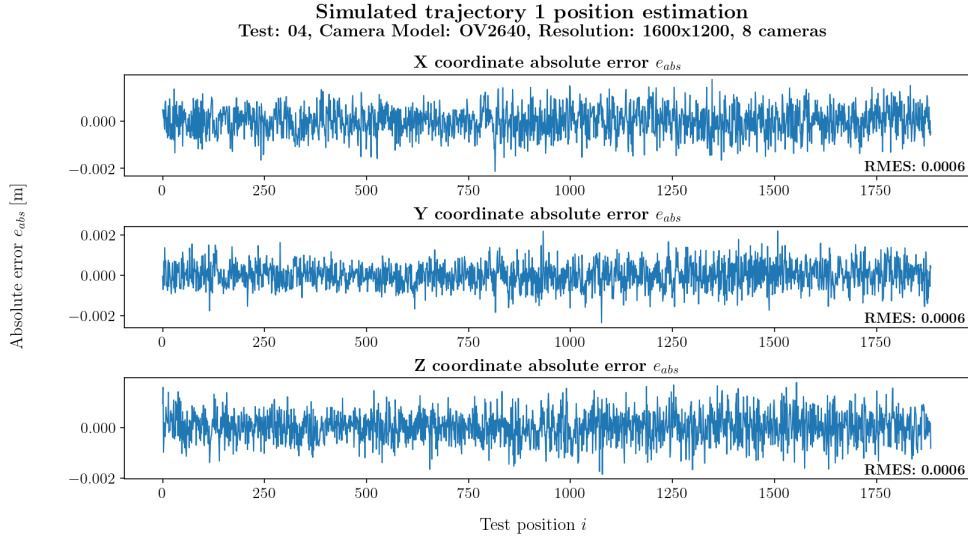


Figure 3.79: Real error in position estimation for trajectory 1 at resolution 1600×1200 and test 04, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

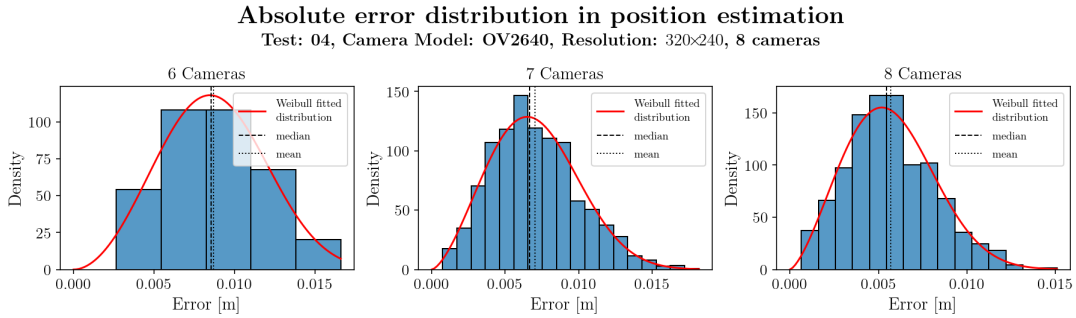


Figure 3.80: Error distribution histogram for trajectory 1 and test 04 at resolution 320×240.

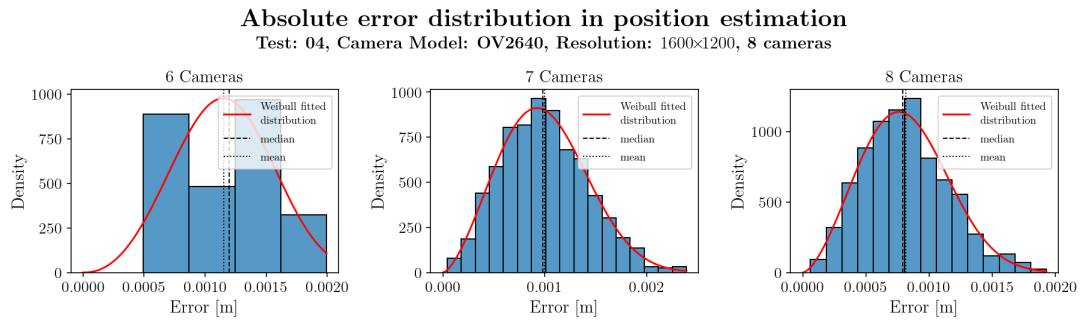


Figure 3.81: Error distribution histogram for trajectory 1 and test 04 at resolution 1600×1200.

3.7 Trajectory 2 simulation results

3.7.1 Analysis of Test 01

Trajectory 2, as described in section 3.4, is a figure 8 trajectory, and this section describes the results of it as captured by cameras in test configuration 1. Figure 3.82 presents the simulated trajectory in a dashed black line and the estimated points in red (which as trajectory 1 examples, due to the scale used in the plots, becomes apparently a connected curve). The left plot is the trajectory projected onto the XY plane, and the right plot is its projection onto the XZ plane. By analysing Table 3.55 and Figure 3.83, it can be observed that the algorithm recovers the beacon position 90% of the time with an error smaller than $e_{abs} \leq 0.0098$ m when using a low resolution of 320×240 . Also, in that same table, it can be seen that the mean error is $\bar{e}_{abs} = 0.0059$ m, standard deviation $\sigma = 0.0031$ m and median $\tilde{e}_{abs} = 0.0053$ m.

When using high resolution of 1600×1200 , Table 3.56 and Figure 3.84 show that the error in position determination decreases to 90% of values with $e_{abs} \leq 0.0017$ m, with mean error $\bar{e}_{abs} = 0.0009$ m, standard deviation $\sigma = 0.0008$ m and median $\tilde{e}_{abs} = 0.0007$ m.

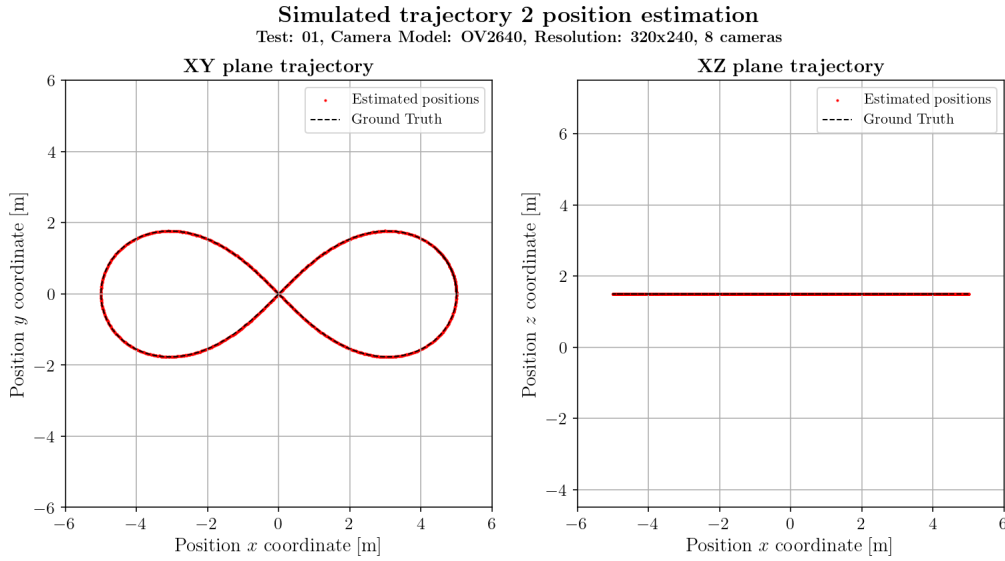


Figure 3.82: Trajectory 2 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 01

Figures 3.85 and 3.86 present the error of the system per coordinate component for resolutions 320×240 and 1600×1200 , respectively, for every point recovered along the trajectory. It can be seen that there are some irregularities in both graphs, especially in the components related to x and y coordinates. This happens when the beacon approaches the intersection point along the figure 8 trajectory, where, in the vicinity of the intersection, the direction vectors from each camera become almost parallel. Small errors in blob detection may cause those deviations near the intersection. If it weren't for these small errors, the algorithm would provide a more exact and regular solution.

With a resolution of 320×240 , the maximum error in x, y coordinates is of the order $e_x \sim e_y \sim 0.015$ m and z coordinate is of the order $e_z \sim 0.005$ m, almost an order of magnitude lower. The root mean square error (RMSE) per coordinate between the simulated trajectory points and the estimated points, were $RMSE_x = 0.0045$ m, $RMSE_y = 0.0048$ m and $RMSE_z = 0.0011$ m, showing that in the z coordinates the measurements were more stable. This may be since the trajectory was contained in the XY plane.

In case of resolution 1600×1200 , the maximum error in x and y coordinates decreased to approximately $e_x \sim e_y \sim 0.005$ m and z coordinate decreased to approximately $e_z \sim 0.0005$ m, again one order of magnitude lower. The root mean square error (RMSE) between the simulated trajectory points and the estimated points resulted in $RMSE_x = 0.0008$ m, $RMSE_y = 0.0008$ m and $RMSE_z = 0.0002$ m.

The absolute error distribution has been plotted in the form of histograms, as in other tests. This trajectory was able to be followed by all the 8 cameras available in the simulation, so Figure 3.87 shows

Distance absolute error e_{abs} per test position
Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

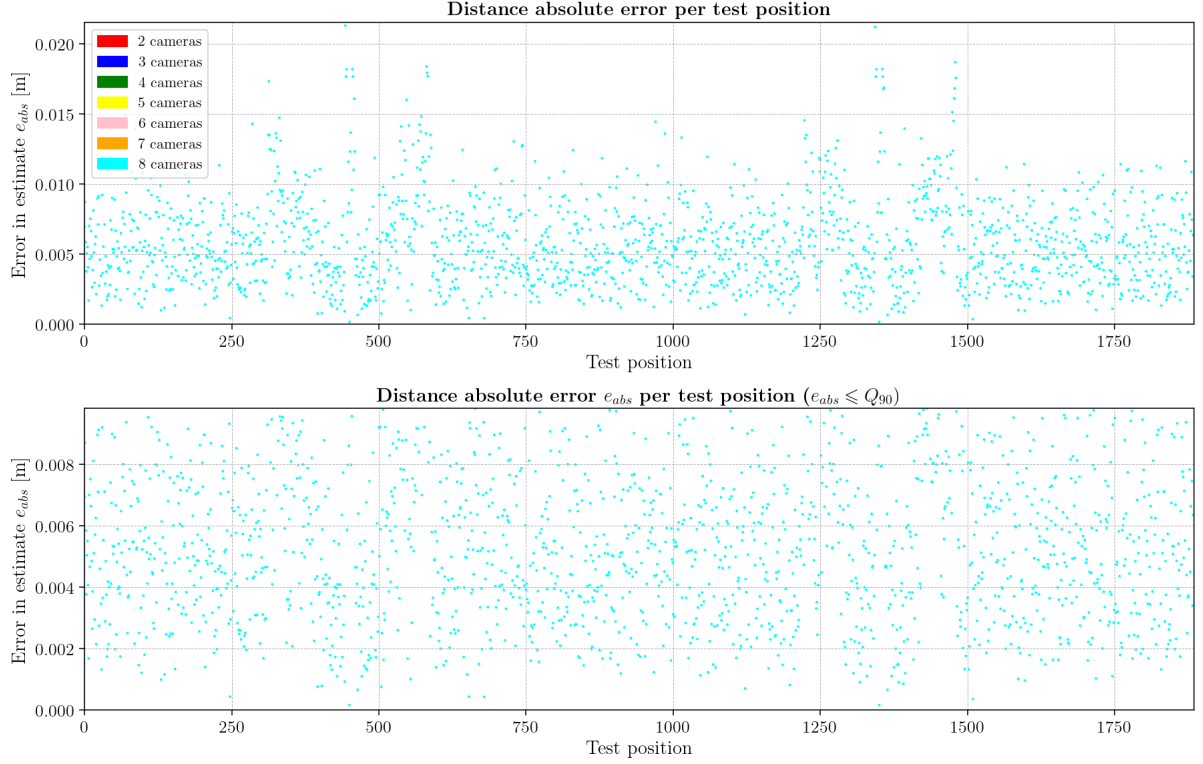


Figure 3.83: Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 01.

Table 3.55: Global statistics and root mean square error for test 01 with resolution 320×240.

	Abs. error	Est. error
count	1884	1884
mean	0.0059	0.0211
std	0.0031	0.0072
min	0.0002	0.0005
25%	0.0036	0.0162
50%	0.0053	0.0202
75%	0.0076	0.0248
90%	0.0098	0.0303
max	0.0213	0.0570
RMSE	0.0066	

Table 3.56: Global statistics and root mean square error for test 01 with resolution 1600×1200.

	Abs. error	Est. error
count	1884	1884
mean	0.0009	0.0033
std	0.0008	0.0016
min	0.0001	0.0006
25%	0.0005	0.0024
50%	0.0007	0.0030
75%	0.0010	0.0036
90%	0.0017	0.0045
max	0.0064	0.0145
RMSE	0.0012	

the histogram for resolution 320×240 and Figure 3.88 for resolution 1600×1200. A Weibull PDF was again fitted to these data distributions, yielding the parameters presented in Tables 3.57 and 3.58 for the respective resolutions. In the case of resolution 320×240, the Weibull PDF has a scale parameter $\lambda = 0.0066\text{m}$, which, like in the systematic cases, approaches the median value of absolute error along the trajectory. In case of resolution 1600×1200, the scale parameter goes down to $\lambda = 0.0010\text{m}$.

Table 3.57: Weibull PDF parameters and bin count per camera used in figure 3.87 for test 01 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	1.9782	0.0000	0.0066	1884	24

Distance absolute error e_{abs} per test position
Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

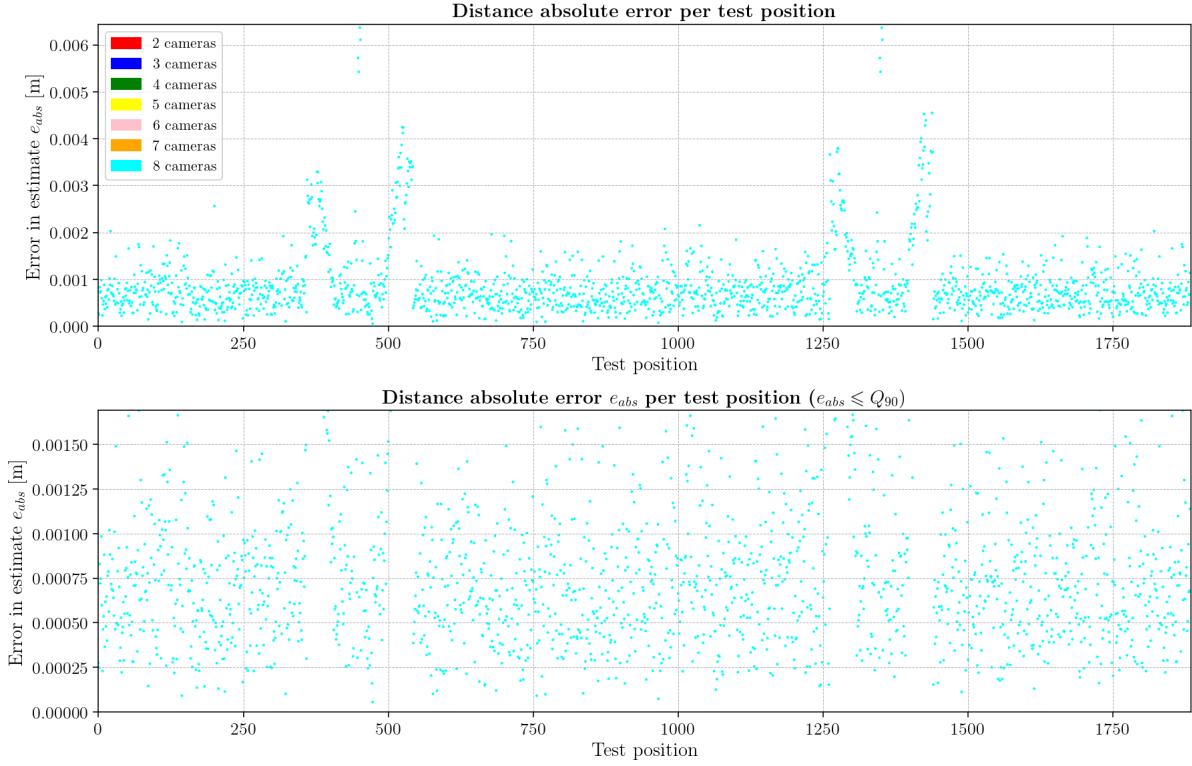


Figure 3.84: Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 01.

Table 3.58: Weibull PDF parameters and bin count per camera used in figure 3.88 for test 01 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	1.4307	0.0000	0.0010	1884	30

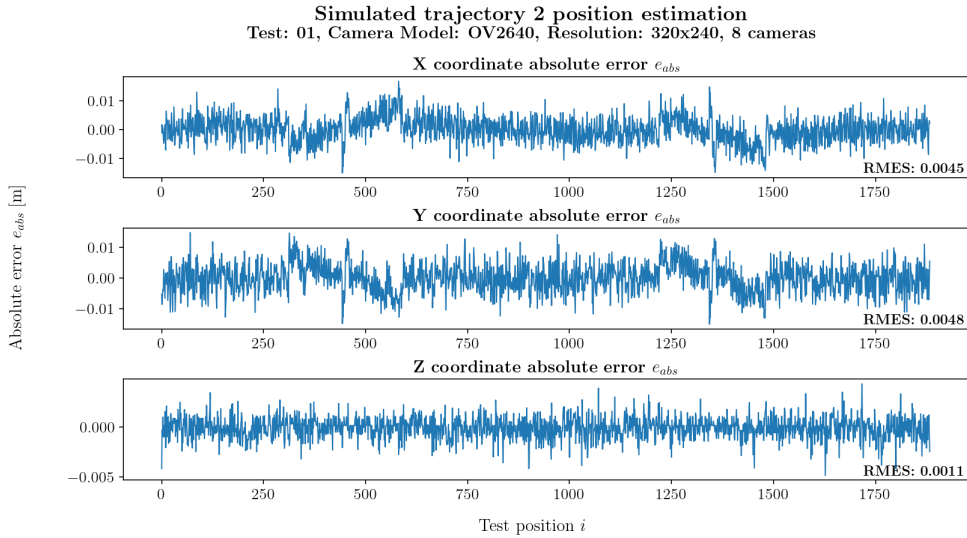


Figure 3.85: Real error in position estimation for trajectory 2 at resolution 320×240 and test 01, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

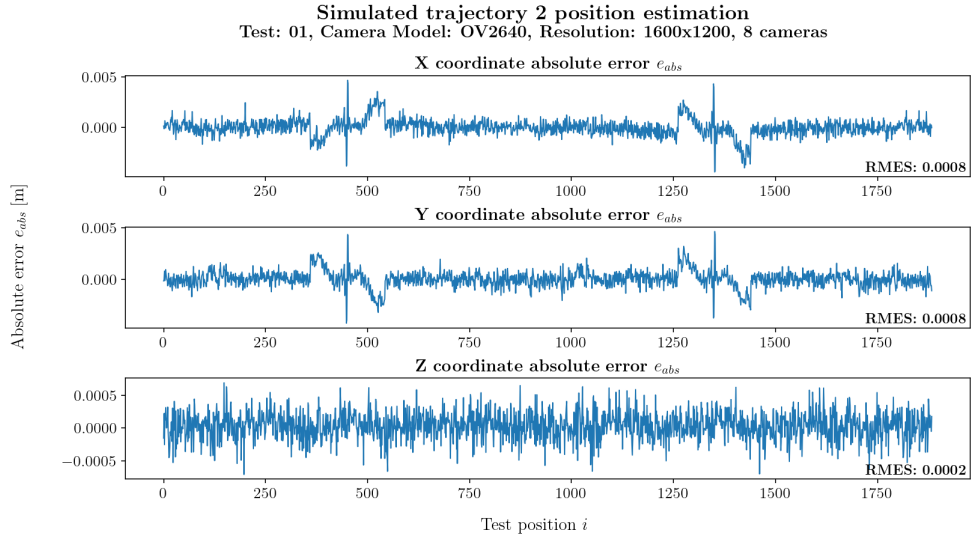


Figure 3.86: Real error in position estimation for trajectory 2 at resolution 1600x1200 and test 01, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

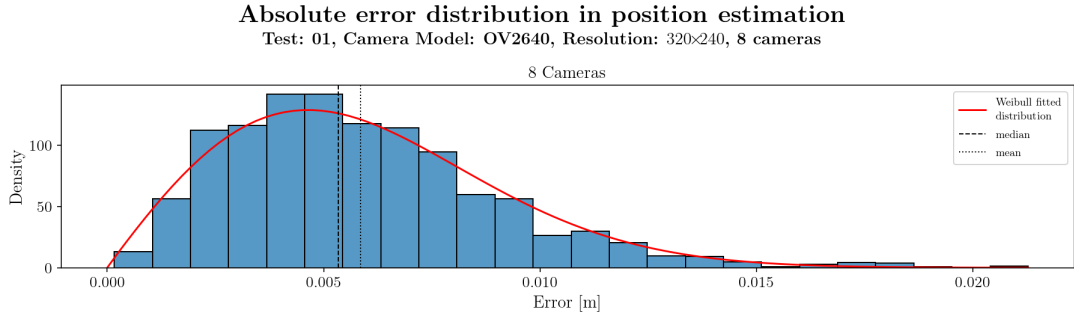


Figure 3.87: Error distribution histogram for trajectory 2 and test 01 at resolution 320x240.

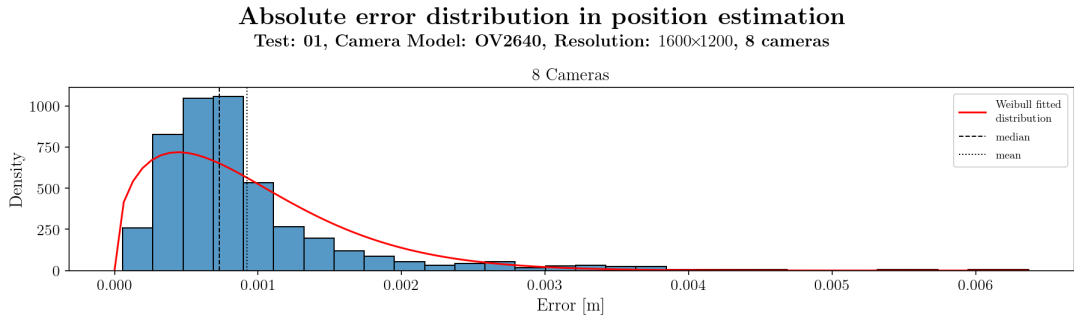


Figure 3.88: Error distribution histogram for trajectory 2 and test 01 at resolution 1600x1200.

3.7.2 Analysis of Test 02

This section describes the results of trajectory 2 as captured by cameras in test configuration 2. Figure 3.89 presents the simulated trajectory in a dashed black line and the estimated points in red. The left plot is the trajectory projected onto the XY plane, and the right plot is its projection onto the XZ plane. In Table 3.59 and Figure 3.90, it can be noticed again the ability of the algorithm to recover the beacon position 90% of the times with an error smaller than $e_{abs} \leq 0.0104$ m when using a low resolution of 320×240 . Also, in that same table, it can be seen that the mean error is $\bar{e}_{abs} = 0.0061$ m, standard deviation $\sigma = 0.0030$ m and median $\tilde{e}_{abs} = 0.0057$ m.

With a resolution of 1600×1200 , the analysis of Table 3.60 and Figure 3.91 demonstrates that the error in position estimation decreases to 90% of values with $e_{abs} \leq 0.0014$ m, with mean error $\bar{e}_{abs} = 0.0009$ m, standard deviation $\sigma = 0.0004$ m and median $\tilde{e}_{abs} = 0.0008$ m.

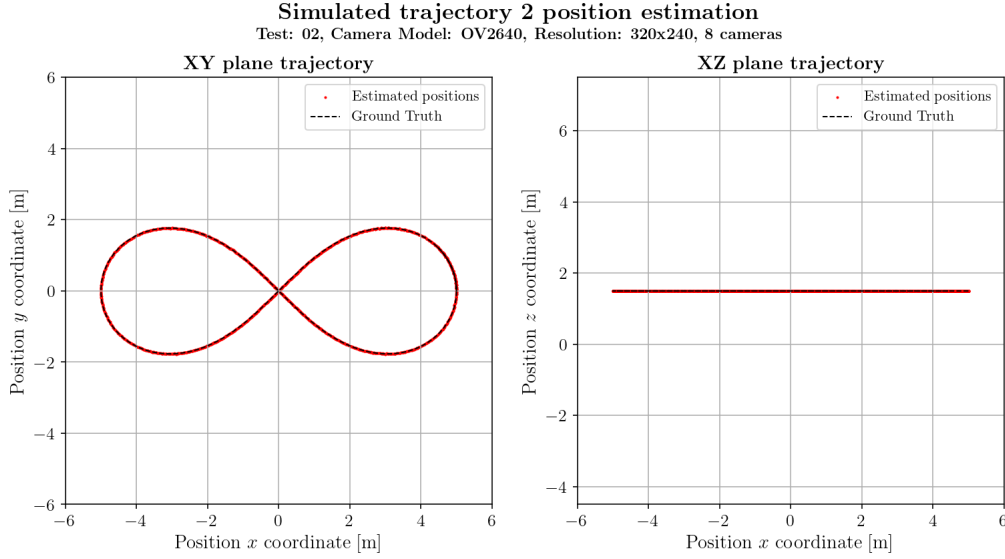


Figure 3.89: Trajectory 2 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 02

In Figures 3.92 and 3.93, the error of the system per coordinate component for resolutions 320×240 and 1600×1200 , respectively, for every point recovered along the trajectory. Like in test 01, on this test configuration of cameras, there are also some irregularities in both graphs, especially in the components related to x and y coordinates. This happens again when the beacon approaches the intersection point along the figure 8 trajectory.

With a resolution of 320×240 , the maximum error in x coordinate is of the order $e_x \sim 0.015$ m, in the y coordinate is of the order $e_y \sim 0.020$ m and z coordinate is of the order $e_z \sim 0.005$ m, again almost an order of magnitude lower. The root mean square error (RMSE) per coordinate between the simulated trajectory points and the estimated points, were $RMSE_x = 0.0043$ m, $RMSE_y = 0.0005$ m and $RMSE_z = 0.0019$ m, showing that in the z coordinates the measurements were more stable. This may be because the trajectory was contained in the XY plane. Also, the difference between the $RMSE_x$ and $RMSE_y$ may be explainable by the configuration of cameras used on this test.

As for resolution 1600×1200 , the maximum error in x and y coordinates decreased to approximately $e_x \sim e_y \sim 0.0025$ m and z coordinate decreased to approximately $e_z \sim 0.001$ m. The root mean square error (RMSE) between the simulated trajectory points and the estimated points resulted in $RMSE_x = 0.0006$ m, $RMSE_y = 0.0006$ m and $RMSE_z = 0.0003$ m.

The histograms below present the absolute error distribution for this test, for both resolutions displayed. All the 8 cameras available in the simulation were again able to follow this trajectory, so Figure 3.94 shows the histogram for resolution 320×240 and Figure 3.95 for resolution 1600×1200 . A Weibull PDF was, as usual, fitted to these data distributions, yielding the parameters presented in Tables 3.61 and 3.62 for the respective resolutions. In the case of resolution 320×240 , the Weibull PDF has a scale parameter

Distance absolute error e_{abs} per test position
Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras

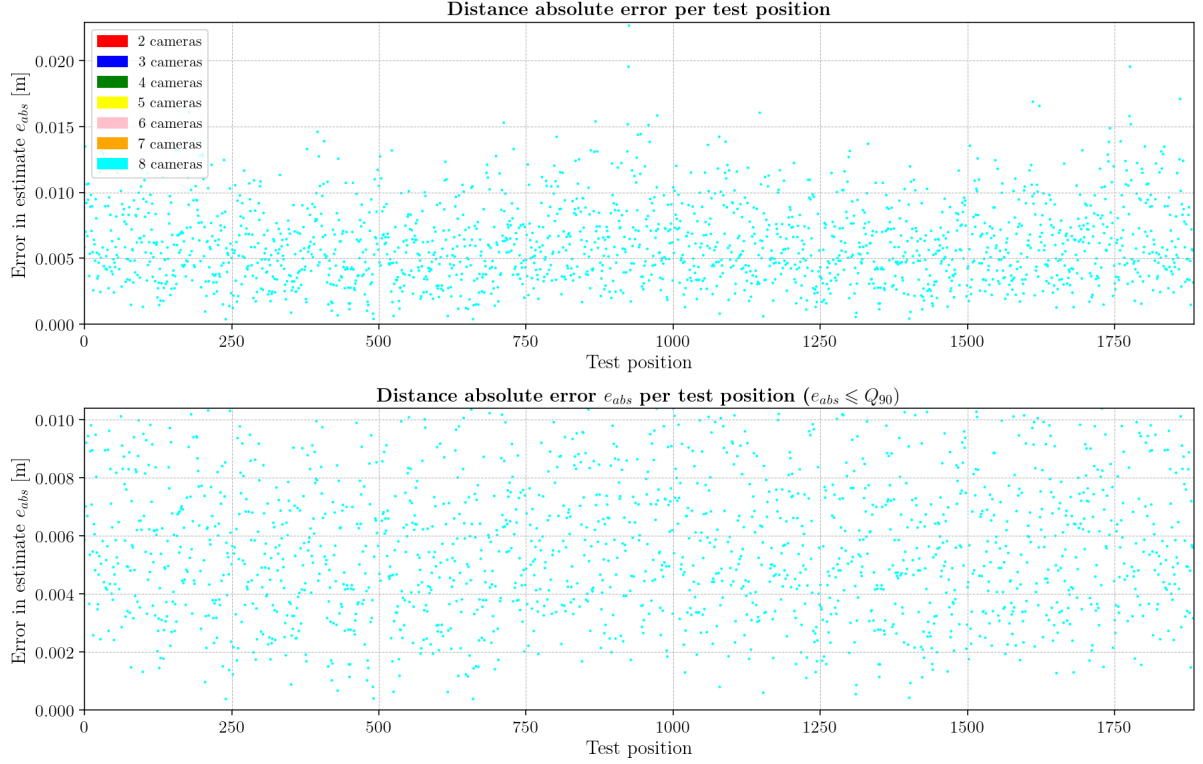


Figure 3.90: Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 02.

Table 3.59: Global statistics and root mean square error for test 02 with resolution 320×240.

	Abs. error	Est. error
count	1884	1884
mean	0.0061	0.0259
std	0.0030	0.0063
min	0.0004	0.0104
25%	0.0039	0.0215
50%	0.0057	0.0252
75%	0.0079	0.0295
90%	0.0104	0.0342
max	0.0226	0.0589
RMSE	0.0069	

Table 3.60: Global statistics and root mean square error for test 02 with resolution 1600×1200.

	Abs. error	Est. error
count	1884	1884
mean	0.0009	0.0038
std	0.0004	0.0010
min	0.0000	0.0013
25%	0.0006	0.0032
50%	0.0008	0.0038
75%	0.0011	0.0044
90%	0.0014	0.0049
max	0.0041	0.0119
RMSE	0.0010	

$\lambda = 0.0070$ m, which, like in the systematic cases, approaches the median value of absolute error along the trajectory. In case of resolution 1600×1200, the scale parameter goes down to $\lambda = 0.0010$ m. In both cases, the scale parameters for both resolutions approach the medians for the corresponding resolutions, as presented in Table 3.59 and Table 3.60.

Table 3.61: Weibull PDF parameters and bin count per camera used in figure 3.94 for test 02 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.1480	0.0000	0.0070	1884	26

Distance absolute error e_{abs} per test position
Test: 02, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

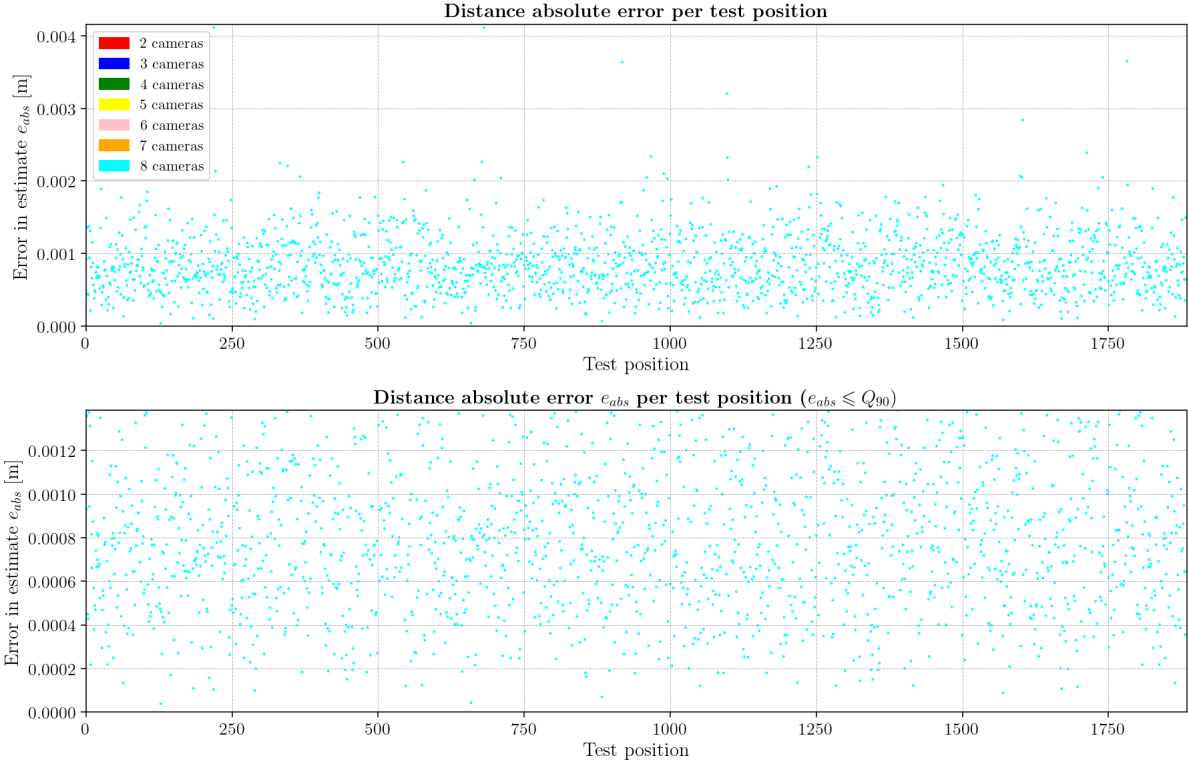


Figure 3.91: Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 02.

Table 3.62: Weibull PDF parameters and bin count per camera used in figure 3.95 for test 02 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.1315	0.0000	0.0010	1884	35

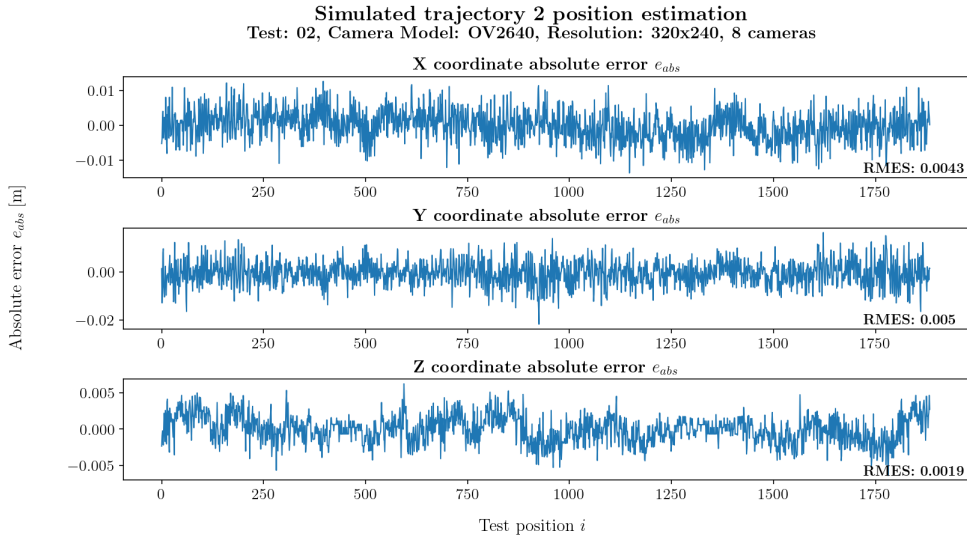


Figure 3.92: Real error in position estimation for trajectory 2 at resolution 320×240 and test 02, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

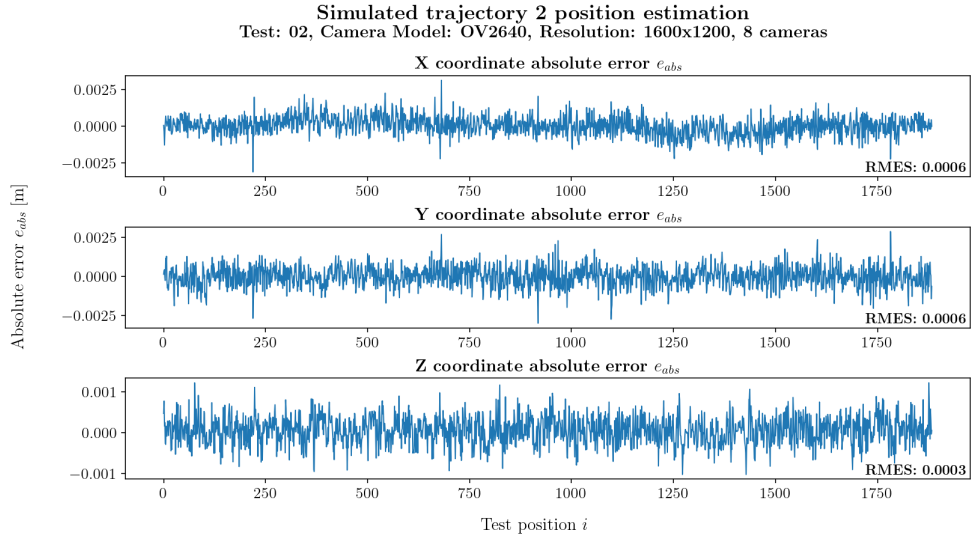


Figure 3.93: Real error in position estimation for trajectory 2 at resolution 1600x1200 and test 02, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

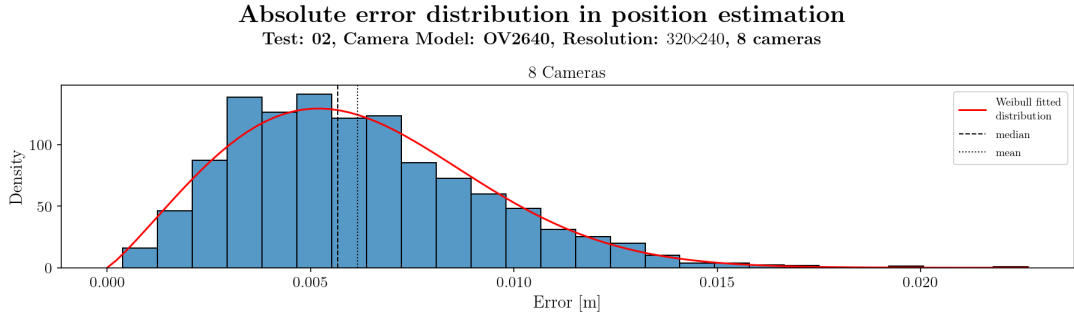


Figure 3.94: Error distribution histogram for trajectory 2 and test 02 at resolution 320x240.

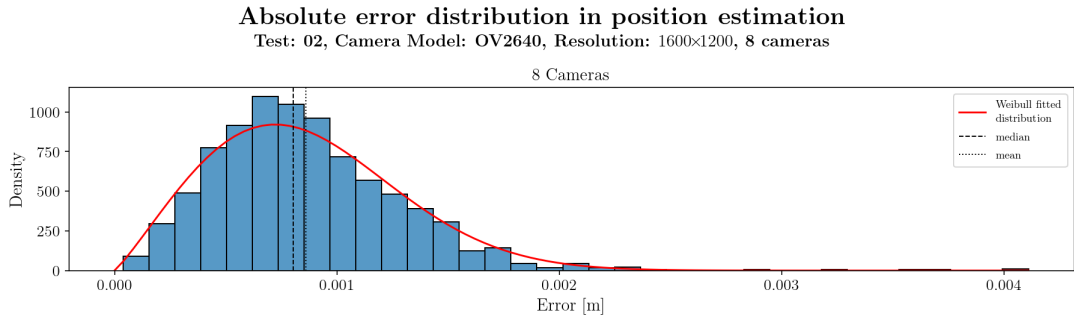


Figure 3.95: Error distribution histogram for trajectory 2 and test 02 at resolution 1600x1200.

3.7.3 Analysis of Test 03

Test camera configuration number 3 is evaluated in this section, analysing the data of estimation of trajectory 2, as in previous tests. The simulated trajectory in a dashed black line with the estimated points in red is presented in Figure 3.96. The left plot is the trajectory projected onto the XY plane, and the right plot is its projection onto the XZ plane. In Table 3.63 and Figure 3.97, it can be noticed again the ability of the algorithm to recover the beacon position 90% of the times with an error smaller than $e_{abs} \leq 0.0079$ m when using a low resolution of 320×240 . Also, in that same table, it can be seen that the mean error is $\bar{e}_{abs} = 0.0049$ m, standard deviation $\sigma = 0.0021$ m and median $\tilde{e}_{abs} = 0.0047$ m.

Looking now at the high resolution of 1600×1200 , the analysis of Table 3.64 and Figure 3.98 demonstrates that the error in position estimation reduces to 90% of values with $e_{abs} \leq 0.0012$ m, with mean error $\bar{e}_{abs} = 0.0008$ m, standard deviation $\sigma = 0.0003$ m and median $\tilde{e}_{abs} = 0.0007$ m.

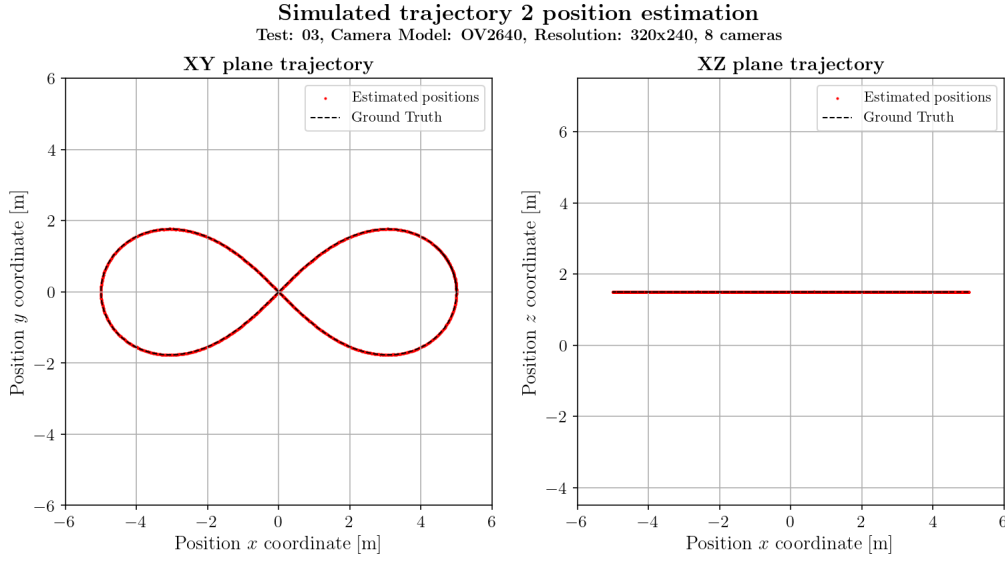


Figure 3.96: Trajectory 2 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 03

Figures 3.99 and 3.100 display again the error of the system per coordinate component for resolutions 320×240 and 1600×1200 , respectively, for every point recovered along the trajectory. Like in tests 1 and 2, in this third test, there are also some irregularities in both graphs, like in previous cases, more noticeable in the components related to x and y coordinates. This happens once again when the beacon approaches the intersection point along the figure 8 trajectory.

At resolution of 320×240 , the maximum error in x and y coordinates is of the order $e_x \sim e_y \sim 0.015$ m and z coordinate is on a smaller order of $e_z \sim 0.005$ m. The root mean square error (RMSE) per coordinate between the simulated trajectory points and the estimated points, were $\text{RMSE}_x = 0.0033$ m, $\text{RMSE}_y = 0.0033$ m and $\text{RMSE}_z = 0.0026$ m, showing that in the z coordinates the measurements were more once again accurate. This may again be since the trajectory was contained in the XY plane.

At resolution 1600×1200 , the maximum error in x reduces to the order of $e_x \sim 0.002$ m and y coordinate reduces to approximately $e_y \sim 0.001$ m while z coordinate decreases to approximately $e_z \sim 0.001$ m. The root mean square error (RMSE) between the simulated trajectory points and the estimated points resulted in $\text{RMSE}_x = 0.0005$ m, $\text{RMSE}_y = 0.0005$ m and $\text{RMSE}_z = 0.0004$ m.

Also, histograms of the absolute error distribution for this test are presented for both the lowest and the highest resolutions simulated. All the 8 cameras available in the simulation were once more able to follow this trajectory, so Figure 3.101 shows the histogram for resolution 320×240 and Figure 3.102 for resolution 1600×1200 . A Weibull PDF was also fitted to these data distributions, yielding the parameters presented in Tables 3.65 and 3.66 for the respective resolutions. In the case of resolution 320×240 , the Weibull PDF has a scale parameter $\lambda = 0.0055$ m, which, like in the systematic cases, approaches the median value of absolute error along the trajectory. In case of resolution 1600×1200 , the scale parameter goes down

Distance absolute error e_{abs} per test position
Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras

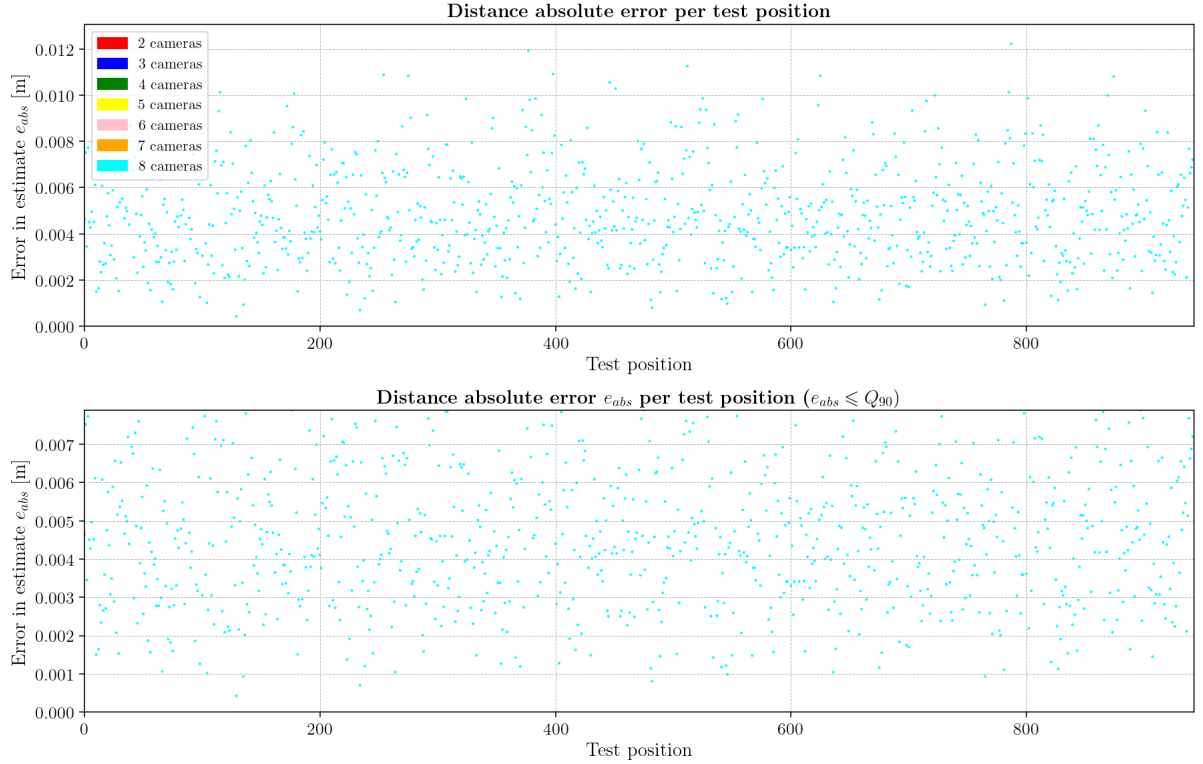


Figure 3.97: Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 03.

Table 3.63: Global statistics and root mean square error for test 03 with resolution 320×240.

	Abs. error	Est. error
count	1884	1884
mean	0.0049	0.0238
std	0.0021	0.0054
min	0.0004	0.0073
25%	0.0033	0.0200
50%	0.0047	0.0235
75%	0.0063	0.0272
90%	0.0079	0.0311
max	0.0129	0.0428
RMSE	0.0054	

Table 3.64: Global statistics and root mean square error for test 03 with resolution 1600×1200.

	Abs. error	Est. error
count	1884	1884
mean	0.0008	0.0037
std	0.0003	0.0008
min	0.0001	0.0015
25%	0.0005	0.0032
50%	0.0007	0.0037
75%	0.0010	0.0042
90%	0.0012	0.0048
max	0.0028	0.0068
RMSE	0.0008	

to $\lambda = 0.0009$ m. In both cases, the scale parameters for both resolutions approach the medians for the corresponding resolutions, as presented in Table 3.63 and Table 3.64.

Table 3.65: Weibull PDF parameters and bin count per camera used in figure 3.101 for test 03 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.4447	0.0000	0.0055	1884	21

Distance absolute error e_{abs} per test position
Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

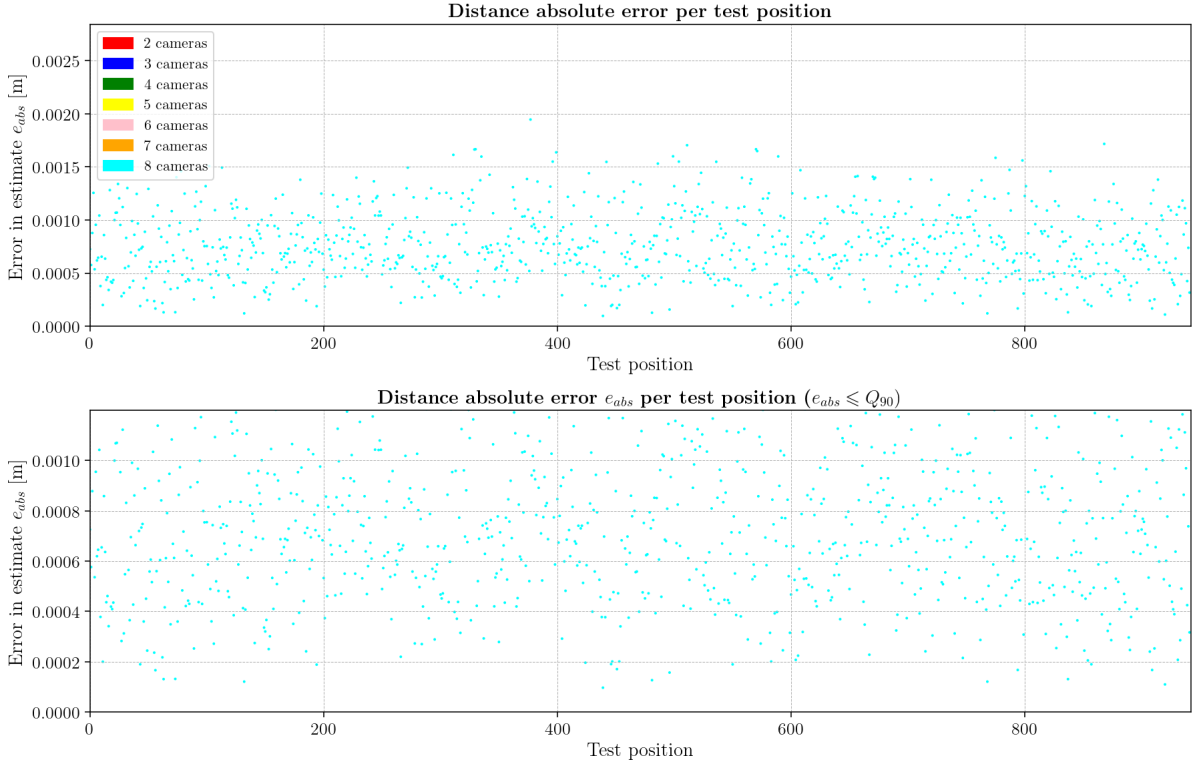


Figure 3.98: Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 03.

Table 3.66: Weibull PDF parameters and bin count per camera used in figure 3.102 for test 03 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
8	2.4538	0.0000	0.0009	1884	30

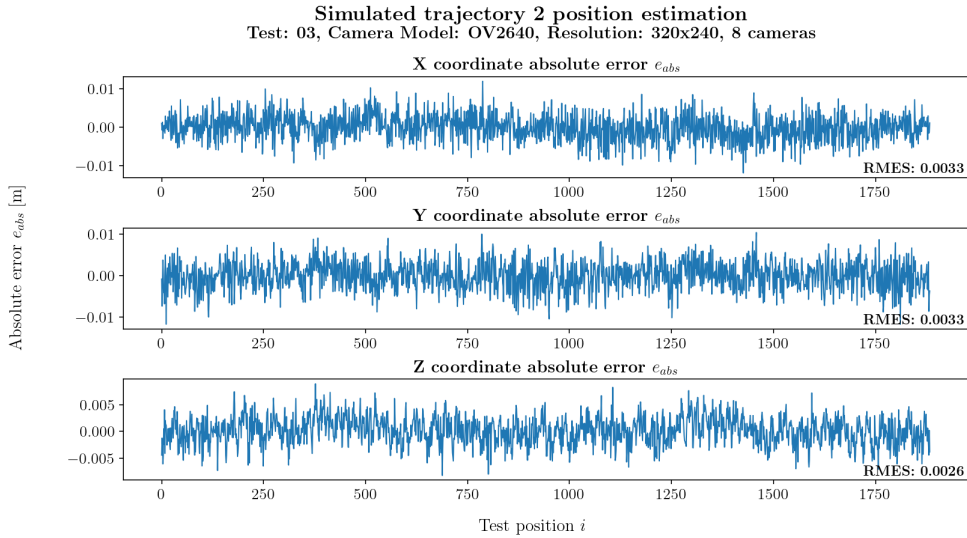


Figure 3.99: Real error in position estimation for trajectory 2 at resolution 320×240 and test 03, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

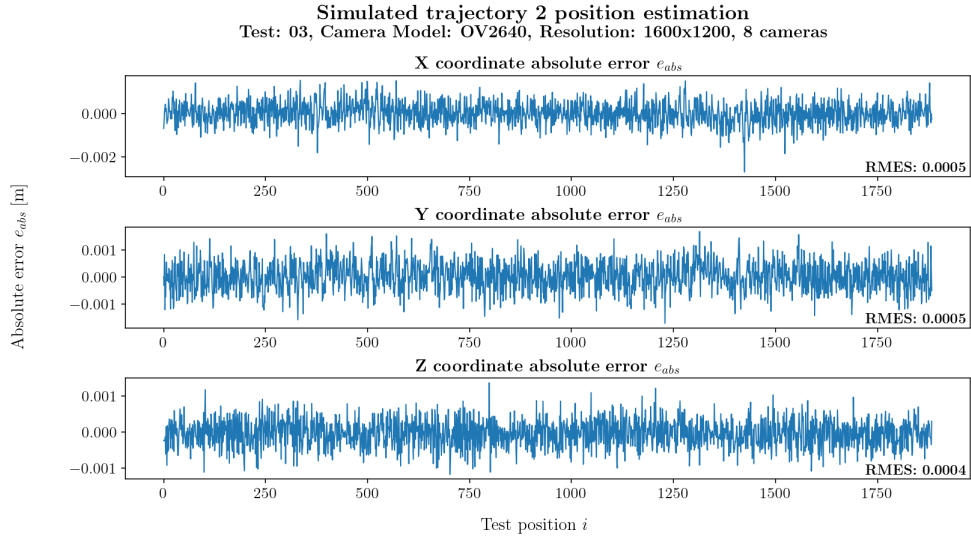


Figure 3.100: Real error in position estimation for trajectory 2 at resolution 1600x1200 and test 03, for x coordinate (top), y coordinate (middle) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

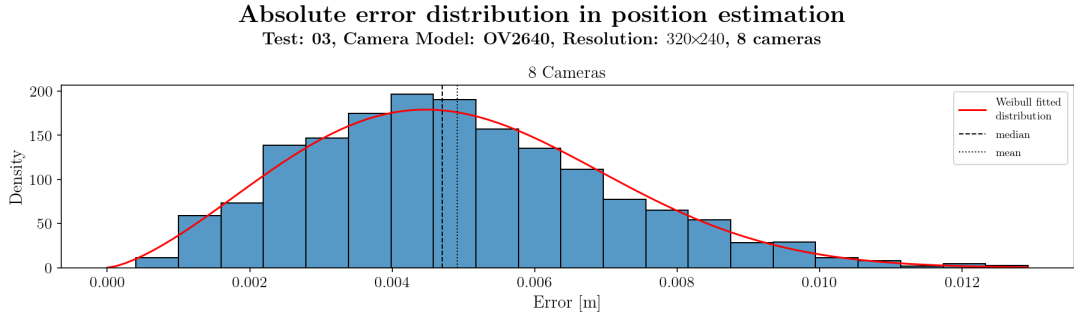


Figure 3.101: Error distribution histogram for trajectory 2 and test 03 at resolution 320x240.

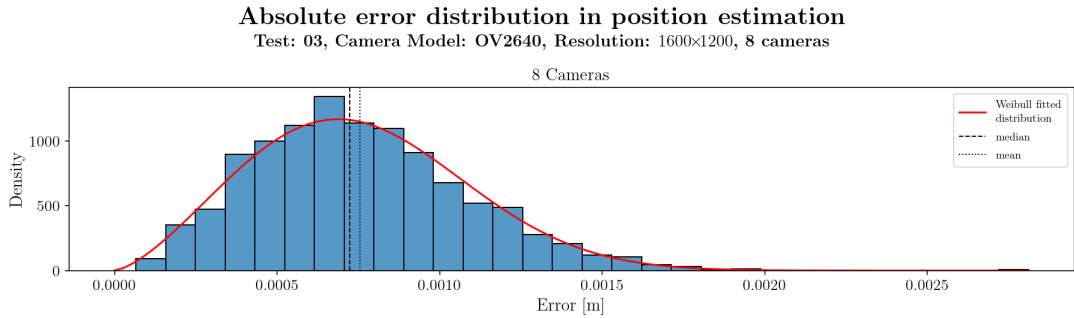


Figure 3.102: Error distribution histogram for trajectory 2 and test 03 at resolution 1600x1200.

3.7.4 Analysis of Test 04

The final camera configuration test 4 applied to the figure 8 trajectory is evaluated in this section. The simulated trajectory in a dashed black line with the estimated points in red is presented in Figure 3.103. As previously described, the left plot is the trajectory projected onto the XY plane, and the right plot is its projection onto the XZ plane. In Table 3.67 and Figure 3.104, it can be reconfirmed the ability of the algorithm to recover the beacon position 90% of the time with an error smaller than $e_{abs} \leq 0.0100$ m when using a low resolution of 320×240 . Also, in that same table, it can be seen that the mean error is $\bar{e}_{abs} = 0.0064$ m, standard deviation $\sigma = 0.0028$ m and median $\tilde{e}_{abs} = 0.0062$ m.

Looking now at the high resolution of 1600×1200 , Table 3.68 and Figure 3.105 demonstrate that the error in position estimation reduces to 90% of values with $e_{abs} \leq 0.0015$ m, with mean error $\bar{e}_{abs} = 0.0009$ m, standard deviation $\sigma = 0.0004$ m and median $\tilde{e}_{abs} = 0.0009$ m.

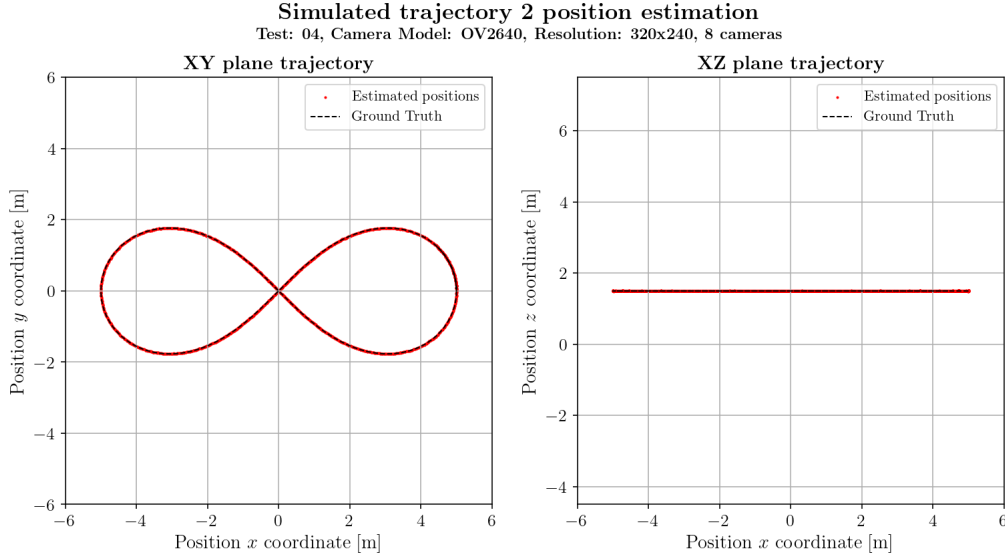


Figure 3.103: Trajectory 2 ground truth (dashed black line) and measured position using this work's algorithm, for resolution 320×240 and test 04

The error of the system per coordinate component for resolutions 320×240 and 1600×1200 is presented in figures 3.106 and 3.107, respectively, for every point recovered along the trajectory. Like in tests 1 and 2, on this third test, there are also some irregularities in both graphs, like in previous cases, more noticeable in the components related to x and y coordinates. This happens once again when the beacon approaches the intersection point along the figure 8 trajectory.

At resolution of 320×240 , the maximum error in x, y and z coordinates is of the order $e_x \sim e_y \sim e_z \sim 0.015$ m. The root mean square error (RMSE) per coordinate between the simulated trajectory points and the estimated points for this test configuration were $RMSE_x = 0.0039$ m, $RMSE_y = 0.0042$ m and $RMSE_z = 0.004$ m, showing that in the z coordinates the measurements were more once again accurate. This may again be due to the fact that the trajectory was contained in the XY plane.

For resolution 1600×1200 , the maximum error in x, y and z coordinates decreased to approximately $e_x \sim e_y \sim e_z \sim 0.002$ m. The root mean square error (RMSE) between the simulated trajectory points and the estimated points computed for each coordinate were $RMSE_x = 0.0006$ m, $RMSE_y = 0.0006$ m and $RMSE_z = 0.0006$ m.

The error data distribution is again plotted in the histograms for both resolutions presented in this section. From all the 8 cameras available in the simulation, there were positions along the trajectory that were only detected by 7 cameras. Figure 3.108 shows the histograms for resolution 320×240 and Figure 3.109 for resolution 1600×1200 . A Weibull PDF was once more fitted to these data distributions, with the output parameters compiled in Tables 3.69 and 3.70 for the respective resolutions. In the case of resolution 320×240 , the Weibull PDF has a scale parameter $\lambda = 0.0067$ m, which, like in the systematic cases, approaches the median value of absolute error along the trajectory. In case of resolution 1600×1200 ,

Distance absolute error e_{abs} per test position
Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

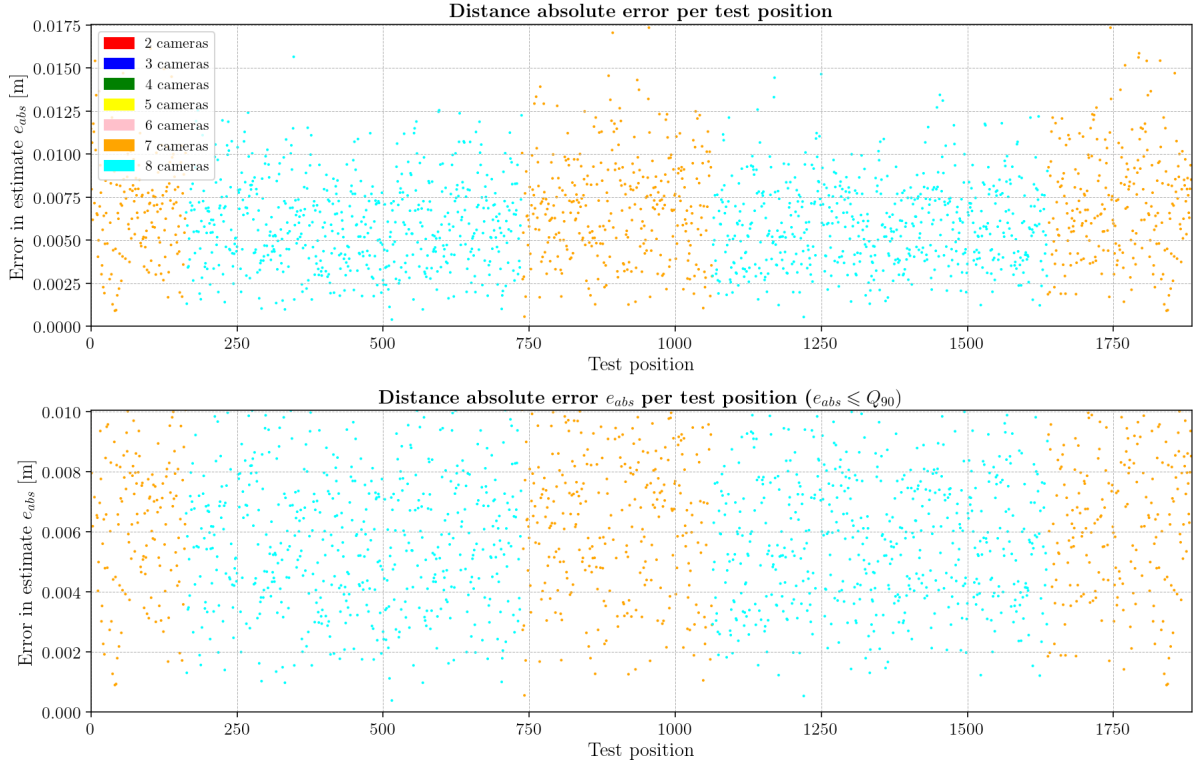


Figure 3.104: Absolute error e_{abs} per point in the trajectory for resolution 320×240 and test 04.

Table 3.67: Global statistics and root mean square error for test 04 with resolution 320×240.

	Abs. error	Est. error
count	1884	1884
mean	0.0064	0.0301
std	0.0028	0.0071
min	0.0004	0.0081
25%	0.0042	0.0249
50%	0.0062	0.0301
75%	0.0081	0.0347
90%	0.0100	0.0397
max	0.0173	0.0539
RMSE	0.0070	

Table 3.68: Global statistics and root mean square error for test 04 with resolution 1600×1200.

	Abs. error	Est. error
count	1884	1884
mean	0.0009	0.0044
std	0.0004	0.0009
min	0.0001	0.0008
25%	0.0006	0.0037
50%	0.0009	0.0043
75%	0.0012	0.0050
90%	0.0015	0.0056
max	0.0025	0.0073
RMSE	0.0010	

the scale parameter goes down to $\lambda = 0.0009$ m. In both cases, the scale parameters for both resolutions approach the medians for the corresponding resolutions, as presented in Table 3.67 and Table 3.68.

Table 3.69: Weibull PDF parameters and bin count per camera used in figure 3.108 for test 04 and resolution 320×240.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
7	2.4420	0.0000	0.0080	735	15
8	2.5339	0.0000	0.0067	1149	19

Distance absolute error e_{abs} per test position
Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

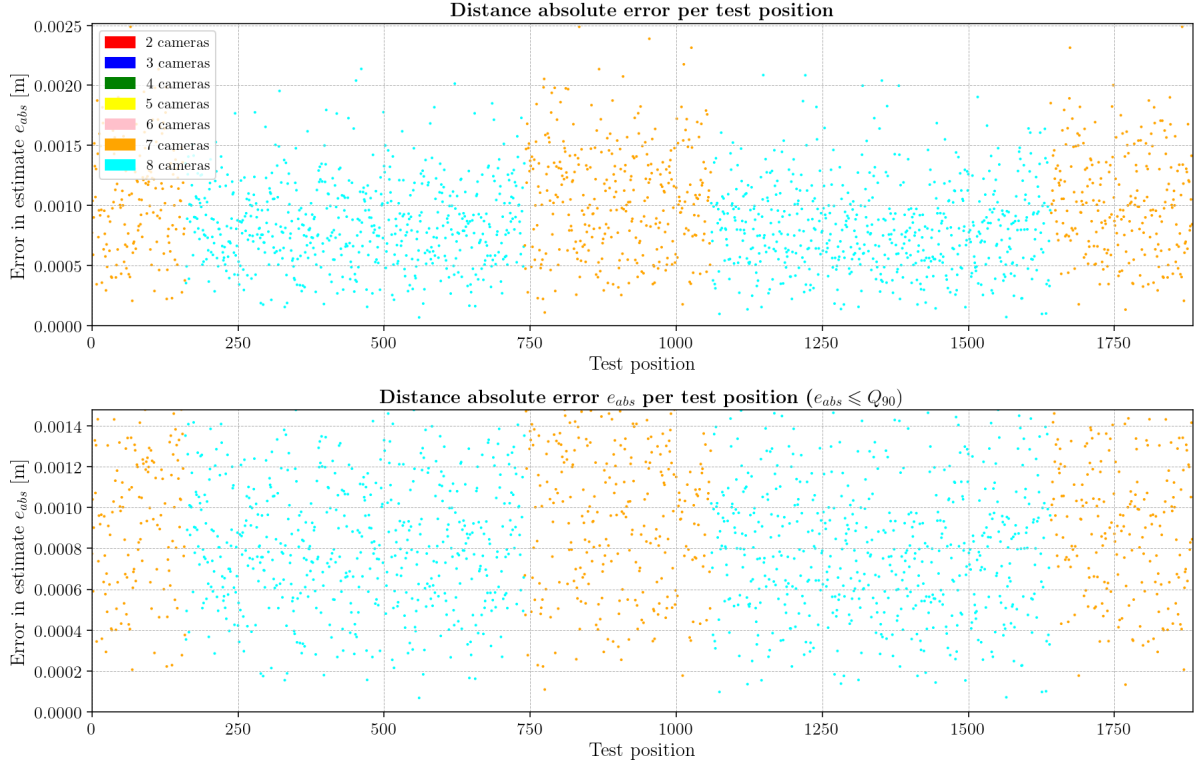


Figure 3.105: Absolute error e_{abs} per point in the trajectory for resolution 1600×1200 and test 04.

Table 3.70: Weibull PDF parameters and bin count per camera used in figure 3.109 for test 04 and resolution 1600×1200.

N ^{er} Cameras	Shape	Loc	Scale	Count	N ^{er} bins
7	2.5592	0.0000	0.0012	722	14
8	2.4784	0.0000	0.0009	1162	18

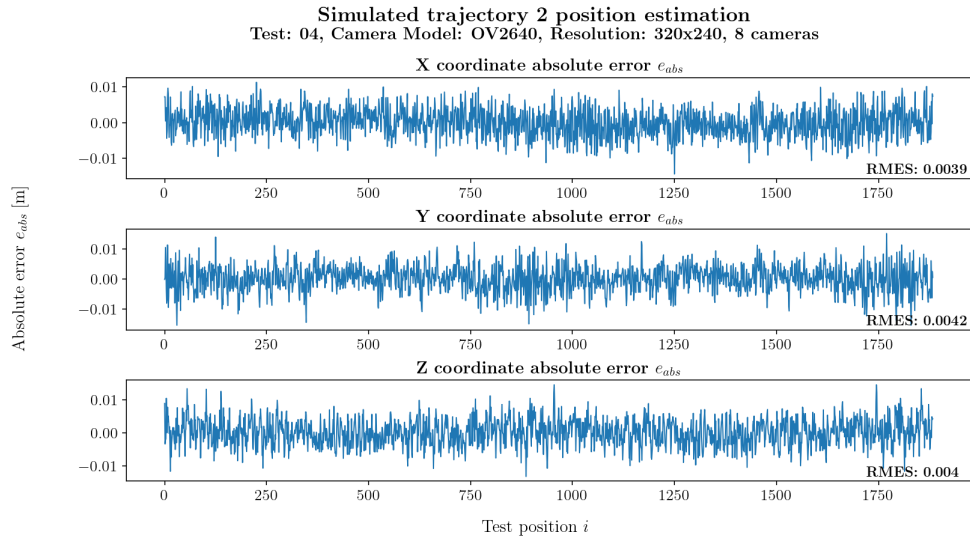


Figure 3.106: Real error in position estimation for trajectory 2 at resolution 320×240 and test 04, for x coordinate (top), y coordinate (centre) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

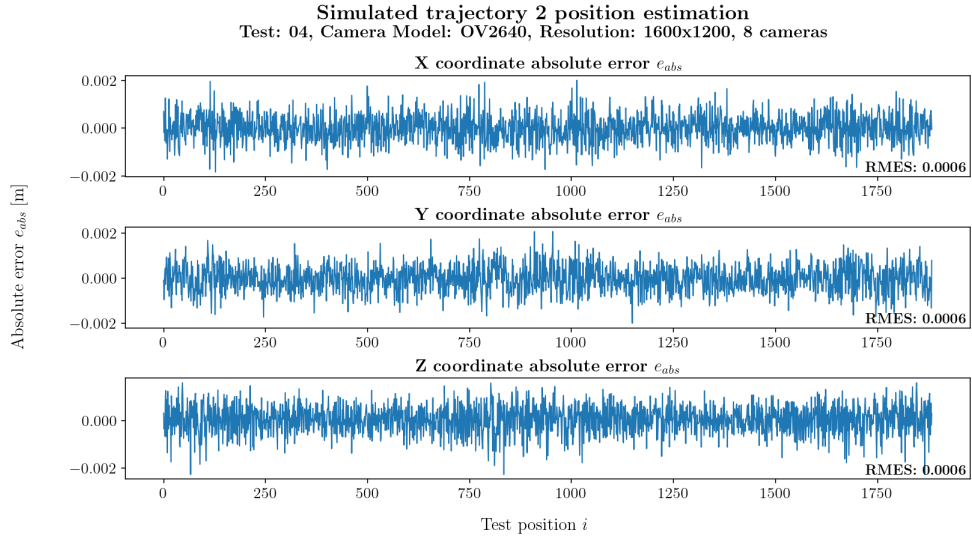


Figure 3.107: Real error in position estimation for trajectory 2 at resolution 1600x1200 and test 04, for x coordinate (top), y coordinate (centre) and z coordinate (bottom). The root mean squared error of the measurements is indicated at the bottom right of each coordinate plot.

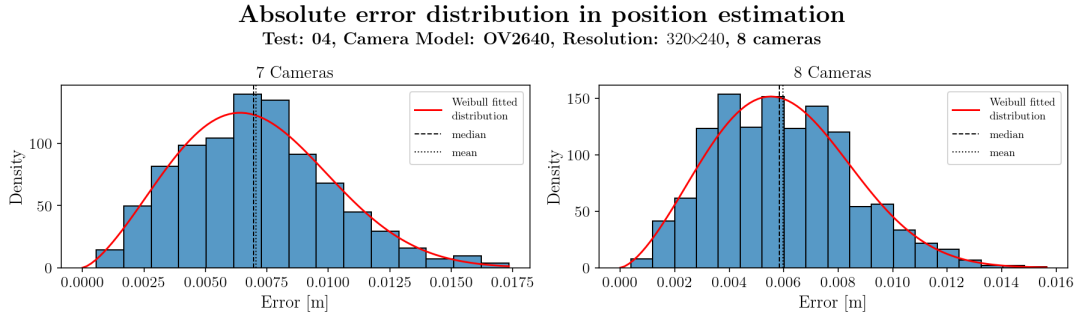


Figure 3.108: Error distribution histogram for trajectory 2 and test 04 at resolution 320x240.

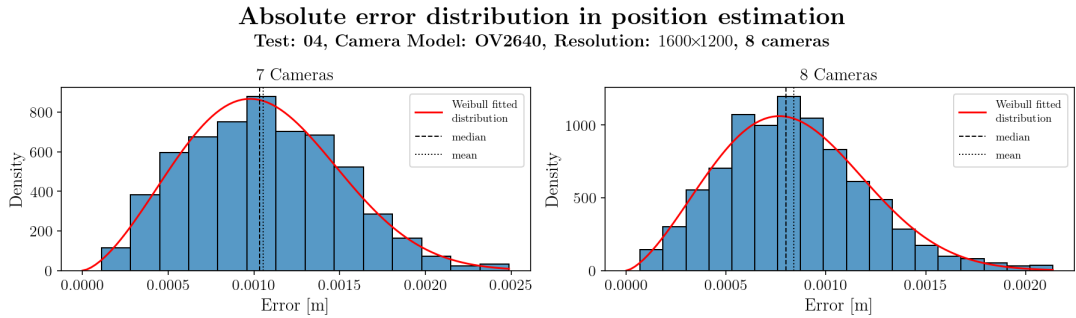


Figure 3.109: Error distribution histogram for trajectory 2 and test 04 at resolution 1600x1200.

Chapter 4

Implementation

This chapter presents the cameras and support electronics used to implement a test version of the system. For this purpose, economic prototyping cameras were used, namely the ESP32-CAM with the OV2640 sensor described in more detail in subsection 4.1.1. In this same section, it is also presented the 3D printed enclosure for the cameras and its support base. In section 4.1.2, the infrared (IR) beacon developed to test the system is presented. In section 4.2 is described the Wi-Fi configuration of cameras and the communication protocol created to send the images from the cameras to the tracking software. Lastly, in section 4.3, the tracking software written in Python is described, which implements the position estimation and blob detection functions that are the basis of the system, as described in chapter 2.2.

4.1 Hardware

This section focuses on presenting the hardware used to implement the system proposed in this study, namely the cameras and microcontroller chosen to implement the system (subsection 4.1.1) and the beacon design (subsection 4.1.2).

4.1.1 Cameras

To test the ideas behind this work, the consumer electronics and prototyping board ESP32 CAM was chosen [29]. These were mainly chosen because of ease of programming (they can either be programmed in C using Arduino IDE or with MicroPython using a Python IDE) and because of their inexpensiveness. The prototyping boards are reasonable for creating a test version of the system planned for this work. They provide remote connections via Wi-Fi or Bluetooth, and they have the possibility of connecting to an external antenna that increases the range of connection to these cameras (see Figure 4.1). Table 4.1 compiles the main technical specifications of the ESP32-S microcontroller present in these boards, with more information available in the data-sheet from the manufacturers [29]. They have proven to be reliable and a great option to implement a prototype of the system.

To keep the cameras working in a powered stand-alone fashion, we opted to provide power from a typical 5V power bank. To be able to do so, without having to build custom support electronics, expansion boards were used that already have a mini USB connector built in. This connector also became useful in programming the cameras by enabling the programming of the boards without using another microcontroller to act as a bridge between the programming computer and the microcontroller.

To physically support the cameras, 3D printed enclosures have been made, based on free models already available online, with small modifications to receive the filter holder and the support for the case (for example, to allow space for the USB cable that connects to the Powerbank to fit into its respective connector). Figure 4.2 shows these 3D parts produced, along with the IR filter used.

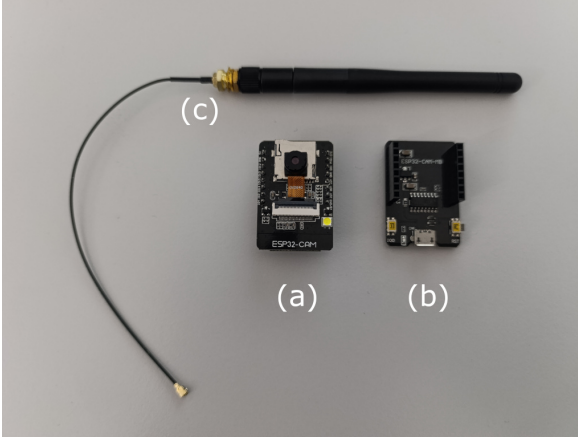


Figure 4.1: ESP32-CAM used in this work (a), with sensor OV2640 attached. Also in the image are the expansion board (b) and the external antenna (c).

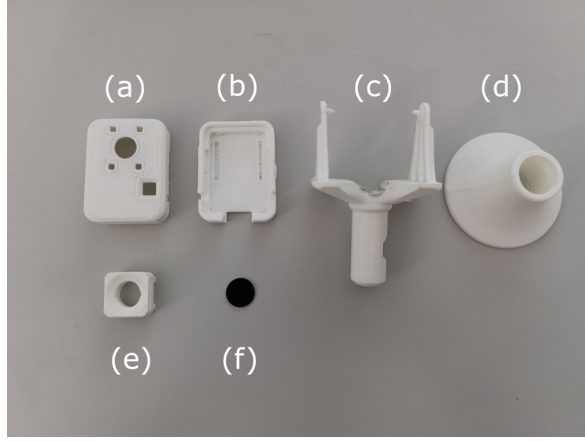


Figure 4.2: 3D printed camera casing (a, b) and support (c). Also shown in the image are the support base (d), the filter holder (e), and the IR filter (f).

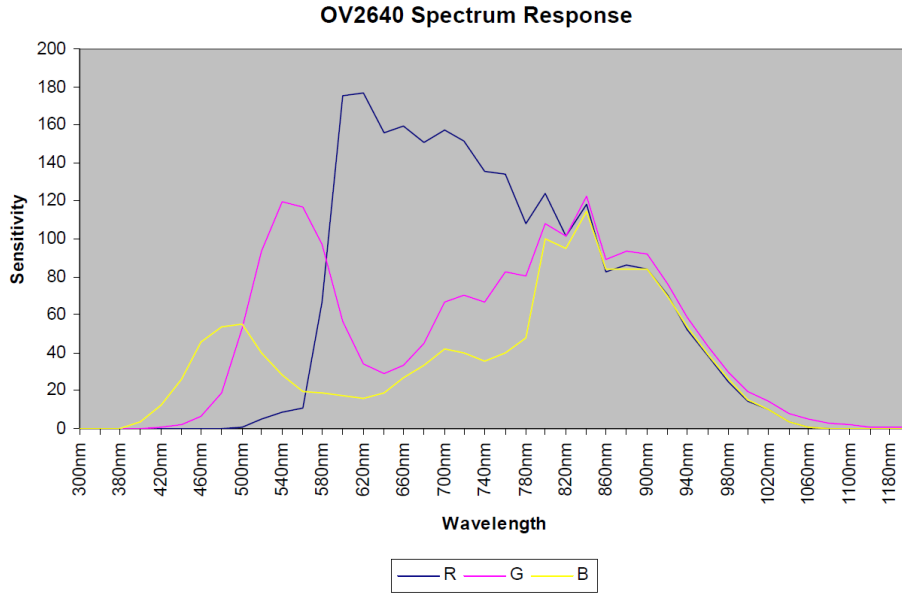


Figure 4.3: Spectral response of image sensor OV2640, adapted from the sensor data sheet.

Table 4.1: Technical specifications of the ESP32-CAM

Feature	Specification
Microcontroller	ESP32-S (dual-core, 32-bit LX6, up to 240 MHz)
RAM	520 kB SRAM
Flash memory	4 MB (external SPI)
Supported camera	OV2640 (2 Mp), OV7670
Camera resolutions	QQVGA (160×120) up to UXGA (1600×1200)
Wireless connectivity	Wi-Fi 802.11 b/g/n (up to 150 Mbit s ⁻¹)
Bluetooth	Not available
Available GPIOs	9 pins (multiplexed)
Communication interfaces	UART, SPI, I ² C, PWM, ADC
MicroSD card slot	Up to 4 GB (SPI/SDIO)
Power supply voltage	5 V (internal regulator to 3.3 V)
Typical current consumption	160 mA to 260 mA (Wi-Fi transmission)
Dimensions	27 mm × 40.5 mm × 4.5 mm

Table 4.2: Technical specifications of the OV2640 image sensor

Feature	Specification
Sensor type	CMOS image sensor
Resolution	2 Mp (1600 × 1200)
Output formats	YUV, JPEG, RGB565, RAW
Image size options	QQVGA (160×120) up to UXGA (1600×1200)
Lens	Focal length 2.96 mm, F/2.8
Diagonal field of view (FOV)	65° (typical, depends on module lens)
Pixel size	2.2 μm × 2.2 μm
Shutter type	Rolling shutter
Maximum frame rate	15 fps @ UXGA, 30 fps @ SVGA (800×600)
Supply voltage	2.5 V (core), 1.8 V (I/O), 3.0 V (analog)
Power consumption	60 mW (typical)
Operating temperature	−30 °C to 70 °C
Package	CSP2, 24 pin

The OV2640 is a 2K camera designed around 2005, which was common to find in first-generation smartphones. Its specifications are compiled into Table 4.2, and more info is available in the respective data-sheet [23]. In the referred data sheet, there is information regarding the spectral response of the camera, which is presented in Figure 4.3, showing that maximum sensitivity in both R, G and B channels is located around 860 nm, very close to the chosen beacon IR emission wavelength, described in the next section.

4.1.2 Beacon

The beacon idealised in this work, to be tracked by the cameras, was projected to emit infrared (IR) light, having in mind the simplification of the blob detection algorithm. By using IR, the cameras could be fitted with an IR pass band filter, rejecting most of the visible spectrum, and this way reducing the chances of detecting false positives. By analysing the spectral response of the OV2640 sensor (see 4.3), a 3W IR LED with 850 nm wavelength is used as the light-emitting component of the beacon. This LED has a rather small viewing angle, so in order to spread the emitted light and enable the detection from wider angles between cameras and beacon, a Fresnel lens was adapted into the PCB. Usually, these Fresnel lenses are used in the reverse process of concentrating light into an IR detector, i.e. motion detectors and such. The use of this Fresnel lens proved to work as expected, spreading the light from the IR LED and enabling detection from wider angles.

Figure 4.4 shows the electronics schematics for the beacon, which incorporates 4 resistors to dissipate power, as the IR LED requires a rather large amount of current. It also serves as a connector for power cables, and it was also prepared with support for USB power connection, having in mind the use of power banks to provide power to the beacon.

Figures 4.5 and 4.6 show the designed Printed Circuit Board (PCB) for this project. In Figures 4.7 and 4.8, the constructed beacon is presented.

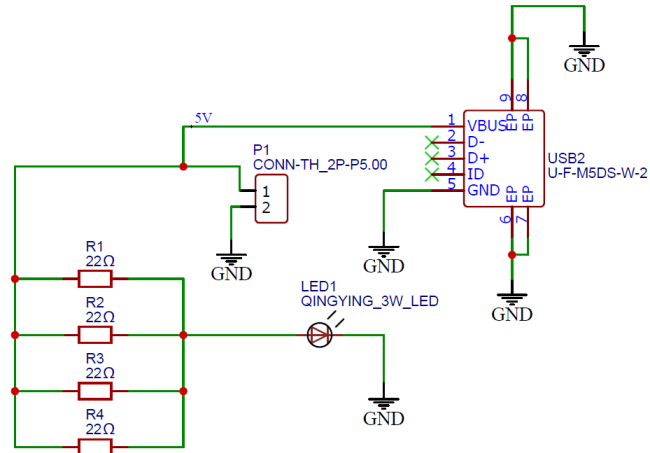


Figure 4.4: IR Beacon schematic circuit used to develop the PCB

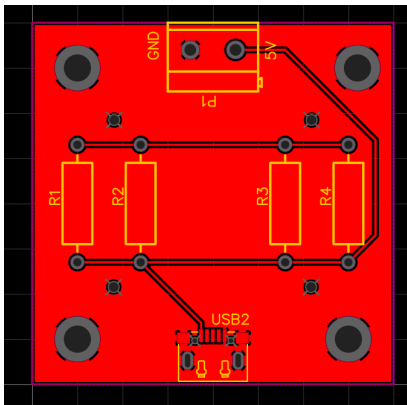


Figure 4.5: Top layer of the IR Beacon PCB

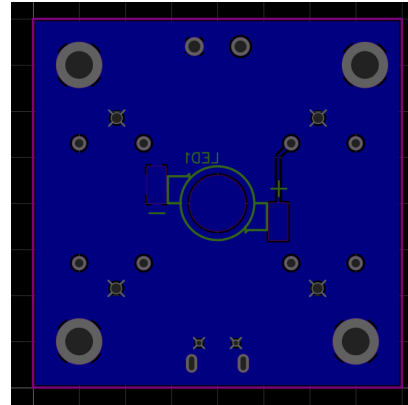


Figure 4.6: Bottom layer of the IR Beacon PCB

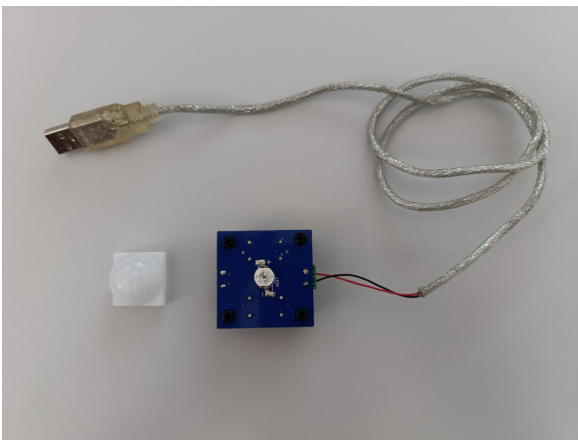


Figure 4.7: IR Beacon built for this project - top side

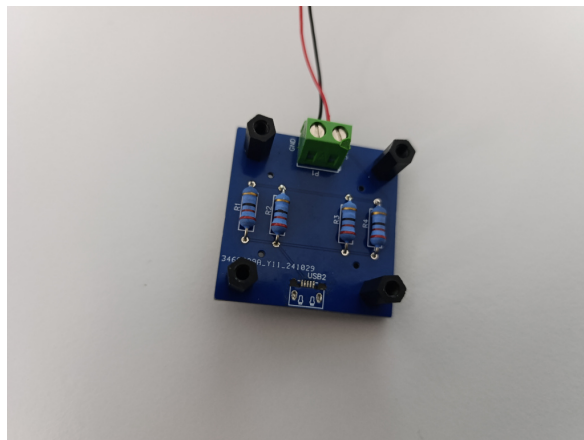


Figure 4.8: IR Beacon built for this project - bottom side

4.2 Communication

The ESP32-CAM were programmed to communicate with the tracking software (described in section 4.3) via Wi-Fi. To achieve this goal, the example code provided with the ESP32 Arduino library “CameraWebServer.ino” was modified to include the possibility of creating an Access Point, over which the users can configure the Wi-Fi network they want to connect the cameras to and define the camera name in the system.

This modified version of the code was then renamed as “CameraWebServerAP.ino”. Figure 4.9 shows the camera Access-Point page as viewed from a mobile device (Figure 4.10 shows the same screen for pc users), which is used to configure the network to be used and the camera name in the system. This page is reachable when the camera is not able to connect to the currently saved network in EEPROM, or if there is no saved network, in which case the camera turns on the access point. After connecting to the access point, the user can use a browser to connect to the 192.168.4.1 IP address. The configuration web page was programmed in HTML language, and C code was added into the “CameraWebSeverAp.ino” to handle the Wi-Fi connection.

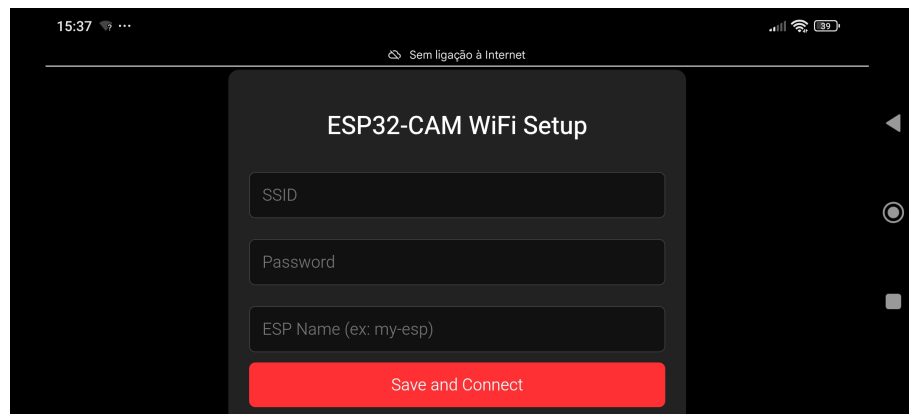


Figure 4.9: Camera access point screen, where the user can insert the details of the Wi-Fi network desired to be used, the Wi-Fi password and the name to be given to the camera, as viewed in a mobile device.

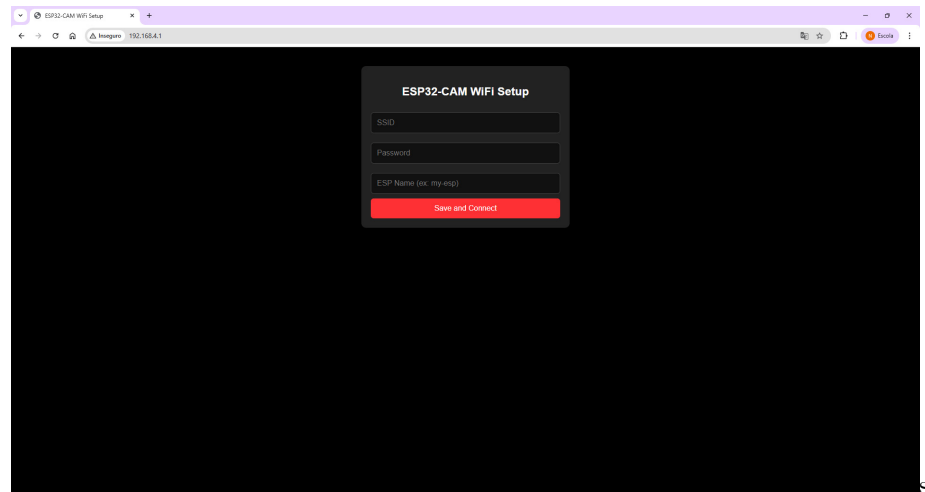


Figure 4.10: Camera access point screen as in figure 4.9, but from a computer.

After a successful connection to the desired Wi-Fi network, the ESP32-Cam starts the camera capture and streams the data to a web page, accessible via browser, to which an interface was programmed in Python, enabling the direct capture from the developed software. Also, there is a control page, reachable from a browser (see Figure 4.11), to which was also programmed an interface from the Python software. This enables remote capture and control of the cameras.

To access the camera’s live feed, the URL used is based on the camera name given by the user, following the pattern:

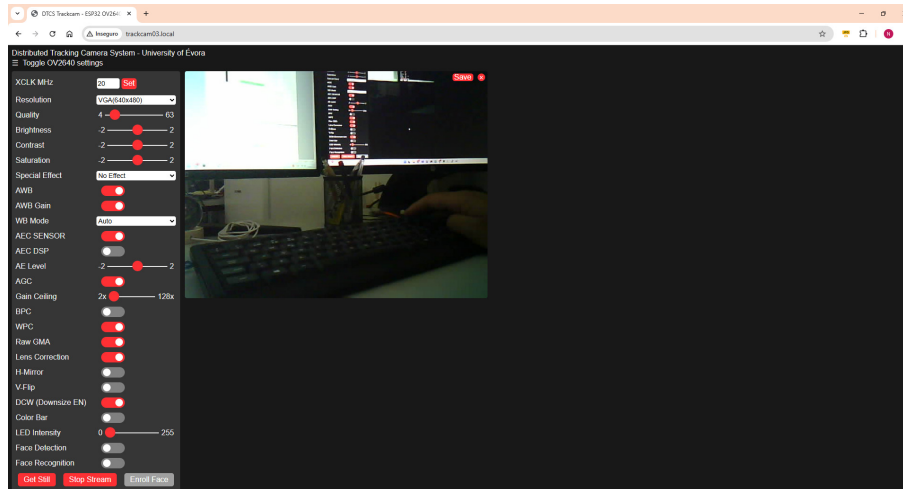


Figure 4.11: Camera control screen after successful connection of camera to Wi-Fi network.

`http://<camera_name>.local/stream`

for example, if the camera has been named by the user as “trackcam01” the corresponding live feed URL is going to be accessible at

`http://trackcam01.local/stream.`

4.3 Distributed Tracking Camera Software

This section describes the software developed in Python in order to provide an easy way for users to interface with the Distributed Tracking Cameras System proposed in this work. All the functions that implement the blob detection (see Chapter 2, subsection 2.2.2) and the position estimation algorithm are programmed into it. It will be described in some more detail below and in the subsection 4.3.2, the communication interface with other systems (e.g., an autonomous robot, another computer or a microcontroller) is also described. However, this is a preliminary version of the software that contains only minimal functionality to make the system work.

4.3.1 Description of the implemented software

Here, the Python software named DTCS (Distributed Tracking Camera System), which was created to interface the cameras and provide an estimation of location based on the blob detections from each camera in the system, is described.

As stated in the introduction of this chapter, all the functionality regarding blob detection and position estimation has been programmed as the base of the software. A Python class was created to interface the cameras via Wi-Fi, and into that class, all the properties of the physical camera are stored (e.g. camera position and orientation, sensor size and resolution, etc.). This camera class also implements methods for blob detection, and each camera runs in its own thread, separated from the main function that runs continually during the execution of the software.

Figure 4.12 shows the Graphical User Interface (GUI) created to ease the use of the software by any user and to provide an intuitive interface to the functionality of the software.

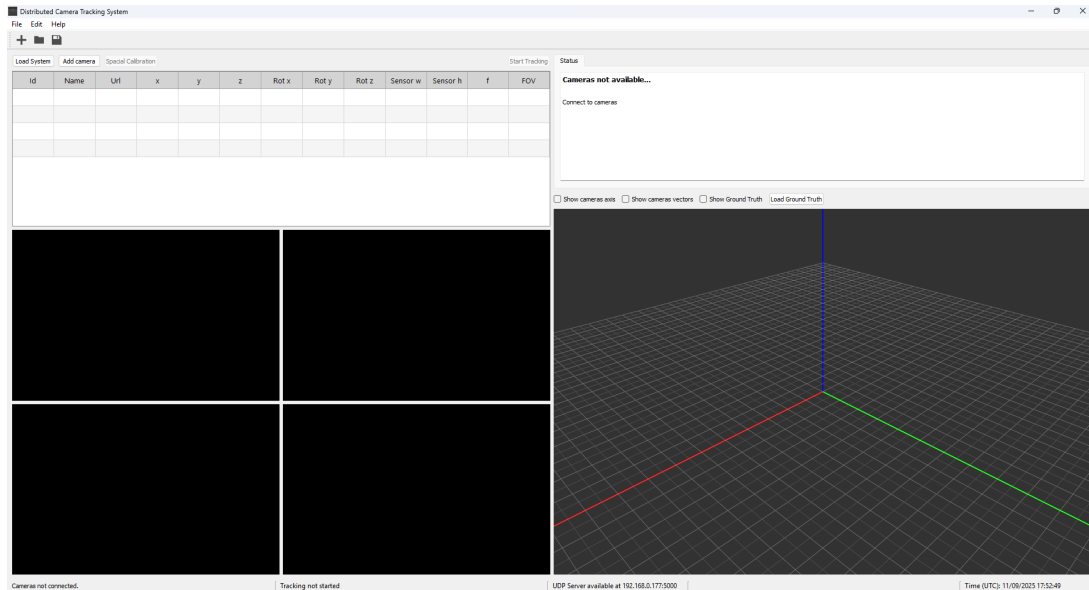


Figure 4.12: Main window of the Distributed Camera System software developed in Python, using PyQt5.

This main window was created using PyQt5 [30] and has buttons to load a file containing the camera’s information being used in the system, or in case such files don’t exist (e.g. in a first installation of the cameras system), there is another button that opens a pop-up window that enables the user to create cameras in the system. This window is called “Add new camera” (see Figure 4.13) and has fields where the user can input the properties of each camera. Pressing the “OK” button closes the window and loads the camera into the system. The user can add up to 8 cameras in the current version of the software.

It should be noted that for the remote connection over Wi-Fi to the cameras to work successfully, both the computer running the DTCS software and the cameras must be connected to the same Wi-Fi network, which should also be configured in the cameras as explained in section 4.2.

After loading all the cameras into the system, the user can save a “scenario”, which basically saves the information of all the cameras into a file that can be loaded in a later execution of the software. To save

a camera configuration, the user can use the menu “File” - “Save”.

This file will always be valid if the physical disposition of cameras has not changed from one execution to another. To load the file, the user can press the button “Load System” or go to the menu “File” - “Open”.

Accessible over the top menu via “Edit” - “Preferences”, there is also a “Preferences” window (Figure 4.14), where the user can provide a location to store Log files, and where the IP address where the position estimation server is running can be identified. This IP address cannot be changed by the user, but there is an option to define the associated Port.

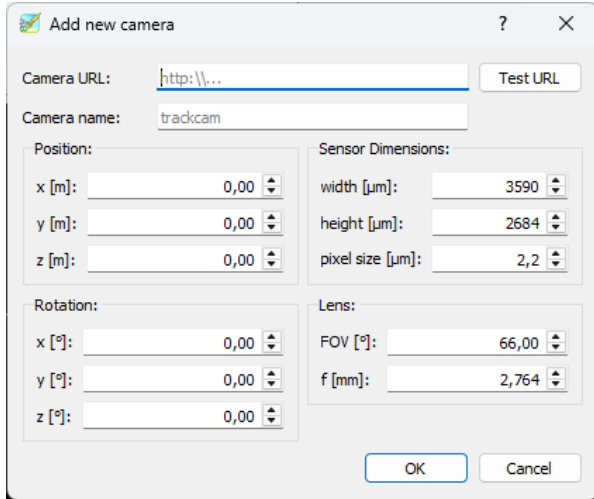


Figure 4.13: DTCS “Add New Camera” menu.’

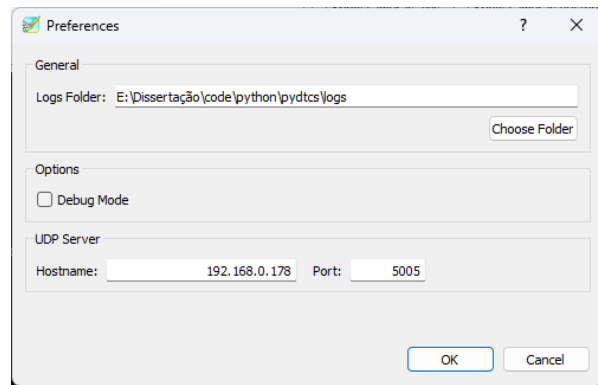


Figure 4.14: DTCS “Preferences” menu.

After at least 2 cameras have been loaded into the system, the “Start Tracking” button becomes enabled, allowing the user to start tracking when pressing it. Figure 4.16 shows the software while tracking using cameras.

After a camera is connected to the system, a Tab becomes available that allows the control of settings of that camera remotely, like resolution, exposure, gain, etc. Each camera has its own control tab, as seen in Figure 4.15.

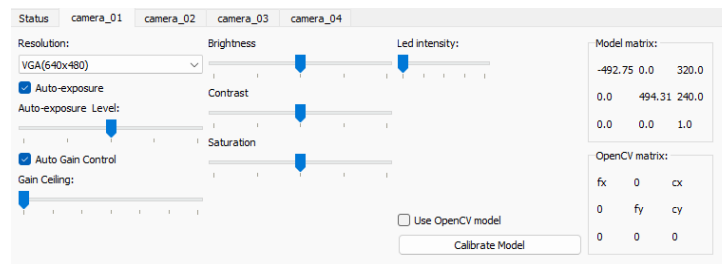


Figure 4.15: Zoom into the camera controls tab, displaying some of its interfaces to the remote camera

It is visible in the lower right corner of the GUI a 3D render of the scene, where cameras are displayed. The user can interface with this 3D visualisation with the mouse, allowing rotation and translation of the field in view. While tracking, this scene can also be enabled to show the \mathbf{v} vectors from each camera, as seen in the aforementioned figure. A zoomed view of this 3D window is presented in Figure 4.17, where it is also visible that the “Status” Tab displays values of position and cameras tracking the object. Cameras with a green bounding box are tracking; if not tracking, they become greyed out.

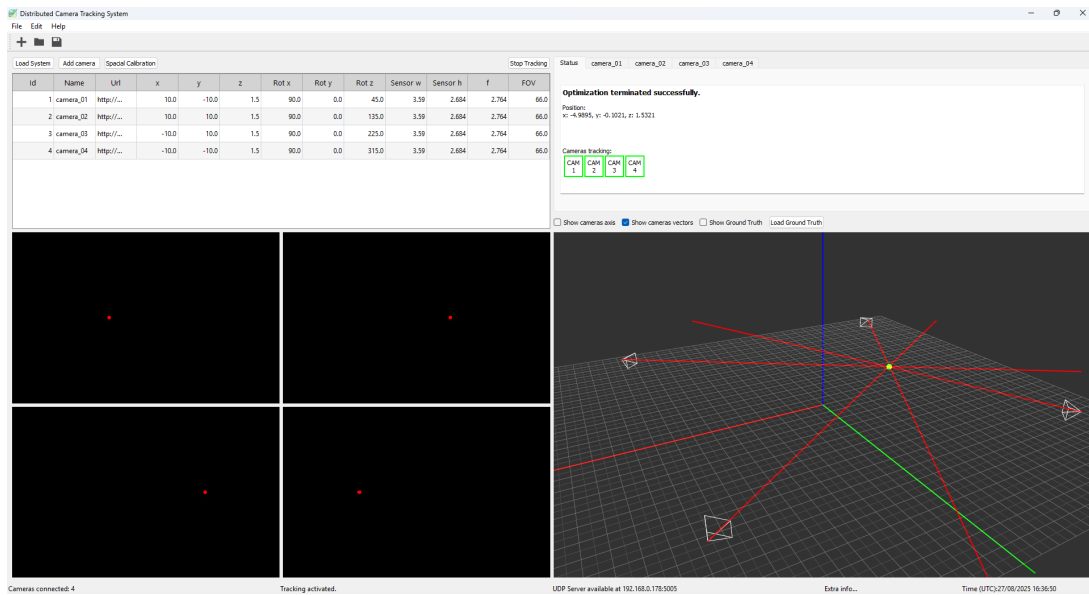


Figure 4.16: Main window of the Distributed Camera System software with cameras connected and tracking a blob, estimating its position.

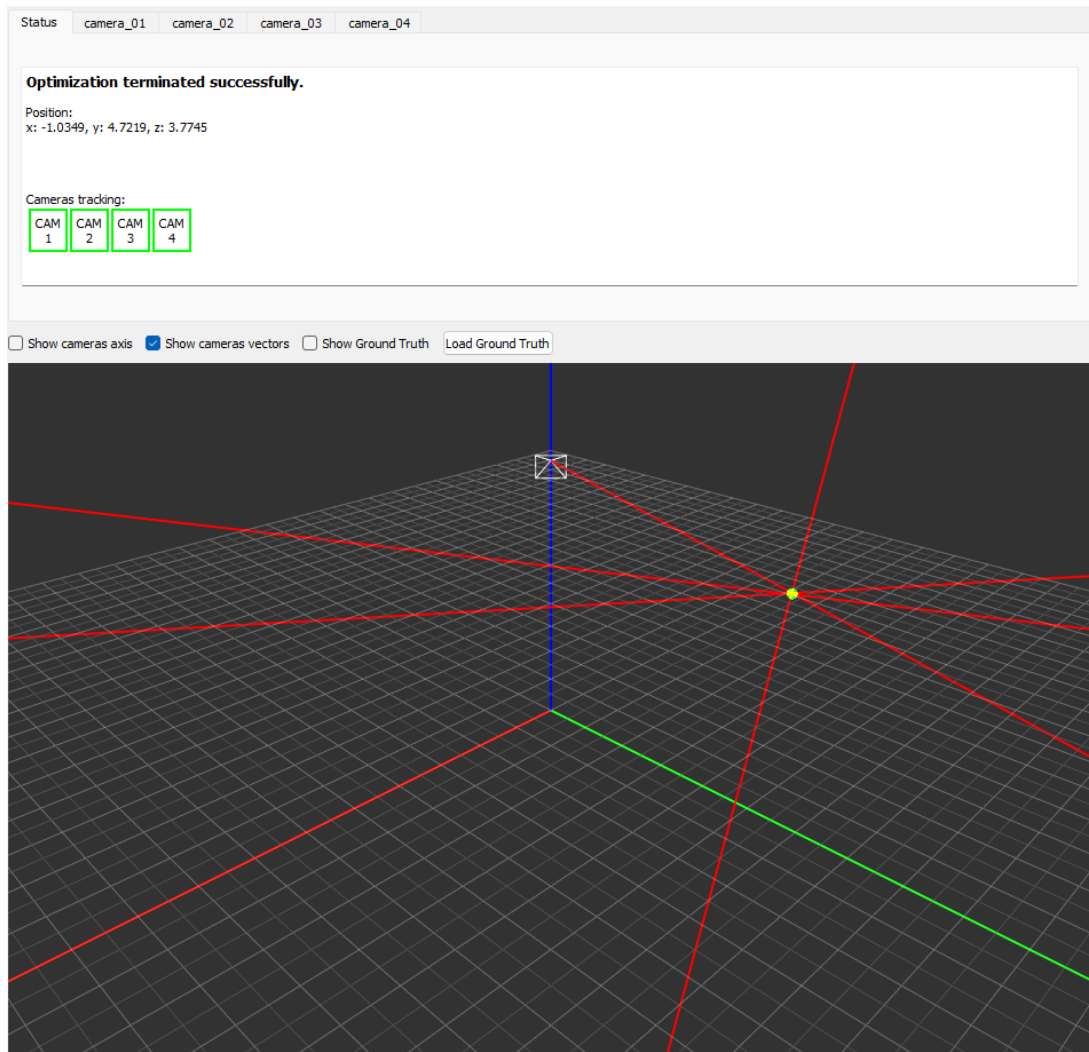


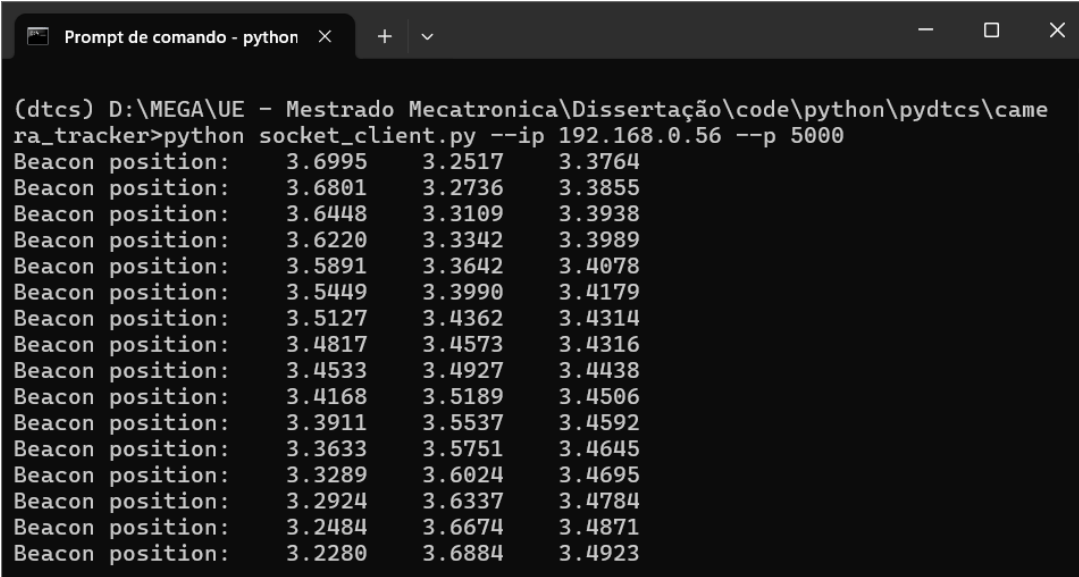
Figure 4.17: Zoom into the Main window of the Distributed Camera System software, showing the “Status” bar with cameras connected and tracking a blob, estimating its position. Cameras are also displaying their predicted vectors.

4.3.2 Communication with Robotic Systems

The purpose of this software is to provide a precise location of a beacon inside a space of interest. However, it is predictable that only knowing the location may not be very useful, if it can not be used to, e.g. provide navigation data to a robotic system or tracking data to another software. Having this in mind, it was programmed into the tracking software a tracking navigation data server, using “sockets” in Python.

When tracking is enabled, a server is started and waits for client connections. After clients connect to the server, it starts streaming the position data of the beacon being measured in real time. Each line of data sent by the server contains a sequence of x, y and z coordinates in meters, as seen, for example, in Figure 4.18.

The client socket software currently only prints to the terminal the positions received, as long as the server continues to send messages. But the sample script can later be used to instead of printing the coordinates, to feed a stream of positional data to a navigation and guidance routine in another software, e.g., a ROS-based system.



```
(dtcs) D:\MEGA\UE - Mestrado Mecatronica\Dissertação\code\python\pydtcs\came
ra_tracker>python socket_client.py --ip 192.168.0.56 --p 5000
Beacon position: 3.6995 3.2517 3.3764
Beacon position: 3.6801 3.2736 3.3855
Beacon position: 3.6448 3.3109 3.3938
Beacon position: 3.6220 3.3342 3.3989
Beacon position: 3.5891 3.3642 3.4078
Beacon position: 3.5449 3.3990 3.4179
Beacon position: 3.5127 3.4362 3.4314
Beacon position: 3.4817 3.4573 3.4316
Beacon position: 3.4533 3.4927 3.4438
Beacon position: 3.4168 3.5189 3.4506
Beacon position: 3.3911 3.5537 3.4592
Beacon position: 3.3633 3.5751 3.4645
Beacon position: 3.3289 3.6024 3.4695
Beacon position: 3.2924 3.6337 3.4784
Beacon position: 3.2484 3.6674 3.4871
Beacon position: 3.2280 3.6884 3.4923
```

Figure 4.18: Output of the socket client when connected to the DTCS embedded server. It shows the beacon position being transmitted over the socket connection.

The execution of the script can be done using the terminal, with the following command:

*

```
1 python socket_client.py --ip <<ip address of server>> --p <<port>>
```

where <<ip address of server>> should be replaced by the local IP of the server machine (which can be found in the DTCS GUI) and <<port>> should be replaced by the port defined in the system. By default, port 5000 is selected.

For example, in the case of one wanting to test the system locally, running both the client Python script and DTCS server on the same computer, the client Python script can be called like:

*

```
2 python socket_client.py --ip 127.0.0.1 --port 5000
```


4.4 Real tests of the system

This section details some of the physical tests done on the proposed visual tracking system. It presents the approaches used in assessing the accuracy and precision of the system, despite being done at a reduced scale compared with the simulations. It also presents some of the difficulties faced during the execution of these tests.

The section starts by presenting in subsection 4.4.1 the physical setup of tests and cameras positioning and alignment. subsection 4.4.2 describes the measurements calibration paper sheet prepared to assist in the estimation of the physical world position of the beacon.

In subsection 4.4.3, the accuracy and precision of the system are analysed. Along with the statistical treatment of the positions measured, in subsection 4.4.4, the responsiveness of the system is also inspected by analysing the frame rate of the system.

4.4.1 Tests setup

To test the tracking system proposed in this work, an experimental setup was implemented in one of the University Physics Department Laboratories. In the laboratory bench, 4 cameras were set up around the corners and their positions measured. The orientations were estimated by trigonometry, by aiming the cameras at the centre of the table and using the measured positions of the cameras.

In the centre of the bench table, a A0 calibration paper sheet (described in next section) was glued to the table, to serve as a reference system for manual measurement of cameras position (in relation to the centre of the calibration paper) and to help in the manual measurement of the beacon position, to compare with the measurements given by the tracking system. This can be seen on Figure 4.19

The cameras were powered with individual power banks and connected to a local Wi-Fi network created for this test.

During these tests, a low quality in blob detection was mainly due to low signal-to-noise ratio (SNR) in the blob, compared to the image background. In other words, the illumination of the room was too intense, causing difficulties in detecting the 3W beacon.

To solve this issue, the room was darkened, and in this way, the beacon SNR increased considerably. The tests were then performed under these conditions.

After the camera setup and the software connection, the tracking function was started, allowing the beacon to be moved manually over the sheet while the software tracked its position. All the position estimations done during these tests were confirmed only by visual comparison of the value given by the software with the beacon position over the calibration sheet. In most cases, these values agreed with a difference that could go up to 20 mm. Figure 4.21, the beacon can be seen in one of those test positions during the tests.

In order to have a more concise analysis of the results of the tests, 10 positions were measured over the calibration sheet and recorded into a text file for later analysis, and while the software was tracking it, it generated a log file with all the positions estimated by the system. This analysis is presented in subsection 4.4.3.

In Figure 4.20, a camera with the respective Powerbank is shown.

During the tests, a resolution of 640×480 was used in the cameras, and Table 4.3 shows the measured camera positions and orientations.

Table 4.3: Camera measured positions used in the laboratory testing of the system.

Camera	x (m)	y (m)	z (m)	α (°)	β (°)	γ (°)
1	1.4665	-0.647	0.090	90	0	65.7
2	1.4665	0.647	0.090	90	0	114.3
3	-1.4665	0.647	0.090	90	0	245.7
4	1.4665	-0.647	0.090	90	0	294.3

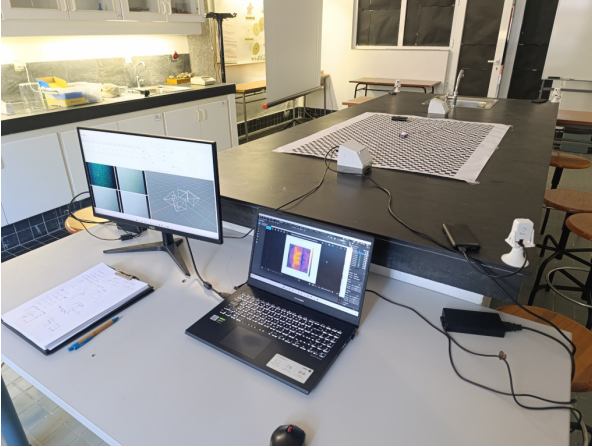


Figure 4.19: Laboratory set-up for the Distributed Camera System testing.



Figure 4.20: One of the cameras set up for live tests.

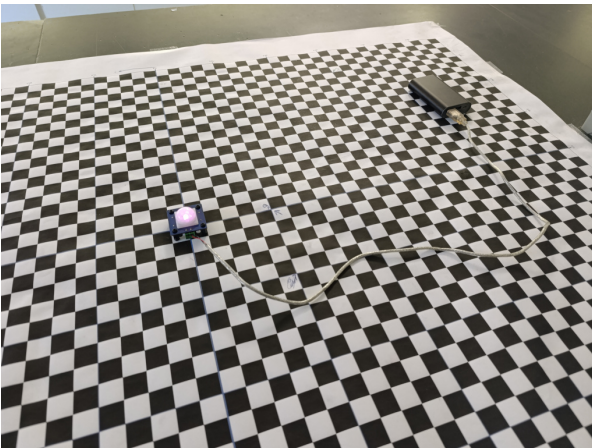


Figure 4.21: A0-sized checkerboard used as a reference for beacon coordinate measurement, to provide a ground truth for comparison with the output of the system.

4.4.2 Pattern for ground truth assessment

In order to have a procedure that could provide some form of ground truth information about the location of the beacon, it was decided to create a calibration sheet.

As a requirement, this sheet should be large enough to cover a considerable area. With this in mind, the sheet was designed as an A0 paper sheet.

Onto this sheet, a checkerboard pattern was drawn, with each square sized with a width of 2 cm, as seen in the scaled reproduction presented in Figure 4.22.

In the drawing, a reference system axis for the coordinates was also included, and a scale along the borders, to assist in the measurement of coordinates during the tests. Figure 4.23 shows the zoomed area marked by the bottom orange dashed line rectangle in Figure 4.22, and Figure 4.24 shows the central zoomed area indicated by the central orange dashed line rectangle. This image shows the circular mark pattern that was designed in the squares around the origin of the coordinate system as a way of visually determining the orientation of the reference system.

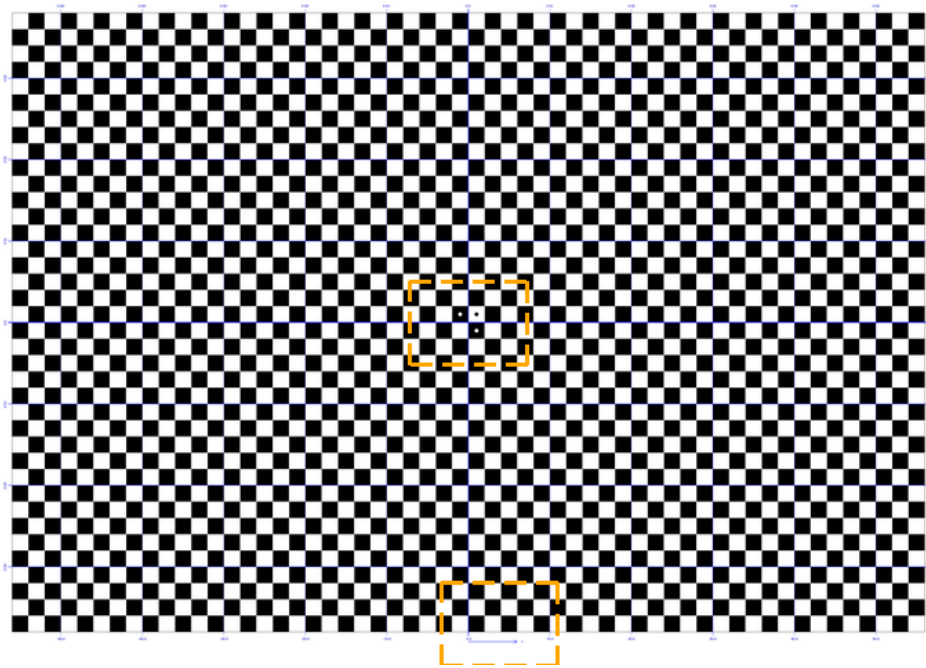


Figure 4.22: Measurements calibration paper sheet created to assist in the definition of a ground truth for the system.

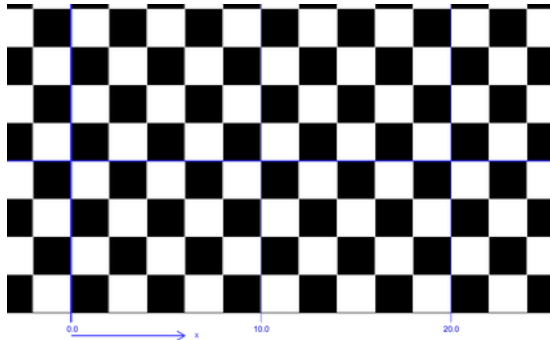


Figure 4.23: Calibration sheet zoomed in to show the indication of the axis defined and the scale (in cm).

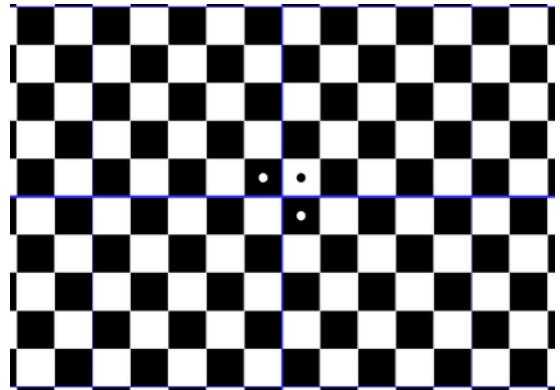


Figure 4.24: Calibration sheet zoomed in to show the indication of the coordinate system origin and some alignment markers, to help identify coordinate directions at a distance.

4.4.3 Accuracy and precision of the system

This section aims to present an estimation of the accuracy of the system and its precision. By accuracy, it is intended that the system is able to retrieve in as exact a way as possible, the real coordinates of the beacon. Precision, on the other hand, will measure how repeatable the measurements provided by the system.

Here, results from the analysis of one of the log files of the laboratory tests are presented. To gather the information regarding each point in a systematic way that would be easy to study after the test, the following approach was used:

1. Enable tracking on the software. This starts the tracking procedure, but also records continuously a log file of estimated positions. If the beacon is not detected by more than 1 camera, an entry with coordinates replaced by “nan” (not a number) was recorded in the log file for that timestamp.
2. With the beacon covered, it would be positioned in a given position chosen randomly over the calibration sheet. That position was then recorded in a table, and the beacon uncovered.
3. As soon as the beacon was uncovered, the software would detect it and record its estimated position in the log file. The software was allowed to measure the beacon over several frames, spanning a few seconds. That would generate in the log file a sequence of continuous positions recorded.
4. If this position was the last to be tested, the tracking function of the software was stopped. Otherwise, steps 2 and 3 would be repeated for another beacon position.

This procedure allowed a parsing of the log by separating the measured positions using the intervals with “nan” recorded in the file as separators. This procedure worked very well, producing 10 separate groups of measurements, consistent with the manually recorded positions of the beacon as it was moved. Table 4.4 gives the manually recorded positions of the beacon, used as a ground truth of the system. These measurements were made with a maximum error of ± 0.005 m. Figure 4.25 shows these positions over a grid, matching the coordinate system used in the calibration sheet.

Table 4.4: Test positions used for accuracy and precision testing of the system.

Position	x (m)	y (m)	z (m)
1	-0.100	0.000	0.020
2	0.220	0.100	0.020
3	-0.160	-0.160	0.020
4	0.380	0.080	0.142
5	-0.500	0.300	0.142
6	-0.120	-0.240	0.142
7	0.160	-0.140	0.020
8	-0.160	0.140	0.020
9	0.000	0.000	0.142
10	-0.420	-0.020	0.142

After the log file data separation into groups of measurements for each test position, statistics were used to estimate the mean position per coordinate of all tracking system individual position estimations for the respective test position. Also, the standard deviation of each coordinate. The standard deviation is used to establish a precision value for the system because it measures the repeatability of the system. For the estimation of accuracy, the Euclidean distance between the mean estimated position by the software and the real test position was used. Figures 4.26–4.35 present the data collected visually in the top plot, with the test position marked by a black cross in the centre of the image, the estimated position by the software in each frame as blue dots and the mean position as a red + sign. The lengths of the lines that compose the red + sign are of different sizes, because they are scaled according to the standard deviation of each coordinate. Below each one of the scatter plots, there are also histograms of the position estimations per coordinate.

Table 4.5 contains all the statistics calculated for these positions. For each position, it contains the real position measured in the calibration sheet (per coordinate), the mean estimated position by the algorithm (per coordinate) and the respective standard deviation, used to estimate system precision. The last columns contain the Accuracy value (or the absolute error between the real position and the estimated

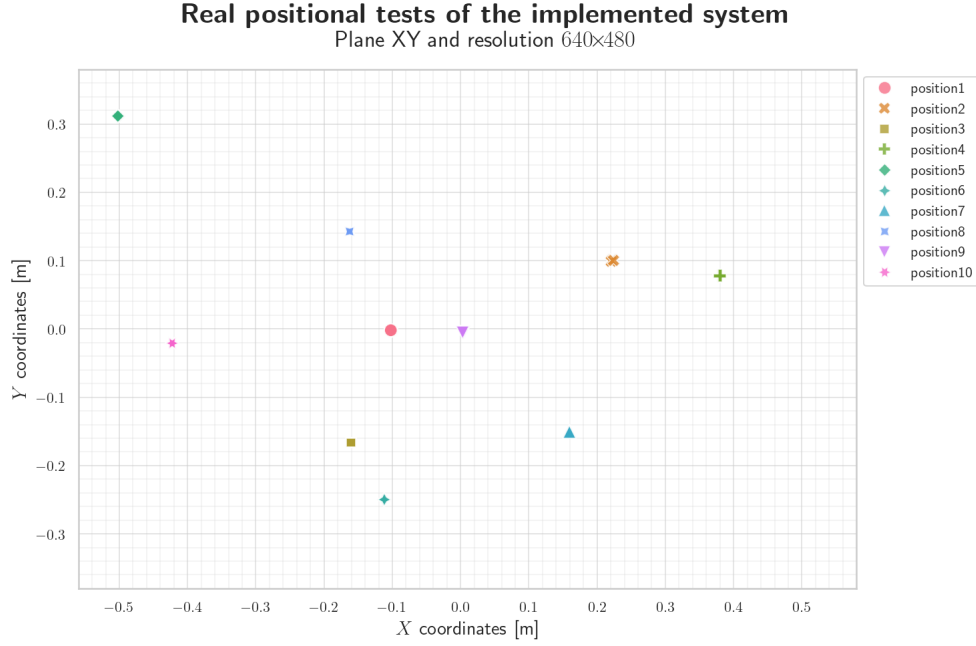


Figure 4.25: Positions used for accuracy and precision assessment of the system over the space defined by the pattern described above (see 4.4.2). Also shown with different marks and colours are the mean estimated positions by the system, with its radius scaled proportionally to 2 times its standard deviation.

position) and the Count of frames used in estimating these values.

The final line of the table gives global values, calculated as the means of the estimates. It was obtained that the precision in x was of $\sigma_x = 0.0003$ m, in y of $\sigma_y = 0.0002$ m and in z of $\sigma_z = 0.0001$ m. The overall accuracy was of $e_{abs} = 0.0069$ m, with a minimum value of $e_{abs}^{min} = 0.0028$ m at test position number 10, and a maximum error in accuracy of $e_{abs}^{max} = 0.0129$ m in test position number 6.

Table 4.5: Test positions tested in the live tests for accuracy and precision assessment. Each line contains the real position of the beacon, the estimated mean and standard deviation of the beacon position estimate, the number of points used (counts) and the estimated accuracy for each test

Test Position	Real [m]			Estimated Mean [m]			Estimated Std [m]			Accuracy [m]	Count
	x	y	z	x_{est}	y_{est}	z_{est}	σ_x	σ_y	σ_z	e_{abs}	
1	-0.1000	0.0000	0.0200	-0.1020	-0.0022	0.0226	0.0003	0.0002	0.0001	0.0039	678
2	0.2200	0.1000	0.0200	0.2224	0.1007	0.0232	0.0003	0.0002	0.0002	0.0041	671
3	-0.1600	-0.1600	0.0200	-0.1603	-0.1666	0.0235	0.0004	0.0003	0.0002	0.0075	810
4	0.3800	0.0800	0.1420	0.3796	0.0778	0.1451	0.0002	0.0002	0.0002	0.0038	518
5	-0.5000	0.3000	0.1420	-0.5017	0.3117	0.1399	0.0004	0.0002	0.0001	0.0120	471
6	-0.1200	-0.2400	0.1420	-0.1120	-0.2498	0.1444	0.0004	0.0002	0.0001	0.0129	246
7	0.1600	-0.1400	0.0200	0.1594	-0.1503	0.0254	0.0003	0.0002	0.0001	0.0116	447
8	-0.1600	0.1400	0.0200	-0.1622	0.1427	0.0215	0.0003	0.0002	0.0001	0.0038	211
9	0.0000	0.0000	0.1420	0.0036	-0.0050	0.1441	0.0003	0.0002	0.0001	0.0065	270
10	-0.4200	-0.0200	0.1420	-0.4227	-0.0208	0.1421	0.0003	0.0001	0.0001	0.0028	1046
Global Results							0.0003	0.0002	0.0001	0.0069	

These results provide evidence that the system is precise and that it can be accurate. Accuracy is still the worst result of this test, even if it only fails by a few mm. But this was caused most likely by the hand-measured position of cameras and, more importantly, by the fact that camera orientations were estimated, not measured. This could cause easily propagated errors in the position estimations, resulting in less accuracy of the measurements.

Accuracy and precision of the system

Position 1 measured with resolution 640x480

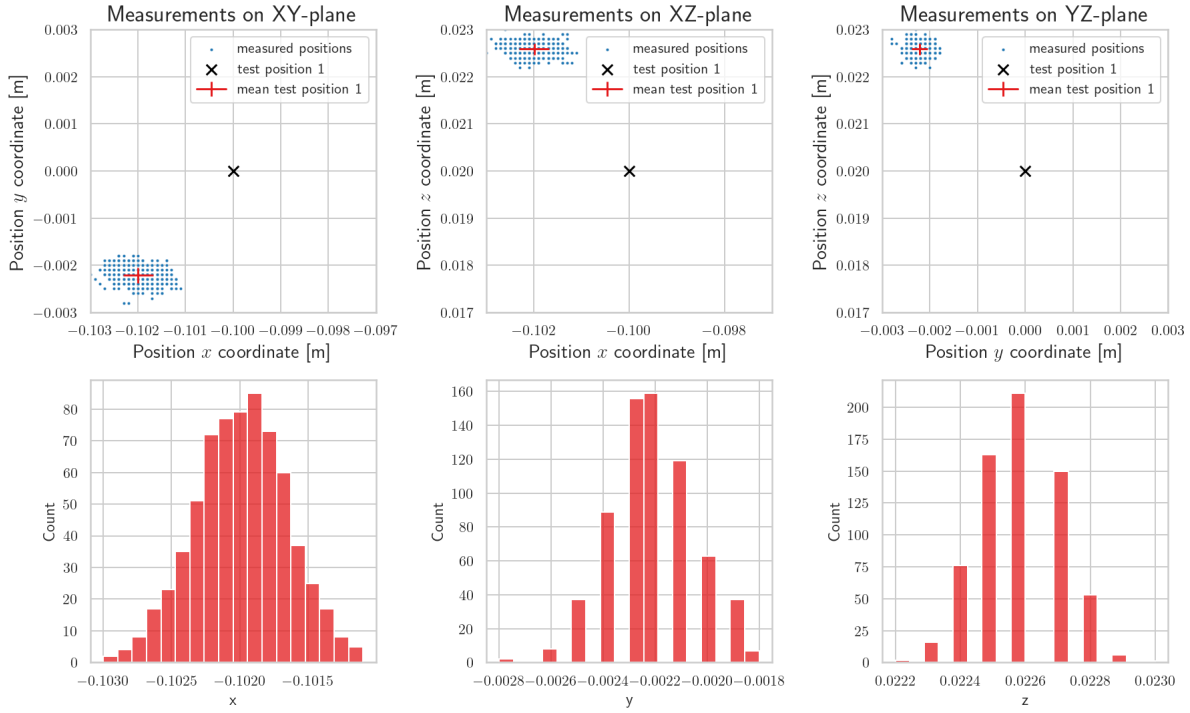


Figure 4.26: Top row: Position 1 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system

Position 2 measured with resolution 640x480

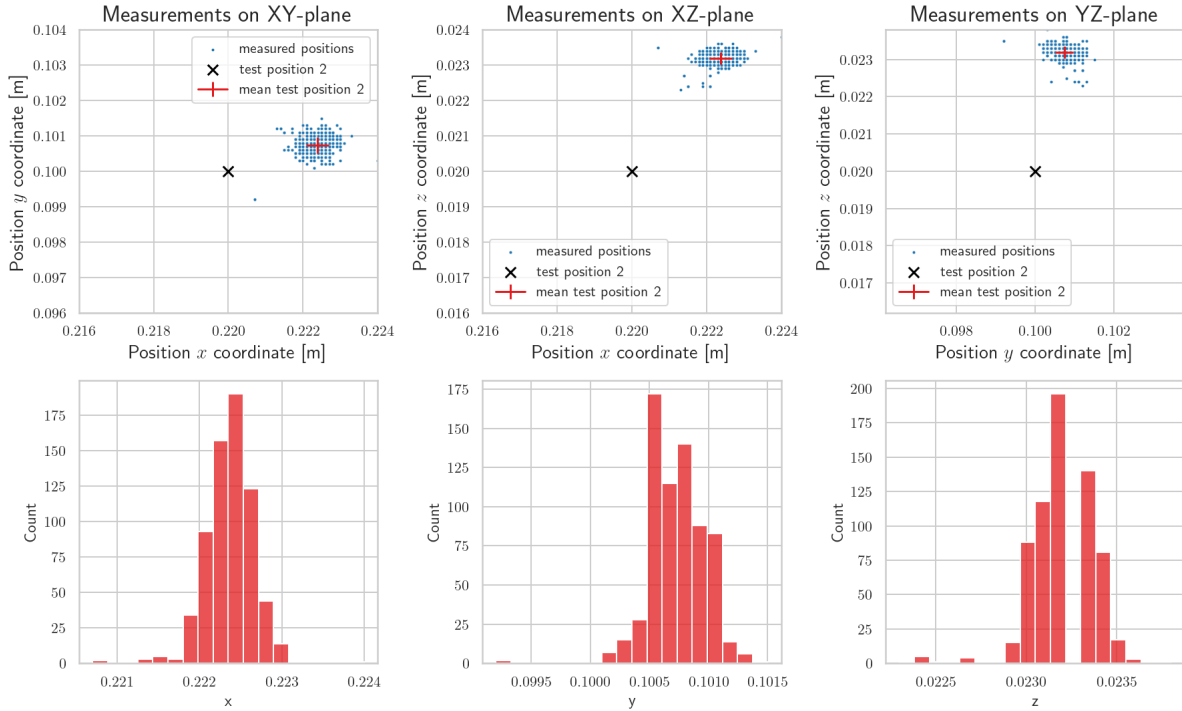


Figure 4.27: Top row: Position 2 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system Position 3 measured with resolution 640×480

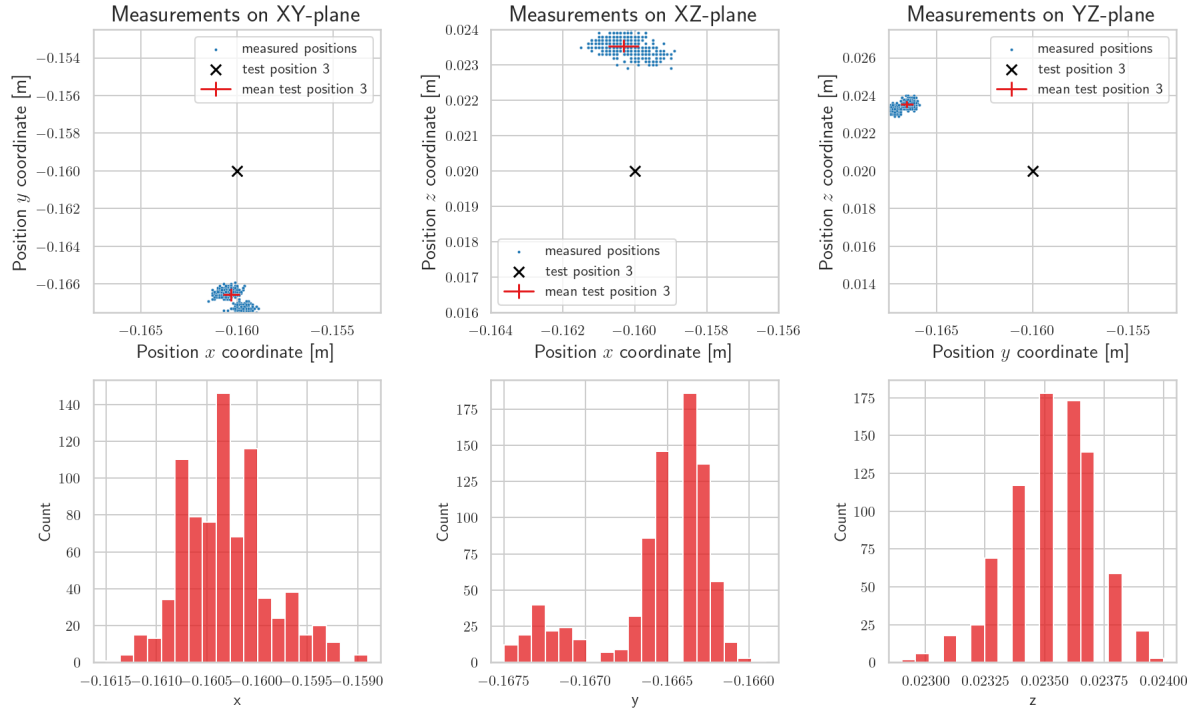


Figure 4.28: Top row: Position 3 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system Position 4 measured with resolution 640×480

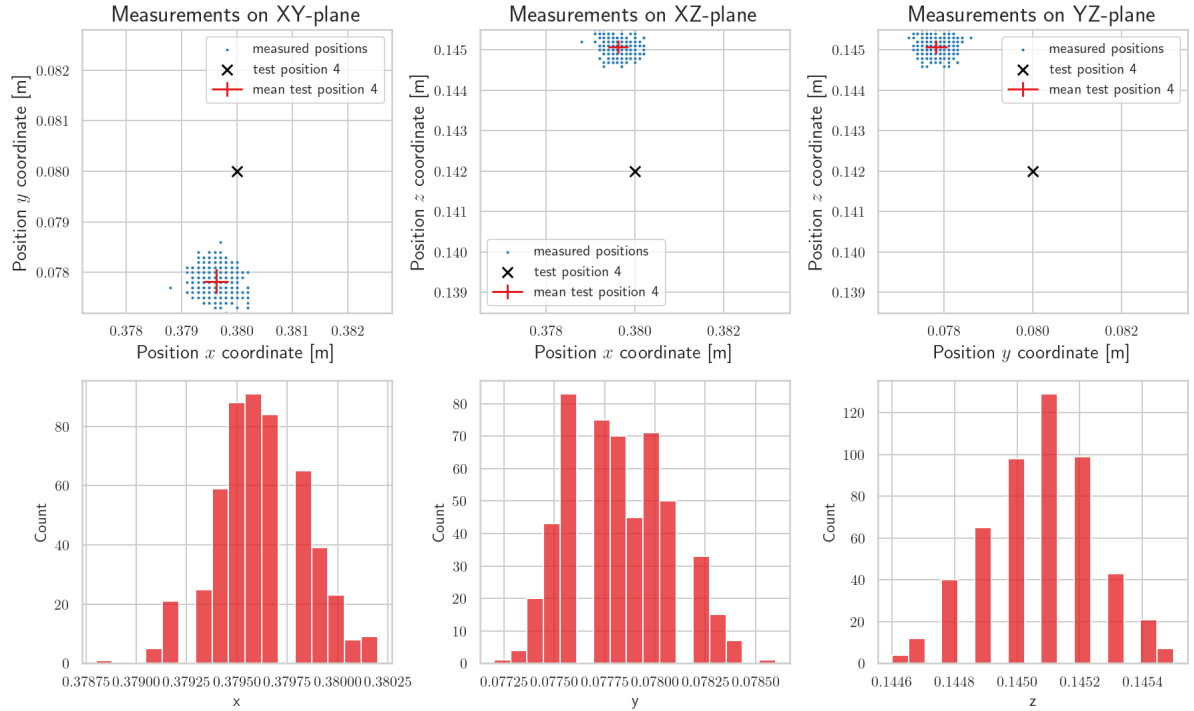


Figure 4.29: Top row: Position 4 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system

Position 5 measured with resolution 640x480

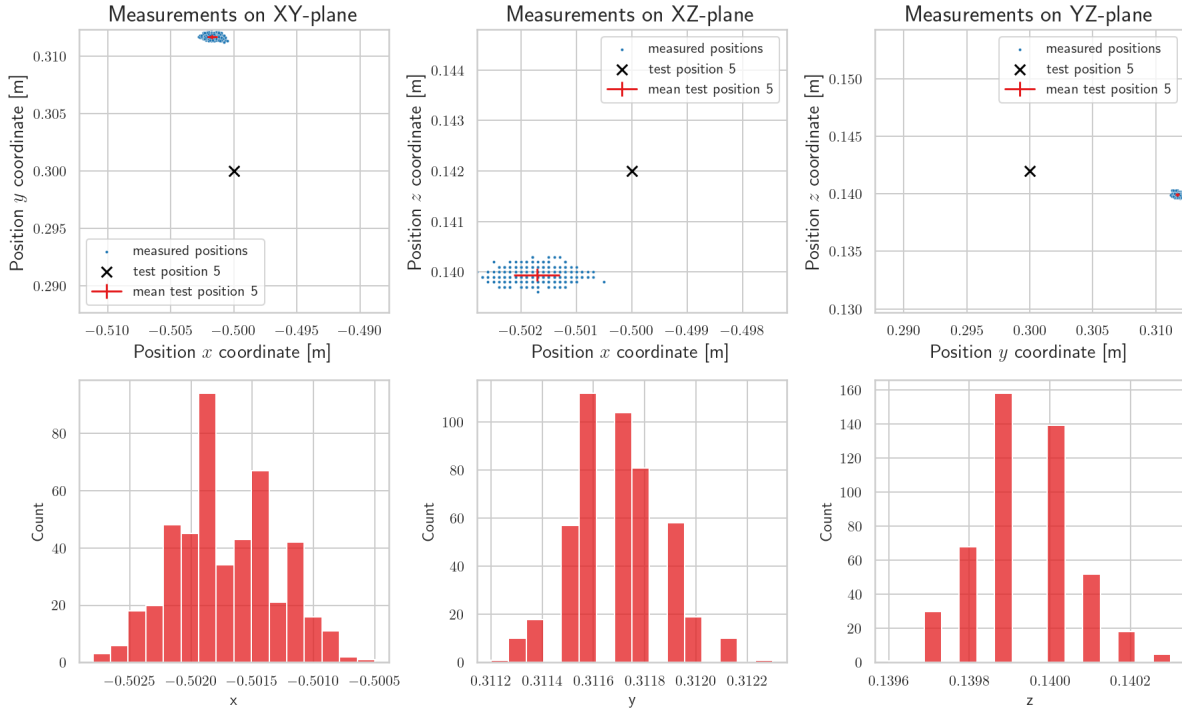


Figure 4.30: Top row: Position 5 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system

Position 6 measured with resolution 640x480

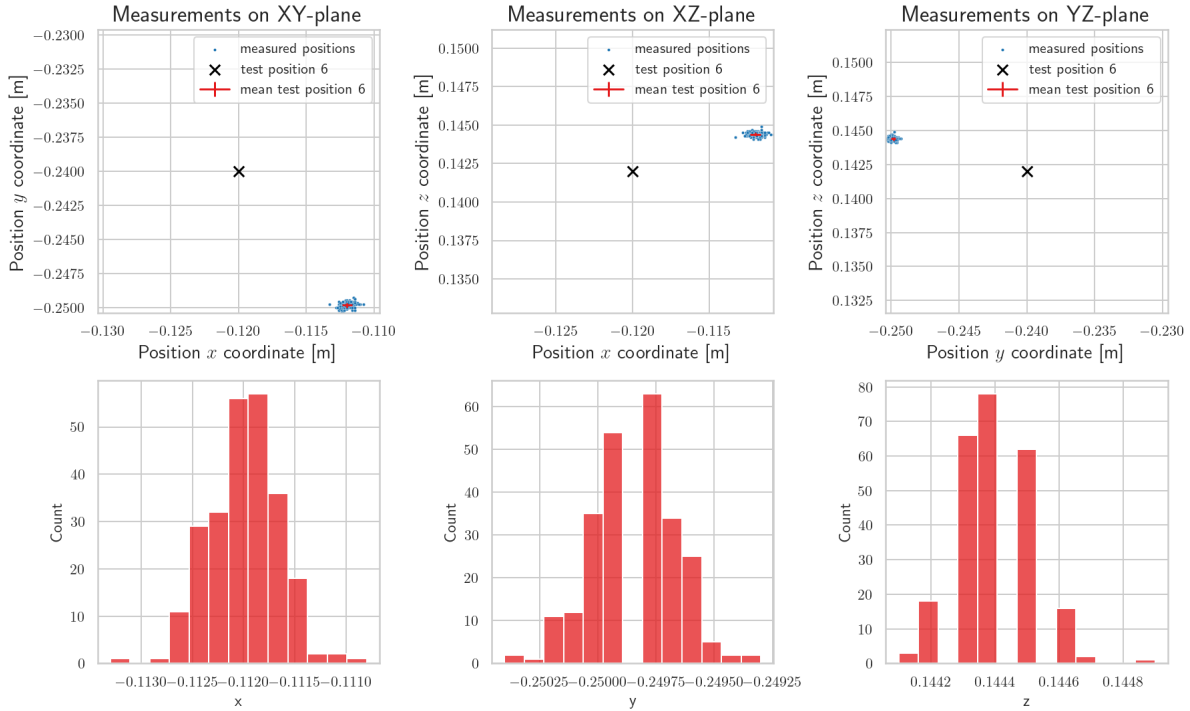


Figure 4.31: Top row: Position 6 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system

Position 7 measured with resolution 640×480

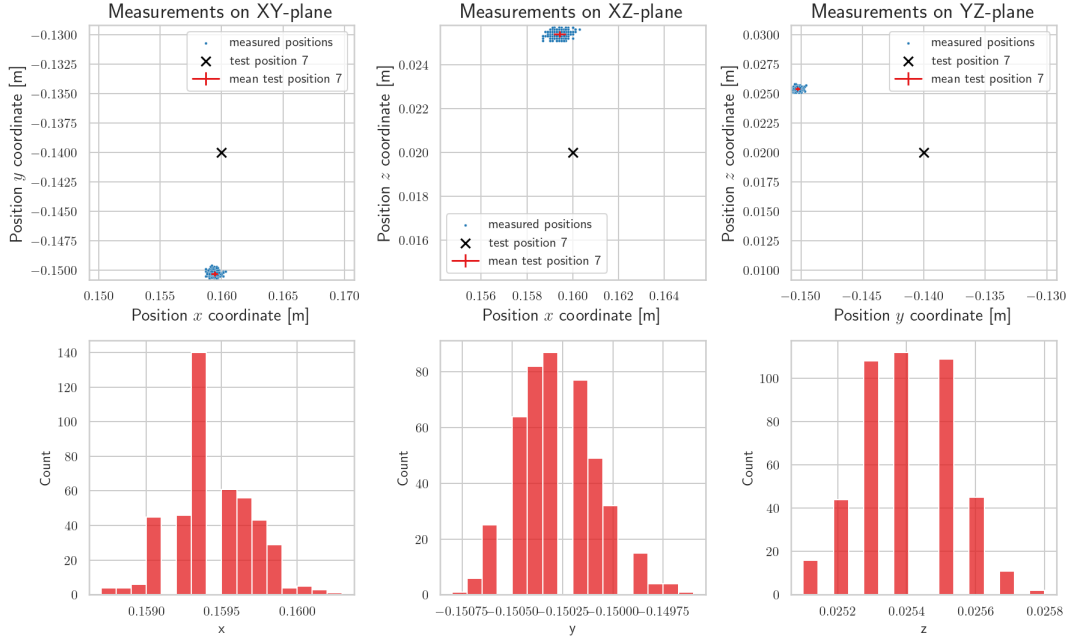


Figure 4.32: Top row: Position 7 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system

Position 8 measured with resolution 640×480

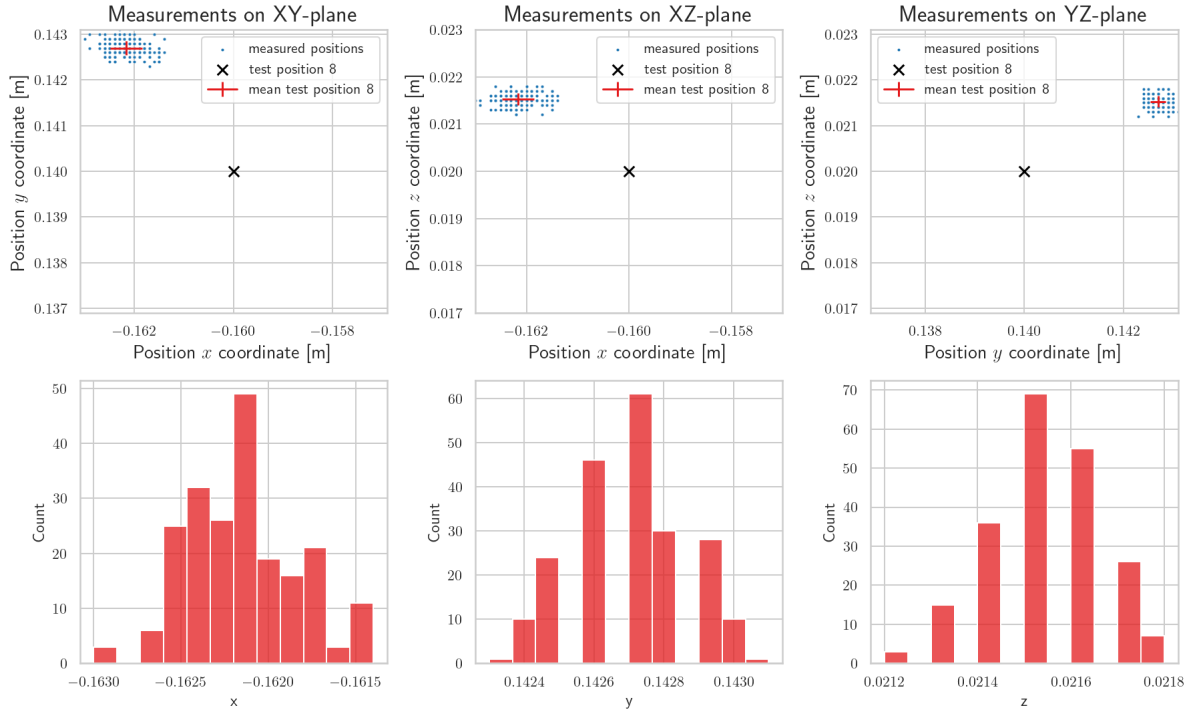


Figure 4.33: Top row: Position 8 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system

Position 9 measured with resolution 640x480

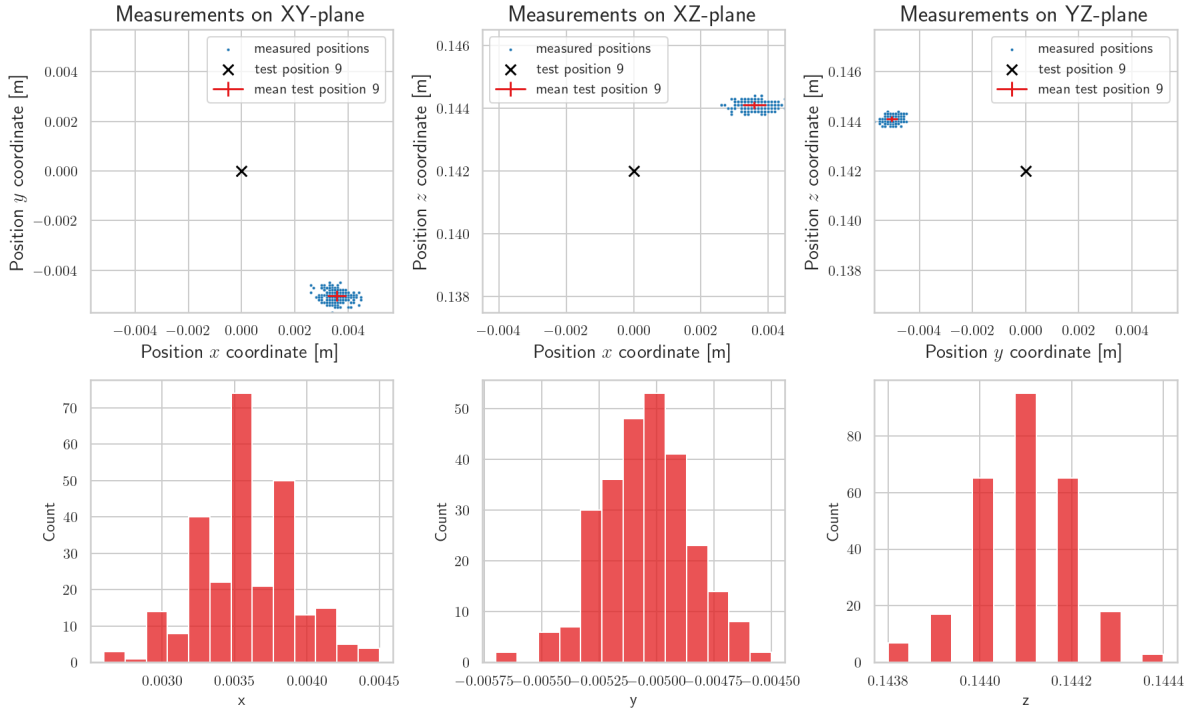


Figure 4.34: Top row: Position 9 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

Accuracy and precision of the system

Position 10 measured with resolution 640x480

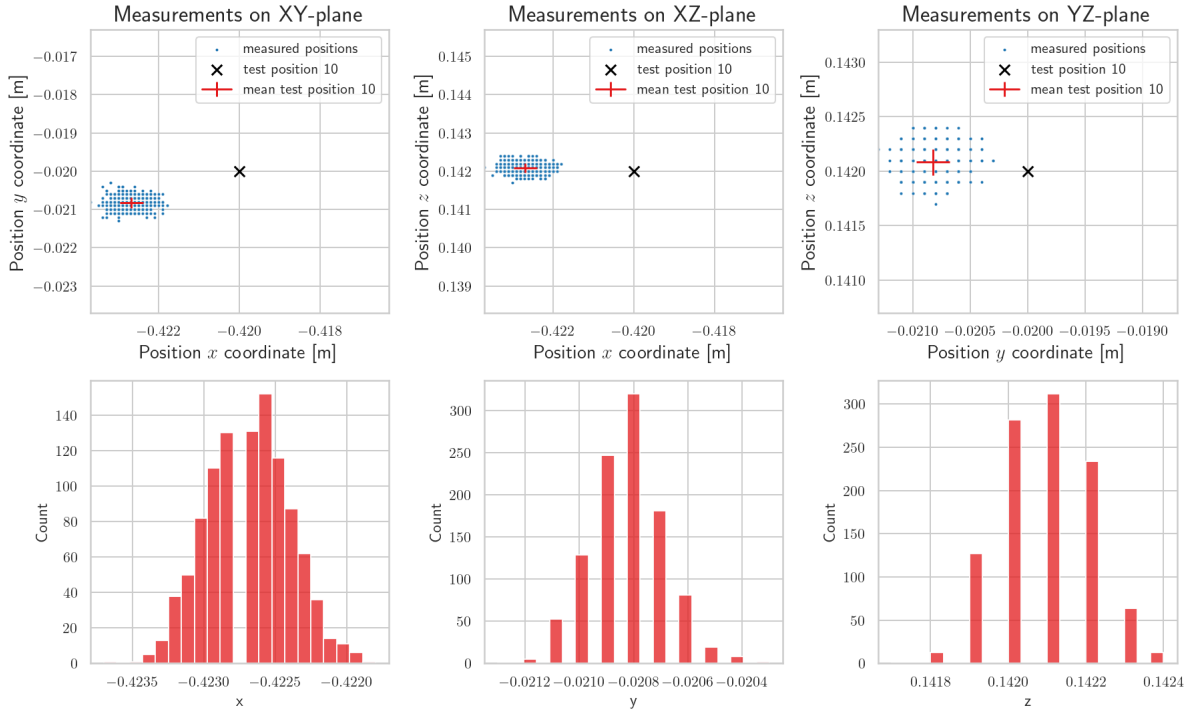


Figure 4.35: Top row: Position 10 estimated by the designed tracking system, using the DTCS software (blue dots), mean estimated position (red plus sign) and real position of beacon (black plus sign). Bottom row: Histogram of x (left), y (centre) and z (right) estimated coordinates distribution.

4.4.4 Sampling frequency of the system

Using the fact that the log files contained lines that started with a timestamp, it was possible to estimate the instantaneous sampling frequency f_s in Hz of the system by computing the inverse of the time difference between two consecutive measurement timestamps, according to the following expression.

$$f_s = \frac{1}{t_i - t_{i-1}} \quad (4.1)$$

where t_i is the timestamp of frame i and t_{i-1} the timestamp of the previous frame $i - 1$.

Figure 4.36 presents the estimation of the sampling frequency as calculated using (4.1) for the whole test in the top plot. The gaps between estimations correspond to the times when the beacon was covered (to be moved to another test position). From these 10 test positions, 4 were randomly chosen to provide a zoomed view of that specific test position estimation. These results are presented for positions 1, 2, 5 and 10.

From this data, it is possible to estimate a mean sampling frequency \bar{f}_s for the system, which resulted in $\bar{f}_s = 13.80$ Hz.

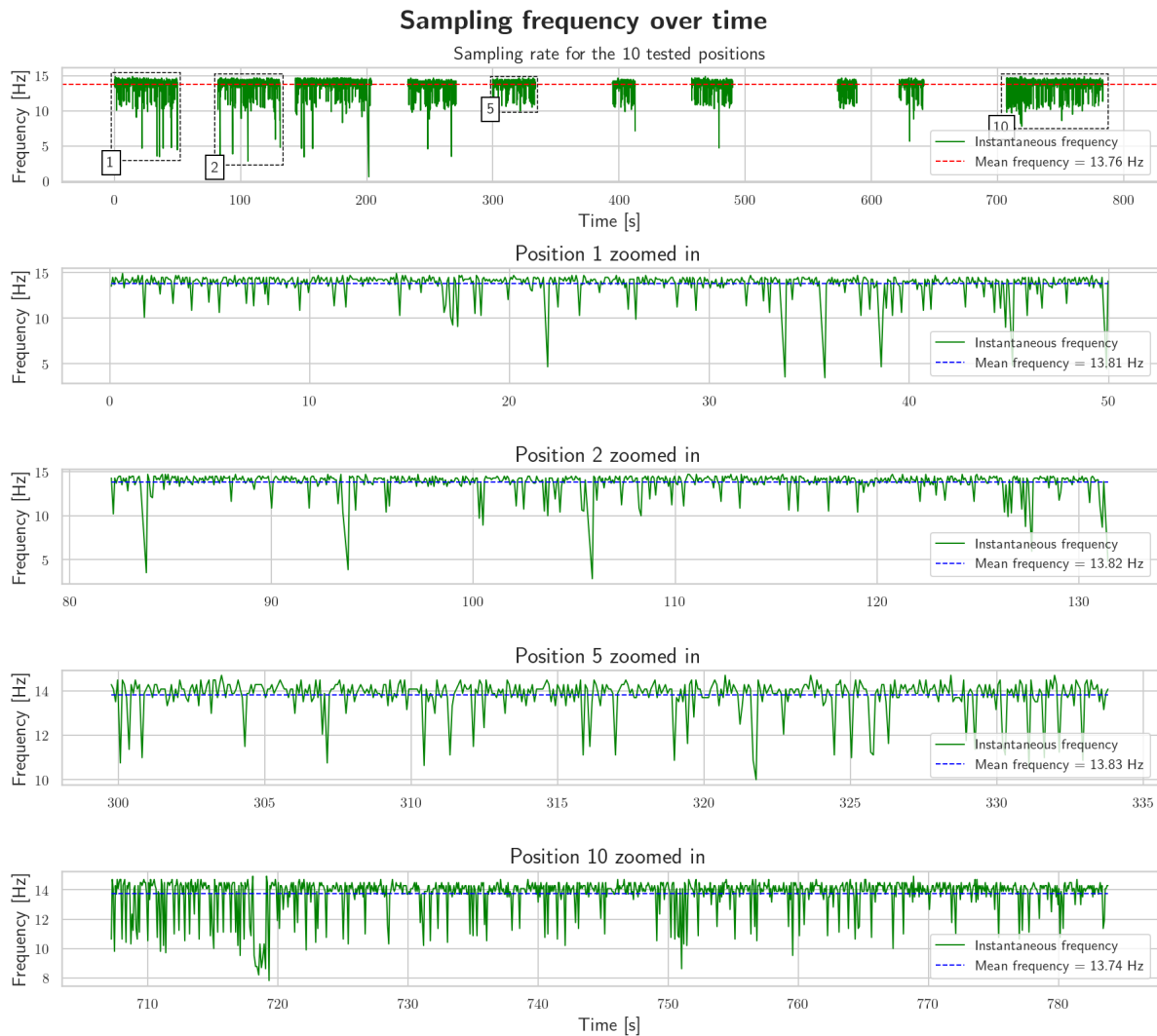


Figure 4.36: Framerate of the system during live test. From top to bottom are presented the framerate for all tests and zoomed-in plots of framerate for test positions 1, 2, 5 and 10, respectively.

Chapter 5

Conclusion

5.1 System analysis and applicability

The proposed visual tracking seems to be viable for production using low-cost cameras and a micro-controller, with the estimation algorithm being run over a computer with access to the camera's video feed.

Simulations have shown that if the orientations and positions of cameras are precisely known, the system can recover the position of an object within a few millimetres with a low resolution of 320×240. When going to a higher resolution, the absolute error of the system decreases to a few tenths of a millimetre, depending on the camera's distribution.

The analysis of those simulations has shown that 2 cameras are the least number that must be used to track a beacon, but with a high probability of large absolute errors in the position estimation. With 3 cameras detecting the beacon, the system still produces some error in the position estimation, but already with a much higher accuracy. Still, there is some probability of having errors within 10 cm. With 4 or more cameras, the absolute error of the system reduces to a few millimetres in the worst-case scenario.

The practical tests of the DTCS system proposed in this work, done in the University of Évora Physics Laboratories, have shown that the system can track the beacon with a precision of within 0.003 mm, with the tests not providing good enough data to assess the accuracy, mainly due to poor camera orientation determination.

Given all the above, depending on the application, this may be a suitable methodology to track a robotic system and provide it with an external position estimation, assisting its own navigation and path-finding algorithms, and at the same time, providing an alternative to GPS indoors. It also indicates that it could be a replacement for a more accurate, but also much more expensive system, like Vicon or OptiTrack.

5.2 System Current Limitations

Current limitations of this project involve mainly that the position of cameras in relation to a user-defined reference frame can be achieved with some accuracy by manual methods; however, the orientation of the cameras is very difficult to accurately measure. Because of this, inaccurate measurements of camera angles produce inaccurate object position estimations.

Another limitation is that in its current state, the software is only able to detect one beacon at a time and is restricted to IR beacons. It would probably be desirable to have the capacity to detect more than one beacon at a time.

Furthermore, the system is also susceptible to ambient light conditions (like other visual systems such as Vicon or OptiTrack), requiring, for now, an illumination-controlled setting.

5.3 Future Work

As future work, one important feature to add to the software will be the ability of tracking more than one beacon at a time, and also the capacity to distinguish them apart. This will enable the tracking of not only the position of a system, but also the orientation, if the relative positions between beacons (on the robotic system reference frame) are known *a priori*.

Another important feature to add is the capacity for auto-calibration of camera poses. This work was already ongoing at the time of the dissertation writing. This will enable the recovery of the camera poses accurately, by using beacons for which positions in real-world reference are also known by the user.

One evolution of the system would be to recognise coloured high-power LEDs, with different colours, instead of IR beacons. This would enable the recovery of the robotic system's pose, given that the detection of those different LEDs was successful.

It could even be incorporated into the system a Convolutional Neural Network, trained to recognise the object, instead of the beacon, that would generate image coordinates for several keypoints of the robotic system being tracked and outputting its image coordinates, just like if they were beacons, and from those coordinates extract the robot pose in the world reference frame.

Bibliography

- [1] V. Verma, M. W. Maimone, D. M. Gaines, R. Francis, T. A. Estlin, S. R. Kuhn, G. R. Rabideau, S. A. Chien, M. M. McHenry, E. J. Graser, A. L. Rankin, and E. R. Thiel, “Autonomous robotics is driving perseverance rover’s progress on mars,” *Science Robotics*, vol. 8, no. 80, p. eadi3099, 2023. [Online]. Available: <https://www.science.org/doi/abs/10.1126/scirobotics.adi3099>
- [2] A. Waga, S. Benhlima, A. Bekri, J. Abdouni, and F. Z. Saber, “A survey on autonomous navigation for mobile robots: From traditional techniques to deep learning and large language models,” vol. 37, no. 7, p. 198. [Online]. Available: <https://doi.org/10.1007/s44443-025-00216-x>
- [3] K. Katona, H. A. Neamah, and P. Korondi, “Obstacle avoidance and path planning methods for autonomous navigation of mobile robot,” *Sensors*, vol. 24, no. 11, 2024. [Online]. Available: <https://www.mdpi.com/1424-8220/24/11/3573>
- [4] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, “Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam,” *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [5] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, “Event-based vision: A survey,” 04 2019.
- [6] E. A. Rodríguez-Martínez, W. Flores-Fuentes, F. Achakir, O. Sergiyenko, and F. N. Murrieta-Rico, “Vision-based navigation and perception for autonomous robots: Sensors, slam, control strategies, and cross-domain applications—a review,” *Eng*, vol. 6, no. 7, 2025. [Online]. Available: <https://www.mdpi.com/2673-4117/6/7/153>
- [7] J. Wang, Z. Yu, D. Zhou, J. Shi, and R. Deng, “Vision-based deep reinforcement learning of unmanned aerial vehicle (uav) autonomous navigation using privileged information,” *Drones*, vol. 8, no. 12, 2024. [Online]. Available: <https://www.mdpi.com/2504-446X/8/12/782>
- [8] “Marvelmind indoor navigation system,” 2020, acedido em setembro de 2025. [Online]. Available: <https://marvelmind.com/>
- [9] “Vicon motion systems,” 2017, acedido em setembro de 2025. [Online]. Available: <https://www.vicon.com/>
- [10] “Optitrack — motion capture systems,” <https://optitrack.com/>, NaturalPoint, Inc., 2025, acedido em Setembro de 2025.
- [11] D. Roorda and F. van der Helm, “Optical motion capture: Theory and applications,” *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 5, pp. 1235–1245, 2019.
- [12] M. Yavari, “Indoor real-time positioning using ultra-wideband technology,” 2015, com avaliação em LOS e NLOS, fusão com IMU.
- [13] M. F. R. Al-Okby, S. Junginger, T. Roddelkopf, and K. Thurow, “Uwb-based real-time indoor positioning systems: A comprehensive review,” *Applied Sciences*, vol. 14, no. 23, p. 11005, 2024.
- [14] M. F. B. e. a. Amirmudin, “Design and evaluation of ultra wideband for localization of mobile robots,” 2022, 20 cm em NLOS, 10-20 cm em LOS.
- [15] M. Ziegler, A. E. Kianfar, T. Hartmann, and E. Clausen, “Development and evaluation of a uwb-based indoor positioning system for underground mine environments,” vol. 40, no. 4, pp. 1021–1040. [Online]. Available: <https://doi.org/10.1007/s42461-023-00797-z>

- [16] P. Merriaux, Y. Dupuis, R. Bouteau, P. Vasseur, and X. Savatier, "A study of vicon system positioning performance," *Sensors*, vol. 17, no. 7, p. 1591, 2017.
- [17] OpenCV. (2025) Camera calibration and 3d reconstruction. [Online]. Available: https://docs.opencv.org/4.x/d9/d0c/group_calib3d.html
- [18] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. USA: Prentice Hall PTR, 1998.
- [19] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, Í. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.
- [20] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," in *International journal of computer vision*, vol. 60, no. 2. Springer, 2004, pp. 91–110.
- [21] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: <http://www.blender.org>
- [22] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.
- [23] I. OmniVision Technologies, "Ov2640 color cmos uxga (2.0 megapixel) camerachiptm with image array and image processor," https://www.uctronics.com/download/OV2640_DS.pdf, 2024, accessed: 2024-07-09.
- [24] hjmediastudios, "C-130 transport," <http://www.blendswap.com/blends/view/67155>, 2013.
- [25] Blend Swap. (2025) Blend swap - free blender 3d models. Online community for sharing Creative Commons Blender models. [Online]. Available: <https://www.blendswap.com>
- [26] E. W. Weisstein, "Lemniscate," <https://mathworld.wolfram.com/Lemniscate.html>, 2001, from MathWorld—A Wolfram Resource. Accessed: 2025-07-22.
- [27] W. Weibull, "A statistical distribution function of wide applicability," *Journal of Applied Mechanics*, vol. 18, no. 3, pp. 293–297, 1951.
- [28] J. F. Lawless, *Statistical Models and Methods for Lifetime Data*, 2nd ed. New York: Wiley-Interscience, 2003.
- [29] Espressif Systems, "ESP32-CAM Development Board Datasheet," <https://media.digikey.com/pdf/Data%20Sheets/DFRobot%20PDFs/DFR0602-Web.pdf>, Apr 2018, accessed: 2025-07-22.
- [30] R. C. Limited, *PyQt5 Reference Guide*, 2016, version 5. [Online]. Available: <https://www.riverbankcomputing.com/static/Docs/PyQt5/>

Appendix A

Systematic Simulations Datasets

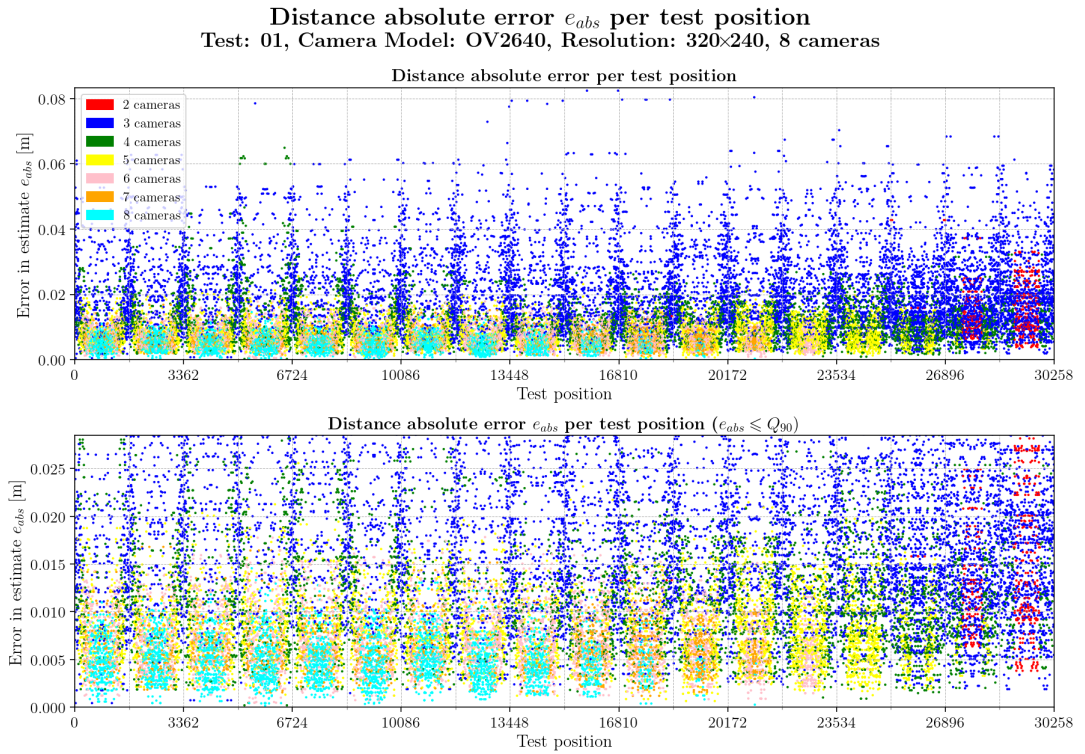
This appendix presents all the datasets generated for the systematic simulations. Each section is related to one of the five simulated camera distribution scenarios, and data for all five camera resolutions simulated, namely 320×240, 640×480, 800×600, 1024×768, 1600×1200 are presented in tables and graphics.

The interpretation of the graphics and tables is the same as discussed in section 3.5.

A.1 Test 01

Absolute error per iteration and resolution

Resolution: 320×240



Error per iteration for resolution 320×240 and test01

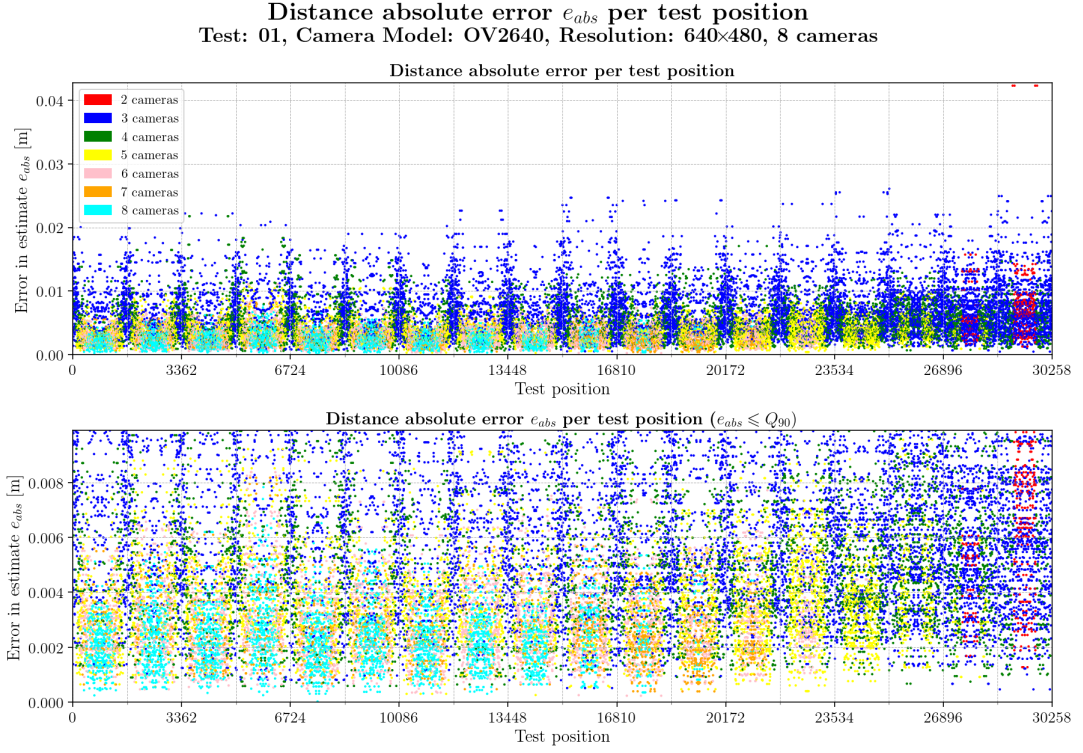
Table A.1: Statistics per number of cameras observing the beacon for test 01 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	420	11400	4340	4440	3808	1848	3897
mean	0.0154	0.0220	0.0122	0.0083	0.0072	0.0062	0.0050
std	0.0079	0.0121	0.0070	0.0038	0.0032	0.0026	0.0023
min	0.0038	0.0004	0.0002	0.0006	0.0005	0.0012	0.0002
25%	0.0095	0.0131	0.0076	0.0055	0.0048	0.0044	0.0033
50%	0.0131	0.0193	0.0110	0.0079	0.0068	0.0059	0.0048
75%	0.0208	0.0285	0.0152	0.0107	0.0094	0.0077	0.0064
max	0.0427	0.0824	0.0649	0.0256	0.0178	0.0168	0.0146

Table A.2: Global statistics and root mean square error for test 01 with resolution 320×240.

	Abs. error	Est. error
count	30153	30153
mean	0.0135	0.0235
std	0.0108	0.0093
min	0.0002	0.0000
25%	0.0060	0.0171
50%	0.0099	0.0230
75%	0.0175	0.0293
90%	0.0285	0.0357
max	0.0824	0.0629
RMSE	0.0173	

Resolution: 640×480



Error per iteration for resolution 640×480 and test01

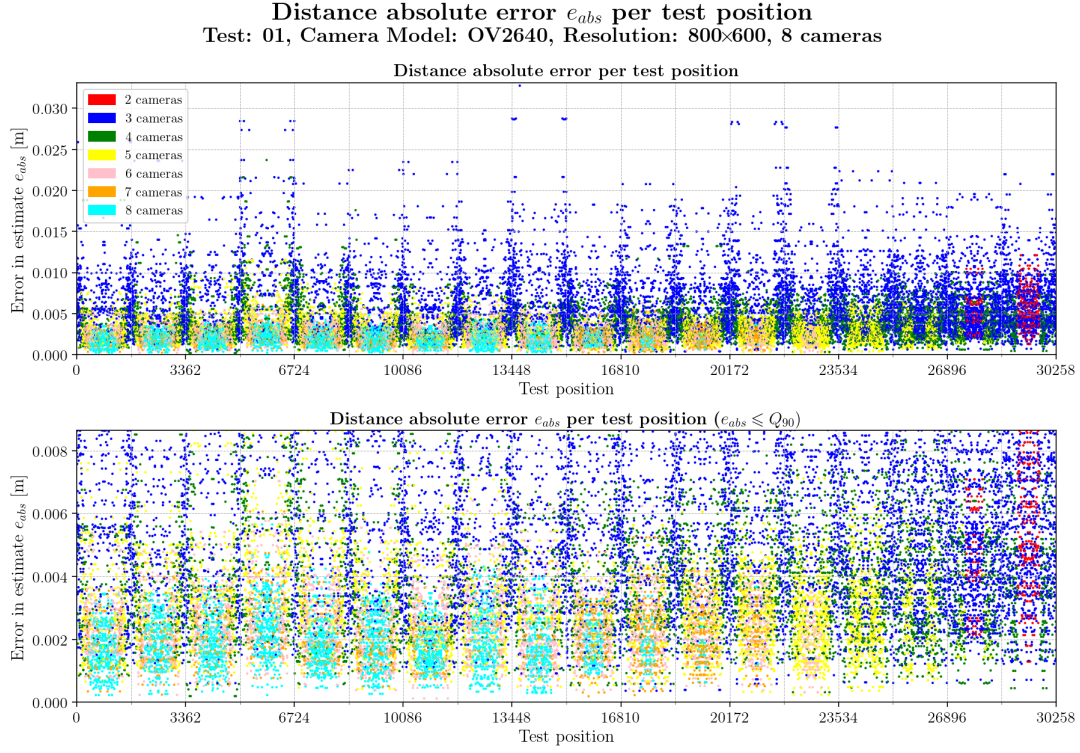
Table A.3: Statistics per number of cameras observing the beacon for test 01 with resolution 640×480.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	360	10896	4836	4540	3840	1784	3969
mean	0.0073	0.0076	0.0054	0.0037	0.0030	0.0027	0.0024
std	0.0050	0.0040	0.0029	0.0017	0.0014	0.0012	0.0011
min	0.0013	0.0005	0.0006	0.0003	0.0002	0.0004	0.0000
25%	0.0041	0.0046	0.0033	0.0025	0.0020	0.0018	0.0016
50%	0.0066	0.0069	0.0049	0.0035	0.0029	0.0025	0.0023
75%	0.0088	0.0097	0.0068	0.0046	0.0038	0.0034	0.0030
max	0.0423	0.0261	0.0218	0.0107	0.0111	0.0082	0.0078

Table A.4: Global statistics and root mean square error for test 01 with resolution 640×480.

	Abs. error	Est. error
count	30225	30225
mean	0.0051	0.0095
std	0.0036	0.0035
min	0.0000	0.0001
25%	0.0026	0.0070
50%	0.0041	0.0095
75%	0.0067	0.0120
90%	0.0099	0.0141
max	0.0423	0.0225
RMSE	0.0063	

Resolution: 800×600



Error per iteration for resolution 800×600 and test01

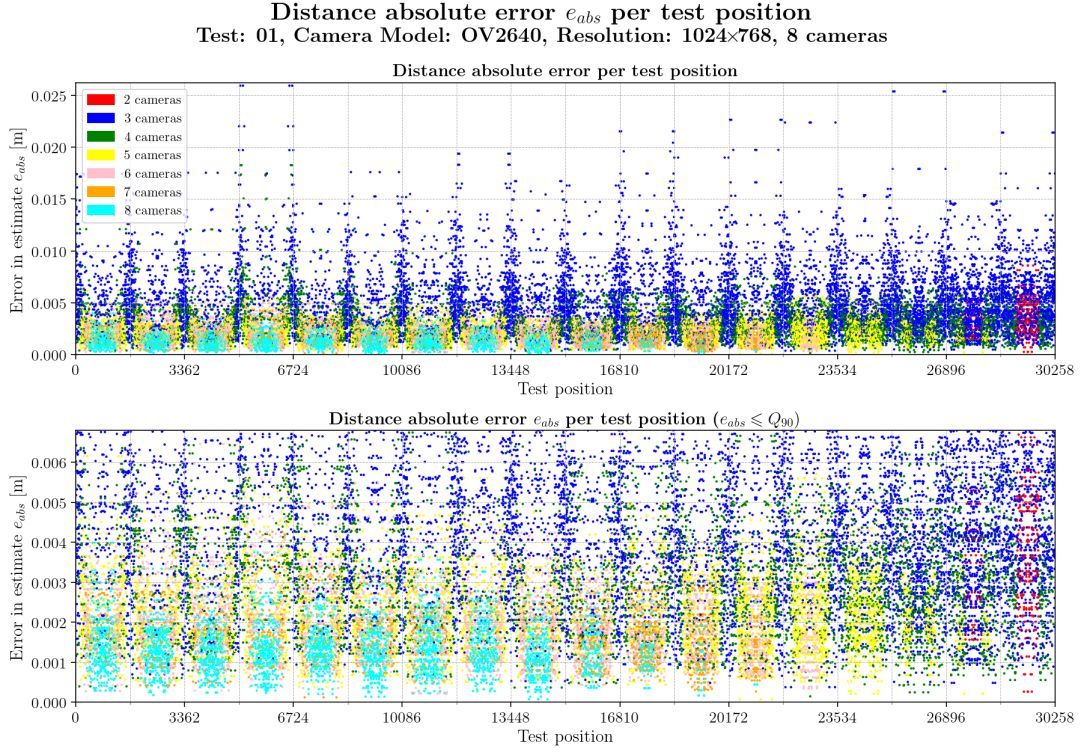
Table A.5: Statistics per number of cameras observing the beacon for test 01 with resolution 800×600.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	356	10876	4860	4540	3840	1784	3969
mean	0.0057	0.0068	0.0042	0.0031	0.0025	0.0021	0.0019
std	0.0023	0.0040	0.0023	0.0015	0.0011	0.0009	0.0008
min	0.0013	0.0004	0.0002	0.0002	0.0001	0.0002	0.0002
25%	0.0041	0.0039	0.0025	0.0020	0.0017	0.0015	0.0013
50%	0.0053	0.0059	0.0038	0.0029	0.0024	0.0020	0.0018
75%	0.0071	0.0087	0.0053	0.0039	0.0032	0.0027	0.0024
max	0.0121	0.0327	0.0237	0.0115	0.0070	0.0061	0.0057

Table A.6: Global statistics and root mean square error for test 01 with resolution 800×600.

	Abs. error	Est. error
count	30225	30225
mean	0.0044	0.0078
std	0.0034	0.0029
min	0.0001	0.0000
25%	0.0021	0.0059
50%	0.0033	0.0078
75%	0.0055	0.0097
90%	0.0086	0.0116
max	0.0327	0.0183
RMSE	0.0055	

Resolution: 1024×768



Error per iteration for resolution 1024×768 and test01

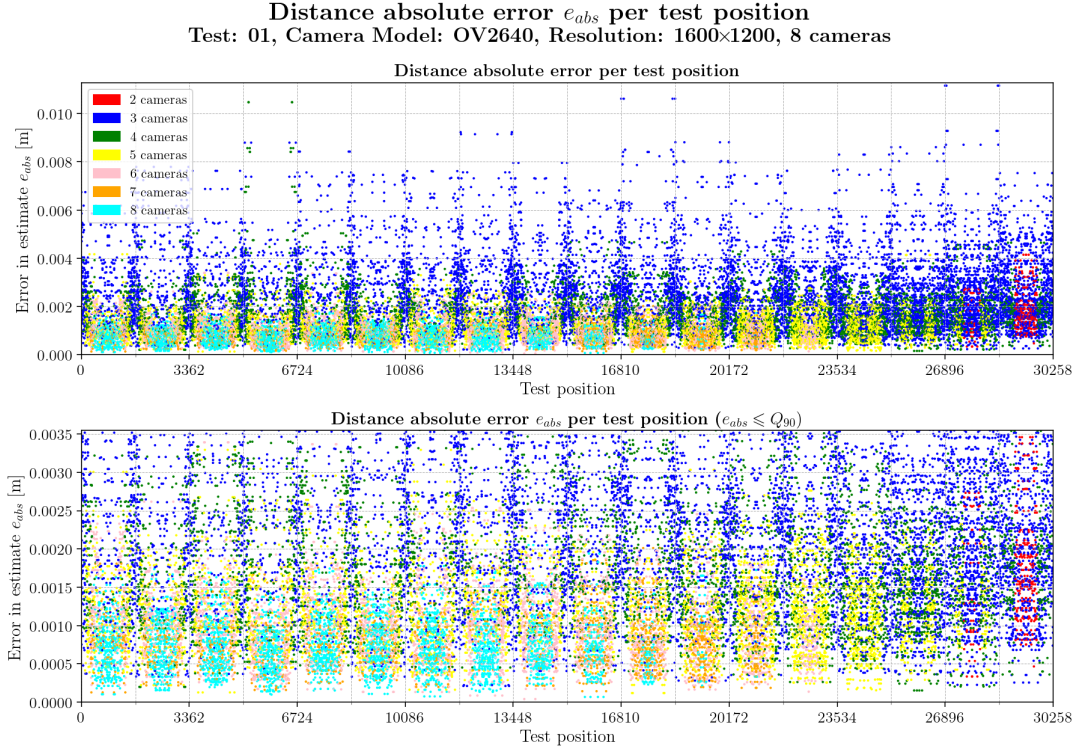
Table A.7: Statistics per number of cameras observing the beacon for test 01 with resolution 1024×768.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	320	10888	4844	4580	3840	1784	3969
mean	0.0037	0.0056	0.0033	0.0023	0.0020	0.0017	0.0014
std	0.0016	0.0032	0.0018	0.0010	0.0009	0.0007	0.0006
min	0.0003	0.0004	0.0001	0.0001	0.0002	0.0001	0.0001
25%	0.0025	0.0034	0.0021	0.0016	0.0014	0.0012	0.0009
50%	0.0036	0.0049	0.0030	0.0022	0.0019	0.0017	0.0013
75%	0.0049	0.0070	0.0041	0.0030	0.0026	0.0021	0.0017
max	0.0086	0.0259	0.0183	0.0062	0.0065	0.0043	0.0040

Table A.8: Global statistics and root mean square error for test 01 with resolution 1024×768.

	Abs. error	Est. error
count	30225	30225
mean	0.0035	0.0061
std	0.0027	0.0023
min	0.0001	0.0000
25%	0.0016	0.0046
50%	0.0027	0.0061
75%	0.0044	0.0076
90%	0.0068	0.0090
max	0.0259	0.0167
RMSE	0.0044	

Resolution: 1600×1200



Error per iteration for resolution 1600×1200 and test01

Table A.9: Statistics per number of cameras observing the beacon for test 01 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	320	10888	4820	4552	3892	1784	3969
mean	0.0018	0.0028	0.0018	0.0012	0.0010	0.0008	0.0007
std	0.0008	0.0016	0.0009	0.0005	0.0005	0.0003	0.0003
min	0.0003	0.0002	0.0002	0.0001	0.0000	0.0001	0.0001
25%	0.0012	0.0017	0.0012	0.0009	0.0007	0.0005	0.0005
50%	0.0016	0.0025	0.0016	0.0012	0.0010	0.0008	0.0007
75%	0.0022	0.0036	0.0023	0.0015	0.0013	0.0010	0.0009
max	0.0041	0.0112	0.0105	0.0042	0.0034	0.0020	0.0021

Table A.10: Global statistics and root mean square error for test 01 with resolution 1600×1200.

	Abs. error	Est. error
count	30225	30225
mean	0.0018	0.0032
std	0.0013	0.0012
min	0.0000	0.0000
25%	0.0009	0.0024
50%	0.0014	0.0032
75%	0.0023	0.0040
90%	0.0035	0.0048
max	0.0112	0.0095
RMSE	0.0022	

Error and camera coverage maps

Error maps for resolution 320×240

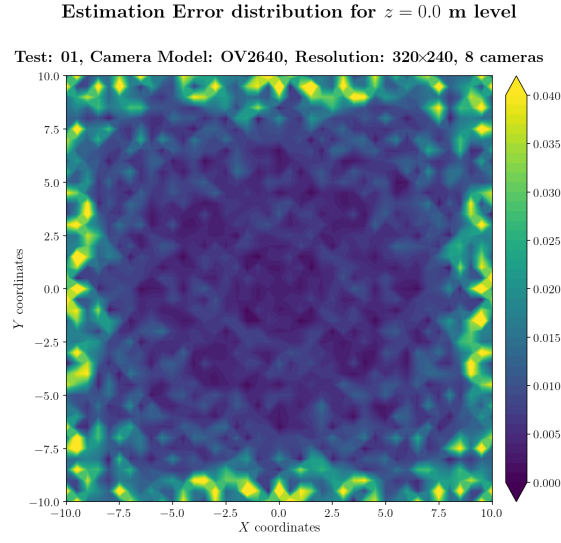


Figure A.1: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.000$ m.

Camera Coverage Spatial Distribution for $z = 0.0$ m level

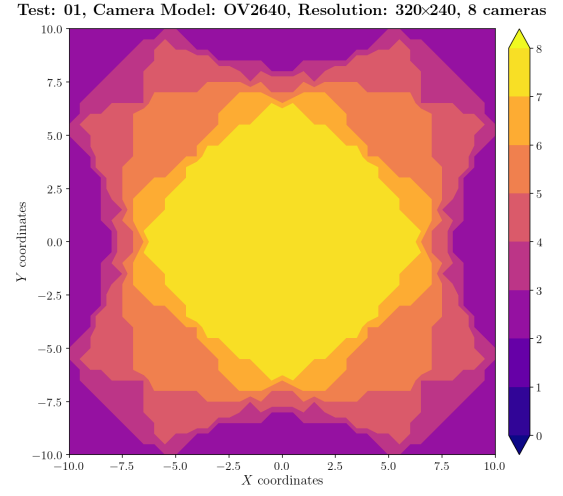


Figure A.2: Camera coverage for test 01 at level $z = 0.000$ m and resolution 320×240.

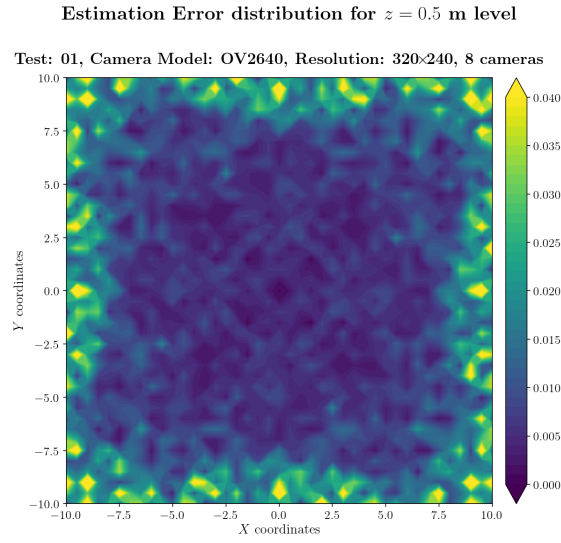


Figure A.3: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.500$ m.

Camera Coverage Spatial Distribution for $z = 0.5$ m level

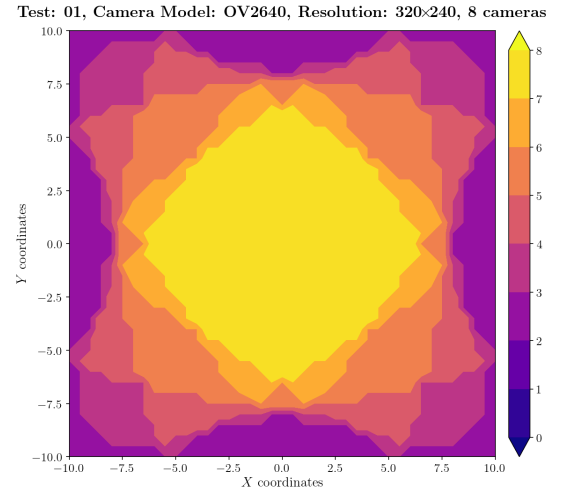
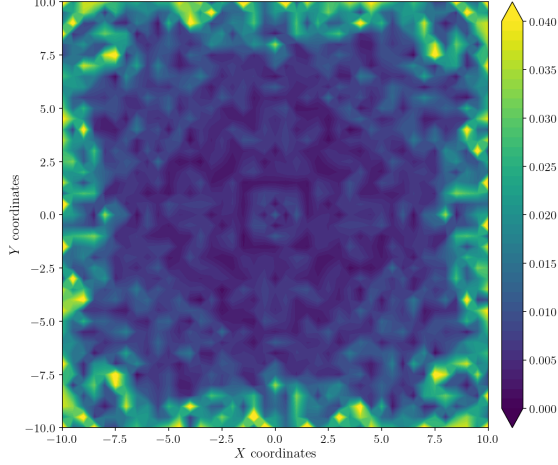


Figure A.4: Camera coverage for test 01 at level $z = 0.500$ m and resolution 320×240.

Estimation Error distribution for $z = 1.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

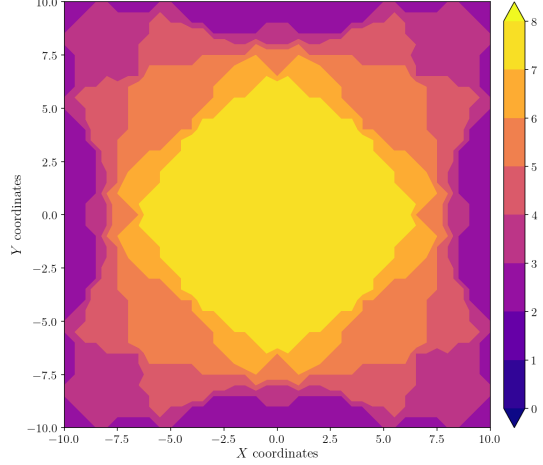
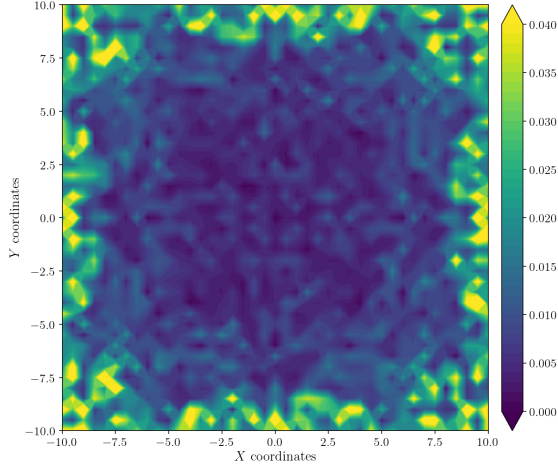


Figure A.5: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.000$ m.

Figure A.6: Camera coverage for test 01 at level $z = 1.000$ m and resolution 320×240.

Estimation Error distribution for $z = 1.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

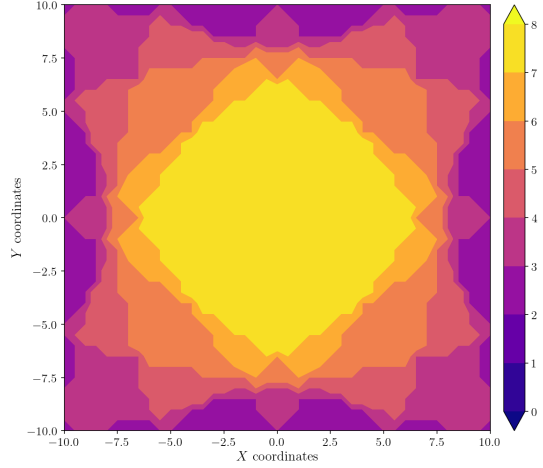
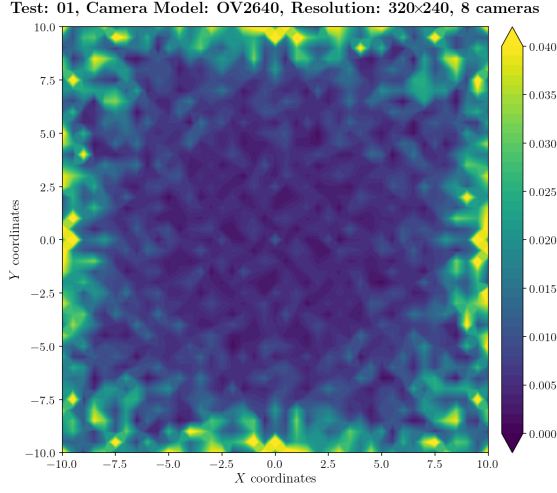


Figure A.7: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.500$ m.

Figure A.8: Camera coverage for test 01 at level $z = 1.500$ m and resolution 320×240.

Estimation Error distribution for $z = 2.0$ m level



Camera Coverage Spatial Distribution for $z = 2.0$ m level

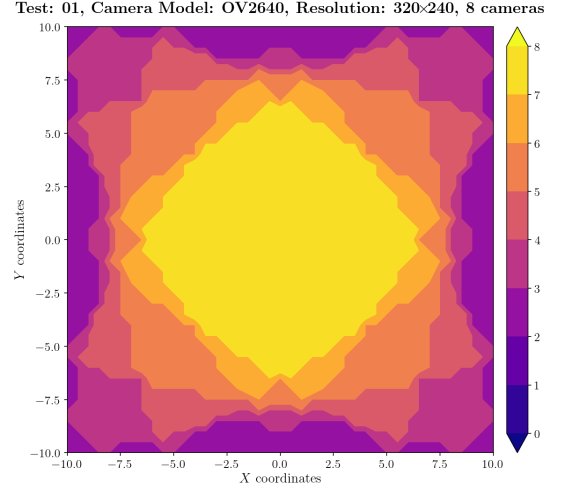
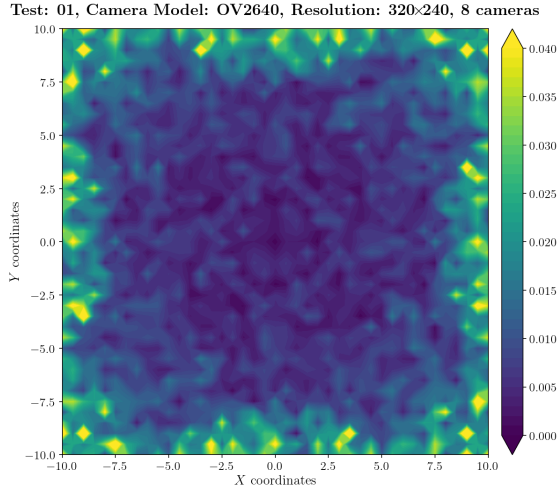


Figure A.9: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.000$ m.

Figure A.10: Camera coverage for test 01 at level $z = 2.000$ m and resolution 320×240.

Estimation Error distribution for $z = 2.5$ m level



Camera Coverage Spatial Distribution for $z = 2.5$ m level

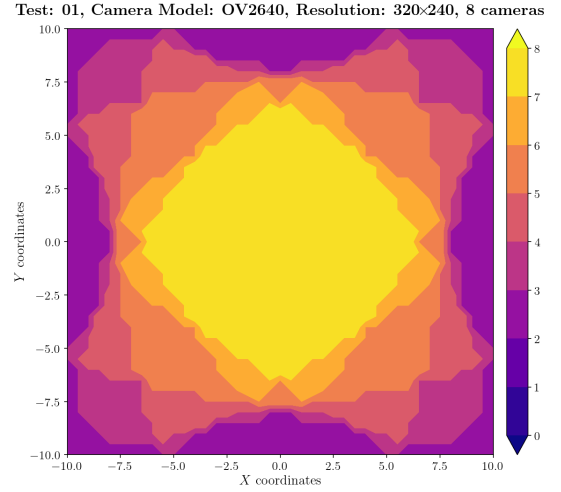
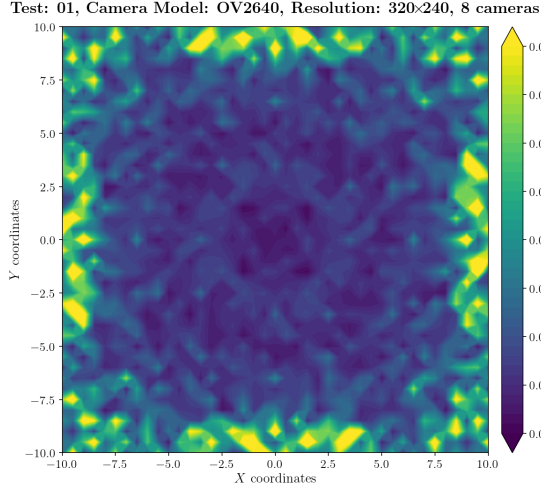


Figure A.11: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.500$ m.

Figure A.12: Camera coverage for test 01 at level $z = 2.500$ m and resolution 320×240.

Estimation Error distribution for $z = 3.0$ m level



Camera Coverage Spacial Distribution for $z = 3.0$ m level

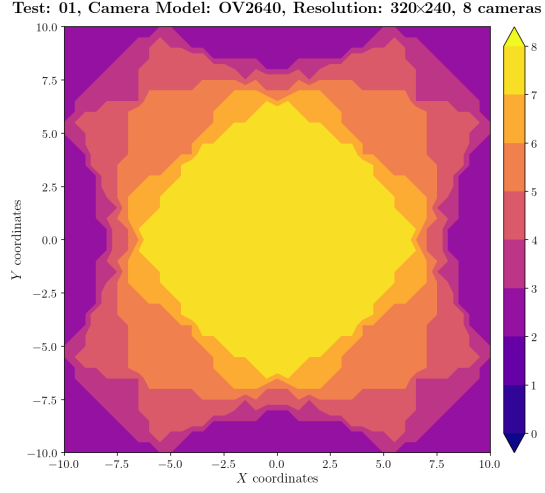
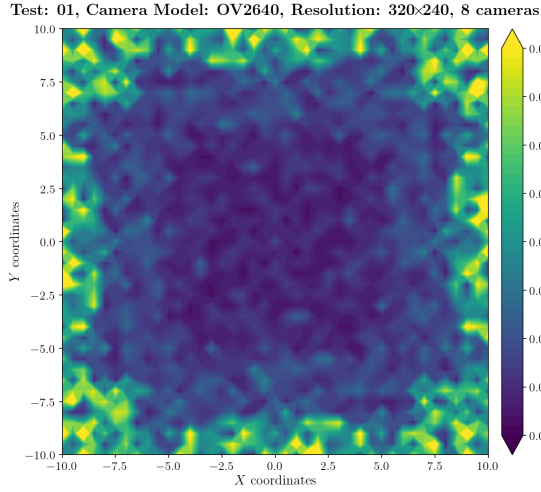


Figure A.13: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.000$ m.

Figure A.14: Camera coverage for test 01 at level $z = 3.000$ m and resolution 320×240.

Estimation Error distribution for $z = 3.5$ m level



Camera Coverage Spacial Distribution for $z = 3.5$ m level

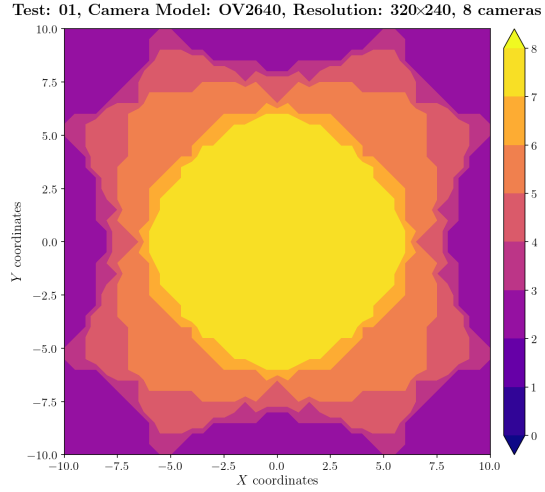
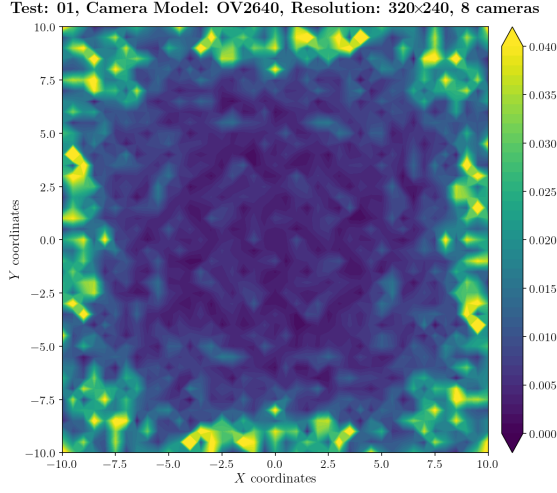


Figure A.15: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.500$ m.

Figure A.16: Camera coverage for test 01 at level $z = 3.500$ m and resolution 320×240.

Estimation Error distribution for $z = 4.0$ m level



Camera Coverage Spatial Distribution for $z = 4.0$ m level

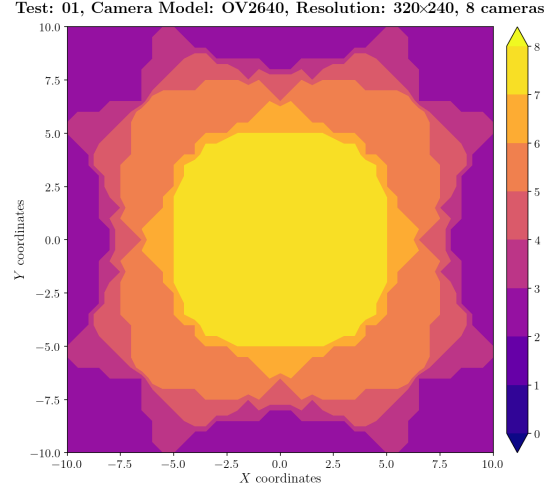
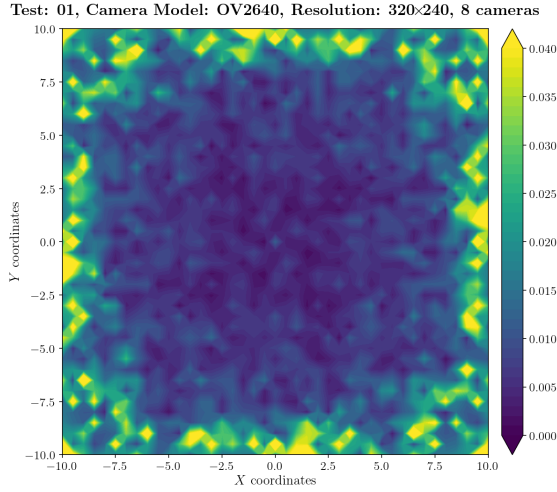


Figure A.17: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.000$ m.

Figure A.18: Camera coverage for test 01 at level $z = 4.000$ m and resolution 320×240.

Estimation Error distribution for $z = 4.5$ m level



Camera Coverage Spatial Distribution for $z = 4.5$ m level

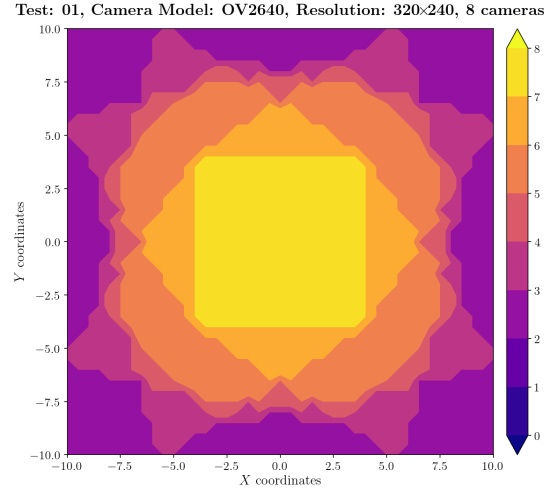
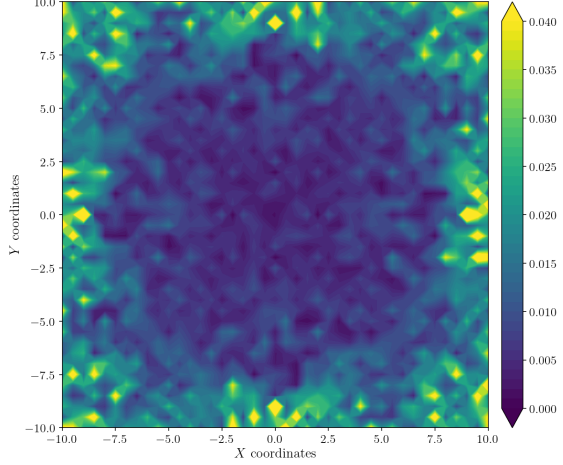


Figure A.19: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.500$ m.

Figure A.20: Camera coverage for test 01 at level $z = 4.500$ m and resolution 320×240.

Estimation Error distribution for $z = 5.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

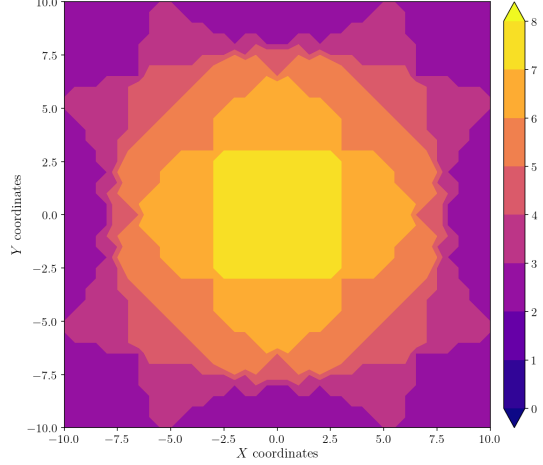
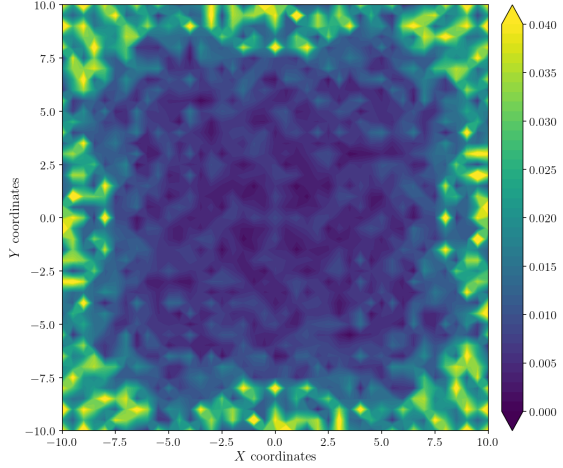


Figure A.21: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 5.000$ m.

Figure A.22: Camera coverage for test 01 at level $z = 5.000$ m and resolution 320×240.

Estimation Error distribution for $z = 5.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

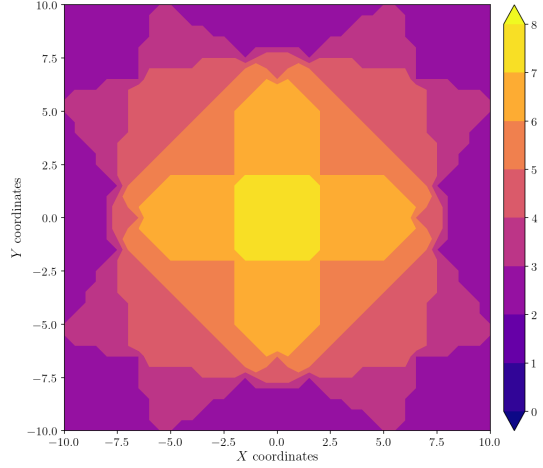


Figure A.23: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 5.500$ m.

Figure A.24: Camera coverage for test 01 at level $z = 5.500$ m and resolution 320×240.

Estimation Error distribution for $z = 6.0$ m level

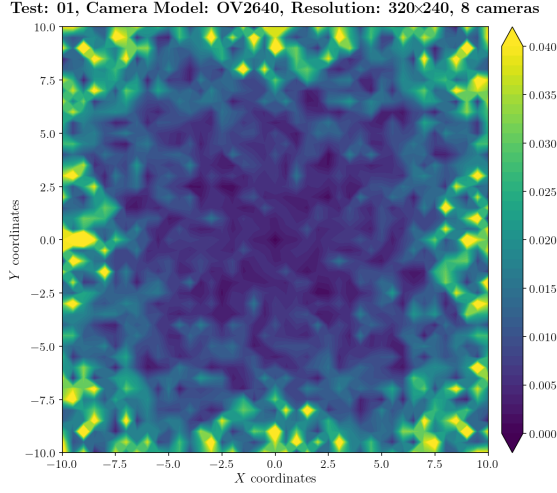


Figure A.25: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 6.000$ m.

Camera Coverage Spacial Distribution for $z = 6.0$ m level

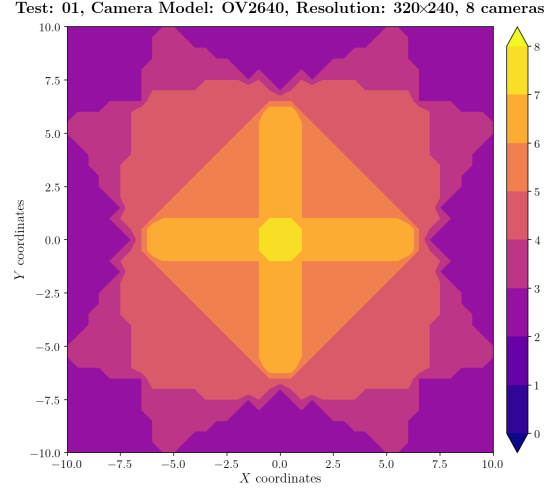


Figure A.26: Camera coverage for test 01 at level $z = 6.000$ m and resolution 320×240.

Estimation Error distribution for $z = 6.5$ m level

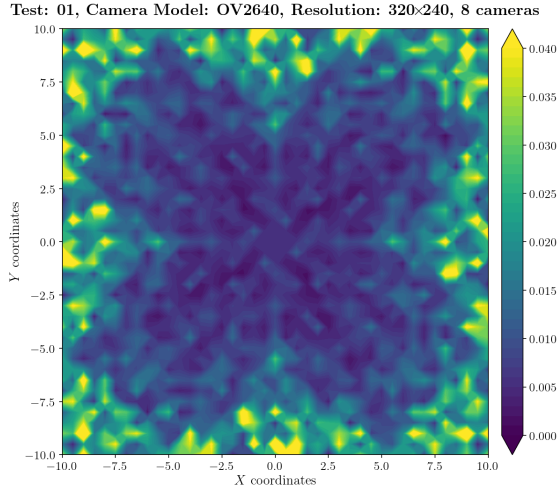


Figure A.27: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 6.500$ m.

Camera Coverage Spacial Distribution for $z = 6.5$ m level

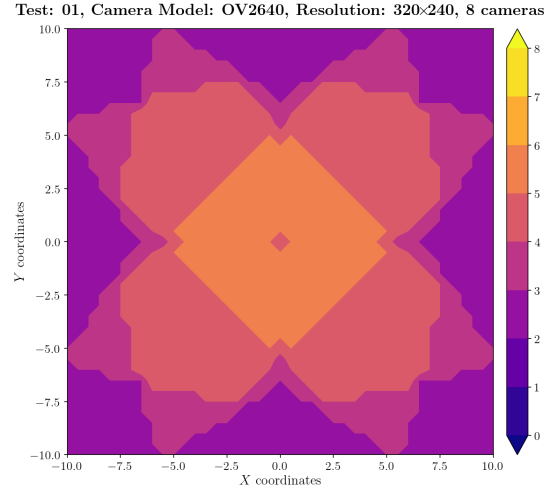
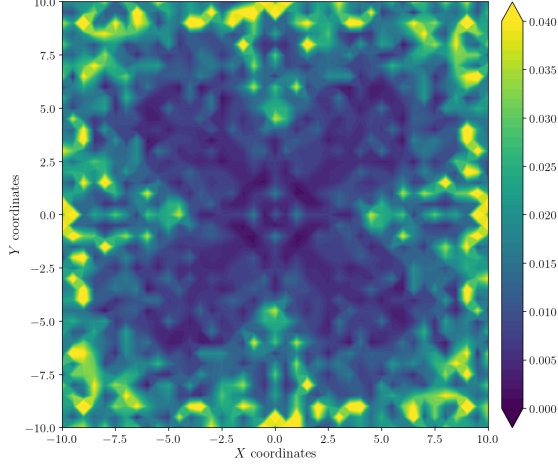


Figure A.28: Camera coverage for test 01 at level $z = 6.500$ m and resolution 320×240.

Estimation Error distribution for $z = 7.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

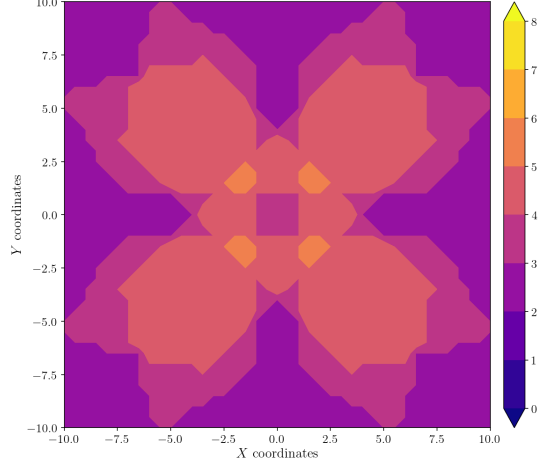
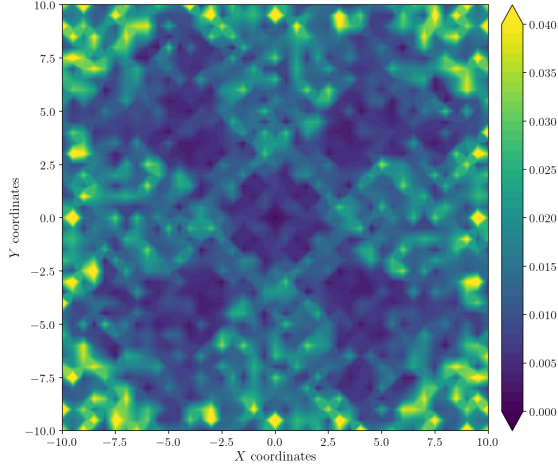


Figure A.29: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 7.000$ m.

Figure A.30: Camera coverage for test 01 at level $z = 7.000$ m and resolution 320×240.

Estimation Error distribution for $z = 7.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 320×240, 8 cameras

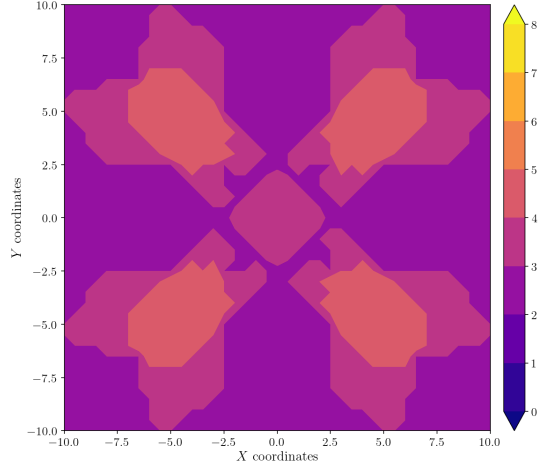


Figure A.31: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 7.500$ m.

Figure A.32: Camera coverage for test 01 at level $z = 7.500$ m and resolution 320×240.

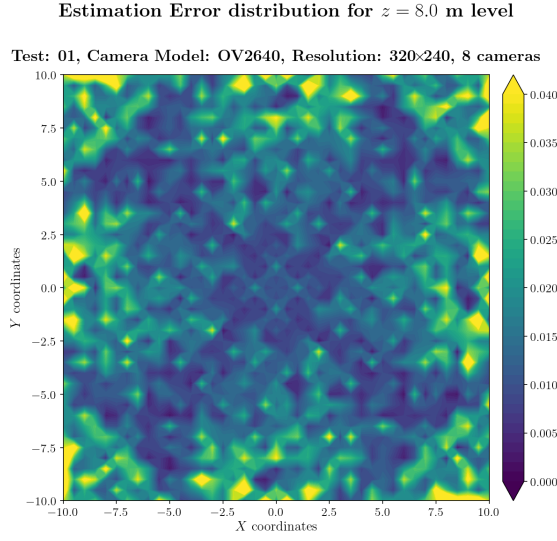


Figure A.33: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 8.000$ m.

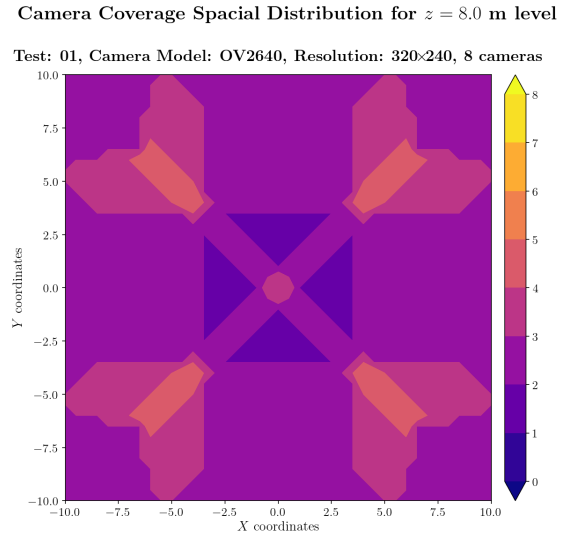


Figure A.34: Camera coverage for test 01 at level $z = 8.000$ m and resolution 320×240.

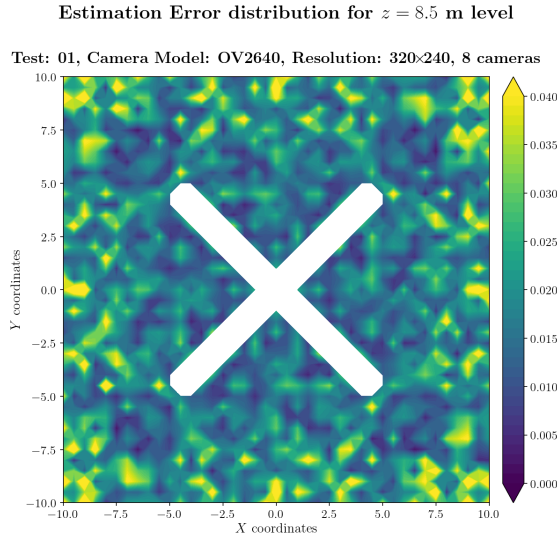


Figure A.35: Estimation error contour map for test 01 with 8 cameras using resolution of 320×240 pixels, at level $z = 8.500$ m.

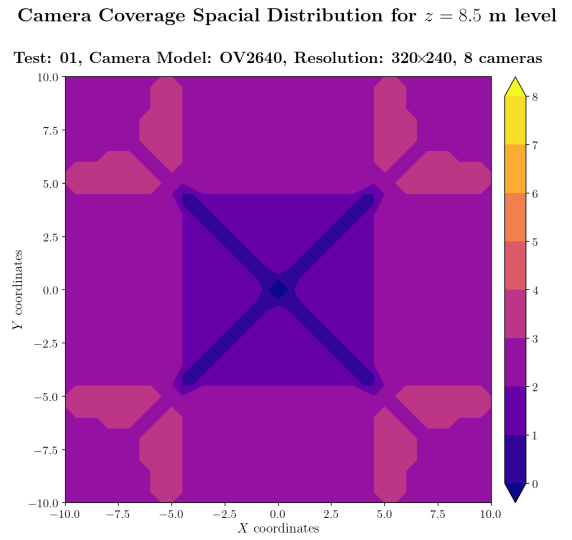


Figure A.36: Camera coverage for test 01 at level $z = 8.500$ m and resolution 320×240.

Error maps for resolution 640×480

Estimation Error distribution for $z = 0.0$ m level

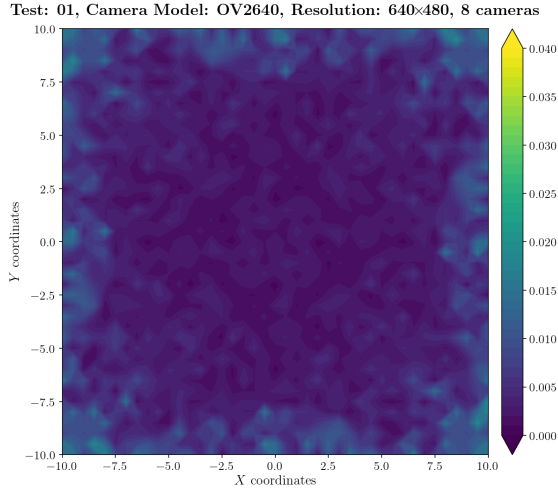


Figure A.37: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

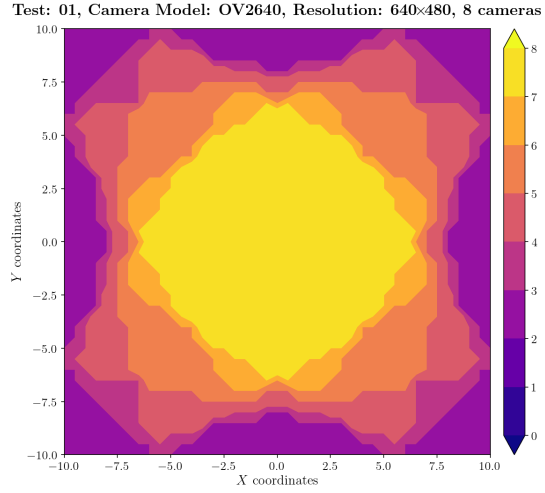


Figure A.38: Camera coverage for test 01 at level $z = 0.000$ m and resolution 640×480.

Estimation Error distribution for $z = 0.5$ m level

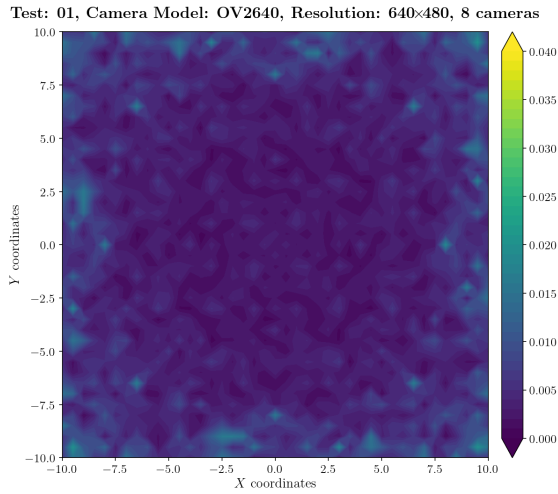


Figure A.39: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.500$ m.

Camera Coverage Spacial Distribution for $z = 0.5$ m level

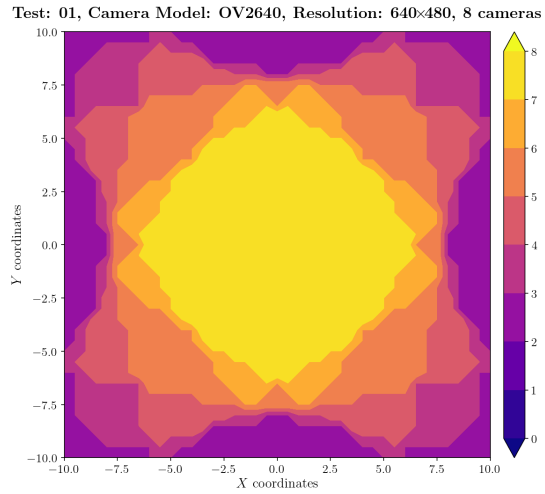


Figure A.40: Camera coverage for test 01 at level $z = 0.500$ m and resolution 640×480.

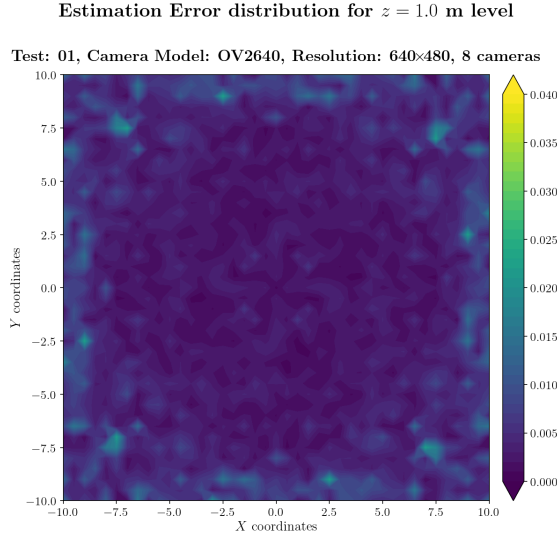


Figure A.41: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.000$ m.

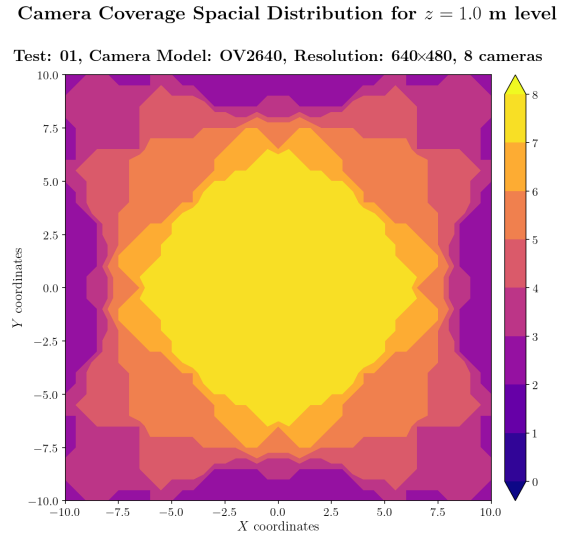


Figure A.42: Camera coverage for test 01 at level $z = 1.000$ m and resolution 640×480.

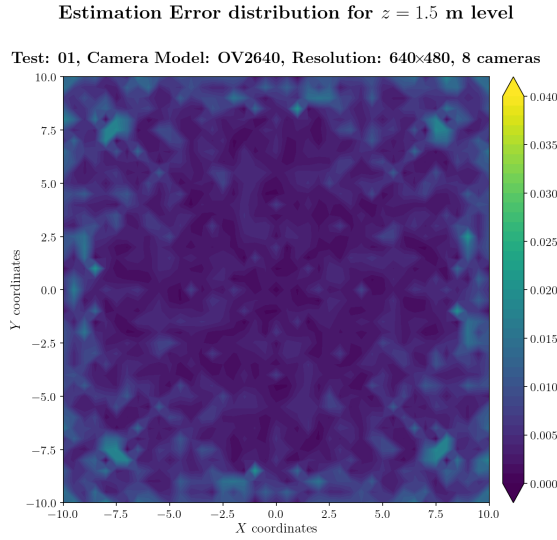


Figure A.43: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.500$ m.

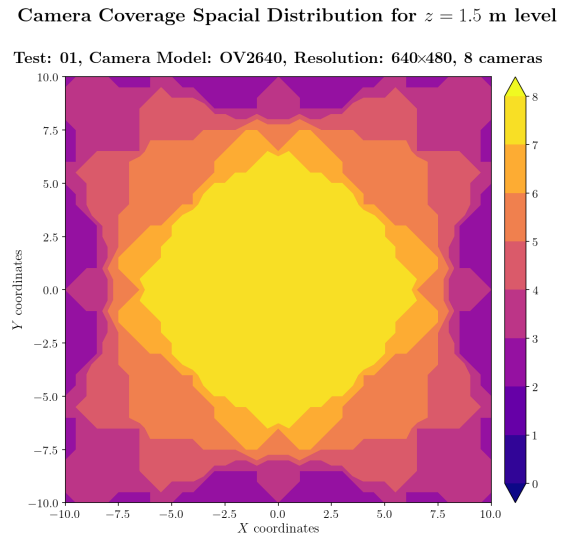
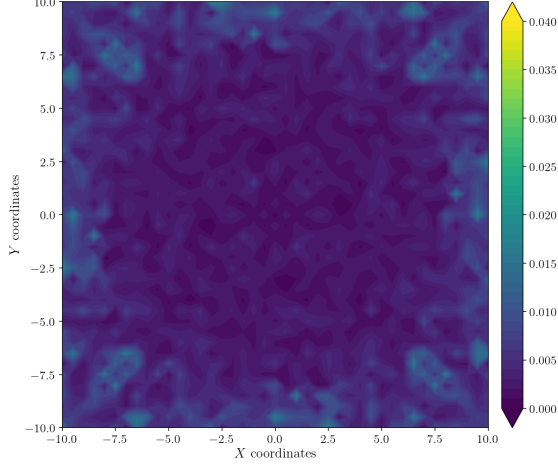


Figure A.44: Camera coverage for test 01 at level $z = 1.500$ m and resolution 640×480.

Estimation Error distribution for $z = 2.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras

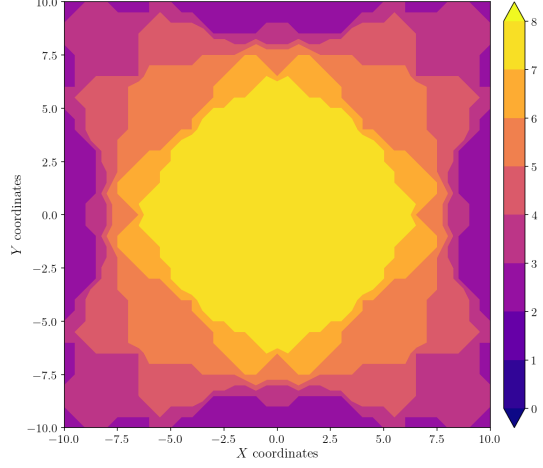
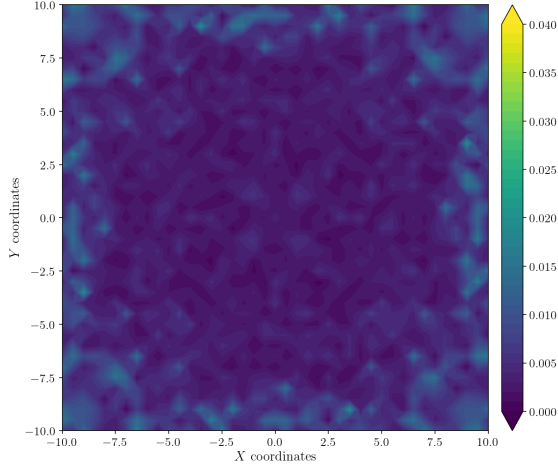


Figure A.45: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.000$ m.

Figure A.46: Camera coverage for test 01 at level $z = 2.000$ m and resolution 640×480.

Estimation Error distribution for $z = 2.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras

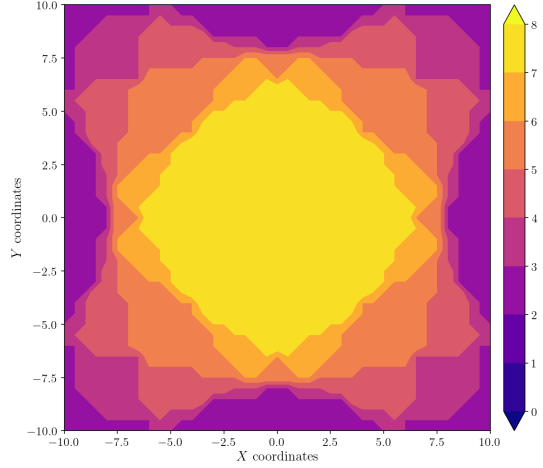


Figure A.47: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.500$ m.

Figure A.48: Camera coverage for test 01 at level $z = 2.500$ m and resolution 640×480.

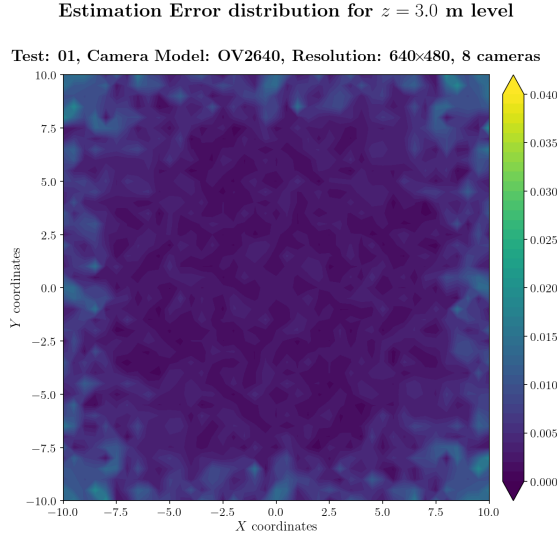


Figure A.49: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.000$ m.

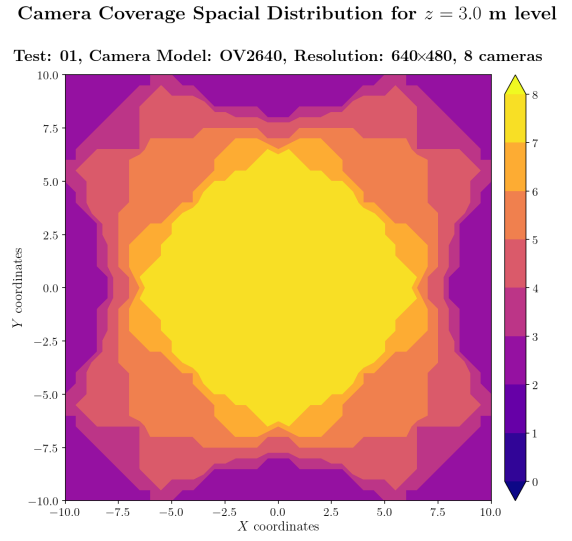


Figure A.50: Camera coverage for test 01 at level $z = 3.000$ m and resolution 640×480.

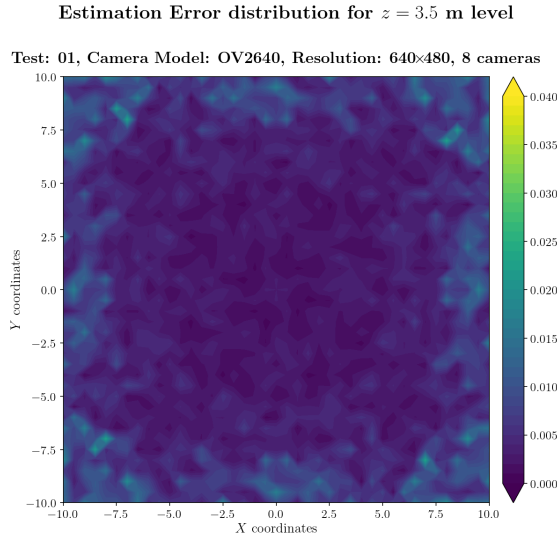


Figure A.51: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.500$ m.

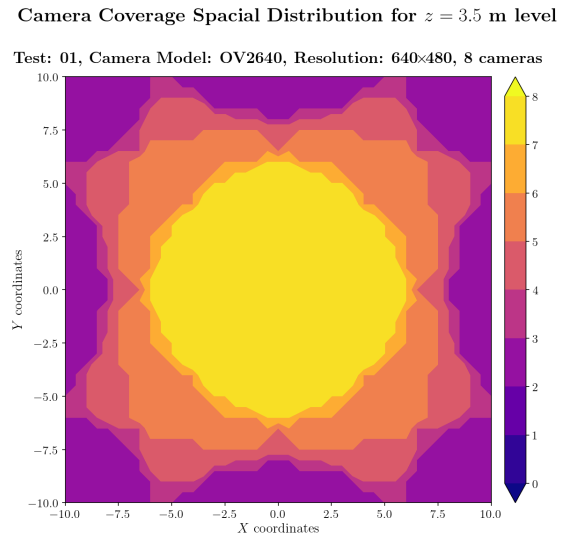
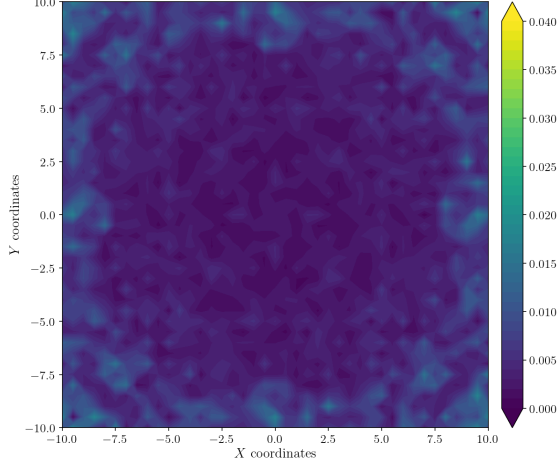


Figure A.52: Camera coverage for test 01 at level $z = 3.500$ m and resolution 640×480.

Estimation Error distribution for $z = 4.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras

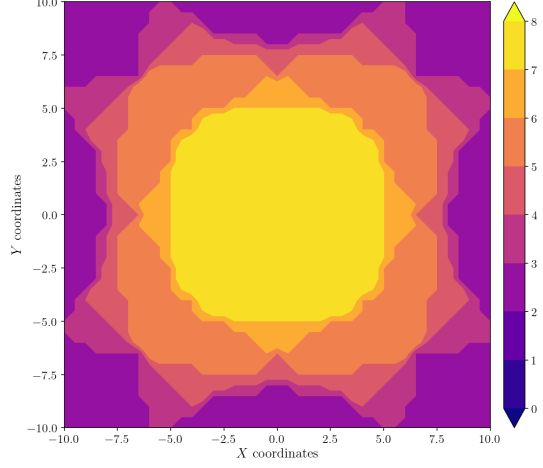
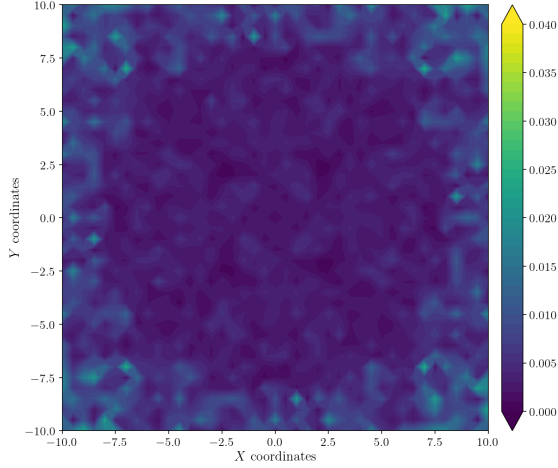


Figure A.53: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.000$ m.

Figure A.54: Camera coverage for test 01 at level $z = 4.000$ m and resolution 640×480.

Estimation Error distribution for $z = 4.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras

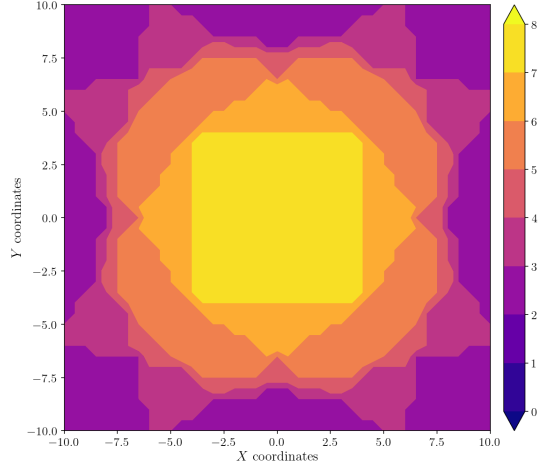


Figure A.55: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.500$ m.

Figure A.56: Camera coverage for test 01 at level $z = 4.500$ m and resolution 640×480.

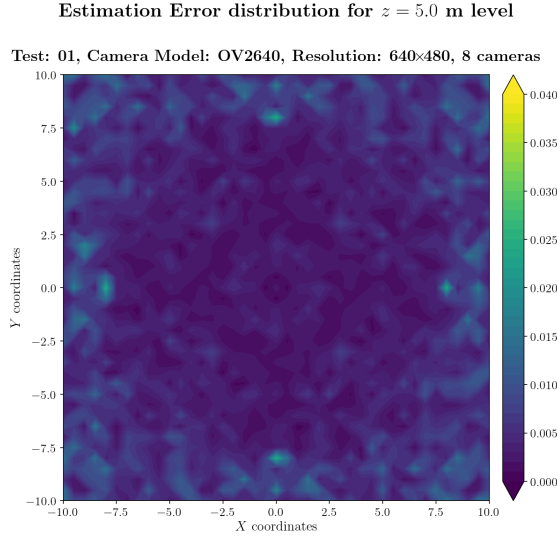


Figure A.57: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 5.000$ m.

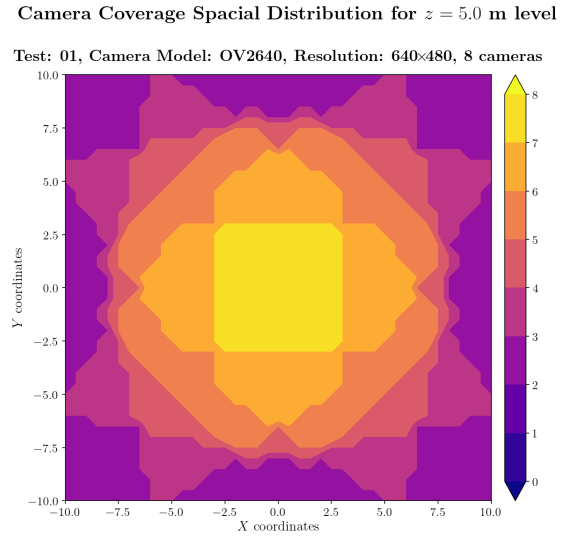


Figure A.58: Camera coverage for test 01 at level $z = 5.000$ m and resolution 640×480.

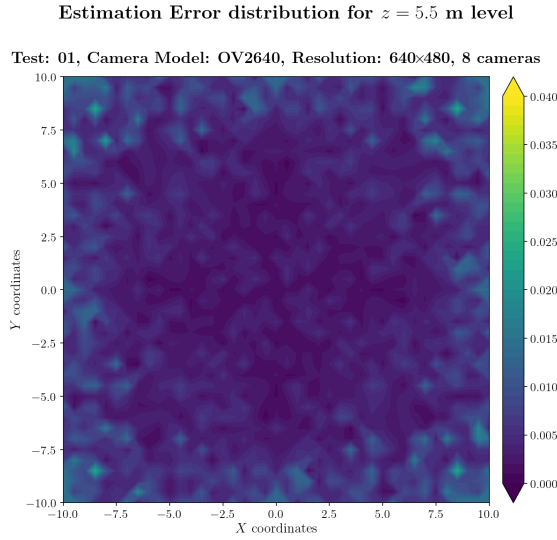


Figure A.59: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 5.500$ m.

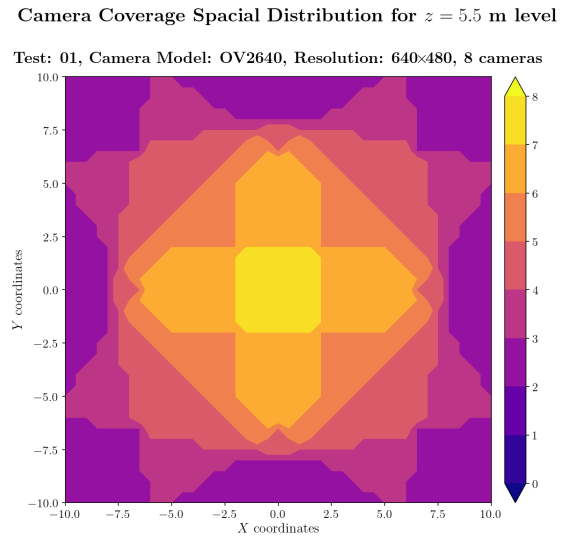
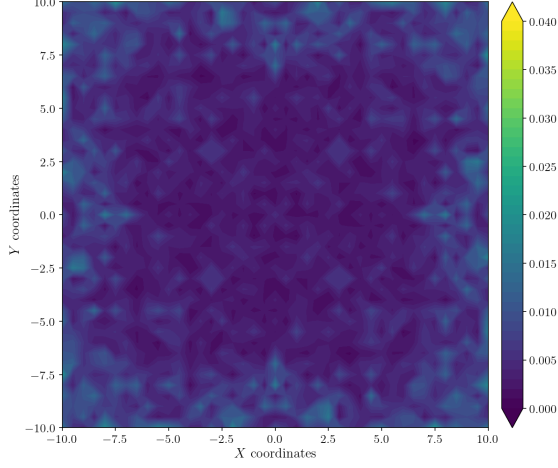


Figure A.60: Camera coverage for test 01 at level $z = 5.500$ m and resolution 640×480.

Estimation Error distribution for $z = 6.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras

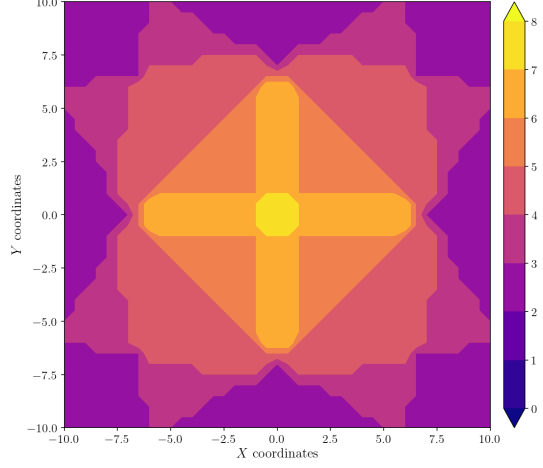
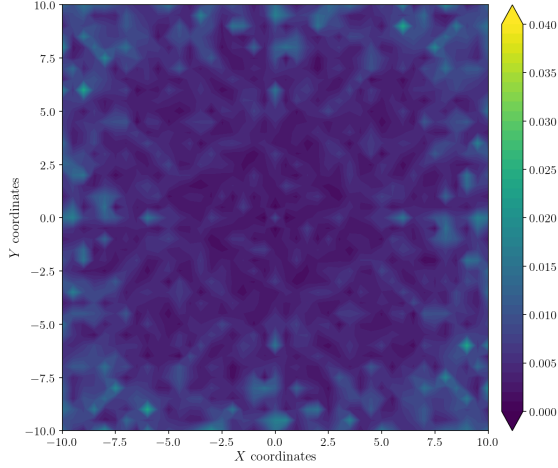


Figure A.61: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 6.000$ m.

Figure A.62: Camera coverage for test 01 at level $z = 6.000$ m and resolution 640×480.

Estimation Error distribution for $z = 6.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras

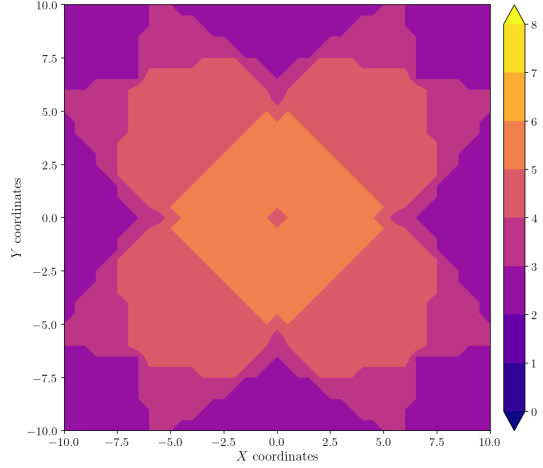


Figure A.63: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 6.500$ m.

Figure A.64: Camera coverage for test 01 at level $z = 6.500$ m and resolution 640×480.

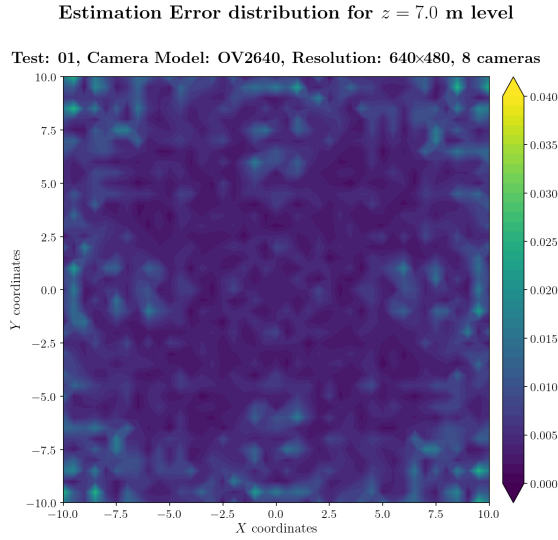


Figure A.65: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 7.000$ m.

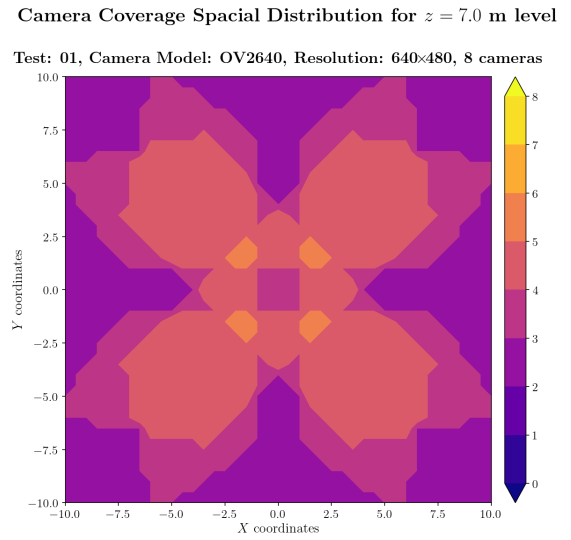


Figure A.66: Camera coverage for test 01 at level $z = 7.000$ m and resolution 640×480.

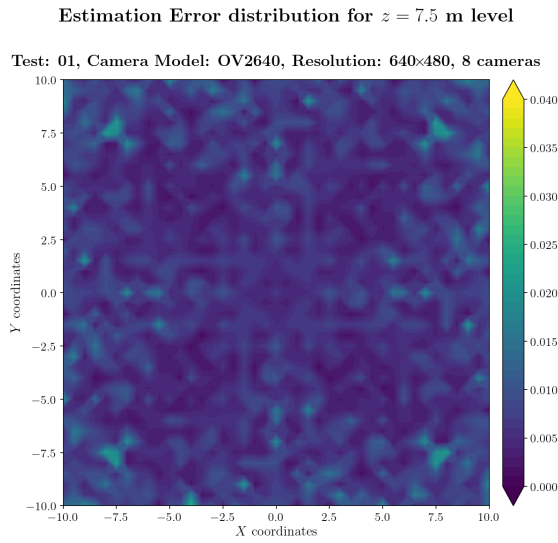


Figure A.67: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 7.500$ m.

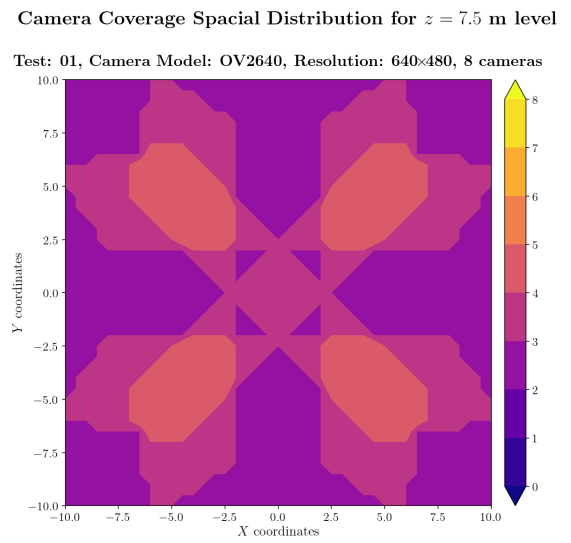
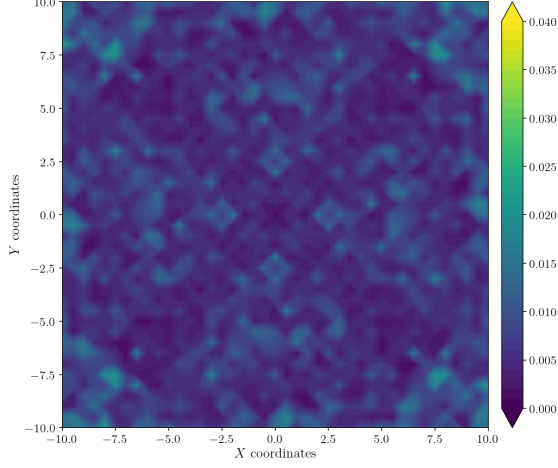


Figure A.68: Camera coverage for test 01 at level $z = 7.500$ m and resolution 640×480.

Estimation Error distribution for $z = 8.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras

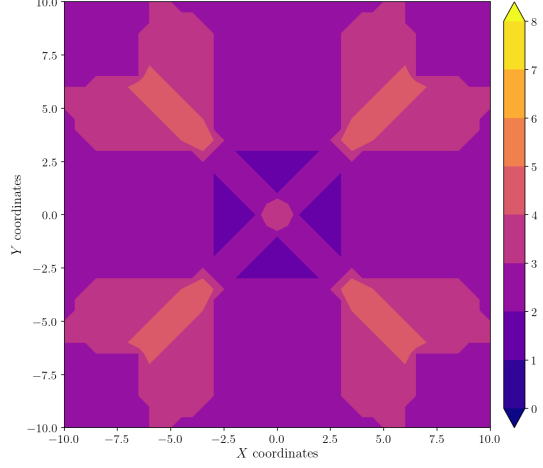
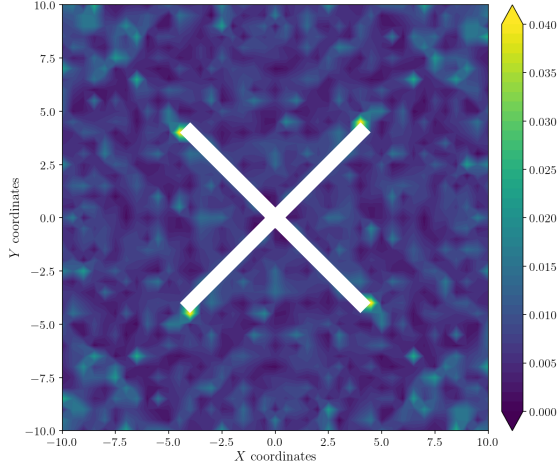


Figure A.69: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 8.000$ m.

Figure A.70: Camera coverage for test 01 at level $z = 8.000$ m and resolution 640×480.

Estimation Error distribution for $z = 8.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 640×480, 8 cameras

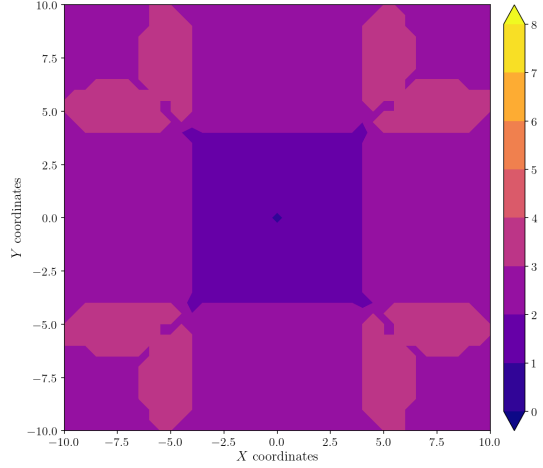


Figure A.71: Estimation error contour map for test 01 with 8 cameras using resolution of 640×480 pixels, at level $z = 8.500$ m.

Figure A.72: Camera coverage for test 01 at level $z = 8.500$ m and resolution 640×480.

Error maps for resolution 800×600

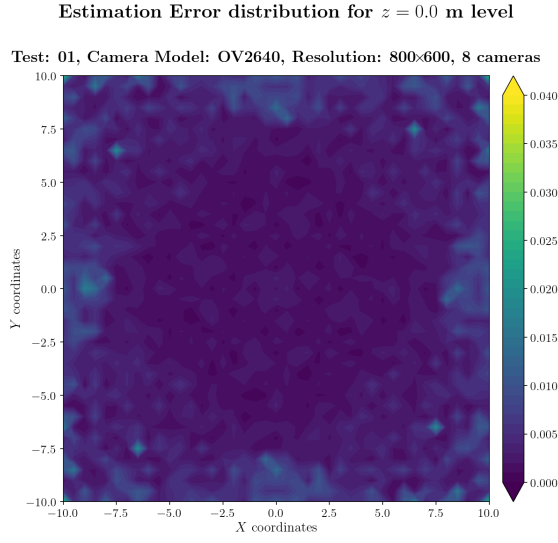


Figure A.73: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.000$ m.

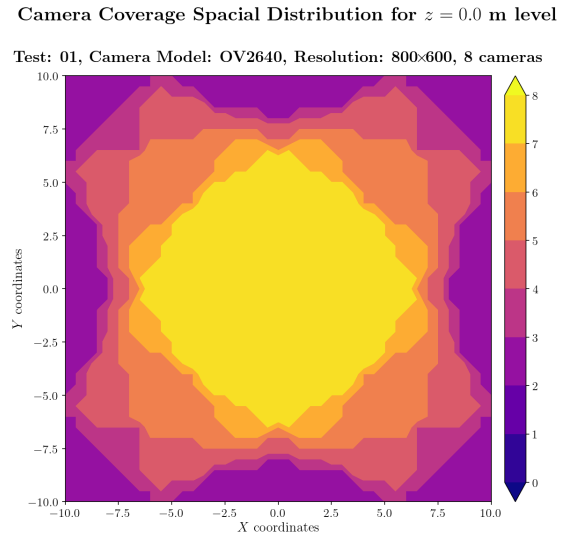


Figure A.74: Camera coverage for test 01 at level $z = 0.000$ m and resolution 800×600.

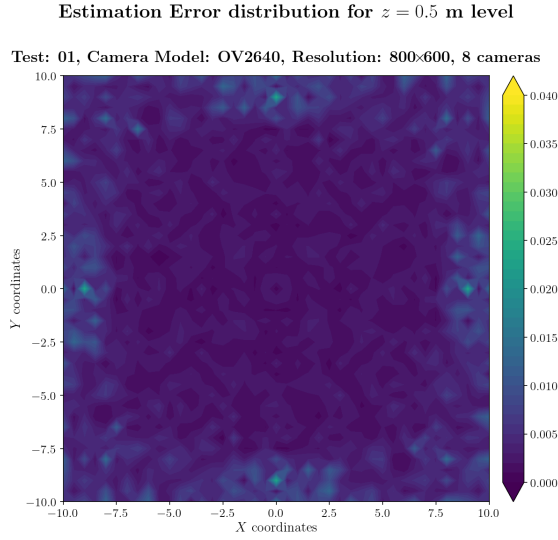


Figure A.75: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.500$ m.

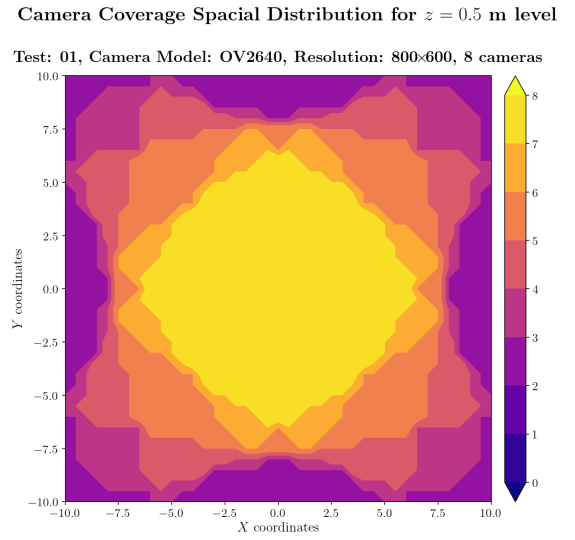
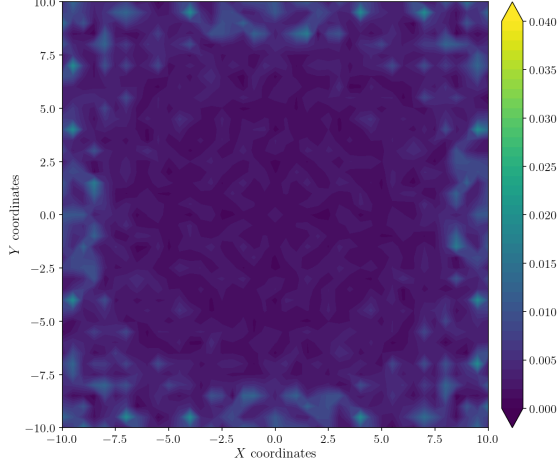


Figure A.76: Camera coverage for test 01 at level $z = 0.500$ m and resolution 800×600.

Estimation Error distribution for $z = 1.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras

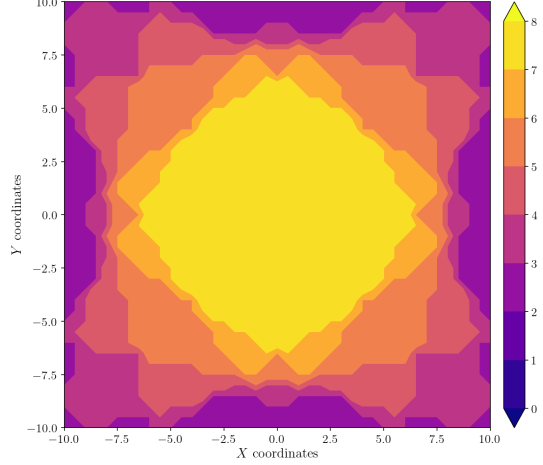
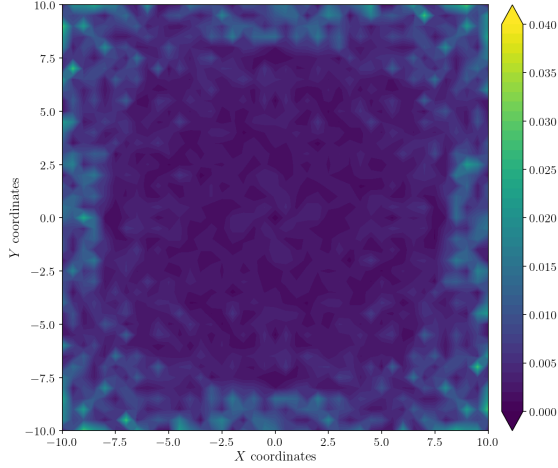


Figure A.77: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.000$ m.

Figure A.78: Camera coverage for test 01 at level $z = 1.000$ m and resolution 800×600.

Estimation Error distribution for $z = 1.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras

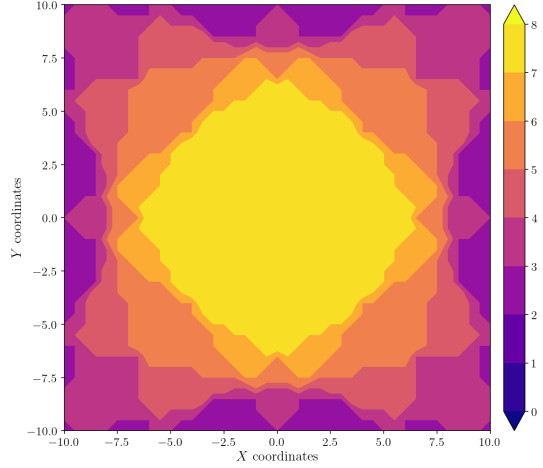
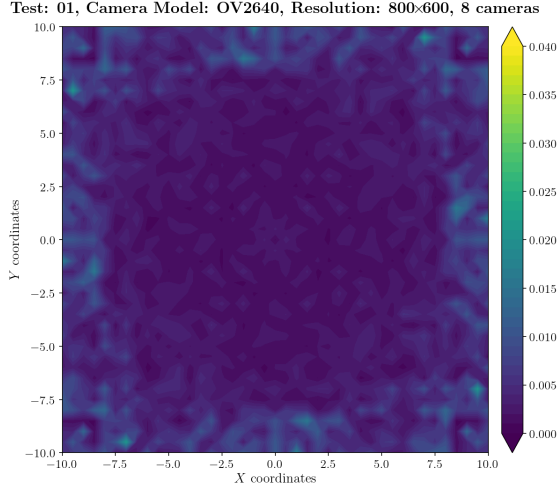


Figure A.79: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.500$ m.

Figure A.80: Camera coverage for test 01 at level $z = 1.500$ m and resolution 800×600.

Estimation Error distribution for $z = 2.0$ m level



Camera Coverage Spatial Distribution for $z = 2.0$ m level

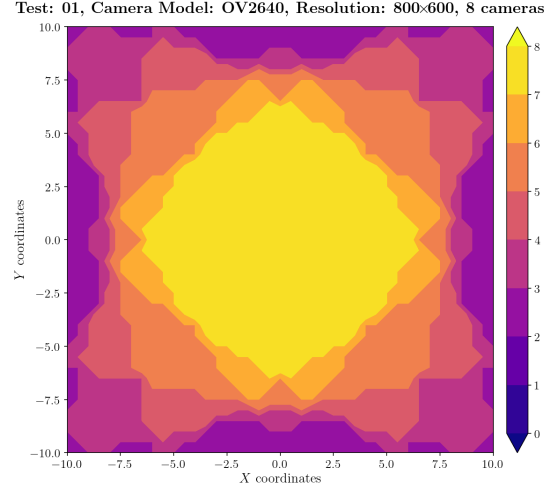
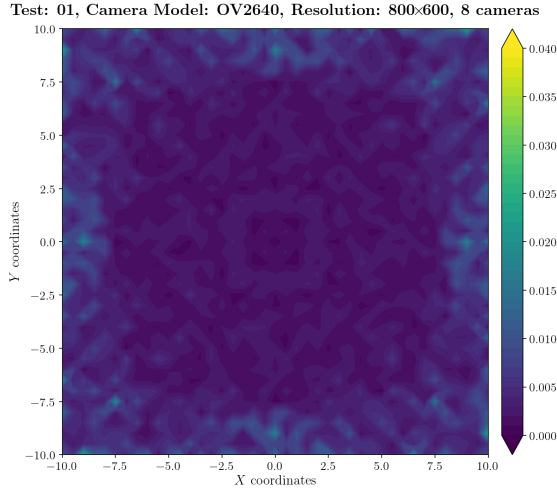


Figure A.81: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.000$ m.

Figure A.82: Camera coverage for test 01 at level $z = 2.000$ m and resolution 800×600.

Estimation Error distribution for $z = 2.5$ m level



Camera Coverage Spatial Distribution for $z = 2.5$ m level

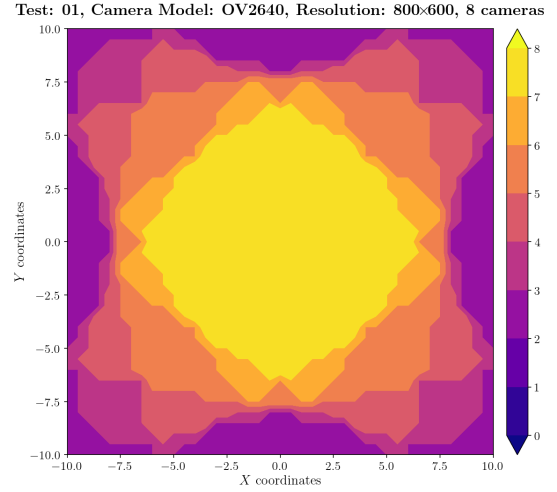
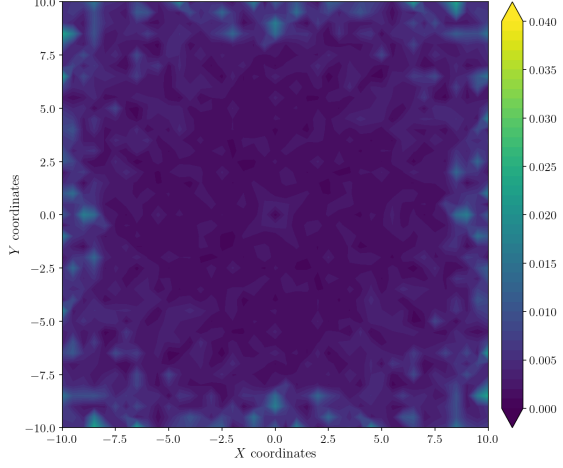


Figure A.83: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.500$ m.

Figure A.84: Camera coverage for test 01 at level $z = 2.500$ m and resolution 800×600.

Estimation Error distribution for $z = 3.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras

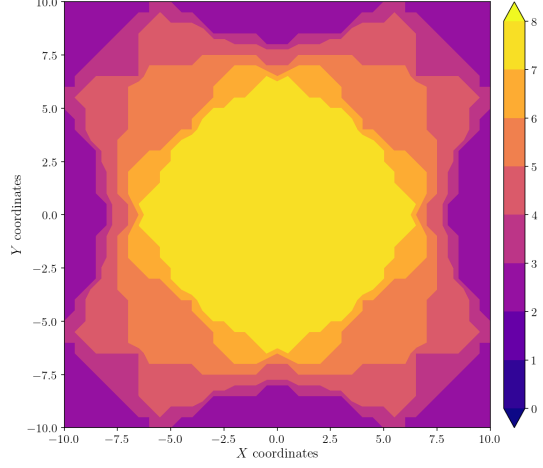
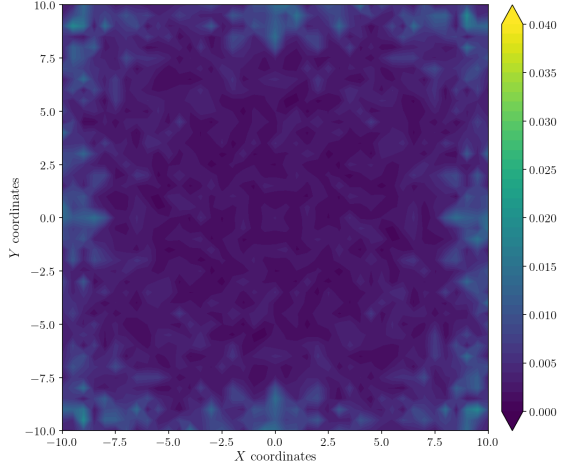


Figure A.85: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.000$ m.

Figure A.86: Camera coverage for test 01 at level $z = 3.000$ m and resolution 800×600.

Estimation Error distribution for $z = 3.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras

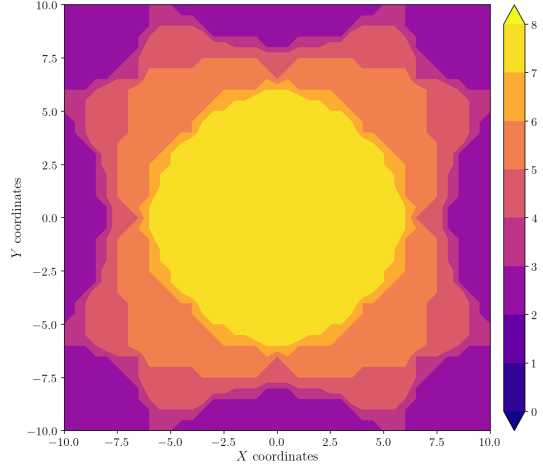


Figure A.87: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.500$ m.

Figure A.88: Camera coverage for test 01 at level $z = 3.500$ m and resolution 800×600.

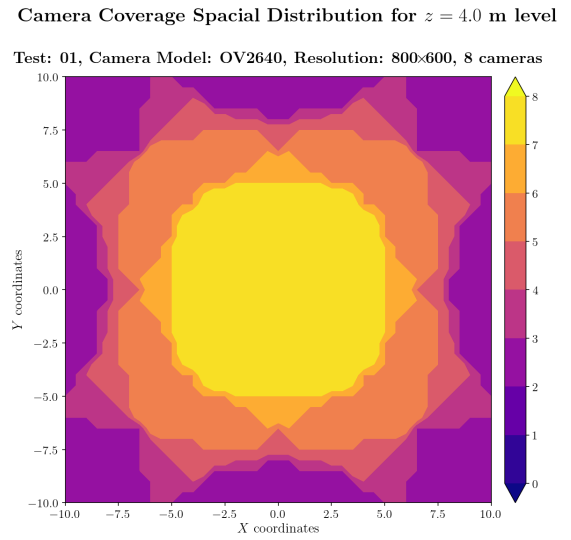
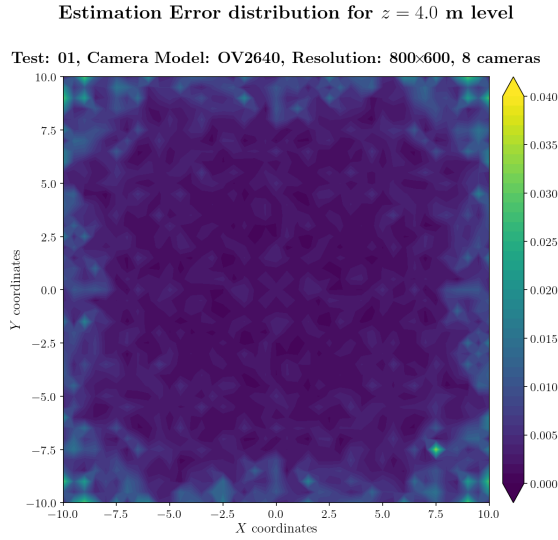


Figure A.89: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.000$ m.

Figure A.90: Camera coverage for test 01 at level $z = 4.000$ m and resolution 800×600.

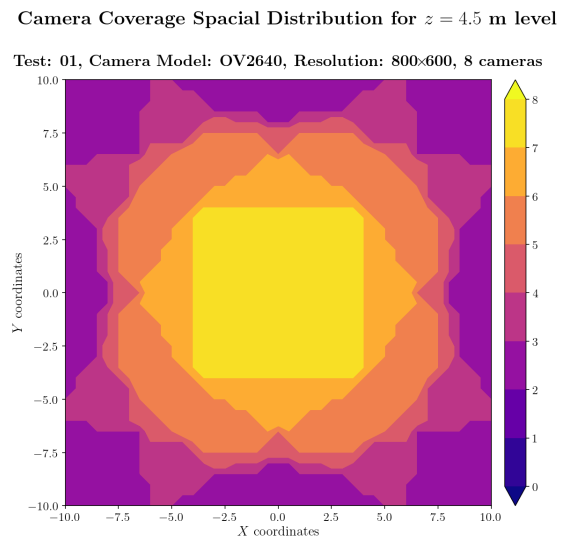
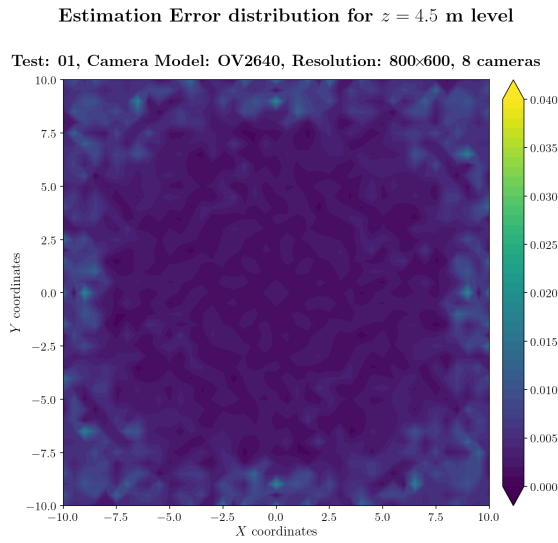
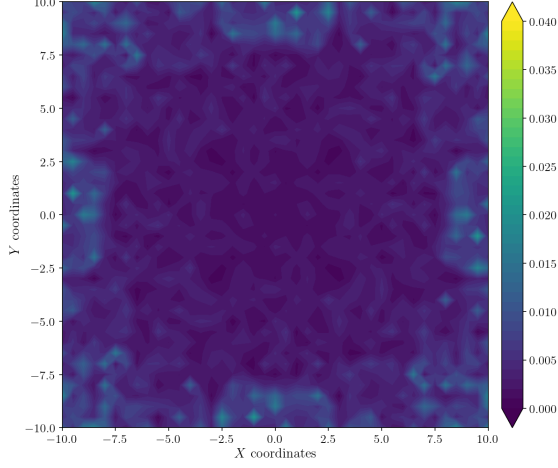


Figure A.91: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.500$ m.

Figure A.92: Camera coverage for test 01 at level $z = 4.500$ m and resolution 800×600.

Estimation Error distribution for $z = 5.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras

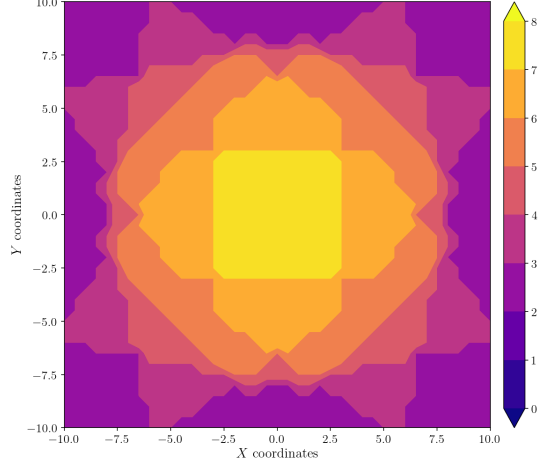
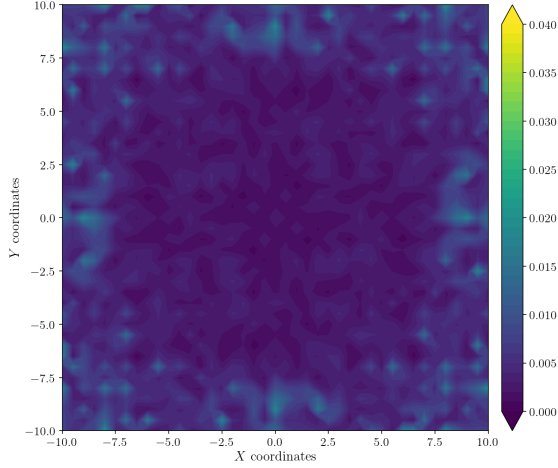


Figure A.93: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 5.000$ m.

Figure A.94: Camera coverage for test 01 at level $z = 5.000$ m and resolution 800×600.

Estimation Error distribution for $z = 5.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras

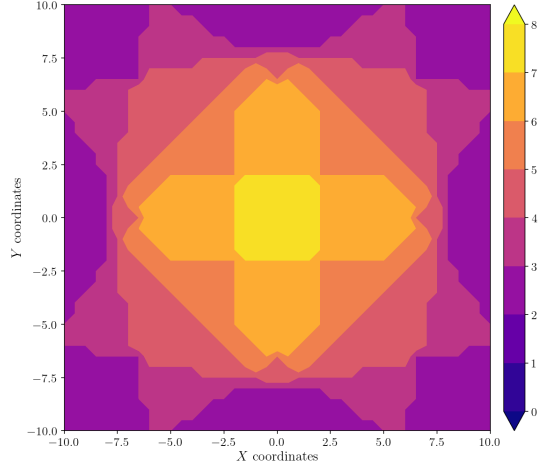


Figure A.95: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 5.500$ m.

Figure A.96: Camera coverage for test 01 at level $z = 5.500$ m and resolution 800×600.

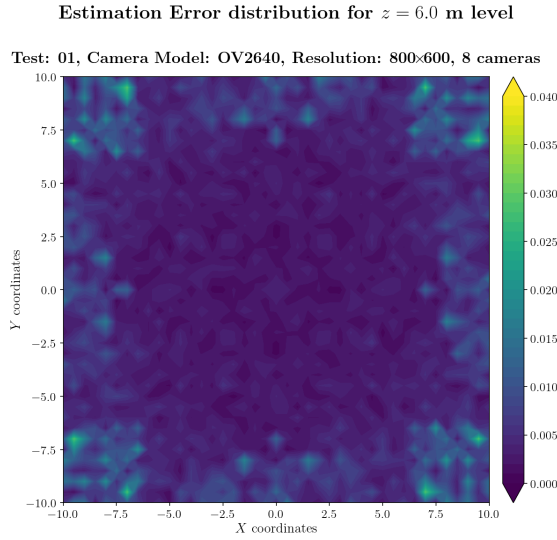


Figure A.97: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 6.000$ m.

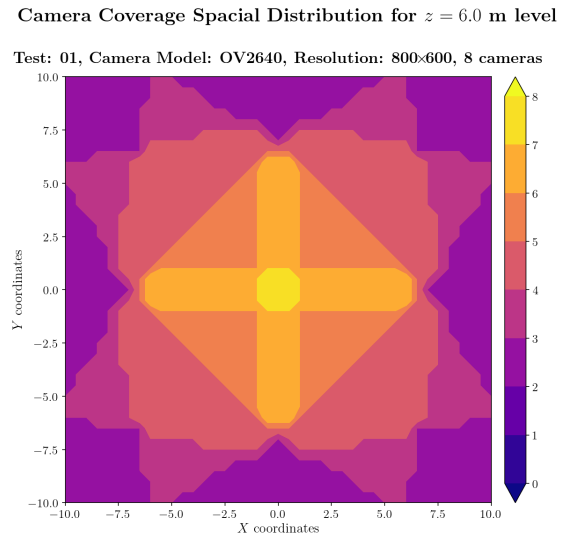


Figure A.98: Camera coverage for test 01 at level $z = 6.000$ m and resolution 800×600.

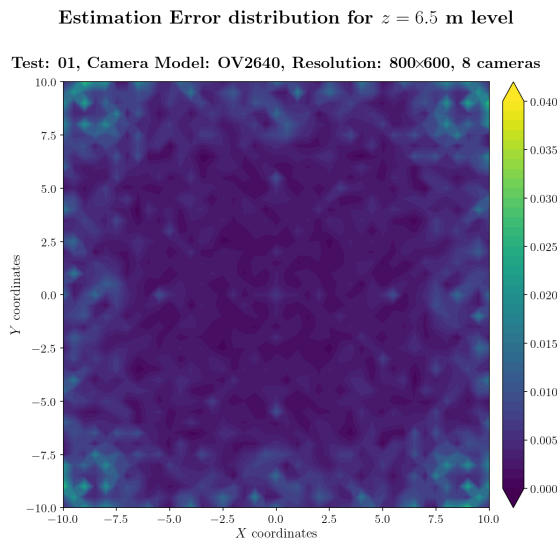


Figure A.99: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 6.500$ m.

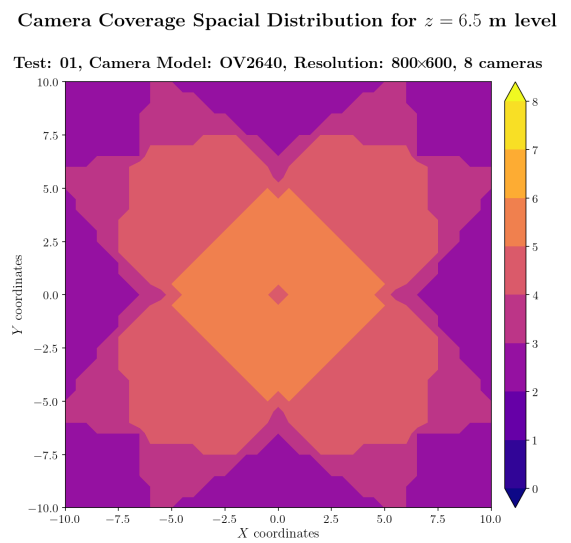
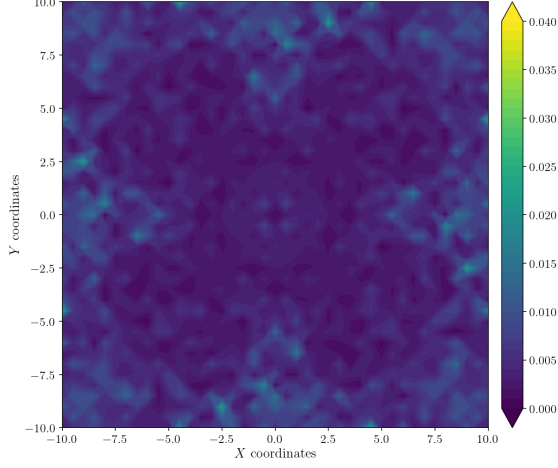


Figure A.100: Camera coverage for test 01 at level $z = 6.500$ m and resolution 800×600.

Estimation Error distribution for $z = 7.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras

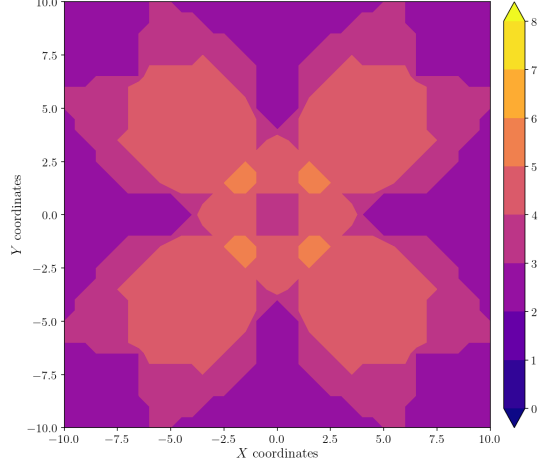
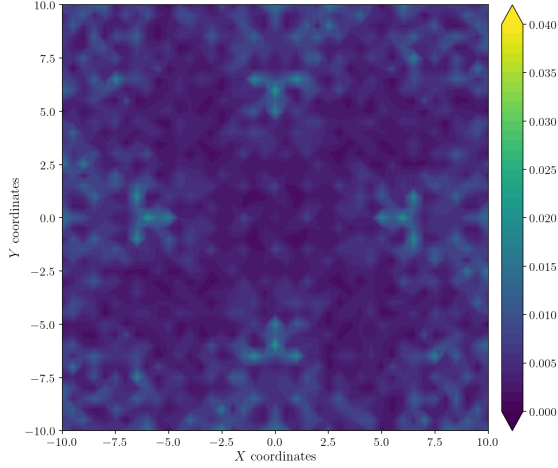


Figure A.101: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 7.000$ m.

Figure A.102: Camera coverage for test 01 at level $z = 7.000$ m and resolution 800×600.

Estimation Error distribution for $z = 7.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 800×600, 8 cameras

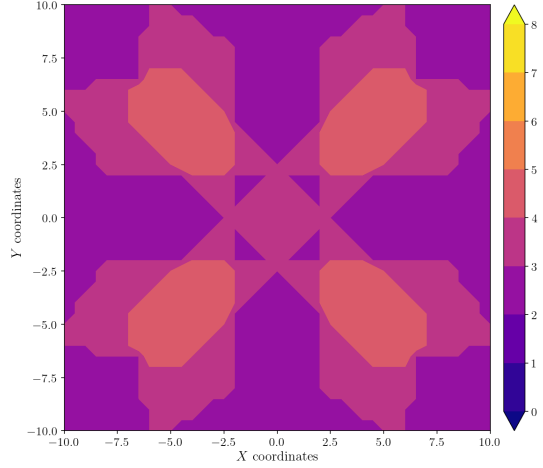


Figure A.103: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 7.500$ m.

Figure A.104: Camera coverage for test 01 at level $z = 7.500$ m and resolution 800×600.

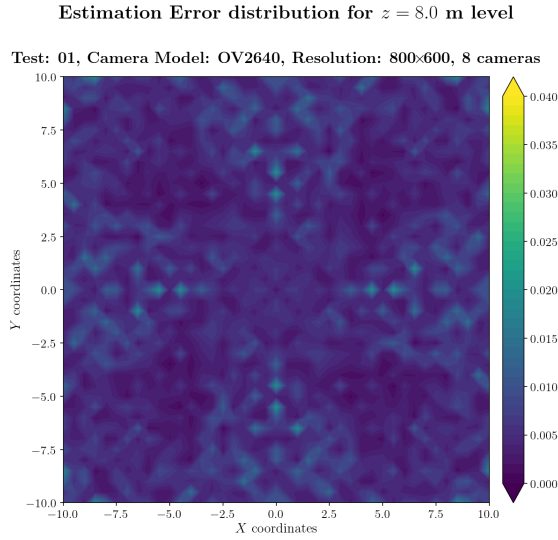


Figure A.105: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 8.000$ m.

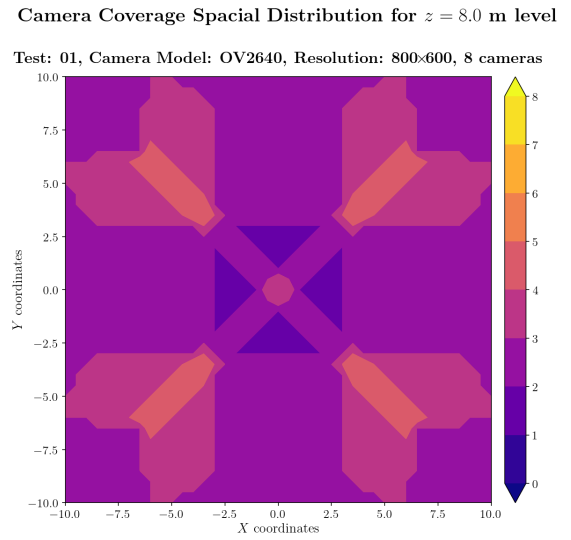


Figure A.106: Camera coverage for test 01 at level $z = 8.000$ m and resolution 800×600.

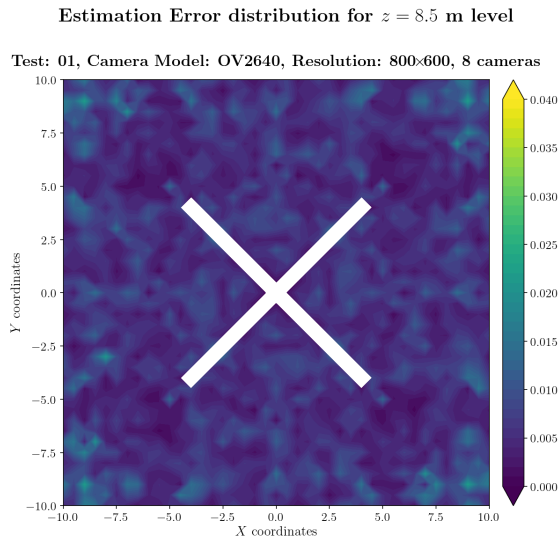


Figure A.107: Estimation error contour map for test 01 with 8 cameras using resolution of 800×600 pixels, at level $z = 8.500$ m.

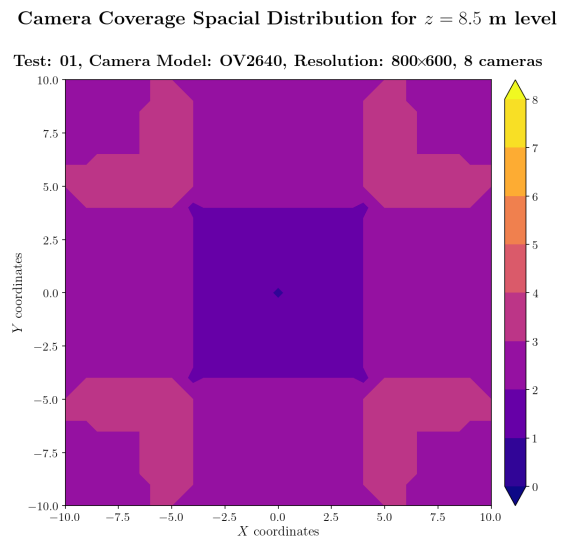


Figure A.108: Camera coverage for test 01 at level $z = 8.500$ m and resolution 800×600.

Error maps for resolution 1024×768

Estimation Error distribution for $z = 0.0$ m level

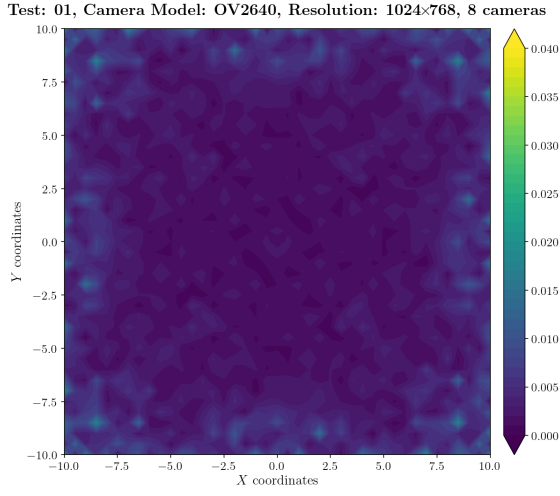


Figure A.109: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

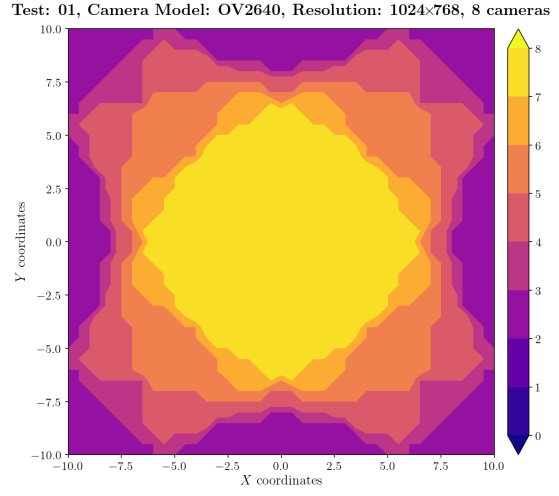


Figure A.110: Camera coverage for test 01 at level $z = 0.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 0.5$ m level

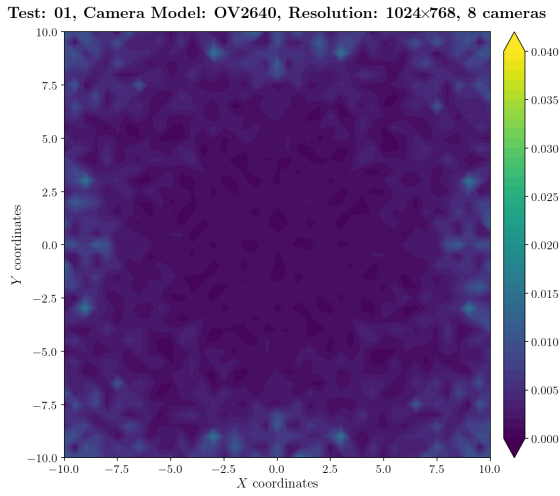


Figure A.111: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.500$ m.

Camera Coverage Spacial Distribution for $z = 0.5$ m level

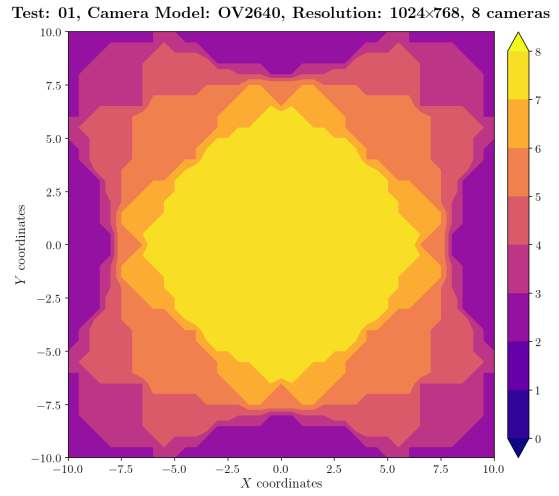


Figure A.112: Camera coverage for test 01 at level $z = 0.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.0$ m level

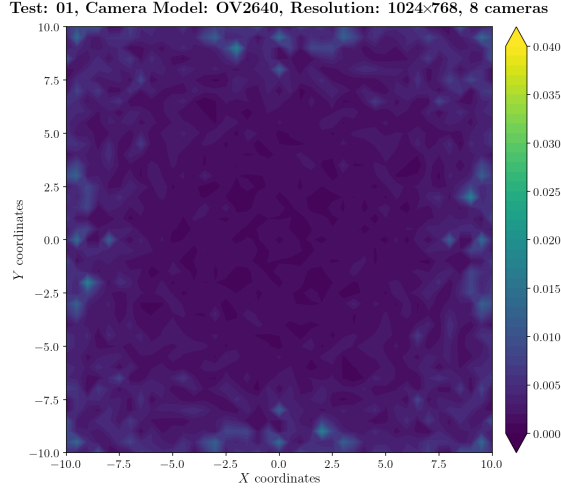


Figure A.113: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.000$ m.

Camera Coverage Spatial Distribution for $z = 1.0$ m level

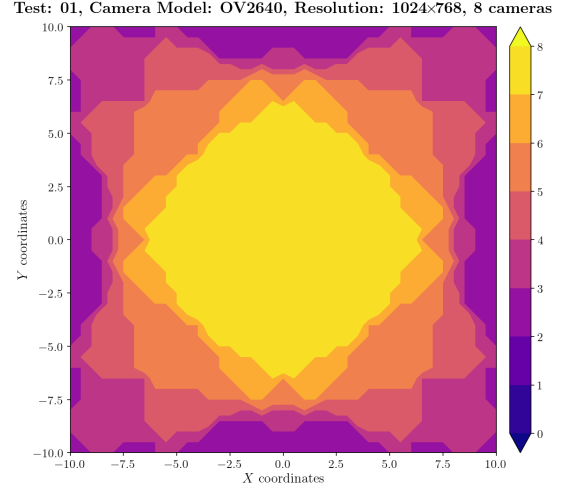


Figure A.114: Camera coverage for test 01 at level $z = 1.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.5$ m level

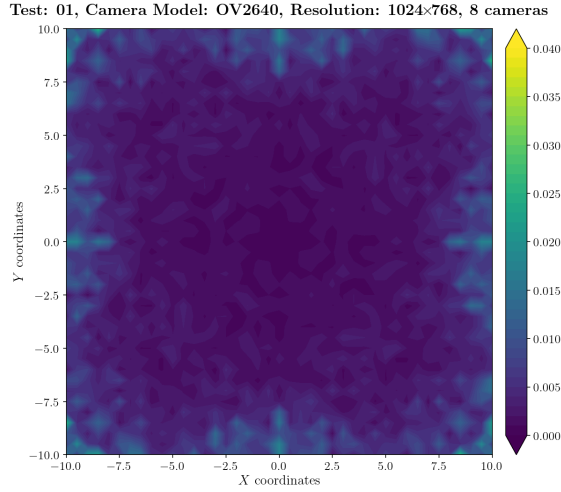


Figure A.115: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.500$ m.

Camera Coverage Spatial Distribution for $z = 1.5$ m level

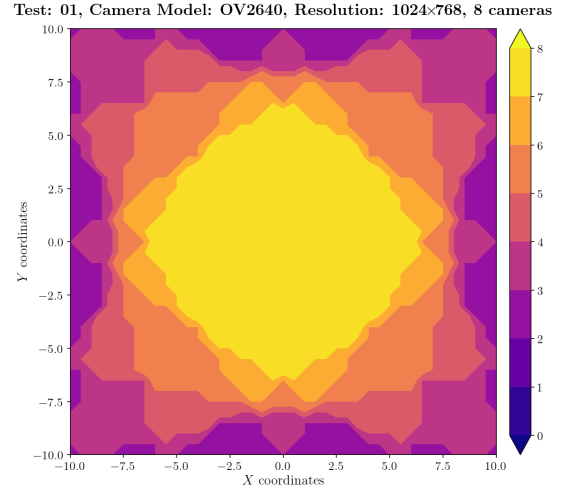
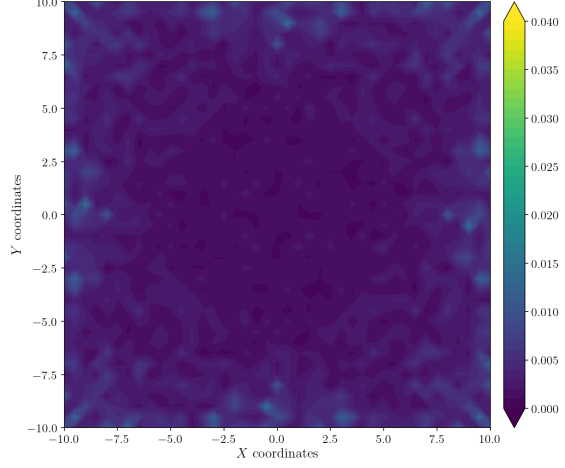


Figure A.116: Camera coverage for test 01 at level $z = 1.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 2.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

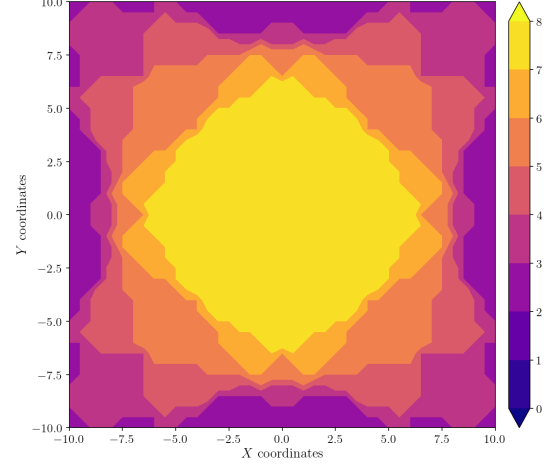
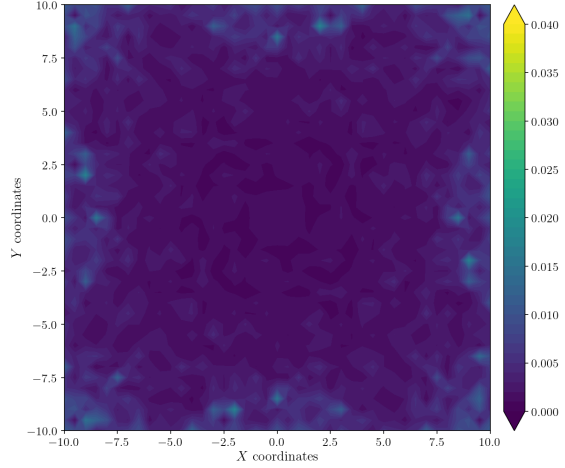


Figure A.117: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.000$ m.

Figure A.118: Camera coverage for test 01 at level $z = 2.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 2.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

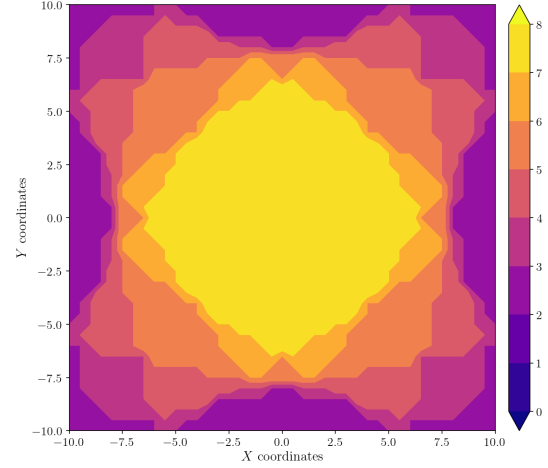


Figure A.119: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.500$ m.

Figure A.120: Camera coverage for test 01 at level $z = 2.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.0$ m level

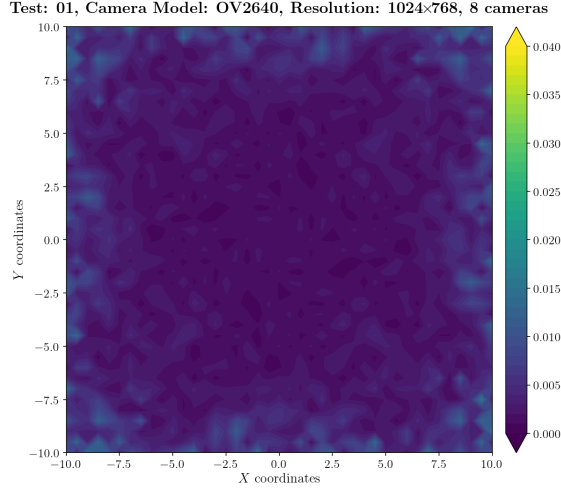


Figure A.121: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.000$ m.

Camera Coverage Spatial Distribution for $z = 3.0$ m level

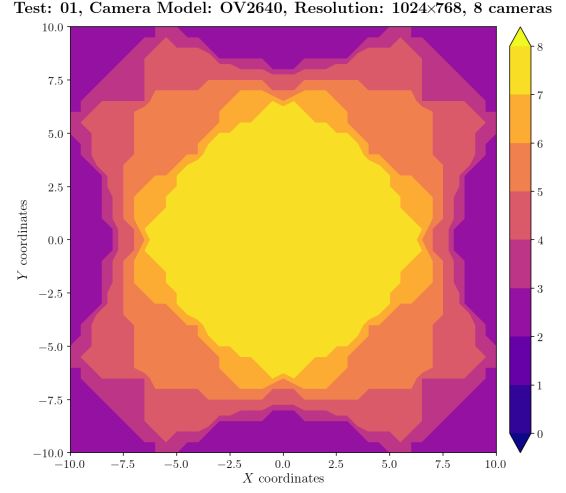


Figure A.122: Camera coverage for test 01 at level $z = 3.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.5$ m level

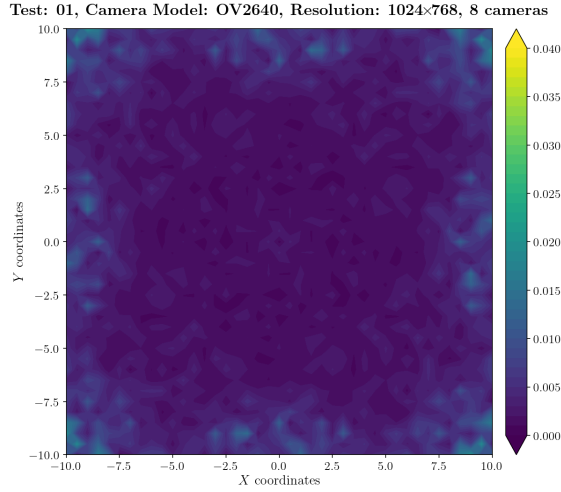


Figure A.123: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.500$ m.

Camera Coverage Spatial Distribution for $z = 3.5$ m level

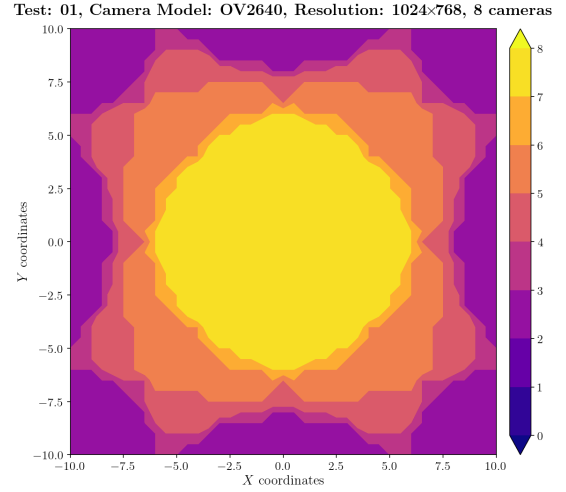
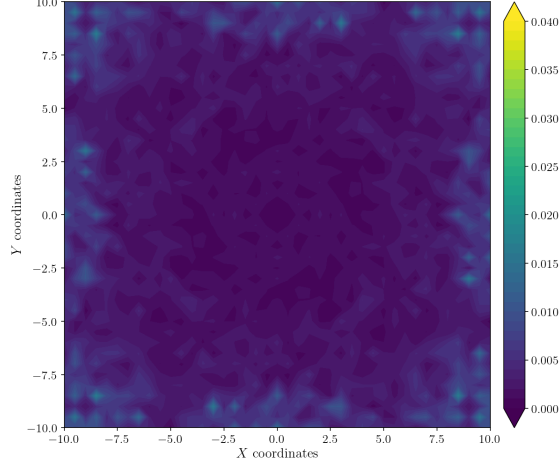


Figure A.124: Camera coverage for test 01 at level $z = 3.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 4.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

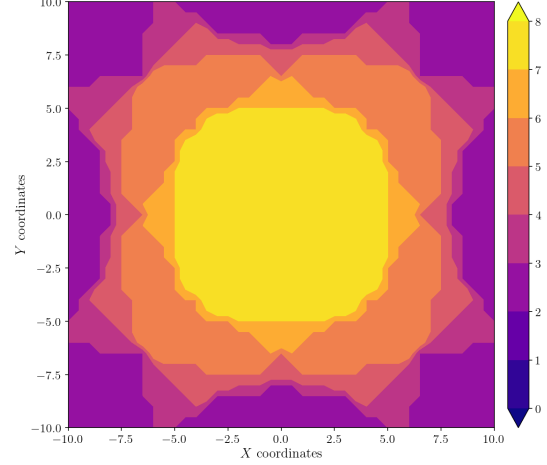
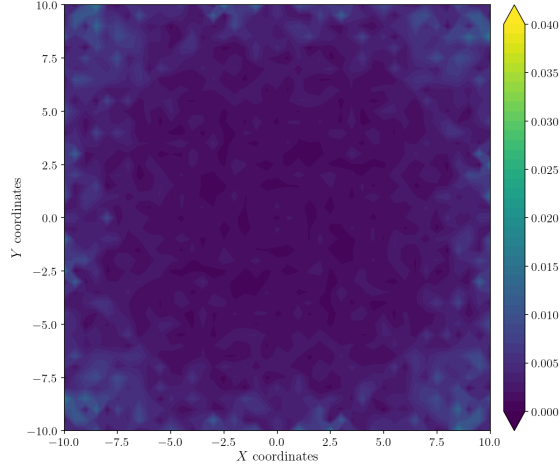


Figure A.125: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.000$ m.

Figure A.126: Camera coverage for test 01 at level $z = 4.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 4.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

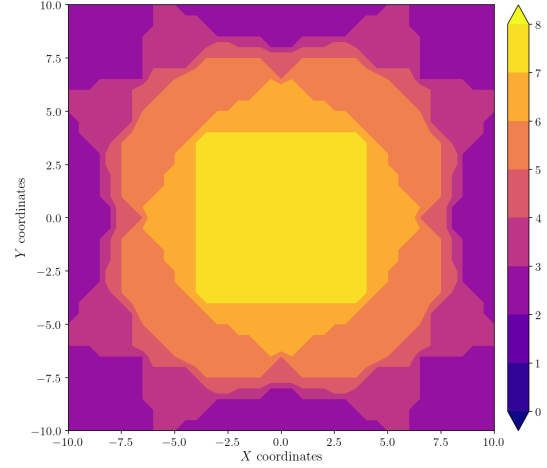


Figure A.127: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.500$ m.

Figure A.128: Camera coverage for test 01 at level $z = 4.500$ m and resolution 1024×768.

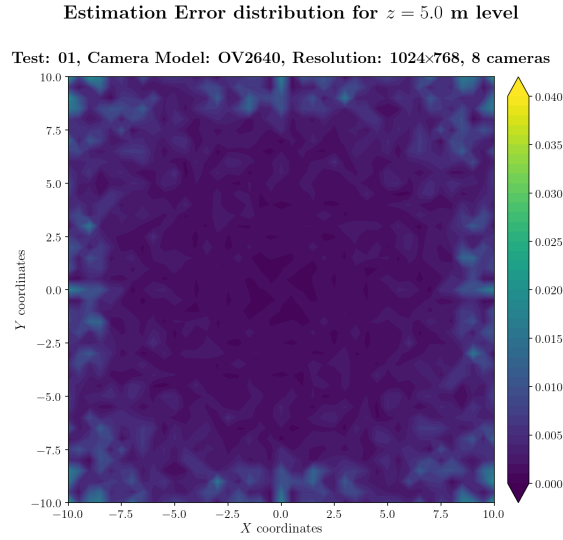


Figure A.129: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 5.000$ m.

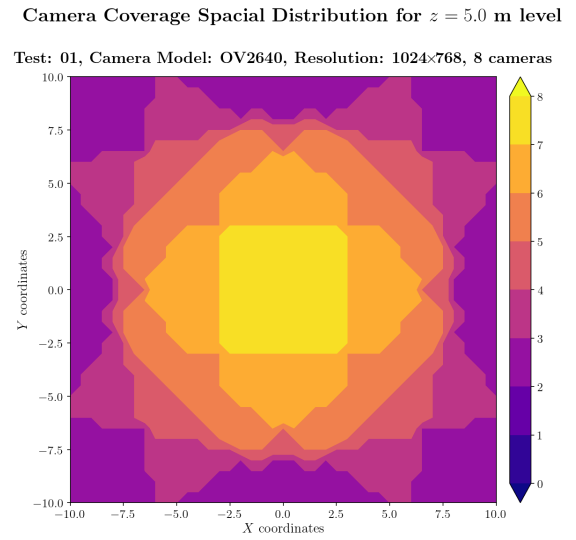


Figure A.130: Camera coverage for test 01 at level $z = 5.000$ m and resolution 1024×768.

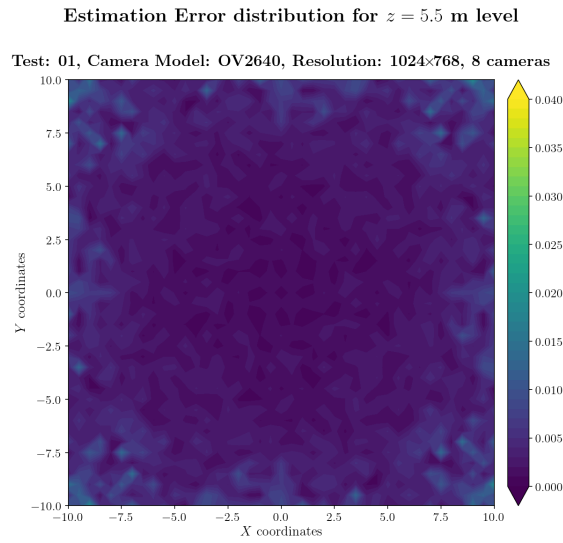


Figure A.131: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 5.500$ m.

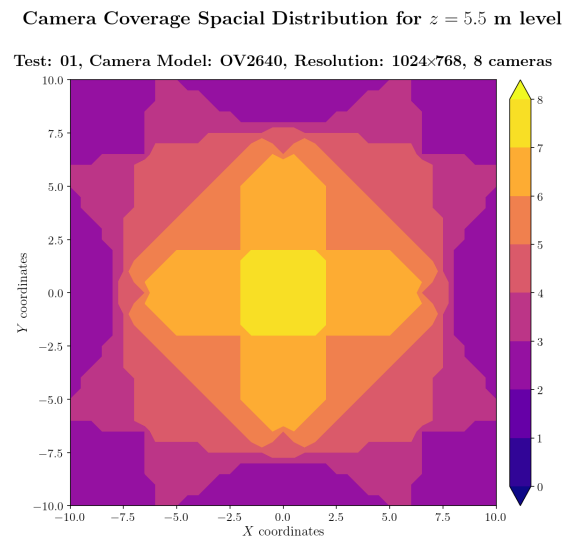
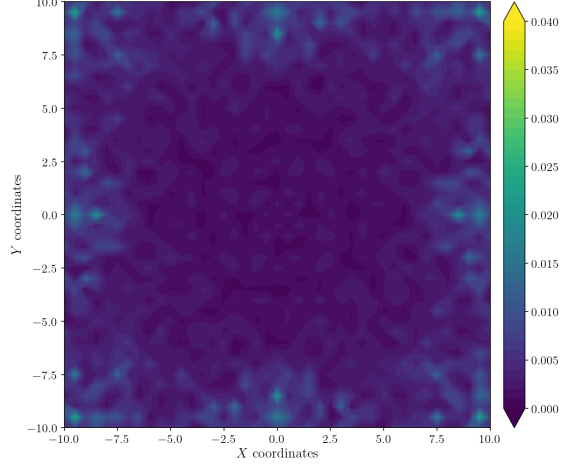


Figure A.132: Camera coverage for test 01 at level $z = 5.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 6.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

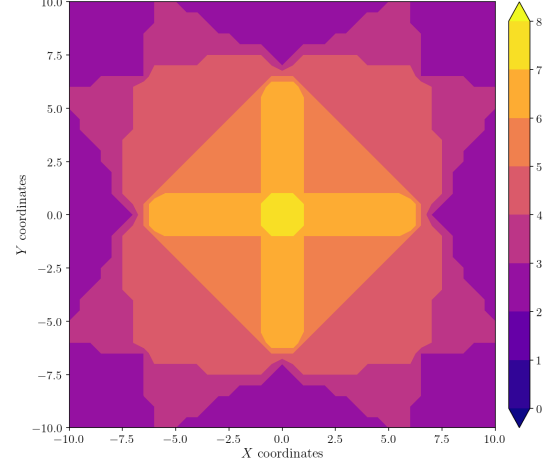
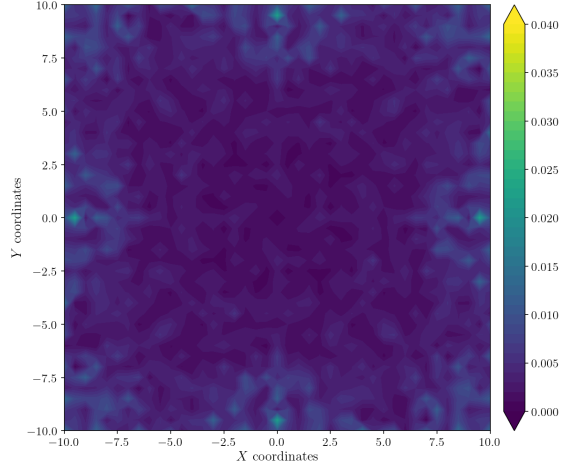


Figure A.133: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 6.000$ m.

Figure A.134: Camera coverage for test 01 at level $z = 6.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 6.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

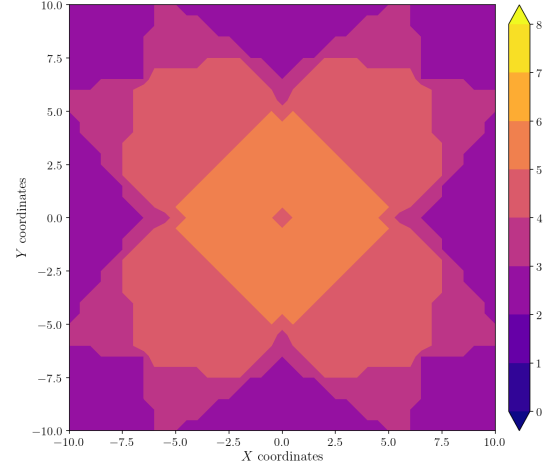


Figure A.135: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 6.500$ m.

Figure A.136: Camera coverage for test 01 at level $z = 6.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 7.0$ m level

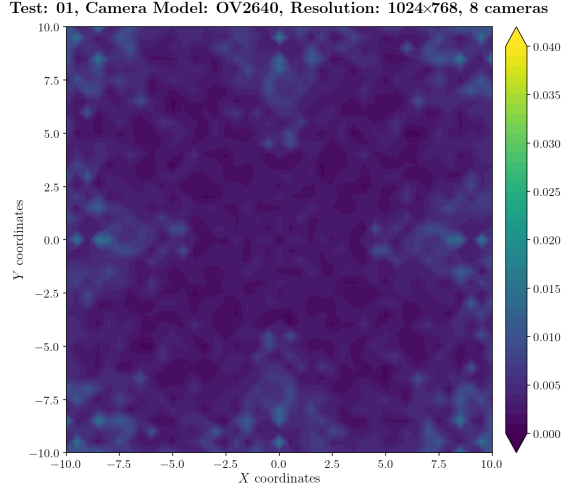


Figure A.137: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 7.000$ m.

Camera Coverage Spacial Distribution for $z = 7.0$ m level

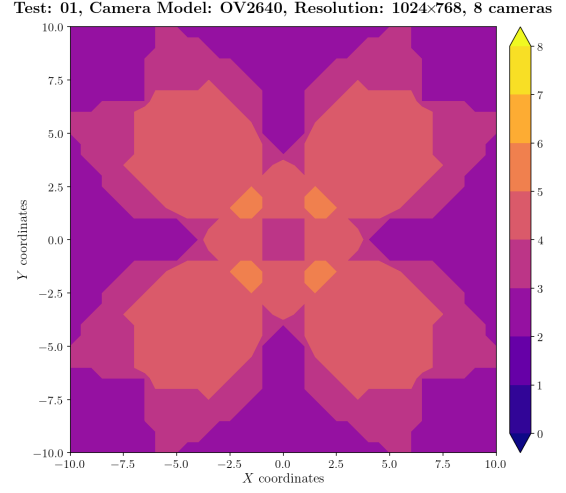


Figure A.138: Camera coverage for test 01 at level $z = 7.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 7.5$ m level

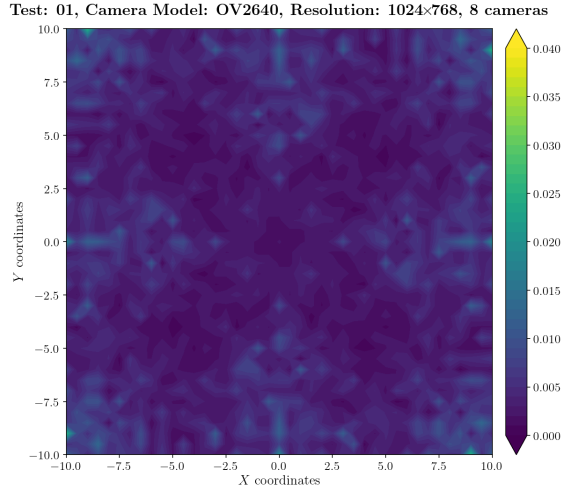


Figure A.139: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 7.500$ m.

Camera Coverage Spacial Distribution for $z = 7.5$ m level

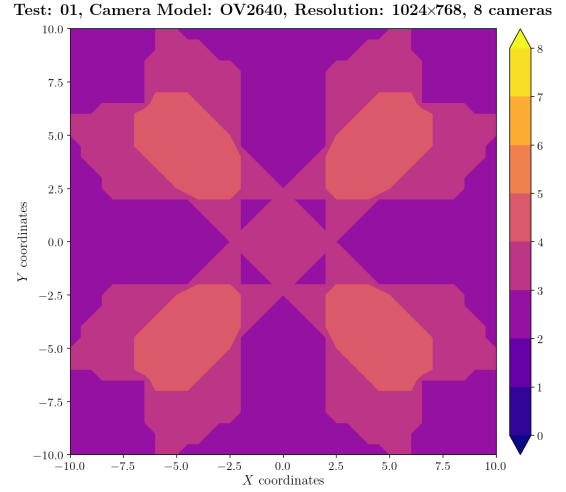
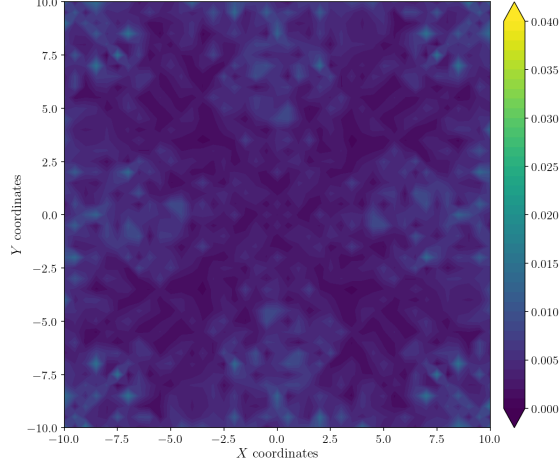


Figure A.140: Camera coverage for test 01 at level $z = 7.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 8.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

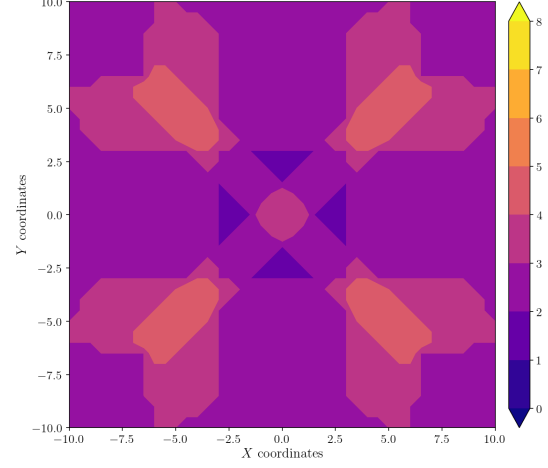
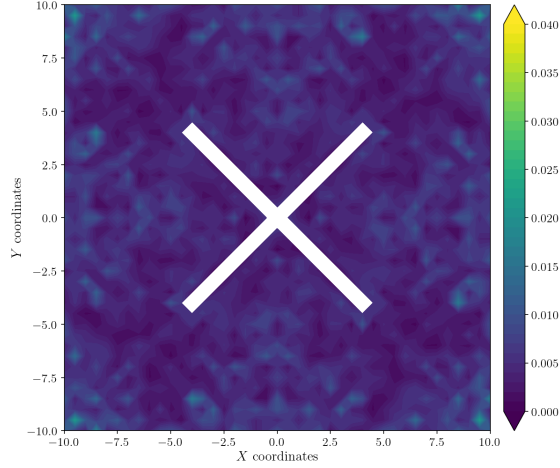


Figure A.141: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 8.000$ m.

Figure A.142: Camera coverage for test 01 at level $z = 8.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 8.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

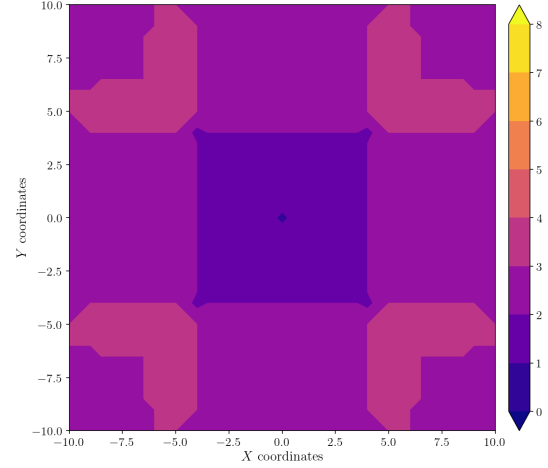


Figure A.143: Estimation error contour map for test 01 with 8 cameras using resolution of 1024×768 pixels, at level $z = 8.500$ m.

Figure A.144: Camera coverage for test 01 at level $z = 8.500$ m and resolution 1024×768.

Error maps for resolution 1600×1200

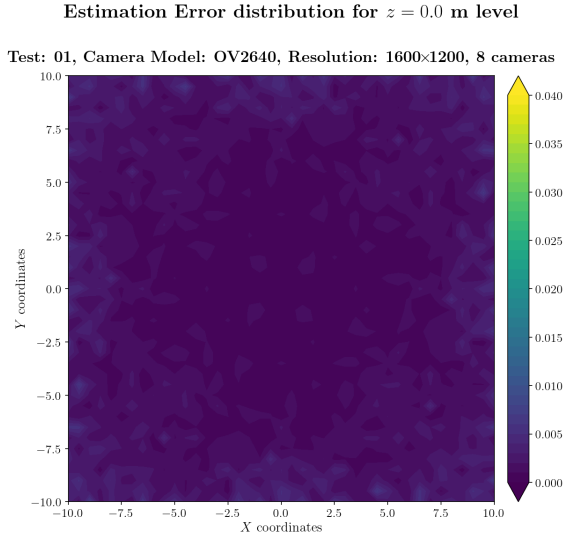


Figure A.145: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.000$ m.

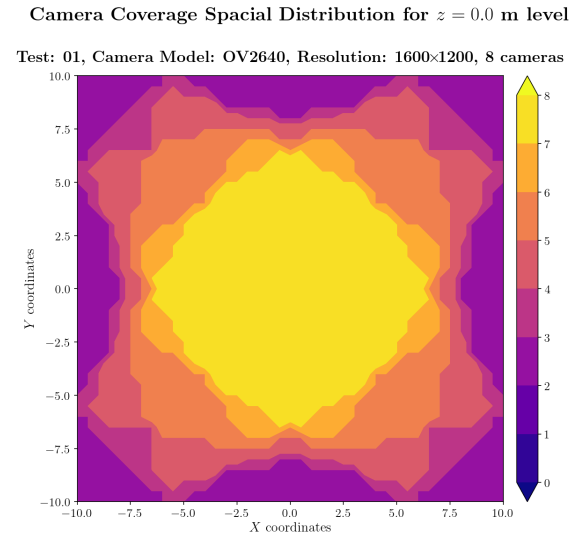


Figure A.146: Camera coverage for test 01 at level $z = 0.000$ m and resolution 1600×1200.

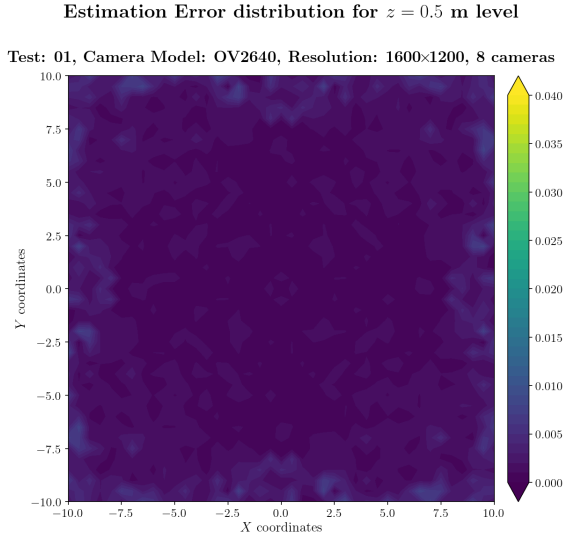


Figure A.147: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.500$ m.

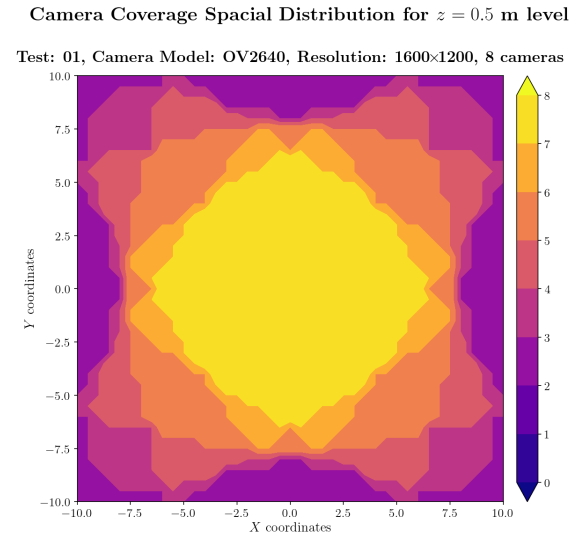


Figure A.148: Camera coverage for test 01 at level $z = 0.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 1.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

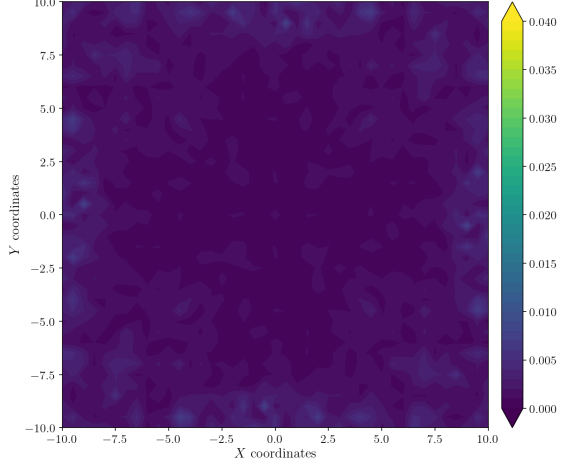


Figure A.149: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.000$ m.

Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

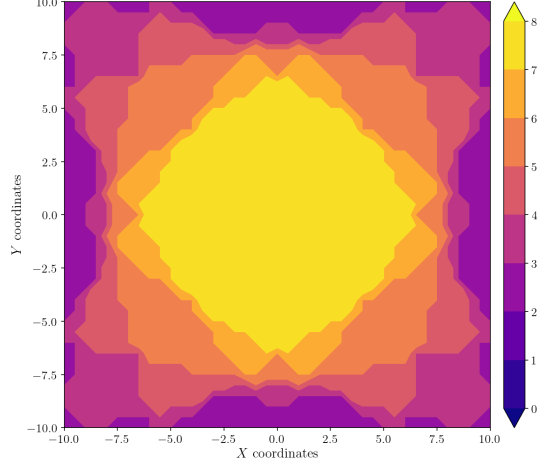


Figure A.150: Camera coverage for test 01 at level $z = 1.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 1.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

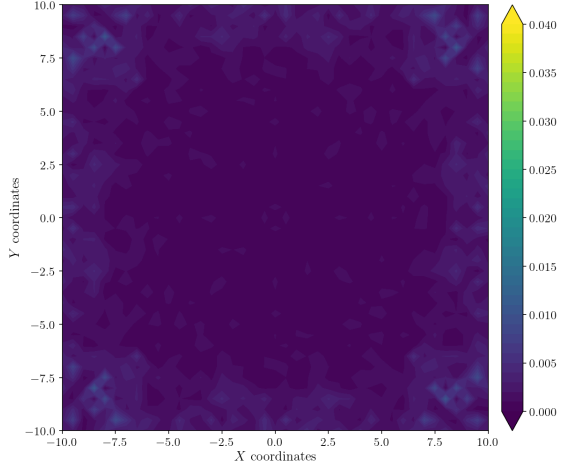


Figure A.151: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.500$ m.

Camera Coverage Spacial Distribution for $z = 1.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

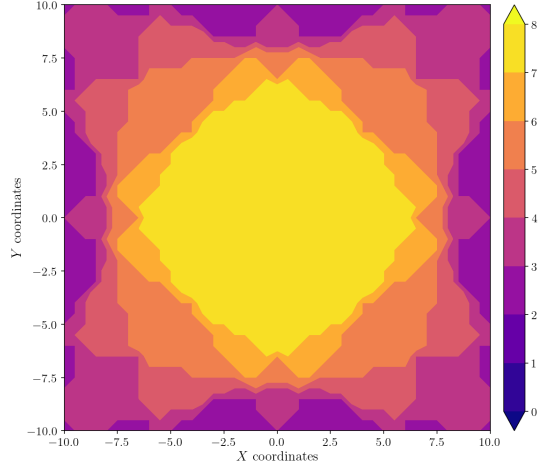


Figure A.152: Camera coverage for test 01 at level $z = 1.500$ m and resolution 1600×1200.

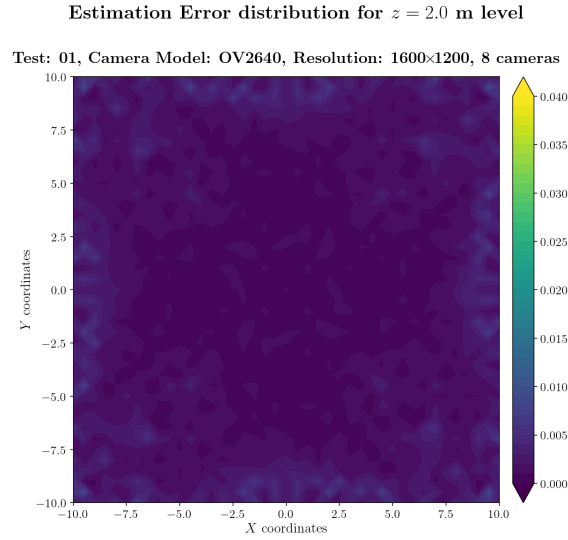


Figure A.153: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.000$ m.

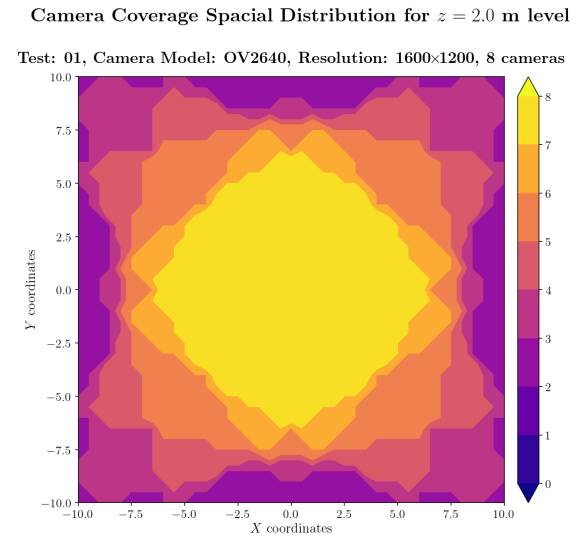


Figure A.154: Camera coverage for test 01 at level $z = 2.000$ m and resolution 1600×1200.

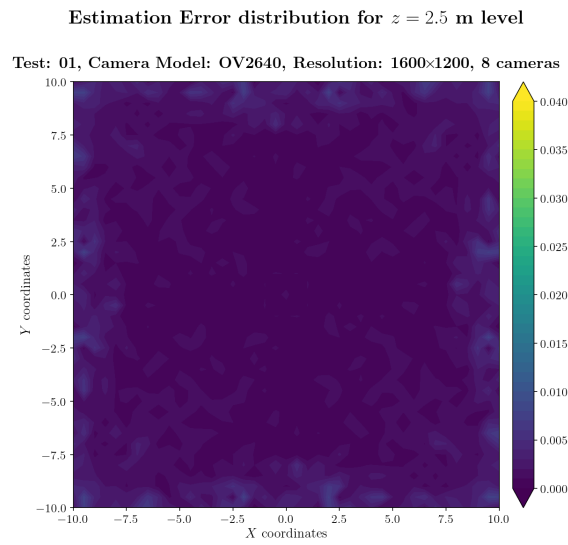


Figure A.155: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.500$ m.

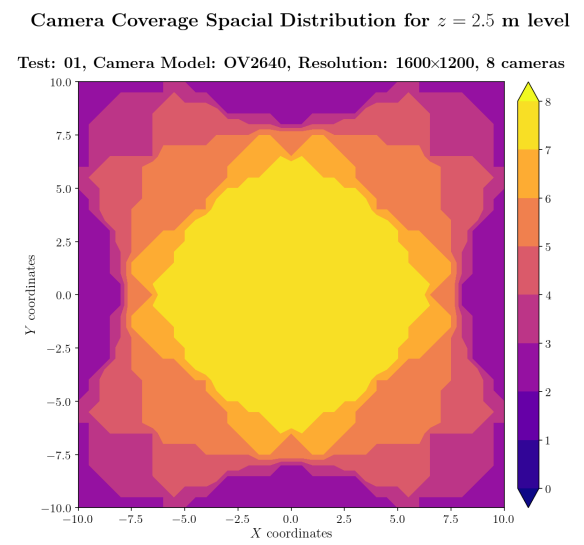
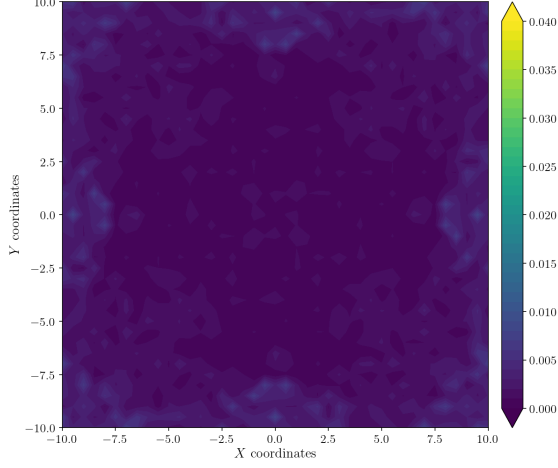


Figure A.156: Camera coverage for test 01 at level $z = 2.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 3.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

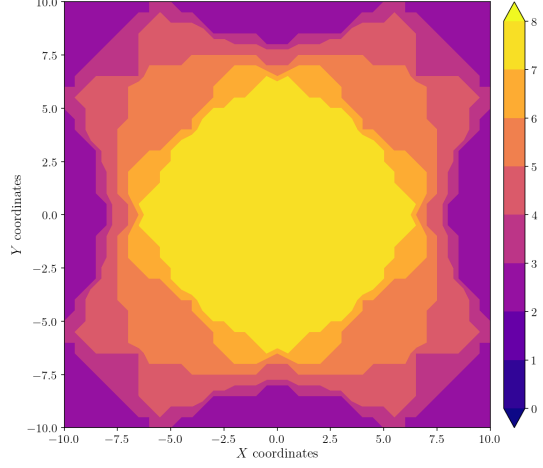
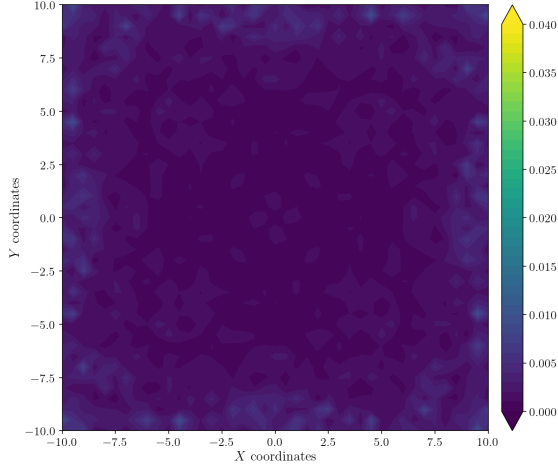


Figure A.157: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.000$ m.

Figure A.158: Camera coverage for test 01 at level $z = 3.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 3.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

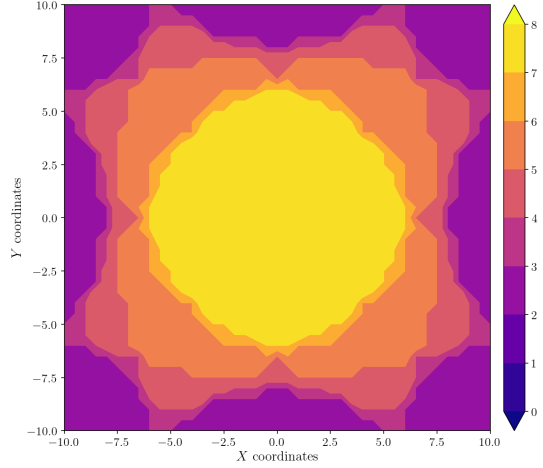


Figure A.159: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.500$ m.

Figure A.160: Camera coverage for test 01 at level $z = 3.500$ m and resolution 1600×1200.

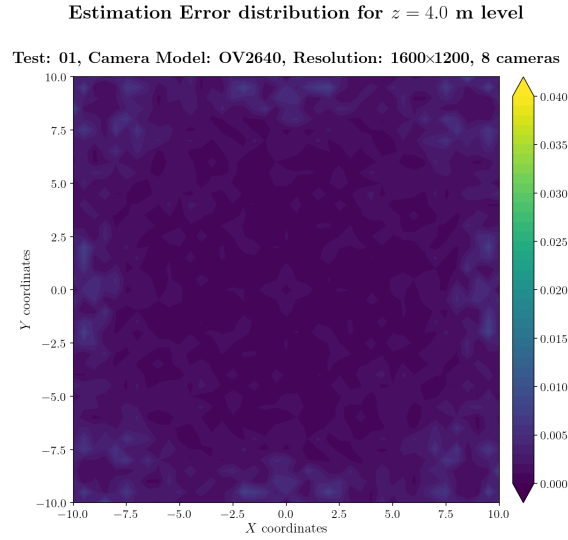


Figure A.161: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.000$ m.

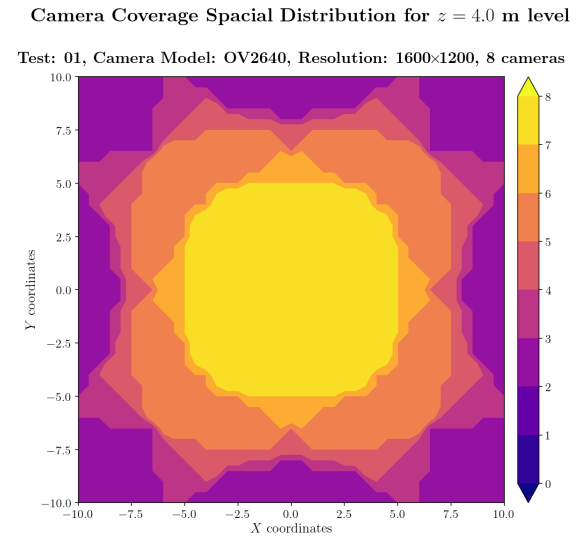


Figure A.162: Camera coverage for test 01 at level $z = 4.000$ m and resolution 1600×1200.

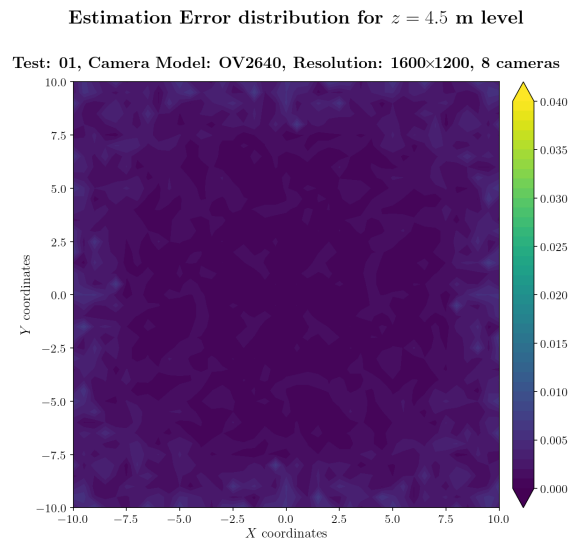


Figure A.163: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.500$ m.

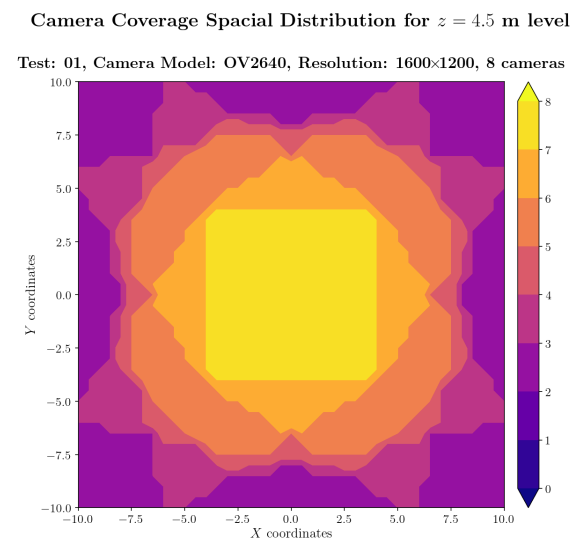
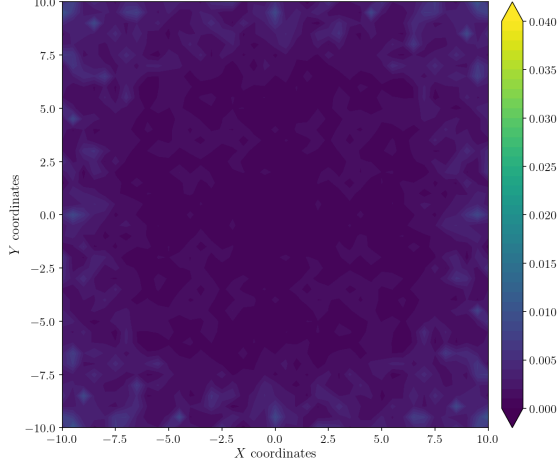


Figure A.164: Camera coverage for test 01 at level $z = 4.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 5.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

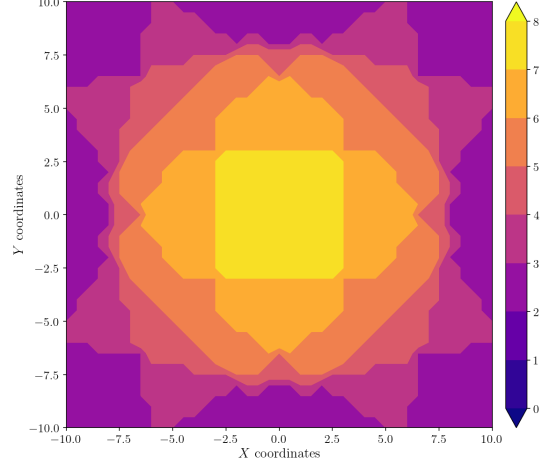
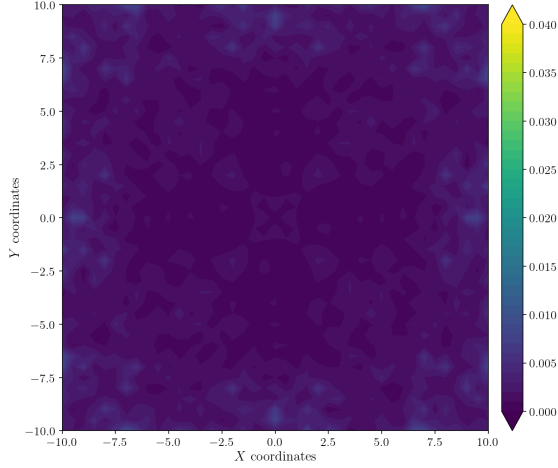


Figure A.165: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 5.000$ m.

Figure A.166: Camera coverage for test 01 at level $z = 5.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 5.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

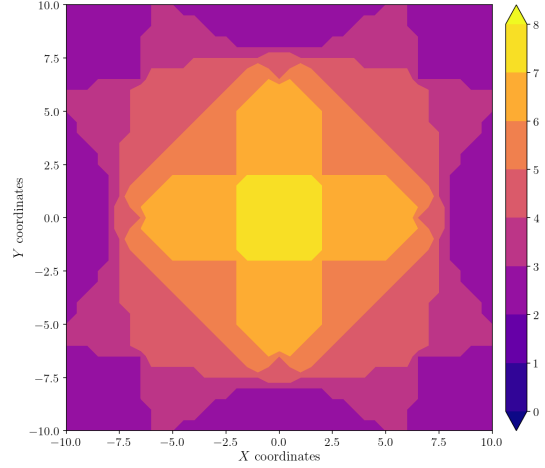


Figure A.167: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 5.500$ m.

Figure A.168: Camera coverage for test 01 at level $z = 5.500$ m and resolution 1600×1200.

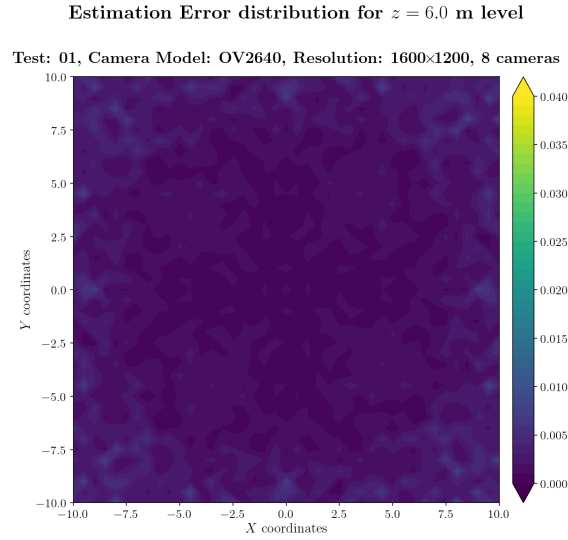


Figure A.169: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 6.000$ m.

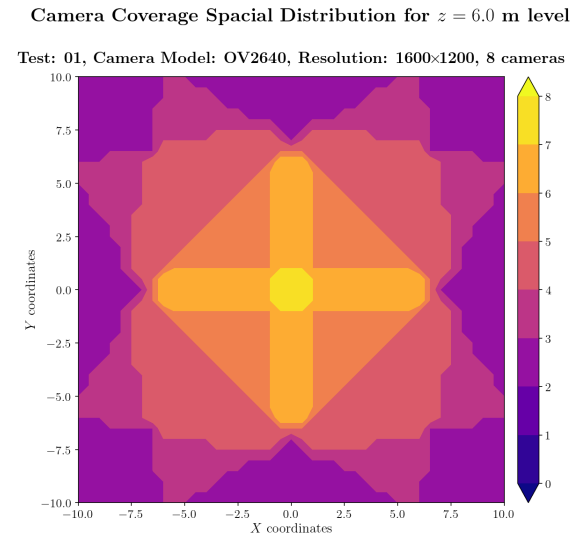


Figure A.170: Camera coverage for test 01 at level $z = 6.000$ m and resolution 1600×1200.

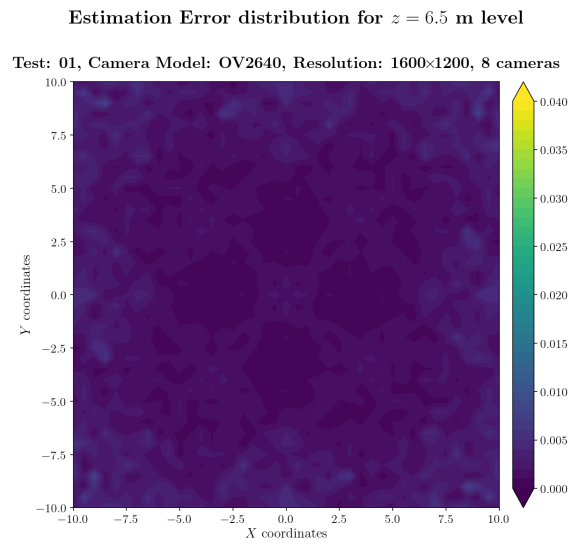


Figure A.171: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 6.500$ m.

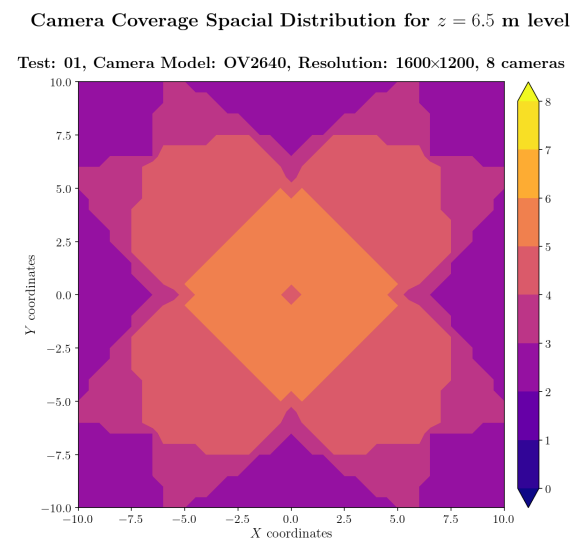
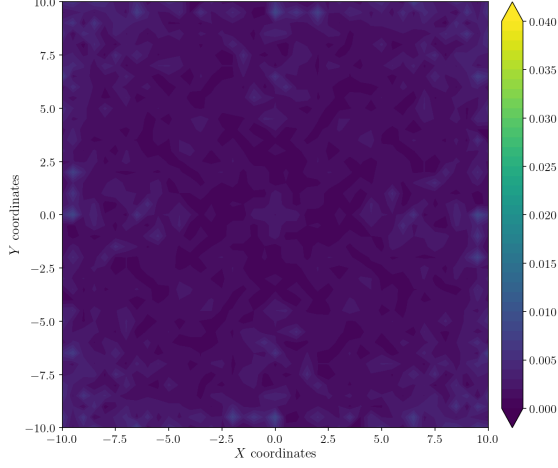


Figure A.172: Camera coverage for test 01 at level $z = 6.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 7.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.0$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

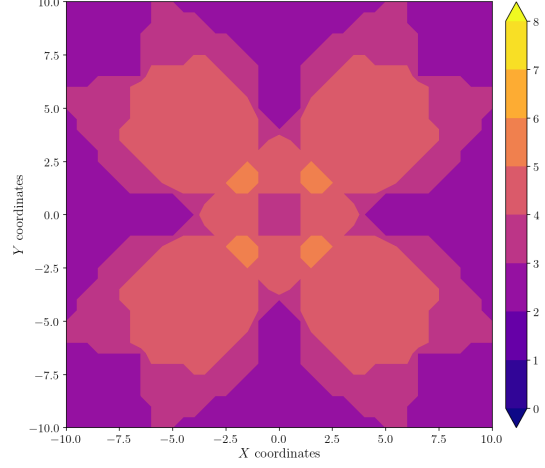
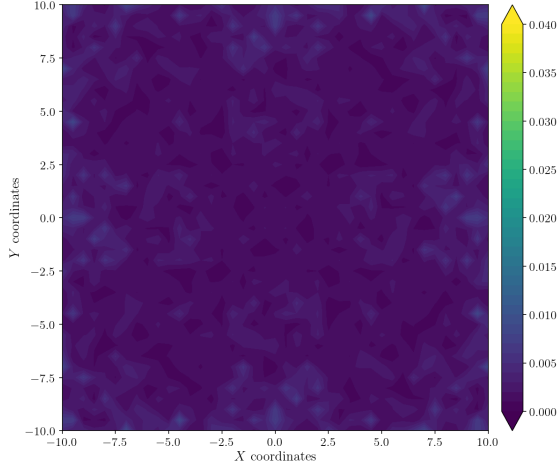


Figure A.173: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 7.000$ m.

Figure A.174: Camera coverage for test 01 at level $z = 7.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 7.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.5$ m level

Test: 01, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

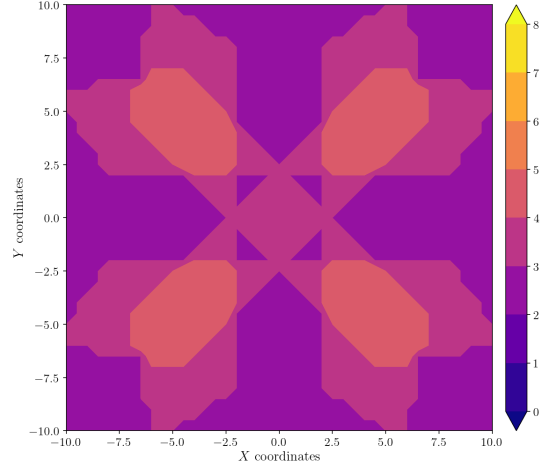


Figure A.175: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 7.500$ m.

Figure A.176: Camera coverage for test 01 at level $z = 7.500$ m and resolution 1600×1200.

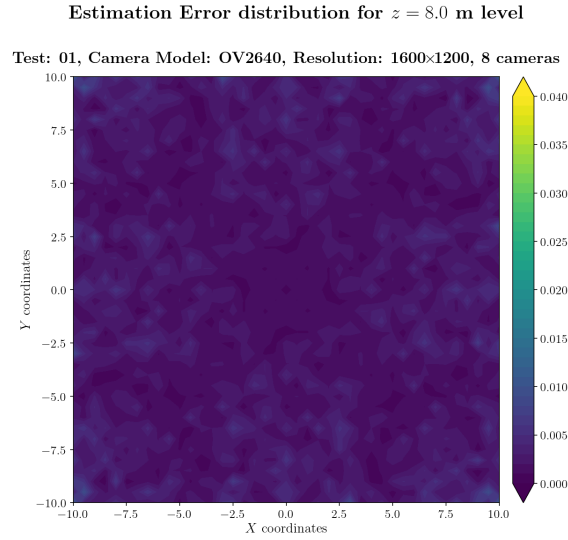


Figure A.177: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 8.000$ m.

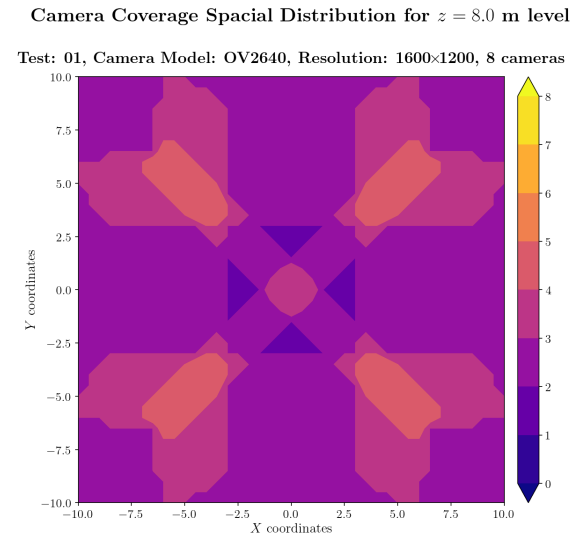


Figure A.178: Camera coverage for test 01 at level $z = 8.000$ m and resolution 1600×1200.

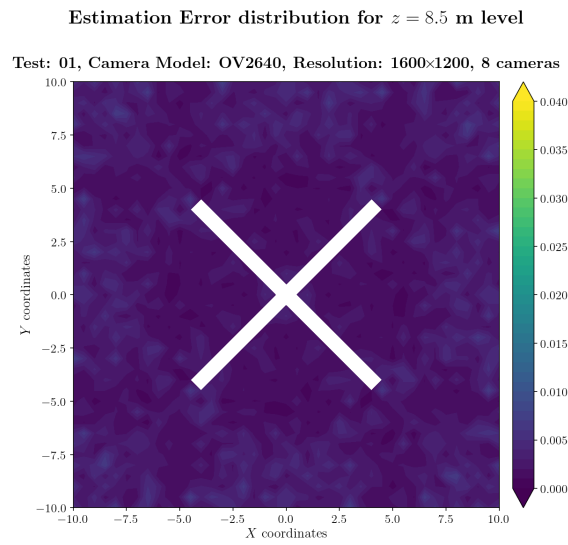


Figure A.179: Estimation error contour map for test 01 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 8.500$ m.

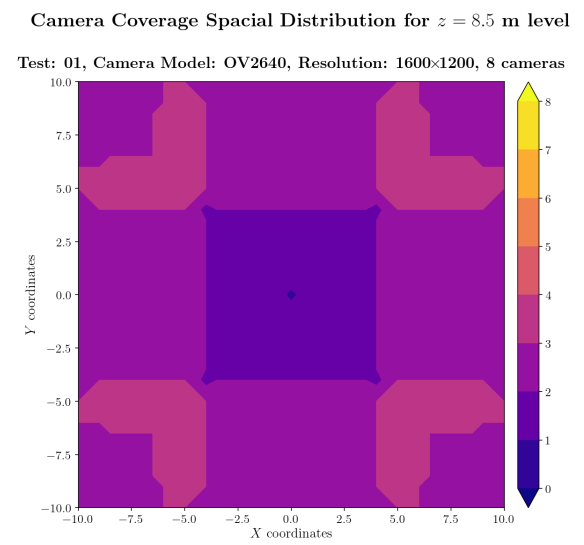
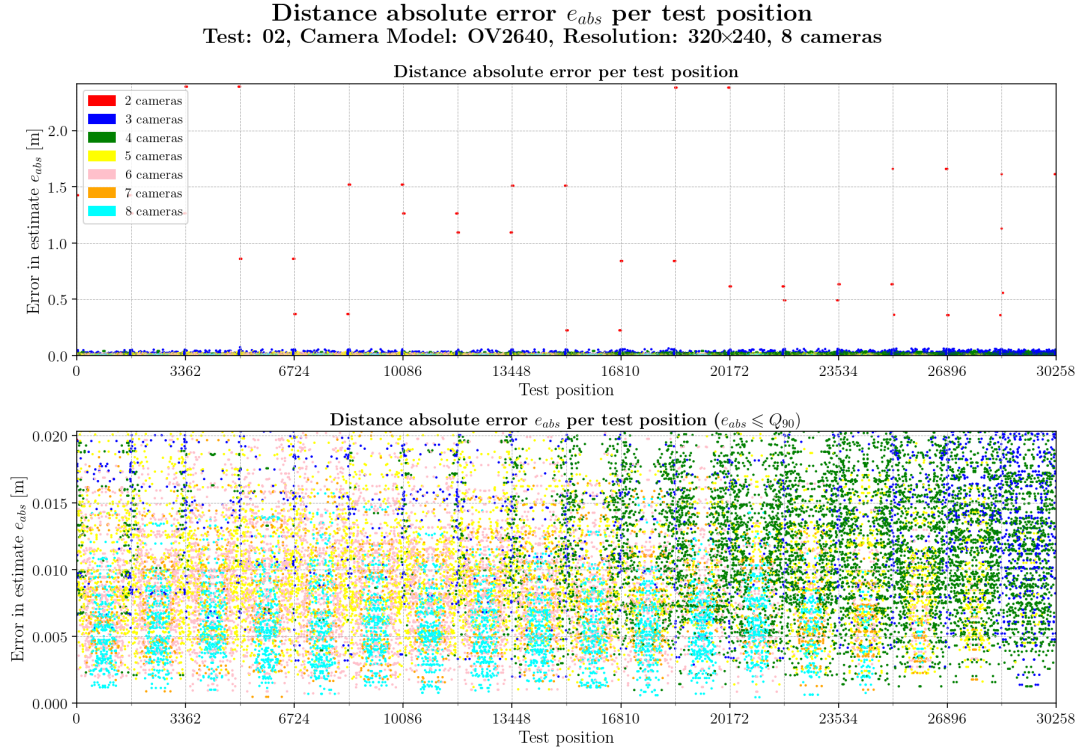


Figure A.180: Camera coverage for test 01 at level $z = 8.500$ m and resolution 1600×1200.

A.2 Test 02

Absolute Error per iteration and resolution

Resolution: 320×240



Error per iteration for resolution 320×240 and test02

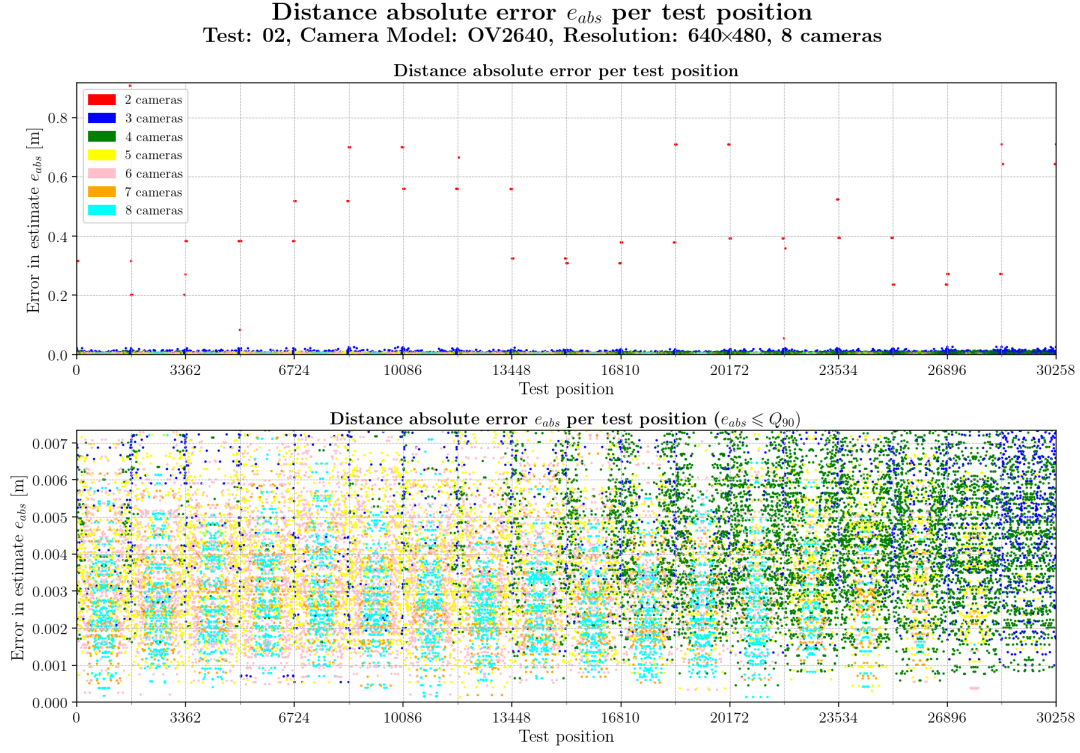
Table A.11: Statistics per number of cameras observing the beacon for test 02 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	80	3628	8061	5444	6548	1900	4597
mean	1.0074	0.0222	0.0132	0.0103	0.0090	0.0079	0.0058
std	0.6843	0.0119	0.0061	0.0048	0.0043	0.0038	0.0026
min	0.0197	0.0014	0.0010	0.0010	0.0007	0.0005	0.0004
25%	0.4612	0.0135	0.0089	0.0070	0.0059	0.0049	0.0039
50%	0.8594	0.0200	0.0127	0.0095	0.0084	0.0074	0.0056
75%	1.5088	0.0289	0.0167	0.0129	0.0114	0.0102	0.0074
max	2.3882	0.0739	0.0427	0.0391	0.0344	0.0215	0.0157

Table A.12: Global statistics and root mean square error for test 02 with resolution 320×240.

	Abs. error	Est. error
count	30258	30258
mean	0.0140	0.0290
std	0.0624	0.0096
min	0.0004	0.0017
25%	0.0064	0.0224
50%	0.0096	0.0283
75%	0.0144	0.0349
90%	0.0203	0.0414
max	2.3882	0.0792
RMSE	0.0640	

Resolution: 640×480



Error per iteration for resolution 640×480 and test02

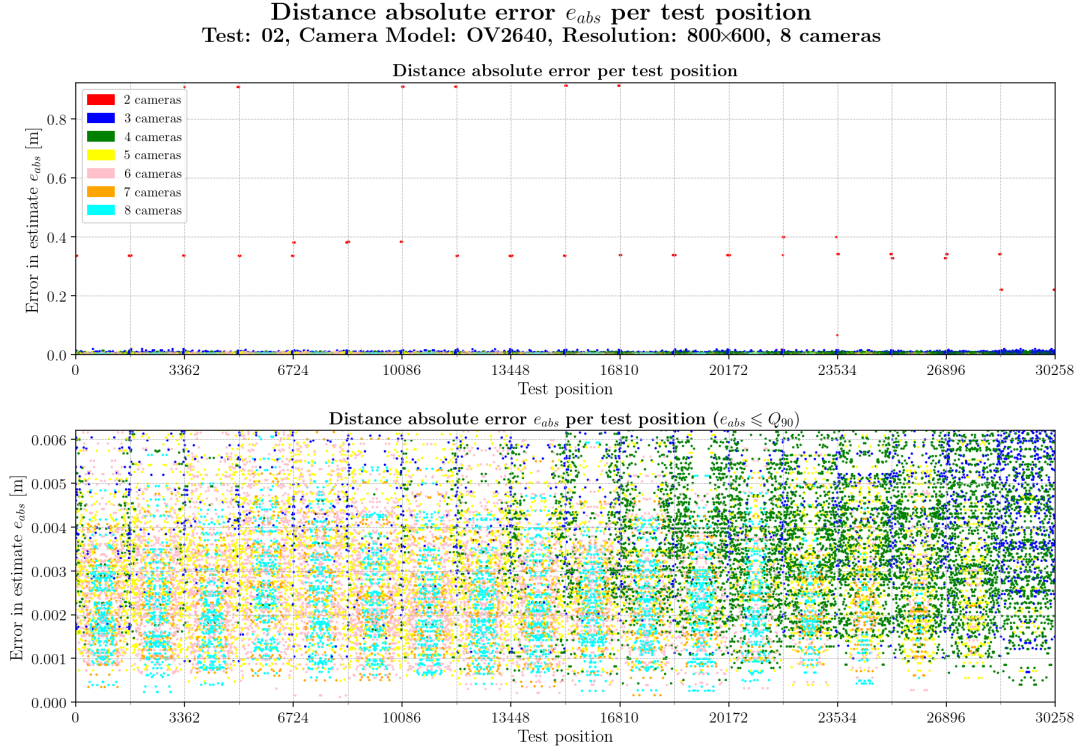
Table A.13: Statistics per number of cameras observing the beacon for test 02 with resolution 640×480.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	72	3436	8165	5488	6508	1924	4665
mean	0.4326	0.0076	0.0049	0.0039	0.0035	0.0029	0.0027
std	0.1750	0.0039	0.0020	0.0017	0.0016	0.0013	0.0012
min	0.0550	0.0005	0.0005	0.0001	0.0002	0.0002	0.0001
25%	0.3136	0.0047	0.0034	0.0026	0.0023	0.0020	0.0018
50%	0.3826	0.0071	0.0047	0.0037	0.0033	0.0027	0.0026
75%	0.5588	0.0098	0.0061	0.0049	0.0045	0.0036	0.0034
max	0.9070	0.0262	0.0151	0.0108	0.0130	0.0088	0.0087

Table A.14: Global statistics and root mean square error for test 02 with resolution 640×480.

	Abs. error	Est. error
count	30258	30258
mean	0.0053	0.0109
std	0.0227	0.0033
min	0.0001	0.0005
25%	0.0026	0.0087
50%	0.0037	0.0109
75%	0.0053	0.0132
90%	0.0073	0.0152
max	0.9070	0.0249
RMSE	0.0233	

Resolution: 800×600



Error per iteration for resolution 800×600 and test02

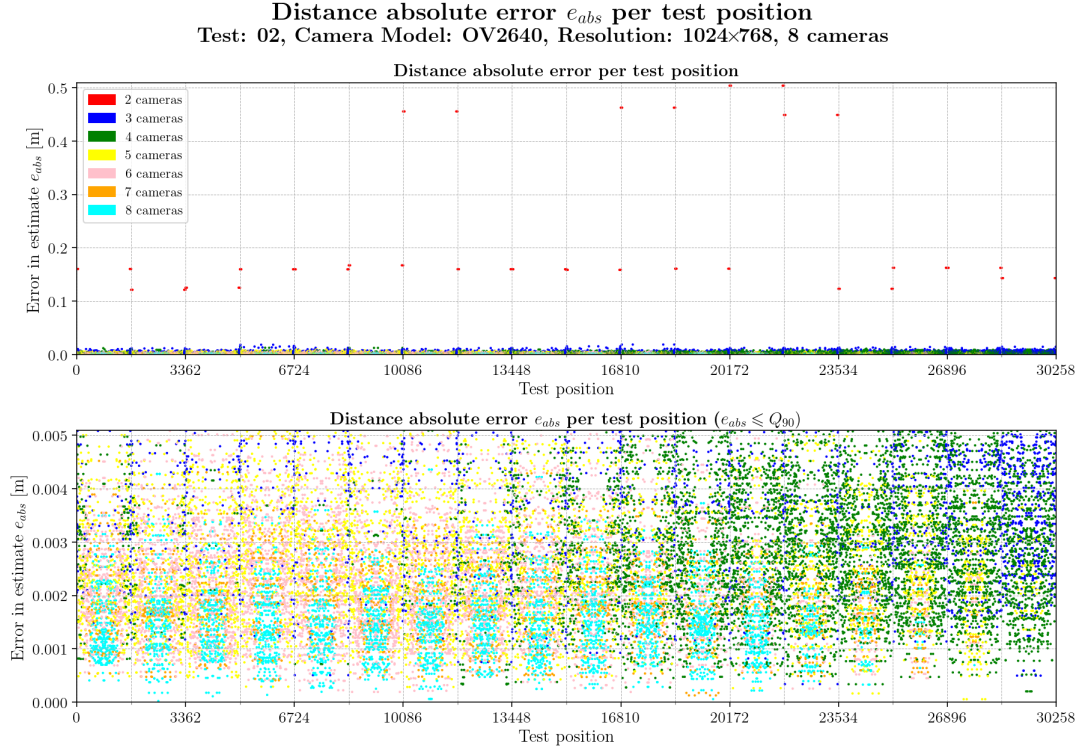
Table A.15: Statistics per number of cameras observing the beacon for test 02 with resolution 800×600.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	72	3420	8137	5524	6440	1996	4669
mean	0.4208	0.0062	0.0041	0.0032	0.0029	0.0024	0.0023
std	0.2201	0.0030	0.0018	0.0015	0.0015	0.0011	0.0011
min	0.0170	0.0007	0.0004	0.0004	0.0001	0.0002	0.0002
25%	0.3356	0.0040	0.0028	0.0021	0.0019	0.0016	0.0015
50%	0.3378	0.0058	0.0039	0.0030	0.0027	0.0023	0.0022
75%	0.3830	0.0079	0.0052	0.0041	0.0036	0.0030	0.0029
max	0.9131	0.0190	0.0135	0.0097	0.0101	0.0065	0.0096

Table A.16: Global statistics and root mean square error for test 02 with resolution 800×600.

	Abs. error	Est. error
count	30258	30258
mean	0.0045	0.0092
std	0.0231	0.0029
min	0.0001	0.0003
25%	0.0021	0.0072
50%	0.0031	0.0090
75%	0.0044	0.0111
90%	0.0062	0.0129
max	0.9131	0.0299
RMSE	0.0235	

Resolution: 1024×768



Error per iteration for resolution 1024×768 and test02

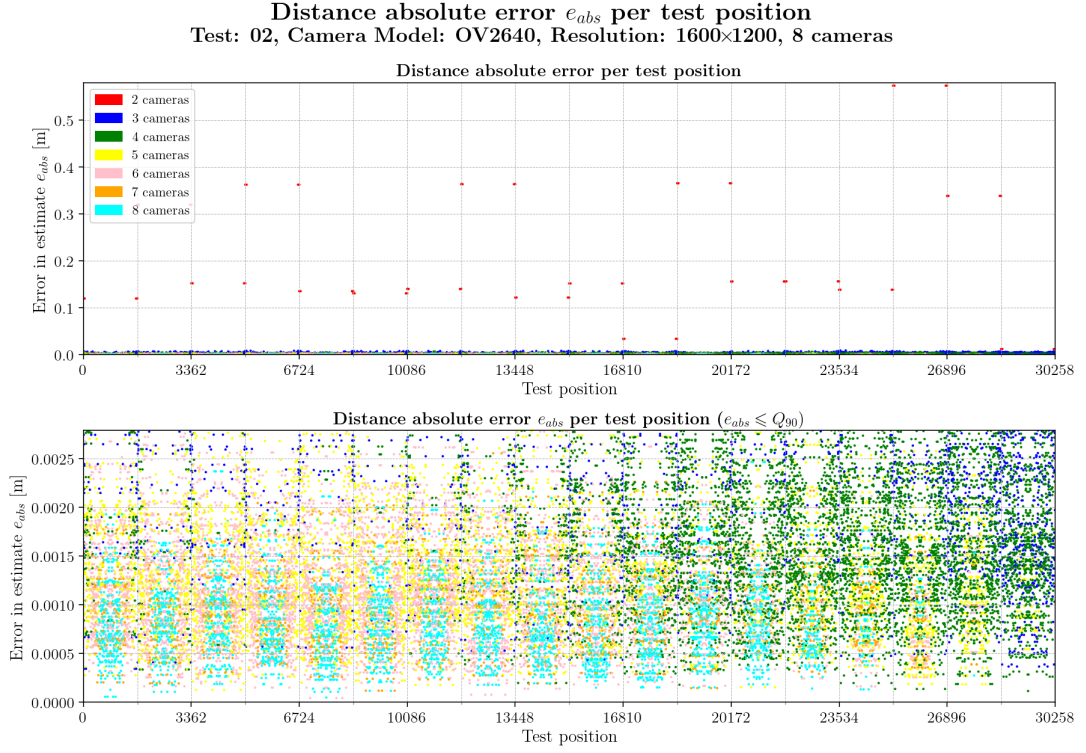
Table A.17: Statistics per number of cameras observing the beacon for test 02 with resolution 1024×768.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	72	3408	8133	5508	6444	2024	4669
mean	0.2219	0.0050	0.0033	0.0028	0.0023	0.0019	0.0015
std	0.1333	0.0026	0.0016	0.0014	0.0011	0.0008	0.0007
min	0.1213	0.0003	0.0002	0.0001	0.0002	0.0001	0.0000
25%	0.1586	0.0032	0.0022	0.0018	0.0016	0.0013	0.0010
50%	0.1601	0.0046	0.0031	0.0026	0.0022	0.0018	0.0014
75%	0.1672	0.0062	0.0042	0.0036	0.0030	0.0023	0.0019
max	0.5034	0.0188	0.0128	0.0096	0.0072	0.0050	0.0044

Table A.18: Global statistics and root mean square error for test 02 with resolution 1024×768.

	Abs. error	Est. error
count	30258	30258
mean	0.0034	0.0073
std	0.0126	0.0024
min	0.0000	0.0003
25%	0.0016	0.0057
50%	0.0024	0.0072
75%	0.0036	0.0089
90%	0.0051	0.0105
max	0.5034	0.0164
RMSE	0.0130	

Resolution: 1600×1200



Error per iteration for resolution 1600×1200 and test02

Table A.19: Statistics per number of cameras observing the beacon for test 02 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	72	3360	8145	5448	6532	2020	4681
mean	0.2094	0.0031	0.0018	0.0014	0.0012	0.0010	0.0008
std	0.1409	0.0015	0.0009	0.0006	0.0005	0.0004	0.0004
min	0.0118	0.0003	0.0001	0.0001	0.0000	0.0001	0.0000
25%	0.1306	0.0020	0.0012	0.0009	0.0008	0.0006	0.0005
50%	0.1519	0.0029	0.0017	0.0013	0.0011	0.0009	0.0008
75%	0.3384	0.0040	0.0023	0.0018	0.0015	0.0012	0.0011
max	0.5734	0.0089	0.0066	0.0042	0.0047	0.0029	0.0031

Table A.20: Global statistics and root mean square error for test 02 with resolution 1600×1200.

	Abs. error	Est. error
count	30258	30258
mean	0.0020	0.0040
std	0.0123	0.0013
min	0.0000	0.0004
25%	0.0008	0.0031
50%	0.0013	0.0039
75%	0.0019	0.0047
90%	0.0028	0.0056
max	0.5734	0.0149
RMSE	0.0124	

Error and camera coverage maps

Error maps for resolution 320×240

Estimation Error distribution for $z = 0.0$ m level

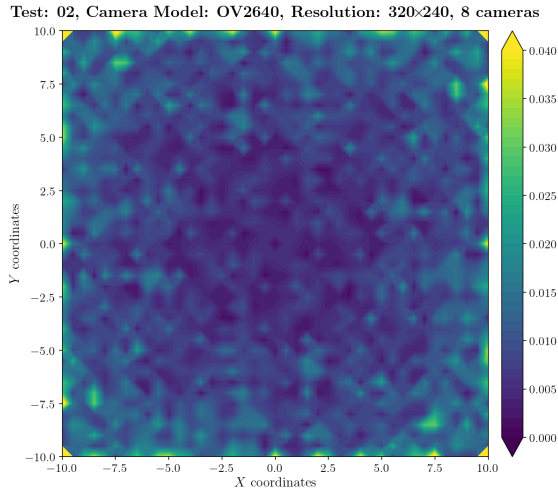


Figure A.181: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

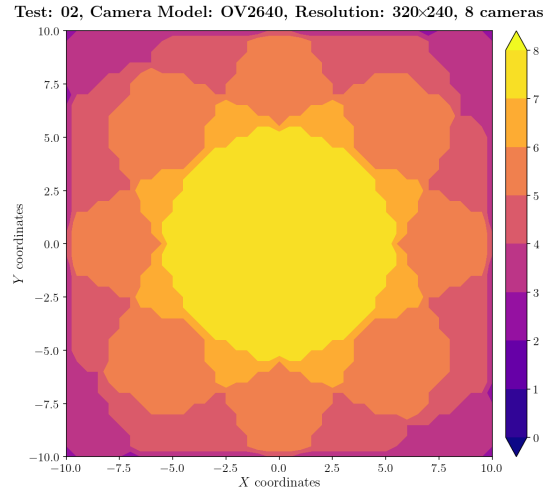


Figure A.182: Camera coverage for test 02 at level $z = 0.000$ m and resolution 320×240.

Estimation Error distribution for $z = 0.5$ m level

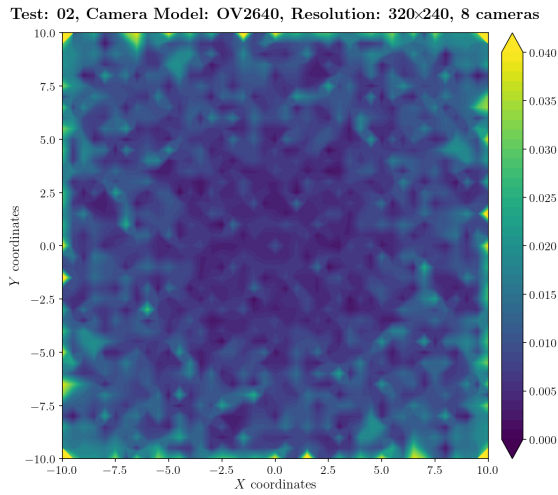


Figure A.183: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.500$ m.

Camera Coverage Spacial Distribution for $z = 0.5$ m level

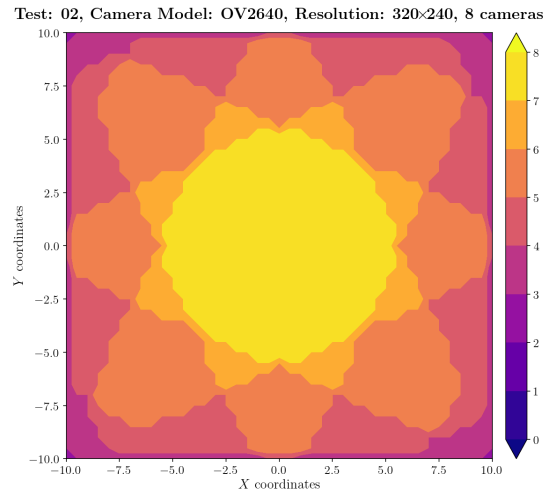


Figure A.184: Camera coverage for test 02 at level $z = 0.500$ m and resolution 320×240.

Estimation Error distribution for $z = 1.0$ m level

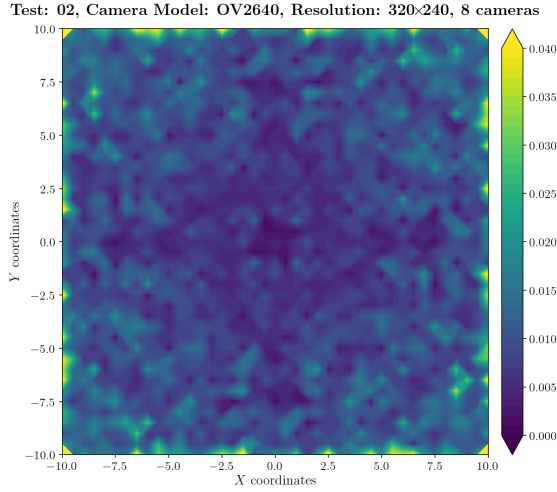


Figure A.185: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.000$ m.

Camera Coverage Spatial Distribution for $z = 1.0$ m level

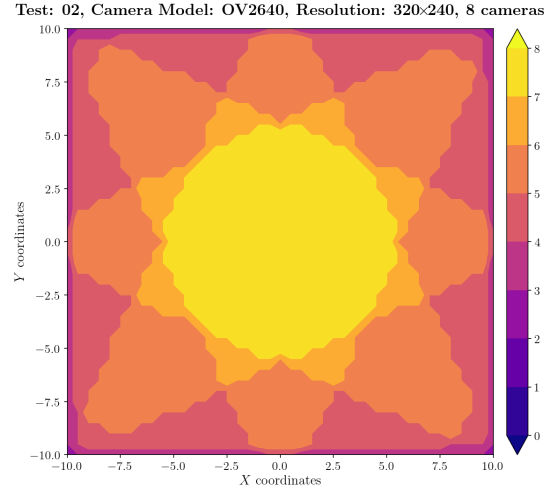


Figure A.186: Camera coverage for test 02 at level $z = 1.000$ m and resolution 320×240.

Estimation Error distribution for $z = 1.5$ m level

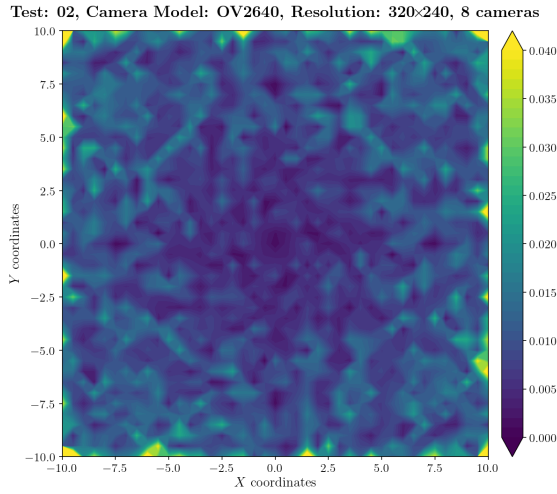


Figure A.187: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.500$ m.

Camera Coverage Spatial Distribution for $z = 1.5$ m level

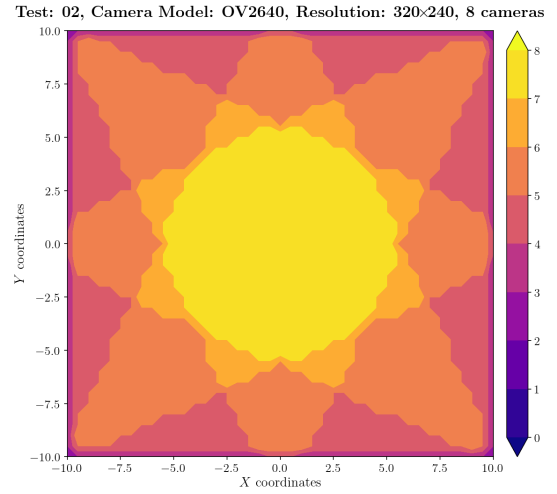
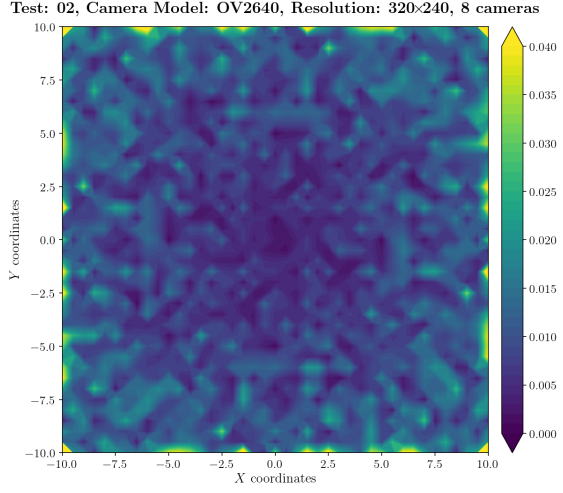


Figure A.188: Camera coverage for test 02 at level $z = 1.500$ m and resolution 320×240.

Estimation Error distribution for $z = 2.0$ m level



Camera Coverage Spacial Distribution for $z = 2.0$ m level

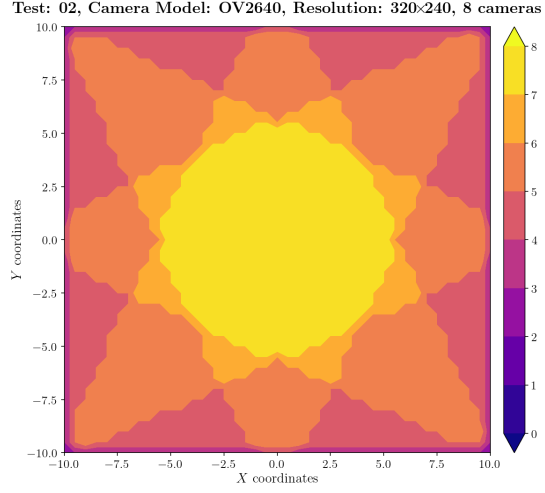
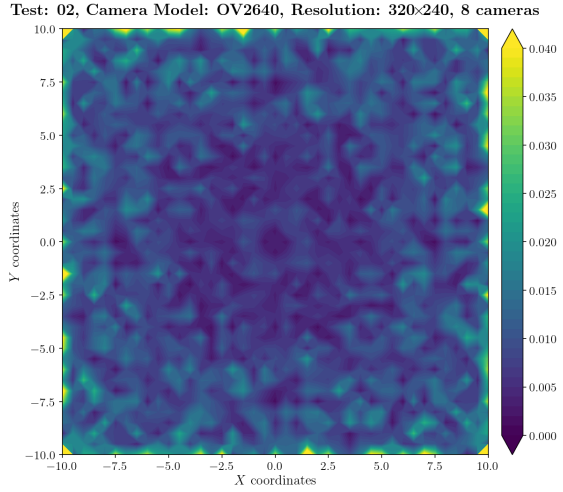


Figure A.189: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.000$ m.

Figure A.190: Camera coverage for test 02 at level $z = 2.000$ m and resolution 320×240.

Estimation Error distribution for $z = 2.5$ m level



Camera Coverage Spacial Distribution for $z = 2.5$ m level

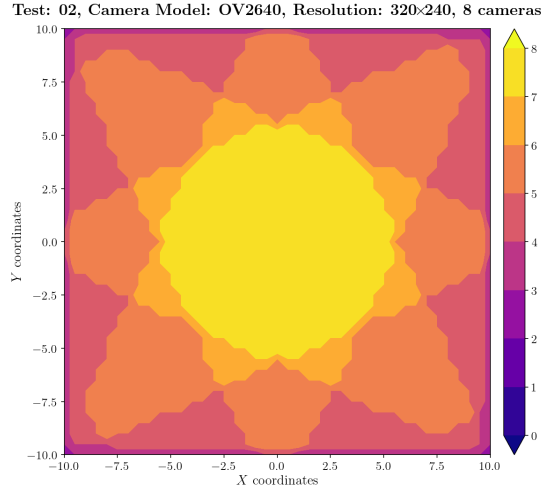


Figure A.191: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.500$ m.

Figure A.192: Camera coverage for test 02 at level $z = 2.500$ m and resolution 320×240.

Estimation Error distribution for $z = 3.0$ m level

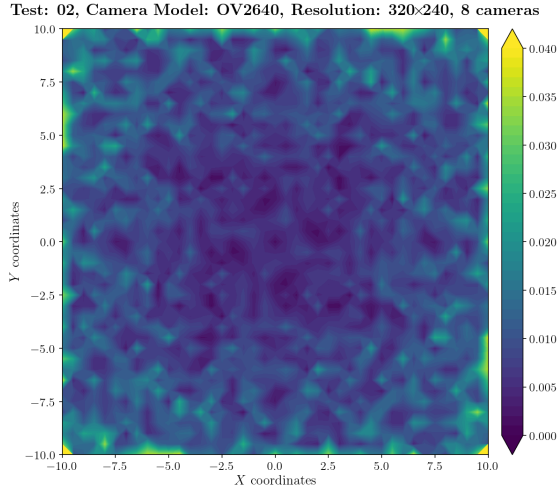


Figure A.193: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.000$ m.

Camera Coverage Spatial Distribution for $z = 3.0$ m level

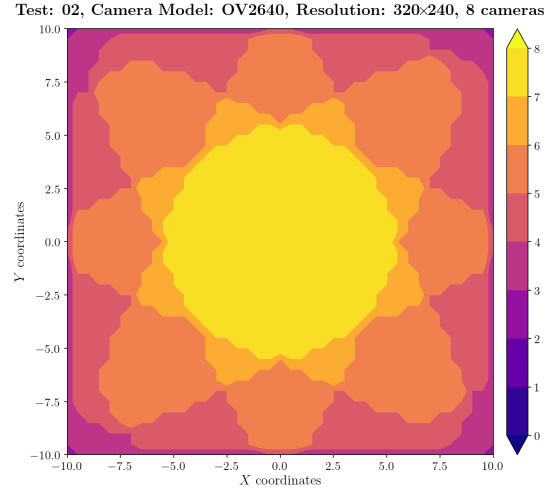


Figure A.194: Camera coverage for test 02 at level $z = 3.000$ m and resolution 320×240.

Estimation Error distribution for $z = 3.5$ m level

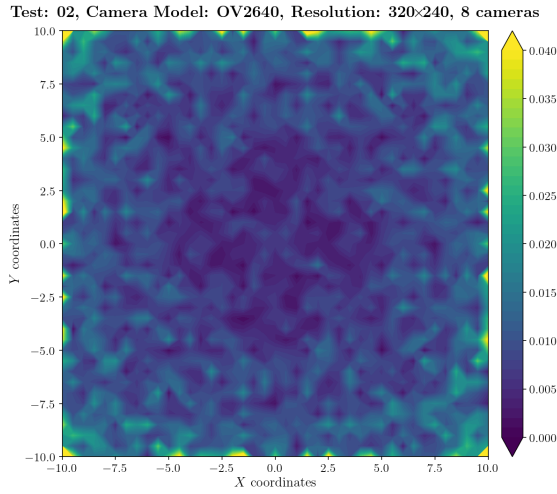


Figure A.195: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.500$ m.

Camera Coverage Spatial Distribution for $z = 3.5$ m level

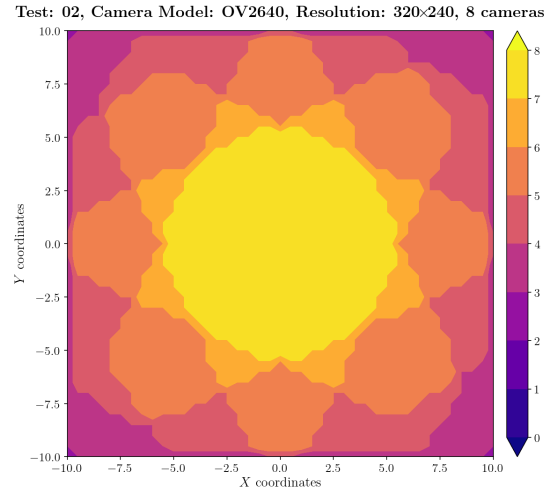


Figure A.196: Camera coverage for test 02 at level $z = 3.500$ m and resolution 320×240.

Estimation Error distribution for $z = 4.0$ m level

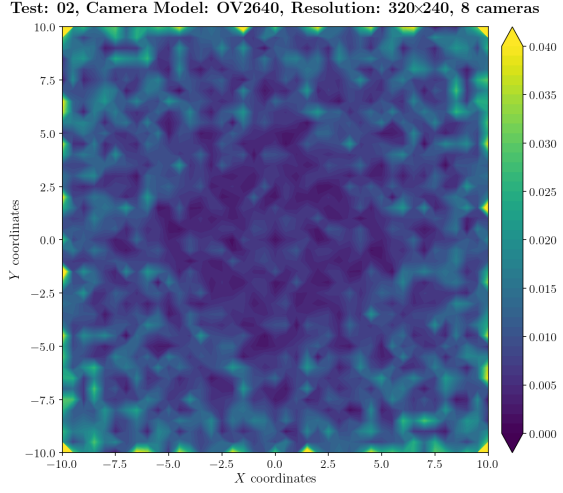


Figure A.197: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.000$ m.

Camera Coverage Spacial Distribution for $z = 4.0$ m level

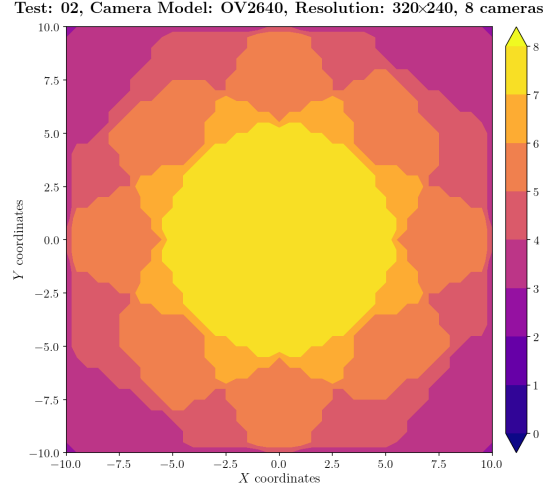


Figure A.198: Camera coverage for test 02 at level $z = 4.000$ m and resolution 320×240.

Estimation Error distribution for $z = 4.5$ m level

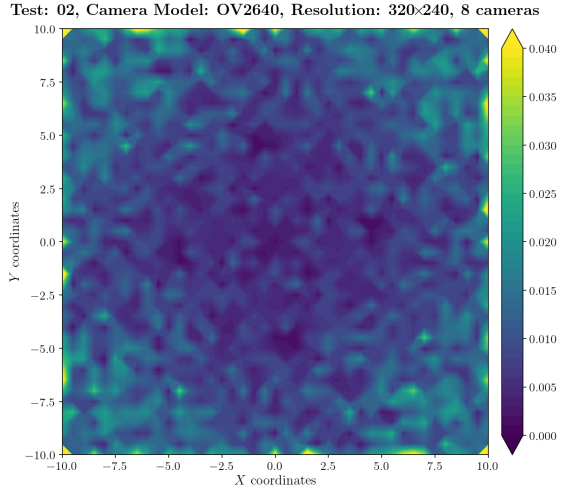


Figure A.199: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.500$ m.

Camera Coverage Spacial Distribution for $z = 4.5$ m level

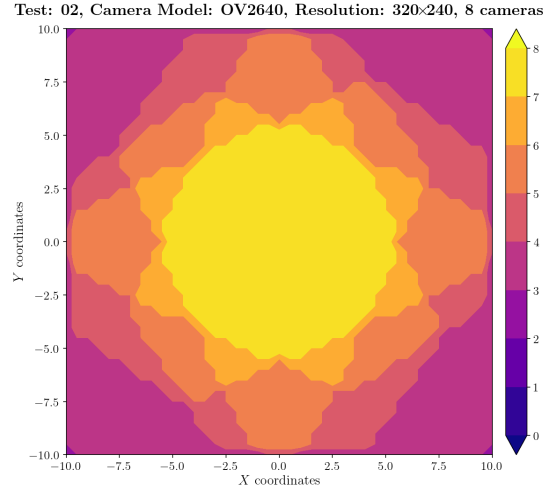


Figure A.200: Camera coverage for test 02 at level $z = 4.500$ m and resolution 320×240.

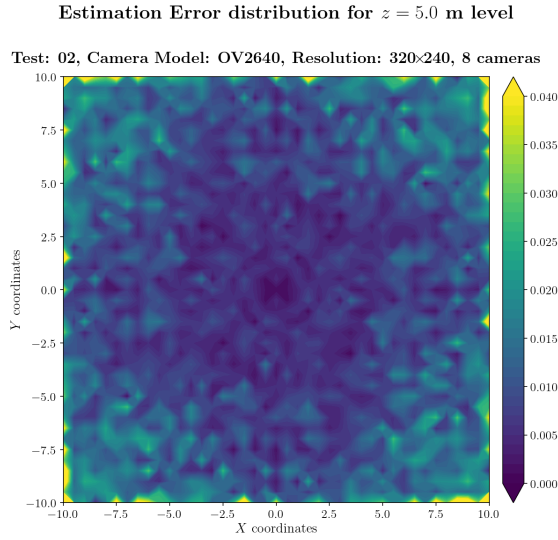


Figure A.201: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 5.000$ m.

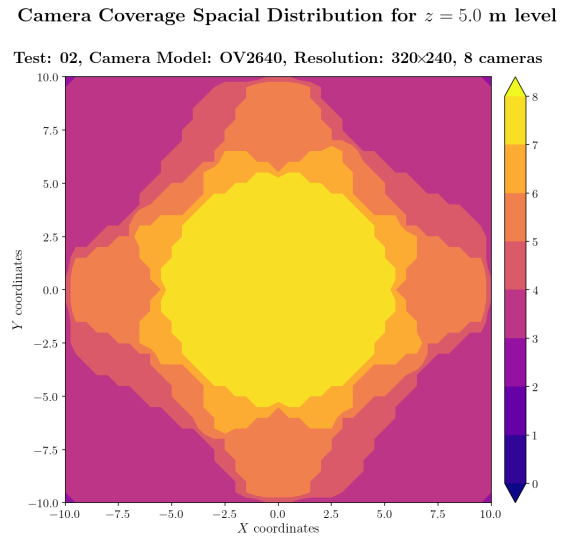


Figure A.202: Camera coverage for test 02 at level $z = 5.000$ m and resolution 320×240.

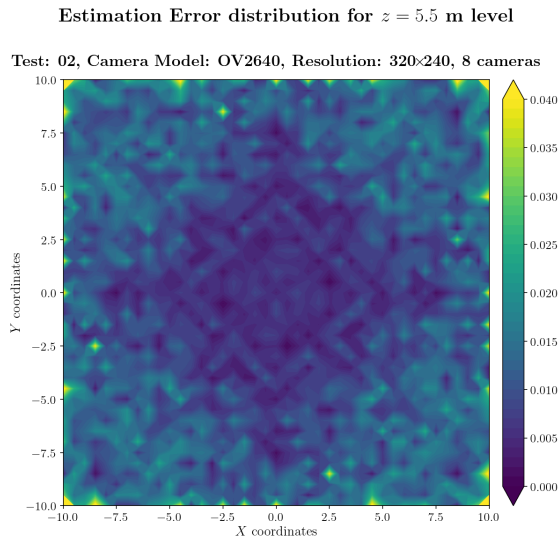


Figure A.203: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 5.500$ m.

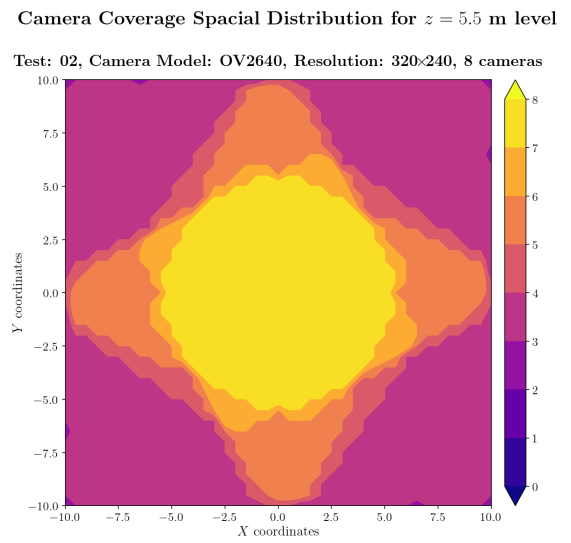
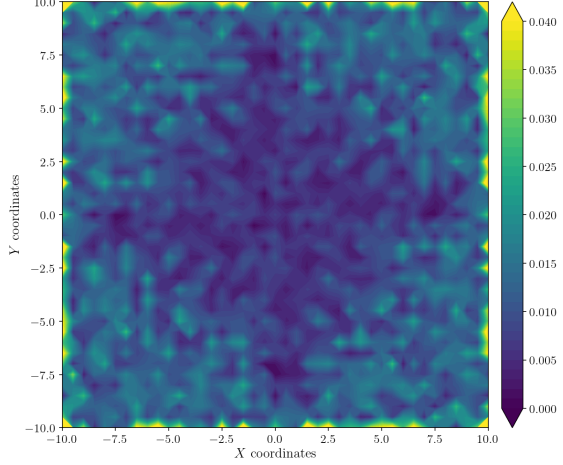


Figure A.204: Camera coverage for test 02 at level $z = 5.500$ m and resolution 320×240.

Estimation Error distribution for $z = 6.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras

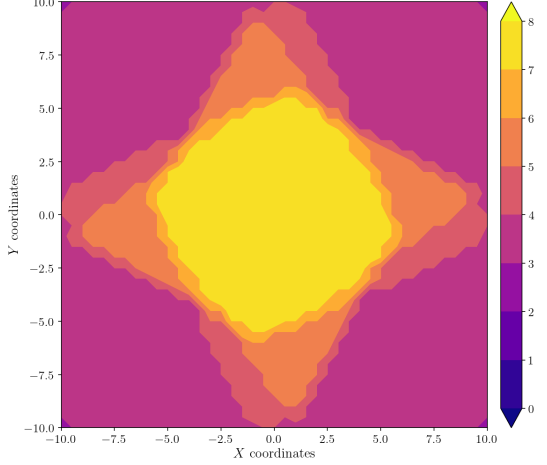
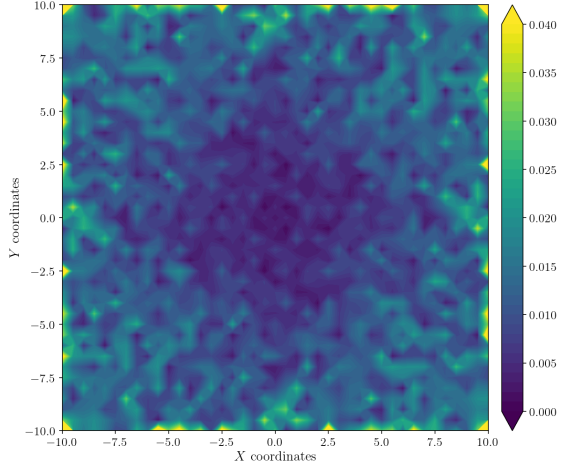


Figure A.205: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 6.000$ m.

Figure A.206: Camera coverage for test 02 at level $z = 6.000$ m and resolution 320×240.

Estimation Error distribution for $z = 6.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras

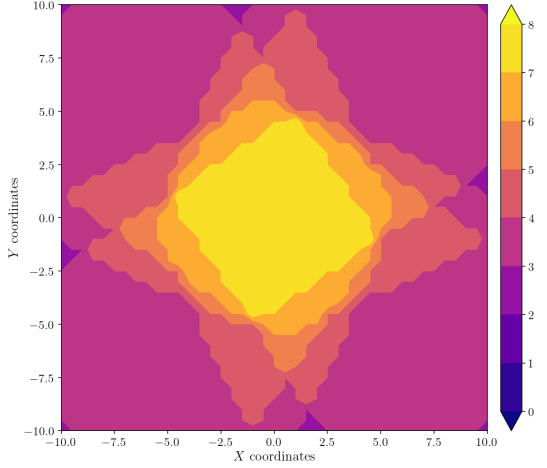


Figure A.207: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 6.500$ m.

Figure A.208: Camera coverage for test 02 at level $z = 6.500$ m and resolution 320×240.

Estimation Error distribution for $z = 7.0$ m level

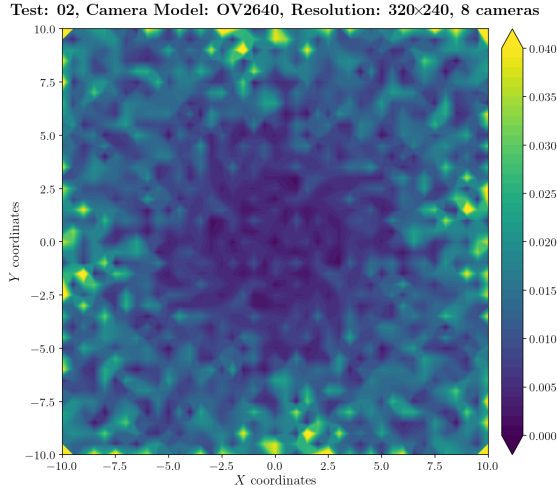


Figure A.209: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 7.000$ m.

Camera Coverage Spatial Distribution for $z = 7.0$ m level

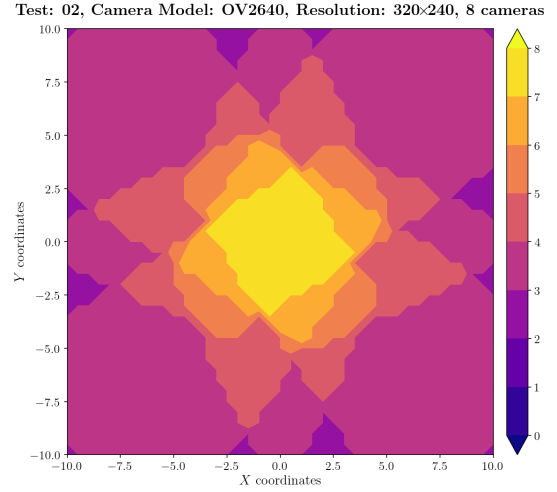


Figure A.210: Camera coverage for test 02 at level $z = 7.000$ m and resolution 320×240.

Estimation Error distribution for $z = 7.5$ m level

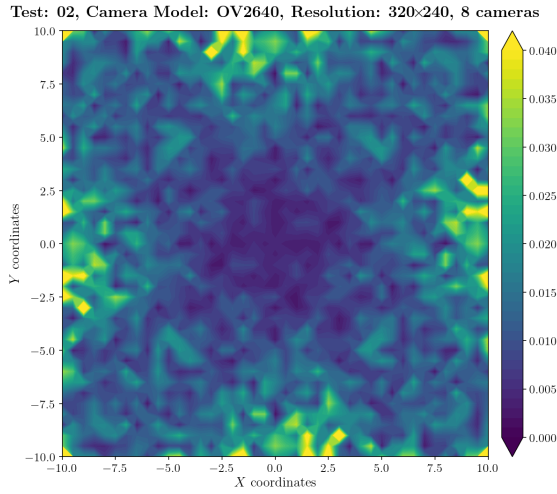


Figure A.211: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 7.500$ m.

Camera Coverage Spatial Distribution for $z = 7.5$ m level

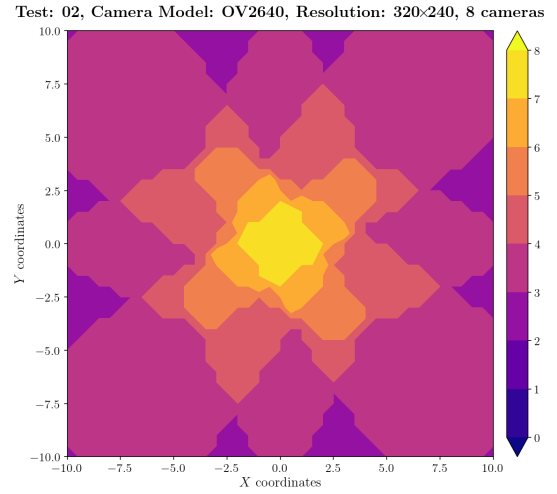
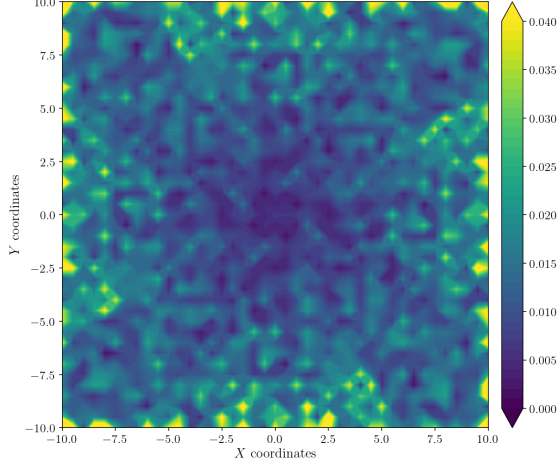


Figure A.212: Camera coverage for test 02 at level $z = 7.500$ m and resolution 320×240.

Estimation Error distribution for $z = 8.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras

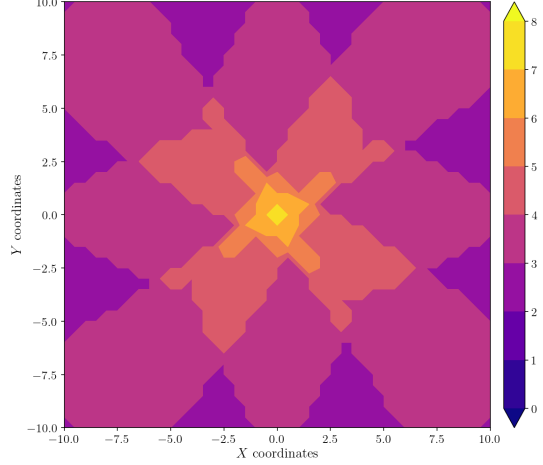
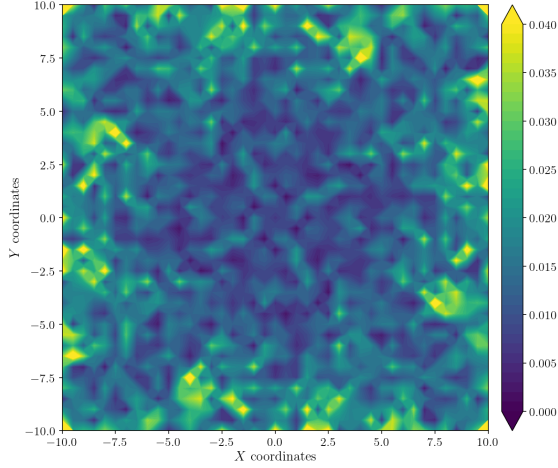


Figure A.213: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 8.000$ m.

Figure A.214: Camera coverage for test 02 at level $z = 8.000$ m and resolution 320×240.

Estimation Error distribution for $z = 8.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 320×240, 8 cameras

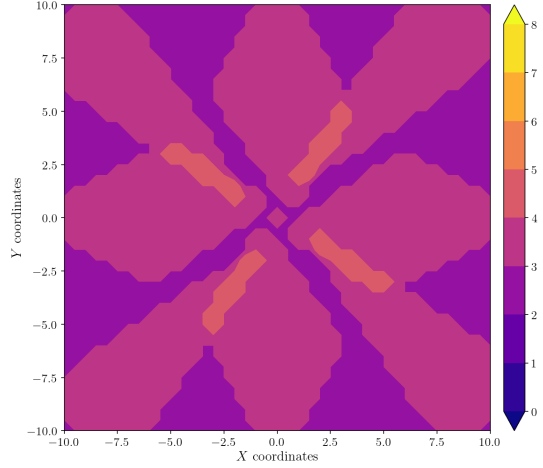


Figure A.215: Estimation error contour map for test 02 with 8 cameras using resolution of 320×240 pixels, at level $z = 8.500$ m.

Figure A.216: Camera coverage for test 02 at level $z = 8.500$ m and resolution 320×240.

Error maps for resolution 640×480

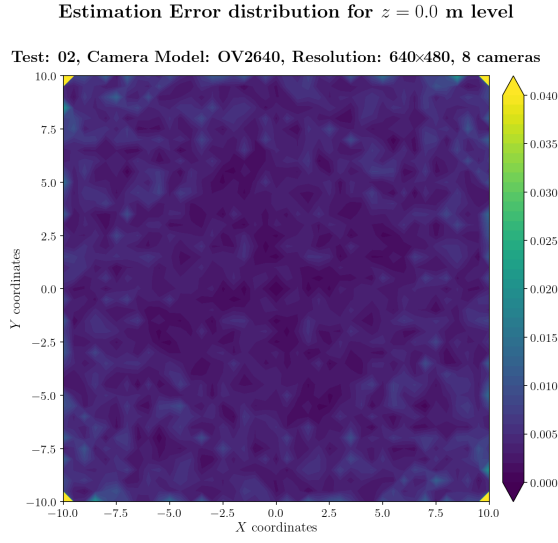


Figure A.217: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.000$ m.

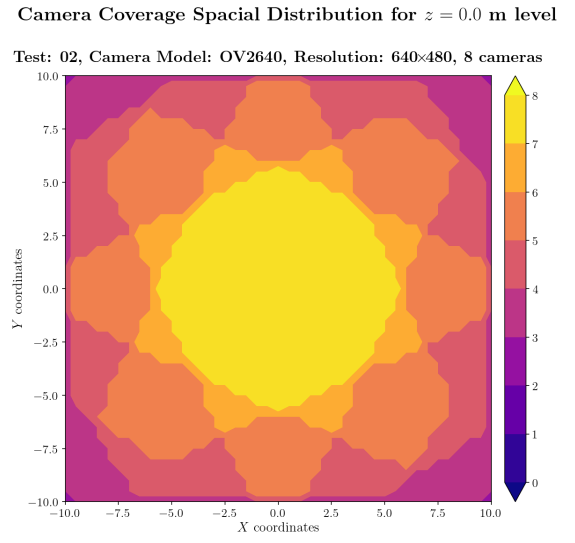


Figure A.218: Camera coverage for test 02 at level $z = 0.000$ m and resolution 640×480.

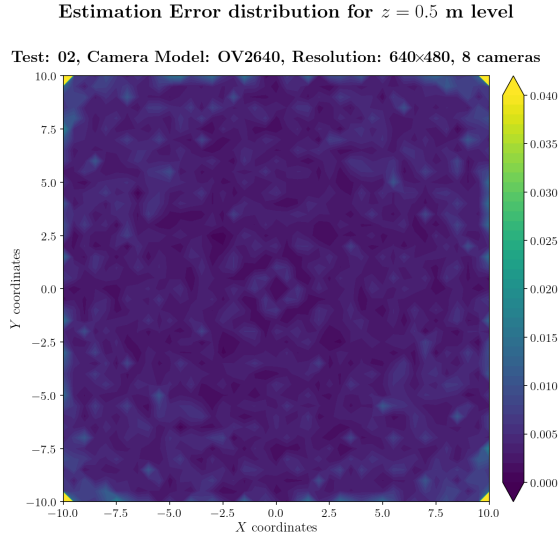


Figure A.219: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.500$ m.

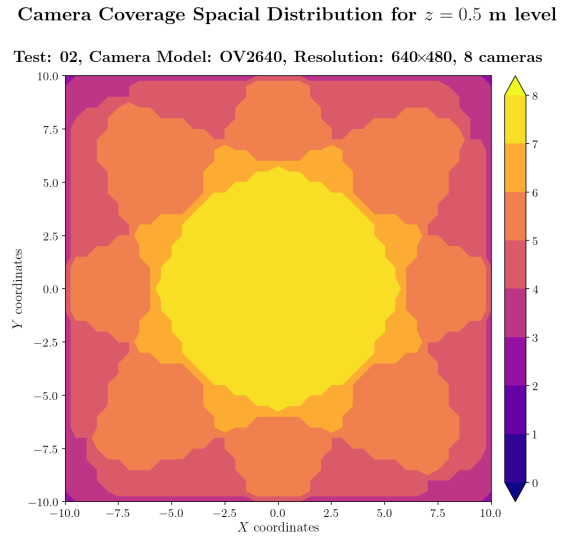
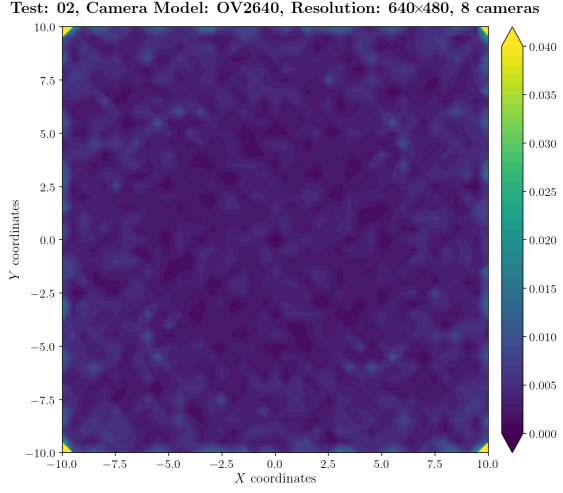


Figure A.220: Camera coverage for test 02 at level $z = 0.500$ m and resolution 640×480.

Estimation Error distribution for $z = 1.0$ m level



Camera Coverage Spacial Distribution for $z = 1.0$ m level

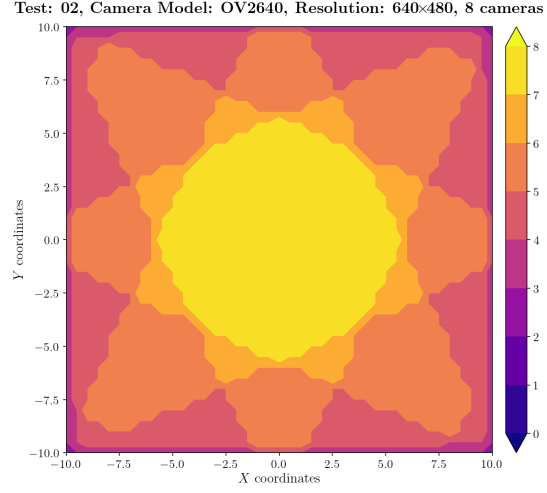
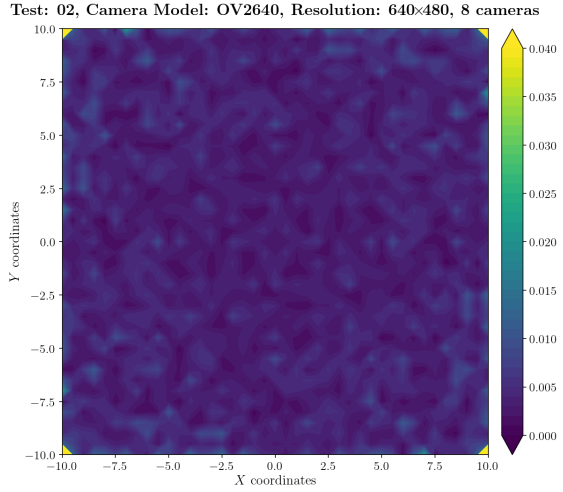


Figure A.221: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.000$ m.

Figure A.222: Camera coverage for test 02 at level $z = 1.000$ m and resolution 640×480.

Estimation Error distribution for $z = 1.5$ m level



Camera Coverage Spacial Distribution for $z = 1.5$ m level

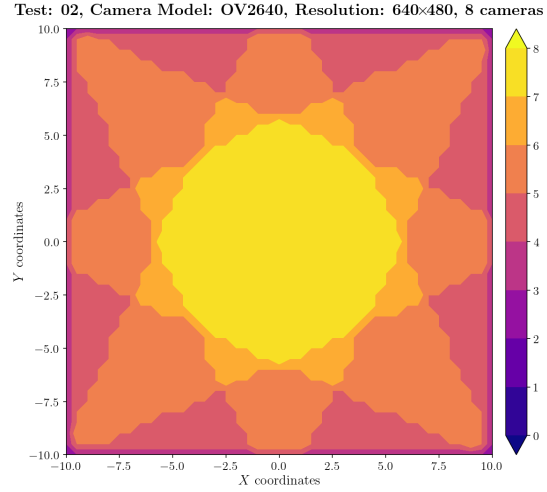


Figure A.223: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.500$ m.

Figure A.224: Camera coverage for test 02 at level $z = 1.500$ m and resolution 640×480.

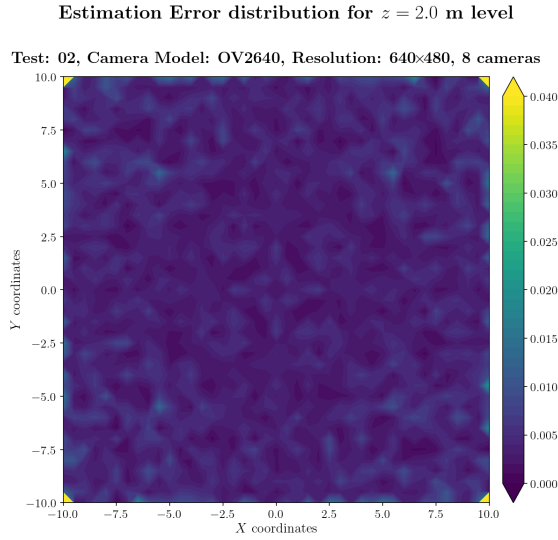


Figure A.225: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.000$ m.

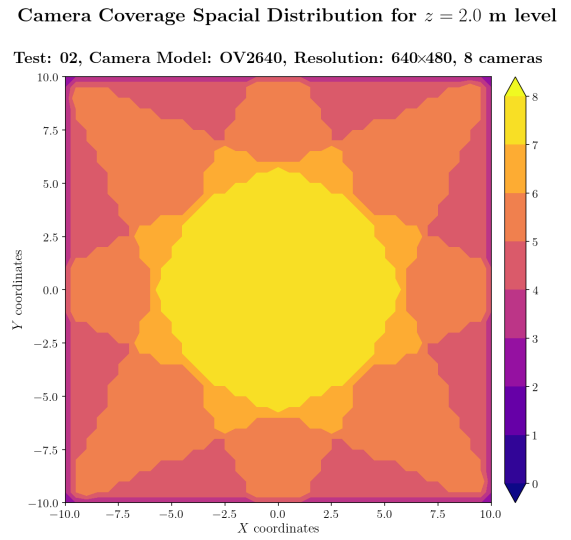


Figure A.226: Camera coverage for test 02 at level $z = 2.000$ m and resolution 640×480.

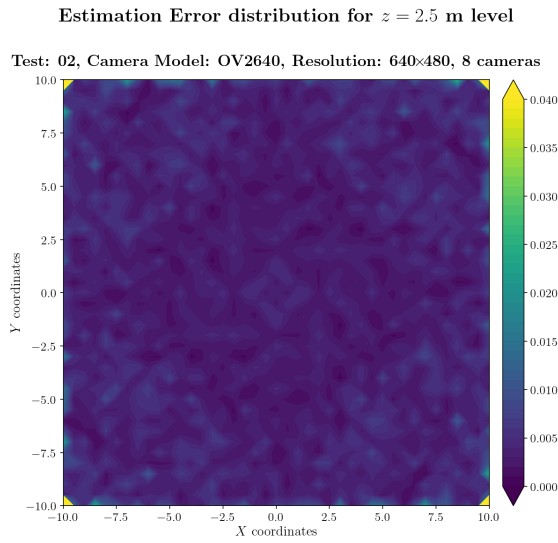


Figure A.227: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.500$ m.

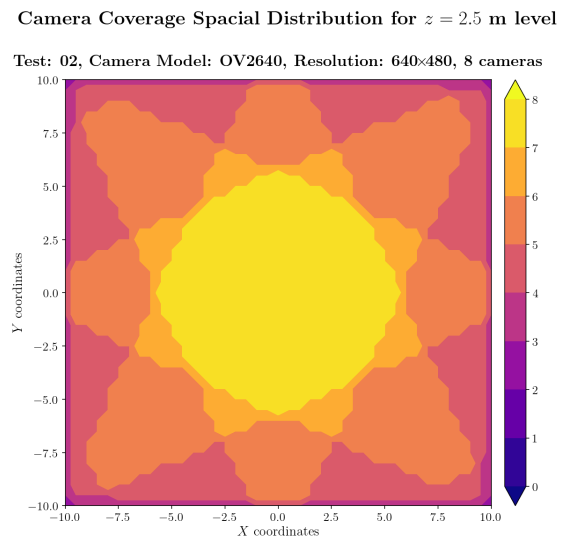
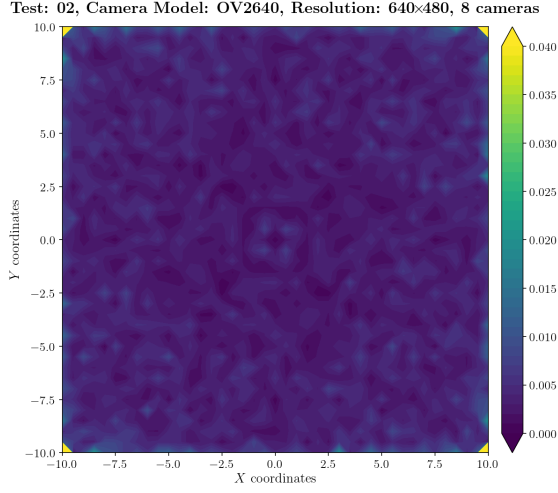


Figure A.228: Camera coverage for test 02 at level $z = 2.500$ m and resolution 640×480.

Estimation Error distribution for $z = 3.0$ m level



Camera Coverage Spacial Distribution for $z = 3.0$ m level

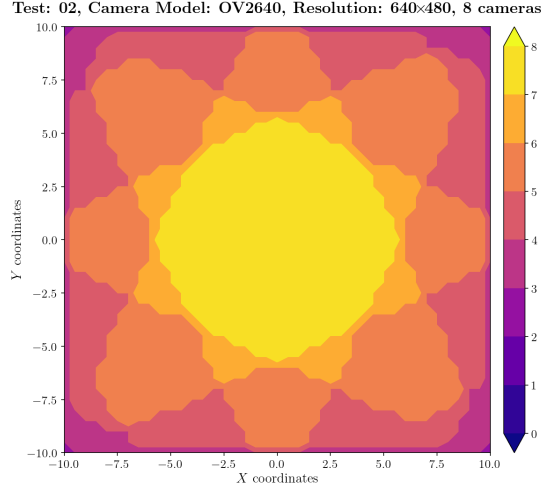
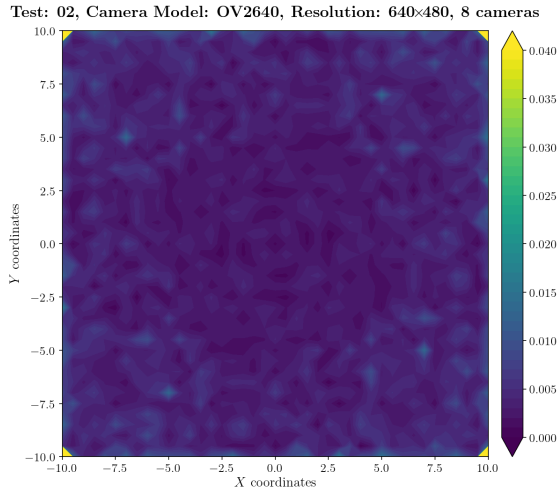


Figure A.229: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.000$ m.

Figure A.230: Camera coverage for test 02 at level $z = 3.000$ m and resolution 640×480.

Estimation Error distribution for $z = 3.5$ m level



Camera Coverage Spacial Distribution for $z = 3.5$ m level

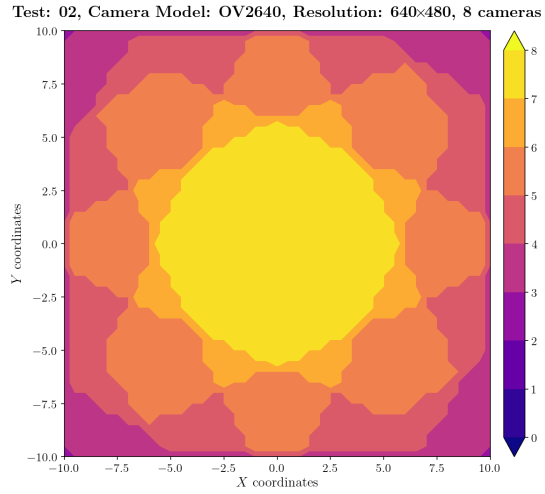


Figure A.231: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.500$ m.

Figure A.232: Camera coverage for test 02 at level $z = 3.500$ m and resolution 640×480.

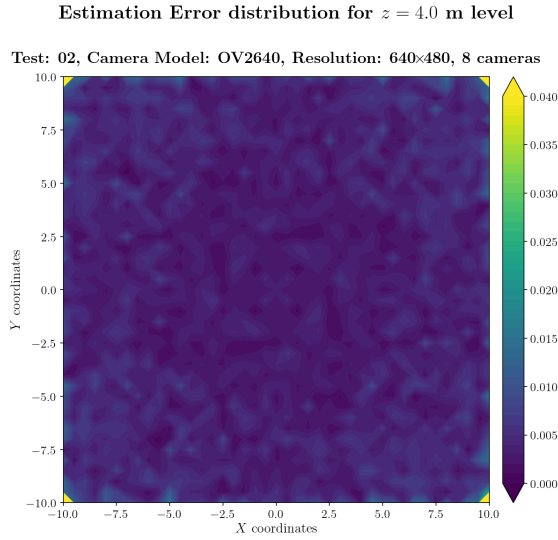


Figure A.233: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.000$ m.

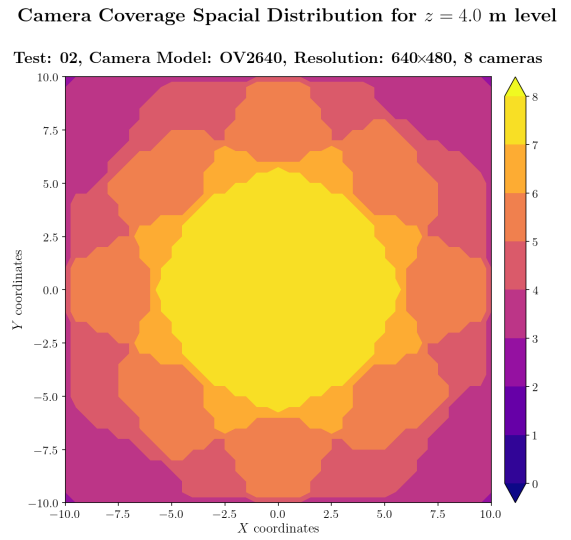


Figure A.234: Camera coverage for test 02 at level $z = 4.000$ m and resolution 640×480.

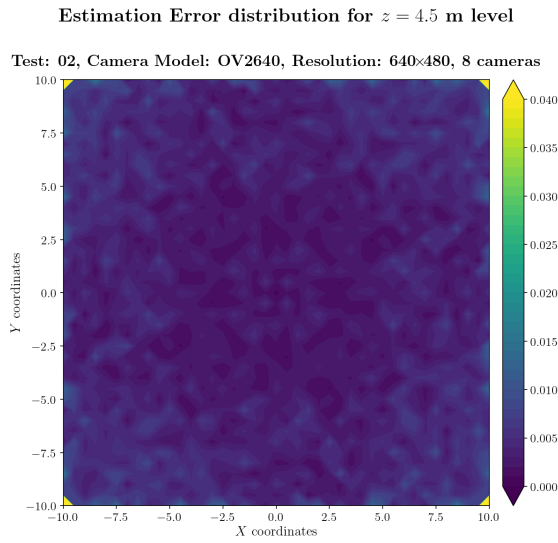


Figure A.235: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.500$ m.

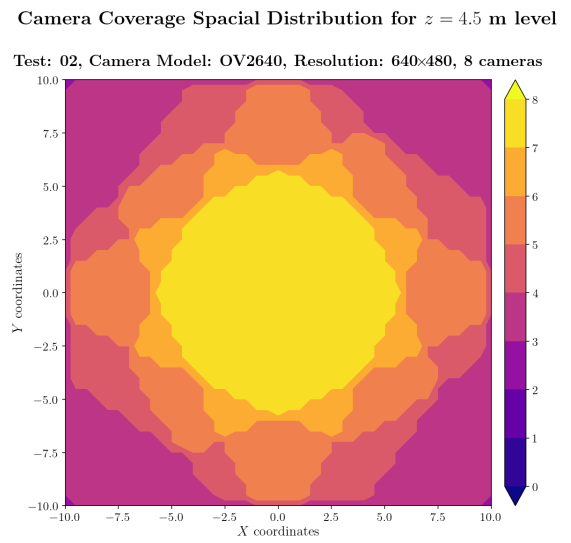
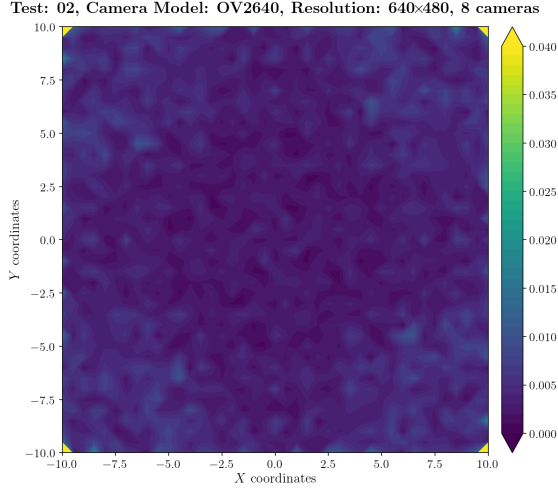


Figure A.236: Camera coverage for test 02 at level $z = 4.500$ m and resolution 640×480.

Estimation Error distribution for $z = 5.0$ m level



Camera Coverage Spacial Distribution for $z = 5.0$ m level

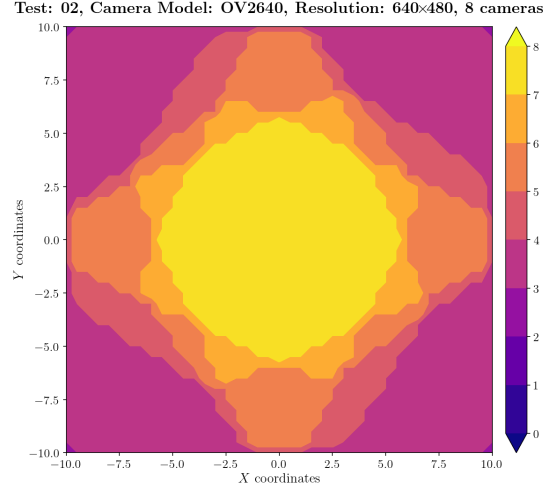
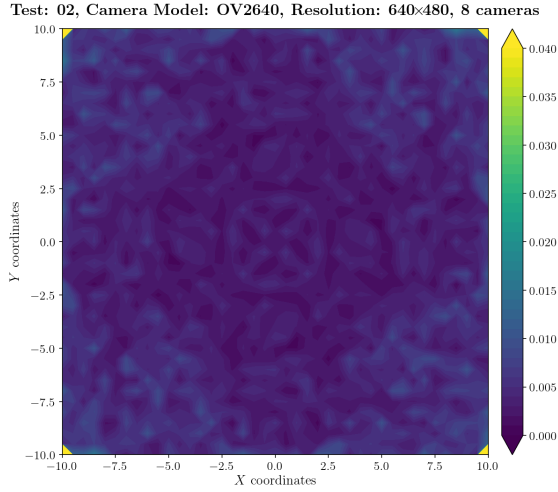


Figure A.237: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 5.000$ m.

Figure A.238: Camera coverage for test 02 at level $z = 5.000$ m and resolution 640×480.

Estimation Error distribution for $z = 5.5$ m level



Camera Coverage Spacial Distribution for $z = 5.5$ m level

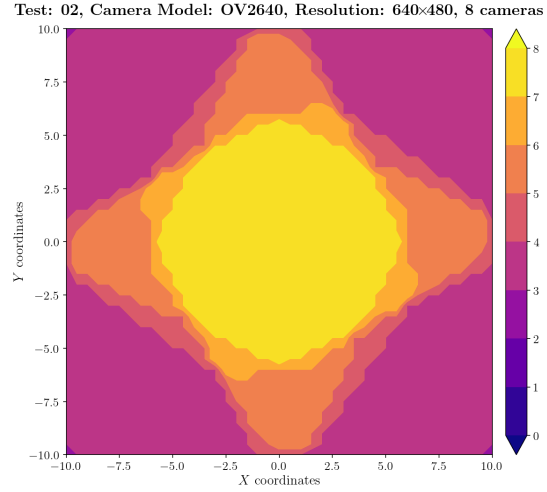


Figure A.239: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 5.500$ m.

Figure A.240: Camera coverage for test 02 at level $z = 5.500$ m and resolution 640×480.

Estimation Error distribution for $z = 6.0$ m level

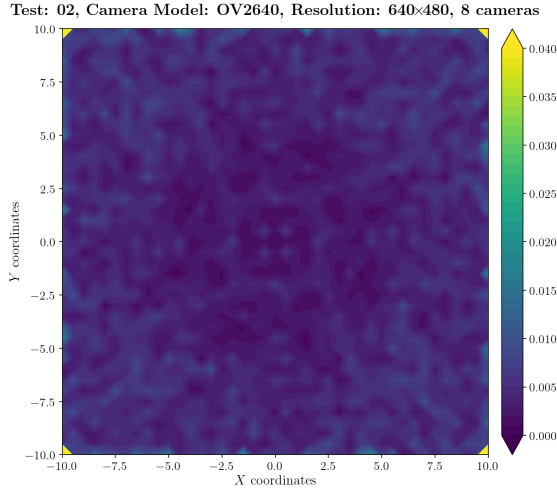


Figure A.241: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 6.000$ m.

Camera Coverage Spatial Distribution for $z = 6.0$ m level

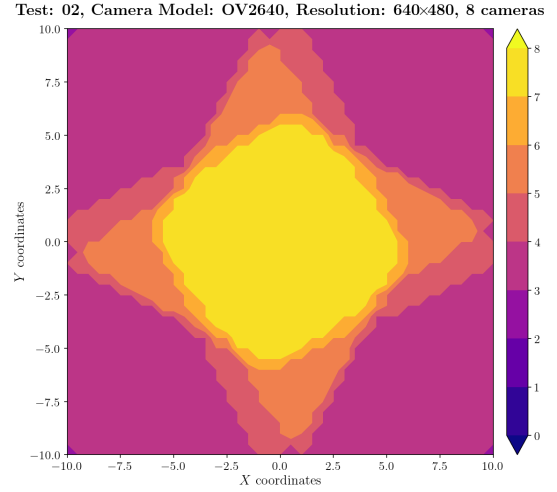


Figure A.242: Camera coverage for test 02 at level $z = 6.000$ m and resolution 640×480.

Estimation Error distribution for $z = 6.5$ m level

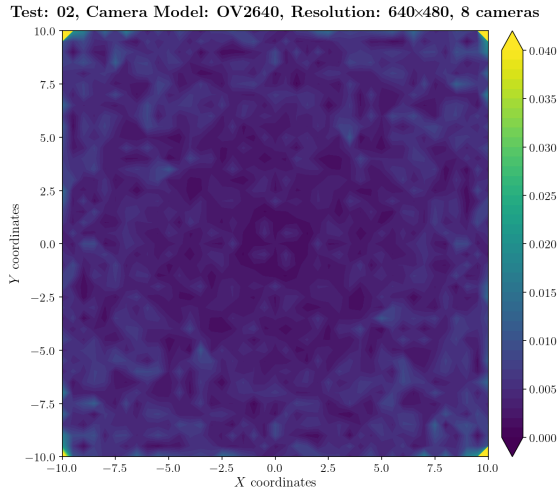


Figure A.243: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 6.500$ m.

Camera Coverage Spatial Distribution for $z = 6.5$ m level

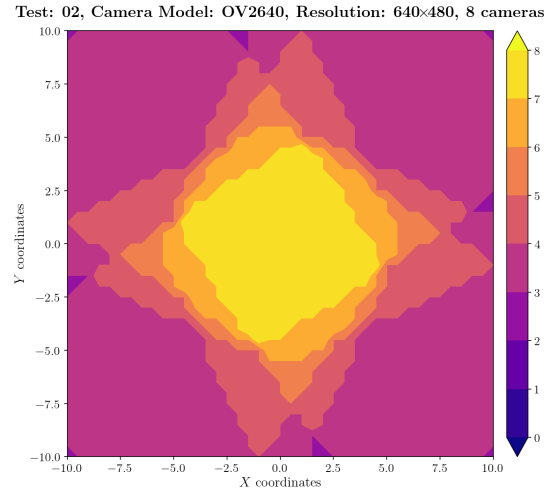
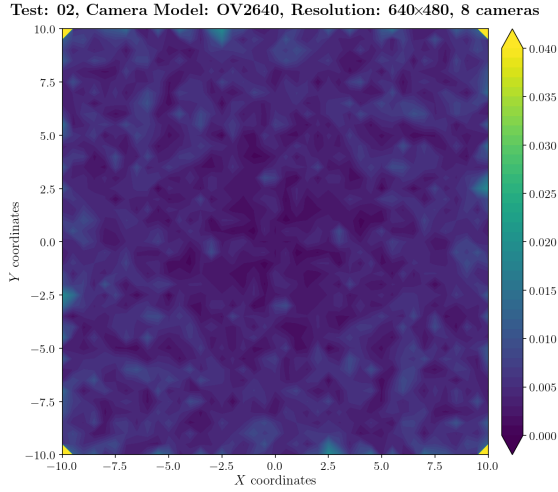


Figure A.244: Camera coverage for test 02 at level $z = 6.500$ m and resolution 640×480.

Estimation Error distribution for $z = 7.0$ m level



Camera Coverage Spacial Distribution for $z = 7.0$ m level

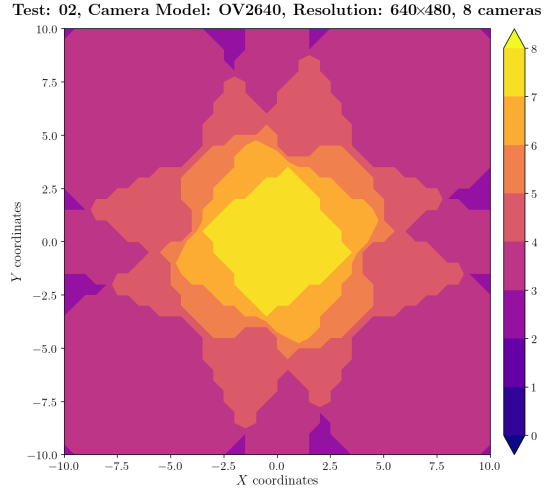
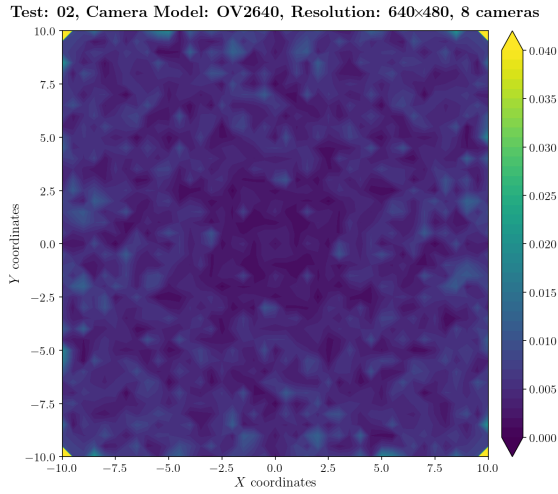


Figure A.245: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 7.000$ m.

Figure A.246: Camera coverage for test 02 at level $z = 7.000$ m and resolution 640×480.

Estimation Error distribution for $z = 7.5$ m level



Camera Coverage Spacial Distribution for $z = 7.5$ m level

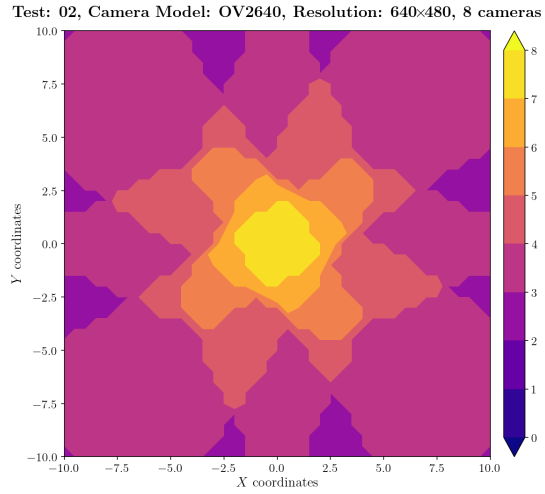


Figure A.247: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 7.500$ m.

Figure A.248: Camera coverage for test 02 at level $z = 7.500$ m and resolution 640×480.

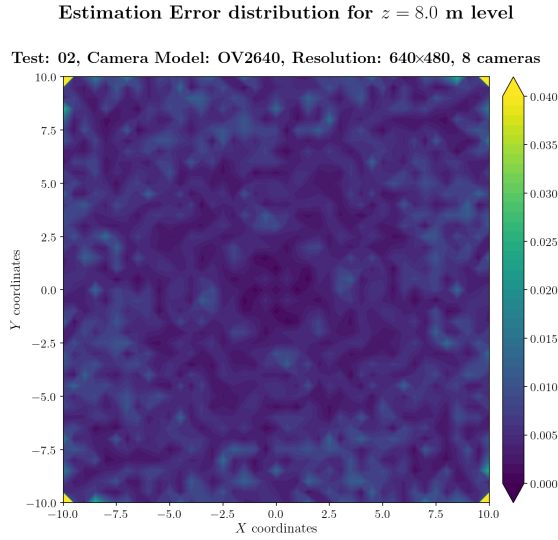


Figure A.249: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 8.000$ m.

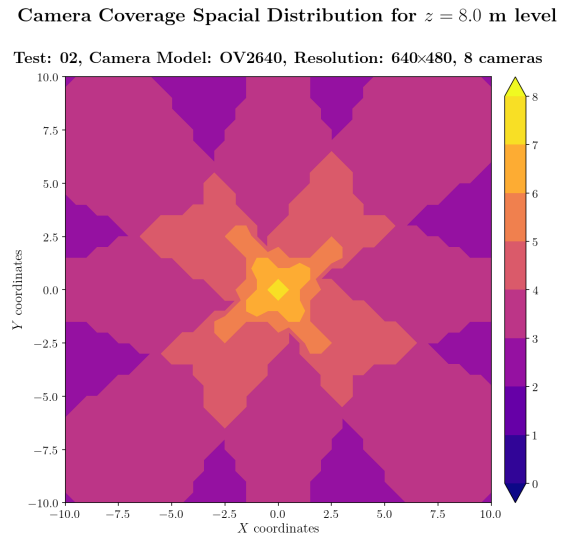


Figure A.250: Camera coverage for test 02 at level $z = 8.000$ m and resolution 640×480.

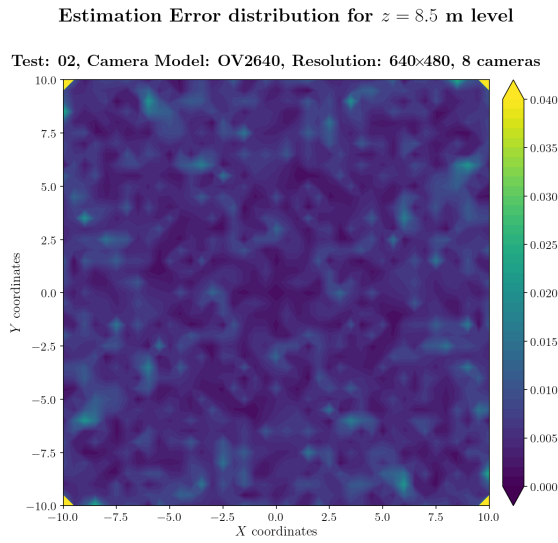


Figure A.251: Estimation error contour map for test 02 with 8 cameras using resolution of 640×480 pixels, at level $z = 8.500$ m.

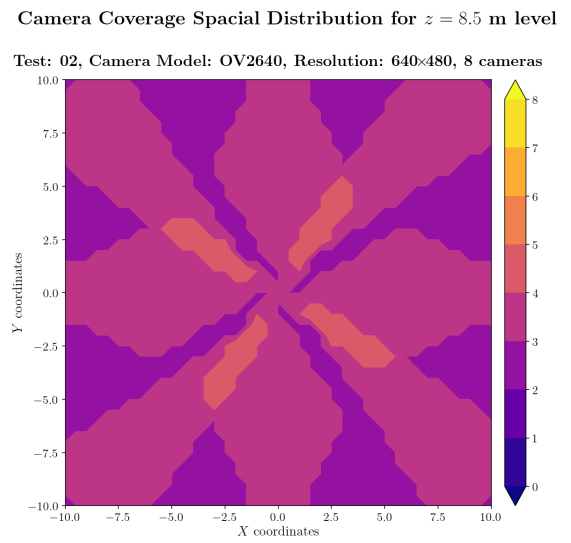


Figure A.252: Camera coverage for test 02 at level $z = 8.500$ m and resolution 640×480.

Error maps for resolution 800×600

Estimation Error distribution for $z = 0.0$ m level

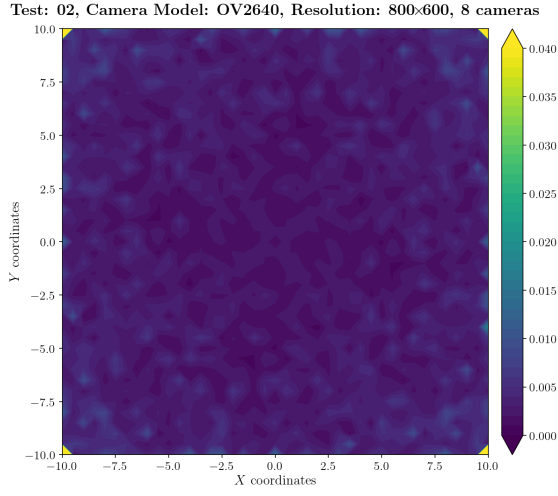


Figure A.253: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

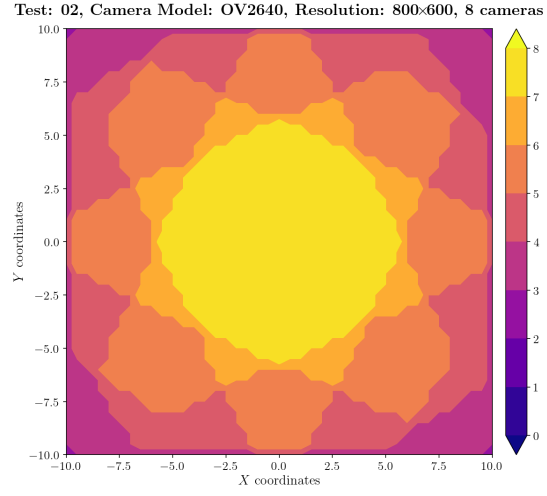


Figure A.254: Camera coverage for test 02 at level $z = 0.000$ m and resolution 800×600.

Estimation Error distribution for $z = 0.5$ m level

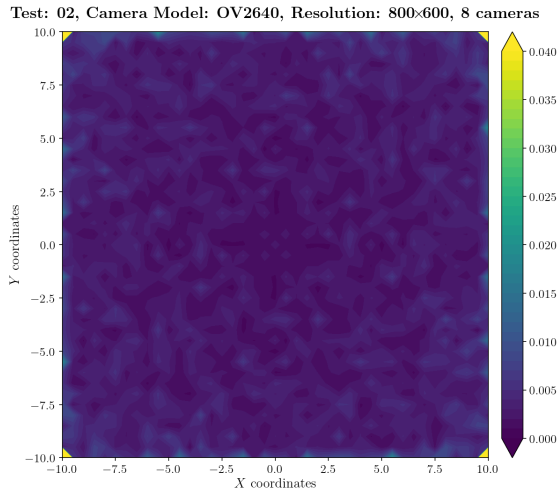


Figure A.255: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.500$ m.

Camera Coverage Spacial Distribution for $z = 0.5$ m level

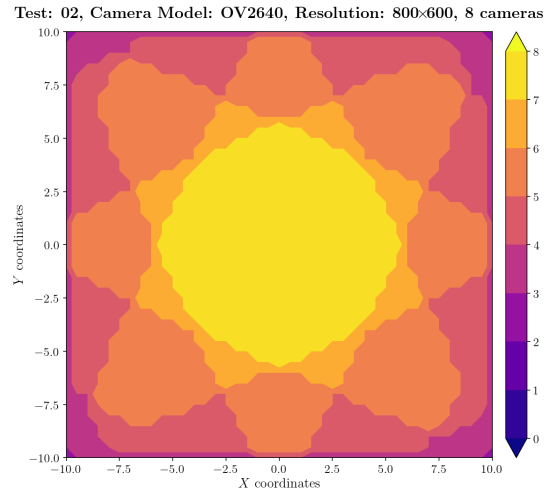


Figure A.256: Camera coverage for test 02 at level $z = 0.500$ m and resolution 800×600.

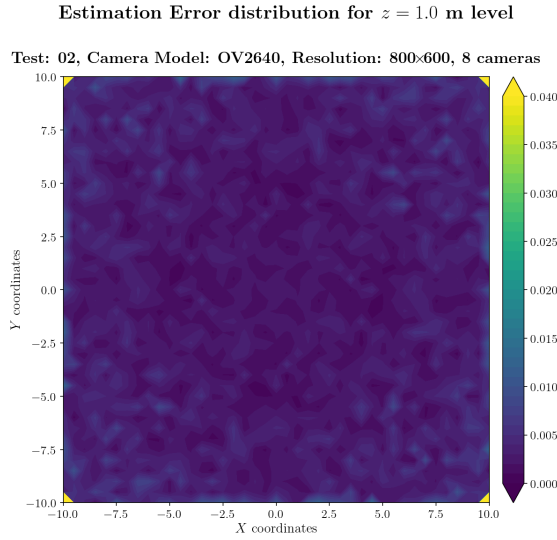


Figure A.257: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.000$ m.

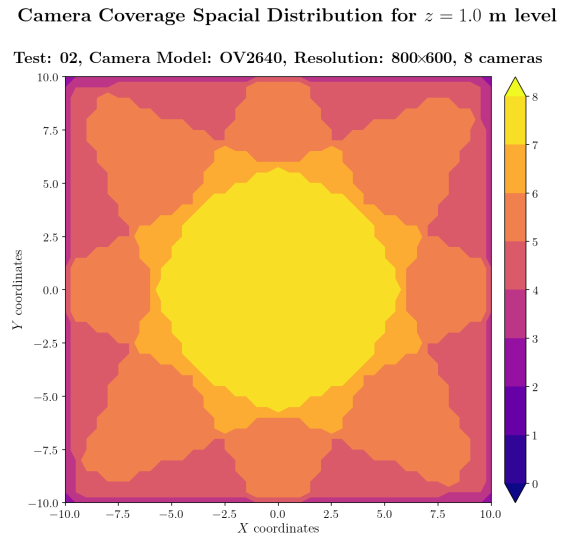


Figure A.258: Camera coverage for test 02 at level $z = 1.000$ m and resolution 800×600.

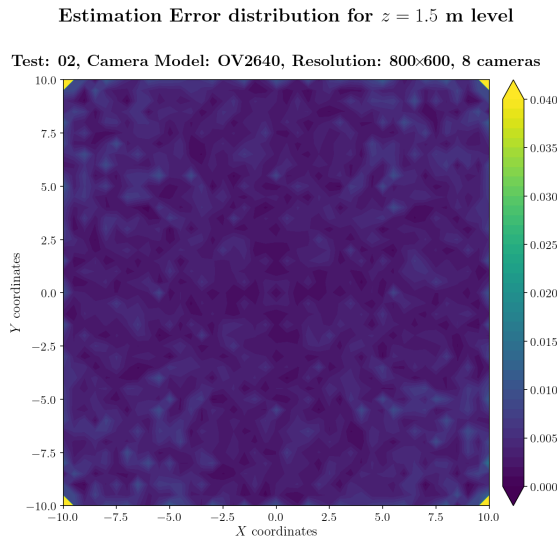


Figure A.259: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.500$ m.

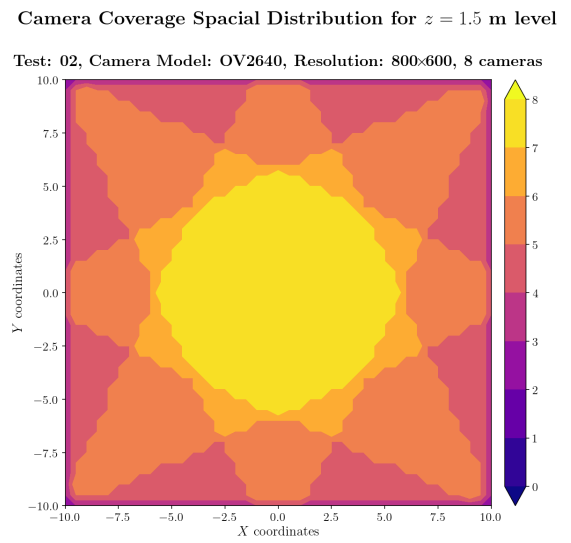
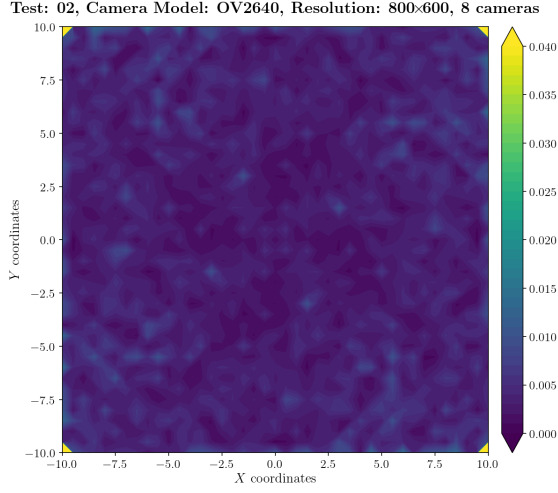


Figure A.260: Camera coverage for test 02 at level $z = 1.500$ m and resolution 800×600.

Estimation Error distribution for $z = 2.0$ m level



Camera Coverage Spacial Distribution for $z = 2.0$ m level

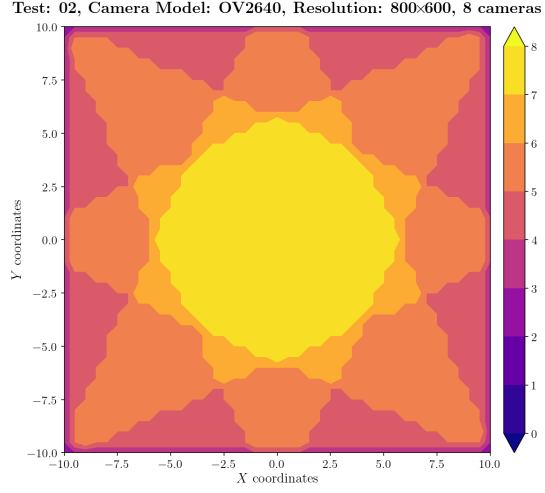
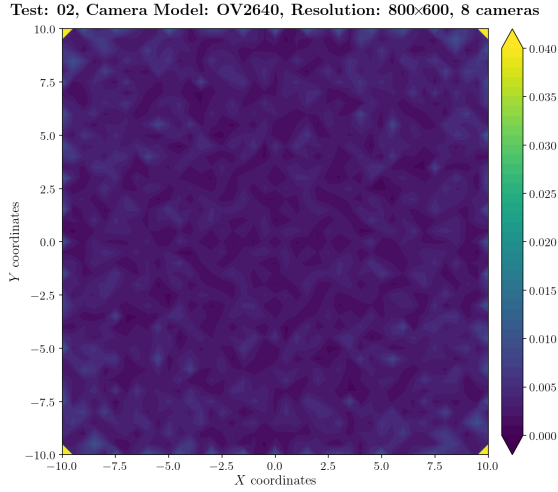


Figure A.261: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.000$ m.

Figure A.262: Camera coverage for test 02 at level $z = 2.000$ m and resolution 800×600.

Estimation Error distribution for $z = 2.5$ m level



Camera Coverage Spacial Distribution for $z = 2.5$ m level

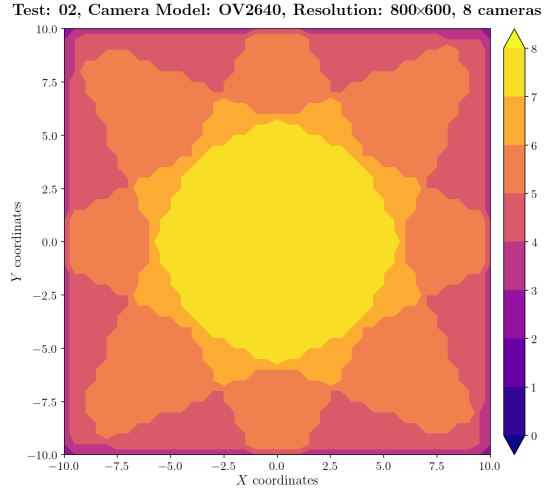


Figure A.263: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.500$ m.

Figure A.264: Camera coverage for test 02 at level $z = 2.500$ m and resolution 800×600.

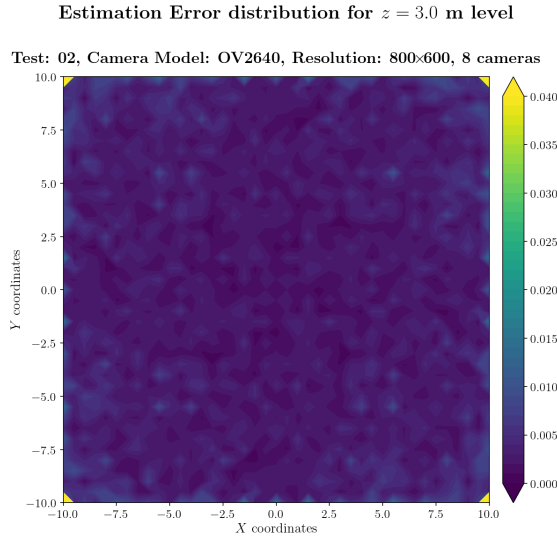


Figure A.265: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.000$ m.

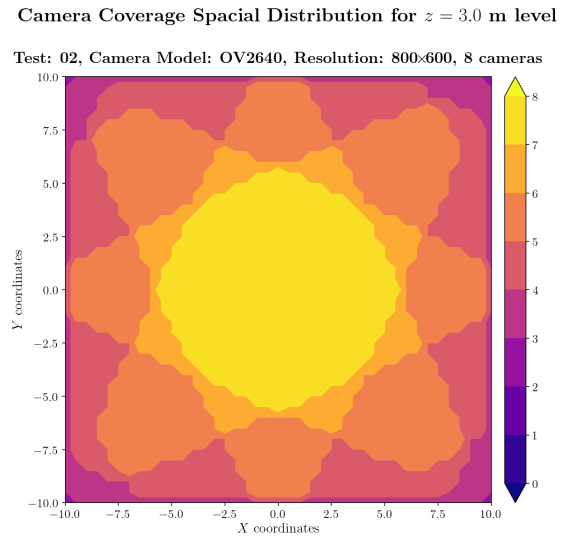


Figure A.266: Camera coverage for test 02 at level $z = 3.000$ m and resolution 800×600.

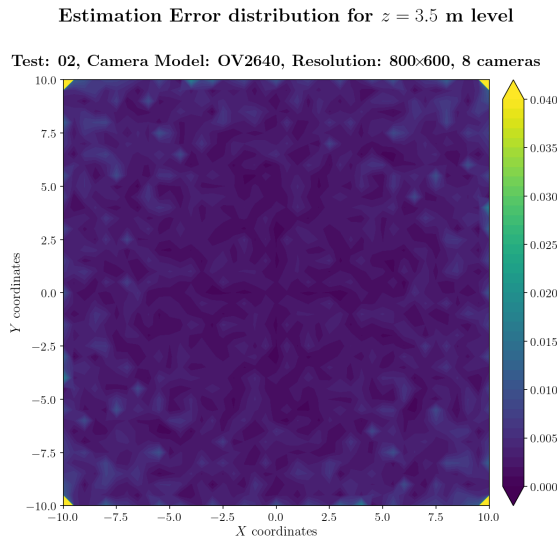


Figure A.267: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.500$ m.

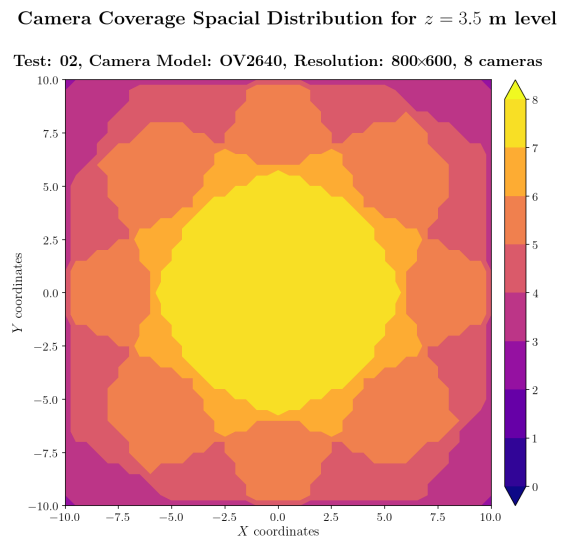


Figure A.268: Camera coverage for test 02 at level $z = 3.500$ m and resolution 800×600.

Estimation Error distribution for $z = 4.0$ m level

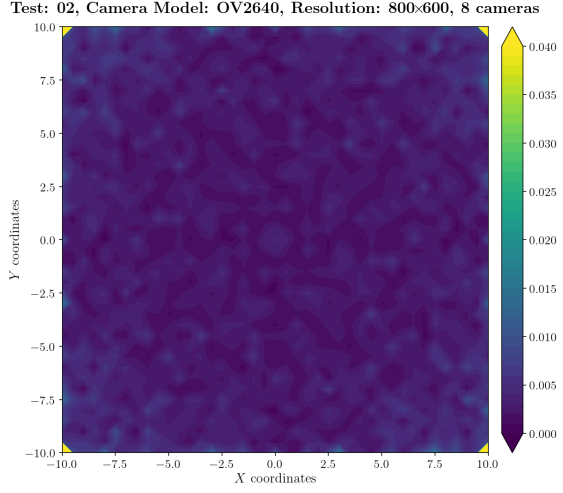


Figure A.269: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.000$ m.

Camera Coverage Spacial Distribution for $z = 4.0$ m level

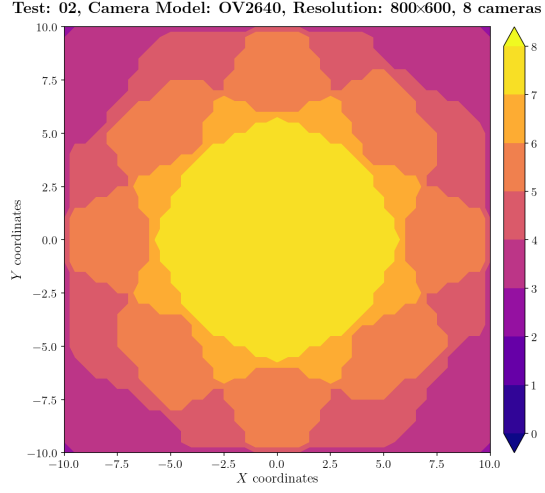


Figure A.270: Camera coverage for test 02 at level $z = 4.000$ m and resolution 800×600.

Estimation Error distribution for $z = 4.5$ m level

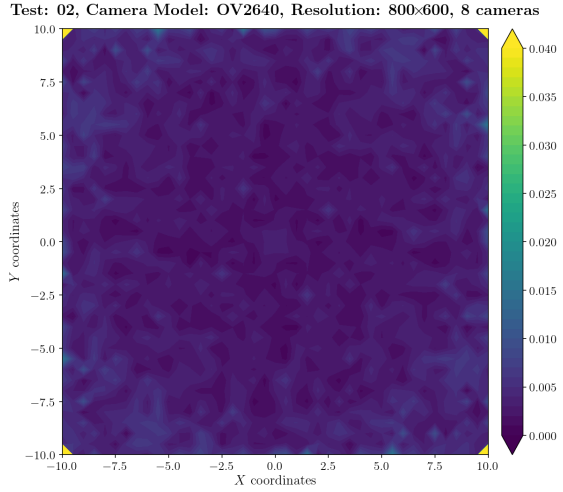


Figure A.271: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.500$ m.

Camera Coverage Spacial Distribution for $z = 4.5$ m level

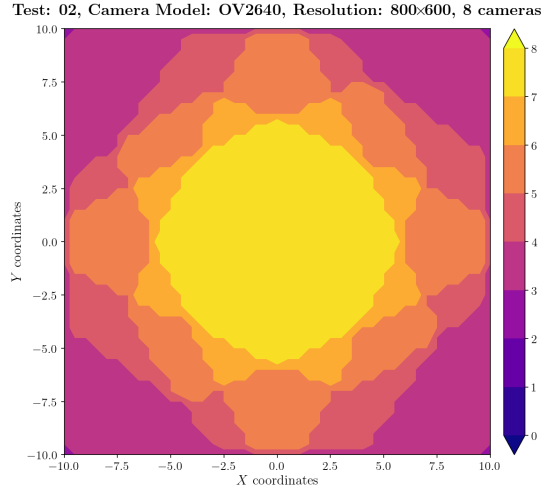


Figure A.272: Camera coverage for test 02 at level $z = 4.500$ m and resolution 800×600.

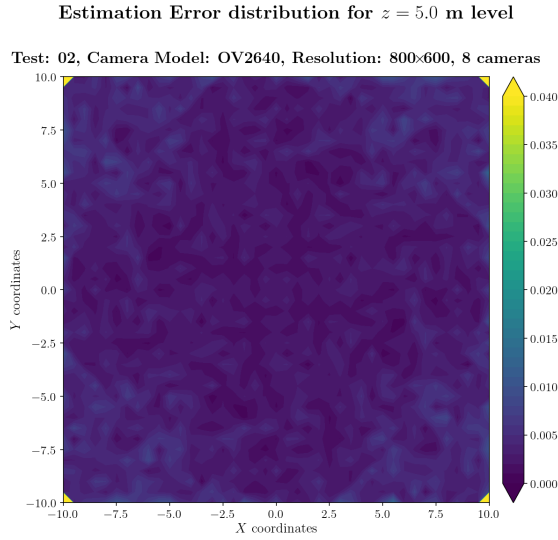


Figure A.273: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 5.000$ m.

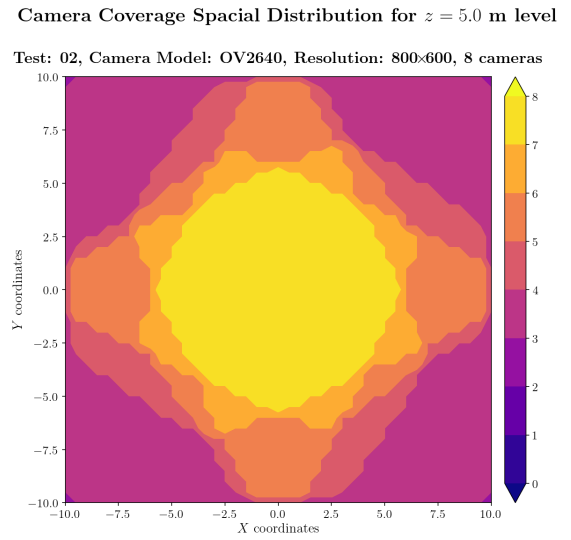


Figure A.274: Camera coverage for test 02 at level $z = 5.000$ m and resolution 800×600.

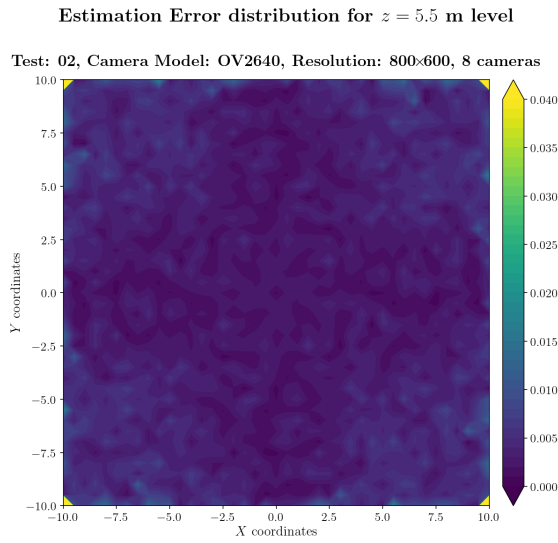


Figure A.275: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 5.500$ m.

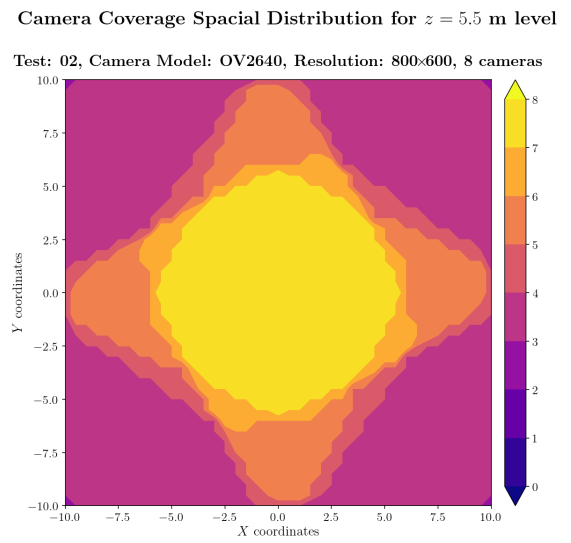
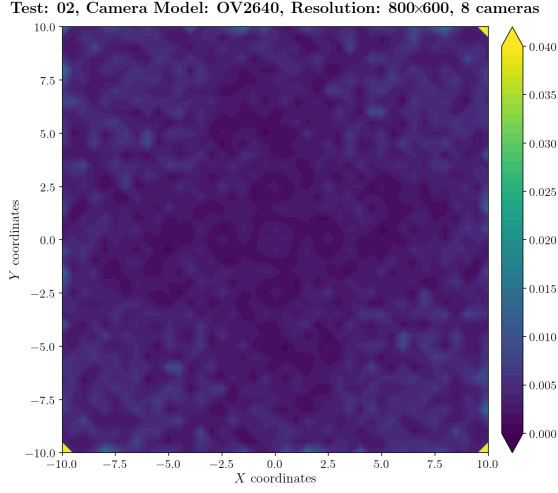


Figure A.276: Camera coverage for test 02 at level $z = 5.500$ m and resolution 800×600.

Estimation Error distribution for $z = 6.0$ m level



Camera Coverage Spacial Distribution for $z = 6.0$ m level

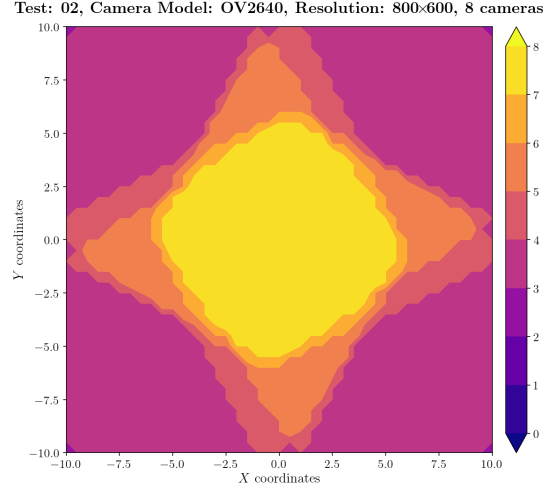
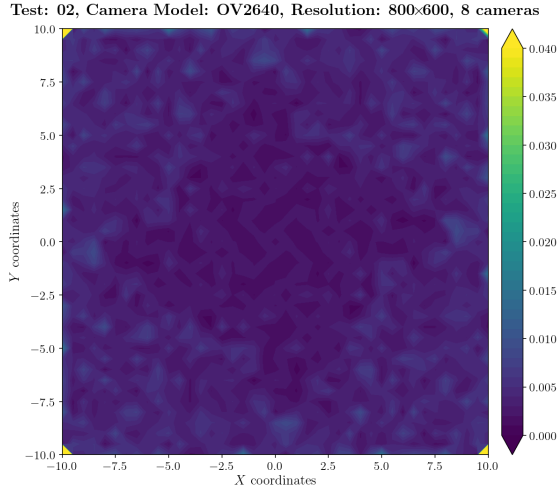


Figure A.277: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 6.000$ m.

Figure A.278: Camera coverage for test 02 at level $z = 6.000$ m and resolution 800×600.

Estimation Error distribution for $z = 6.5$ m level



Camera Coverage Spacial Distribution for $z = 6.5$ m level

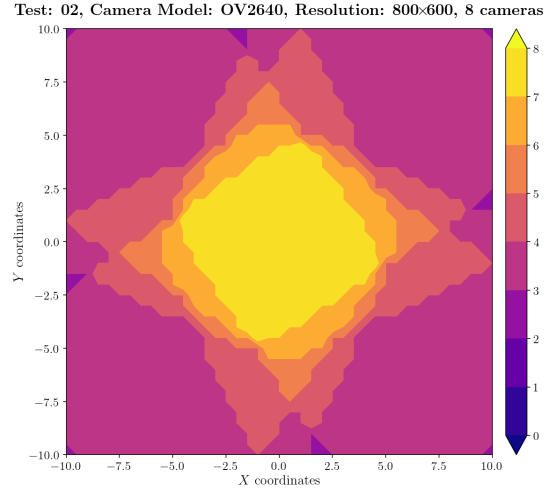


Figure A.279: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 6.500$ m.

Figure A.280: Camera coverage for test 02 at level $z = 6.500$ m and resolution 800×600.

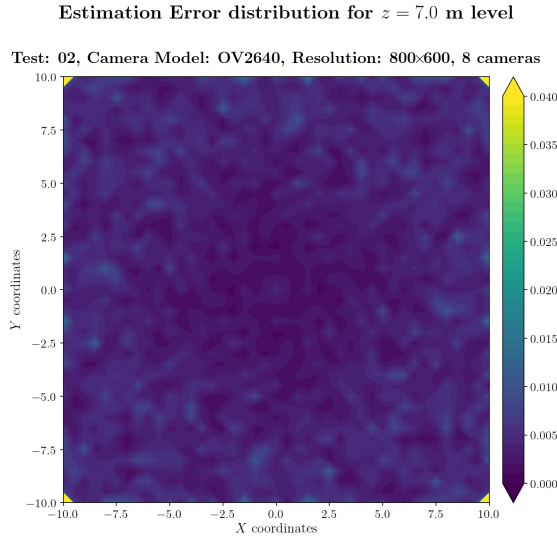


Figure A.281: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 7.000$ m.

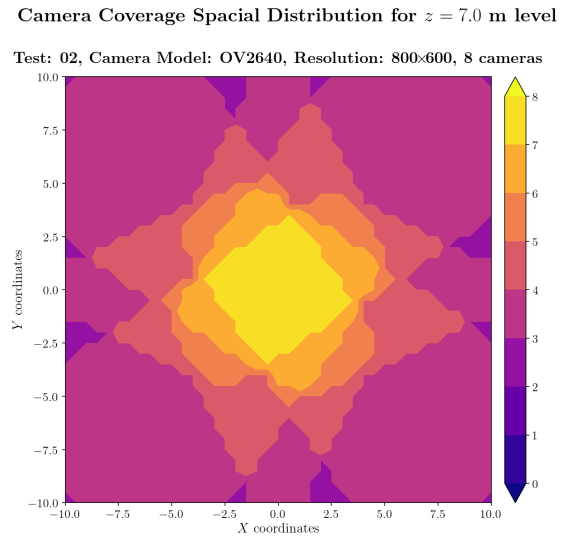


Figure A.282: Camera coverage for test 02 at level $z = 7.000$ m and resolution 800×600.

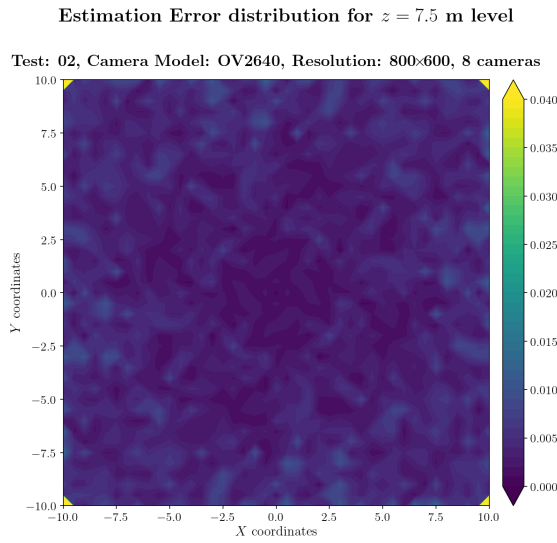


Figure A.283: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 7.500$ m.

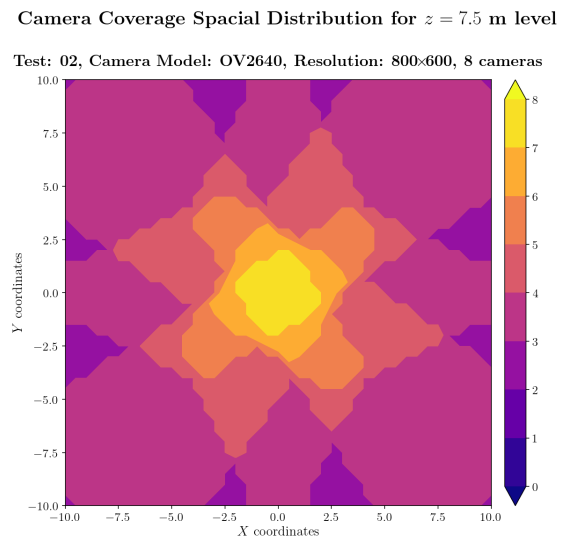
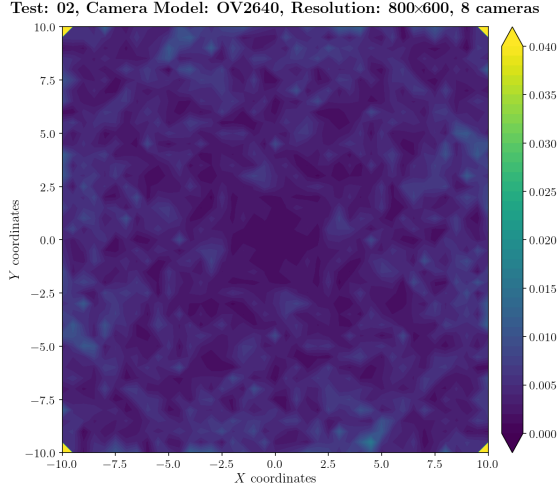


Figure A.284: Camera coverage for test 02 at level $z = 7.500$ m and resolution 800×600.

Estimation Error distribution for $z = 8.0$ m level



Camera Coverage Spacial Distribution for $z = 8.0$ m level

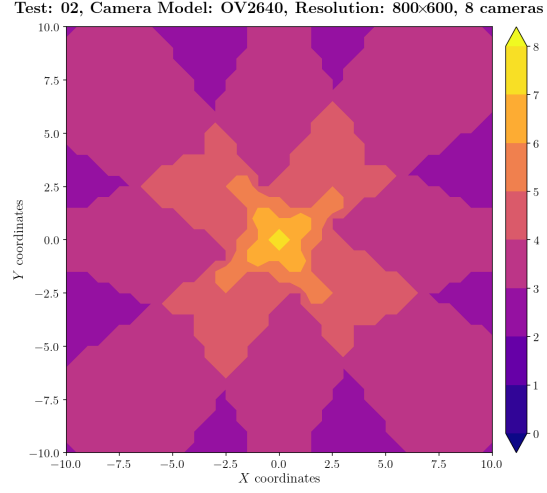
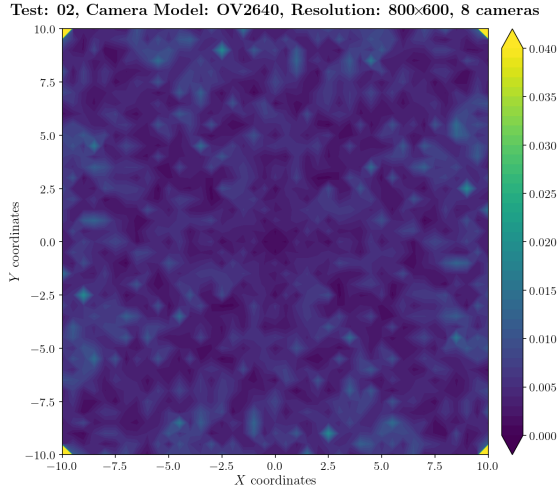


Figure A.285: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 8.000$ m.

Figure A.286: Camera coverage for test 02 at level $z = 8.000$ m and resolution 800×600.

Estimation Error distribution for $z = 8.5$ m level



Camera Coverage Spacial Distribution for $z = 8.5$ m level

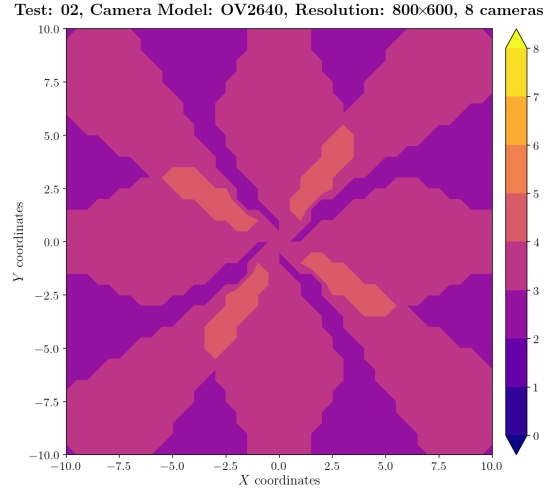


Figure A.287: Estimation error contour map for test 02 with 8 cameras using resolution of 800×600 pixels, at level $z = 8.500$ m.

Figure A.288: Camera coverage for test 02 at level $z = 8.500$ m and resolution 800×600.

Error maps for resolution 1024×768

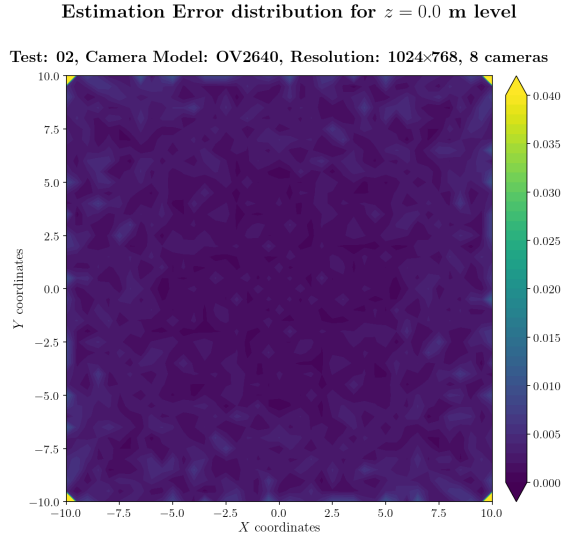


Figure A.289: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.000$ m.

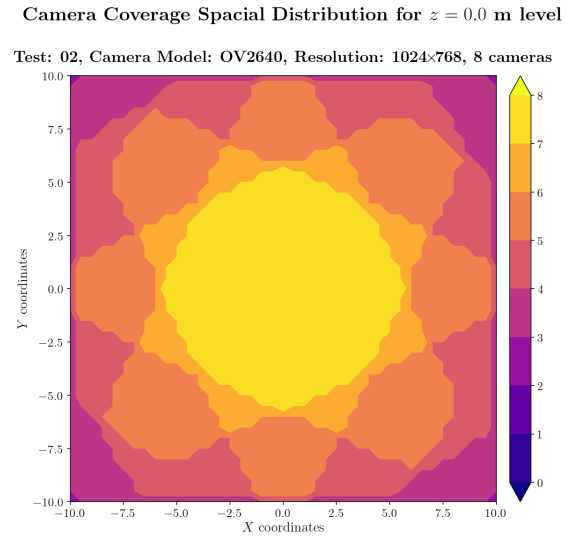


Figure A.290: Camera coverage for test 02 at level $z = 0.000$ m and resolution 1024×768.

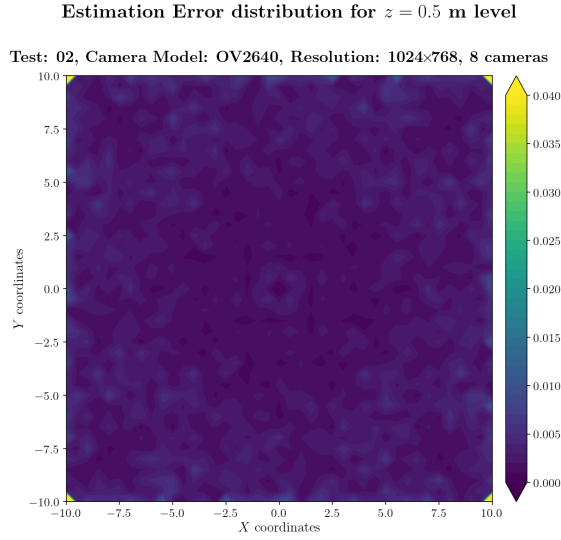


Figure A.291: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.500$ m.

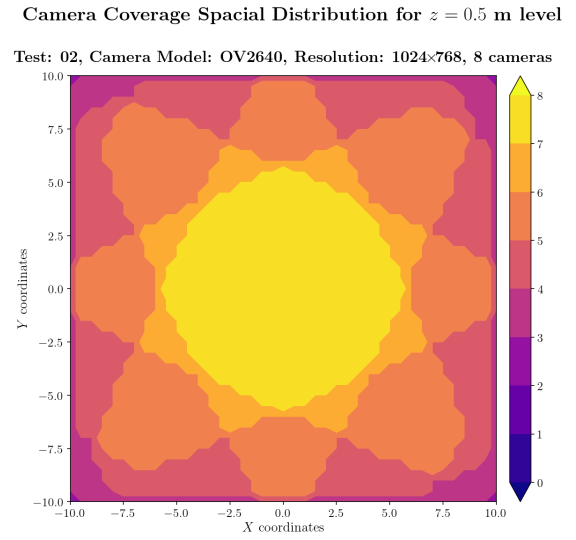
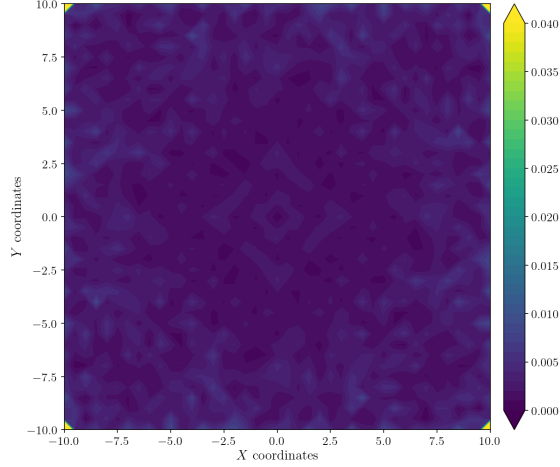


Figure A.292: Camera coverage for test 02 at level $z = 0.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

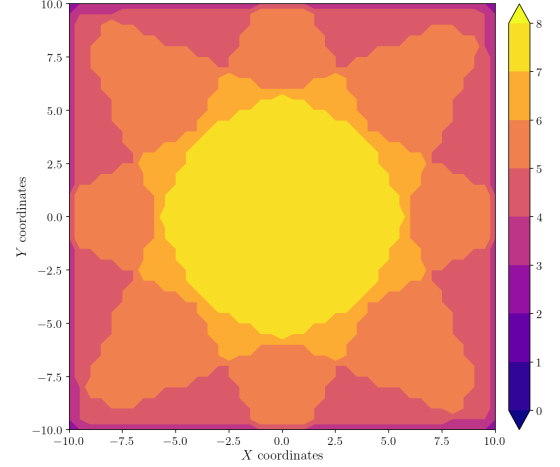
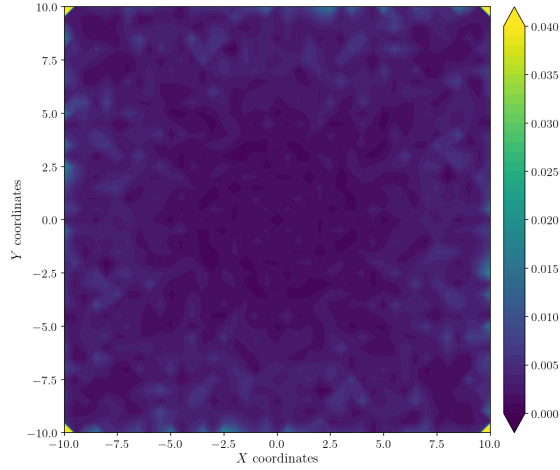


Figure A.293: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.000$ m.

Figure A.294: Camera coverage for test 02 at level $z = 1.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

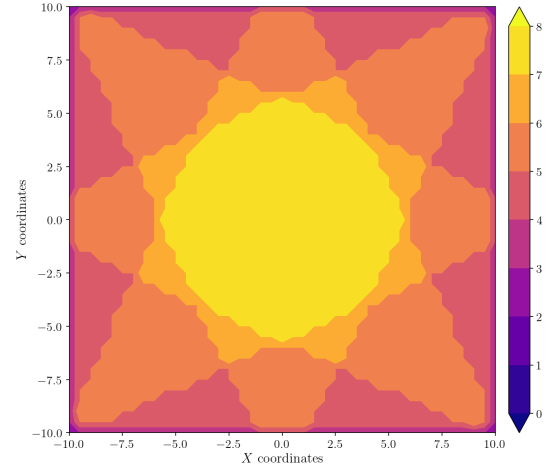


Figure A.295: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.500$ m.

Figure A.296: Camera coverage for test 02 at level $z = 1.500$ m and resolution 1024×768.

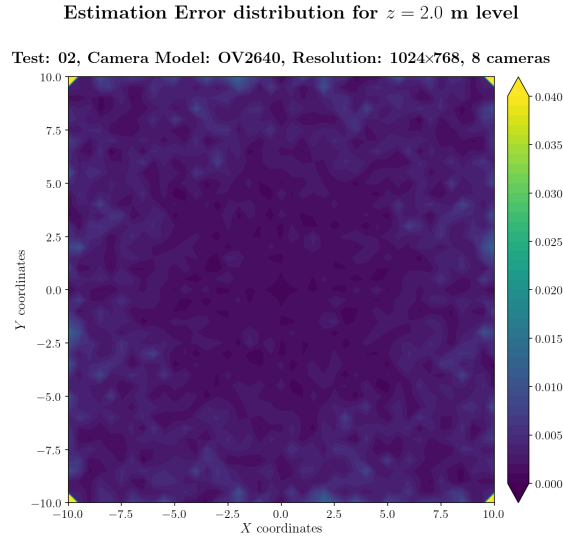


Figure A.297: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.000$ m.

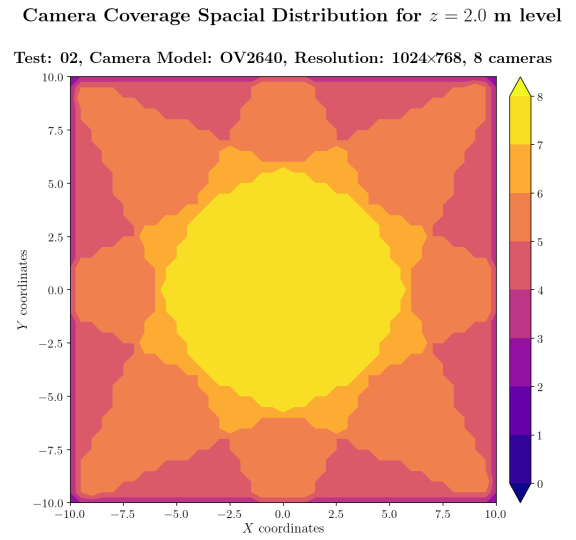


Figure A.298: Camera coverage for test 02 at level $z = 2.000$ m and resolution 1024×768.

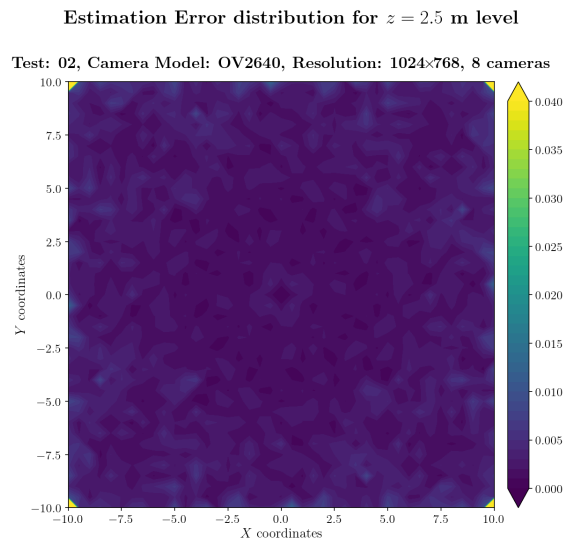


Figure A.299: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.500$ m.

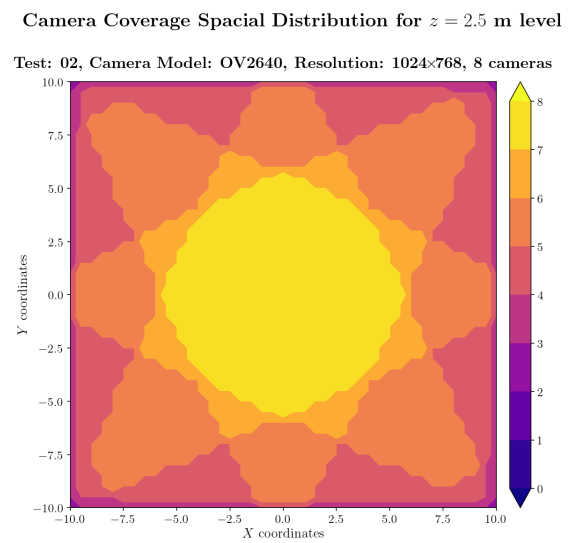
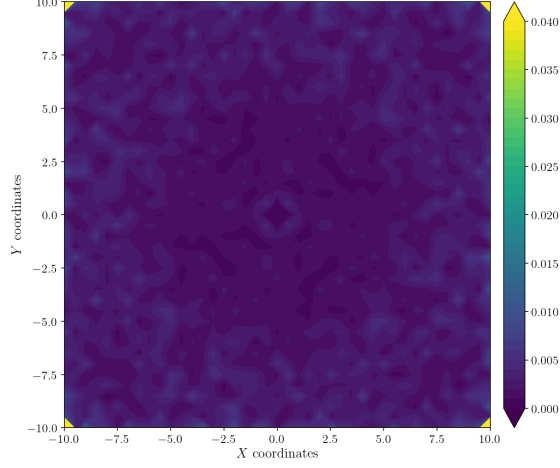


Figure A.300: Camera coverage for test 02 at level $z = 2.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

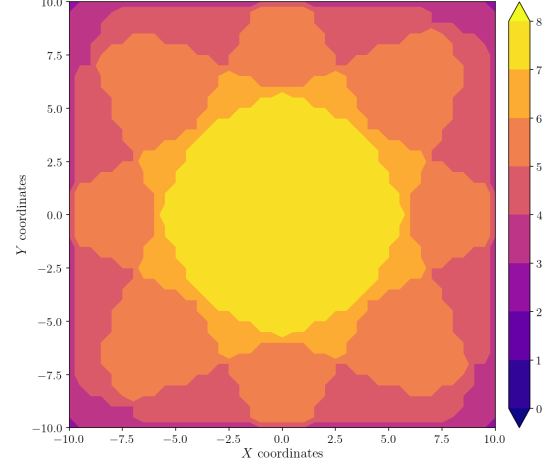
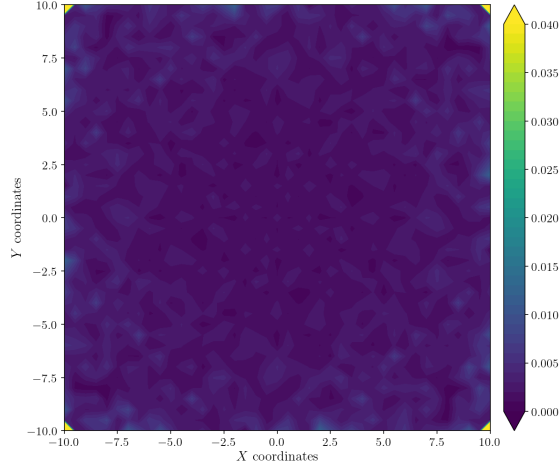


Figure A.301: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.000$ m.

Figure A.302: Camera coverage for test 02 at level $z = 3.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

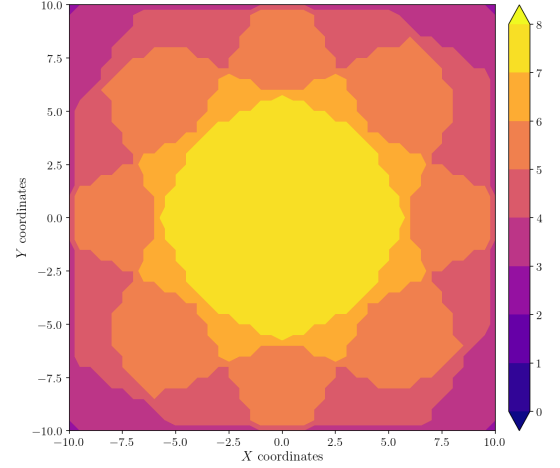


Figure A.303: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.500$ m.

Figure A.304: Camera coverage for test 02 at level $z = 3.500$ m and resolution 1024×768.

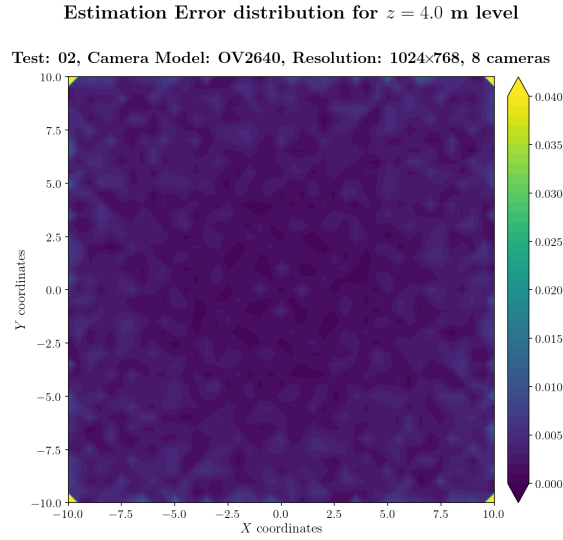


Figure A.305: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.000$ m.

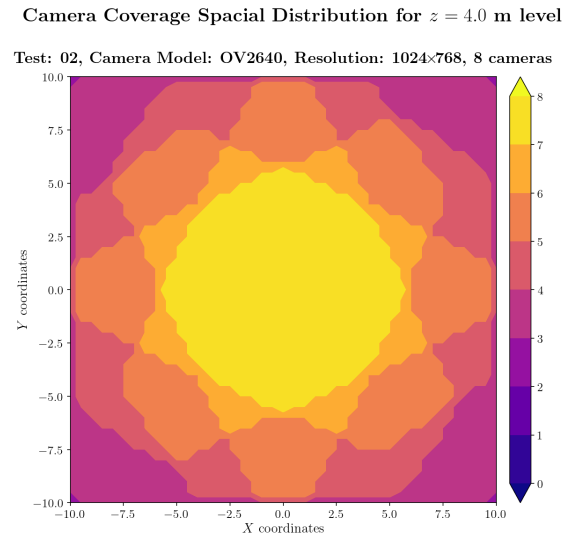


Figure A.306: Camera coverage for test 02 at level $z = 4.000$ m and resolution 1024×768.

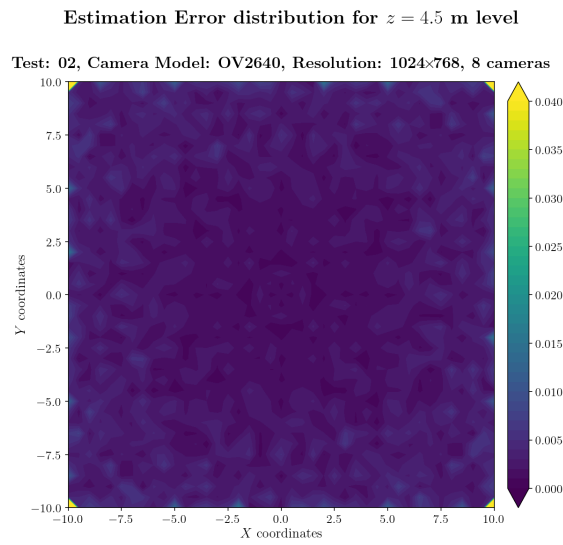


Figure A.307: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.500$ m.

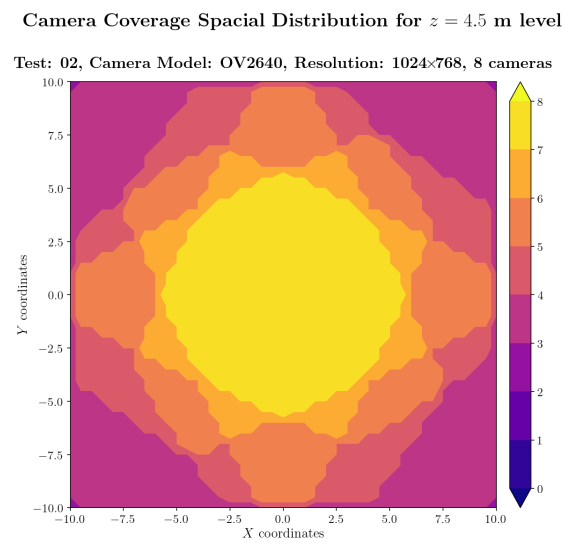
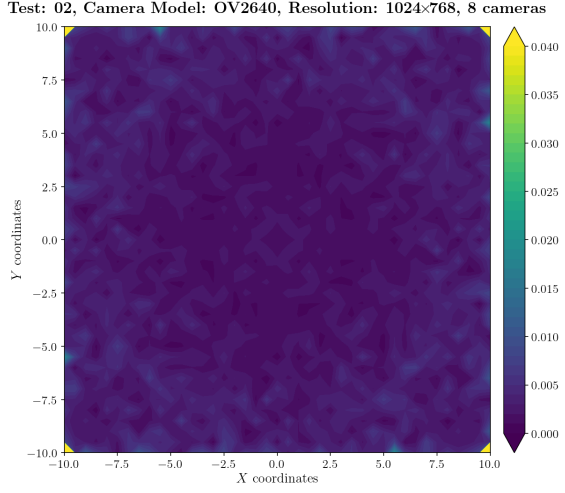


Figure A.308: Camera coverage for test 02 at level $z = 4.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 5.0$ m level



Camera Coverage Spacial Distribution for $z = 5.0$ m level

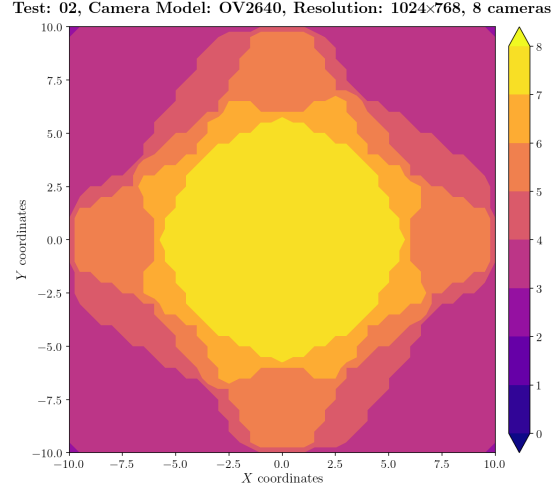
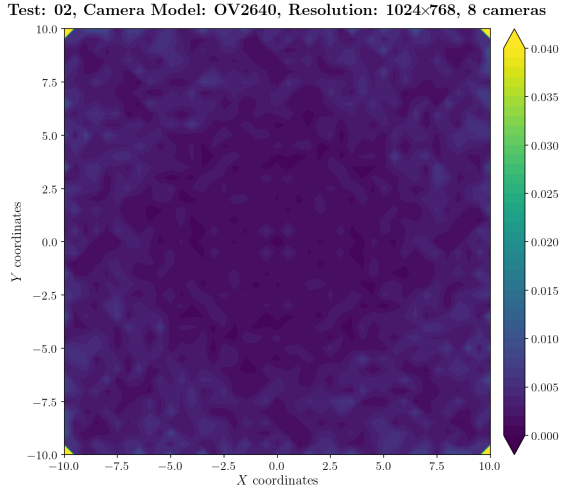


Figure A.309: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 5.000$ m.

Figure A.310: Camera coverage for test 02 at level $z = 5.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 5.5$ m level



Camera Coverage Spacial Distribution for $z = 5.5$ m level

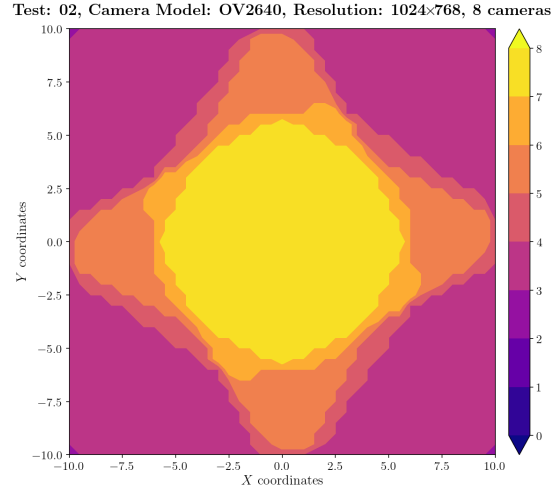


Figure A.311: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 5.500$ m.

Figure A.312: Camera coverage for test 02 at level $z = 5.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 6.0$ m level

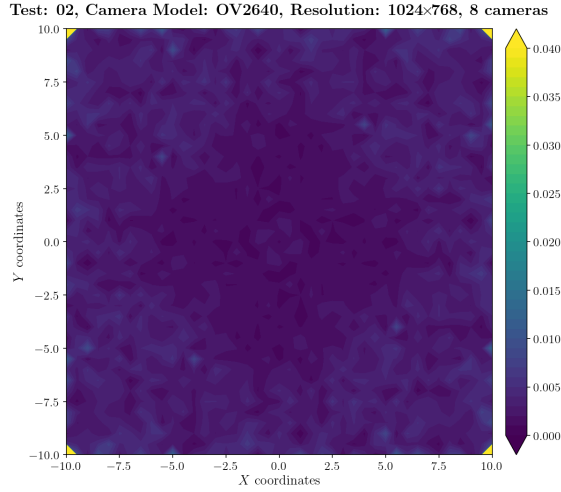


Figure A.313: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 6.000$ m.

Camera Coverage Spatial Distribution for $z = 6.0$ m level

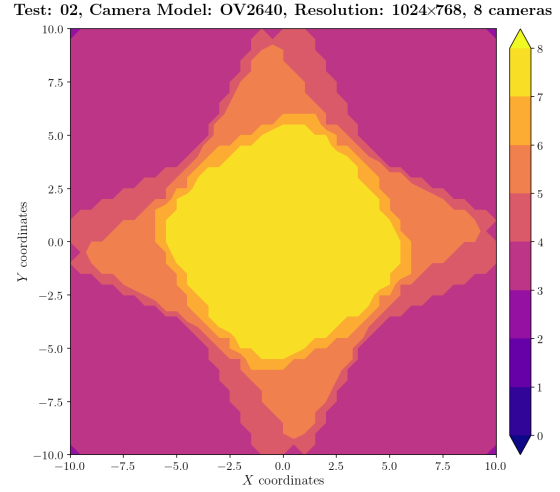


Figure A.314: Camera coverage for test 02 at level $z = 6.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 6.5$ m level

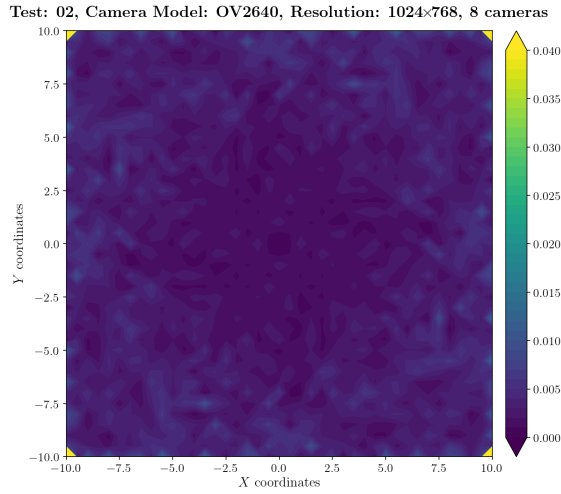


Figure A.315: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 6.500$ m.

Camera Coverage Spatial Distribution for $z = 6.5$ m level

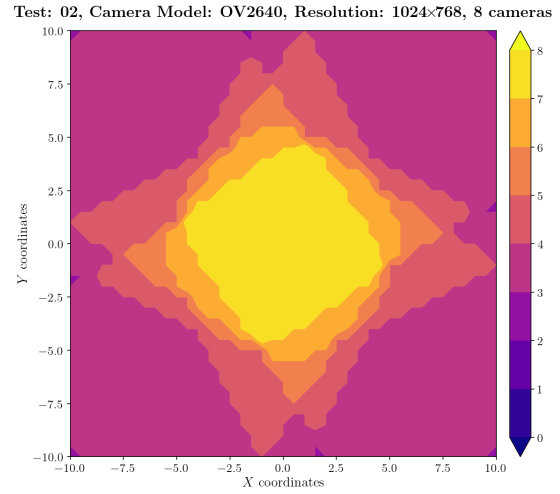
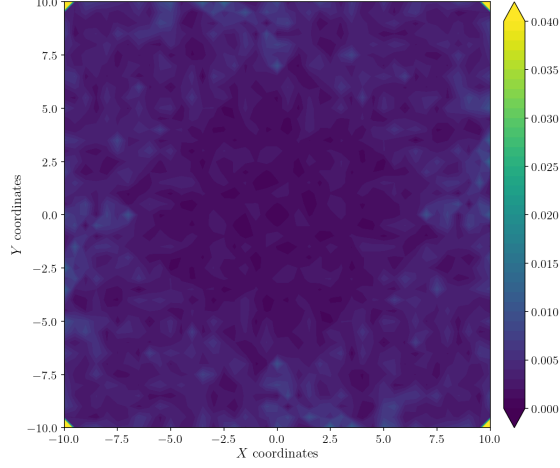


Figure A.316: Camera coverage for test 02 at level $z = 6.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 7.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

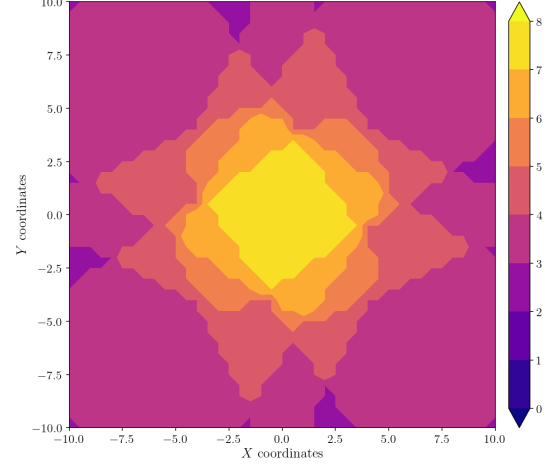
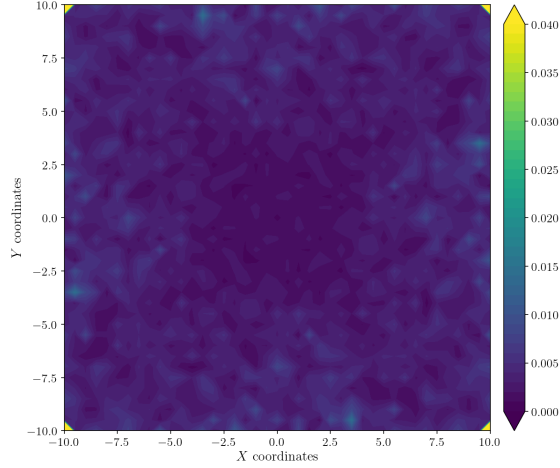


Figure A.317: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 7.000$ m.

Figure A.318: Camera coverage for test 02 at level $z = 7.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 7.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

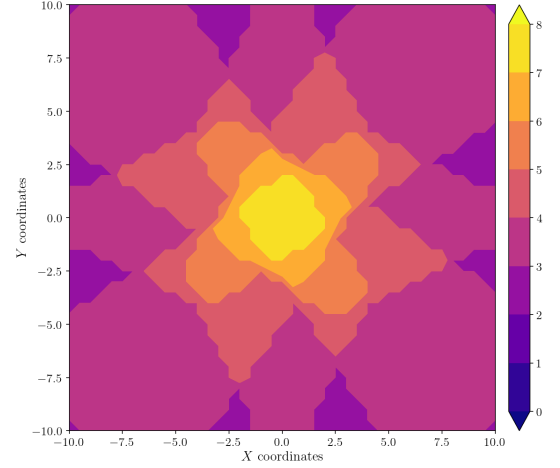


Figure A.319: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 7.500$ m.

Figure A.320: Camera coverage for test 02 at level $z = 7.500$ m and resolution 1024×768.

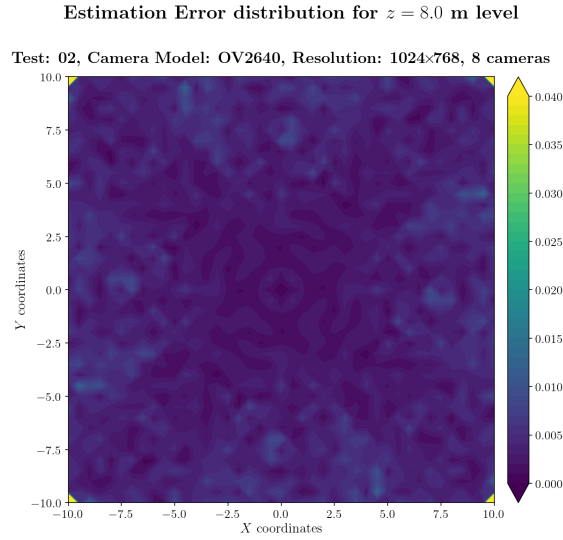


Figure A.321: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 8.000$ m.

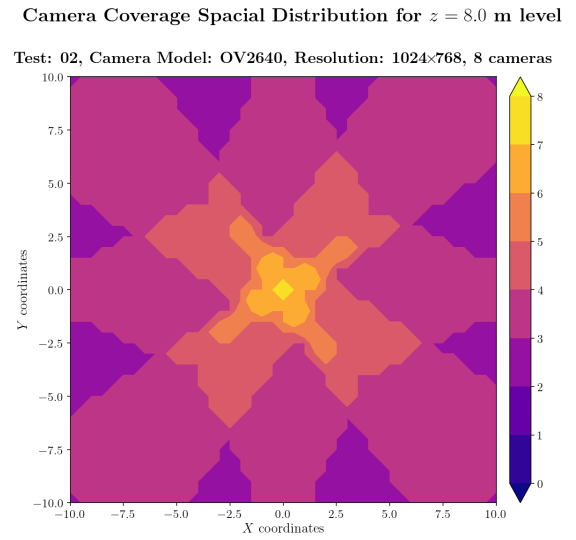


Figure A.322: Camera coverage for test 02 at level $z = 8.000$ m and resolution 1024×768.

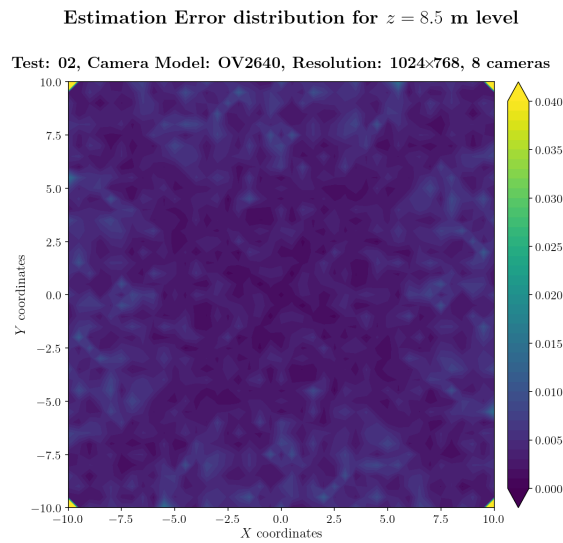


Figure A.323: Estimation error contour map for test 02 with 8 cameras using resolution of 1024×768 pixels, at level $z = 8.500$ m.

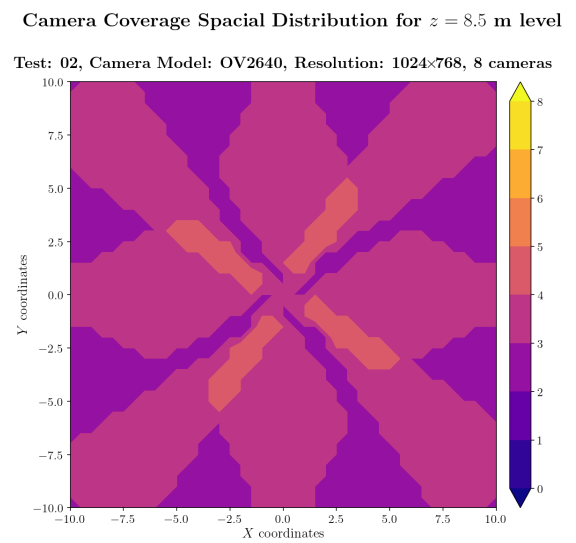


Figure A.324: Camera coverage for test 02 at level $z = 8.500$ m and resolution 1024×768.

Error maps for resolution 1600×1200

Estimation Error distribution for $z = 0.0$ m level

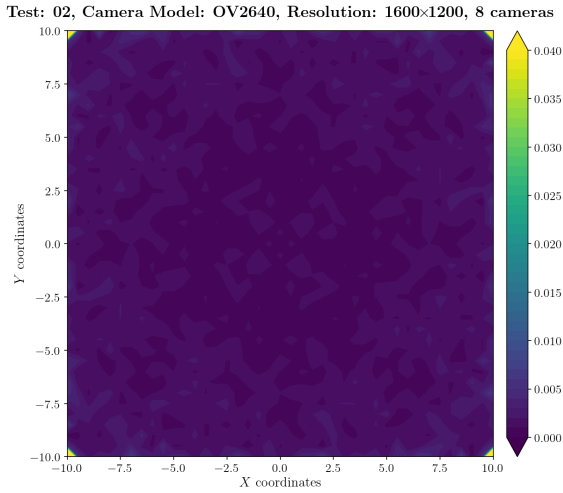


Figure A.325: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

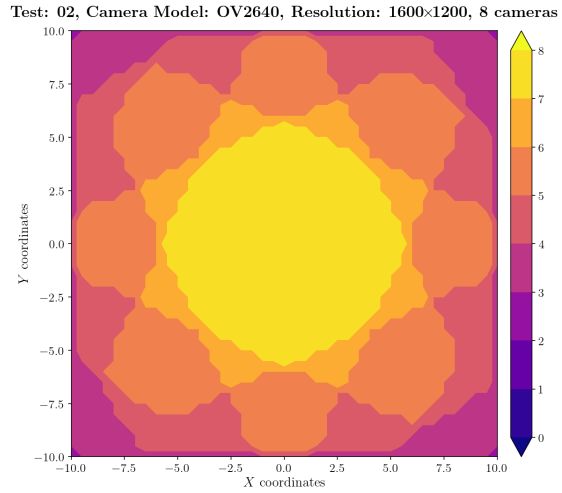


Figure A.326: Camera coverage for test 02 at level $z = 0.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 0.5$ m level

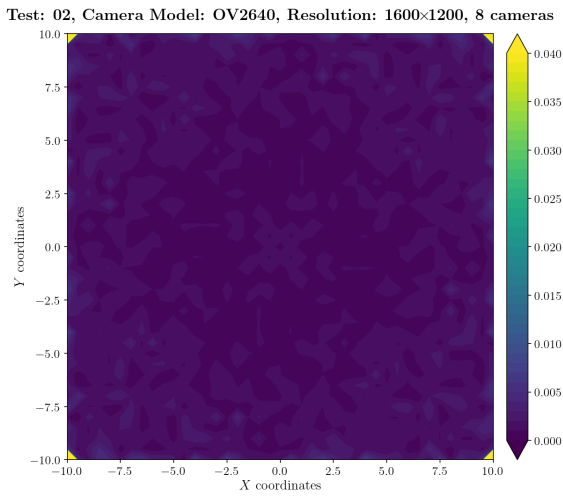


Figure A.327: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.500$ m.

Camera Coverage Spacial Distribution for $z = 0.5$ m level

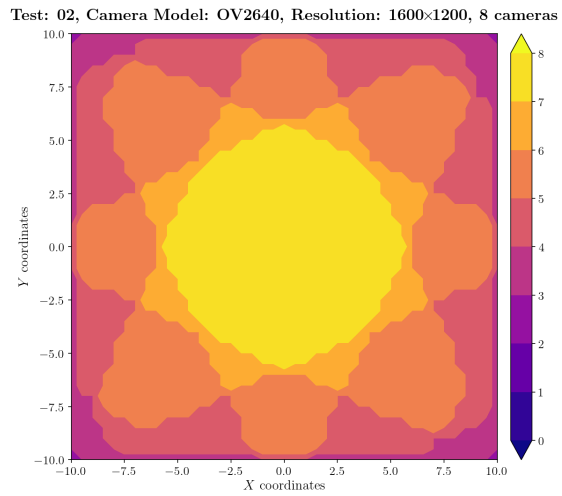


Figure A.328: Camera coverage for test 02 at level $z = 0.500$ m and resolution 1600×1200.

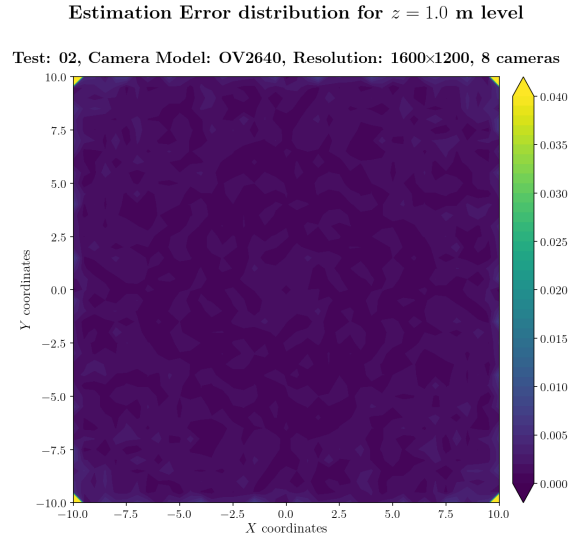


Figure A.329: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.000$ m.

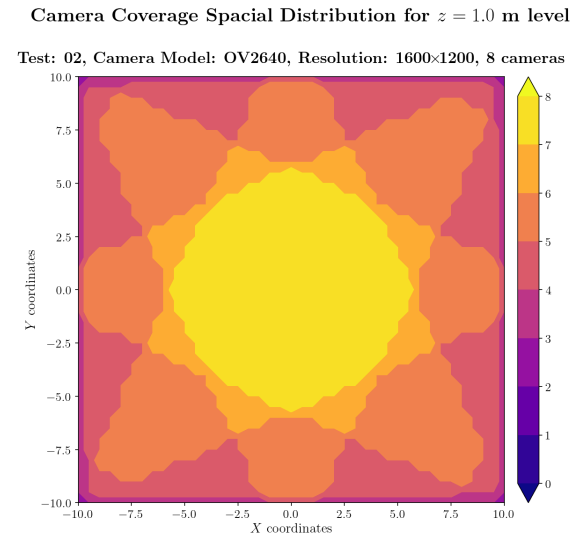


Figure A.330: Camera coverage for test 02 at level $z = 1.000$ m and resolution 1600×1200.

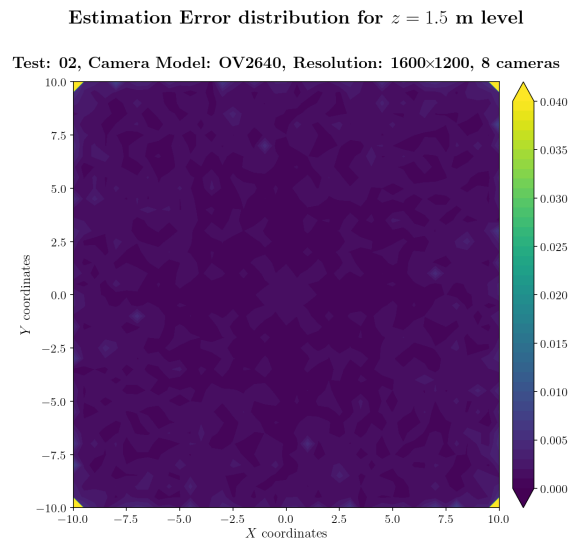


Figure A.331: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.500$ m.

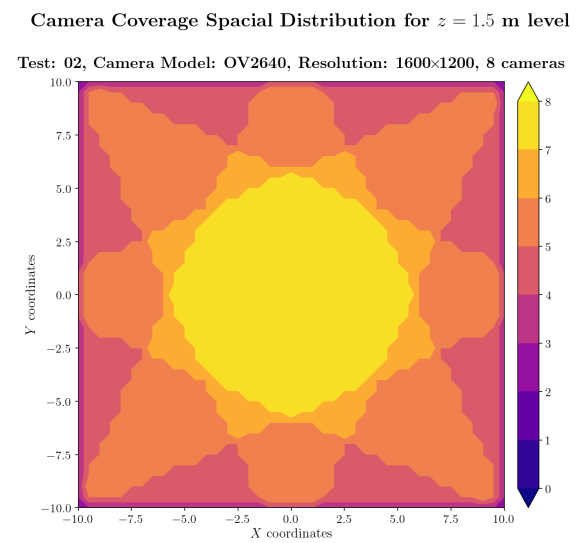
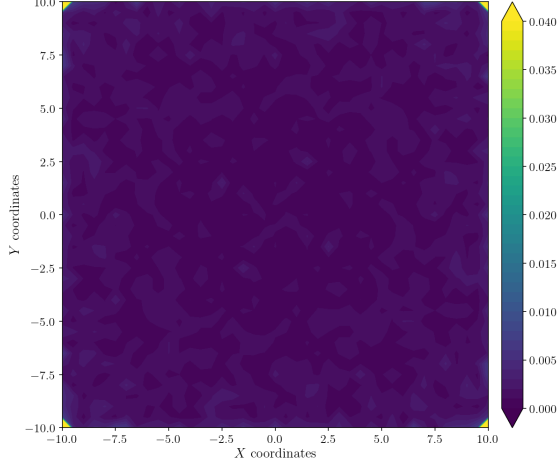


Figure A.332: Camera coverage for test 02 at level $z = 1.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 2.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 02, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

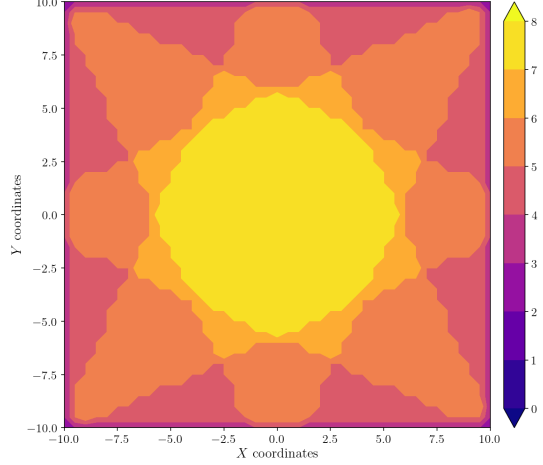
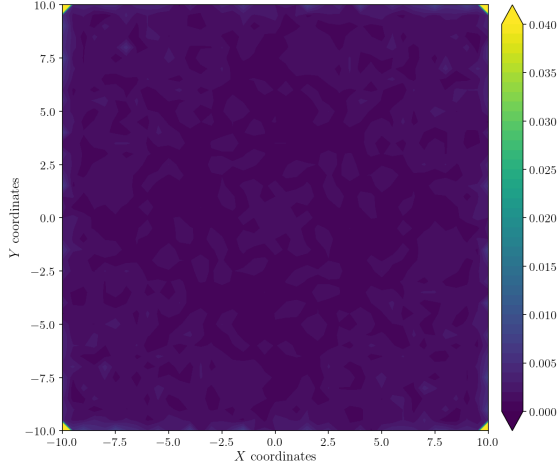


Figure A.333: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.000$ m.

Figure A.334: Camera coverage for test 02 at level $z = 2.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 2.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.5$ m level

Test: 02, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

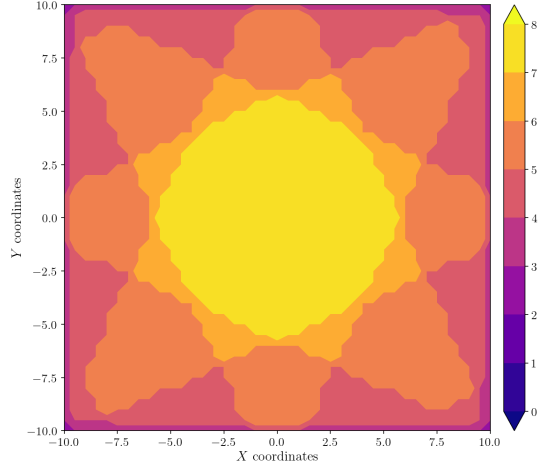


Figure A.335: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.500$ m.

Figure A.336: Camera coverage for test 02 at level $z = 2.500$ m and resolution 1600×1200.

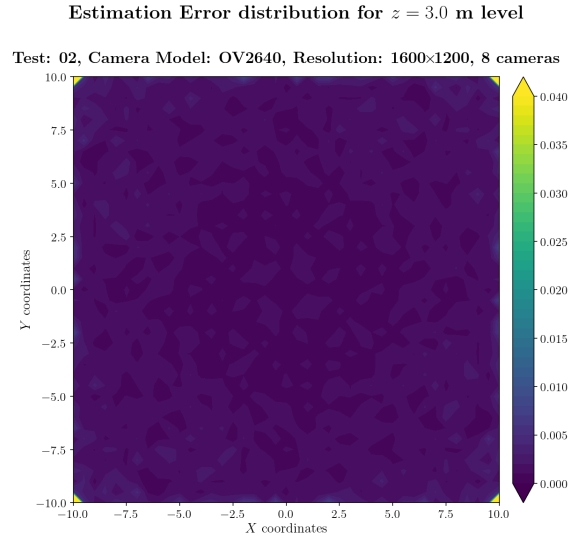


Figure A.337: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.000$ m.

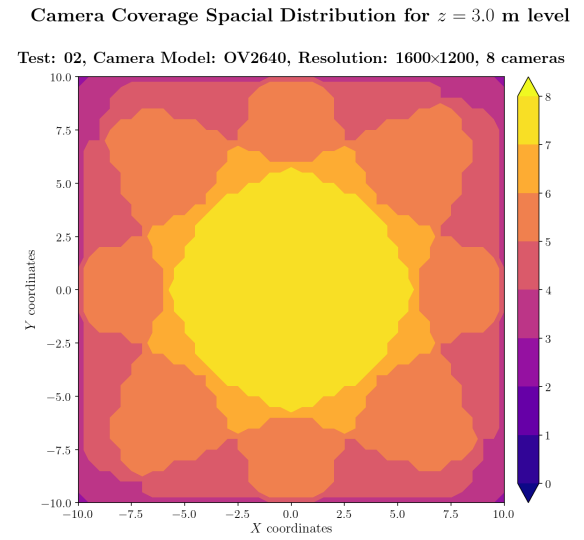


Figure A.338: Camera coverage for test 02 at level $z = 3.000$ m and resolution 1600×1200.

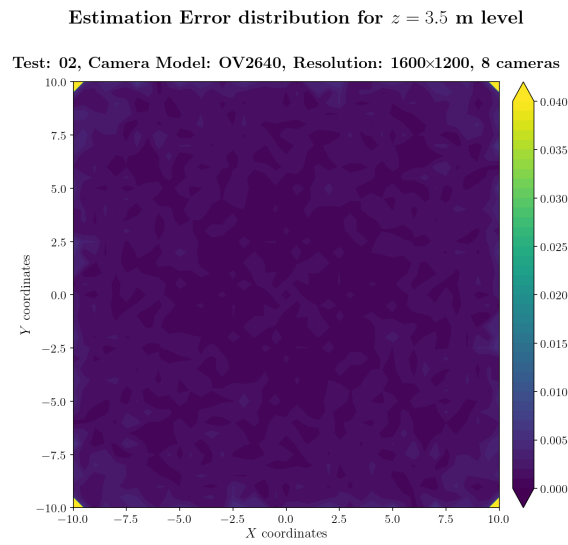


Figure A.339: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.500$ m.

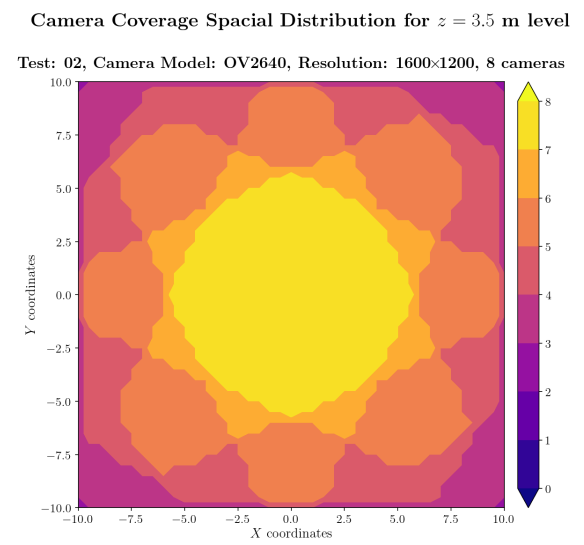
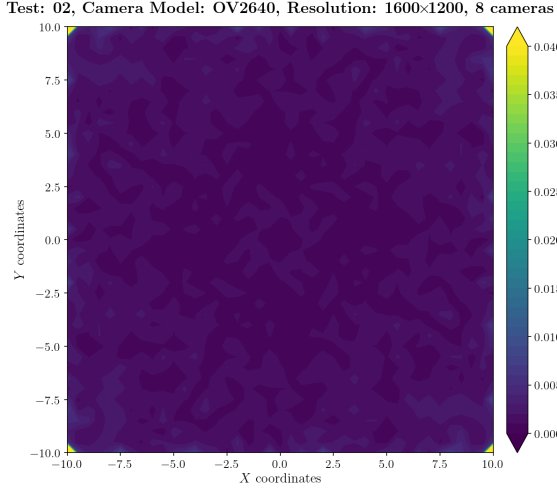


Figure A.340: Camera coverage for test 02 at level $z = 3.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 4.0$ m level



Camera Coverage Spacial Distribution for $z = 4.0$ m level

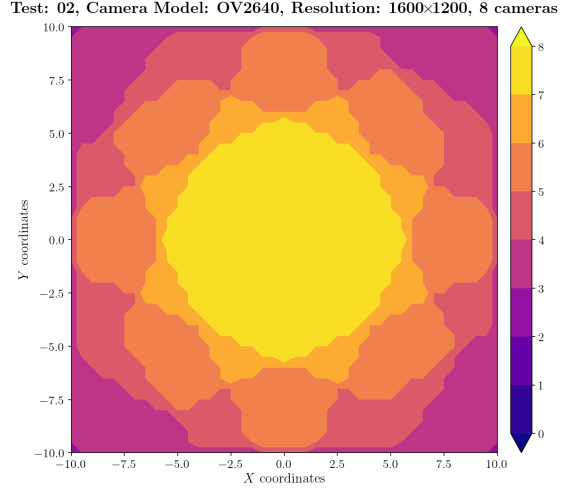
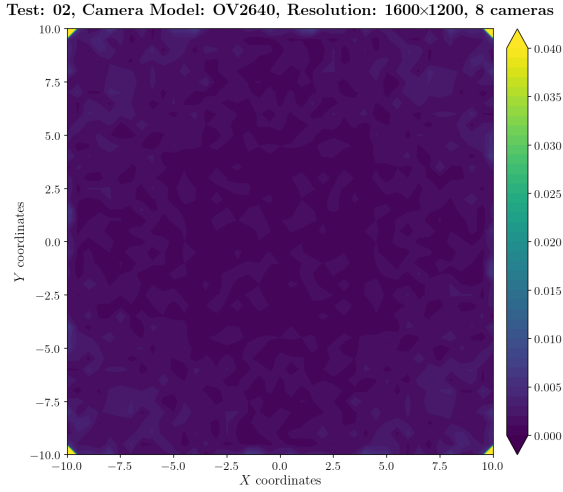


Figure A.341: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.000$ m.

Figure A.342: Camera coverage for test 02 at level $z = 4.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 4.5$ m level



Camera Coverage Spacial Distribution for $z = 4.5$ m level

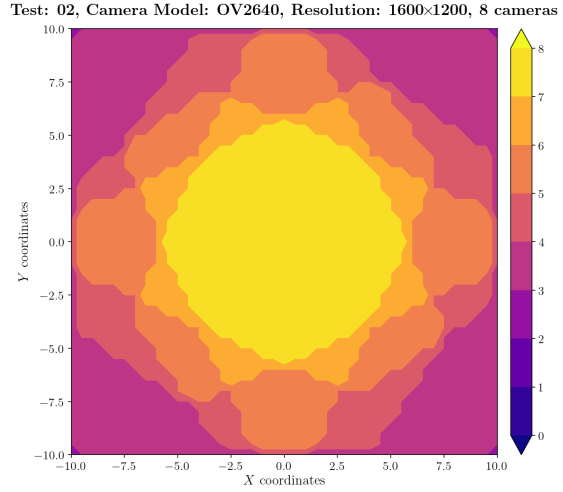


Figure A.343: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.500$ m.

Figure A.344: Camera coverage for test 02 at level $z = 4.500$ m and resolution 1600×1200.

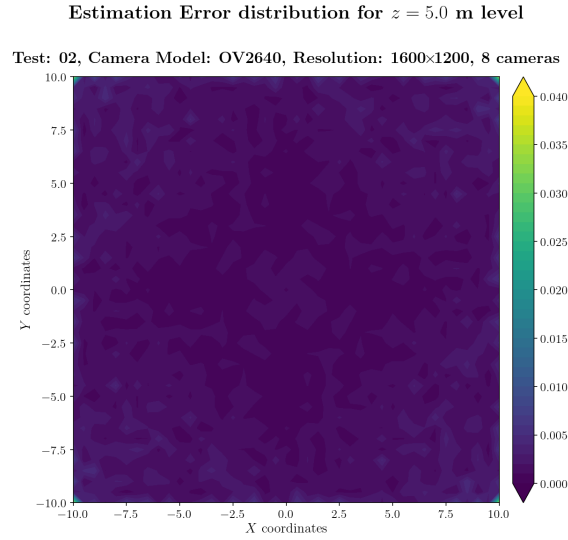


Figure A.345: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 5.000$ m.

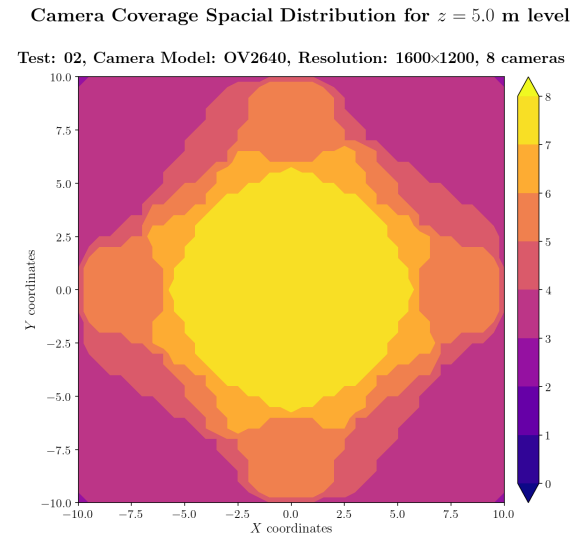


Figure A.346: Camera coverage for test 02 at level $z = 5.000$ m and resolution 1600×1200.

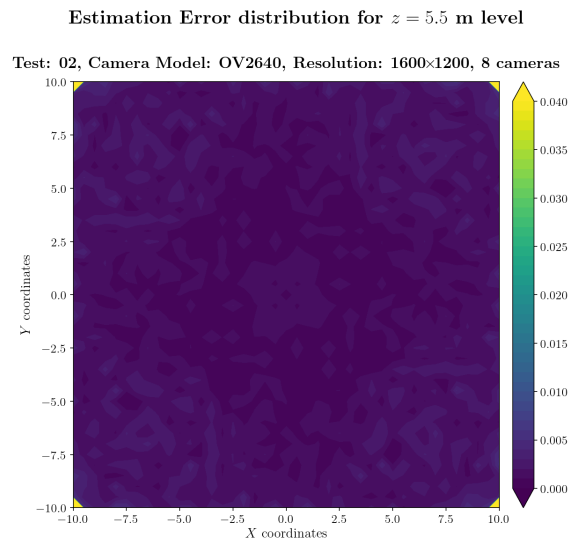


Figure A.347: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 5.500$ m.

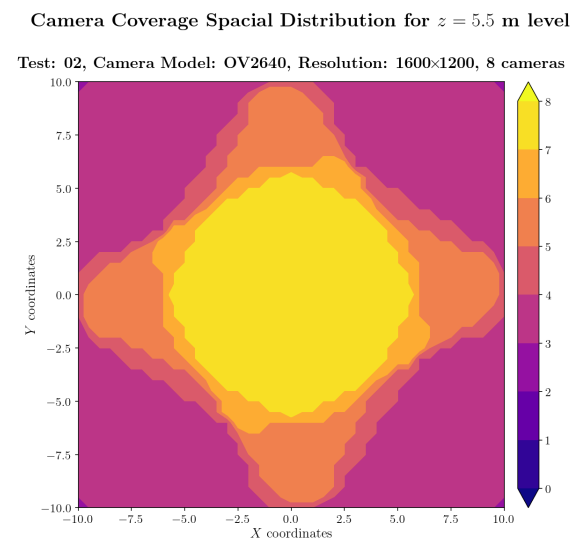


Figure A.348: Camera coverage for test 02 at level $z = 5.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 6.0$ m level

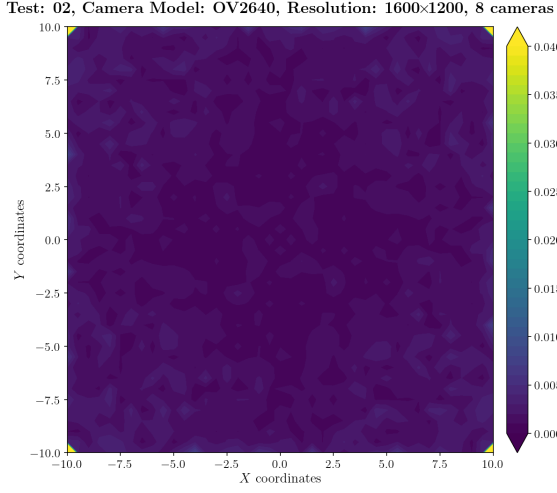


Figure A.349: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 6.000$ m.

Camera Coverage Spacial Distribution for $z = 6.0$ m level

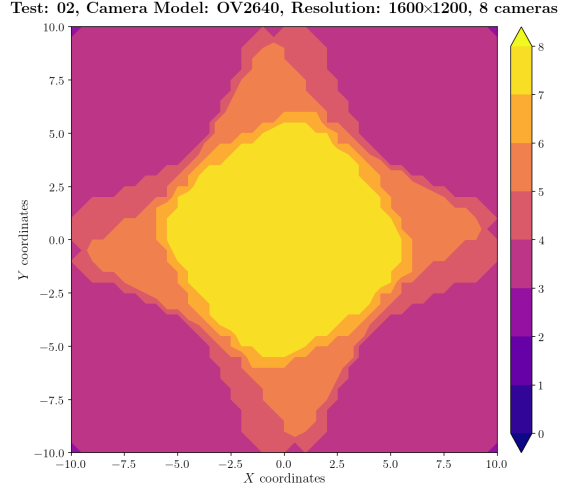


Figure A.350: Camera coverage for test 02 at level $z = 6.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 6.5$ m level

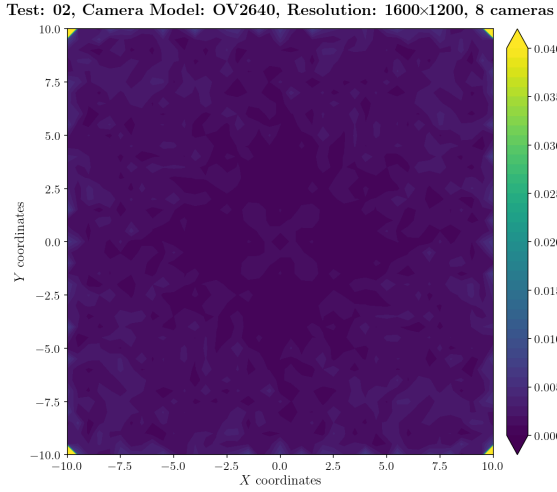


Figure A.351: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 6.500$ m.

Camera Coverage Spacial Distribution for $z = 6.5$ m level

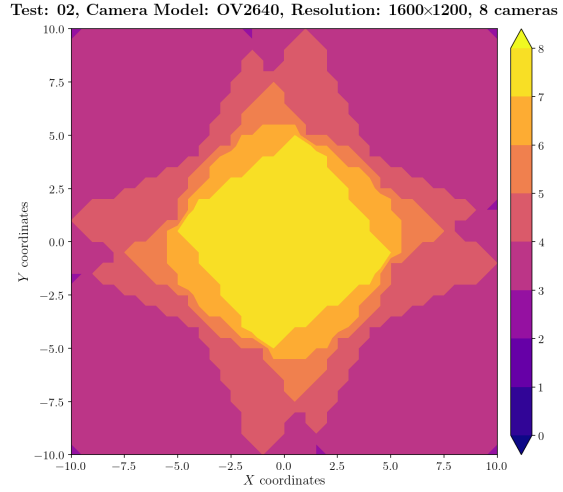


Figure A.352: Camera coverage for test 02 at level $z = 6.500$ m and resolution 1600×1200.

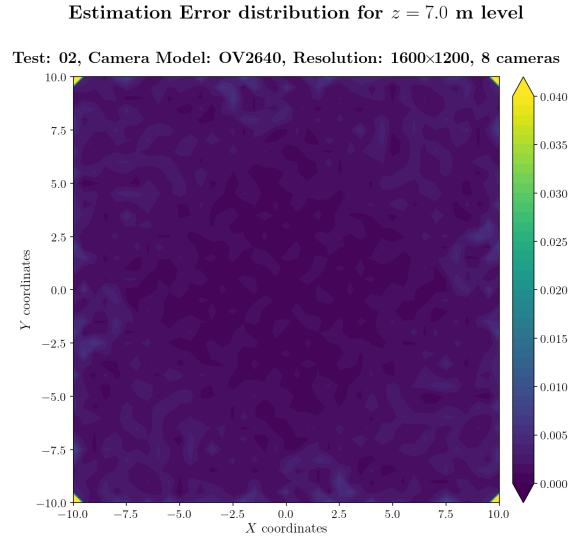


Figure A.353: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 7.000$ m.

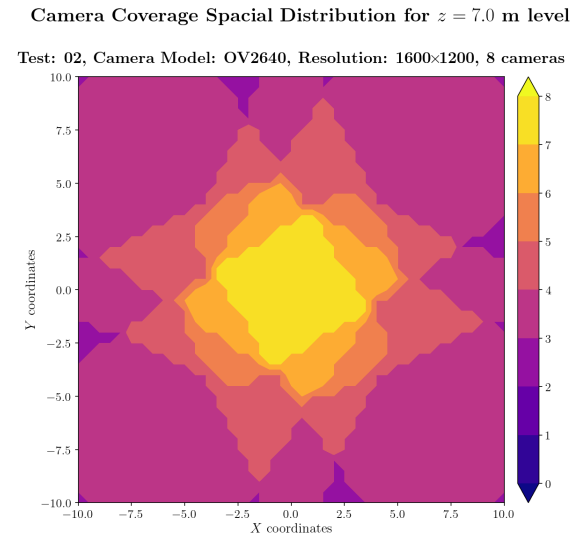


Figure A.354: Camera coverage for test 02 at level $z = 7.000$ m and resolution 1600×1200.

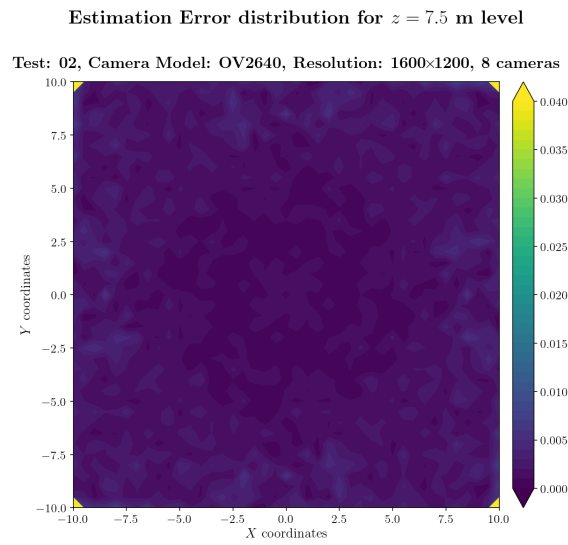


Figure A.355: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 7.500$ m.

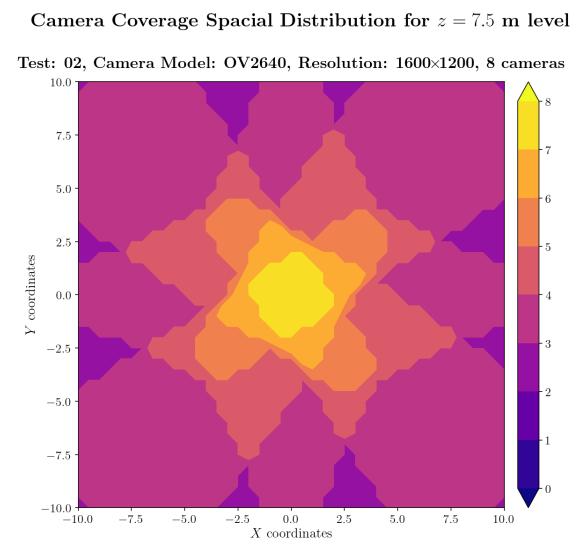
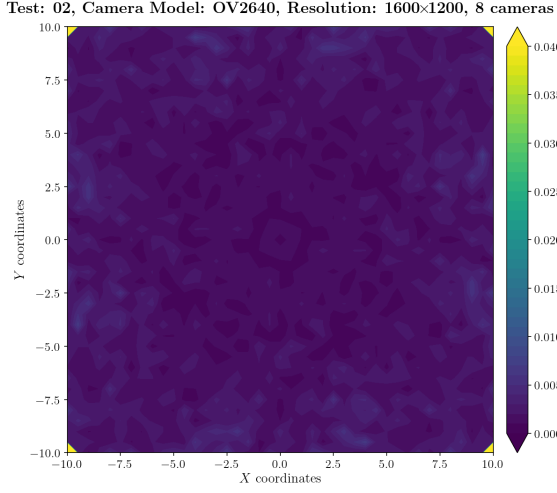


Figure A.356: Camera coverage for test 02 at level $z = 7.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 8.0$ m level



Camera Coverage Spacial Distribution for $z = 8.0$ m level

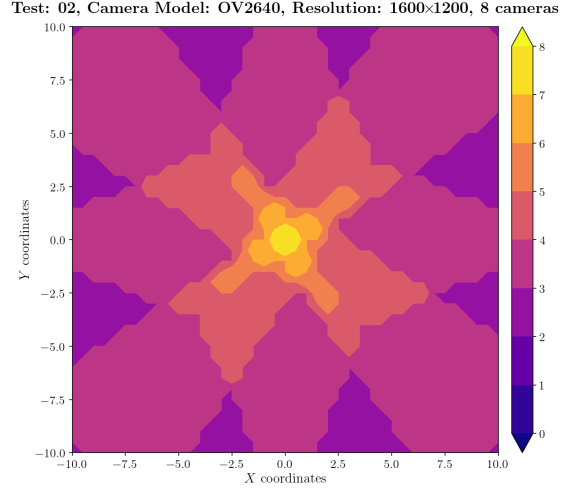
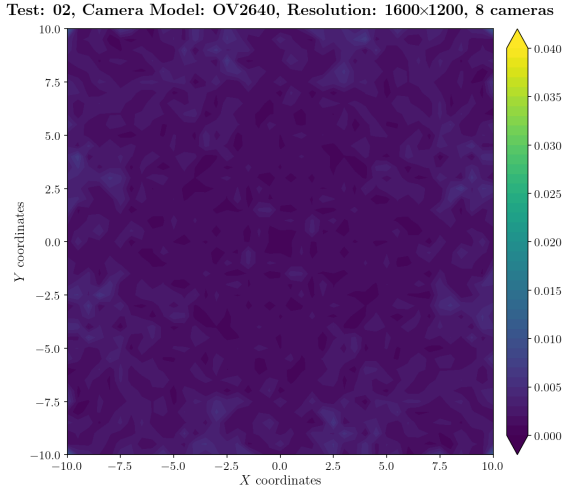


Figure A.357: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 8.000$ m.

Figure A.358: Camera coverage for test 02 at level $z = 8.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 8.5$ m level



Camera Coverage Spacial Distribution for $z = 8.5$ m level

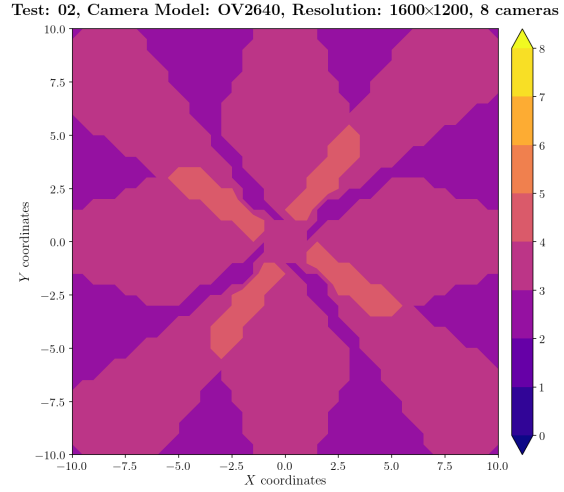


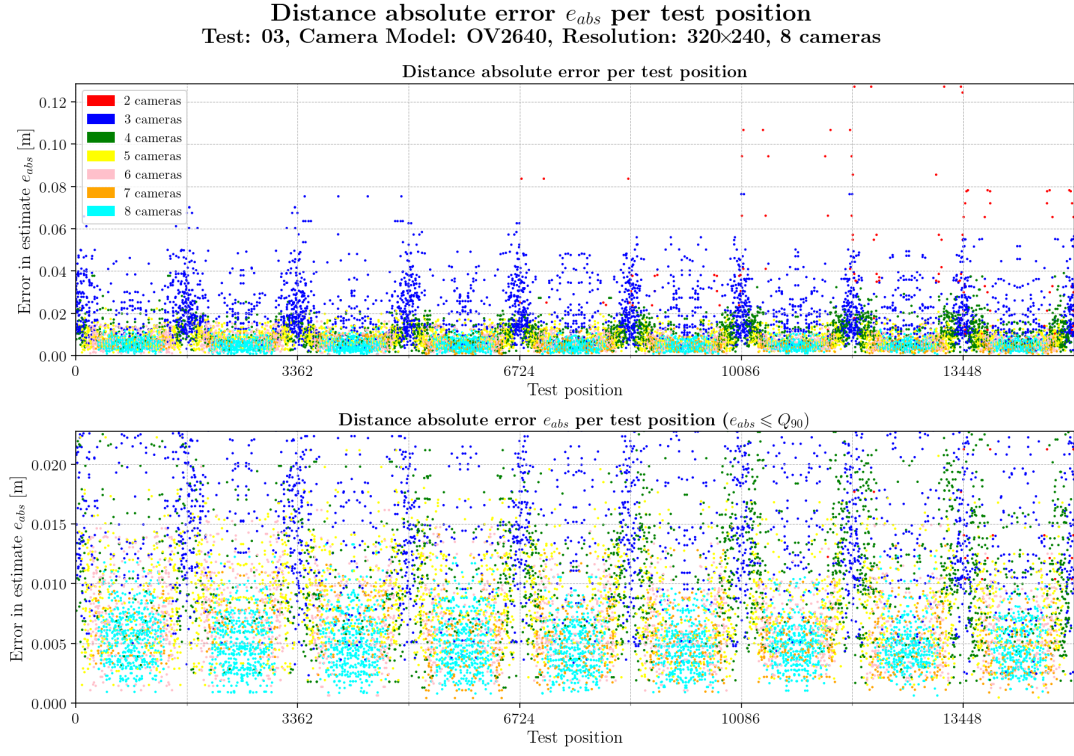
Figure A.359: Estimation error contour map for test 02 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 8.500$ m.

Figure A.360: Camera coverage for test 02 at level $z = 8.500$ m and resolution 1600×1200.

A.3 Test 03

Absolute Error per iteration and resolution

Resolution: 320×240



Error per iteration for resolution 320×240 and test03

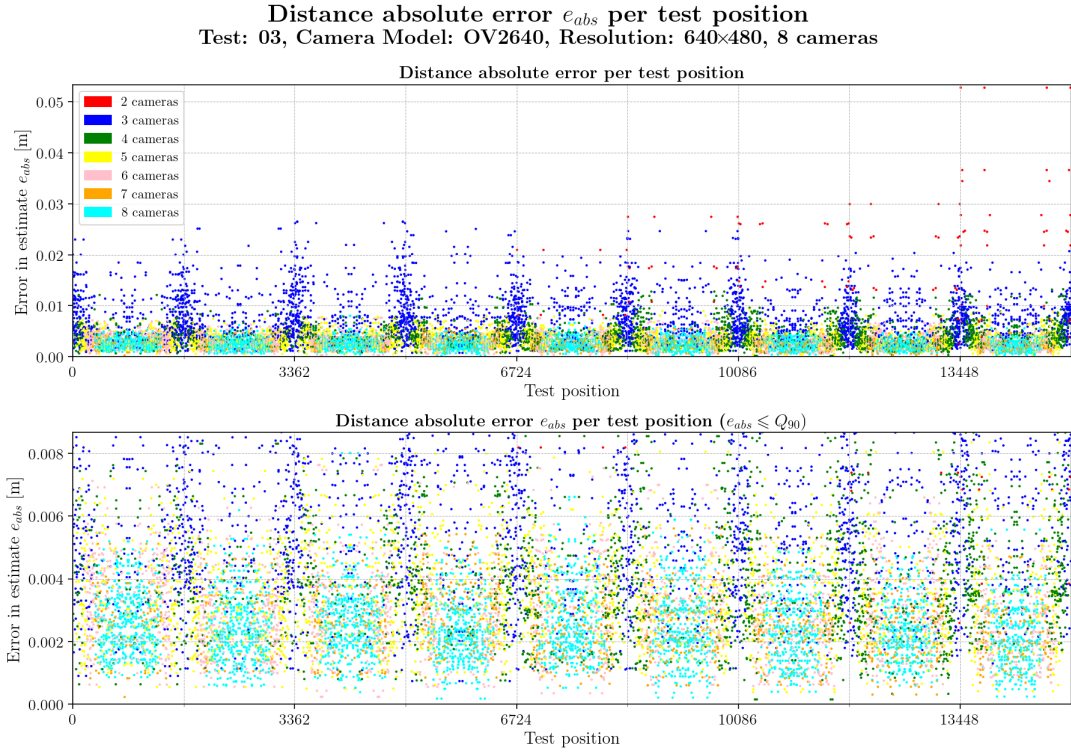
Table A.21: Statistics per number of cameras observing the beacon for test 03 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	176	3772	2434	2454	2380	1264	4330
mean	0.0494	0.0218	0.0119	0.0086	0.0069	0.0054	0.0049
std	0.0305	0.0123	0.0060	0.0036	0.0031	0.0024	0.0021
min	0.0104	0.0014	0.0013	0.0005	0.0005	0.0004	0.0004
25%	0.0239	0.0126	0.0077	0.0059	0.0046	0.0037	0.0033
50%	0.0386	0.0189	0.0110	0.0084	0.0067	0.0052	0.0047
75%	0.0655	0.0286	0.0150	0.0110	0.0090	0.0070	0.0062
max	0.1272	0.0763	0.0390	0.0222	0.0198	0.0141	0.0133

Table A.22: Global statistics and root mean square error for test 03 with resolution 320×240.

	Abs. error	Est. error
count	16810	16810
mean	0.0110	0.0238
std	0.0105	0.0084
min	0.0004	0.0001
25%	0.0049	0.0180
50%	0.0078	0.0233
75%	0.0131	0.0294
90%	0.0227	0.0345
max	0.1272	0.0700
RMSE	0.0152	

Resolution: 640×480



Error per iteration for resolution 640×480 and test03

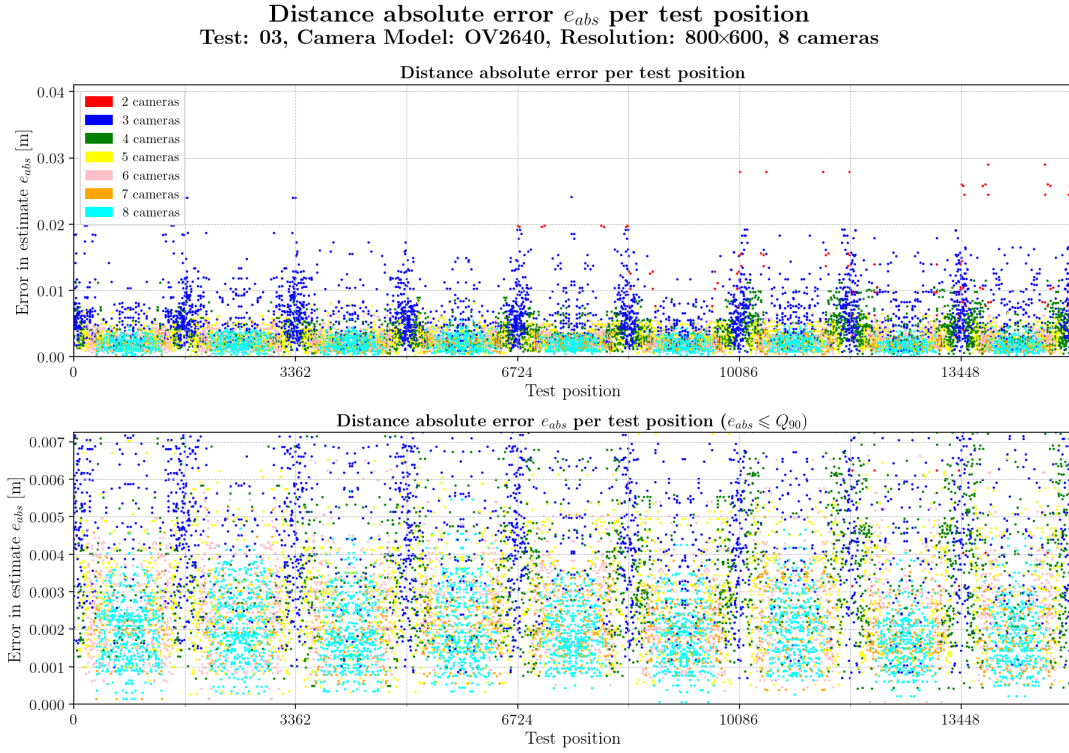
Table A.23: Statistics per number of cameras observing the beacon for test 03 with resolution 640×480.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	168	3732	2428	2404	2440	1304	4334
mean	0.0176	0.0081	0.0046	0.0035	0.0030	0.0025	0.0024
std	0.0104	0.0045	0.0026	0.0016	0.0014	0.0011	0.0011
min	0.0024	0.0007	0.0002	0.0004	0.0002	0.0002	0.0002
25%	0.0098	0.0046	0.0028	0.0022	0.0020	0.0017	0.0017
50%	0.0166	0.0072	0.0040	0.0033	0.0028	0.0025	0.0023
75%	0.0242	0.0105	0.0057	0.0045	0.0038	0.0033	0.0031
max	0.0528	0.0265	0.0184	0.0094	0.0076	0.0056	0.0080

Table A.24: Global statistics and root mean square error for test 03 with resolution 640×480.

	Abs. error	Est. error
count	16810	16810
mean	0.0044	0.0096
std	0.0037	0.0033
min	0.0002	0.0001
25%	0.0022	0.0074
50%	0.0033	0.0096
75%	0.0051	0.0118
90%	0.0087	0.0140
max	0.0528	0.0220
RMSE	0.0058	

Resolution: 800×600



Error per iteration for resolution 800×600 and test03

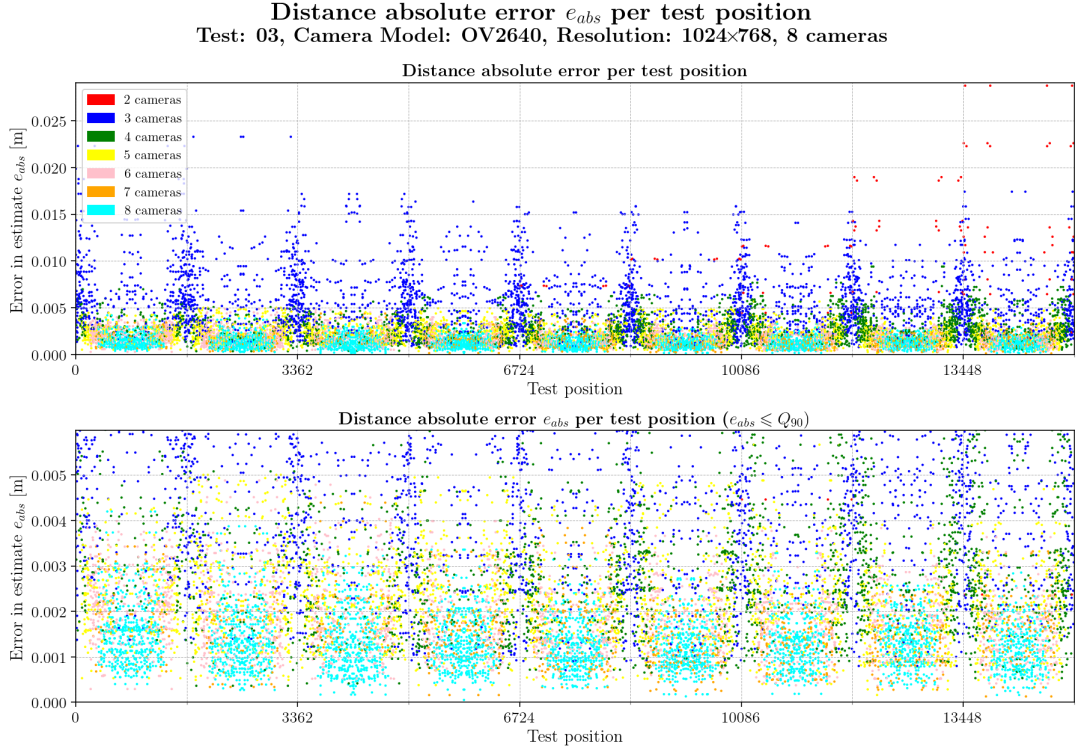
Table A.25: Statistics per number of cameras observing the beacon for test 03 with resolution 800×600.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	160	3732	2412	2420	2448	1288	4350
mean	0.0135	0.0065	0.0041	0.0030	0.0026	0.0021	0.0019
std	0.0083	0.0037	0.0023	0.0014	0.0012	0.0008	0.0009
min	0.0031	0.0007	0.0004	0.0003	0.0001	0.0001	0.0001
25%	0.0080	0.0039	0.0024	0.0020	0.0017	0.0015	0.0013
50%	0.0111	0.0057	0.0037	0.0028	0.0025	0.0020	0.0018
75%	0.0156	0.0085	0.0052	0.0038	0.0034	0.0026	0.0024
max	0.0406	0.0241	0.0136	0.0081	0.0074	0.0052	0.0057

Table A.26: Global statistics and root mean square error for test 03 with resolution 800×600.

	Abs. error	Est. error
count	16810	16810
mean	0.0036	0.0081
std	0.0030	0.0027
min	0.0001	0.0001
25%	0.0018	0.0063
50%	0.0027	0.0082
75%	0.0043	0.0099
90%	0.0072	0.0115
max	0.0406	0.0185
RMSE	0.0047	

Resolution: 1024×768



Error per iteration for resolution 1024×768 and test03

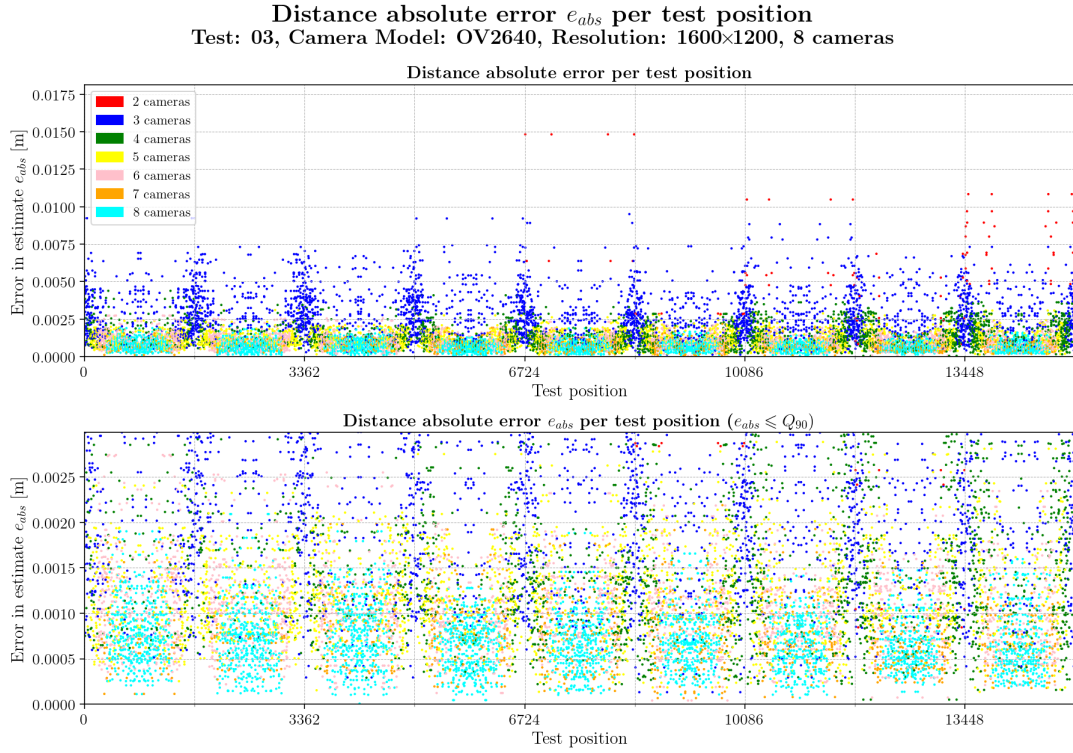
Table A.27: Statistics per number of cameras observing the beacon for test 03 with resolution 1024×768.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	152	3724	2412	2436	2440	1292	4354
mean	0.0116	0.0059	0.0030	0.0023	0.0019	0.0015	0.0013
std	0.0065	0.0034	0.0015	0.0010	0.0008	0.0007	0.0006
min	0.0024	0.0007	0.0004	0.0002	0.0001	0.0001	0.0000
25%	0.0067	0.0034	0.0020	0.0016	0.0013	0.0010	0.0009
50%	0.0108	0.0052	0.0028	0.0023	0.0018	0.0015	0.0012
75%	0.0143	0.0075	0.0038	0.0030	0.0024	0.0020	0.0016
max	0.0287	0.0233	0.0108	0.0057	0.0049	0.0044	0.0042

Table A.28: Global statistics and root mean square error for test 03 with resolution 1024×768.

	Abs. error	Est. error
count	16810	16810
mean	0.0029	0.0062
std	0.0027	0.0022
min	0.0000	0.0000
25%	0.0013	0.0048
50%	0.0021	0.0061
75%	0.0034	0.0075
90%	0.0060	0.0090
max	0.0287	0.0156
RMSE	0.0040	

Resolution: 1600×1200



Error per iteration for resolution 1600×1200 and test03

Table A.29: Statistics per number of cameras observing the beacon for test 03 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	152	3680	2432	2460	2440	1284	4362
mean	0.0060	0.0029	0.0015	0.0012	0.0010	0.0008	0.0007
std	0.0037	0.0016	0.0008	0.0005	0.0005	0.0004	0.0003
min	0.0008	0.0001	0.0001	0.0001	0.0000	0.0001	0.0000
25%	0.0030	0.0017	0.0009	0.0009	0.0007	0.0005	0.0005
50%	0.0054	0.0025	0.0014	0.0012	0.0010	0.0007	0.0007
75%	0.0080	0.0039	0.0020	0.0016	0.0013	0.0010	0.0009
max	0.0179	0.0095	0.0047	0.0031	0.0027	0.0022	0.0021

Table A.30: Global statistics and root mean square error for test 03 with resolution 1600×1200.

	Abs. error	Est. error
count	16810	16810
mean	0.0015	0.0033
std	0.0013	0.0011
min	0.0000	0.0000
25%	0.0007	0.0025
50%	0.0011	0.0032
75%	0.0018	0.0040
90%	0.0030	0.0047
max	0.0179	0.0076
RMSE	0.0020	

Error and camera coverage maps

Error maps for resolution 320×240

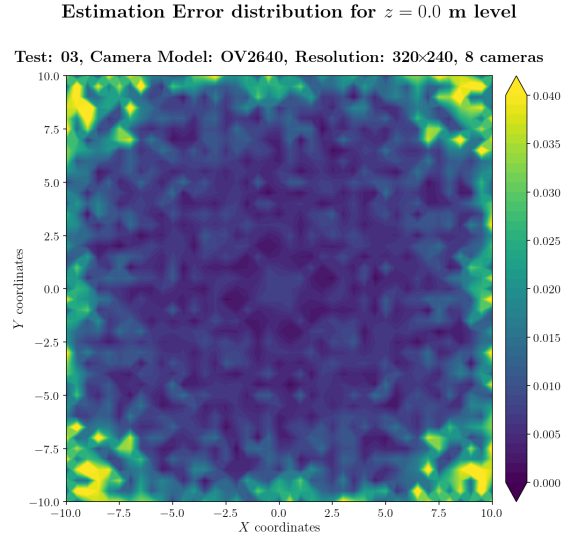


Figure A.361: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.000$ m.

Camera Coverage Spatial Distribution for $z = 0.0$ m level

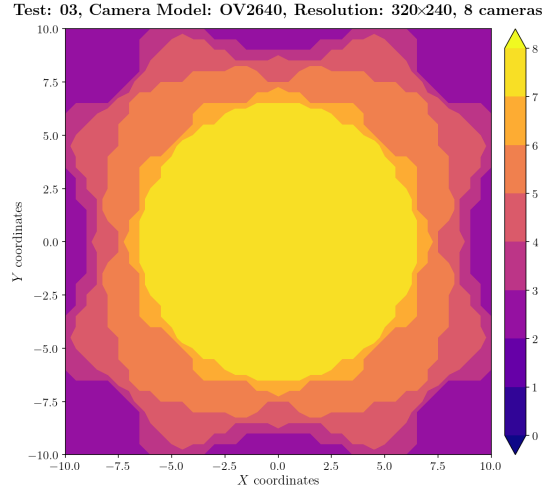


Figure A.362: Camera coverage for test 03 at level $z = 0.000$ m and resolution 320×240.

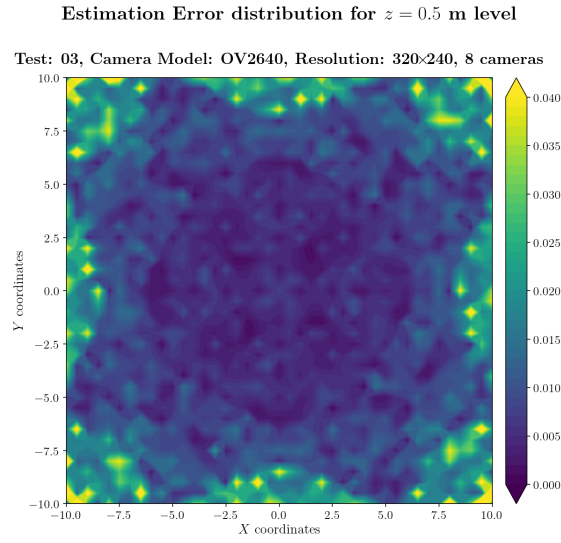


Figure A.363: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.500$ m.

Camera Coverage Spatial Distribution for $z = 0.5$ m level

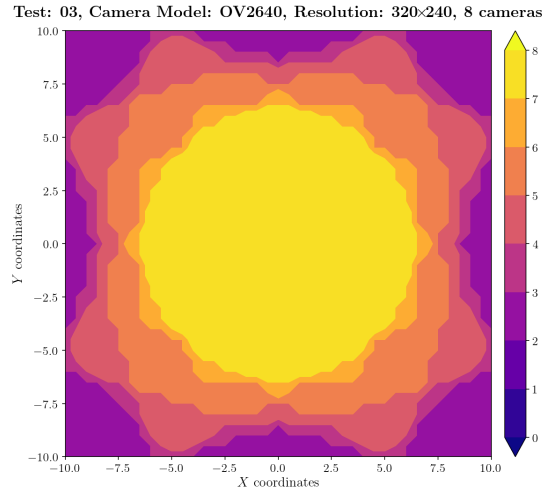
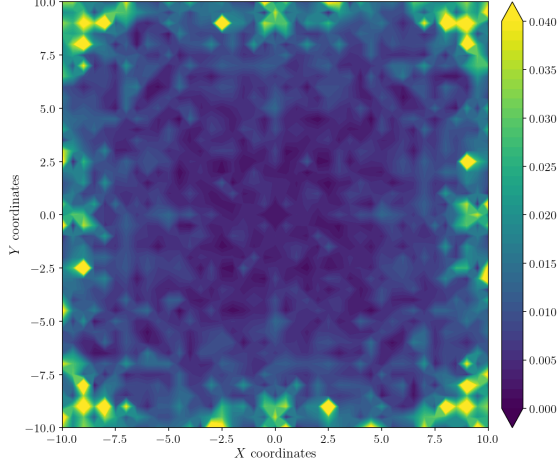


Figure A.364: Camera coverage for test 03 at level $z = 0.500$ m and resolution 320×240.

Estimation Error distribution for $z = 1.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras

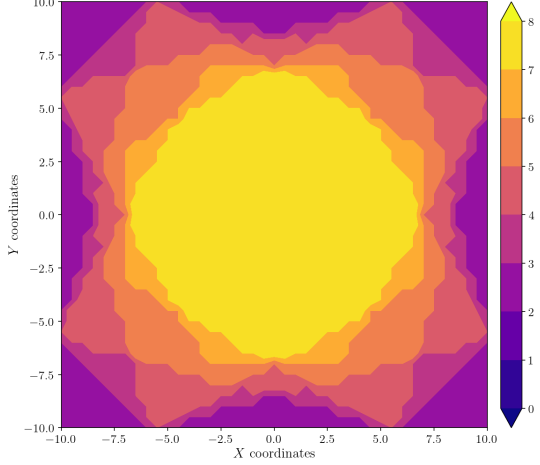
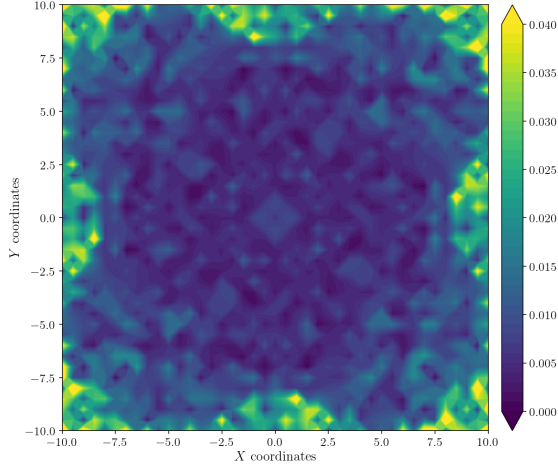


Figure A.365: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.000$ m.

Figure A.366: Camera coverage for test 03 at level $z = 1.000$ m and resolution 320×240.

Estimation Error distribution for $z = 1.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras

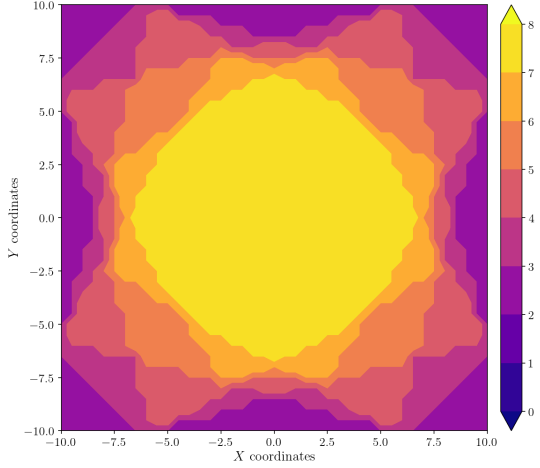


Figure A.367: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.500$ m.

Figure A.368: Camera coverage for test 03 at level $z = 1.500$ m and resolution 320×240.

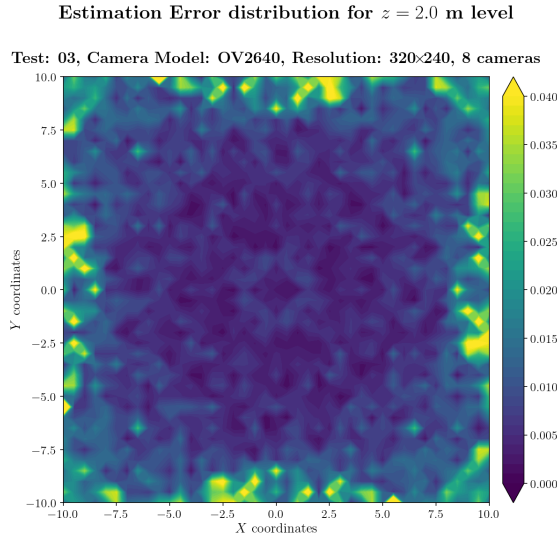


Figure A.369: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.000$ m.

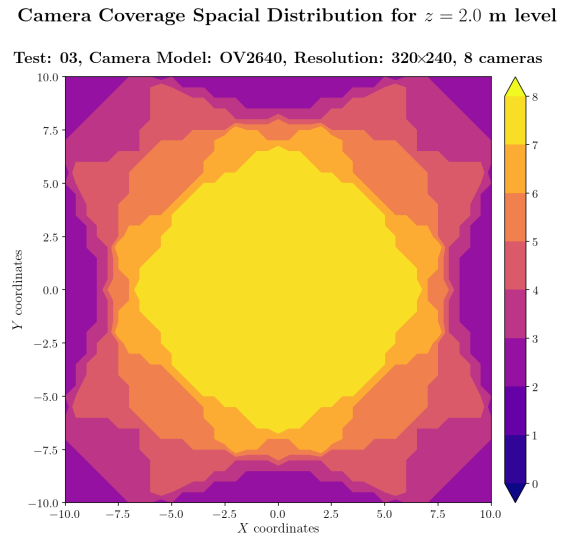


Figure A.370: Camera coverage for test 03 at level $z = 2.000$ m and resolution 320×240.

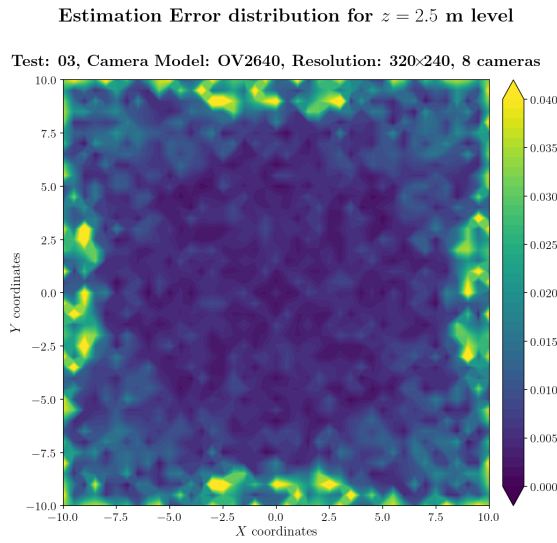


Figure A.371: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.500$ m.

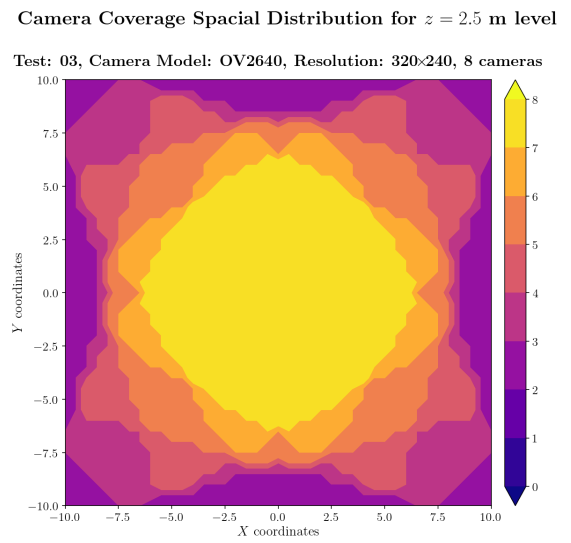
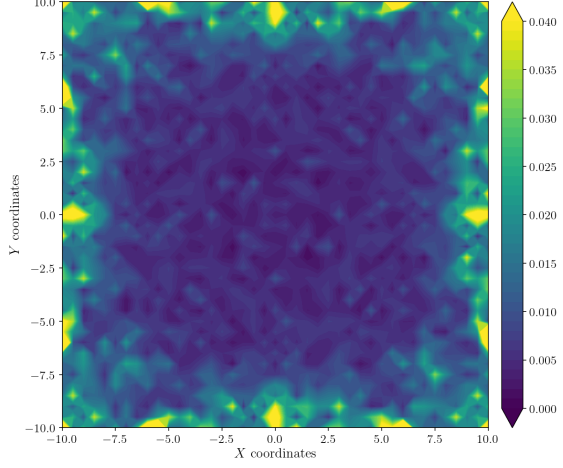


Figure A.372: Camera coverage for test 03 at level $z = 2.500$ m and resolution 320×240.

Estimation Error distribution for $z = 3.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras

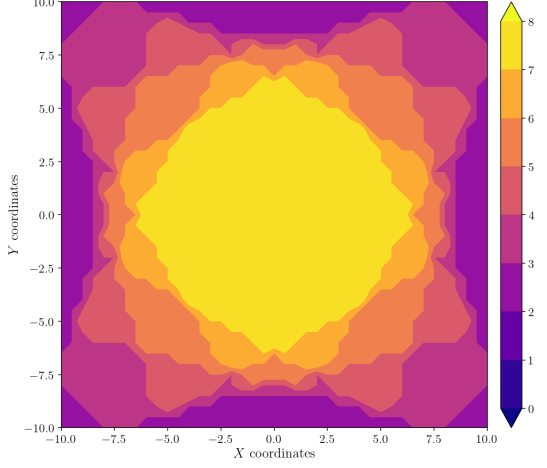
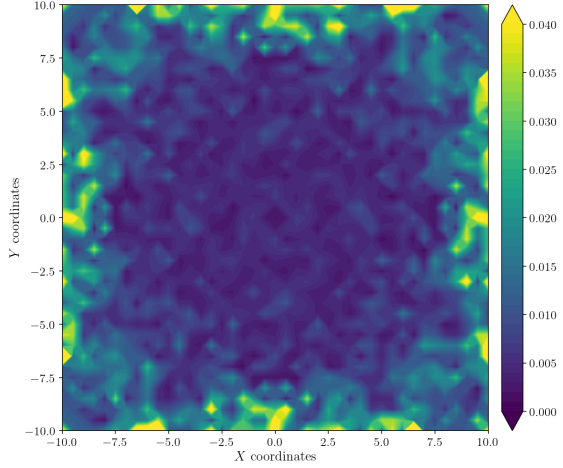


Figure A.373: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.000$ m.

Figure A.374: Camera coverage for test 03 at level $z = 3.000$ m and resolution 320×240.

Estimation Error distribution for $z = 3.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 320×240, 8 cameras

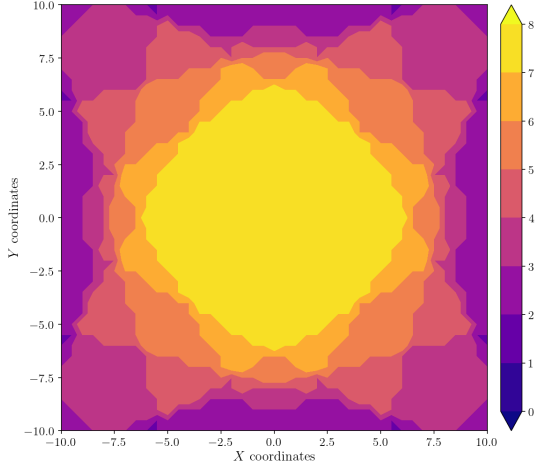


Figure A.375: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.500$ m.

Figure A.376: Camera coverage for test 03 at level $z = 3.500$ m and resolution 320×240.

Estimation Error distribution for $z = 4.0$ m level

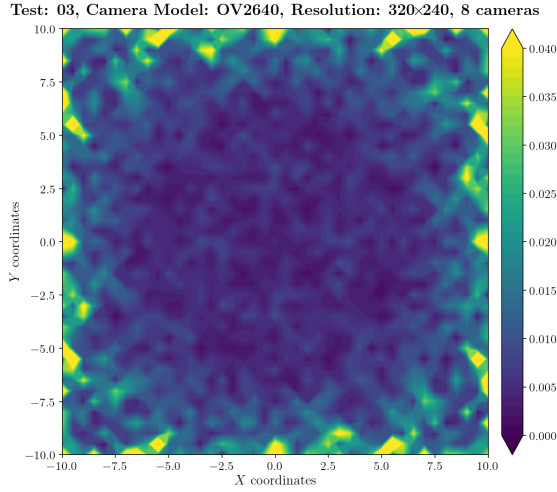


Figure A.377: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.000$ m.

Camera Coverage Spatial Distribution for $z = 4.0$ m level

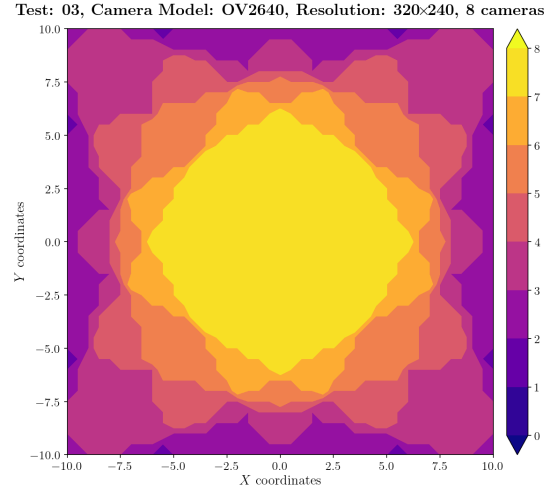


Figure A.378: Camera coverage for test 03 at level $z = 4.000$ m and resolution 320×240.

Estimation Error distribution for $z = 4.5$ m level

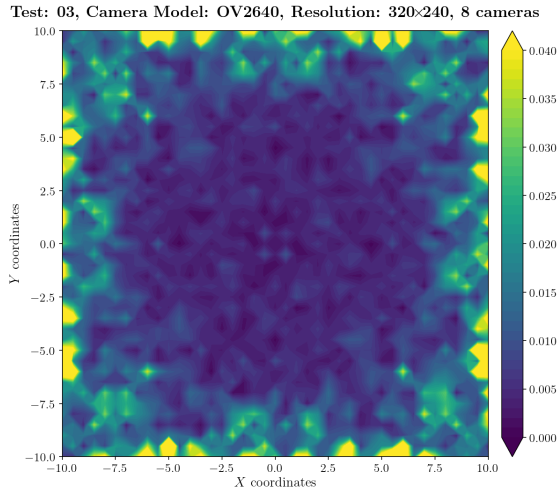


Figure A.379: Estimation error contour map for test 03 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.500$ m.

Camera Coverage Spatial Distribution for $z = 4.5$ m level

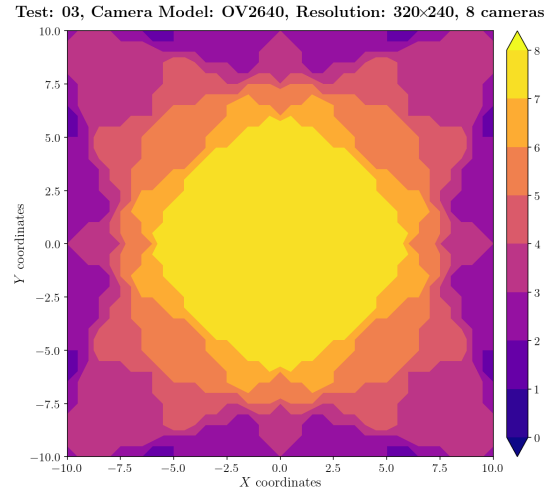


Figure A.380: Camera coverage for test 03 at level $z = 4.500$ m and resolution 320×240.

Error maps for resolution 640×480

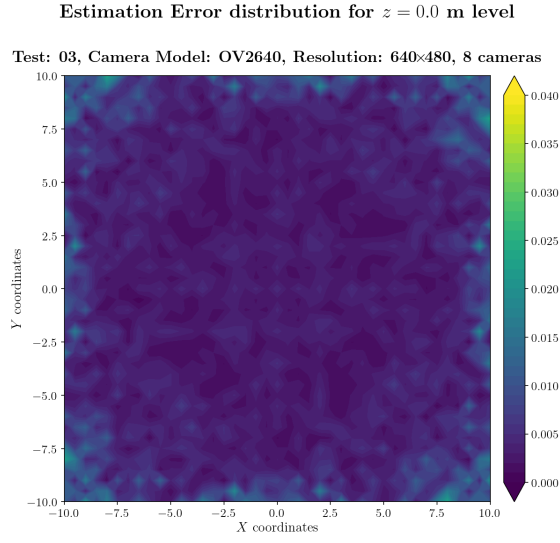


Figure A.381: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.000$ m.

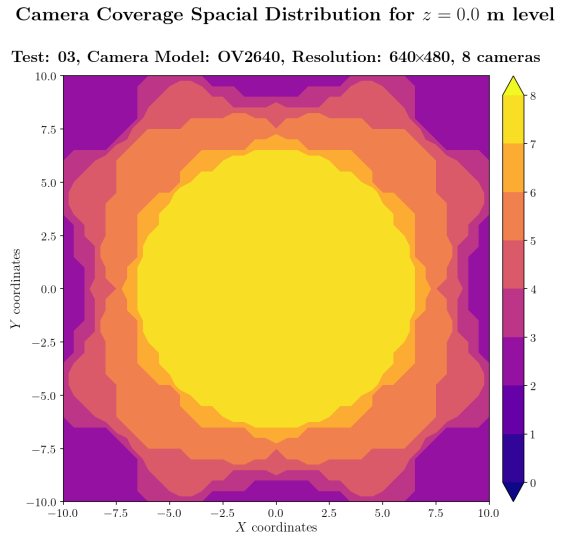


Figure A.382: Camera coverage for test 03 at level $z = 0.000$ m and resolution 640×480.

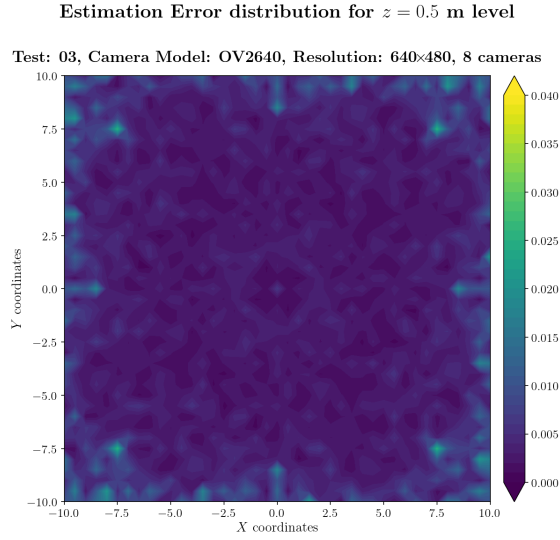


Figure A.383: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.500$ m.

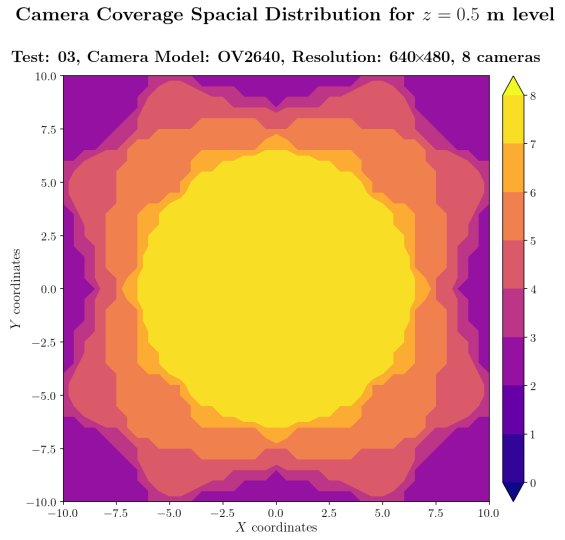


Figure A.384: Camera coverage for test 03 at level $z = 0.500$ m and resolution 640×480.

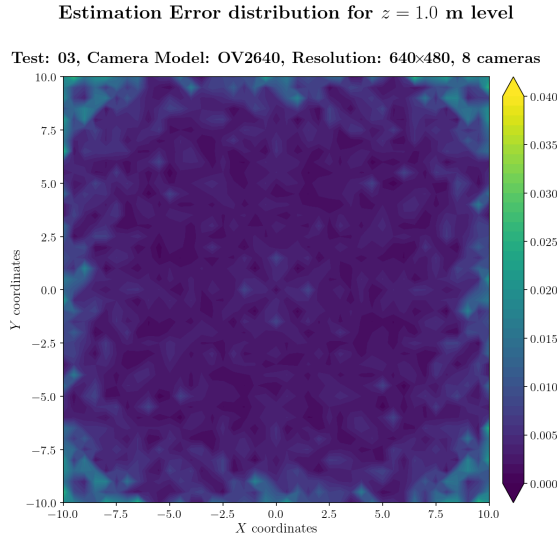


Figure A.385: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.000$ m.

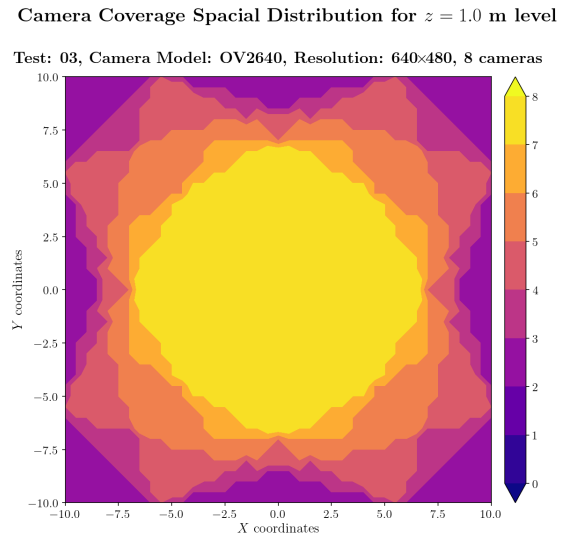


Figure A.386: Camera coverage for test 03 at level $z = 1.000$ m and resolution 640×480.

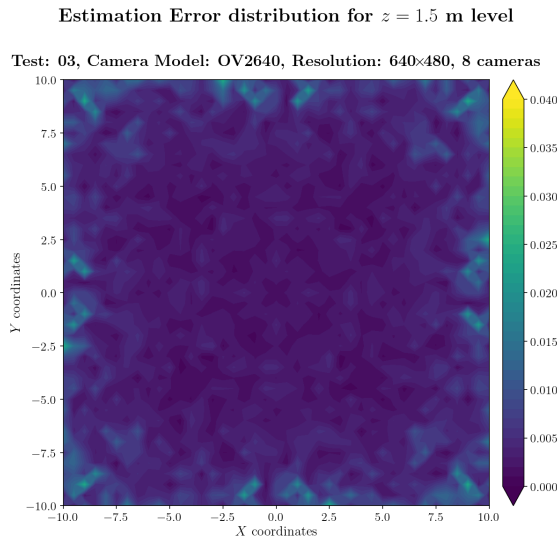


Figure A.387: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.500$ m.

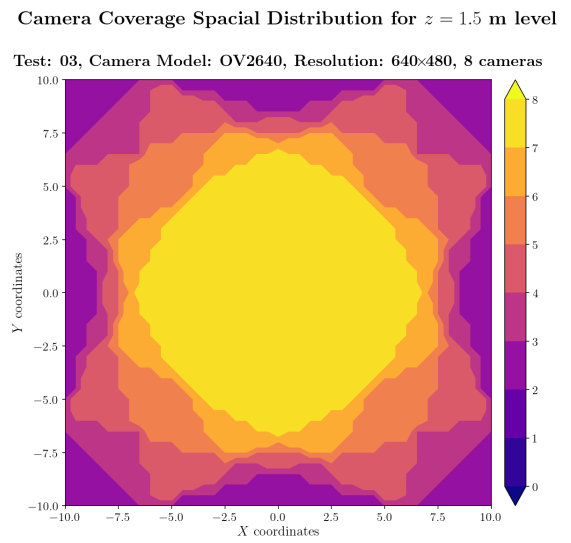
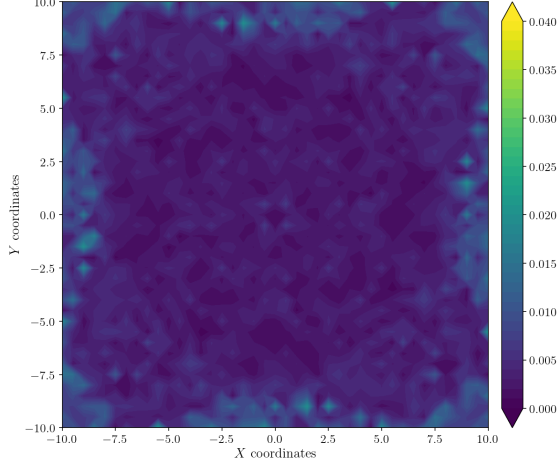


Figure A.388: Camera coverage for test 03 at level $z = 1.500$ m and resolution 640×480.

Estimation Error distribution for $z = 2.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 640×480, 8 cameras

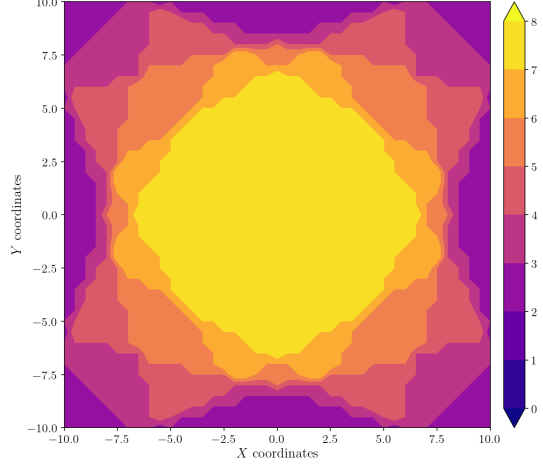
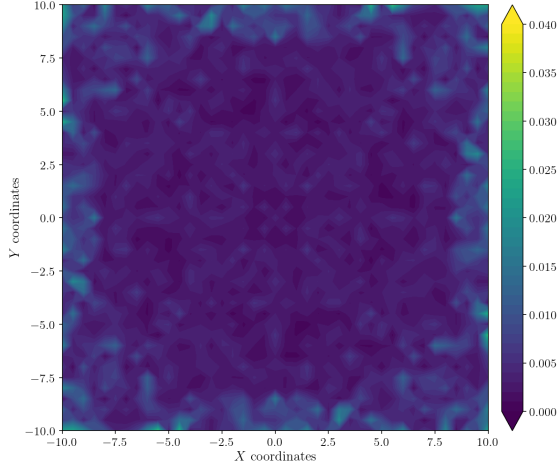


Figure A.389: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.000$ m.

Figure A.390: Camera coverage for test 03 at level $z = 2.000$ m and resolution 640×480.

Estimation Error distribution for $z = 2.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 640×480, 8 cameras

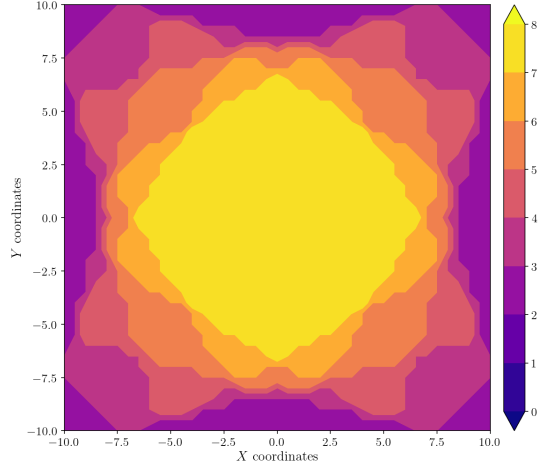


Figure A.391: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.500$ m.

Figure A.392: Camera coverage for test 03 at level $z = 2.500$ m and resolution 640×480.

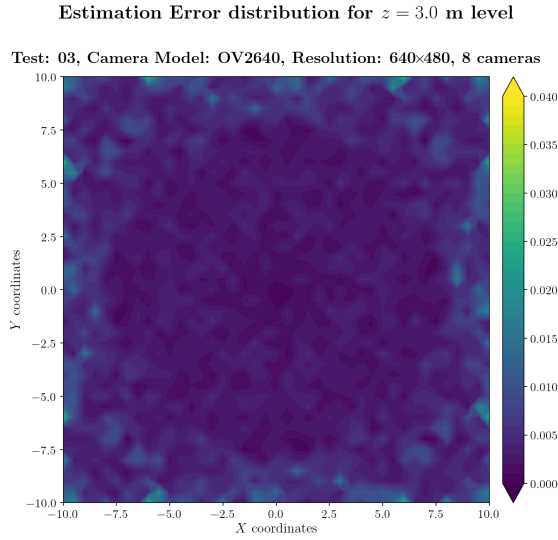


Figure A.393: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.000$ m.

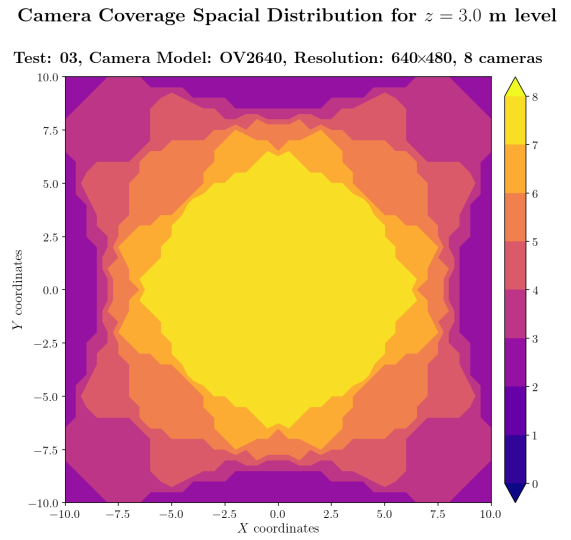


Figure A.394: Camera coverage for test 03 at level $z = 3.000$ m and resolution 640×480.

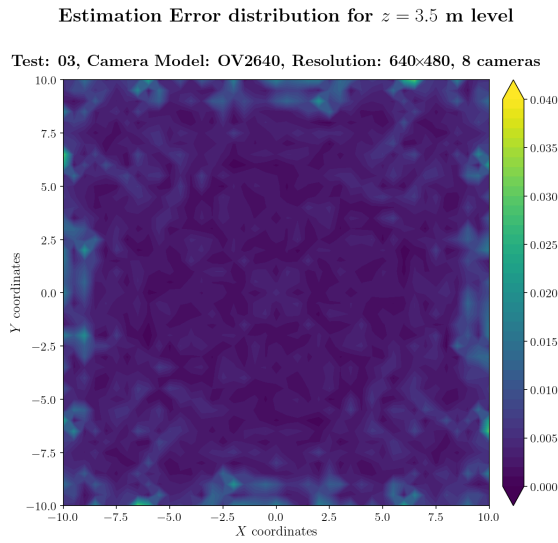


Figure A.395: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.500$ m.

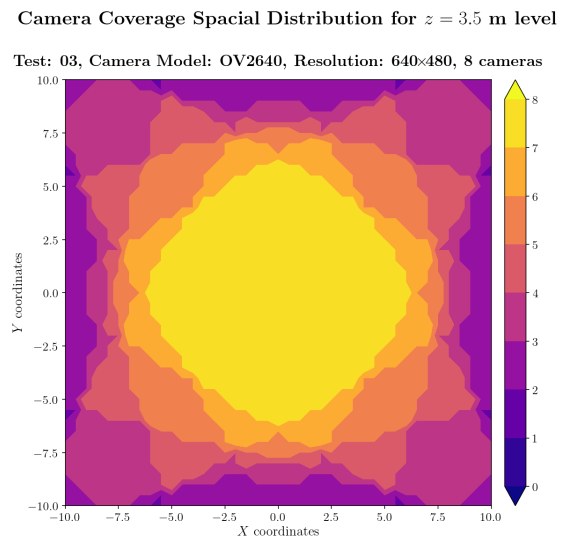
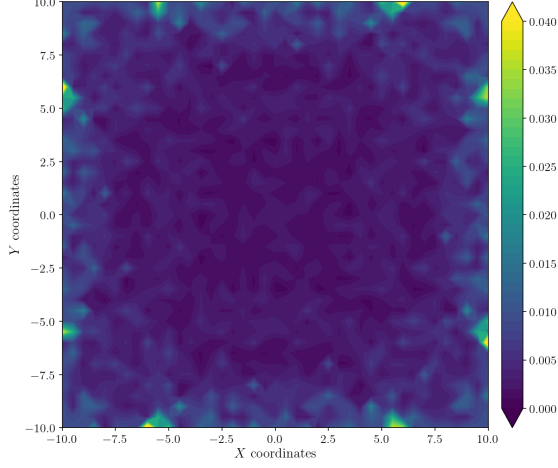


Figure A.396: Camera coverage for test 03 at level $z = 3.500$ m and resolution 640×480.

Estimation Error distribution for $z = 4.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 640×480, 8 cameras

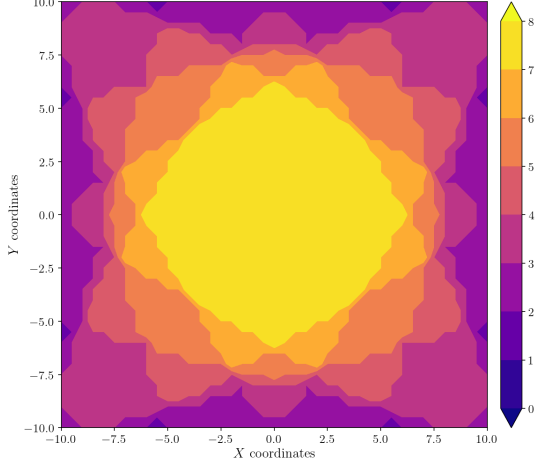
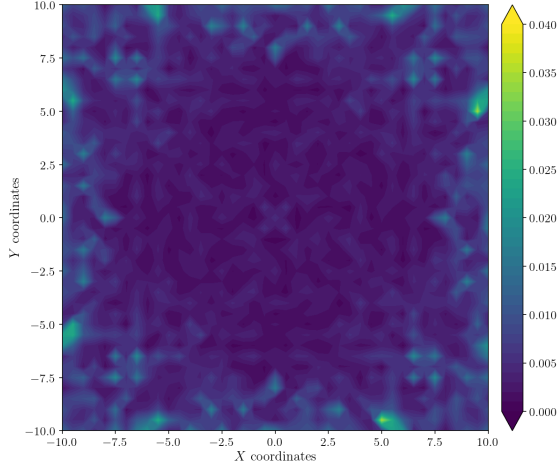


Figure A.397: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.000$ m.

Figure A.398: Camera coverage for test 03 at level $z = 4.000$ m and resolution 640×480.

Estimation Error distribution for $z = 4.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 640×480, 8 cameras

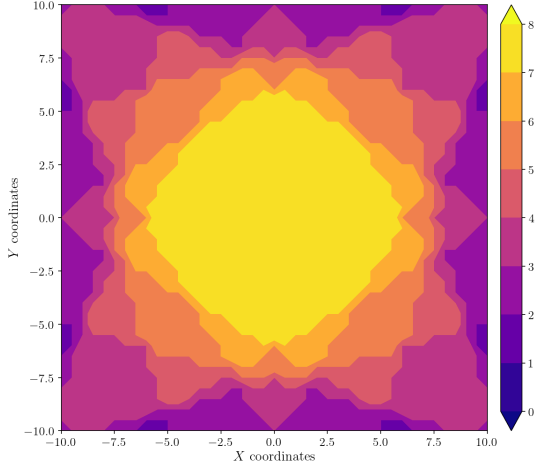


Figure A.399: Estimation error contour map for test 03 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.500$ m.

Figure A.400: Camera coverage for test 03 at level $z = 4.500$ m and resolution 640×480.

Error maps for resolution 800×600

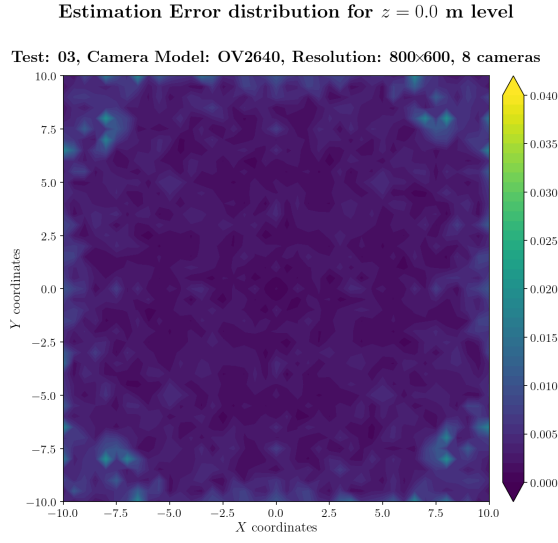


Figure A.401: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.000$ m.

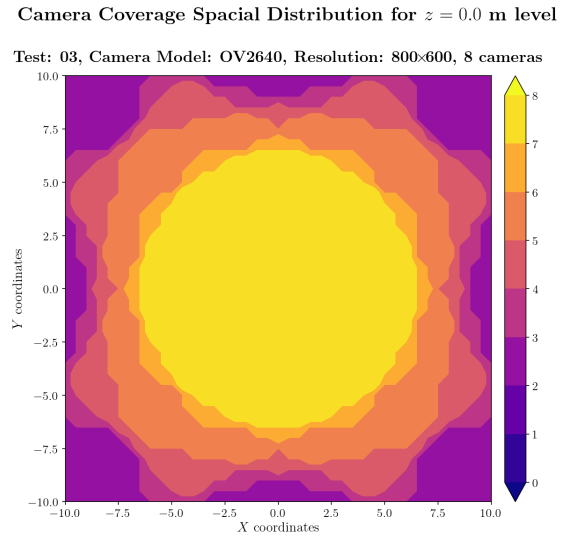


Figure A.402: Camera coverage for test 03 at level $z = 0.000$ m and resolution 800×600.

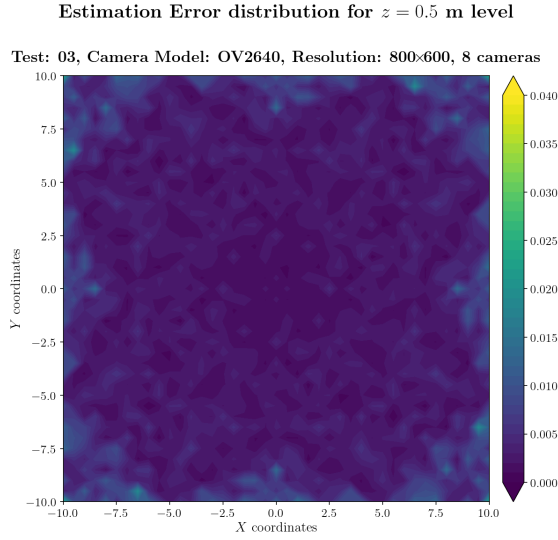


Figure A.403: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.500$ m.

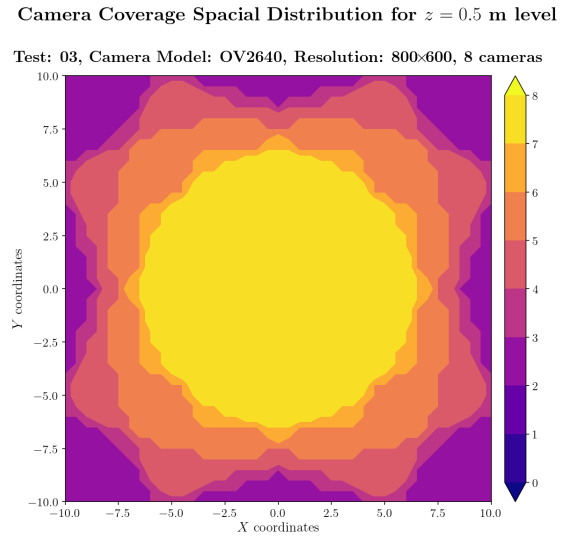
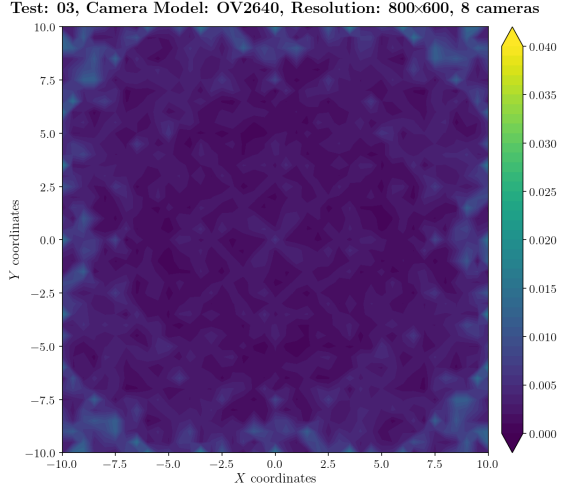


Figure A.404: Camera coverage for test 03 at level $z = 0.500$ m and resolution 800×600.

Estimation Error distribution for $z = 1.0$ m level



Camera Coverage Spacial Distribution for $z = 1.0$ m level

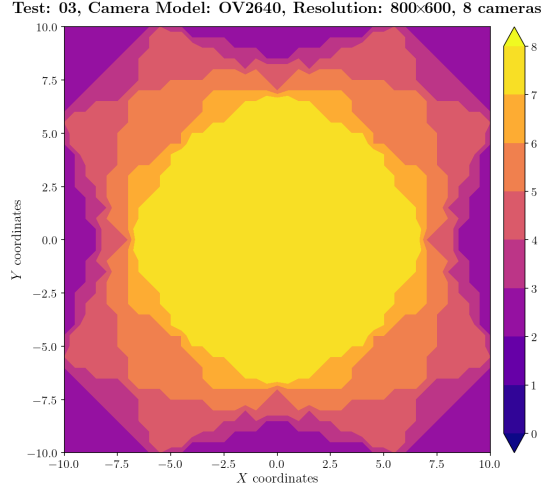
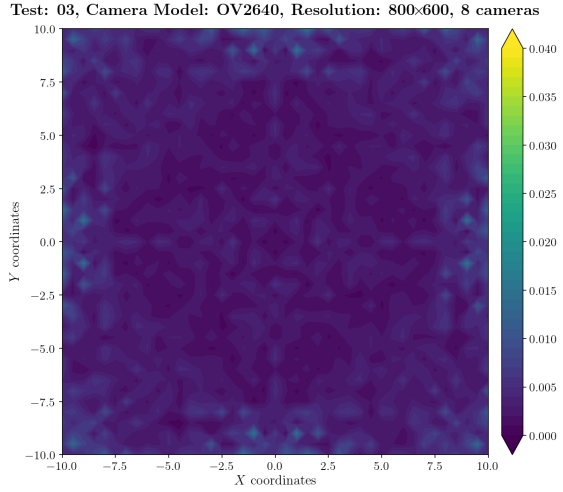


Figure A.405: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.000$ m.

Figure A.406: Camera coverage for test 03 at level $z = 1.000$ m and resolution 800×600.

Estimation Error distribution for $z = 1.5$ m level



Camera Coverage Spacial Distribution for $z = 1.5$ m level

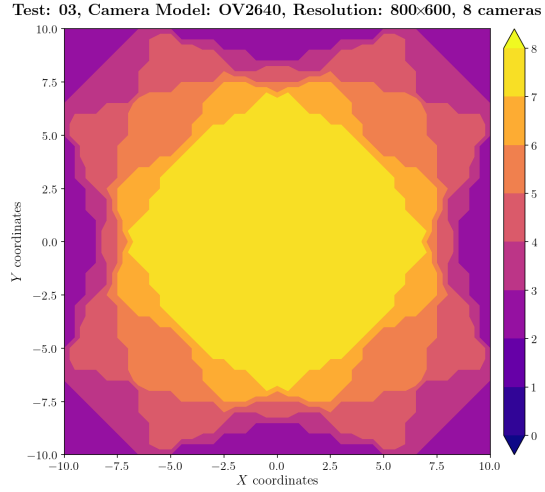


Figure A.407: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.500$ m.

Figure A.408: Camera coverage for test 03 at level $z = 1.500$ m and resolution 800×600.

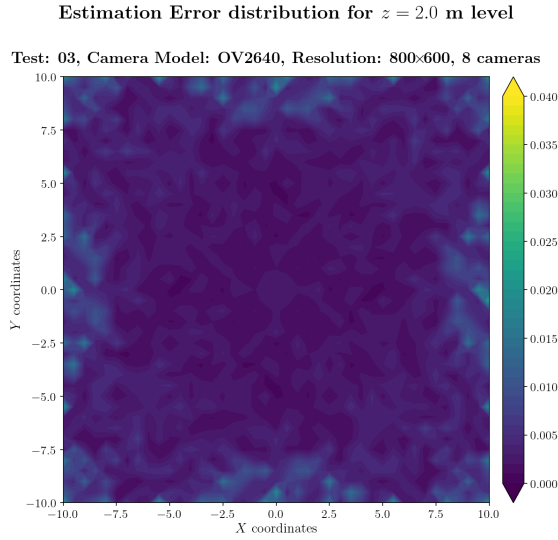


Figure A.409: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.000$ m.

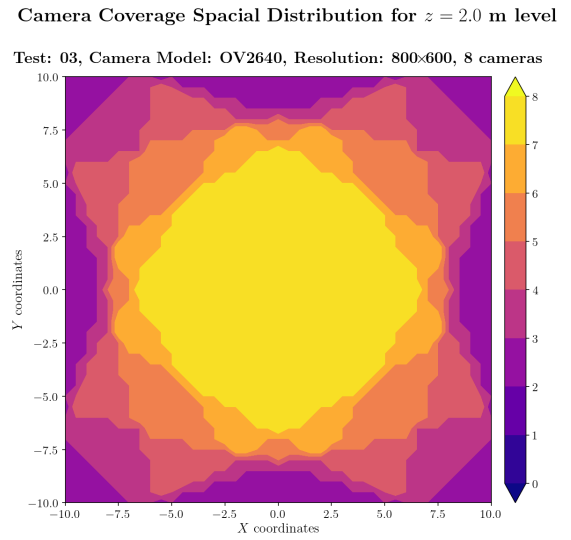


Figure A.410: Camera coverage for test 03 at level $z = 2.000$ m and resolution 800×600.

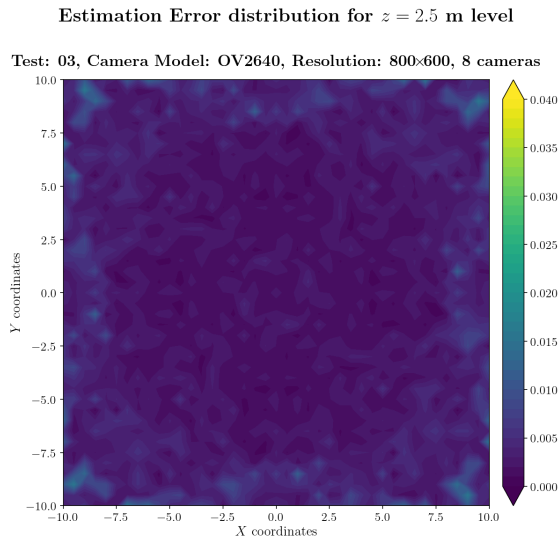


Figure A.411: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.500$ m.

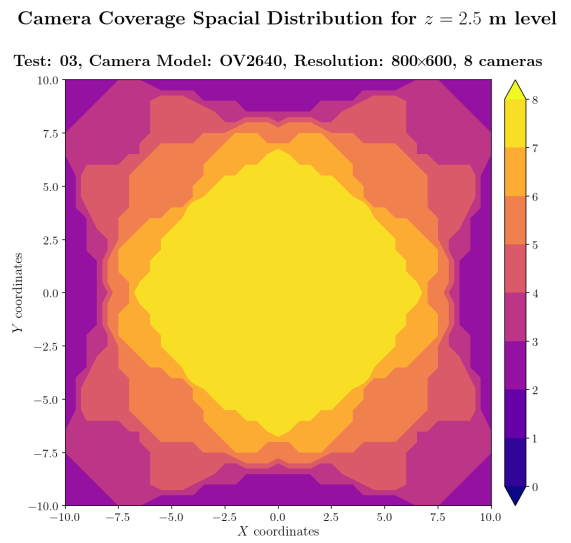
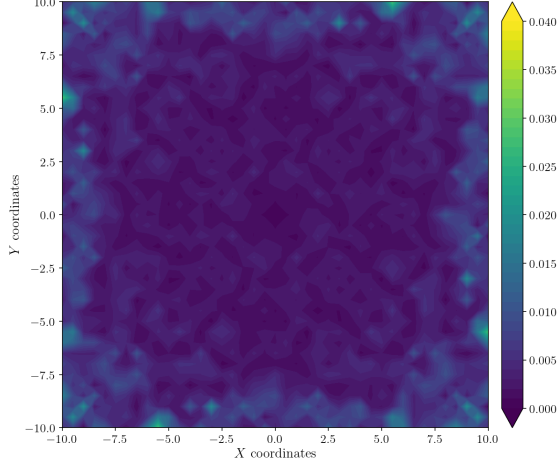


Figure A.412: Camera coverage for test 03 at level $z = 2.500$ m and resolution 800×600.

Estimation Error distribution for $z = 3.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 800×600, 8 cameras

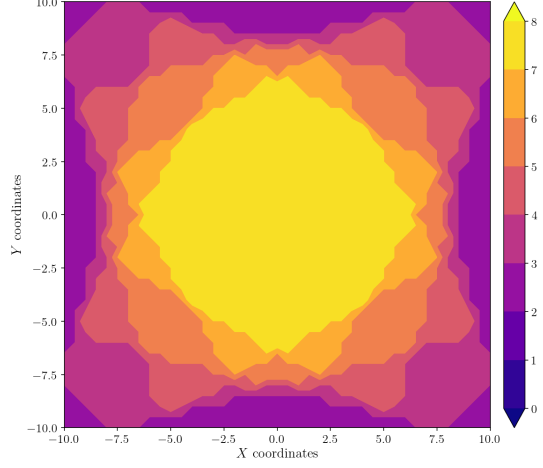
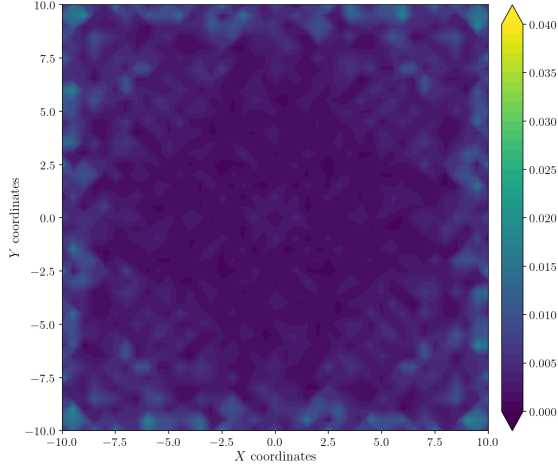


Figure A.413: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.000$ m.

Figure A.414: Camera coverage for test 03 at level $z = 3.000$ m and resolution 800×600.

Estimation Error distribution for $z = 3.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 800×600, 8 cameras

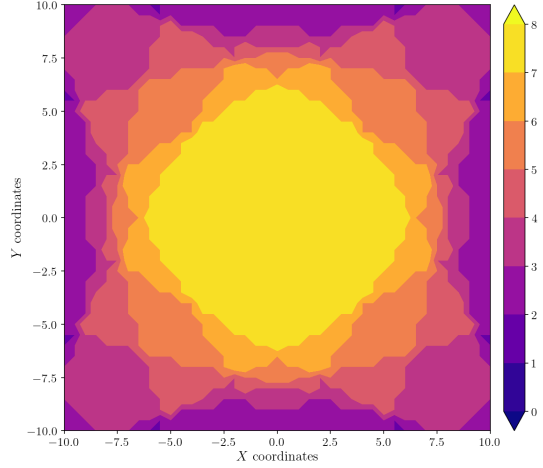


Figure A.415: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.500$ m.

Figure A.416: Camera coverage for test 03 at level $z = 3.500$ m and resolution 800×600.

Estimation Error distribution for $z = 4.0$ m level

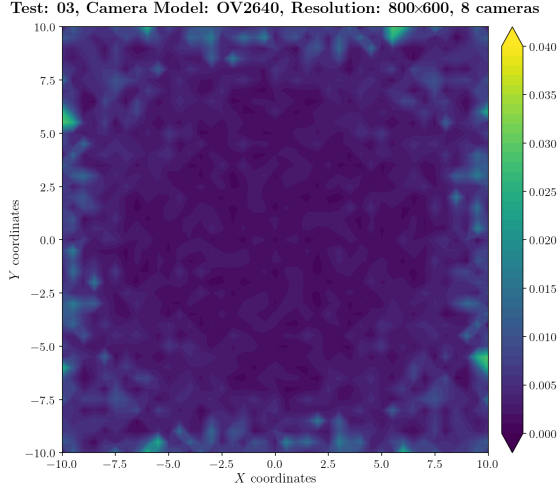


Figure A.417: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.000$ m.

Camera Coverage Spatial Distribution for $z = 4.0$ m level

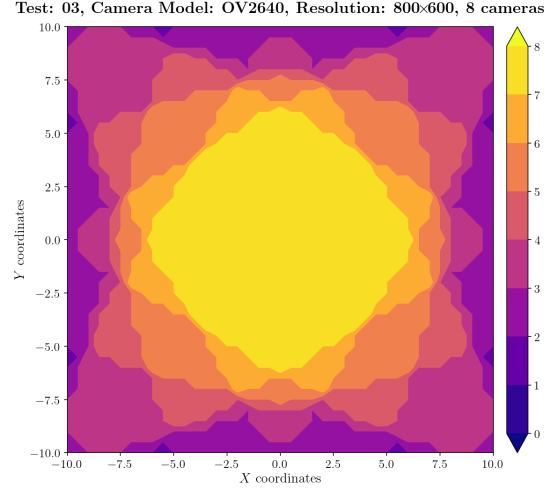


Figure A.418: Camera coverage for test 03 at level $z = 4.000$ m and resolution 800×600.

Estimation Error distribution for $z = 4.5$ m level

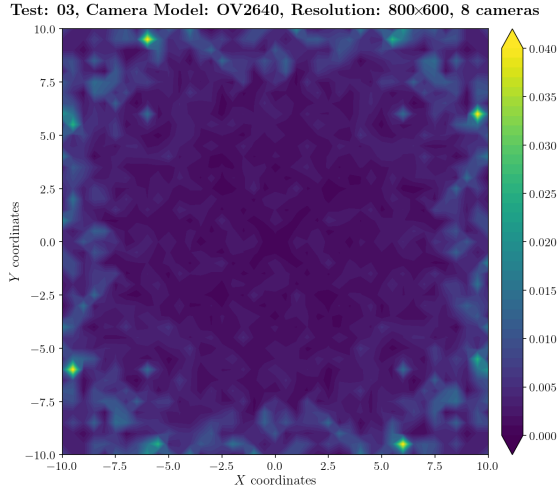


Figure A.419: Estimation error contour map for test 03 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.500$ m.

Camera Coverage Spatial Distribution for $z = 4.5$ m level

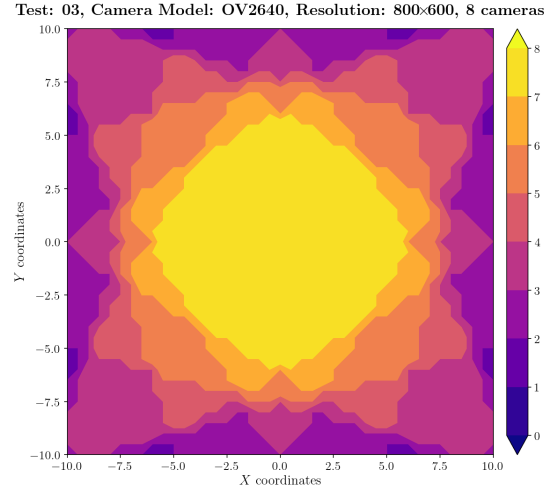


Figure A.420: Camera coverage for test 03 at level $z = 4.500$ m and resolution 800×600.

Error maps for resolution 1024×768

Estimation Error distribution for $z = 0.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

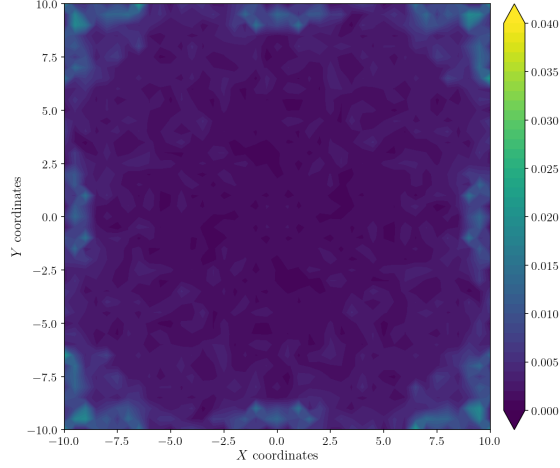


Figure A.421: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

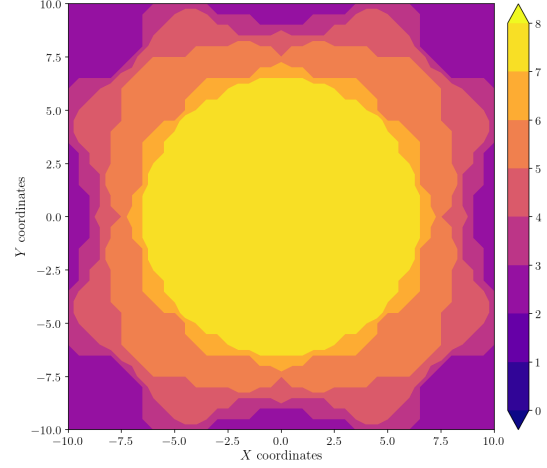


Figure A.422: Camera coverage for test 03 at level $z = 0.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 0.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

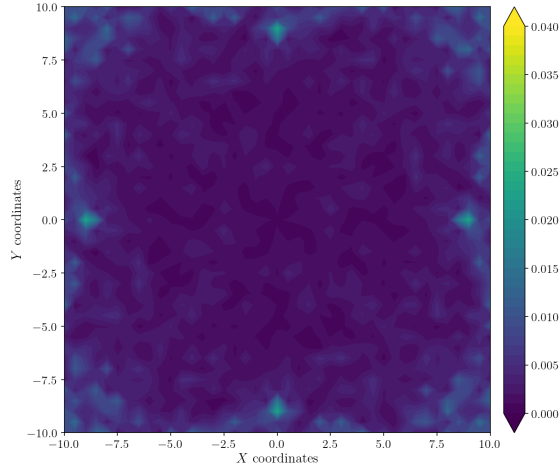


Figure A.423: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.500$ m.

Camera Coverage Spacial Distribution for $z = 0.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

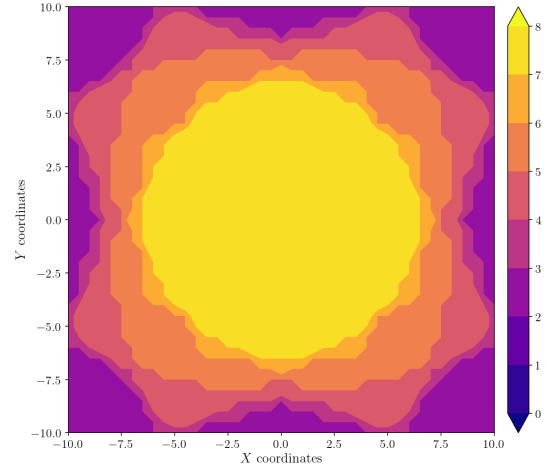


Figure A.424: Camera coverage for test 03 at level $z = 0.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.0$ m level

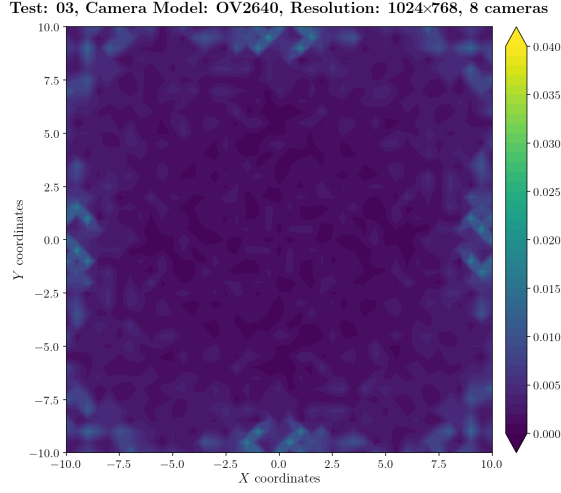


Figure A.425: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.000$ m.

Camera Coverage Spacial Distribution for $z = 1.0$ m level

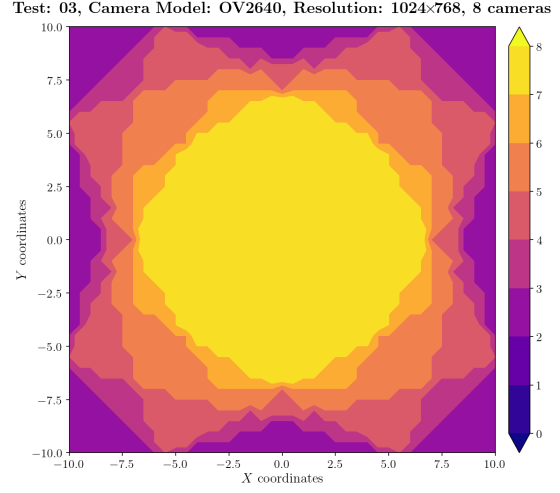


Figure A.426: Camera coverage for test 03 at level $z = 1.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.5$ m level

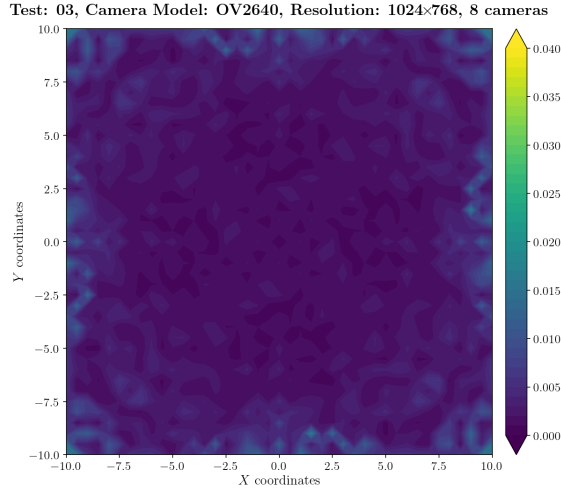


Figure A.427: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.500$ m.

Camera Coverage Spacial Distribution for $z = 1.5$ m level

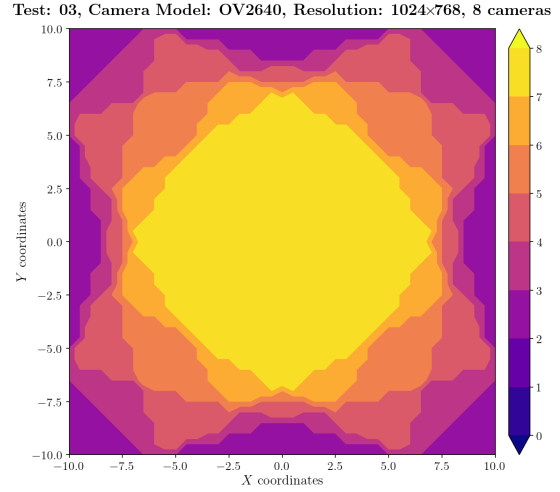
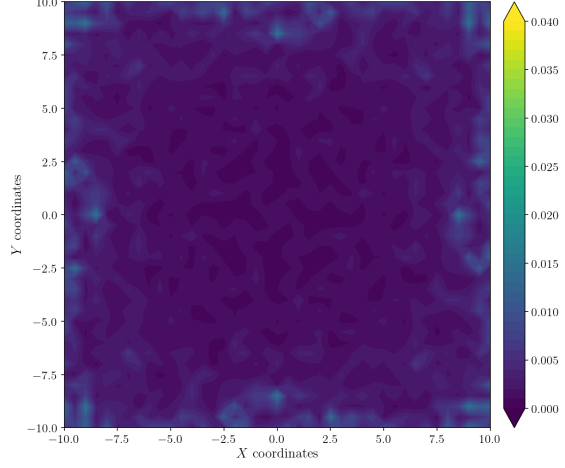


Figure A.428: Camera coverage for test 03 at level $z = 1.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 2.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

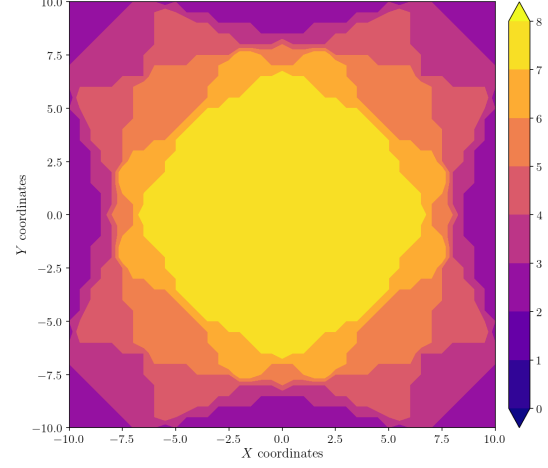
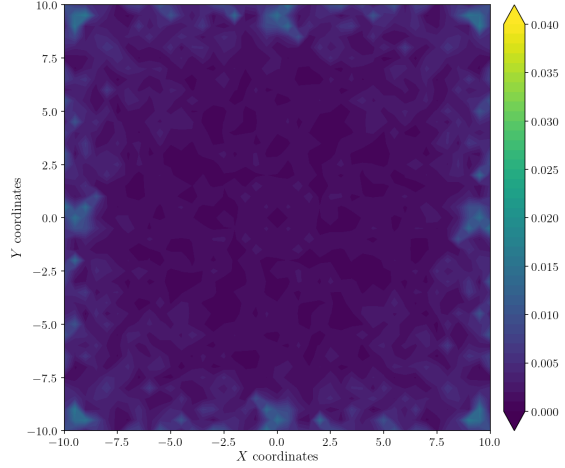


Figure A.429: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.000$ m.

Figure A.430: Camera coverage for test 03 at level $z = 2.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 2.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

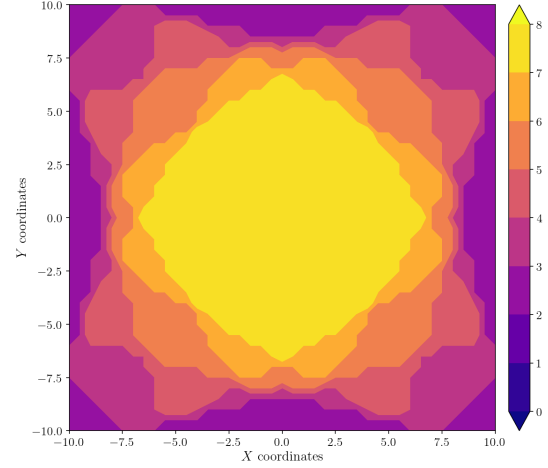


Figure A.431: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.500$ m.

Figure A.432: Camera coverage for test 03 at level $z = 2.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.0$ m level

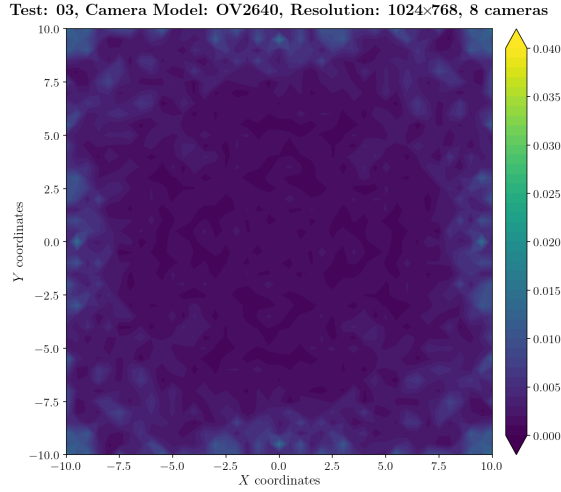


Figure A.433: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.000$ m.

Camera Coverage Spacial Distribution for $z = 3.0$ m level

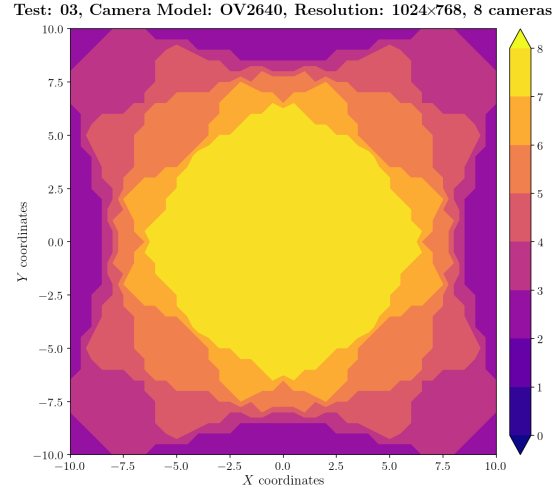


Figure A.434: Camera coverage for test 03 at level $z = 3.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.5$ m level

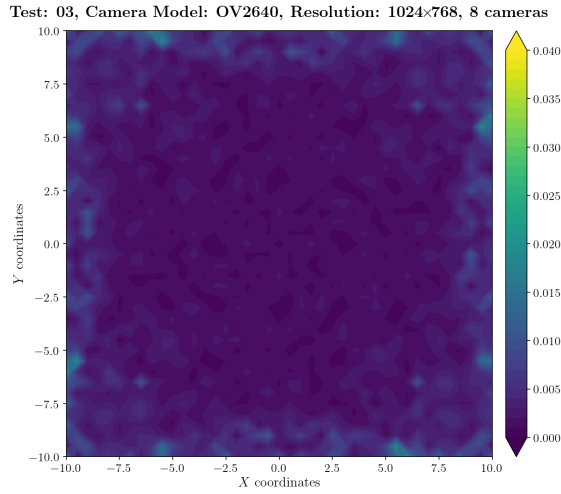


Figure A.435: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.500$ m.

Camera Coverage Spacial Distribution for $z = 3.5$ m level

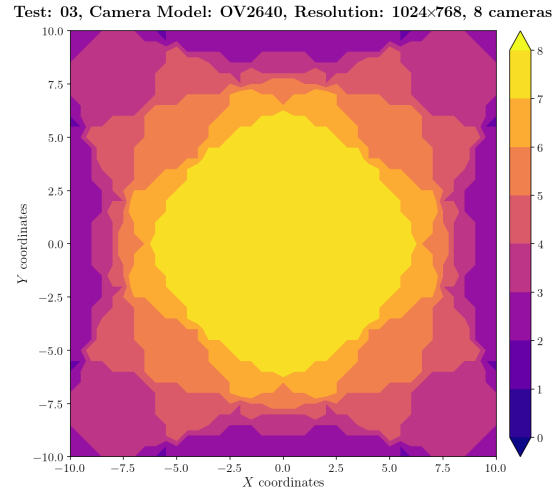
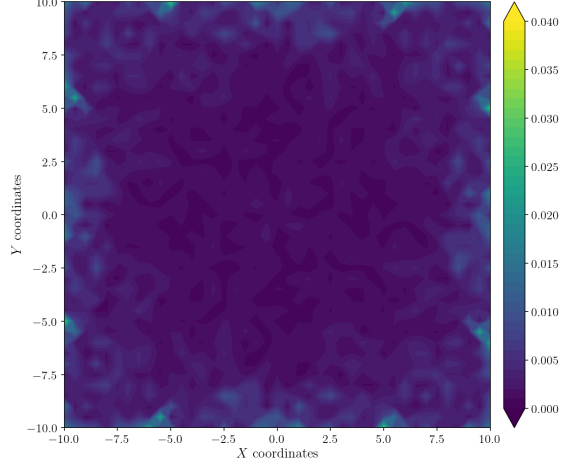


Figure A.436: Camera coverage for test 03 at level $z = 3.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 4.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

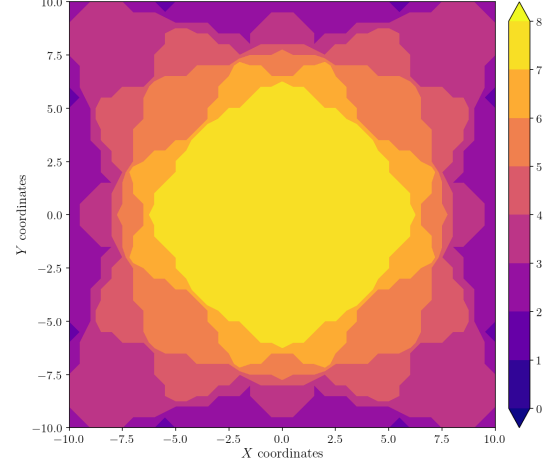
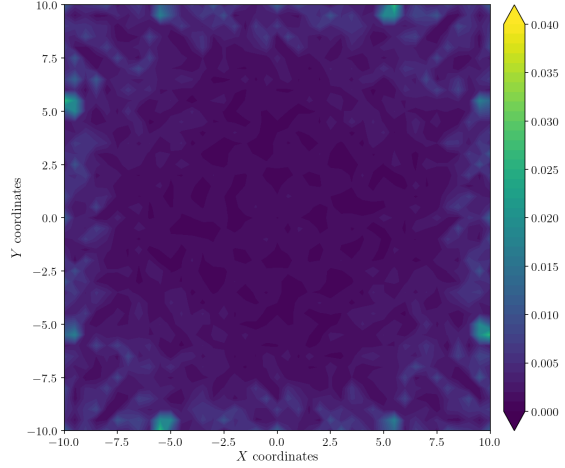


Figure A.437: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.000$ m.

Figure A.438: Camera coverage for test 03 at level $z = 4.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 4.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

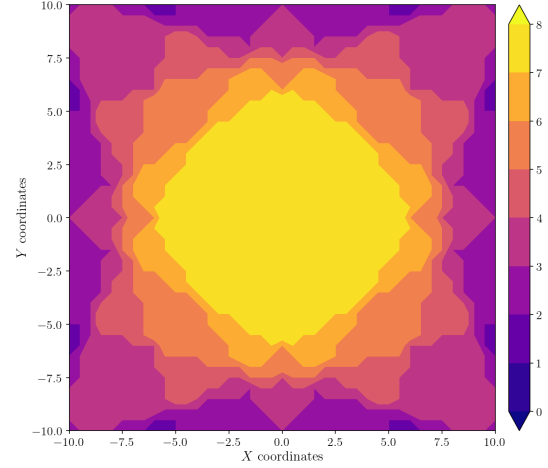


Figure A.439: Estimation error contour map for test 03 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.500$ m.

Figure A.440: Camera coverage for test 03 at level $z = 4.500$ m and resolution 1024×768.

Error maps for resolution 1600×1200

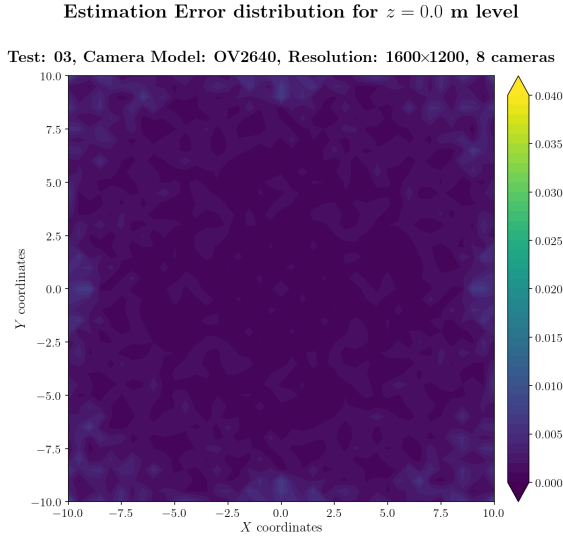


Figure A.441: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.000$ m.

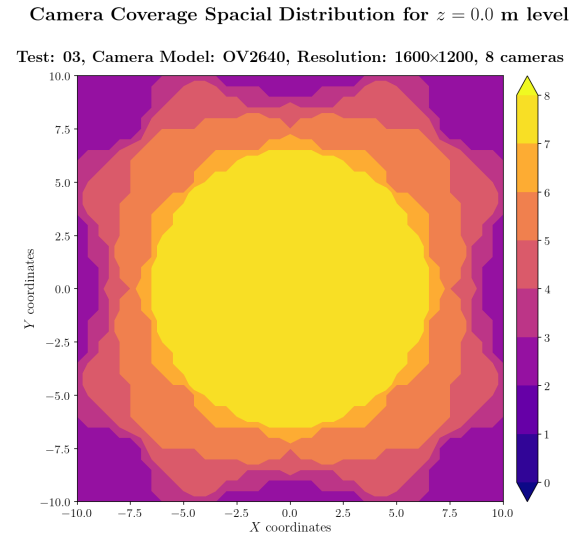


Figure A.442: Camera coverage for test 03 at level $z = 0.000$ m and resolution 1600×1200.

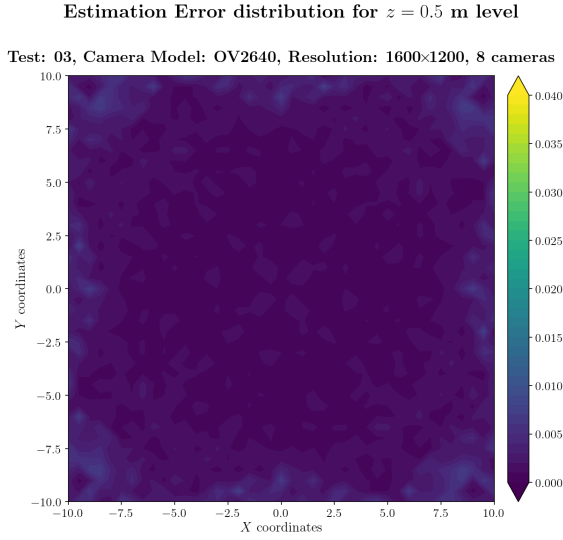


Figure A.443: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.500$ m.

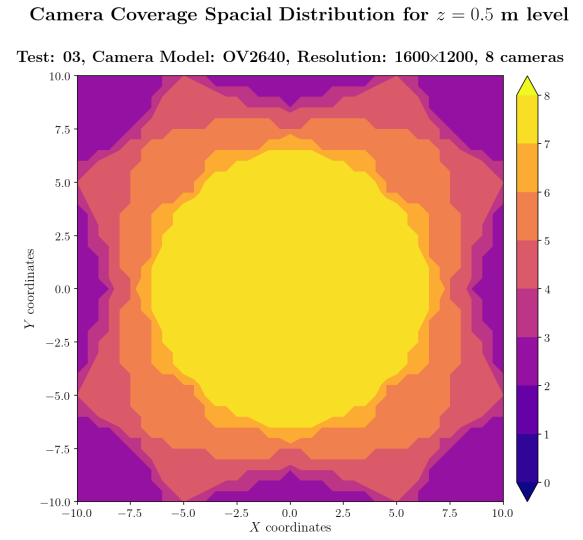
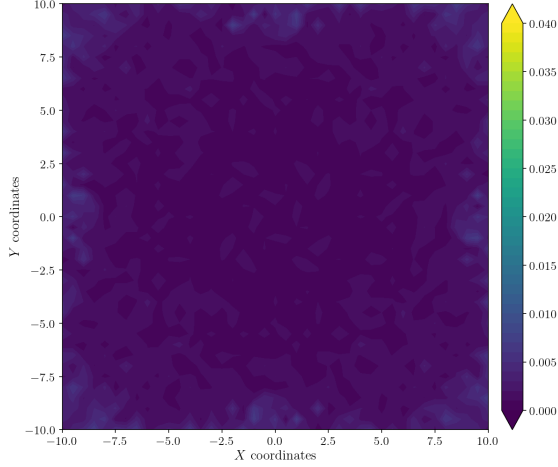


Figure A.444: Camera coverage for test 03 at level $z = 0.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 1.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

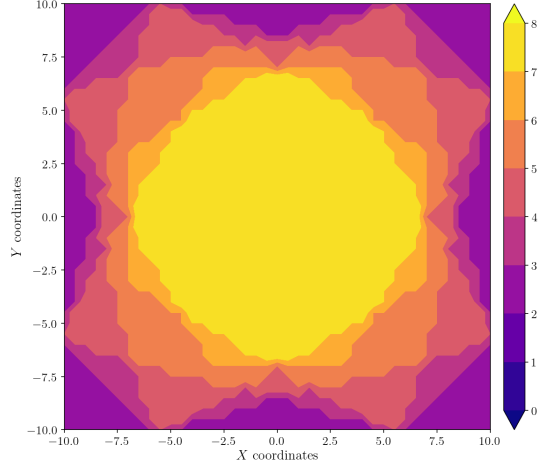
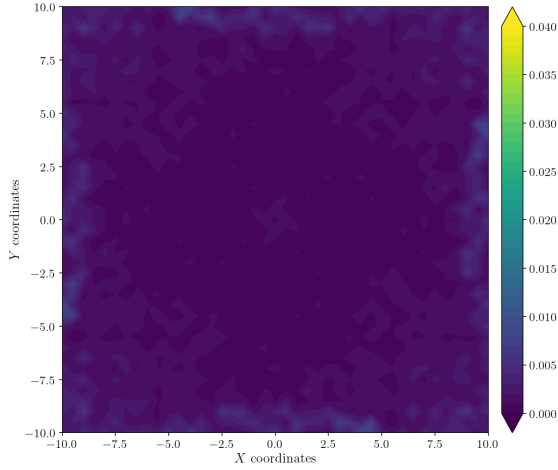


Figure A.445: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.000$ m.

Figure A.446: Camera coverage for test 03 at level $z = 1.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 1.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

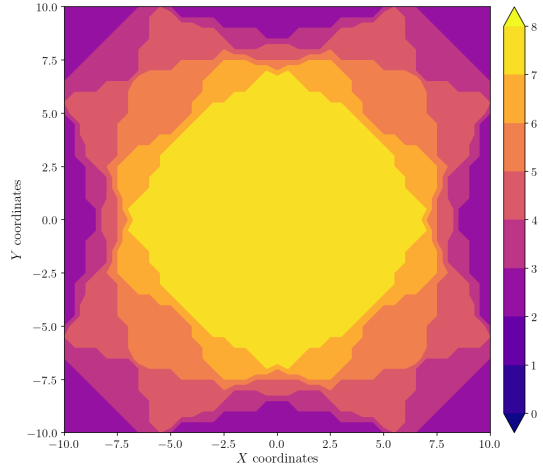


Figure A.447: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.500$ m.

Figure A.448: Camera coverage for test 03 at level $z = 1.500$ m and resolution 1600×1200.

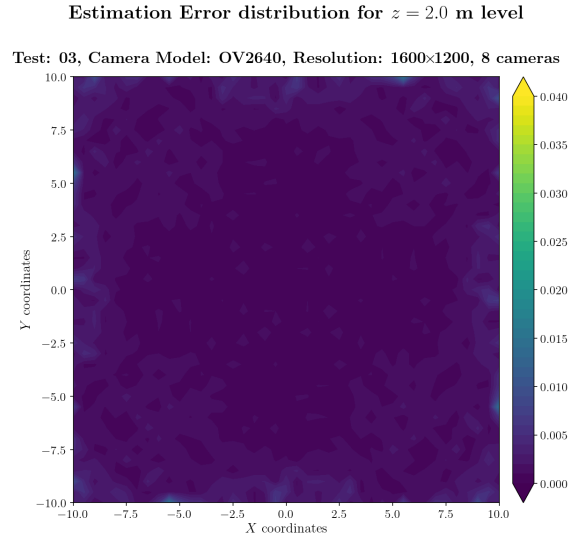


Figure A.449: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.000$ m.

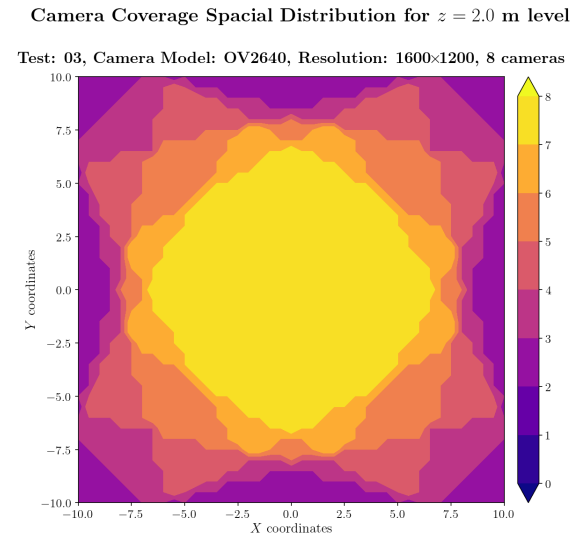


Figure A.450: Camera coverage for test 03 at level $z = 2.000$ m and resolution 1600×1200.

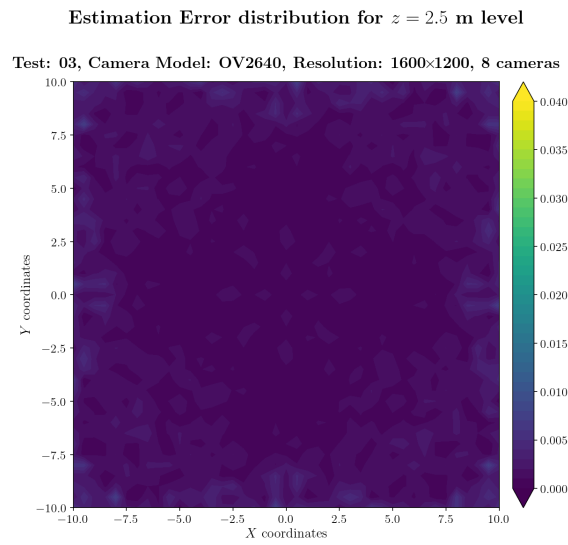


Figure A.451: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.500$ m.

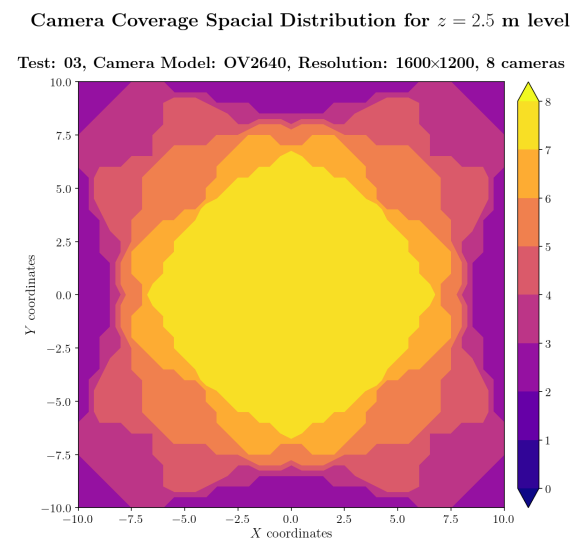
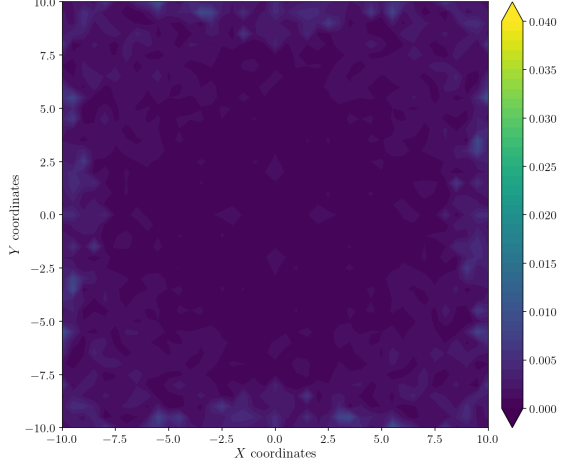


Figure A.452: Camera coverage for test 03 at level $z = 2.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 3.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

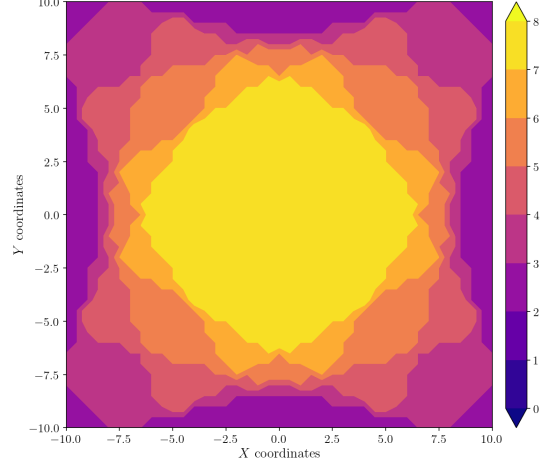
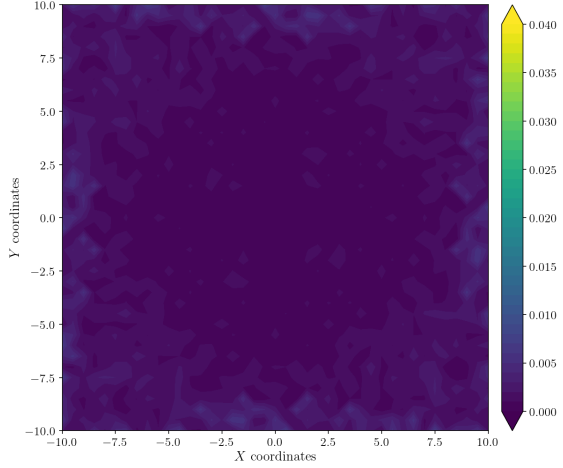


Figure A.453: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.000$ m.

Figure A.454: Camera coverage for test 03 at level $z = 3.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 3.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.5$ m level

Test: 03, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

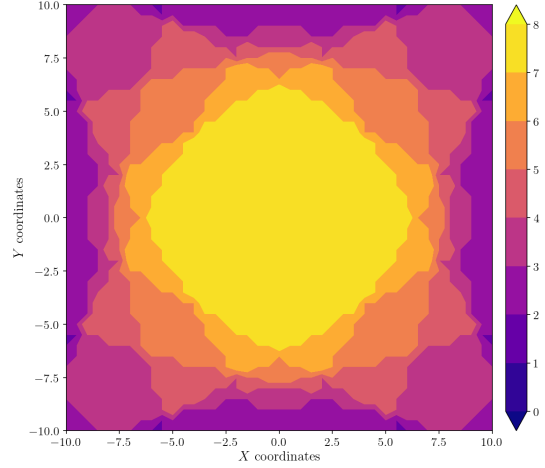


Figure A.455: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.500$ m.

Figure A.456: Camera coverage for test 03 at level $z = 3.500$ m and resolution 1600×1200.

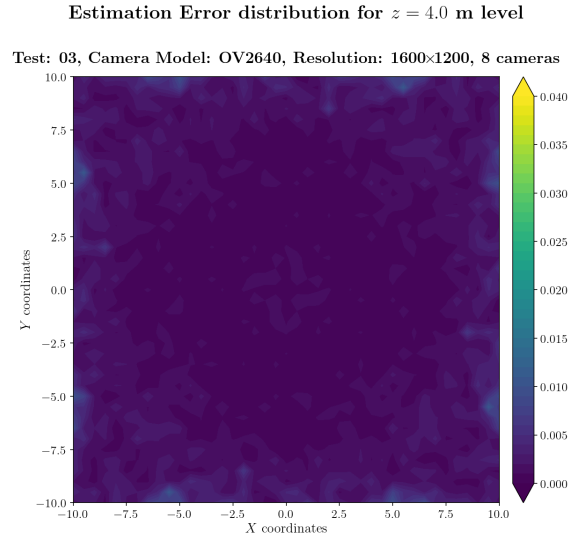


Figure A.457: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.000$ m.

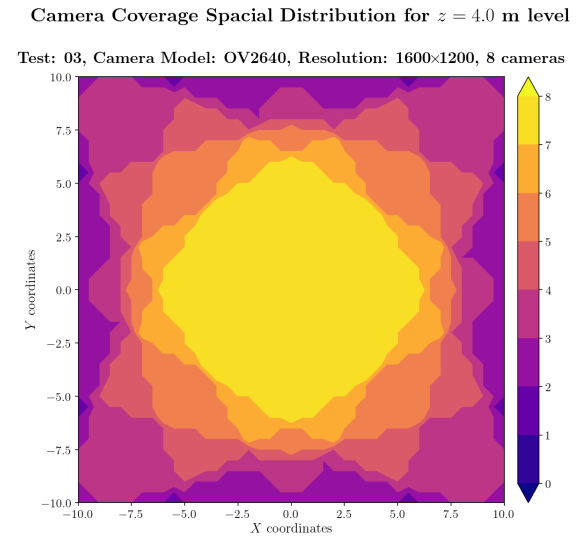


Figure A.458: Camera coverage for test 03 at level $z = 4.000$ m and resolution 1600×1200.

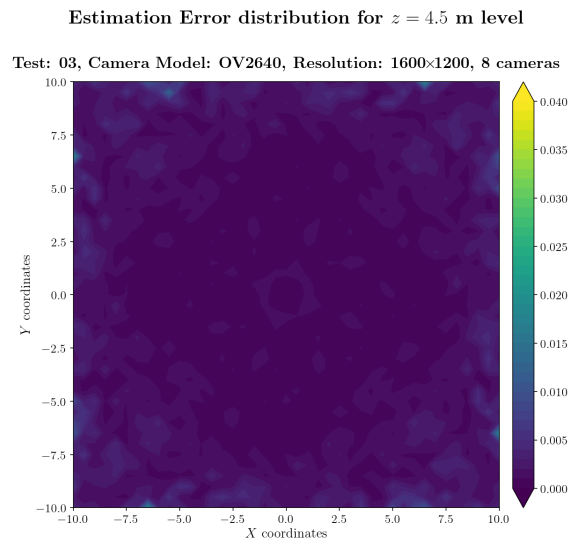


Figure A.459: Estimation error contour map for test 03 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.500$ m.

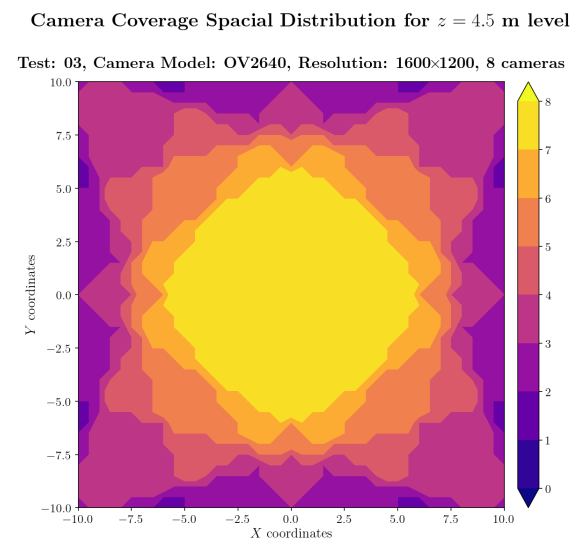
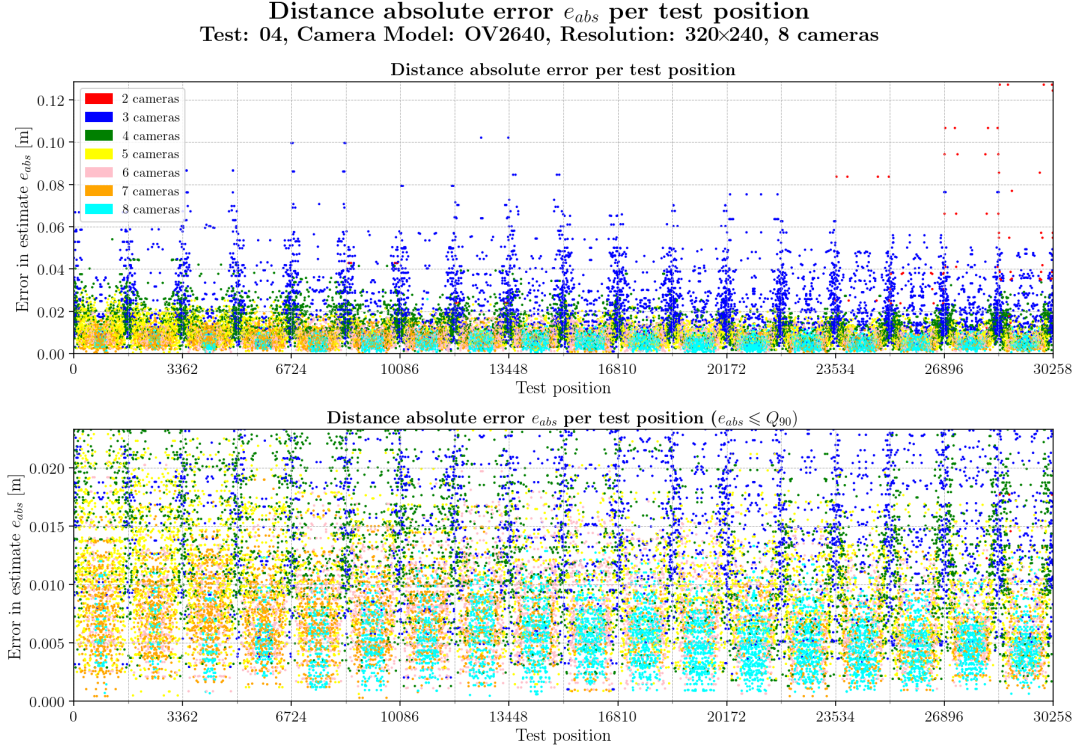


Figure A.460: Camera coverage for test 03 at level $z = 4.500$ m and resolution 1600×1200.

A.4 Test 04

Absolute Error per iteration and resolution

Resolution: 320×240



Error per iteration for resolution 320×240 and test04

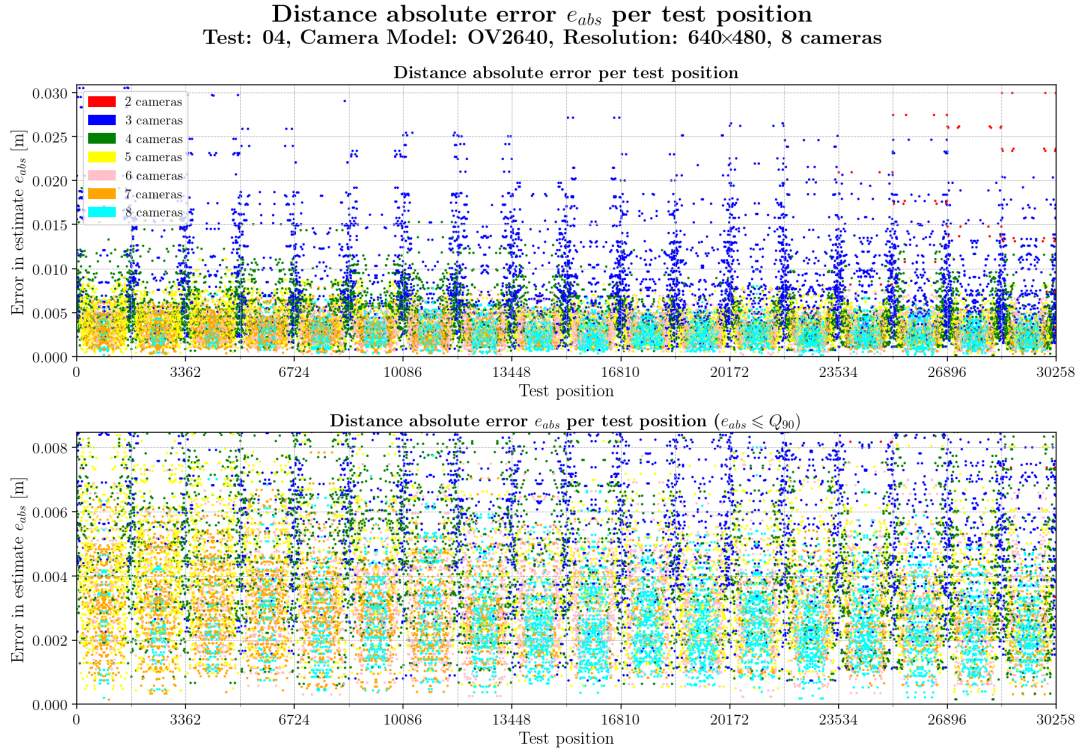
Table A.31: Statistics per number of cameras observing the beacon for test 04 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	80	5728	4593	4859	4772	4096	6130
mean	0.0530	0.0244	0.0141	0.0103	0.0078	0.0069	0.0053
std	0.0313	0.0141	0.0068	0.0046	0.0035	0.0031	0.0023
min	0.0122	0.0010	0.0013	0.0005	0.0006	0.0003	0.0005
25%	0.0305	0.0139	0.0093	0.0068	0.0052	0.0046	0.0035
50%	0.0386	0.0209	0.0130	0.0098	0.0075	0.0067	0.0051
75%	0.0688	0.0322	0.0180	0.0131	0.0100	0.0088	0.0067
max	0.1272	0.1021	0.0541	0.0301	0.0245	0.0190	0.0257

Table A.32: Global statistics and root mean square error for test 04 with resolution 320×240.

	Abs. error	Est. error
count	30258	30258
mean	0.0118	0.0280
std	0.0103	0.0095
min	0.0003	0.0001
25%	0.0056	0.0214
50%	0.0088	0.0275
75%	0.0142	0.0341
90%	0.0233	0.0402
max	0.1272	0.0701
RMSE	0.0156	

Resolution: 640×480



Error per iteration for resolution 640×480 and test04

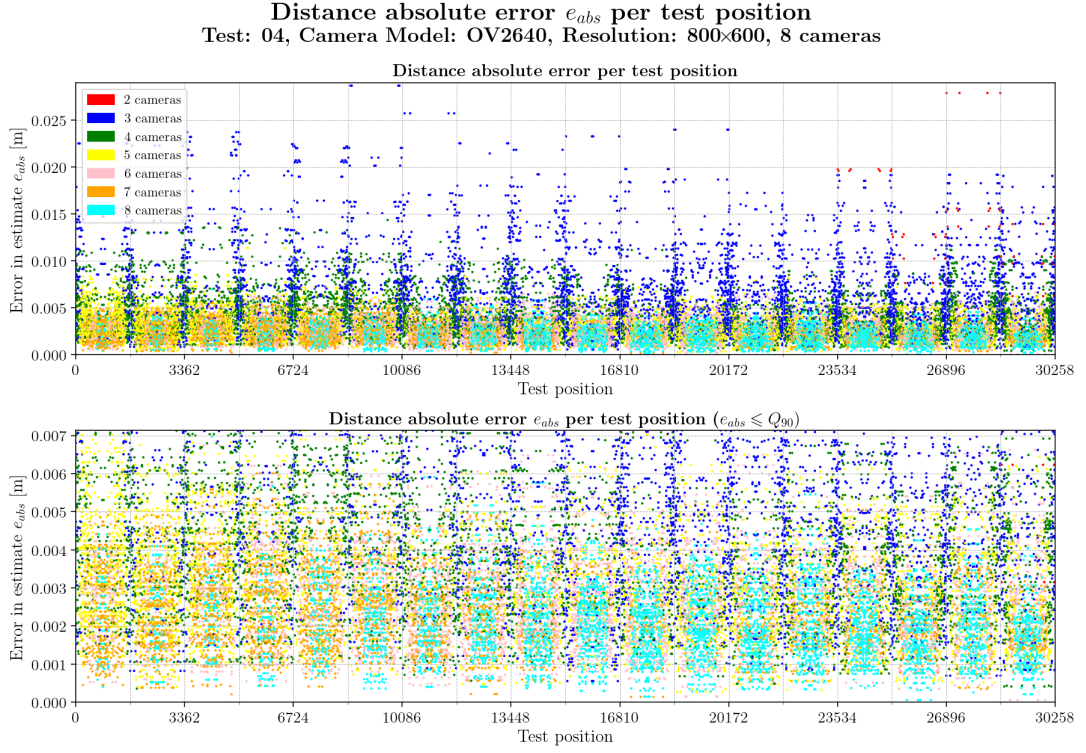
Table A.33: Statistics per number of cameras observing the beacon for test 04 with resolution 640×480.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	72	5616	4580	4800	4844	4172	6174
mean	0.0166	0.0089	0.0051	0.0039	0.0031	0.0029	0.0026
std	0.0082	0.0051	0.0026	0.0017	0.0014	0.0013	0.0011
min	0.0024	0.0007	0.0002	0.0004	0.0002	0.0001	0.0002
25%	0.0108	0.0051	0.0032	0.0026	0.0021	0.0020	0.0018
50%	0.0161	0.0076	0.0046	0.0037	0.0030	0.0028	0.0024
75%	0.0236	0.0119	0.0065	0.0050	0.0040	0.0037	0.0033
max	0.0299	0.0305	0.0191	0.0113	0.0077	0.0078	0.0080

Table A.34: Global statistics and root mean square error for test 04 with resolution 640×480.

	Abs. error	Est. error
count	30258	30258
mean	0.0045	0.0108
std	0.0036	0.0035
min	0.0001	0.0001
25%	0.0023	0.0084
50%	0.0035	0.0108
75%	0.0053	0.0132
90%	0.0085	0.0153
max	0.0305	0.0320
RMSE	0.0057	

Resolution: 800×600



Error per iteration for resolution 800×600 and test04

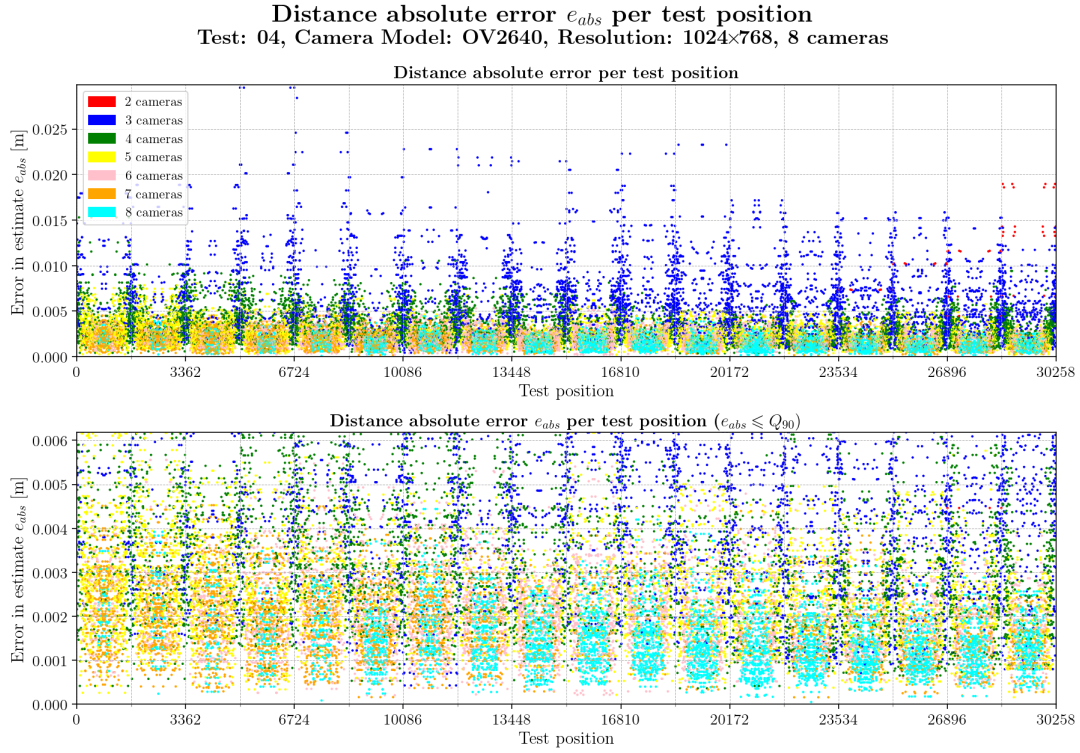
Table A.35: Statistics per number of cameras observing the beacon for test 04 with resolution 800×600.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	64	5576	4608	4800	4860	4168	6182
mean	0.0130	0.0071	0.0046	0.0033	0.0027	0.0024	0.0021
std	0.0058	0.0043	0.0023	0.0015	0.0012	0.0010	0.0009
min	0.0031	0.0006	0.0004	0.0003	0.0001	0.0001	0.0001
25%	0.0097	0.0040	0.0028	0.0022	0.0018	0.0016	0.0014
50%	0.0127	0.0062	0.0042	0.0031	0.0026	0.0023	0.0019
75%	0.0154	0.0091	0.0058	0.0041	0.0035	0.0031	0.0026
max	0.0279	0.0287	0.0143	0.0113	0.0077	0.0060	0.0070

Table A.36: Global statistics and root mean square error for test 04 with resolution 800×600.

	Abs. error	Est. error
count	30258	30258
mean	0.0037	0.0090
std	0.0029	0.0030
min	0.0001	0.0001
25%	0.0019	0.0071
50%	0.0029	0.0090
75%	0.0044	0.0110
90%	0.0071	0.0129
max	0.0287	0.0194
RMSE	0.0048	

Resolution: 1024×768



Error per iteration for resolution 1024×768 and test04

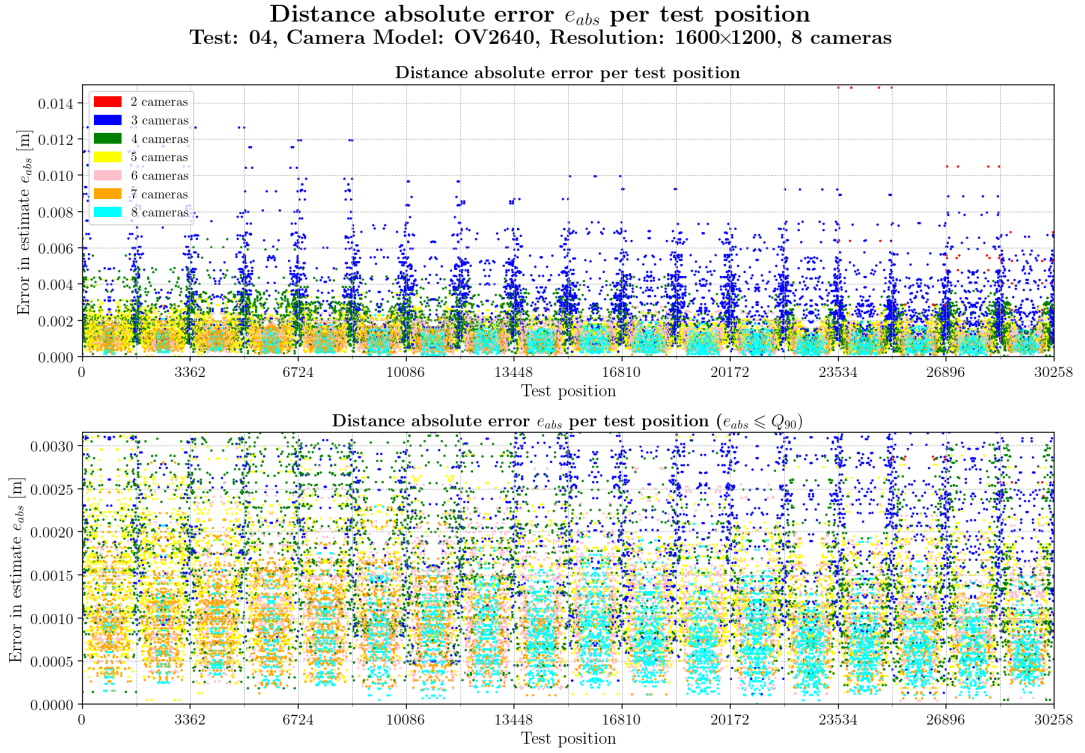
Table A.37: Statistics per number of cameras observing the beacon for test 04 with resolution 1024×768.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	56	5544	4628	4816	4848	4180	6186
mean	0.0111	0.0065	0.0036	0.0027	0.0021	0.0019	0.0015
std	0.0043	0.0038	0.0018	0.0013	0.0009	0.0008	0.0007
min	0.0045	0.0004	0.0004	0.0002	0.0002	0.0002	0.0000
25%	0.0073	0.0038	0.0023	0.0018	0.0014	0.0012	0.0010
50%	0.0109	0.0057	0.0033	0.0025	0.0020	0.0018	0.0014
75%	0.0137	0.0084	0.0046	0.0033	0.0027	0.0024	0.0019
max	0.0190	0.0295	0.0153	0.0081	0.0056	0.0046	0.0044

Table A.38: Global statistics and root mean square error for test 04 with resolution 1024×768.

	Abs. error	Est. error
count	30258	30258
mean	0.0031	0.0072
std	0.0027	0.0024
min	0.0000	0.0000
25%	0.0015	0.0055
50%	0.0023	0.0070
75%	0.0037	0.0088
90%	0.0062	0.0104
max	0.0295	0.0182
RMSE	0.0041	

Resolution: 1600×1200



Error per iteration for resolution 1600×1200 and test04

Table A.39: Statistics per number of cameras observing the beacon for test 04 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	56	5464	4664	4800	4888	4184	6202
mean	0.0051	0.0033	0.0018	0.0014	0.0011	0.0009	0.0008
std	0.0036	0.0019	0.0010	0.0006	0.0005	0.0004	0.0004
min	0.0008	0.0001	0.0001	0.0000	0.0000	0.0001	0.0000
25%	0.0028	0.0019	0.0011	0.0009	0.0008	0.0006	0.0005
50%	0.0048	0.0029	0.0017	0.0013	0.0011	0.0009	0.0007
75%	0.0058	0.0043	0.0024	0.0017	0.0014	0.0012	0.0010
max	0.0148	0.0126	0.0065	0.0042	0.0032	0.0028	0.0023

Table A.40: Global statistics and root mean square error for test 04 with resolution 1600×1200.

	Abs. error	Est. error
count	30258	30258
mean	0.0016	0.0038
std	0.0013	0.0013
min	0.0000	0.0000
25%	0.0008	0.0030
50%	0.0012	0.0038
75%	0.0019	0.0046
90%	0.0032	0.0055
max	0.0148	0.0119
RMSE	0.0021	

Error and camera coverage maps

Error maps for resolution 320×240

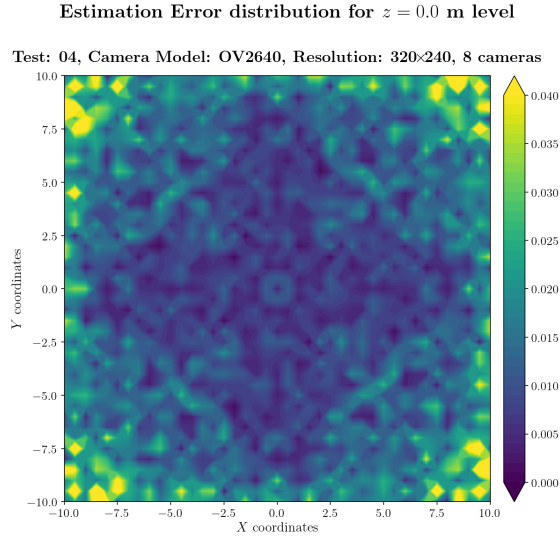


Figure A.461: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.000$ m.

Camera Coverage Spatial Distribution for $z = 0.0$ m level

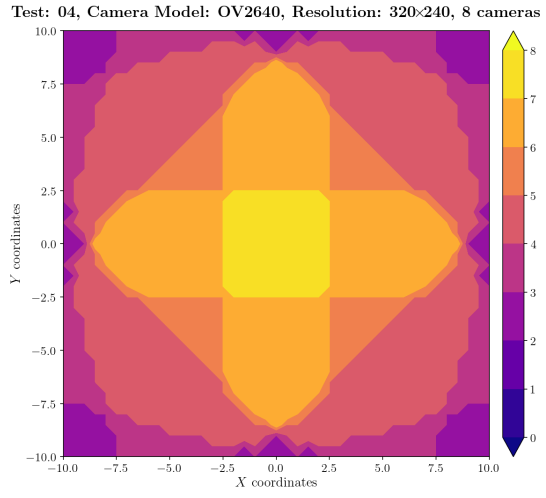


Figure A.462: Camera coverage for test 04 at level $z = 0.000$ m and resolution 320×240.

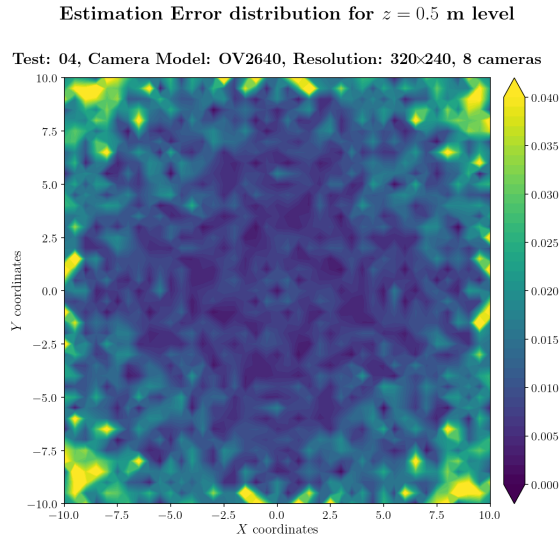


Figure A.463: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.500$ m.

Camera Coverage Spatial Distribution for $z = 0.5$ m level

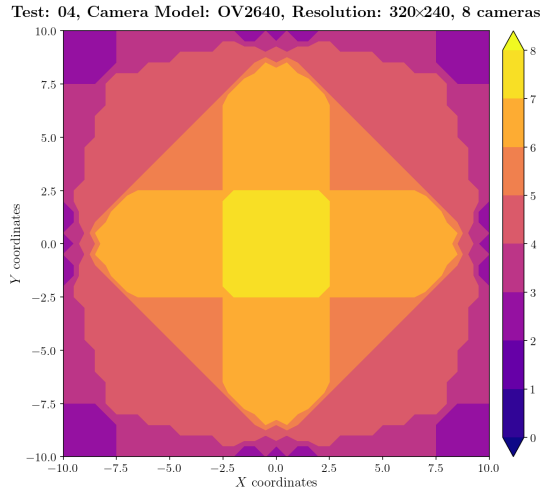


Figure A.464: Camera coverage for test 04 at level $z = 0.500$ m and resolution 320×240.

Estimation Error distribution for $z = 1.0$ m level

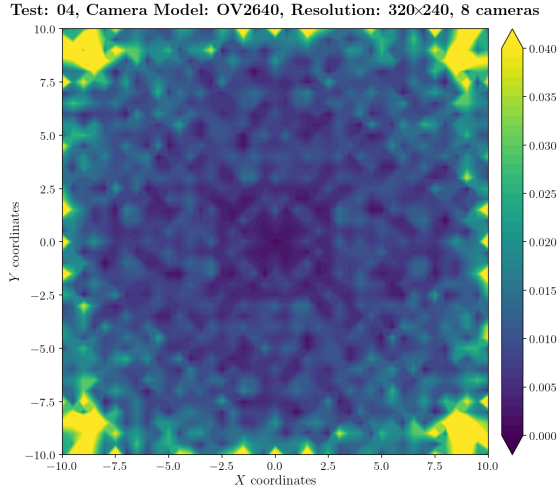


Figure A.465: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.000$ m.

Camera Coverage Spatial Distribution for $z = 1.0$ m level

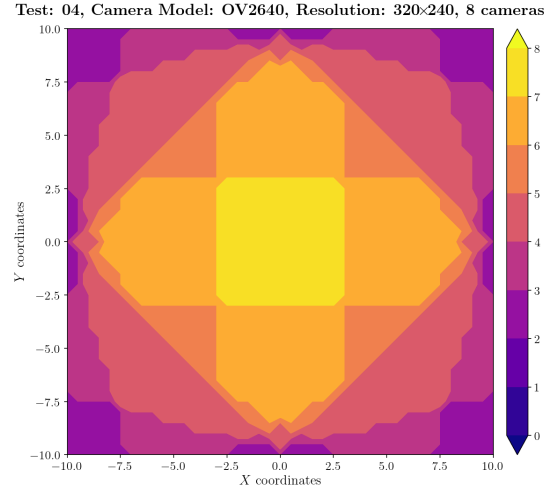


Figure A.466: Camera coverage for test 04 at level $z = 1.000$ m and resolution 320×240.

Estimation Error distribution for $z = 1.5$ m level

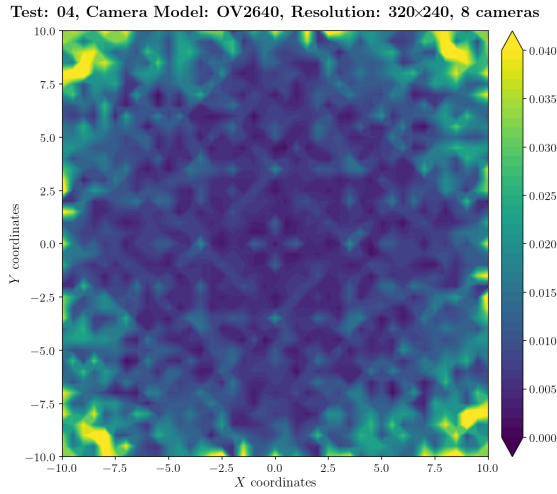


Figure A.467: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.500$ m.

Camera Coverage Spatial Distribution for $z = 1.5$ m level

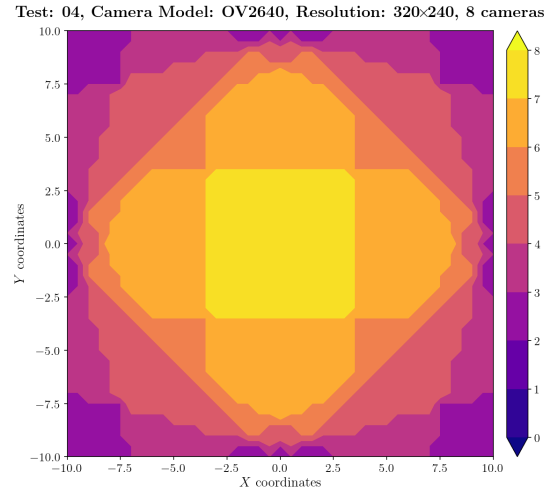
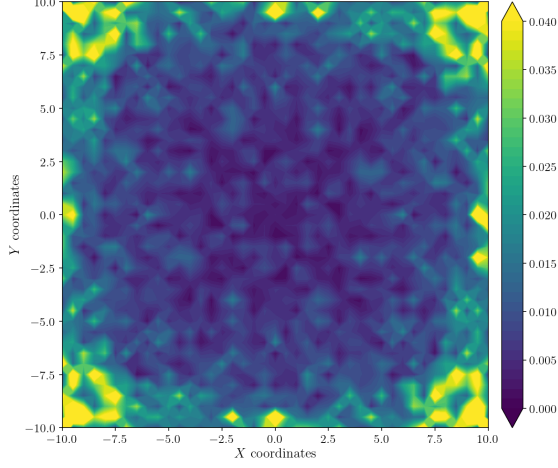


Figure A.468: Camera coverage for test 04 at level $z = 1.500$ m and resolution 320×240.

Estimation Error distribution for $z = 2.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

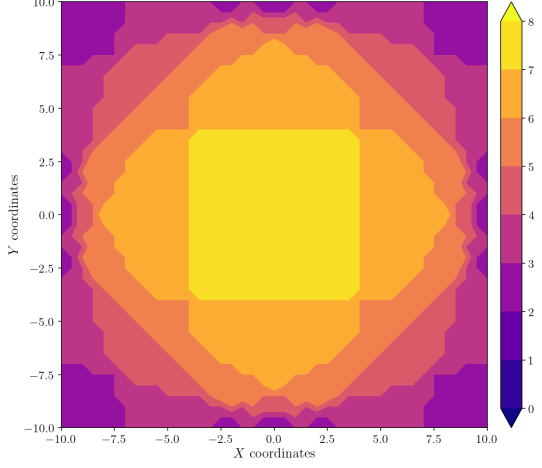
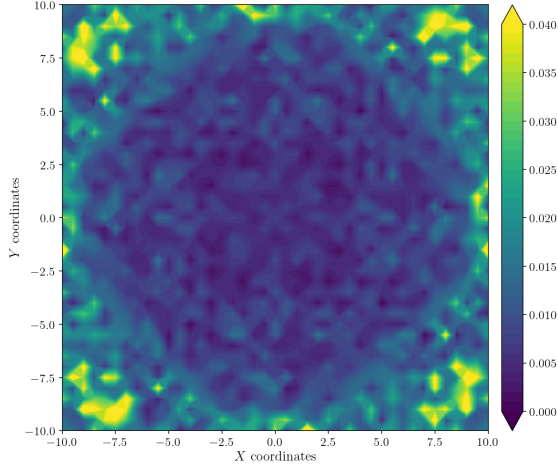


Figure A.469: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.000$ m.

Figure A.470: Camera coverage for test 04 at level $z = 2.000$ m and resolution 320×240.

Estimation Error distribution for $z = 2.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

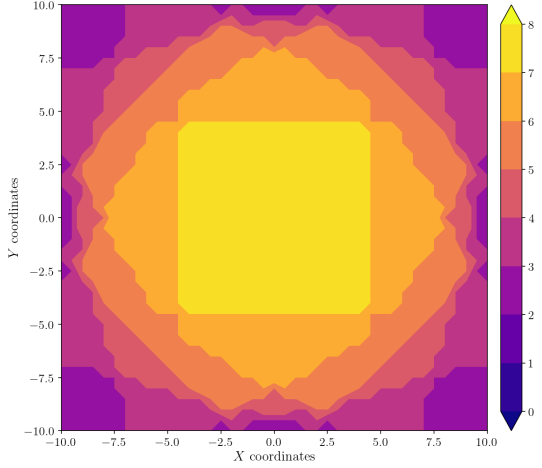


Figure A.471: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.500$ m.

Figure A.472: Camera coverage for test 04 at level $z = 2.500$ m and resolution 320×240.

Estimation Error distribution for $z = 3.0$ m level

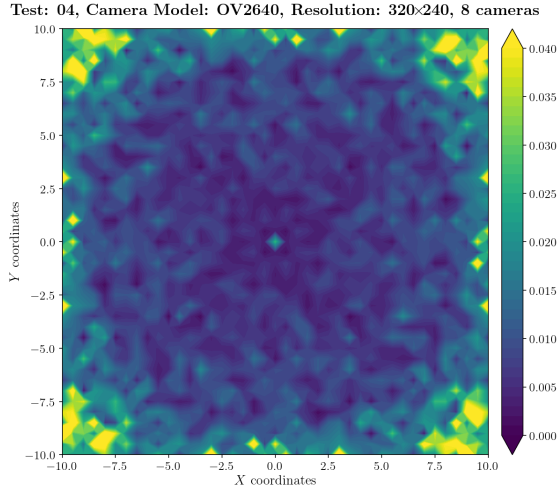


Figure A.473: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.000$ m.

Camera Coverage Spatial Distribution for $z = 3.0$ m level

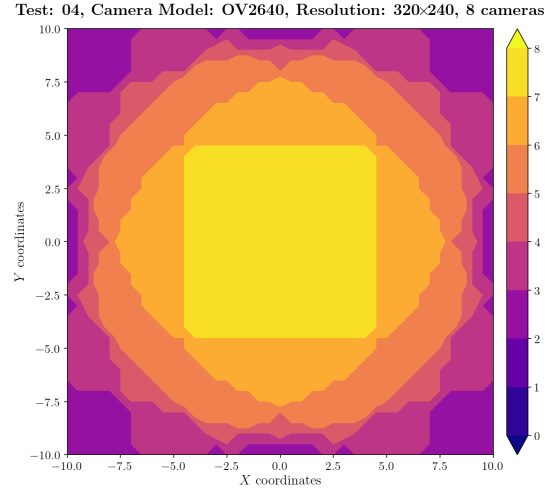


Figure A.474: Camera coverage for test 04 at level $z = 3.000$ m and resolution 320×240.

Estimation Error distribution for $z = 3.5$ m level

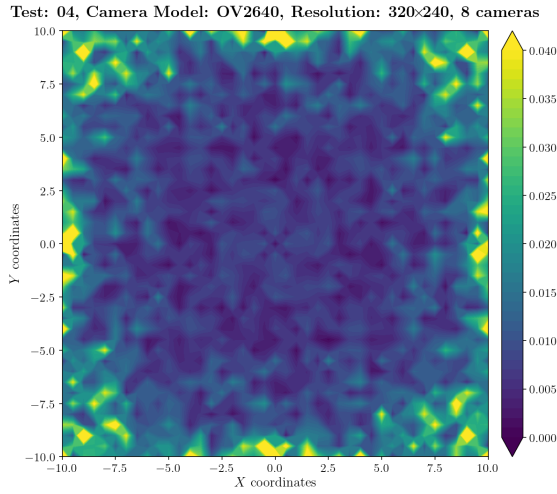


Figure A.475: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.500$ m.

Camera Coverage Spatial Distribution for $z = 3.5$ m level

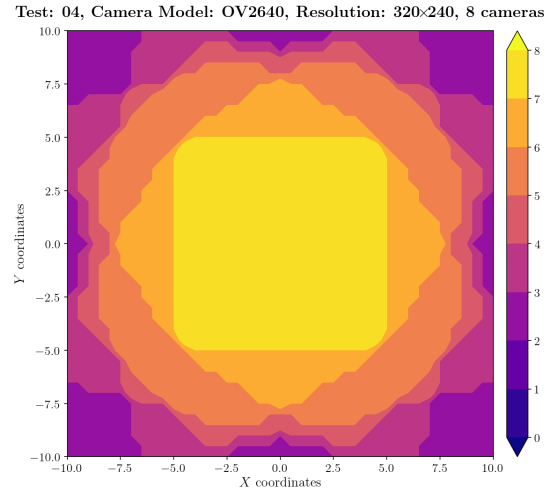


Figure A.476: Camera coverage for test 04 at level $z = 3.500$ m and resolution 320×240.

Estimation Error distribution for $z = 4.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

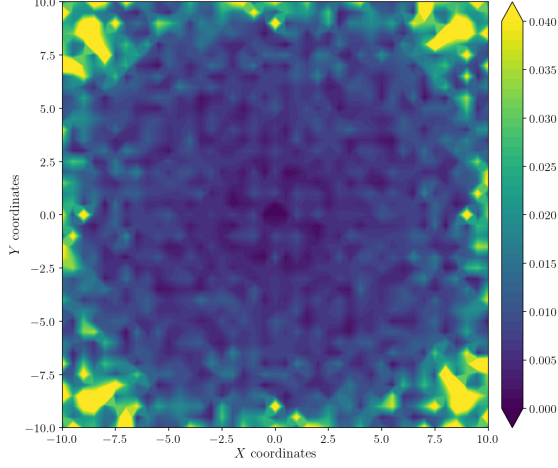


Figure A.477: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.000$ m.

Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

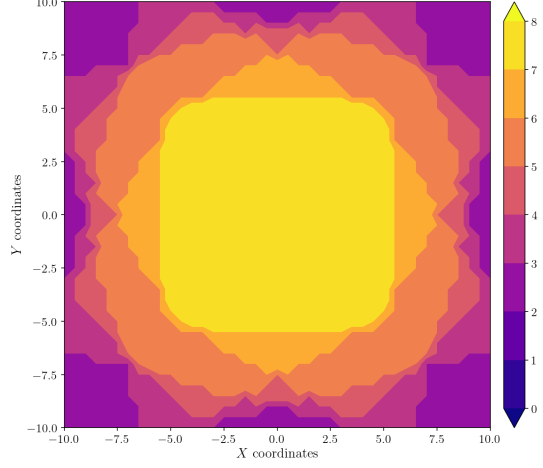


Figure A.478: Camera coverage for test 04 at level $z = 4.000$ m and resolution 320×240.

Estimation Error distribution for $z = 4.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

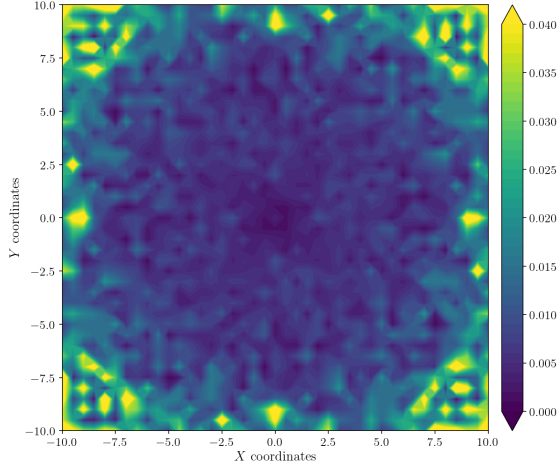


Figure A.479: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.500$ m.

Camera Coverage Spacial Distribution for $z = 4.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

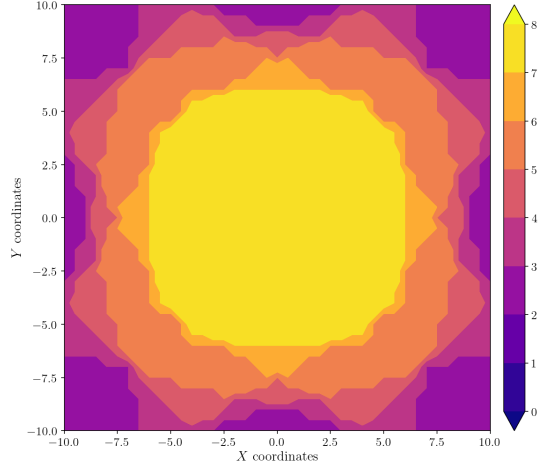


Figure A.480: Camera coverage for test 04 at level $z = 4.500$ m and resolution 320×240.

Estimation Error distribution for $z = 5.0$ m level

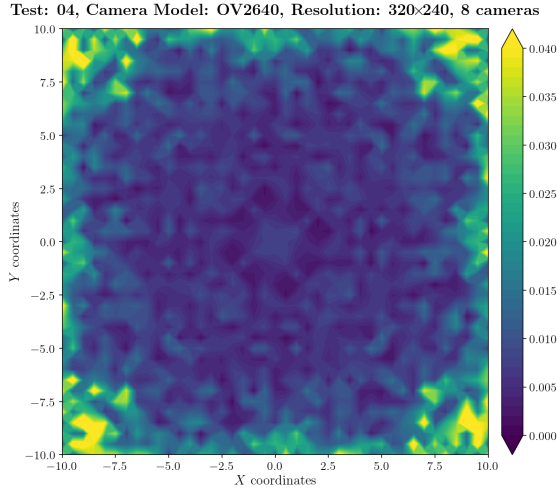


Figure A.481: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 5.000$ m.

Camera Coverage Spacial Distribution for $z = 5.0$ m level

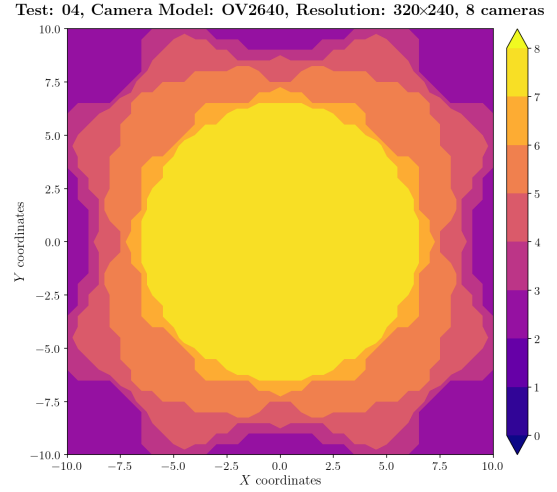


Figure A.482: Camera coverage for test 04 at level $z = 5.000$ m and resolution 320×240.

Estimation Error distribution for $z = 5.5$ m level

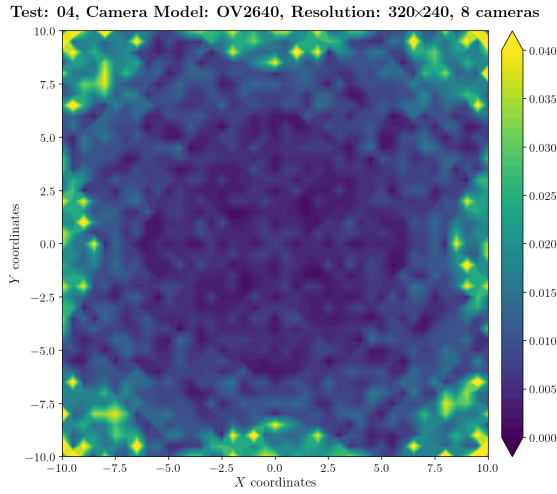


Figure A.483: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 5.500$ m.

Camera Coverage Spacial Distribution for $z = 5.5$ m level

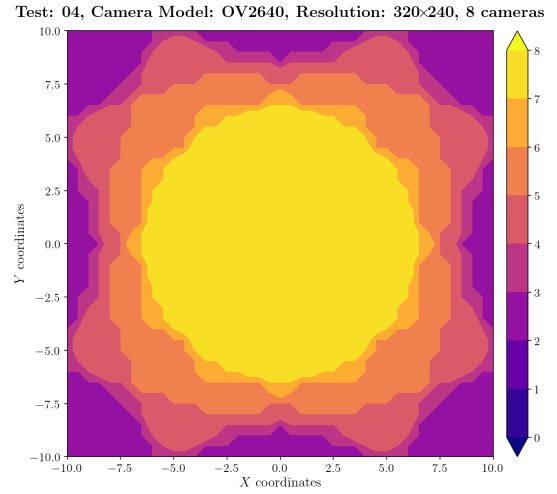
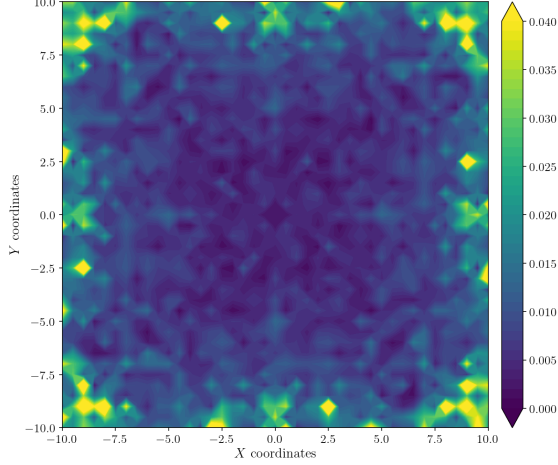


Figure A.484: Camera coverage for test 04 at level $z = 5.500$ m and resolution 320×240.

Estimation Error distribution for $z = 6.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

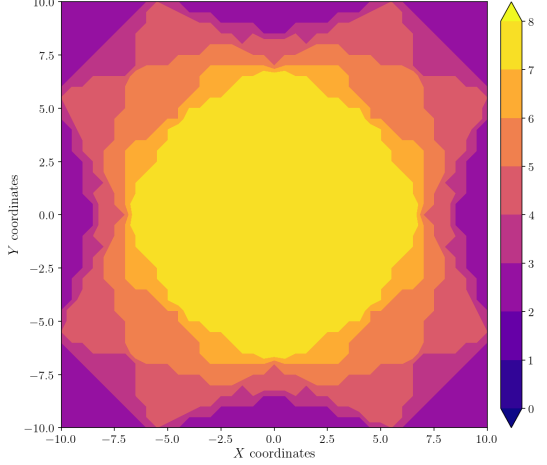
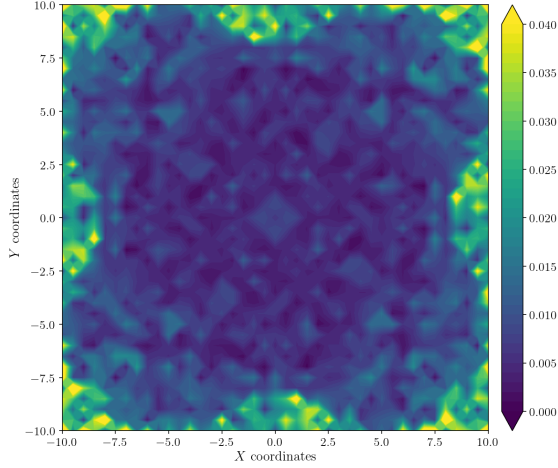


Figure A.485: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 6.000$ m.

Figure A.486: Camera coverage for test 04 at level $z = 6.000$ m and resolution 320×240.

Estimation Error distribution for $z = 6.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

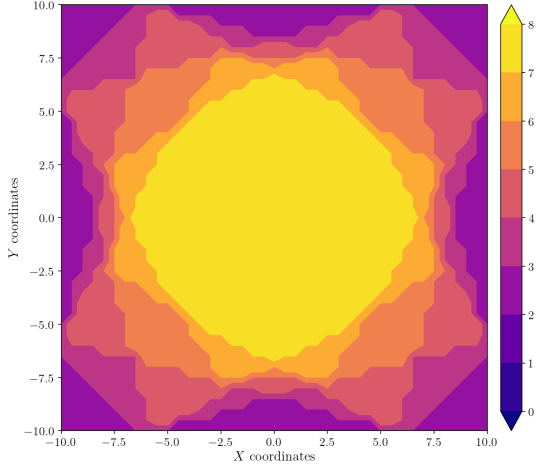


Figure A.487: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 6.500$ m.

Figure A.488: Camera coverage for test 04 at level $z = 6.500$ m and resolution 320×240.

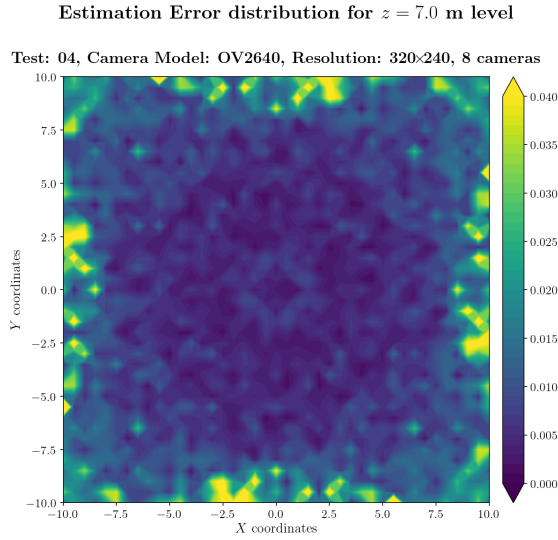


Figure A.489: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 7.000$ m.

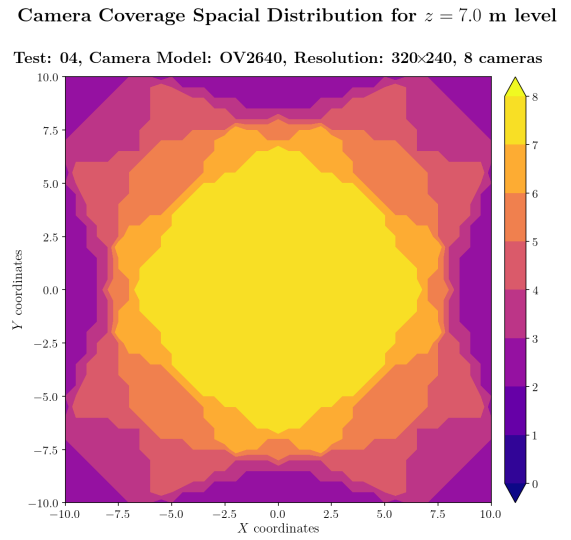


Figure A.490: Camera coverage for test 04 at level $z = 7.000$ m and resolution 320×240.

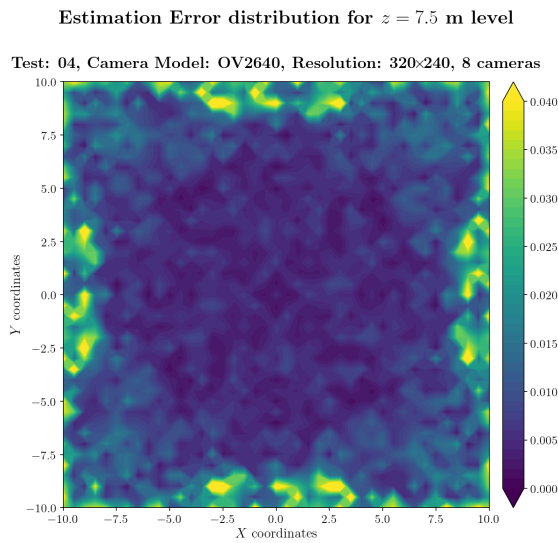


Figure A.491: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 7.500$ m.

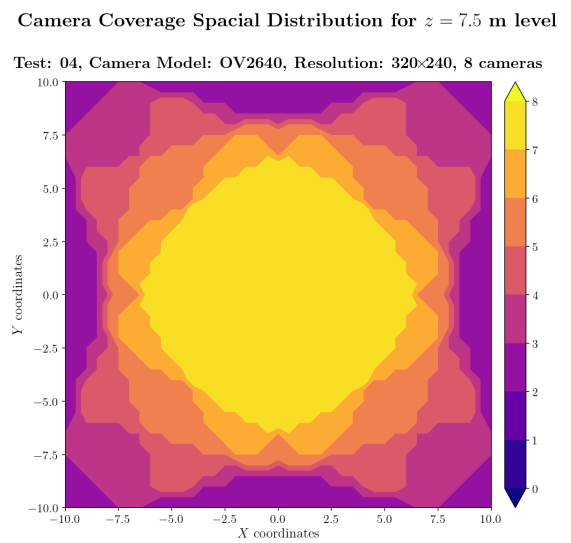
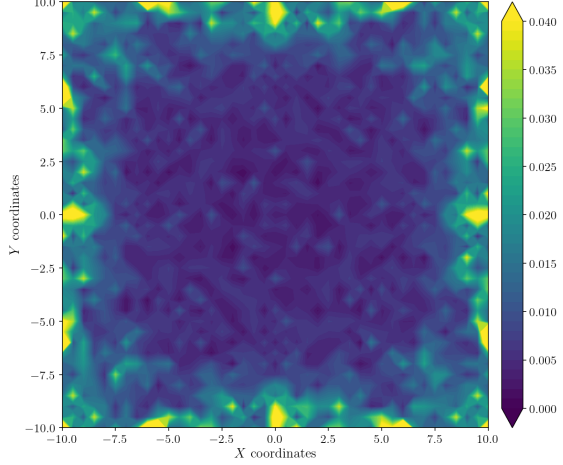


Figure A.492: Camera coverage for test 04 at level $z = 7.500$ m and resolution 320×240.

Estimation Error distribution for $z = 8.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

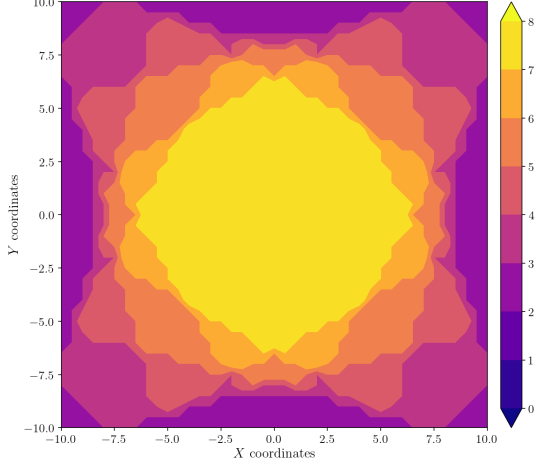
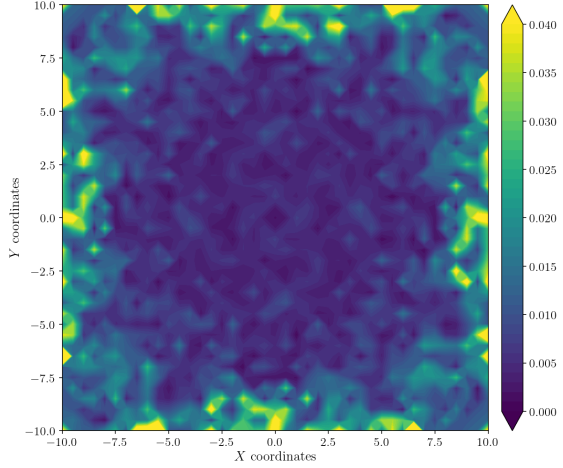


Figure A.493: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 8.000$ m.

Figure A.494: Camera coverage for test 04 at level $z = 8.000$ m and resolution 320×240.

Estimation Error distribution for $z = 8.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 320×240, 8 cameras

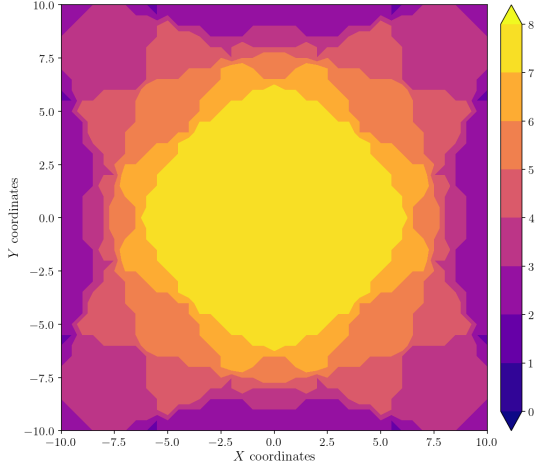


Figure A.495: Estimation error contour map for test 04 with 8 cameras using resolution of 320×240 pixels, at level $z = 8.500$ m.

Figure A.496: Camera coverage for test 04 at level $z = 8.500$ m and resolution 320×240.

Error maps for resolution 640×480

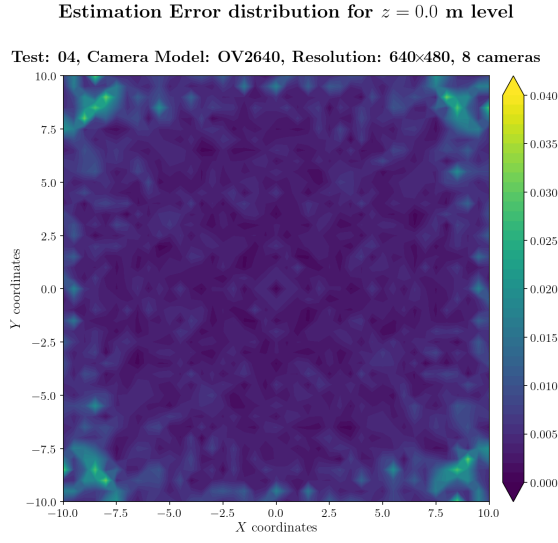


Figure A.497: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.000$ m.

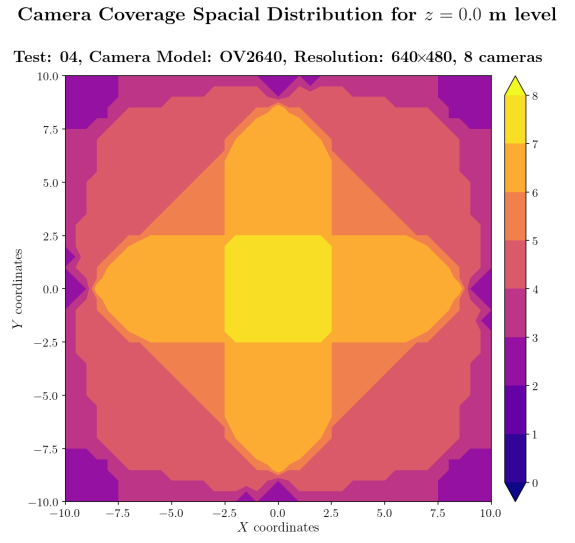


Figure A.498: Camera coverage for test 04 at level $z = 0.000$ m and resolution 640×480.

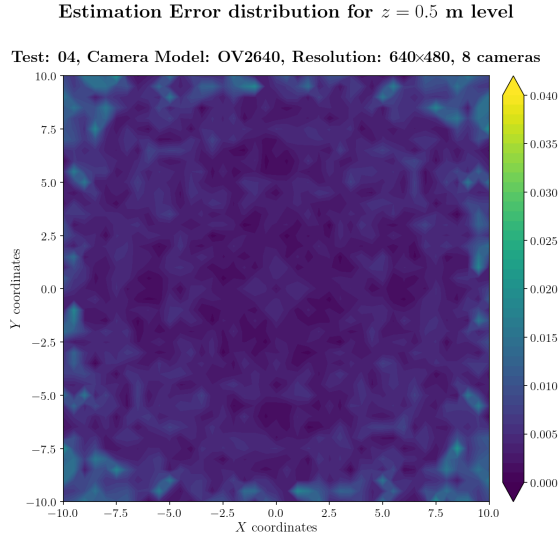


Figure A.499: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.500$ m.

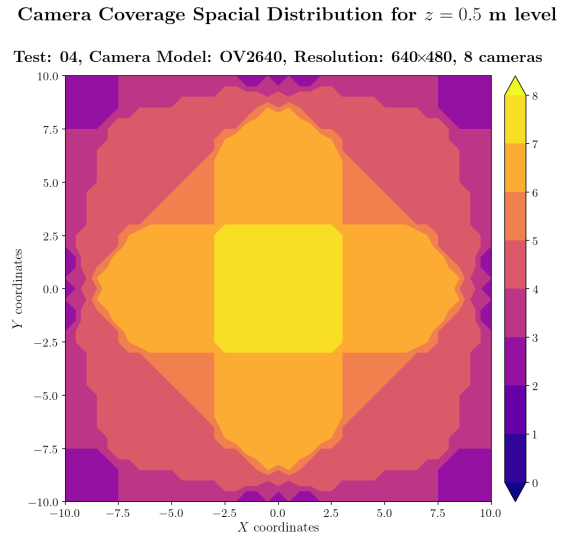
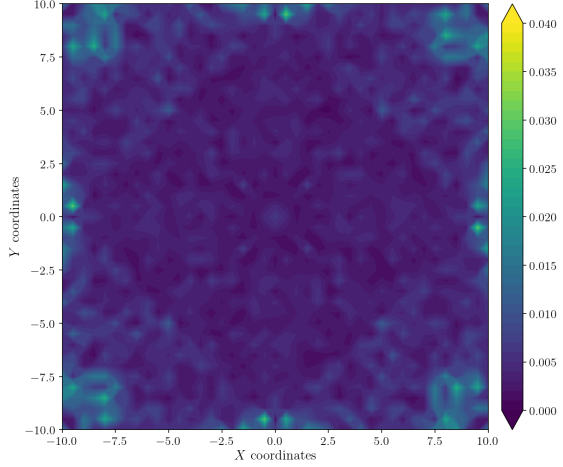


Figure A.500: Camera coverage for test 04 at level $z = 0.500$ m and resolution 640×480.

Estimation Error distribution for $z = 1.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 640×480, 8 cameras

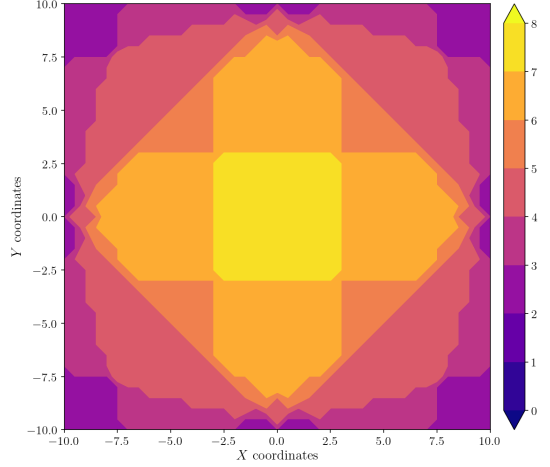
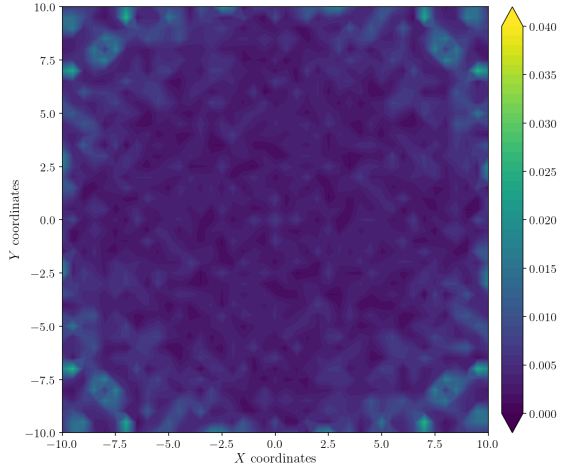


Figure A.501: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.000$ m.

Figure A.502: Camera coverage for test 04 at level $z = 1.000$ m and resolution 640×480.

Estimation Error distribution for $z = 1.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 640×480, 8 cameras

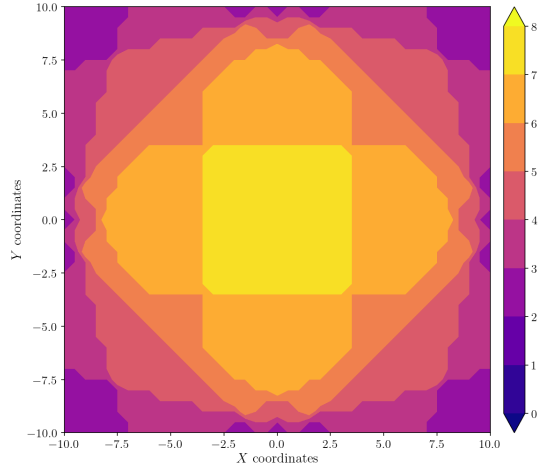


Figure A.503: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.500$ m.

Figure A.504: Camera coverage for test 04 at level $z = 1.500$ m and resolution 640×480.

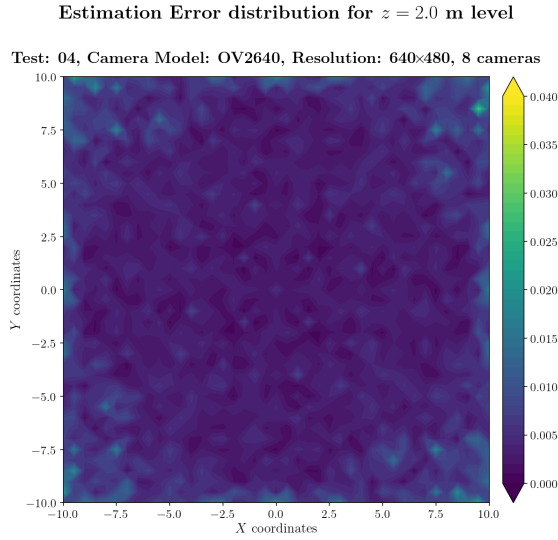


Figure A.505: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.000$ m.

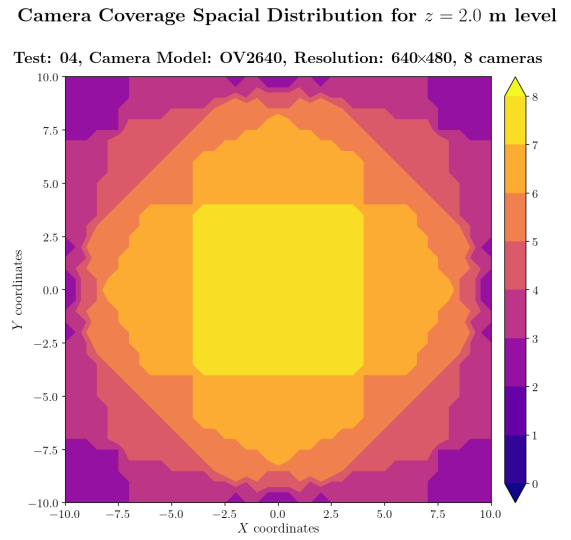


Figure A.506: Camera coverage for test 04 at level $z = 2.000$ m and resolution 640×480.

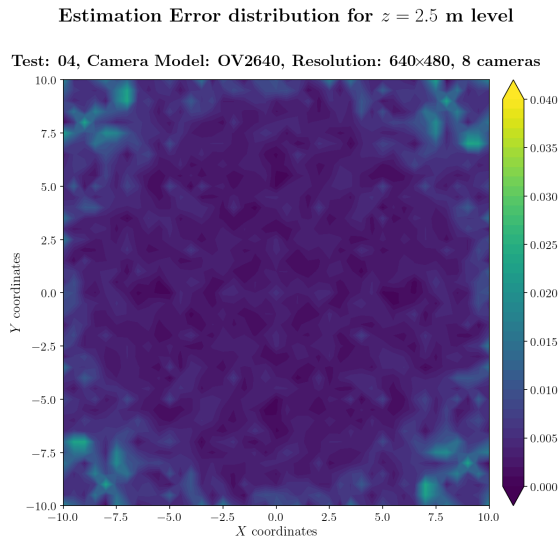


Figure A.507: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.500$ m.

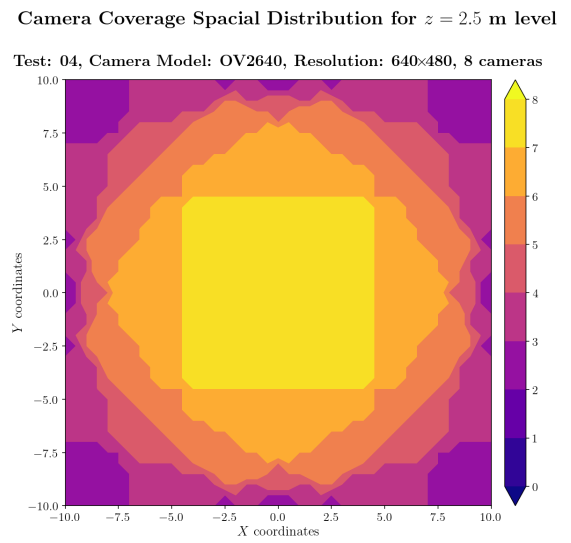


Figure A.508: Camera coverage for test 04 at level $z = 2.500$ m and resolution 640×480.

Estimation Error distribution for $z = 3.0$ m level

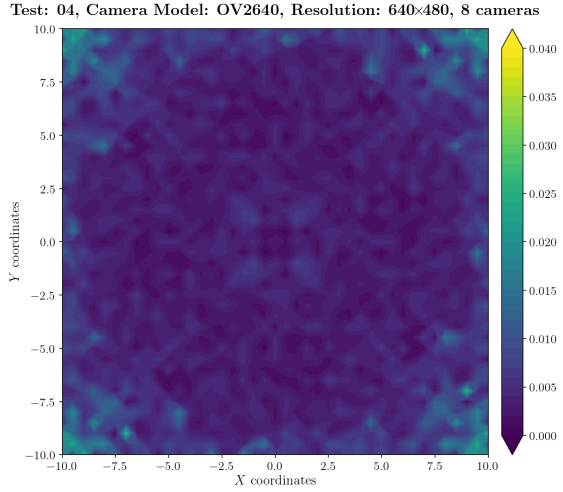


Figure A.509: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.000$ m.

Camera Coverage Spacial Distribution for $z = 3.0$ m level

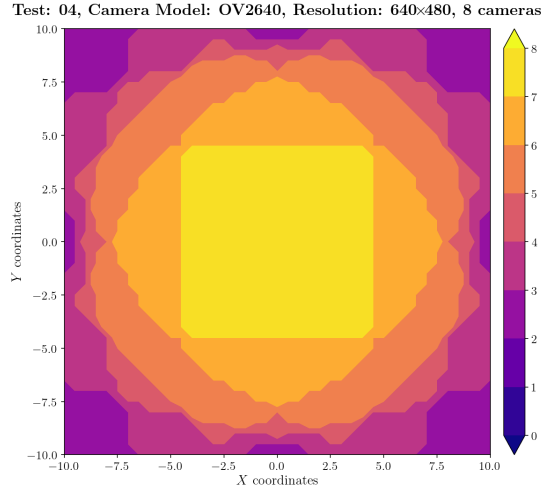


Figure A.510: Camera coverage for test 04 at level $z = 3.000$ m and resolution 640×480.

Estimation Error distribution for $z = 3.5$ m level

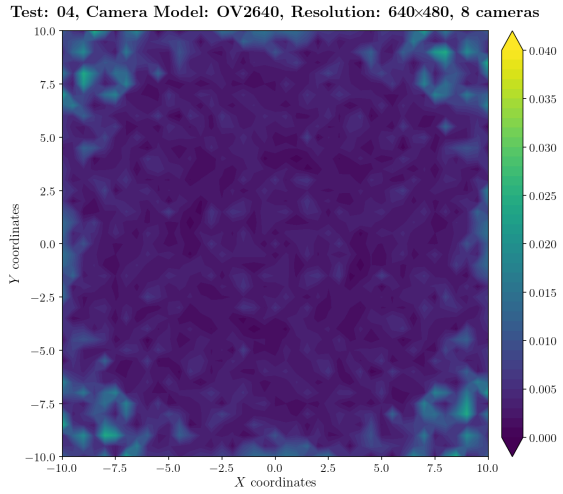


Figure A.511: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.500$ m.

Camera Coverage Spacial Distribution for $z = 3.5$ m level

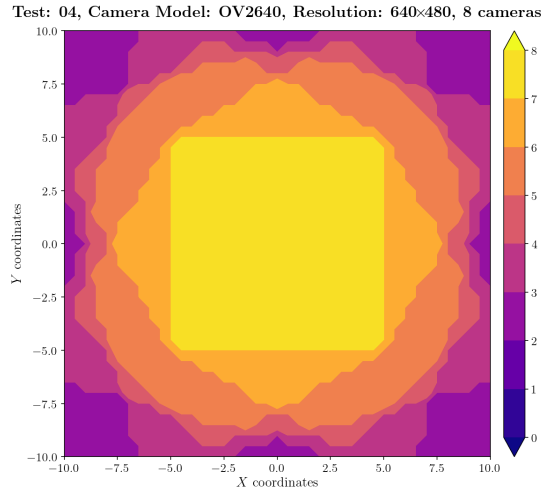


Figure A.512: Camera coverage for test 04 at level $z = 3.500$ m and resolution 640×480.

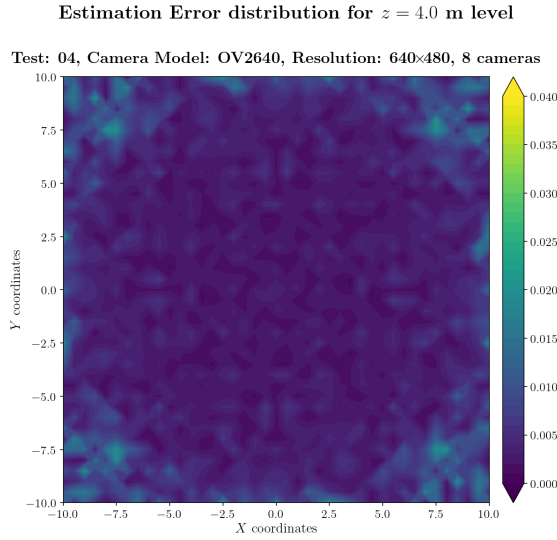


Figure A.513: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.000$ m.

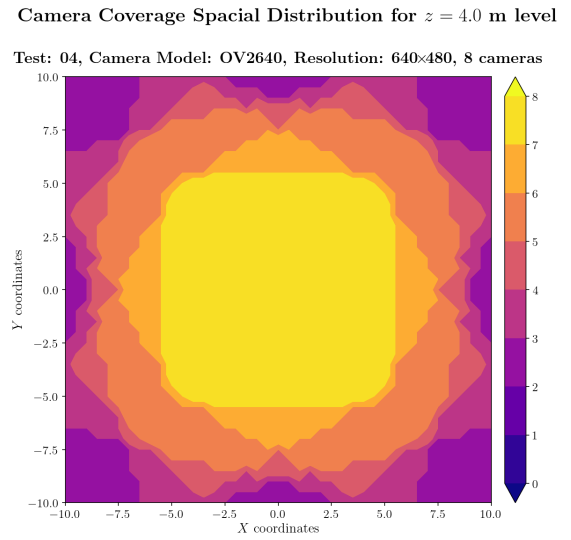


Figure A.514: Camera coverage for test 04 at level $z = 4.000$ m and resolution 640×480.

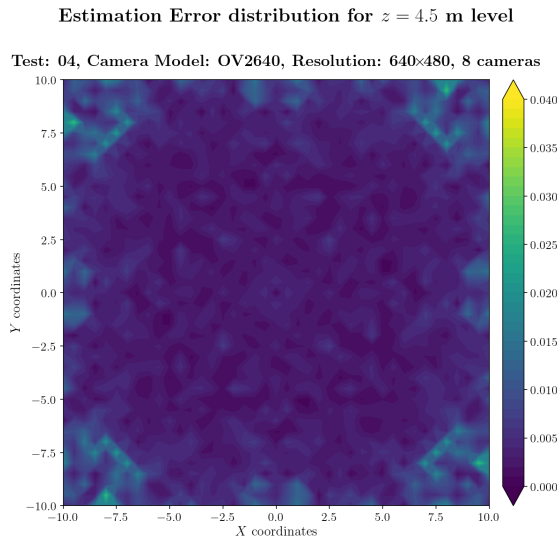


Figure A.515: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.500$ m.

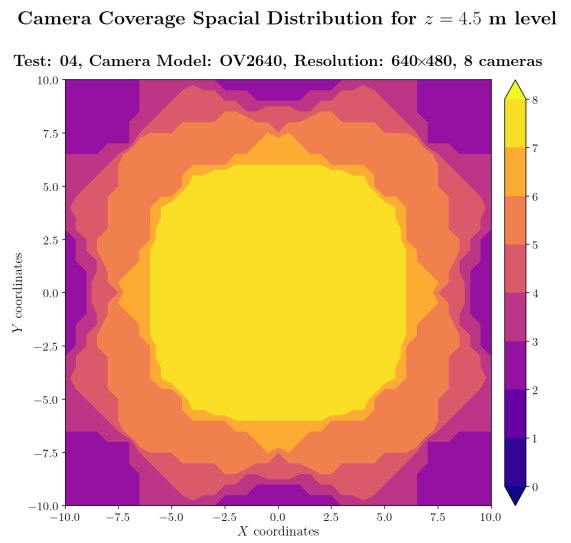


Figure A.516: Camera coverage for test 04 at level $z = 4.500$ m and resolution 640×480.

Estimation Error distribution for $z = 5.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 640×480, 8 cameras

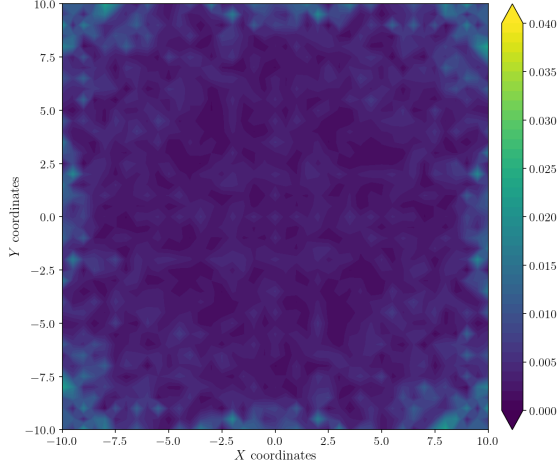


Figure A.517: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 5.000$ m.

Camera Coverage Spacial Distribution for $z = 5.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 640×480, 8 cameras

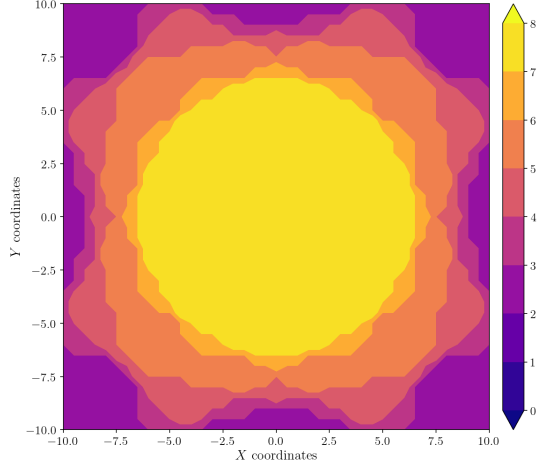


Figure A.518: Camera coverage for test 04 at level $z = 5.000$ m and resolution 640×480.

Estimation Error distribution for $z = 5.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 640×480, 8 cameras

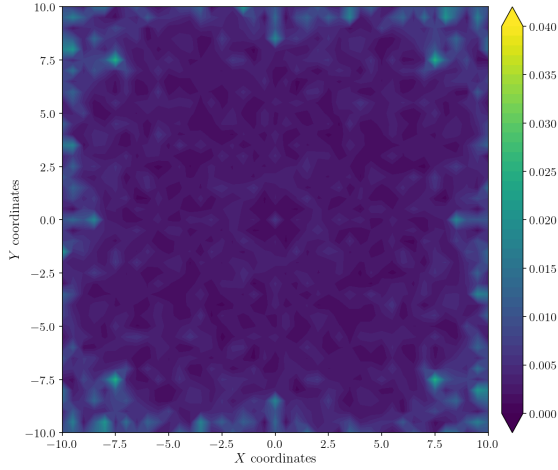


Figure A.519: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 5.500$ m.

Camera Coverage Spacial Distribution for $z = 5.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 640×480, 8 cameras

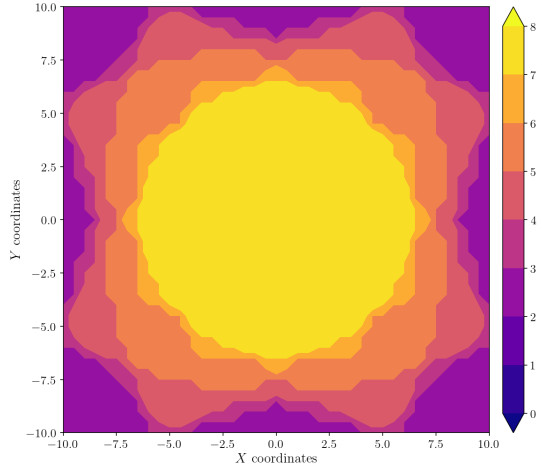


Figure A.520: Camera coverage for test 04 at level $z = 5.500$ m and resolution 640×480.

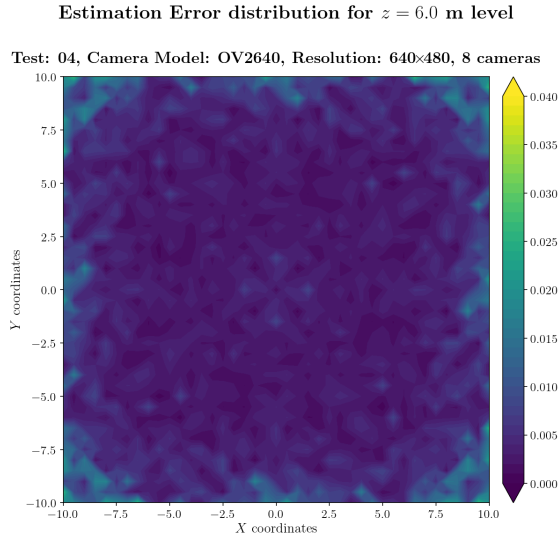


Figure A.521: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 6.000$ m.

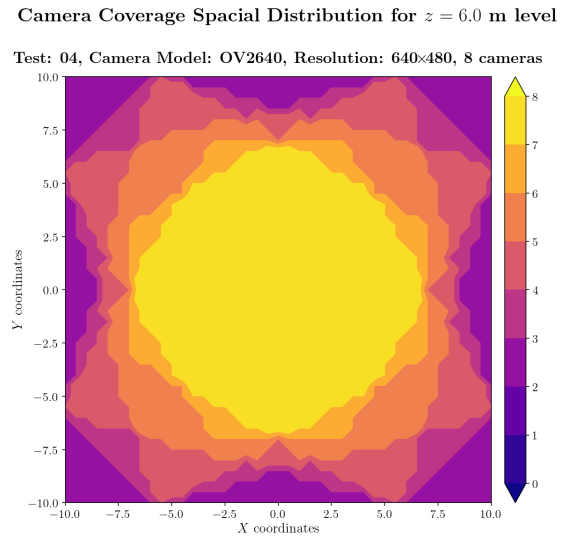


Figure A.522: Camera coverage for test 04 at level $z = 6.000$ m and resolution 640×480.

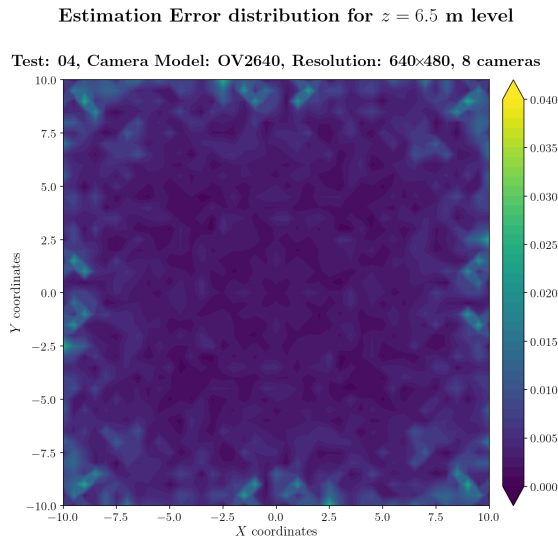


Figure A.523: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 6.500$ m.

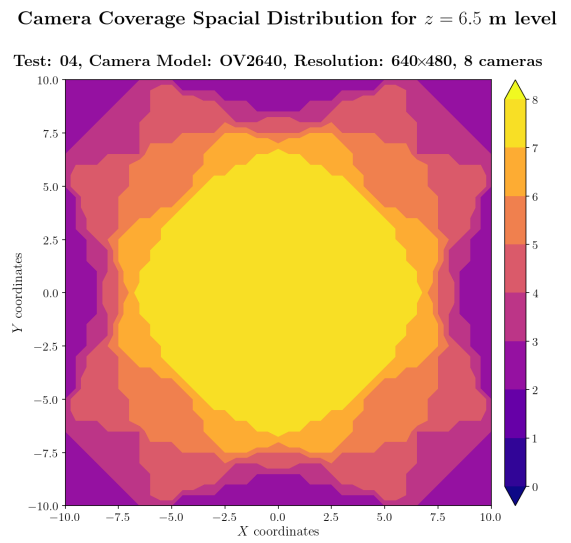
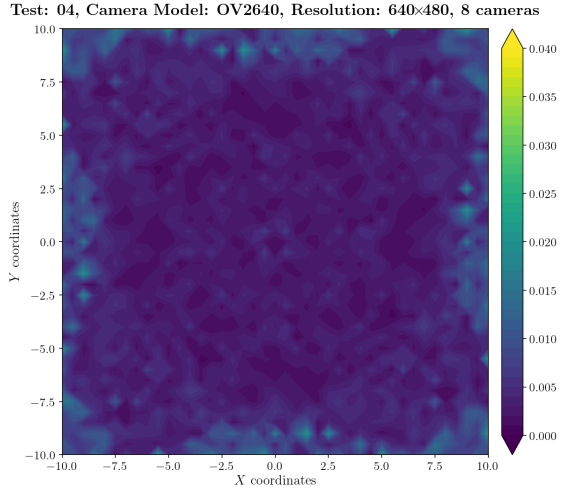


Figure A.524: Camera coverage for test 04 at level $z = 6.500$ m and resolution 640×480.

Estimation Error distribution for $z = 7.0$ m level



Camera Coverage Spacial Distribution for $z = 7.0$ m level

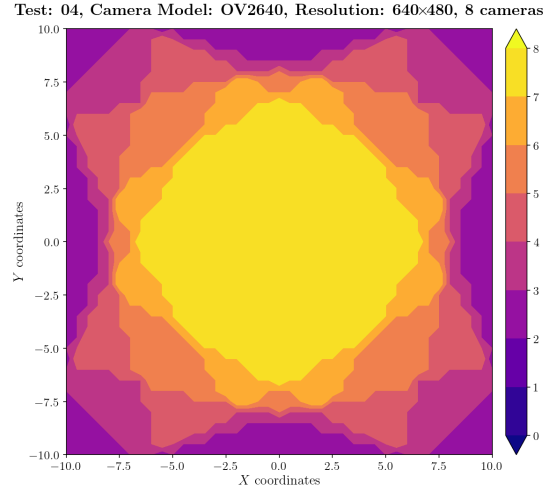
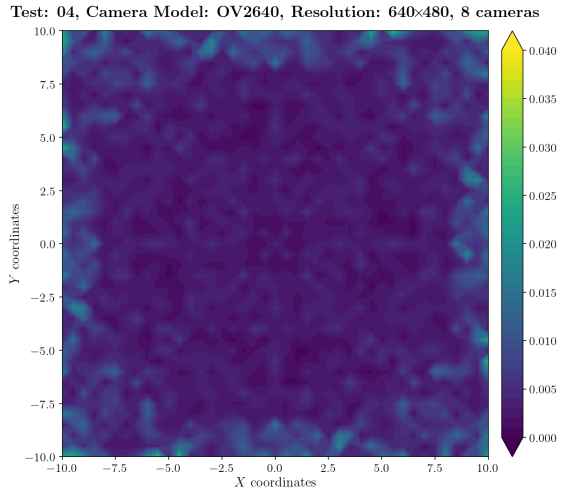


Figure A.525: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 7.000$ m.

Figure A.526: Camera coverage for test 04 at level $z = 7.000$ m and resolution 640×480.

Estimation Error distribution for $z = 7.5$ m level



Camera Coverage Spacial Distribution for $z = 7.5$ m level

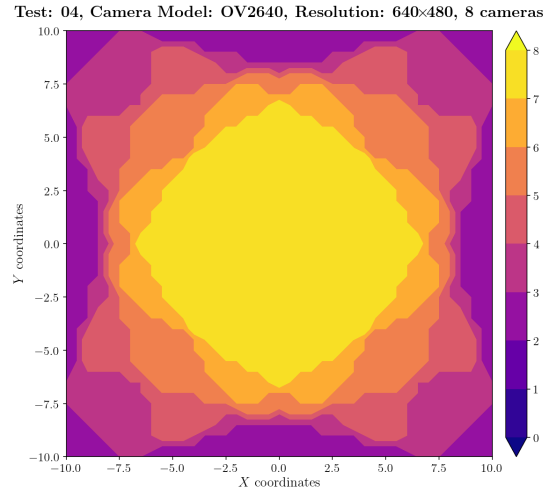


Figure A.527: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 7.500$ m.

Figure A.528: Camera coverage for test 04 at level $z = 7.500$ m and resolution 640×480.

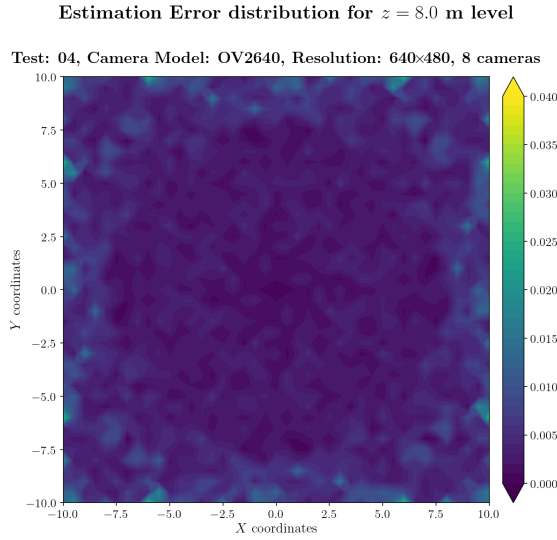


Figure A.529: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 8.000$ m.

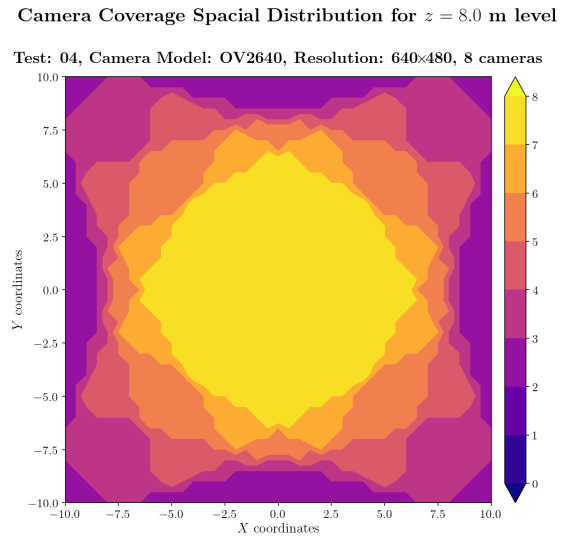


Figure A.530: Camera coverage for test 04 at level $z = 8.000$ m and resolution 640×480.

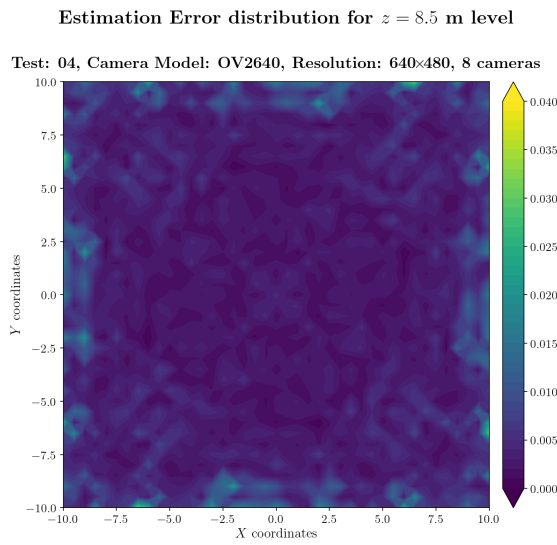


Figure A.531: Estimation error contour map for test 04 with 8 cameras using resolution of 640×480 pixels, at level $z = 8.500$ m.

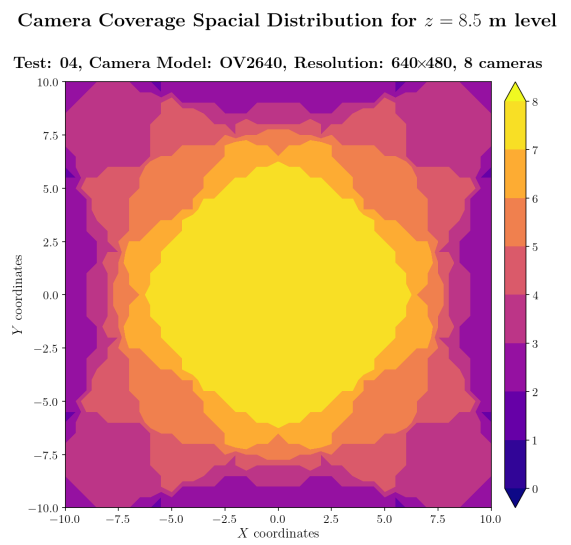


Figure A.532: Camera coverage for test 04 at level $z = 8.500$ m and resolution 640×480.

Error maps for resolution 800×600

Estimation Error distribution for $z = 0.0$ m level

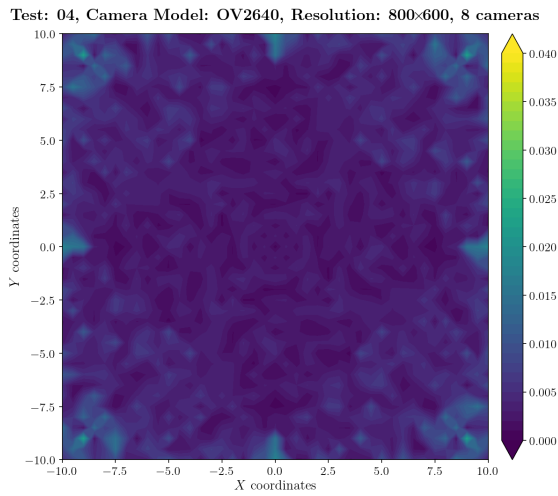


Figure A.533: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

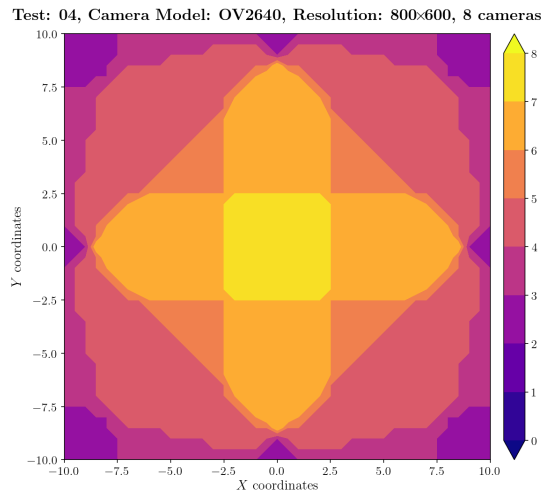


Figure A.534: Camera coverage for test 04 at level $z = 0.000$ m and resolution 800×600.

Estimation Error distribution for $z = 0.5$ m level

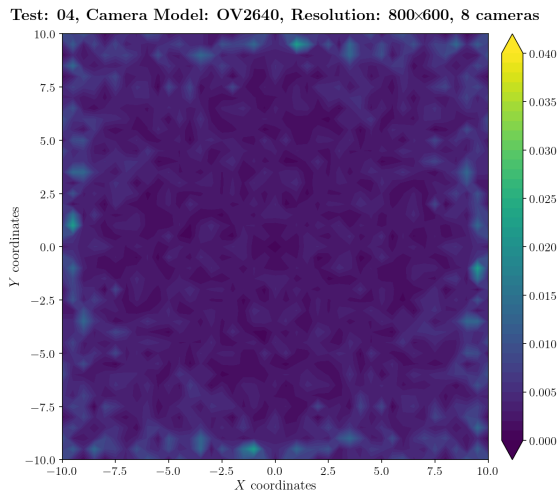


Figure A.535: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.500$ m.

Camera Coverage Spacial Distribution for $z = 0.5$ m level

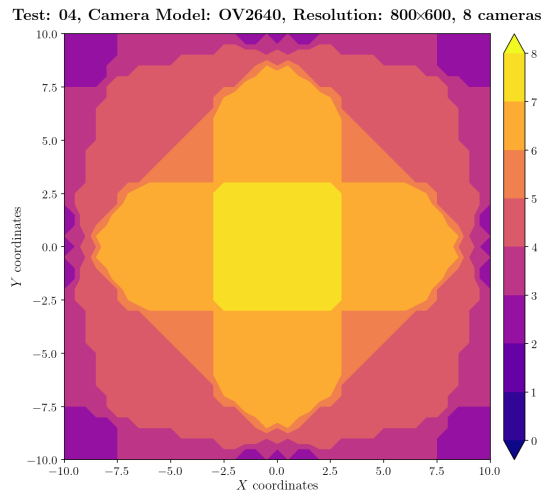


Figure A.536: Camera coverage for test 04 at level $z = 0.500$ m and resolution 800×600.

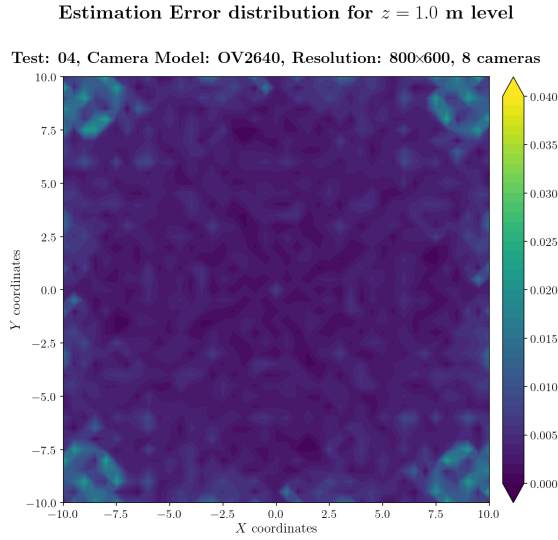


Figure A.537: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.000$ m.

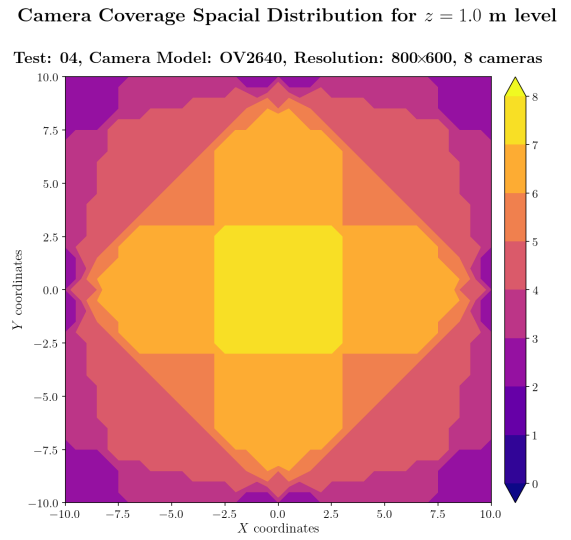


Figure A.538: Camera coverage for test 04 at level $z = 1.000$ m and resolution 800×600.

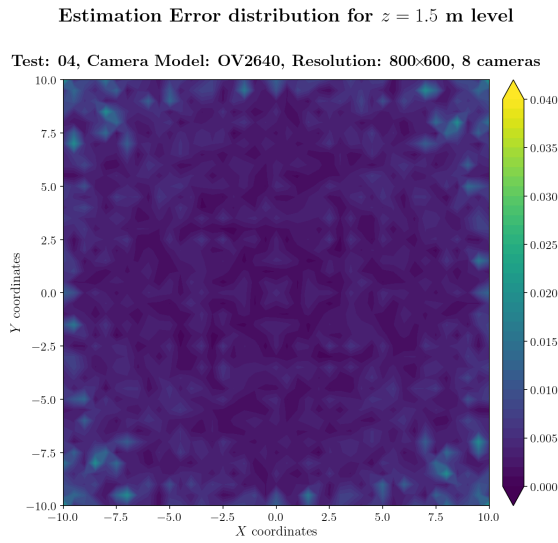


Figure A.539: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.500$ m.

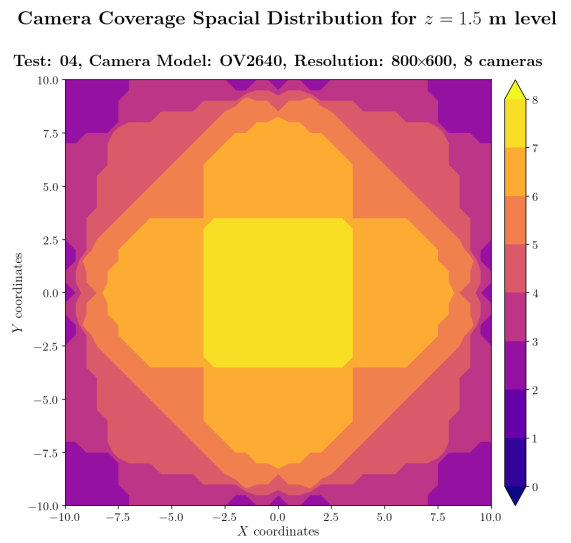


Figure A.540: Camera coverage for test 04 at level $z = 1.500$ m and resolution 800×600.

Estimation Error distribution for $z = 2.0$ m level

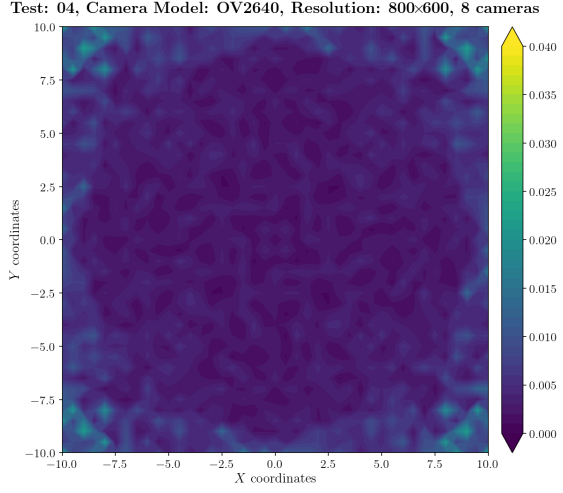


Figure A.541: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.000$ m.

Camera Coverage Spacial Distribution for $z = 2.0$ m level

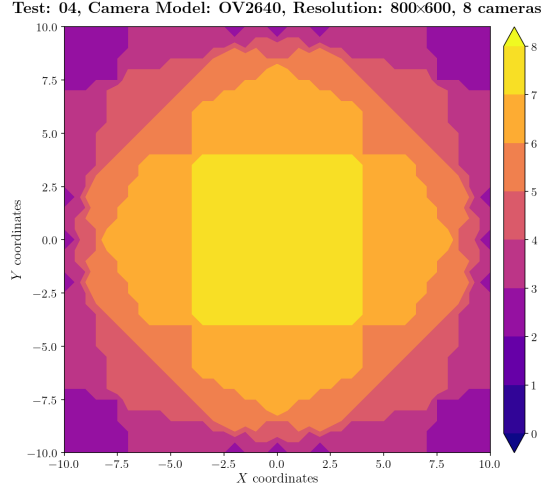


Figure A.542: Camera coverage for test 04 at level $z = 2.000$ m and resolution 800×600.

Estimation Error distribution for $z = 2.5$ m level

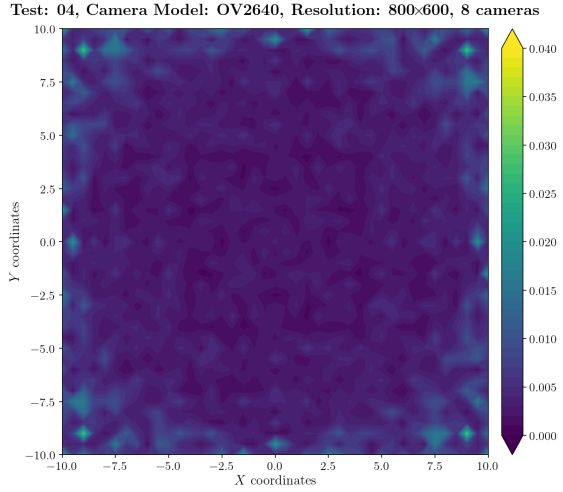


Figure A.543: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.500$ m.

Camera Coverage Spacial Distribution for $z = 2.5$ m level

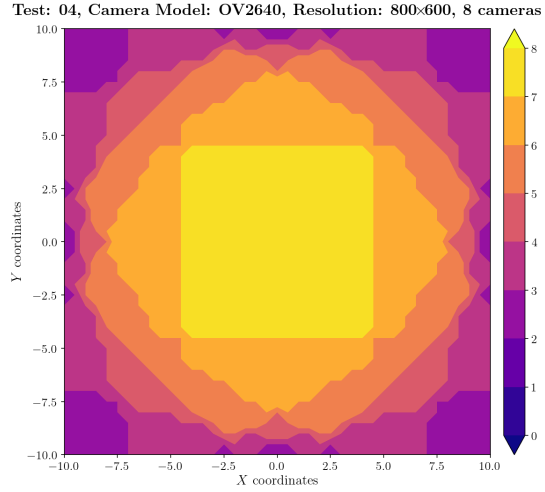


Figure A.544: Camera coverage for test 04 at level $z = 2.500$ m and resolution 800×600.

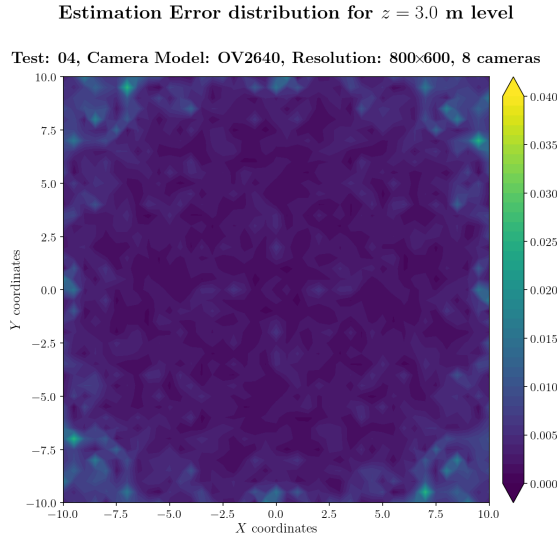


Figure A.545: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.000$ m.

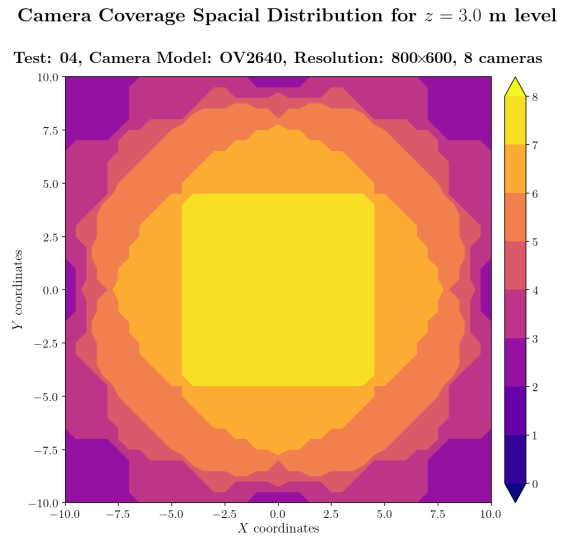


Figure A.546: Camera coverage for test 04 at level $z = 3.000$ m and resolution 800×600.

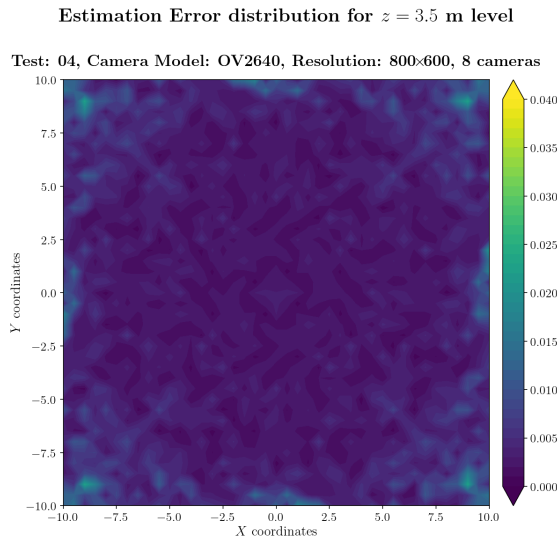


Figure A.547: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.500$ m.

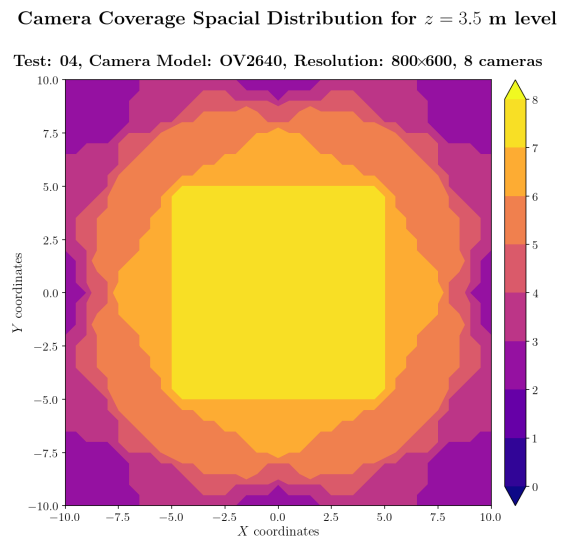
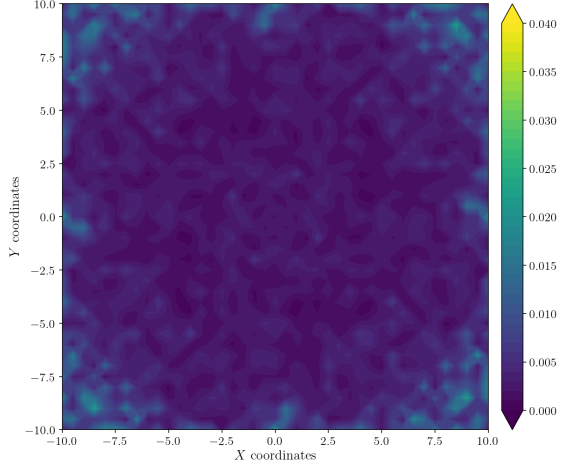


Figure A.548: Camera coverage for test 04 at level $z = 3.500$ m and resolution 800×600.

Estimation Error distribution for $z = 4.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 800×600, 8 cameras

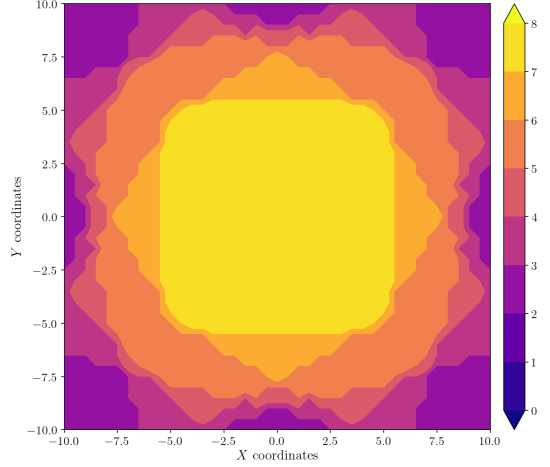
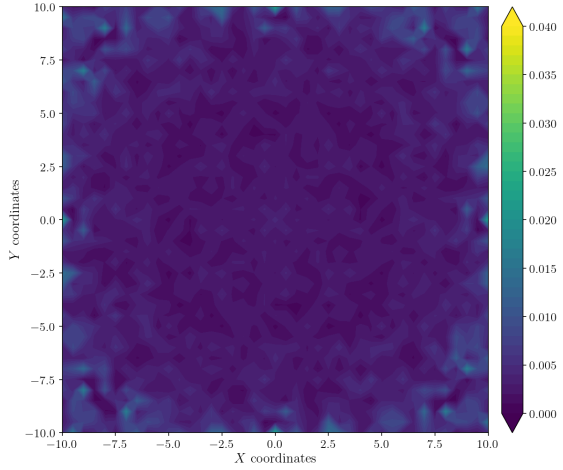


Figure A.549: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.000$ m.

Figure A.550: Camera coverage for test 04 at level $z = 4.000$ m and resolution 800×600.

Estimation Error distribution for $z = 4.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 800×600, 8 cameras

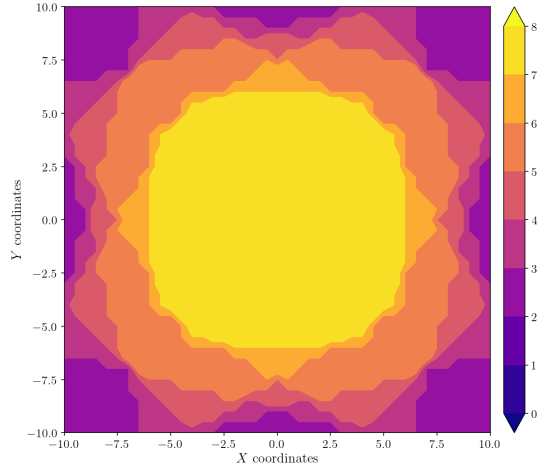


Figure A.551: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.500$ m.

Figure A.552: Camera coverage for test 04 at level $z = 4.500$ m and resolution 800×600.

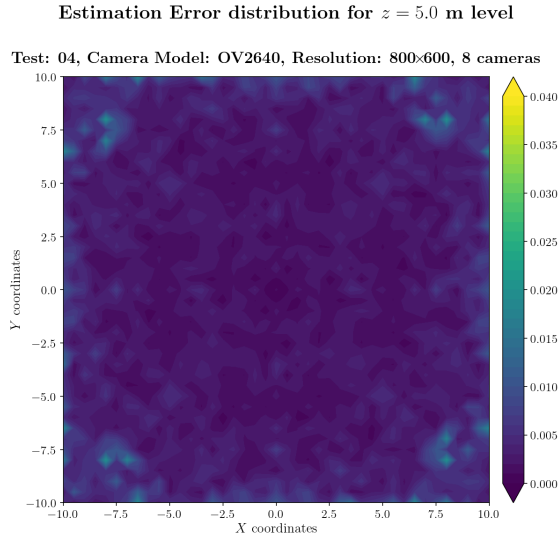


Figure A.553: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 5.000$ m.

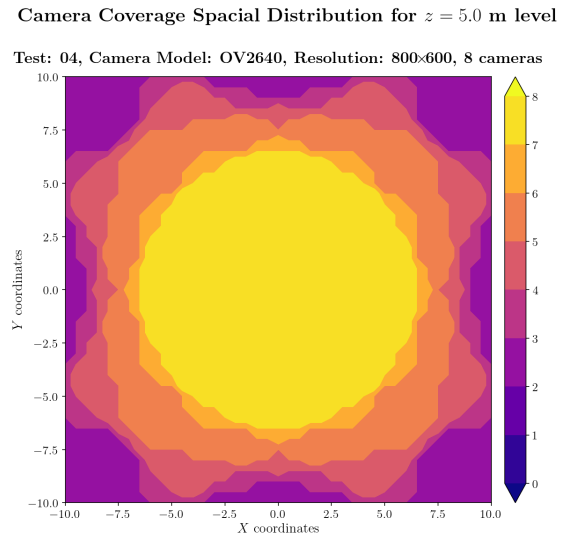


Figure A.554: Camera coverage for test 04 at level $z = 5.000$ m and resolution 800×600.

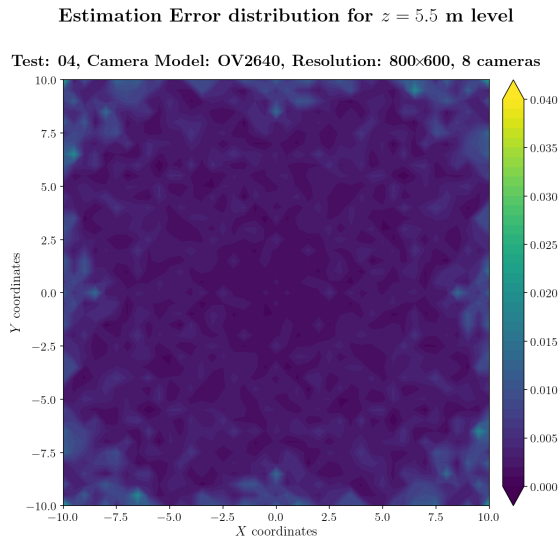


Figure A.555: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 5.500$ m.

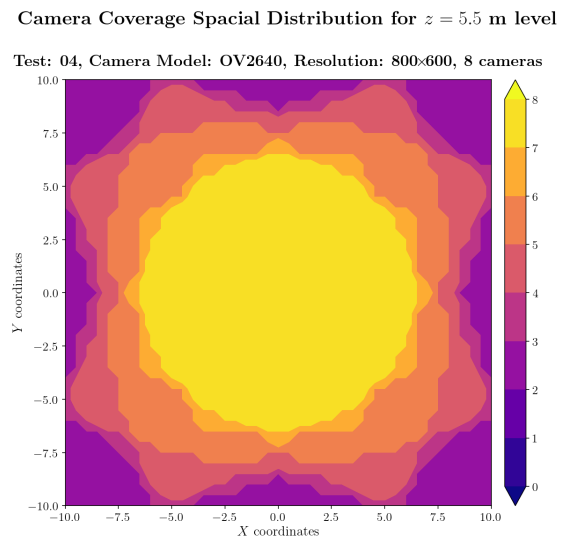


Figure A.556: Camera coverage for test 04 at level $z = 5.500$ m and resolution 800×600.

Estimation Error distribution for $z = 6.0$ m level

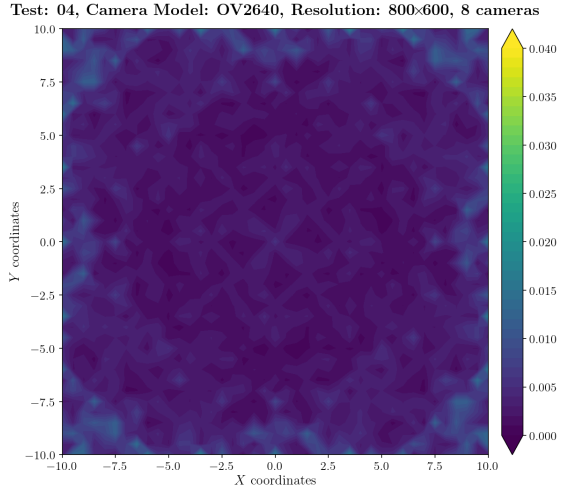


Figure A.557: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 6.000$ m.

Camera Coverage Spacial Distribution for $z = 6.0$ m level

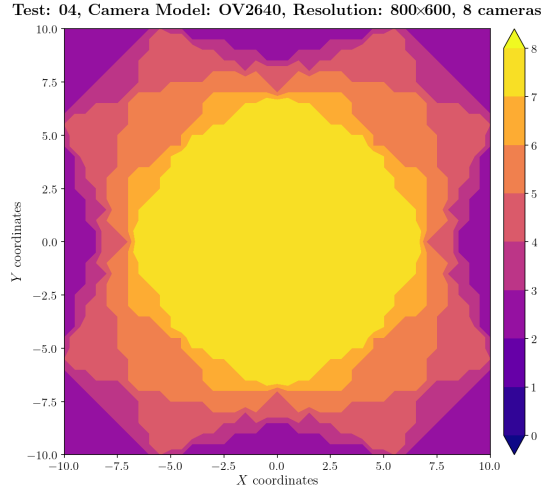


Figure A.558: Camera coverage for test 04 at level $z = 6.000$ m and resolution 800×600.

Estimation Error distribution for $z = 6.5$ m level

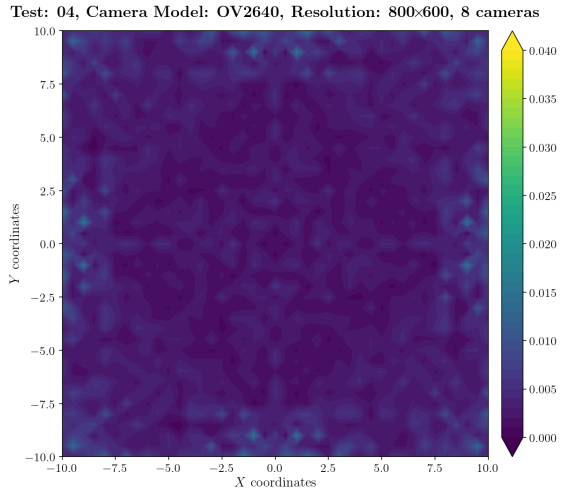


Figure A.559: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 6.500$ m.

Camera Coverage Spacial Distribution for $z = 6.5$ m level

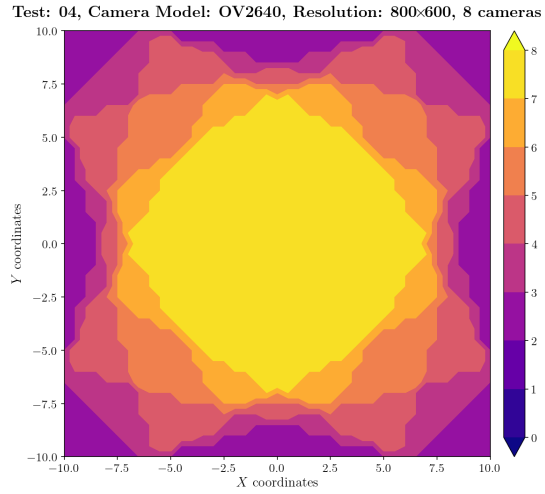


Figure A.560: Camera coverage for test 04 at level $z = 6.500$ m and resolution 800×600.

Estimation Error distribution for $z = 7.0$ m level

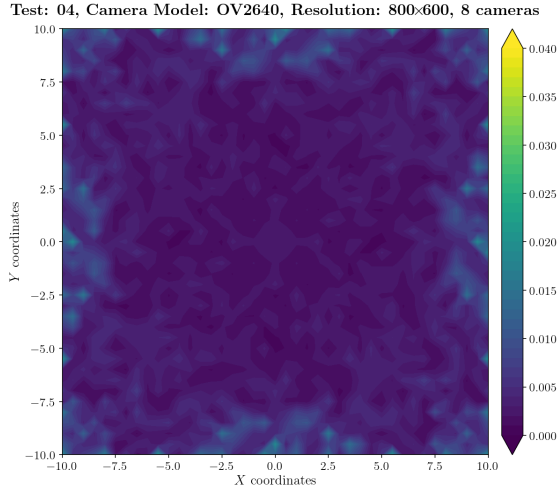


Figure A.561: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 7.000$ m.

Camera Coverage Spatial Distribution for $z = 7.0$ m level

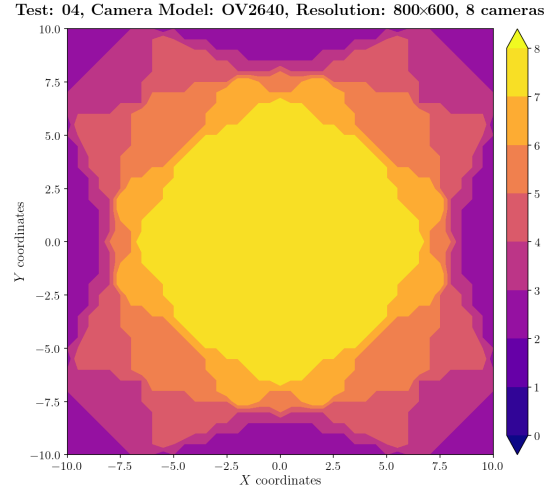


Figure A.562: Camera coverage for test 04 at level $z = 7.000$ m and resolution 800×600.

Estimation Error distribution for $z = 7.5$ m level

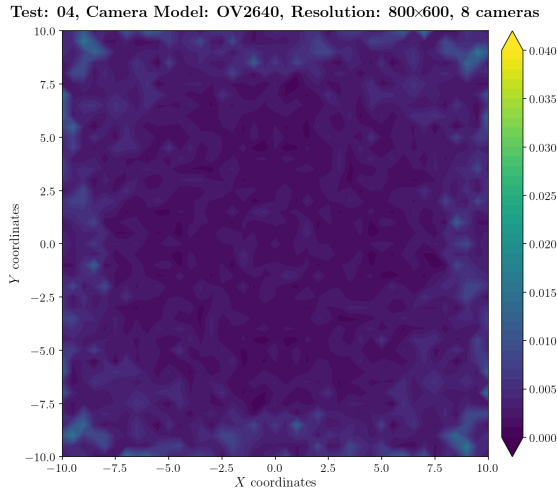


Figure A.563: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 7.500$ m.

Camera Coverage Spatial Distribution for $z = 7.5$ m level

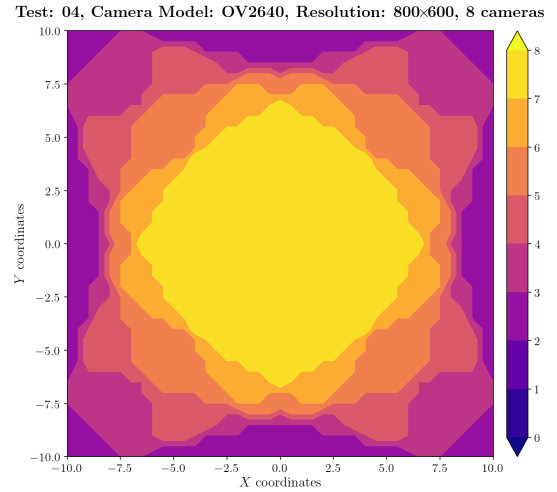
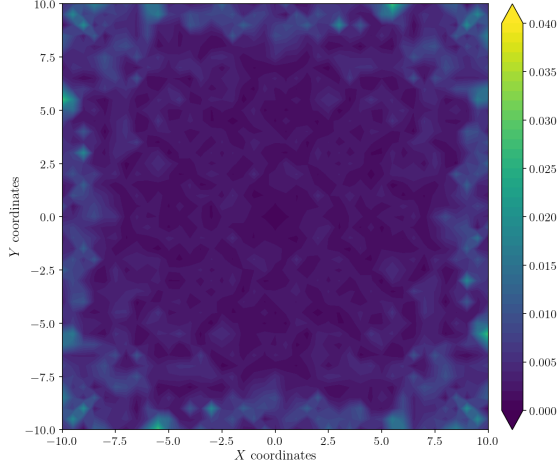


Figure A.564: Camera coverage for test 04 at level $z = 7.500$ m and resolution 800×600.

Estimation Error distribution for $z = 8.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 800×600, 8 cameras

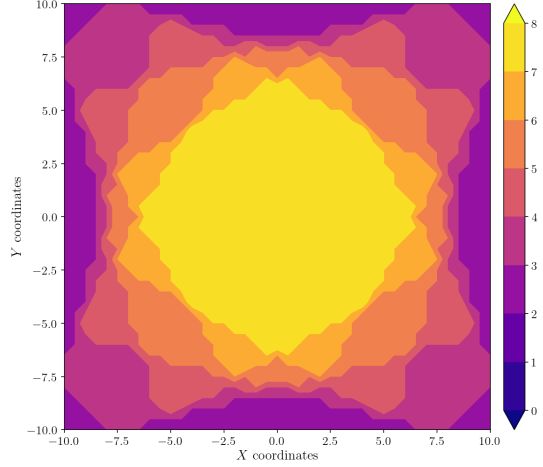
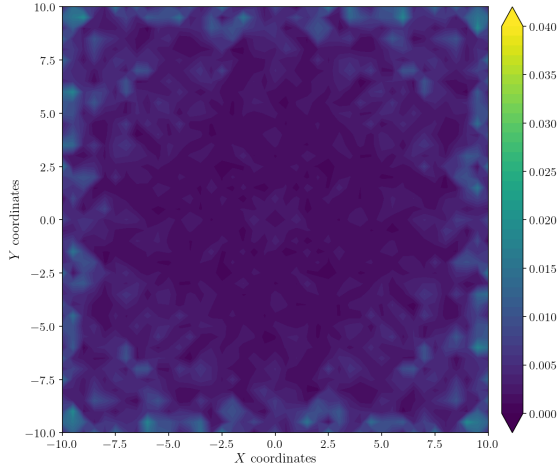


Figure A.565: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 8.000$ m.

Figure A.566: Camera coverage for test 04 at level $z = 8.000$ m and resolution 800×600.

Estimation Error distribution for $z = 8.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 800×600, 8 cameras

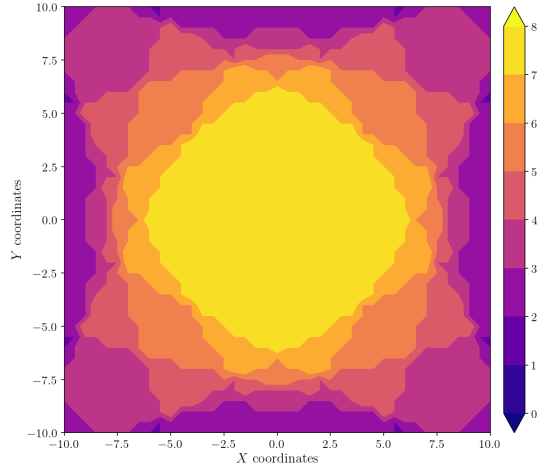


Figure A.567: Estimation error contour map for test 04 with 8 cameras using resolution of 800×600 pixels, at level $z = 8.500$ m.

Figure A.568: Camera coverage for test 04 at level $z = 8.500$ m and resolution 800×600.

Error maps for resolution 1024×768

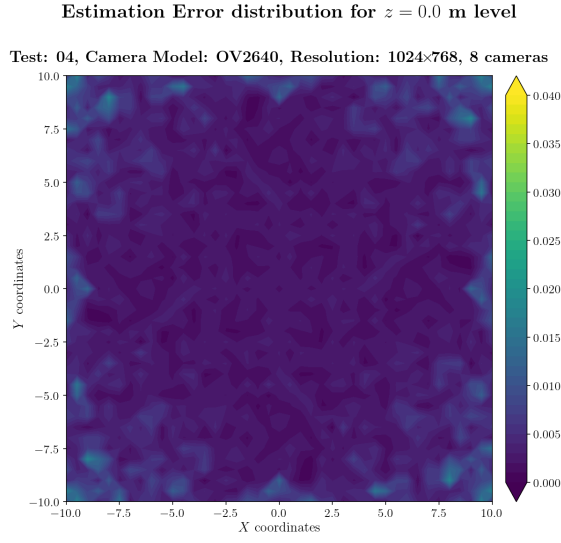


Figure A.569: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.000$ m.

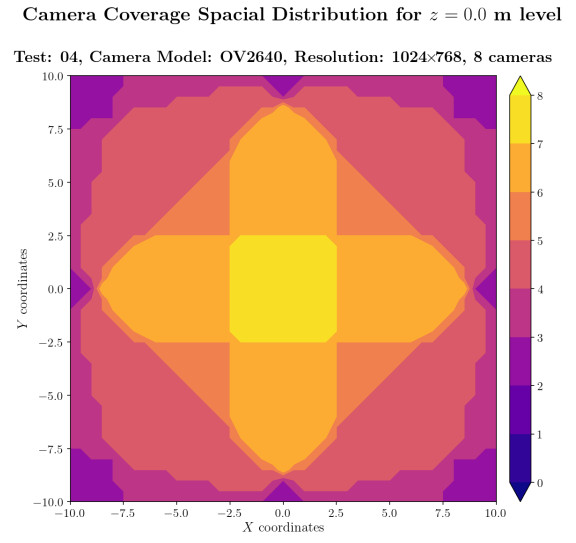


Figure A.570: Camera coverage for test 04 at level $z = 0.000$ m and resolution 1024×768.

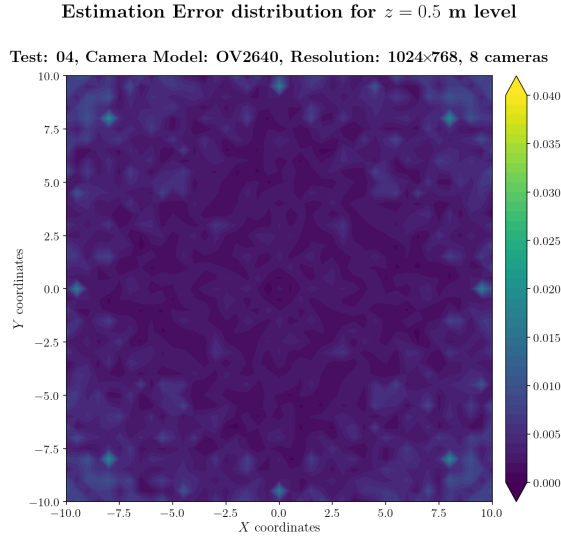


Figure A.571: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.500$ m.

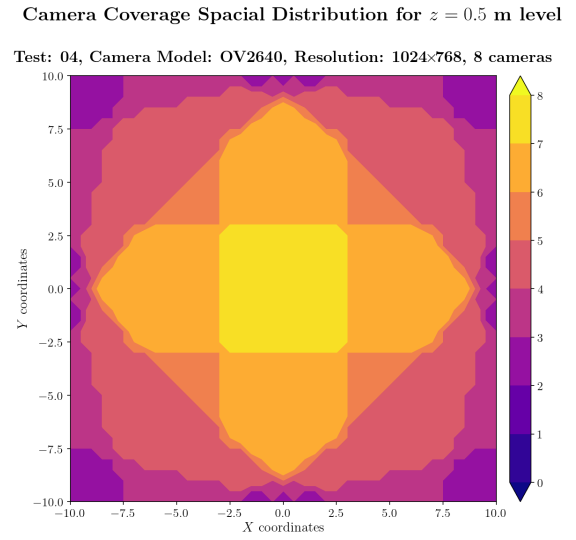
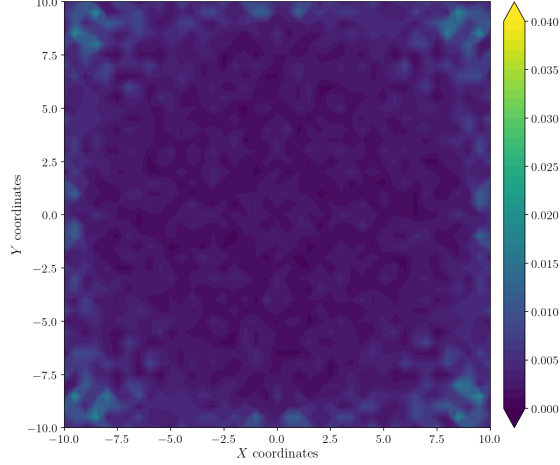


Figure A.572: Camera coverage for test 04 at level $z = 0.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

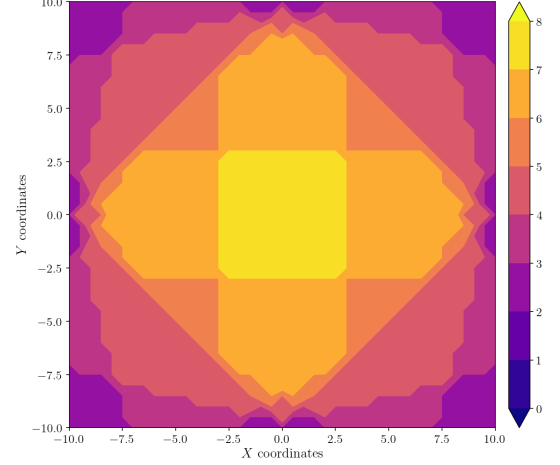
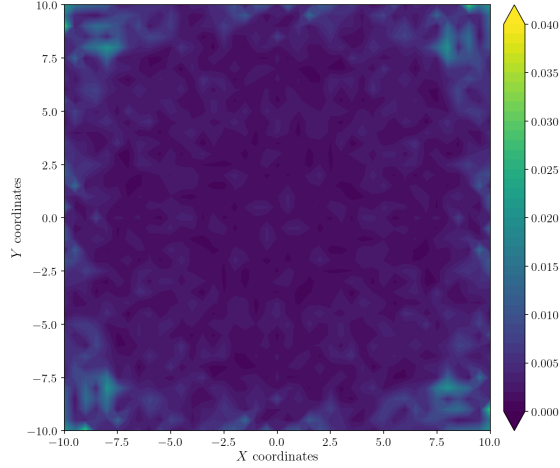


Figure A.573: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.000$ m.

Figure A.574: Camera coverage for test 04 at level $z = 1.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 1.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

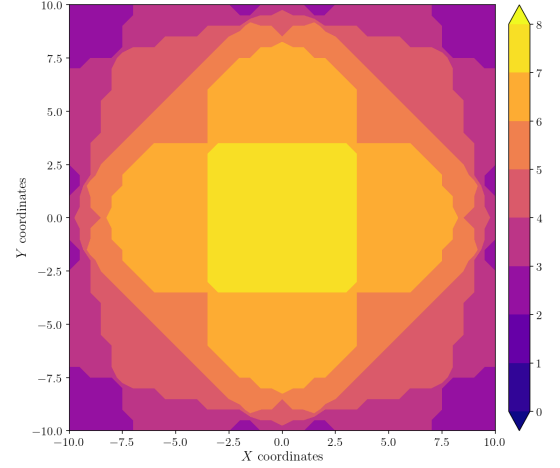


Figure A.575: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.500$ m.

Figure A.576: Camera coverage for test 04 at level $z = 1.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 2.0$ m level

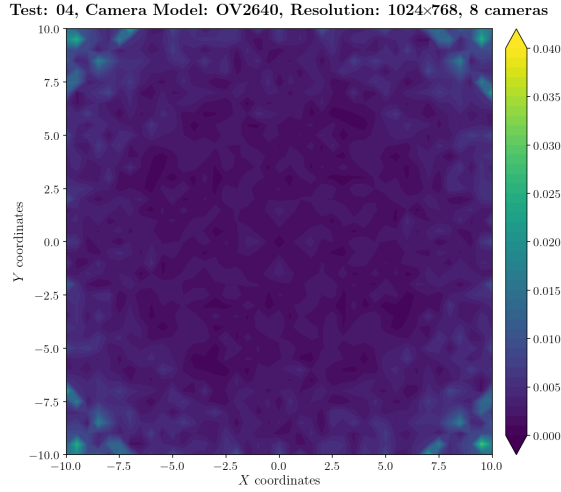


Figure A.577: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.000$ m.

Camera Coverage Spatial Distribution for $z = 2.0$ m level

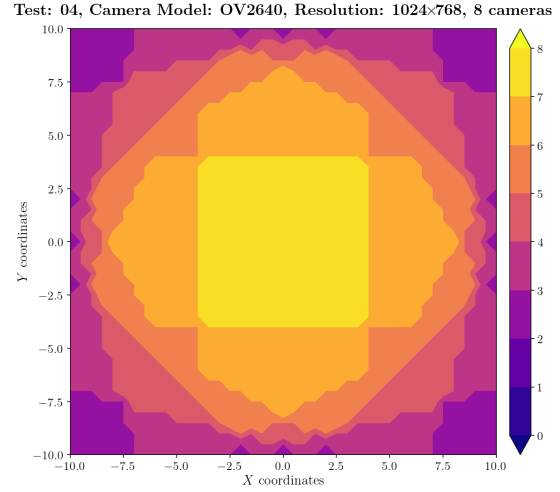


Figure A.578: Camera coverage for test 04 at level $z = 2.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 2.5$ m level

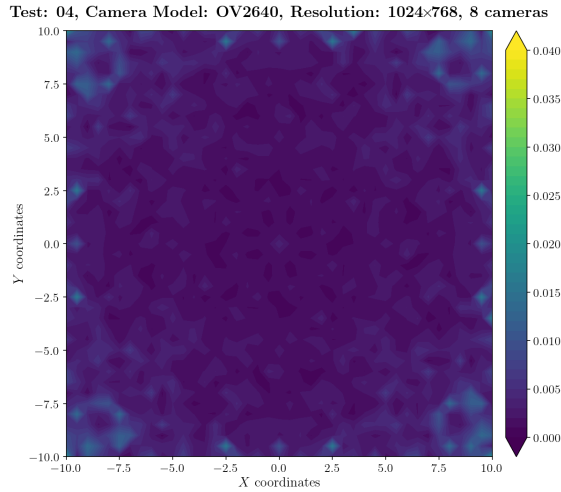


Figure A.579: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.500$ m.

Camera Coverage Spatial Distribution for $z = 2.5$ m level

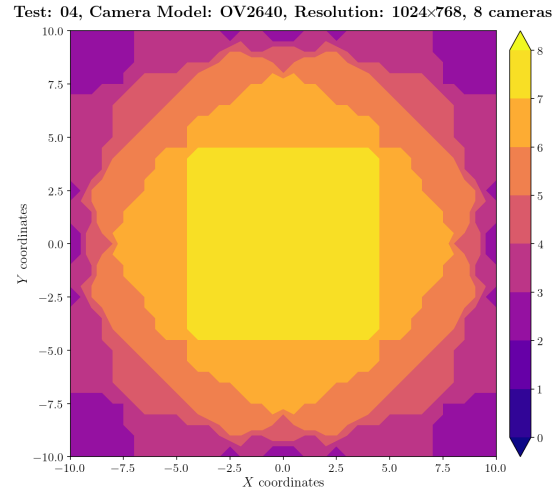
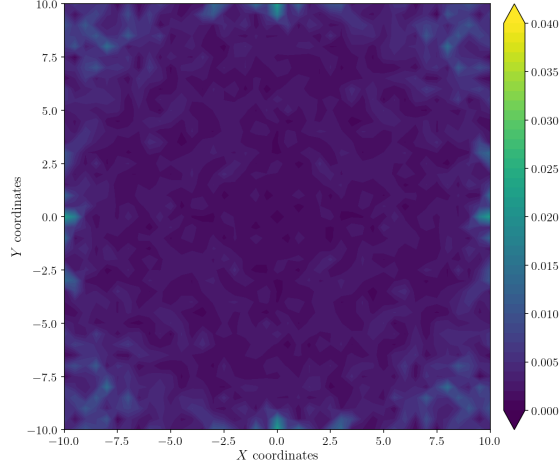


Figure A.580: Camera coverage for test 04 at level $z = 2.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

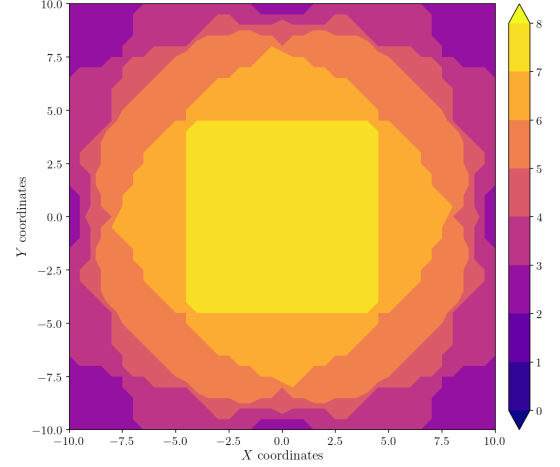
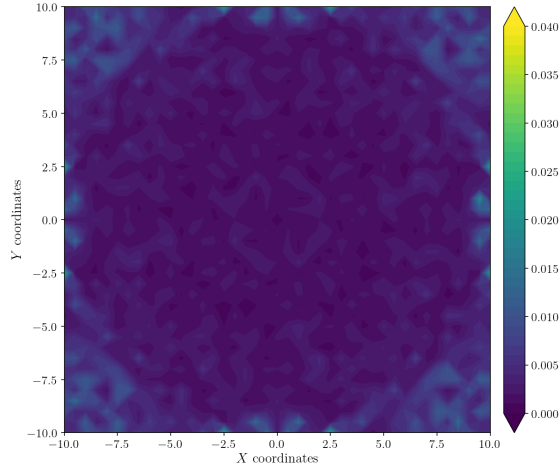


Figure A.581: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.000$ m.

Figure A.582: Camera coverage for test 04 at level $z = 3.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

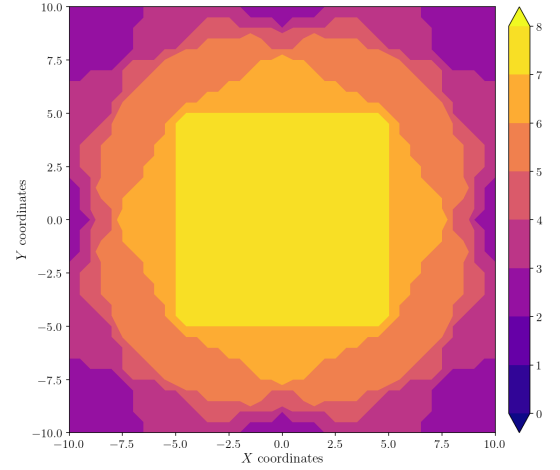


Figure A.583: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.500$ m.

Figure A.584: Camera coverage for test 04 at level $z = 3.500$ m and resolution 1024×768.

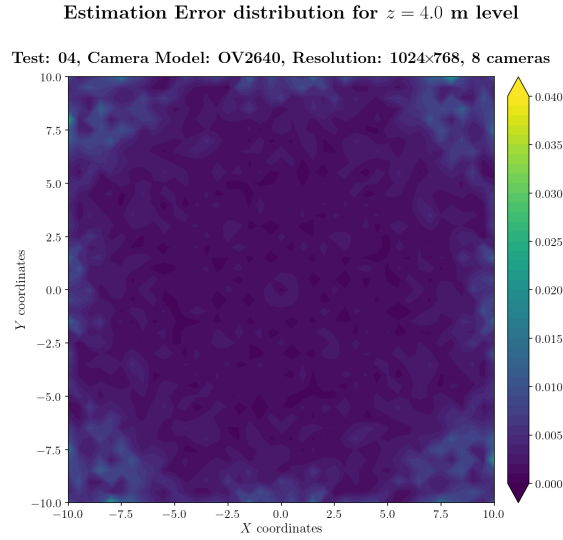


Figure A.585: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.000$ m.

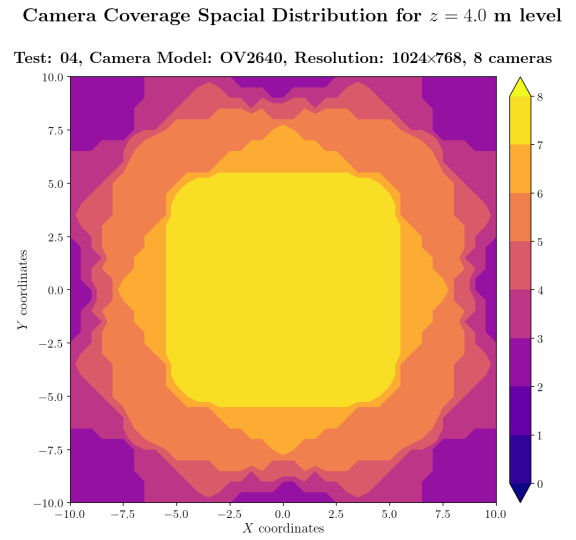


Figure A.586: Camera coverage for test 04 at level $z = 4.000$ m and resolution 1024×768.

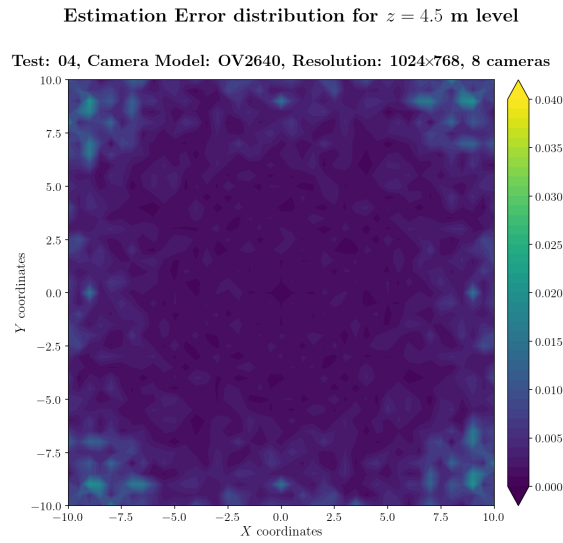


Figure A.587: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.500$ m.

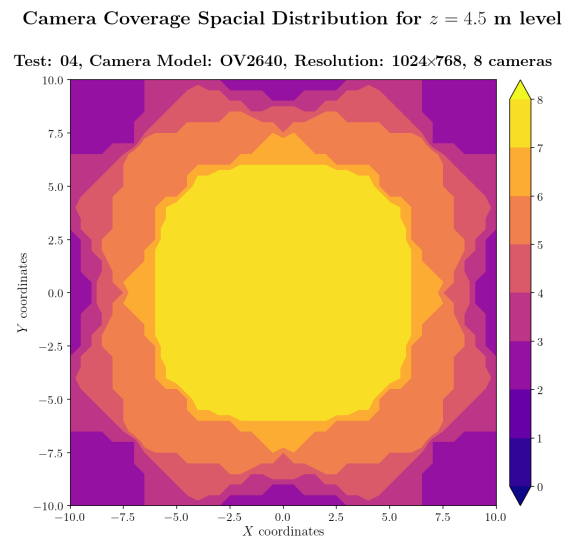
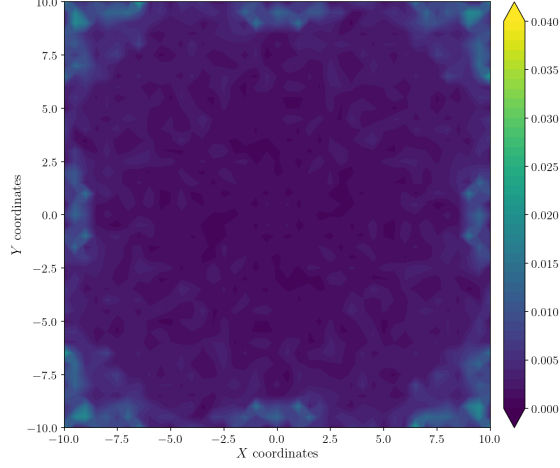


Figure A.588: Camera coverage for test 04 at level $z = 4.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 5.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

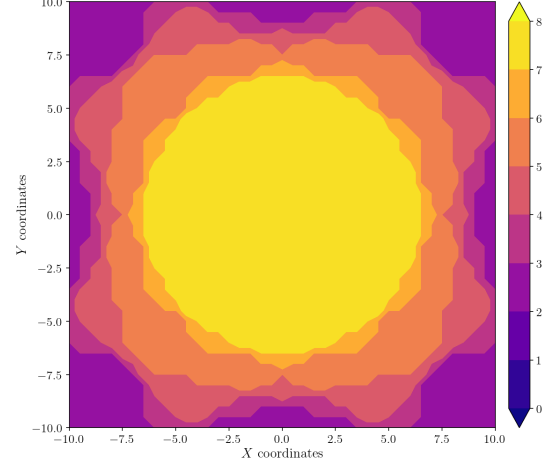
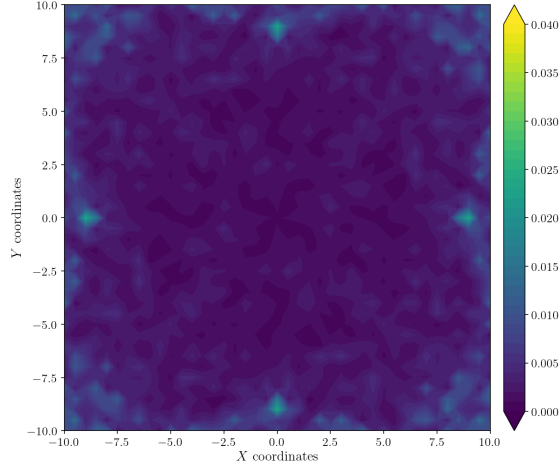


Figure A.589: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 5.000$ m.

Figure A.590: Camera coverage for test 04 at level $z = 5.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 5.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

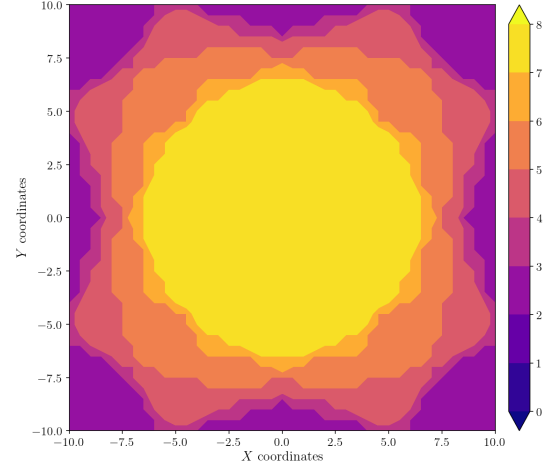


Figure A.591: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 5.500$ m.

Figure A.592: Camera coverage for test 04 at level $z = 5.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 6.0$ m level

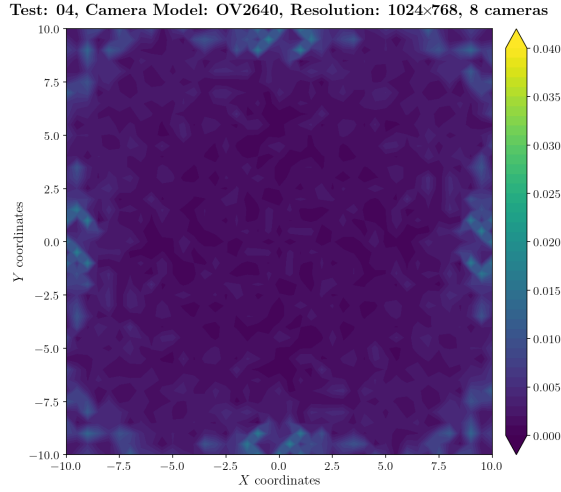


Figure A.593: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 6.000$ m.

Camera Coverage Spacial Distribution for $z = 6.0$ m level

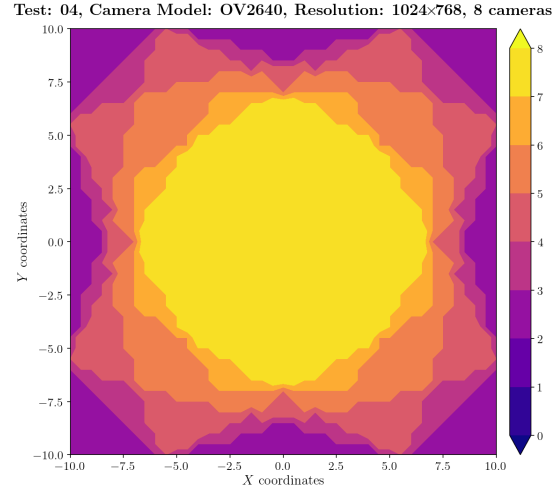


Figure A.594: Camera coverage for test 04 at level $z = 6.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 6.5$ m level

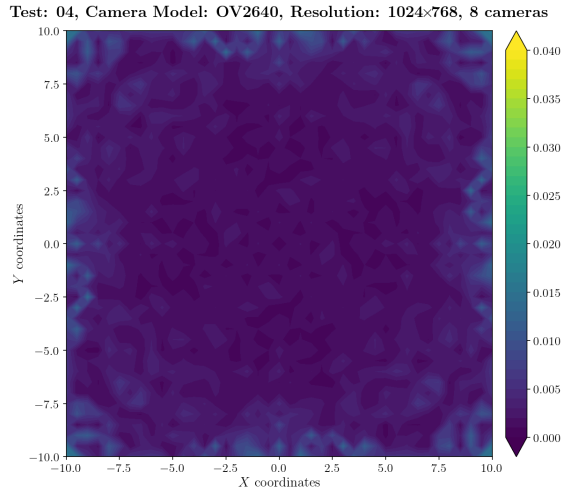


Figure A.595: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 6.500$ m.

Camera Coverage Spacial Distribution for $z = 6.5$ m level

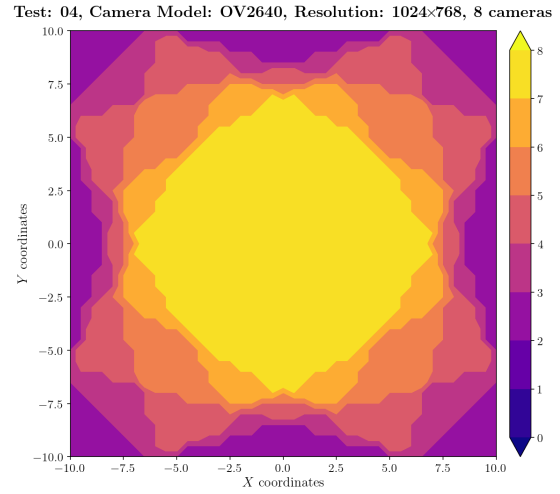
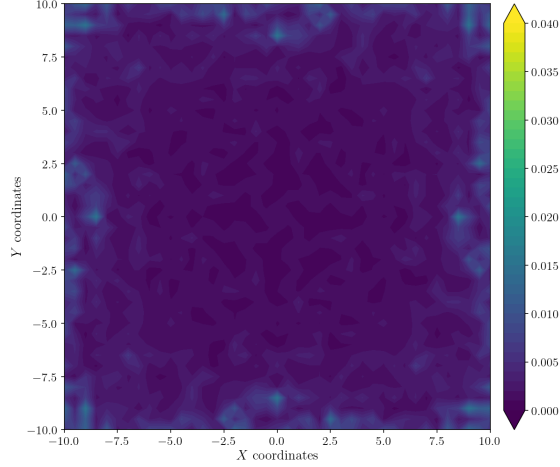


Figure A.596: Camera coverage for test 04 at level $z = 6.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 7.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

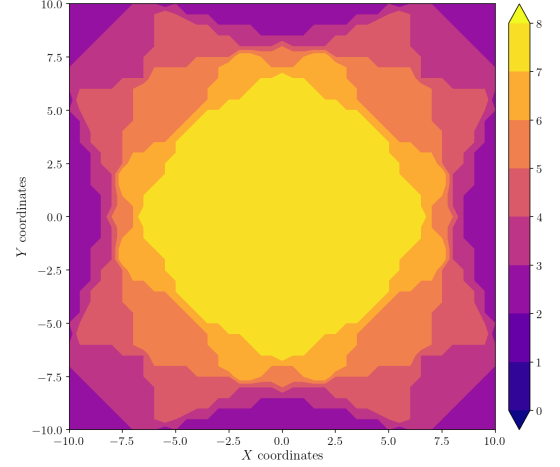
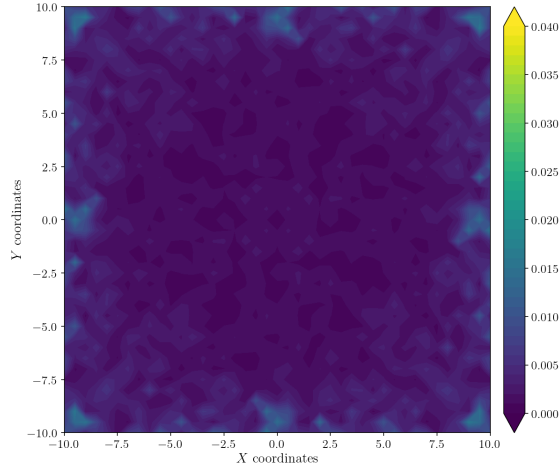


Figure A.597: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 7.000$ m.

Figure A.598: Camera coverage for test 04 at level $z = 7.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 7.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

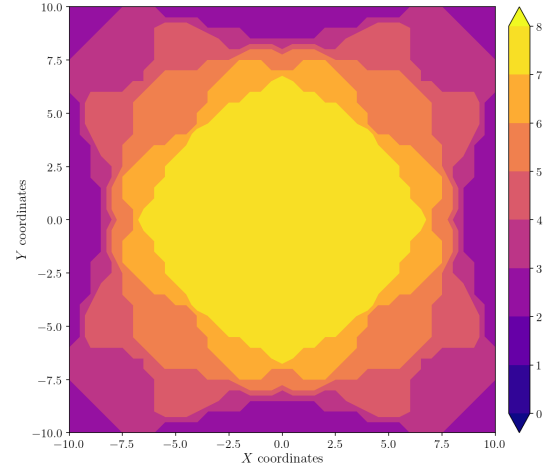


Figure A.599: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 7.500$ m.

Figure A.600: Camera coverage for test 04 at level $z = 7.500$ m and resolution 1024×768.

Estimation Error distribution for $z = 8.0$ m level

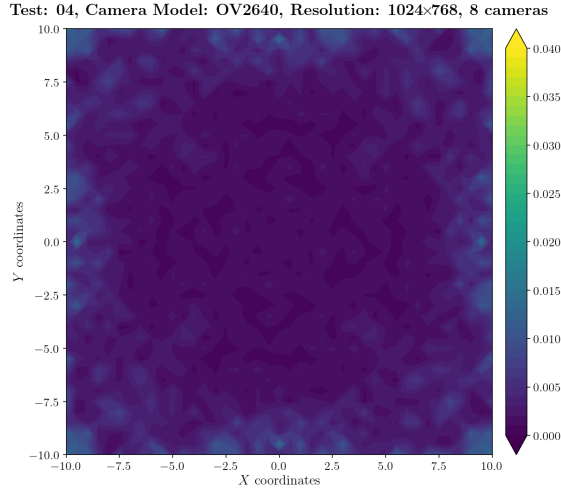


Figure A.601: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 8.000$ m.

Camera Coverage Spacial Distribution for $z = 8.0$ m level

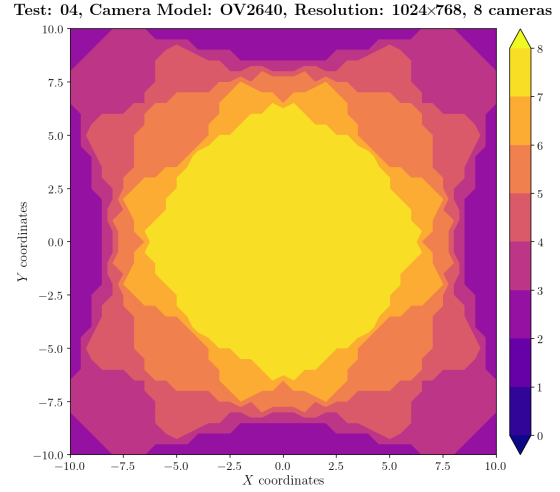


Figure A.602: Camera coverage for test 04 at level $z = 8.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 8.5$ m level

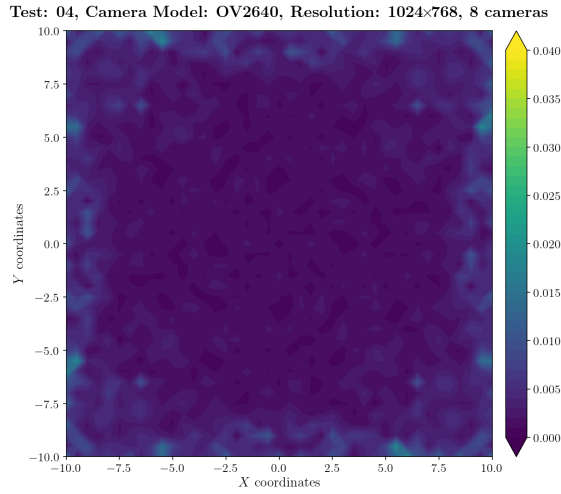


Figure A.603: Estimation error contour map for test 04 with 8 cameras using resolution of 1024×768 pixels, at level $z = 8.500$ m.

Camera Coverage Spacial Distribution for $z = 8.5$ m level

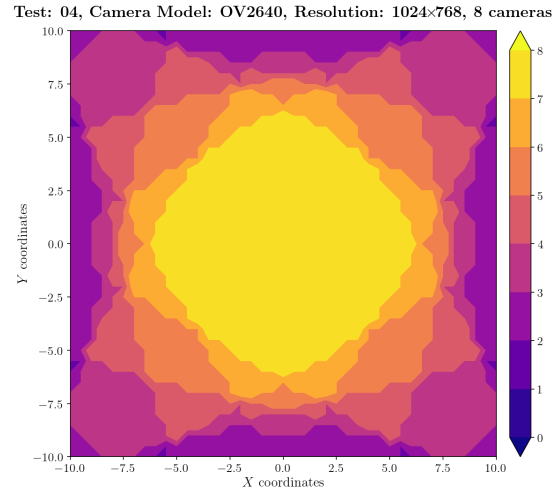


Figure A.604: Camera coverage for test 04 at level $z = 8.500$ m and resolution 1024×768.

Error maps for resolution 1600×1200

Estimation Error distribution for $z = 0.0$ m level

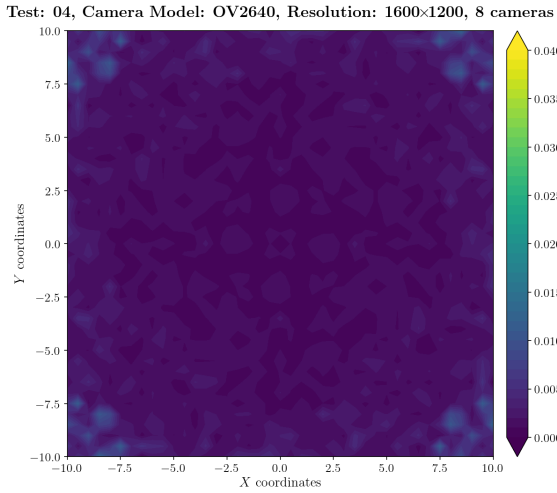


Figure A.605: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

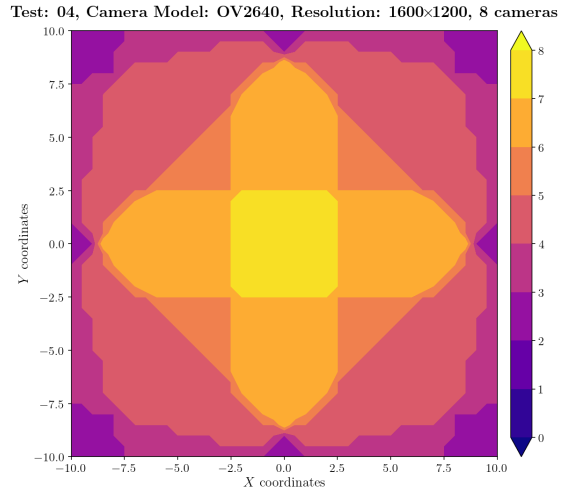


Figure A.606: Camera coverage for test 04 at level $z = 0.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 0.5$ m level

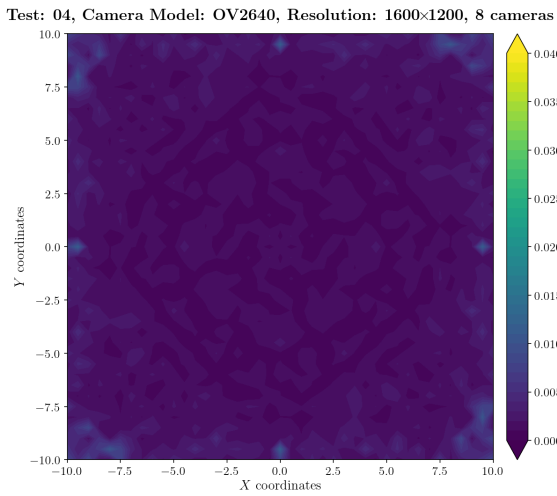


Figure A.607: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.500$ m.

Camera Coverage Spacial Distribution for $z = 0.5$ m level

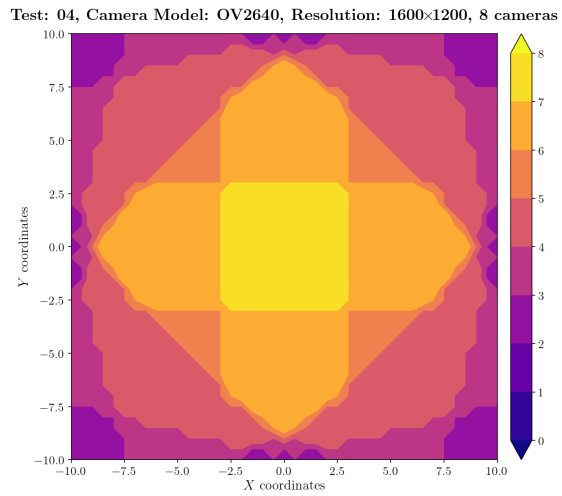


Figure A.608: Camera coverage for test 04 at level $z = 0.500$ m and resolution 1600×1200.

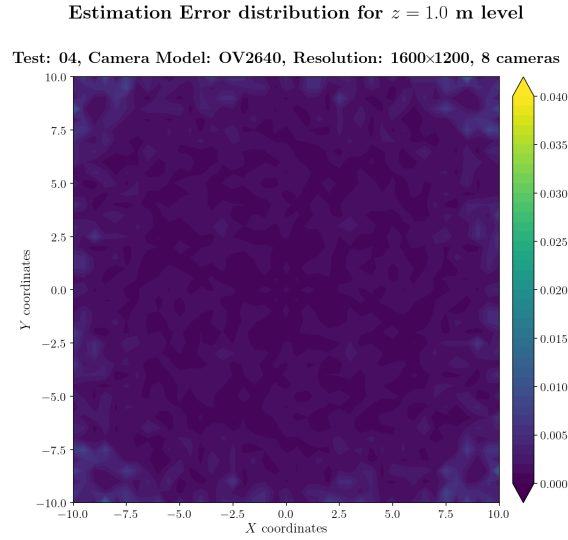


Figure A.609: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.000$ m.

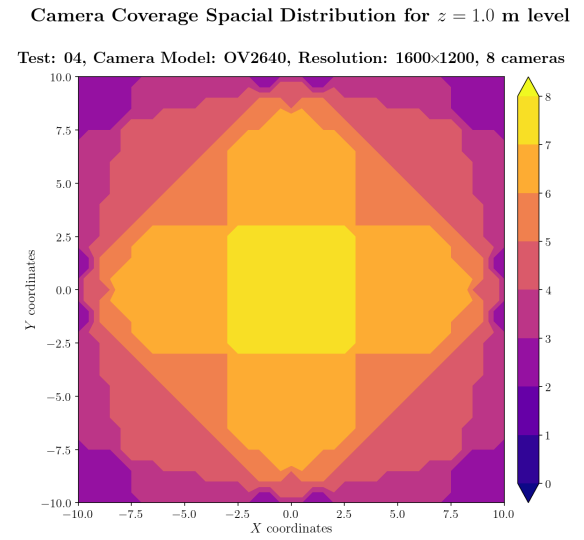


Figure A.610: Camera coverage for test 04 at level $z = 1.000$ m and resolution 1600×1200.

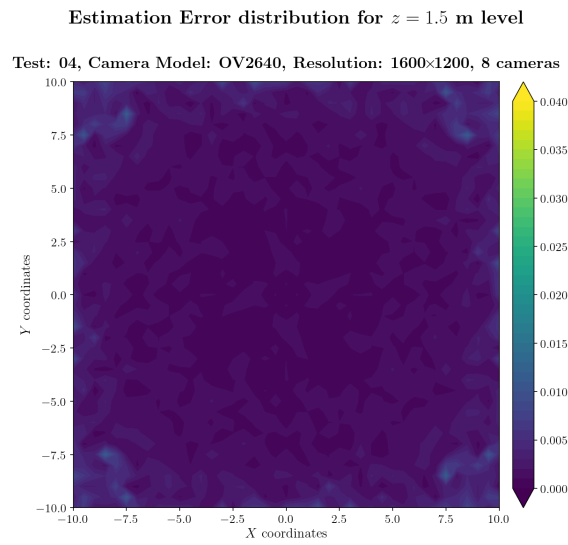


Figure A.611: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.500$ m.

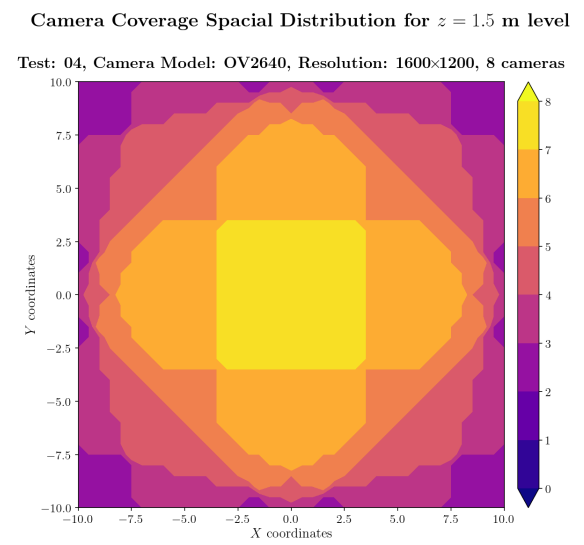
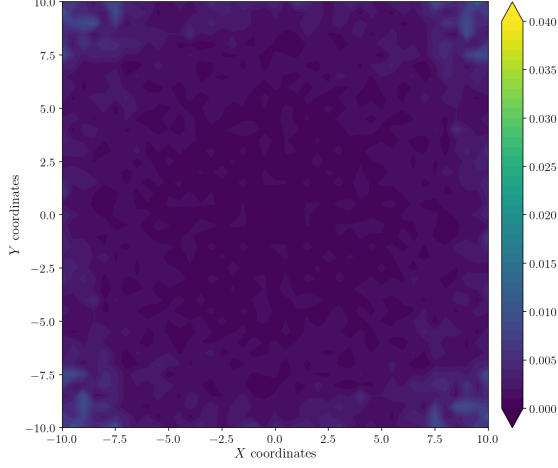


Figure A.612: Camera coverage for test 04 at level $z = 1.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 2.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

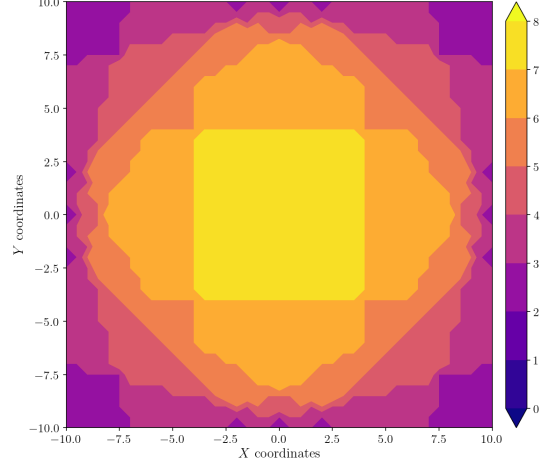
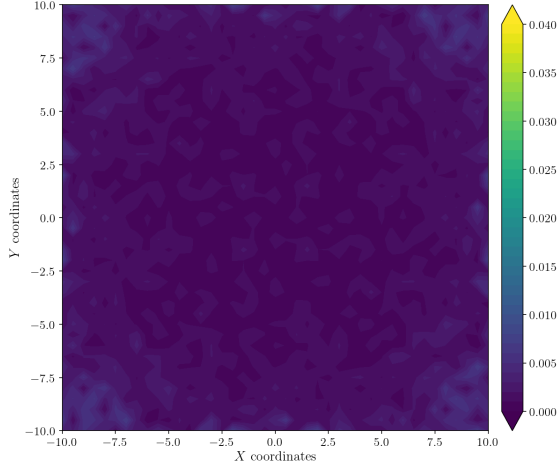


Figure A.613: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.000$ m.

Figure A.614: Camera coverage for test 04 at level $z = 2.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 2.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

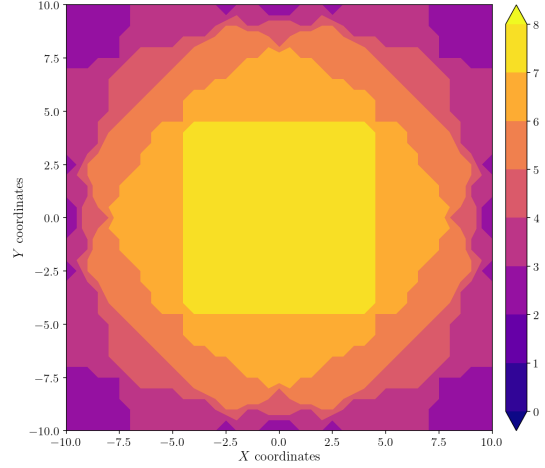


Figure A.615: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.500$ m.

Figure A.616: Camera coverage for test 04 at level $z = 2.500$ m and resolution 1600×1200.

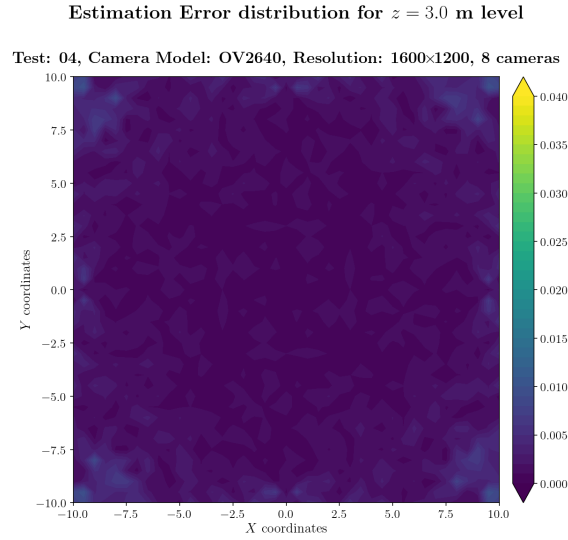


Figure A.617: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.000$ m.

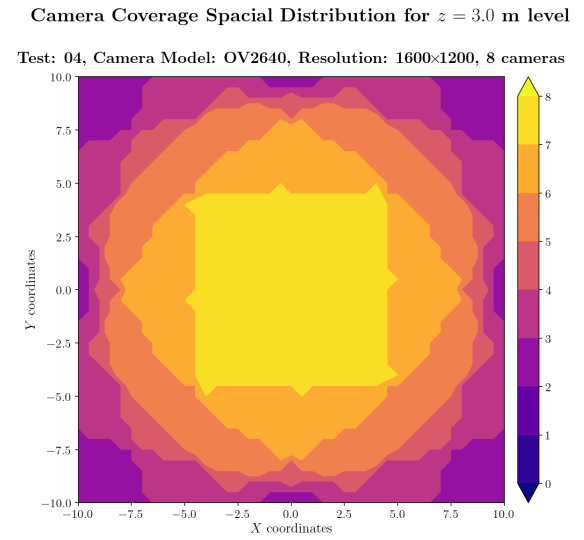


Figure A.618: Camera coverage for test 04 at level $z = 3.000$ m and resolution 1600×1200.

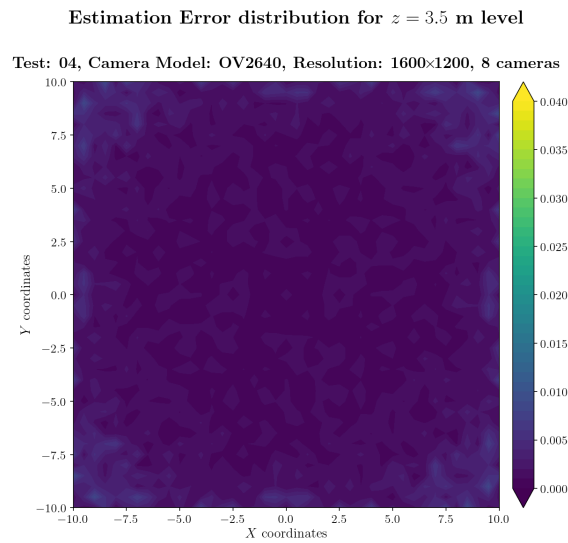


Figure A.619: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.500$ m.

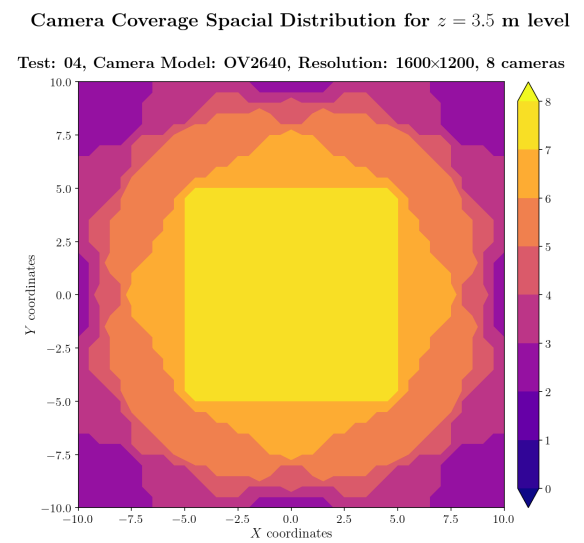
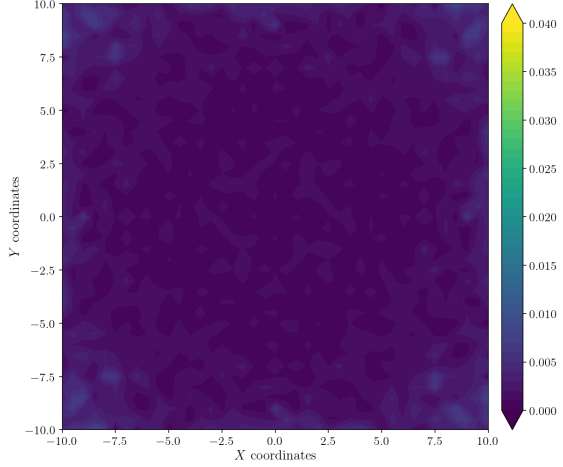


Figure A.620: Camera coverage for test 04 at level $z = 3.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 4.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

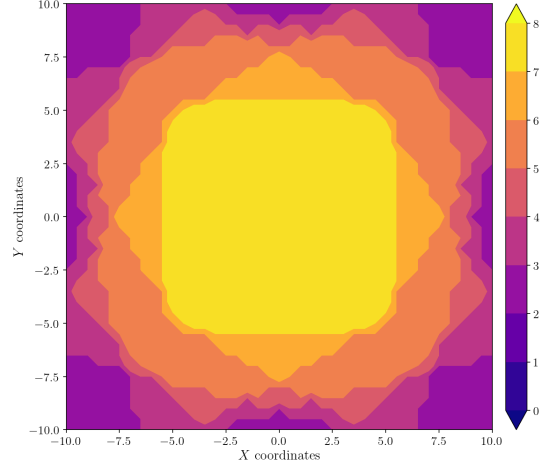
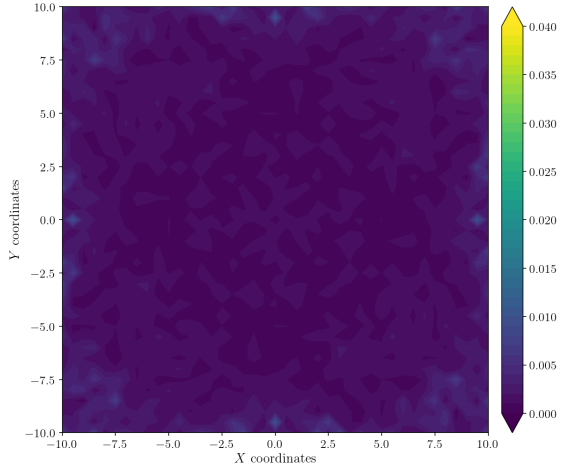


Figure A.621: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.000$ m.

Figure A.622: Camera coverage for test 04 at level $z = 4.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 4.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

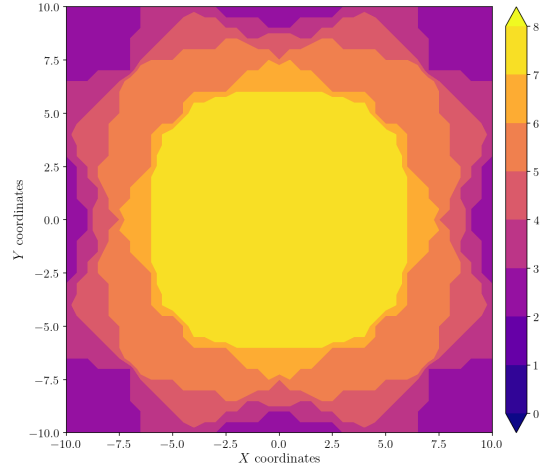


Figure A.623: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.500$ m.

Figure A.624: Camera coverage for test 04 at level $z = 4.500$ m and resolution 1600×1200.

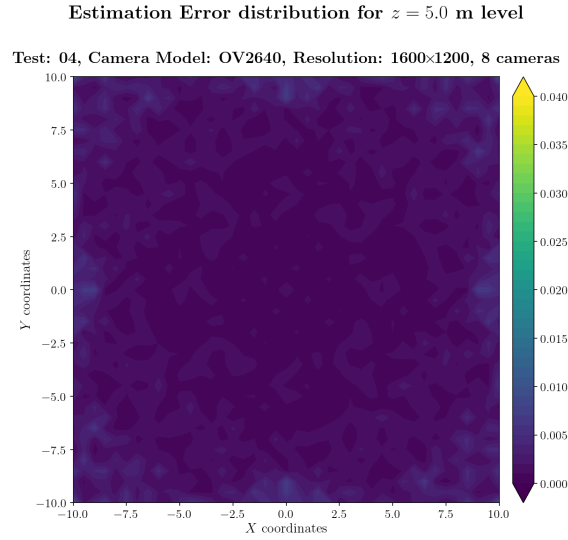


Figure A.625: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 5.000$ m.

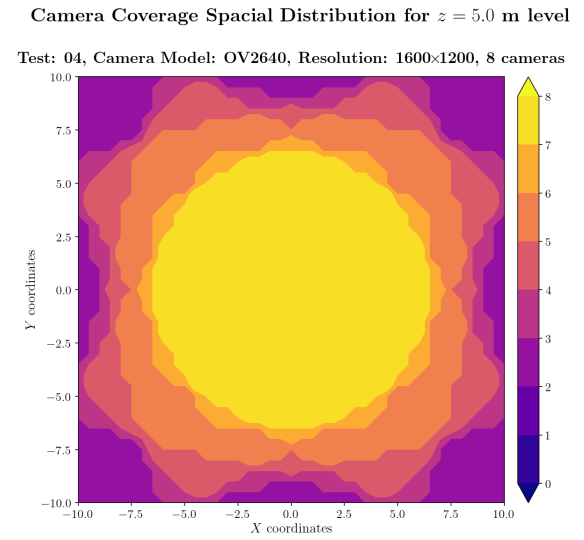


Figure A.626: Camera coverage for test 04 at level $z = 5.000$ m and resolution 1600×1200.

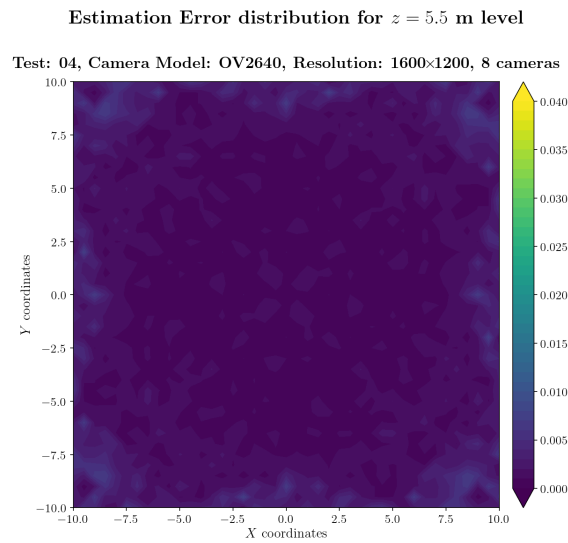


Figure A.627: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 5.500$ m.

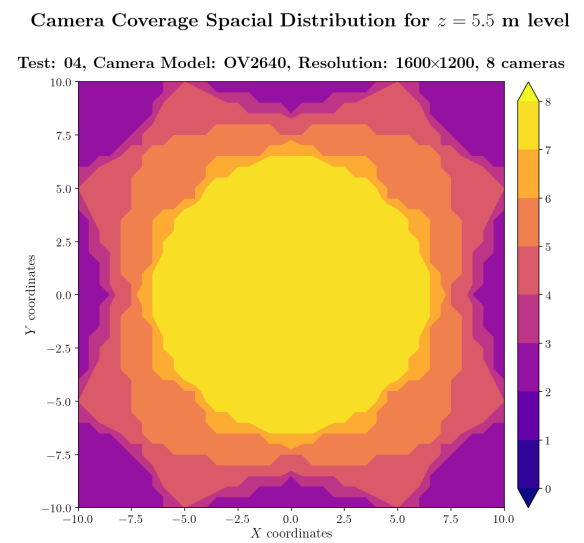
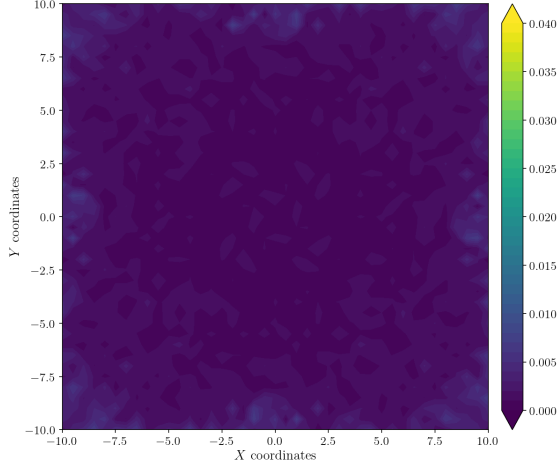


Figure A.628: Camera coverage for test 04 at level $z = 5.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 6.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

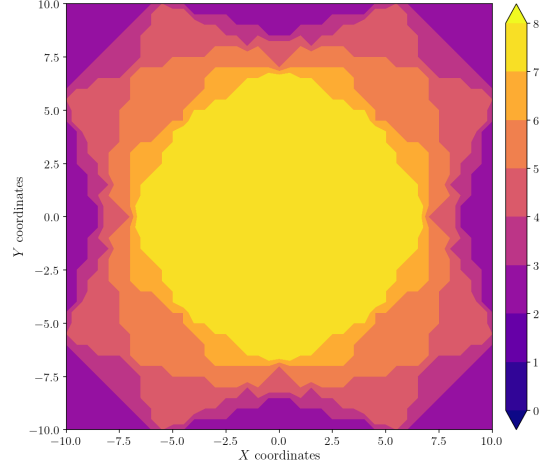
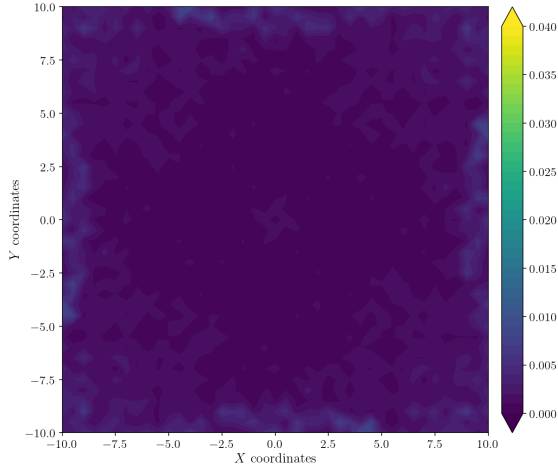


Figure A.629: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 6.000$ m.

Figure A.630: Camera coverage for test 04 at level $z = 6.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 6.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

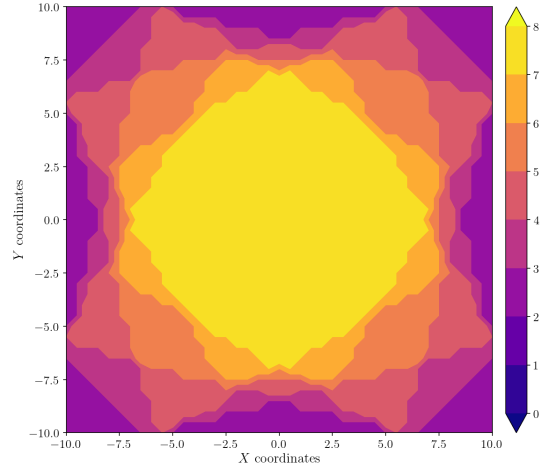


Figure A.631: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 6.500$ m.

Figure A.632: Camera coverage for test 04 at level $z = 6.500$ m and resolution 1600×1200.

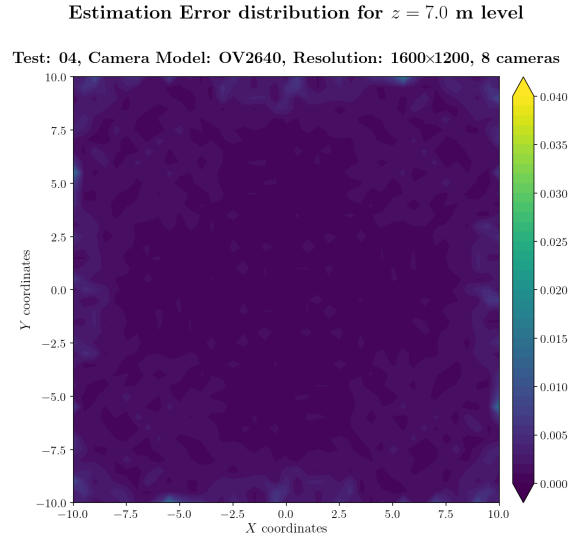


Figure A.633: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 7.000$ m.

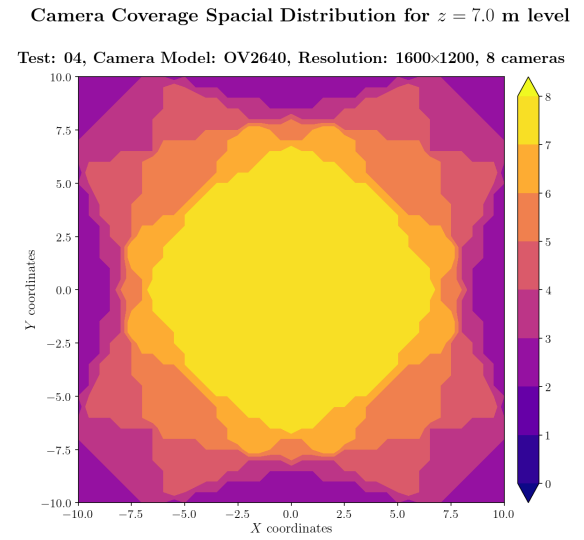


Figure A.634: Camera coverage for test 04 at level $z = 7.000$ m and resolution 1600×1200.

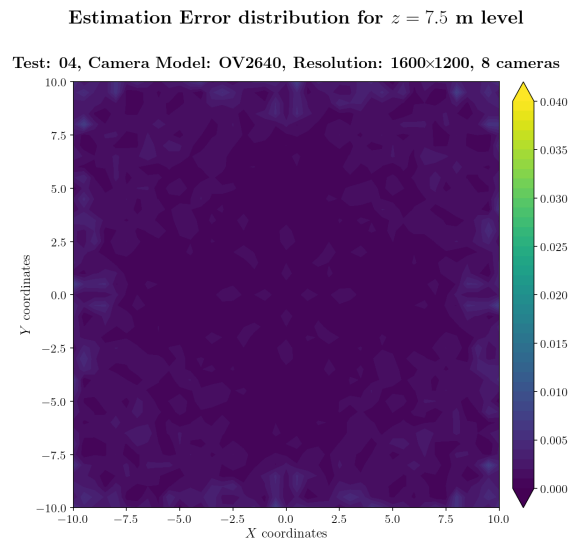


Figure A.635: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 7.500$ m.

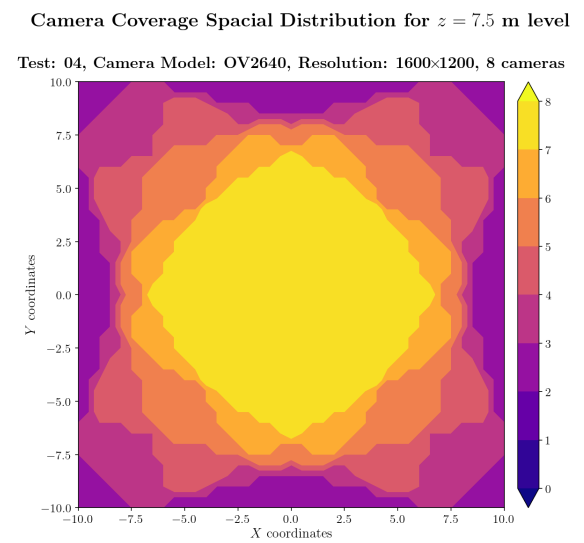
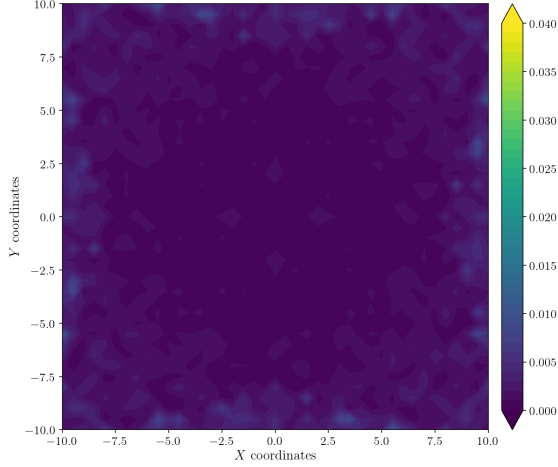


Figure A.636: Camera coverage for test 04 at level $z = 7.500$ m and resolution 1600×1200.

Estimation Error distribution for $z = 8.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.0$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

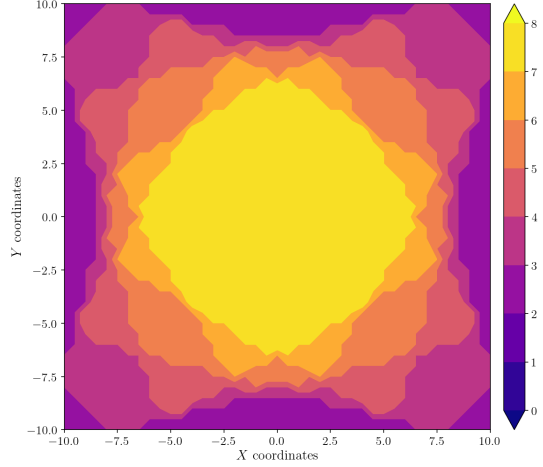
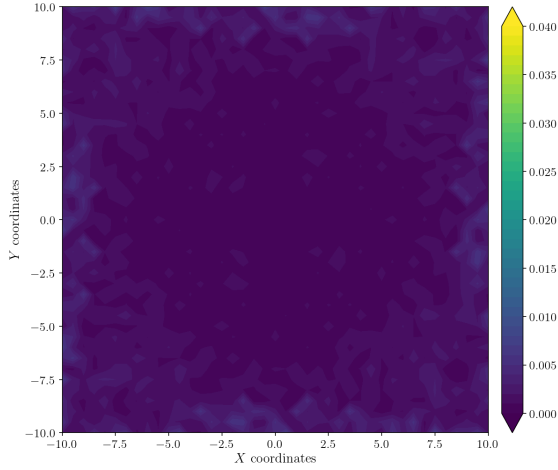


Figure A.637: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 8.000$ m.

Figure A.638: Camera coverage for test 04 at level $z = 8.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 8.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.5$ m level

Test: 04, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

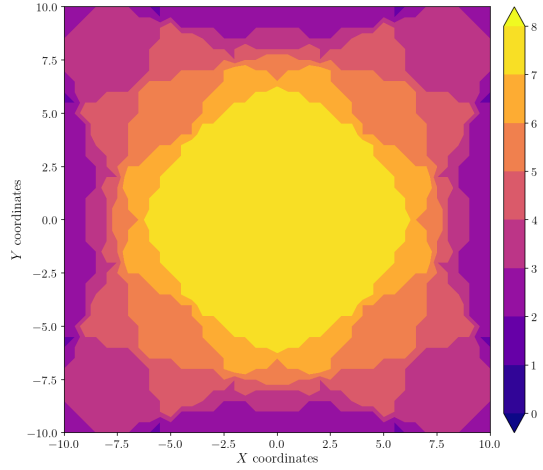


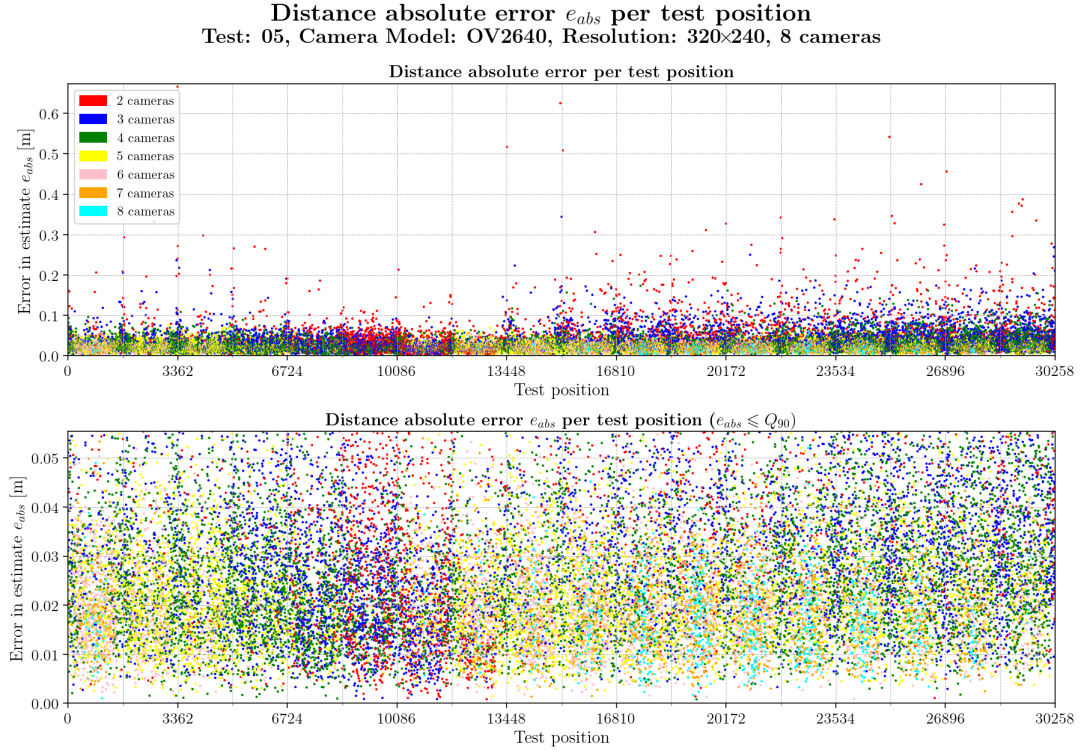
Figure A.639: Estimation error contour map for test 04 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 8.500$ m.

Figure A.640: Camera coverage for test 04 at level $z = 8.500$ m and resolution 1600×1200.

A.5 Test 05

Absolute Error per iteration and resolution

Resolution: 320×240



Error per iteration for resolution 320×240 and test05

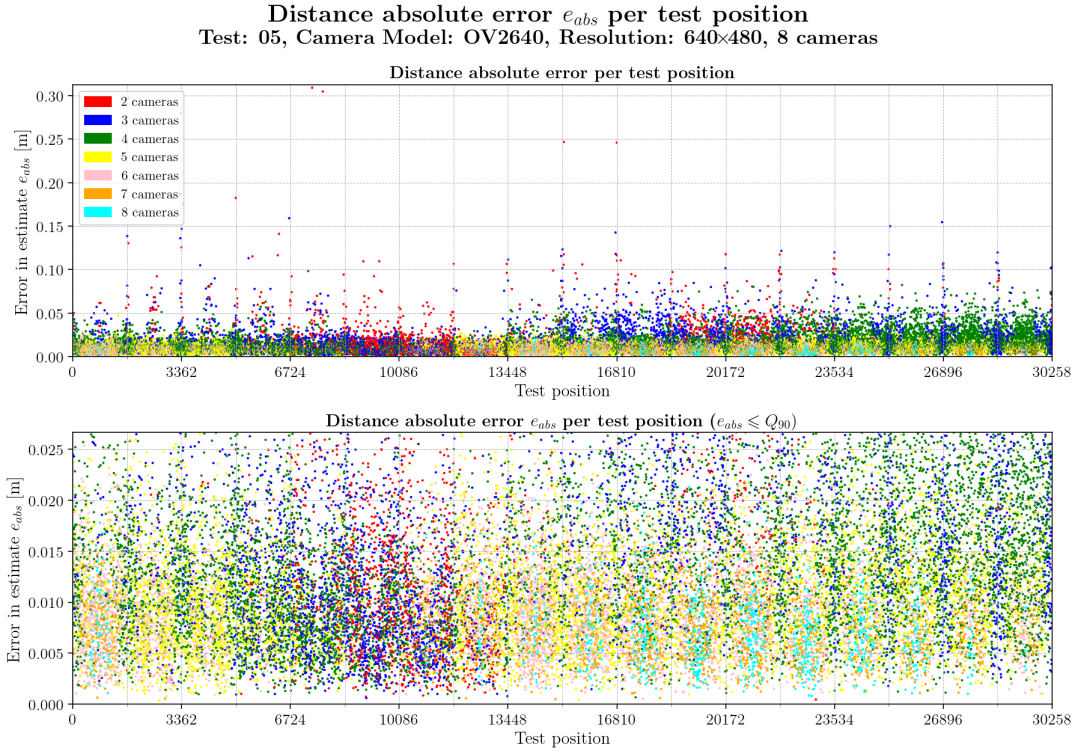
Table A.41: Statistics per number of cameras observing the beacon for test 05 with resolution 320×240.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	3224	6007	7347	5349	3640	2018	1047
mean	0.0549	0.0387	0.0290	0.0228	0.0201	0.0188	0.0173
std	0.0541	0.0277	0.0178	0.0111	0.0093	0.0087	0.0076
min	0.0008	0.0008	0.0008	0.0014	0.0009	0.0013	0.0010
25%	0.0222	0.0202	0.0168	0.0149	0.0135	0.0122	0.0117
50%	0.0400	0.0320	0.0251	0.0212	0.0186	0.0177	0.0163
75%	0.0668	0.0485	0.0364	0.0289	0.0256	0.0237	0.0218
max	0.6663	0.3438	0.1727	0.0838	0.0746	0.0578	0.0601

Table A.42: Global statistics and root mean square error for test 05 with resolution 320×240.

	Abs. error	Est. error
count	28632	28632
mean	0.0305	0.0528
std	0.0271	0.0253
min	0.0008	0.0000
25%	0.0158	0.0349
50%	0.0238	0.0523
75%	0.0360	0.0698
90%	0.0554	0.0855
max	0.6663	0.1880
RMSE	0.0408	

Resolution: 640×480



Error per iteration for resolution 640×480 and test05

Table A.43: Statistics per number of cameras observing the beacon for test 05 with resolution 640×480.

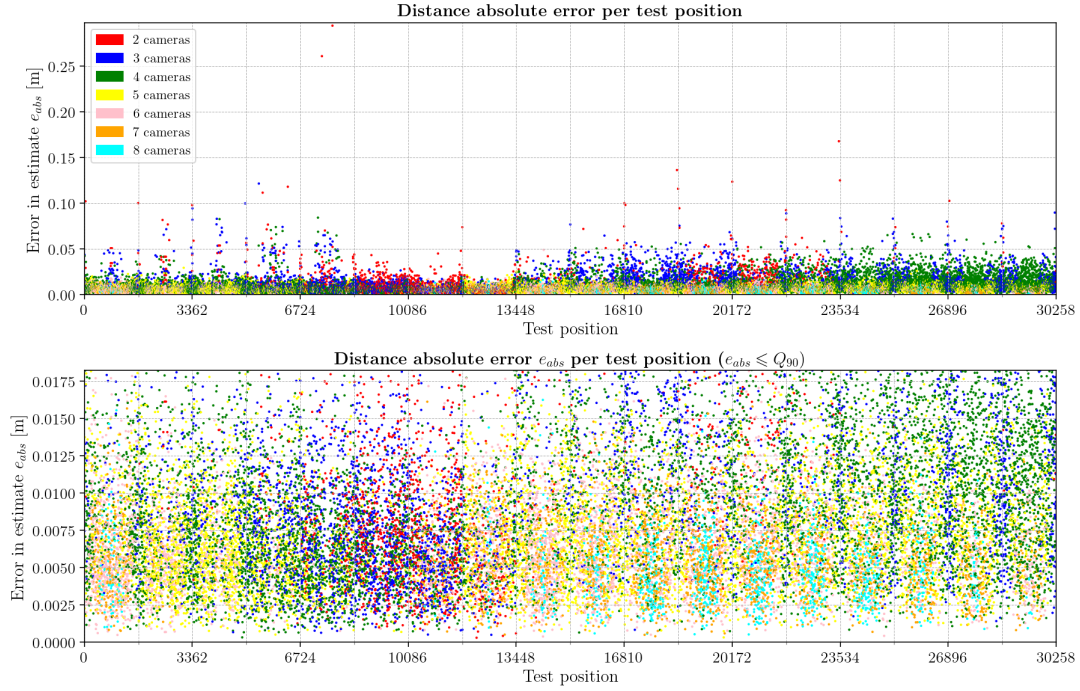
	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	1964	4616	8793	5852	4179	2439	1370
mean	0.0232	0.0197	0.0165	0.0104	0.0090	0.0081	0.0073
std	0.0261	0.0165	0.0111	0.0052	0.0042	0.0037	0.0034
min	0.0005	0.0006	0.0006	0.0004	0.0005	0.0005	0.0008
25%	0.0084	0.0088	0.0088	0.0066	0.0060	0.0054	0.0049
50%	0.0151	0.0152	0.0136	0.0096	0.0086	0.0077	0.0069
75%	0.0292	0.0250	0.0208	0.0133	0.0115	0.0103	0.0093
max	0.3092	0.1592	0.0984	0.0473	0.0498	0.0231	0.0213

Table A.44: Global statistics and root mean square error for test 05 with resolution 640×480.

	Abs. error	Est. error
count	29213	29213
mean	0.0140	0.0263
std	0.0126	0.0122
min	0.0004	0.0000
25%	0.0069	0.0177
50%	0.0107	0.0261
75%	0.0166	0.0345
90%	0.0266	0.0422
max	0.3092	0.0902
RMSE	0.0189	

Resolution: 800×600

Distance absolute error e_{abs} per test position
Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Error per iteration for resolution 800×600 and test05

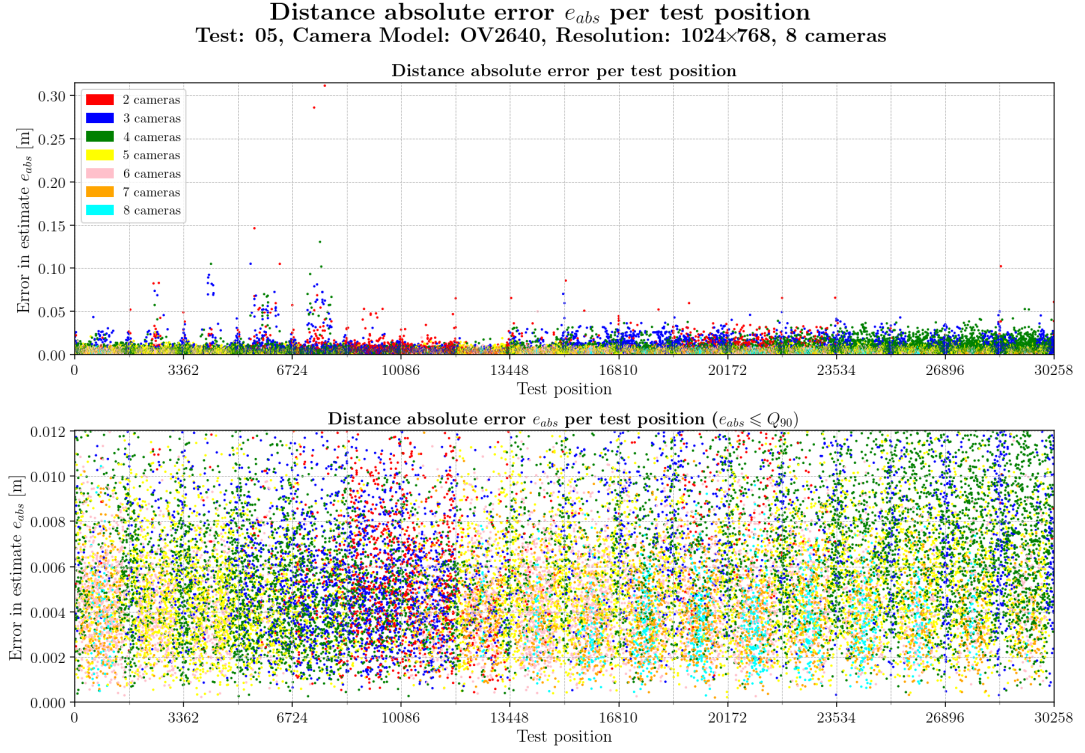
Table A.45: Statistics per number of cameras observing the beacon for test 05 with resolution 800×600.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	1912	4619	8815	5868	4179	2452	1376
mean	0.0150	0.0137	0.0115	0.0073	0.0064	0.0055	0.0048
std	0.0173	0.0116	0.0085	0.0037	0.0031	0.0026	0.0023
min	0.0003	0.0005	0.0003	0.0003	0.0004	0.0004	0.0005
25%	0.0058	0.0063	0.0059	0.0046	0.0041	0.0037	0.0031
50%	0.0098	0.0101	0.0093	0.0067	0.0059	0.0052	0.0044
75%	0.0184	0.0172	0.0143	0.0094	0.0082	0.0070	0.0062
max	0.2940	0.1214	0.0841	0.0293	0.0489	0.0161	0.0162

Table A.46: Global statistics and root mean square error for test 05 with resolution 800×600.

	Abs. error	Est. error
count	29221	29221
mean	0.0097	0.0185
std	0.0088	0.0089
min	0.0003	0.0000
25%	0.0048	0.0124
50%	0.0074	0.0180
75%	0.0114	0.0241
90%	0.0182	0.0298
max	0.2940	0.1209
RMSE	0.0131	

Resolution: 1024×768



Error per iteration for resolution 1024×768 and test05

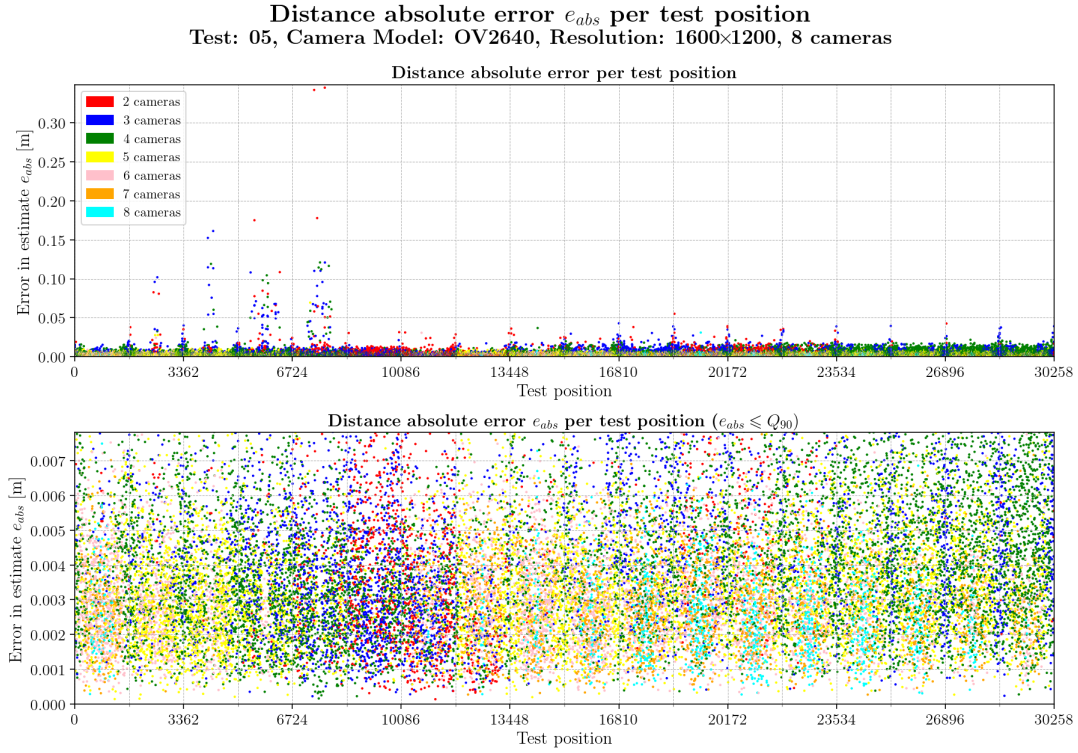
Table A.47: Statistics per number of cameras observing the beacon for test 05 with resolution 1024×768.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	1891	4606	8782	5918	4189	2456	1383
mean	0.0105	0.0097	0.0078	0.0052	0.0045	0.0040	0.0036
std	0.0141	0.0088	0.0063	0.0025	0.0023	0.0018	0.0016
min	0.0004	0.0003	0.0003	0.0002	0.0002	0.0002	0.0003
25%	0.0044	0.0045	0.0042	0.0033	0.0030	0.0026	0.0024
50%	0.0071	0.0070	0.0063	0.0048	0.0043	0.0037	0.0035
75%	0.0122	0.0117	0.0094	0.0066	0.0057	0.0050	0.0046
max	0.3114	0.1052	0.1306	0.0258	0.0500	0.0141	0.0191

Table A.48: Global statistics and root mean square error for test 05 with resolution 1024×768.

	Abs. error	Est. error
count	29225	29225
mean	0.0067	0.0129
std	0.0067	0.0061
min	0.0002	0.0000
25%	0.0035	0.0090
50%	0.0052	0.0127
75%	0.0078	0.0164
90%	0.0120	0.0200
max	0.3114	0.1243
RMSE	0.0095	

Resolution: 1600×1200



Error per iteration for resolution 1600×1200 and test05

Table A.49: Statistics per number of cameras observing the beacon for test 05 with resolution 1600×1200.

	2 cameras	3 cameras	4 cameras	5 cameras	6 cameras	7 cameras	8 cameras
count	1882	4590	8759	5939	4199	2473	1386
mean	0.0072	0.0065	0.0050	0.0034	0.0030	0.0028	0.0026
std	0.0145	0.0086	0.0050	0.0019	0.0015	0.0012	0.0014
min	0.0001	0.0002	0.0002	0.0001	0.0003	0.0003	0.0003
25%	0.0027	0.0029	0.0027	0.0022	0.0020	0.0019	0.0017
50%	0.0047	0.0046	0.0040	0.0031	0.0028	0.0027	0.0024
75%	0.0081	0.0073	0.0061	0.0043	0.0038	0.0035	0.0033
max	0.3452	0.1612	0.1209	0.0693	0.0306	0.0081	0.0309

Table A.50: Global statistics and root mean square error for test 05 with resolution 1600×1200.

	Abs. error	Est. error
count	29228	29228
mean	0.0045	0.0084
std	0.0060	0.0051
min	0.0001	0.0000
25%	0.0023	0.0058
50%	0.0034	0.0082
75%	0.0051	0.0107
90%	0.0078	0.0128
max	0.3452	0.1647
RMSE	0.0075	

Error and camera coverage maps

Error maps for resolution 320×240

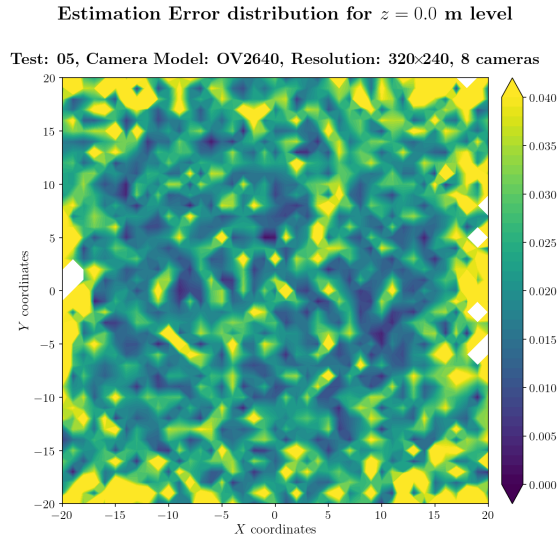


Figure A.641: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 0.000$ m.

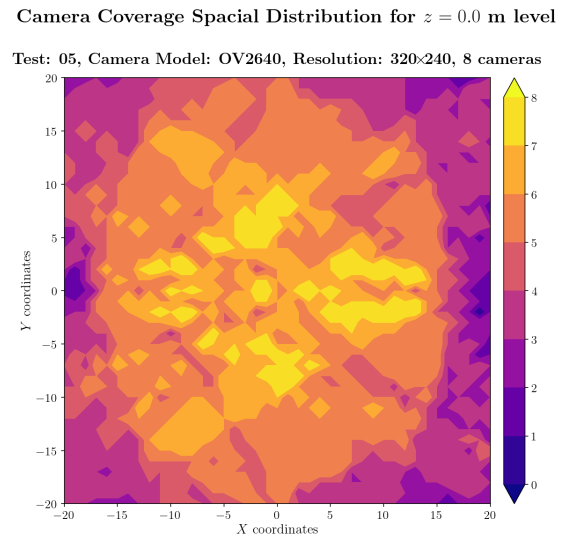


Figure A.642: Camera coverage for test 05 at level $z = 0.000$ m and resolution 320×240.

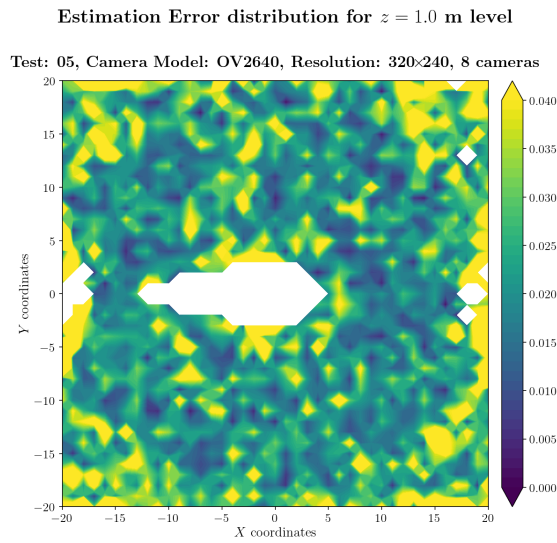


Figure A.643: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 1.000$ m.

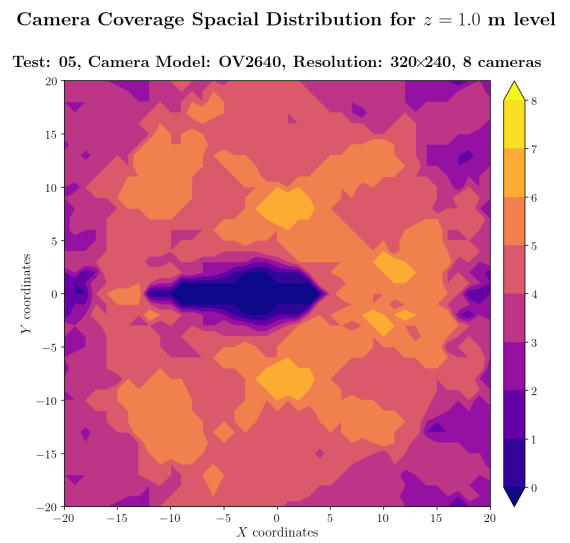
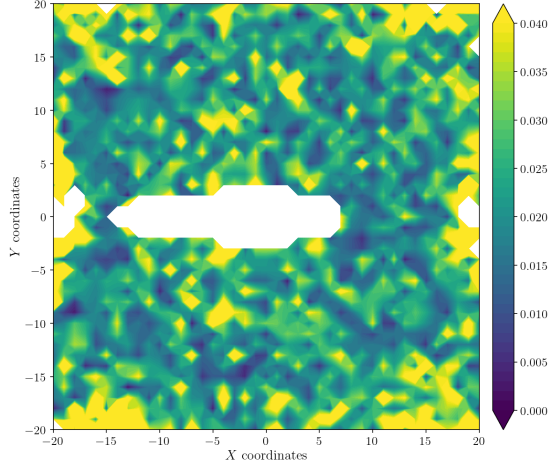


Figure A.644: Camera coverage for test 05 at level $z = 1.000$ m and resolution 320×240.

Estimation Error distribution for $z = 2.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

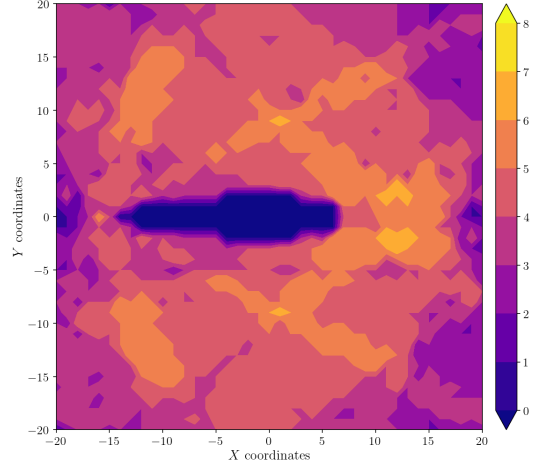
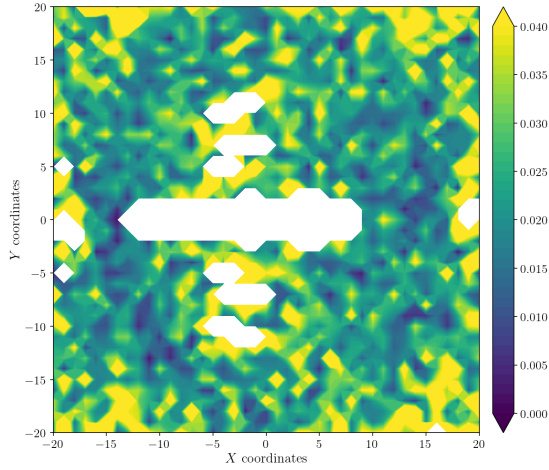


Figure A.645: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 2.000$ m.

Figure A.646: Camera coverage for test 05 at level $z = 2.000$ m and resolution 320×240.

Estimation Error distribution for $z = 3.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

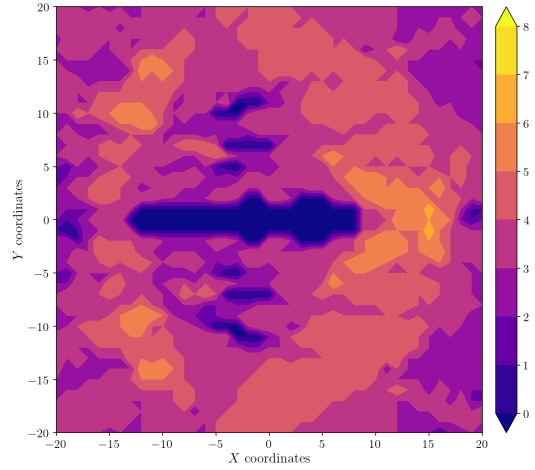


Figure A.647: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 3.000$ m.

Figure A.648: Camera coverage for test 05 at level $z = 3.000$ m and resolution 320×240.

Estimation Error distribution for $z = 4.0$ m level

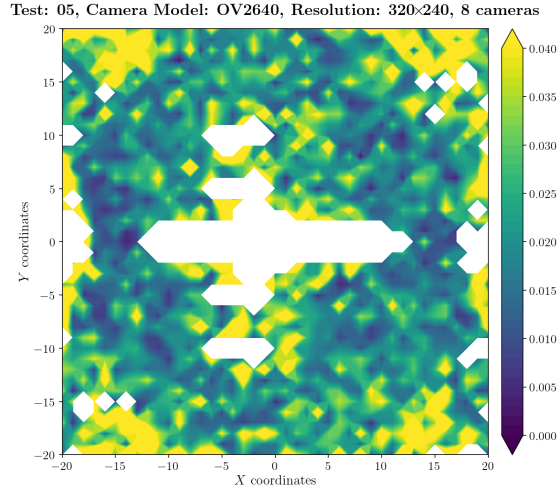


Figure A.649: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 4.000$ m.

Camera Coverage Spatial Distribution for $z = 4.0$ m level

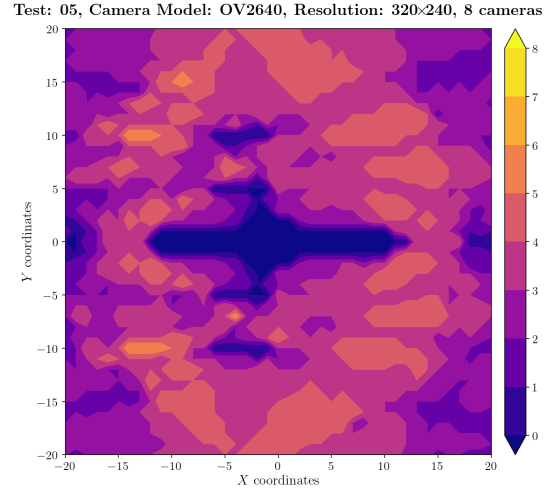


Figure A.650: Camera coverage for test 05 at level $z = 4.000$ m and resolution 320×240.

Estimation Error distribution for $z = 5.0$ m level

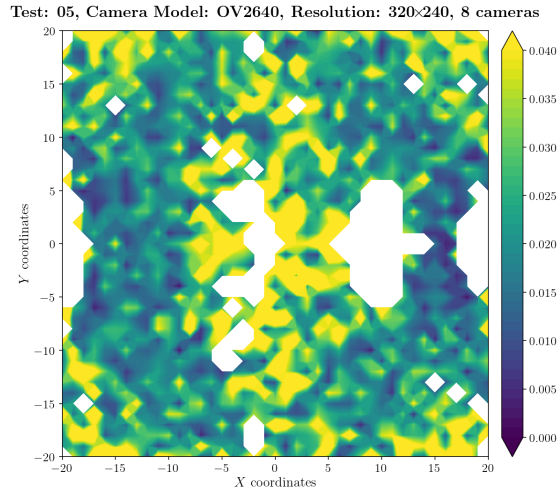


Figure A.651: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 5.000$ m.

Camera Coverage Spatial Distribution for $z = 5.0$ m level

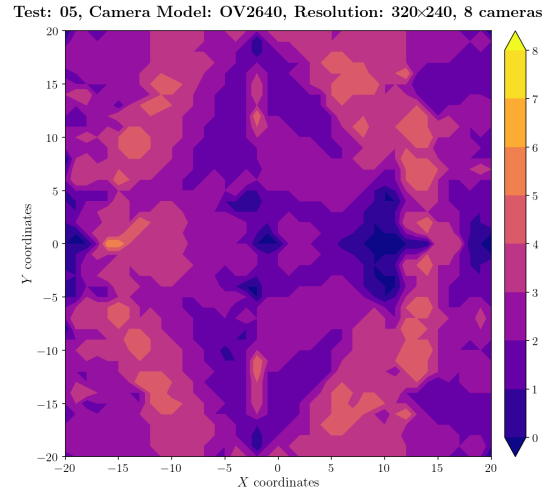
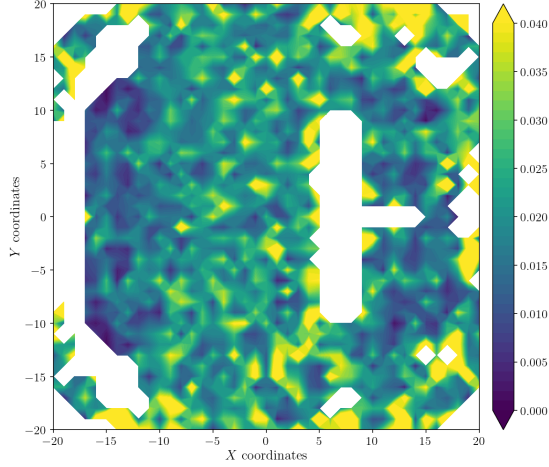


Figure A.652: Camera coverage for test 05 at level $z = 5.000$ m and resolution 320×240.

Estimation Error distribution for $z = 6.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

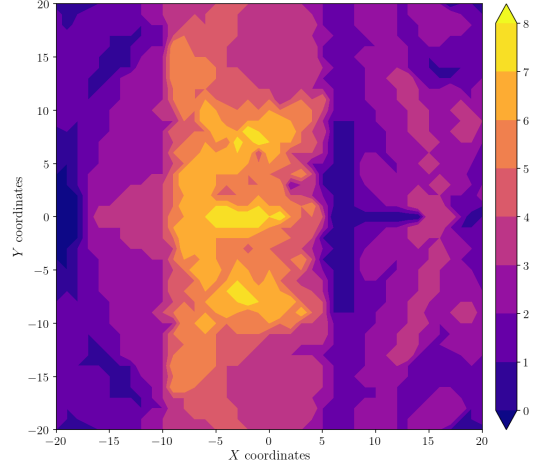
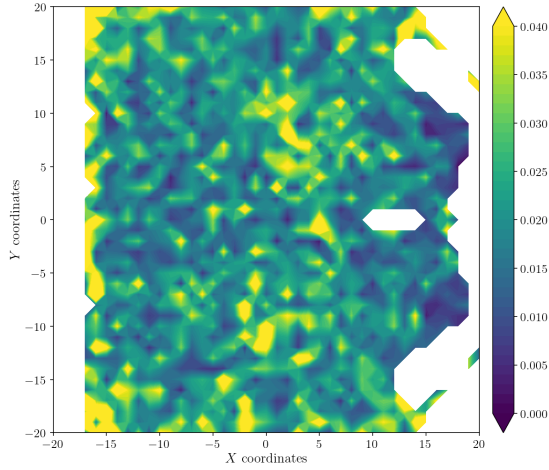


Figure A.653: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 6.000$ m.

Figure A.654: Camera coverage for test 05 at level $z = 6.000$ m and resolution 320×240.

Estimation Error distribution for $z = 7.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

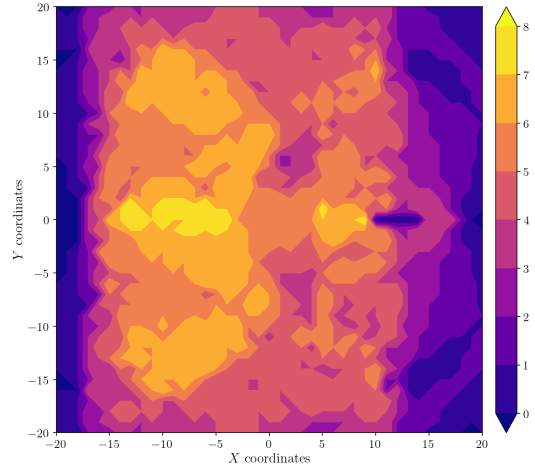


Figure A.655: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 7.000$ m.

Figure A.656: Camera coverage for test 05 at level $z = 7.000$ m and resolution 320×240.

Estimation Error distribution for $z = 8.0$ m level

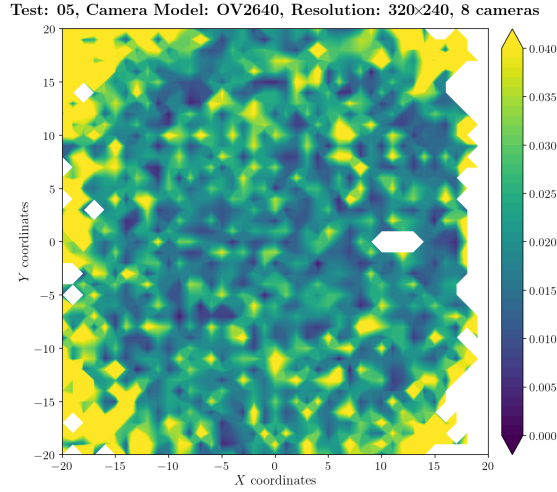


Figure A.657: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 8.000$ m.

Camera Coverage Spatial Distribution for $z = 8.0$ m level

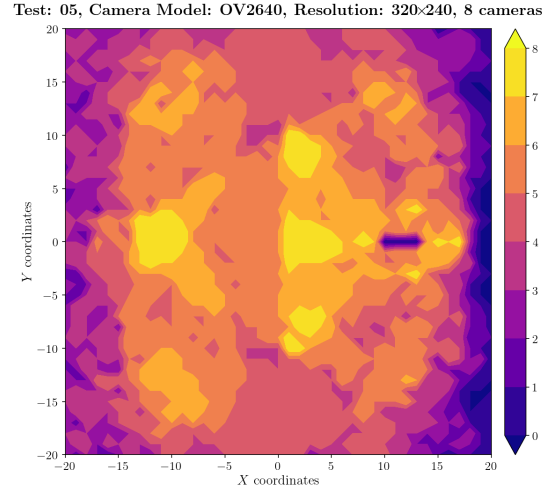


Figure A.658: Camera coverage for test 05 at level $z = 8.000$ m and resolution 320×240.

Estimation Error distribution for $z = 9.0$ m level

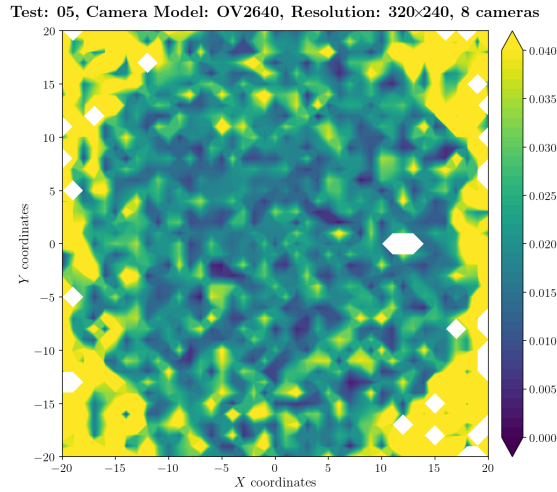


Figure A.659: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 9.000$ m.

Camera Coverage Spatial Distribution for $z = 9.0$ m level

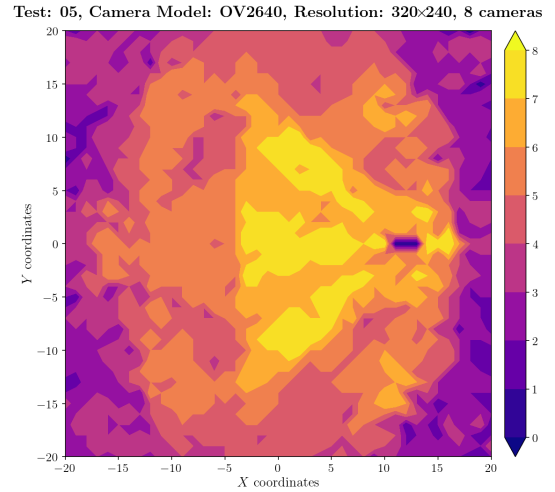


Figure A.660: Camera coverage for test 05 at level $z = 9.000$ m and resolution 320×240.

Estimation Error distribution for $z = 10.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

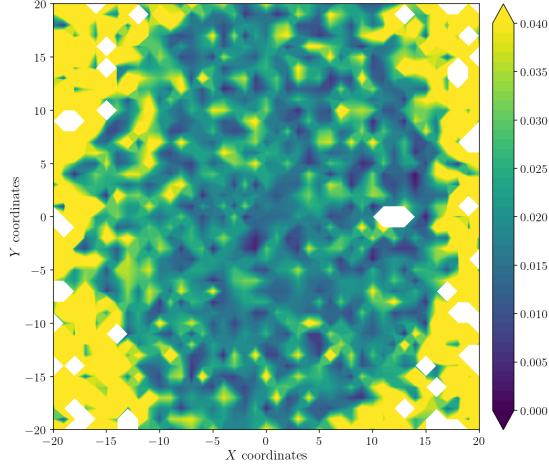


Figure A.661: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 10.000$ m.

Camera Coverage Spacial Distribution for $z = 10.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

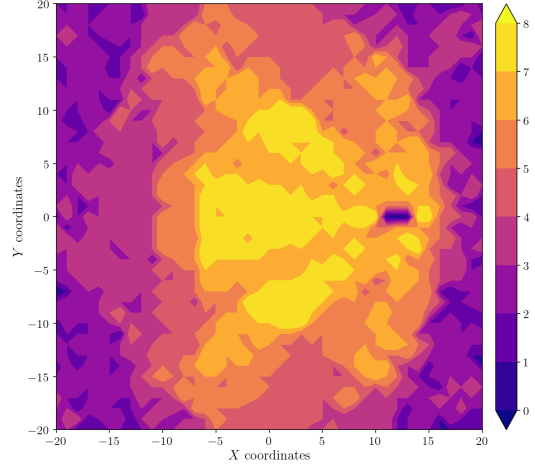


Figure A.662: Camera coverage for test 05 at level $z = 10.000$ m and resolution 320×240.

Estimation Error distribution for $z = 11.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

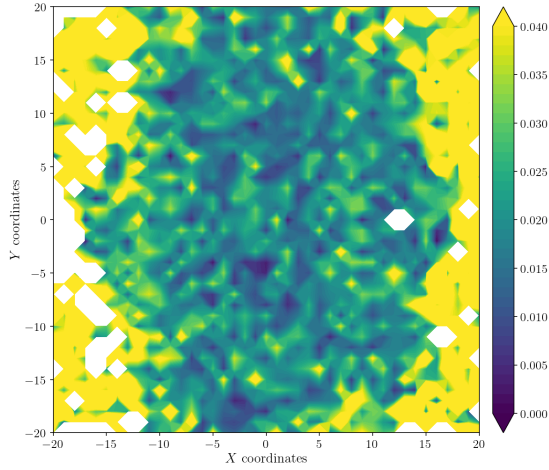


Figure A.663: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 11.000$ m.

Camera Coverage Spacial Distribution for $z = 11.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

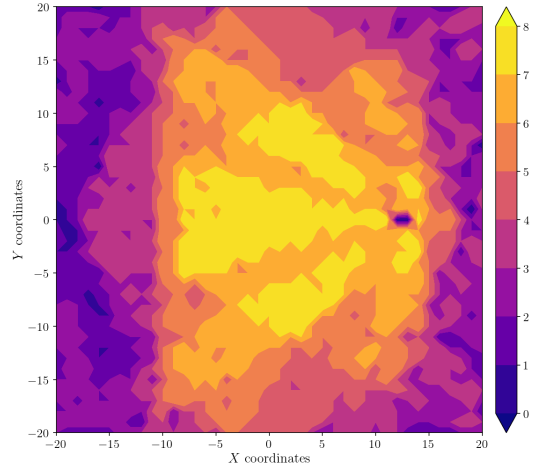


Figure A.664: Camera coverage for test 05 at level $z = 11.000$ m and resolution 320×240.

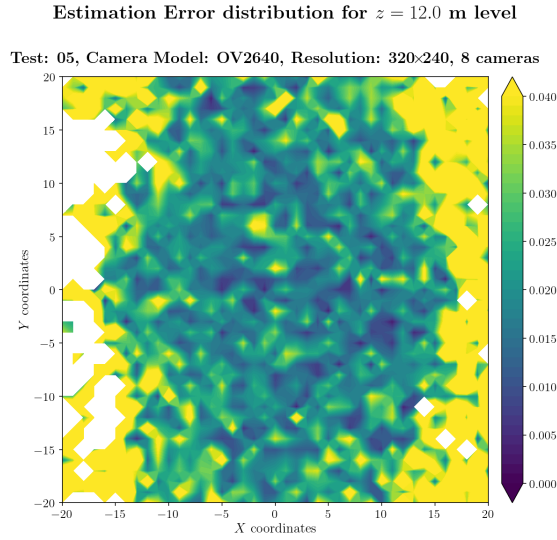


Figure A.665: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 12.000$ m.

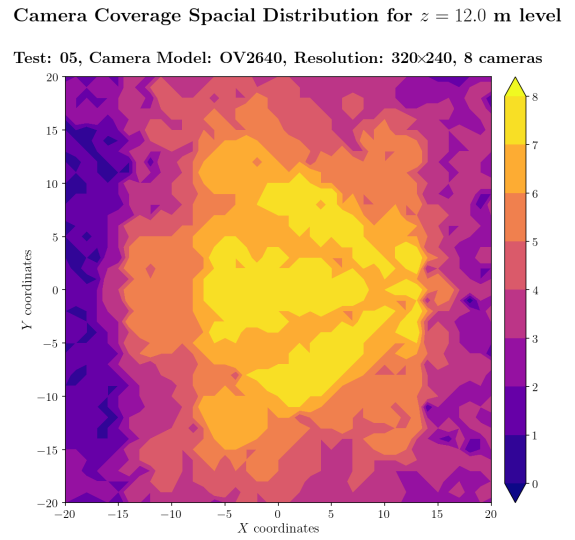


Figure A.666: Camera coverage for test 05 at level $z = 12.000$ m and resolution 320×240.

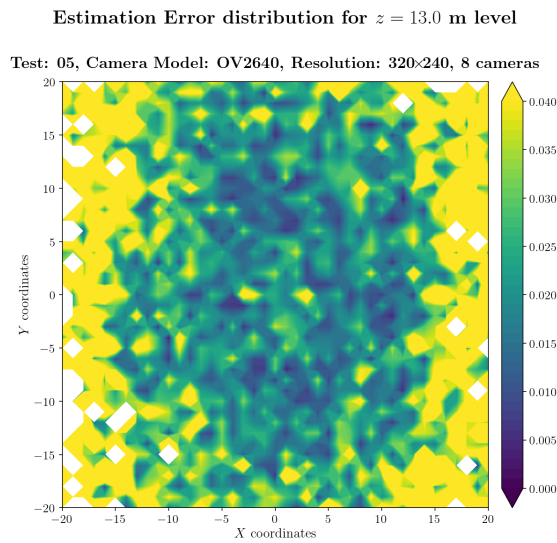


Figure A.667: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 13.000$ m.

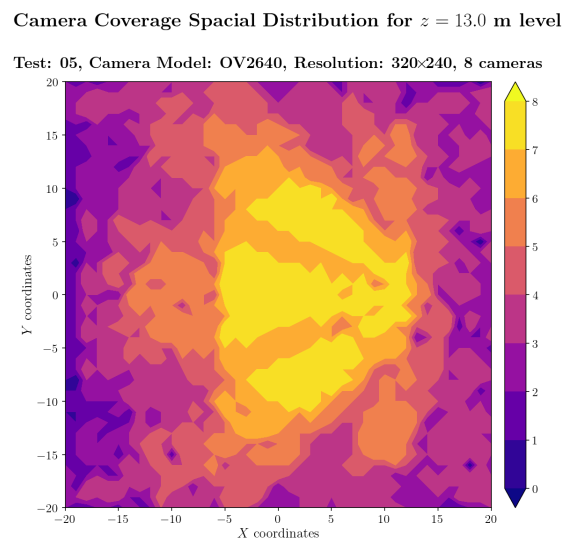


Figure A.668: Camera coverage for test 05 at level $z = 13.000$ m and resolution 320×240.

Estimation Error distribution for $z = 14.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

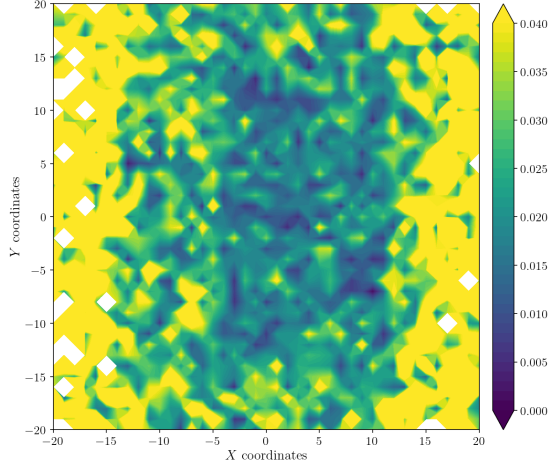


Figure A.669: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 14.000$ m.

Camera Coverage Spacial Distribution for $z = 14.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

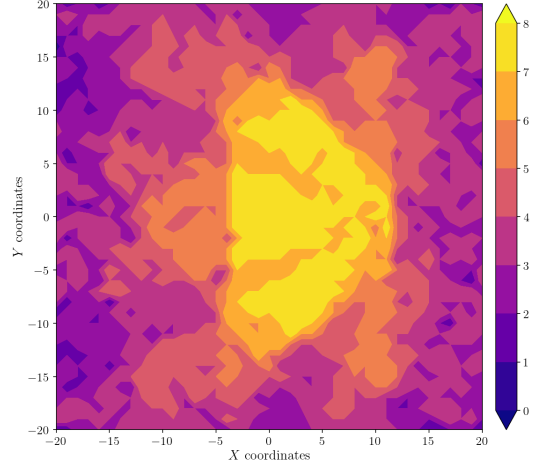


Figure A.670: Camera coverage for test 05 at level $z = 14.000$ m and resolution 320×240.

Estimation Error distribution for $z = 15.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

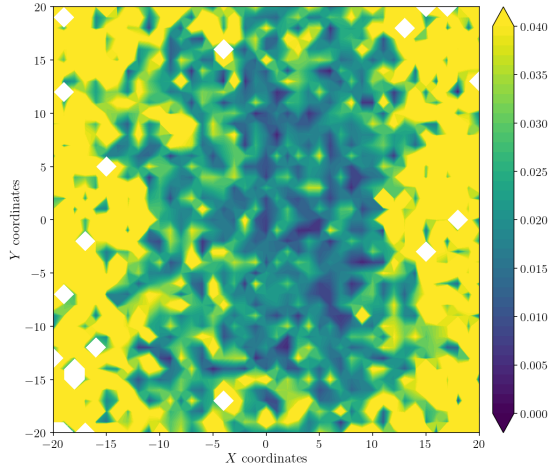


Figure A.671: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 15.000$ m.

Camera Coverage Spacial Distribution for $z = 15.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 320×240, 8 cameras

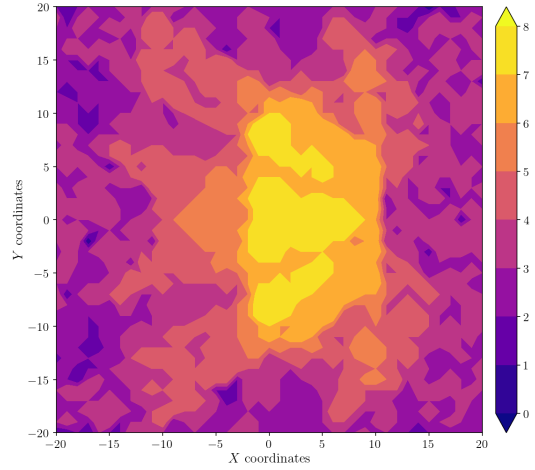


Figure A.672: Camera coverage for test 05 at level $z = 15.000$ m and resolution 320×240.

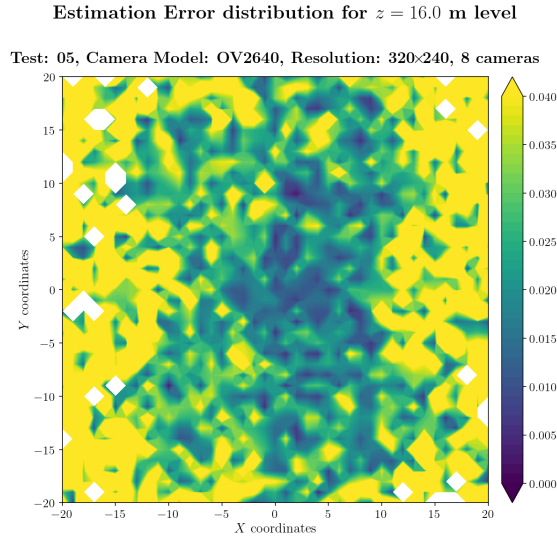


Figure A.673: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 16.000$ m.

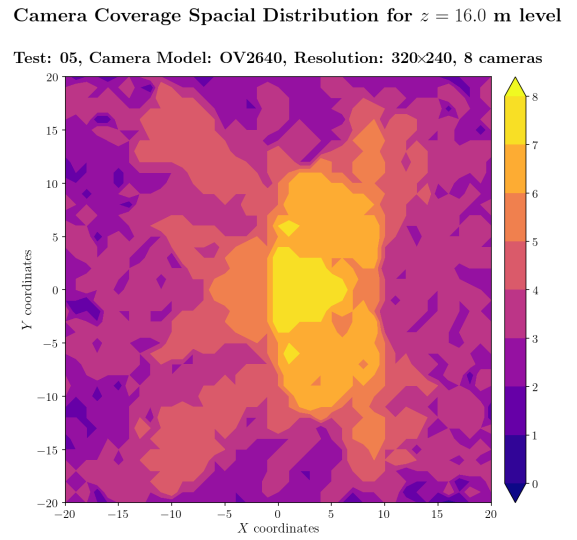


Figure A.674: Camera coverage for test 05 at level $z = 16.000$ m and resolution 320×240.

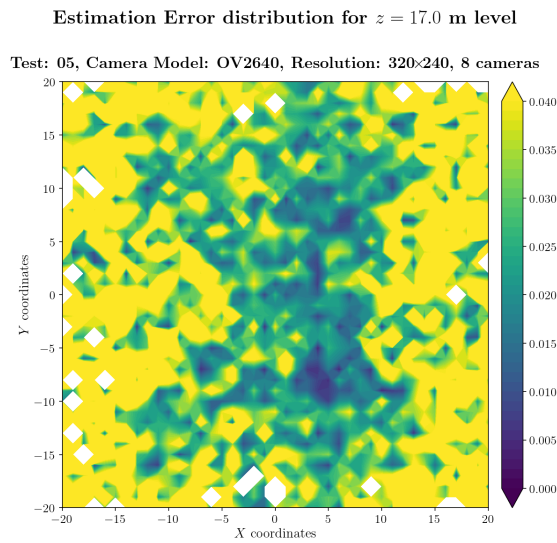


Figure A.675: Estimation error contour map for test 05 with 8 cameras using resolution of 320×240 pixels, at level $z = 17.000$ m.

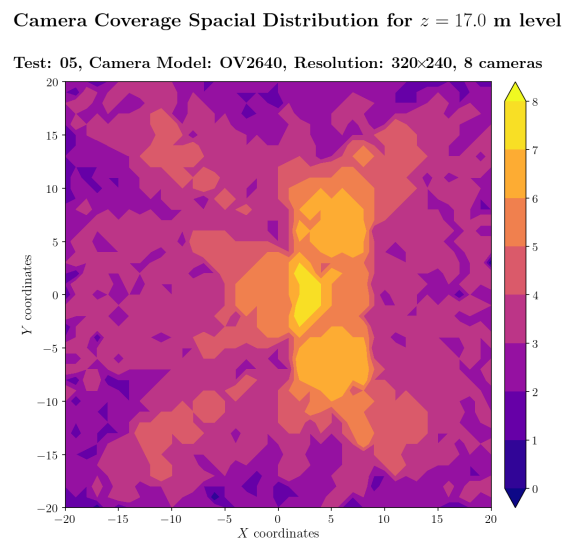


Figure A.676: Camera coverage for test 05 at level $z = 17.000$ m and resolution 320×240.

Error maps for resolution 640×480

Estimation Error distribution for $z = 0.0$ m level

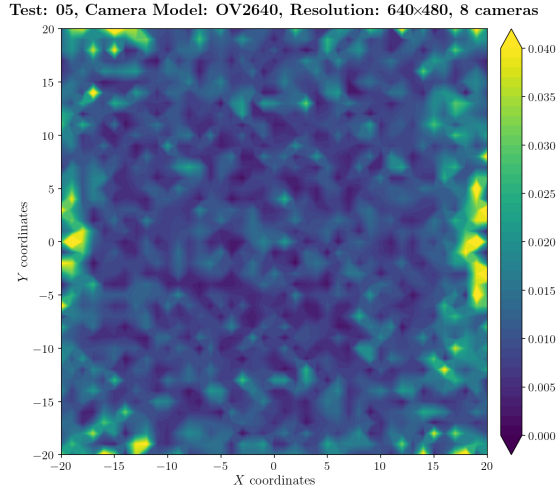


Figure A.677: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

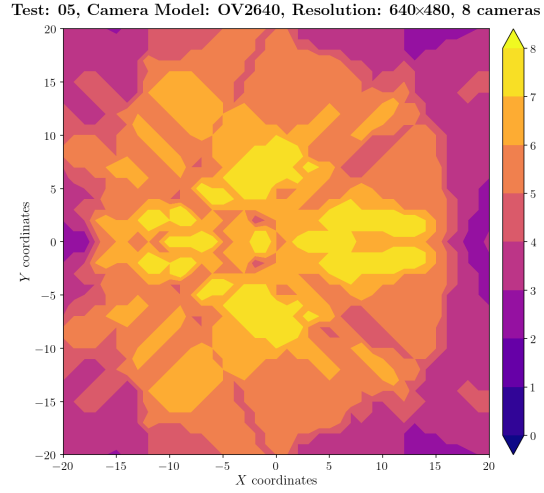


Figure A.678: Camera coverage for test 05 at level $z = 0.000$ m and resolution 640×480.

Estimation Error distribution for $z = 1.0$ m level

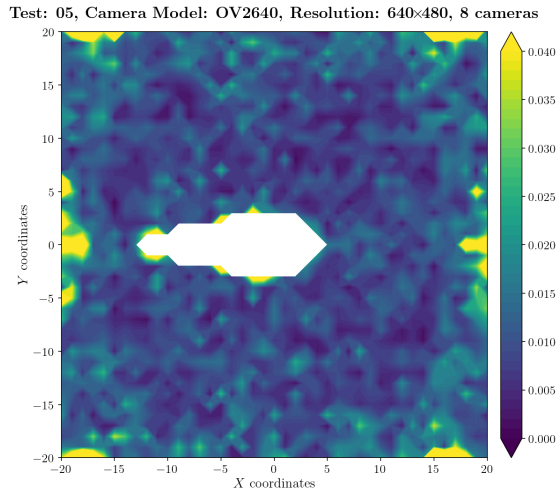


Figure A.679: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 1.000$ m.

Camera Coverage Spacial Distribution for $z = 1.0$ m level

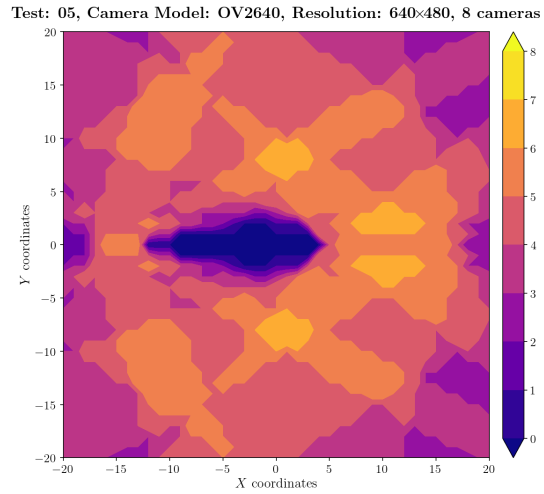


Figure A.680: Camera coverage for test 05 at level $z = 1.000$ m and resolution 640×480.

Estimation Error distribution for $z = 2.0$ m level

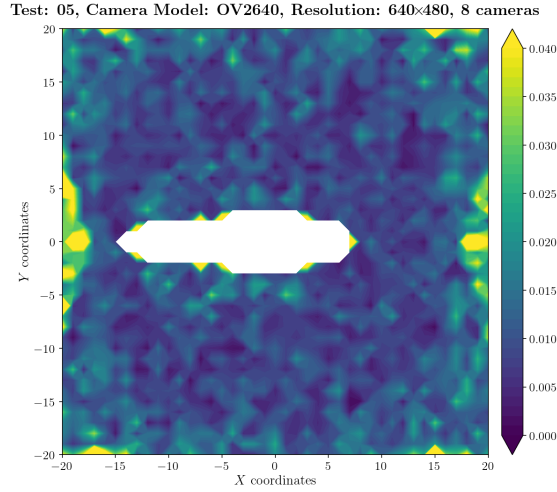


Figure A.681: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 2.000$ m.

Camera Coverage Spacial Distribution for $z = 2.0$ m level

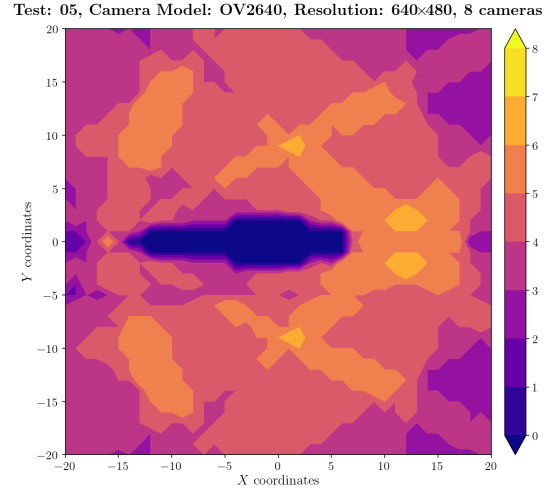


Figure A.682: Camera coverage for test 05 at level $z = 2.000$ m and resolution 640×480.

Estimation Error distribution for $z = 3.0$ m level

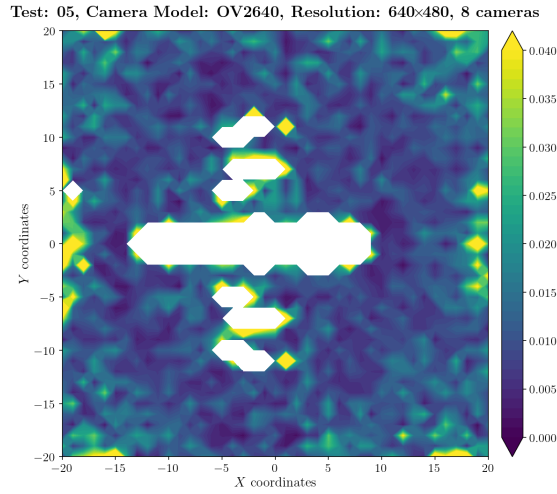


Figure A.683: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 3.000$ m.

Camera Coverage Spacial Distribution for $z = 3.0$ m level

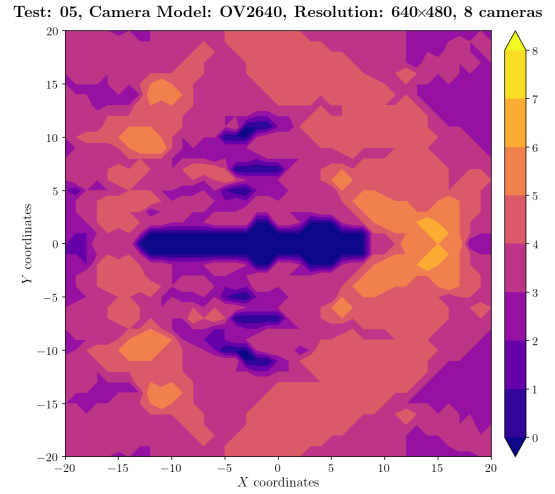
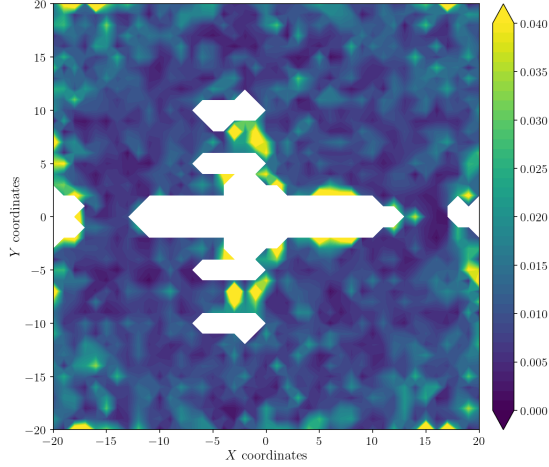


Figure A.684: Camera coverage for test 05 at level $z = 3.000$ m and resolution 640×480.

Estimation Error distribution for $z = 4.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

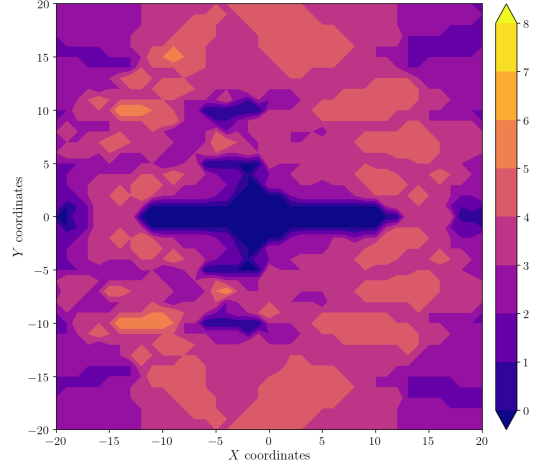
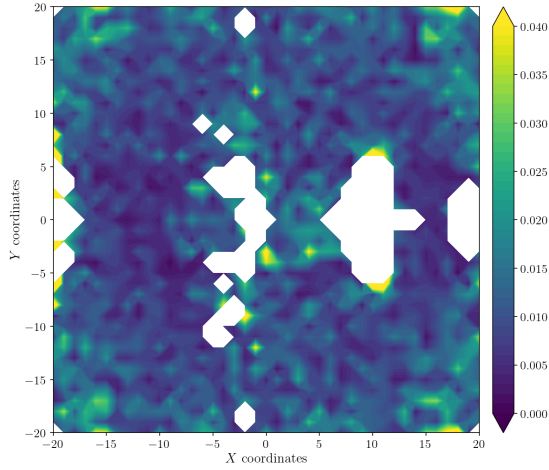


Figure A.685: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 4.000$ m.

Figure A.686: Camera coverage for test 05 at level $z = 4.000$ m and resolution 640×480.

Estimation Error distribution for $z = 5.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

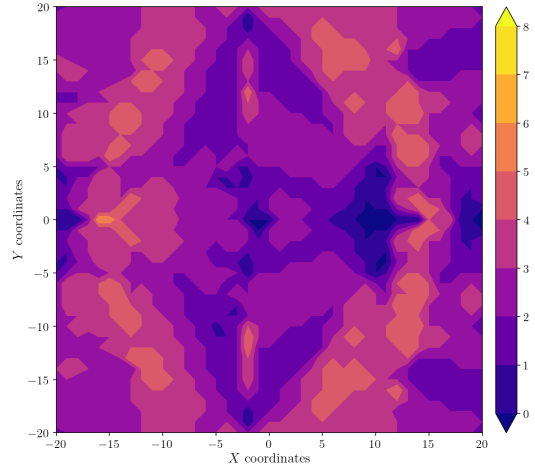


Figure A.687: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 5.000$ m.

Figure A.688: Camera coverage for test 05 at level $z = 5.000$ m and resolution 640×480.

Estimation Error distribution for $z = 6.0$ m level

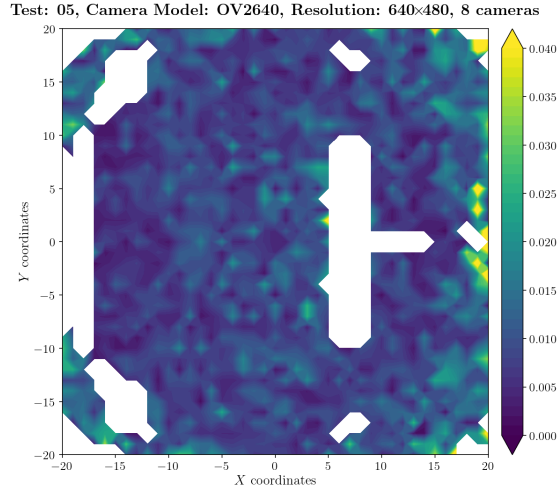


Figure A.689: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 6.000$ m.

Camera Coverage Spatial Distribution for $z = 6.0$ m level

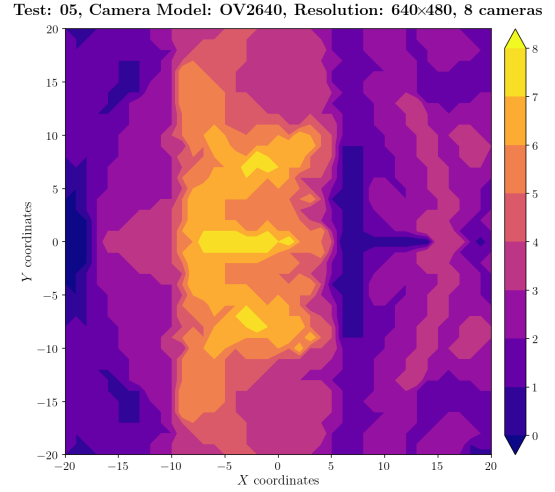


Figure A.690: Camera coverage for test 05 at level $z = 6.000$ m and resolution 640×480.

Estimation Error distribution for $z = 7.0$ m level

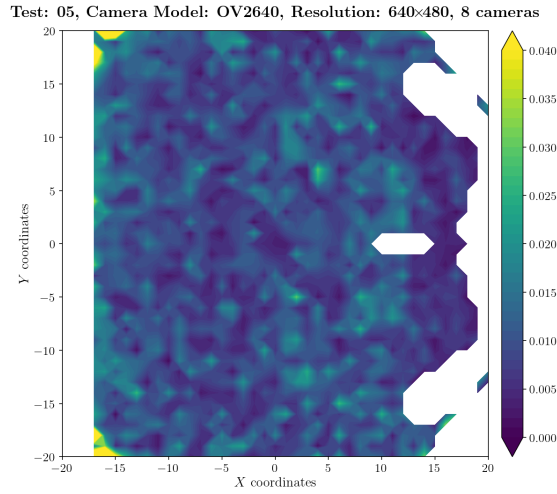


Figure A.691: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 7.000$ m.

Camera Coverage Spatial Distribution for $z = 7.0$ m level

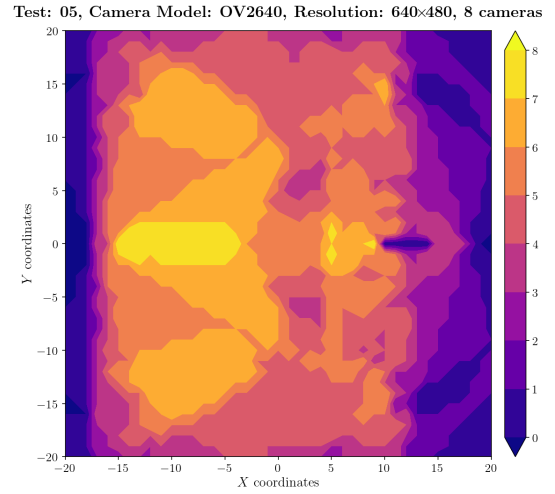
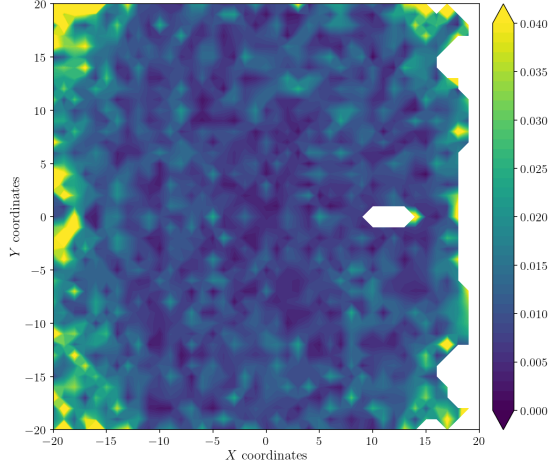


Figure A.692: Camera coverage for test 05 at level $z = 7.000$ m and resolution 640×480.

Estimation Error distribution for $z = 8.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

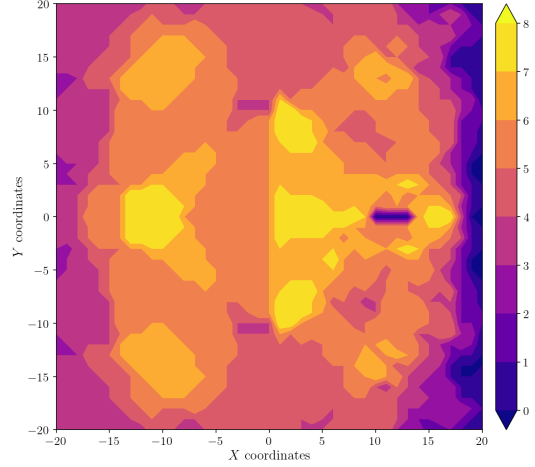
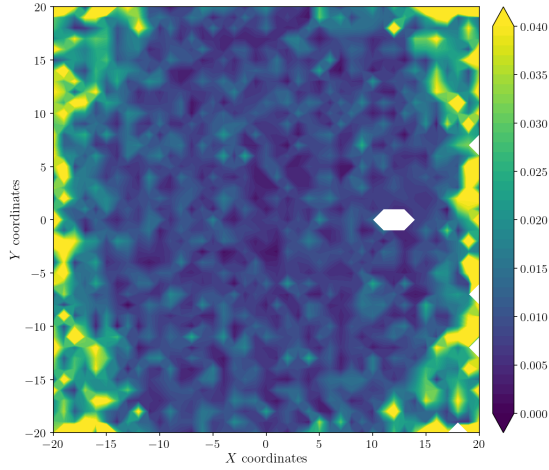


Figure A.693: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 8.000$ m.

Figure A.694: Camera coverage for test 05 at level $z = 8.000$ m and resolution 640×480.

Estimation Error distribution for $z = 9.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 9.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

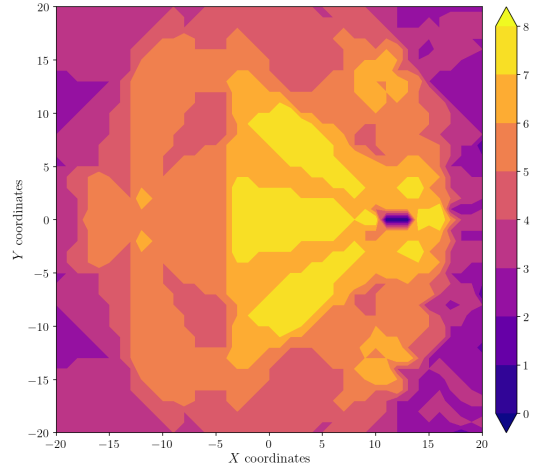


Figure A.695: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 9.000$ m.

Figure A.696: Camera coverage for test 05 at level $z = 9.000$ m and resolution 640×480.

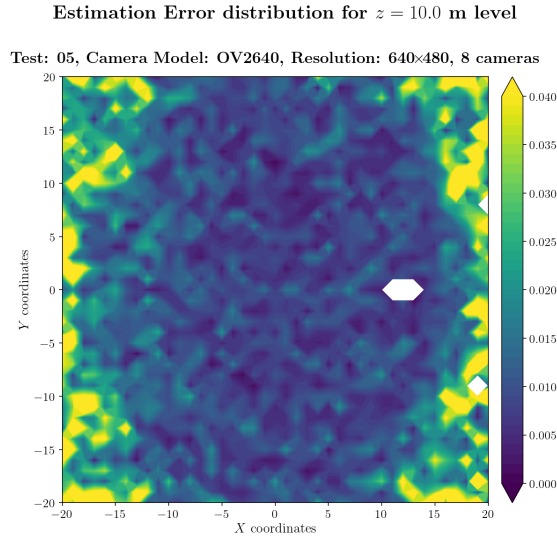


Figure A.697: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 10.000$ m.

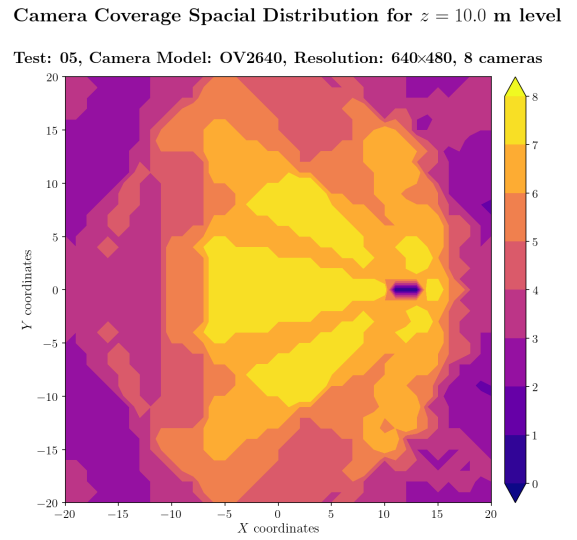


Figure A.698: Camera coverage for test 05 at level $z = 10.000$ m and resolution 640×480.

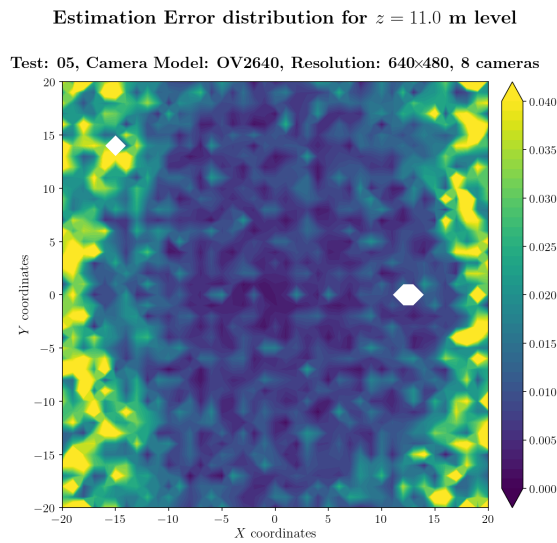


Figure A.699: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 11.000$ m.

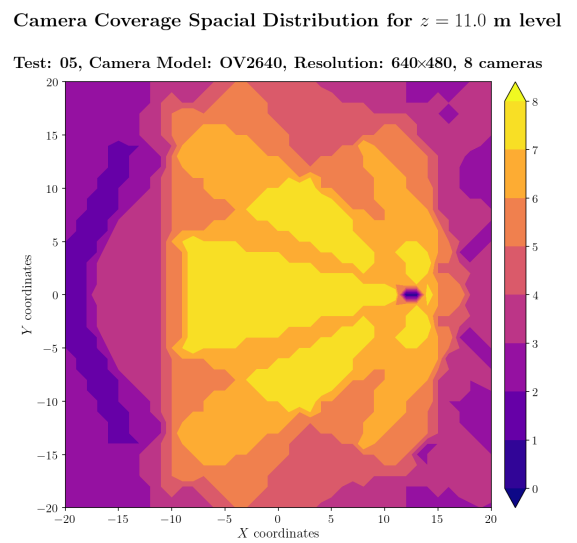


Figure A.700: Camera coverage for test 05 at level $z = 11.000$ m and resolution 640×480.

Estimation Error distribution for $z = 12.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

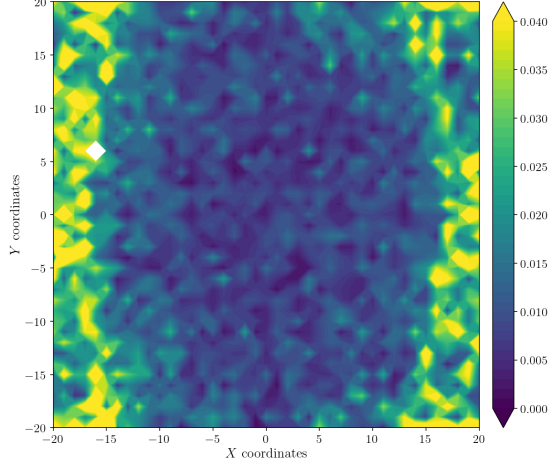


Figure A.701: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 12.000$ m.

Camera Coverage Spacial Distribution for $z = 12.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

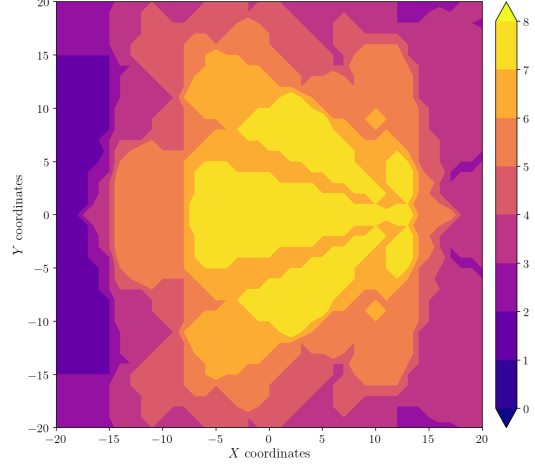


Figure A.702: Camera coverage for test 05 at level $z = 12.000$ m and resolution 640×480.

Estimation Error distribution for $z = 13.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

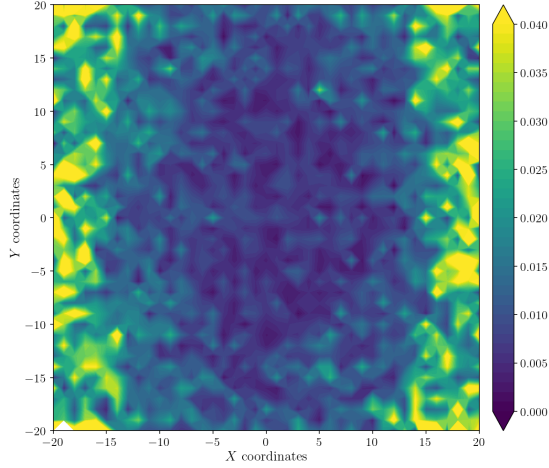


Figure A.703: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 13.000$ m.

Camera Coverage Spacial Distribution for $z = 13.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

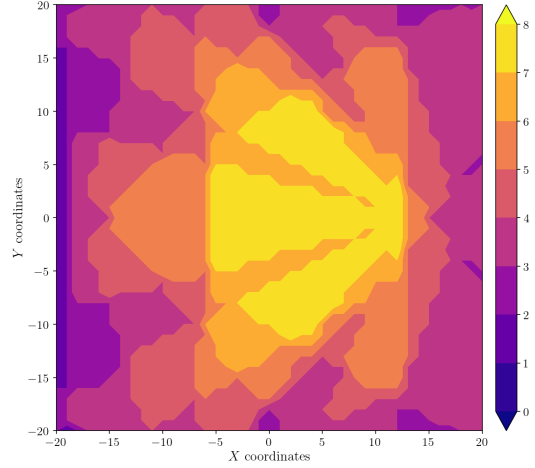


Figure A.704: Camera coverage for test 05 at level $z = 13.000$ m and resolution 640×480.

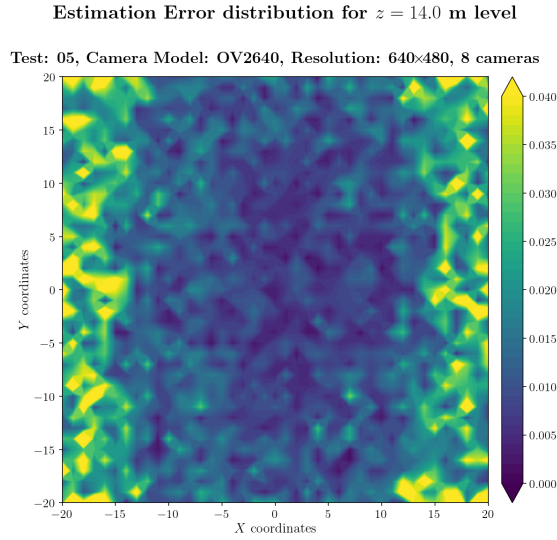


Figure A.705: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 14.000$ m.

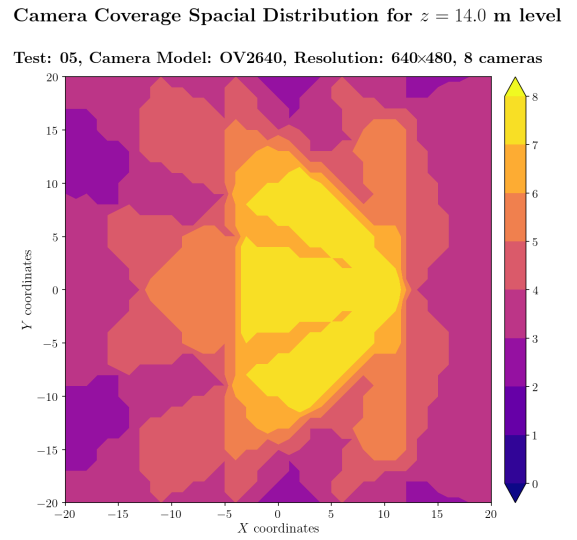


Figure A.706: Camera coverage for test 05 at level $z = 14.000$ m and resolution 640×480.

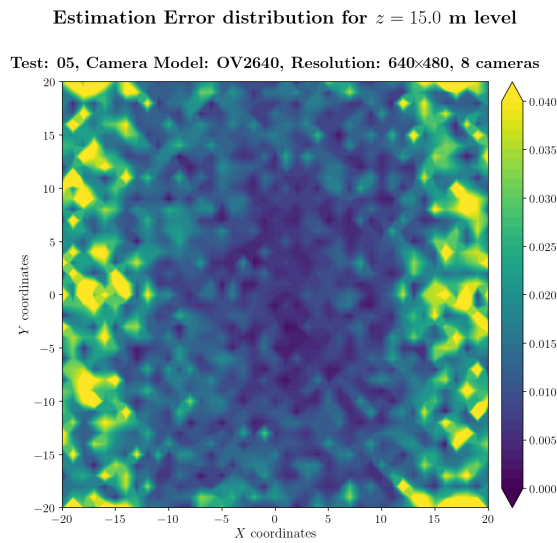


Figure A.707: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 15.000$ m.

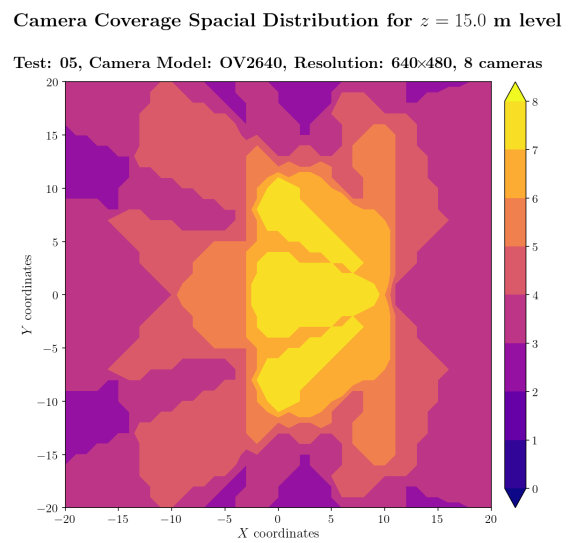
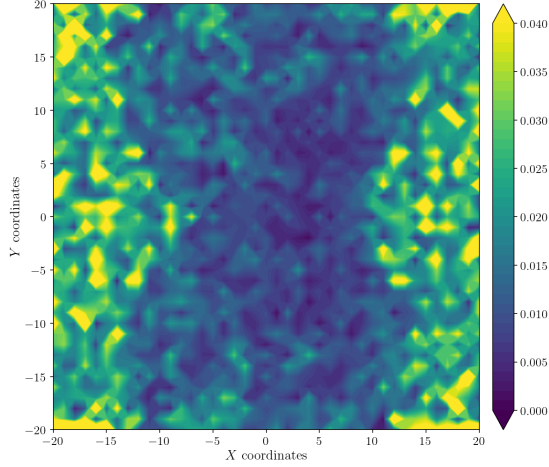


Figure A.708: Camera coverage for test 05 at level $z = 15.000$ m and resolution 640×480.

Estimation Error distribution for $z = 16.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 16.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

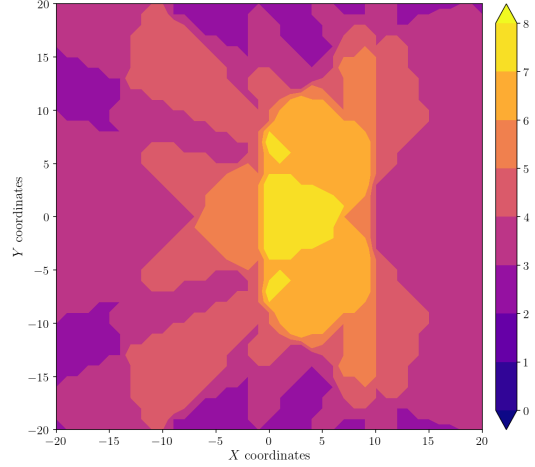
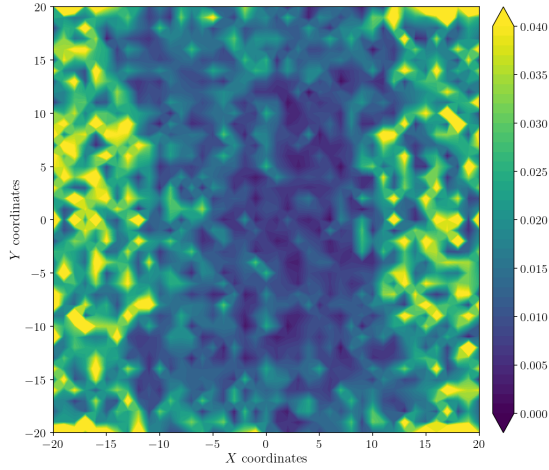


Figure A.709: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 16.000$ m.

Figure A.710: Camera coverage for test 05 at level $z = 16.000$ m and resolution 640×480.

Estimation Error distribution for $z = 17.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras



Camera Coverage Spacial Distribution for $z = 17.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 640×480, 8 cameras

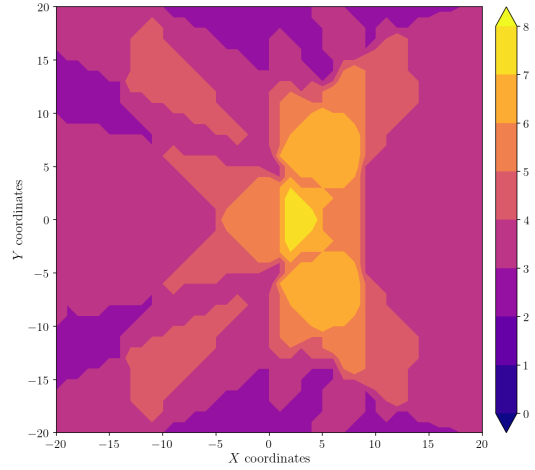


Figure A.711: Estimation error contour map for test 05 with 8 cameras using resolution of 640×480 pixels, at level $z = 17.000$ m.

Figure A.712: Camera coverage for test 05 at level $z = 17.000$ m and resolution 640×480.

Error maps for resolution 800×600

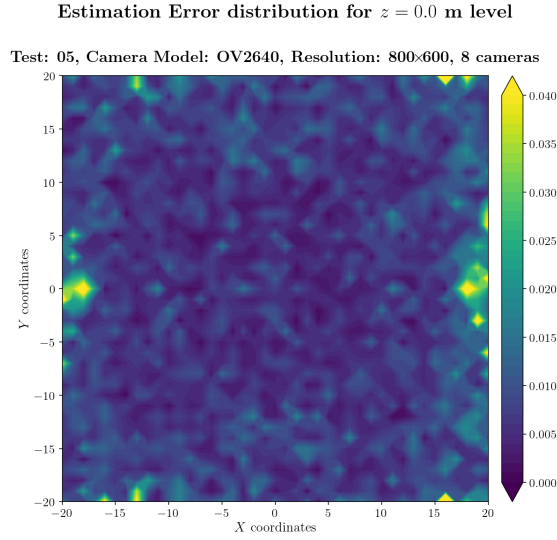


Figure A.713: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 0.000$ m.

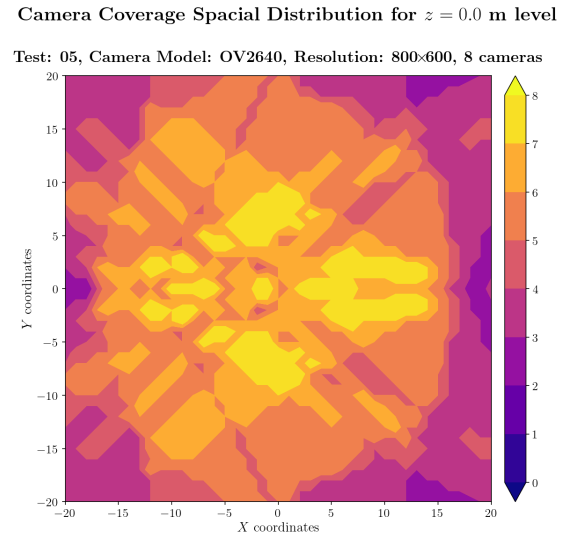


Figure A.714: Camera coverage for test 05 at level $z = 0.000$ m and resolution 800×600.

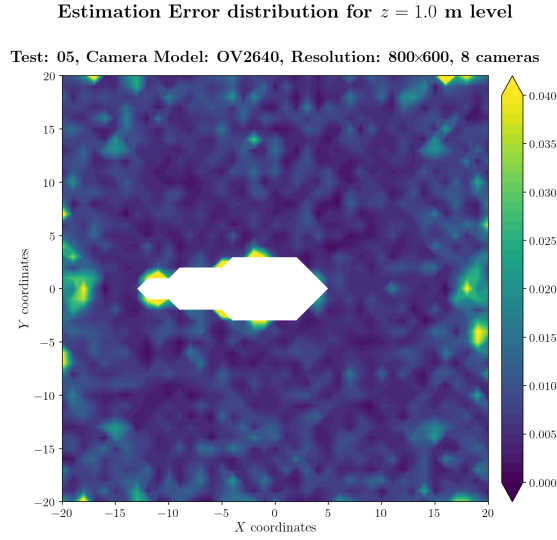


Figure A.715: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 1.000$ m.

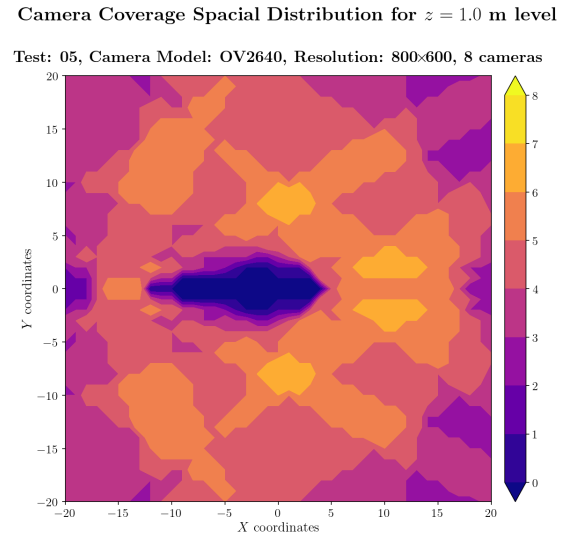
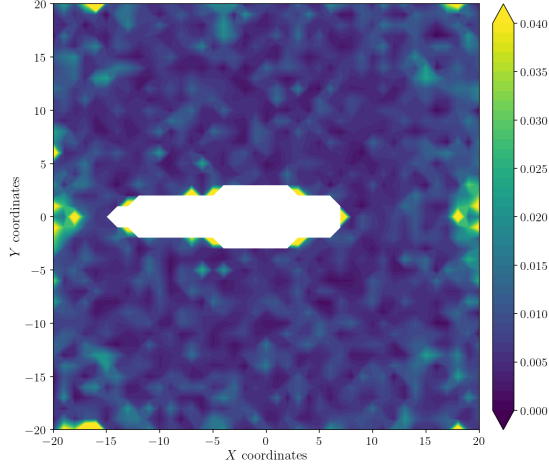


Figure A.716: Camera coverage for test 05 at level $z = 1.000$ m and resolution 800×600.

Estimation Error distribution for $z = 2.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

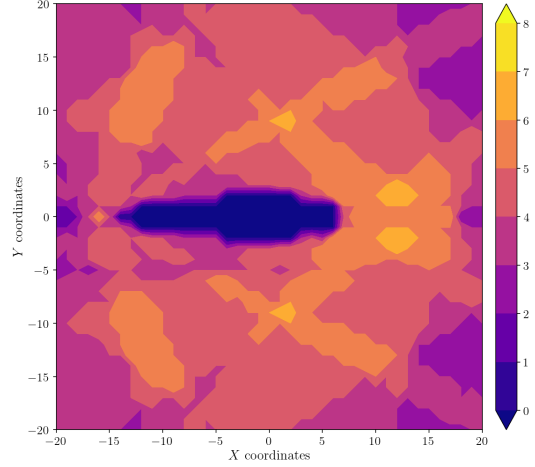
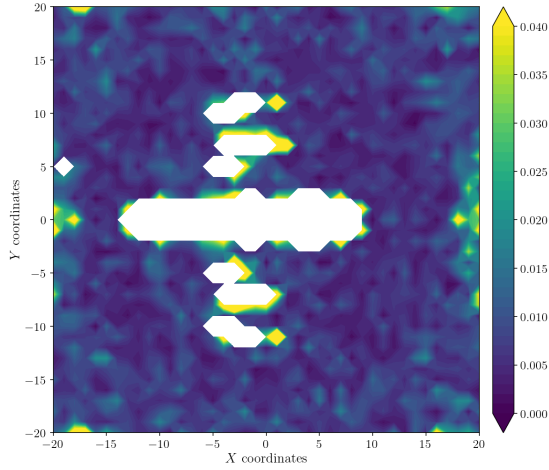


Figure A.717: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 2.000$ m.

Figure A.718: Camera coverage for test 05 at level $z = 2.000$ m and resolution 800×600.

Estimation Error distribution for $z = 3.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

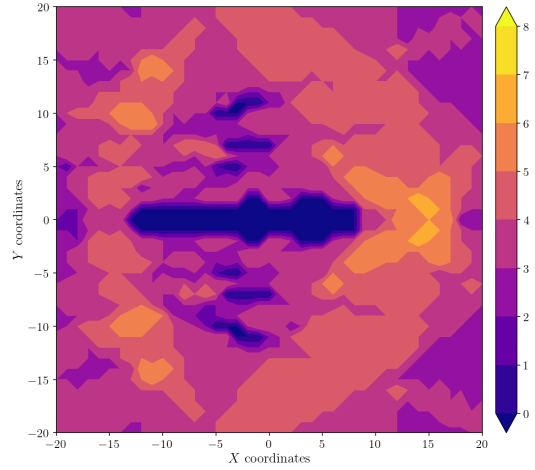


Figure A.719: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 3.000$ m.

Figure A.720: Camera coverage for test 05 at level $z = 3.000$ m and resolution 800×600.

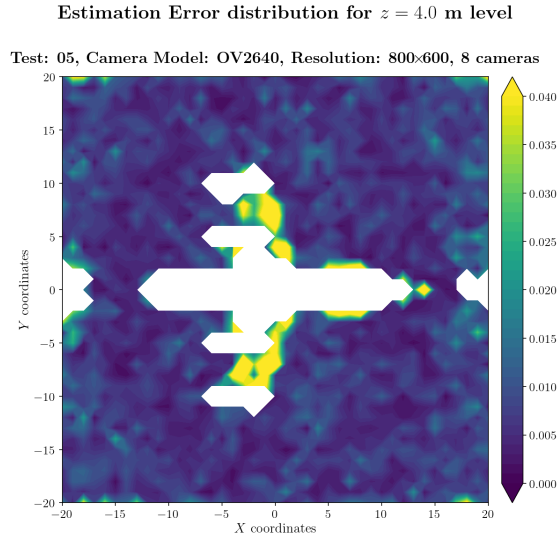


Figure A.721: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 4.000$ m.

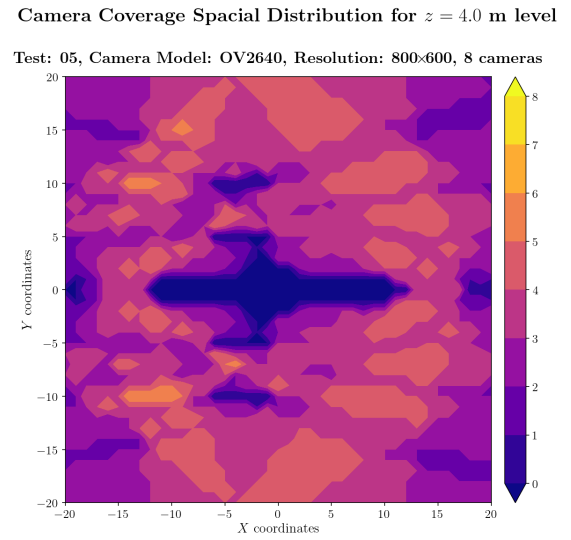


Figure A.722: Camera coverage for test 05 at level $z = 4.000$ m and resolution 800×600.

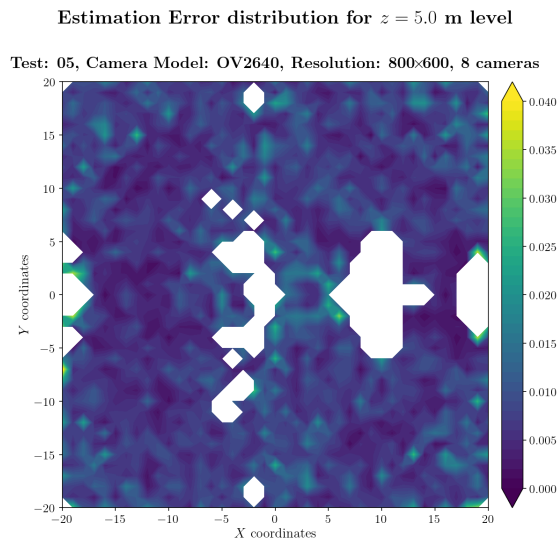


Figure A.723: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 5.000$ m.

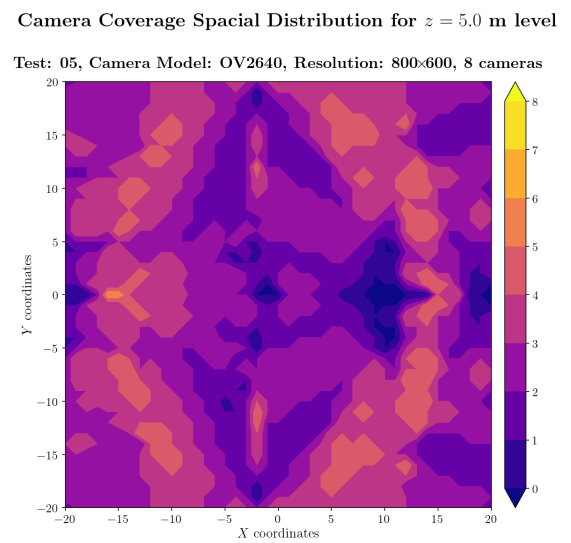


Figure A.724: Camera coverage for test 05 at level $z = 5.000$ m and resolution 800×600.

Estimation Error distribution for $z = 6.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

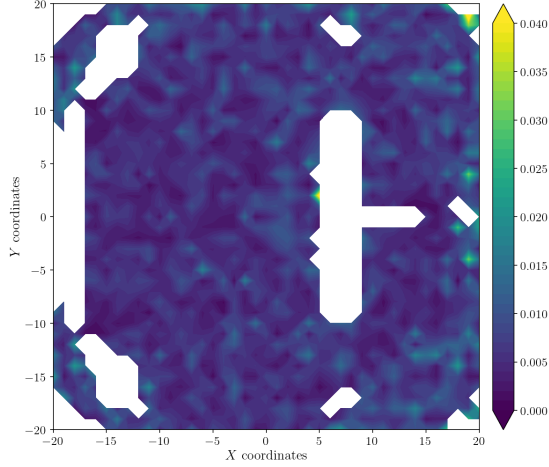


Figure A.725: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 6.000$ m.

Camera Coverage Spacial Distribution for $z = 6.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

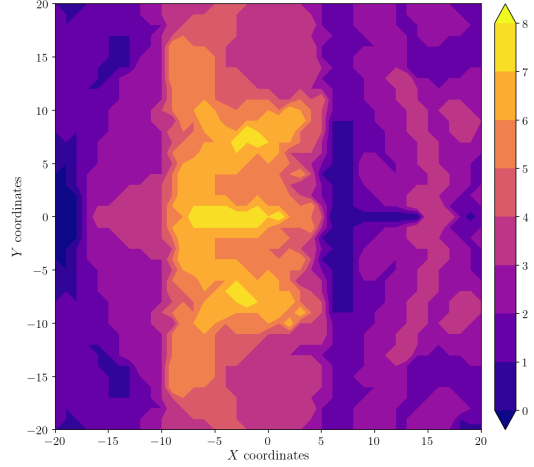


Figure A.726: Camera coverage for test 05 at level $z = 6.000$ m and resolution 800×600.

Estimation Error distribution for $z = 7.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

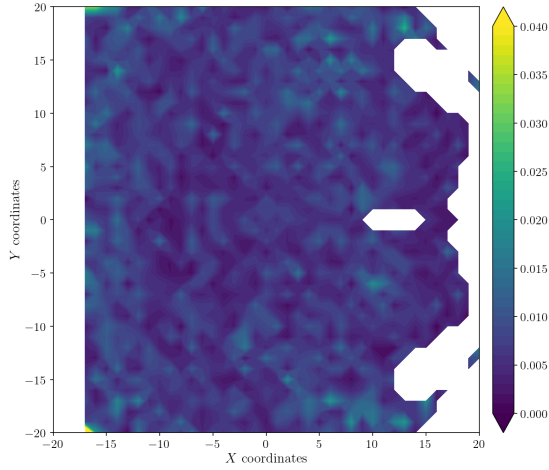


Figure A.727: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 7.000$ m.

Camera Coverage Spacial Distribution for $z = 7.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

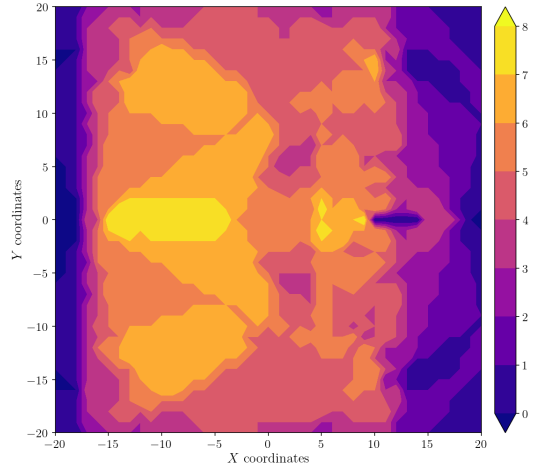


Figure A.728: Camera coverage for test 05 at level $z = 7.000$ m and resolution 800×600.

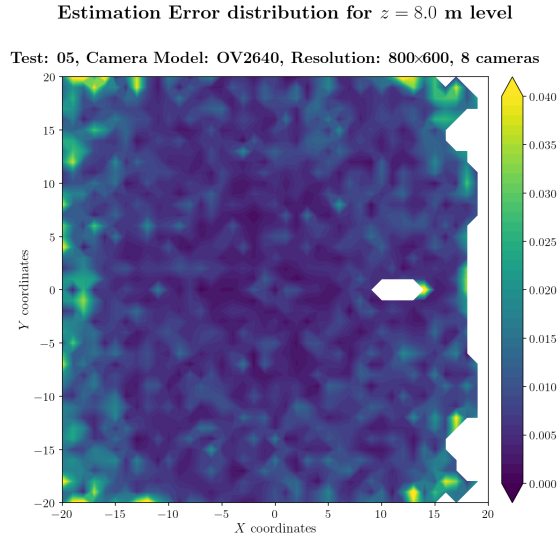


Figure A.729: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 8.000$ m.

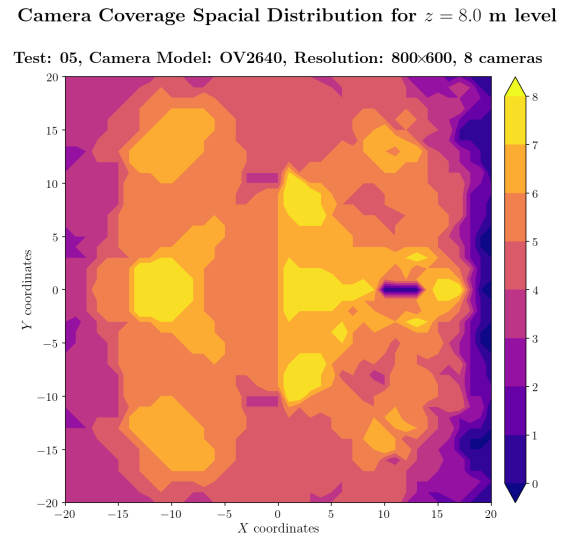


Figure A.730: Camera coverage for test 05 at level $z = 8.000$ m and resolution 800×600.

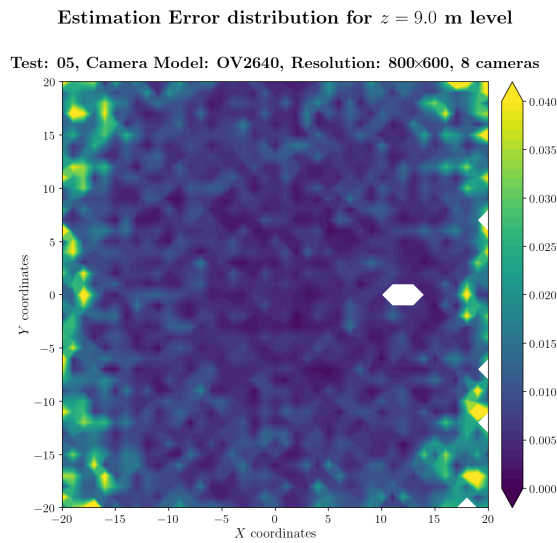


Figure A.731: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 9.000$ m.

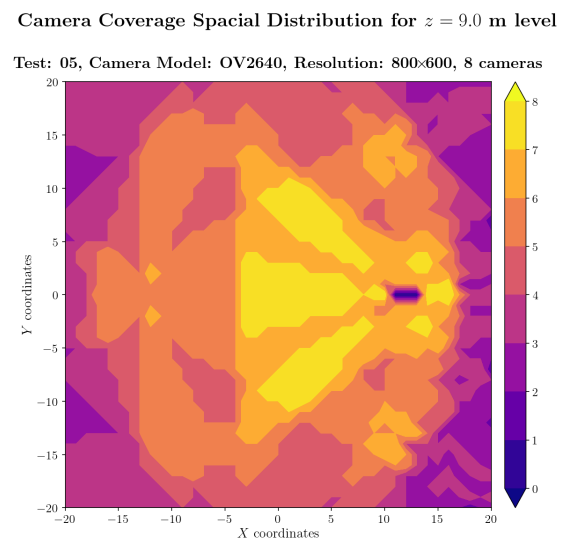
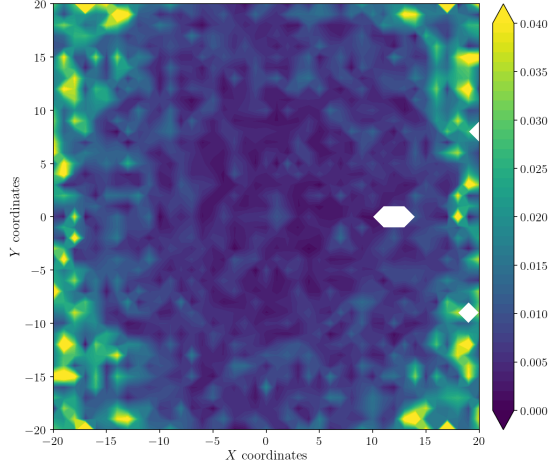


Figure A.732: Camera coverage for test 05 at level $z = 9.000$ m and resolution 800×600.

Estimation Error distribution for $z = 10.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 10.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

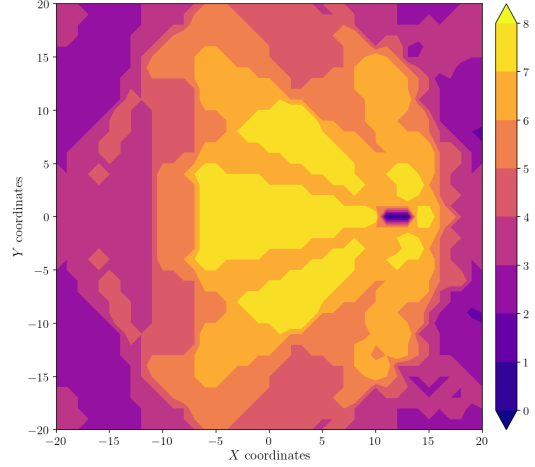
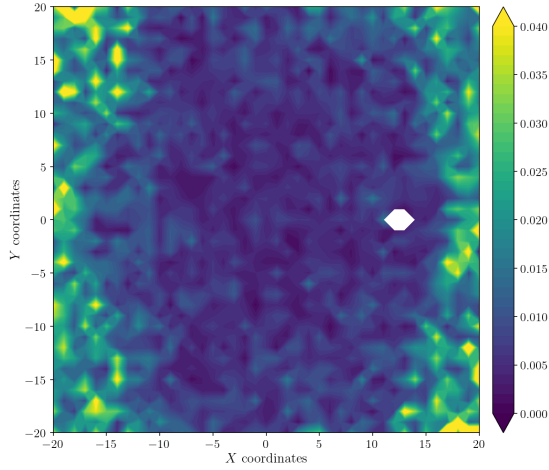


Figure A.733: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 10.000$ m.

Figure A.734: Camera coverage for test 05 at level $z = 10.000$ m and resolution 800×600.

Estimation Error distribution for $z = 11.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras



Camera Coverage Spacial Distribution for $z = 11.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

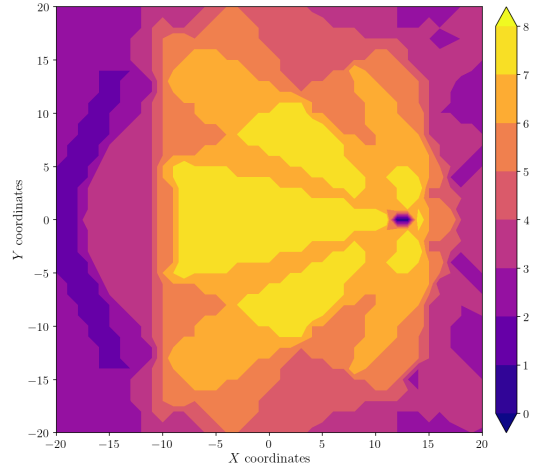


Figure A.735: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 11.000$ m.

Figure A.736: Camera coverage for test 05 at level $z = 11.000$ m and resolution 800×600.

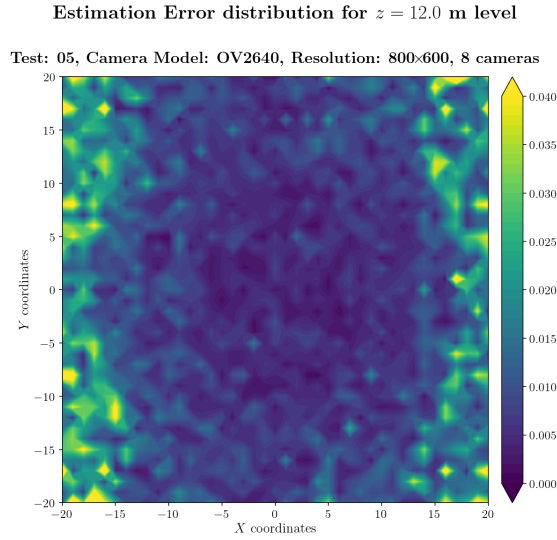


Figure A.737: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 12.000$ m.

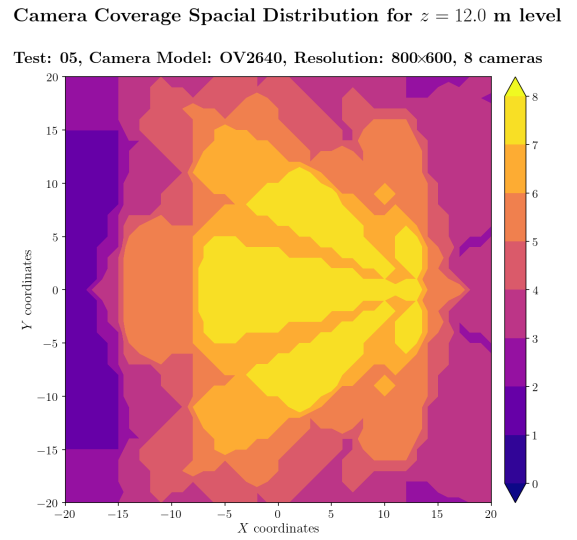


Figure A.738: Camera coverage for test 05 at level $z = 12.000$ m and resolution 800×600.

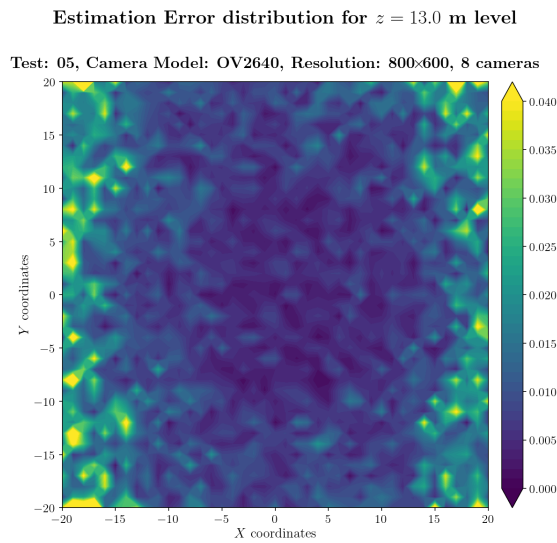


Figure A.739: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 13.000$ m.

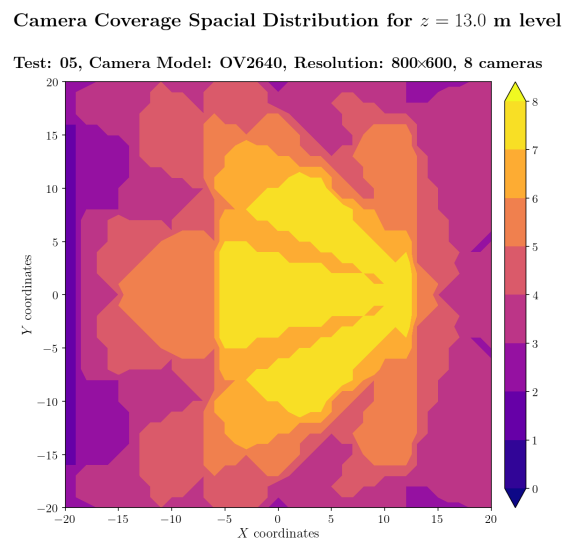


Figure A.740: Camera coverage for test 05 at level $z = 13.000$ m and resolution 800×600.

Estimation Error distribution for $z = 14.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

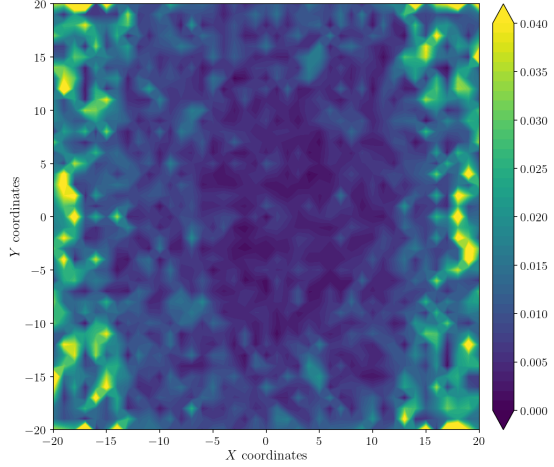


Figure A.741: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 14.000$ m.

Camera Coverage Spacial Distribution for $z = 14.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

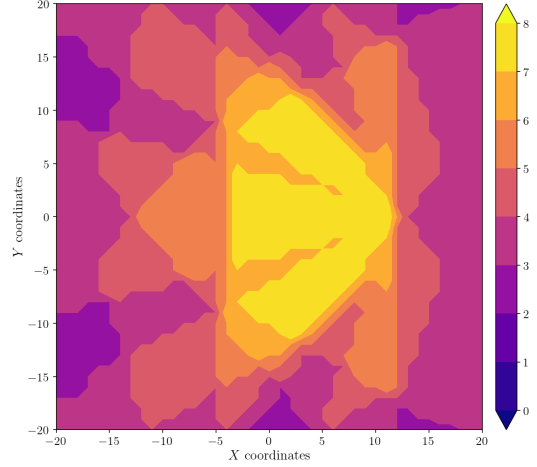


Figure A.742: Camera coverage for test 05 at level $z = 14.000$ m and resolution 800×600.

Estimation Error distribution for $z = 15.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

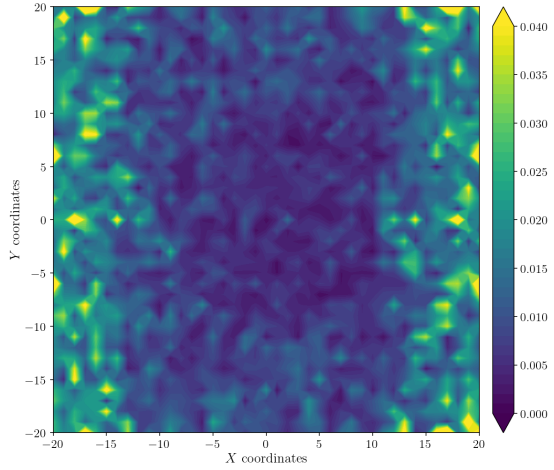


Figure A.743: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 15.000$ m.

Camera Coverage Spacial Distribution for $z = 15.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 800×600, 8 cameras

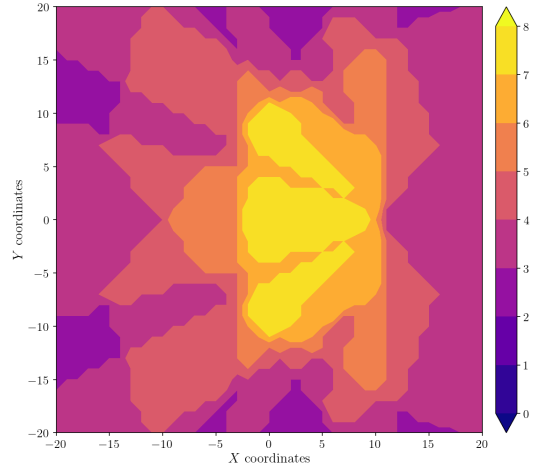


Figure A.744: Camera coverage for test 05 at level $z = 15.000$ m and resolution 800×600.

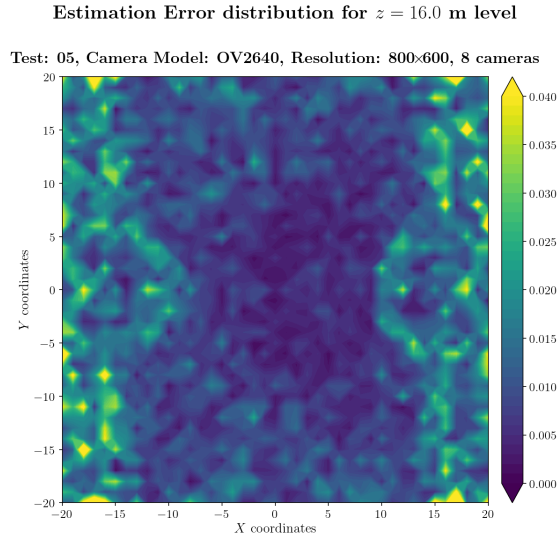


Figure A.745: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 16.000$ m.

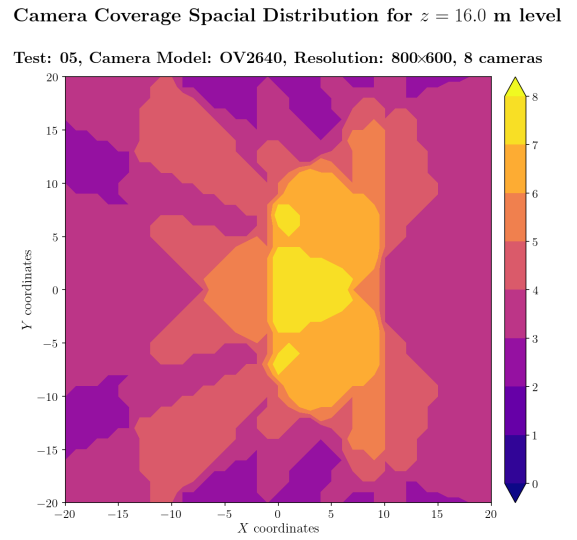


Figure A.746: Camera coverage for test 05 at level $z = 16.000$ m and resolution 800×600.

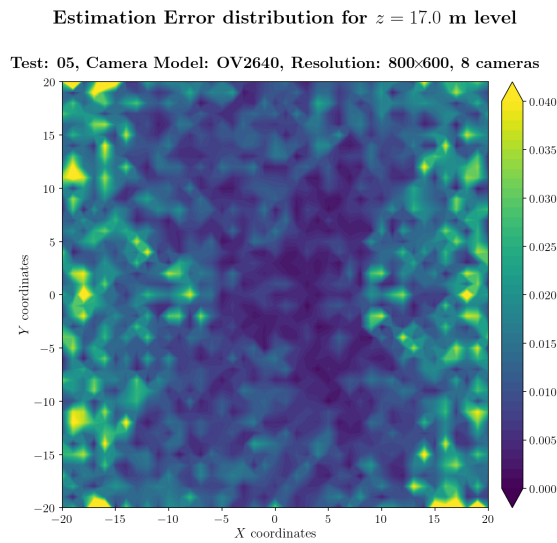


Figure A.747: Estimation error contour map for test 05 with 8 cameras using resolution of 800×600 pixels, at level $z = 17.000$ m.

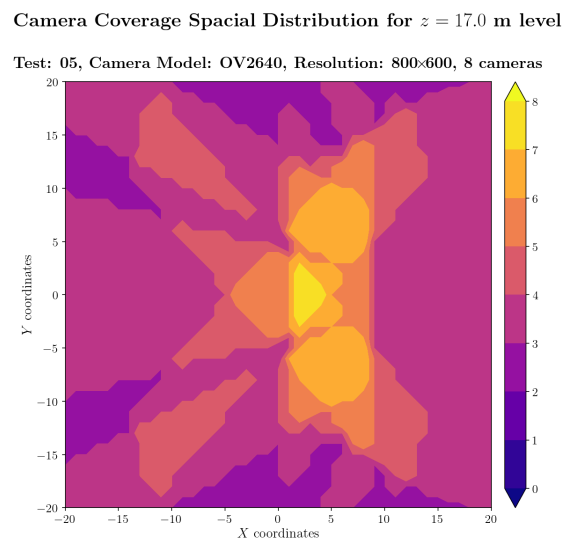


Figure A.748: Camera coverage for test 05 at level $z = 17.000$ m and resolution 800×600.

Error maps for resolution 1024×768

Estimation Error distribution for $z = 0.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

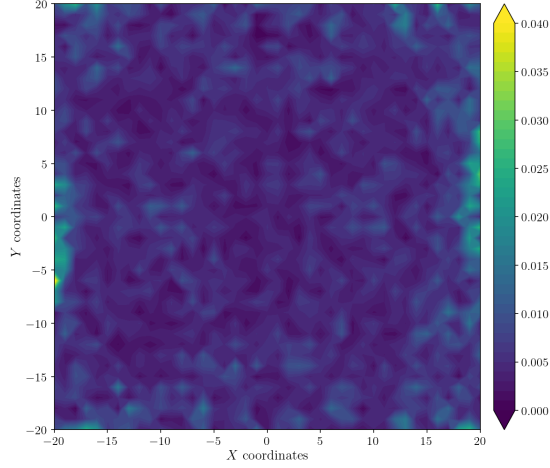


Figure A.749: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 0.000$ m.

Camera Coverage Spacial Distribution for $z = 0.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

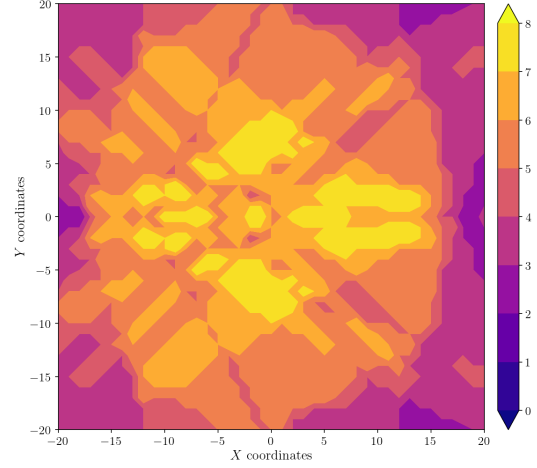


Figure A.750: Camera coverage for test 05 at level $z = 0.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 1.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

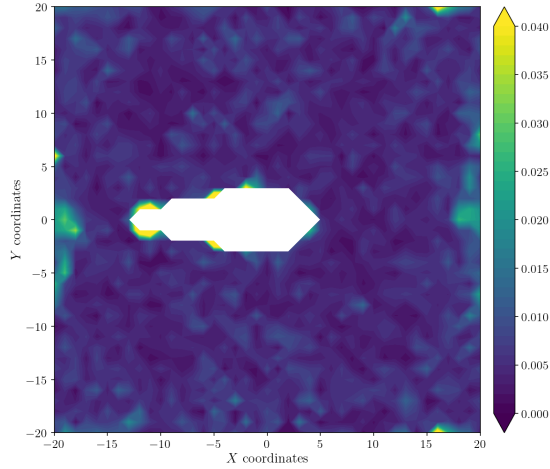


Figure A.751: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 1.000$ m.

Camera Coverage Spacial Distribution for $z = 1.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

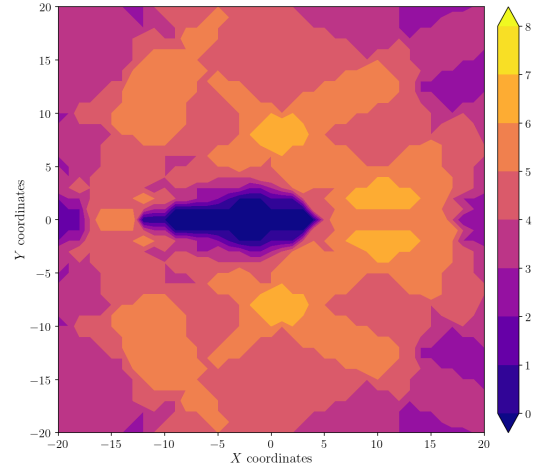


Figure A.752: Camera coverage for test 05 at level $z = 1.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 2.0$ m level

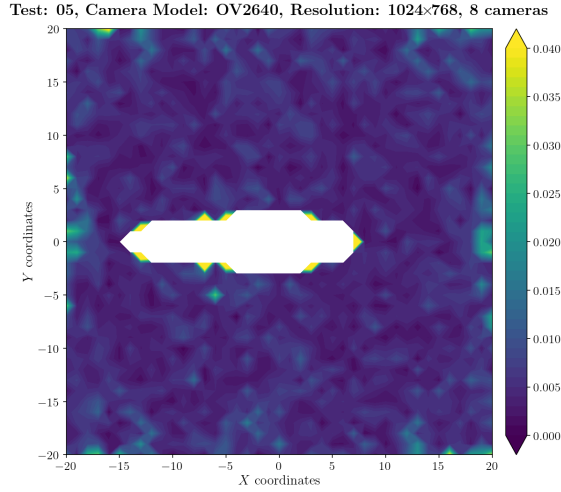


Figure A.753: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 2.000$ m.

Camera Coverage Spacial Distribution for $z = 2.0$ m level

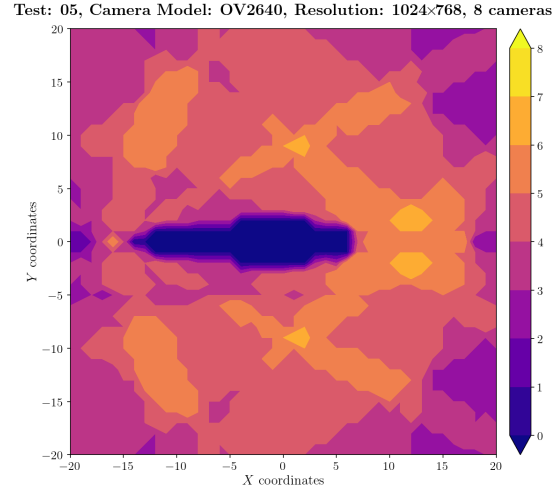


Figure A.754: Camera coverage for test 05 at level $z = 2.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 3.0$ m level

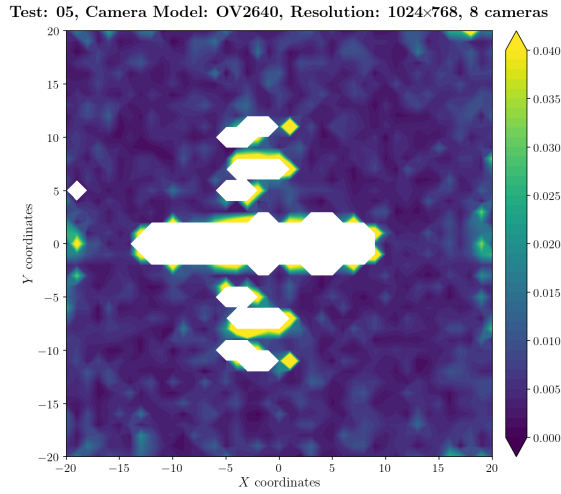


Figure A.755: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 3.000$ m.

Camera Coverage Spacial Distribution for $z = 3.0$ m level

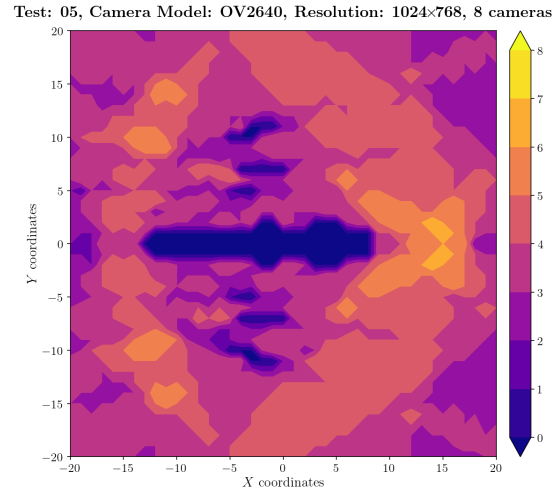
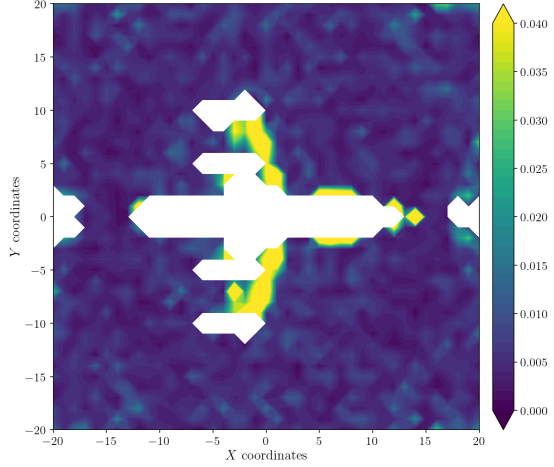


Figure A.756: Camera coverage for test 05 at level $z = 3.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 4.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 4.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

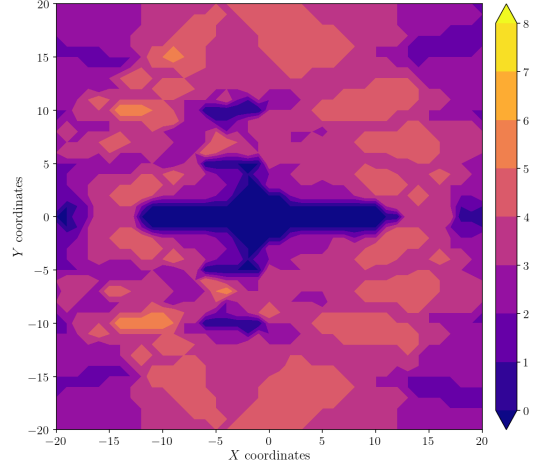
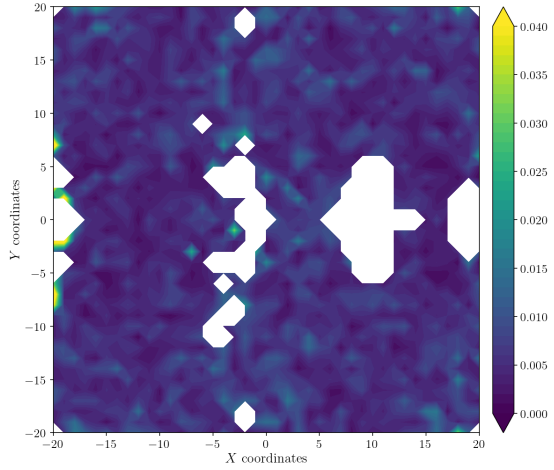


Figure A.757: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 4.000$ m.

Figure A.758: Camera coverage for test 05 at level $z = 4.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 5.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 5.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

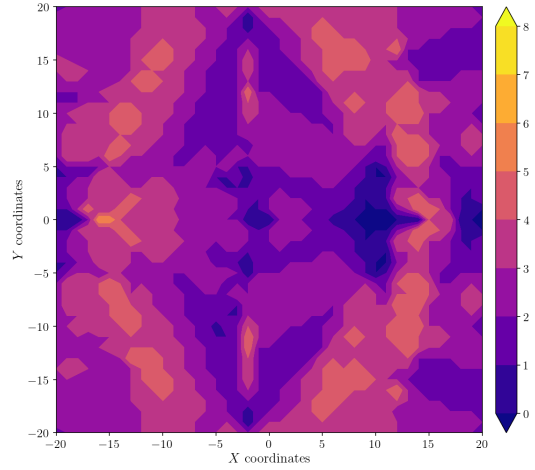


Figure A.759: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 5.000$ m.

Figure A.760: Camera coverage for test 05 at level $z = 5.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 6.0$ m level

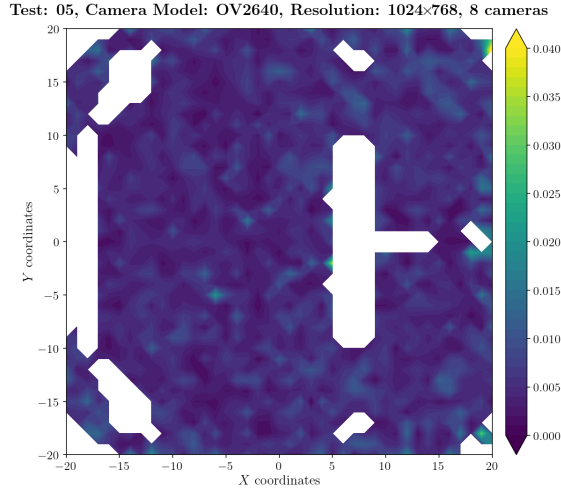


Figure A.761: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 6.000$ m.

Camera Coverage Spacial Distribution for $z = 6.0$ m level

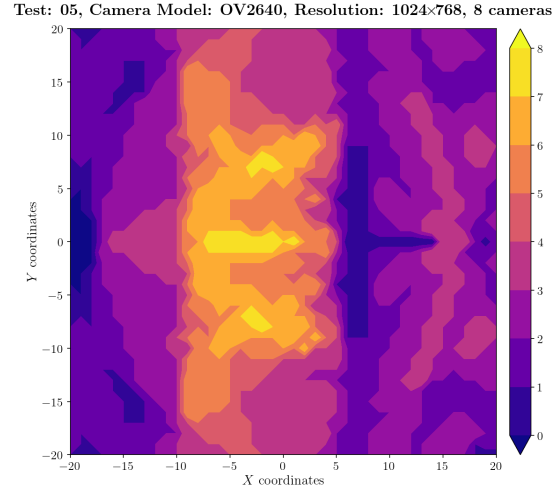


Figure A.762: Camera coverage for test 05 at level $z = 6.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 7.0$ m level

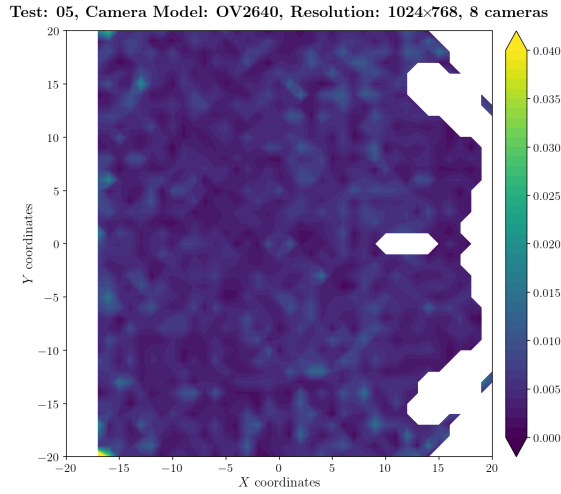


Figure A.763: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 7.000$ m.

Camera Coverage Spacial Distribution for $z = 7.0$ m level

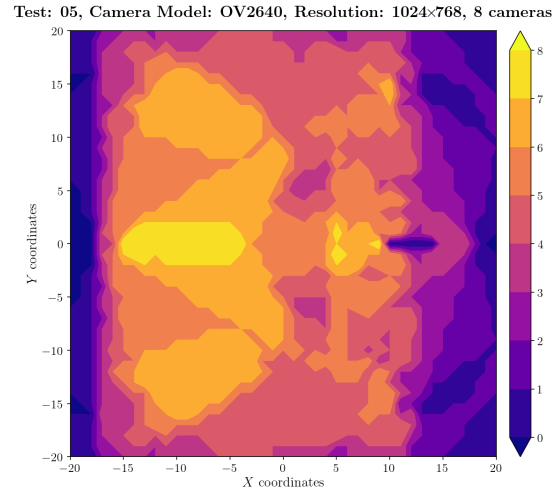
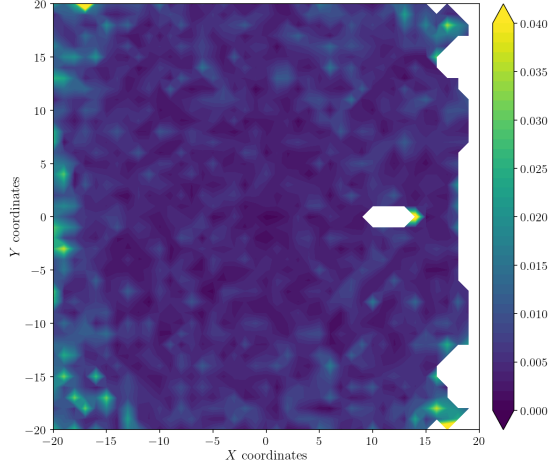


Figure A.764: Camera coverage for test 05 at level $z = 7.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 8.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 8.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

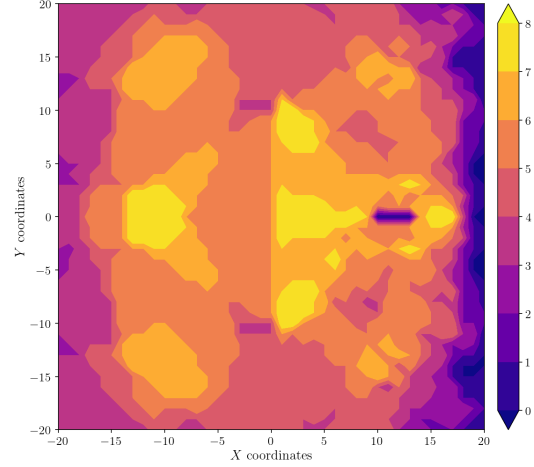
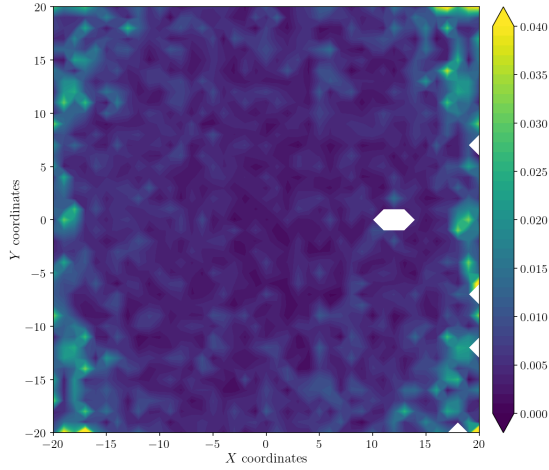


Figure A.765: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 8.000$ m.

Figure A.766: Camera coverage for test 05 at level $z = 8.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 9.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 9.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

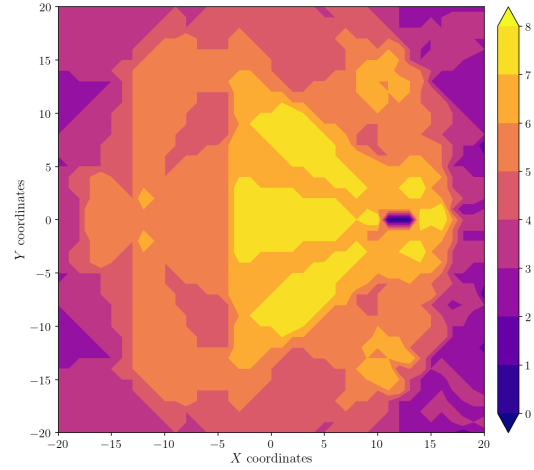


Figure A.767: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 9.000$ m.

Figure A.768: Camera coverage for test 05 at level $z = 9.000$ m and resolution 1024×768.

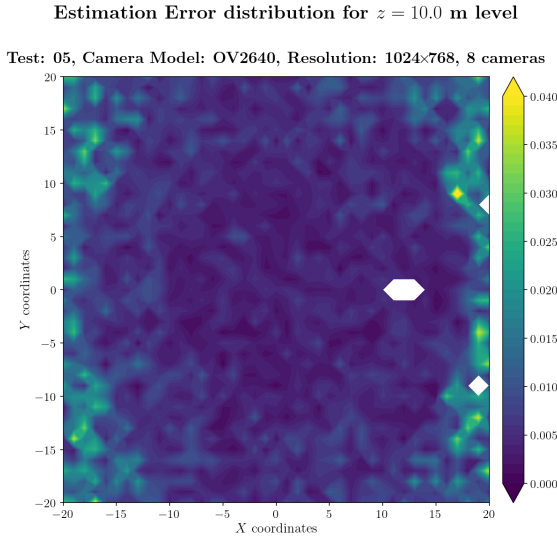


Figure A.769: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 10.000$ m.

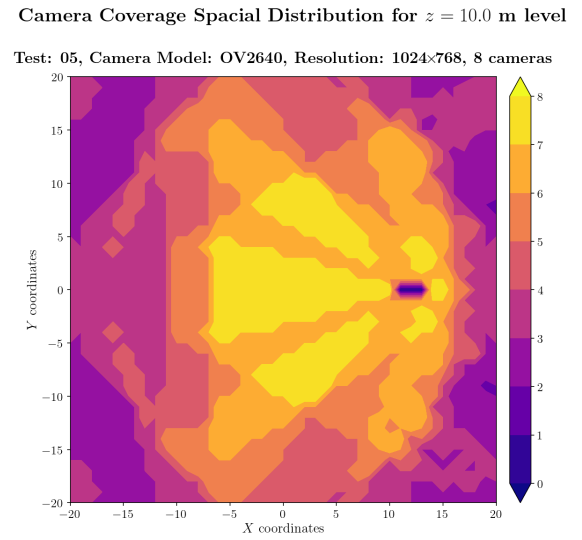


Figure A.770: Camera coverage for test 05 at level $z = 10.000$ m and resolution 1024×768.

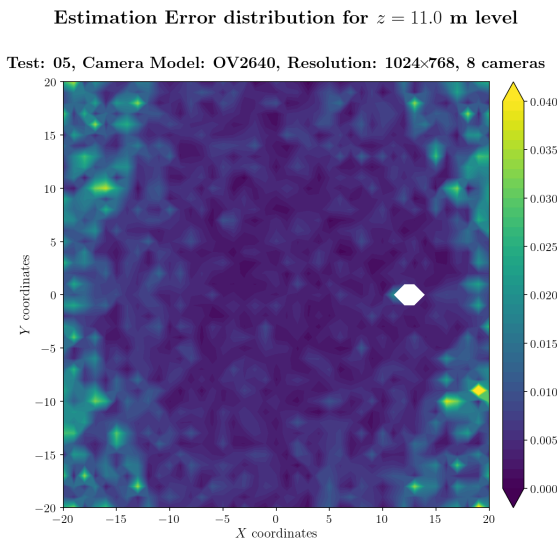


Figure A.771: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 11.000$ m.

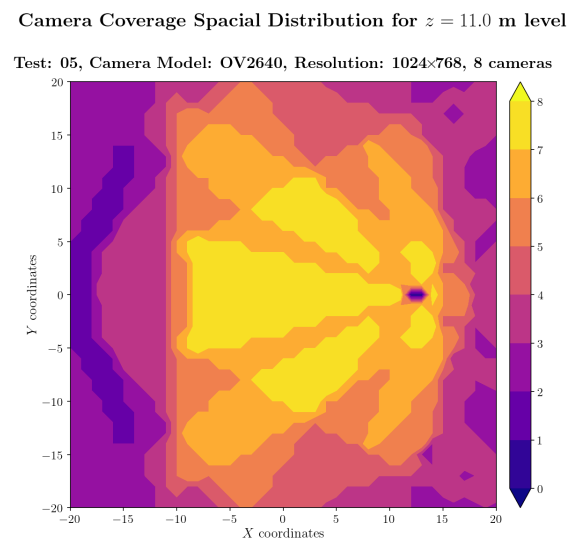
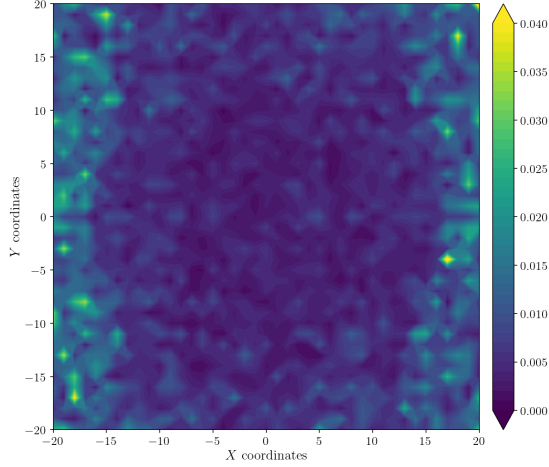


Figure A.772: Camera coverage for test 05 at level $z = 11.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 12.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 12.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

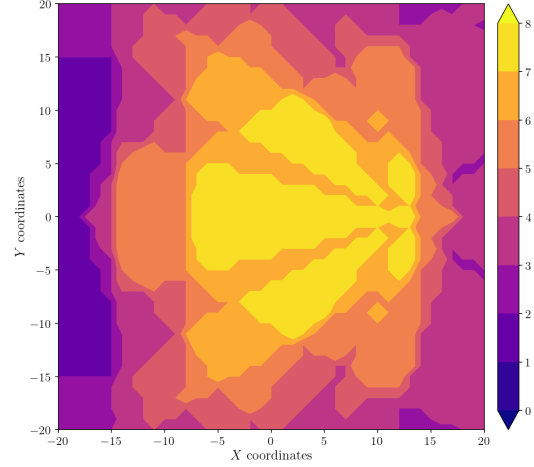
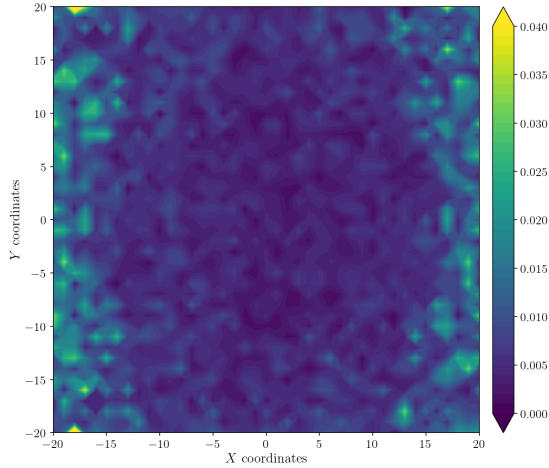


Figure A.773: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 12.000$ m.

Figure A.774: Camera coverage for test 05 at level $z = 12.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 13.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 13.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

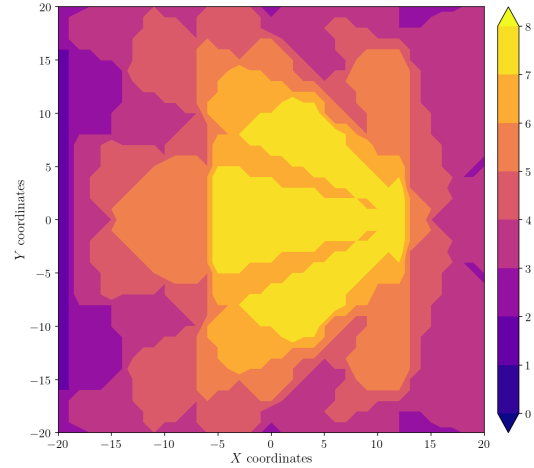


Figure A.775: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 13.000$ m.

Figure A.776: Camera coverage for test 05 at level $z = 13.000$ m and resolution 1024×768.

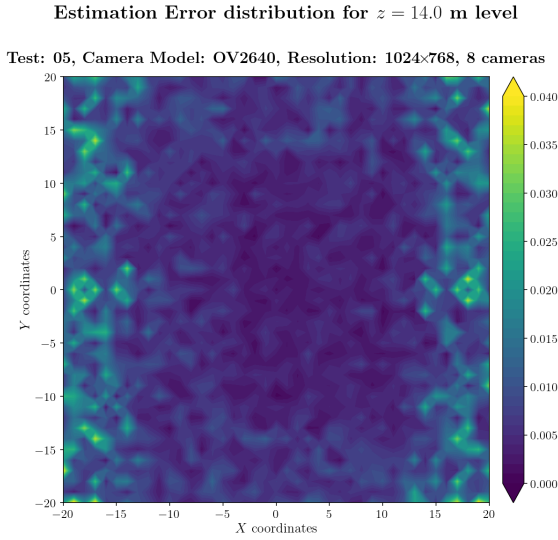


Figure A.777: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 14.000$ m.

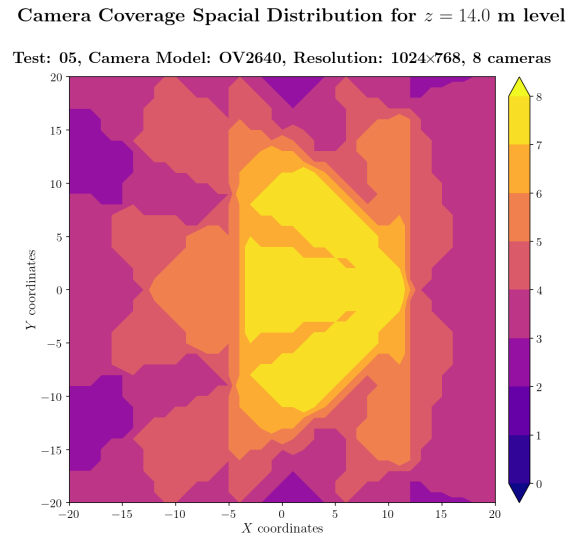


Figure A.778: Camera coverage for test 05 at level $z = 14.000$ m and resolution 1024×768.

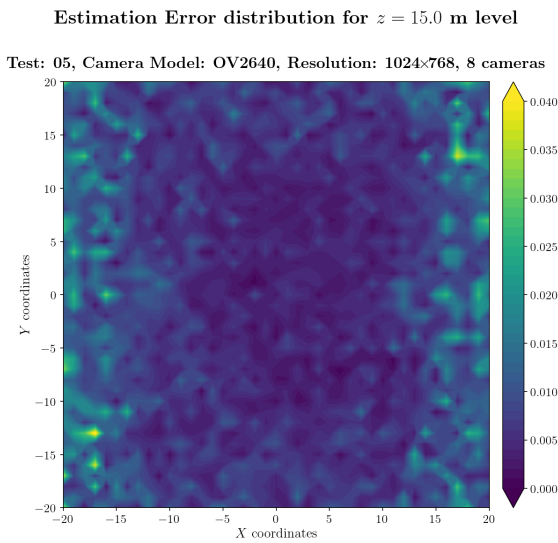


Figure A.779: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 15.000$ m.

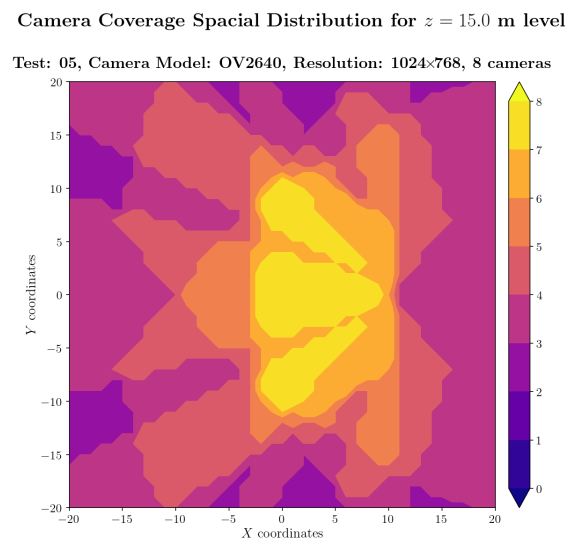
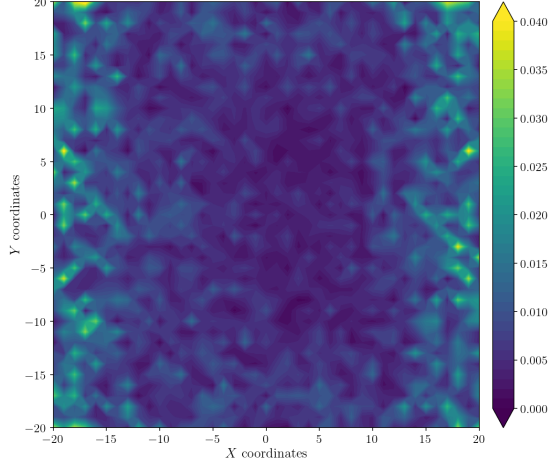


Figure A.780: Camera coverage for test 05 at level $z = 15.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 16.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 16.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

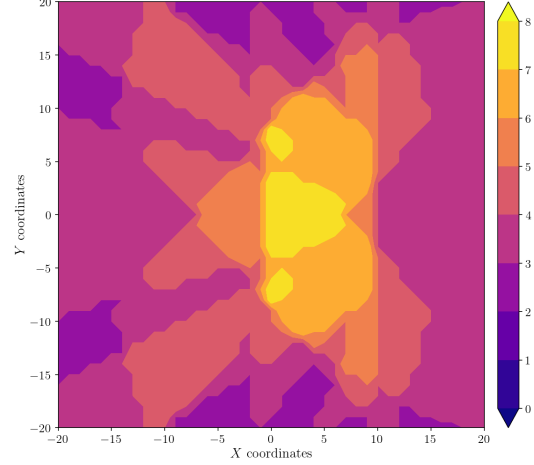
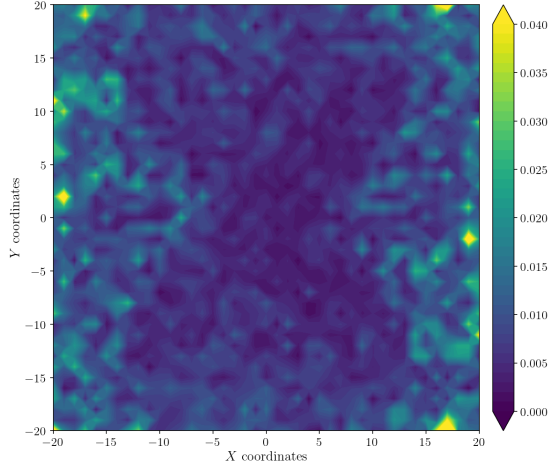


Figure A.781: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 16.000$ m.

Figure A.782: Camera coverage for test 05 at level $z = 16.000$ m and resolution 1024×768.

Estimation Error distribution for $z = 17.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras



Camera Coverage Spacial Distribution for $z = 17.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1024×768, 8 cameras

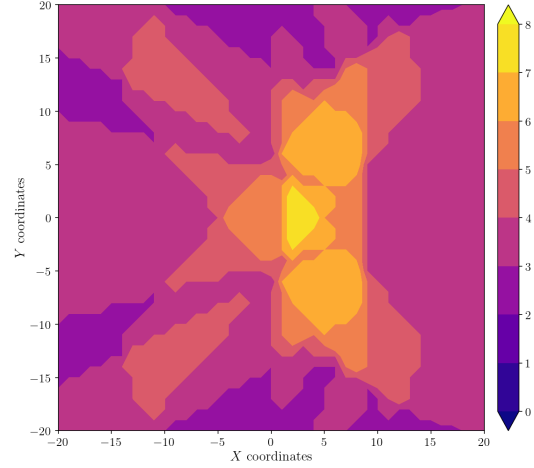


Figure A.783: Estimation error contour map for test 05 with 8 cameras using resolution of 1024×768 pixels, at level $z = 17.000$ m.

Figure A.784: Camera coverage for test 05 at level $z = 17.000$ m and resolution 1024×768.

Error maps for resolution 1600×1200

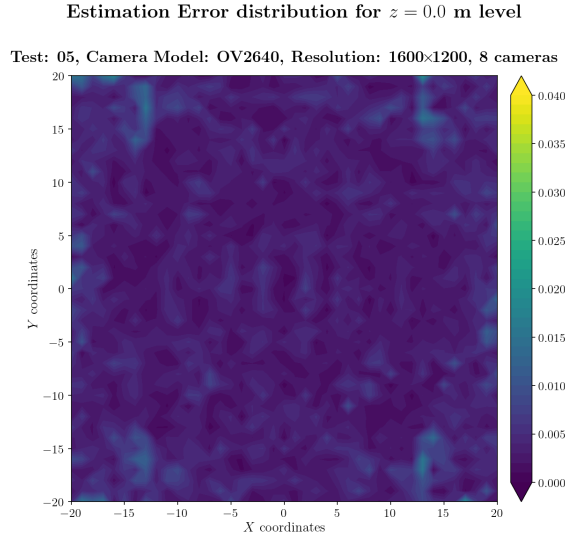


Figure A.785: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 0.000$ m.

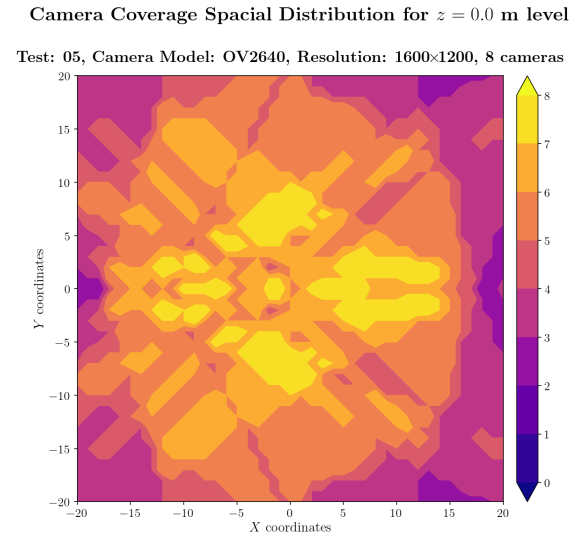


Figure A.786: Camera coverage for test 05 at level $z = 0.000$ m and resolution 1600×1200.

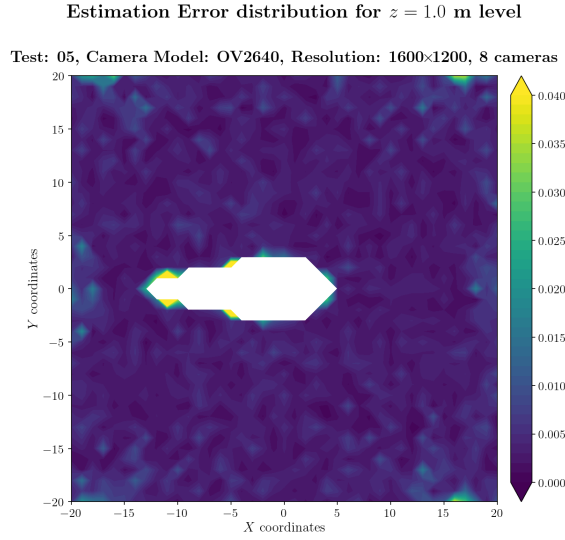


Figure A.787: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 1.000$ m.

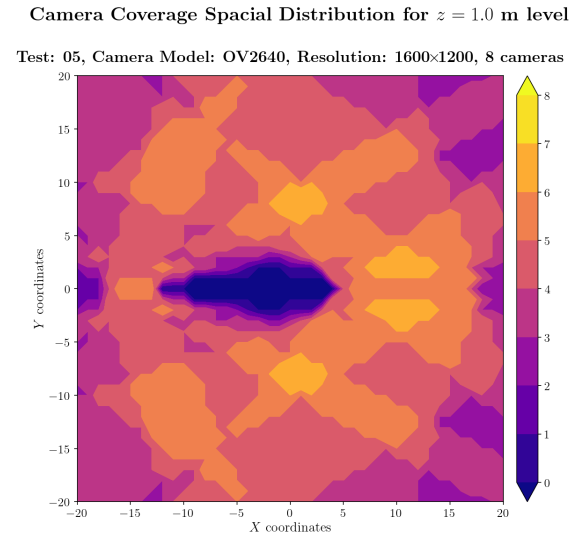


Figure A.788: Camera coverage for test 05 at level $z = 1.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 2.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

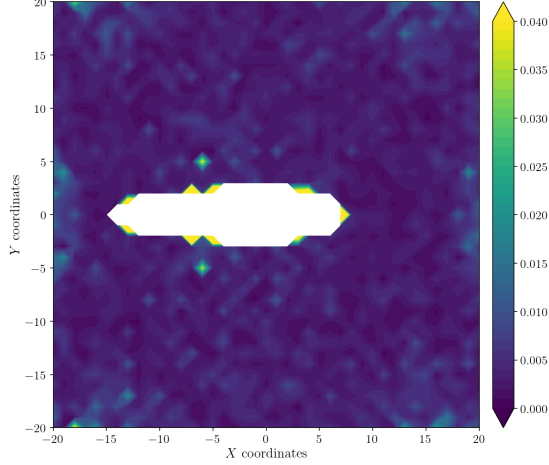


Figure A.789: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 2.000$ m.

Camera Coverage Spacial Distribution for $z = 2.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

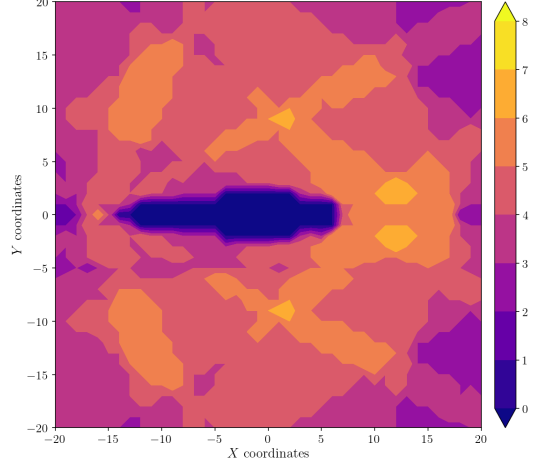


Figure A.790: Camera coverage for test 05 at level $z = 2.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 3.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

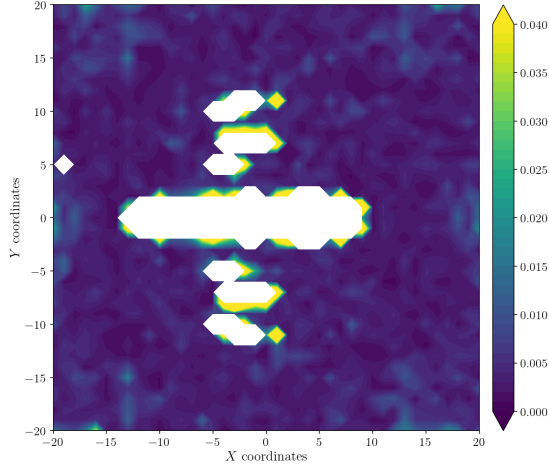


Figure A.791: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 3.000$ m.

Camera Coverage Spacial Distribution for $z = 3.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

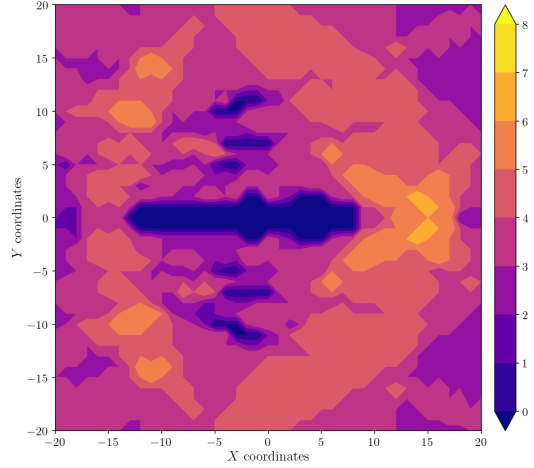


Figure A.792: Camera coverage for test 05 at level $z = 3.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 4.0$ m level

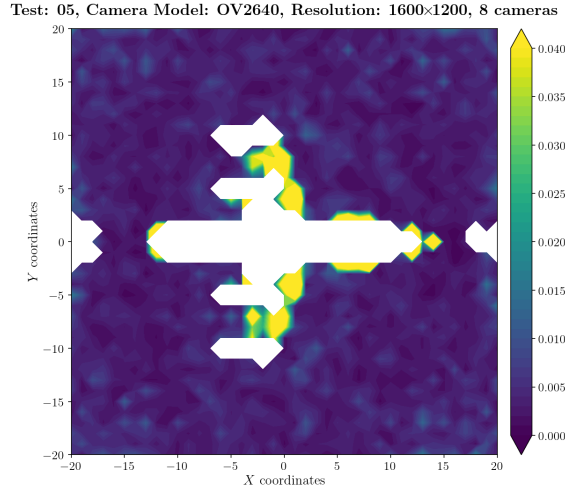


Figure A.793: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 4.000$ m.

Camera Coverage Spatial Distribution for $z = 4.0$ m level

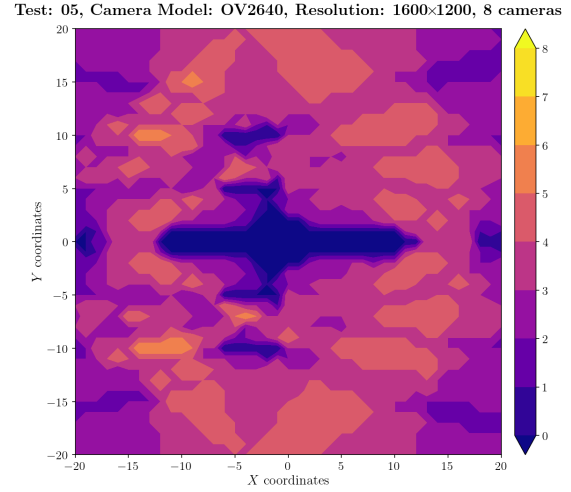


Figure A.794: Camera coverage for test 05 at level $z = 4.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 5.0$ m level

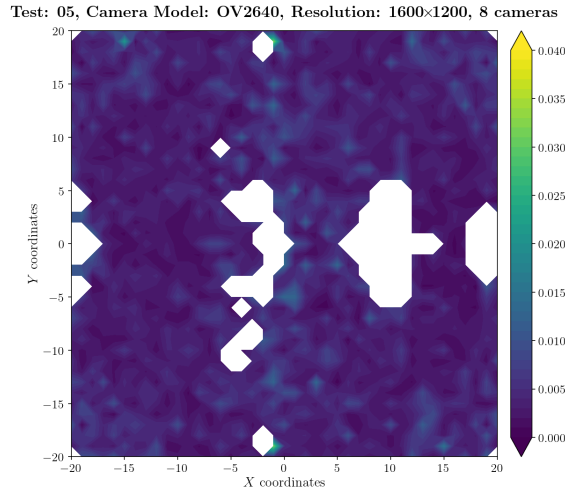


Figure A.795: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 5.000$ m.

Camera Coverage Spatial Distribution for $z = 5.0$ m level

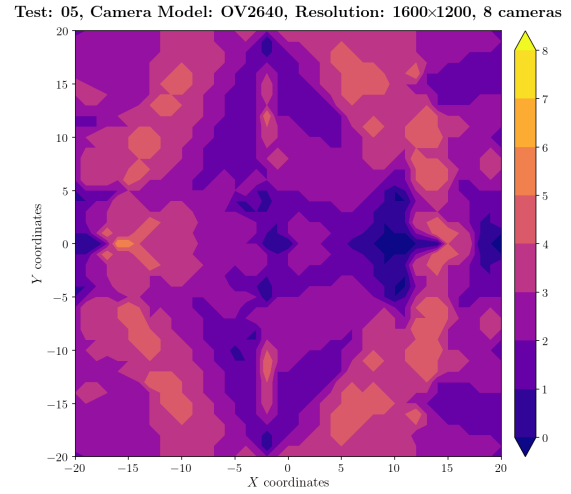
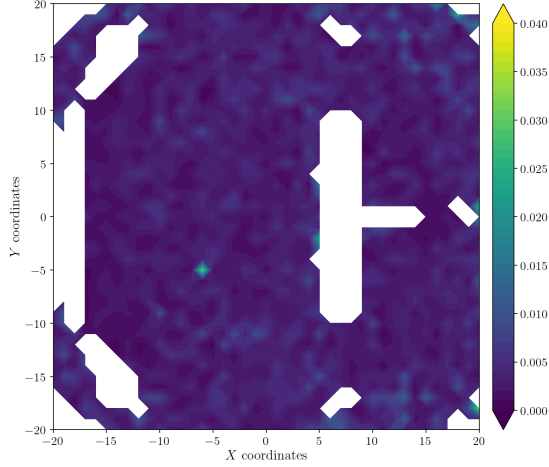


Figure A.796: Camera coverage for test 05 at level $z = 5.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 6.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 6.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

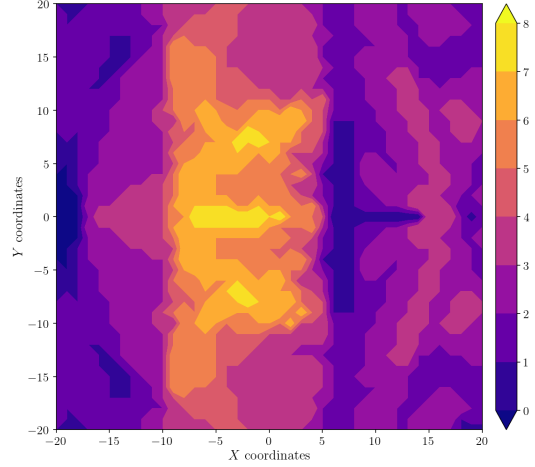
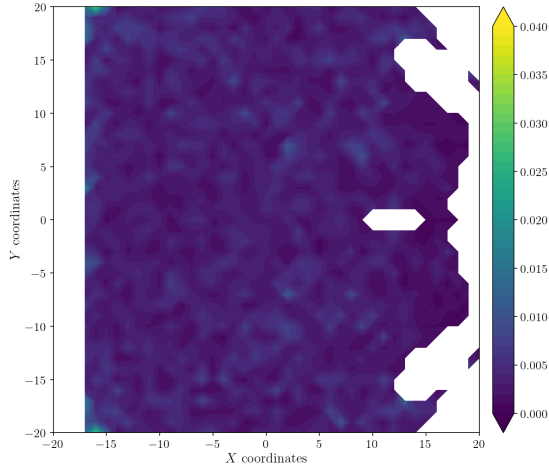


Figure A.797: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 6.000$ m.

Figure A.798: Camera coverage for test 05 at level $z = 6.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 7.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras



Camera Coverage Spacial Distribution for $z = 7.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

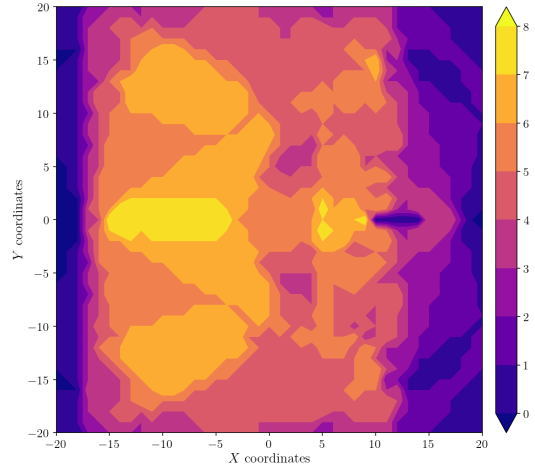


Figure A.799: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 7.000$ m.

Figure A.800: Camera coverage for test 05 at level $z = 7.000$ m and resolution 1600×1200.

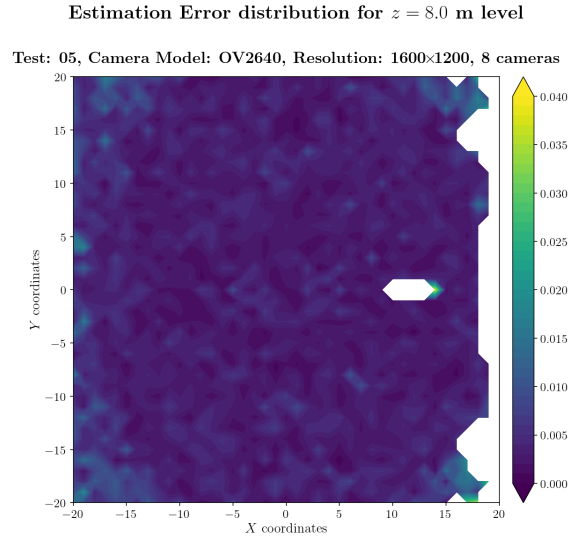


Figure A.801: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 8.000$ m.

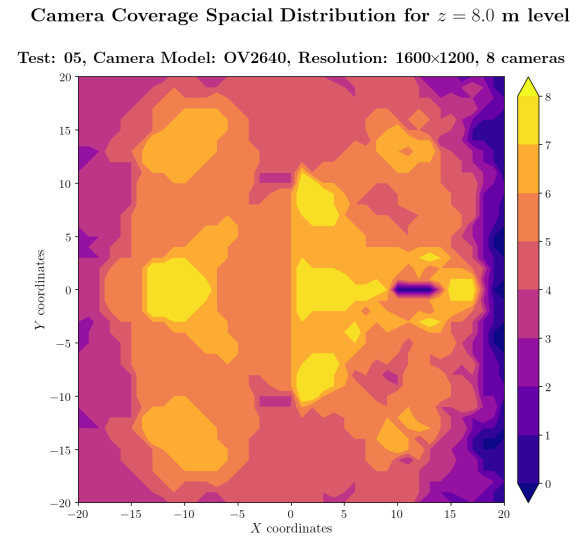


Figure A.802: Camera coverage for test 05 at level $z = 8.000$ m and resolution 1600×1200.

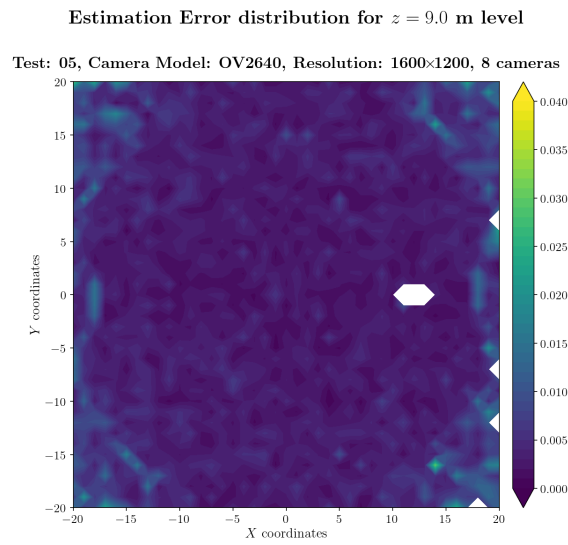


Figure A.803: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 9.000$ m.

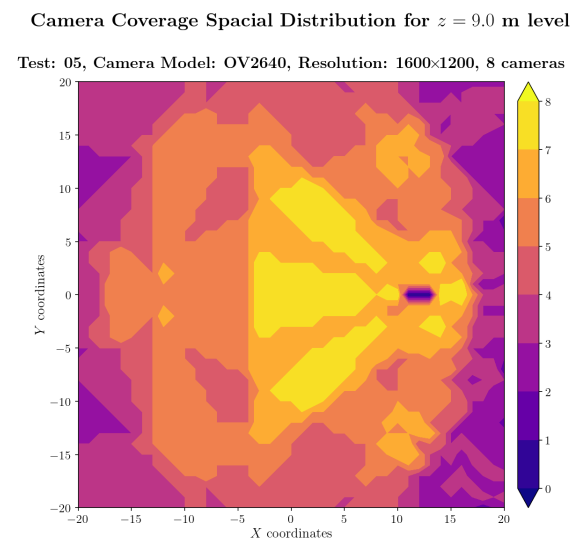


Figure A.804: Camera coverage for test 05 at level $z = 9.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 10.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

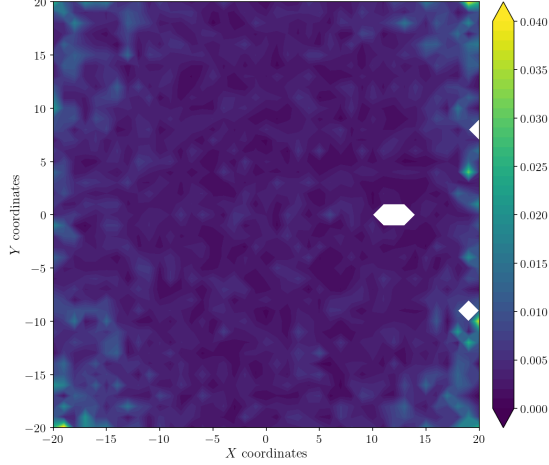


Figure A.805: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 10.000$ m.

Camera Coverage Spacial Distribution for $z = 10.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

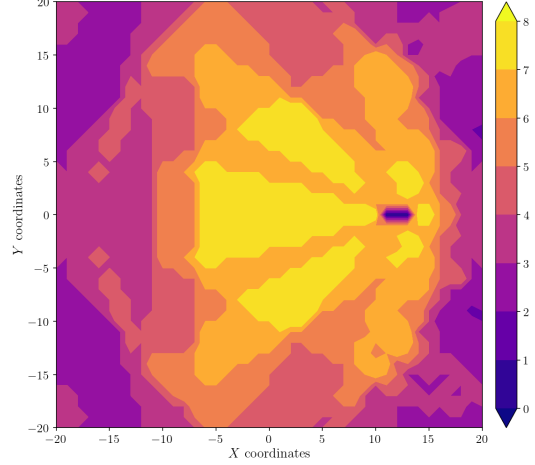


Figure A.806: Camera coverage for test 05 at level $z = 10.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 11.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

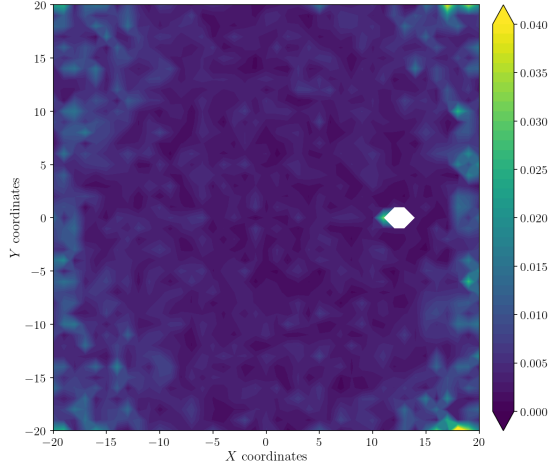


Figure A.807: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 11.000$ m.

Camera Coverage Spacial Distribution for $z = 11.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

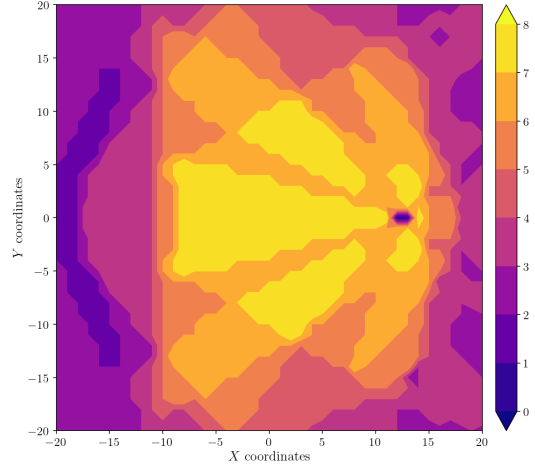


Figure A.808: Camera coverage for test 05 at level $z = 11.000$ m and resolution 1600×1200.

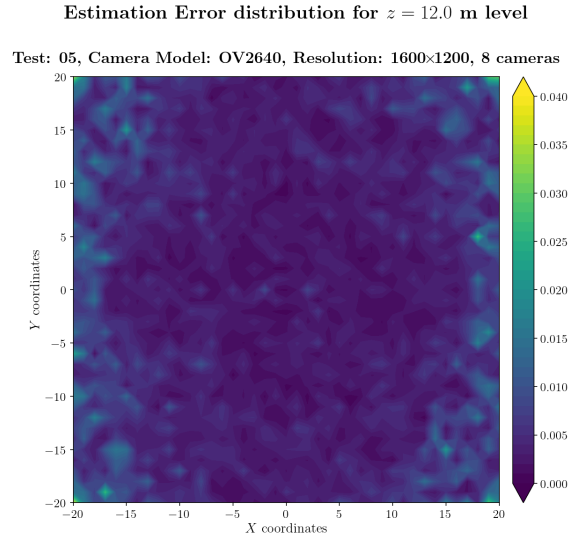


Figure A.809: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 12.000$ m.

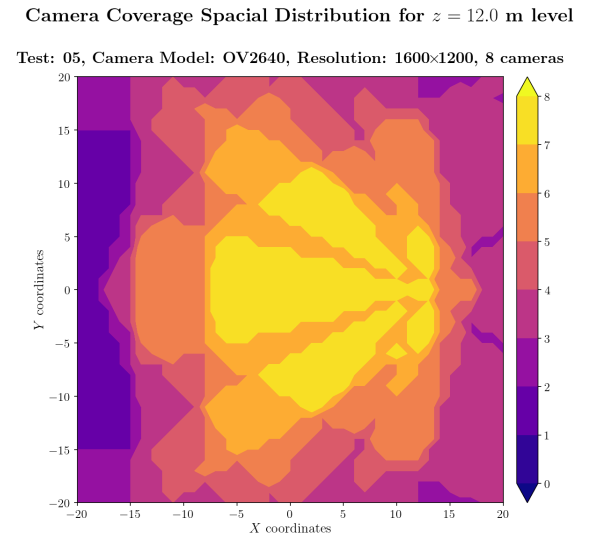


Figure A.810: Camera coverage for test 05 at level $z = 12.000$ m and resolution 1600×1200.

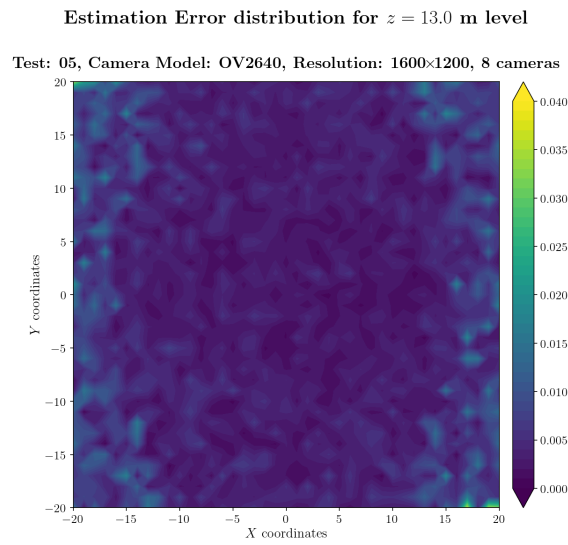


Figure A.811: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 13.000$ m.

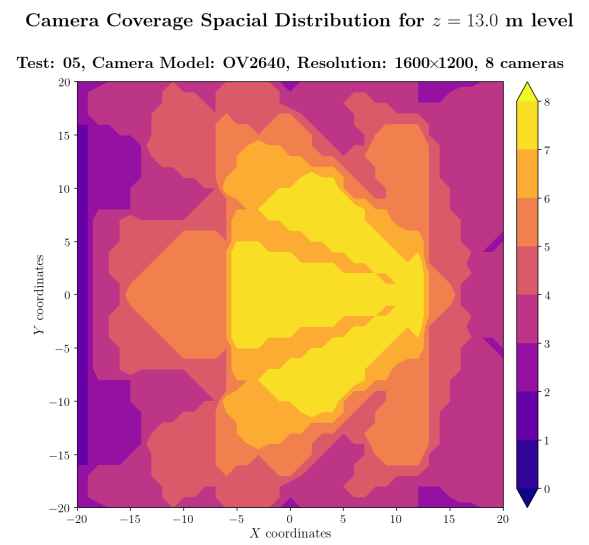


Figure A.812: Camera coverage for test 05 at level $z = 13.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 14.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

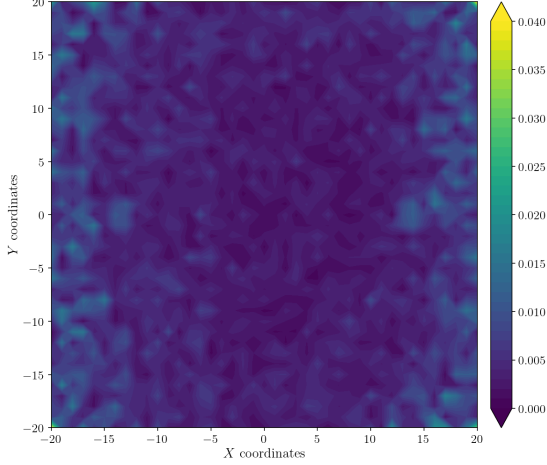


Figure A.813: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 14.000$ m.

Camera Coverage Spacial Distribution for $z = 14.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

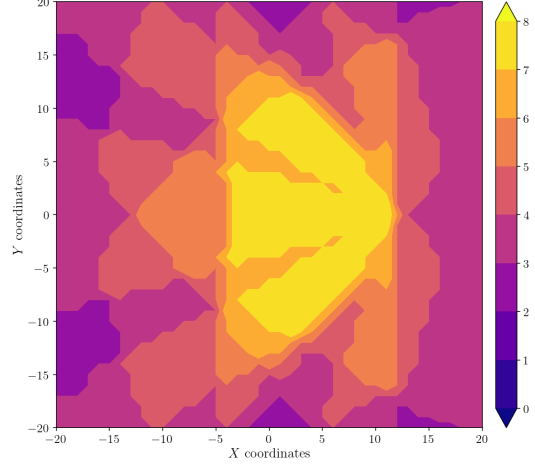


Figure A.814: Camera coverage for test 05 at level $z = 14.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 15.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

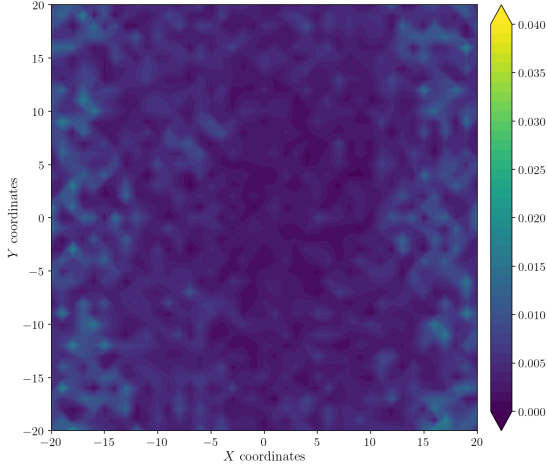


Figure A.815: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 15.000$ m.

Camera Coverage Spacial Distribution for $z = 15.0$ m level

Test: 05, Camera Model: OV2640, Resolution: 1600×1200, 8 cameras

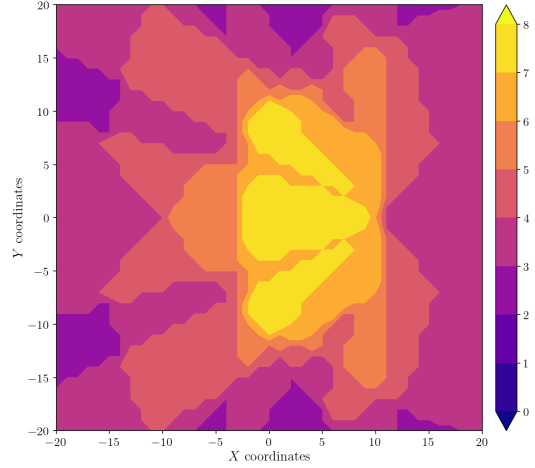


Figure A.816: Camera coverage for test 05 at level $z = 15.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 16.0$ m level

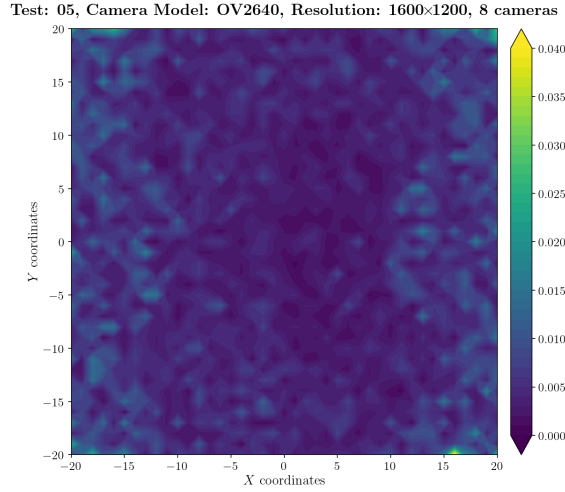


Figure A.817: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 16.000$ m.

Camera Coverage Spatial Distribution for $z = 16.0$ m level

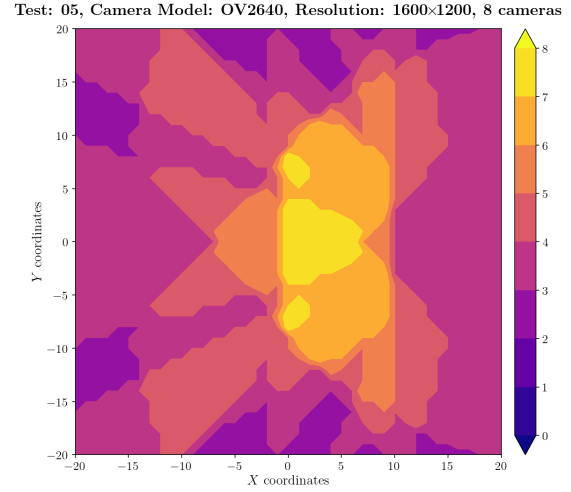


Figure A.818: Camera coverage for test 05 at level $z = 16.000$ m and resolution 1600×1200.

Estimation Error distribution for $z = 17.0$ m level

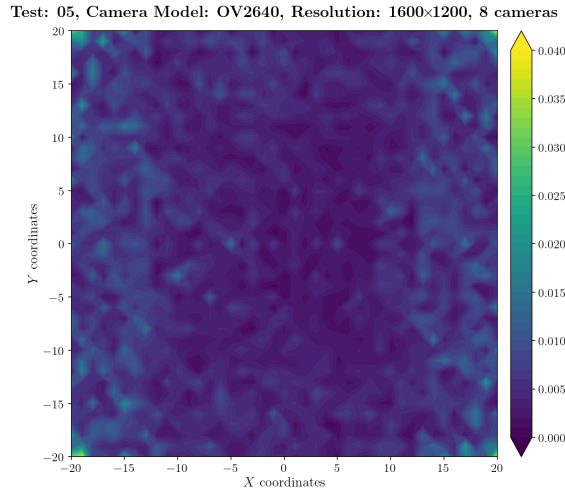


Figure A.819: Estimation error contour map for test 05 with 8 cameras using resolution of 1600×1200 pixels, at level $z = 17.000$ m.

Camera Coverage Spatial Distribution for $z = 17.0$ m level

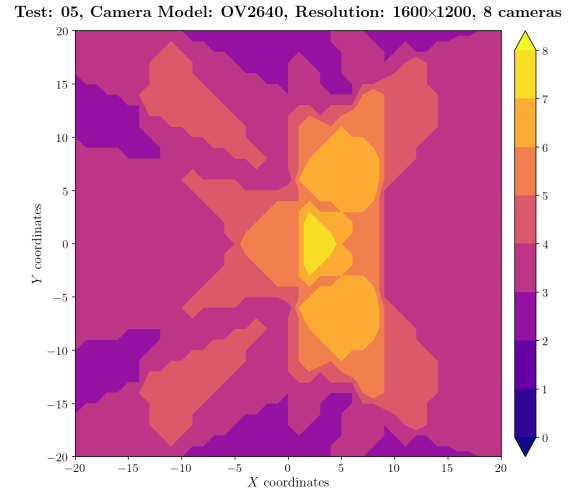


Figure A.820: Camera coverage for test 05 at level $z = 17.000$ m and resolution 1600×1200.