FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Robot Self-Localization in Dynamic Environments

Carlos Miguel Correia da Costa

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Armando Jorge Miranda de Sousa (Ph.D.)

Second Supervisor: Germano Manuel Correia dos Santos Veiga (Ph.D.)

January 26, 2015

© Carlos Miguel Correia da Costa, 2015

Robot Self-Localization in Dynamic Environments

Carlos Miguel Correia da Costa

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Doctor Luis Teixeira
External Examiner: Doctor Artur Pereira
Supervisor: Doctor Armando Sousa
Second Supervisor: Doctor Germano Veiga

January 26, 2015

Abstract

Mobile robot platforms capable of operating safely and accurately in dynamic environments can have a multitude of applications, ranging from simple delivery tasks to advanced assembly operations. These abilities rely heavily on a robust navigation stack, which requires a stable and accurate localization system.

This dissertation proposes an efficient, modular, extensible and easy to configure 3/6 Degrees of Freedom (DoF) localization system, capable of operating on a wide range of mobile robot platforms and environments. It is able to reliably estimate the global position using feature matching and is capable of achieving high accuracy pose tracking with a range of point cloud registration algorithms. It can use several point cloud sensing devices (such as LIght Detection And Ranging (LIDAR) or RGB-D cameras) and by using reference maps it does not require ambient modification. Moreover, it can update the localization map at runtime and dynamically adjust its operation rate based on the predicted robot velocity in order to use the minimum amount of hardware resources. It also offers a detailed analysis of each pose estimation, providing information about the percentage of registered inliers, the root mean square error of the inliers, the angular distribution of the inliers and outliers, the pose corrections that were performed in relation to the expected position and in case of initial pose estimation it also gives the distribution of the acceptable initial poses, which can be very valuable information for a navigation supervisor when the robot is in ambiguous areas that are very similar in different parts of the known environment.

The Robot Operating System (ROS) implementation was tested in several dynamic indoor environments using three mobile robot platforms equipped with LIDARs and RGB-D cameras (two of them are well know public datasets). Overall tests using sensor data from simulation and retrieved from the robot platforms performed in a high end laptop with an Intel Core i7 3630QM processor, 16GB DDR3 of memory and NVIDIA GTX680M graphics card, demonstrated high accuracy in complex dynamic environments, with less than 1-2 cm in translation error and less than 1-3 degrees in rotation error (in 3 and 6 DoF respectively). Execution times ranged from 5 to 35 milliseconds in both 3 DoF and 6 DoF.

The sub centimeter accuracy achieved by the proposed localization system along with the dynamic map update capability and the need of no artificial landmarks will allow the fast deployment of mobile robot platforms capable of operating safely and accurately in cluttered environments. Moreover, the robustness to dynamic objects will grant the possibility to use robots as coworkers, helping humans perform their work more efficiently and thus reducing the overall production costs.

Resumo

Plataformas móveis robóticas capazes de operar com precisão e de forma segura em ambientes dinâmicos têm um alargado espetro de aplicações, desde simples entregas de objetos até operações complexas de montagem. Para atingir estes requisitos de operação é necessário um sistema de navegação robusto, que por sua vez requer um módulo de localização preciso e estável.

Esta dissertação descreve um sistema de localização para 3/6 graus de liberdade (DoF), que é eficiente, modular, extensível e fácil de configurar, capaz de operar num alargado conjunto de plataformas móveis e ambientes. É capaz de estimar a posição inicial de um robô usando métodos de associação de características geométricas e consegue seguir a sua pose com alta precisão através de algoritmos de registo de nuvens de pontos. A sua implementação consegue tirar partido de vários sensores laser e câmaras RGB-D e não necessita de marcadores artificiais ou modificação do ambiente. Possui ainda a capacidade de atualizar o mapa incrementalmente e ajustar a sua frequência de funcionamento de acordo com a velocidade do robô de forma a usar o mínimo de recursos computacionais possível. Para facilitar a avaliação da qualidade da localização para operações críticas, cada estimativa da pose do robô é acompanhada com a análise do registo da nuvem de pontos, contendo informação acerca da percentagem de pontos corretamente registados, a raiz quadrada do erro quadrático médio, a distribuição angular dos pontos classificados como pertencentes e não pertencentes ao mapa de referência, as correções aplicadas à estimativa da pose e no caso de ser efetuada localização global também é disponibilizada a distribuição das poses iniciais aceitáveis, o que pode ser informação bastante útil para um supervisor de navegação quando o robô está em posições ambíguas do ambiente nas quais existe geometria semelhante em sítios diferentes do mapa.

A implementação em ROS foi testada em vários ambientes dinâmicos recorrendo a duas plataformas móveis equipadas com LIDARs e câmaras RGB-D. Os resultados obtidos usando dados de simulação e recolhidos das plataformas robóticas realizados num portátil com CPU Intel Core i7 3630QM, 16GB DDR3 de memória e placa gráfica NVIDIA GTX680M demonstraram que o sistema consegue fazer a estimativa da pose do robô com um erro de translação inferior a 1-2 centímetros e um erro de rotação abaixo de 1-3 graus (em 3 e 6 DoF respectivamente). Os tempos de execução oscilaram entre 5 e 35 milissegundos para uma configuração 3 DoF e 6 DoF.

A alta precisão disponibilizada pelo sistema de localização proposto em conjunto com a sua capacidade para atualizar o mapa incrementalmente e de não necessitar marcadores artificiais, irá permitir o desenvolvimento de plataformas robóticas móveis capazes de operar em ambientes não estruturados. Por outro lado, a sua robustez em relação a objetos dinâmicos abre a possibilidade dos robôs colaborarem com humanos para melhorar a produtividade global de uma dada tarefa e assim reduzir os custos de produção.

Acknowledgments

I would like to express my gratitude to my supervisors and the INESC team, for their helpful contributions. Their experience and expertise significantly improved the quality of this dissertation.

I am also very grateful for the knowledge and experiences that I gather from my friends, teachers and colleagues over the years and for the brilliant work developed by the ROS and PCL community.

And of course, to my family, for their continuous support.

Carlos Miguel Correia da Costa

"Perfection is achieved, not when there is nothing more to add,
but when there is nothing left to take away."

Antoine de Saint-Exupéry

Contents

1 Introduction **1**
- 1.1 Context . . . 1
- 1.2 Project . . . 1
- 1.3 Motivation and objectives . . . 2
- 1.4 Contributions . . . 2
- 1.5 Dissertation outline . . . 2

2 Localization methods **3**
- 2.1 Proprioceptive methods . . . 3
  - 2.1.1 Odometry . . . 3
  - 2.1.2 Dead reckoning . . . 4
- 2.2 Exteroceptive methods . . . 4
  - 2.2.1 Trilateration methods . . . 4
    - 2.2.1.1 Global Navigation Satellite System (GNSS) . . . 4
    - 2.2.1.2 Differential Global Positioning System (DGPS) . . . 5
    - 2.2.1.3 Assisted Global Positioning System (AGPS) . . . 5
    - 2.2.1.4 Signal strength geolocation methods . . . 6
  - 2.2.2 Celestial navigation . . . 6
  - 2.2.3 Landmark methods . . . 7
  - 2.2.4 Point cloud methods . . . 7
  - 2.2.5 Probabilistic methods . . . 8
    - 2.2.5.1 Monte Carlo Localization (MCL) . . . 8
    - 2.2.5.2 Kalman filters . . . 9
    - 2.2.5.3 Perfect Match . . . 10
  - 2.2.6 Simultaneous Localization And Mapping (SLAM) . . . 10
- 2.3 Summary . . . 11

3 Relevant software / hardware technologies **13**
- 3.1 ROS . . . 13
  - 3.1.1 Architecture . . . 13
    - 3.1.1.1 Nodes . . . 14
    - 3.1.1.2 Nodelets . . . 15
    - 3.1.1.3 Topics . . . 15
    - 3.1.1.4 Services . . . 15
    - 3.1.1.5 Actions . . . 16
  - 3.1.2 Build system . . . 16
  - 3.1.3 Development tools . . . 16
    - 3.1.3.1 Graphical User Interface tools . . . 16

3.1.3.2 Command line tools . . . 17
3.2 PCL . . . 17
3.3 Gazebo . . . 18
3.4 Point cloud acquisition . . . 19
3.4.1 Structured light methods . . . 19
3.4.2 Time of Flight methods . . . 20
3.4.2.1 Light waves . . . 20
3.4.2.2 Radio waves . . . 21
3.4.2.3 Sound waves . . . 22
3.4.3 Stereo vision . . . 23

4 Localization system **25**
4.1 Overview . . . 25
4.2 Pipeline configuration . . . 27
4.3 Reference map . . . 27
4.4 Point cloud assembly . . . 28
4.5 Point cloud search data structures . . . 28
4.5.1 Voxel grids . . . 29
4.5.2 Octrees . . . 29
4.5.3 k-d trees . . . 30
4.6 Filtering and down sampling . . . 30
4.6.1 Point cloud downsampling . . . 31
4.6.1.1 Voxel grid sampling . . . 31
4.6.1.2 Random sampling . . . 31
4.6.1.3 Covariance sampling . . . 31
4.6.2 Outlier removal . . . 32
4.6.2.1 Distance filter . . . 32
4.6.2.2 Passthrough filter . . . 32
4.6.2.3 Radius outlier removal filter . . . 33
4.6.2.4 Statistical outlier removal filter . . . 33
4.6.3 Surface and object reconstruction and resampling . . . 33
4.7 Normal estimation . . . 34
4.7.1 Line normal estimation . . . 35
4.7.2 Surface normal estimation . . . 35
4.8 Keypoint detection . . . 35
4.9 Keypoint description . . . 36
4.10 Cloud registration . . . 36
4.10.1 Initial alignment with keypoints descriptor matching . . . 36
4.10.2 Final alignment with point cloud error minimization . . . 37
4.11 Outlier detection . . . 37
4.12 Localization validation . . . 38
4.13 Dynamic map update . . . 38

5 Localization system evaluation **39**
5.1 Testing platforms . . . 39
5.1.1 Jarvis platform . . . 39
5.1.2 Pioneer 3-DX platform . . . 40
5.1.3 Guardian platform . . . 41
5.2 Testing environments . . . 41

5.2.1 Jarvis in RoboCup field . . . 42
5.2.2 Guardian in structured environment . . . 42
5.2.3 Pioneer in industrial hall . . . 44
5.2.4 Kinect in flying arena . . . 44
5.3 3 DoF localization system tests . . . 45
5.3.1 Overview . . . 45
5.3.2 Laser assembly with spherical linear interpolation . . . 48
5.3.3 Point cloud preprocessing . . . 50
5.3.4 SLAM . . . 50
5.3.5 Point cloud registration . . . 54
5.3.6 Translation and rotation errors . . . 56
5.3.7 Computation time . . . 57
5.4 6 DoF localization system tests . . . 58
5.4.1 Overview . . . 58
5.4.2 Point cloud preprocessing . . . 58
5.4.3 SLAM . . . 60
5.4.4 Point cloud registration . . . 62
5.4.5 Translation and rotation errors . . . 63
5.4.6 Computation time . . . 64

6 Conclusions **65**
6.1 Main contributions . . . 65
6.2 Future work . . . 66

A 3 DoF localization system results **67**
A.1 Jarvis robot tests . . . 68
A.1.1 Rounded path using the Jarvis robot at 5 cm/s . . . 68
A.1.2 Rounded path using the Jarvis robot at 30 cm/s . . . 84
A.1.3 Complex path using the Jarvis robot at 5 cm/s . . . 100
A.1.4 Complex path using the Jarvis robot at 50-30-50-10 cm/s . . . 116
A.1.5 Complex path using the Jarvis robot at 5 cm/s without laser spherical linear interpolation . . . 132
A.1.6 Complex path using the Jarvis robot at 50-30-50-10 cm/s without laser spherical linear interpolation . . . 143
A.2 Pioneer robot tests . . . 154
A.2.1 360 path using the Pioneer robot at 23 cm/s . . . 154
A.2.2 First SLAM path using the Pioneer robot at 26 cm/s . . . 170
A.2.3 Second SLAM path using the Pioneer robot at 19 cm/s . . . 186
A.2.4 Third SLAM path using the Pioneer robot at 16 cm/s . . . 202
A.3 Guardian simulator tests . . . 218
A.3.1 Wall following path using the simulated Guardian robot at 30 cm/s in the static environment . . . 218
A.3.2 Wall following path using the simulated Guardian robot at 30 cm/s in the static cluttered environment . . . 233
A.3.3 Wall following path using the simulated Guardian robot at 30 cm/s in the dynamic cluttered environment . . . 248

B 6 DoF localization system results **263**
B.1 Kinect tests 264
B.1.1 Kinect with overview flying motion at 30 cm/s 264
B.1.2 Kinect with translation motions at 20 cm/s 276
B.1.3 Kinect with rotations motion at 10 cm/s 288

References **301**

List of Figures

2.1.2.1 Different types of optical encoders used to measure distances . . . 4
2.2.1.1 Trilateration technique in a global localization system . . . 5
2.2.2.1 Circle of position in celestial navigation . . . 6
2.2.4.1 ICP point cloud matching . . . 7
2.2.5.1 MCL animation . . . 8
2.2.5.2 MCL redistribution of particles . . . 8
2.2.5.3 MCL position refinement . . . 8
2.2.5.4 MCL position estimation . . . 8
2.2.5.5 Unscented Kalman Filter . . . 9
2.2.5.6 Position estimate . . . 10
2.2.5.7 Position associated error . . . 10

3.1.0.1 ROS logo . . . 13
3.2.0.1 PCL logo . . . 17
3.2.0.2 Point Cloud Library . . . 17
3.3.0.3 Gazebo logo . . . 18
3.3.0.4 Integration of ROS and Gazebo . . . 18
3.4.1.1 Structured light system diagram . . . 19
3.4.1.2 Kinect 2 sensor . . . 19
3.4.1.3 Structure IO sensor . . . 19
3.4.2.1 LIDAR scan . . . 20
3.4.2.2 Time of Flight system . . . 21
3.4.2.3 Mesa SR4000 . . . 21
3.4.2.4 2D SICK NAV 350 . . . 21
3.4.2.5 3D Velodyne HDL-64E . . . 21
3.4.2.6 RADAR scan of two ships . . . 22
3.4.2.7 QT50R-AFH RADAR . . . 22
3.4.2.8 SONAR scan of two ships . . . 22
3.4.2.9 SONAR sensor . . . 22
3.4.3.1 Stereo vision system . . . 23
3.4.3.2 PR2 head capable of active stereo vision . . . 23
3.4.3.3 Stereo vision overview . . . 24

4.1.0.1 Localization system brief overview . . . 25
4.1.0.2 Localization system overview . . . 26
4.4.0.3 Laser scans assembled with tilting platform (red points) . . . 28
4.5.1.1 Voxel grid applied over trees point clouds . . . 29
4.5.2.1 Octree of a stag beetle . . . 29

4.5.3.1 2-d tree . . . 30
4.5.3.2 3-d tree . . . 30
4.6.1.1 Table point cloud before (left) and after (right) voxel grid downsampling . . 31
4.6.2.1 Statistical outlier removal filter . . . 33
4.6.3.1 Surface smoothing using moving least squares algorithm . . . 34
4.6.3.2 Surface normals refinement using moving least squares algorithm . . . 34
4.7.2.1 Point neighborhood for normal estimation . . . 35
4.11.0.1 Point cloud registration with outlier detection . . . 37

5.1.1.1 Jarvis testing platform . . . 40
5.1.2.1 Pioneer 3-DX testing platform [SEE+12] . . . 40
5.1.3.1 Guardian testing platform . . . 41
5.1.3.2 Guardian Gazebo simulation . . . 41
5.2.1.1 Jarvis testing environment . . . 42
5.2.2.1 Guardian testing environment . . . 43
5.2.2.2 Guardian cluttered (left) and dynamic (right) testing environment . . . 43
5.2.2.3 Interactive CAD of Guardian testing environment (.u3d file) . . . 43
5.2.3.1 Industrial hall with and without objects in the center [SEE+12] . . . 44
5.2.4.1 Flying arena environment [PMC+11] . . . 44
5.3.2.1 Large laser deformation on opposite walls when the robot is rotating . . . 48
5.3.2.2 Correction of the large laser deformation on opposite walls with spherical linear interpolation . . . 49
5.3.2.3 Laser deformation when the robot is rotating . . . 49
5.3.2.4 Correction of the laser deformation with spherical linear interpolation . . . 49
5.3.4.1 Map made with the localization system using full integration in conjunction with OctoMap . . . 51
5.3.4.2 Map manually corrected using the previous figure as starting point . . . 51
5.3.4.3 Map made with the GMapping package playing the rosbag at 10% speed . . 51
5.3.4.4 Map made with the GMapping package playing the rosbag in real time . . . 52
5.3.4.5 Map made using the ground truth poses provided by the Raptor-E cameras . 52
5.3.4.6 Original map from room i-108 . . . 52
5.3.4.7 Updated map of room i-108 using full integration . . . 53
5.3.4.8 Original map of structured environment . . . 53
5.3.4.9 Updated map of structured environment using partial integration (outliers) . . 53
5.3.5.1 Laser scans assembled on top of the map using the localization system poses 55
5.3.5.2 Laser scans assembled on top of the map using the AMCL poses . . . 55
5.3.5.3 Initial pose estimation using feature matching in Guardian static environment 55
5.3.5.4 Initial pose estimation using feature matching in Jarvis environment . . . 56
5.3.5.5 Initial pose estimation distribution in Jarvis environment . . . 56
5.3.6.1 Probability distributions for the localization system translation errors . . . 57
5.3.6.2 Probability distributions for the localization system rotation errors . . . 57
5.3.7.1 Probability distributions for the localization system computation time . . . 57
5.4.2.1 Example of filtered point cloud from Kinect data . . . 58
5.4.3.1 3D SLAM using the ground truth poses . . . 60
5.4.3.2 3D SLAM using the localization system with surface reconstruction . . . 61
5.4.4.1 Poses estimated by the ground truth and localization system . . . 62
5.4.4.2 Poses estimated by the *ethzasl_icp_mapper* ROS package [PMC+11] . . . 62

5.4.4.3 Point clouds assembled on top of the map using the localization system poses (for the top part of the figure: green arrows → ground truth poses, red arrows → localization system poses) . . . 63
5.4.4.4 Point clouds assembled on top of the map using the ground truth poses . . . 63
5.4.5.1 Probability distributions for the localization system translation errors . . . 64
5.4.5.2 Probability distributions for the localization system rotation errors . . . 64
5.4.6.1 Probability distributions for the localization system global computation time 64

A.1.1.1 Animation of laser scans using the ground truth poses . . . 68
A.1.1.2 Animation of laser scans using the localization system poses . . . 68
A.1.1.3 Animation of laser scans using the AMCL poses . . . 68
A.1.1.4 Laser scans assembled on top of the map using the ground truth poses . . . 69
A.1.1.5 Laser scans assembled on top of the map using the localization system poses 69
A.1.1.6 Laser scans assembled on top of the map using the AMCL poses . . . 69
A.1.1.7 Poses estimated by the ground truth, localization system, AMCL and odometry 70
A.1.1.8 Evolution of x position over time . . . 70
A.1.1.9 Evolution of y position over time . . . 70
A.1.1.10 Distance traveled between consecutive pose estimations . . . 71
A.1.1.11 Estimated linear velocity . . . 71
A.1.1.12 Estimated linear acceleration . . . 71
A.1.1.13 Angular differences between consecutive pose estimations . . . 72
A.1.1.14 Estimated angular velocity . . . 72
A.1.1.15 Estimated angular acceleration . . . 72
A.1.1.16 Odometry translation errors . . . 73
A.1.1.17 Localization system translation errors . . . 73
A.1.1.18 AMCL translation errors . . . 73
A.1.1.19 Odometry translation errors components . . . 74
A.1.1.20 Localization system translation errors components . . . 74
A.1.1.21 AMCL translation errors components . . . 74
A.1.1.22 Probability distributions for the odometry translation errors . . . 75
A.1.1.23 Probability distributions for the localization system translation errors . . . 75
A.1.1.24 Probability distributions for the AMCL translation errors . . . 75
A.1.1.25 Odometry rotation errors axis . . . 76
A.1.1.26 Localization system rotation errors axis . . . 76
A.1.1.27 AMCL rotation errors axis . . . 76
A.1.1.28 Odometry rotation errors . . . 77
A.1.1.29 Localization system rotation errors . . . 77
A.1.1.30 AMCL rotation errors . . . 77
A.1.1.31 Probability distributions for the odometry rotation errors . . . 78
A.1.1.32 Probability distributions for the localization system rotation errors . . . 78
A.1.1.33 Probability distributions for the AMCL rotation errors . . . 78
A.1.1.34 Translation corrections performed by the localization system . . . 79
A.1.1.35 Translation corrections components performed by the localization system . . . 79
A.1.1.36 Probability distributions for the translation corrections performed by the localization system . . . 79
A.1.1.37 Rotation corrections (axis) performed by the localization system . . . 80
A.1.1.38 Rotation corrections performed by the localization system . . . 80
A.1.1.39 Probability distributions for the rotation corrections performed by the localization system . . . 80

A.1.1.40 Point clouds size in several of the localization system processing stages . . . 81
A.1.1.41 Number of registered points . . . 81
A.1.1.42 Probability distributions for the ambient point cloud outlier percentage . . . 81
A.1.1.43 Angular distribution percentage of registered points . . . 82
A.1.1.44 Root Mean Square Error of the inliers . . . 82
A.1.1.45 Probability distributions for the Root Mean Square Error of the inliers . . . 82
A.1.1.46 Localization system computation times . . . 83
A.1.1.47 Probability distributions for the point cloud registration time . . . 83
A.1.1.48 Probability distributions for the localization system global computation time 83
A.1.2.1 Animation of laser scans using the ground truth poses . . . 84
A.1.2.2 Animation of laser scans using the localization system poses . . . 84
A.1.2.3 Animation of laser scans using the AMCL poses . . . 84
A.1.2.4 Laser scans assembled on top of the map using the ground truth poses . . . 85
A.1.2.5 Laser scans assembled on top of the map using the localization system poses 85
A.1.2.6 Laser scans assembled on top of the map using the AMCL poses . . . 85
A.1.2.7 Poses estimated by the ground truth, localization system, AMCL and odometry 86
A.1.2.8 Evolution of x position over time . . . 86
A.1.2.9 Evolution of y position over time . . . 86
A.1.2.10 Distance traveled between consecutive pose estimations . . . 87
A.1.2.11 Estimated linear velocity . . . 87
A.1.2.12 Estimated linear acceleration . . . 87
A.1.2.13 Angular differences between consecutive pose estimations . . . 88
A.1.2.14 Estimated angular velocity . . . 88
A.1.2.15 Estimated angular acceleration . . . 88
A.1.2.16 Odometry translation errors . . . 89
A.1.2.17 Localization system translation errors . . . 89
A.1.2.18 AMCL translation errors . . . 89
A.1.2.19 Odometry translation errors components . . . 90
A.1.2.20 Localization system translation errors components . . . 90
A.1.2.21 AMCL translation errors components . . . 90
A.1.2.22 Probability distributions for the odometry translation errors . . . 91
A.1.2.23 Probability distributions for the localization system translation errors . . . 91
A.1.2.24 Probability distributions for the AMCL translation errors . . . 91
A.1.2.25 Odometry rotation errors axis . . . 92
A.1.2.26 Localization system rotation errors axis . . . 92
A.1.2.27 AMCL rotation errors axis . . . 92
A.1.2.28 Odometry rotation errors . . . 93
A.1.2.29 Localization system rotation errors . . . 93
A.1.2.30 AMCL rotation errors . . . 93
A.1.2.31 Probability distributions for the odometry rotation errors . . . 94
A.1.2.32 Probability distributions for the localization system rotation errors . . . 94
A.1.2.33 Probability distributions for the AMCL rotation errors . . . 94
A.1.2.34 Translation corrections performed by the localization system . . . 95
A.1.2.35 Translation corrections components performed by the localization system . . 95
A.1.2.36 Probability distributions for the translation corrections performed by the localization system . . . 95
A.1.2.37 Rotation corrections (axis) performed by the localization system . . . 96
A.1.2.38 Rotation corrections performed by the localization system . . . 96

A.1.2.39 Probability distributions for the rotation corrections performed by the localization system . . . 96
A.1.2.40 Point clouds size in several of the localization system processing stages . . . 97
A.1.2.41 Number of registered points . . . 97
A.1.2.42 Probability distributions for the ambient point cloud outlier percentage . . . 97
A.1.2.43 Angular distribution percentage of registered points . . . 98
A.1.2.44 Root Mean Square Error of the inliers . . . 98
A.1.2.45 Probability distributions for the Root Mean Square Error of the inliers . . . 98
A.1.2.46 Localization system computation times . . . 99
A.1.2.47 Probability distributions for the point cloud registration time . . . 99
A.1.2.48 Probability distributions for the localization system global computation time 99
A.1.3.1 Animation of laser scans using the ground truth poses . . . 100
A.1.3.2 Animation of laser scans using the localization system poses . . . 100
A.1.3.3 Animation of laser scans using the AMCL poses . . . 100
A.1.3.4 Laser scans assembled on top of the map using the ground truth poses . . . 101
A.1.3.5 Laser scans assembled on top of the map using the localization system poses 101
A.1.3.6 Laser scans assembled on top of the map using the AMCL poses . . . 101
A.1.3.7 Poses estimated by the ground truth, localization system, AMCL and odometry 102
A.1.3.8 Evolution of x position over time . . . 102
A.1.3.9 Evolution of y position over time . . . 102
A.1.3.10 Distance traveled between consecutive pose estimations . . . 103
A.1.3.11 Estimated linear velocity . . . 103
A.1.3.12 Estimated linear acceleration . . . 103
A.1.3.13 Angular differences between consecutive pose estimations . . . 104
A.1.3.14 Estimated angular velocity . . . 104
A.1.3.15 Estimated angular acceleration . . . 104
A.1.3.16 Odometry translation errors . . . 105
A.1.3.17 Localization system translation errors . . . 105
A.1.3.18 AMCL translation errors . . . 105
A.1.3.19 Odometry translation errors components . . . 106
A.1.3.20 Localization system translation errors components . . . 106
A.1.3.21 AMCL translation errors components . . . 106
A.1.3.22 Probability distributions for the odometry translation errors . . . 107
A.1.3.23 Probability distributions for the localization system translation errors . . . 107
A.1.3.24 Probability distributions for the AMCL translation errors . . . 107
A.1.3.25 Odometry rotation errors axis . . . 108
A.1.3.26 Localization system rotation errors axis . . . 108
A.1.3.27 AMCL rotation errors axis . . . 108
A.1.3.28 Odometry rotation errors . . . 109
A.1.3.29 Localization system rotation errors . . . 109
A.1.3.30 AMCL rotation errors . . . 109
A.1.3.31 Probability distributions for the odometry rotation errors . . . 110
A.1.3.32 Probability distributions for the localization system rotation errors . . . 110
A.1.3.33 Probability distributions for the AMCL rotation errors . . . 110
A.1.3.34 Translation corrections performed by the localization system . . . 111
A.1.3.35 Translation corrections components performed by the localization system . . . 111
A.1.3.36 Probability distributions for the translation corrections performed by the localization system . . . 111

A.1.3.37 Rotation corrections (axis) performed by the localization system 112
A.1.3.38 Rotation corrections performed by the localization system 112
A.1.3.39 Probability distributions for the rotation corrections performed by the localization system 112
A.1.3.40 Point clouds size in several of the localization system processing stages . . . 113
A.1.3.41 Number of registered points 113
A.1.3.42 Probability distributions for the ambient point cloud outlier percentage . . . 113
A.1.3.43 Angular distribution percentage of registered points 114
A.1.3.44 Root Mean Square Error of the inliers 114
A.1.3.45 Probability distributions for the Root Mean Square Error of the inliers 114
A.1.3.46 Localization system computation times 115
A.1.3.47 Probability distributions for the point cloud registration time 115
A.1.3.48 Probability distributions for the localization system global computation time 115
A.1.4.1 Animation of laser scans using the ground truth poses 116
A.1.4.2 Animation of laser scans using the localization system poses 116
A.1.4.3 Animation of laser scans using the AMCL poses 116
A.1.4.4 Laser scans assembled on top of the map using the ground truth poses 117
A.1.4.5 Laser scans assembled on top of the map using the localization system poses 117
A.1.4.6 Laser scans assembled on top of the map using the AMCL poses 117
A.1.4.7 Poses estimated by the ground truth, localization system, AMCL and odometry 118
A.1.4.8 Evolution of x position over time 118
A.1.4.9 Evolution of y position over time 118
A.1.4.10 Distance traveled between consecutive pose estimations 119
A.1.4.11 Estimated linear velocity 119
A.1.4.12 Estimated linear acceleration 119
A.1.4.13 Angular differences between consecutive pose estimations 120
A.1.4.14 Estimated angular velocity 120
A.1.4.15 Estimated angular acceleration 120
A.1.4.16 Odometry translation errors 121
A.1.4.17 Localization system translation errors 121
A.1.4.18 AMCL translation errors 121
A.1.4.19 Odometry translation errors components 122
A.1.4.20 Localization system translation errors components 122
A.1.4.21 AMCL translation errors components 122
A.1.4.22 Probability distributions for the odometry translation errors 123
A.1.4.23 Probability distributions for the localization system translation errors 123
A.1.4.24 Probability distributions for the AMCL translation errors 123
A.1.4.25 Odometry rotation errors axis 124
A.1.4.26 Localization system rotation errors axis 124
A.1.4.27 AMCL rotation errors axis 124
A.1.4.28 Odometry rotation errors 125
A.1.4.29 Localization system rotation errors 125
A.1.4.30 AMCL rotation errors 125
A.1.4.31 Probability distributions for the odometry rotation errors 126
A.1.4.32 Probability distributions for the localization system rotation errors 126
A.1.4.33 Probability distributions for the AMCL rotation errors 126
A.1.4.34 Translation corrections performed by the localization system 127
A.1.4.35 Translation corrections components performed by the localization system . . 127

A.1.4.36 Probability distributions for the translation corrections performed by the localization system . . . 127
A.1.4.37 Rotation corrections (axis) performed by the localization system . . . 128
A.1.4.38 Rotation corrections performed by the localization system . . . 128
A.1.4.39 Probability distributions for the rotation corrections performed by the localization system . . . 128
A.1.4.40 Point clouds size in several of the localization system processing stages . . . 129
A.1.4.41 Number of registered points . . . 129
A.1.4.42 Probability distributions for the ambient point cloud outlier percentage . . . 129
A.1.4.43 Angular distribution percentage of registered points . . . 130
A.1.4.44 Root Mean Square Error of the inliers . . . 130
A.1.4.45 Probability distributions for the Root Mean Square Error of the inliers . . . 130
A.1.4.46 Localization system computation times . . . 131
A.1.4.47 Probability distributions for the point cloud registration time . . . 131
A.1.4.48 Probability distributions for the localization system global computation time 131
A.1.5.1 Animation of laser scans using spherical linear interpolation . . . 132
A.1.5.2 Animation of laser scans without using spherical linear interpolation . . . 132
A.1.5.3 Laser scans assembled in the previous test using laser spherical linear interpolation . . . 133
A.1.5.4 Laser scans assembled without using laser spherical linear interpolation . . . 133
A.1.5.5 Poses estimated by the ground truth, localization system and odometry . . . 133
A.1.5.6 Evolution of x position over time . . . 134
A.1.5.7 Evolution of y position over time . . . 134
A.1.5.8 Distance traveled between consecutive pose estimations . . . 135
A.1.5.9 Angular differences between consecutive pose estimations . . . 135
A.1.5.10 Localization system translation errors . . . 136
A.1.5.11 Localization system translation errors components . . . 136
A.1.5.12 Probability distributions for the localization system translation errors . . . 136
A.1.5.13 Localization system rotation errors axis . . . 137
A.1.5.14 Localization system rotation errors . . . 137
A.1.5.15 Probability distributions for the localization system rotation errors . . . 137
A.1.5.16 Translation corrections performed by the localization system . . . 138
A.1.5.17 Translation corrections components performed by the localization system . . . 138
A.1.5.18 Probability distributions for the translation corrections performed by the localization system . . . 138
A.1.5.19 Rotation corrections (axis) performed by the localization system . . . 139
A.1.5.20 Rotation corrections performed by the localization system . . . 139
A.1.5.21 Probability distributions for the rotation corrections performed by the localization system . . . 139
A.1.5.22 Point clouds size in several of the localization system processing stages . . . 140
A.1.5.23 Number of registered points . . . 140
A.1.5.24 Probability distributions for the ambient point cloud outlier percentage . . . 140
A.1.5.25 Angular distribution percentage of registered points . . . 141
A.1.5.26 Root Mean Square Error of the inliers . . . 141
A.1.5.27 Probability distributions for the Root Mean Square Error of the inliers . . . 141
A.1.5.28 Localization system computation times . . . 142
A.1.5.29 Probability distributions for the point cloud registration time . . . 142
A.1.5.30 Probability distributions for the localization system global computation time 142

A.1.6.1 Animation of laser scans using spherical linear interpolation 143
A.1.6.2 Animation of laser scans without using spherical linear interpolation 143
A.1.6.3 Laser scans assembled in the previous test using laser spherical linear interpolation 144
A.1.6.4 Laser scans assembled without using laser spherical linear interpolation . . . 144
A.1.6.5 Poses estimated by the ground truth, localization system and odometry . . . 144
A.1.6.6 Evolution of x position over time 145
A.1.6.7 Evolution of y position over time 145
A.1.6.8 Distance traveled between consecutive pose estimations 146
A.1.6.9 Angular differences between consecutive pose estimations 146
A.1.6.10 Localization system translation errors 147
A.1.6.11 Localization system translation errors components 147
A.1.6.12 Probability distributions for the localization system translation errors 147
A.1.6.13 Localization system rotation errors axis 148
A.1.6.14 Localization system rotation errors 148
A.1.6.15 Probability distributions for the localization system rotation errors 148
A.1.6.16 Translation corrections performed by the localization system 149
A.1.6.17 Translation corrections components performed by the localization system . . 149
A.1.6.18 Probability distributions for the translation corrections performed by the localization system 149
A.1.6.19 Rotation corrections (axis) performed by the localization system 150
A.1.6.20 Rotation corrections performed by the localization system 150
A.1.6.21 Probability distributions for the rotation corrections performed by the localization system 150
A.1.6.22 Point clouds size in several of the localization system processing stages . . . 151
A.1.6.23 Number of registered points 151
A.1.6.24 Probability distributions for the ambient point cloud outlier percentage . . . 151
A.1.6.25 Angular distribution percentage of registered points 152
A.1.6.26 Root Mean Square Error of the inliers 152
A.1.6.27 Probability distributions for the Root Mean Square Error of the inliers 152
A.1.6.28 Localization system computation times 153
A.1.6.29 Probability distributions for the point cloud registration time 153
A.1.6.30 Probability distributions for the localization system global computation time 153
A.2.1.1 Animation of laser scans using the ground truth poses 154
A.2.1.2 Animation of laser scans using the localization system poses 154
A.2.1.3 Animation of laser scans using the AMCL poses 154
A.2.1.4 Laser scans assembled on top of the map using the ground truth poses 155
A.2.1.5 Laser scans assembled on top of the map using the localization system poses 155
A.2.1.6 Laser scans assembled on top of the map using the AMCL poses 155
A.2.1.7 Poses estimated by the ground truth, localization system, AMCL and odometry 156
A.2.1.8 Evolution of x position over time 156
A.2.1.9 Evolution of y position over time 156
A.2.1.10 Distance traveled between consecutive pose estimations 157
A.2.1.11 Estimated linear velocity 157
A.2.1.12 Estimated linear acceleration 157
A.2.1.13 Angular differences between consecutive pose estimations 158
A.2.1.14 Estimated angular velocity 158
A.2.1.15 Estimated angular acceleration 158

A.2.1.16 Odometry translation errors . . . 159
A.2.1.17 Localization system translation errors . . . 159
A.2.1.18 AMCL translation errors . . . 159
A.2.1.19 Odometry translation errors components . . . 160
A.2.1.20 Localization system translation errors components . . . 160
A.2.1.21 AMCL translation errors components . . . 160
A.2.1.22 Probability distributions for the odometry translation errors . . . 161
A.2.1.23 Probability distributions for the localization system translation errors . . . 161
A.2.1.24 Probability distributions for the AMCL translation errors . . . 161
A.2.1.25 Odometry rotation errors axis . . . 162
A.2.1.26 Localization system rotation errors axis . . . 162
A.2.1.27 AMCL rotation errors axis . . . 162
A.2.1.28 Odometry rotation errors . . . 163
A.2.1.29 Localization system rotation errors . . . 163
A.2.1.30 AMCL rotation errors . . . 163
A.2.1.31 Probability distributions for the odometry rotation errors . . . 164
A.2.1.32 Probability distributions for the localization system rotation errors . . . 164
A.2.1.33 Probability distributions for the AMCL rotation errors . . . 164
A.2.1.34 Translation corrections performed by the localization system . . . 165
A.2.1.35 Translation corrections components performed by the localization system . . . 165
A.2.1.36 Probability distributions for the translation corrections performed by the localization system . . . 165
A.2.1.37 Rotation corrections (axis) performed by the localization system . . . 166
A.2.1.38 Rotation corrections performed by the localization system . . . 166
A.2.1.39 Probability distributions for the rotation corrections performed by the localization system . . . 166
A.2.1.40 Point clouds size in several of the localization system processing stages . . . 167
A.2.1.41 Number of registered points . . . 167
A.2.1.42 Probability distributions for the ambient point cloud outlier percentage . . . 167
A.2.1.43 Angular distribution percentage of registered points . . . 168
A.2.1.44 Root Mean Square Error of the inliers . . . 168
A.2.1.45 Probability distributions for the Root Mean Square Error of the inliers . . . 168
A.2.1.46 Localization system computation times . . . 169
A.2.1.47 Probability distributions for the point cloud registration time . . . 169
A.2.1.48 Probability distributions for the localization system global computation time 169
A.2.2.1 Animation of laser scans using the ground truth poses . . . 170
A.2.2.2 Animation of laser scans using the localization system poses . . . 170
A.2.2.3 Animation of laser scans using the AMCL poses . . . 170
A.2.2.4 Laser scans assembled on top of the map using the ground truth poses . . . 171
A.2.2.5 Laser scans assembled on top of the map using the localization system poses 171
A.2.2.6 Laser scans assembled on top of the map using the AMCL poses . . . 171
A.2.2.7 Poses estimated by the ground truth, localization system, AMCL and odometry 172
A.2.2.8 Evolution of x position over time . . . 172
A.2.2.9 Evolution of y position over time . . . 172
A.2.2.10 Distance traveled between consecutive pose estimations . . . 173
A.2.2.11 Estimated linear velocity . . . 173
A.2.2.12 Estimated linear acceleration . . . 173
A.2.2.13 Angular differences between consecutive pose estimations . . . 174

A.2.2.14 Estimated angular velocity . . . 174
A.2.2.15 Estimated angular acceleration . . . 174
A.2.2.16 Odometry translation errors . . . 175
A.2.2.17 Localization system translation errors . . . 175
A.2.2.18 AMCL translation errors . . . 175
A.2.2.19 Odometry translation errors components . . . 176
A.2.2.20 Localization system translation errors components . . . 176
A.2.2.21 AMCL translation errors components . . . 176
A.2.2.22 Probability distributions for the odometry translation errors . . . 177
A.2.2.23 Probability distributions for the localization system translation errors . . . 177
A.2.2.24 Probability distributions for the AMCL translation errors . . . 177
A.2.2.25 Odometry rotation errors axis . . . 178
A.2.2.26 Localization system rotation errors axis . . . 178
A.2.2.27 AMCL rotation errors axis . . . 178
A.2.2.28 Odometry rotation errors . . . 179
A.2.2.29 Localization system rotation errors . . . 179
A.2.2.30 AMCL rotation errors . . . 179
A.2.2.31 Probability distributions for the odometry rotation errors . . . 180
A.2.2.32 Probability distributions for the localization system rotation errors . . . 180
A.2.2.33 Probability distributions for the AMCL rotation errors . . . 180
A.2.2.34 Translation corrections performed by the localization system . . . 181
A.2.2.35 Translation corrections components performed by the localization system . . . 181
A.2.2.36 Probability distributions for the translation corrections performed by the localization system . . . 181
A.2.2.37 Rotation corrections (axis) performed by the localization system . . . 182
A.2.2.38 Rotation corrections performed by the localization system . . . 182
A.2.2.39 Probability distributions for the rotation corrections performed by the localization system . . . 182
A.2.2.40 Point clouds size in several of the localization system processing stages . . . 183
A.2.2.41 Number of registered points . . . 183
A.2.2.42 Probability distributions for the ambient point cloud outlier percentage . . . 183
A.2.2.43 Angular distribution percentage of registered points . . . 184
A.2.2.44 Root Mean Square Error of the inliers . . . 184
A.2.2.45 Probability distributions for the Root Mean Square Error of the inliers . . . 184
A.2.2.46 Localization system computation times . . . 185
A.2.2.47 Probability distributions for the point cloud registration time . . . 185
A.2.2.48 Probability distributions for the localization system global computation time 185
A.2.3.1 Animation of laser scans using the ground truth poses . . . 186
A.2.3.2 Animation of laser scans using the localization system poses . . . 186
A.2.3.3 Animation of laser scans using the AMCL poses . . . 186
A.2.3.4 Laser scans assembled on top of the map using the ground truth poses . . . 187
A.2.3.5 Laser scans assembled on top of the map using the localization system poses 187
A.2.3.6 Laser scans assembled on top of the map using the AMCL poses . . . 187
A.2.3.7 Poses estimated by the ground truth, localization system, AMCL and odometry 188
A.2.3.8 Evolution of x position over time . . . 188
A.2.3.9 Evolution of y position over time . . . 188
A.2.3.10 Distance traveled between consecutive pose estimations . . . 189
A.2.3.11 Estimated linear velocity . . . 189

A.2.3.12 Estimated linear acceleration . . . 189
A.2.3.13 Angular differences between consecutive pose estimations . . . 190
A.2.3.14 Estimated angular velocity . . . 190
A.2.3.15 Estimated angular acceleration . . . 190
A.2.3.16 Odometry translation errors . . . 191
A.2.3.17 Localization system translation errors . . . 191
A.2.3.18 AMCL translation errors . . . 191
A.2.3.19 Odometry translation errors components . . . 192
A.2.3.20 Localization system translation errors components . . . 192
A.2.3.21 AMCL translation errors components . . . 192
A.2.3.22 Probability distributions for the odometry translation errors . . . 193
A.2.3.23 Probability distributions for the localization system translation errors . . . 193
A.2.3.24 Probability distributions for the AMCL translation errors . . . 193
A.2.3.25 Odometry rotation errors axis . . . 194
A.2.3.26 Localization system rotation errors axis . . . 194
A.2.3.27 AMCL rotation errors axis . . . 194
A.2.3.28 Odometry rotation errors . . . 195
A.2.3.29 Localization system rotation errors . . . 195
A.2.3.30 AMCL rotation errors . . . 195
A.2.3.31 Probability distributions for the odometry rotation errors . . . 196
A.2.3.32 Probability distributions for the localization system rotation errors . . . 196
A.2.3.33 Probability distributions for the AMCL rotation errors . . . 196
A.2.3.34 Translation corrections performed by the localization system . . . 197
A.2.3.35 Translation corrections components performed by the localization system . . . 197
A.2.3.36 Probability distributions for the translation corrections performed by the localization system . . . 197
A.2.3.37 Rotation corrections (axis) performed by the localization system . . . 198
A.2.3.38 Rotation corrections performed by the localization system . . . 198
A.2.3.39 Probability distributions for the rotation corrections performed by the localization system . . . 198
A.2.3.40 Point clouds size in several of the localization system processing stages . . . 199
A.2.3.41 Number of registered points . . . 199
A.2.3.42 Probability distributions for the ambient point cloud outlier percentage . . . 199
A.2.3.43 Angular distribution percentage of registered points . . . 200
A.2.3.44 Root Mean Square Error of the inliers . . . 200
A.2.3.45 Probability distributions for the Root Mean Square Error of the inliers . . . 200
A.2.3.46 Localization system computation times . . . 201
A.2.3.47 Probability distributions for the point cloud registration time . . . 201
A.2.3.48 Probability distributions for the localization system global computation time 201
A.2.4.1 Animation of laser scans using the ground truth poses . . . 202
A.2.4.2 Animation of laser scans using the localization system poses . . . 202
A.2.4.3 Animation of laser scans using the AMCL poses . . . 202
A.2.4.4 Laser scans assembled on top of the map using the ground truth poses . . . 203
A.2.4.5 Laser scans assembled on top of the map using the localization system poses 203
A.2.4.6 Laser scans assembled on top of the map using the AMCL poses . . . 203
A.2.4.7 Poses estimated by the ground truth, localization system, AMCL and odometry 204
A.2.4.8 Evolution of x position over time . . . 204
A.2.4.9 Evolution of y position over time . . . 204

A.2.4.10 Distance traveled between consecutive pose estimations . . . 205
A.2.4.11 Estimated linear velocity . . . 205
A.2.4.12 Estimated linear acceleration . . . 205
A.2.4.13 Angular differences between consecutive pose estimations . . . 206
A.2.4.14 Estimated angular velocity . . . 206
A.2.4.15 Estimated angular acceleration . . . 206
A.2.4.16 Odometry translation errors . . . 207
A.2.4.17 Localization system translation errors . . . 207
A.2.4.18 AMCL translation errors . . . 207
A.2.4.19 Odometry translation errors components . . . 208
A.2.4.20 Localization system translation errors components . . . 208
A.2.4.21 AMCL translation errors components . . . 208
A.2.4.22 Probability distributions for the odometry translation errors . . . 209
A.2.4.23 Probability distributions for the localization system translation errors . . . 209
A.2.4.24 Probability distributions for the AMCL translation errors . . . 209
A.2.4.25 Odometry rotation errors axis . . . 210
A.2.4.26 Localization system rotation errors axis . . . 210
A.2.4.27 AMCL rotation errors axis . . . 210
A.2.4.28 Odometry rotation errors . . . 211
A.2.4.29 Localization system rotation errors . . . 211
A.2.4.30 AMCL rotation errors . . . 211
A.2.4.31 Probability distributions for the odometry rotation errors . . . 212
A.2.4.32 Probability distributions for the localization system rotation errors . . . 212
A.2.4.33 Probability distributions for the AMCL rotation errors . . . 212
A.2.4.34 Translation corrections performed by the localization system . . . 213
A.2.4.35 Translation corrections components performed by the localization system . . . 213
A.2.4.36 Probability distributions for the translation corrections performed by the localization system . . . 213
A.2.4.37 Rotation corrections (axis) performed by the localization system . . . 214
A.2.4.38 Rotation corrections performed by the localization system . . . 214
A.2.4.39 Probability distributions for the rotation corrections performed by the localization system . . . 214
A.2.4.40 Point clouds size in several of the localization system processing stages . . . 215
A.2.4.41 Number of registered points . . . 215
A.2.4.42 Probability distributions for the ambient point cloud outlier percentage . . . 215
A.2.4.43 Angular distribution percentage of registered points . . . 216
A.2.4.44 Root Mean Square Error of the inliers . . . 216
A.2.4.45 Probability distributions for the Root Mean Square Error of the inliers . . . 216
A.2.4.46 Localization system computation times . . . 217
A.2.4.47 Probability distributions for the point cloud registration time . . . 217
A.2.4.48 Probability distributions for the localization system global computation time 217
A.3.1.1 Animation of laser scans using the localization system poses . . . 218
A.3.1.2 Laser scans assembled on top of the map using the localization system poses 218
A.3.1.3 Poses estimated by the ground truth, localization system and odometry . . . 219
A.3.1.4 Evolution of x position over time . . . 219
A.3.1.5 Evolution of y position over time . . . 219
A.3.1.6 Distance traveled between consecutive pose estimations . . . 220
A.3.1.7 Estimated linear velocity . . . 220

A.3.1.8 Estimated linear acceleration . . . 220
A.3.1.9 Angular differences between consecutive pose estimations . . . 221
A.3.1.10 Estimated angular velocity . . . 221
A.3.1.11 Estimated angular acceleration . . . 221
A.3.1.12 Odometry translation errors . . . 222
A.3.1.13 Localization system translation errors . . . 222
A.3.1.14 Odometry translation errors components . . . 223
A.3.1.15 Localization system translation errors components . . . 223
A.3.1.16 Probability distributions for the odometry translation errors . . . 224
A.3.1.17 Probability distributions for the localization system translation errors . . . 224
A.3.1.18 Odometry rotation errors axis . . . 225
A.3.1.19 Localization system rotation errors axis . . . 225
A.3.1.20 Odometry rotation errors . . . 226
A.3.1.21 Localization system rotation errors . . . 226
A.3.1.22 Probability distributions for the odometry rotation errors . . . 227
A.3.1.23 Probability distributions for the localization system rotation errors . . . 227
A.3.1.24 Translation corrections performed by the localization system . . . 228
A.3.1.25 Translation corrections components performed by the localization system . . . 228
A.3.1.26 Probability distributions for the translation corrections performed by the localization system . . . 228
A.3.1.27 Rotation corrections (axis) performed by the localization system . . . 229
A.3.1.28 Rotation corrections performed by the localization system . . . 229
A.3.1.29 Probability distributions for the rotation corrections performed by the localization system . . . 229
A.3.1.30 Point clouds size in several of the localization system processing stages . . . 230
A.3.1.31 Number of registered points . . . 230
A.3.1.32 Probability distributions for the ambient point cloud outlier percentage . . . 230
A.3.1.33 Angular distribution percentage of registered points . . . 231
A.3.1.34 Root Mean Square Error of the inliers . . . 231
A.3.1.35 Probability distributions for the Root Mean Square Error of the inliers . . . 231
A.3.1.36 Localization system computation times . . . 232
A.3.1.37 Probability distributions for the point cloud registration time . . . 232
A.3.1.38 Probability distributions for the localization system global computation time 232
A.3.2.1 Animation of laser scans using the localization system poses . . . 233
A.3.2.2 Laser scans assembled on top of the map using the localization system poses 233
A.3.2.3 Poses estimated by the ground truth, localization system and odometry . . . 234
A.3.2.4 Evolution of x position over time . . . 234
A.3.2.5 Evolution of y position over time . . . 234
A.3.2.6 Distance traveled between consecutive pose estimations . . . 235
A.3.2.7 Estimated linear velocity . . . 235
A.3.2.8 Estimated linear acceleration . . . 235
A.3.2.9 Angular differences between consecutive pose estimations . . . 236
A.3.2.10 Estimated angular velocity . . . 236
A.3.2.11 Estimated angular acceleration . . . 236
A.3.2.12 Odometry translation errors . . . 237
A.3.2.13 Localization system translation errors . . . 237
A.3.2.14 Odometry translation errors components . . . 238
A.3.2.15 Localization system translation errors components . . . 238

A.3.2.16 Probability distributions for the odometry translation errors 239
A.3.2.17 Probability distributions for the localization system translation errors 239
A.3.2.18 Odometry rotation errors axis 240
A.3.2.19 Localization system rotation errors axis 240
A.3.2.20 Odometry rotation errors 241
A.3.2.21 Localization system rotation errors 241
A.3.2.22 Probability distributions for the odometry rotation errors 242
A.3.2.23 Probability distributions for the localization system rotation errors 242
A.3.2.24 Translation corrections performed by the localization system 243
A.3.2.25 Translation corrections components performed by the localization system . . 243
A.3.2.26 Probability distributions for the translation corrections performed by the localization system 243
A.3.2.27 Rotation corrections (axis) performed by the localization system 244
A.3.2.28 Rotation corrections performed by the localization system 244
A.3.2.29 Probability distributions for the rotation corrections performed by the localization system 244
A.3.2.30 Point clouds size in several of the localization system processing stages . . . 245
A.3.2.31 Number of registered points 245
A.3.2.32 Probability distributions for the ambient point cloud outlier percentage . . . 245
A.3.2.33 Angular distribution percentage of registered points 246
A.3.2.34 Root Mean Square Error of the inliers 246
A.3.2.35 Probability distributions for the Root Mean Square Error of the inliers 246
A.3.2.36 Localization system computation times 247
A.3.2.37 Probability distributions for the point cloud registration time 247
A.3.2.38 Probability distributions for the localization system global computation time 247
A.3.3.1 Animation of laser scans using the localization system poses 248
A.3.3.2 Laser scans assembled on top of the map using the localization system poses 248
A.3.3.3 Poses estimated by the ground truth, localization system and odometry . . . 249
A.3.3.4 Evolution of x position over time 249
A.3.3.5 Evolution of y position over time 249
A.3.3.6 Distance traveled between consecutive pose estimations 250
A.3.3.7 Estimated linear velocity 250
A.3.3.8 Estimated linear acceleration 250
A.3.3.9 Angular differences between consecutive pose estimations 251
A.3.3.10 Estimated angular velocity 251
A.3.3.11 Estimated angular acceleration 251
A.3.3.12 Odometry translation errors 252
A.3.3.13 Localization system translation errors 252
A.3.3.14 Odometry translation errors components 253
A.3.3.15 Localization system translation errors components 253
A.3.3.16 Probability distributions for the odometry translation errors 254
A.3.3.17 Probability distributions for the localization system translation errors 254
A.3.3.18 Odometry rotation errors axis 255
A.3.3.19 Localization system rotation errors axis 255
A.3.3.20 Odometry rotation errors 256
A.3.3.21 Localization system rotation errors 256
A.3.3.22 Probability distributions for the odometry rotation errors 257
A.3.3.23 Probability distributions for the localization system rotation errors 257

A.3.3.24 Translation corrections performed by the localization system 258
A.3.3.25 Translation corrections components performed by the localization system . . 258
A.3.3.26 Probability distributions for the translation corrections performed by the localization system 258
A.3.3.27 Rotation corrections (axis) performed by the localization system 259
A.3.3.28 Rotation corrections performed by the localization system 259
A.3.3.29 Probability distributions for the rotation corrections performed by the localization system 259
A.3.3.30 Point clouds size in several of the localization system processing stages . . . 260
A.3.3.31 Number of registered points 260
A.3.3.32 Probability distributions for the ambient point cloud outlier percentage . . . 260
A.3.3.33 Angular distribution percentage of registered points 261
A.3.3.34 Root Mean Square Error of the inliers 261
A.3.3.35 Probability distributions for the Root Mean Square Error of the inliers 261
A.3.3.36 Localization system computation times 262
A.3.3.37 Probability distributions for the point cloud registration time 262
A.3.3.38 Probability distributions for the localization system global computation time 262

B.1.1.1 Animation of point clouds using the localization system poses 264
B.1.1.2 Poses estimated by the ground truth and localization system 264
B.1.1.3 Poses estimated by the *ethzasl_icp_mapper* package [PMC+11] 264
B.1.1.4 Point clouds assembled on top of the map using the ground truth poses . . . 265
B.1.1.5 Point clouds assembled on top of the map using the localization system poses 265
B.1.1.6 Evolution of x position over time 266
B.1.1.7 Evolution of y position over time 266
B.1.1.8 Evolution of z position over time 266
B.1.1.9 Distance traveled between consecutive pose estimations 267
B.1.1.10 Estimated linear velocity 267
B.1.1.11 Estimated linear acceleration 267
B.1.1.12 Angular differences between consecutive pose estimations 268
B.1.1.13 Estimated angular velocity 268
B.1.1.14 Estimated angular acceleration 268
B.1.1.15 Localization system translation errors 269
B.1.1.16 Localization system translation errors components 269
B.1.1.17 Probability distributions for the localization system translation errors 269
B.1.1.18 Localization system rotation errors axis 270
B.1.1.19 Localization system rotation errors 270
B.1.1.20 Probability distributions for the localization system rotation errors 270
B.1.1.21 Translation corrections performed by the localization system 271
B.1.1.22 Translation corrections components performed by the localization system . . 271
B.1.1.23 Probability distributions for the translation corrections performed by the localization system 271
B.1.1.24 Rotation corrections (axis) performed by the localization system 272
B.1.1.25 Rotation corrections performed by the localization system 272
B.1.1.26 Probability distributions for the rotation corrections performed by the localization system 272
B.1.1.27 Point clouds size in several of the localization system processing stages . . . 273
B.1.1.28 Number of registered points 273
B.1.1.29 Probability distributions for the ambient point cloud outlier percentage . . . 273

B.1.1.30 Root Mean Square Error of the inliers . . . 274
B.1.1.31 Probability distributions for the Root Mean Square Error of the inliers . . . 274
B.1.1.32 Localization system computation times . . . 275
B.1.1.33 Probability distributions for the point cloud registration time . . . 275
B.1.1.34 Probability distributions for the localization system global computation time 275
B.1.2.1 Animation of point clouds using the localization system poses . . . 276
B.1.2.2 Poses estimated by the ground truth and localization system . . . 276
B.1.2.3 Point clouds assembled on top of the map using the ground truth poses . . . 277
B.1.2.4 Point clouds assembled on top of the map using the localization system poses 277
B.1.2.5 Evolution of x position over time . . . 278
B.1.2.6 Evolution of y position over time . . . 278
B.1.2.7 Evolution of z position over time . . . 278
B.1.2.8 Distance traveled between consecutive pose estimations . . . 279
B.1.2.9 Estimated linear velocity . . . 279
B.1.2.10 Estimated linear acceleration . . . 279
B.1.2.11 Angular differences between consecutive pose estimations . . . 280
B.1.2.12 Estimated angular velocity . . . 280
B.1.2.13 Estimated angular acceleration . . . 280
B.1.2.14 Localization system translation errors . . . 281
B.1.2.15 Localization system translation errors components . . . 281
B.1.2.16 Probability distributions for the localization system translation errors . . . 281
B.1.2.17 Localization system rotation errors axis . . . 282
B.1.2.18 Localization system rotation errors . . . 282
B.1.2.19 Probability distributions for the localization system rotation errors . . . 282
B.1.2.20 Translation corrections performed by the localization system . . . 283
B.1.2.21 Translation corrections components performed by the localization system . . 283
B.1.2.22 Probability distributions for the translation corrections performed by the localization system . . . 283
B.1.2.23 Rotation corrections (axis) performed by the localization system . . . 284
B.1.2.24 Rotation corrections performed by the localization system . . . 284
B.1.2.25 Probability distributions for the rotation corrections performed by the localization system . . . 284
B.1.2.26 Point clouds size in several of the localization system processing stages . . . 285
B.1.2.27 Number of registered points . . . 285
B.1.2.28 Probability distributions for the ambient point cloud outlier percentage . . . 285
B.1.2.29 Root Mean Square Error of the inliers . . . 286
B.1.2.30 Probability distributions for the Root Mean Square Error of the inliers . . . 286
B.1.2.31 Localization system computation times . . . 287
B.1.2.32 Probability distributions for the point cloud registration time . . . 287
B.1.2.33 Probability distributions for the localization system global computation time 287
B.1.3.1 Animation of point clouds using the localization system poses . . . 288
B.1.3.2 Poses estimated by the ground truth and localization system . . . 288
B.1.3.3 Point clouds assembled on top of the map using the ground truth poses . . . 289
B.1.3.4 Point clouds assembled on top of the map using the localization system poses 289
B.1.3.5 Evolution of x position over time . . . 290
B.1.3.6 Evolution of y position over time . . . 290
B.1.3.7 Evolution of z position over time . . . 290
B.1.3.8 Distance traveled between consecutive pose estimations . . . 291

B.1.3.9 Estimated linear velocity . . . 291
B.1.3.10 Estimated linear acceleration . . . 291
B.1.3.11 Angular differences between consecutive pose estimations . . . 292
B.1.3.12 Estimated angular velocity . . . 292
B.1.3.13 Estimated angular acceleration . . . 292
B.1.3.14 Localization system translation errors . . . 293
B.1.3.15 Localization system translation errors components . . . 293
B.1.3.16 Probability distributions for the localization system translation errors . . . 293
B.1.3.17 Localization system rotation errors axis . . . 294
B.1.3.18 Localization system rotation errors . . . 294
B.1.3.19 Probability distributions for the localization system rotation errors . . . 294
B.1.3.20 Translation corrections performed by the localization system . . . 295
B.1.3.21 Translation corrections components performed by the localization system . . . 295
B.1.3.22 Probability distributions for the translation corrections performed by the localization system . . . 295
B.1.3.23 Rotation corrections (axis) performed by the localization system . . . 296
B.1.3.24 Rotation corrections performed by the localization system . . . 296
B.1.3.25 Probability distributions for the rotation corrections performed by the localization system . . . 296
B.1.3.26 Point clouds size in several of the localization system processing stages . . . 297
B.1.3.27 Number of registered points . . . 297
B.1.3.28 Probability distributions for the ambient point cloud outlier percentage . . . 297
B.1.3.29 Root Mean Square Error of the inliers . . . 298
B.1.3.30 Probability distributions for the Root Mean Square Error of the inliers . . . 298
B.1.3.31 Localization system computation times . . . 299
B.1.3.32 Probability distributions for the point cloud registration time . . . 299
B.1.3.33 Probability distributions for the localization system global computation time 299

LIST OF FIGURES

List of Tables

2.1 Overview of self-localization approaches 12

5.1 LIDARs hardware specifications 41
5.2 3 DoF localization system results - (*) Most relevant experiments 46
5.3 3 DoF odometry and AMCL results - (*) Most relevant experiments 47
5.4 6 DoF results 59

Abbreviations

AGPS Assisted Global Positioning System.
AMCL Adaptive Monte Carlo Localization.
API Application Programming Interface.

CAD Computer Aided Design.
CARLoS Cooperative Autonomous Robot for Large Open Spaces.

DoF Degrees of Freedom.

EKF Extended Kalman Filter.
ESF Ensemble of Shape Functions.

FPFH Fast Point Feature Histogram.

GLONASS GLObalnaya NAvigatsionnaya Sputnikovaya Sistema.
GNSS Global Navigation Satellite System.
GPS Global Positioning System.
GPU Graphics Processing Unit.

ICP Iterative Closest Point.
IMU Inertial Measurement Unit.
IP Internet Protocol.
ISS3D 3D Intrinsic Shape Signatures.

LIDAR LIght Detection And Ranging.

MCL Monte Carlo Localization.

NDT Normal Distributions Transform.

PCA Principal Component Analysis.
PCL Point Cloud Library.
PFH Point Feature Histogram.

RADAR RAdio Detection And Ranging.
RANSAC Random Sample Consensus.
RMSE Root Mean Square Error.
ROS Robot Operating System.

SAC-IA Sample Consensus Initial Alignment.
SC3D Shape Context 3D.
SHOT Signature of Histograms of Orientations.

SIFT Scale Invariant Feature Transform.
SLAM Simultaneous Localization And Mapping.
SONAR SOund Navigation And Ranging.

TCP Transmission Control Protocol.
ToA Time of Arrival.
ToF Time of Flight.
TTFF Time To First Fix.

UDP User Datagram Protocol.
UKF Unscented Kalman Filter.
USC Unique Shape Context.

XML Extensible Markup Language.
XML-RPC Extensible Markup Language Remote Procedure Call.

Chapter 1

Introduction

This chapter provides an overview about the motivations and objectives of this dissertation along with its practical applications.

1.1 Context

Humanity has sought a reliable method of navigation ever since it started to explore the world. It began with simple landmark reference points for local travels, then perfected celestial navigation for global journeys, and when it finally conquered space, it deployed a global system for high accuracy localization. Autonomous robots face the same problem, because in order to be able to navigate with precision, they first need to know their location.

Over the years, several localization methods have been proposed and refined, according to the navigation environment and the accuracy requirements. Some are meant for high precision local navigation, while others provide an approximate global position.

A robot capable of operating safely and accurately in a dynamic environment can have innumerous applications, ranging from simple delivery tasks to advanced assembly. Besides improving productivity by performing repetitive tasks with precision and speed, robots can also act as coworkers, helping humans perform their jobs more efficiently and thus, reducing the overall production costs.

1.2 Project

Cooperative Autonomous Robot for Large Open Spaces (CARLoS)[1] is a European research project that aims to develop an autonomous robot capable of performing repetitive tasks alongside human co-workers in dynamic environments. The robot will operate in shipyards and is intended to perform fit-out operations, such as stud welding and projection mapping of Computer Aided Design

[1] http://carlosproject.eu/

(CAD) drawings. Stud welding is a repetitive task that provides structural support for other components, such as heat insulation layers or electrical systems. Projection mapping of CAD drawings or other important information will help human co-workers assemble components faster, because it will mark the exact position in which they must be installed.

1.3 Motivation and objectives

With the increase of competitiveness in the current globalized trading markets, companies are trying to reduce production costs and improve the productivity of their assets. Robots can help achieve these goals by performing the simple and repetitive jobs while giving humans more free time to perform the complex and creative tasks.

Mobile platforms equipped with robotic arms provide a flexible way to automate a wide range of tasks that must be performed over large areas. However, before performing the intended operations they first need to know where they are and how they can reach the desired location. Moreover, given their limited computational resources and energy storages, they require efficient, reliable and accurate control systems capable to operate in real time.

1.4 Contributions

This dissertation introduces an efficient, modular and extensible 3/6 DoF localization system for mobile robot platforms capable of operating accurately and reliably in dynamic environments. It is a multi-level registration pipeline that uses geometric features to estimate the initial position of a robot platform and point cloud registration algorithms to track its pose. The tracking subsystem can have two different configurations. One tuned for maximum efficiency used for the normal operation of the mobile platform and another for unlikely situations that may require more robust registration algorithms / configurations. It also supports incremental map update and can adjust its operation rate based on the estimated robot velocity. For critical operations, it provides a detailed analysis of the tracking quality and when initial pose estimation is required it gives the distribution of the acceptable poses, which can be very valuable information if there are several areas in the known map with very similar geometry.

1.5 Dissertation outline

The remaining of this dissertation is split over 5 chapters. Chapter 2 provides an overview of the main localization methods available for mobile robot platforms. Chapter 3 introduces the main frameworks used to implement the self-localization system. Chapter 4 details the architecture and ROS implementation of the proposed 3/6 DoF self-localization system. Chapter 5 evaluates the results achieved with the localization system in several testing environments. Finally, chapter 6 presents the conclusions of this dissertation and suggestions for future work.

Chapter 2

Localization methods

Self-localization is critical to any autonomous robot that must navigate the environment and requires the calculation of a 3/6 DoF pose in a given world coordinate system.

Over the years, several approaches were developed according to the precision requirements and the environment in which the robot is designed to operate. Some require support systems to calculate the position, while others are completely autonomous, allowing the robot to localize itself without any outside dependencies. Also, some systems have limited range while others can only be effective in open spaces. Moreover, several of these methods were designed to cope with errors in the localization sensors and tolerate temporary malfunctions.

The following sections will introduce some of the most used self-localization systems that can be employed in the estimation of a robot's pose.

2.1 Proprioceptive methods

Proprioceptive methods rely on internal information that the robot possesses about its own systems operation and movement in order to update the current pose estimation. As a result, they allow the robot to operate without an external support system.

Since these methods don't correct their estimations based on environment observations, they are bound to have significant cumulative errors. As such, in order to maintain an accurate estimation of a robot's pose, they are usually combined with exteroceptive systems.

2.1.1 Odometry

Odometry estimates the current pose by integrating the velocity of the robot over time. This velocity is usually calculated by measuring the number of rotations of the wheels (using optical encoders like the ones shown in figure 2.1.2.1). This method can provide a viable approximated location, as can be seen in [RKZ13].

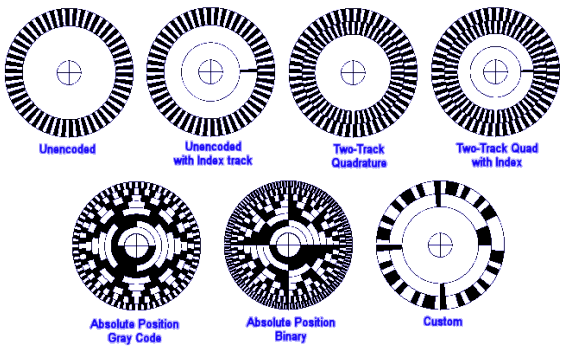


Figure 2.1.2.1: Different types of optical encoders used to measure distances[1]

2.1.2 Dead reckoning

Dead reckoning is an extension of the odometry method, in which the acceleration and angular velocity are used to improve the localization estimations.

Other sensors, such as accelerometers and gyroscopes, can also be used to improve the position estimation [Ibr10] and provide the robot orientation.

2.2 Exteroceptive methods

Exteroceptive methods use a range of sensors to analyze the environment and retrieve the necessary information to perform localization.

2.2.1 Trilateration methods

Trilateration is a geometric technique that can be employed in the calculation of absolute or relative positions using distances from known points.

For 3D localization, it usually involves the intersection of 4 or more spheres, in which their radius is the distance to known positions.

2.2.1.1 Global Navigation Satellite System (GNSS)

Global Navigation Satellite System (GNSS) such as the Global Positioning System (GPS) or GLObalnaya NAvigatsionnaya Sputnikovaya Sistema (GLONASS), allow 3D positioning in the planet Earth using a trilateration method [DKU+07].

In these systems, a constellation of satellites broadcasts a radio signal with information about its position along with the time of the message dispatch. Using this data and knowing the speed of the radio waves, the distances to the satellites can be computed.

With at least 3 satellites distances, the 3D position can be calculated, since the Earth can be used as the 4^{th} sphere (figure 2.2.1.1 shows a visual representation of the trilateration technique used in a global localization system).

[1] http://www.mindspring.com/~tom2000/Delphi/Codewheel.html

Given that the correct measurement of the distances to the satellites relies in the accurate computation of the elapsed time between the message dispatch and its reception, it is critical that both the satellites and the receiver have synchronized clocks. This is achieved by using high precision atomic clocks in the satellites and a clock reset technique in the receiver. This reset method relies on the fact that 3 satellite distances will only have a valid location if the clock of the receiver is synchronized. With this knowledge, the receiver can compute the necessary corrections to reset its internal clock to match the satellites time.

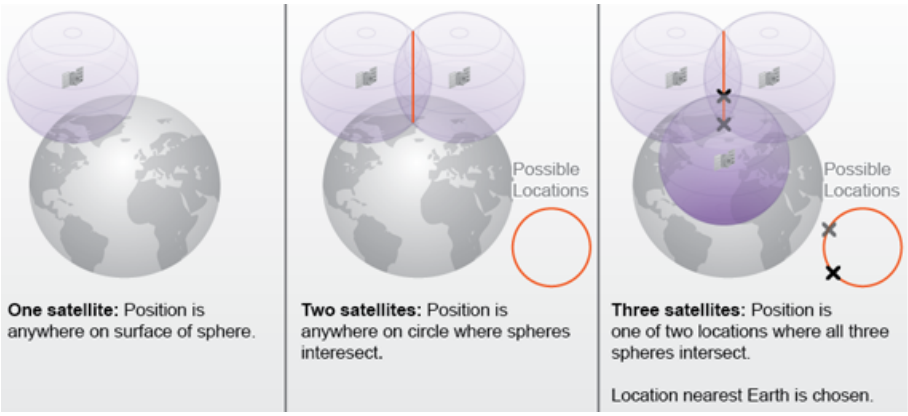


Figure 2.2.1.1: Trilateration technique in a global localization system[2]

2.2.1.2 Differential Global Positioning System (DGPS)

The accuracy of the GPS position can be increased with the help of local broadcast stations. These stations are fixed and provide information about the corrections that can be made to the satellite signals in order to improve the localization precision.

These corrections are useful to mitigate some of the ambient interference that the satellite signals face. These interferences can range from simple signal reflection in the environment landscape to the more complex interactions with the atmosphere, which can change the speed and path of the radio signals.

The computation of the corrections [KKOO07] is based on the fact that these stations are fixed, and as such, they can compare the location given by the satellite signals with their known location. With this position differential, the appropriate corrections can be calculated and broadcasted to the GPS receivers.

2.2.1.3 Assisted Global Positioning System (AGPS)

Assisted Global Positioning System (AGPS) systems are a common method used to speed up the Time To First Fix (TTFF) of a GPS receiver. They usually rely on the cellphone network to provide location estimation and signal corrections [R.48]. This information can greatly reduce the TTFF when there are few satellites visible or their signal is very weak and only temporary available.

[2] https://www.e-education.psu.edu/geog160/node/1923

2.2.1.4 Signal strength geolocation methods

Signal strength geolocation [CK02], also known as fingerprinting localization [BOG+10], is an approximate method that can be used to calculate relative positions.

It relies on the analysis of the signal attenuation from a given access point (like a Wi-Fi router or cellphone tower), to estimate distances. With enough access points (usually 4), an approximate position can be computed.

This type of distance estimation can be useful for indoor navigation, but requires a propagation model of the signal and the environment. If these models aren't accurate, then the localization precision of these methods will be very low.

Although this method is less accurate than the more recent global localization systems (such as GPS), it can be used without human made infrastructures, and as such, is a viable solution in case of temporary disruption of the GPS signal.

2.2.2 Celestial navigation

Celestial navigation [YXL11] relies on the observation of stars, planets or other reference objects, to calculate the latitude and longitude.

The calculation of the position on the surface of the Earth using celestial navigation is similar to trilateration, but in this case, angles are used instead of distances. These angles (delta), are measured between the Earth horizon and the center of the celestial object.

Having the delta, and knowing the relative position of the Earth to the reference object, along with the Greenwich hour time, it is possible to calculate a circle of position (as shown in figure 2.2.2.1).

Having at least 3 circles of position, the latitude and longitude can be computed.

Although this method is less accurate than the more recent global localization systems (such as GPS), it can be used without human made infrastructures, and as such, is a viable solution in case of temporary disruption of the GPS signal.

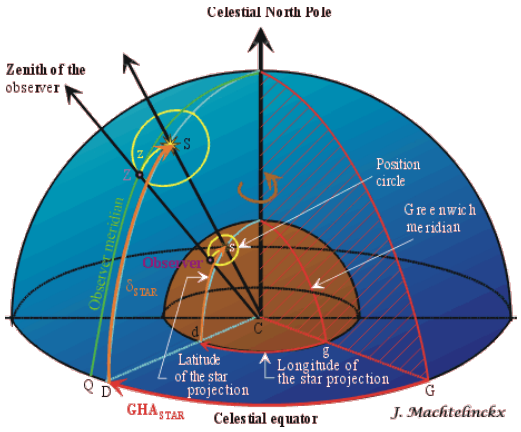


Figure 2.2.2.1: Circle of position in celestial navigation[3]

2.2.3 Landmark methods

Landmark methods [LKP06] can be used to perform relative localization, and are very useful to reduce the required information for navigation.

In these methods a database of markers / environment geometry is stored along with its location, and when the robot recognizes one of these markers, it corrects its proprioceptive methods measures.

It is a simplification of the method that will be presented in the next section, and it is useful for environments that have unique geometry in key positions of the navigation map.

2.2.4 Point cloud methods

Point cloud localization methods can be used to perform relative localization by finding the best point cloud match between the environment and the know map (figure 2.2.4.1 shows its application to small objects). These methods require a 2D or 3D representation of the environment and tend to be used in conjunction with proprioceptive methods (to have an estimation of movement), and also with probabilistic methods (when the point cloud acquisition location is not known).

One of the most used algorithms for 3D point cloud matching is the Iterative Closest Point (ICP) [BM92, Jez08, Zha94, BTP13, CSSK02, DZA+13, ZL11]. It is an iterative algorithm that finds the translation and rotation transformation that minimizes the distances of the corresponding points on both clouds. It finishes when the matching error is below a given value, the matrix transformation between iterations has a translation / rotation below the specified thresholds or when the maximum allowed iterations is reached.

There are several variants that optimize different parts of the algorithm [RL01].

The main steps for each iteration of the ICP algorithm are presented below.

1. Selection of points in one or both point clouds (source and reference clouds)
2. Matching / pairing source points to reference points
3. Weighting the corresponding pairs
4. Rejecting low quality matches (outliers)
5. Assigning an error metric based on the point pairs
   (a) Usually mean square error based on points distance

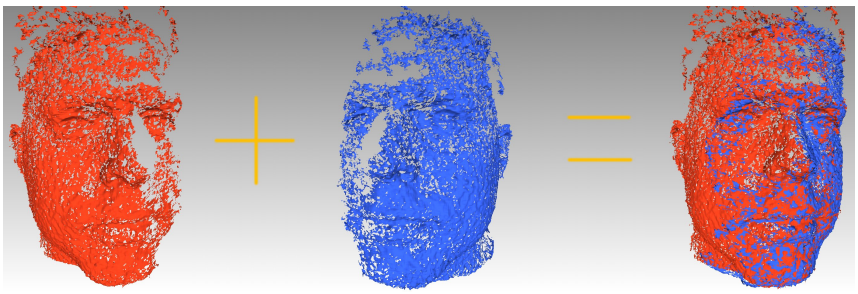


Figure 2.2.4.1: ICP point cloud matching[4]

[3] http://onboardintelligence.com/CelestialNav/Celnav2.aspx

2.2.5 Probabilistic methods

Probabilistic methods aim to reduce the impact of sensor accumulated errors or even temporary malfunctions by using Bayesian estimations and Markov processes.

2.2.5.1 Monte Carlo Localization (MCL)

Monte Carlo Localization (MCL) (also known as particle filter), is a global localization algorithm that estimates the position and orientation of a robot by analyzing and adjusting the distribution and weights of state particles on a given environment [BOG+10, AMGC02, BGFM10, Che03, FBDT99, SS09].

It starts by randomly distributing the state particles on the map, and over time it changes their position and weight according to new sensor readings. The probable location of the robot will be in the area of the map that has the largest cluster of state particles. The figures below show the evolution of the state particles distribution during the robot movement in the environment, and illustrates how the new sensor readings changed the particles clusters[5].

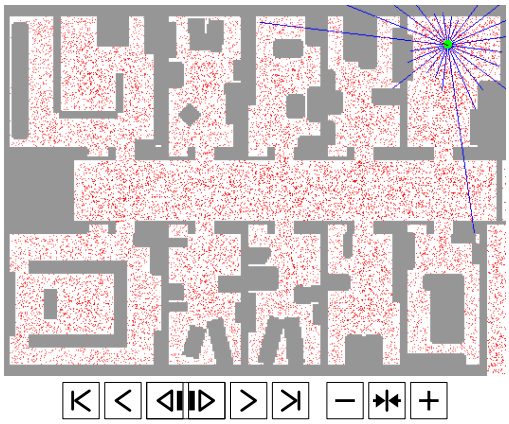


Figure 2.2.5.1: MCL animation

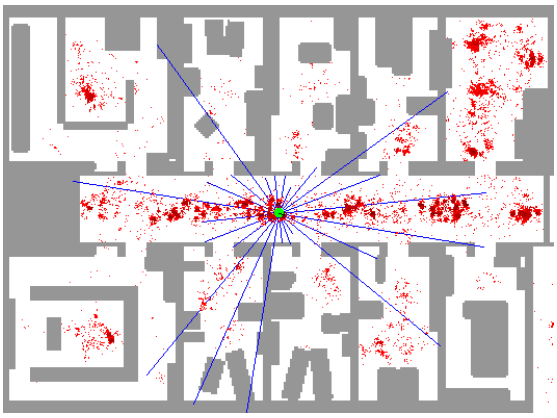


Figure 2.2.5.2: MCL redistribution of particles

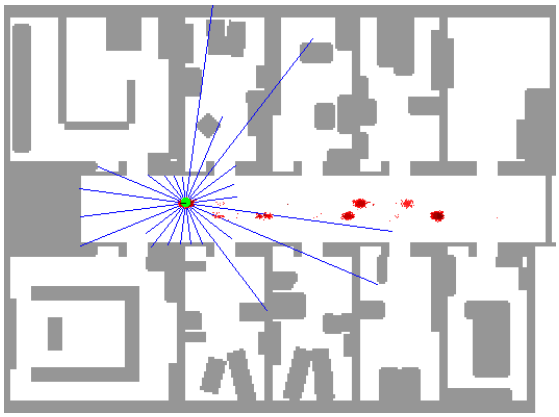


Figure 2.2.5.3: MCL position refinement

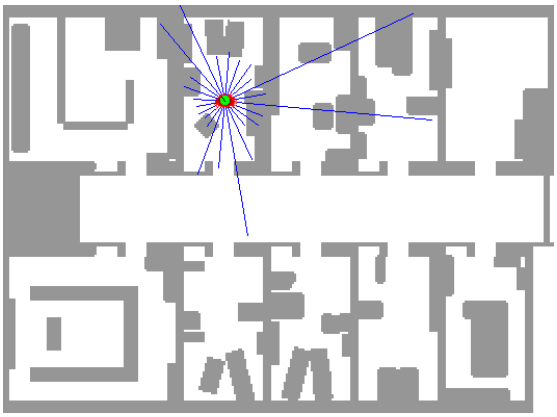


Figure 2.2.5.4: MCL position estimation

[4] http://dynface4d.isr.uc.pt/database.php
[5] http://www.cs.washington.edu/robotics/mcl/

2.2.5.2 Kalman filters

Kalman filters [Kal60] are probabilistic algorithms that estimate a given system state even when it is affected by noise or other errors. They perform linear quadratic estimations to achieve optimal results and can be efficiently implemented to be used in real time systems. They are recursive algorithms based on Marcov processes, and as a result, they only need to know the current system state in order to perform measurement corrections.

The Extended Kalman Filter (EKF) [EW99, Rib04, IKP10, LYL11] is a variant of the Kalman filter, designed to handle non-linear systems by performing linear approximations to the state variables. These approximations may lead to divergence in the estimations, and as such, the EKF can't guarantee optimal results.

The Unscented Kalman Filter (UKF) [JU97, WM00] is another variant of the Kalman filter that was designed for highly non-linear systems. It usually achieves better results than EKF due to its unscented transform.

For the particular case of localization, these algorithms start with an initial estimation of the system state, and for each new position (computed from the sensors data), they predict the estimated robot location (according to the Bayes estimation model and the Gaussian distribution of errors), and then update their internal model of the system (mean and covariance) to incorporate the system evolution.

In figure 2.2.5.5 can be seen that the UKF estimated position (red) is closer to the real position (blue) than the raw sensor measurements (green).

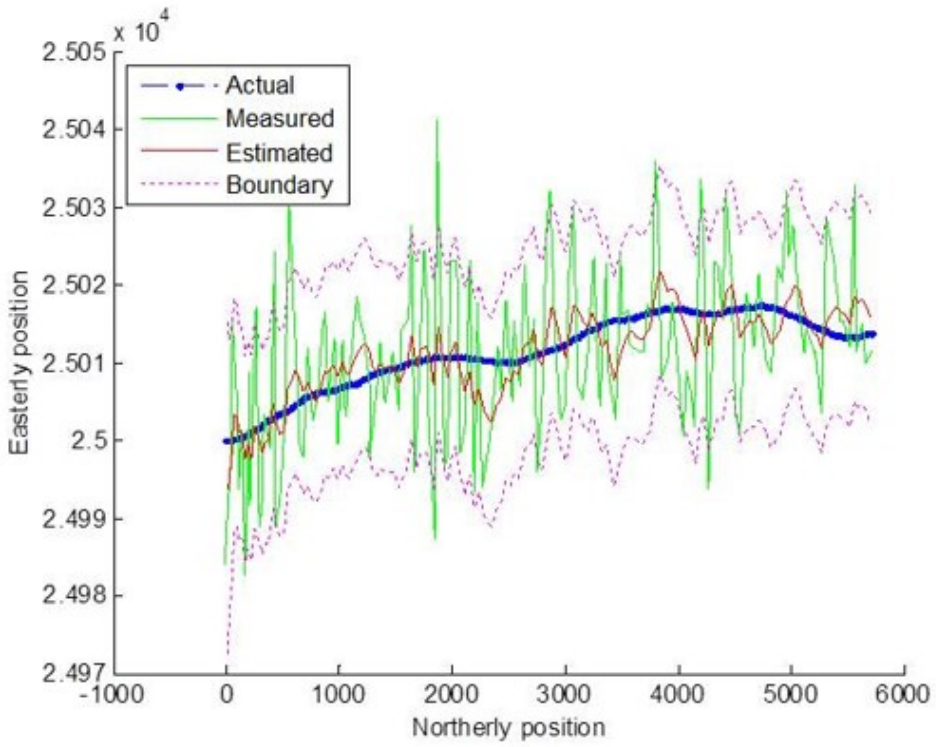


Figure 2.2.5.5: Unscented Kalman Filter[6]

2.2.5.3 Perfect Match

The Perfect Match [LLRM06, PM63] is an efficient self-localization algorithm that is largely used in the Robocup Robotic Soccer Mid Size League. Its main goal is to minimize the localization error by carefully analyzing the know map and selecting the most probable current position using a gradient descent approach. To improve tracking accuracy, the algorithm also uses a stochastic weighted approach.

With the proper configuration, it can achieve a localization accuracy similar to the particle filter, while using about ten times less computations.

An example of the position estimation can be seen in the figures below. Figure 2.2.5.6 shows the probable locations when the robot detects a line on the floor, and figure 2.2.5.7 illustrates their associated errors (brighter areas indicate smaller error). By using a gradient descent, the most probable location was selected (black circle).

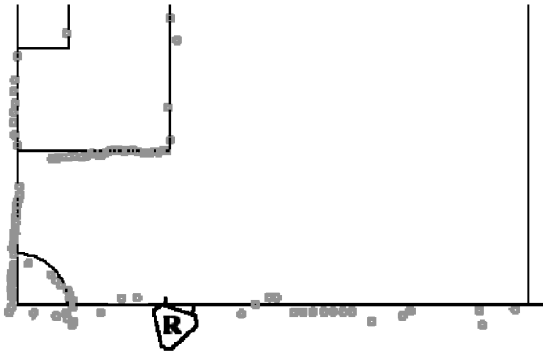


Figure 2.2.5.6: Position estimate

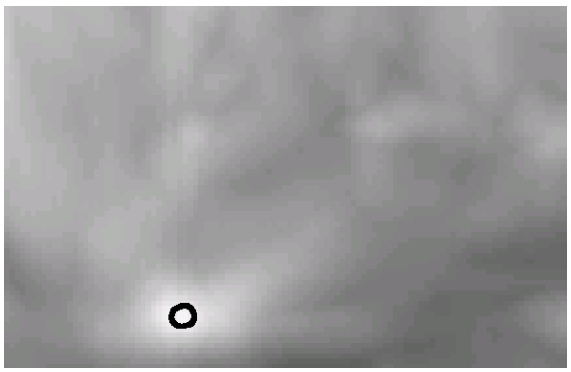


Figure 2.2.5.7: Position associated error

2.2.6 Simultaneous Localization And Mapping (SLAM)

Simultaneous Localization And Mapping (SLAM) [Thr02] is a very effective approach to either explore unknown environments or update a known map. It relies on both proprioceptive and exteroceptive methods to perform localization and mapping. It is also very useful to make the robot navigation more robust in dynamic environments, in which the topology of the world may change considerably over time.

There are numerous approaches to perform SLAM [TGCV12]. Some optimized for exploration and others for map improvement. For the exploration tasks, proprioceptive methods play a critical role, and are usually paired with probabilistic methods, such as Kalman filters, in order to reliably map the environment. For map correction and improvement, several probabilistic methods can be employed according to the precision required. For high accuracy 3D mapping, the ICP algorithm can be used to build accurate point clouds of the environment.

[6] http://www.lzcheng.com/courseworks/kalmanfilter

2.3 Summary

This chapter introduced several localization systems that can be used in different types of environments and with different degrees of accuracy. It started with the simple proprioceptive methods and then moved on to the more robust exteroceptive approaches.

Some techniques can be combined to improve the pose estimate or to make the localization more efficient to a particular type of environment.

For outside tasks, a GNSS approach can give accurate localization estimations with very little computation cost. On the other hand, indoor localization requires more advanced techniques in order to infer the current position based on the analysis of the robot surroundings. These techniques usually start with an estimation of the robot movement and then refine it with probabilistic or geometric methods.

Table 2.1 presents an overview of the mentioned localization techniques.

Table 2.1: Overview of self-localization approaches

Method	*Ideal environment*	*Accuracy*	*Operational cost*	*Computational cost*	*Notes*
Odometry	Any	Low	Low	Low	Position estimation is affected by cumulative errors
Dead reckoning	Any	Low	Low	Low	Pose estimation is affected by cumulative errors
GNSS	Outside	Medium	Low	Low	Requires a clear line of sight to at least 3 satellites
Signal strength	Any	Low	Low	Low	Requires an accurate model for the signal attenuation
Celestial	Outside	Low	Low	Low	Requires a clear view of the celestial objects and a nautical almanac
Landmark	Any	Medium	Low	Medium	Requires a database of landmarks
ICP	Indoors	High	Medium	High	Requires a detailed 3D representation of the environment
MCL	Indoors	Medium	Medium	Medium	Inefficient for large areas
Kalman filters	Any	Medium	Low	Low	Useful to improve estimations of other methods
Perfect Match	Any	Medium	Low	Low	Not ideal for large or dynamic environments
SLAM	Indoors	Medium	Medium	High	Adapts well to dynamic environments

Chapter 3

Relevant software / hardware technologies

Robot self-localization in complex environments is a multidisciplinary problem that requires advanced computer software systems. In order to speed up the implementation and deployment of the localization system, several frameworks and libraries were used. Among the most important were ROS for the system architecture, Point Cloud Library (PCL) for the point cloud processing and Gazebo for simulation and testing.

3.1 ROS

ROS[1] [QCG+09] is a software framework designed to ease the development of robot systems. It provides seamless integration between hardware drivers and software modules, allowing a fast transition between simulation and deployment.

Figure 3.1.0.1: ROS logo

It's an open source project that offers a distributed computing framework with several core libraries and development tools that aims to speedup software prototyping, testing and deployment.

3.1.1 Architecture

The ROS architecture was designed from the beginning to be a distributed peer-to-peer software framework that could be deployed in several operating systems and implemented in a range of different programming languages. However, given that most of the ROS community prefers open source software, the Ubuntu[2]] operating system is the main developing and testing environment, and as such, the recommended choice for ROS developers. Moreover, considering that robotics

[1] http://www.ros.org/
[2] http://www.ubuntu.com/

research requires software with both performance and maintainability at its core, the C++[3] programming language is used in most of the available packages, along with Python[4] and Java[5].

Being a distributed computing framework, ROS relies in network connections and exchange of messages to perform the intended tasks. As such, its architecture was developed to follow a publish / subscribe pattern (ROS topics[6]) and request and reply communication paradigm (ROS services[7] and actions[8]). This allows ROS nodes[9] (operating system processes) to be deployed in different computing platforms with ease and simplifies testing and exchange of software modules.

The next sections provide a more detailed description of the main ROS architecture concepts.

3.1.1.1 Nodes

ROS nodes are operating system processes that are part of the peer-to-peer communication graph. They are the fundamental building blocks of any ROS system and can be spread among several computing platforms.

In order to manage the communications between nodes, the ROS framework provides a master node (roscore[10]) that uses Extensible Markup Language Remote Procedure Call (XML-RPC) to maintain a communication graph of the system. This allows nodes to be started without knowing the location (Internet Protocol (IP) address and port) of the other nodes in the network and greatly simplifies their integration and exchange. Also, by using ROS launch files[11] (Extensible Markup Language (XML) configuration files), the specification of a system communication data flow and configuration can be easily altered.

Besides handling communications, the master node also manages system configurations through the parameter server. This allows nodes to share and change the system configuration at runtime. However, given that the parameter server is usually queried only when a node starts up, the dynamic reconfigure[12] Application Programming Interface (API) can be used instead, if it is necessary to change the configuration of a node when it is already running. This is achieved by providing a callback that is asynchronously called when a configuration change is requested.

The flexibility provided by ROS in both module integration and exchange can greatly speedup testing and deployment and the possibility of changing the configuration of the system at runtime and restart software modules individually (nodes) is very useful when implementing system supervisors and recovery behaviors.

[3] http://www.cplusplus.com/
[4] https://www.python.org/
[5] https://www.java.com
[6] http://wiki.ros.org/Topics
[7] http://wiki.ros.org/Services
[8] http://wiki.ros.org/actionlib
[9] http://wiki.ros.org/Nodes
[10] http://wiki.ros.org/roscore
[11] http://wiki.ros.org/roslaunch
[12] http://wiki.ros.org/dynamic_reconfigure

3.1.1.2 Nodelets

A nodelet[13] is a special kind of node that aims to reduce the overhead of message exchange. This overhead can be significant when messages are very large, such as point clouds or video. To mitigate this problem, the nodelets exchange pointers to shared memory regions, instead of sending the entire messages between nodes.

To achieve this overhead reduction, some architecture changes are required. The most important being the use of threads instead of processes, and the creation of a superclass that all nodelets must inherit. This leads to the creation of plugin libraries for each nodelet instead of an executable for each node.

Another important change is the introduction of nodelet managers to allow loading and setup of nodelets in different threads inside the same process.

In terms of implementation, the transition from nodes to nodelets requires few code changes and can lead to a significant improvement of the overall system performance.

3.1.1.3 Topics

ROS topics are named communication buses that follow the publish / subscribe pattern. They provide a simple method for exchanging messages between nodes and allow the decoupling of information production and consumption. This is useful when there are multiple sources of the same information or there are multiple consumers that are interested in processing the same data for different purposes. Moreover, this communication architecture allows to log and replay the exchanged messages, which can be helpful to test different algorithms with the same data.

Currently, topics can use either the Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) protocols to exchange messages. The TCP implementation is used by default and creates a bidirectional channel between each producer and subscriber while guaranteeing the delivery of all messages. The UDP implementation uses an unreliable and stateless transport approach, in which a subscriber listens to a given broadcast address, and has no guarantee that will receive all messages. As such, TCP should be used when all messages must be processed, and UDP should be considered when the latency and the TCP overhead are important issues.

3.1.1.4 Services

ROS services are named communication buses that follow the request and reply paradigm, in which a node asks for a given service and receives a response according to the data that was sent in the request message and the state of the service node. They are useful to query other nodes state or to request the execution of some behavior / action.

[13] http://wiki.ros.org/nodelet

3.1.1.5 Actions

ROS actions are a special kind of service in which the progress of the request can be queried. They are very useful when the request might take a long time, and gives the caller the necessary information to supervise the execution of the request and if necessary, terminate its execution.

3.1.2 Build system

The latest ROS build system is named catkin[14], and is the successor of the original rosbuild[15] system. It combines CMake[16] macros and Python scripts to allow building multiple dependent projects at the same time. It is a cross-platform build system, organized in packages and meta-packages (group of packages). Each package is a software module that can produce libraries or binaries from source code.

Catkin was designed to deal with complex build configurations, which in the case of ROS packages involves a considerable amount of build dependencies for each project. As such, catkin provides a build system that can easily find, build and link both ROS and system dependencies. Moreover, it provides install targets to allow faster code releases and simplifies builds from source for the final users.

Other useful features of the catkin build system are the concepts of workspace and overlays. Catkin uses a workspace with out-of-source builds to keep the source code separate from the build files. This allows the code directory structure to be clean of compiler generated files that are platform dependent. Moreover, it simplifies the concurrent usage of packages (overlay), because catkin gives priority to workspace packages (in relation to system packages). This is particularly useful when it is necessary to modify and test some package that has been released and is installed in the system (without having to uninstall the stable release of that package).

Finally, catkin is a cross-platform build system that can be used to build other projects that use CMake and are not related to ROS.

3.1.3 Development tools

ROS provides several development tools that allow introspection and visualization of the system state. They are very useful for testing, debugging and profiling.

The next sections give an overview of the most important ROS tools that are currently available.

3.1.3.1 Graphical User Interface tools

The ROS development tools that have graphical user interfaces are aggregated in the rqt framework[17], and are loaded as plugins at runtime (some can be started as standalone applications).

[14] http://wiki.ros.org/catkin
[15] http://wiki.ros.org/rosbuild
[16] http://www.cmake.org/
[17] http://wiki.ros.org/rqt

Currently there is plugins to visualize sensor data (rviz[18]); introspect the contents of topics; log and replay ROS messages (rosbag[19]); display node, package and coordinate systems graphs; list and filter debug messages; change configuration of running nodes (dynamic reconfigure); monitor nodes memory and processor usage and much more.

3.1.3.2 Command line tools

The ROS command line tools[20] are split across several executables and can be used for advanced introspection (rosnode, rostopic and rosservice), system configuration (rosparam), package building and management (catkin, rosdep and rosinstall) and also to search ROS message types documentation (rosmsg and rossrv).

Finally, it is available a diagnostics tool (roswtf), that can detect packages / dependencies issues and configuration problems.

3.2 PCL

The PCL[21] [RC11] is an open source project that provides algorithms for processing point clouds. These algorithms can be used to filter and register point clouds as well as perform object segmentation, recognition and tracking.

Figure 3.2.0.1: PCL logo

Figure 3.2.0.2 gives an overview of the main modules currently available in PCL.

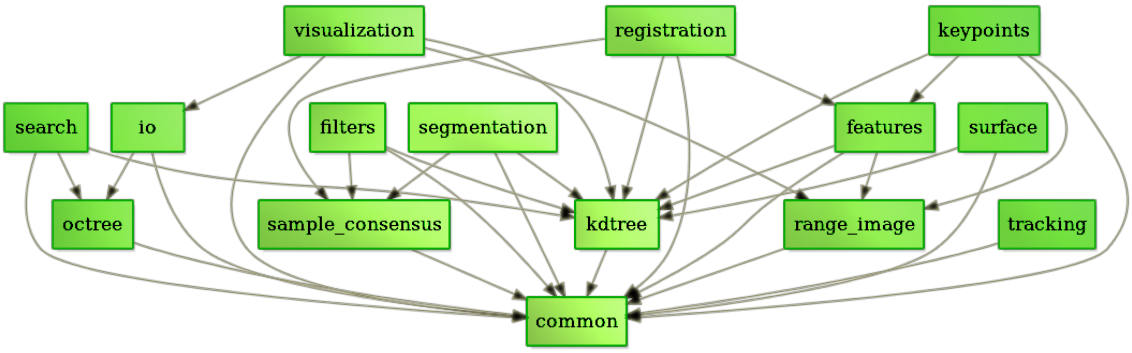


Figure 3.2.0.2: Point Cloud Library[22]

[18] http://wiki.ros.org/rviz
[19] http://wiki.ros.org/rosbag
[20] http://wiki.ros.org/ROS/CommandLineTools
[21] http://pointclouds.org/
[22] http://pointclouds.org/about/

3.3 Gazebo

Gazebo[23] is a 3D multi-robot simulator capable of generating hardware sensor data for different kinds of robots while providing a realistic environment with physics simulation and 3D visualization. It is very useful to speedup testing of algorithms with different types of robots and environments.

Figure 3.3.0.3: Gazebo logo

Figure 3.3.0.4 shows how Gazebo can be used instead of a real robot, without requiring any implementation code modification (because it implements the same ROS interfaces that the hardware drivers use).

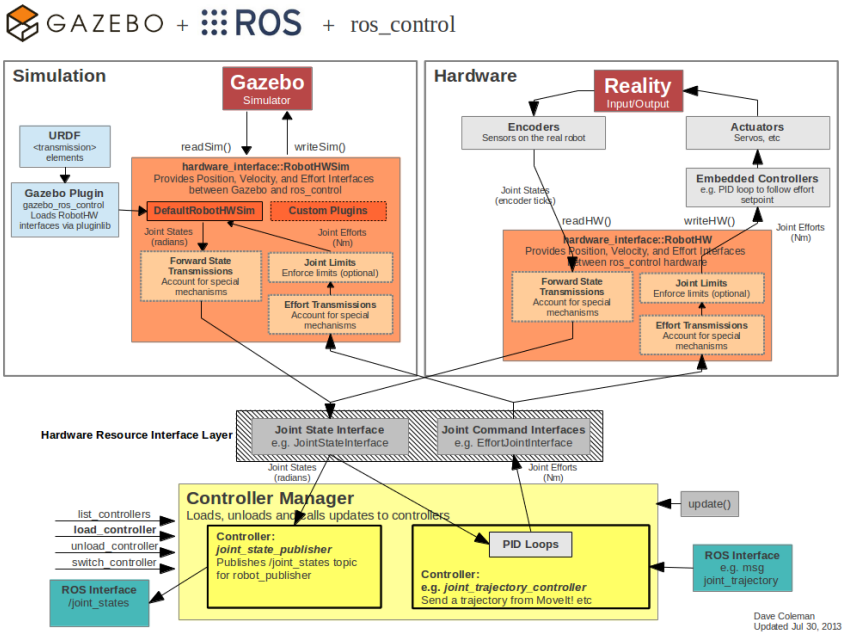


Figure 3.3.0.4: Integration of ROS and Gazebo[24]

[23] http://gazebosim.org/

[24] http://gazebosim.org/tutorials?tut=ros_control

3.4 Point cloud acquisition

Point clouds can be retrieved with a wide range of sensors with varying levels of precision and assembly time [STD09]. The next sections provide a brief overview of the three main groups of methods capable of generating point clouds.

3.4.1 Structured light methods

Structured light methods can retrieve 3D geometry from images by projecting a known pattern into the environment and analyzing its deformation (example in Figure 3.4.1.1). They can achieve sample rates of 30 Hz, and besides 3D geometry they can also retrieve color information.

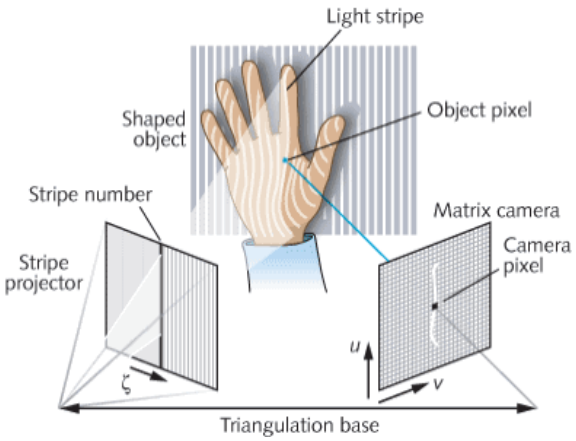


Figure 3.4.1.1: Structured light system diagram[25]

The Kinect 2[26] (seen in figure 3.4.1.2) is an example of a structured light system and can achieve measurements with millimeter accuracy for objects close to the sensor. Another similar sensor is Structure IO[27] which is intended for mobile devices and can be seen in figure 3.4.1.3.

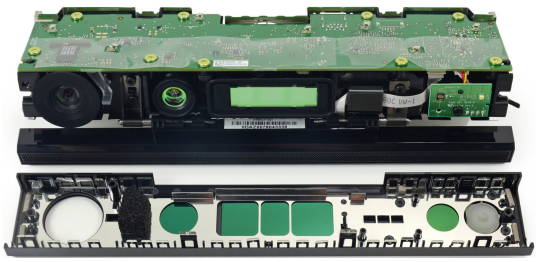


Figure 3.4.1.2: Kinect 2 sensor[28]

Figure 3.4.1.3: Structure IO sensor[29]

[25] http://www.laserfocusworld.com/articles/2011/01/lasers-bring-gesture-recognition-to-the-home.html
[26] http://www.microsoft.com/en-us/kinectforwindows/
[27] http://structure.io/
[28] https://www.ifixit.com/Teardown/Xbox+One+Kinect+Teardown/19725
[29] http://structure.io/press

3.4.2 Time of Flight methods

Time of Flight (ToF) or Time of Arrival (ToA) methods can be used to calculate distances based on the amount of time that a given wave takes from the moment it is created to the moment it is received. By assembling a large amount of sensor readings a 3D representation of the environment can be achieved.

Since these systems rely on active interaction with the environment, they can be used without being affected by lighting interferences. Nevertheless, they should take in consideration the conditions in which the waves propagate and also the geometry of the environment, because it can affect the path that the waves take, and as a result, can lead to the decrease of precision in the measurements.

3.4.2.1 Light waves

Light waves generated with lasers can estimate distances with millimeter accuracy at long ranges (system operation overview in figure 3.4.2.2) and their sensors usually have a low sample rate (below 20 Hz). Systems like LIDAR (2D sensor example in figure 3.4.2.4 and for 3D in figure 3.4.2.5) take advantage of this fact and can be used to obtain a very detailed 3D point cloud of the environment (like the one showed in figure 3.4.2.1). Moreover, some LIDARs can also capture the environment reflectivity / intensity besides their 3D geometry. On the other hand, ToF cameras (example in figure 3.4.2.3) have a very high sample rate (30 Hz or even higher), but have much more measurement error (greater than 1 cm). However, these methods allow the mapping of the environment with low latency, which can be a critical requirement in robots that must react very fast to changes in their surroundings, such as autonomous cars [MCB10].

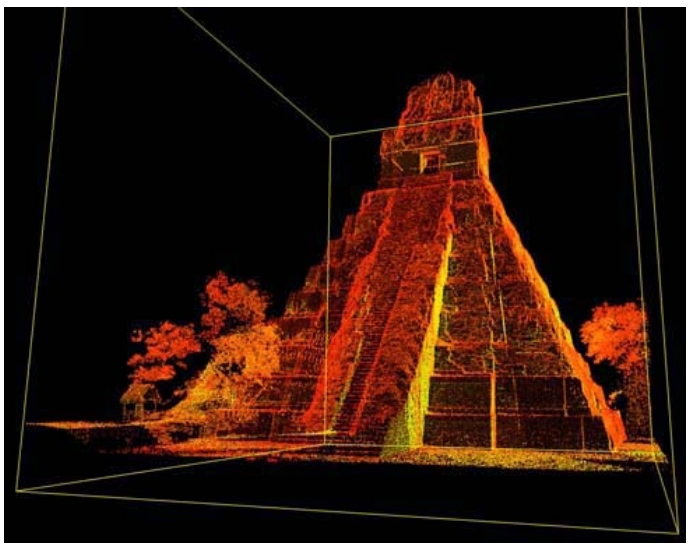


Figure 3.4.2.1: LIDAR scan[30]

[30] http://blogs.scientificamerican.com/cocktail-party-physics/2012/03/12/l-is-for-lidar/

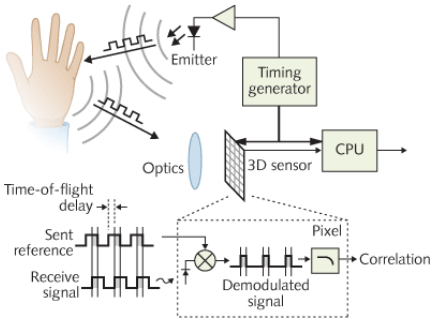


Figure 3.4.2.2: Time of Flight system[31]

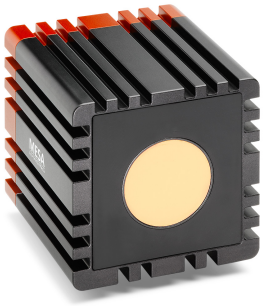


Figure 3.4.2.3: Mesa SR4000[32]

Figure 3.4.2.4: 2D SICK NAV 350[33]

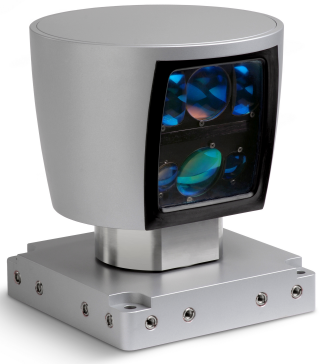


Figure 3.4.2.5: 3D Velodyne HDL-64E[34]

3.4.2.2 Radio waves

Similar to LIDAR, radio waves can be used to calculate distances using the ToF technique. Systems like RAdio Detection And Ranging (RADAR) provide an effective way to calculate the distance, altitude, direction and speed of objects that can be used as landmarks in navigation.

Like any electromagnetic wave localization method, it must take in consideration ambient interferences and even jamming, in order to validate the obtained measures. Moreover, some types of materials with a given geometric configuration might be invisible to RADAR, and as such, critical localization systems might have to employ some additional techniques to ensure the correct mapping of the robot surroundings.

Since RADAR has a less focused beam than LIDAR, it can have considerable less accuracy, as can be seen in figure 3.4.2.6. Nevertheless, it can be an effective method to avoid obstacles [WY07].

[31] http://www.laserfocusworld.com/articles/2011/01/lasers-bring-gesture-recognition-to-the-home.html

[32] http://www.mesa-imaging.ch/products/product-overview/

[33] https://www.sick.com/de/en/detection-and-ranging-solutions/2d-laser-scanners/nav/nav350-3232/p/p256041

[34] http://www.velodynelidar.com/lidar/hdlproducts/hdl64e.aspx

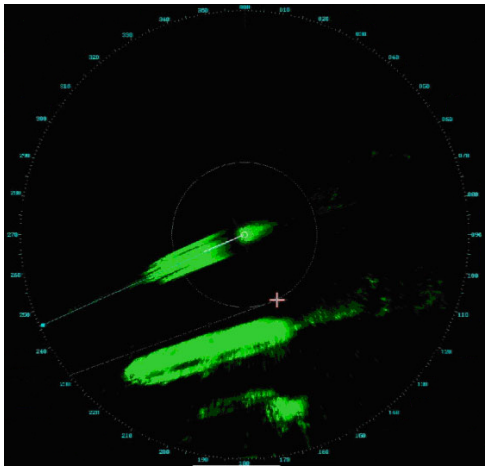


Figure 3.4.2.6: RADAR scan of two ships[35]

Figure 3.4.2.7: QT50R-AFH RADAR[36]

3.4.2.3 Sound waves

Another type of 3D sensor that can be used to map under water environments relies in the acoustic analysis of the reflections of sounds in the surrounding objects. Like the previous methods, SOund Navigation And Ranging (SONAR) can actively scan the environment to calculate the locations of the objects using a ToF technique (as can be seen in figure 3.4.2.8). Although this method is usually applied to underwater mapping, it can also be used in other sound propagation environments, such as air [GWG13].

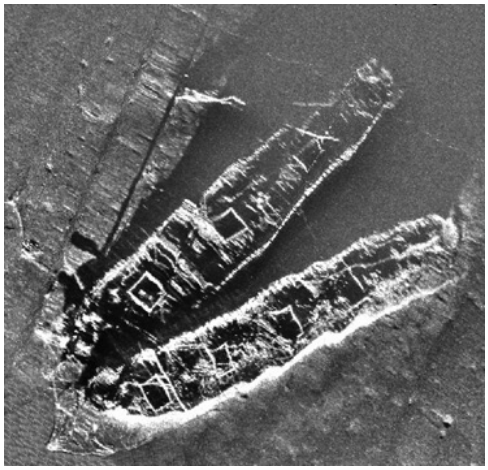


Figure 3.4.2.8: SONAR scan of two ships[37]

Figure 3.4.2.9: SONAR sensor[38]

[35] http://www.sintef.no/Projectweb/STSOps/News/Operational-Aspects-on-Decision-making-in-STS-Lightering

[36] http://www.bannerengineering.com/en-US/products/8/Sensors/658/Radar-Sensors/617/R-GAGE-QT50R-AFH-Adjustable-Field%2C-High-Sensitivity-Sensor/

[37] http://stellwagen.noaa.gov/maritime/palmercrary.html

[38] http://www.parallax.com/product/28015

3.4.3 Stereo vision

Stereo vision systems (example in figure 3.4.3.1) can generate 3D representations of the environment by comparing the displacement of corresponding points in the two ambient images (figure 3.4.3.3 gives an overview of such a system). This can be achieved because the relative position of the cameras is known. As such, points farther away will have smaller displacement between images than points closer to the cameras. In the end, a disparity image is obtained, that can then be converted to a point cloud representation of the environment.

Given that the accuracy of the disparity image relies heavily in the correct matching of points between the left and right image, some stereo vision systems employ active observation by projecting a pattern into the environment in order to refine the point matching (example of hardware setup in figure 3.4.3.2). This can significantly improve the accuracy if the environment has a lot of smooth surfaces with homogeneous colors.

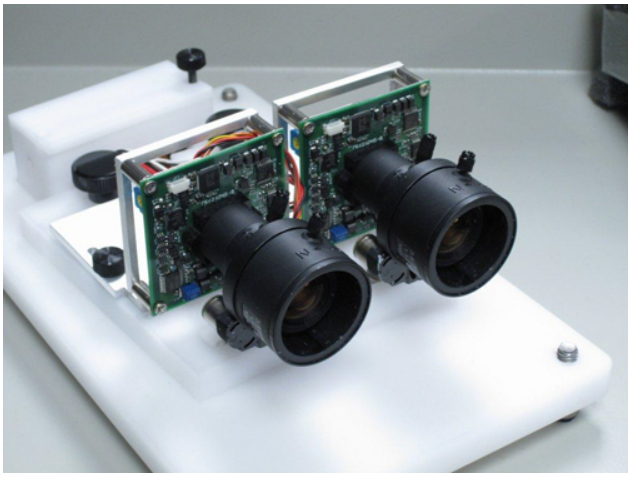


Figure 3.4.3.1: Stereo vision system [Kac13]

Figure 3.4.3.2: PR2 head capable of active stereo vision[39]

[39] https://www.willowgarage.com/pages/pr2/overview

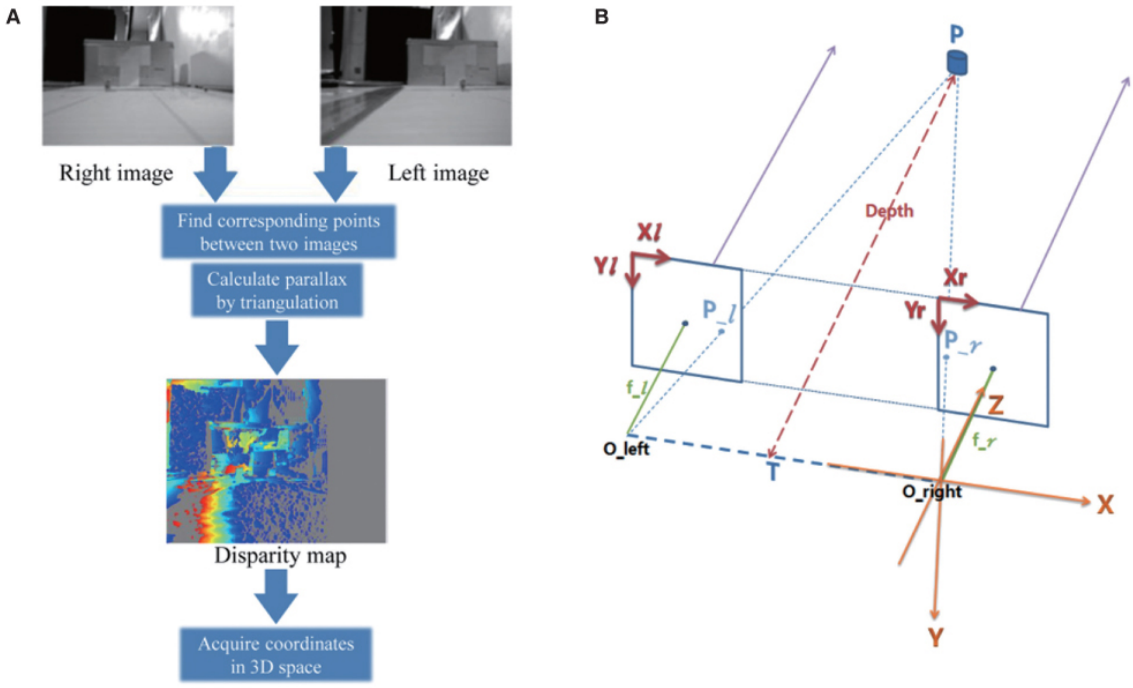


Figure 3.4.3.3: Stereo vision overview [YPH+14]

Chapter 4

Localization system

This chapter details the ROS implementation of the proposed 3/6 DoF self-localization system[1]. It starts with an overview of the main processing stages and then details the control flow and algorithms within the localization pipeline.

4.1 Overview

The self-localization system was implemented as a ROS package and provides 3/6 DoF localization by publishing *geometry_msgs::PoseStamped*[2] and *geometry_msgs::TransformStamped*[3] messages along with a detailed analysis of the pose estimation and registered point cloud (split into inliers and outliers). Moreover, it also gives detailed analysis of the computation runtime of each of its modules in order to pinpoint which algorithms are using more computation resources, which is very useful information when configuring or upgrading the system.

The ROS implementation can receive sensor data through *sensor_msgs::PointCloud2*[4] messages and as a result it can directly use data from RGB-D and ToF cameras. To use LIDARs it provides an assembler that can produce point clouds by merging measurements from several sensor scans using spherical interpolation. As

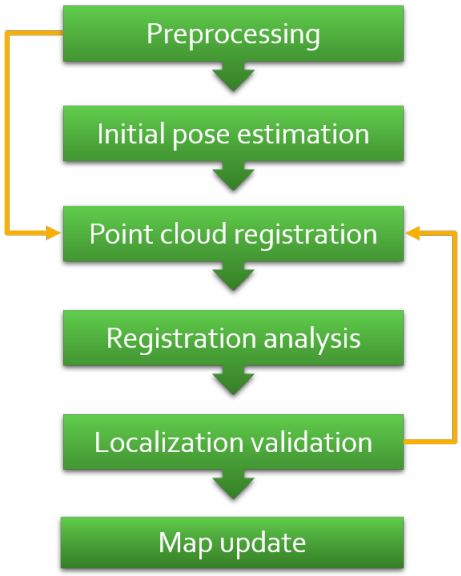


Figure 4.1.0.1: Localization system brief overview

[1] https://github.com/carlosmccosta/dynamic_robot_localization
[2] http://docs.ros.org/api/geometry_msgs/html/msg/PoseStamped.html
[3] http://docs.ros.org/api/geometry_msgs/html/msg/TransformStamped.html
[4] http://docs.ros.org/api/sensor_msgs/html/msg/PointCloud2.html

such, if the LIDAR sensors are mounted on tilting plat-
forms, they can emulate a 3D sensor and retrieve a very detailed view of the environment.

The self-localization system has a modular software architecture and was implemented as several C++ templated shared libraries that can be easily used for other applications besides robot self-localization. As can be seen in figures 4.1.0.1 and 4.1.0.2, it is an extensible and flexible system able to fit the needs of a wide range of mobile platforms. It can be configured as a tracking system, with or without pose recovery and can also have initial pose estimation using feature detection and matching. Moreover, it can dynamically create and update the map if necessary.

It supports two configurable processing pipelines in order to allow fast deployment of robots in large environments. One to process new reference maps and another to localize a mobile robot platform using ambient point clouds. This enables the loading of either processed or unprocessed referenced point clouds and allows a navigation supervisor to dynamically provide the relevant map sections based on the robot position (in order to reduce the computation resources needed).

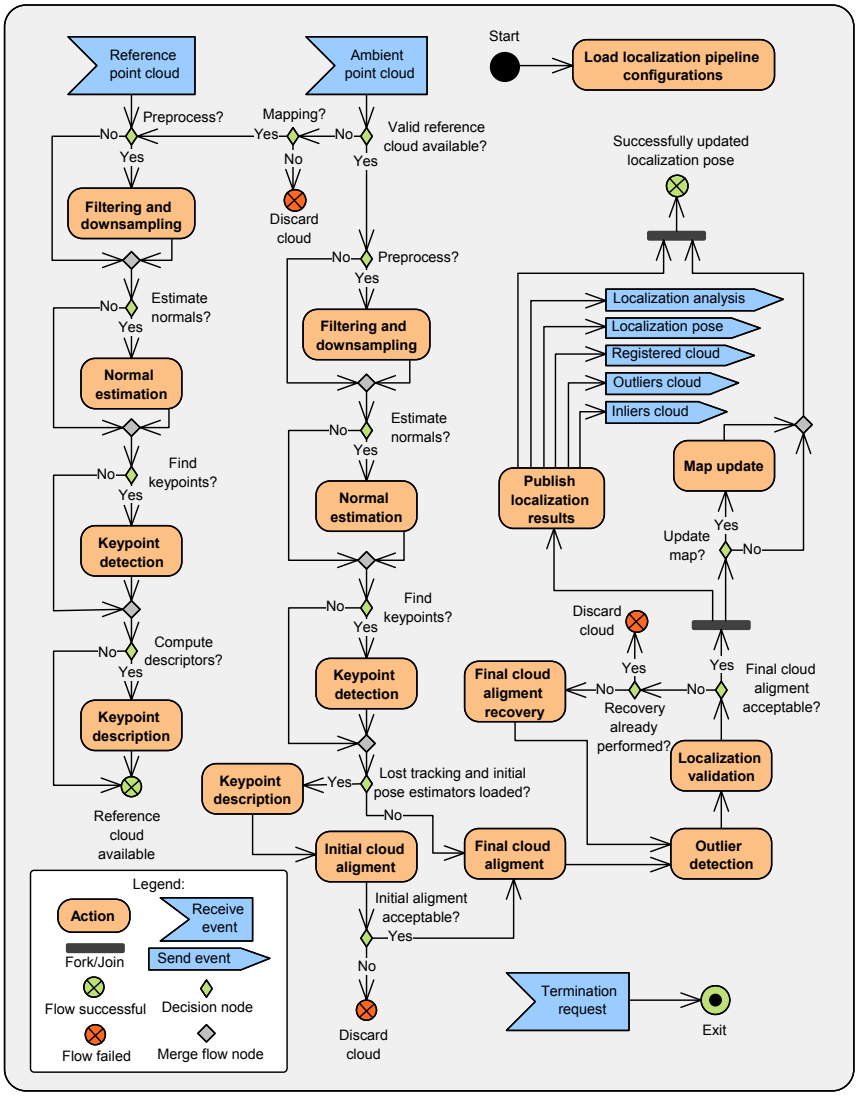


Figure 4.1.0.2: Localization system overview

4.2 Pipeline configuration

The self-localization system was designed to allow fast reconfiguration[5] and parameterization through the use of yaml[6] files and the ROS parameter server[7]. This gives the possibility to quickly tune the localization system to the specific needs of a given mobile platform moving in a particular environment, in order to use the least amount of computational resources possible and without requiring any reprogramming or source code modification. Nevertheless, the system can use a generic configuration if hardware resources are not a concern.

In a typical configuration (following the *Yes* paths of the activity diagram in figure 4.1.0.2), the first time the localization system is used, it receives a raw reference point cloud that is preprocessed and saved to long term memory along with its associated keypoints and keypoint descriptors. This allows a much faster startup the next time the localization system is initialized. After having a reference point cloud, the localization system will estimate the robot pose periodically by analyzing the ambient point cloud sensor data. This data can be preprocessed with several filters and can be associated with computed surface normals. The robot pose estimation is performed by applying a matrix transformation correction to the current robot pose and is based on the registration of the ambient point cloud with the know map. This registration can use a tracking algorithm configuration tuned for efficiency and a second configuration for tracking recovery purposes. These tracking algorithms require a initial pose estimation, and as such, if one isn't available, a third configuration can be employed to estimate the global position of the robot based on geometric features of the environment. The switch between these configuration is based on the analysis of the registered cloud metrics, such as outlier percentage, inliers root mean square error, inliers angular distribution and the registration corrections performed on the ambient point cloud. After successfully performing the robot pose estimation, the map can be updated by either integrating the full registered point cloud, its inliers or its outliers. Finally, like most ROS nodes, the localization system will stop its execution when it receives a termination signal request.

The next sections explain in detail the architecture and algorithms used in each of the processing modules present in figure 4.1.0.2.

4.3 Reference map

The reference point cloud can be loaded from a CAD file, point cloud file or dynamically arrive trough a ROS topic as either a 3 DoF *nav_msgs::OccupancyGrid*[8] or 6 DoF *sensor_msgs::PointCloud2*[9]. This allows a localization supervisor to give only sections of a global map in order to use the least amount of memory and processing power (very large maps have deeper search structures, such as kd-trees, and should be avoided in order to improve the system efficiency).

[5] https://github.com/carlosmccosta/dynamic_robot_localization/blob/hydro-devel/yaml/schema/drl_configs.yaml
[6] http://yaml.org/
[7] http://wiki.ros.org/Parameter%20Server
[8] http://docs.ros.org/api/nav_msgs/html/msg/OccupancyGrid.html
[9] http://docs.ros.org/api/sensor_msgs/html/msg/PointCloud2.html

4.4 Point cloud assembly

The self-localization system can use any sensor that provides point clouds, namely RGB-D / ToF cameras, LIDARs and stereo vision systems. Each of these types of sensors have very different operation rates and measurement accuracy. As such, the localization system allows the assembly of several ambient scans using a circular buffer in order to reduce the impact of sensor noise.

For LIDARs, the system provides a *sensor_msgs::LaserScan*[10] assembler[11] that converts laser measurements in polar coordinates into Cartesian coordinates and projects the points using spherical interpolation (in order to account for laser scan deformation that occurs when the robot is moving and rotating). It can merge scans from several lasers sensors and it will publish the final point cloud after assembling a given number of scans or periodically after a specified duration. These assembly configurations can be changed at runtime based on the robot velocity (for example, when the robots moves slower, more laser scans are assembled for each published point cloud) or through the use of the ROS dynamic reconfigure API, which allows a navigation supervisor to control the rate at which the localization system operates.

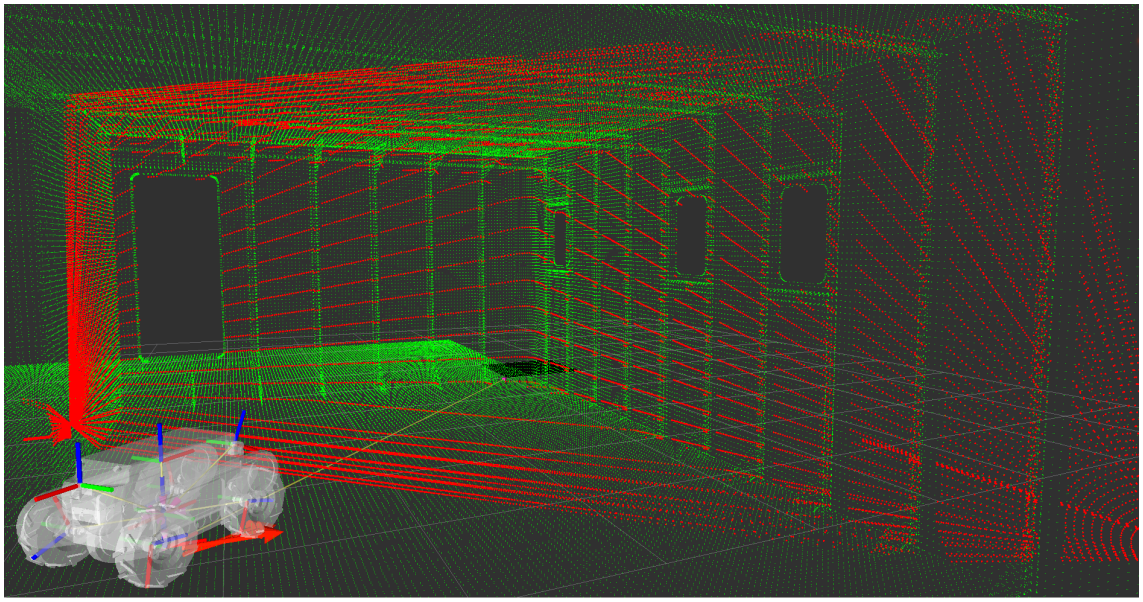


Figure 4.4.0.3: Laser scans assembled with tilting platform (red points)

4.5 Point cloud search data structures

Most of the point cloud algorithms use neighbor searches to analyze the surroundings of a given point. As such, efficient data structures are needed to speedup these operations, in order to execute the algorithms efficiently.

[10] http://docs.ros.org/api/sensor_msgs/html/msg/LaserScan.html
[11] https://github.com/carlosmccosta/laserscan_to_pointcloud

4.5.1 Voxel grids

A voxel grid is a three dimensional space partition data structure that splits the Euclidean space into regular voxels (volume pixel). It can be built very fast but is not very efficient for sparse point clouds. Figure 4.5.1.1 shows voxel grids with different voxel size.

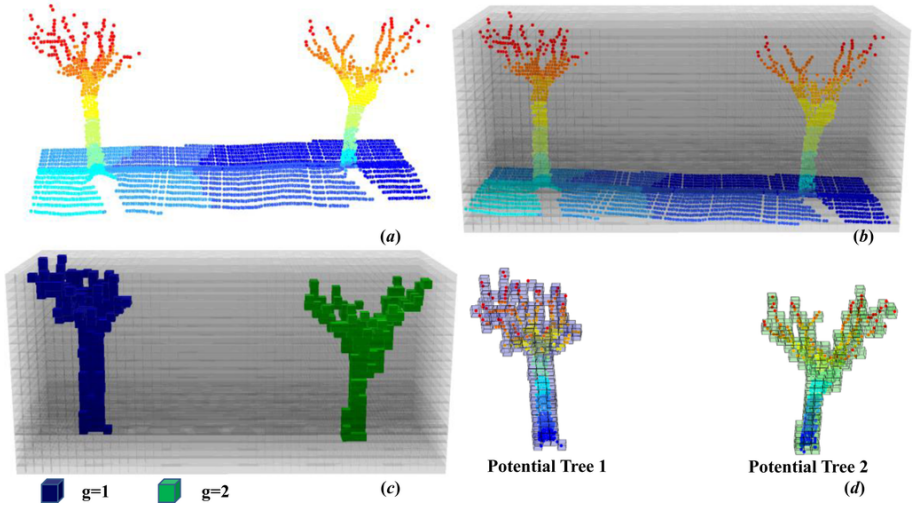


Figure 4.5.1.1: Voxel grid applied over trees point clouds [WYY+13]

4.5.2 Octrees

An octrees is a hierarchical space partition technique that adapts its tree data structure to the distribution of points in the cloud. It accomplishes this by recursively dividing each voxel in 8 octants until the tree depth is reached or when there is no more points in that region of space. This can be seen in figure 4.5.2.1 in which areas with no points have large voxels while areas with high point density have much smaller voxels.

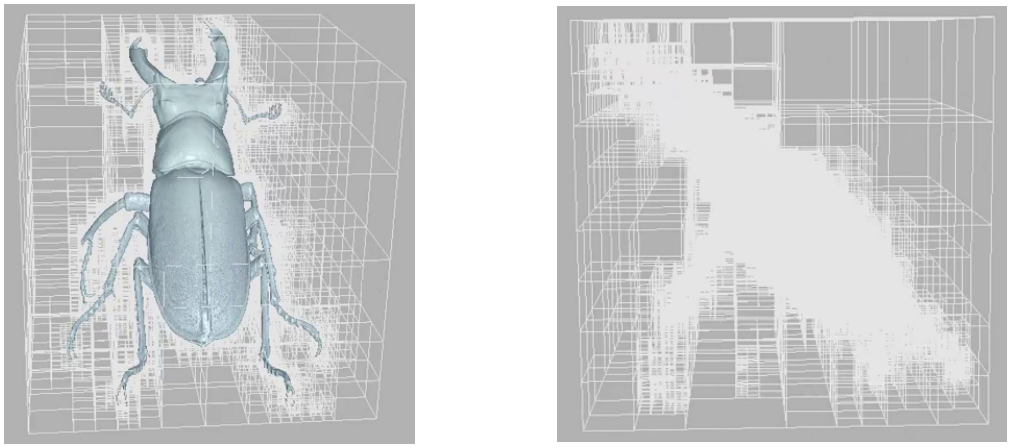


Figure 4.5.2.1: Octree of a stag beetle[12]

[12] http://blog.mpanknin.de/?p=753

4.5.3 k-d trees

A k dimensional tree is a space partition technique that can organize points with k dimensions. It is a generic data structure that can be used for 2D and 3D points (examples in figure 4.5.3.1 and figure 4.5.3.2) as well as any other types of data that have an arbitrary number of dimensions (such as point cloud feature descriptors).

The binary k-d tree is built by successively selecting the median point in each axis until all points are inserted in the tree (the selection of the next axis is performed in a circular way, which in the case of three dimensional data, means that after processing the z axis, the x axis would be selected).

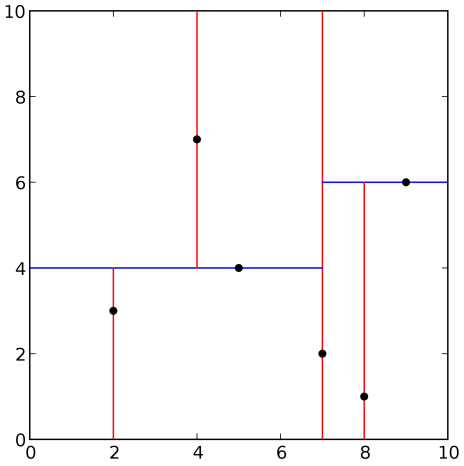


Figure 4.5.3.1: 2-d tree[13]

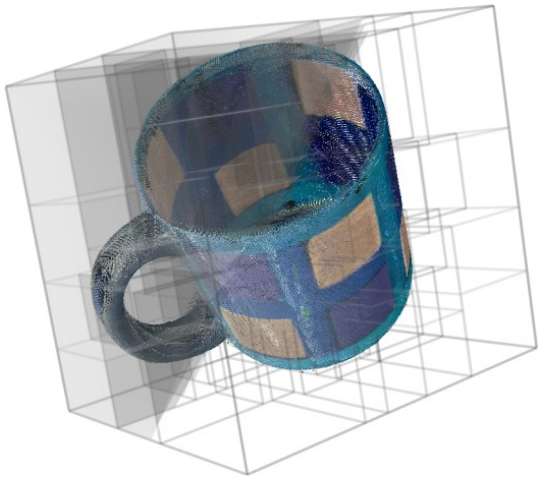


Figure 4.5.3.2: 3-d tree[14]

4.6 Filtering and down sampling

The time it takes to perform cloud registration is proportional to the amount of points in the ambient point cloud and in the reference map. As such, adjusting the level of detail of the point clouds by using voxel grids gives some control over the desired localization accuracy and the computational resources that will be required. This stage is also useful to mitigate the measurement errors of the depth sensors, since the centroid of a voxel that contains points from several scans will be closer to the real surface (if the voxels have dimensions slightly larger than the expected measurement errors).

The localization system allows the application of several preprocessing algorithms (algorithm type / configuration and execution order customizable). The next sections detail some of these methods that can be used to filter and downsample point clouds.

[13] http://en.wikipedia.org/wiki/K-d_tree

[14] http://docs.pointclouds.org/trunk/group__kdtree.html

4.6.1 Point cloud downsampling

Downsampling methods aim to reduce the number of points while maintaining the surface structure of a given point cloud. They can be used to adjust the level of detail according to the point cloud registration precision required.

4.6.1.1 Voxel grid sampling

A voxel grid is a uniform space partition technique that can be used to cluster points according to their Euclidean coordinates. As can be seen in figure 4.6.1.1, it is a very effective method to control the level of detail of a point cloud because it gives the ability to specify the maximum number of points that a region of space should have.

The point cloud downsampling is achieved by replacing each voxel cluster with a single point. The selection of this point can be very fast if the voxel center is used, but computing the centroid of the cluster yields better results because it represents the underlying surface with more accuracy and it attenuates errors in the sensors measurements.

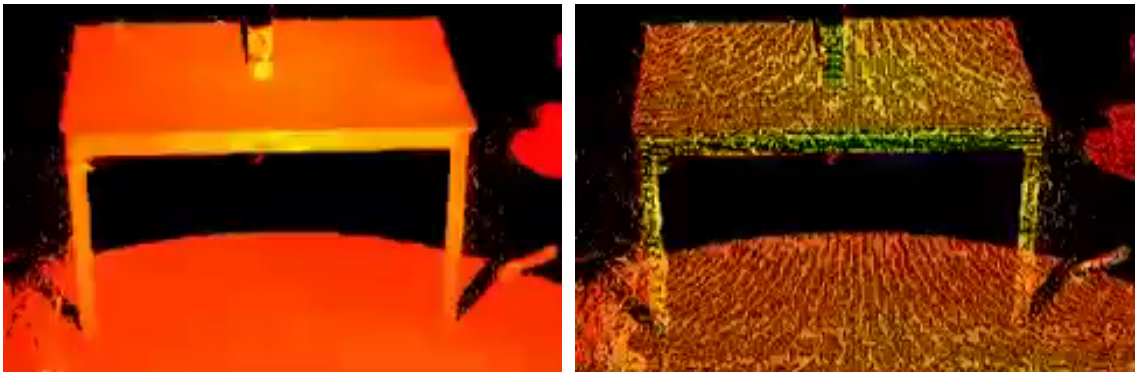


Figure 4.6.1.1: Table point cloud before (left) and after (right) voxel grid downsampling[15]

4.6.1.2 Random sampling

Random sampling [Vit84] is a fast downsampling method that randomly selects points from the input cloud until the specified number of samples is reached. This has the advantage of using real measures instead of downsampled approximations, but also means it is more sensible to sensor measurements noise. However, for outdoor environments or very complex scenes, using the real measurements can be preferable than using cluster centroids because the voxels may not have the necessary resolution or may have a prohibitive computational cost.

4.6.1.3 Covariance sampling

Covariance sampling [GIRL03] is a subsampling method that aims to create a stable downsampled point cloud to be registered with ICP algorithms. It incrementally builds the downsampled point

[15] http://pointclouds.org/documentation/tutorials/voxel_grid.php

cloud while trying to keep the 6 eigenvalues of the covariance matrix as close to each other as possible.

The resultant point cloud has the desired number of points and is stable enough to be matched with ICP point to plane algorithms.

4.6.2 Outlier removal

Some depth sensors can perform measurements with high accuracy, but they have some limitations that can lead to the creation of outliers [Sot06].

One of those limitations can produce shadow points around objects boundaries. This is due to the fact that a portion of the laser beam may hit the object boundary and other part may hit other areas in the object background. And given that most laser range finders use a weighted sum of several beams, this can yield measurements that are not associated with any real object (outliers). Figure 4.6.1.1 shows a considerable amount of these shadow points close to the front table legs. Another issue is related to the angle in which the depth sensor sees the objects areas. If the incidence angle is very low, then it may be difficult to compute its distance and detect if the beam had ambient reflections. This can significant increase the measurements noise or even lead to the creation of outliers. Other common problem is associated with the material properties of the surfaces. For example, objects with very high or very low reflectance, such as metals or glass, can increase the measurements noise. Moreover depending on the combination of surface geometry, material and incidence angle, some objects may even be undetectable by depth sensors.

Given the negative effect that outliers have in surface normal computation and registration algorithms, they should be removed in a preprocessing stage. There are several approaches to perform outlier detection and removal [ZMH10], ranging from simple distance thresholds to more robust statistical analysis. The next sections present some of then that can be useful in a localization system.

4.6.2.1 Distance filter

Given that depth sensors have a maximum recommended distance for their measurements, it is wise to remove points that are close or beyond this limit. Moreover, it may be useful to remove points that are too close to the sensor, because they may belong to the robot itself and not the environment.

This can be achieved by applying a minimum and maximum threshold to the distances returned by the depth sensor.

4.6.2.2 Passthrough filter

A passthrough filter can select or remove points according to their properties.

For outlier removal, it can be used to select points that are within a given bounding box (useful when we already know what area of the environment we want to analyze) or remove points that don't have the appropriate intensity or color.

4.6.2.3 Radius outlier removal filter

The radius outlier removal filter deletes points that don't have a minimum number of neighbors within a specified radius distance. It can be useful when the point density is known and is very effective in removing isolated points.

4.6.2.4 Statistical outlier removal filter

The statistical outlier removal filter [Rus10] performs a global analysis of the distances between points and discards the ones that don't follow the global distance distribution.

It is a robust filter that adapts itself to the point cloud density and is very effective in removing shadow points. To do so, it computes the mean distance that each point has to a given number of neighbors and builds a global distance distribution (example in figure 4.6.2.1). Then, assuming that the distribution is Gaussian, it discards the points that have a distance higher than a given threshold (that is a percentage of the standard deviation of the distance distribution).

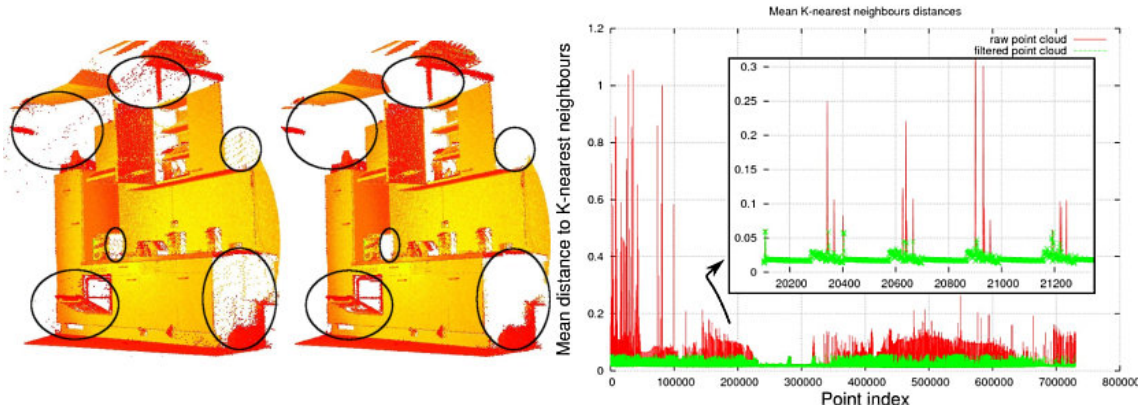


Figure 4.6.2.1: Statistical outlier removal filter[16]

4.6.3 Surface and object reconstruction and resampling

Depending on the level of sensor noise and amount of outliers present in a given point cloud, it may be necessary to employ surface reconstruction techniques to fill gaps in sensor data or correct measurements errors.

Moving least squares [ABCO+03] is a surface reconstruction algorithm that uses higher order bivariate polynomials to fit surfaces to a given set of points. It can be used to fill possible gaps in sensor data, smooth the point cloud (shown in figure 4.6.3.1), refine surface normals (shown in figure 4.6.3.2) and perform downsampling or upsampling.

Surface reconstruction can also be useful when the point cloud is built from several ambient scans with different origins and registered with some alignment errors. In figure 4.6.3.2 can be seen that the normal estimation is not very accurate in the overlapping regions between different

[16] http://pointclouds.org/documentation/tutorials/statistical_outlier.php

scans. This can be solved by estimating the normals using the surfaces computed by the moving least squares algorithm instead of using the point's neighbors.

Figure 4.6.3.1: Surface smoothing using moving least squares algorithm[17]

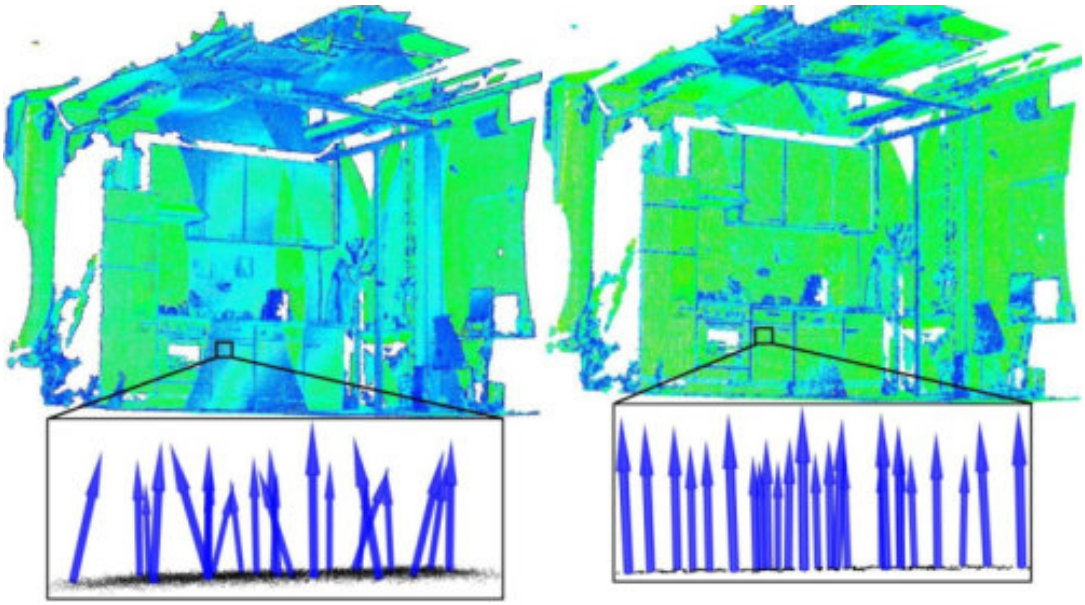


Figure 4.6.3.2: Surface normals refinement using moving least squares algorithm [Rus10]

4.7 Normal estimation

Most of feature detection, description and matching algorithms along with some registration methods rely on the point's surface normal and curvature. These algorithms analyze the neighborhood of a given point in order to compute the line / surface normal, and as such, the correct specification of what points should be included in the estimation is crucial to achieve accurate results. This depends on the environment geometry and the level of detail that is required, and is usually done by

[17] http://pointclouds.org/documentation/tutorials/resampling.php

specifying a radius distance (example in figure 4.7.2.1) or by limiting the number of neighboring points to use.

The next sections describe some of the methods that are supported for 3/6 DoF sensor data.

4.7.1 Line normal estimation

Line normals give information about the spatial disposition and orientation of a given cluster of collinear points. They can be computed using by fitting lines to sensor data using Random Sample Consensus (RANSAC) methods [FB81] and their orientation can be corrected using the sensor's origin.

4.7.2 Surface normal estimation

Surface normals provide information about the orientation of the underlying geometry of a given point cluster. They can be computed using plane fitting methods or using more advanced techniques such as Principal Component Analysis (PCA) [Jol02] or the one presented in section 4.6.3.

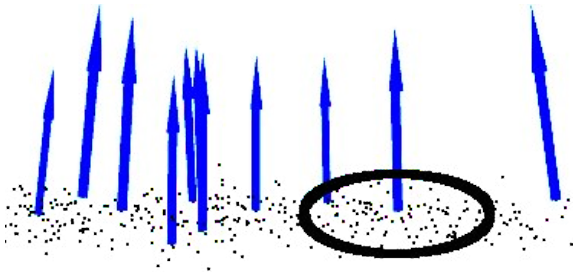


Figure 4.7.2.1: Point neighborhood for normal estimation [Rus10]

4.8 Keypoint detection

Aligning two point clouds with overlapping views of the environment requires the establishment of point correspondences. If both point clouds have similar sensor origins, these can be determined with nearest neighbor's searches and filtered with correspondence rejectors (using other point properties such as reflectance and color). But if they were acquired in two very different positions, then more advanced techniques must be employed.

One of those techniques uses histograms to describe the geometric properties of the environment around a given point. This allows points to be matched even if they have completely different Euclidean coordinates. Also, by using histograms and sampling techniques, these descriptors are much more robust against sensor noise and varying level of point density. However, these advantages come with a heavy computational cost, and as such, point descriptors should only be computed on the most descriptive areas of the environment.

Identifying these environment points is known as feature / keypoint detection [FA14], and usually involves finding interesting points, such as corners and edges. Besides uniqueness, these points must also be repeatable. This means that the detection algorithms should be able to find the

same points even if they are present in different point clouds with sensor noise and varying point density. This is of the utmost importance, because if the same keypoints are not identified on both clouds, then matching the point descriptors will likely fail.

Currently, the localization system can use the Scale Invariant Feature Transform (SIFT) [KL04] algorithm on the point's curvature or the 3D Intrinsic Shape Signatures (ISS3D) [Yu09] keypoint detector on the point's normals.

4.9 Keypoint description

Describing a keypoint usually involves analyzing its neighboring points and computing a given metric or histogram that quantifies the neighbor's relative distribution, their normals angular relation, associated geometry or other metrics that are deemed useful. Several approaches were suggested over the years according to different recognition needs and they are the basis of feature matching algorithms used in the initial pose estimation.

The localization system can use most of the keypoint descriptors available in PCL, namely the Point Feature Histogram (PFH) [RBMB08], the Fast Point Feature Histogram (FPFH) [RBB09], the Signature of Histograms of Orientations (SHOT) [TSD11], the Shape Context 3D (SC3D) [FHK+04], the Unique Shape Context (USC) [TSD10] and the Ensemble of Shape Functions (ESF) [WV11].

4.10 Cloud registration

Point cloud registration is the process of finding the transformation matrix (usually translation and rotation only) that when applied to a given ambient cloud will minimize an error metric (such as the mean square error of the ambient point cloud in relation to a given reference point cloud). Several approaches were suggested over the years and they can be categorized as point or feature cloud registration.

4.10.1 Initial alignment with keypoints descriptor matching

Feature registration is the process of matching keypoint descriptors in order to find an initial alignment between two point clouds. The proposed localization system uses a feature registration method similar to the Sample Consensus Initial Alignment (SAC-IA) algorithm presented in [RBB09]. It uses a RANSAC approach to select the best registration transformation after a given number of iterations. In each iteration a subsample of randomly selected descriptors from the ambient cloud is retrieved. Then for each of these descriptors, k best matching descriptors in the reference point cloud are searched (using a kd-tree) and one of them is chosen randomly (this improves robustness against noise in the sensor data and changes in the environment that are not yet integrated into the map). Later after having filtered these correspondences between ambient and reference descriptors, the registration matrix is computed. If this registration matrix results in

a point cloud overlap that has a minimum of inliers percentage (a point in the ambient cloud is an inlier if it has a point in the reference cloud closer than a given distance), then it is considered an acceptable initial pose and is saved (to allow a localization supervisor to analyze the distribution of the acceptable initial poses). In the end of all iterations, the best initial pose (if found) is refined with a point cloud registration algorithm.

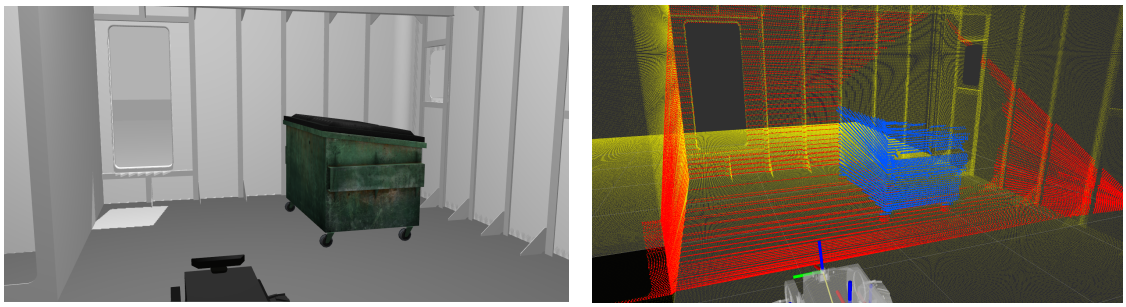
4.10.2 Final alignment with point cloud error minimization

Point cloud registration algorithms (introduced in section 2.2.4) such as ICP [BM92] (with its several known variations [RL01, Pom13] like ICP point-to-point, ICP point-to-point non-linear, ICP point-to-plane and generalized ICP [SHT09]) and the Normal Distributions Transform (NDT) [Mag09] are among the most popular algorithms to register point clouds. They can achieve very accurate cloud registration but they require an approximate initial pose for the registration to successfully converge to a correct solution (otherwise they may achieve only partial cloud overlap or even fail to converge to a valid solution).

4.11 Outlier detection

Detecting which points of the environment are not part of the reference point cloud can be very useful to evaluate the quality of point cloud registration as well as to analyze the presence of previously unknown objects. It's computation splits the ambient cloud into two point sets. One containing inliers (points correctly registered and present in the reference point cloud) and the other having the outliers (points that are either incorrectly registered or not present in the reference cloud).

A given ambient point can be classified as outlier if the corresponding closest point in the reference cloud is farther way than a given distance threshold. These calculation can be done efficiently by using the reference point cloud kd-tree. In figure 4.11.0.1 is an example of a ambient point cloud retrieved with a tilting laser and registered with an indoor map (yellow points). After registration, the ambient point cloud was split into inliers (red points) and outliers (blue points).



(a) Gazebo overview (b) RViz overview

Figure 4.11.0.1: Point cloud registration with outlier detection

4.12 Localization validation

After a point cloud is registered by the localization system, several metrics are calculated in order to evaluate if a valid pose can be retrieved using the registration matrix.

The first computed metrics are the percentage of inliers and the Root Mean Square Error (RMSE) of these inliers. If a minimum number of points was registered and the inlier percentage and root mean square error are acceptable, then the registration is considered successful. However, these registered points can be agglomerated in a small area and may not be representative of the robot location. As such, a second metric is computed that takes into account the angular distribution of these inliers. This metric gives a measurement of how reliable is this registration and is based on the fact that there is high confidence in a given pose estimation when there are correctly registered points all around the robot.

The last metrics are the corrections that the registration matrix introduced. Given that the localization system will be in tracking mode most of the time, it is possible to define how far a new pose can be in relation to the previous accepted location and discard new poses that exceed a given threshold. This is useful to discard pose corrections that are very unlikely to happen, such as the robot moving half a meter between poses when it is expected to move only at 30 cm/s. These situations can happen when there is a sudden decrease in the field of view (that can occur due to sensor occlusion or malfunction) or when large unknown objects very similar to sections of the map appear into the field of view of the robot.

If all these metrics are within acceptable thresholds, then the robot pose can be computed by applying the matrix correction to the initial pose associated with the ambient sensor data. If any of these metrics are not acceptable, then the system can be configured to simply discard this pose estimation and try to estimate the pose in the next sensor data update or it can apply a tracking recovery attempt with a different registration algorithm (or the same algorithm with different parameters). If several consecutive pose estimations are discarded, the system can have a second level of recovery that can be configured to use the initial pose estimation algorithms in order to finally estimate the robot pose and reset the tracking state.

4.13 Dynamic map update

After performing a successful pose estimation, the localization system can update the localization map by either integrating only the unknown objects or the full registered point cloud (it can also be used in conjunction with OctoMap [HWB+13] to perform probabilistic map updates). Integrating only the unknown objects is the recommended approach when there is a known map and the environment is expected to change gradually. This is also more efficient as only the points that need to be integrated are processed and ray traced in OctoMap. On the other hand, integrating the full registered cloud can be desirable if the map of the environment is very incomplete, very outdated or expected to change considerably during the operation of the robot.

Chapter 5

Localization system evaluation

This chapter presents several test scenarios that were devised to evaluate the implemented localization system. They aim to test the accuracy and robustness of the implementation under different environmental conditions and hardware configurations.

5.1 Testing platforms

The localization system was tested on laser sensor data retrieved from three different mobile robot platforms and executed on the same computer in order to allow a direct comparison of computation time. This computer was a Clevo P370EM3 laptop (with a Intel Core i7 3630QM CPU at 2.4GHz, 16 GB of RAM DDR3, NVidia GTX680M graphics card and a Samsung 840 Pro SSD) and it was running Ubuntu 12.04 along with ROS Hydro, PCL 1.7 and Gazebo 1.9.

The sensor data was recorded into rosbags, and is publicly available at[1] along with all the detailed results and experiments videos, in order to allow future comparisons with other localization systems. The hardware specifications of the LIDARs used is presented in table 5.1 and the laser points after downsampling and registration can be seen on the movement paths figures as green circles (inliers) and red circles (outliers).

5.1.1 Jarvis platform

The Jarvis platform is an autonomous ground vehicle equipped with a SICK NAV 350 laser for self-localization (mounted about 2 meters from the floor) and a SICK S3000 laser for collision avoidance (mounted about 0.20 meters from the floor). It uses a tricycle locomotion system with two back wheels and a steerable wheel at the front. In figure 5.1.1.1 the robot is performing a delivery task with the package on top of a moving support. The 3 DoF ground truth was provided by the SICK NAV350 system and relied on 6 laser reflectors (with 9 cm of diameter) to perform the pose estimations (it is certified for robot docking operations with precision up to 4 millimeters).

[1] https://github.com/carlosmccosta/dynamic_robot_localization_tests

Figure 5.1.1.1: Jarvis testing platform

5.1.2 Pioneer 3-DX platform

The Pioneer 3-DX shown in figure 5.1.2.1 is a small lightweight robot equipped with a SICK LMS-200 laser (mounted about 48 cm from the floor) and a Kinect (mounted about 78 cm from the floor). It uses a two-wheel two-motor differential drive locomotion system and can reach a linear speed of 1.2 m/s and angular velocity of 300°/s. The 3 DoF ground truth was provided by 8 Raptor-E cameras [2] and according to [SEE+12] it had less than 1 cm in translation error and less than 0.5 degrees in rotation error.

Figure 5.1.2.1: Pioneer 3-DX testing platform [SEE+12]

[2] http://www.motionanalysis.com/html/movement/raptore.html

5.1.3 Guardian platform

The Guardian platform is an autonomous mobile manipulator equipped with a Hokuyo URG-04LX laser in the front and a Hokuyo URG-04LX_UG01 laser in the back (both mounted about 0.37 meters from the ground). The front laser had a tilting platform which allows 3D mapping of the environment. The arm is a SCHUNK Powerball LWA 4P and in figure 5.1.3.1 it is attached to a stud welding machine (in simulation it is attached to a video projector). It uses a differential drive locomotion system and can be moved with wheels or with tracks. This platform didn't have a certified ground truth and as such, the results could not be quantified with an external localization system (the results performed with the Gazebo simulator will be presented instead - robot model shown in figure 5.1.3.2).

Figure 5.1.3.1: Guardian testing platform

Figure 5.1.3.2: Guardian Gazebo simulation

Table 5.1: LIDARs hardware specifications

Laser model	*Range (meters)*	*Field of view (degrees)*	*Scanning frequency (Hz)*	*Angular resolution (degrees)*	*Statistical error (millimeters)*
SICK NAV350	[0.5..250]	360	8	0.25	15
SICK S3000	[0.1..49]	190	8	0.25	150
SICK LMS200	[0.1..80]	180	10	1.0	35
Hokuyo URG-04LX	[0.06..4.095]	240	10	0.36	10
Hokuyo URG-04LX_UG01	[0.02..4]	240	10	0.36	30

5.2 Testing environments

The localization system was tested in different environments and used the Jarvis platform in a large room with a RoboCup field, the Pioneer 3-DX in a large industrial hall, the Guardian platform in a simulated indoor environment and a Kinect in a flying arena.

5.2.1 Jarvis in RoboCup field

The RoboCup field (shown in figure 5.2.1.1) occupies half of a large room (with 20.5 meters of length and 7.7 meters of depth). It has two doors, several small windows and two large glass openings into the hallway. Several tests were performed with the robot at speeds ranging from 5 cm/s to 50 cm/s in this environment and up to 2 m/s using the Stage simulator[3]. These tests were performed with two different movement paths. The first is a simple rounded path (shown in appendices A.1.1 and A.1.2) that aimed to test the robot in the region of space that had better ground truth (due to its position in relation to the laser reflectors). The second path was more complex and contained several sub paths with different velocities and shapes (shown from appendices A.1.3 to A.1.6). It was intended to evaluate the localization system with typical movements that mobile manipulators require, such as moving forward and backwards with or without angular velocity and stopping at the desired destination.

Figure 5.2.1.1: Jarvis testing environment

5.2.2 Guardian in structured environment

The structured environment simulated in Gazebo is a large room with 12.4 meters of length and 8.4 meters of depth. It has 4 doors, several small windows and the walls have small ledges at regular intervals (as can be seen from figures 5.2.2.1 to 5.2.2.3).

Given that the Guardian mobile manipulator is expected to work on the walls of this environment, several tests were devised with a path following the lower and right wall of the environment (shown from appendices A.3.1 to A.3.3). The first test was done in a static environment clear of unknown objects and was meant to evaluate the best precision that the localization system could achieve. The second test was done in a cluttered environment and was designed to test the robustness of the localization system against static unknown objects, that were placed in the middle of the environment and close to the walls (to block sensor data from reaching known positions and analyze the robustness of matching unknown points that are close to known areas). The last test added a moving car to the scene and aimed to assess the impact of dynamic objects on the point cloud registration algorithms.

[3] http://rtv.github.io/Stage/

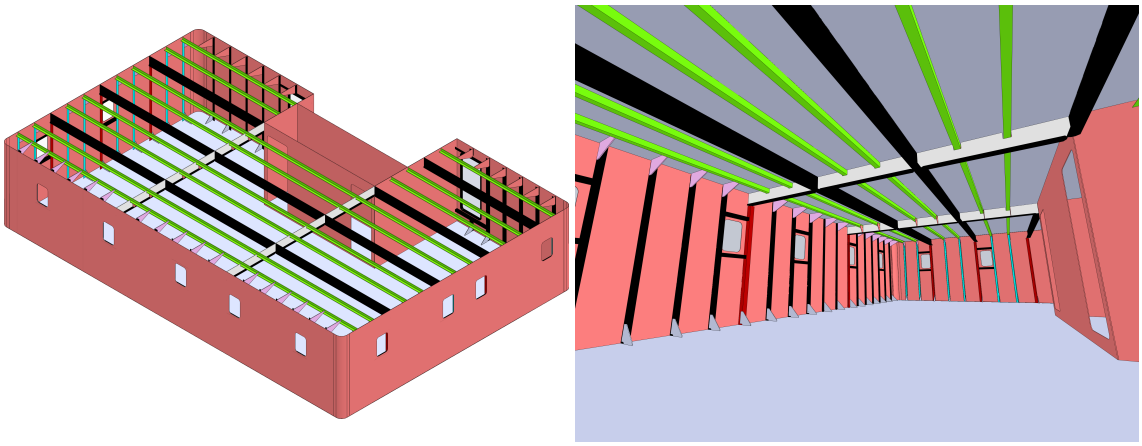


Figure 5.2.2.1: Guardian testing environment

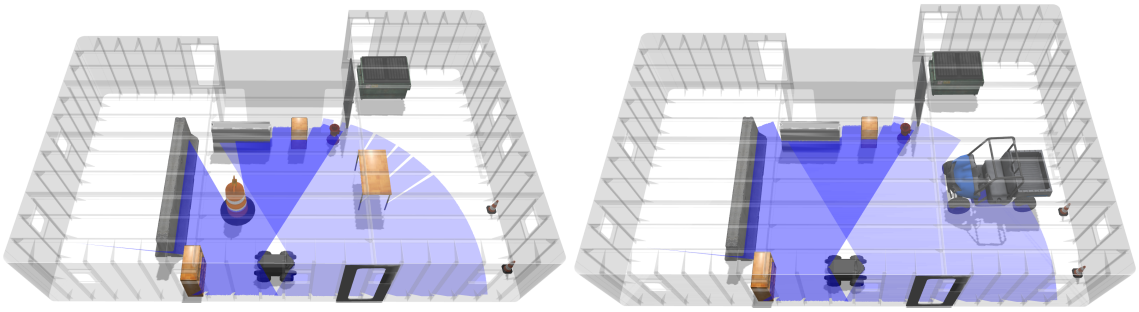


Figure 5.2.2.2: Guardian cluttered (left) and dynamic (right) testing environment

Figure 5.2.2.3: Interactive CAD of Guardian testing environment (.u3d file)

5.2.3 Pioneer in industrial hall

The industrial hall is a large room with several tables and objects spread around. Four tests were performed with the Pioneer in this environment. The first was a 360º path with few objects in the middle of the room, while the remaining 3 were done with a lot of large objects, that significantly reduced the field of view of the robot laser (as can be seen in figure 5.2.3.1).

Figure 5.2.3.1: Industrial hall with and without objects in the center [SEE+12]

5.2.4 Kinect in flying arena

The flying arena shown in figure 5.2.4.1 is a large room in which several objects were added to test 6 DoF pose tracking. In these tests the Kinect was moved by the operator in three different paths. The first was a smooth fly movement over the testing scene, while the other two aimed to test paths with mainly translations and rotations. This environment had a ground truth provided by Vicon cameras and according to the authors of the dataset [PMC+11] it had sub-centimeter accuracy [4].

Figure 5.2.4.1: Flying arena environment [PMC+11]

[4] http://www.vicon.com/

5.3 3 DoF localization system tests

5.3.1 Overview

The main results of the 3 DoF tests performed with the localization system are presented in tables 5.2 and 5.3. They summarizes each test by fitting a normal distribution for each evaluation metric. They were retrieved with a known initial pose and used ICP point-to-point as tracking algorithm and ICP point-to-plane as tracking recovery method. The Jarvis and Pioneer tests had a map built using the localization system in SLAM mode and were manually corrected to achieve a resolution of 10 and 25 mm respectively. The Guardian tests relied on a map built from the CAD model with resolution of 2 mm. A voxel grid of 50 mm was applied to the assembled laser scans in order to reduce the impact of sensor measurement noise and also control the level of detail of the point clouds. The initial pose estimation subsystem used SIFT for keypoint selection, FPFH for keypoint description and the feature matching algorithm described in section 4.10.1 to estimate the initial position and orientation of the robot (figure 5.3.5.3 shows the accepted initial poses when the robot was in the lower right corner of the Guardian test environment).

The next sections will provide an overview analysis of the main 3 DoF results achieved with the proposed localization system (available in appendix A). It will start by explaining how laser spherical interpolation can mitigate point cloud deformation. Later on it will analyze the point cloud preprocessing stage and why it should be carefully tuned to the sensors and ambient geometry. Next it will present the SLAM problem and why it is necessary to have accurate maps in order to achieve precise localization. Finally it will be presented a global analysis of the tests, in which the results achieved by the proposed localization system will be compared with the ground truth, odometry and Adaptive Monte Carlo Localization (AMCL).

Table 5.2: 3 DoF localization system results - (*) Most relevant experiments

Test conditions							Translation error (millimeters)		Rotation error (degrees)		Outliers percentage [0..100]		Global computation time (milliseconds)	
Section	Platform	Ambient	Path shape	Path velocities	Map cell resolution	Nº scans / Nº lasers	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation
A.1.1	Jarvis robot	Cluttered	Rounded	5 cm/s	10 mm	1-4/1	3.384	1.900	0.549	0.087	15.75	3.84	12.052	12.232
A.1.2				30 cm/s	10 mm	1-4/1	12.250	8.901	0.535	0.414	16.08	3.80	21.295	8.352
A.1.3		Cluttered dynamic	Complex	5 cm/s	10 mm	1-4/1	4.280	2.264	0.441	0.070	24.35	5.17	11.776	12.265
A.1.4 (*)				50-30-50-10 cm/s	10 mm	1-4/1	6.422	3.992	0.397	0.099	23.84	5.59	13.376	13.316
A.2.1	Pioneer robot	Cluttered dynamic	360º	23 cm/s	25 mm	1/1	20.105	10.814	5.704	0.662	11.71	3.37	5.046	2.106
A.2.2			SLAM 1	26 cm/s	25 mm	1/1	22.434	12.052	5.429	0.676	4.31	3.84	4.569	2.530
A.2.3			SLAM 2	19 cm/s	25 mm	1/1	19.391	12.568	5.518	0.818	5.09	4.11	4.650	2.180
A.2.4 (*)			SLAM 3	16 cm/s	25 mm	1/1	17.480	8.895	5.455	0.732	4.15	4.26	4.388	2.526
	Jarvis simulator (Stage)	Cluttered	Rounded	5 cm/s	10 mm	2/1	2.817	1.551	0.038	0.026	16.11	0.93	12.600	12.459
				50 cm/s	10 mm	2/1	4.479	3.769	0.106	0.137	15.82	1.14	24.500	13.592
				100 cm/s	10 mm	1/1	4.710	2.788	0.135	0.134	16.13	0.92	16.432	12.291
			Complex	5 cm/s	10 mm	2/1	2.476	1.370	0.017	0.017	17.00	0.92	10.374	12.138
				50-30-50-20 cm/s	10 mm	2/1	3.557	1.800	0.033	0.040	17.61	1.33	14.863	15.898
				200-30-200-30 cm/s	10 mm	2/1	4.199	2.596	0.048	0.110	17.84	1.23	13.150	14.351
A.3.1	Guardian simulator (Gazebo)	Static	Wall follower	5 cm/s	2 mm	2/2	2.642	0.463	0.016	0.023	0.0	0.0	5.723	1.868
				30 cm/s	2 mm	2-4/2	4.654	3.893	0.044	0.070	0.0	0.0	7.784	3.178
A.3.2		Cluttered		5 cm/s	2 mm	2/2	2.912	0.856	0.020	0.026	25.05	13.30	11.889	9.084
				30 cm/s	2 mm	2-4/2	6.092	3.834	0.096	0.085	25.65	11.02	23.925	27.959
A.3.3 (*)		Cluttered dynamic		5 cm/s	2 mm	2/2	4.593	3.802	0.069	0.072	33.33	9.88	22.217	32.408
				30 cm/s	2 mm	2/2	5.897	4.452	0.095	0.104	27.28	13.04	24.409	33.459

Table 5.3: 3 DoF odometry and AMCL results - (*) Most relevant experiments

Test conditions						Odometry translation error (millimeters)		Odometry rotation error (degrees)		AMCL translation error (millimeters)		AMCL rotation error (degrees)	
Section	Platform	Ambient	Path shape	Path velocities	Map cell resolution	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation
A.1.1	Jarvis robot	Cluttered	Rounded	5 cm/s	10 mm	25.353	10.075	0.352	0.206	51.417	20.829	0.442	0.174
A.1.2				30 cm/s	10 mm	103.676	50.992	2.173	1.916	81.368	36.548	1.388	1.085
A.1.3		Cluttered dynamic	Complex	5 cm/s	10 mm	21.506	11.937	0.535	0.360	64.300	13.750	0.316	0.214
A.1.4 (*)				50-30-50-10 cm/s	10 mm	124.269	79.775	2.723	1.496	84.230	29.134	0.595	0.581
A.2.1	Pioneer robot	Cluttered dynamic	360^o	23 cm/s	25 mm	482.439	163.657	12.810	4.468	135.442	66.445	6.462	1.290
A.2.2			SLAM 1	26 cm/s	25 mm	370.283	163.734	8.351	2.825	111.470	46.783	6.496	1.386
A.2.3			SLAM 2	19 cm/s	25 mm	686.983	361.666	11.333	5.423	95.537	47.560	6.957	2.294
A.2.4 (*)			SLAM 3	16 cm/s	25 mm	295.714	151.793	8.858	3.331	108.690	52.341	6.463	1.363

5.3.2 Laser assembly with spherical linear interpolation

The localization system was designed to operate with point cloud sensors (such as the ones presented in section 3.4). For the particular case of data retrieved using LIDARs, there is a very significant problem of point cloud deformation when the robot is moving or rotating at high speeds. This is due to the fact that a typical LIDAR outputs laser scans at a very low rate (8-10 Hz), and as such, assuming that the robot isn't moving when capturing an entire scan will result in deformed point clouds. Given that ROS outputs laser scans (an entire slice of laser measurements from the start to the end of its field of view), and the localization system doesn't have access to individual measurements as they arrive, then one way to mitigate this deformation is by using odometry and / or Inertial Measurement Unit (IMU) information to update the robot pose between the pose corrections performed by the localization system. This allows to use spherical linear interpolation when converting from the raw measurements in polar coordinates into the required Cartesian coordinates in the map frame.

As can be seen in appendix A.1.5, laser scan deformation is negligible when the robot is moving very slowly. However, when traveling at higher speeds, the point cloud deformation poses a serious issue, and the usage of spherical linear interpolation is very important to mitigate it (demonstrated in appendix A.1.6).

In figure 5.3.2.1 can be seen that the laser measurements in front of the robot were deformed outwards, causing the points to be projected outside both walls. Figure 5.3.2.3 shows the same effect, which is now very noticeable since the first laser measurement isn't close to the last one. These deformations were severely diminished when the laser spherical linear interpolation was used (as can be seen in figures 5.3.2.2, 5.3.2.4, A.1.6.3 and A.1.6.4 respectively.

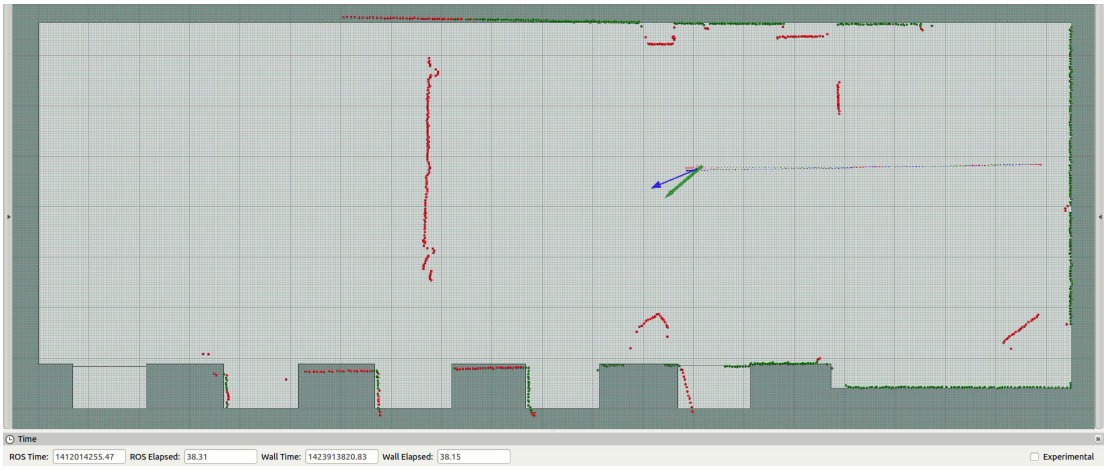


Figure 5.3.2.1: Large laser deformation on opposite walls when the robot is rotating

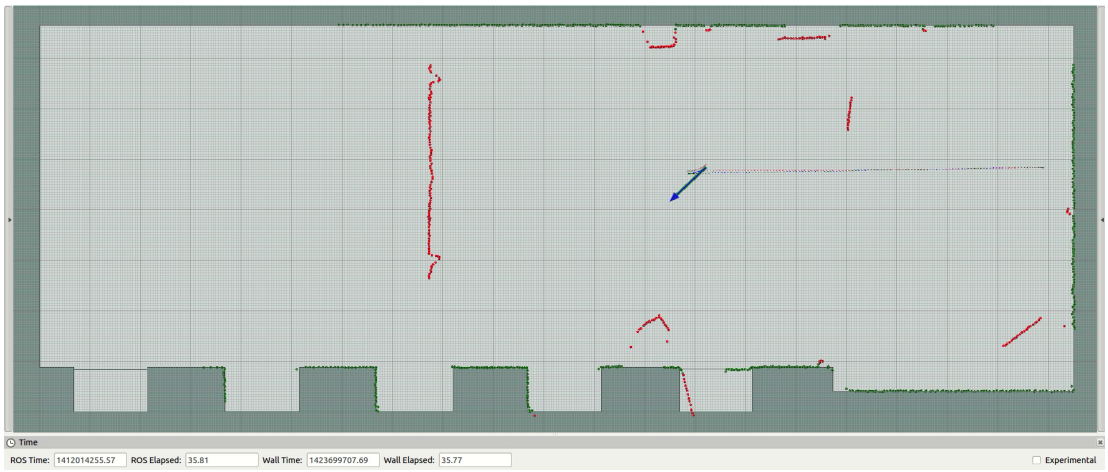


Figure 5.3.2.2: Correction of the large laser deformation on opposite walls with spherical linear interpolation

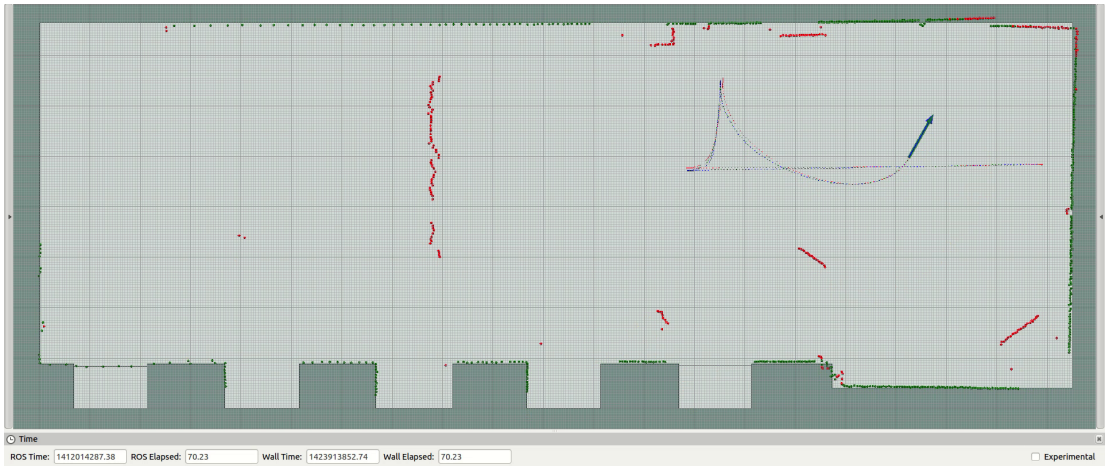


Figure 5.3.2.3: Laser deformation when the robot is rotating

Figure 5.3.2.4: Correction of the laser deformation with spherical linear interpolation

5.3.3 Point cloud preprocessing

The statistical error of the lasers can be mitigated by merging several laser scans before performing a point cloud registration. This is based on the fact that a considerable amount of laser noise follows a Gaussian distribution, and as such, the effective measurement error can be significantly reduced by retrieving the centroids of a voxel grid applied over the laser data. However, assembling too much laser scans can lead to worse pose tracking if the robot is moving at high speeds, because the lasers would be projected into the map frame with a high position error (because odometry accuracy decreases when the robot speed increases and the localization system will not correct it). As such, the implemented laser assembler supports dynamic reconfiguration of the number of laser scans to assemble (or the period of assembly time) based on the robot estimated velocity. This feature besides improving the pose tracking, it also allows a navigation supervisor to regulate the rate at which the localization system operates. This can be very useful for mobile manipulators because the localization system can have a high update rate when the robot is moving fast and a low update rate when it is moving slower or when it is stopped. This allows a more efficient usage of the platform hardware resources while also improving the localization system accuracy.

Besides reducing the sensor measurement errors, this processing stage also allows the removal of laser shadow points caused by veiling, or small cluster of points that may be associated with temporary objects / people. Moreover, given that the time it takes to perform point cloud registration is proportional to the number of points in the reference and ambient point cloud, voxel / random downsampling can be a very effective technique to adjust the level of detail required (given the computational resources available).

5.3.4 SLAM

Any self-localization system requires a map to estimate the pose of the robot when using exteroceptive information. If such map doesn't exist, then the first point cloud can be considered as the initial map, and then it can be updated dynamically as new sensor data is processed. This map update approach can use the full registered point cloud or the inliers / outliers only. Full integration is useful when starting a map from scratch (example in figure 5.3.4.1). Partial integration can be useful when there is a highly detailed map and we only want to add or remove information from it (examples from figures 5.3.4.6 to 5.3.4.9). Partial integration besides reducing the computational resources required it also allows to keep the detail of the source map, avoiding deformations due to sensor measurement noise. This noise is noticeable in the map in figure 5.3.4.7 (updated with full integration) and was avoided in the map in figure 5.3.4.9 (updated with outliers only).

Looking at figures from figures 5.3.4.1 to 5.3.4.5 it can be seen that the proposed localization system was able to achieve equal or even better mapping results than the GMapping[5] SLAM system and even the ground truth provided by the Raptor-E cameras (that seems to be less suitable for mapping than both the proposed localization system and the GMapping ROS package).

[5] http://wiki.ros.org/gmapping

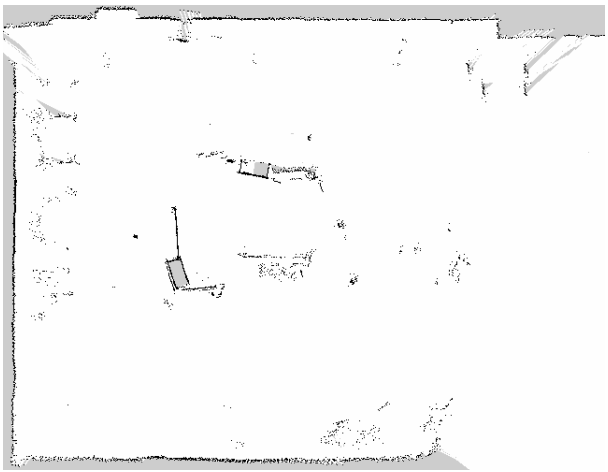


Figure 5.3.4.1: Map made with the localization system using full integration in conjunction with OctoMap

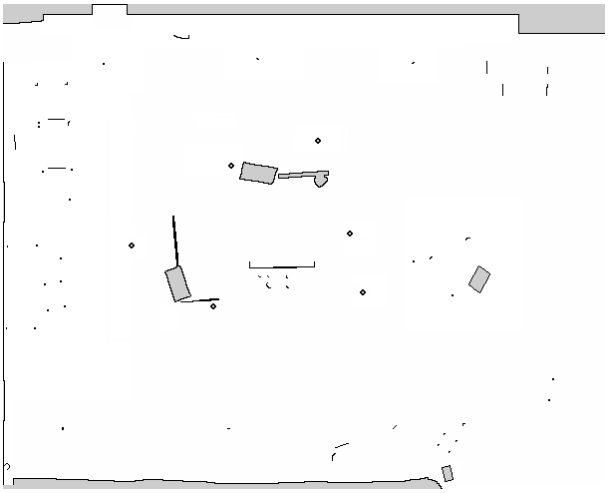


Figure 5.3.4.2: Map manually corrected using the previous figure as starting point

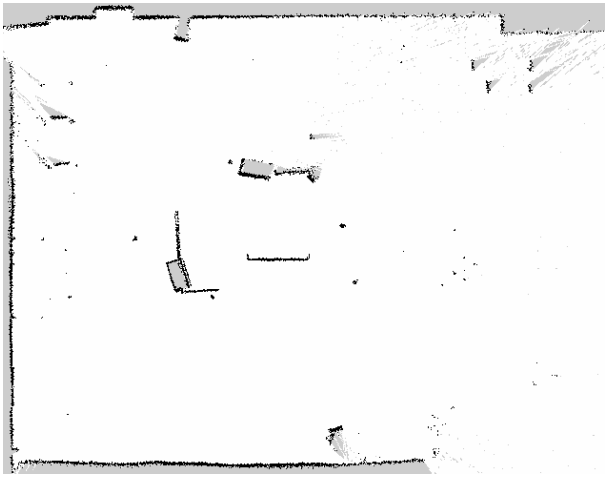


Figure 5.3.4.3: Map made with the GMapping package playing the rosbag at 10% speed

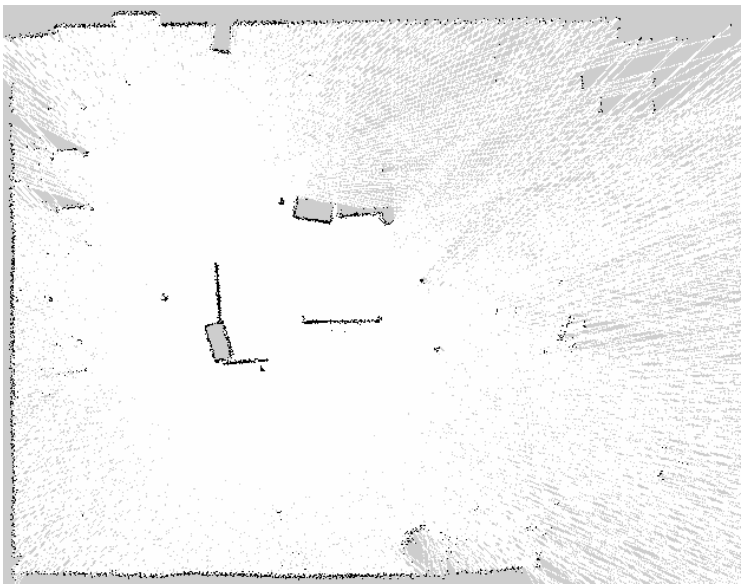


Figure 5.3.4.4: Map made with the GMapping package playing the rosbag in real time

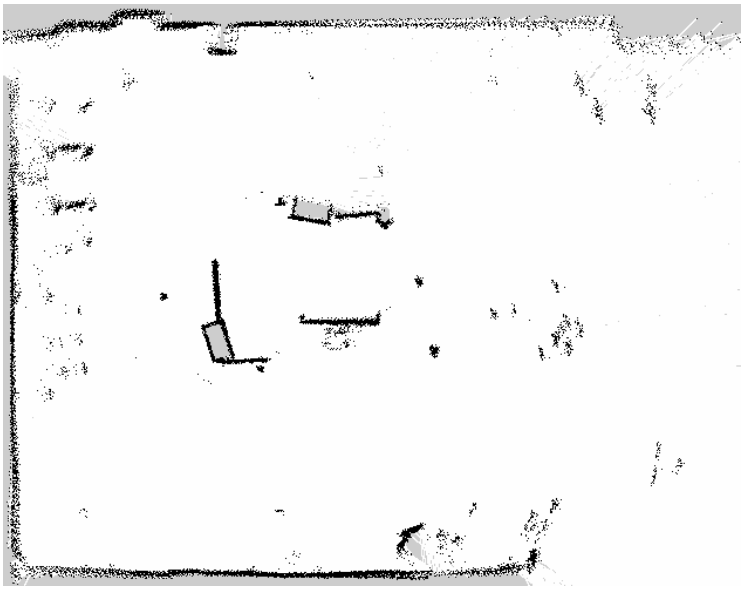


Figure 5.3.4.5: Map made using the ground truth poses provided by the Raptor-E cameras

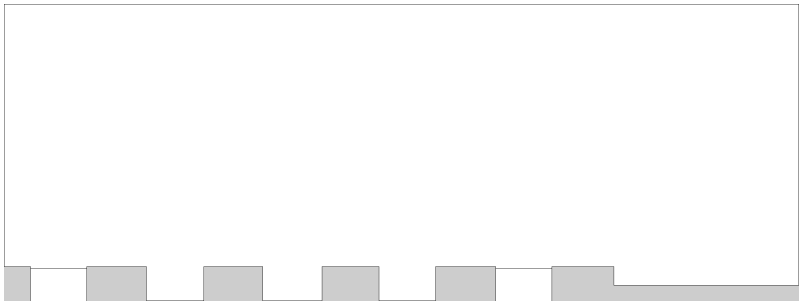


Figure 5.3.4.6: Original map from room i-108

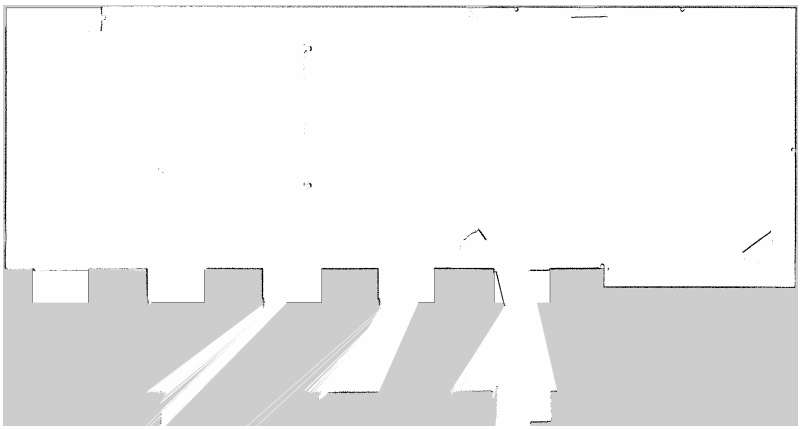


Figure 5.3.4.7: Updated map of room i-108 using full integration

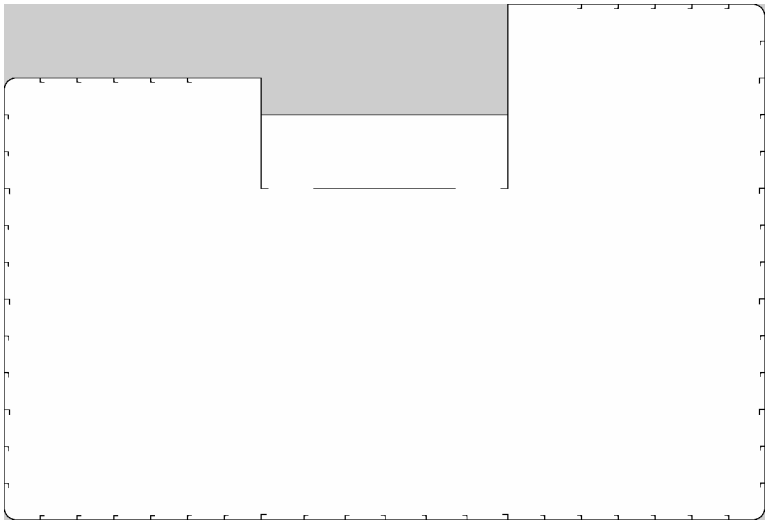


Figure 5.3.4.8: Original map of structured environment

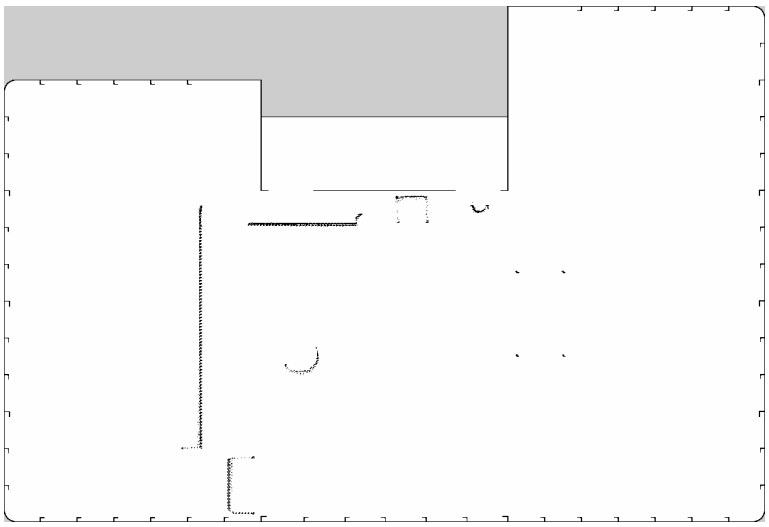


Figure 5.3.4.9: Updated map of structured environment using partial integration (outliers)

5.3.5 Point cloud registration

Looking at the results present in appendix A, such as the laser assembly figures and translation / rotation error graphs from the Jarvis test in the complex path at high velocities (shown from figures 5.3.5.1 to 5.3.6.2), it can be seen that the localization system can registered point clouds with much more accuracy than AMCL (even when using over 5000 particles). The only problems that seem to affect the used algorithms are the point cloud deformation (usually when there is unreliable odometry information, which typically occurs when the robot is moving with high velocities / accelerations), laser measurements errors, unknown objects close to the known reference point cloud and also low resolution maps.

The localization system can tolerate these problems by having a default pipeline configuration for the normal operation of the robot, another for temporary tracking recovery and yet another for initial pose estimation.

The switch between the localization system operation modes using the point cloud registration analysis proved to be a very intuitive and fast process and the overall system configurations seemed to be generic enough to be reused in several types of environments, with different robots equipped with varying types of sensors. Such ease of configuration allows the fast deployment of robots and gives a high confidence that the localization system will remain accurate even in challenging environments. This is not the case with systems such as AMCL, given that they are very dependent on the odometry and laser models, and as such, require constant retuning when one of these models change.

The ability to switch registration algorithms at runtime based on the quality of the estimated pose proved to be an efficient and flexible architecture choice, since it allowed high precision pose tracking with fast registration algorithms and occasional pose tracking recovery with more robust methods (which happened more often in the tests at higher velocities, in which the odometry was significantly worse).

The initial pose estimation using feature matching was very reliable and successfully found the robot location even in low feature surroundings (as can be seen in figures 5.3.5.3 and 5.3.5.4). Moreover, the output of the accepted initial pose estimations allows the detection of similar map locations by a navigation supervisor, which can then plot a path to disambiguate the initial pose guess and avoid dangerous operations at a wrong position.

Figures 5.3.5.3 to 5.3.5.4 show the accepted initial pose estimations using the global localization subsystem presented in section 4.10.1.

The blue circles presented in figure 5.3.5.4 represent the keypoints of the ambient point cloud in the robot start up position, while the violet circles are the reference point cloud keypoints. The green circles are the ambient point cloud laser measurements after performing the initial pose estimation and registration refinement.

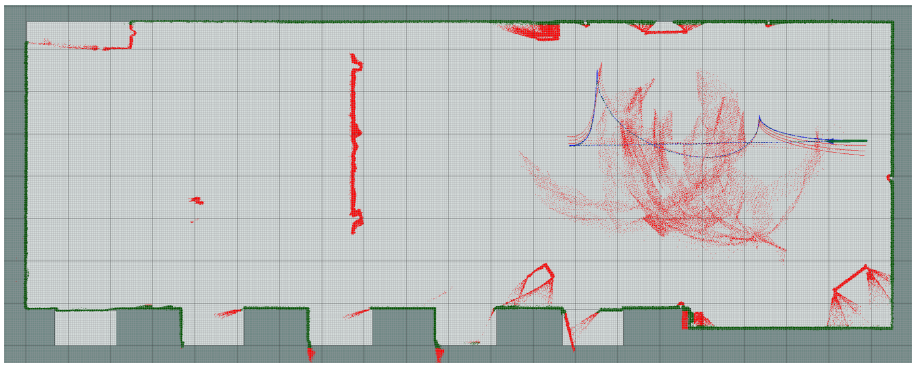


Figure 5.3.5.1: Laser scans assembled on top of the map using the localization system poses

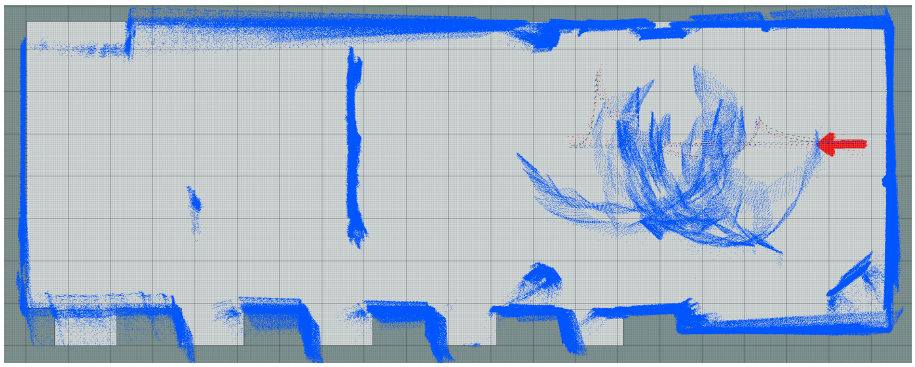


Figure 5.3.5.2: Laser scans assembled on top of the map using the AMCL poses

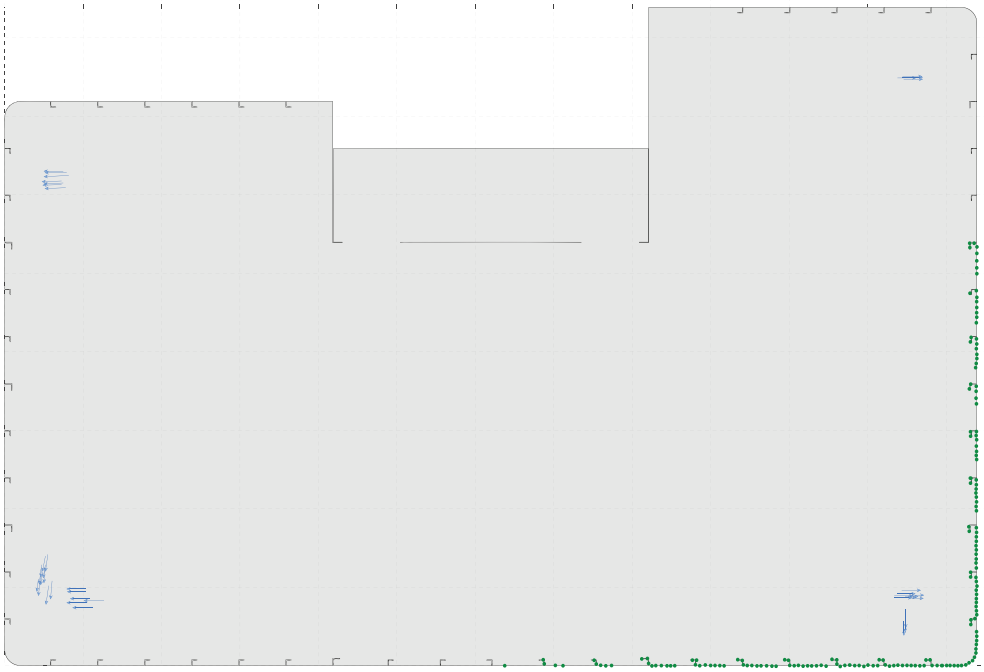


Figure 5.3.5.3: Initial pose estimation using feature matching in Guardian static environment

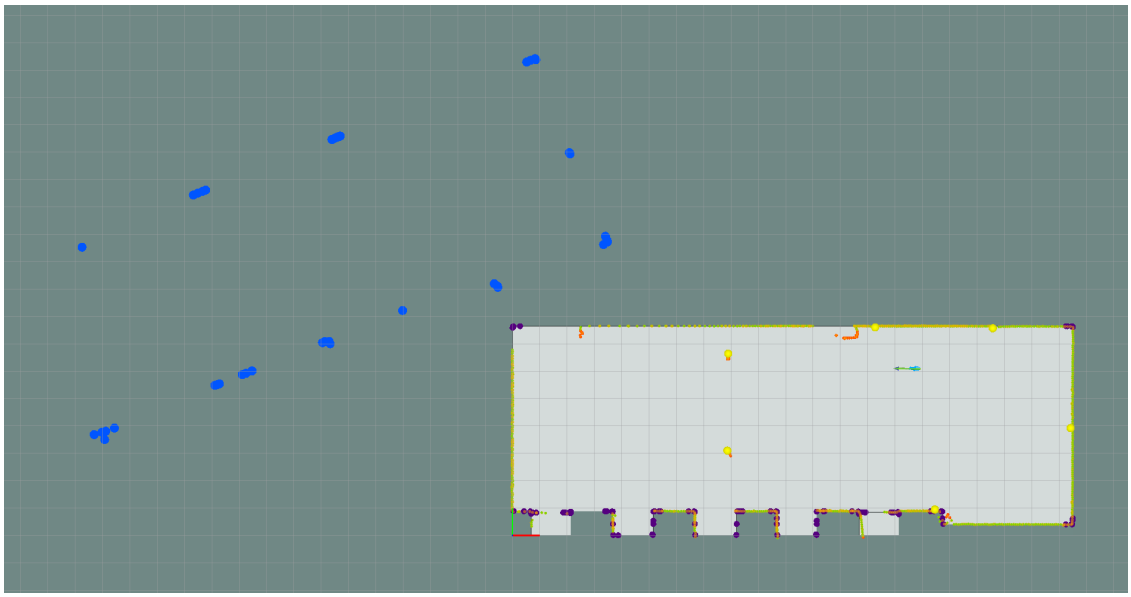


Figure 5.3.5.4: Initial pose estimation using feature matching in Jarvis environment

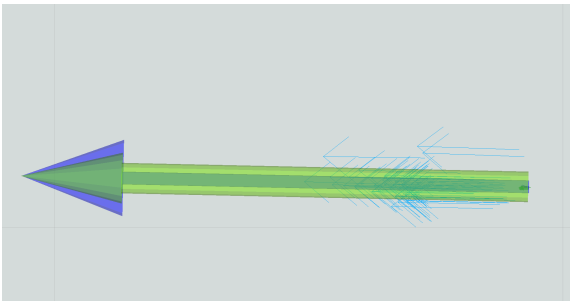


Figure 5.3.5.5: Initial pose estimation distribution in Jarvis environment

5.3.6 Translation and rotation errors

By analyzing tables 5.2 and 5.3 it can be seen that the proposed 3 DoF localization system can achieve pose tracking with less than 1 cm in translation error and less than 1 degree in rotation error.

As expected, the tests at lower velocities had less mean error than the ones at higher speeds while requiring slightly less computation time. Also, the tests on the Jarvis robot had more mean error than the corresponding simulator tests. This is due to laser scan deformation that couldn't be simulated and also because the simulators were only able to add Gaussian noise to the laser measurements. To compensate the simulators limitations, the error in odometry and laser measurements in simulation was deliberately higher than the expected values for the physical platforms. This explains why the localization system needed more time to perform the pose estimations in the simulator tests. It can also be seen that adding unknown objects increased the mean error and required computation time while adding dynamic objects increased these values even further (given that a moving object appears in a laser scan as a deformed version of its static shape).

Comparing the results of the tests performed with Jarvis robot with the tests of the Pioneer robot, it can be seen that even severely reducing the laser range (from 80 to 6 meters), moving the robot in dynamic environments with large occluding objects and even when using a low resolution map (25 mm in the Pioneer tests vs 10 mm in the Jarvis tests), the localization system still managed to track the robot pose with mean translation error (17 mm) bellow the map resolution (25 mm).

These results show that the localization system can reliably achieve high accuracy pose tracking even in dynamic and challenging environments.

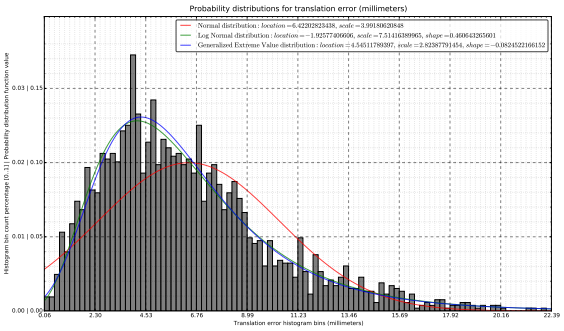


Figure 5.3.6.1: Probability distributions for the localization system translation errors

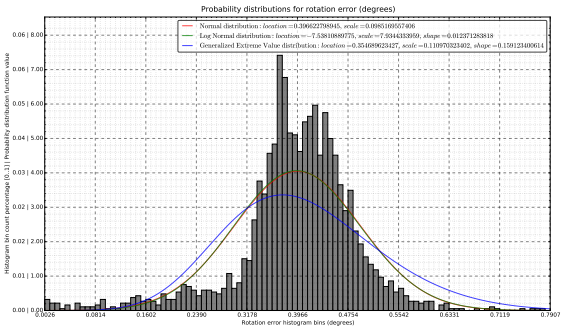


Figure 5.3.6.2: Probability distributions for the localization system rotation errors

5.3.7 Computation time

Looking at the computation time graphs in figure 5.3.7.1 and table 5.2 and appendix A it can be seen that the localization system can register the point clouds very fast (between 5 and 30 milliseconds), which is more than enough to process all the incoming laser scans in real time (typically LIDARs generate scans every 100 milliseconds). Moreover, the computation time seems to be very stable, with occasional peaks due to laser deformation or varying percentage of outliers.

The global computation time is mostly associated with the point cloud registration stage, while the rest of it is due to normal estimation and point cloud preprocessing algorithms.

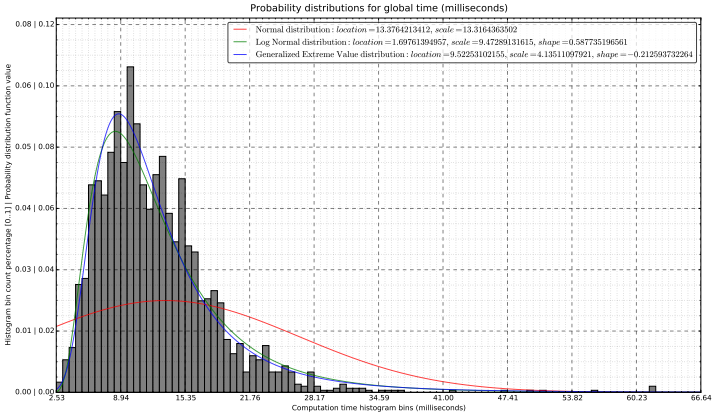


Figure 5.3.7.1: Probability distributions for the localization system computation time

5.4 6 DoF localization system tests

5.4.1 Overview

The main 6 DoF results retrieved with the localization system are shown in table 5.4 and appendix B. The first two experiments were done to evaluate the accuracy of the point cloud registration algorithms while the last one aimed to test the robustness against temporary absence of valid sensor data (in this test the Kinect had periods in which most of its field of view was outside the map).

These tests were retrieved with a known initial pose and used ICP point-to-point as tracking algorithm and ICP point-to-plane as tracking recovery method. The 3D map was done with the localization system in SLAM mode and used continuous surface reconstruction in order to create an accurate representation of the environment and reduce the impact of the measurements noise. This map was later downsampled using a voxel grid with 20 mm cells in order to allow real-time processing of the Kinect data.

The next sections will provide an analysis of the 6 DoF results achieved with the proposed localization system (shown in appendix B). It will start by explaining the importance of point cloud processing and how it helps reduce the cloud registration time. Then it will give an overview of the 3D map building capabilities and it will finished with an analysis of the translation and rotation errors along with the computation time.

5.4.2 Point cloud preprocessing

Point cloud preprocessing can have a very significant role when performing 6 DoF cloud registration for self-localization. This is due to real-time requirements and also because the computation time increases substantially when registering large point clouds. One way to control this problem is by preprocessing the point cloud with a voxel grid to adjust the level of detail and also assign a limit to the number of points that come from sensors. This can be achieved with random sampling or similar point selection techniques, as can be seen in figure 5.4.2.1 (filtered cloud has larger points than the reference cloud). Moreover, depending on the sensor used, it may be wise to restrict the points to a given range, and discard the rest that are too far way (given that these measurements will have more errors).

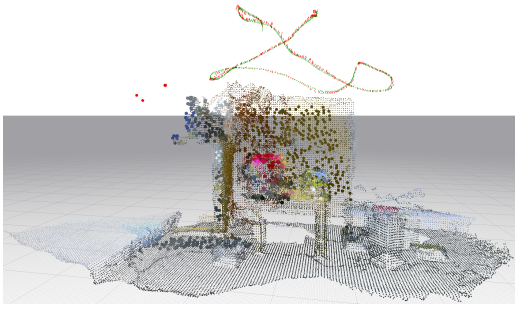


Figure 5.4.2.1: Example of filtered point cloud from Kinect data

Table 5.4: 6 DoF results

Test conditions				Translation error (millimeters)		Rotation error (degrees)		Outliers percentage [0..100]		Global computation time (milliseconds)	
Section	Path shape	Path velocities	Map cell resolution	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation
B.1.1	Overview fly	30 cm/s	20 mm	17.926	9.789	3.027	0.638	0.165	0.172	30.714	11.407
B.1.2	Mainly translations	20 cm/s	20 mm	14.162	7.990	2.486	0.975	0.983	1.874	32.296	10.757
B.1.3	Mainly rotations	10 cm/s	20 mm	16.576	9.389	2.738	0.583	1.214	3.148	29.818	9.134

5.4.3 SLAM

The proposed localization system can perform mapping of the environment and is able to build very detailed point clouds. These point clouds can be continuously re-sampled with the Moving Least Squares algorithm (example in figure 5.4.3.2) in order to reconstruct the surfaces of the environment and attenuate the double wall effects and high sensor noise.

The localization system can also dynamic update the 3D map in order to remove missing objects from the reference point cloud and also add new ones.

Figure 5.4.3.1: 3D SLAM using the ground truth poses

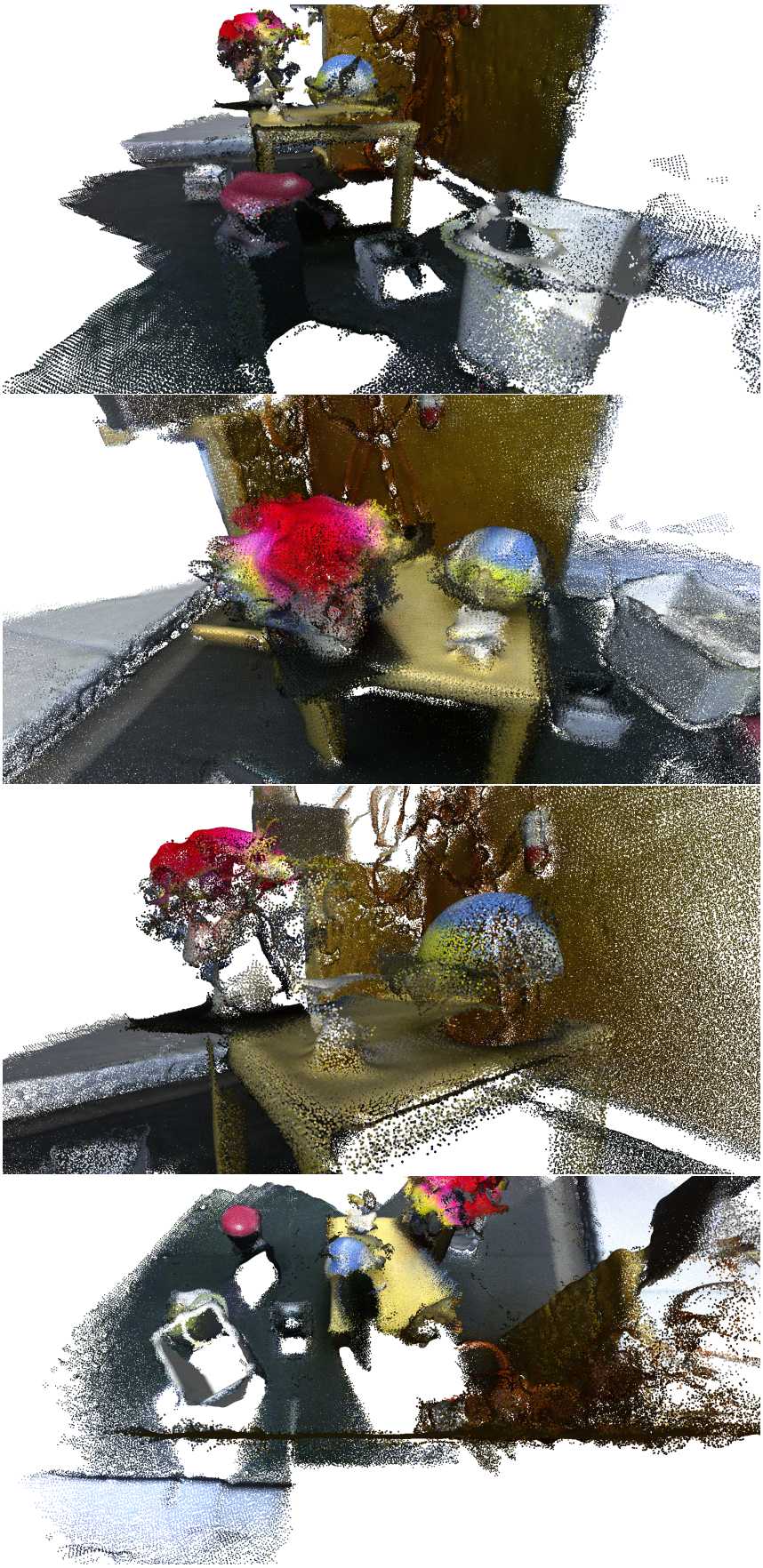


Figure 5.4.3.2: 3D SLAM using the localization system with surface reconstruction

5.4.4 Point cloud registration

Looking at the animated figures in appendix B and the assembly of the registered point clouds (shown in figure 5.4.4.3), it is clear that the proposed localization system is able to register point clouds with high accuracy, even when they are severely downsampled (due to real-time processing constraints). Moreover, the localization system is robust against temporary absence of sensor data (when the field of view of the Kinect was outside the known map) and was able to quickly recover to accurate tracking when valid sensor data was given. In these situations the recovery algorithms were activated, switching the registration algorithm from ICP point-to-point to ICP point-to-plane. Given that these sensor data outages were temporary, the initial pose algorithms weren't necessary. Nevertheless, if the Kinect remained outside the map for a longer period of time, the localization system would alert that the tracking was lost and it would try to find its current pose using feature matching.

Comparing figure 5.4.4.1 and figure 5.4.4.2, it can be seen that the proposed localization system achieved better pose tracking than the *ethzasl_icp_mapper* ROS package [PMC+11] (authors of this Kinect dataset). Moreover, the SLAM map shown in figure 5.4.3.2 seems to have better quality than the map generated using the ground truth poses (shown in figures 5.4.3.1 and 5.4.4.4).

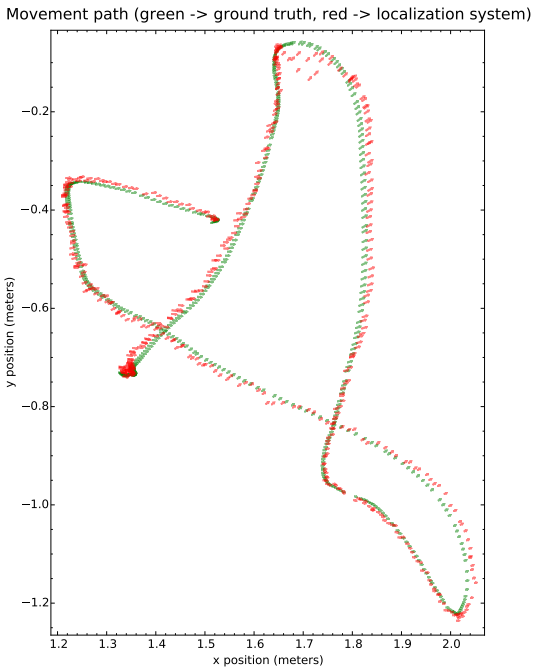


Figure 5.4.4.1: Poses estimated by the ground truth and localization system

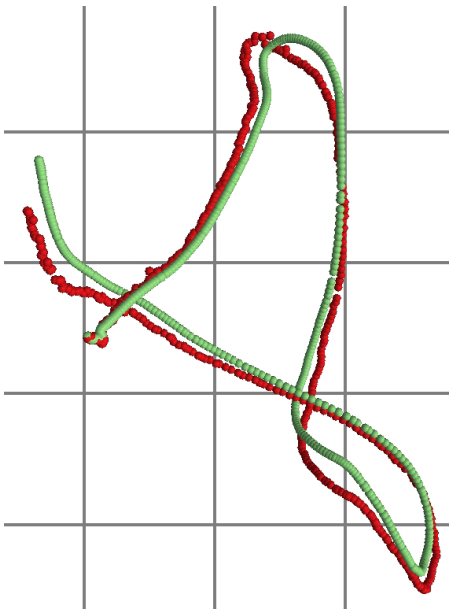


Figure 5.4.4.2: Poses estimated by the *ethzasl_icp_mapper* ROS package [PMC+11]

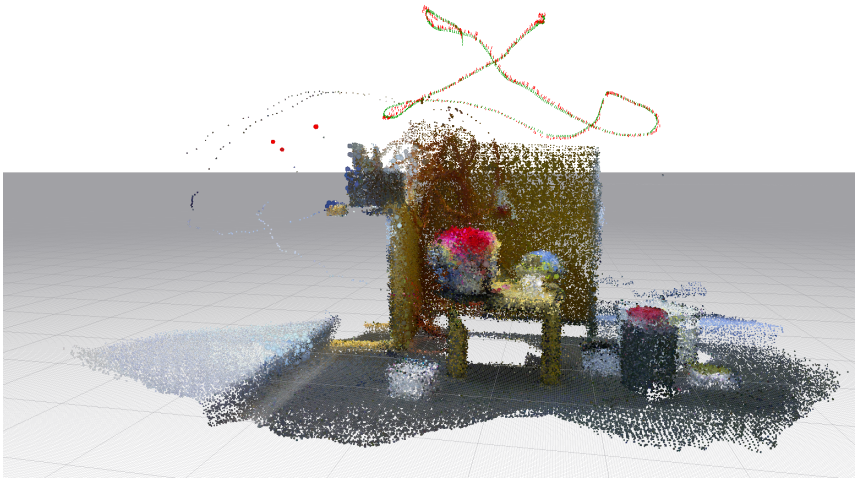


Figure 5.4.4.3: Point clouds assembled on top of the map using the localization system poses (for the top part of the figure: green arrows → ground truth poses, red arrows → localization system poses)

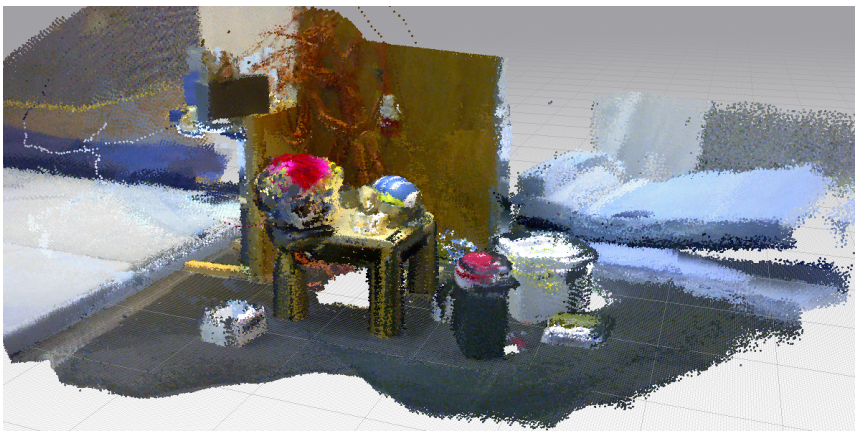


Figure 5.4.4.4: Point clouds assembled on top of the map using the ground truth poses

5.4.5 Translation and rotation errors

The results presented in table 5.4 and appendix B and in figures 5.4.5.1 and 5.4.5.2, shown that the localization system can maintain high accuracy pose tracking, with translation error bellow two centimeters and rotation error around 3 degrees when good sensor data is available. Moreover it can quickly recover from temporary registration problems caused by occlusions or accelerations.

Figures 5.4.4.1 and 5.4.4.2 also shows that the proposed localization system achieved less overall translation error than the tracking system proposed by the authors of this Kinect dataset [PMC+11].

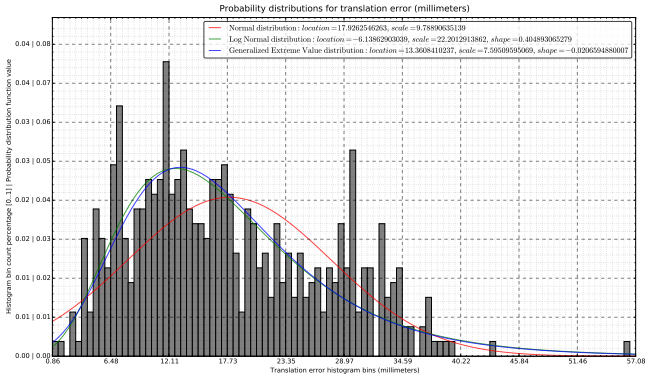


Figure 5.4.5.1: Probability distributions for the localization system translation errors

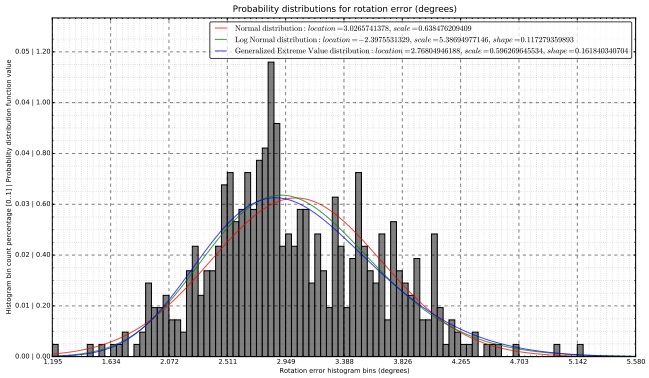


Figure 5.4.5.2: Probability distributions for the localization system rotation errors

5.4.6 Computation time

The localization system was able to achieve a mean computation time low enough to allow real time processing of the Kinect point cloud sensor data (shown in figure 5.4.6.1). Depending on the accuracy requirements, the computation time can be lowered even further by tuning the preprocessing filters described in section 4.6, in order to reduce the number of points used in the cloud registration.

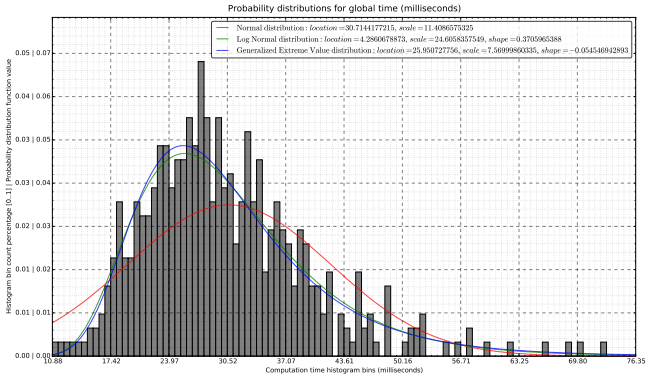


Figure 5.4.6.1: Probability distributions for the localization system global computation time

Chapter 6

Conclusions

The proposed localization system is able to maintain pose tracking with less 1-2 centimeters of translation error and less than a 1-3 degrees of rotation error (in 3 and 6 DoF respectively) with the sensors moving at several velocities even in cluttered and dynamic environments. Moreover, when tracking is lost or no initial pose is given, the system is able to find a valid global pose estimate by switching to more robust registration algorithms that use feature matching. This approach achieved fast pose tracking and reliable initial pose estimation while also providing the set of the accepted initial poses before registration refinement, which can be very valuable information for a navigation supervisor when the robot is in an ambiguous region that can be registered in similar zones of the known map. The system also allows dynamic reconfiguration of the number of laser scans to assemble in order to mitigate laser measurement errors and can adapt its rate of operation according to the robot estimated velocity.

The sub-centimeter accuracy achieved by the proposed localization system along with the dynamic map update capability and the need of no artificial landmarks / ambient modifications will allow the fast deployment of mobile robots capable to operate safely and accurately in cluttered environments.

6.1 Main contributions

The following list gives an overview of the main contributions of the presented work:

- A LIDAR assembler[1] capable of:
  - Merging measurements from several sensors
  - Dynamic adjust its configurations based on the speed of the robot in order to merge more scans when the robot is moving slower
  - Use spherical linear interpolation to reduce scan deformation
  - Emulate a 3D sensor when the LIDAR is mounted on a tilting platform

[1] https://github.com/carlosmccosta/laserscan_to_pointcloud

- An localization system[2] that:
  - Is efficient, modular and extensible to new 3/6 DoF algorithms
  - Can dynamically and incrementally update the localization map
  - Provides the set of acceptable initial poses when reseting tracking (useful for localization supervisors)
  - Gives several localization quality metrics such as percentage, RMSE and angular distribution of the correctly registered points
  - Gives the possibility to use 3 different point cloud registration algorithms based on the tracking state (normal tracking, tracking recovery, initial pose estimation)
- Development of a testing infrastructure to automate the collection, analysis and generation of localization results
- 3 DoF Localization datasets[3]:
  - 4 datasets with subcentimeter ground truth in static and dynamic environments using a robot moving at different speeds
  - 6 Gazebo and 6 Stage simulation datasets
  - Improvement of external 3/6 DoF datasets (ground truth time synchronization and laser transformations calibration)
- Test of the localization system on public datasets and comparison with:
  - Ground truth systems from three different testing environments
  - Widely used localization systems (AMCL and ethzasl-icp-mapper)
  - Well known SLAM system (GMapping)
- In the context of the CARLoS project:
  - The robot hardware configuration proved adequate for the intended tasks and environment
  - The 3 DoF localization system was able to run in real time on the low computational power on-board computer

6.2 Future work

The current implementation of the self-localization system can be further improved with Graphics Processing Unit (GPU) support [TARK10] in order to achieve higher update rates in 6 DoF and also with loop closing algorithms [GSK12] in order to become a complete SLAM system and allow the creation of accurate maps of very large environments. Moreover, it could be extended to support image based localization [Lab14] and semantic perception and mapping of the environment [San13].

[2] https://github.com/carlosmccosta/dynamic_robot_localization
[3] https://github.com/carlosmccosta/dynamic_robot_localization_tests

Appendix A

3 DoF localization system results

The next sections present in detail the results achieved with the localization system configured to 3 DoF.

Each test starts by presenting an interactive sequence of images in order to give an overview of the sensor data retrieved along the movement path of the robot (a regular grid with 1 meter spacing and a inner grid with 5 cm were applied on top of the figures). To allow a more broad analysis of the point cloud registration quality achieved with the proposed localization system, it is also provided figures with all the lasers assembled using the estimated robot poses. For a more visual inspection, the synchronized poses from the localization system (blue arrows), ground truth (green arrows), odometry (red arrows) and AMCL (brown arrows) are also presented. To have a much more detailed description of the robot movement path, several more graphs are shown with information about the evolution of the x and y position, along with the distance that the robot traveled between pose estimations and the associated linear / angular velocities / accelerations. Later on, the translation and rotation errors and their respective probability distributions are shown. To give an overview of the impact of the localization system over the odometry movement estimations, the translation and rotation corrections are also given. In the end, it is presented some graphs with information about the point clouds size in each of the main processing stages of the localization system pipeline, along with the number of registered points, inliers analysis (percentage, RMSE and angular distribution) and finally some more graphs showing how much time the localization system required for each pose estimation.

Only the most representative tests are shown in the next sections. In [1] is available the full test results along with videos and the configurations used.

[1] https://github.com/carlosmccosta/dynamic_robot_localization_tests

A.1 Jarvis robot tests

The next subsections show some of the most relevant results retrieved with the Jarvis robot in the RoboCup field.

A.1.1 Rounded path using the Jarvis robot at 5 cm/s

The figures in this section show the detailed results of the the test performed with the Jarvis robot with a 10 mm resolution map in the RoboCup field following a rounded path and moving at 5 cm/s.

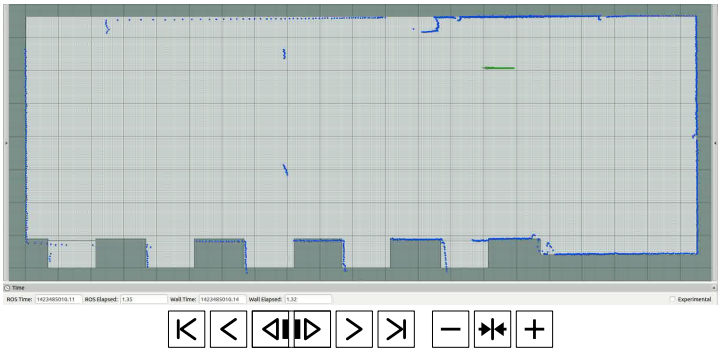


Figure A.1.1.1: Animation of laser scans using the ground truth poses

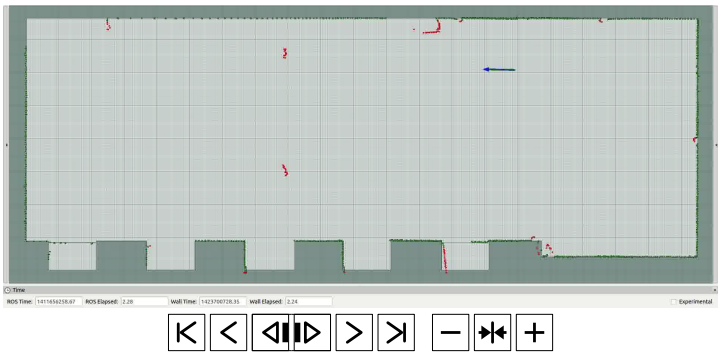


Figure A.1.1.2: Animation of laser scans using the localization system poses

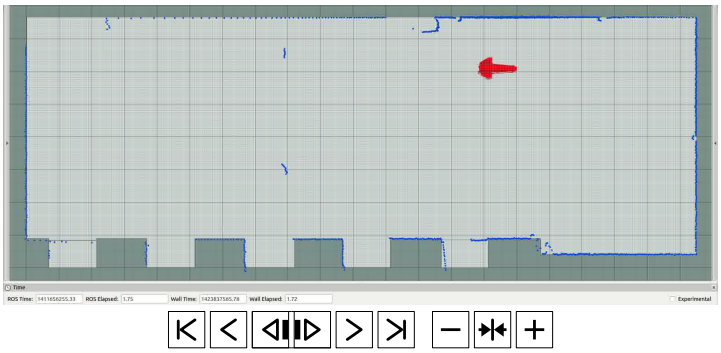


Figure A.1.1.3: Animation of laser scans using the AMCL poses

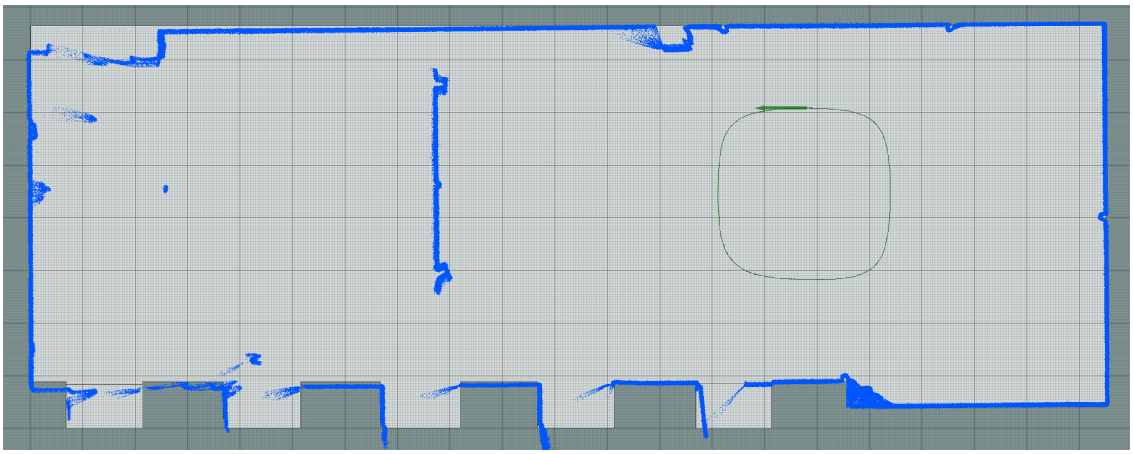


Figure A.1.1.4: Laser scans assembled on top of the map using the ground truth poses

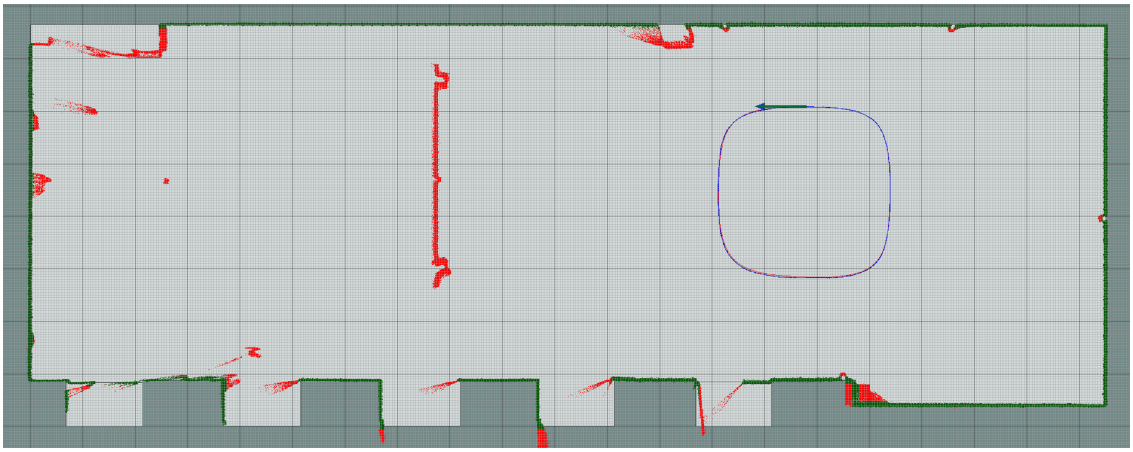


Figure A.1.1.5: Laser scans assembled on top of the map using the localization system poses

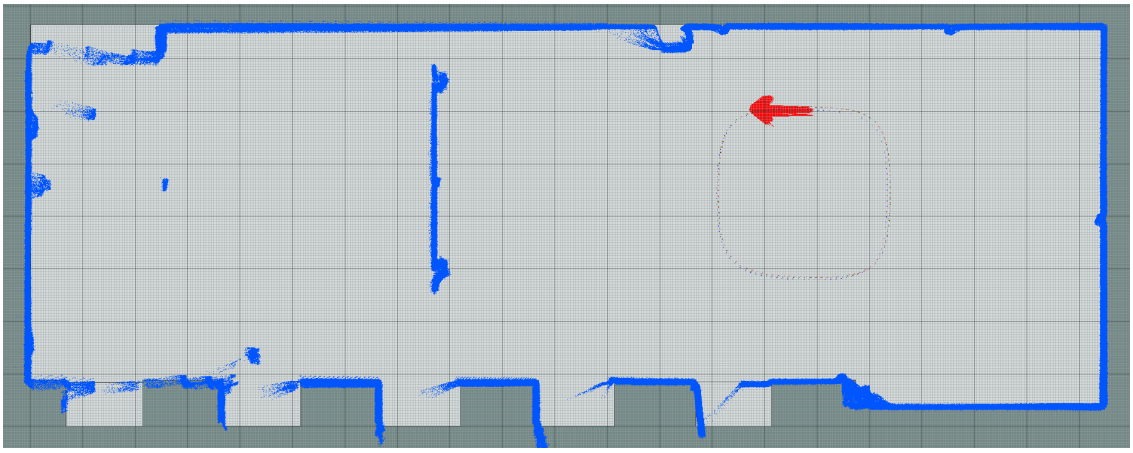


Figure A.1.1.6: Laser scans assembled on top of the map using the AMCL poses

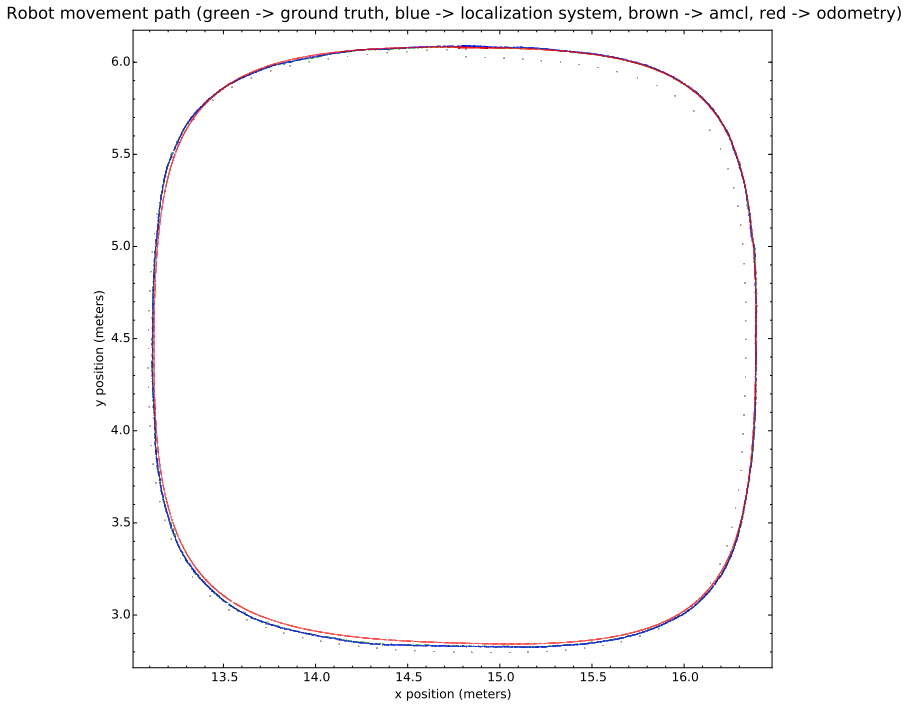


Figure A.1.1.7: Poses estimated by the ground truth, localization system, AMCL and odometry

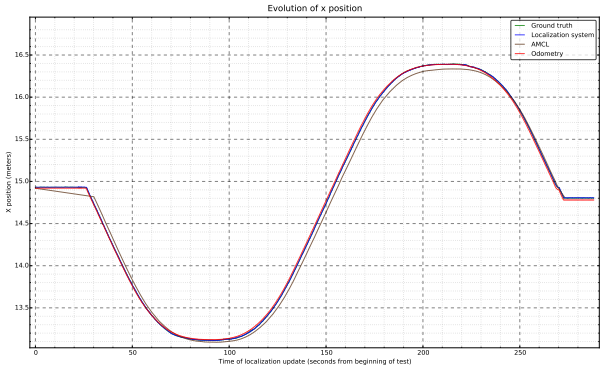


Figure A.1.1.8: Evolution of x position over time

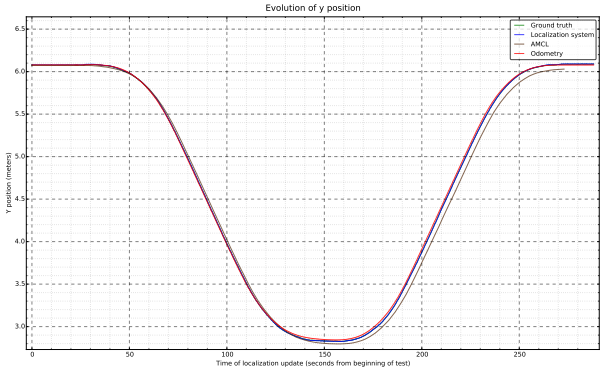


Figure A.1.1.9: Evolution of y position over time

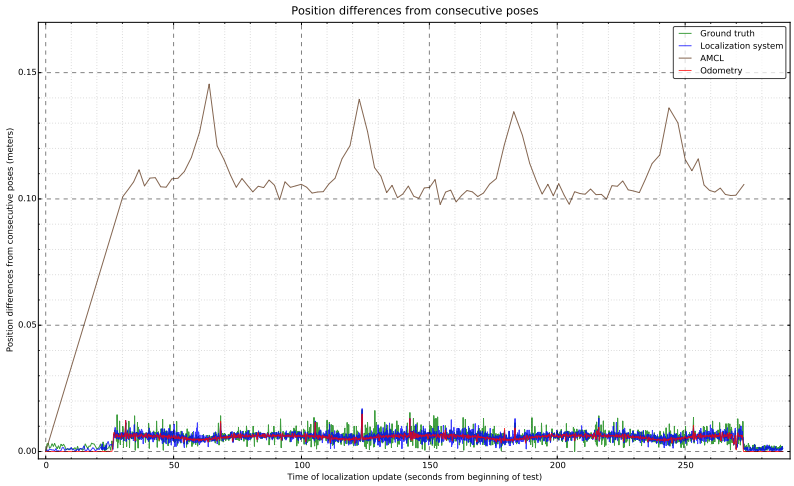


Figure A.1.1.10: Distance traveled between consecutive pose estimations

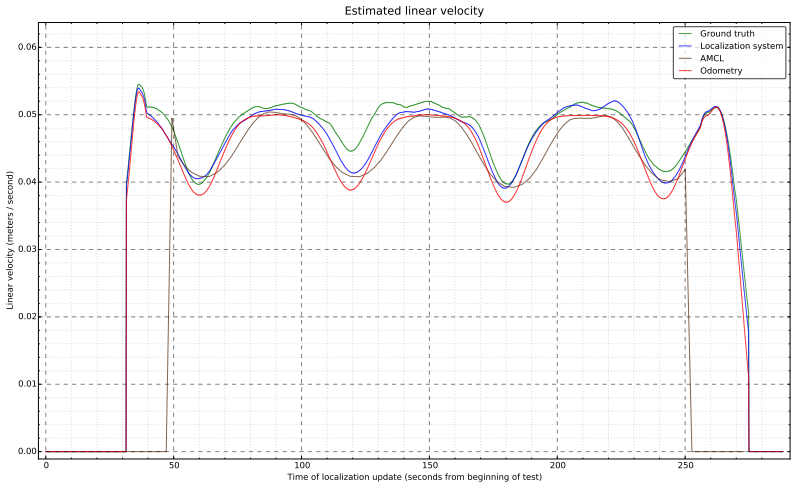


Figure A.1.1.11: Estimated linear velocity

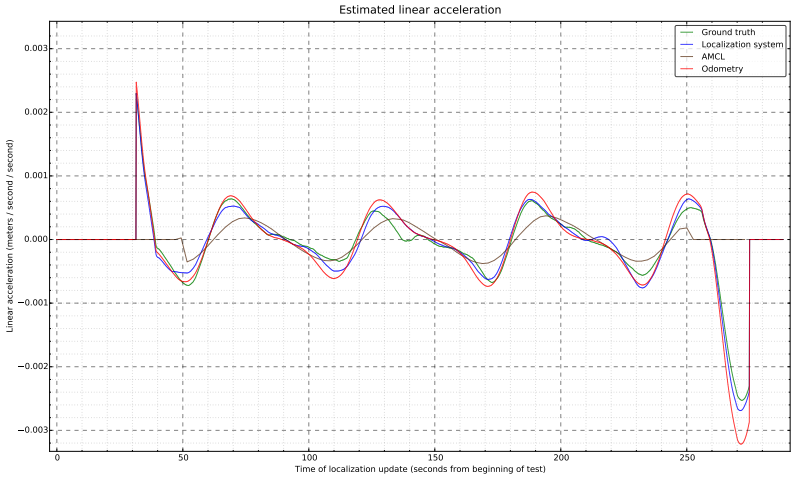


Figure A.1.1.12: Estimated linear acceleration

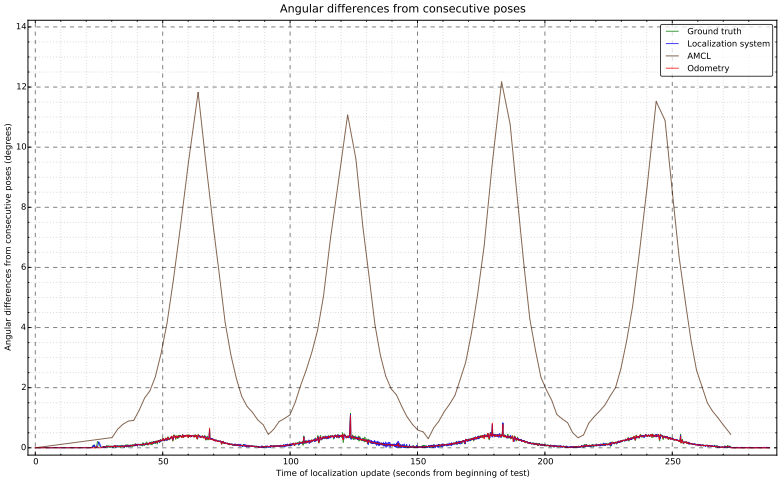


Figure A.1.1.13: Angular differences between consecutive pose estimations

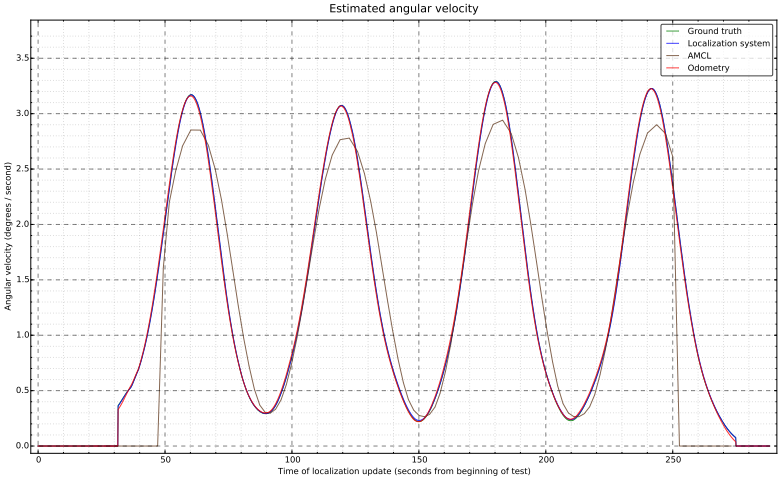


Figure A.1.1.14: Estimated angular velocity

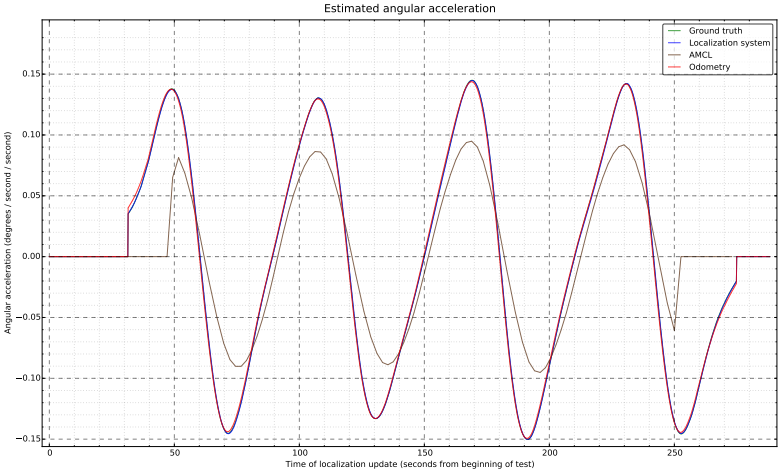


Figure A.1.1.15: Estimated angular acceleration

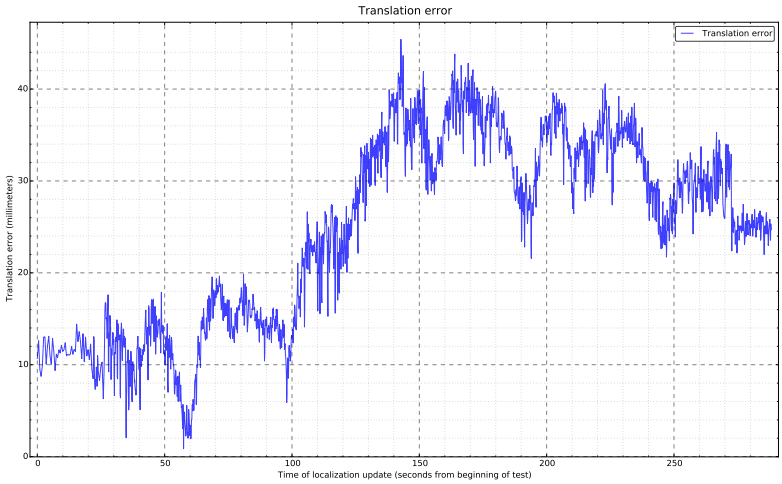


Figure A.1.1.16: Odometry translation errors

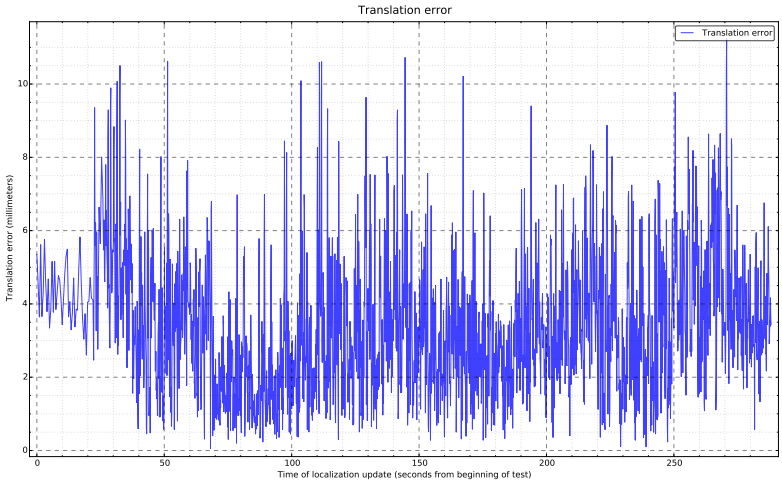


Figure A.1.1.17: Localization system translation errors

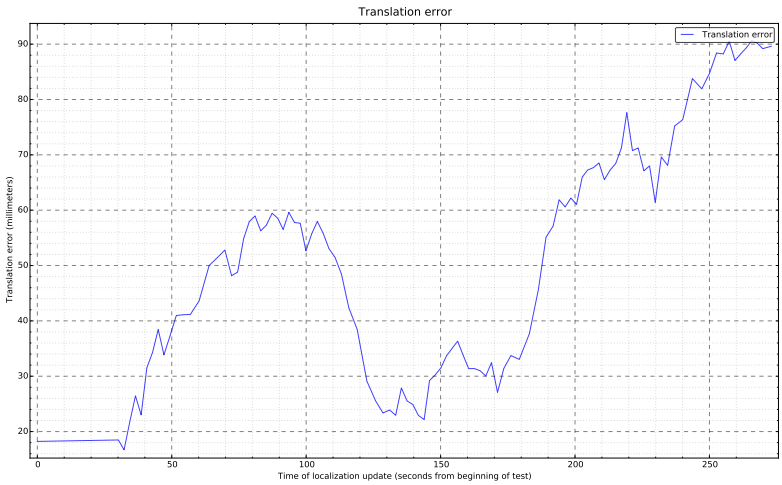


Figure A.1.1.18: AMCL translation errors

Figure A.1.1.19: Odometry translation errors components

Figure A.1.1.20: Localization system translation errors components

Figure A.1.1.21: AMCL translation errors components

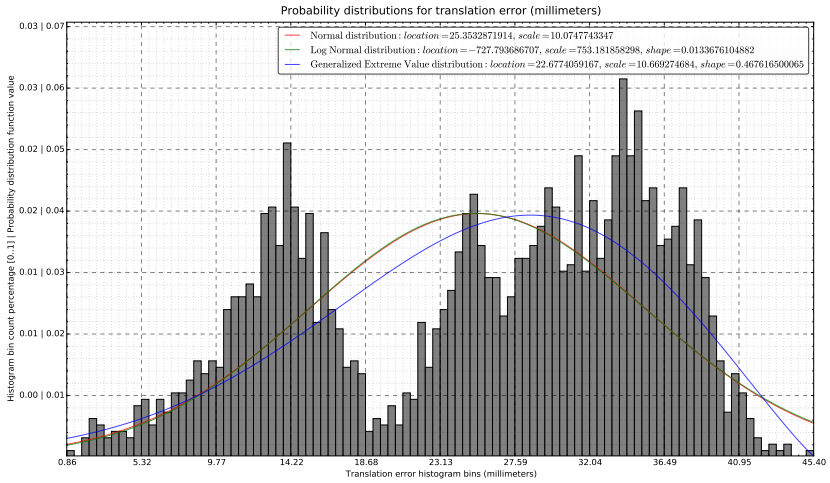


Figure A.1.1.22: Probability distributions for the odometry translation errors

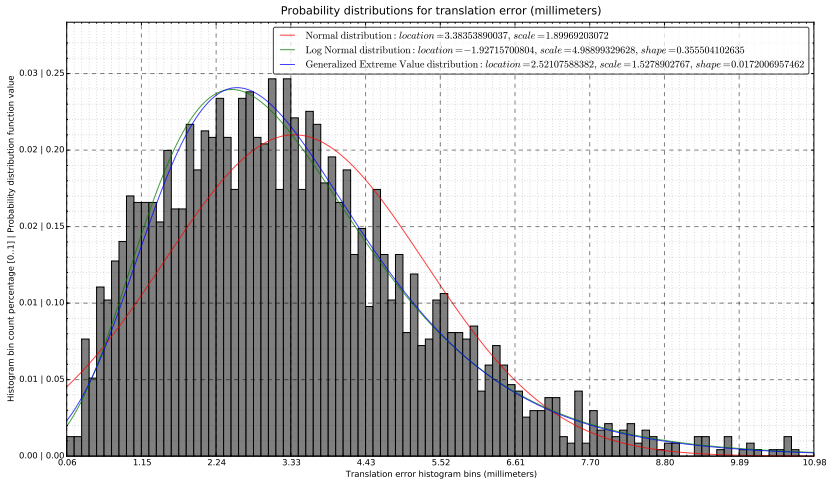


Figure A.1.1.23: Probability distributions for the localization system translation errors

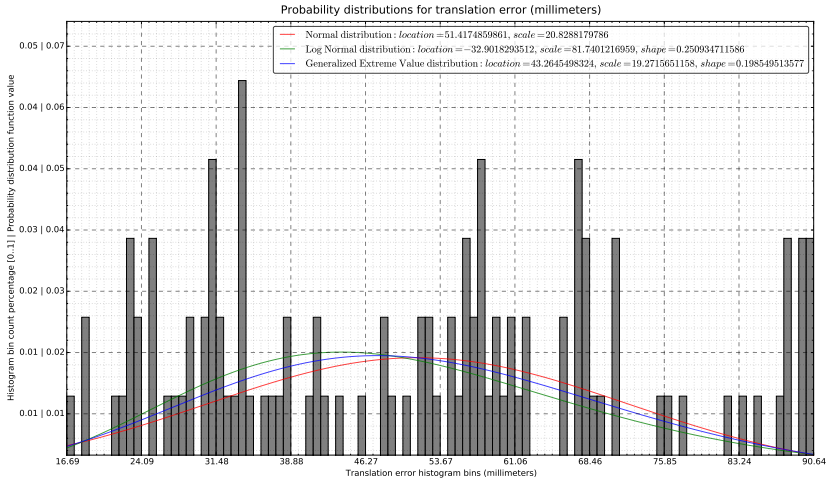


Figure A.1.1.24: Probability distributions for the AMCL translation errors

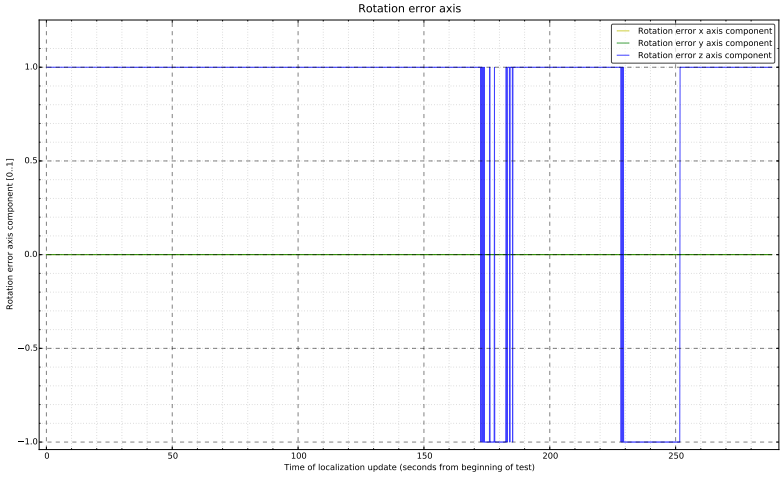


Figure A.1.1.25: Odometry rotation errors axis

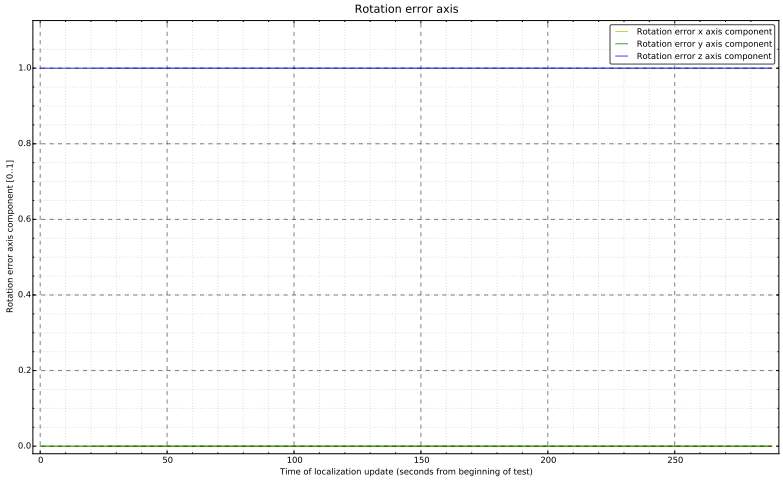


Figure A.1.1.26: Localization system rotation errors axis

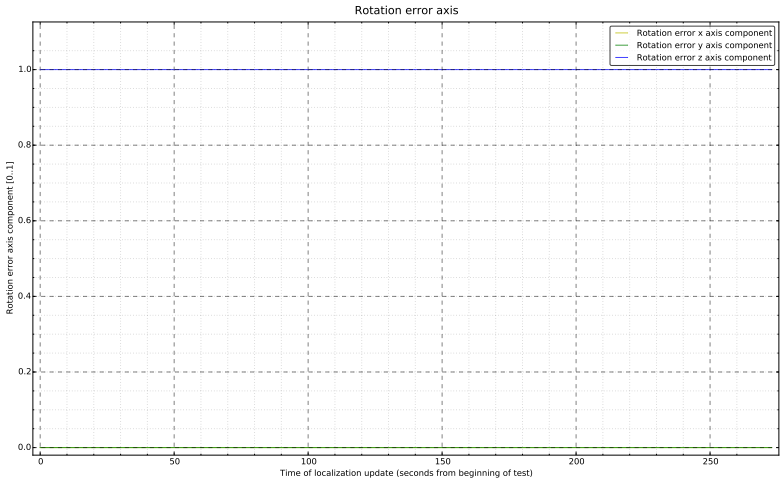


Figure A.1.1.27: AMCL rotation errors axis

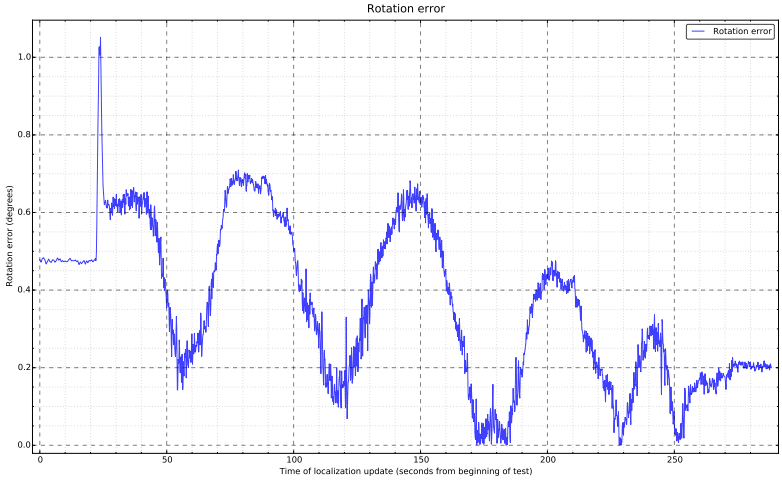


Figure A.1.1.28: Odometry rotation errors

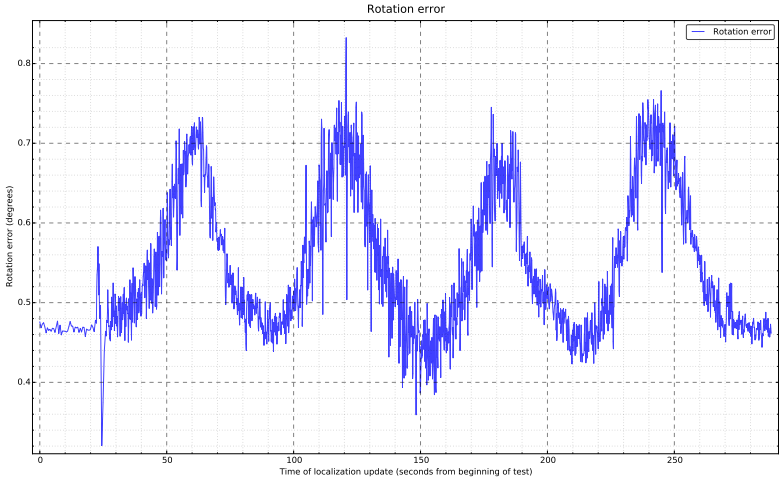


Figure A.1.1.29: Localization system rotation errors

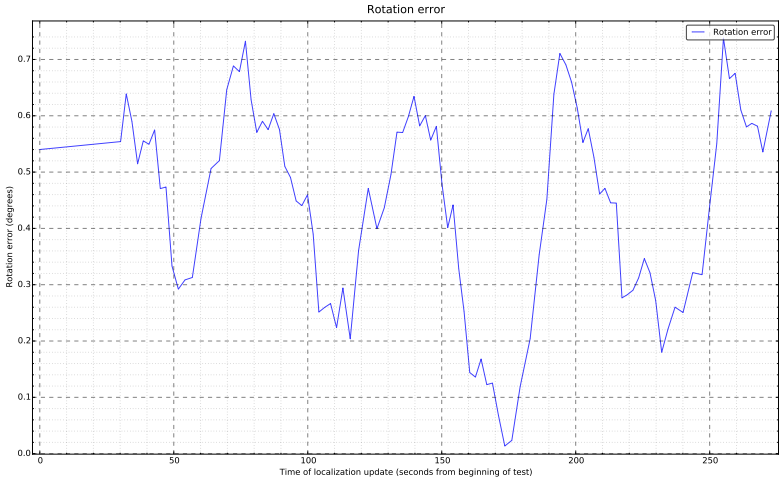


Figure A.1.1.30: AMCL rotation errors

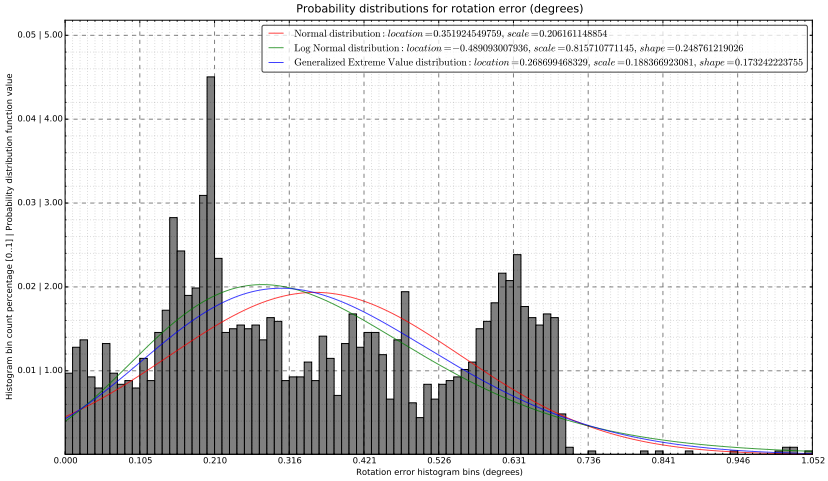


Figure A.1.1.31: Probability distributions for the odometry rotation errors

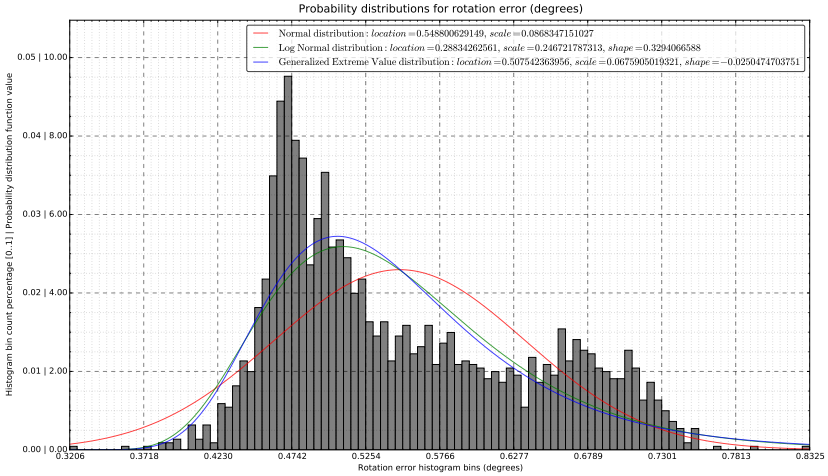


Figure A.1.1.32: Probability distributions for the localization system rotation errors

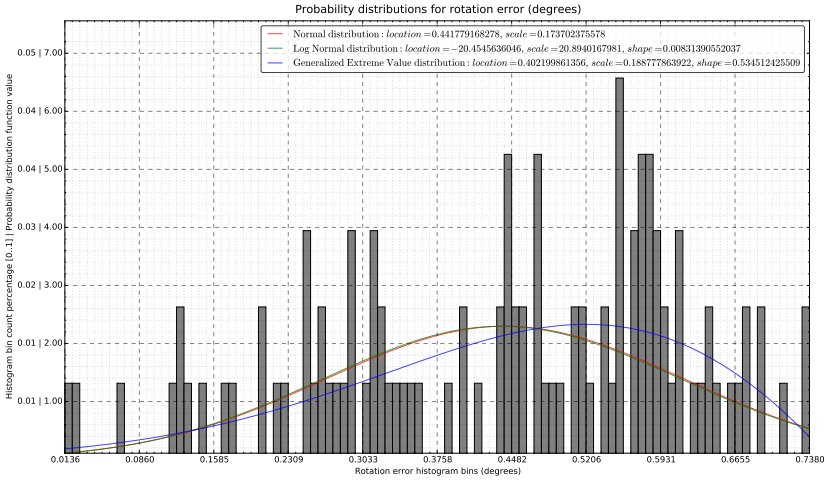


Figure A.1.1.33: Probability distributions for the AMCL rotation errors

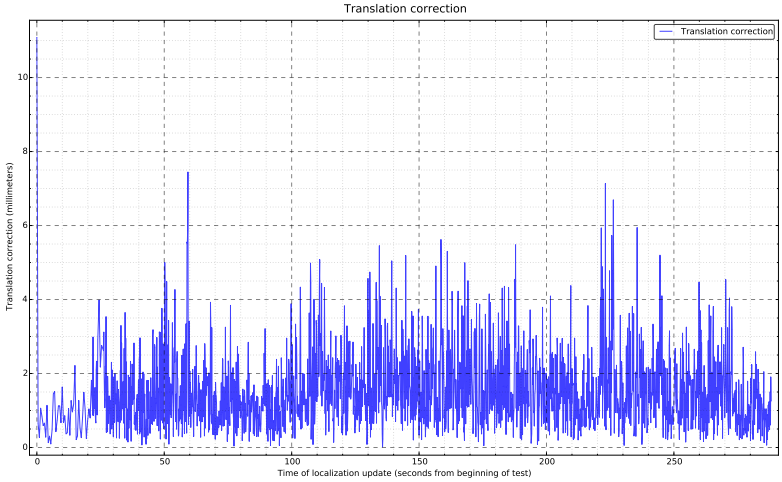


Figure A.1.1.34: Translation corrections performed by the localization system

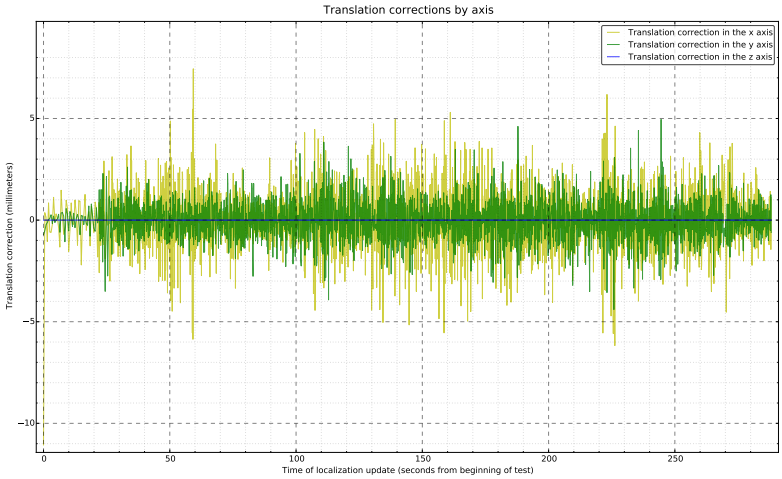


Figure A.1.1.35: Translation corrections components performed by the localization system

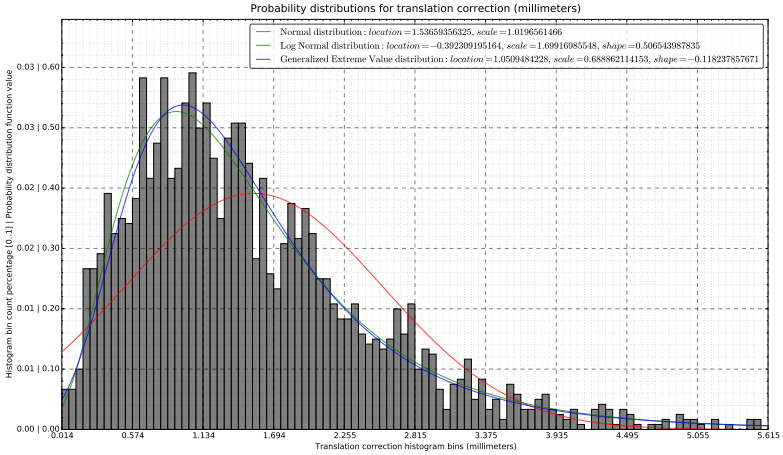


Figure A.1.1.36: Probability distributions for the translation corrections performed by the localization system

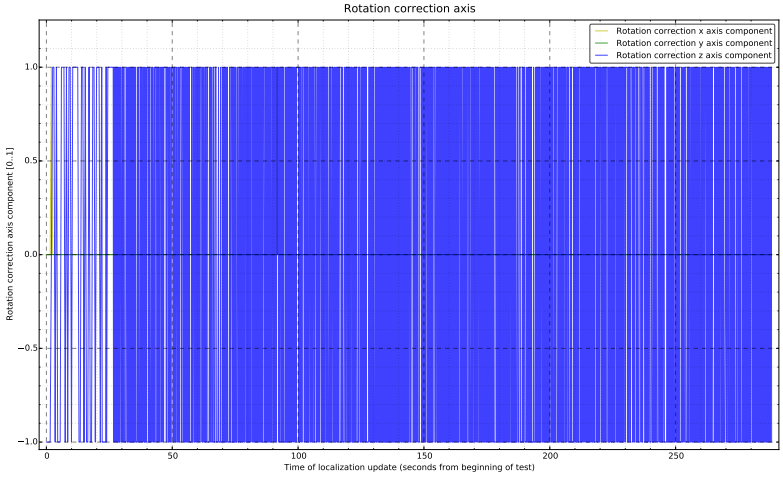


Figure A.1.1.37: Rotation corrections (axis) performed by the localization system

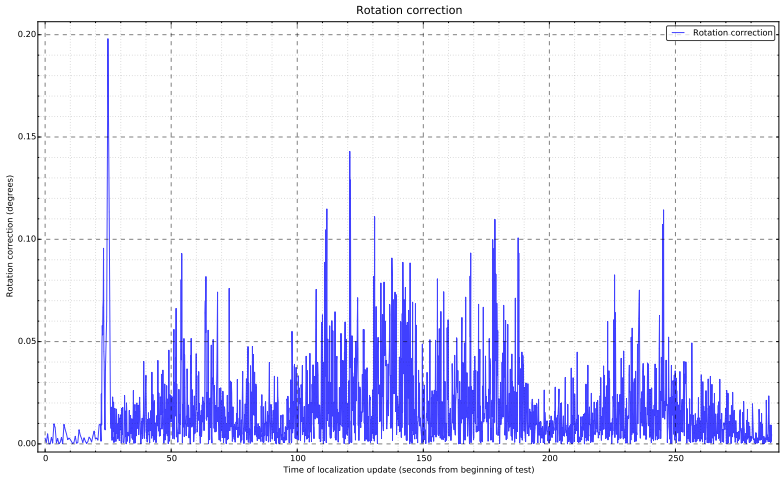


Figure A.1.1.38: Rotation corrections performed by the localization system

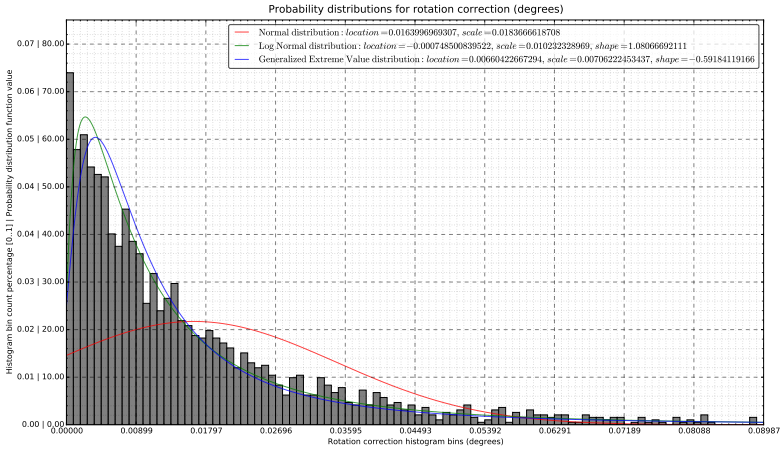


Figure A.1.1.39: Probability distributions for the rotation corrections performed by the localization system

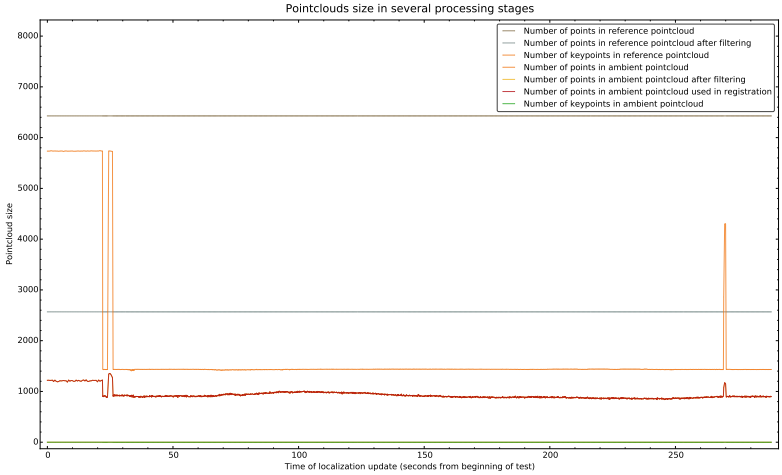


Figure A.1.1.40: Point clouds size in several of the localization system processing stages

Figure A.1.1.41: Number of registered points

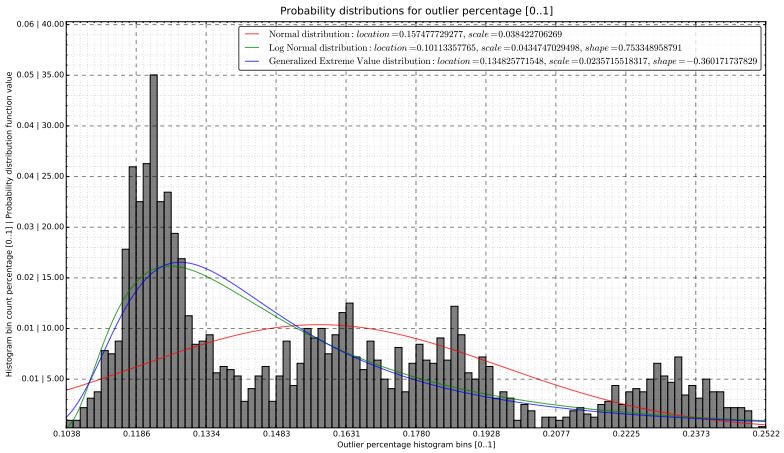


Figure A.1.1.42: Probability distributions for the ambient point cloud outlier percentage

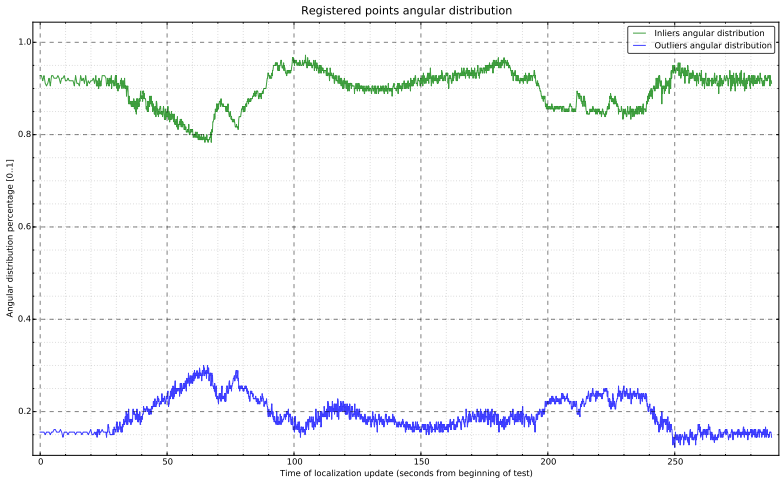


Figure A.1.1.43: Angular distribution percentage of registered points

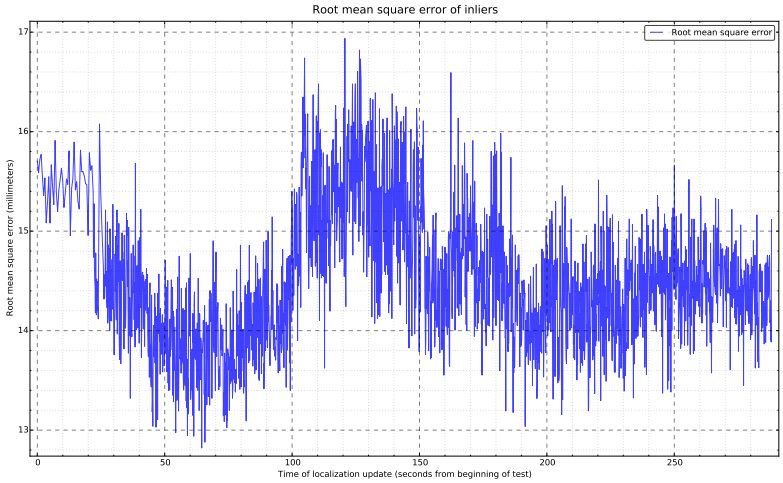


Figure A.1.1.44: Root Mean Square Error of the inliers

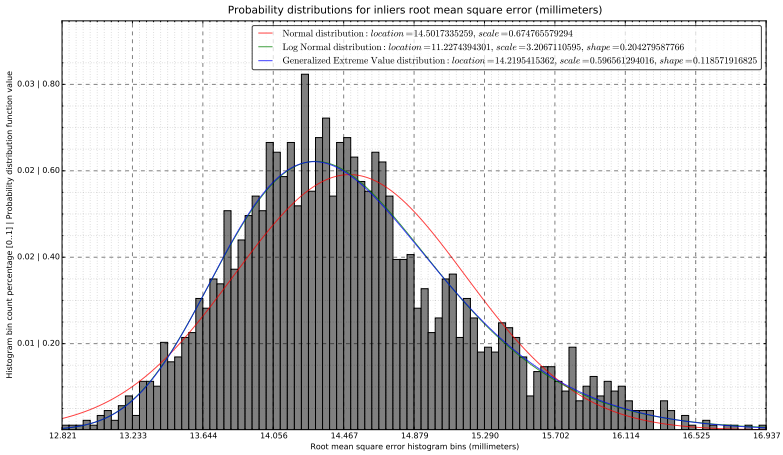


Figure A.1.1.45: Probability distributions for the Root Mean Square Error of the inliers

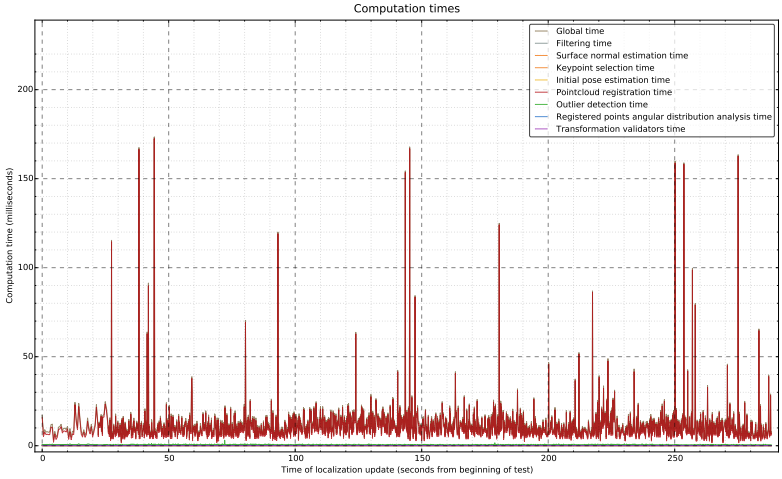


Figure A.1.1.46: Localization system computation times

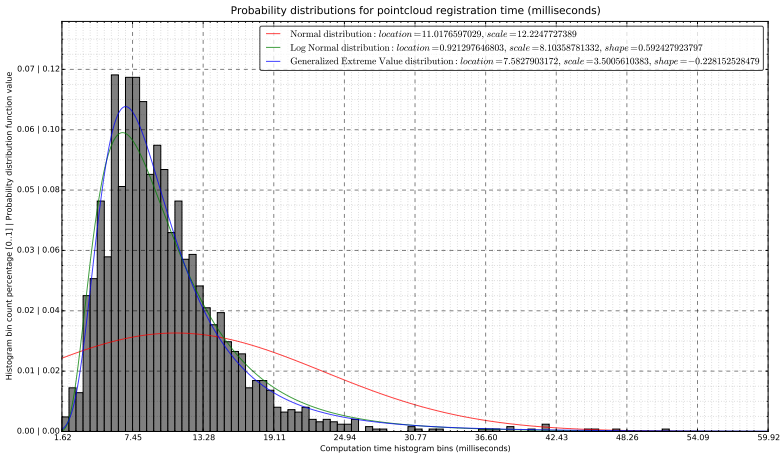


Figure A.1.1.47: Probability distributions for the point cloud registration time

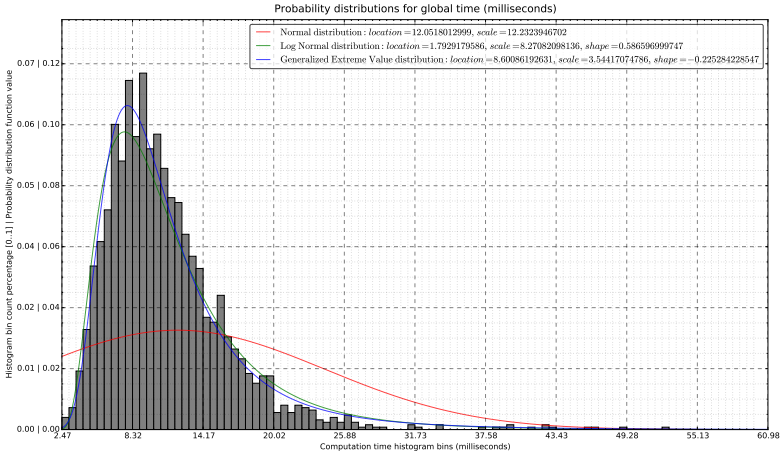


Figure A.1.1.48: Probability distributions for the localization system global computation time

A.1.2 Rounded path using the Jarvis robot at 30 cm/s

The figures in this section show the detailed results of the test performed with the Jarvis robot with a 10 mm resolution map in the RoboCup field following a rounded path and moving at 30 cm/s.

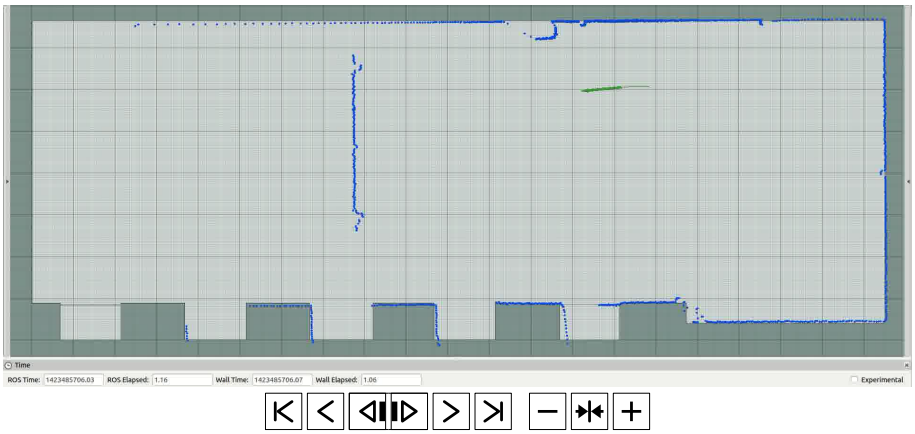


Figure A.1.2.1: Animation of laser scans using the ground truth poses

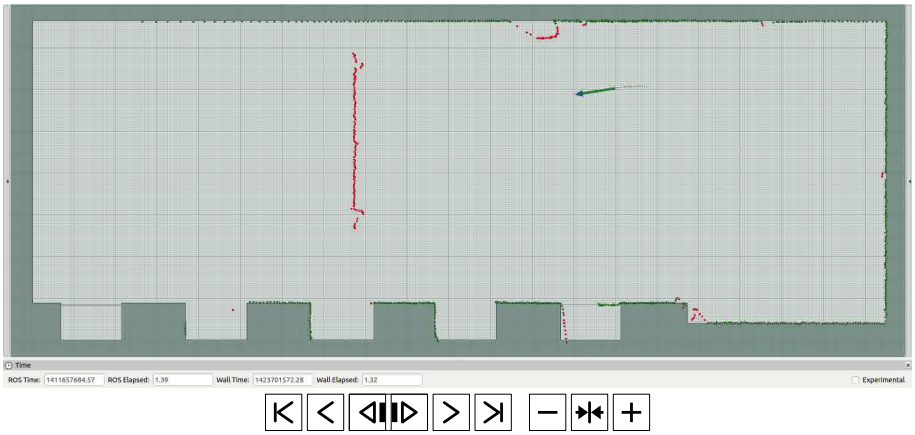


Figure A.1.2.2: Animation of laser scans using the localization system poses

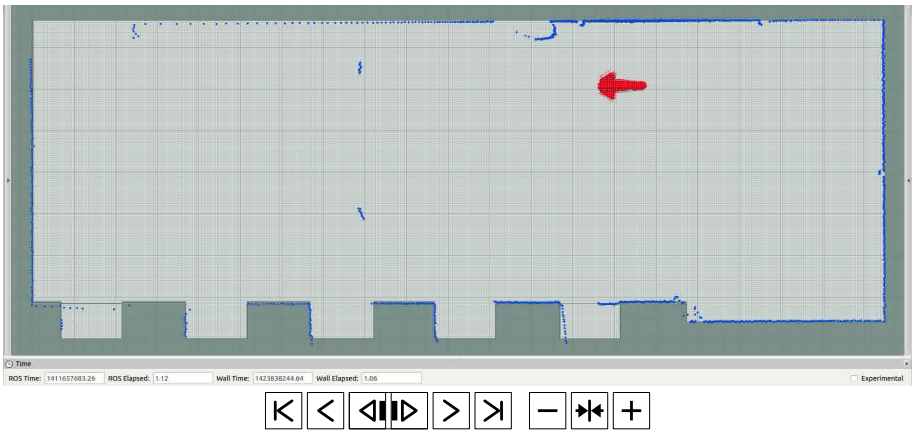


Figure A.1.2.3: Animation of laser scans using the AMCL poses

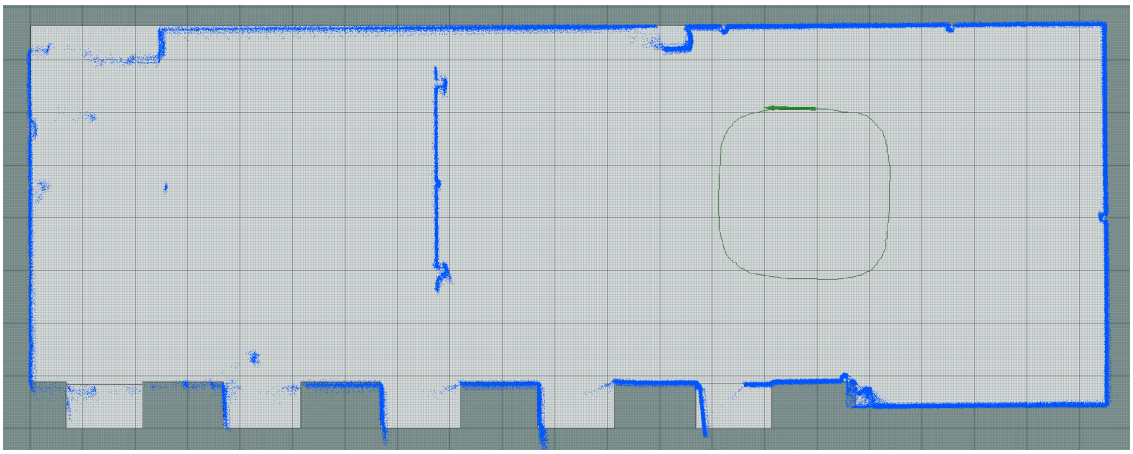


Figure A.1.2.4: Laser scans assembled on top of the map using the ground truth poses

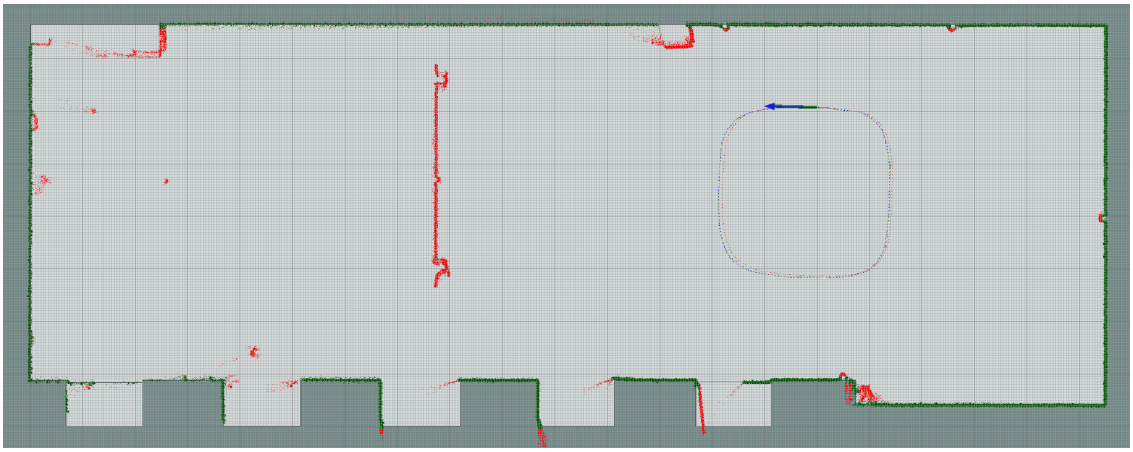


Figure A.1.2.5: Laser scans assembled on top of the map using the localization system poses

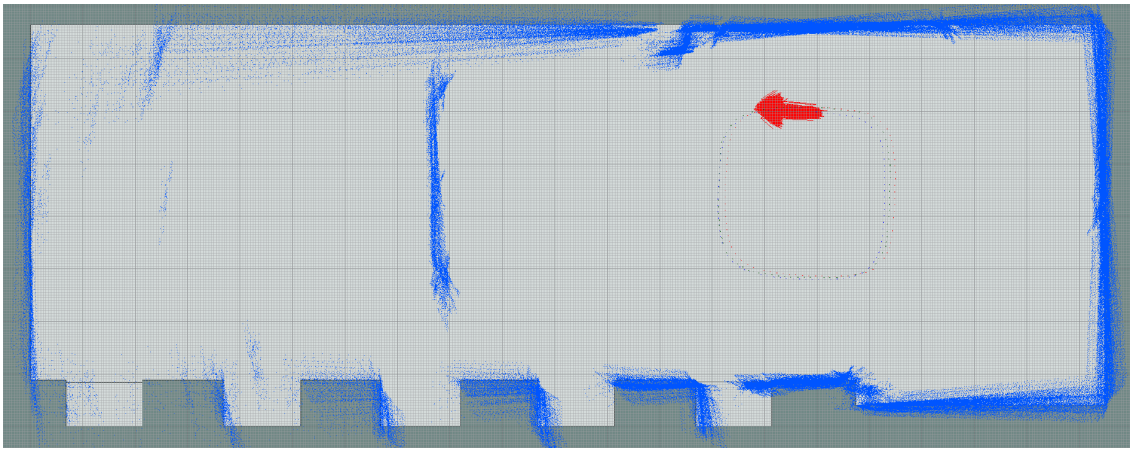


Figure A.1.2.6: Laser scans assembled on top of the map using the AMCL poses

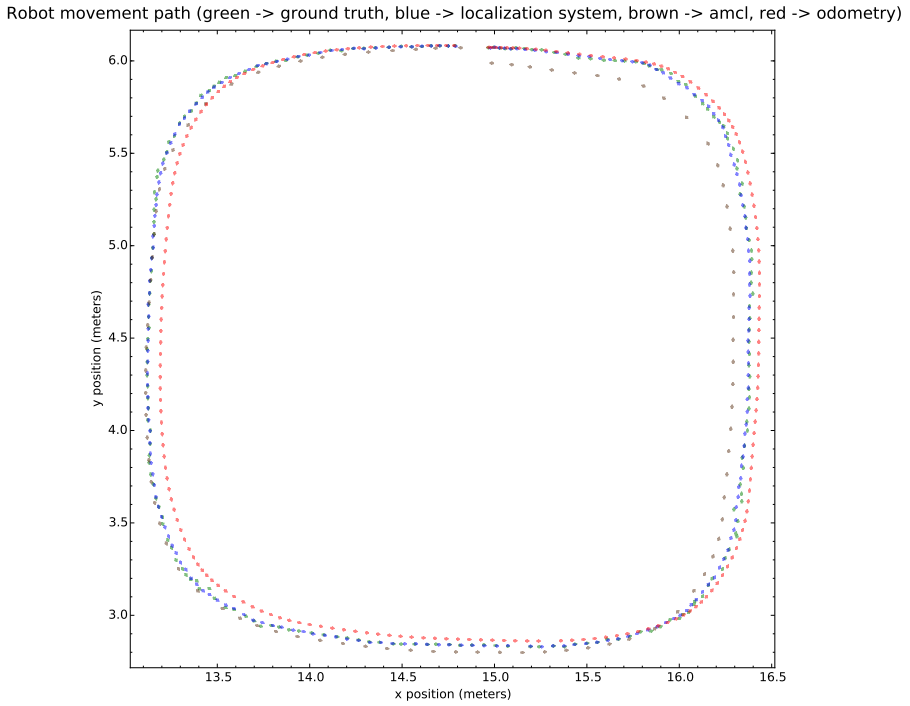


Figure A.1.2.7: Poses estimated by the ground truth, localization system, AMCL and odometry

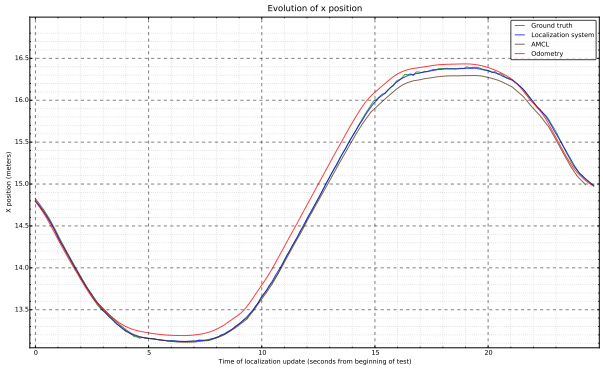


Figure A.1.2.8: Evolution of x position over time

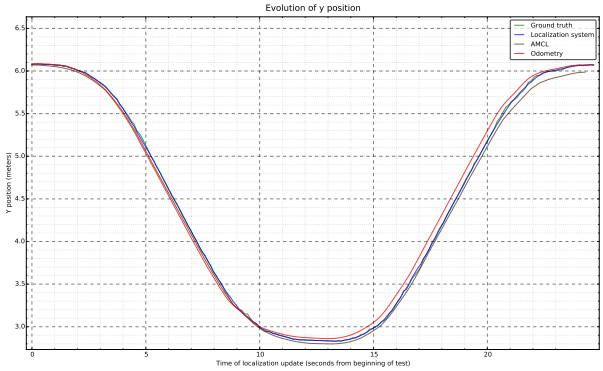


Figure A.1.2.9: Evolution of y position over time

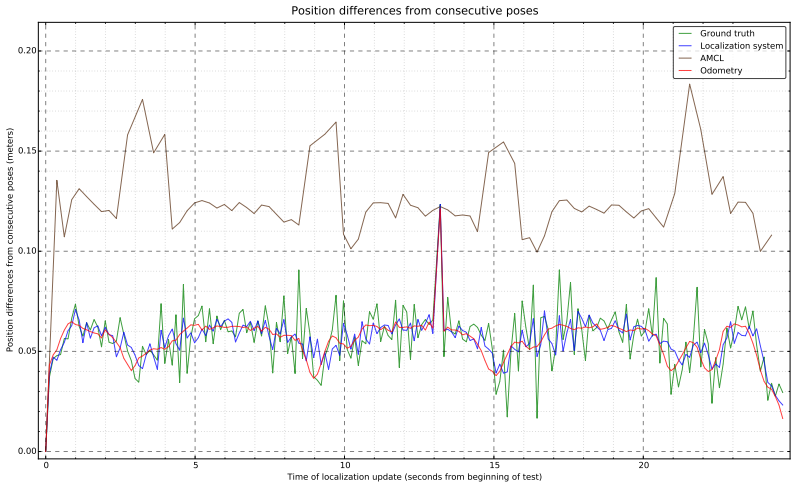


Figure A.1.2.10: Distance traveled between consecutive pose estimations

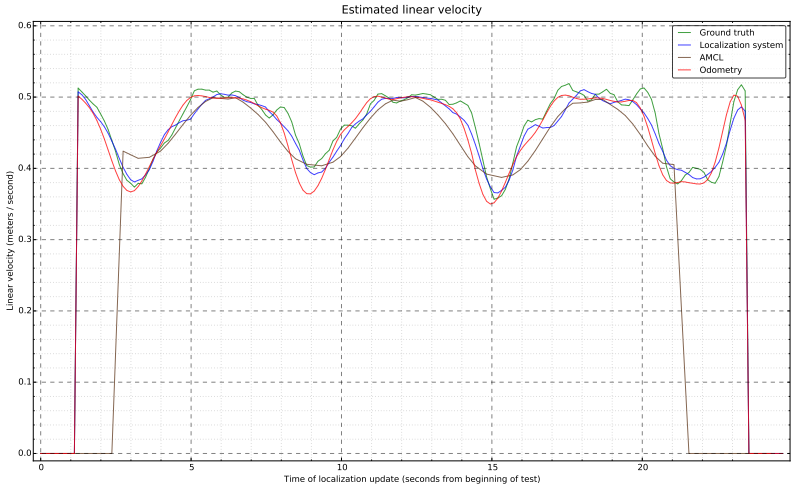


Figure A.1.2.11: Estimated linear velocity

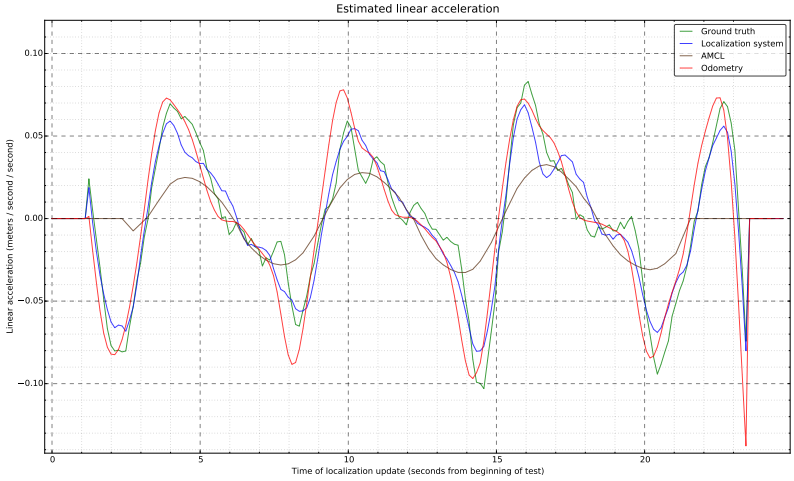


Figure A.1.2.12: Estimated linear acceleration

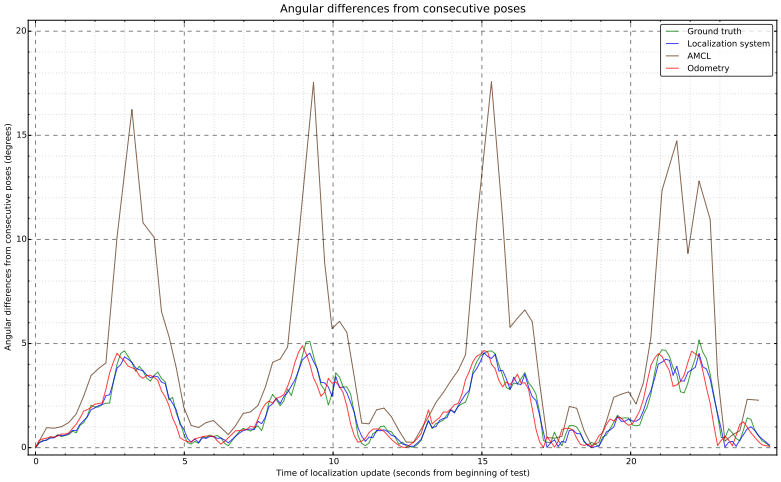


Figure A.1.2.13: Angular differences between consecutive pose estimations

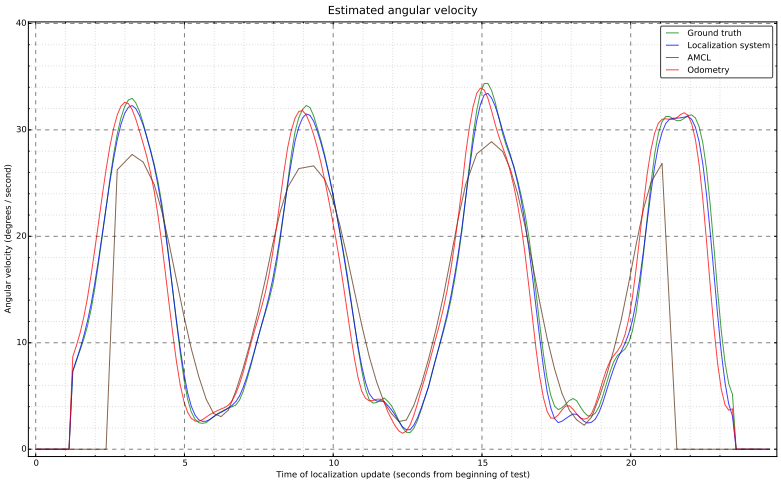


Figure A.1.2.14: Estimated angular velocity

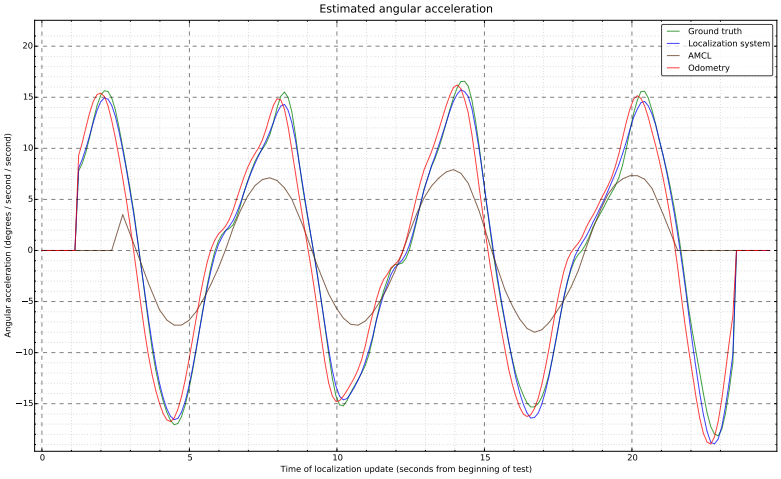


Figure A.1.2.15: Estimated angular acceleration

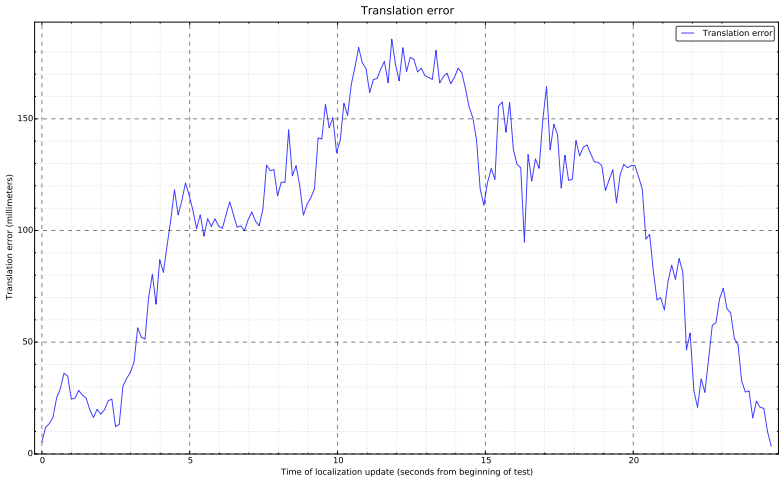


Figure A.1.2.16: Odometry translation errors

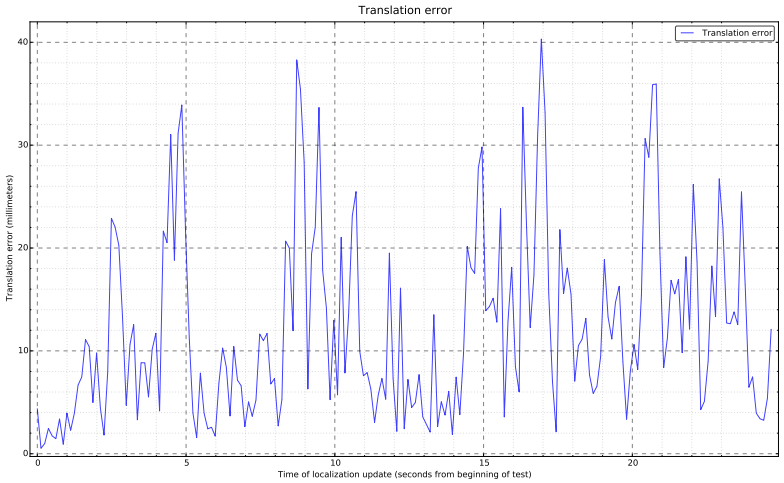


Figure A.1.2.17: Localization system translation errors

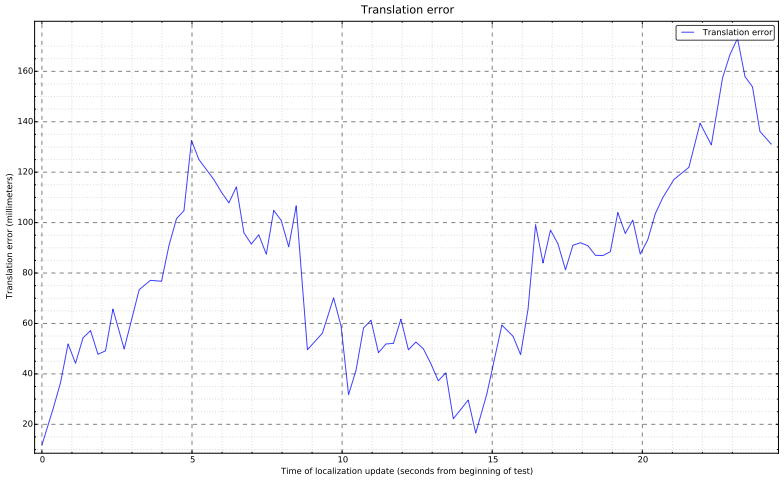


Figure A.1.2.18: AMCL translation errors

Figure A.1.2.19: Odometry translation errors components

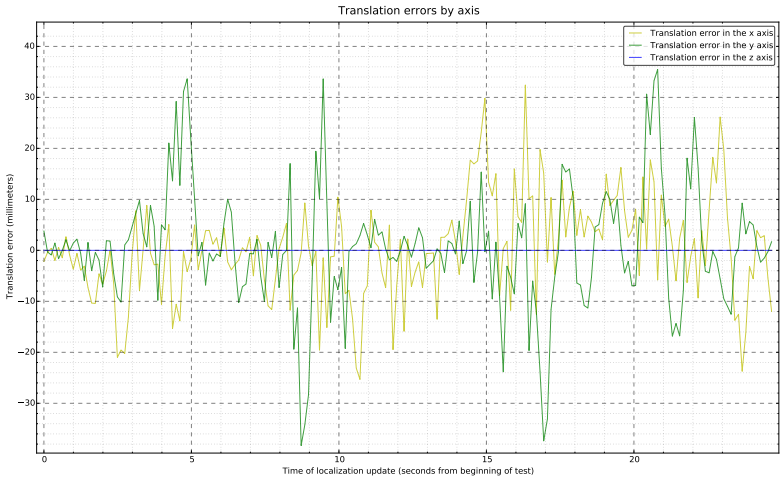


Figure A.1.2.20: Localization system translation errors components

Figure A.1.2.21: AMCL translation errors components

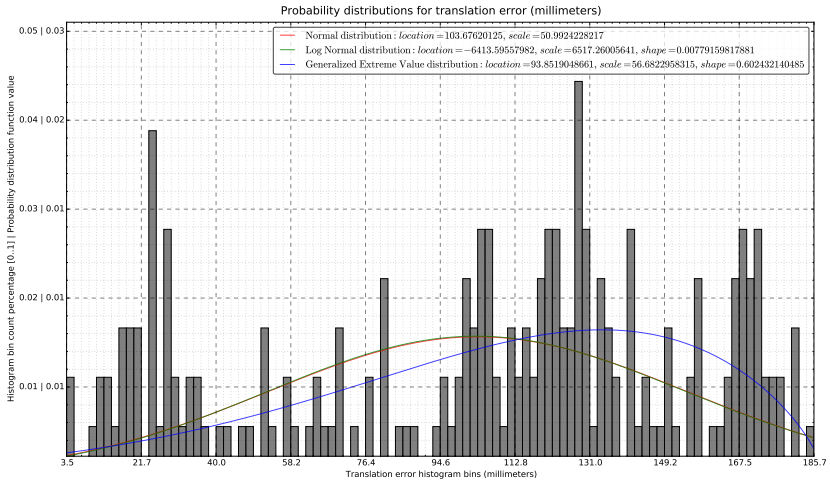


Figure A.1.2.22: Probability distributions for the odometry translation errors

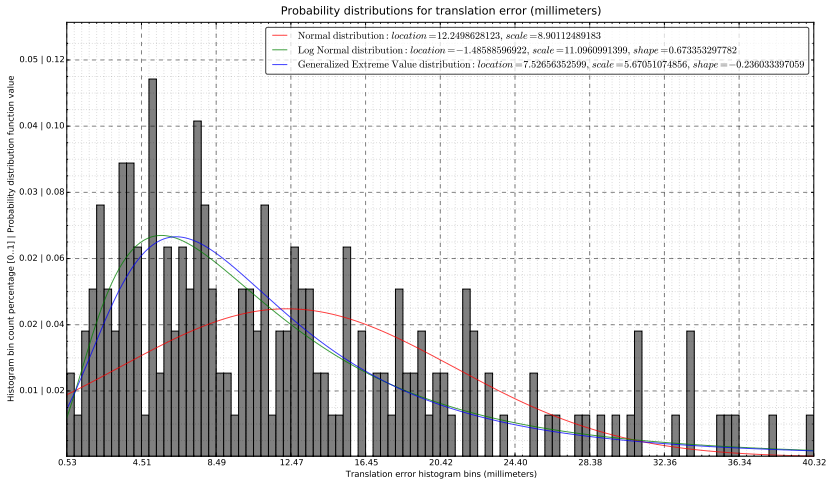


Figure A.1.2.23: Probability distributions for the localization system translation errors

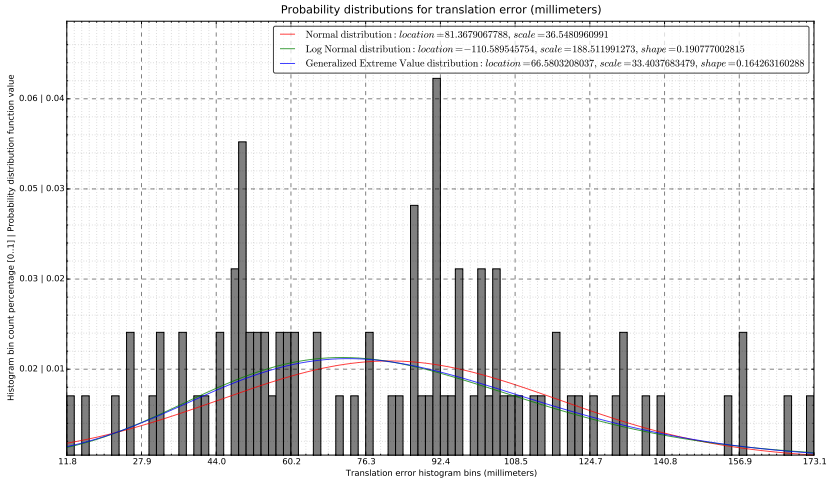


Figure A.1.2.24: Probability distributions for the AMCL translation errors

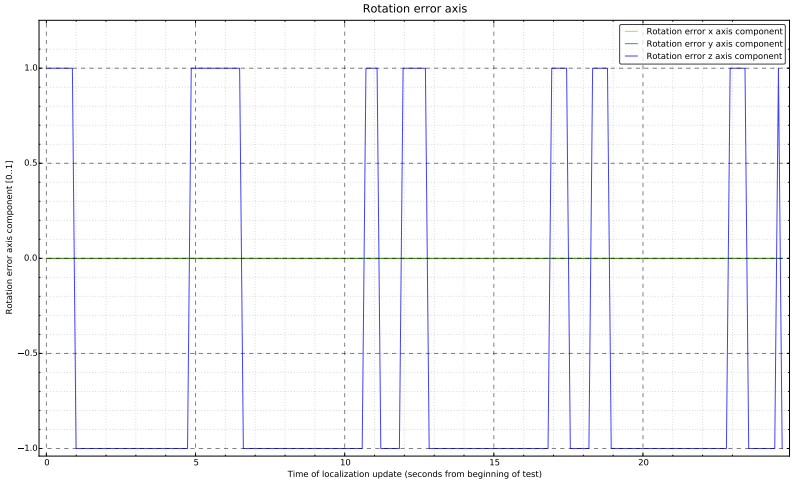


Figure A.1.2.25: Odometry rotation errors axis

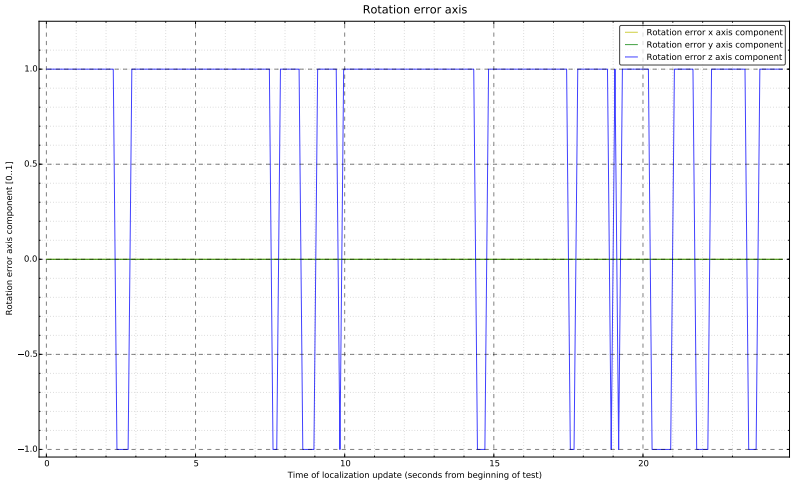


Figure A.1.2.26: Localization system rotation errors axis

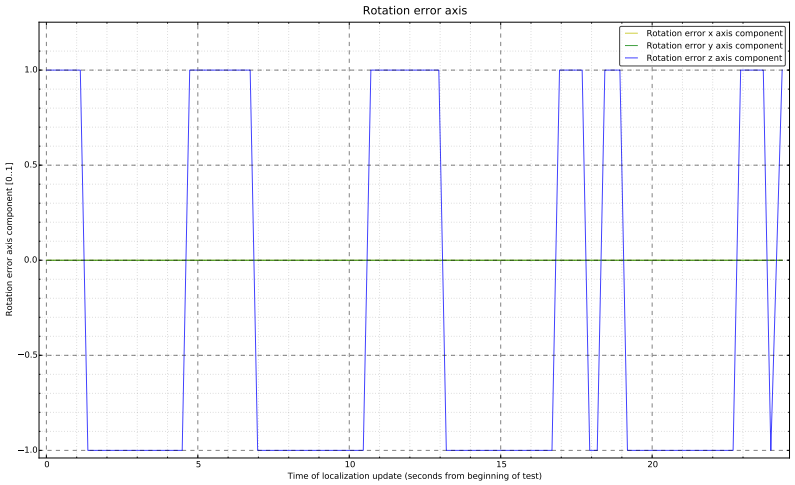


Figure A.1.2.27: AMCL rotation errors axis

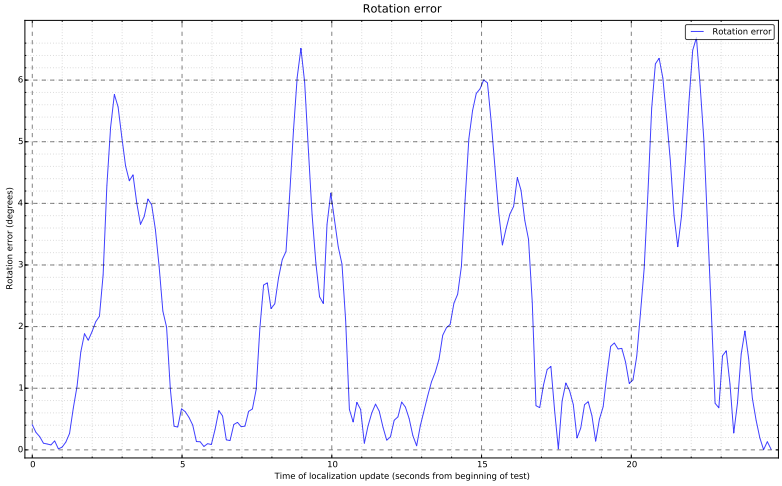


Figure A.1.2.28: Odometry rotation errors

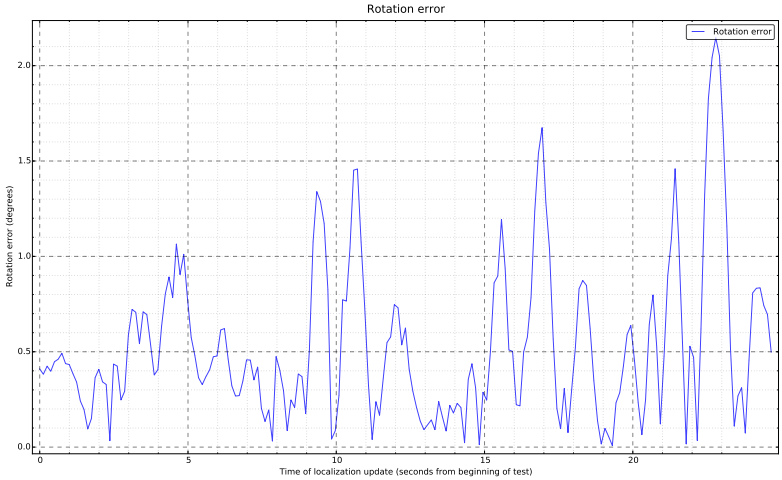


Figure A.1.2.29: Localization system rotation errors

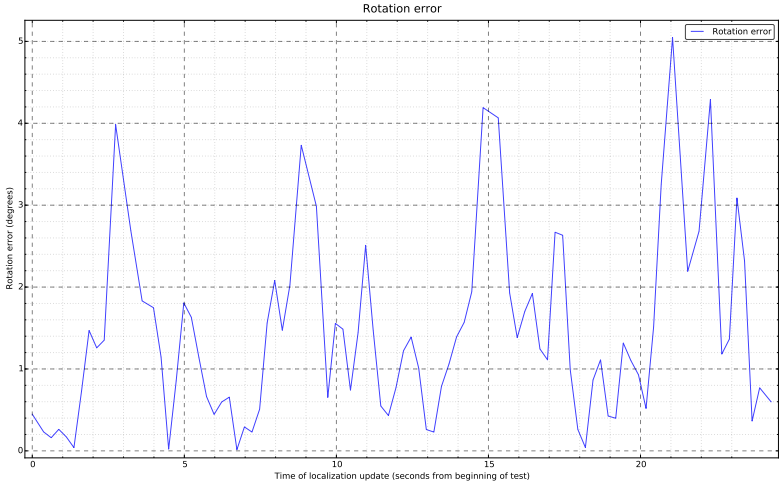


Figure A.1.2.30: AMCL rotation errors

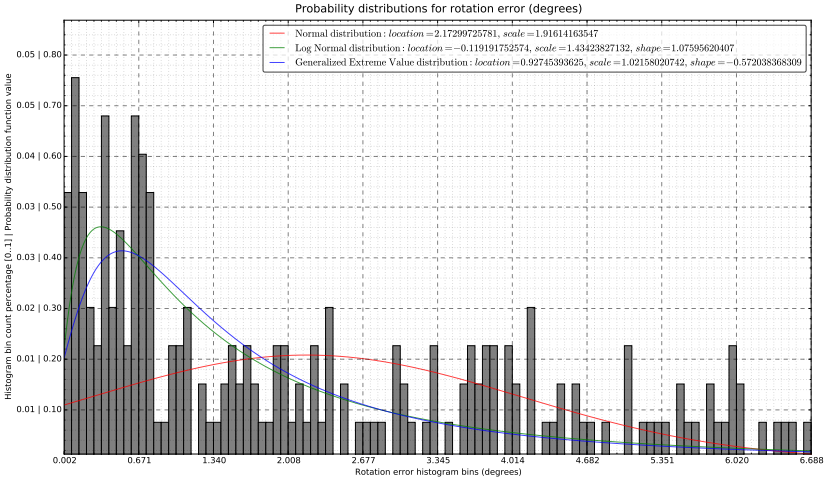


Figure A.1.2.31: Probability distributions for the odometry rotation errors

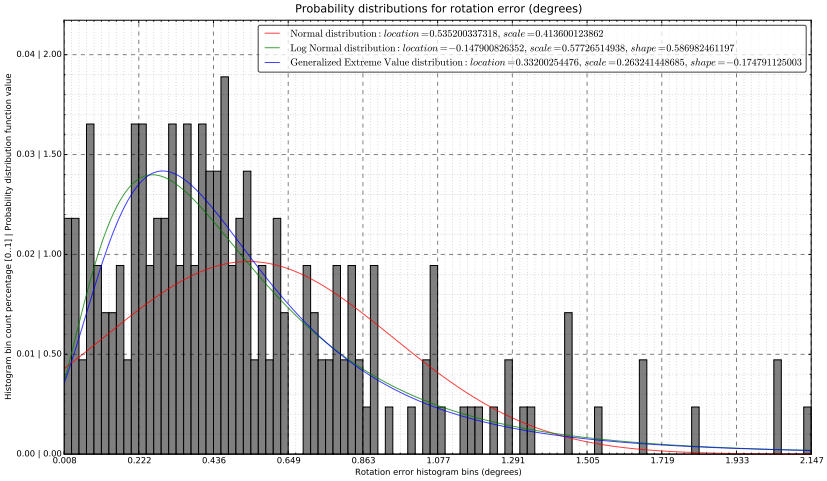


Figure A.1.2.32: Probability distributions for the localization system rotation errors

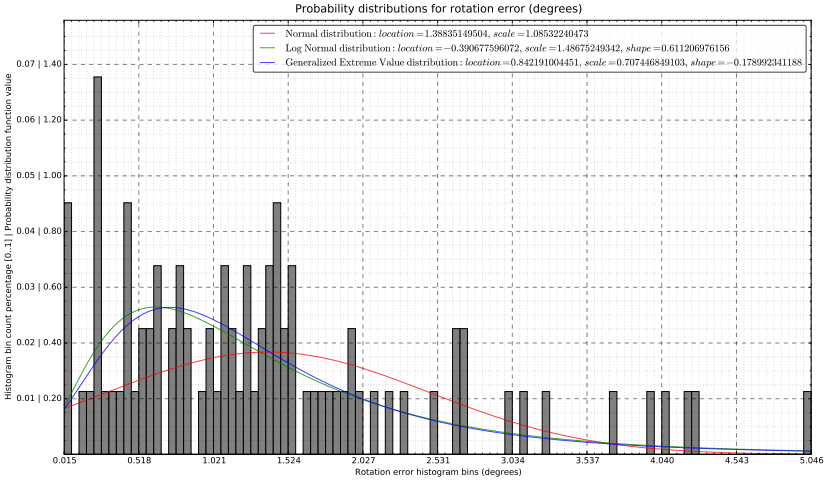


Figure A.1.2.33: Probability distributions for the AMCL rotation errors

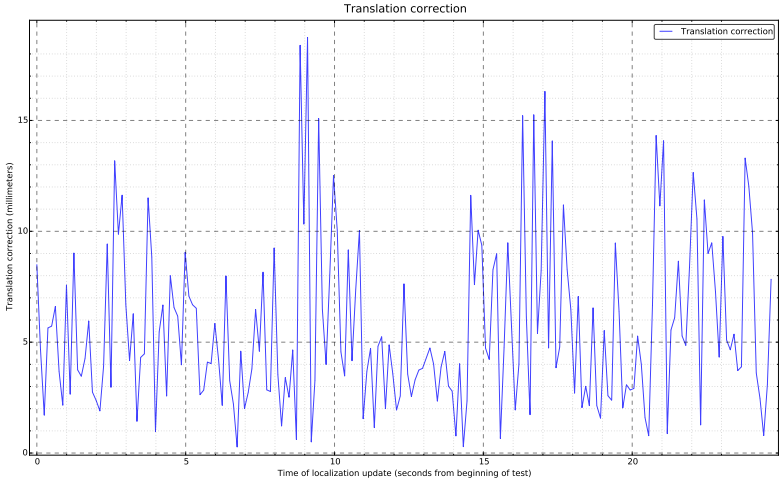


Figure A.1.2.34: Translation corrections performed by the localization system

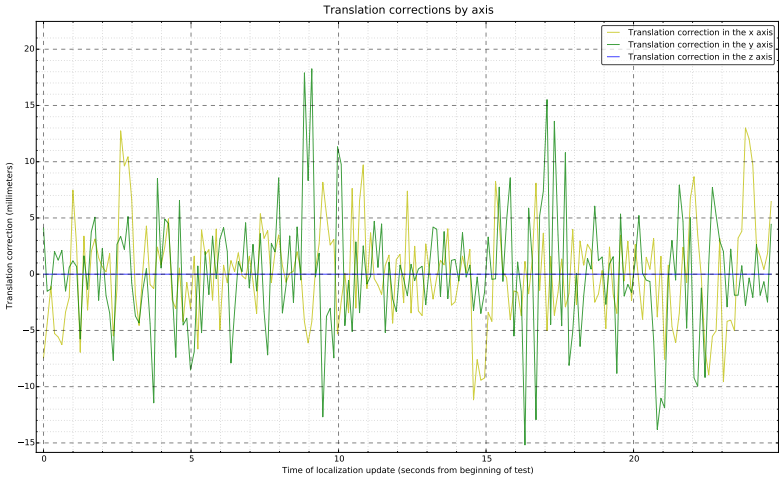


Figure A.1.2.35: Translation corrections components performed by the localization system

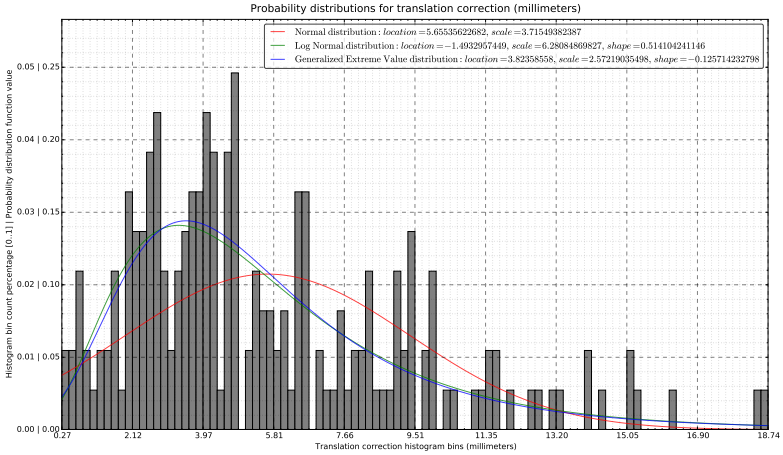


Figure A.1.2.36: Probability distributions for the translation corrections performed by the localization system

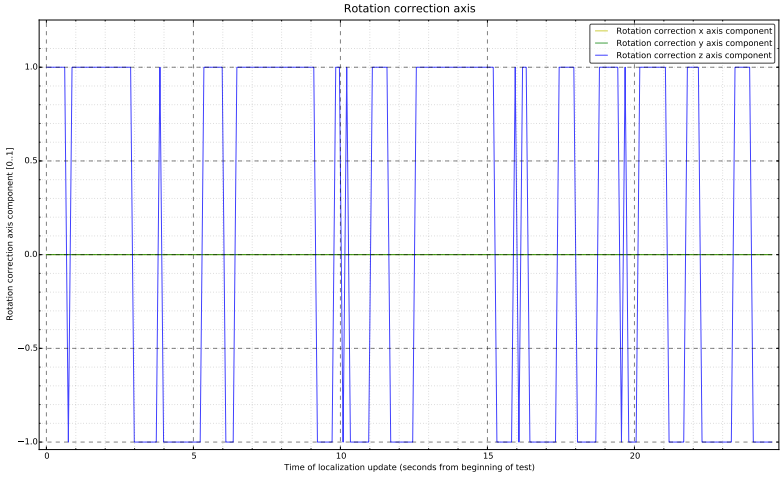


Figure A.1.2.37: Rotation corrections (axis) performed by the localization system

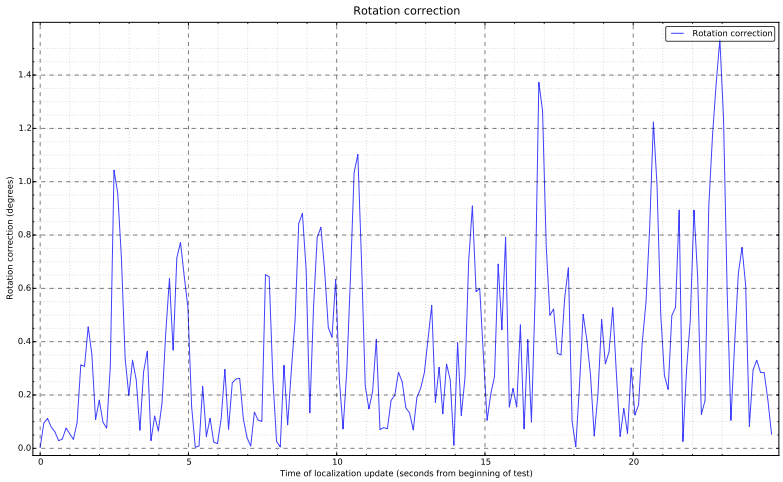


Figure A.1.2.38: Rotation corrections performed by the localization system

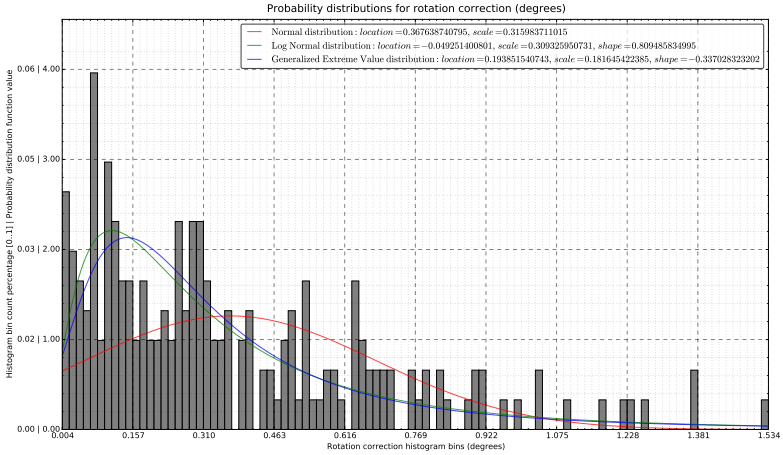


Figure A.1.2.39: Probability distributions for the rotation corrections performed by the localization system

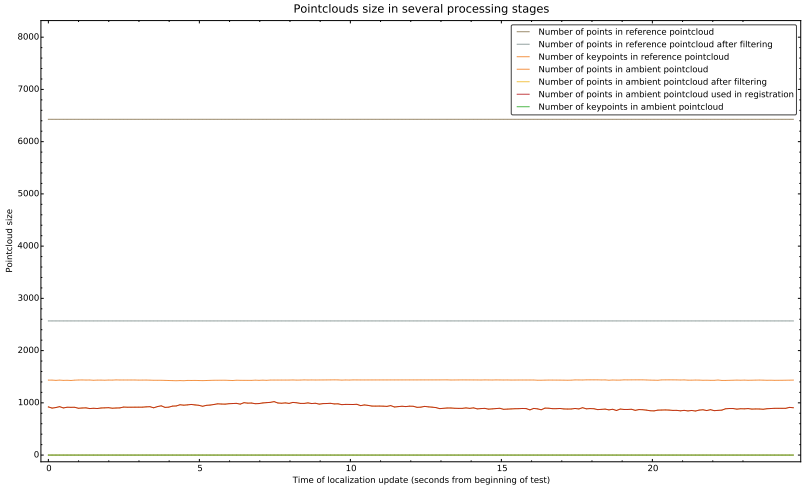


Figure A.1.2.40: Point clouds size in several of the localization system processing stages

Figure A.1.2.41: Number of registered points

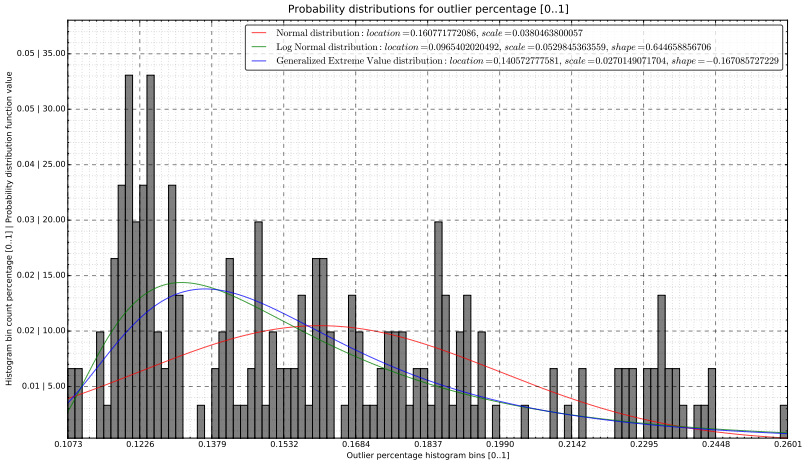


Figure A.1.2.42: Probability distributions for the ambient point cloud outlier percentage

Figure A.1.2.43: Angular distribution percentage of registered points

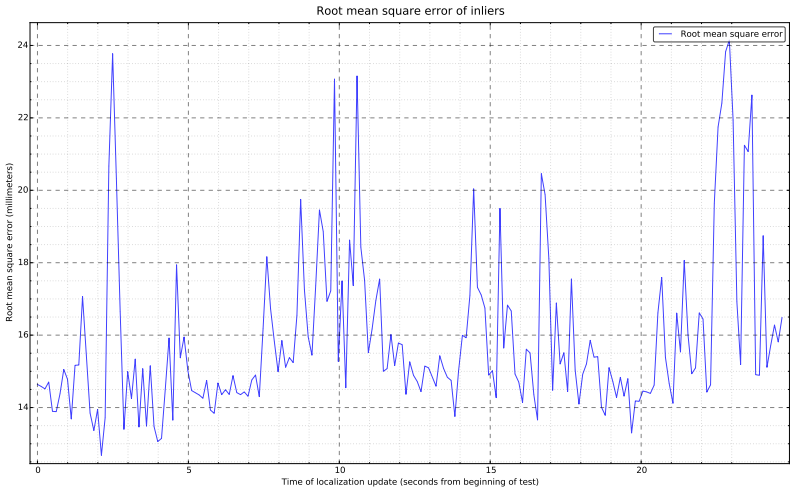


Figure A.1.2.44: Root Mean Square Error of the inliers

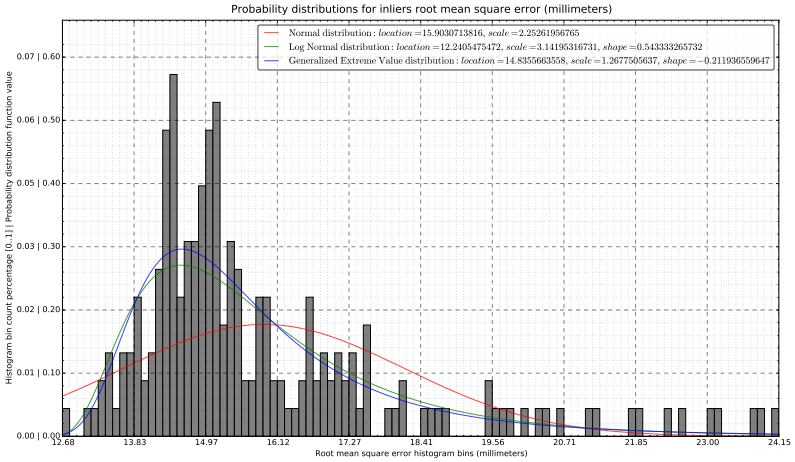


Figure A.1.2.45: Probability distributions for the Root Mean Square Error of the inliers

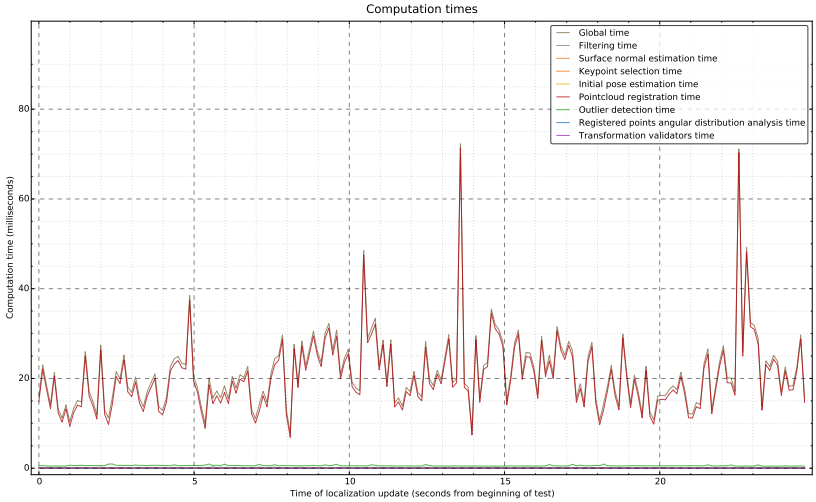


Figure A.1.2.46: Localization system computation times

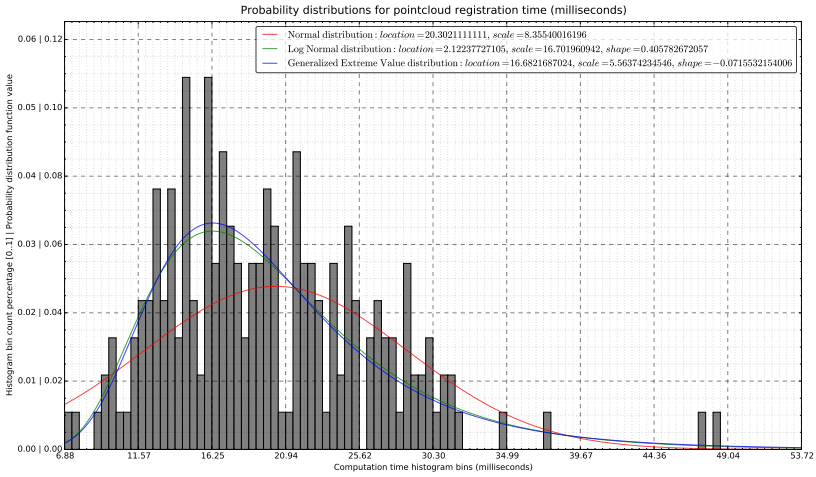


Figure A.1.2.47: Probability distributions for the point cloud registration time

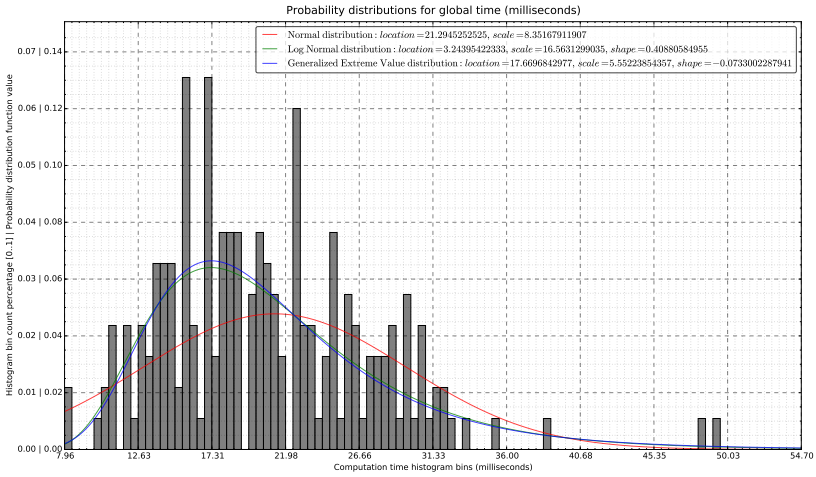


Figure A.1.2.48: Probability distributions for the localization system global computation time

A.1.3 Complex path using the Jarvis robot at 5 cm/s

The figures in this section show the detailed results of the Jarvis robot with a 10 mm resolution map in the RoboCup field following a complex path and moving at 5 cm/s.

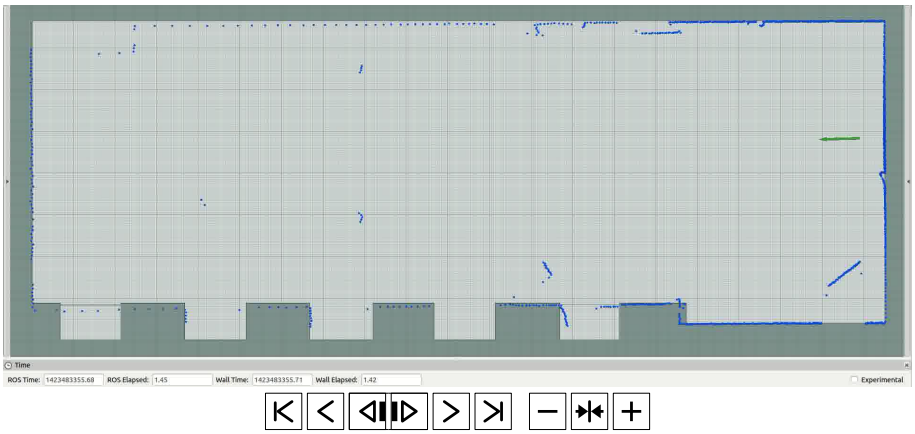


Figure A.1.3.1: Animation of laser scans using the ground truth poses

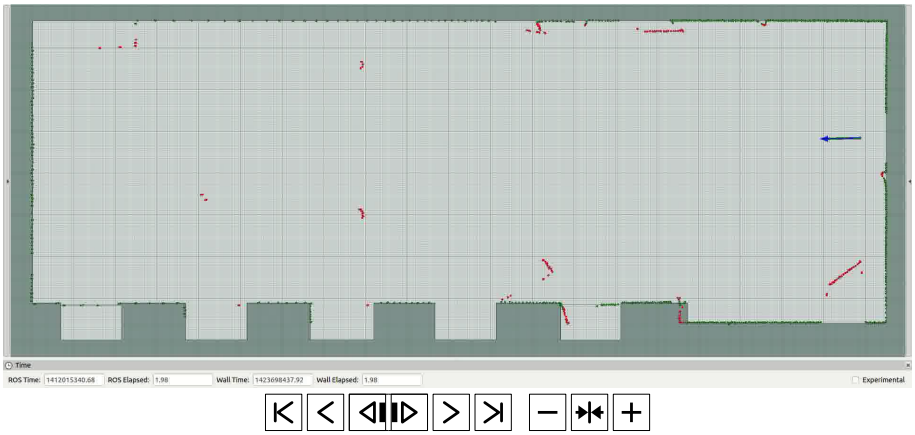


Figure A.1.3.2: Animation of laser scans using the localization system poses

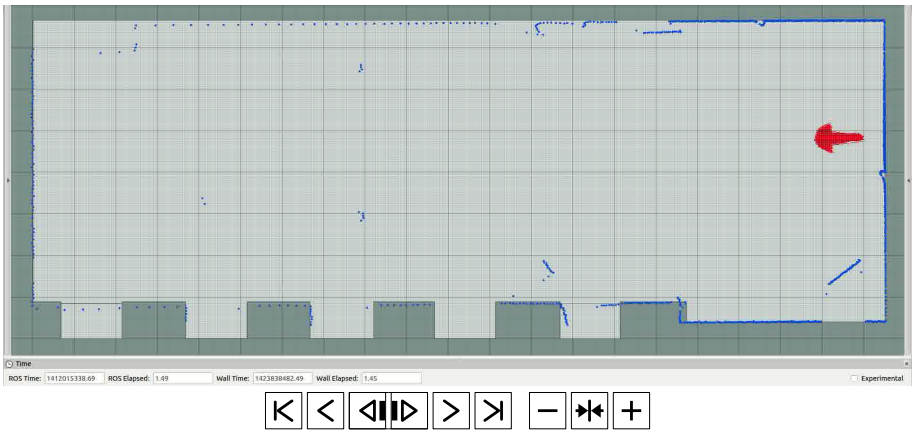


Figure A.1.3.3: Animation of laser scans using the AMCL poses

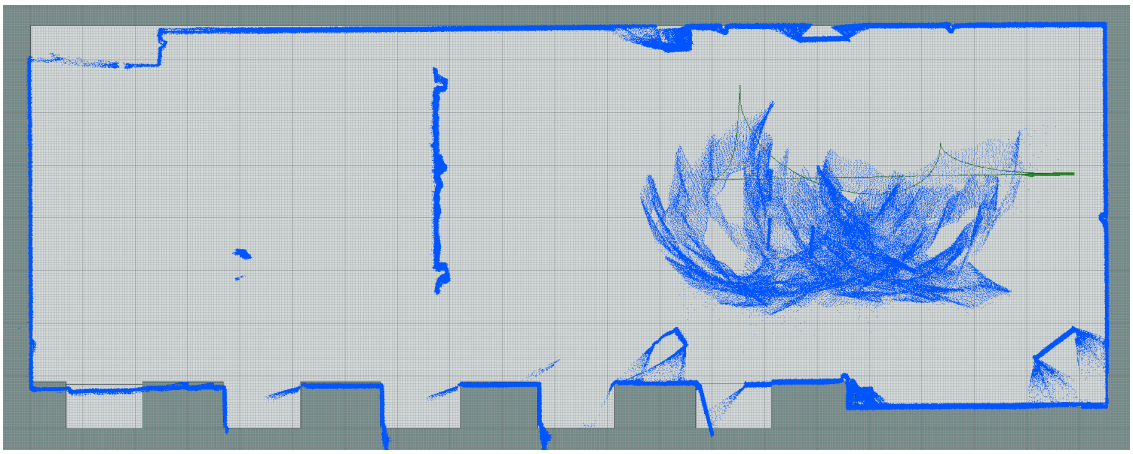


Figure A.1.3.4: Laser scans assembled on top of the map using the ground truth poses

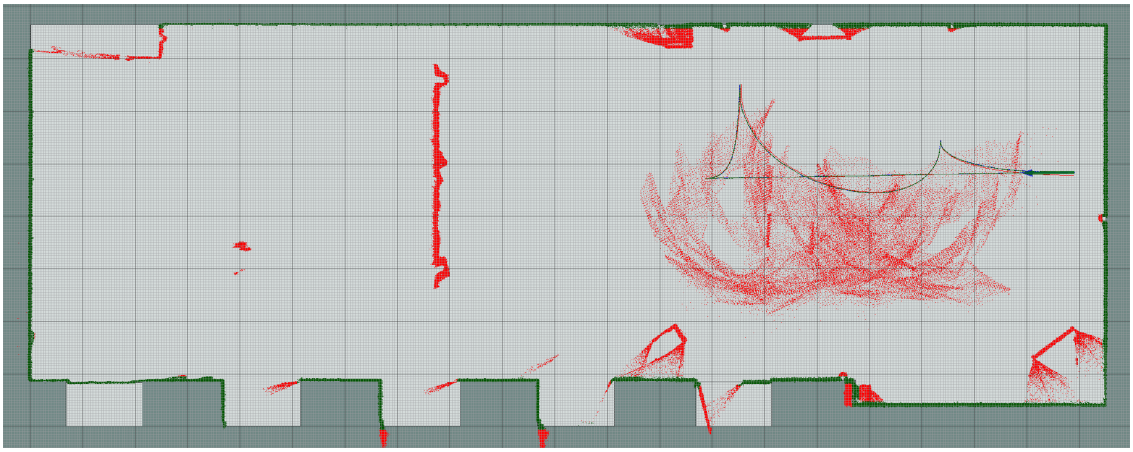


Figure A.1.3.5: Laser scans assembled on top of the map using the localization system poses

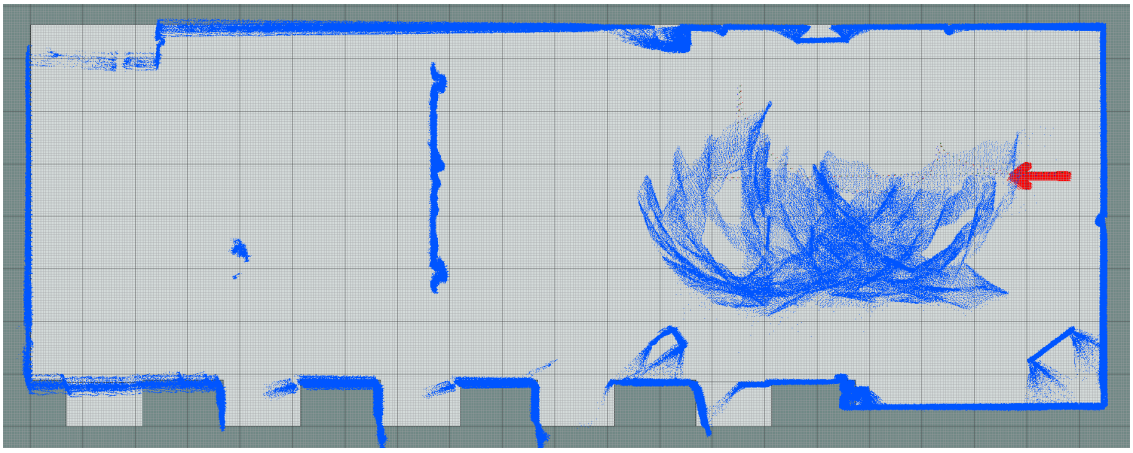


Figure A.1.3.6: Laser scans assembled on top of the map using the AMCL poses

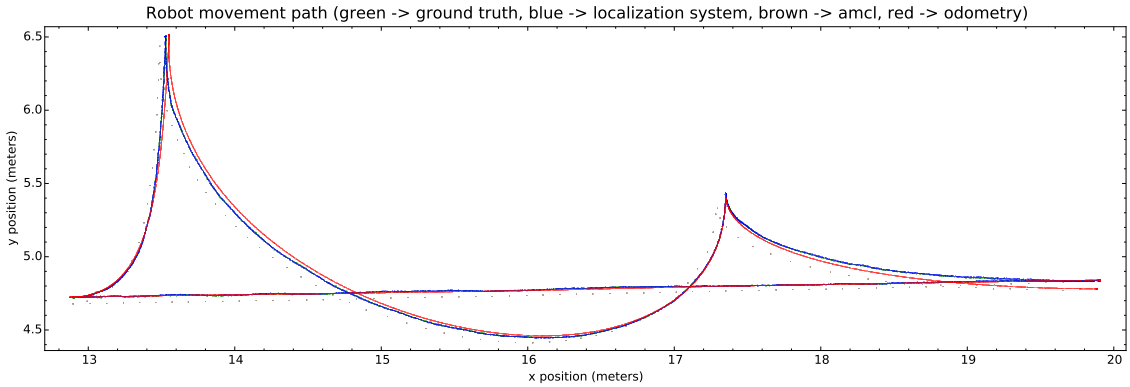


Figure A.1.3.7: Poses estimated by the ground truth, localization system, AMCL and odometry

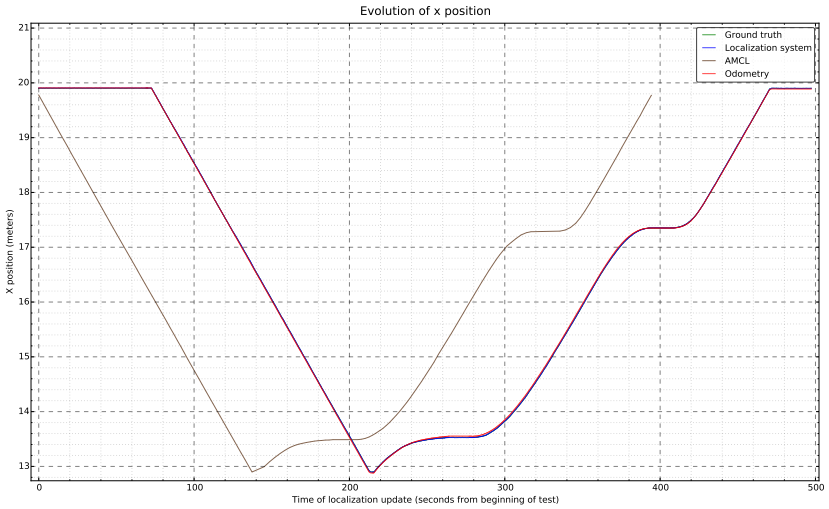


Figure A.1.3.8: Evolution of x position over time

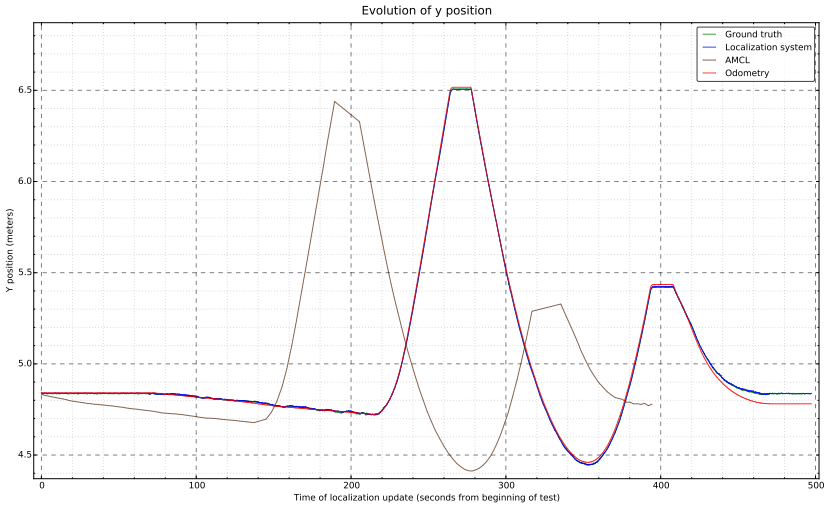


Figure A.1.3.9: Evolution of y position over time

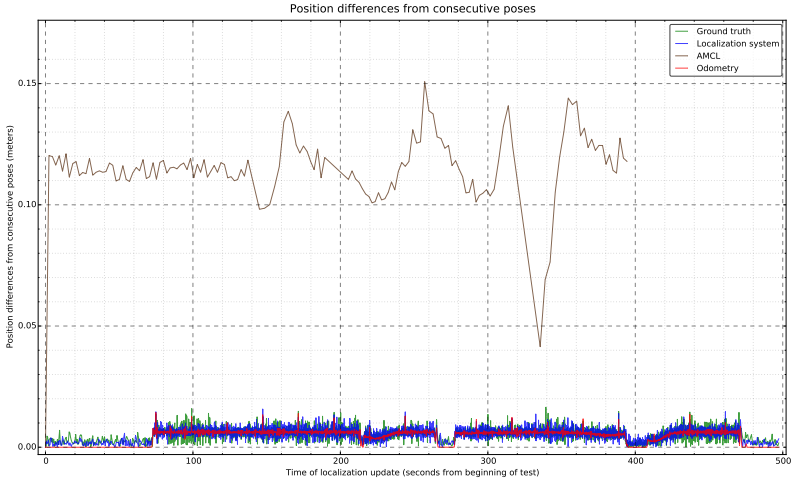


Figure A.1.3.10: Distance traveled between consecutive pose estimations

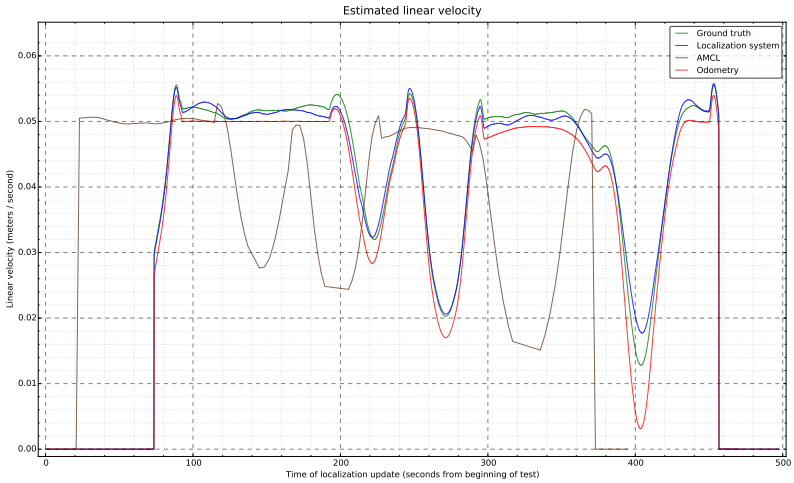


Figure A.1.3.11: Estimated linear velocity

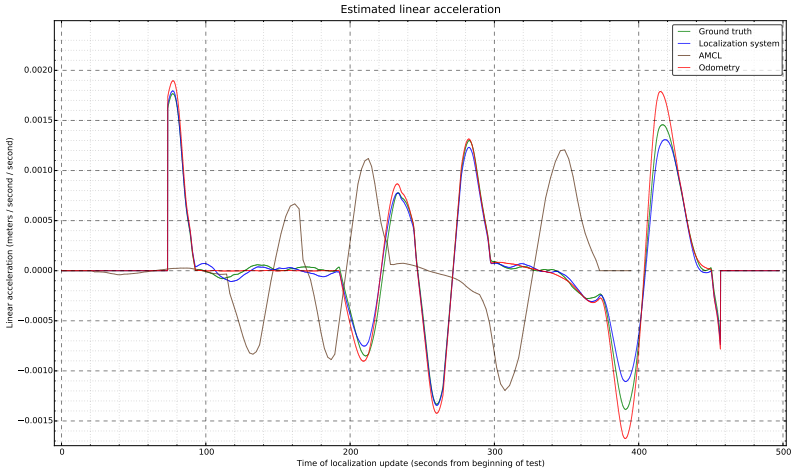


Figure A.1.3.12: Estimated linear acceleration

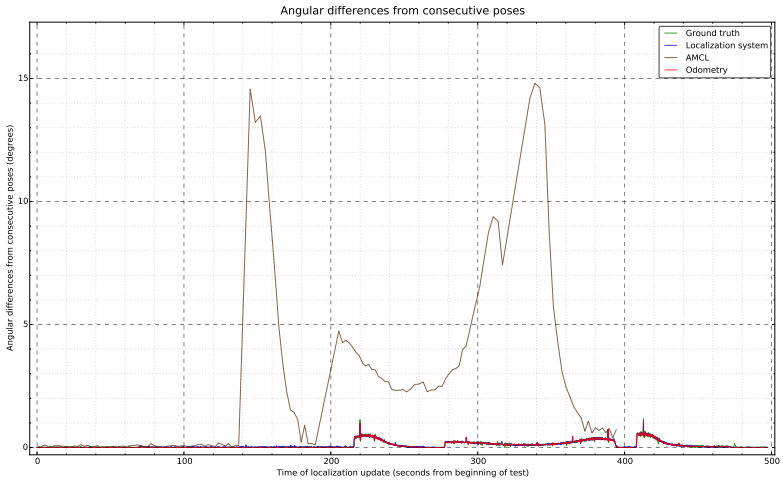


Figure A.1.3.13: Angular differences between consecutive pose estimations

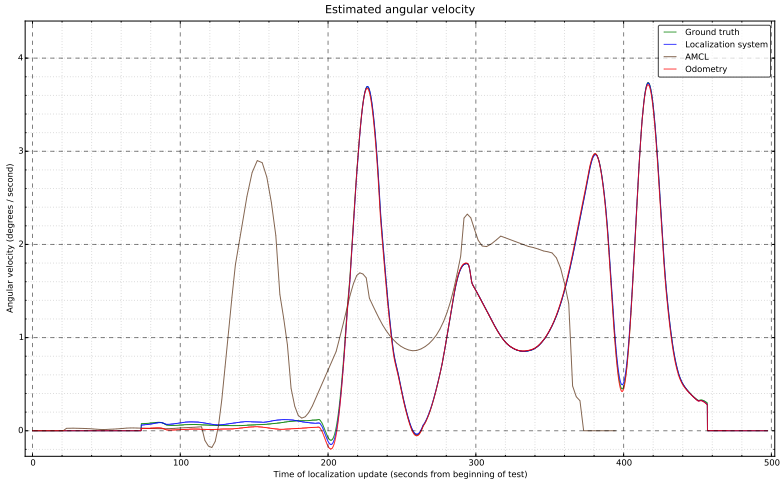


Figure A.1.3.14: Estimated angular velocity

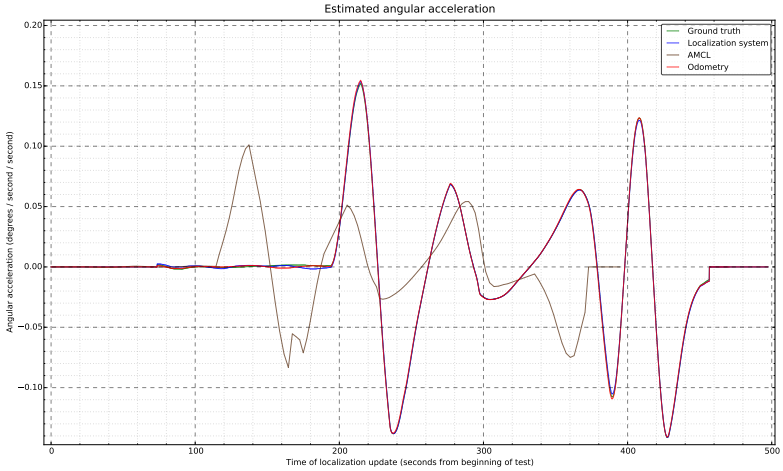


Figure A.1.3.15: Estimated angular acceleration

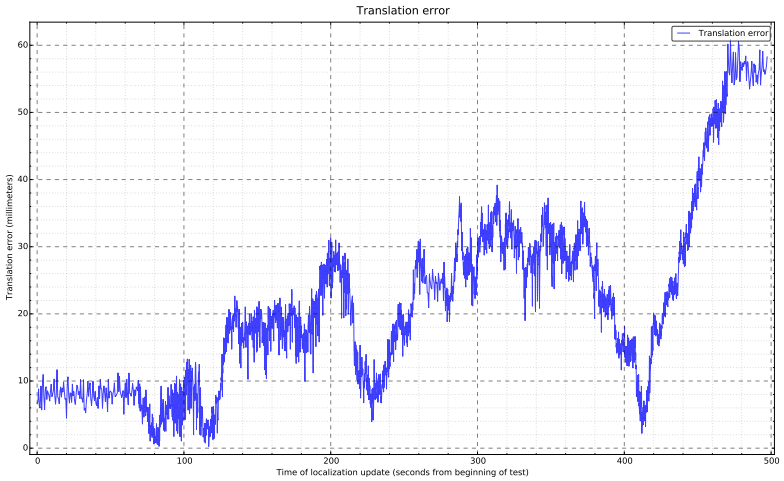


Figure A.1.3.16: Odometry translation errors

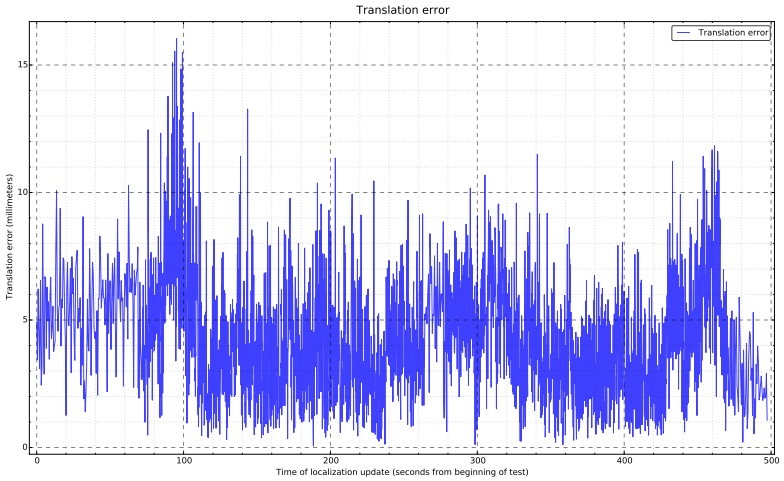


Figure A.1.3.17: Localization system translation errors

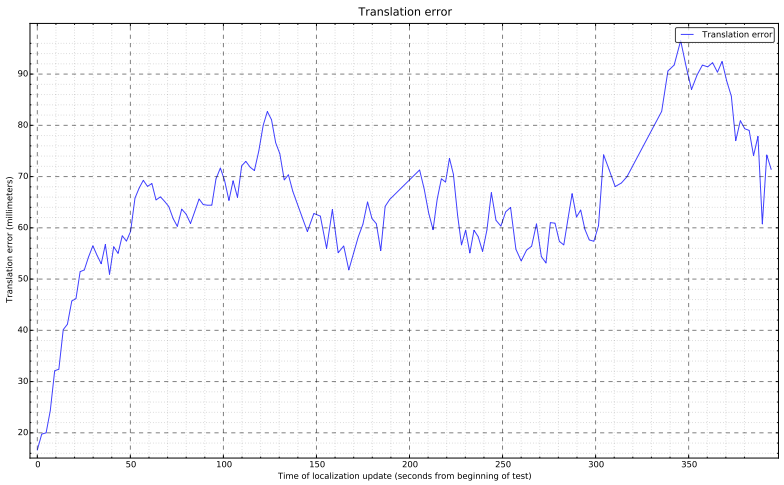


Figure A.1.3.18: AMCL translation errors

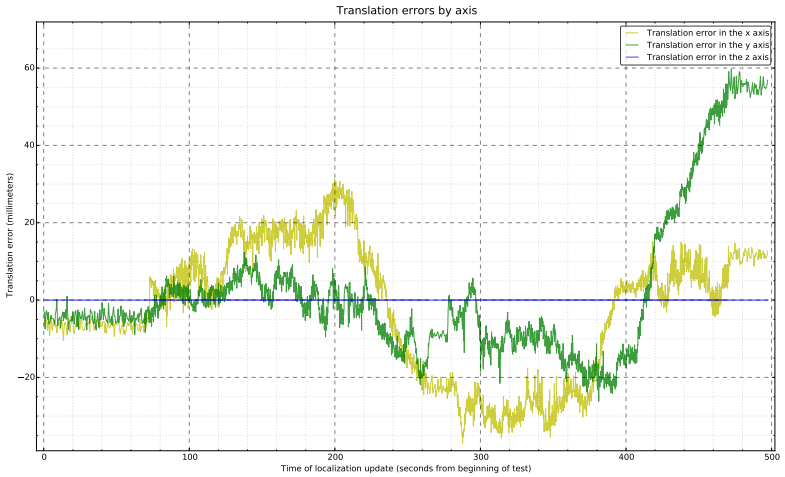


Figure A.1.3.19: Odometry translation errors components

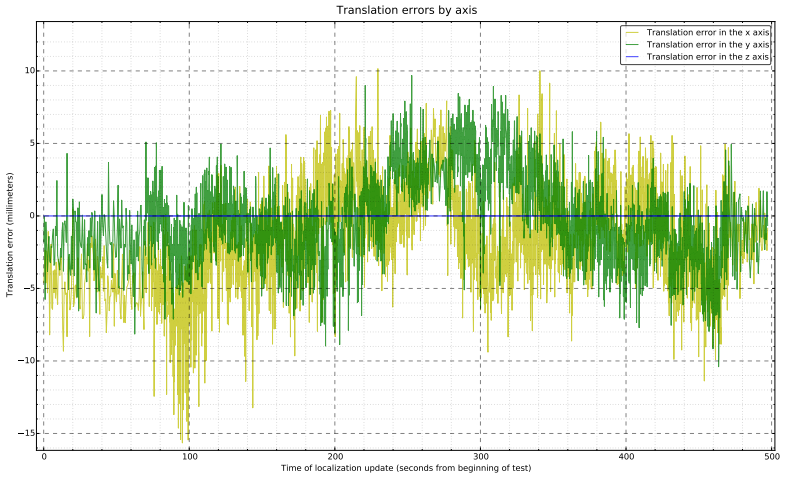


Figure A.1.3.20: Localization system translation errors components

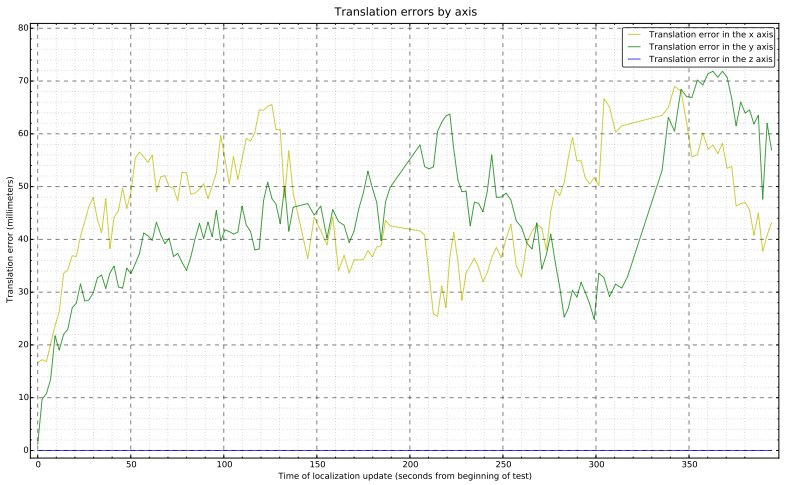


Figure A.1.3.21: AMCL translation errors components

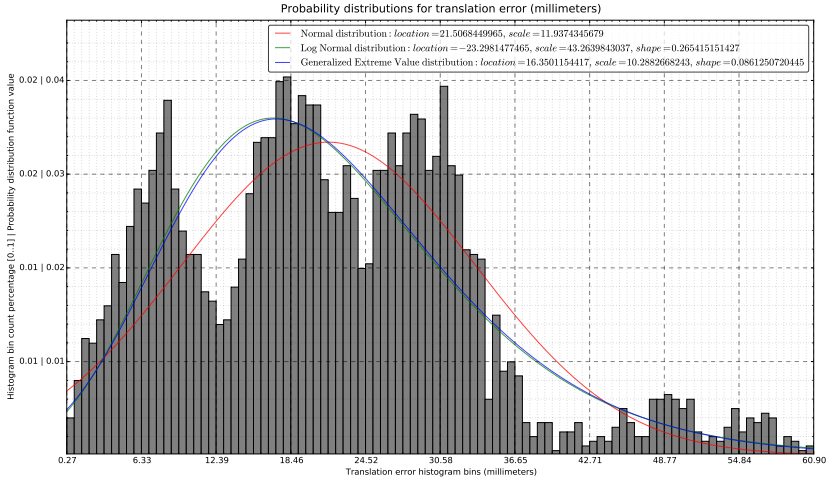


Figure A.1.3.22: Probability distributions for the odometry translation errors

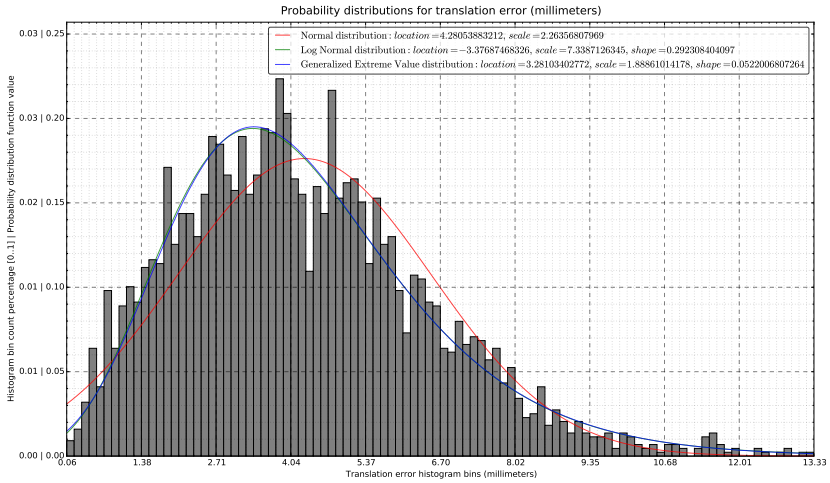


Figure A.1.3.23: Probability distributions for the localization system translation errors

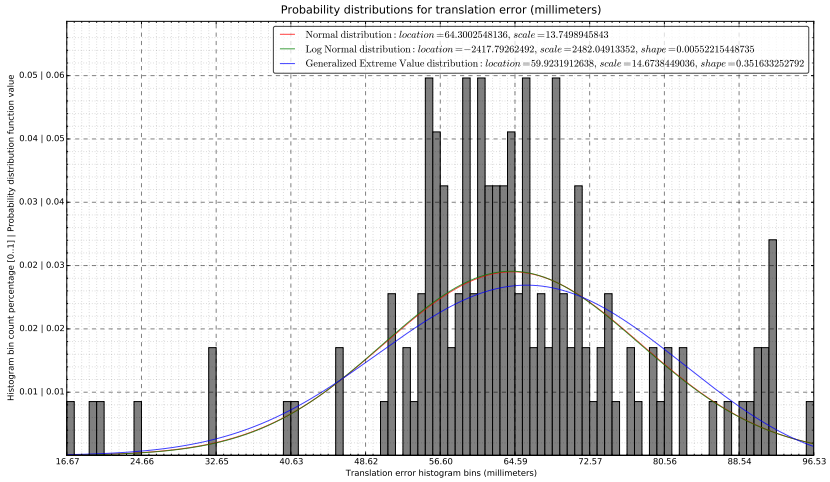


Figure A.1.3.24: Probability distributions for the AMCL translation errors

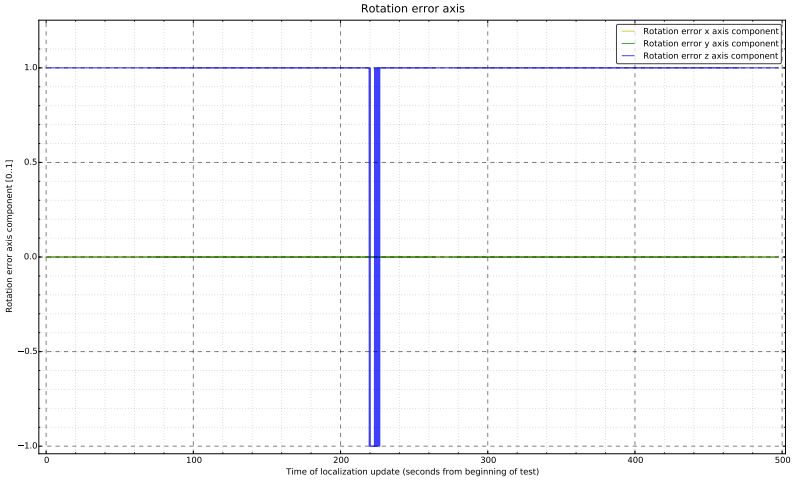


Figure A.1.3.25: Odometry rotation errors axis

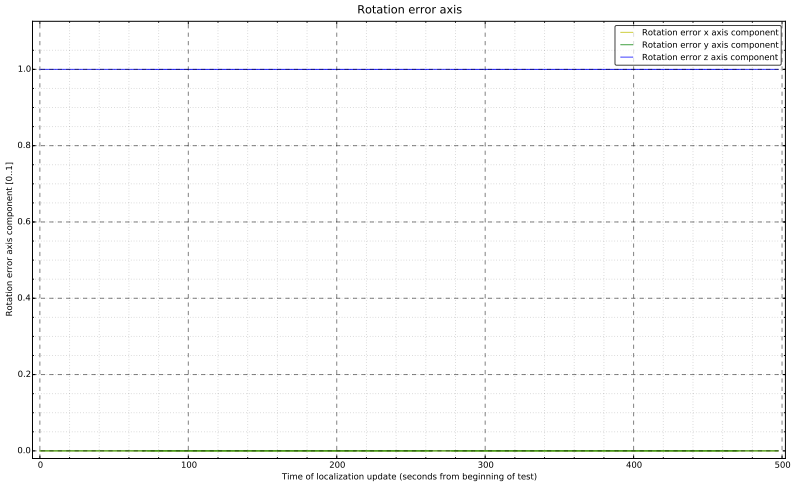


Figure A.1.3.26: Localization system rotation errors axis

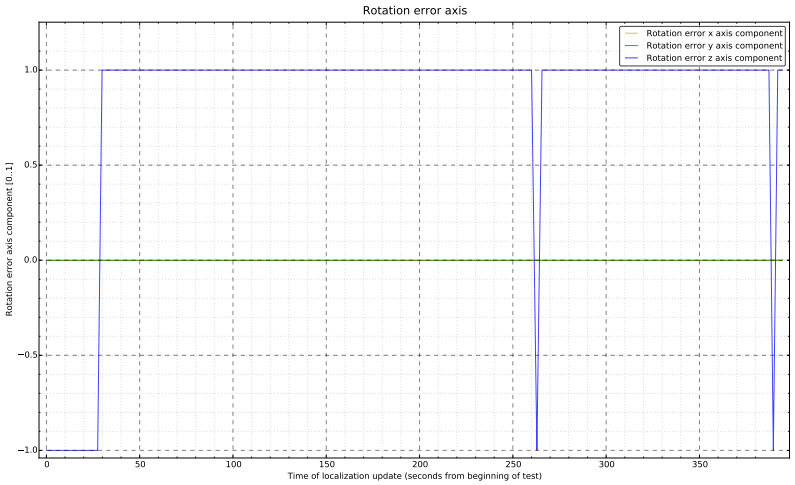


Figure A.1.3.27: AMCL rotation errors axis

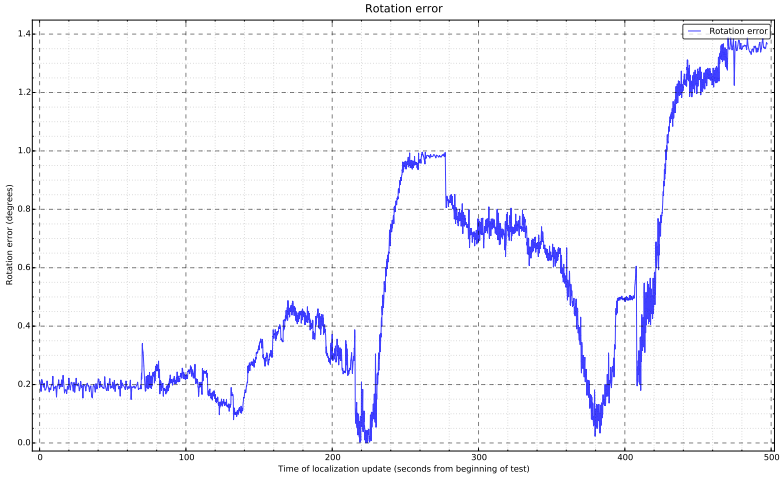


Figure A.1.3.28: Odometry rotation errors

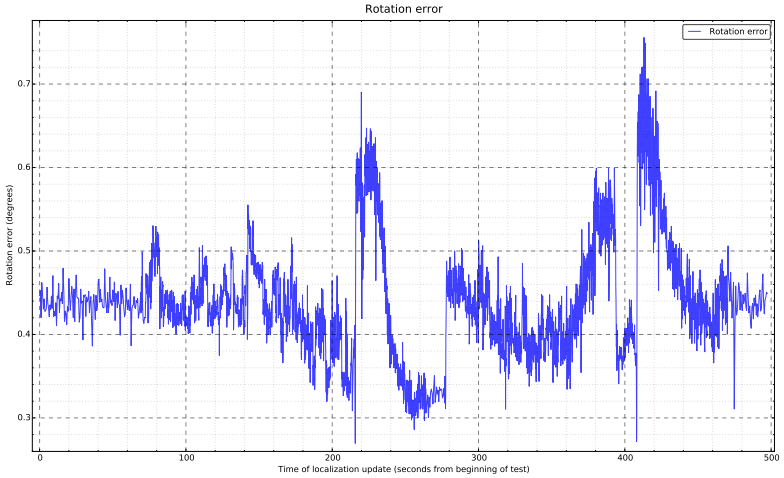


Figure A.1.3.29: Localization system rotation errors

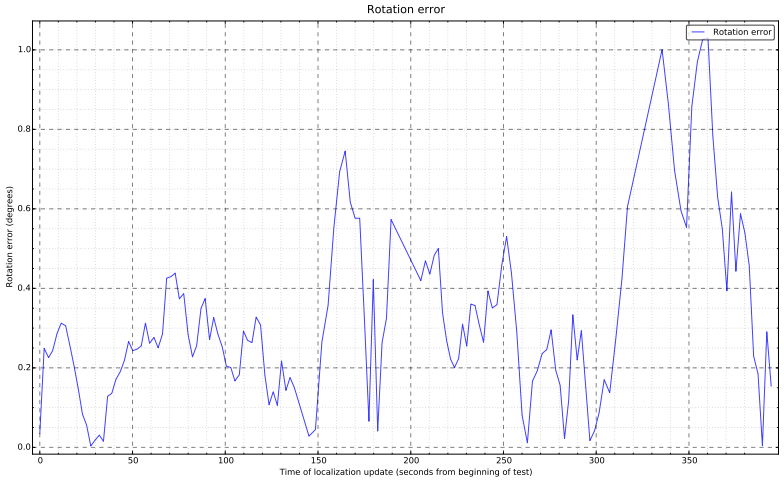


Figure A.1.3.30: AMCL rotation errors

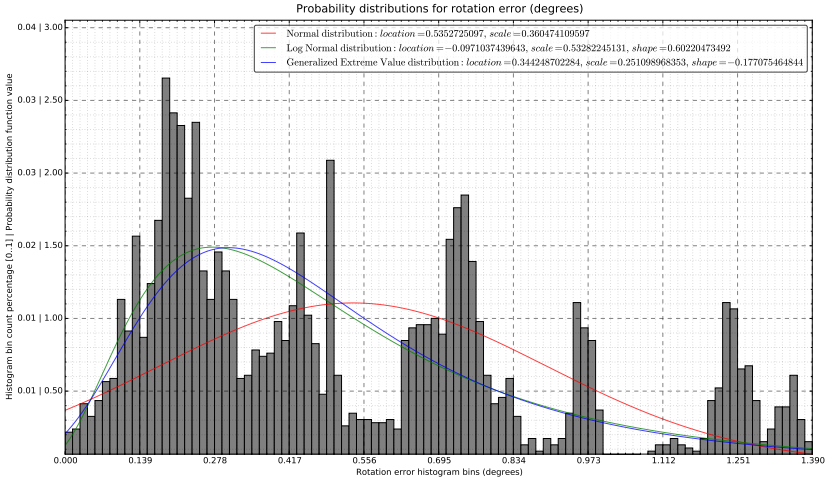


Figure A.1.3.31: Probability distributions for the odometry rotation errors

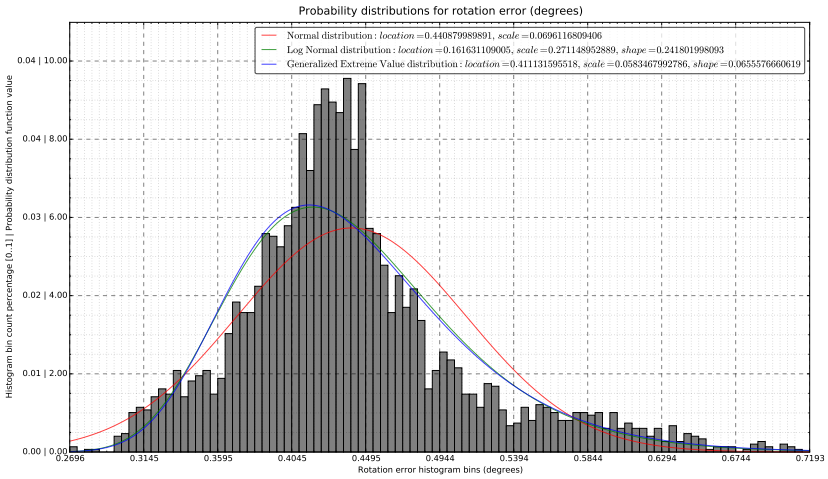


Figure A.1.3.32: Probability distributions for the localization system rotation errors

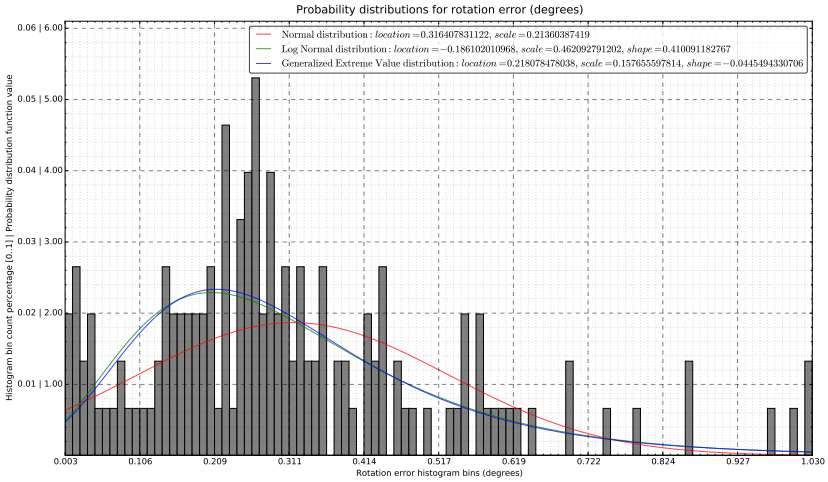


Figure A.1.3.33: Probability distributions for the AMCL rotation errors

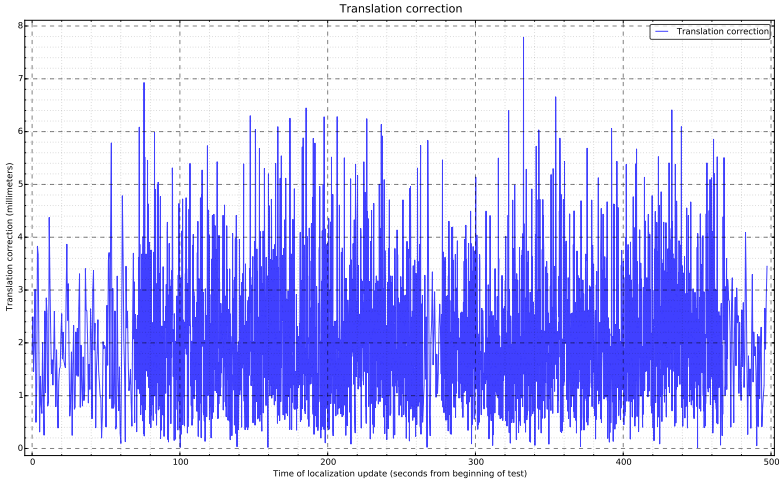


Figure A.1.3.34: Translation corrections performed by the localization system

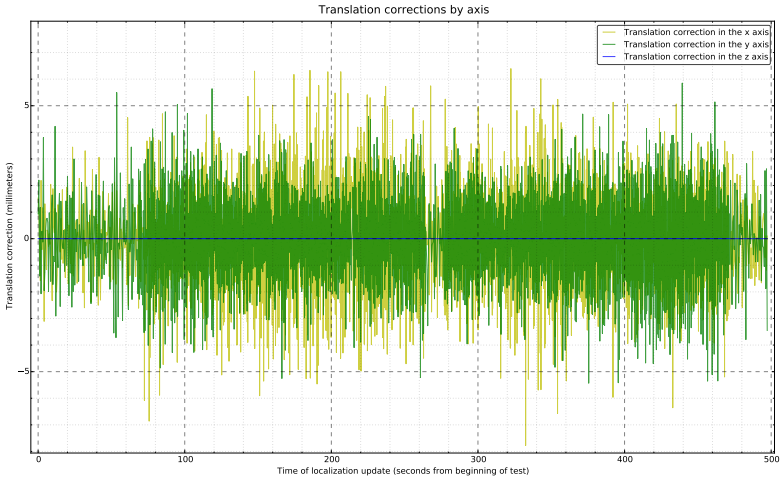


Figure A.1.3.35: Translation corrections components performed by the localization system

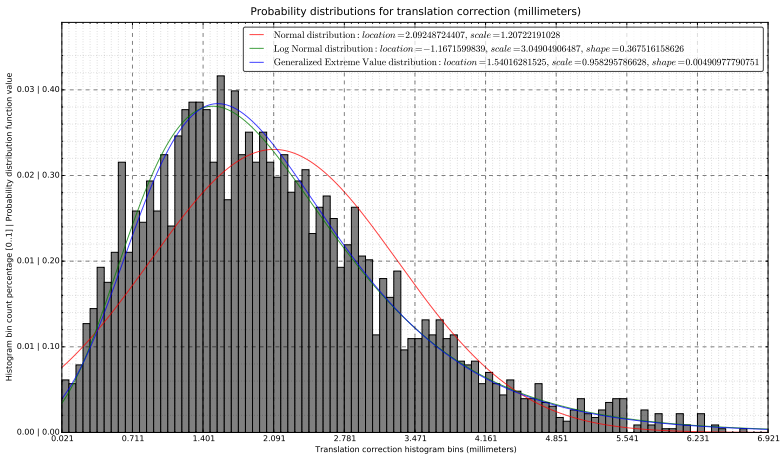


Figure A.1.3.36: Probability distributions for the translation corrections performed by the localization system

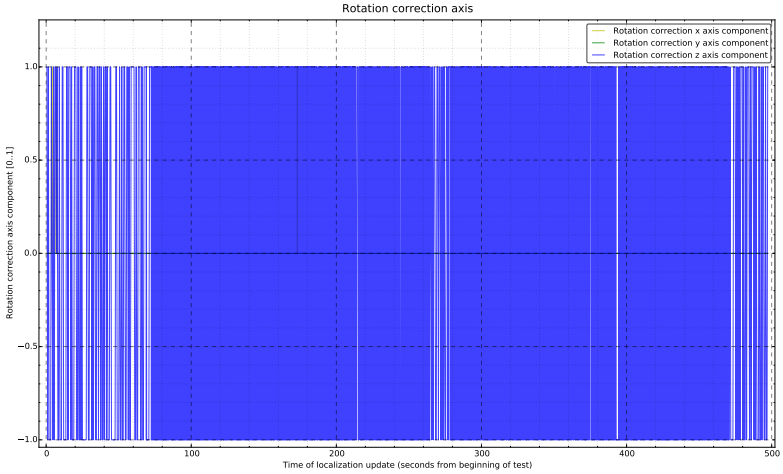


Figure A.1.3.37: Rotation corrections (axis) performed by the localization system

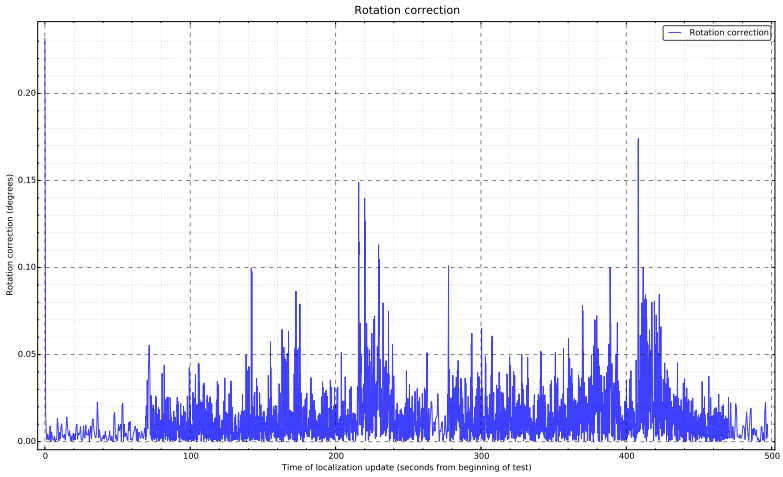


Figure A.1.3.38: Rotation corrections performed by the localization system

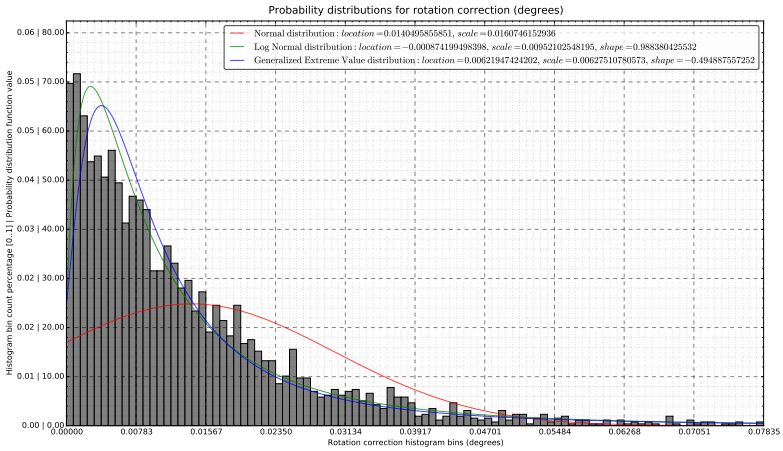


Figure A.1.3.39: Probability distributions for the rotation corrections performed by the localization system

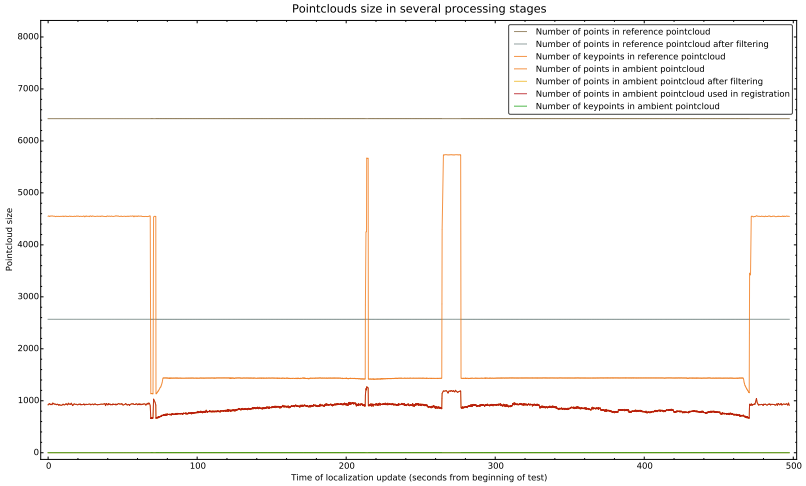


Figure A.1.3.40: Point clouds size in several of the localization system processing stages

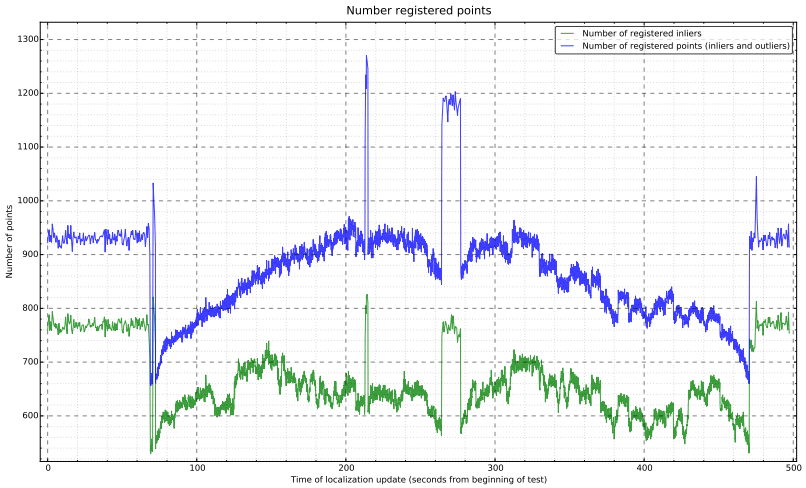


Figure A.1.3.41: Number of registered points

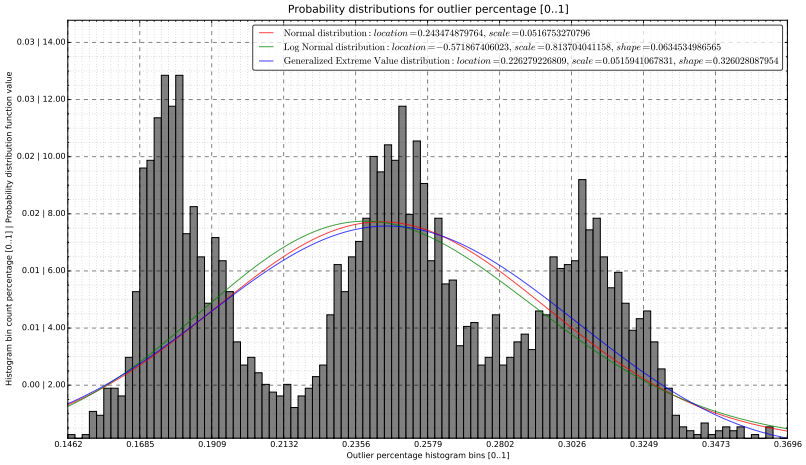


Figure A.1.3.42: Probability distributions for the ambient point cloud outlier percentage

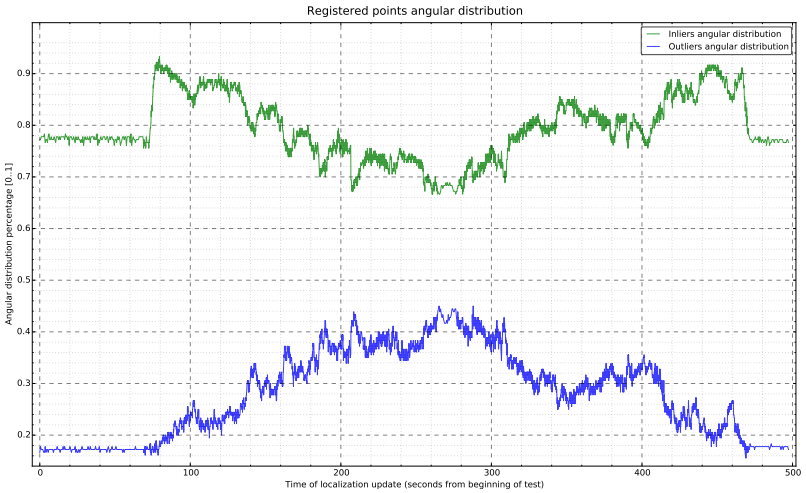


Figure A.1.3.43: Angular distribution percentage of registered points

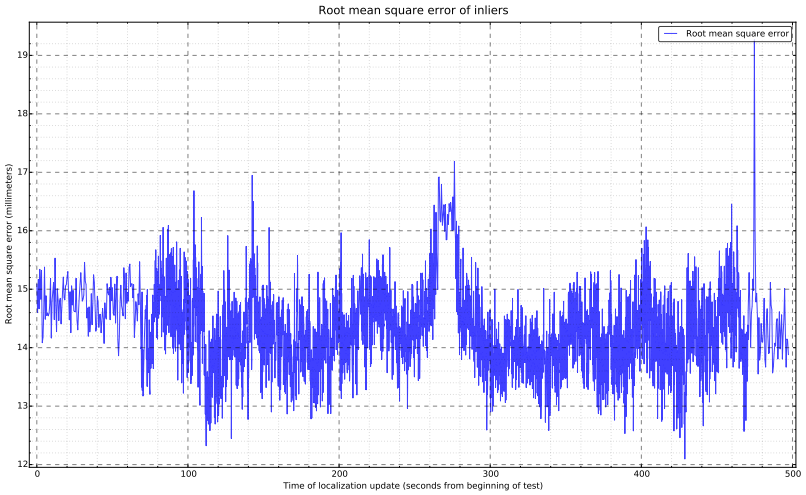


Figure A.1.3.44: Root Mean Square Error of the inliers

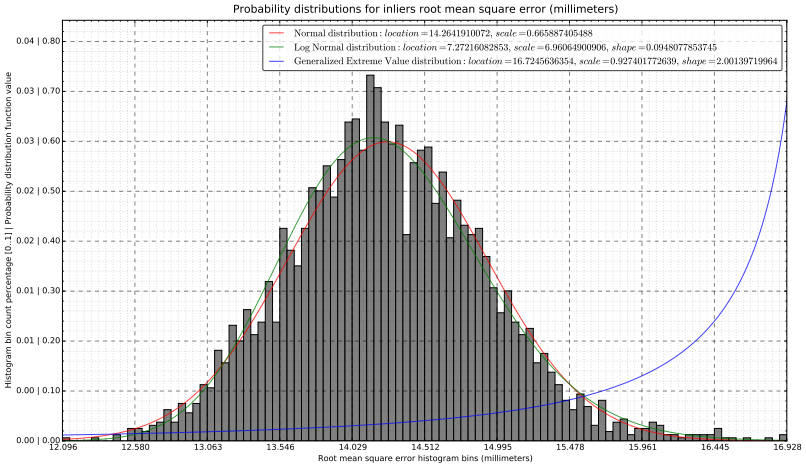


Figure A.1.3.45: Probability distributions for the Root Mean Square Error of the inliers

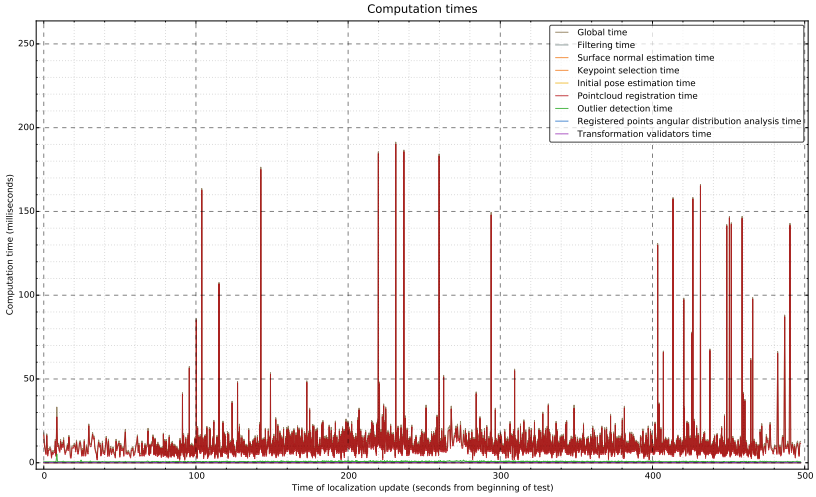


Figure A.1.3.46: Localization system computation times

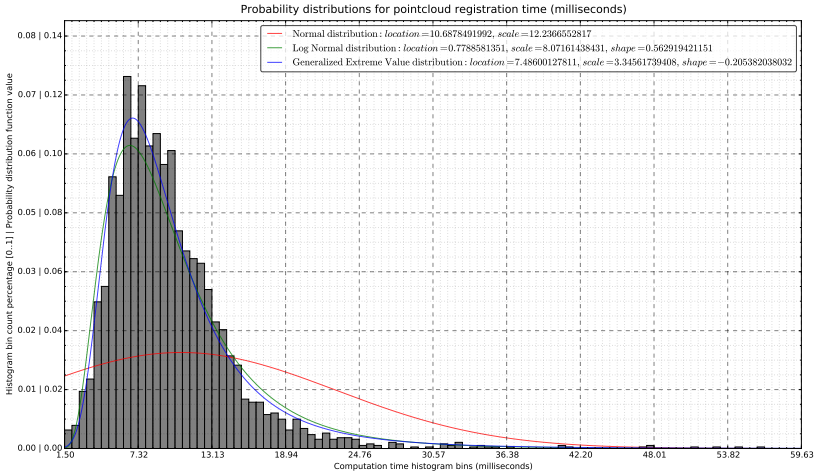


Figure A.1.3.47: Probability distributions for the point cloud registration time

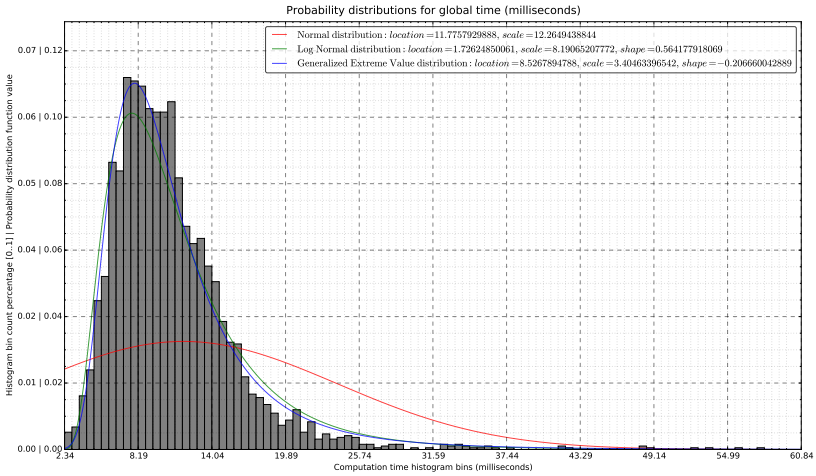


Figure A.1.3.48: Probability distributions for the localization system global computation time

A.1.4 Complex path using the Jarvis robot at 50-30-50-10 cm/s

The figures in this section show the detailed results of the Jarvis robot with a 10 mm resolution map in the RoboCup field following a complex path and moving at 50-30-50-10 cm/s.

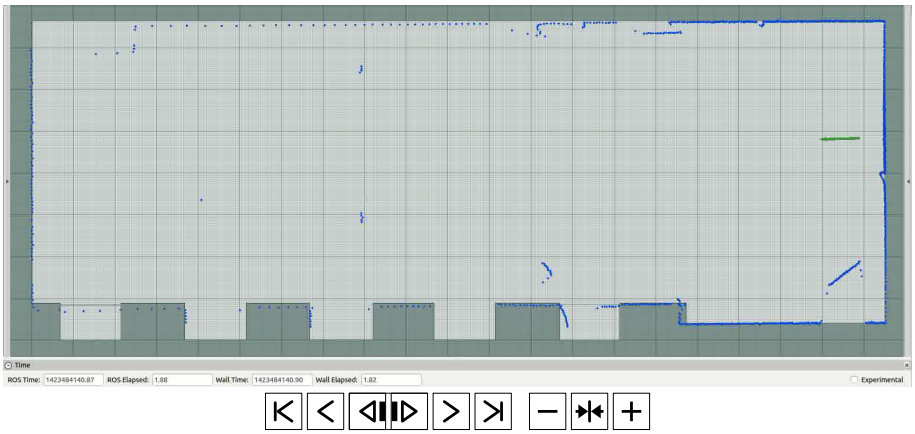


Figure A.1.4.1: Animation of laser scans using the ground truth poses

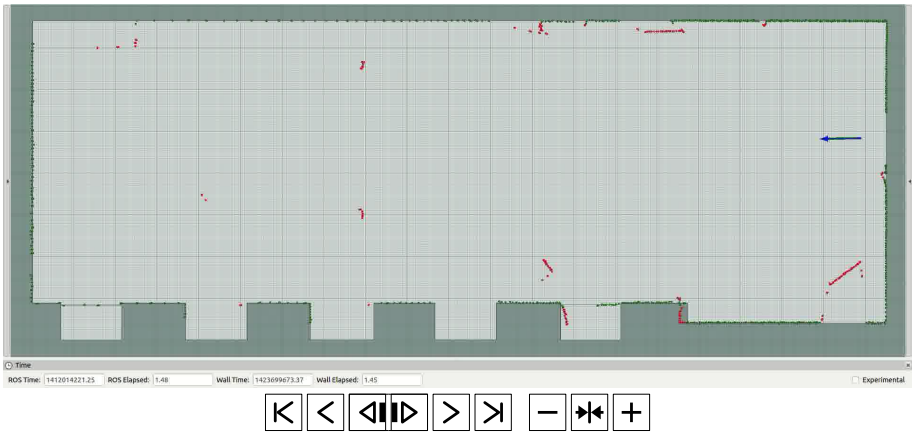


Figure A.1.4.2: Animation of laser scans using the localization system poses

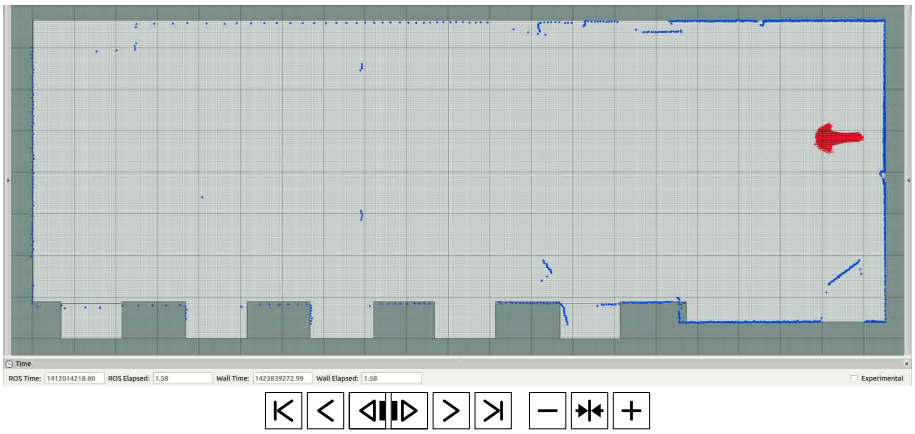


Figure A.1.4.3: Animation of laser scans using the AMCL poses

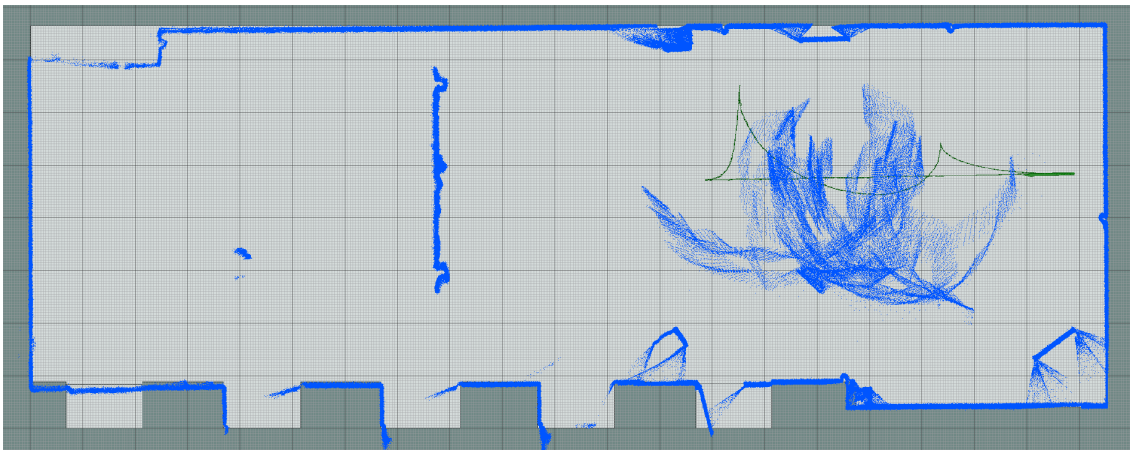


Figure A.1.4.4: Laser scans assembled on top of the map using the ground truth poses

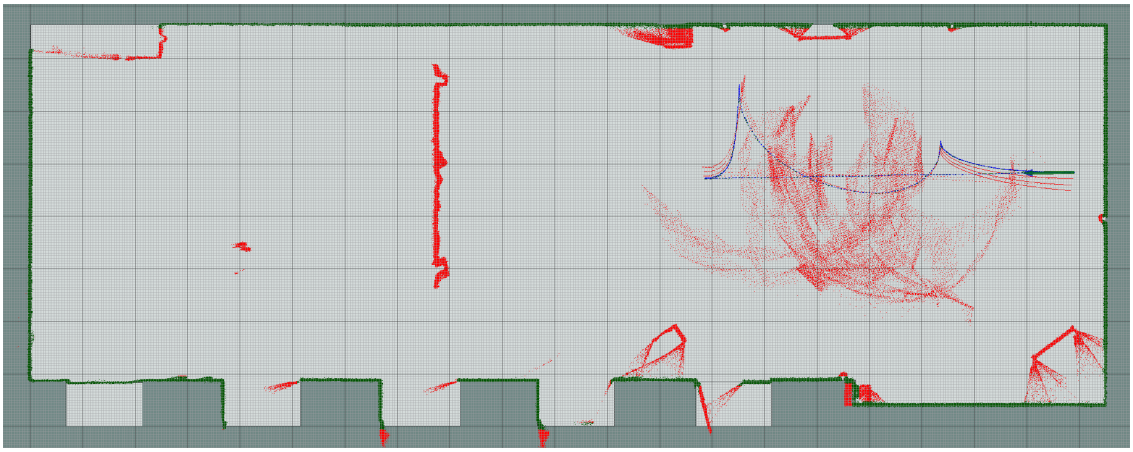


Figure A.1.4.5: Laser scans assembled on top of the map using the localization system poses

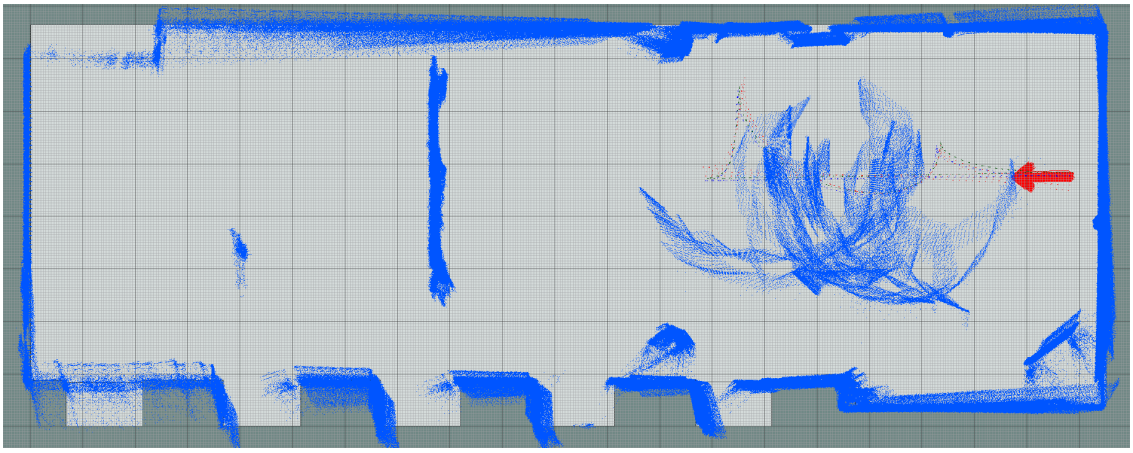


Figure A.1.4.6: Laser scans assembled on top of the map using the AMCL poses

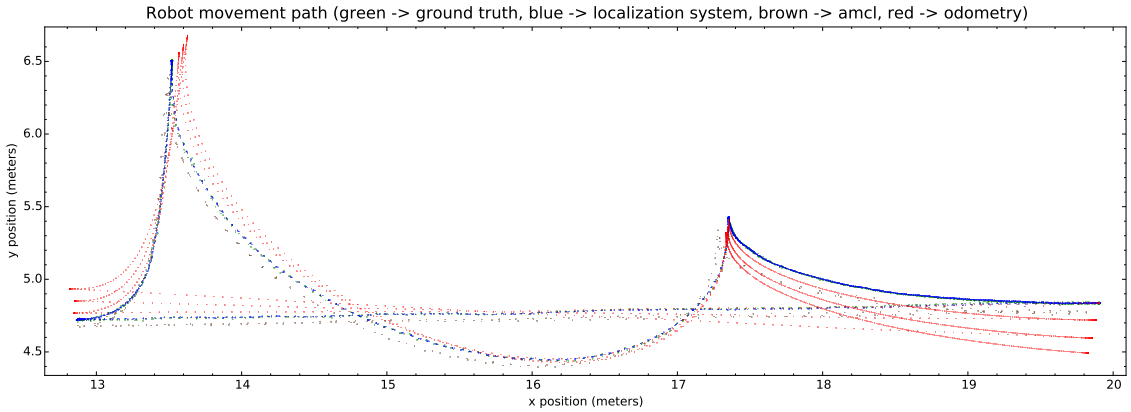


Figure A.1.4.7: Poses estimated by the ground truth, localization system, AMCL and odometry

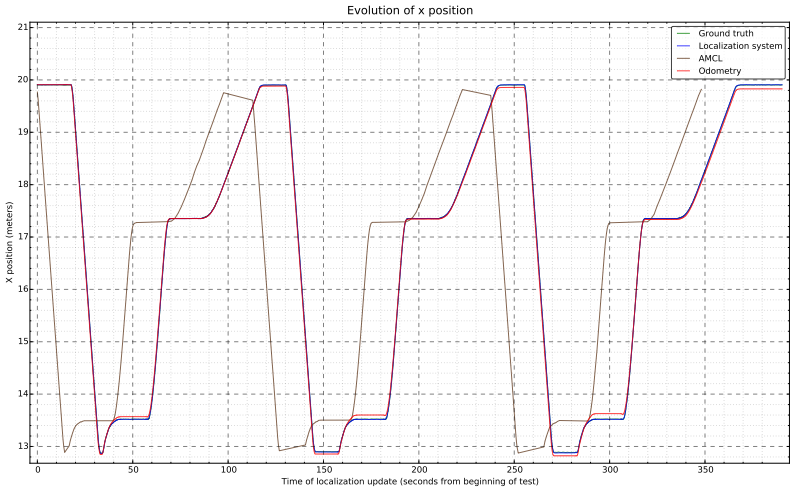


Figure A.1.4.8: Evolution of x position over time

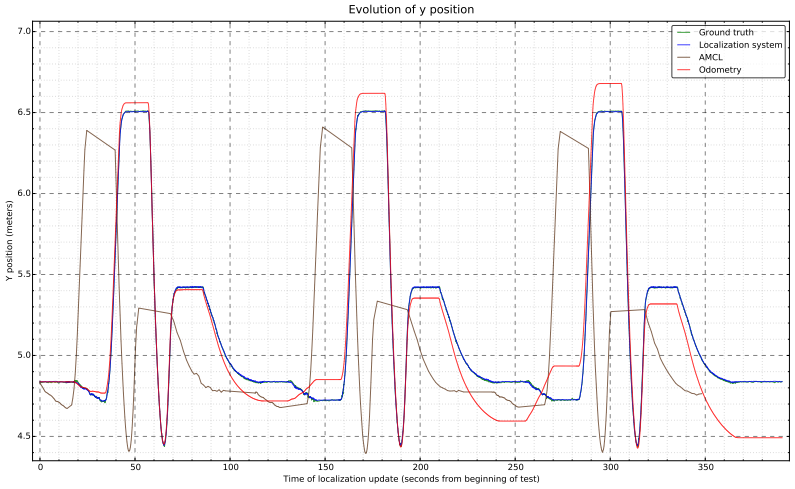


Figure A.1.4.9: Evolution of y position over time

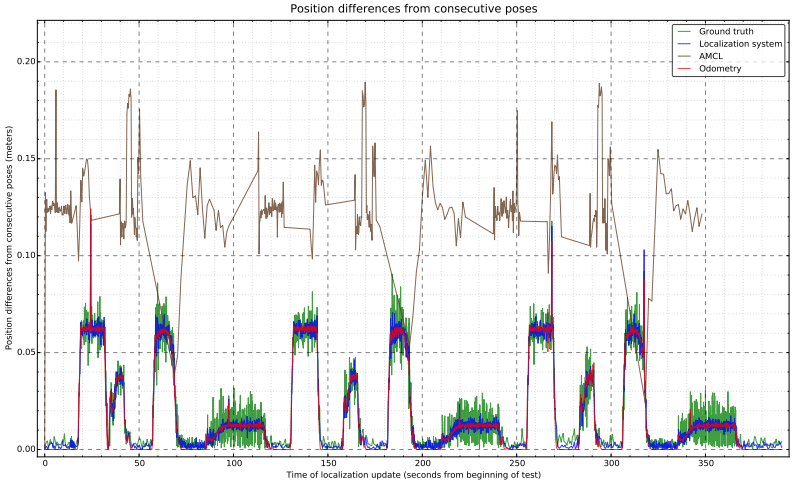


Figure A.1.4.10: Distance traveled between consecutive pose estimations

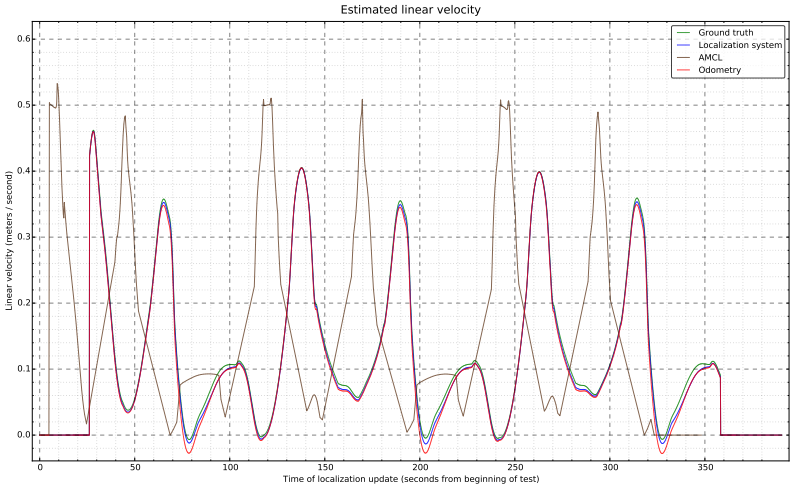


Figure A.1.4.11: Estimated linear velocity

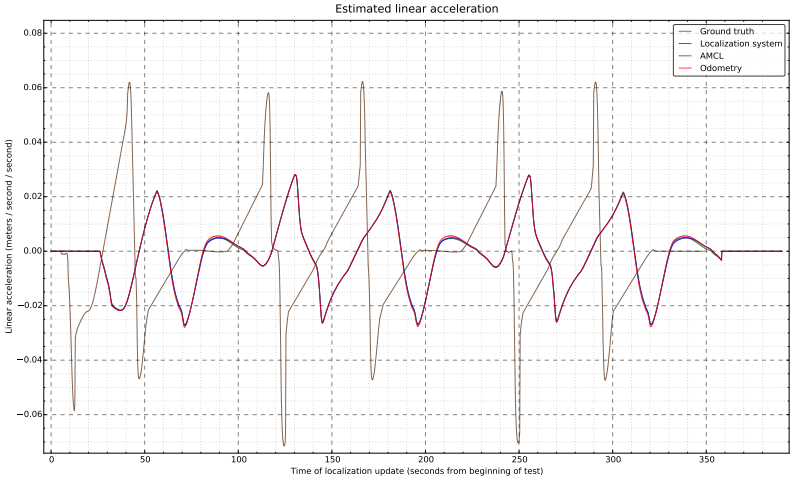


Figure A.1.4.12: Estimated linear acceleration

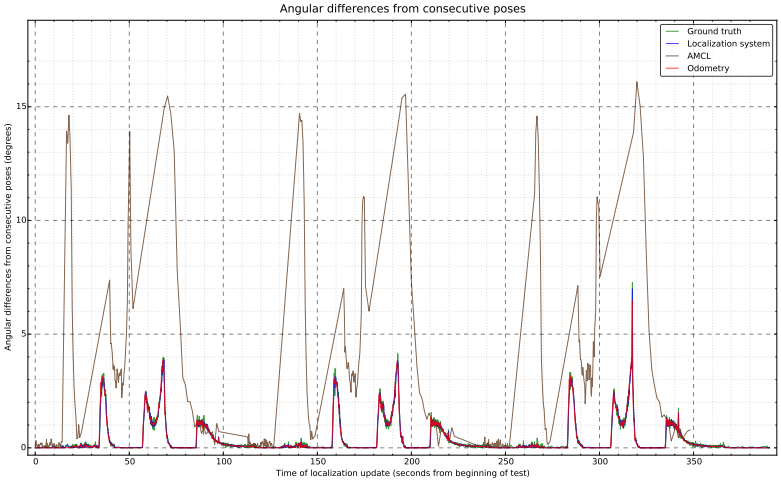


Figure A.1.4.13: Angular differences between consecutive pose estimations

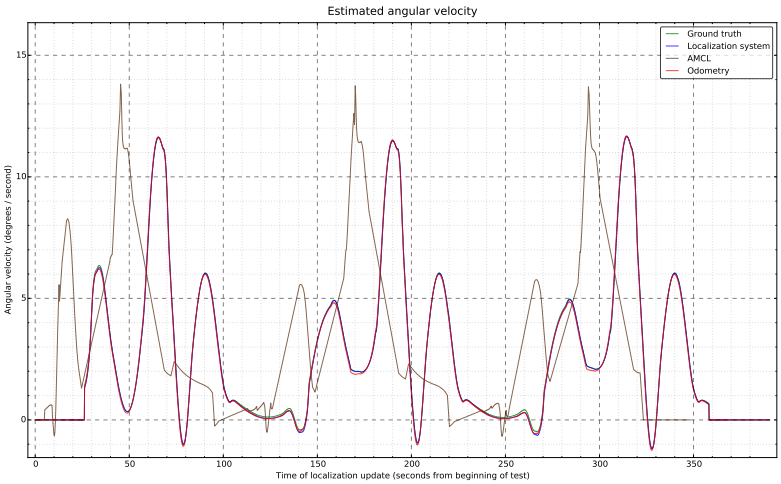


Figure A.1.4.14: Estimated angular velocity

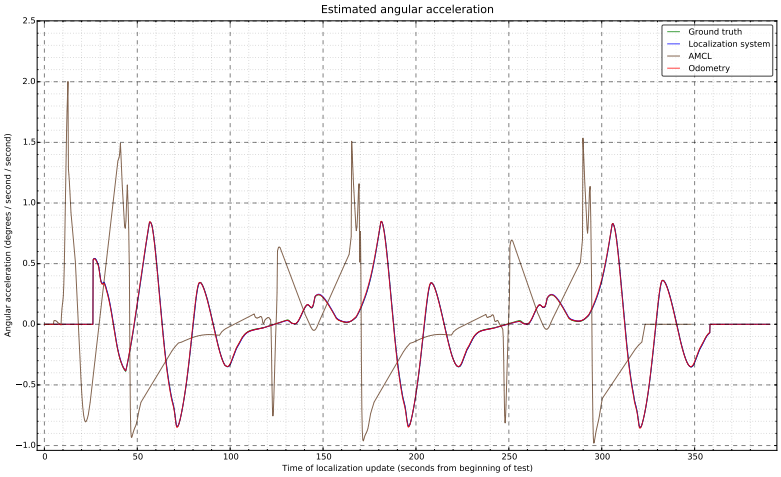


Figure A.1.4.15: Estimated angular acceleration

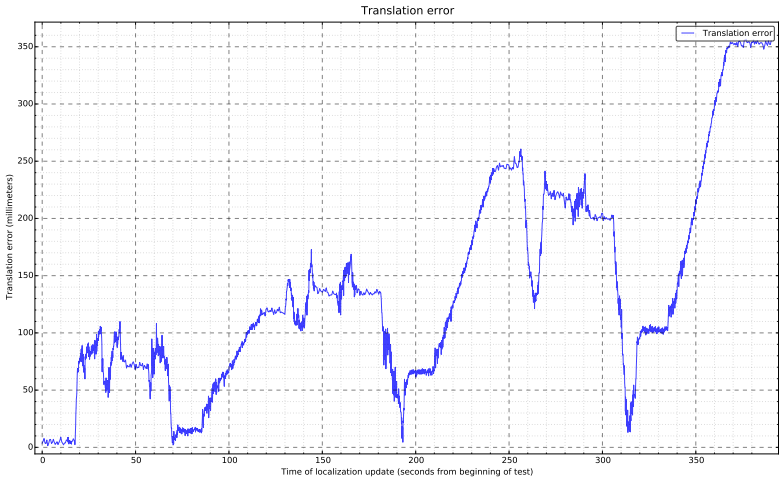


Figure A.1.4.16: Odometry translation errors

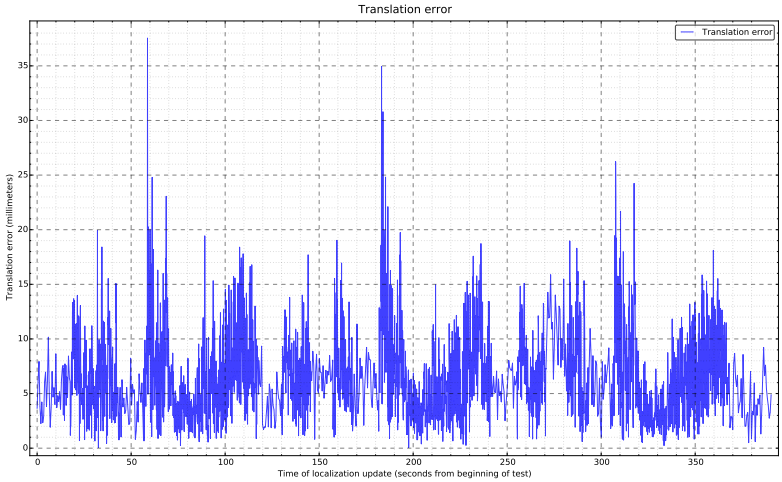


Figure A.1.4.17: Localization system translation errors

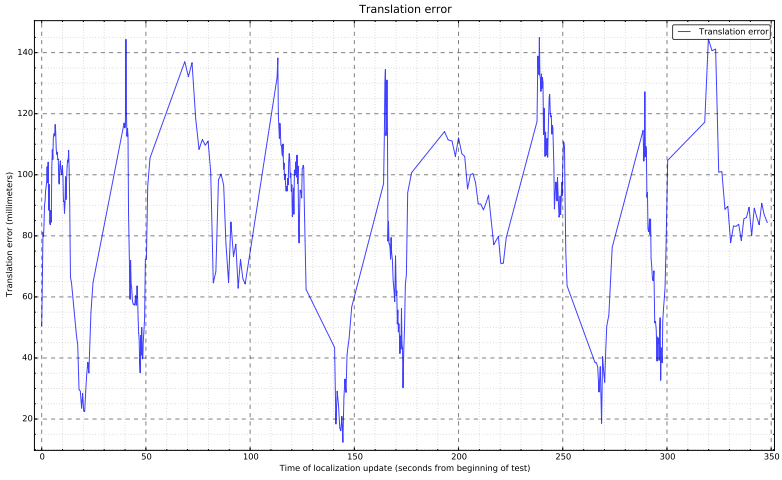


Figure A.1.4.18: AMCL translation errors

Figure A.1.4.19: Odometry translation errors components

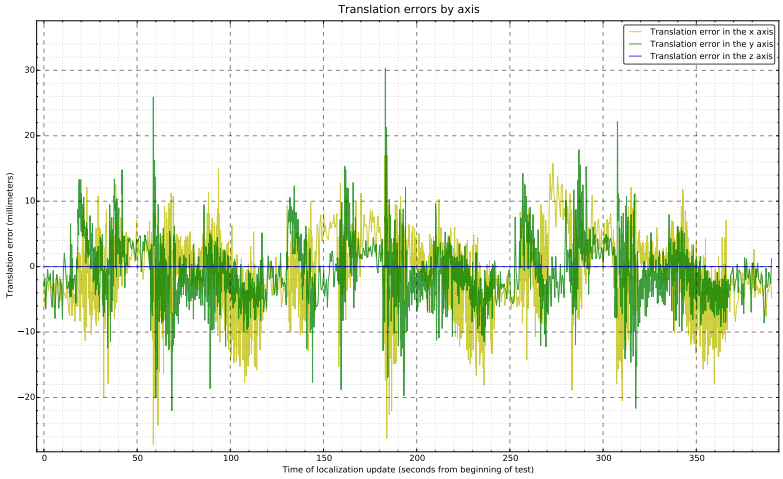


Figure A.1.4.20: Localization system translation errors components

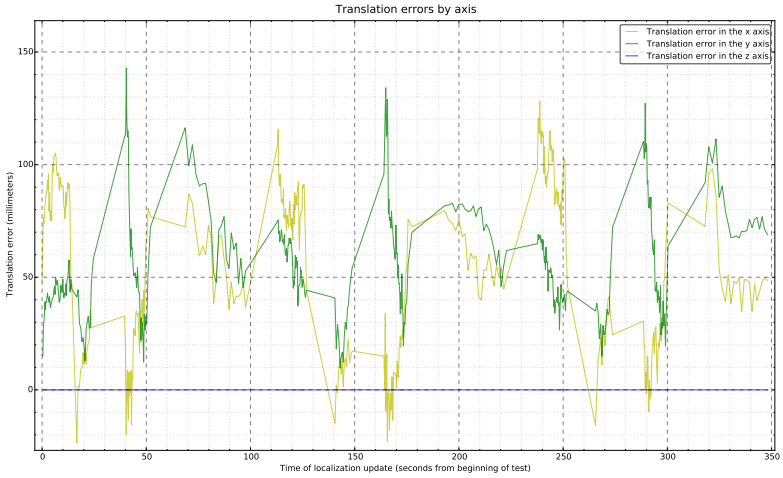


Figure A.1.4.21: AMCL translation errors components

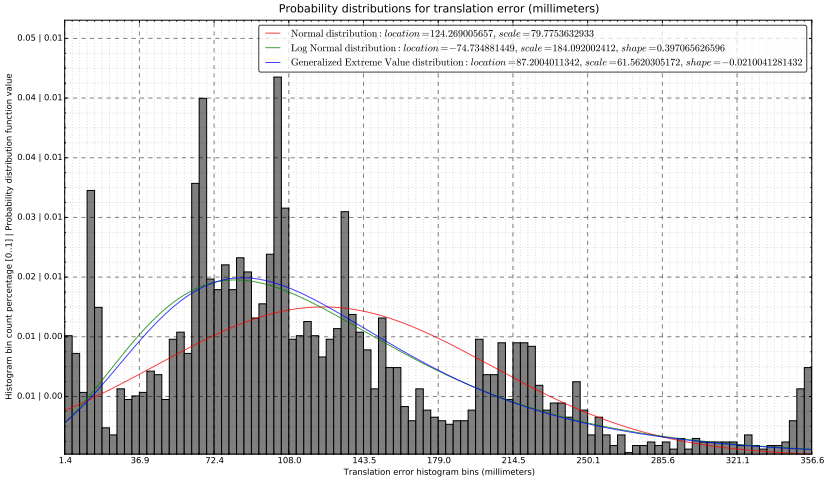


Figure A.1.4.22: Probability distributions for the odometry translation errors

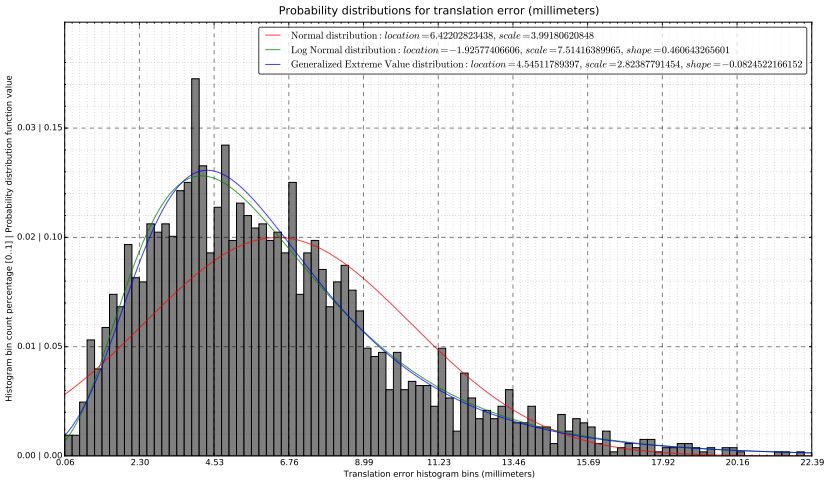


Figure A.1.4.23: Probability distributions for the localization system translation errors

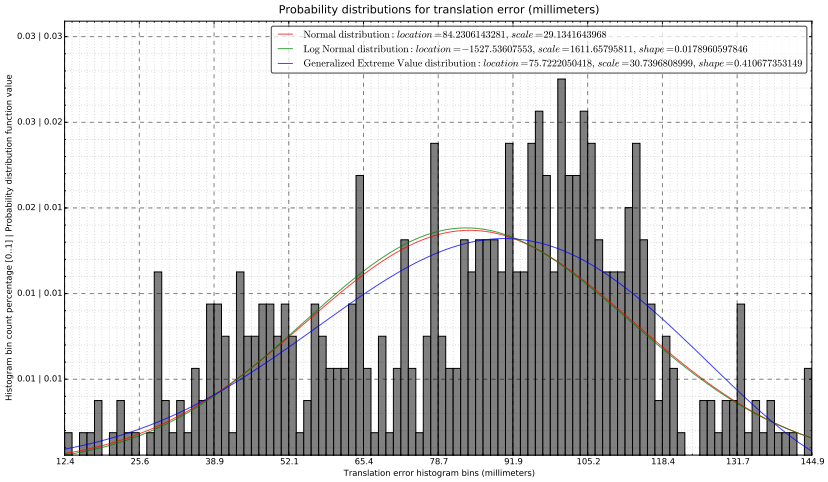


Figure A.1.4.24: Probability distributions for the AMCL translation errors

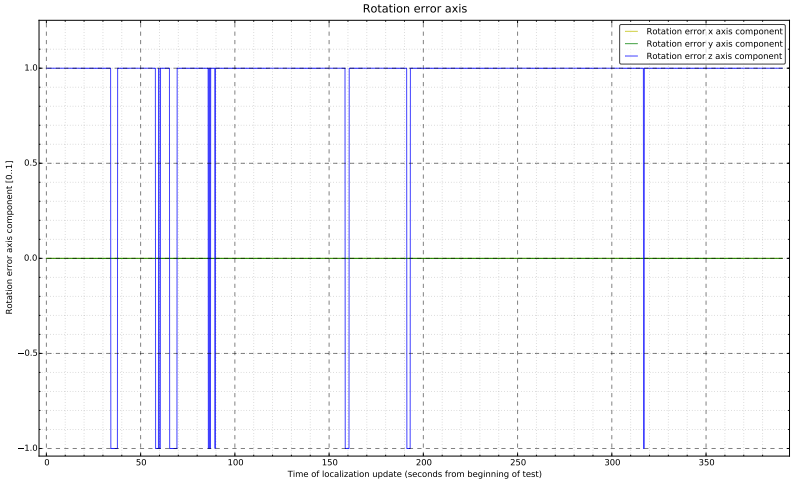


Figure A.1.4.25: Odometry rotation errors axis

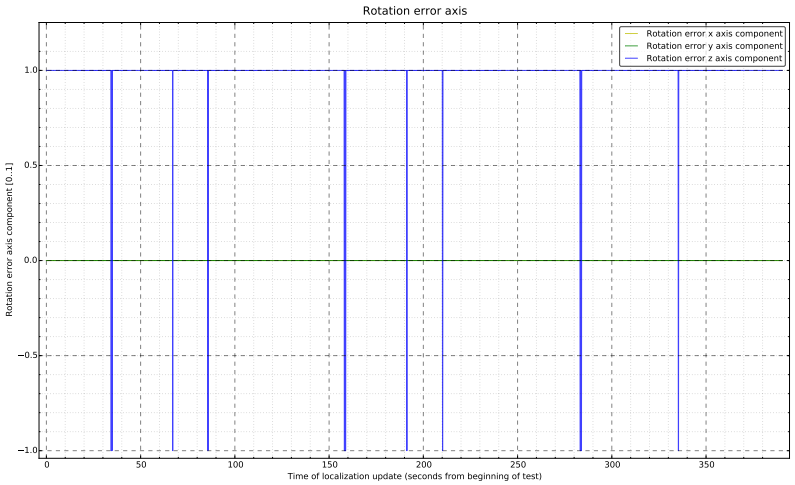


Figure A.1.4.26: Localization system rotation errors axis

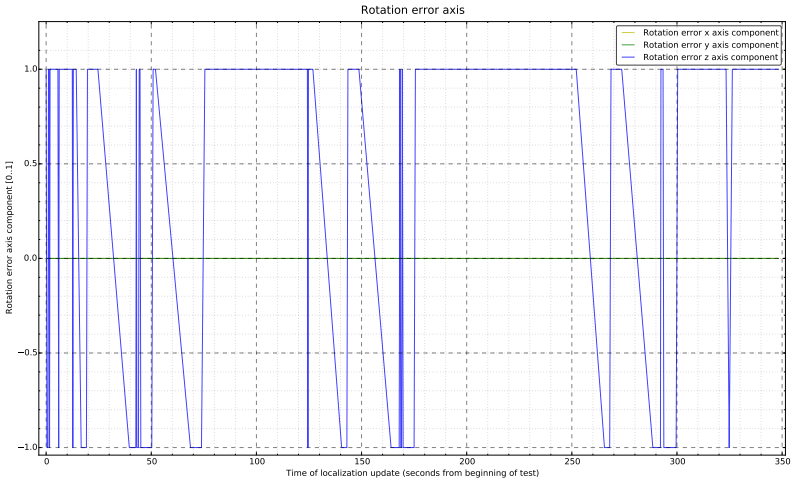


Figure A.1.4.27: AMCL rotation errors axis

Figure A.1.4.28: Odometry rotation errors

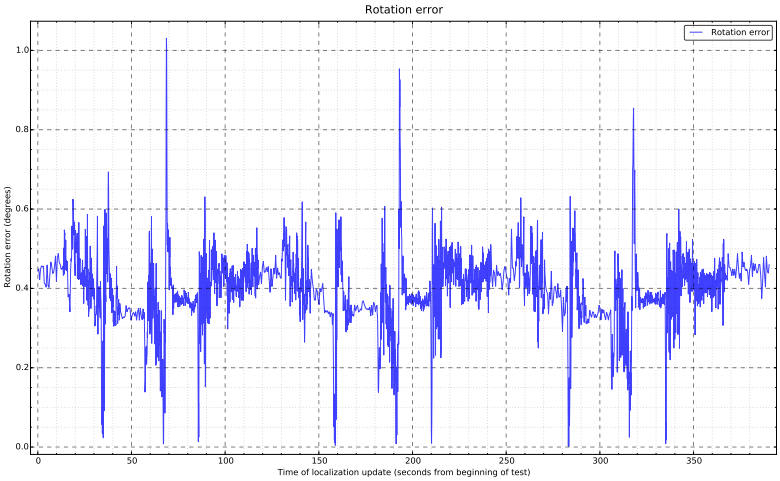


Figure A.1.4.29: Localization system rotation errors

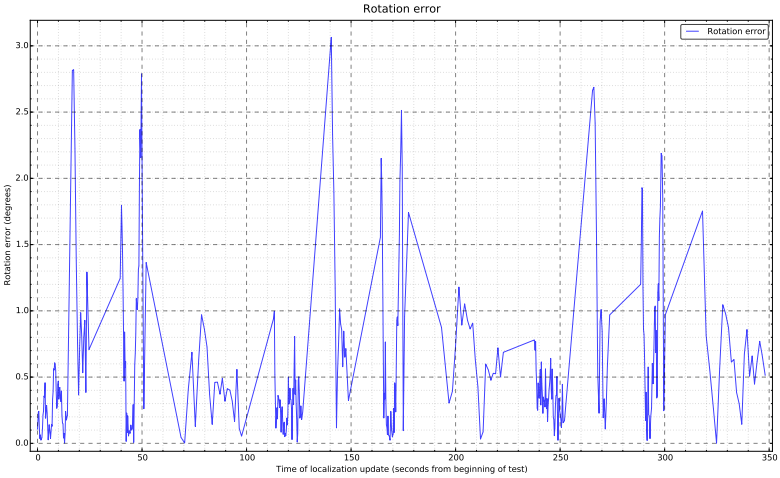


Figure A.1.4.30: AMCL rotation errors

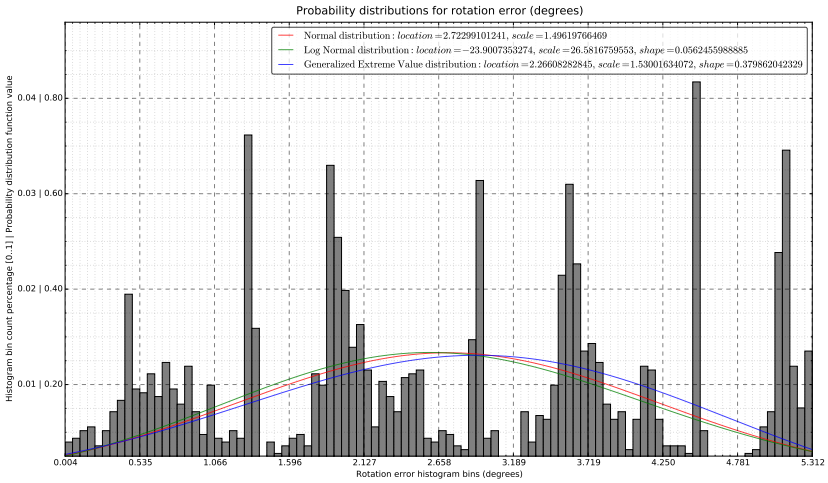


Figure A.1.4.31: Probability distributions for the odometry rotation errors

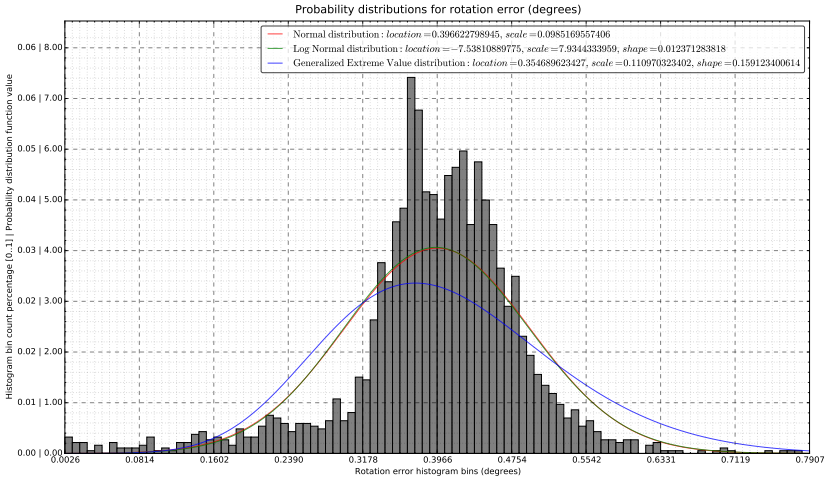


Figure A.1.4.32: Probability distributions for the localization system rotation errors

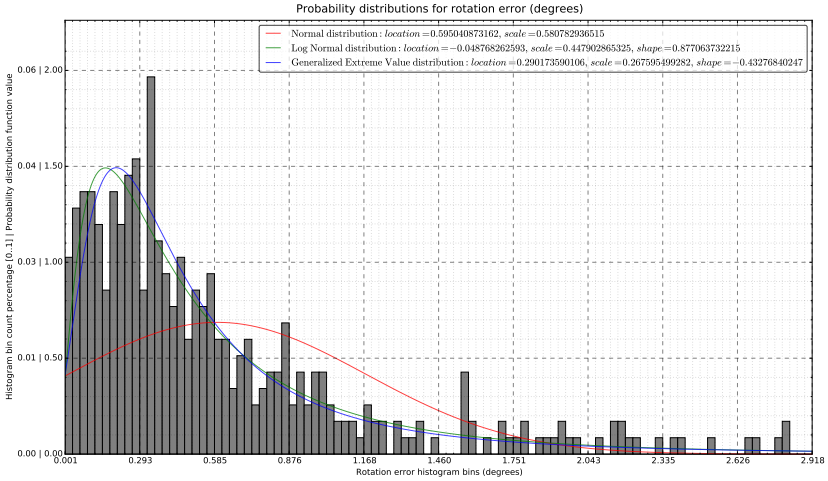


Figure A.1.4.33: Probability distributions for the AMCL rotation errors

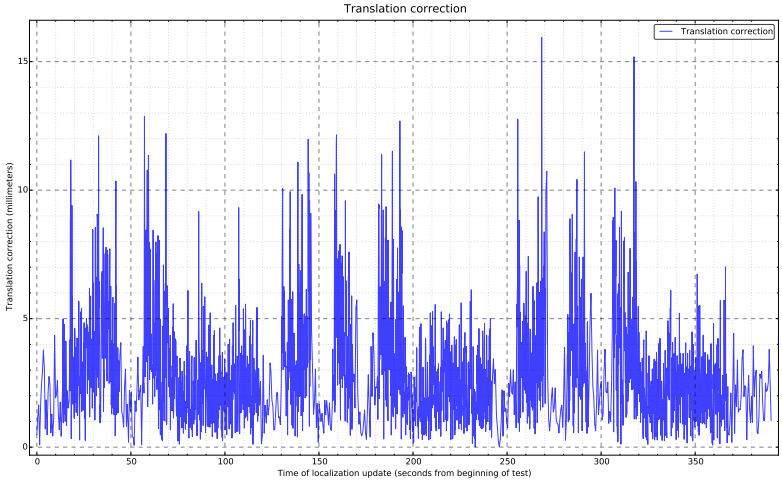


Figure A.1.4.34: Translation corrections performed by the localization system

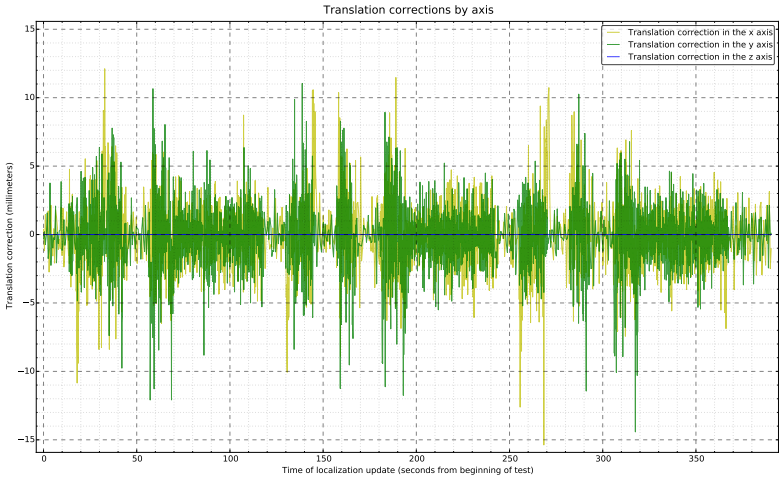


Figure A.1.4.35: Translation corrections components performed by the localization system

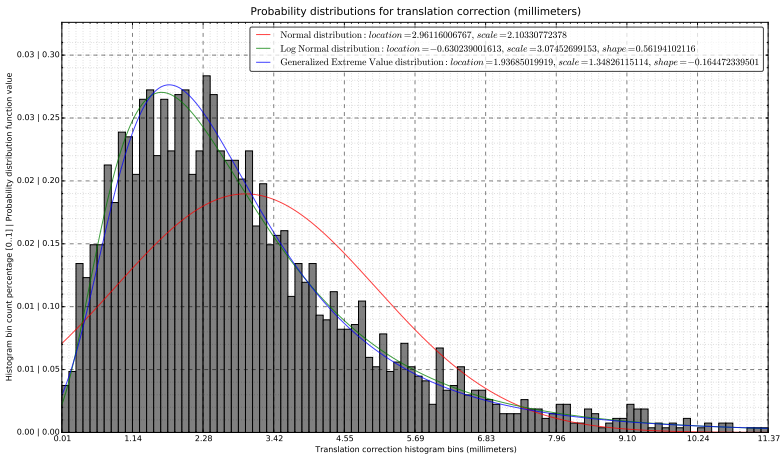


Figure A.1.4.36: Probability distributions for the translation corrections performed by the localization system

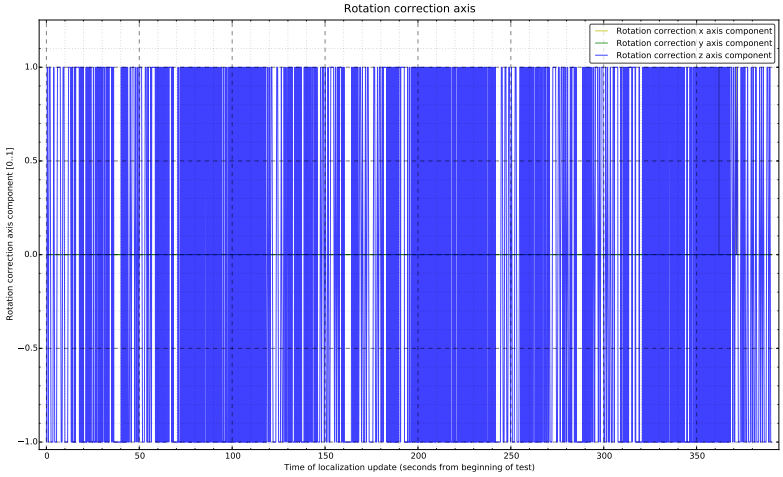


Figure A.1.4.37: Rotation corrections (axis) performed by the localization system

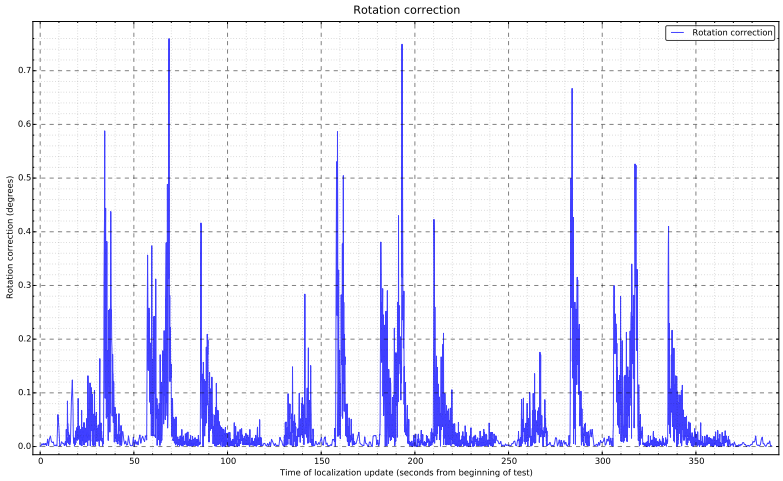


Figure A.1.4.38: Rotation corrections performed by the localization system

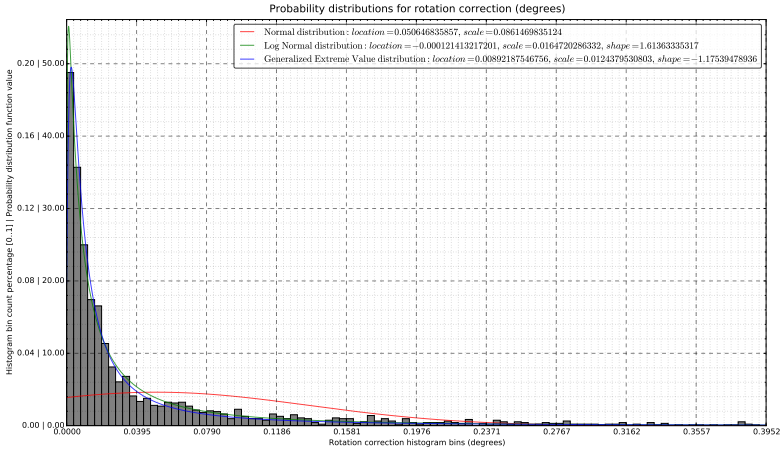


Figure A.1.4.39: Probability distributions for the rotation corrections performed by the localization system

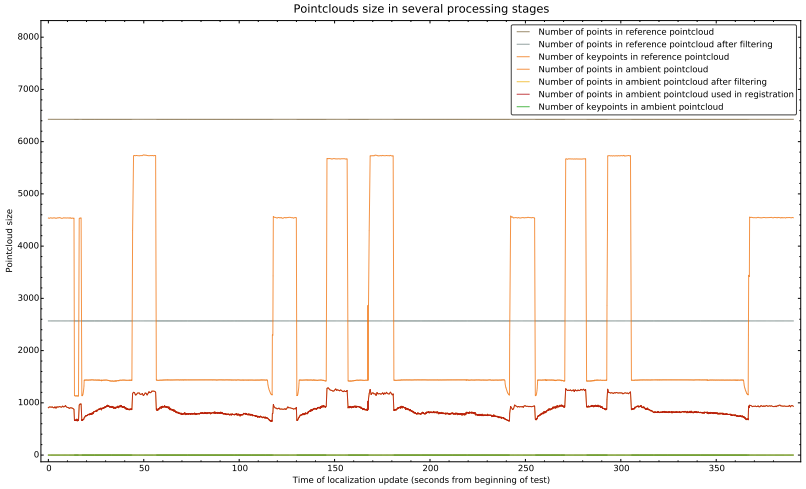


Figure A.1.4.40: Point clouds size in several of the localization system processing stages

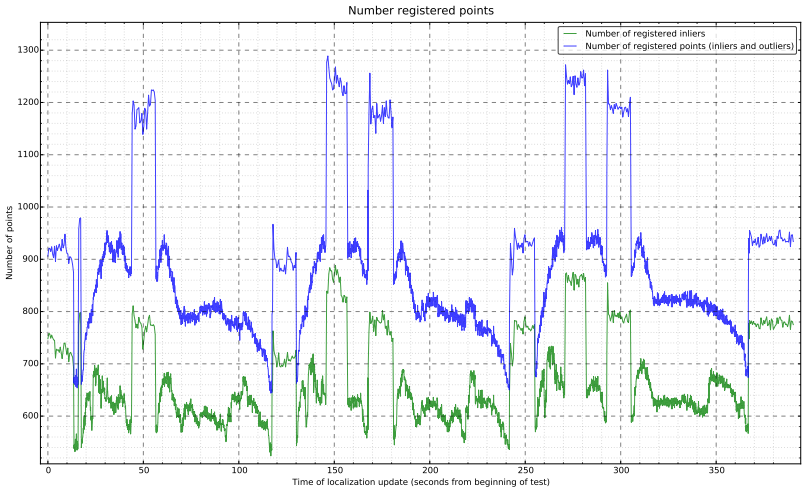


Figure A.1.4.41: Number of registered points

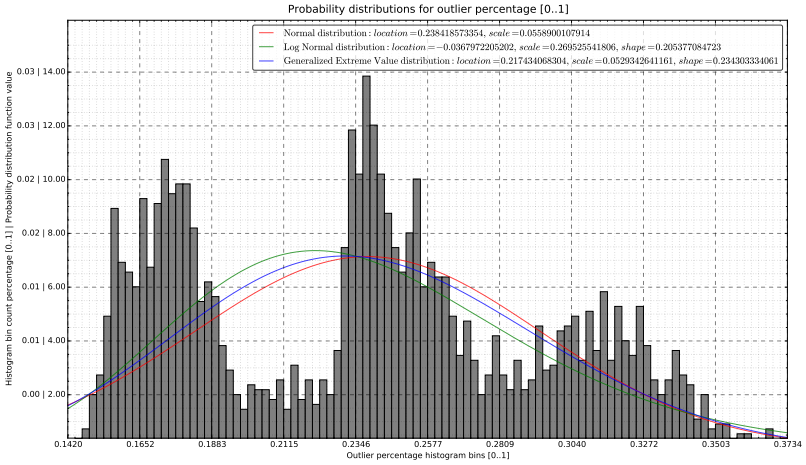


Figure A.1.4.42: Probability distributions for the ambient point cloud outlier percentage

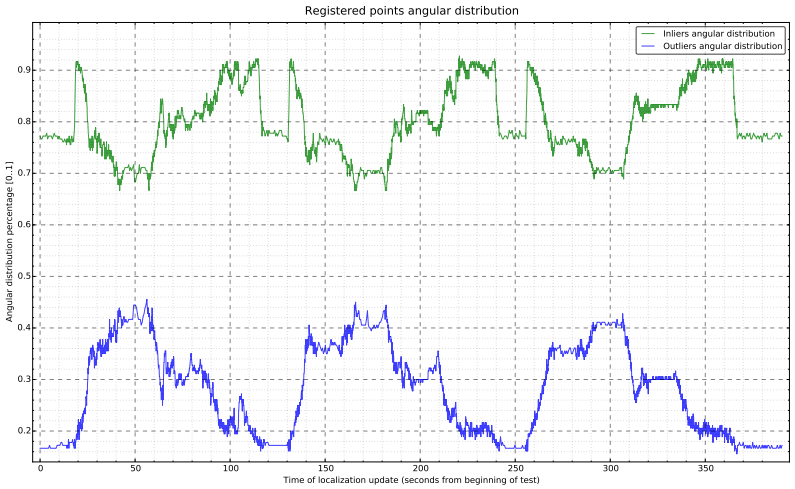


Figure A.1.4.43: Angular distribution percentage of registered points

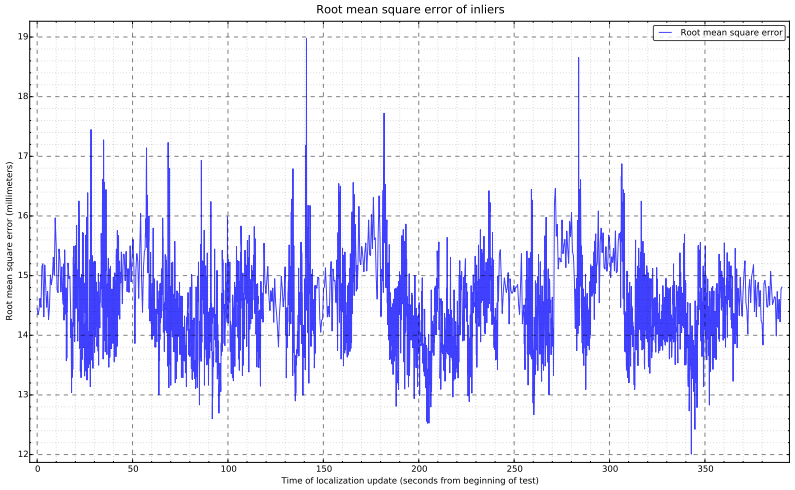


Figure A.1.4.44: Root Mean Square Error of the inliers

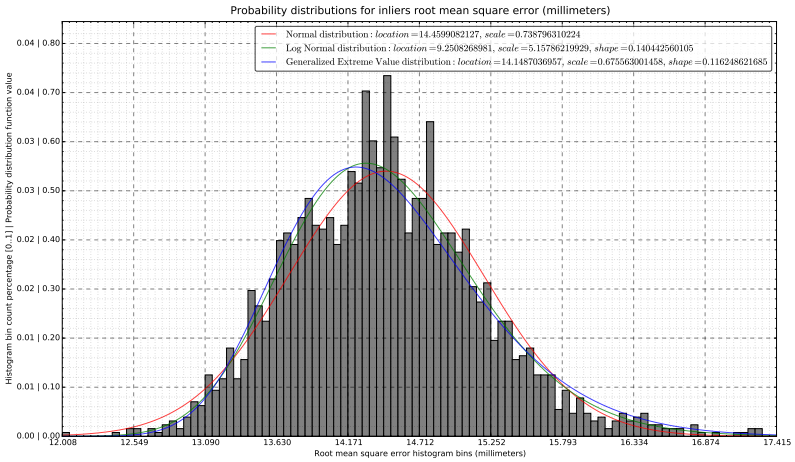


Figure A.1.4.45: Probability distributions for the Root Mean Square Error of the inliers

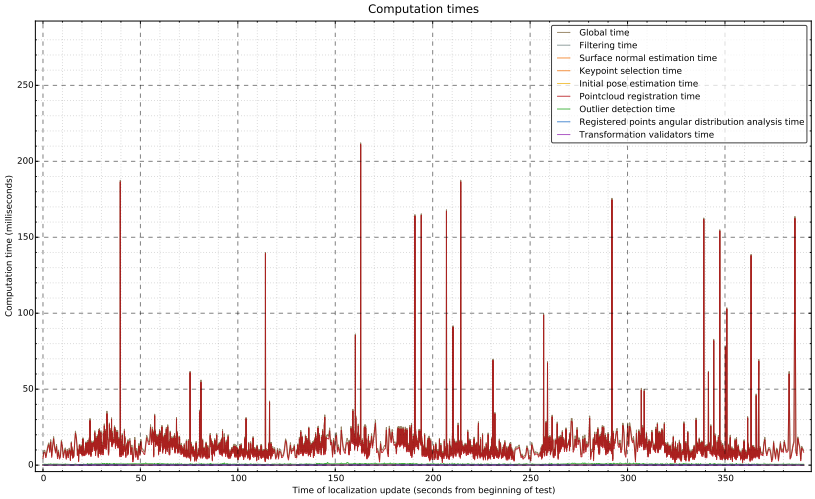


Figure A.1.4.46: Localization system computation times

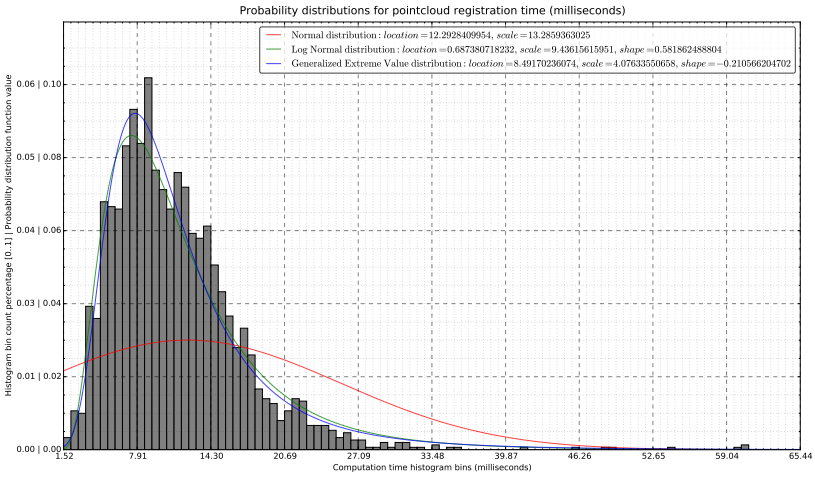


Figure A.1.4.47: Probability distributions for the point cloud registration time

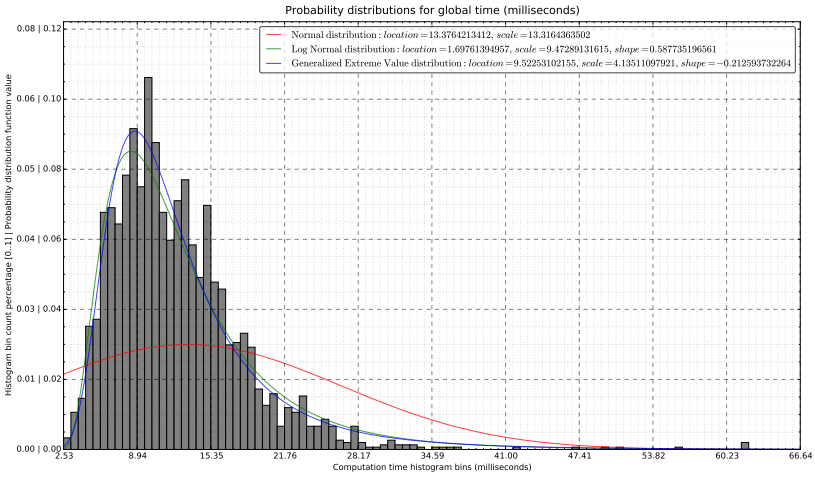


Figure A.1.4.48: Probability distributions for the localization system global computation time

A.1.5 Complex path using the Jarvis robot at 5 cm/s without laser spherical linear interpolation

The figures in this section show the results of a test performed with same configurations as the previous section, but now without using laser spherical linear interpolation.

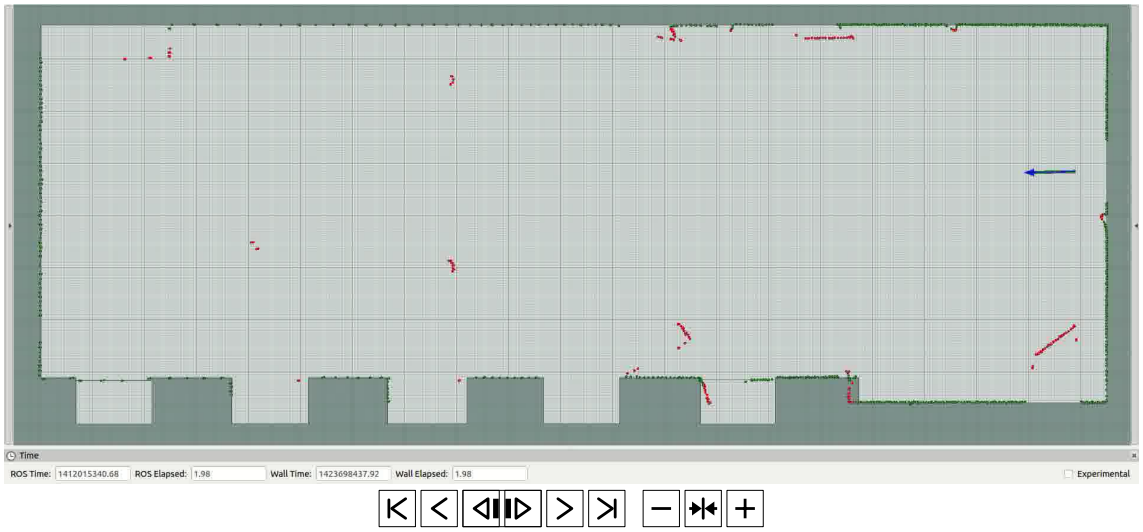


Figure A.1.5.1: Animation of laser scans using spherical linear interpolation

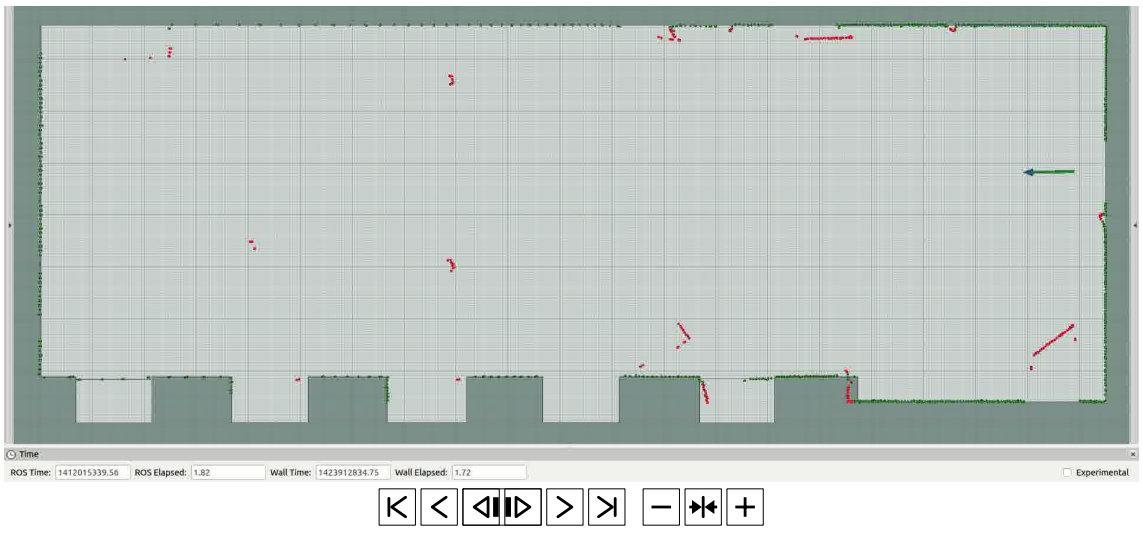


Figure A.1.5.2: Animation of laser scans without using spherical linear interpolation

Figure A.1.5.3: Laser scans assembled in the previous test using laser spherical linear interpolation

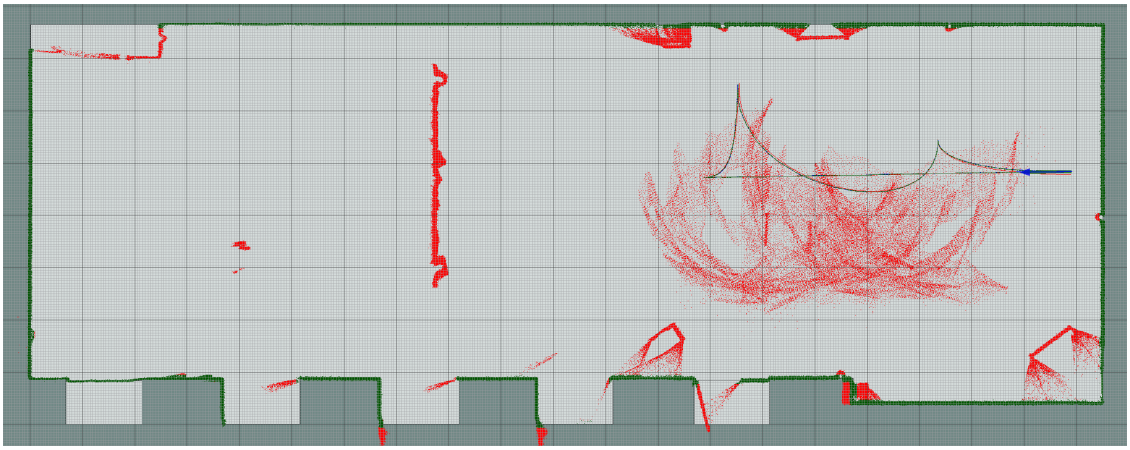


Figure A.1.5.4: Laser scans assembled without using laser spherical linear interpolation

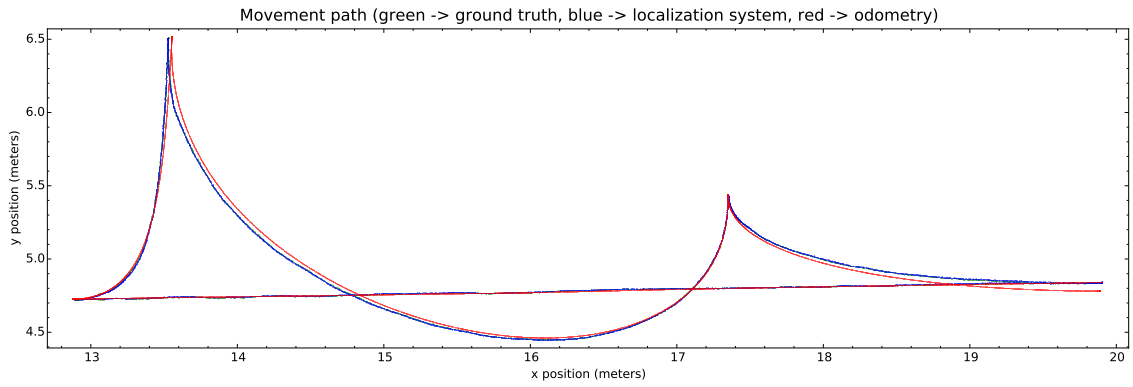


Figure A.1.5.5: Poses estimated by the ground truth, localization system and odometry

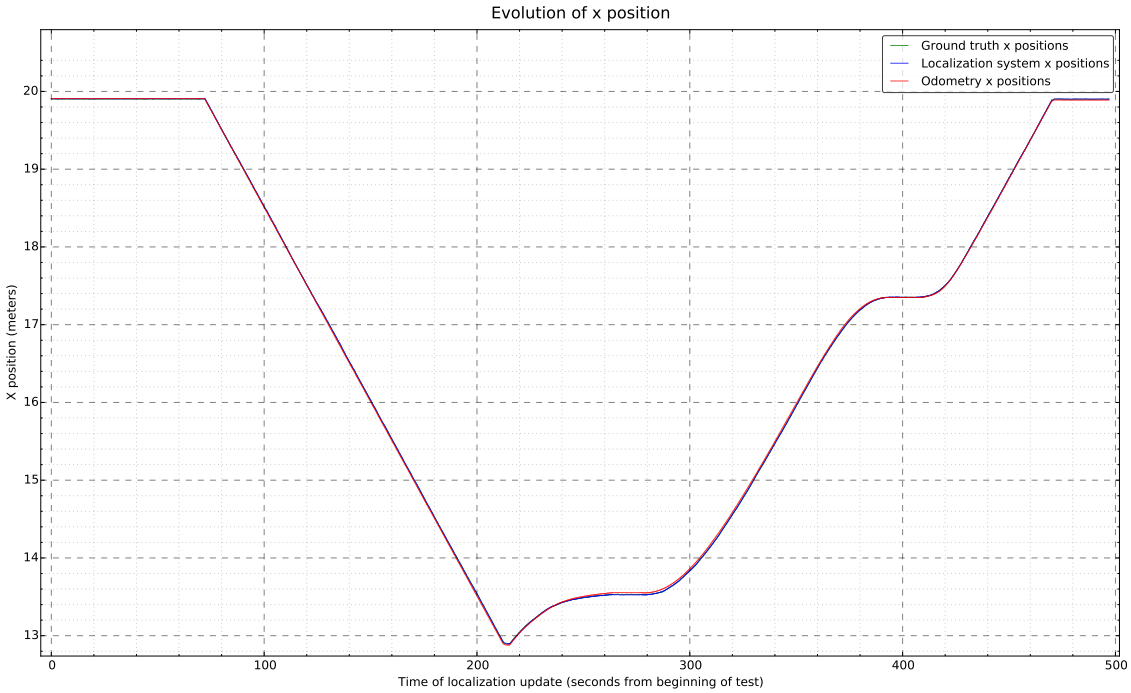


Figure A.1.5.6: Evolution of x position over time

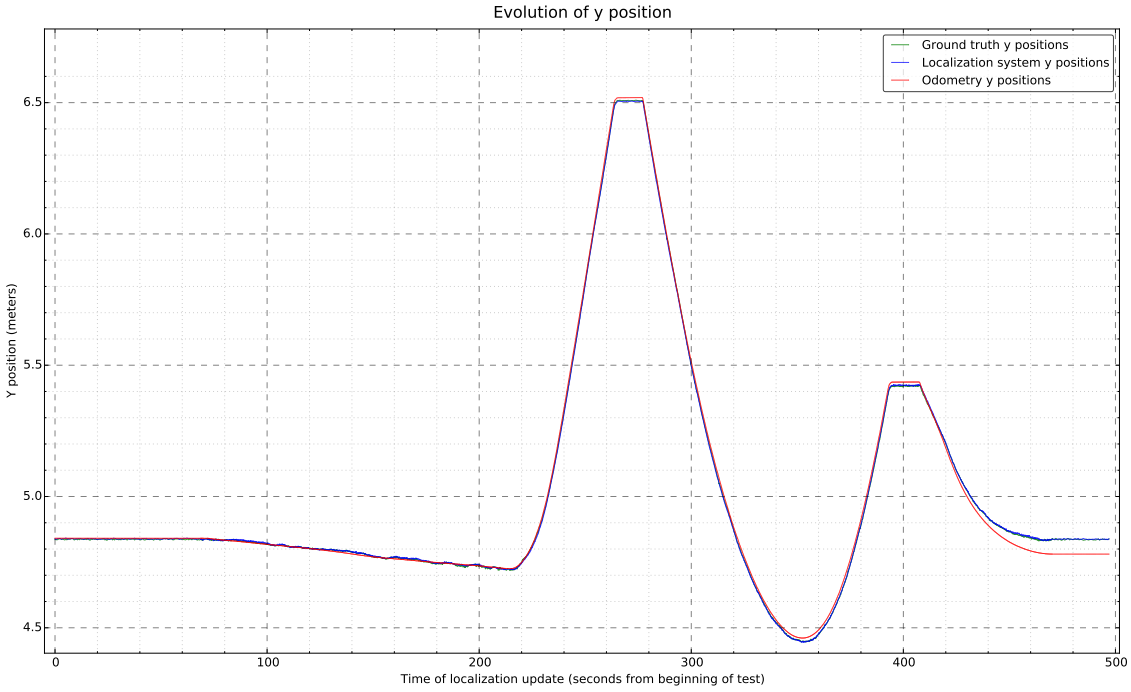


Figure A.1.5.7: Evolution of y position over time

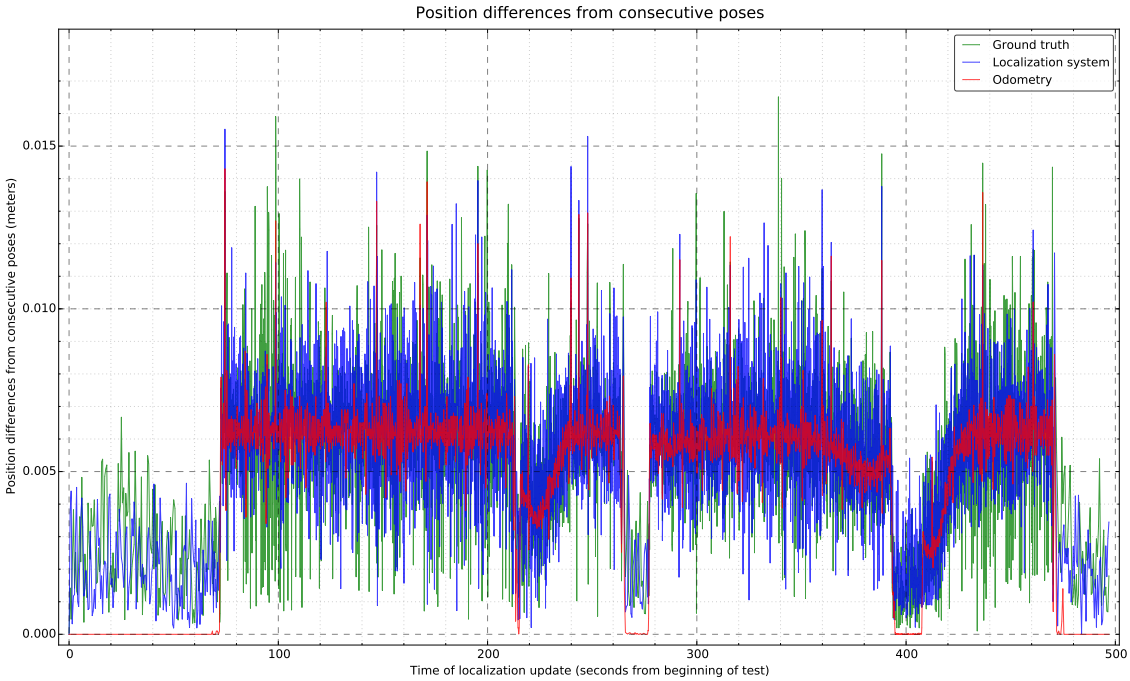


Figure A.1.5.8: Distance traveled between consecutive pose estimations

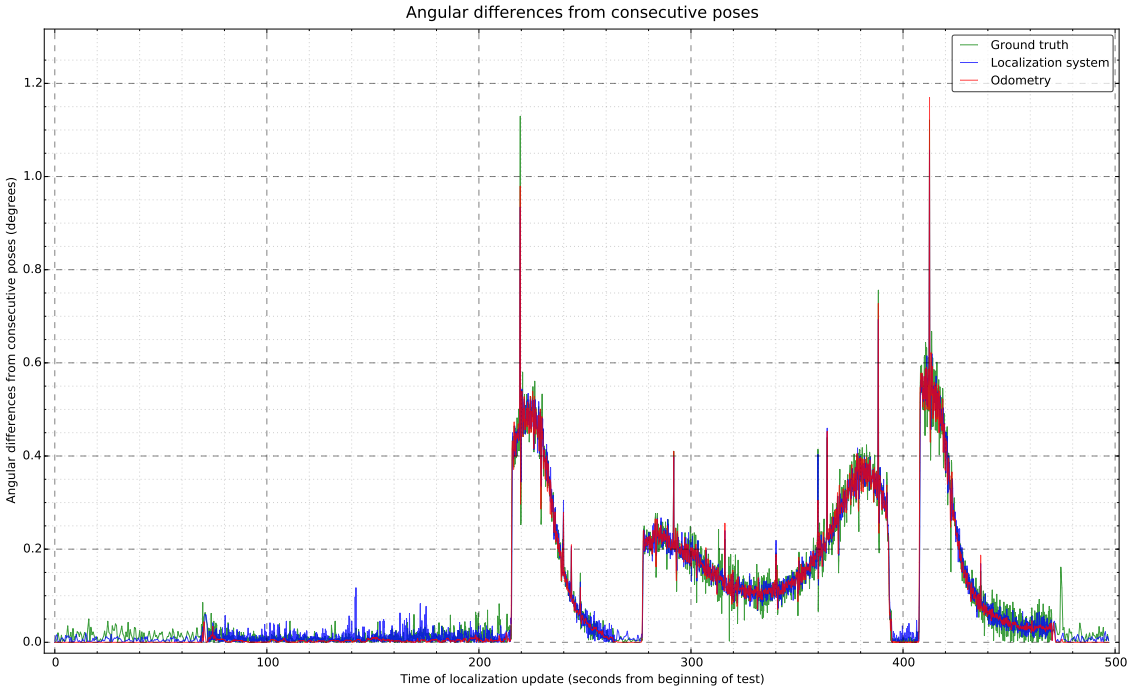


Figure A.1.5.9: Angular differences between consecutive pose estimations

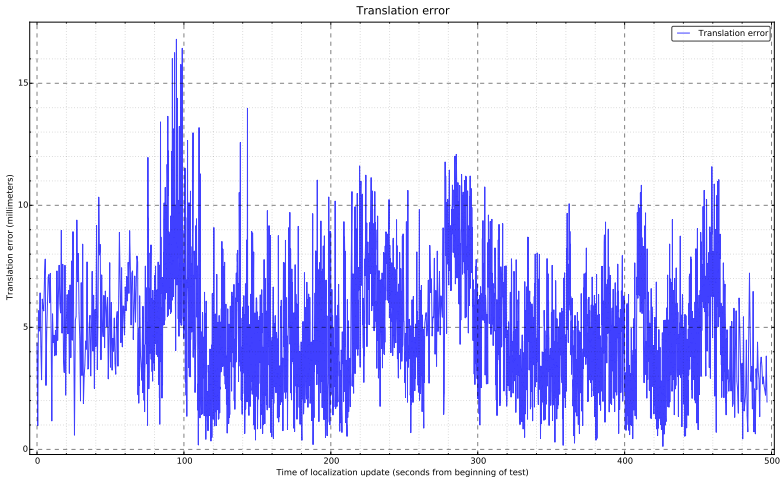


Figure A.1.5.10: Localization system translation errors

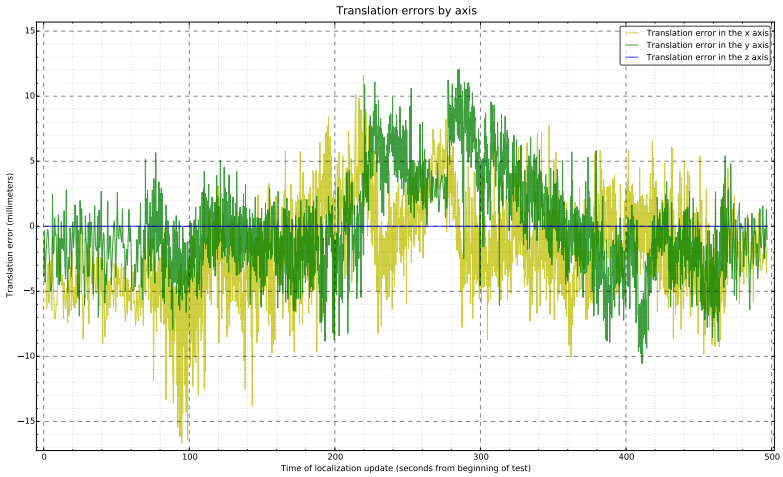


Figure A.1.5.11: Localization system translation errors components

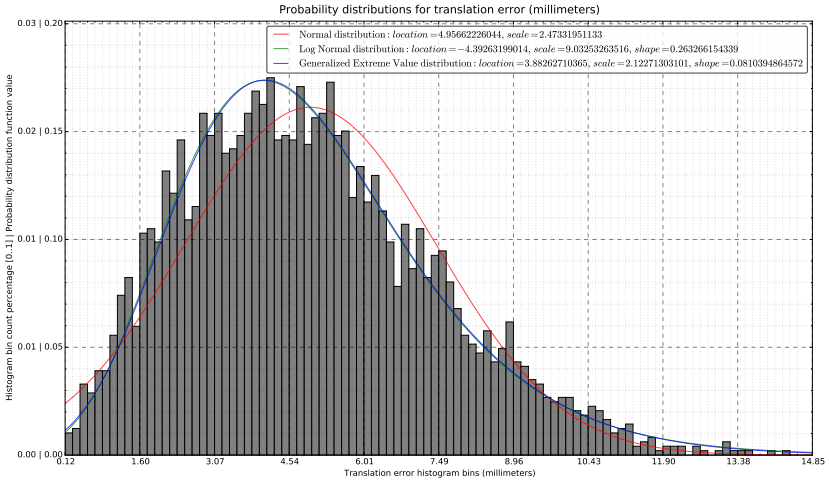


Figure A.1.5.12: Probability distributions for the localization system translation errors

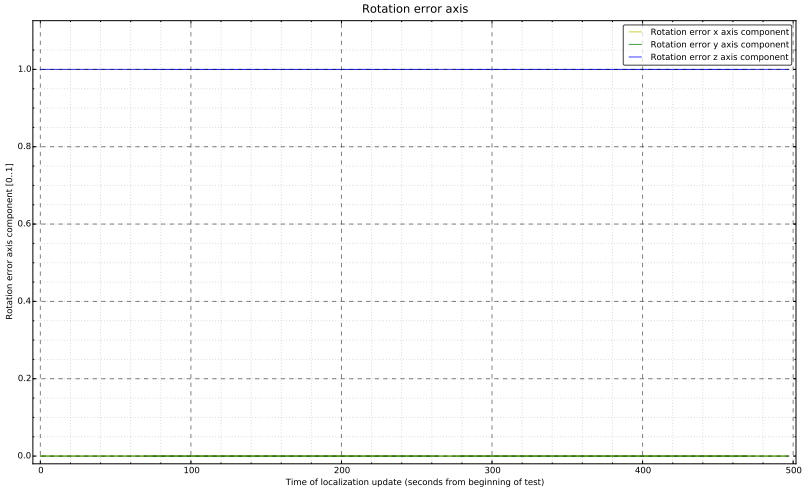


Figure A.1.5.13: Localization system rotation errors axis

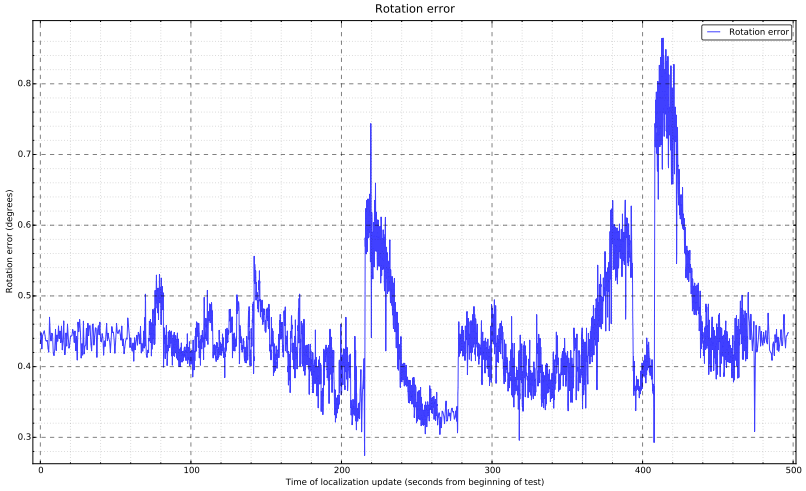


Figure A.1.5.14: Localization system rotation errors

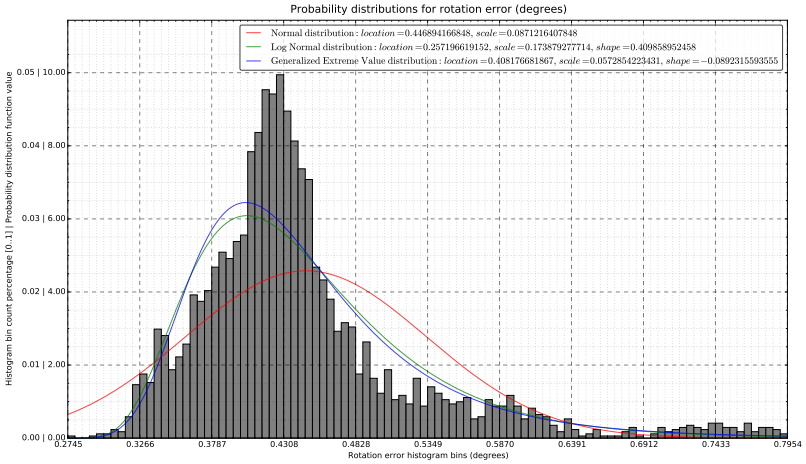


Figure A.1.5.15: Probability distributions for the localization system rotation errors

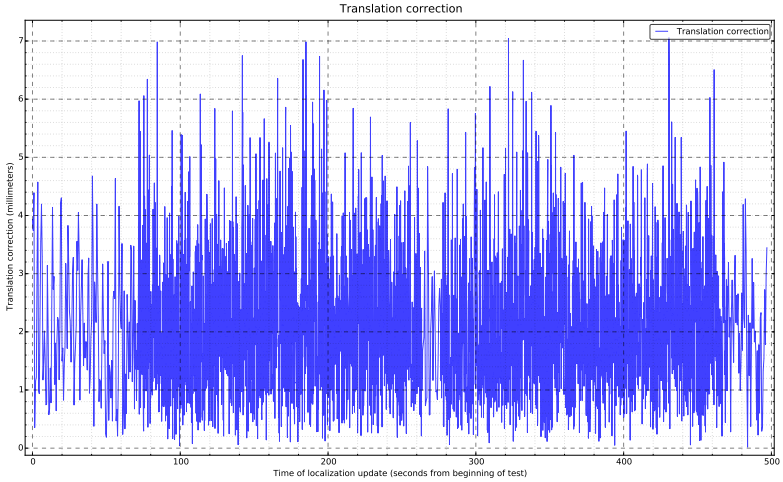


Figure A.1.5.16: Translation corrections performed by the localization system

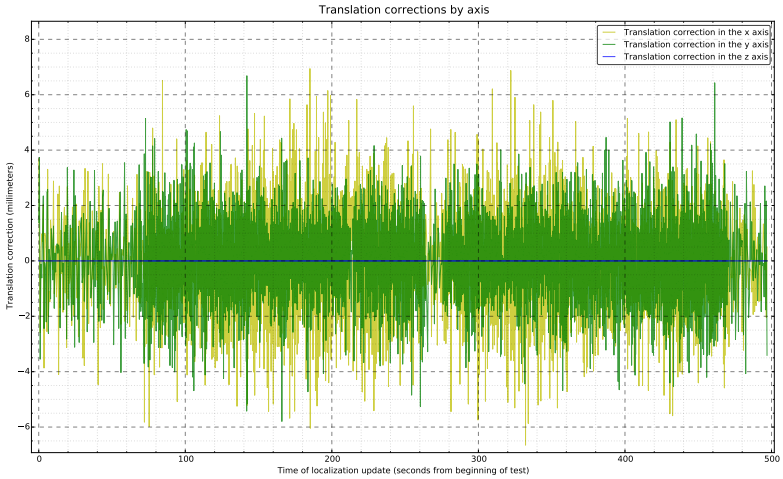


Figure A.1.5.17: Translation corrections components performed by the localization system

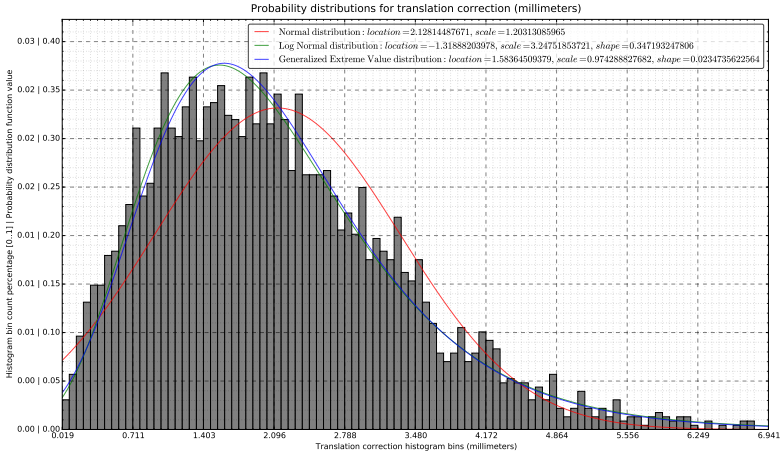


Figure A.1.5.18: Probability distributions for the translation corrections performed by the localization system

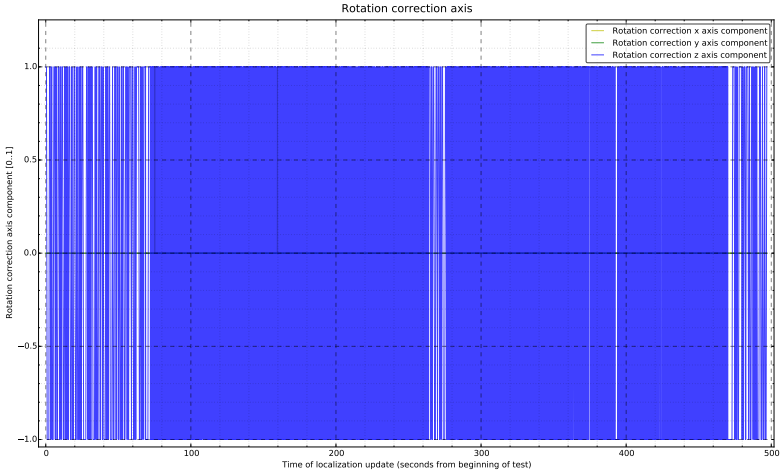


Figure A.1.5.19: Rotation corrections (axis) performed by the localization system

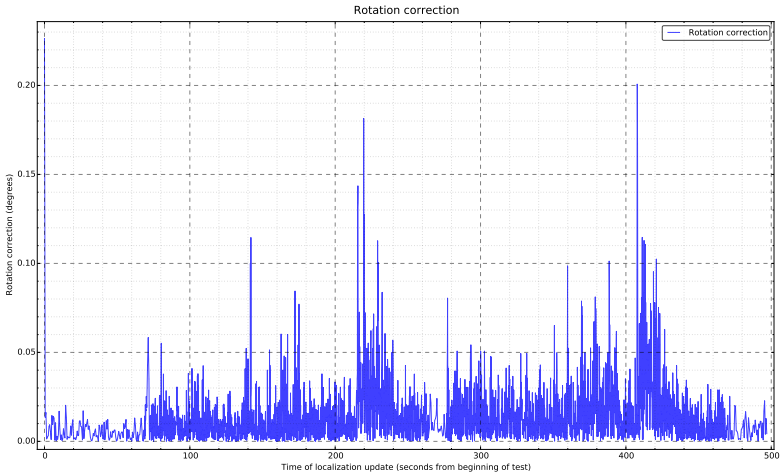


Figure A.1.5.20: Rotation corrections performed by the localization system

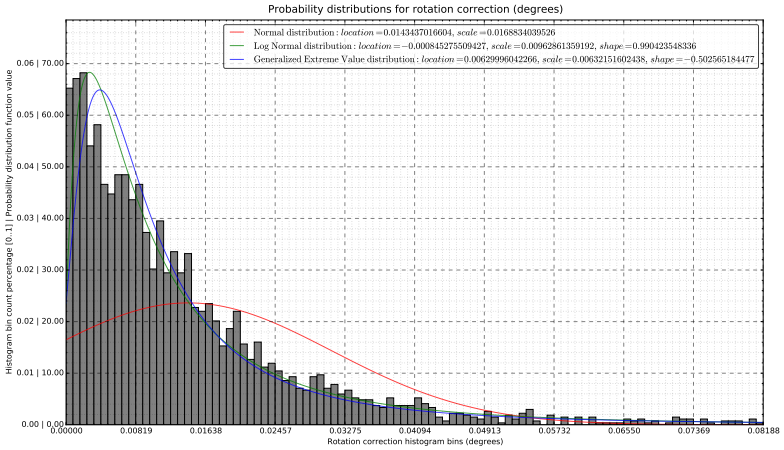


Figure A.1.5.21: Probability distributions for the rotation corrections performed by the localization system

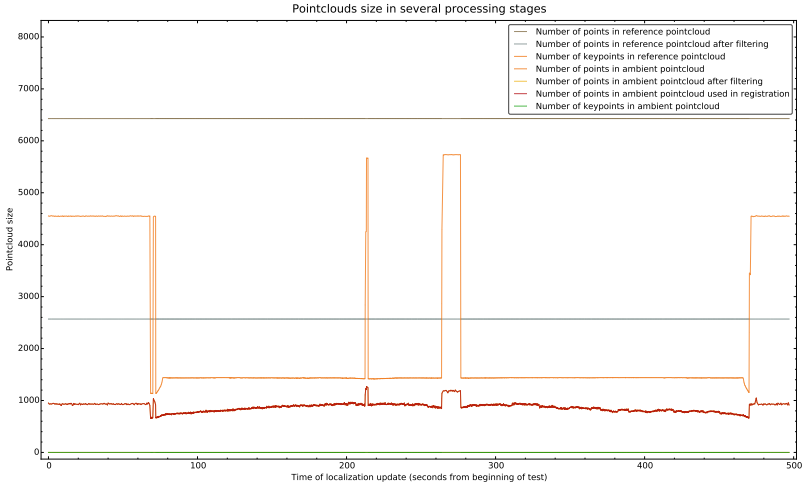


Figure A.1.5.22: Point clouds size in several of the localization system processing stages

Figure A.1.5.23: Number of registered points

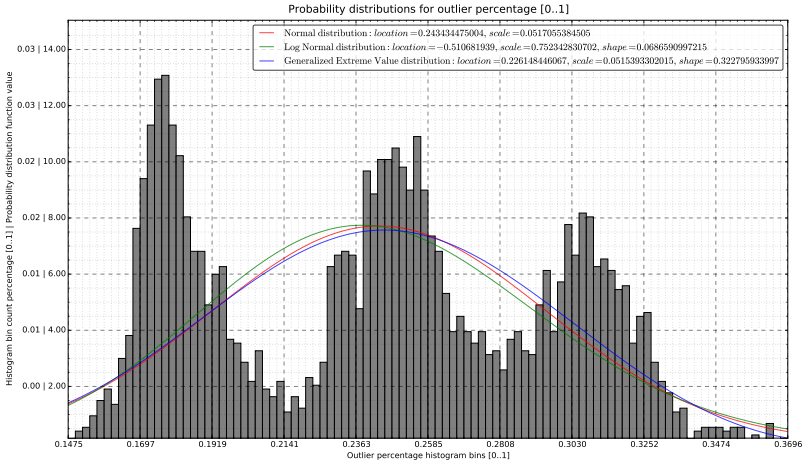


Figure A.1.5.24: Probability distributions for the ambient point cloud outlier percentage

Figure A.1.5.25: Angular distribution percentage of registered points

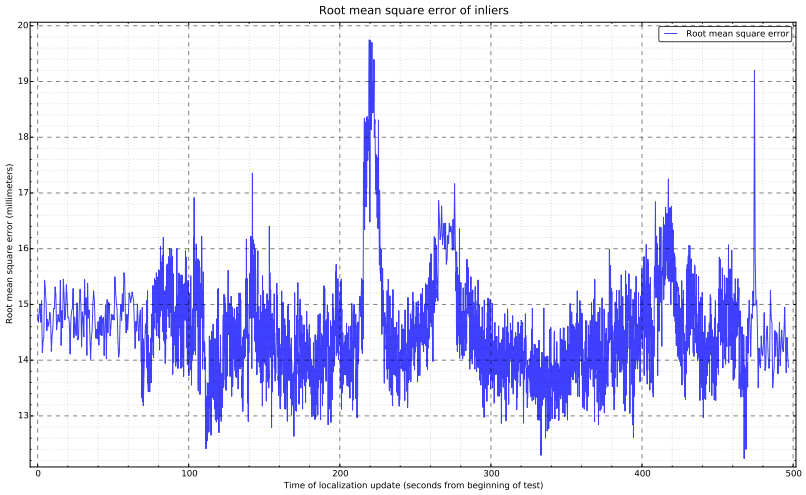


Figure A.1.5.26: Root Mean Square Error of the inliers

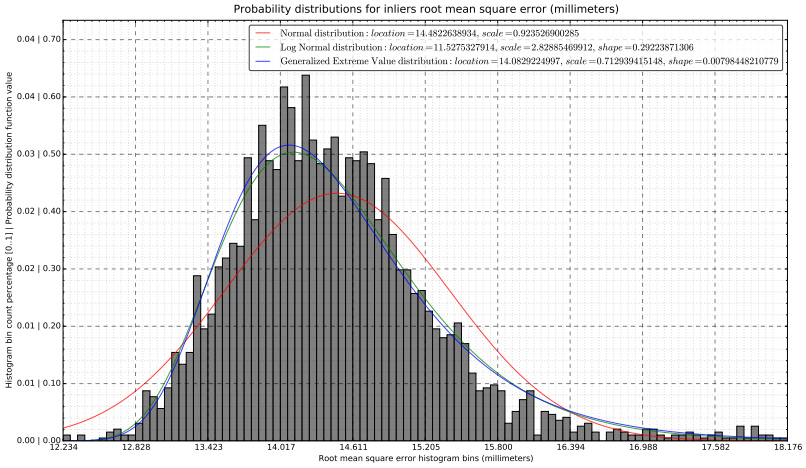


Figure A.1.5.27: Probability distributions for the Root Mean Square Error of the inliers

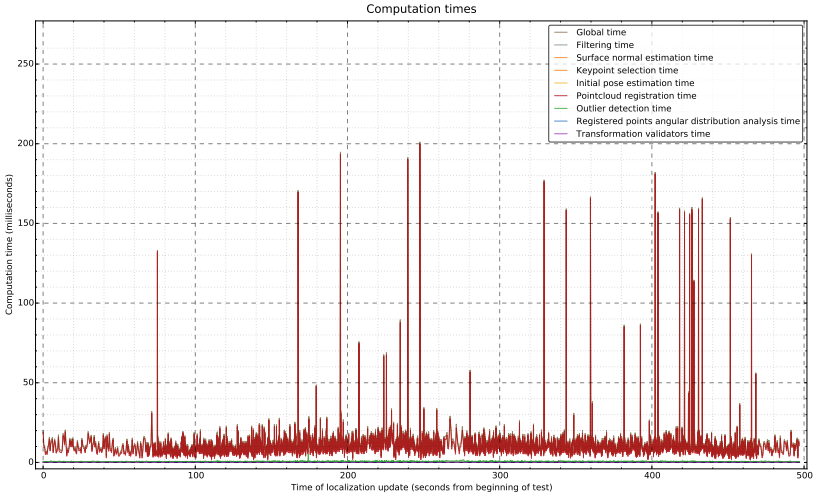


Figure A.1.5.28: Localization system computation times

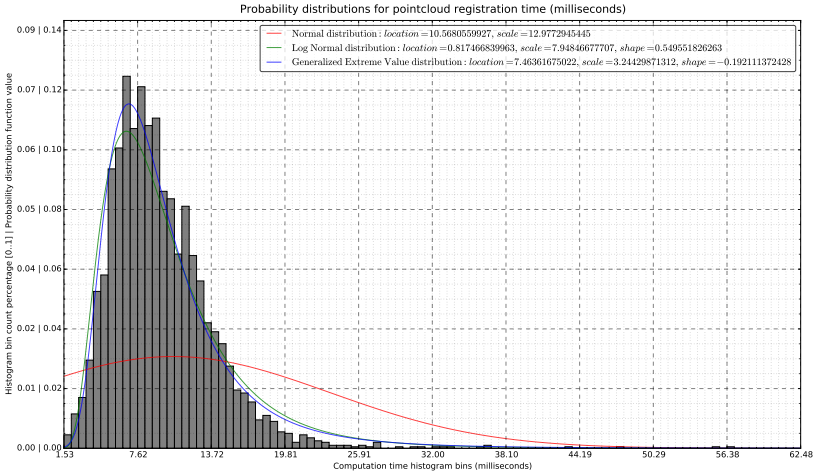


Figure A.1.5.29: Probability distributions for the point cloud registration time

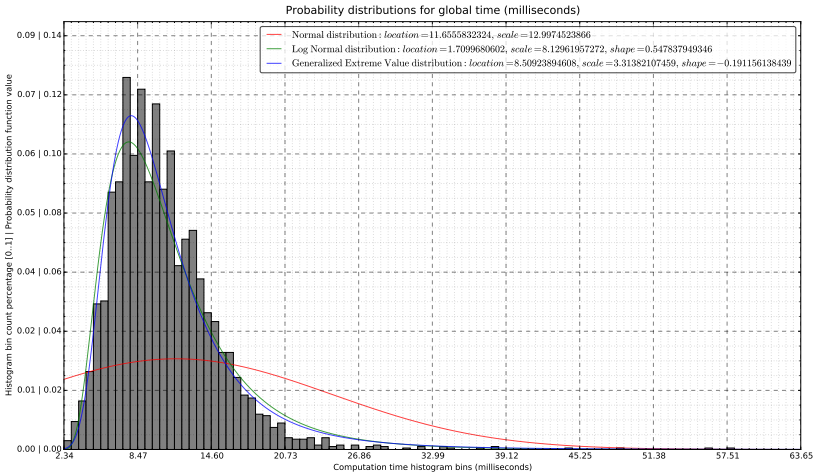


Figure A.1.5.30: Probability distributions for the localization system global computation time

A.1.6 Complex path using the Jarvis robot at 50-30-50-10 cm/s without laser spherical linear interpolation

The figures in this section show the results of a test performed with same configurations as the previous section, but now without using laser spherical linear interpolation.

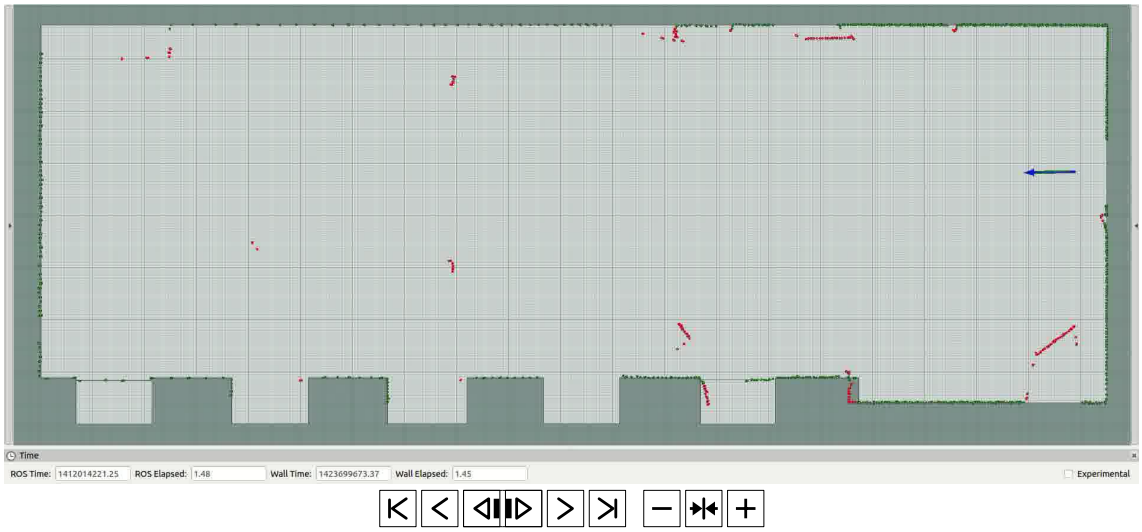


Figure A.1.6.1: Animation of laser scans using spherical linear interpolation

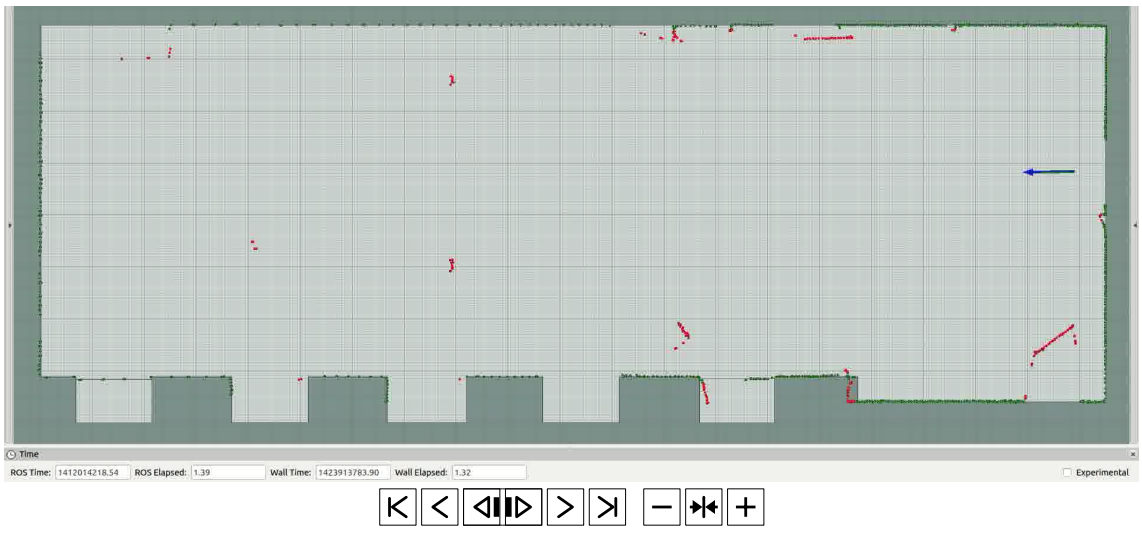


Figure A.1.6.2: Animation of laser scans without using spherical linear interpolation

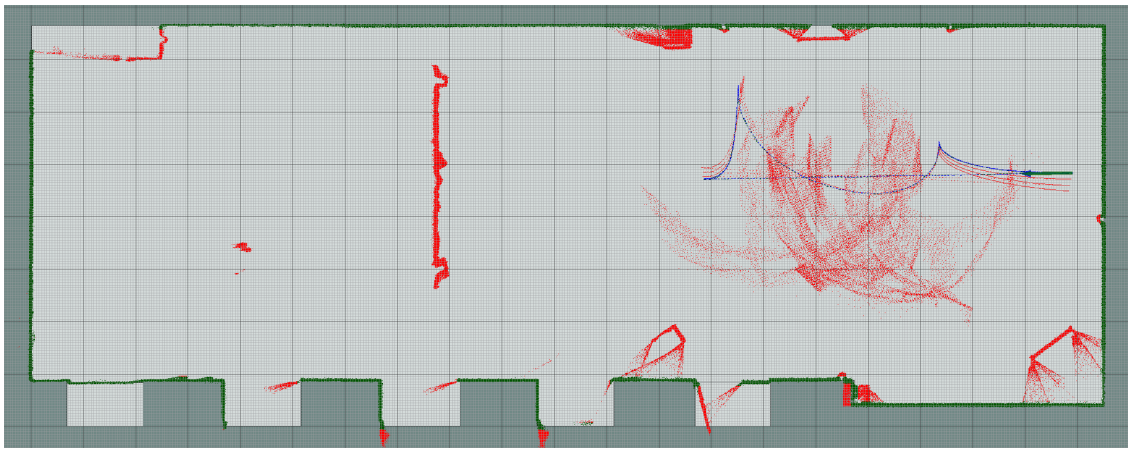


Figure A.1.6.3: Laser scans assembled in the previous test using laser spherical linear interpolation

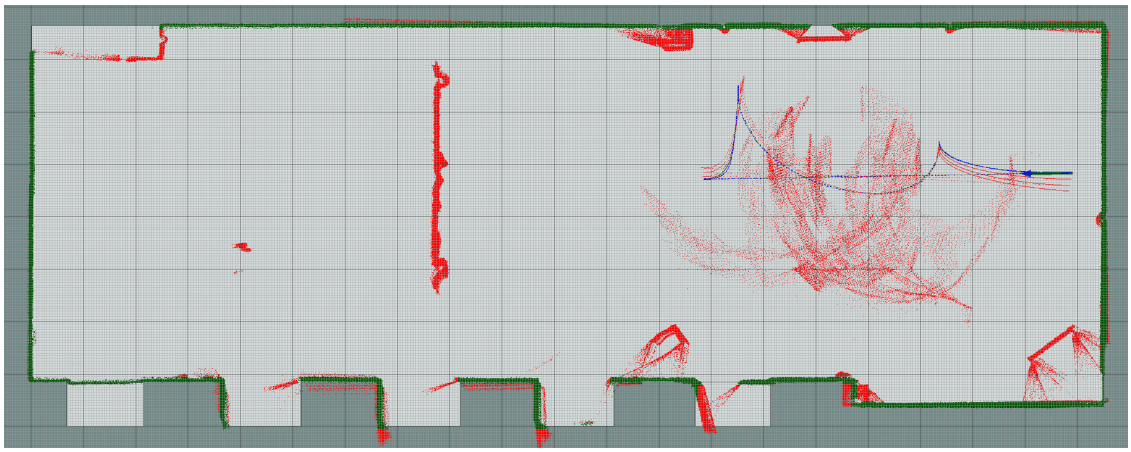


Figure A.1.6.4: Laser scans assembled without using laser spherical linear interpolation

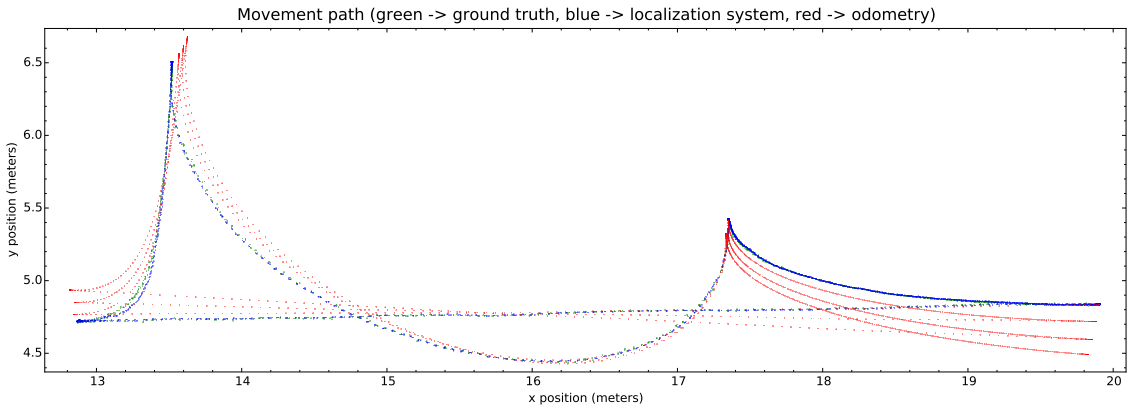


Figure A.1.6.5: Poses estimated by the ground truth, localization system and odometry

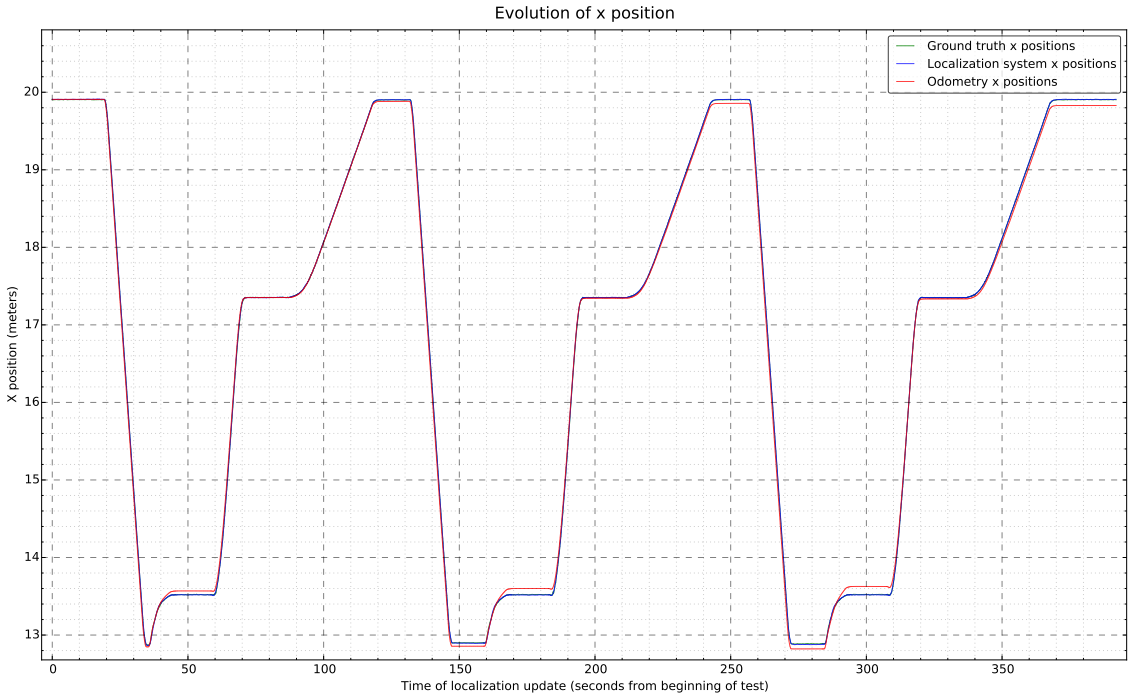


Figure A.1.6.6: Evolution of x position over time

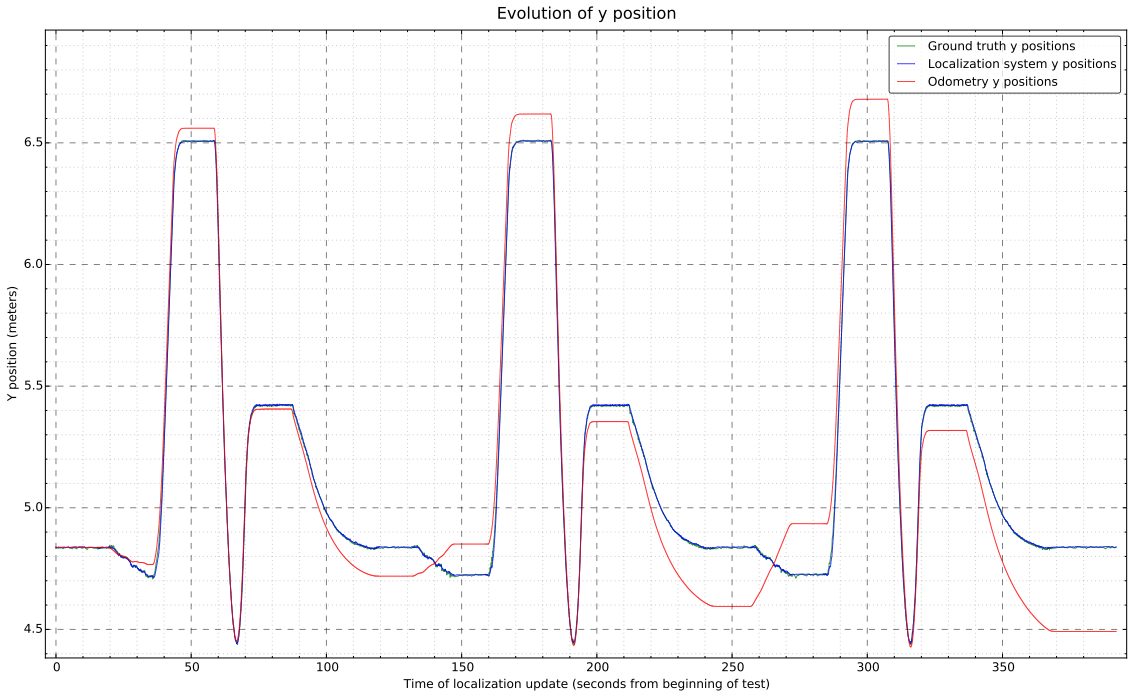


Figure A.1.6.7: Evolution of y position over time

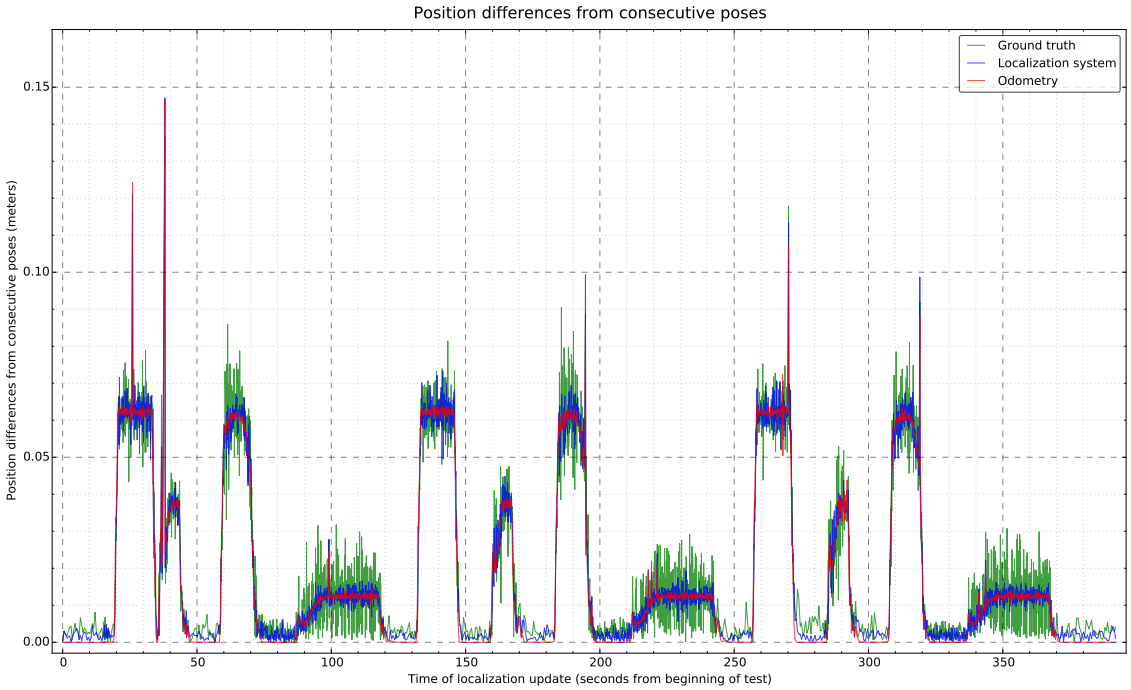


Figure A.1.6.8: Distance traveled between consecutive pose estimations

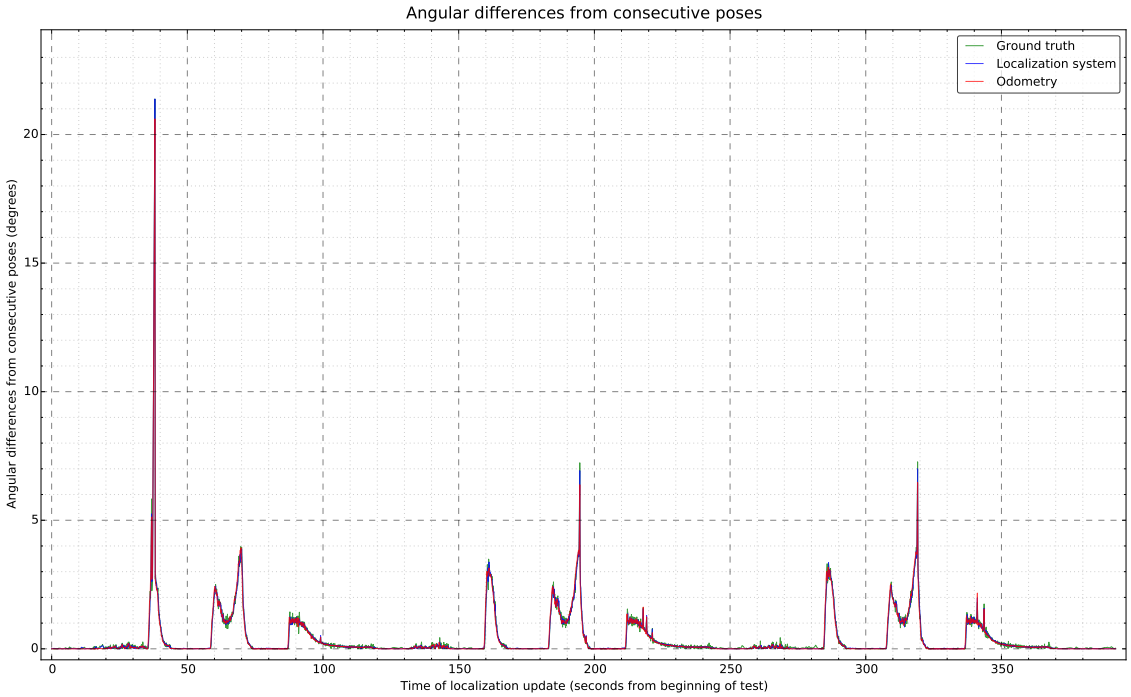


Figure A.1.6.9: Angular differences between consecutive pose estimations

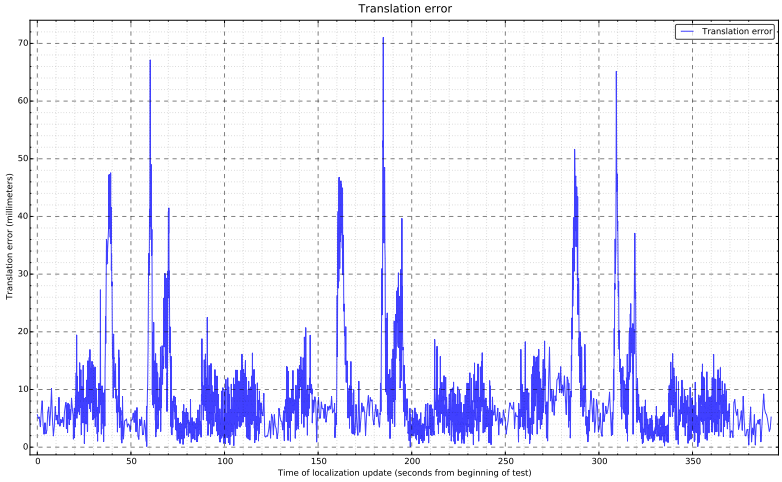


Figure A.1.6.10: Localization system translation errors

Figure A.1.6.11: Localization system translation errors components

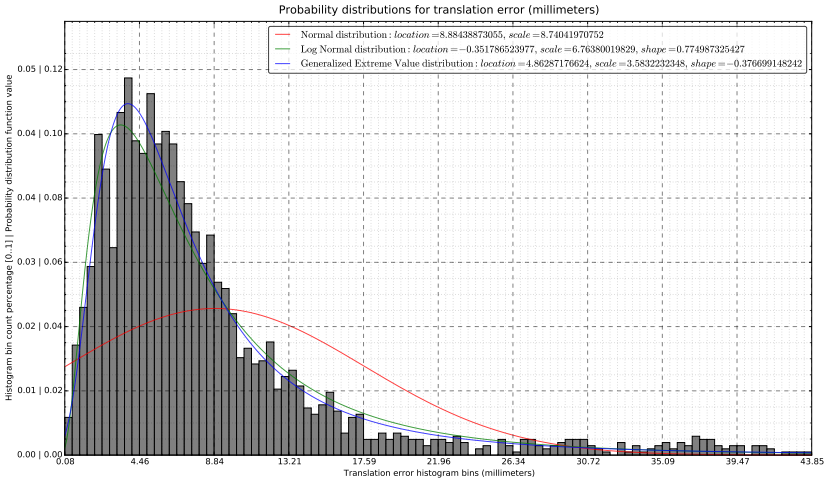


Figure A.1.6.12: Probability distributions for the localization system translation errors

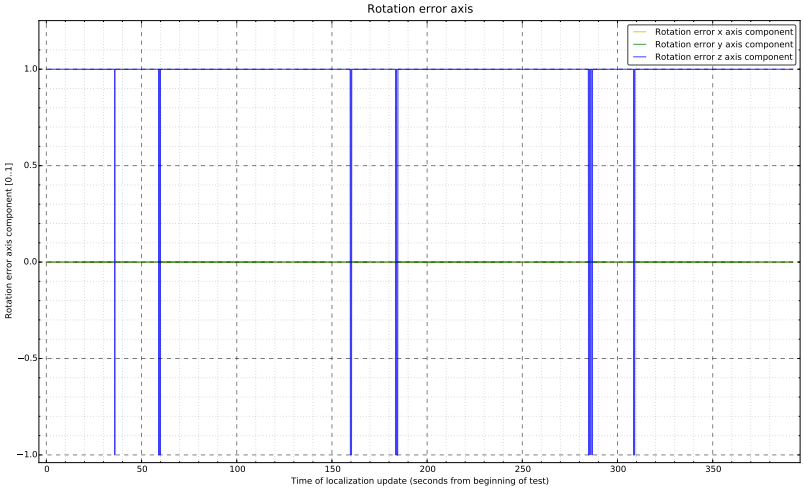


Figure A.1.6.13: Localization system rotation errors axis

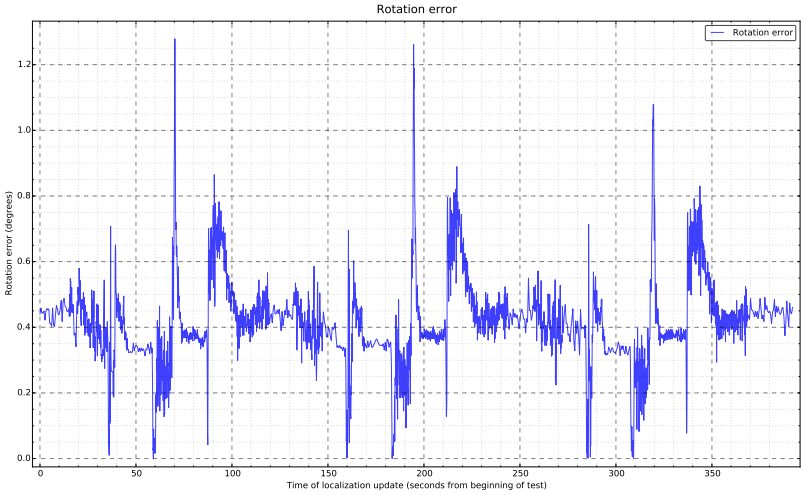


Figure A.1.6.14: Localization system rotation errors

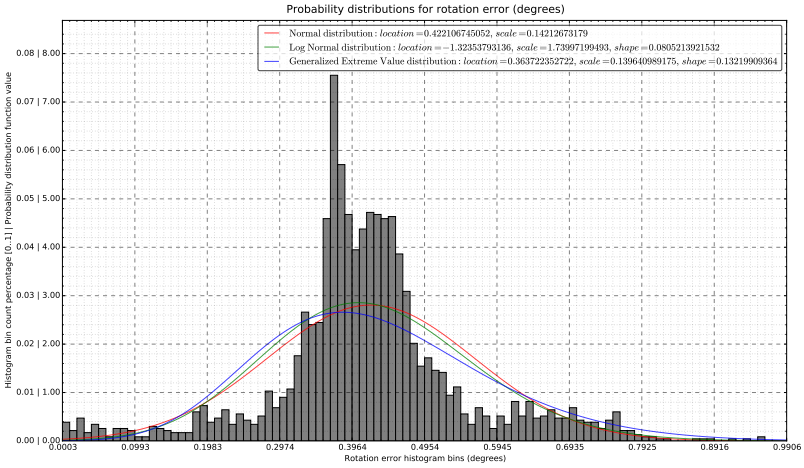


Figure A.1.6.15: Probability distributions for the localization system rotation errors

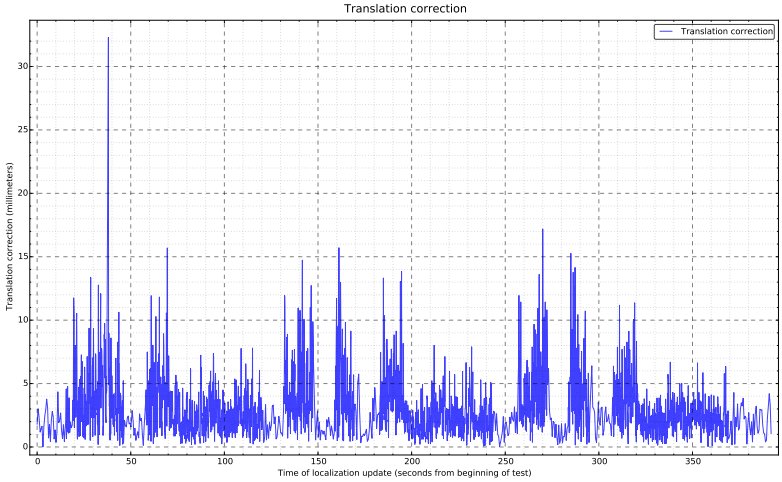


Figure A.1.6.16: Translation corrections performed by the localization system

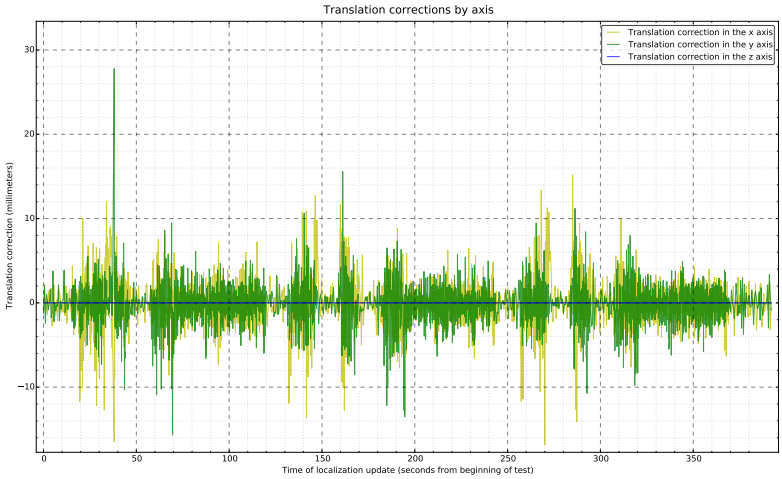


Figure A.1.6.17: Translation corrections components performed by the localization system

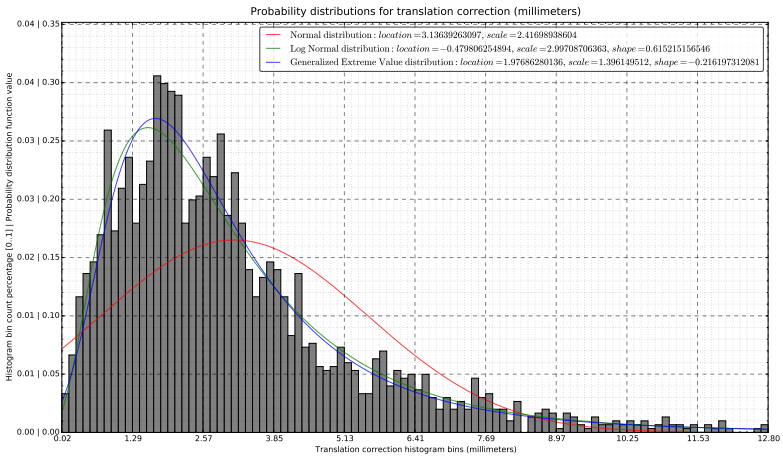


Figure A.1.6.18: Probability distributions for the translation corrections performed by the localization system

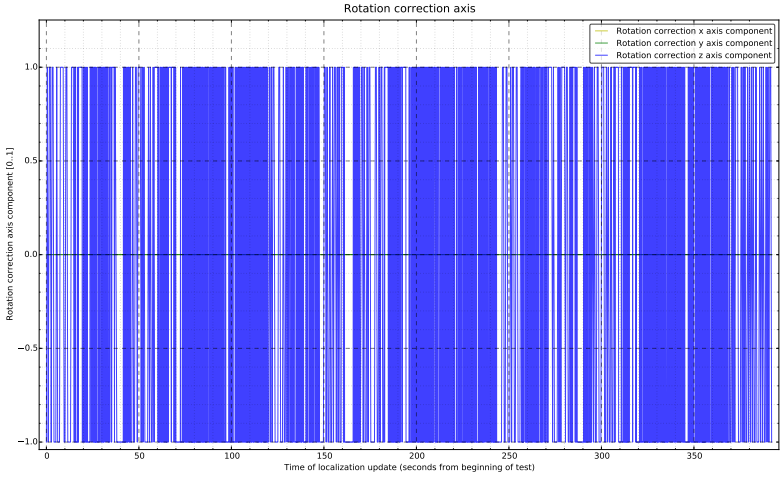


Figure A.1.6.19: Rotation corrections (axis) performed by the localization system

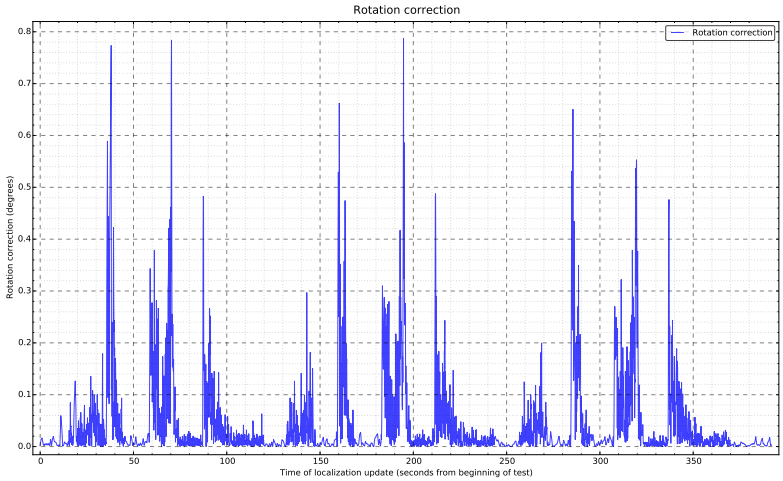


Figure A.1.6.20: Rotation corrections performed by the localization system

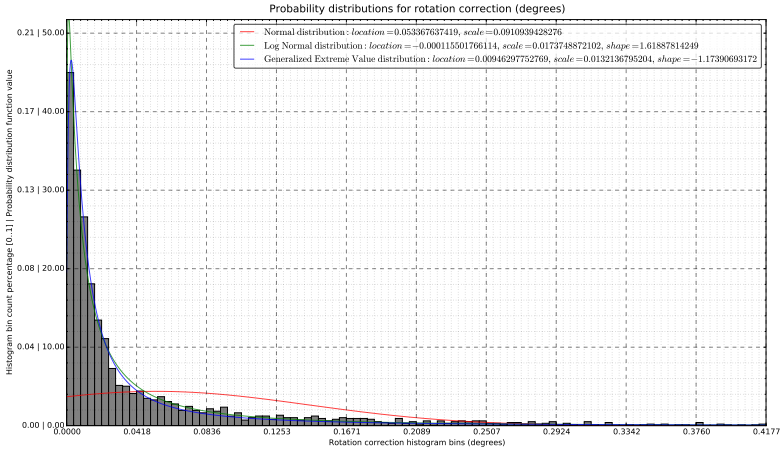


Figure A.1.6.21: Probability distributions for the rotation corrections performed by the localization system

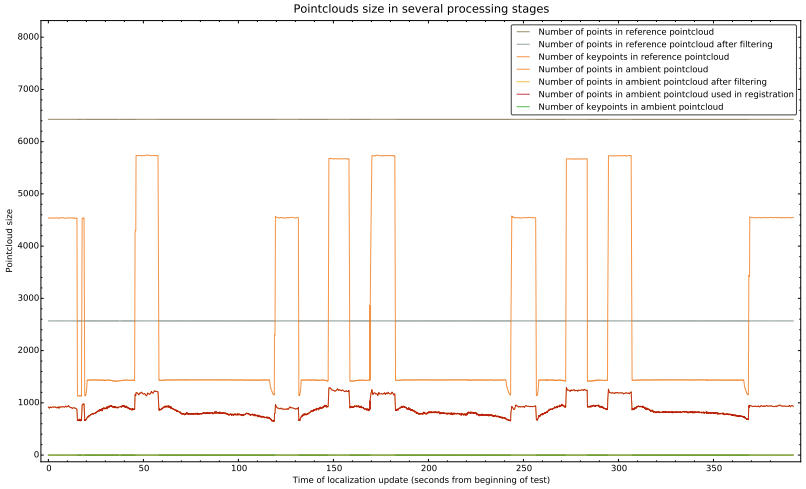


Figure A.1.6.22: Point clouds size in several of the localization system processing stages

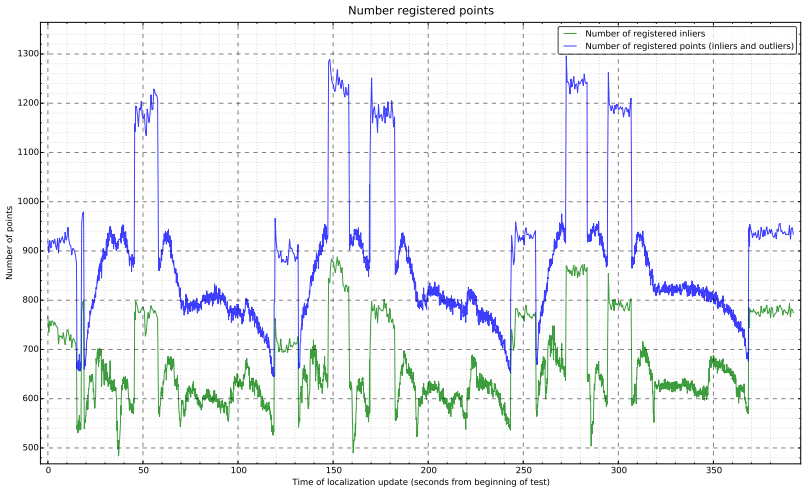


Figure A.1.6.23: Number of registered points

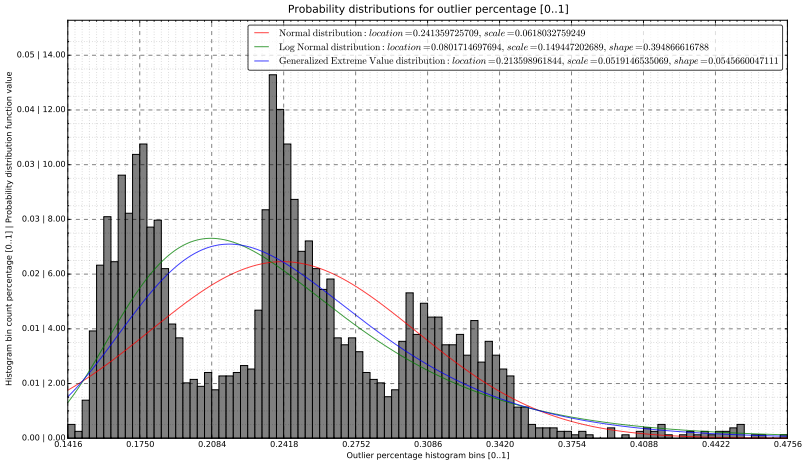


Figure A.1.6.24: Probability distributions for the ambient point cloud outlier percentage

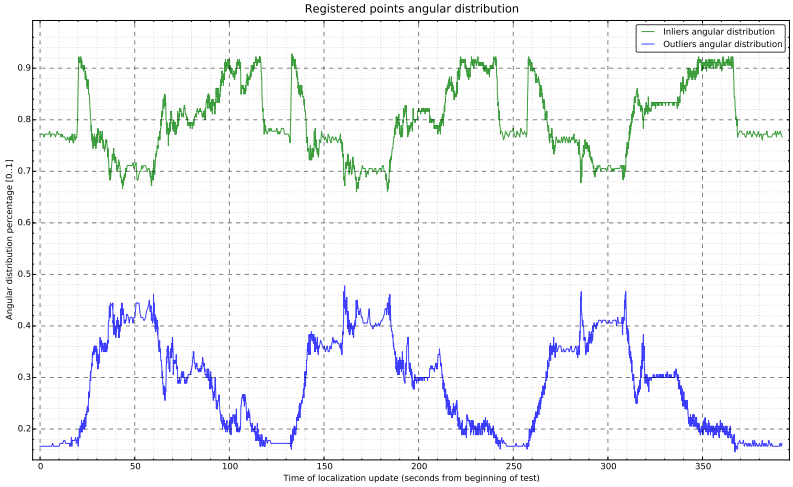


Figure A.1.6.25: Angular distribution percentage of registered points

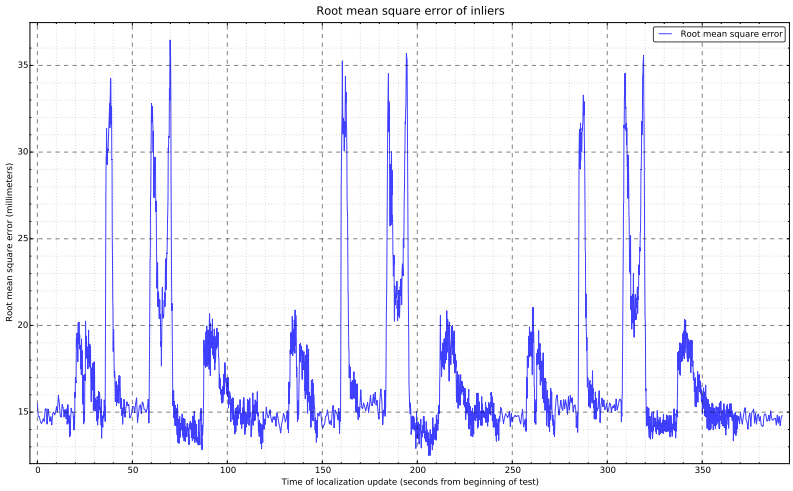


Figure A.1.6.26: Root Mean Square Error of the inliers

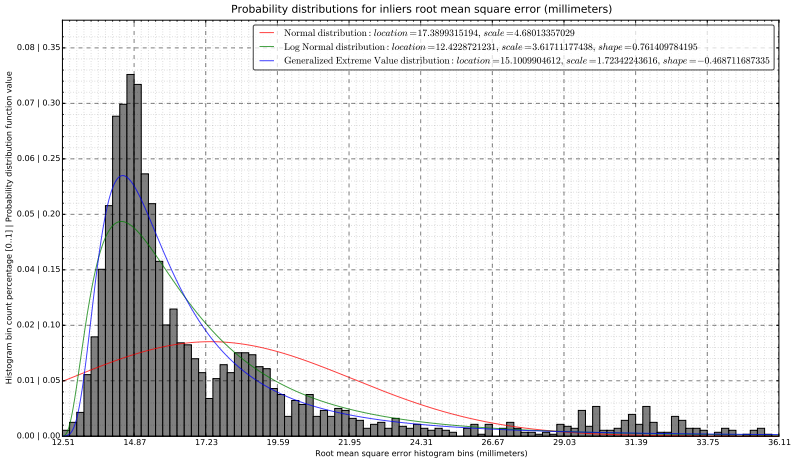


Figure A.1.6.27: Probability distributions for the Root Mean Square Error of the inliers

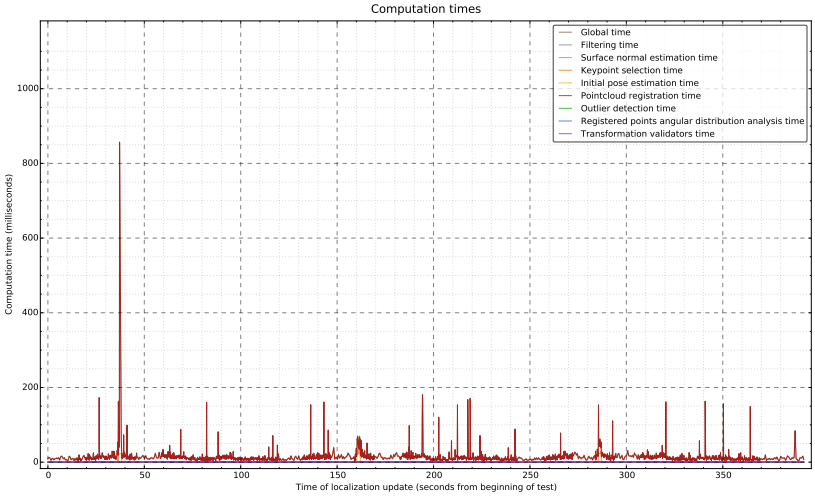


Figure A.1.6.28: Localization system computation times

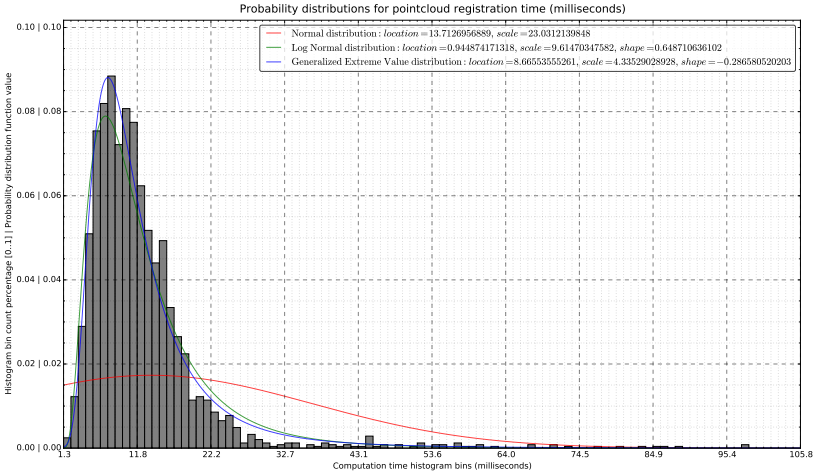


Figure A.1.6.29: Probability distributions for the point cloud registration time

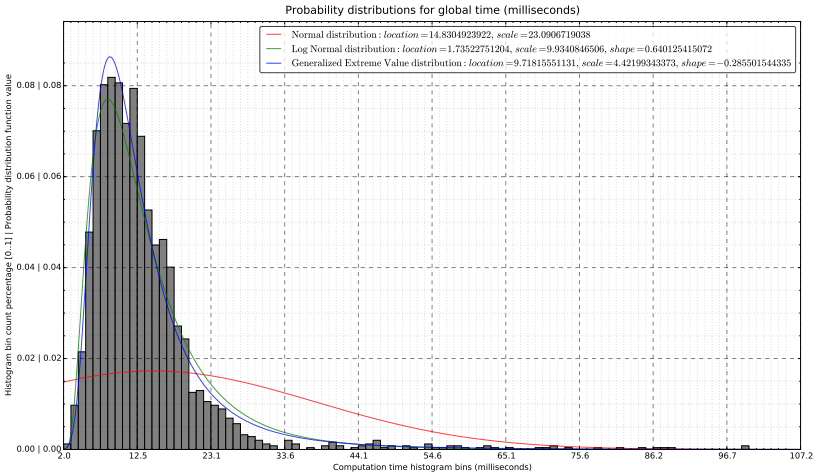


Figure A.1.6.30: Probability distributions for the localization system global computation time

A.2 Pioneer robot tests

The next subsections show some of the most relevant results retrieved with the Pioneer robot in the industrial hall.

A.2.1 360 path using the Pioneer robot at 23 cm/s

The figures in this section show the detailed results of test performed with the Pioneer robot with a 25 mm resolution map in the industrial hall following a 360^{o} path and moving at 23 cm/s.

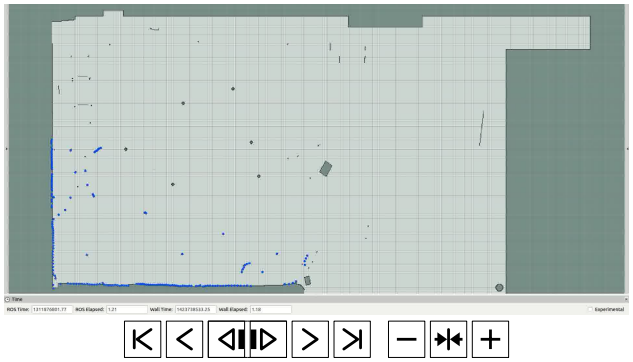


Figure A.2.1.1: Animation of laser scans using the ground truth poses

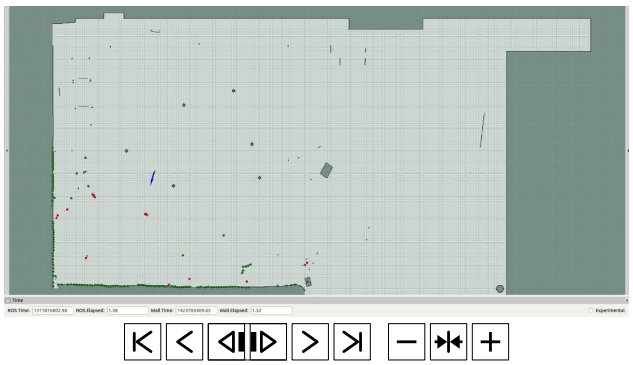


Figure A.2.1.2: Animation of laser scans using the localization system poses

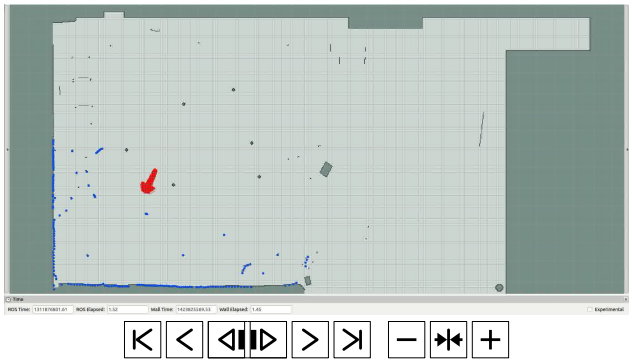


Figure A.2.1.3: Animation of laser scans using the AMCL poses

Figure A.2.1.4: Laser scans assembled on top of the map using the ground truth poses

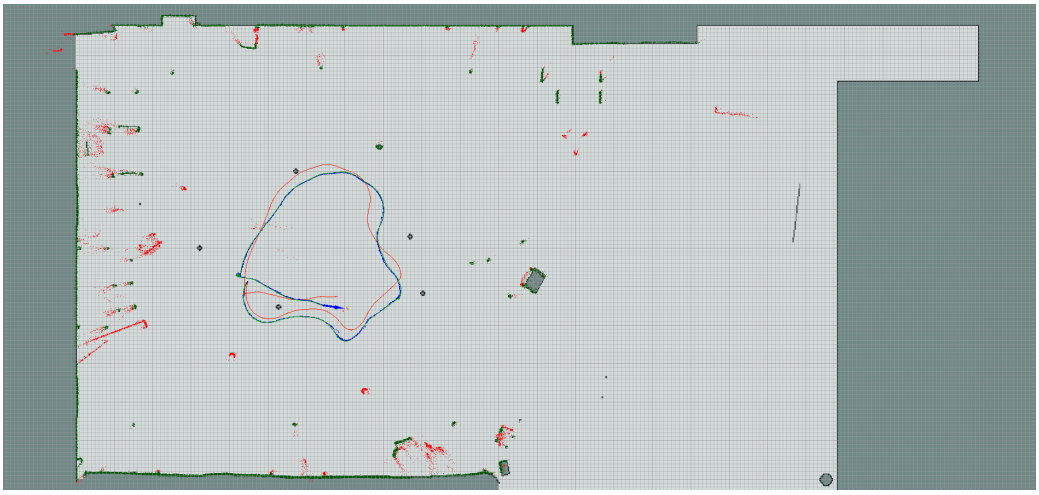


Figure A.2.1.5: Laser scans assembled on top of the map using the localization system poses

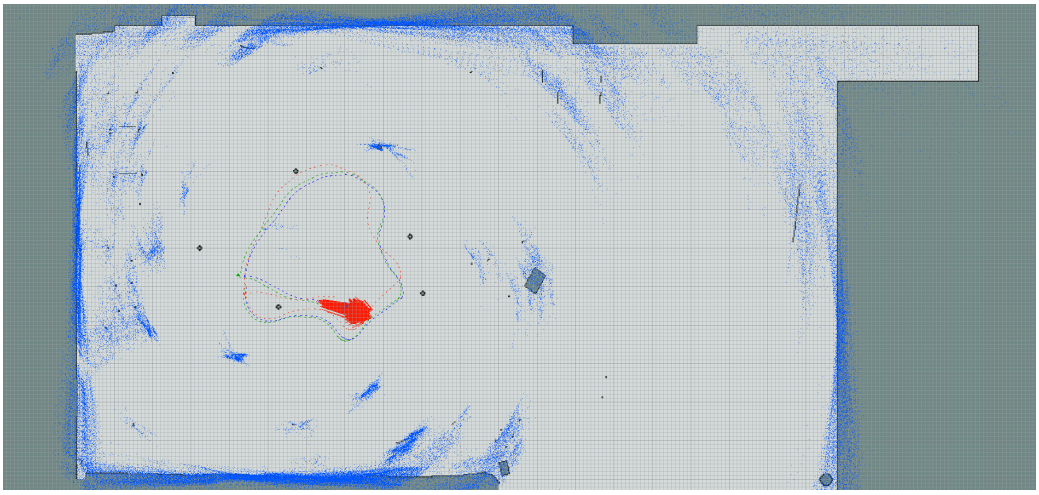


Figure A.2.1.6: Laser scans assembled on top of the map using the AMCL poses

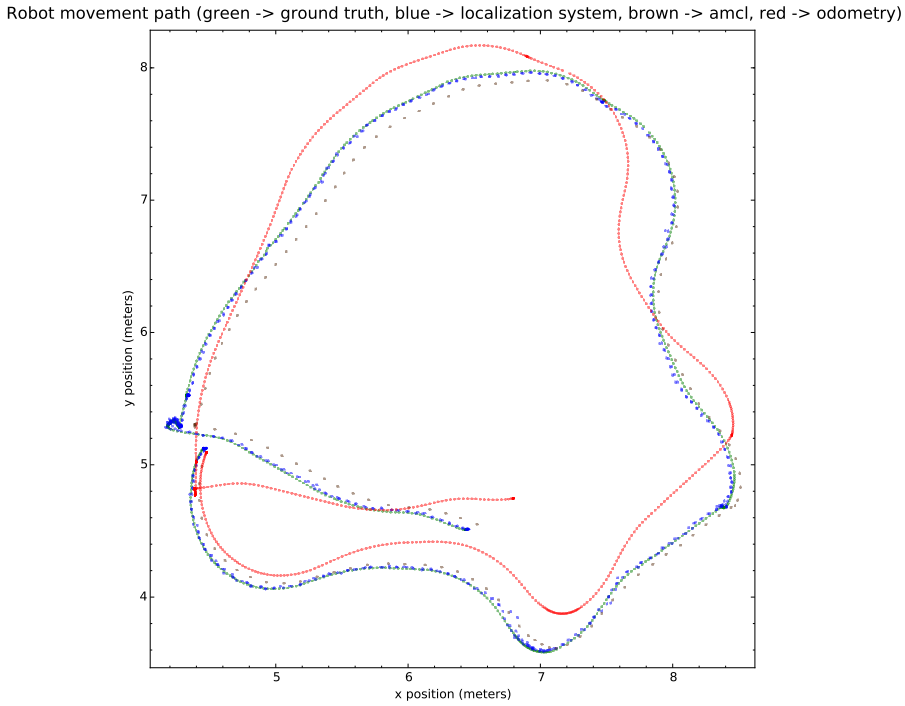


Figure A.2.1.7: Poses estimated by the ground truth, localization system, AMCL and odometry

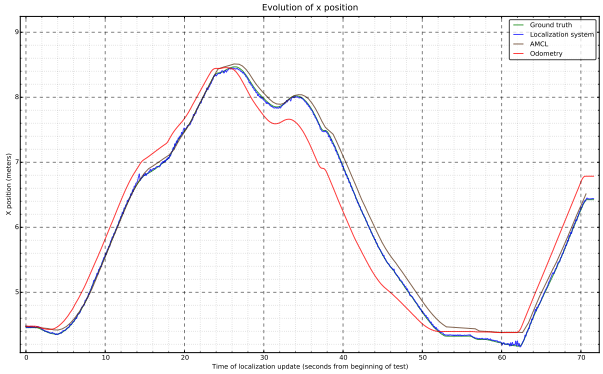


Figure A.2.1.8: Evolution of x position over time

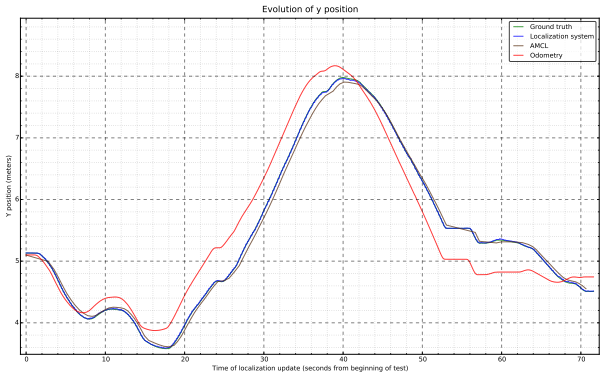


Figure A.2.1.9: Evolution of y position over time

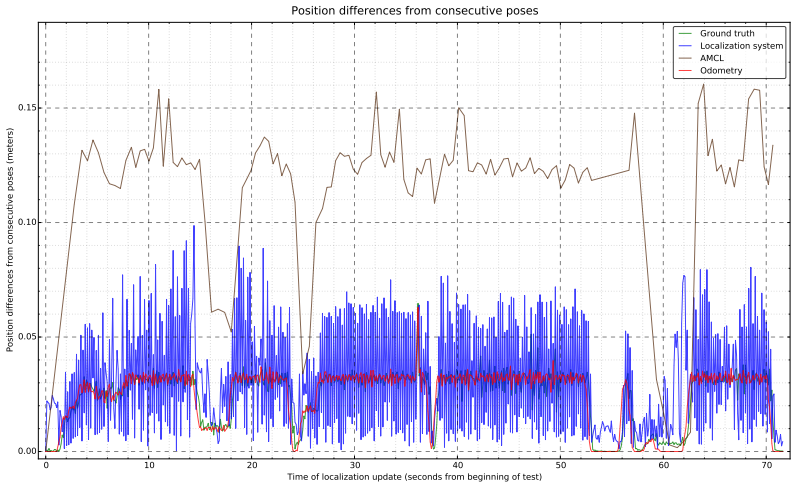


Figure A.2.1.10: Distance traveled between consecutive pose estimations

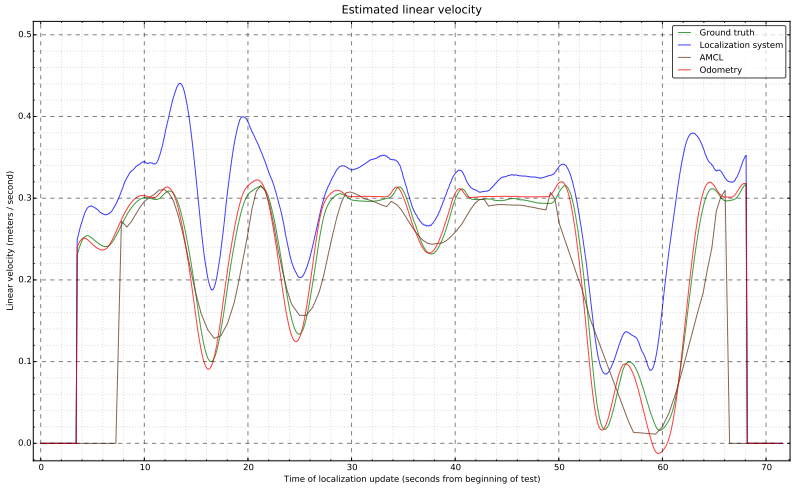


Figure A.2.1.11: Estimated linear velocity

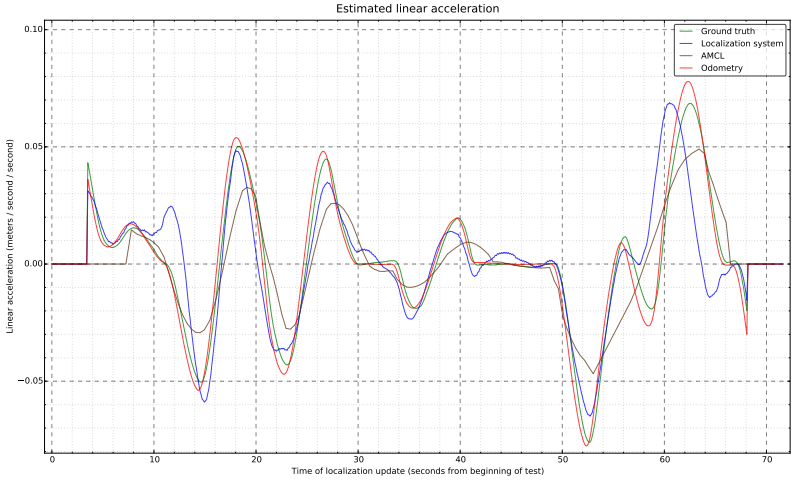


Figure A.2.1.12: Estimated linear acceleration

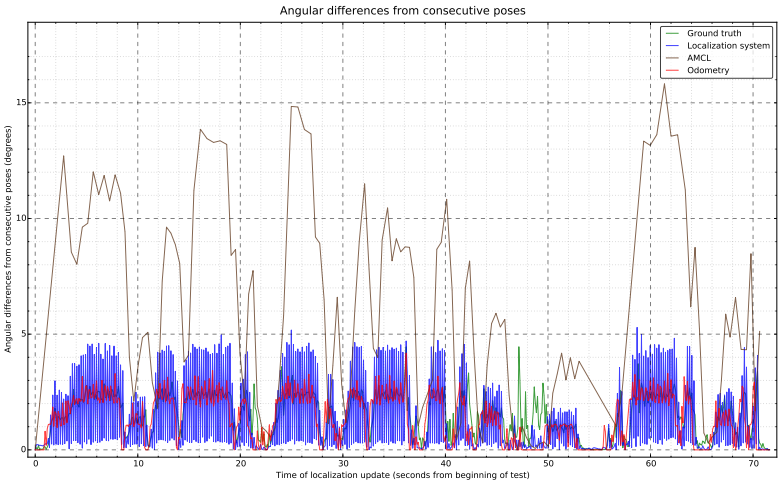


Figure A.2.1.13: Angular differences between consecutive pose estimations

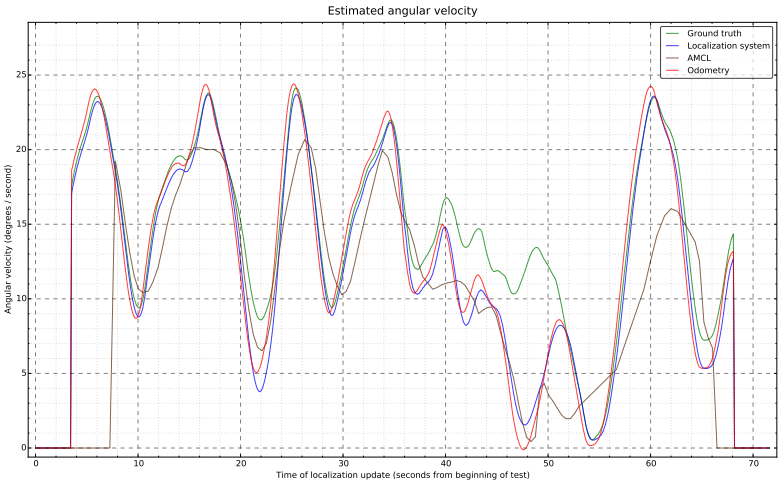


Figure A.2.1.14: Estimated angular velocity

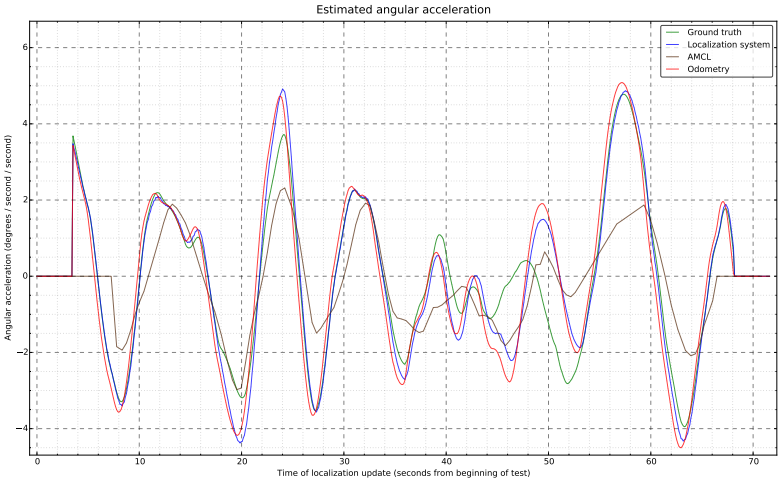


Figure A.2.1.15: Estimated angular acceleration

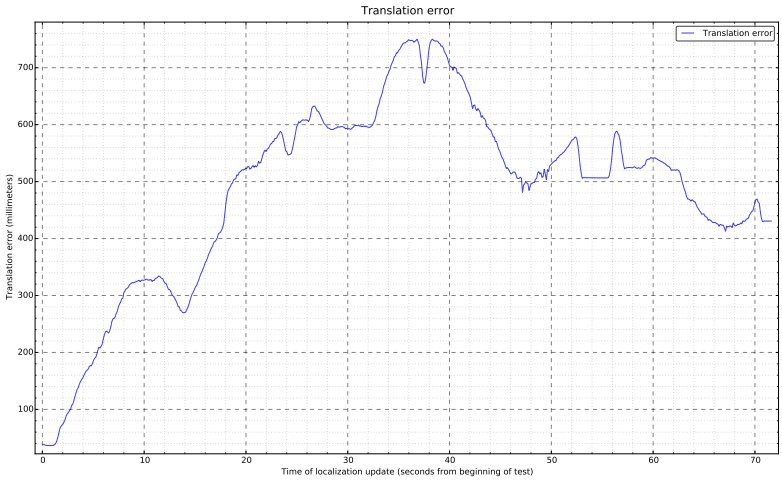


Figure A.2.1.16: Odometry translation errors

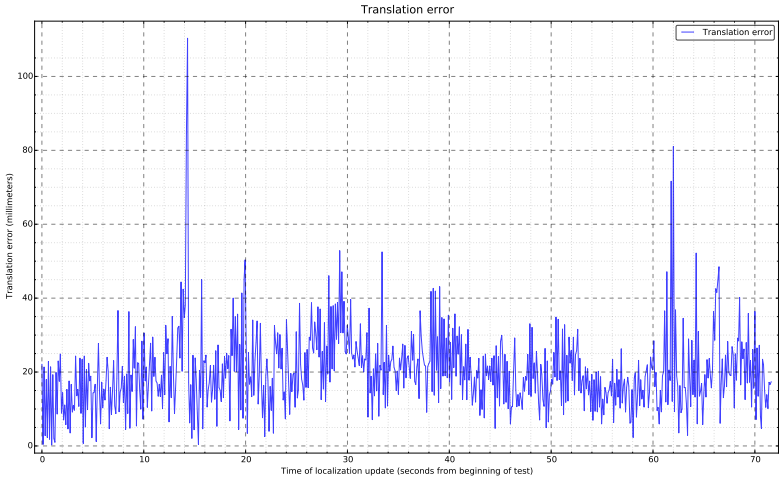


Figure A.2.1.17: Localization system translation errors

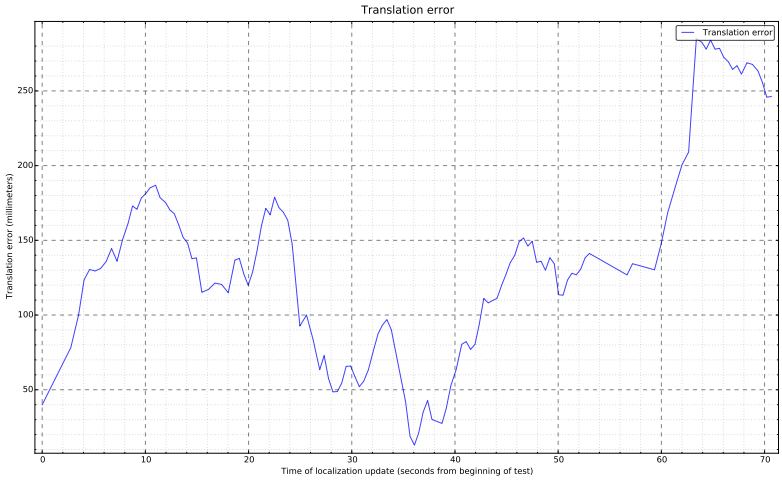


Figure A.2.1.18: AMCL translation errors

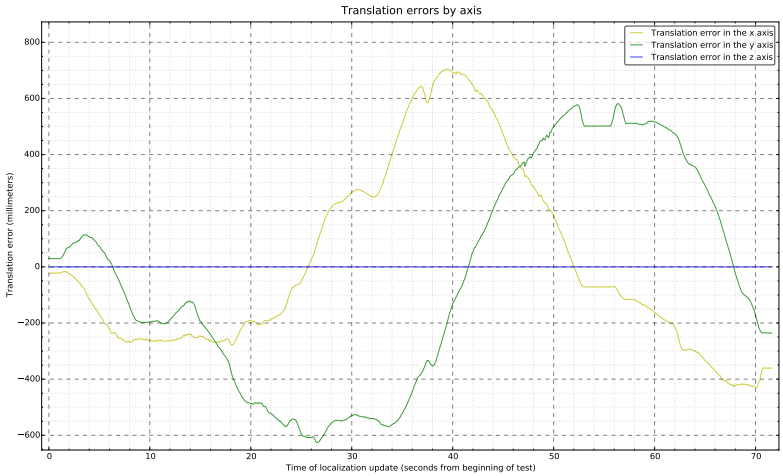


Figure A.2.1.19: Odometry translation errors components

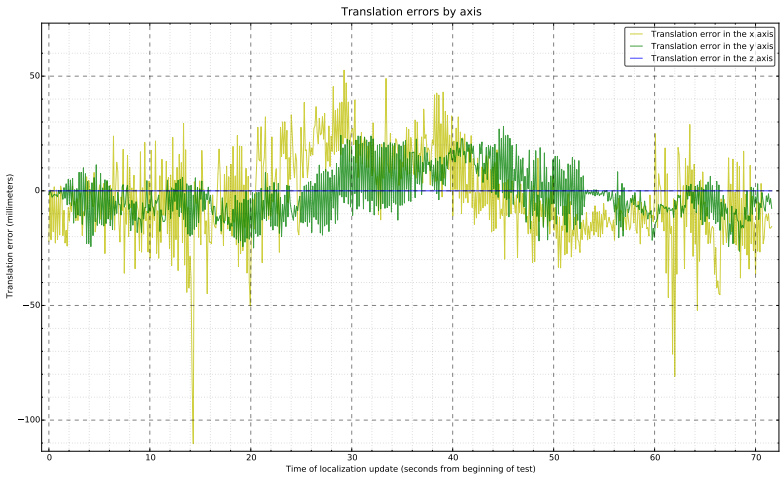


Figure A.2.1.20: Localization system translation errors components

Figure A.2.1.21: AMCL translation errors components

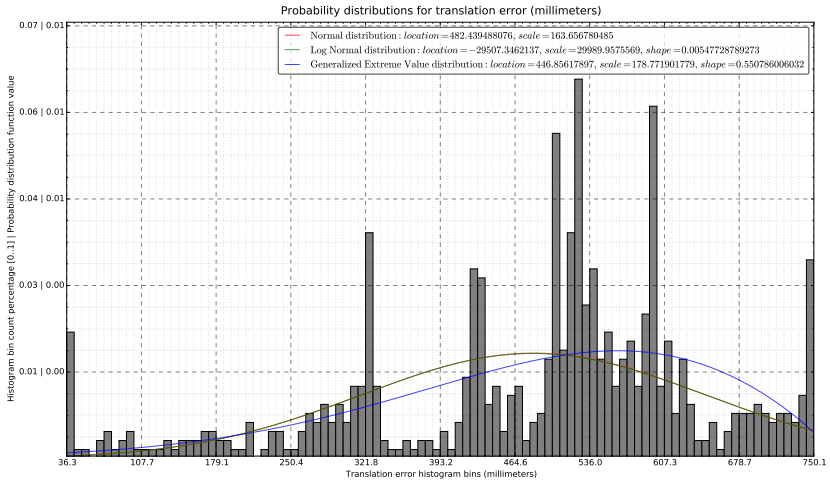


Figure A.2.1.22: Probability distributions for the odometry translation errors

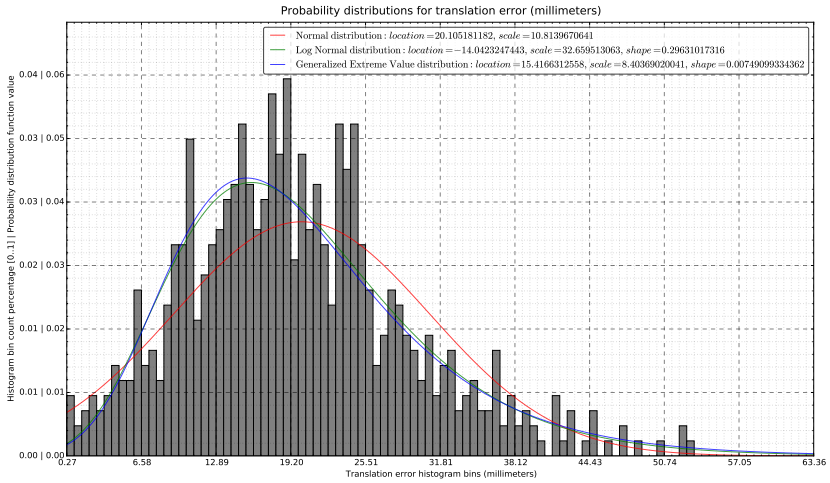


Figure A.2.1.23: Probability distributions for the localization system translation errors

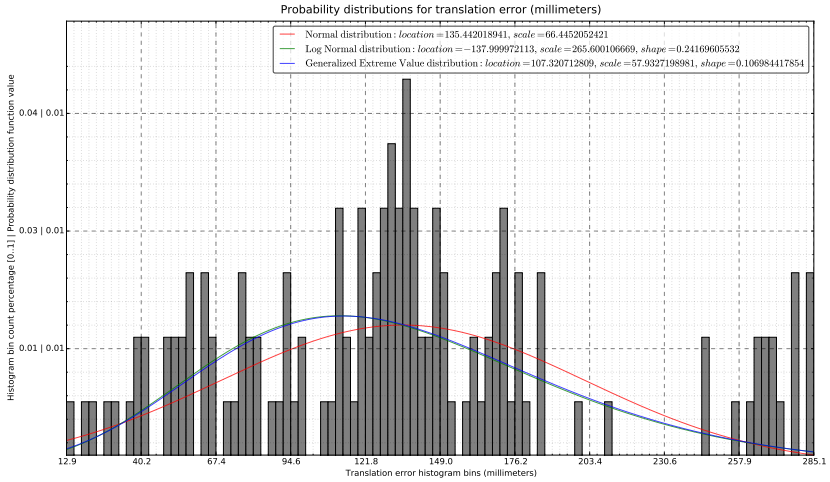


Figure A.2.1.24: Probability distributions for the AMCL translation errors

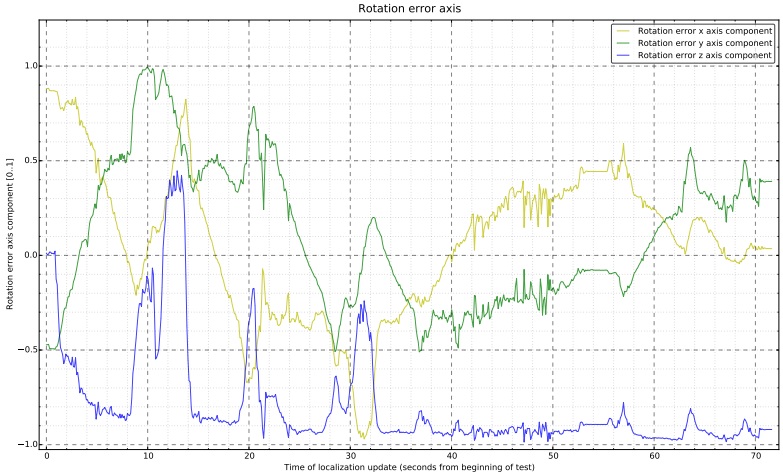


Figure A.2.1.25: Odometry rotation errors axis

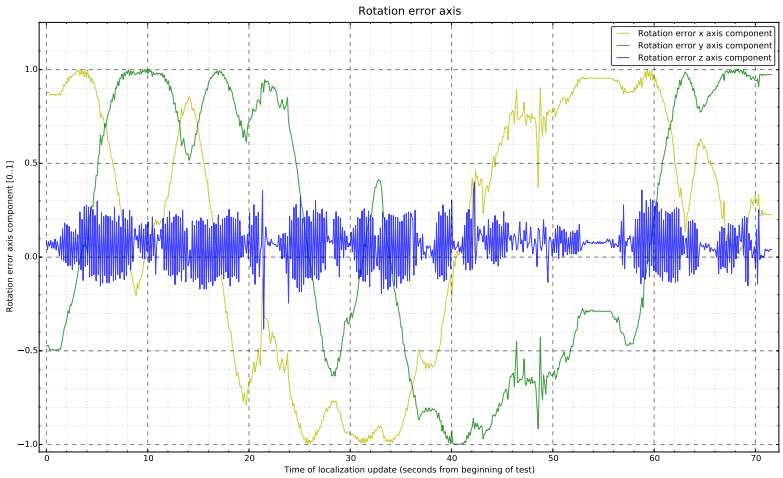


Figure A.2.1.26: Localization system rotation errors axis

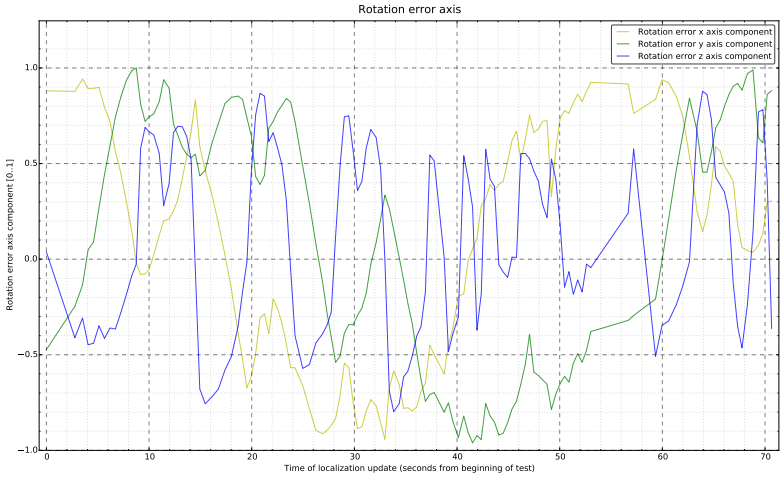


Figure A.2.1.27: AMCL rotation errors axis

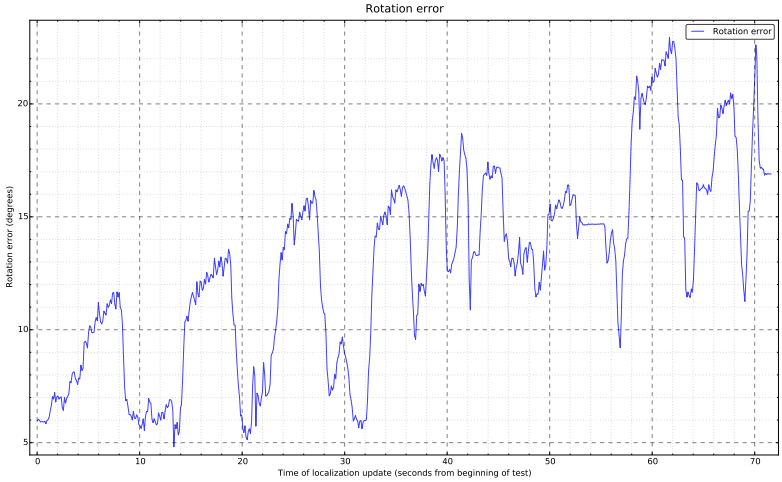


Figure A.2.1.28: Odometry rotation errors

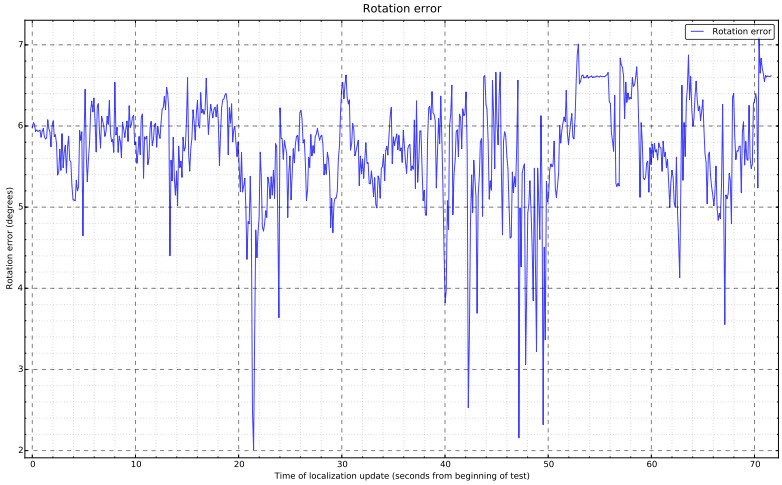


Figure A.2.1.29: Localization system rotation errors

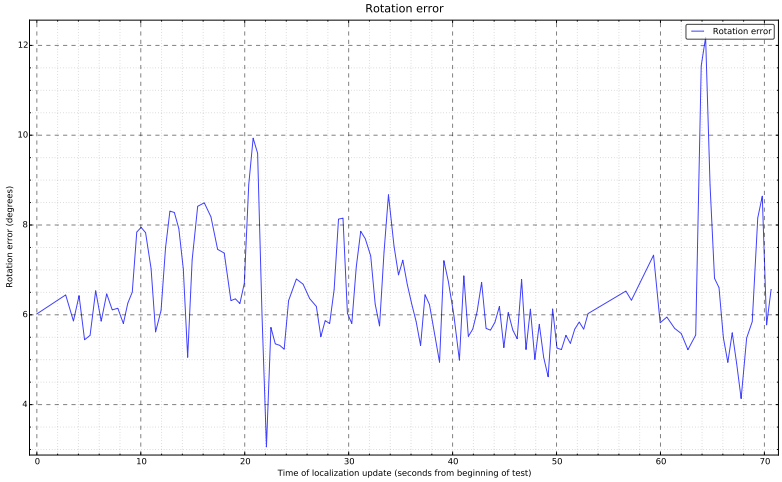


Figure A.2.1.30: AMCL rotation errors

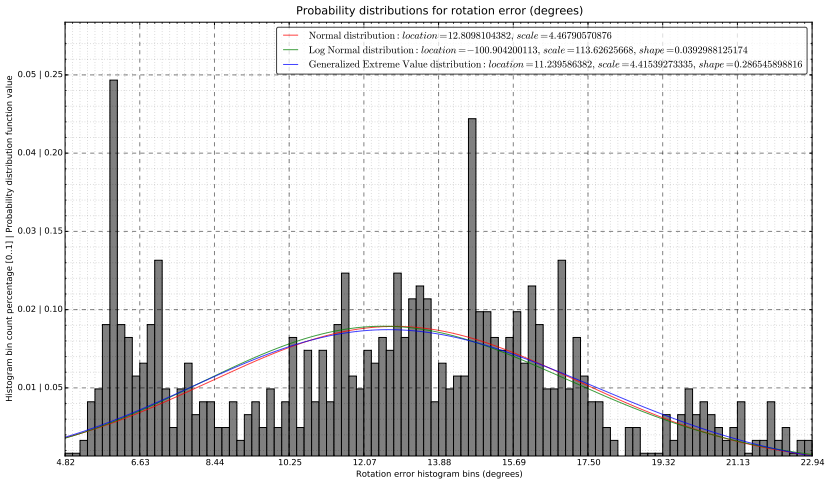


Figure A.2.1.31: Probability distributions for the odometry rotation errors

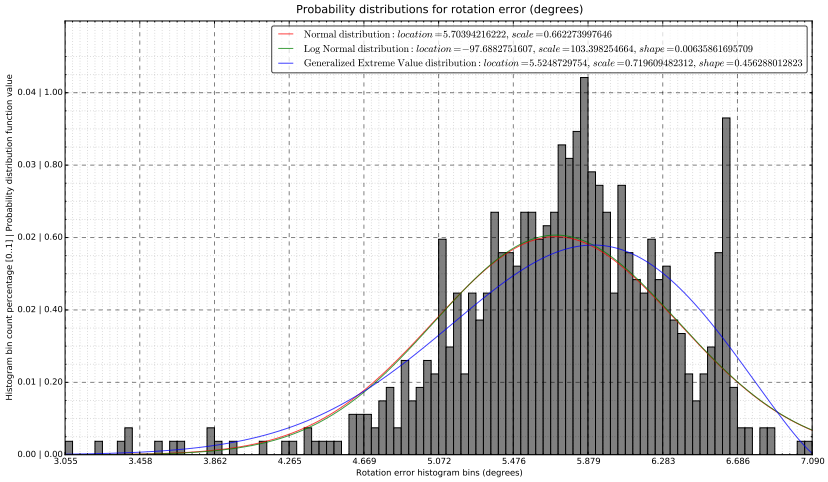


Figure A.2.1.32: Probability distributions for the localization system rotation errors

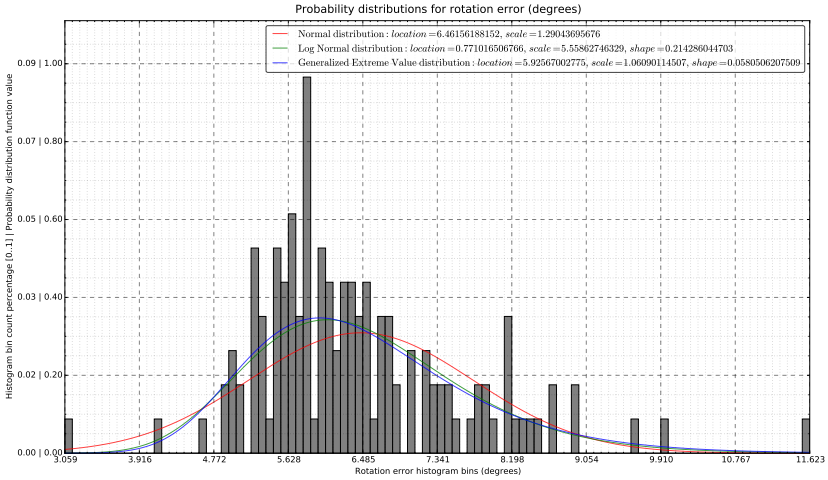


Figure A.2.1.33: Probability distributions for the AMCL rotation errors

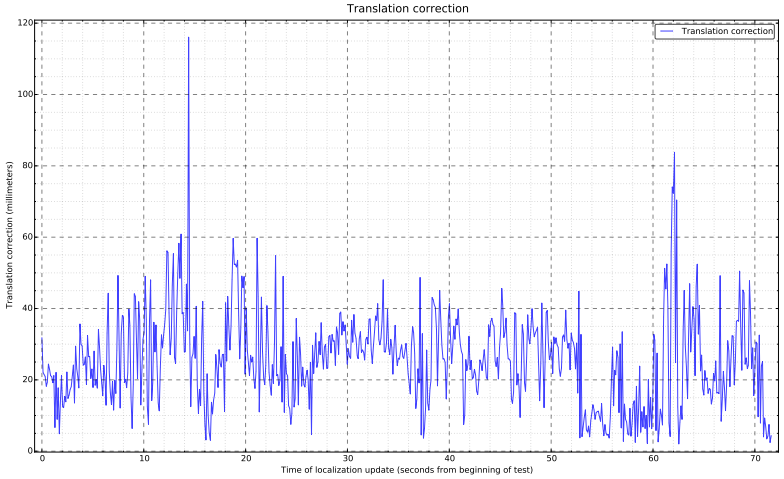


Figure A.2.1.34: Translation corrections performed by the localization system

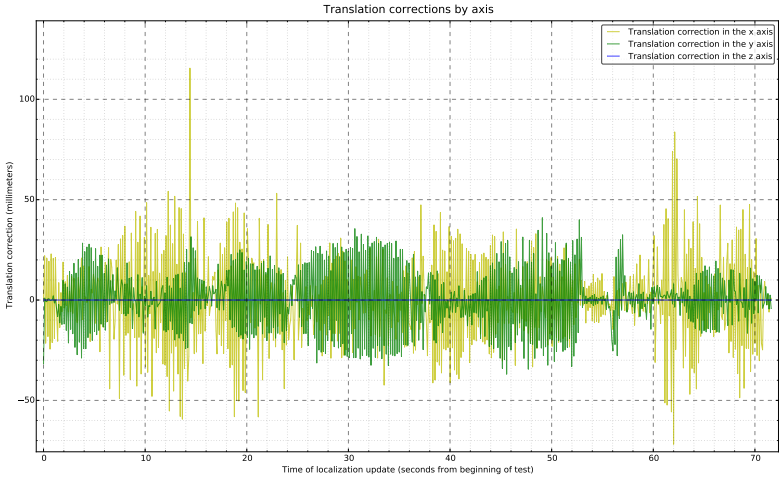


Figure A.2.1.35: Translation corrections components performed by the localization system

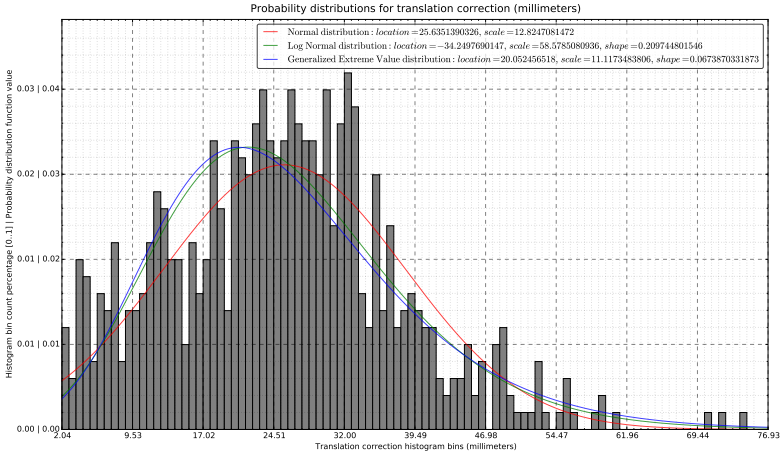


Figure A.2.1.36: Probability distributions for the translation corrections performed by the localization system

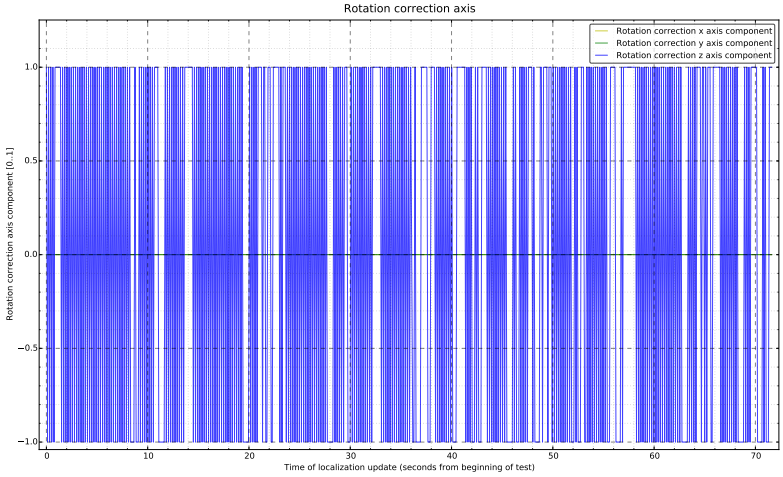


Figure A.2.1.37: Rotation corrections (axis) performed by the localization system

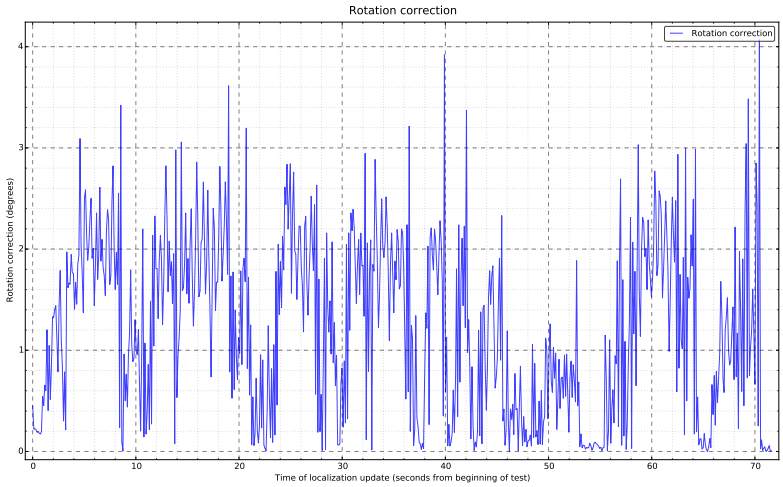


Figure A.2.1.38: Rotation corrections performed by the localization system

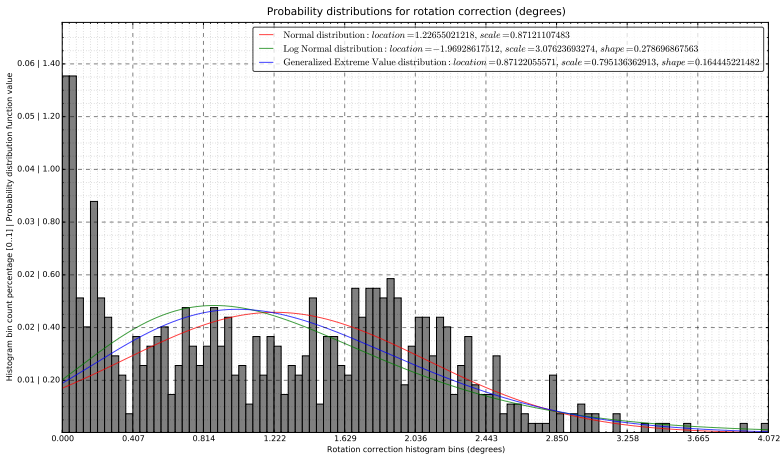


Figure A.2.1.39: Probability distributions for the rotation corrections performed by the localization system

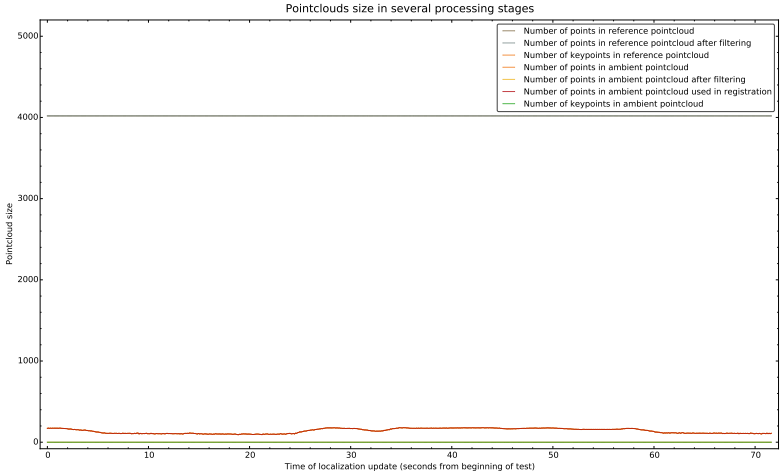


Figure A.2.1.40: Point clouds size in several of the localization system processing stages

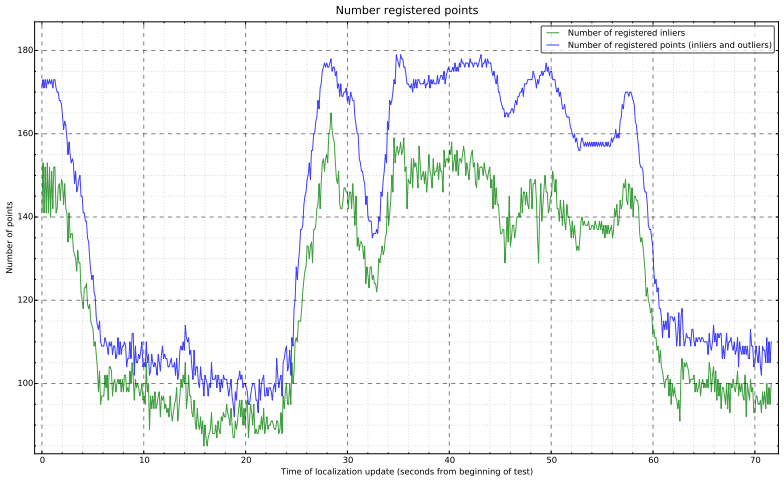


Figure A.2.1.41: Number of registered points

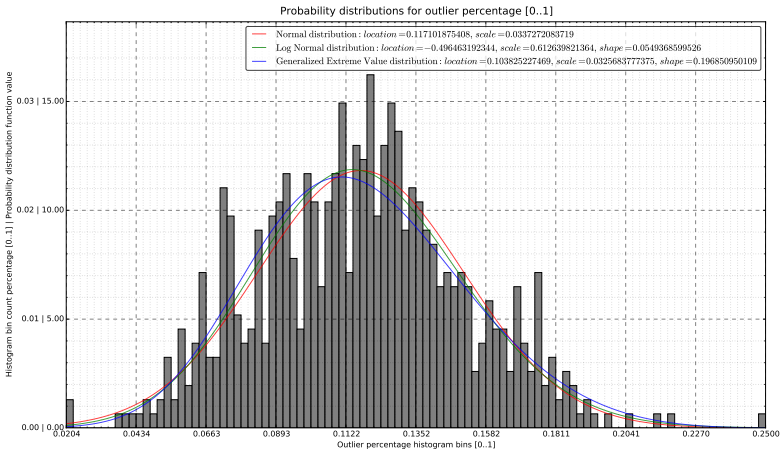


Figure A.2.1.42: Probability distributions for the ambient point cloud outlier percentage

Figure A.2.1.43: Angular distribution percentage of registered points

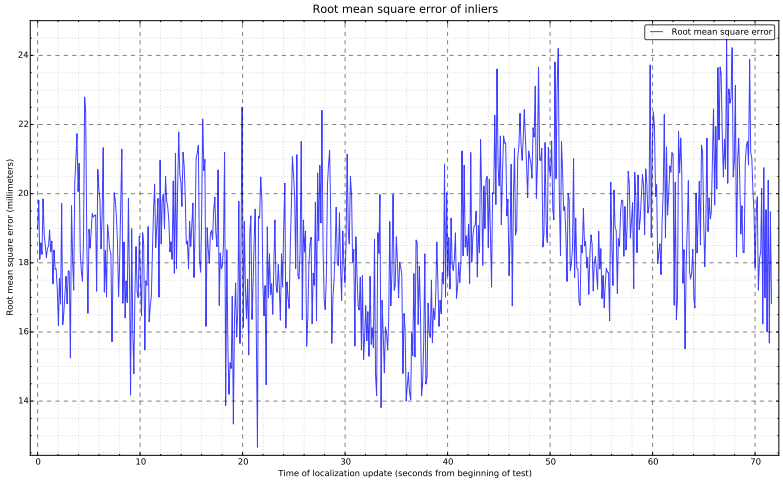


Figure A.2.1.44: Root Mean Square Error of the inliers

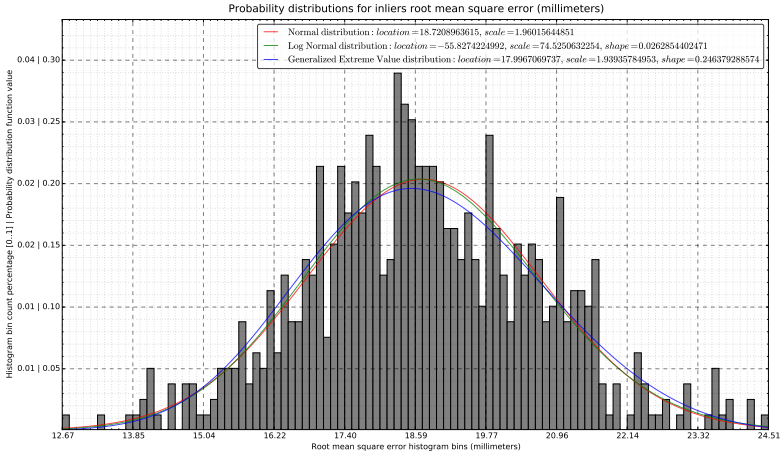


Figure A.2.1.45: Probability distributions for the Root Mean Square Error of the inliers

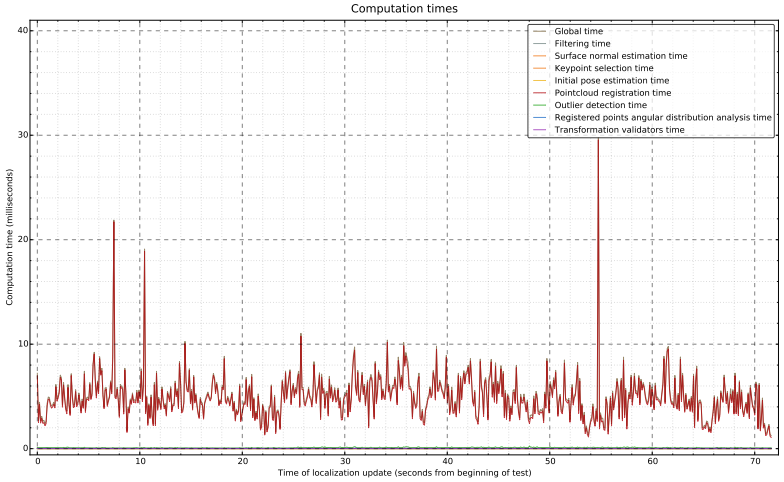


Figure A.2.1.46: Localization system computation times

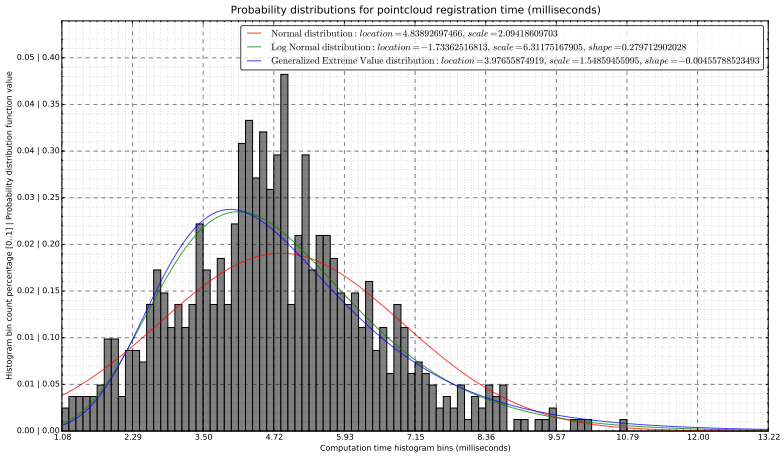


Figure A.2.1.47: Probability distributions for the point cloud registration time

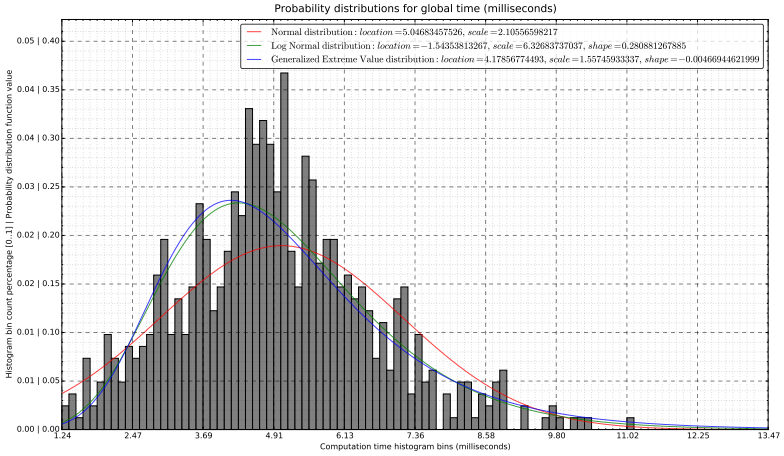


Figure A.2.1.48: Probability distributions for the localization system global computation time

A.2.2 First SLAM path using the Pioneer robot at 26 cm/s

The figures in this section show the detailed results of the test performed with the Pioneer robot with a 25 mm resolution map in the industrial hall following the first SLAM path and moving at 26 cm/s.

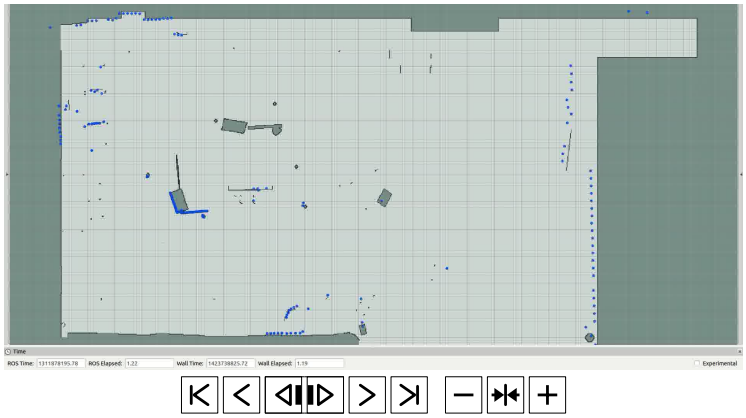


Figure A.2.2.1: Animation of laser scans using the ground truth poses

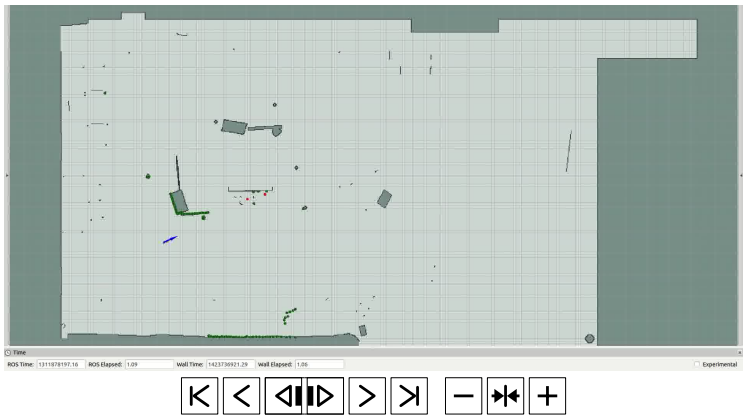


Figure A.2.2.2: Animation of laser scans using the localization system poses

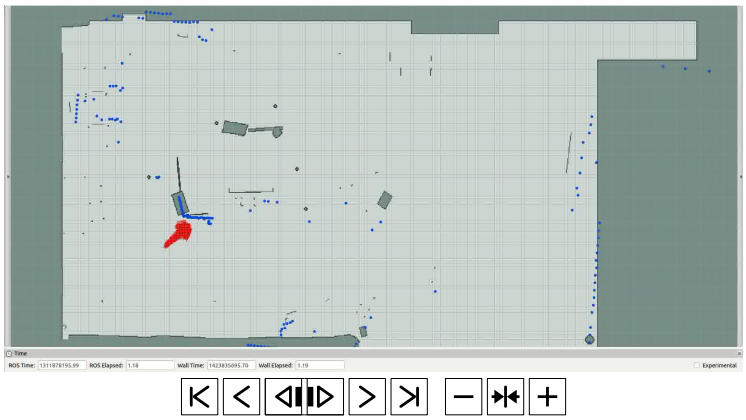


Figure A.2.2.3: Animation of laser scans using the AMCL poses

Figure A.2.2.4: Laser scans assembled on top of the map using the ground truth poses

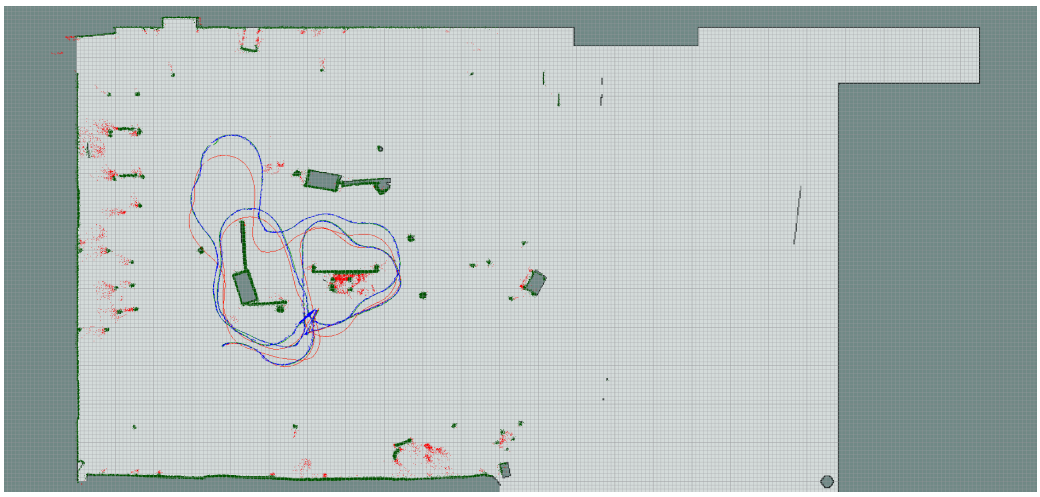


Figure A.2.2.5: Laser scans assembled on top of the map using the localization system poses

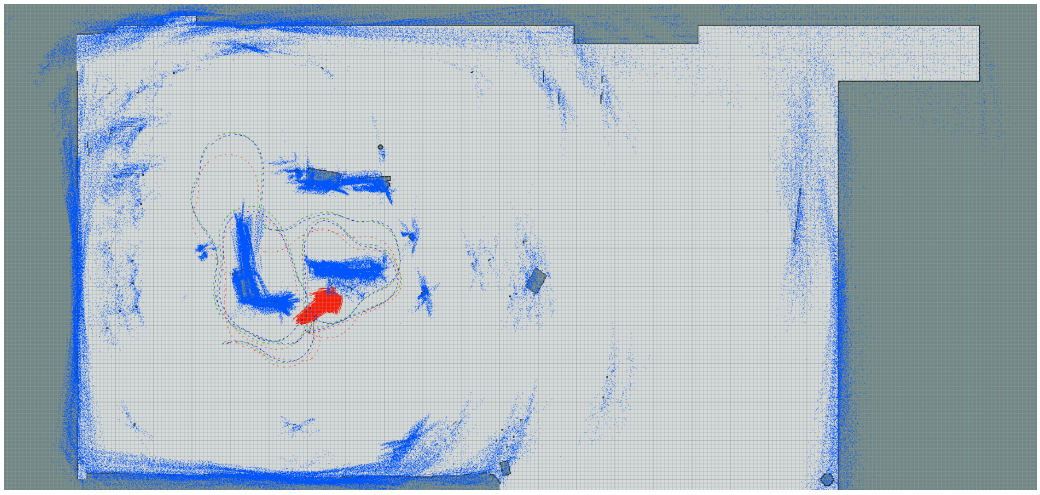


Figure A.2.2.6: Laser scans assembled on top of the map using the AMCL poses

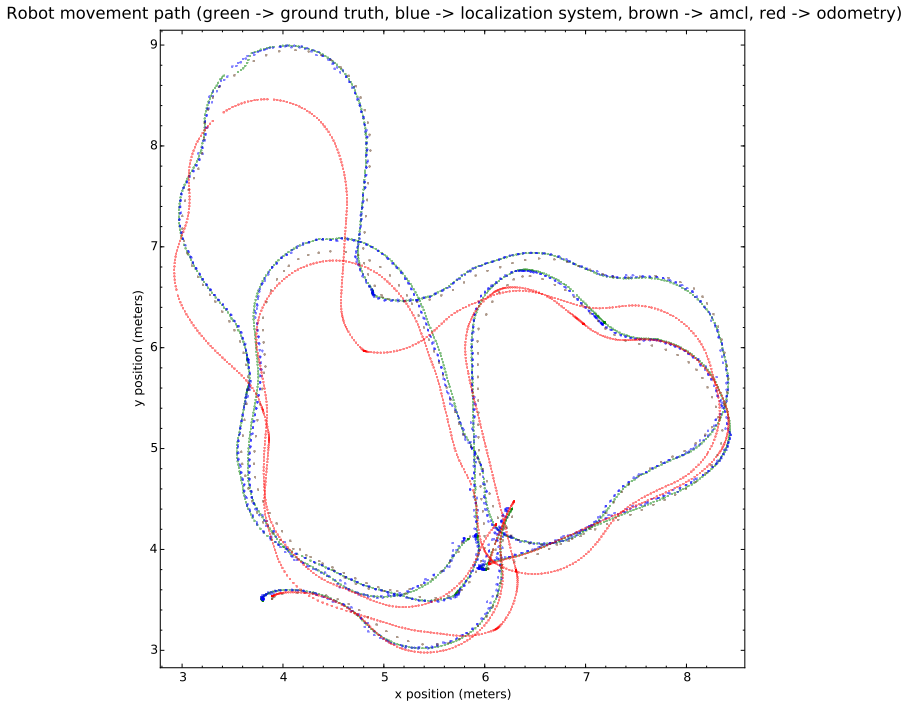


Figure A.2.2.7: Poses estimated by the ground truth, localization system, AMCL and odometry

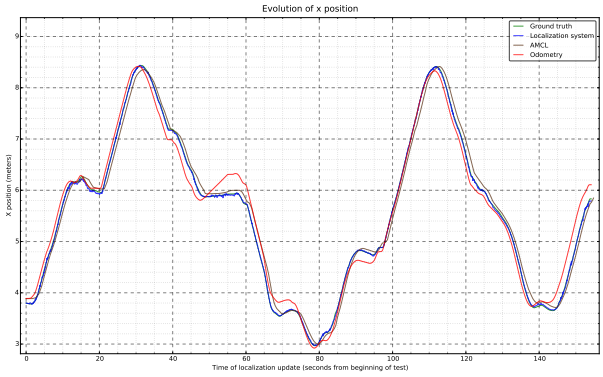


Figure A.2.2.8: Evolution of x position over time

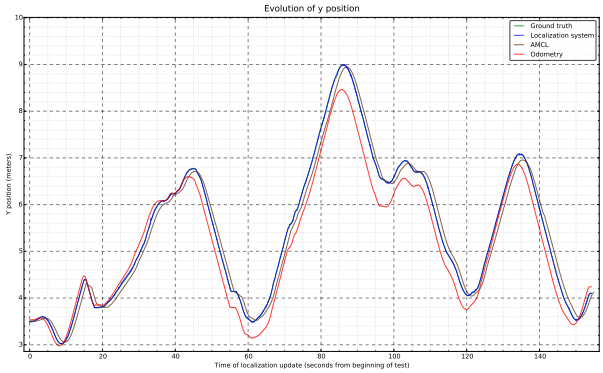


Figure A.2.2.9: Evolution of y position over time

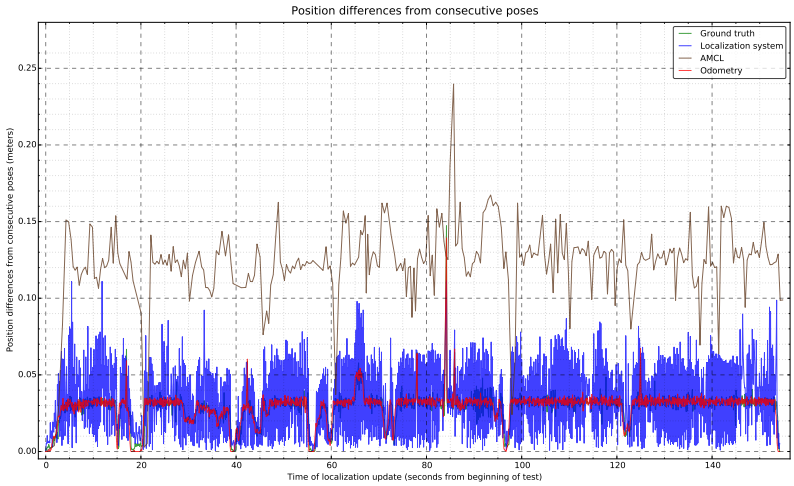


Figure A.2.2.10: Distance traveled between consecutive pose estimations

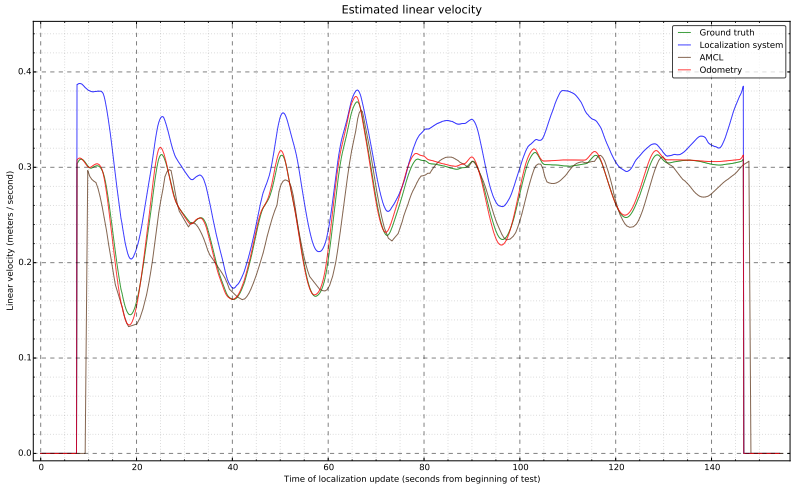


Figure A.2.2.11: Estimated linear velocity

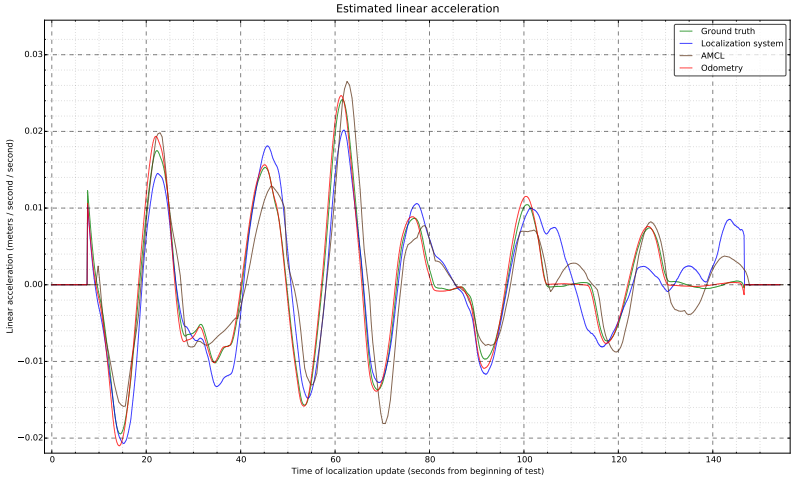


Figure A.2.2.12: Estimated linear acceleration

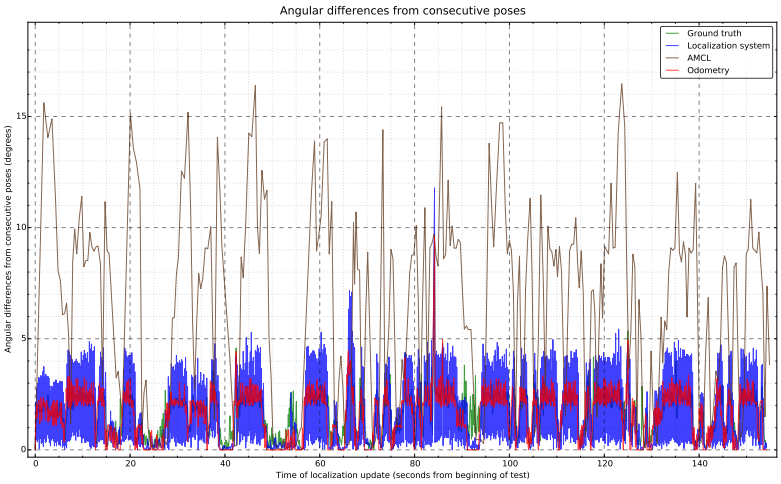


Figure A.2.2.13: Angular differences between consecutive pose estimations

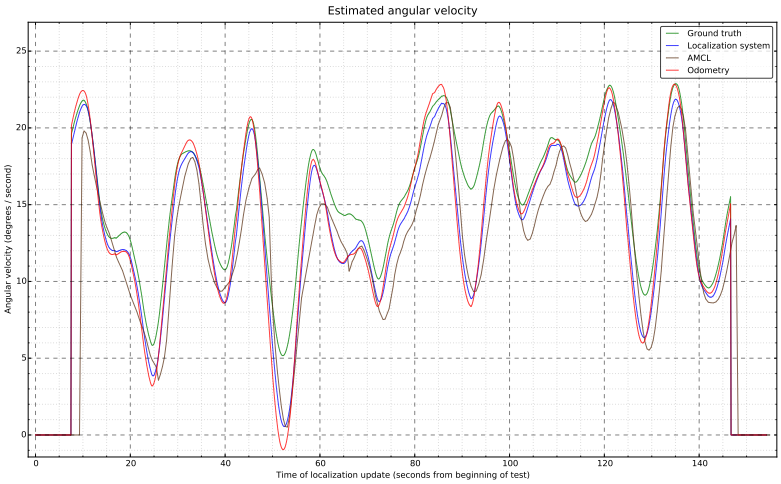


Figure A.2.2.14: Estimated angular velocity

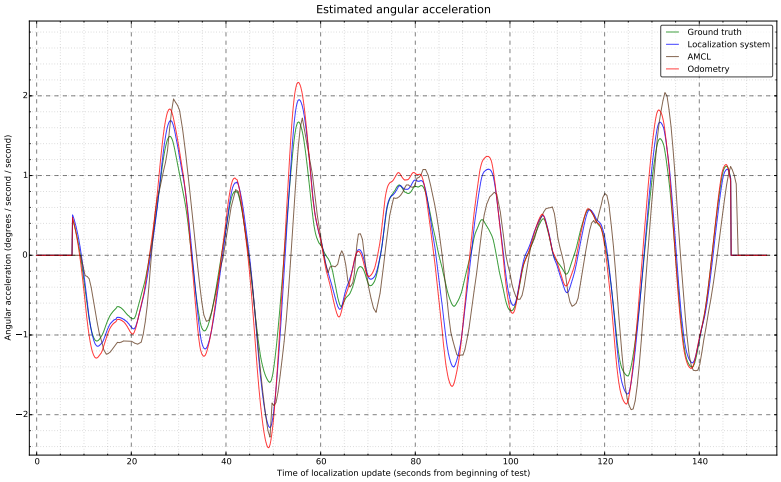


Figure A.2.2.15: Estimated angular acceleration

Figure A.2.2.16: Odometry translation errors

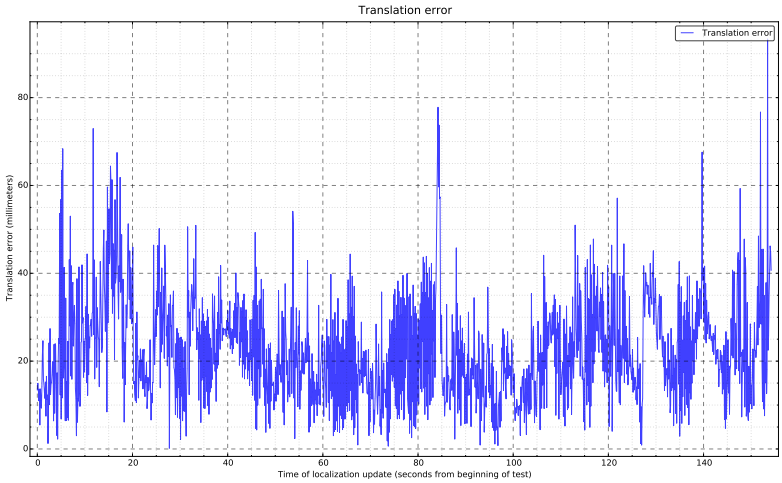


Figure A.2.2.17: Localization system translation errors

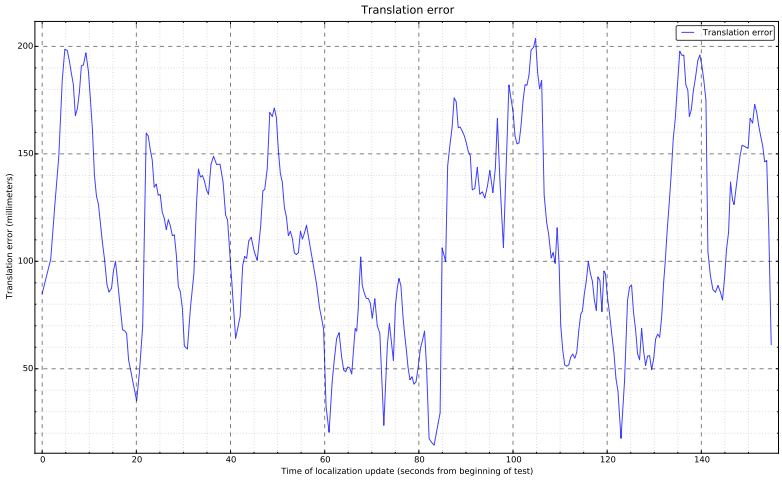


Figure A.2.2.18: AMCL translation errors

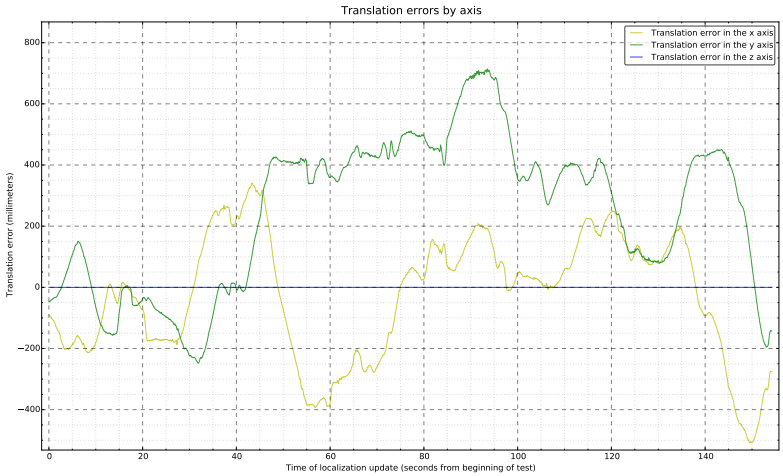


Figure A.2.2.19: Odometry translation errors components

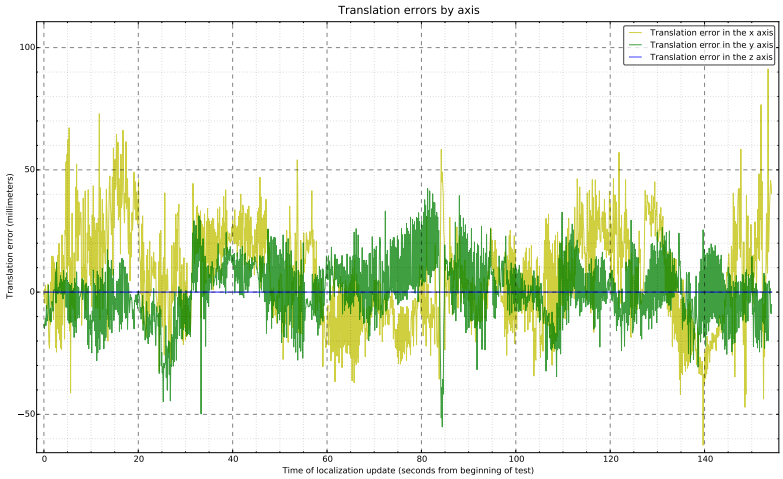


Figure A.2.2.20: Localization system translation errors components

Figure A.2.2.21: AMCL translation errors components

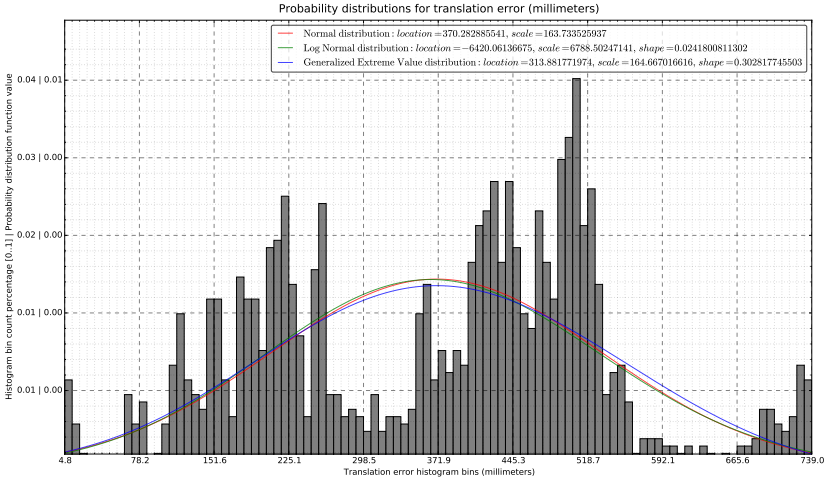


Figure A.2.2.22: Probability distributions for the odometry translation errors

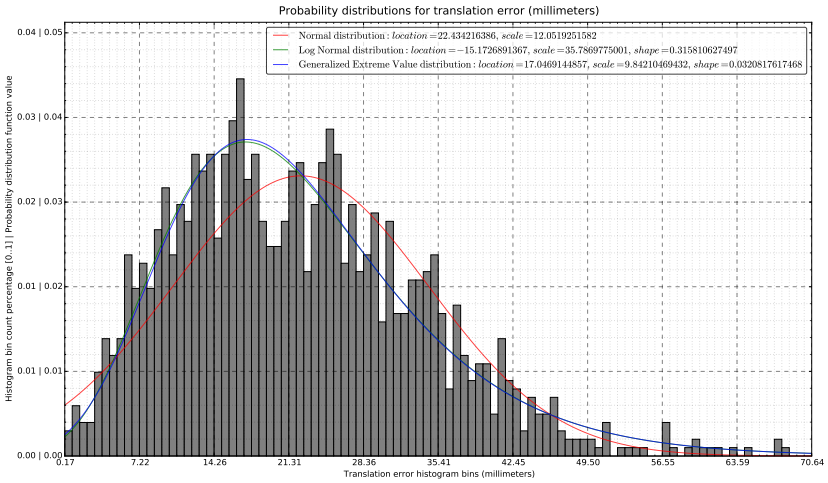


Figure A.2.2.23: Probability distributions for the localization system translation errors

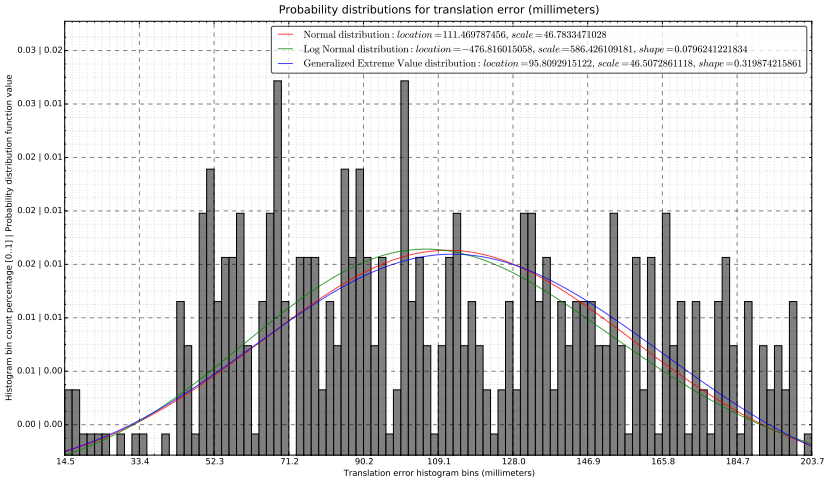


Figure A.2.2.24: Probability distributions for the AMCL translation errors

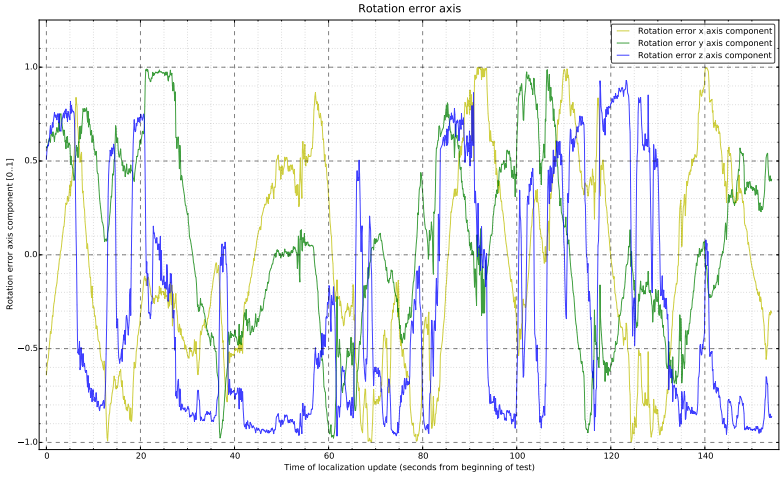


Figure A.2.2.25: Odometry rotation errors axis

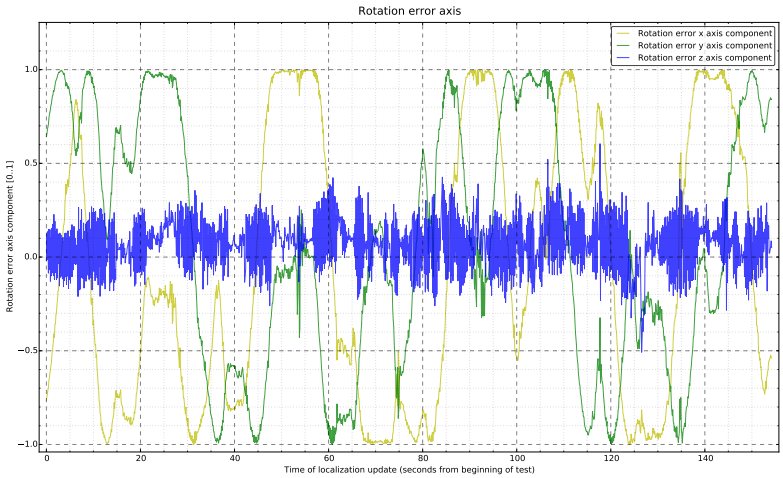


Figure A.2.2.26: Localization system rotation errors axis

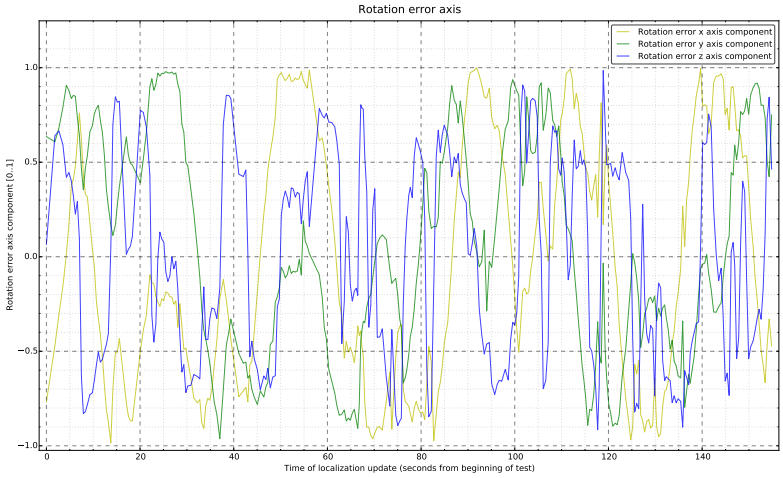


Figure A.2.2.27: AMCL rotation errors axis

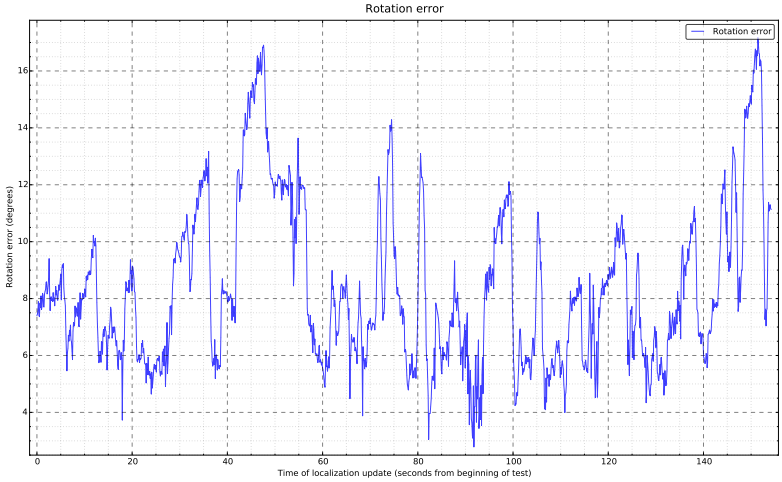


Figure A.2.2.28: Odometry rotation errors

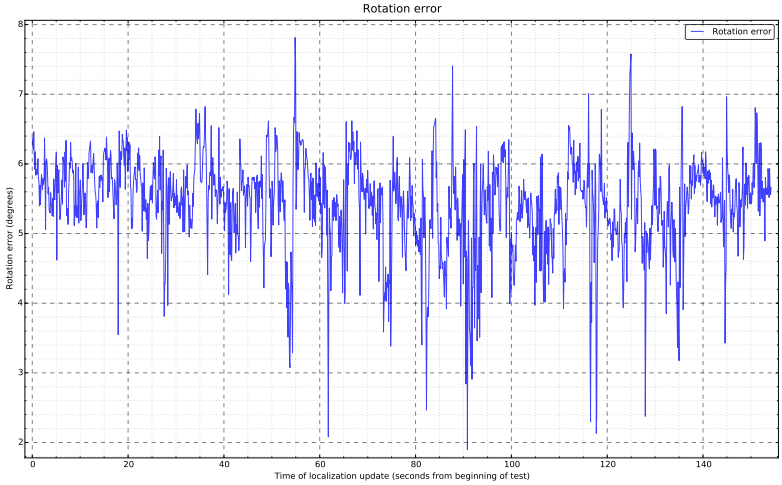


Figure A.2.2.29: Localization system rotation errors

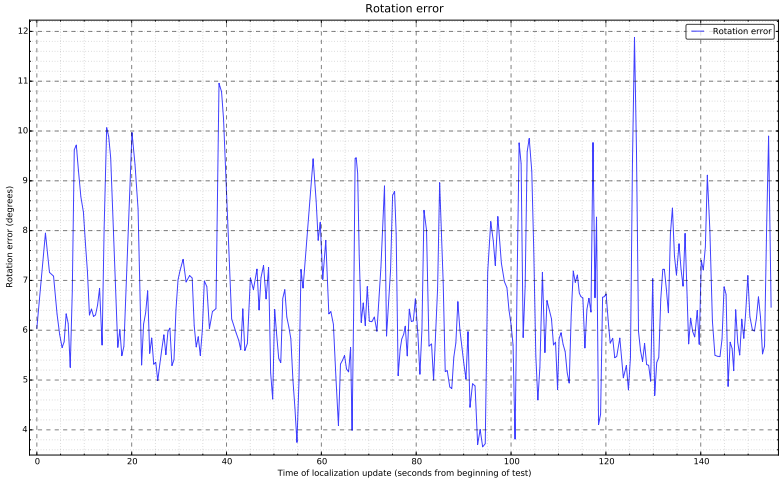


Figure A.2.2.30: AMCL rotation errors

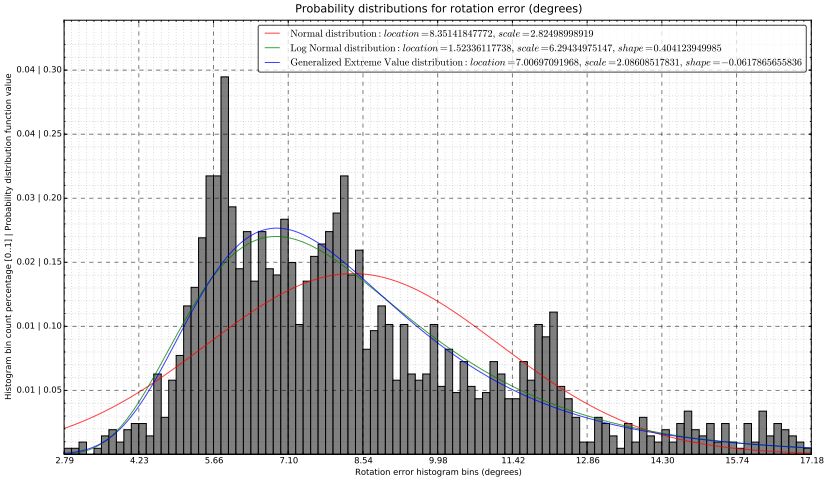


Figure A.2.2.31: Probability distributions for the odometry rotation errors

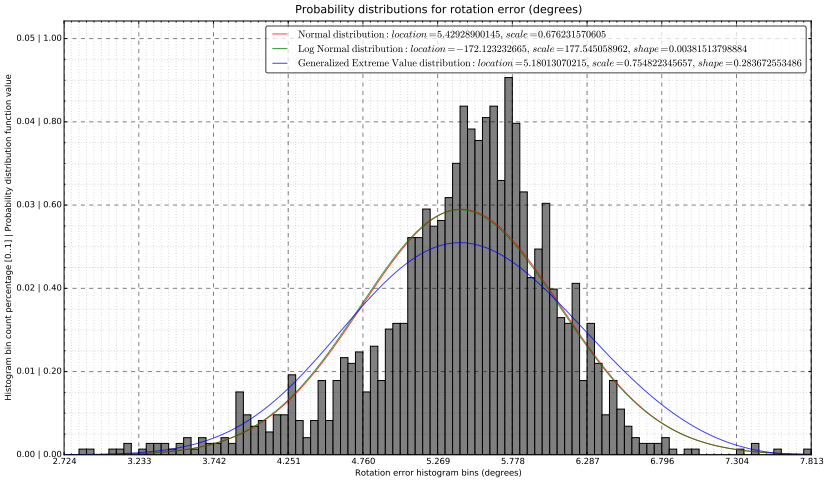


Figure A.2.2.32: Probability distributions for the localization system rotation errors

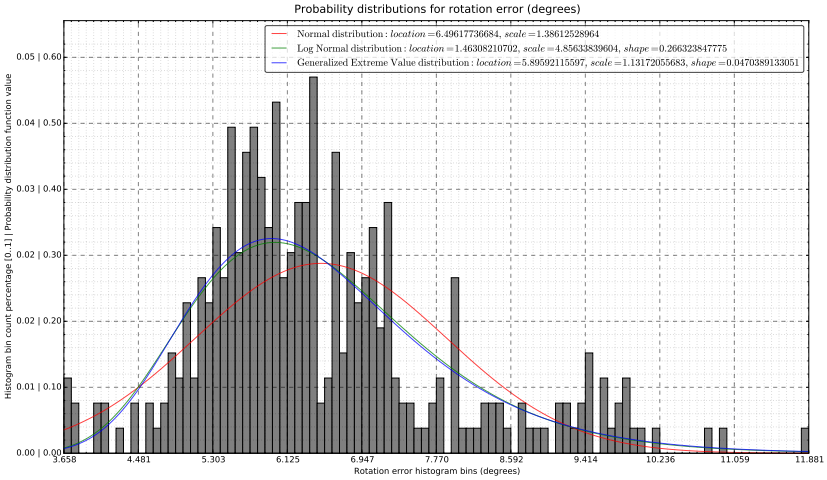


Figure A.2.2.33: Probability distributions for the AMCL rotation errors

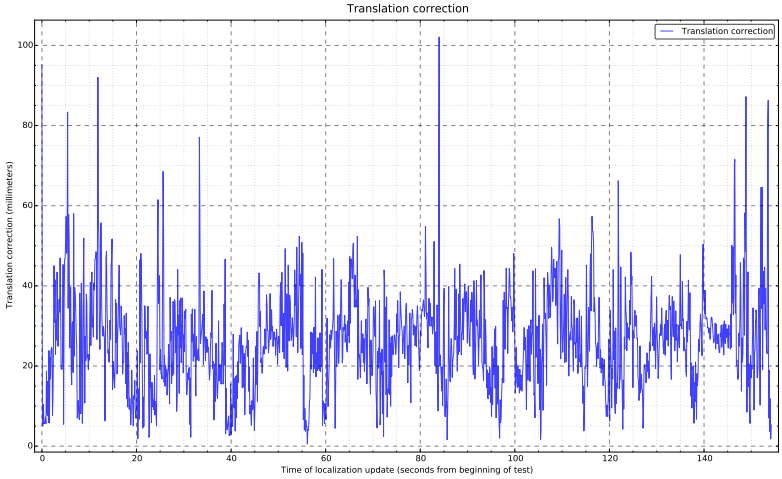


Figure A.2.2.34: Translation corrections performed by the localization system

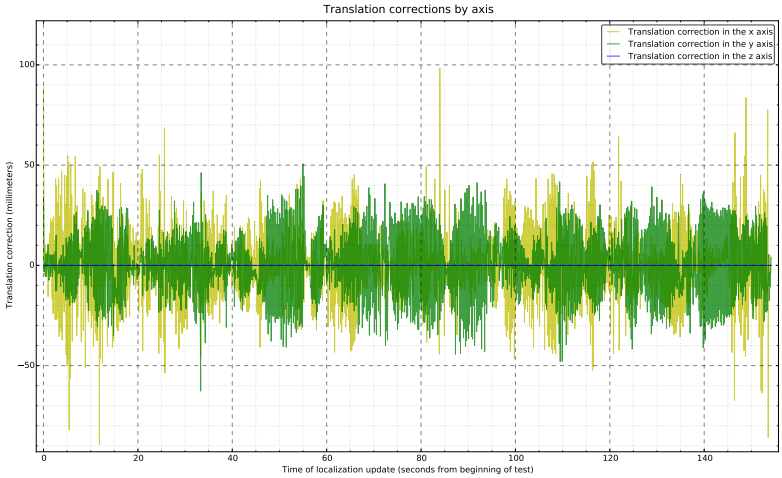


Figure A.2.2.35: Translation corrections components performed by the localization system

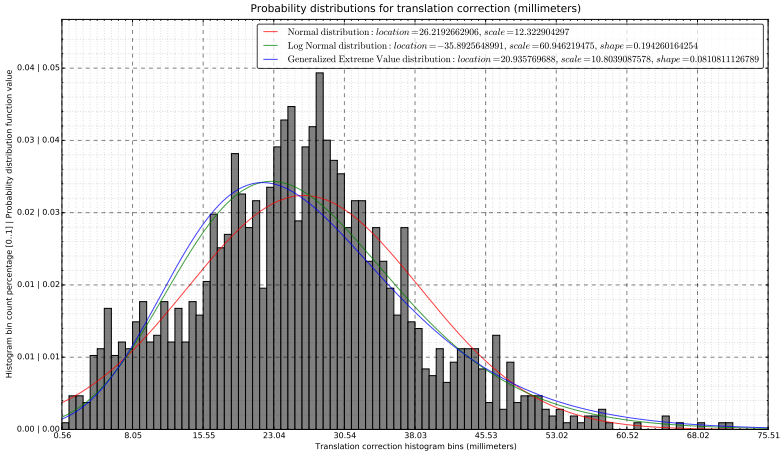


Figure A.2.2.36: Probability distributions for the translation corrections performed by the localization system

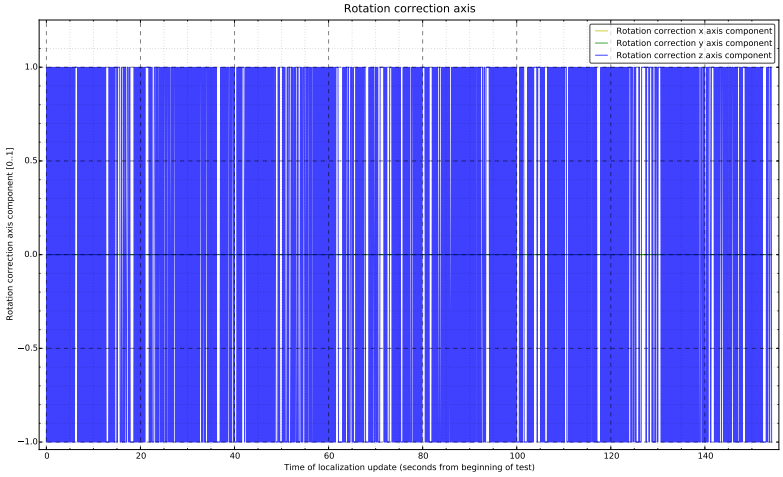


Figure A.2.2.37: Rotation corrections (axis) performed by the localization system

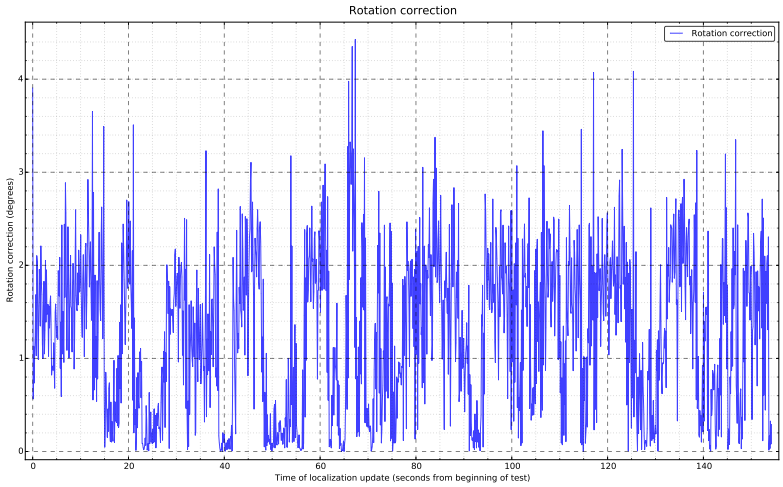


Figure A.2.2.38: Rotation corrections performed by the localization system

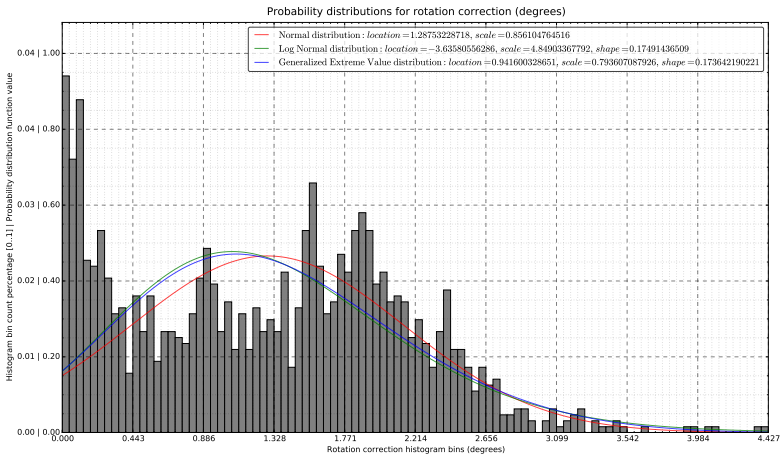


Figure A.2.2.39: Probability distributions for the rotation corrections performed by the localization system

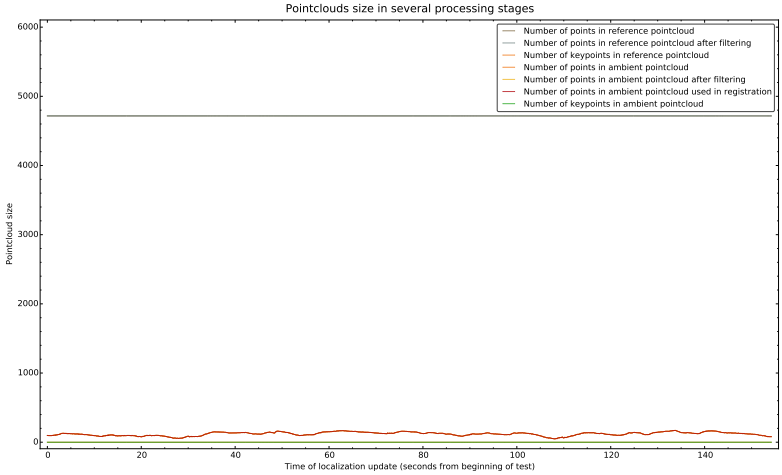


Figure A.2.2.40: Point clouds size in several of the localization system processing stages

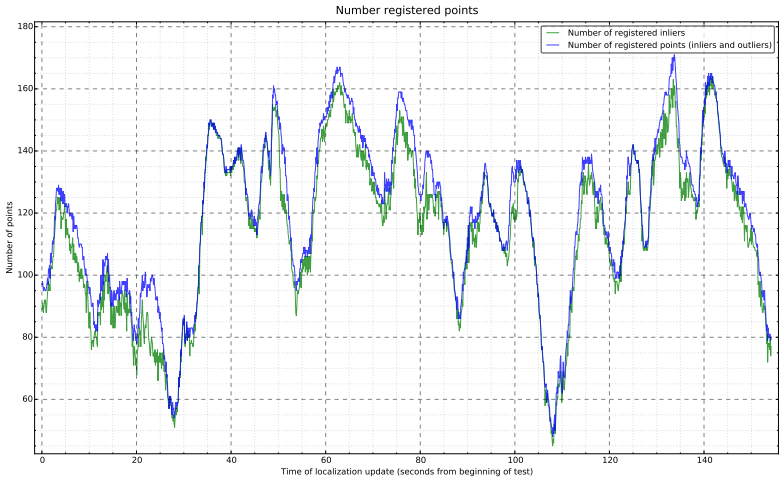


Figure A.2.2.41: Number of registered points

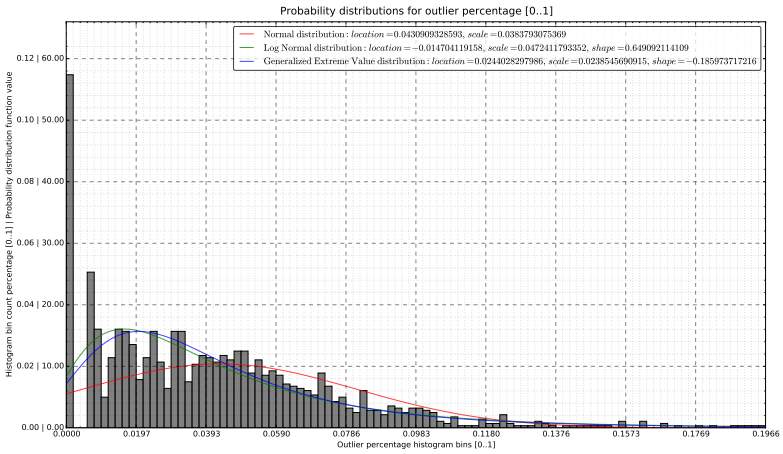


Figure A.2.2.42: Probability distributions for the ambient point cloud outlier percentage

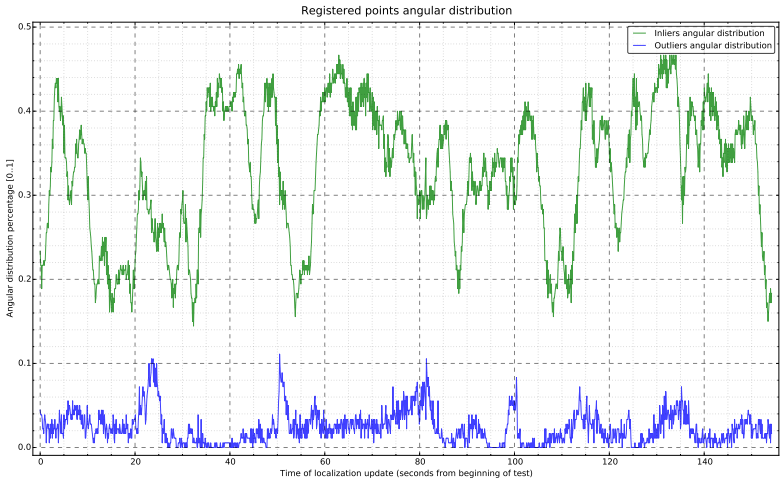


Figure A.2.2.43: Angular distribution percentage of registered points

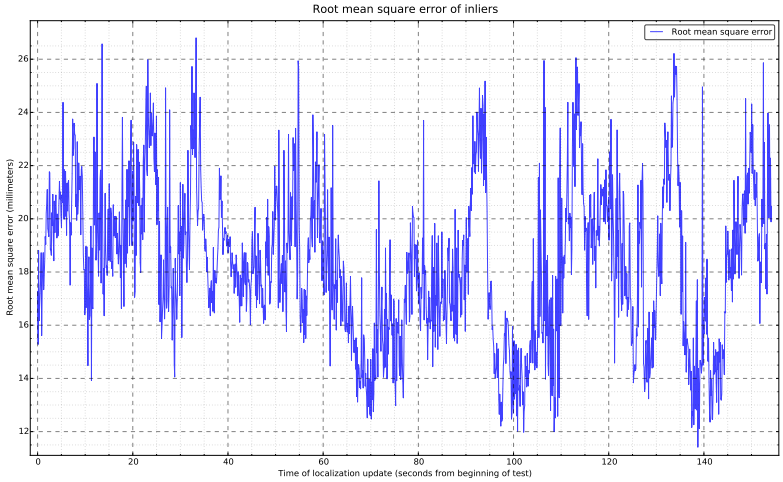


Figure A.2.2.44: Root Mean Square Error of the inliers

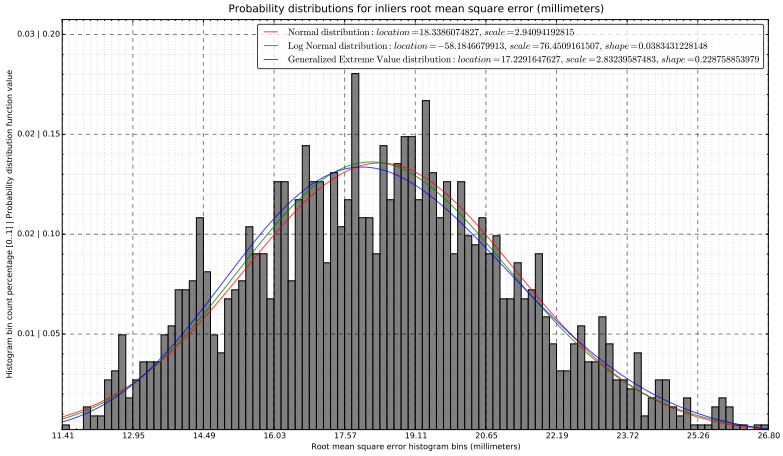


Figure A.2.2.45: Probability distributions for the Root Mean Square Error of the inliers

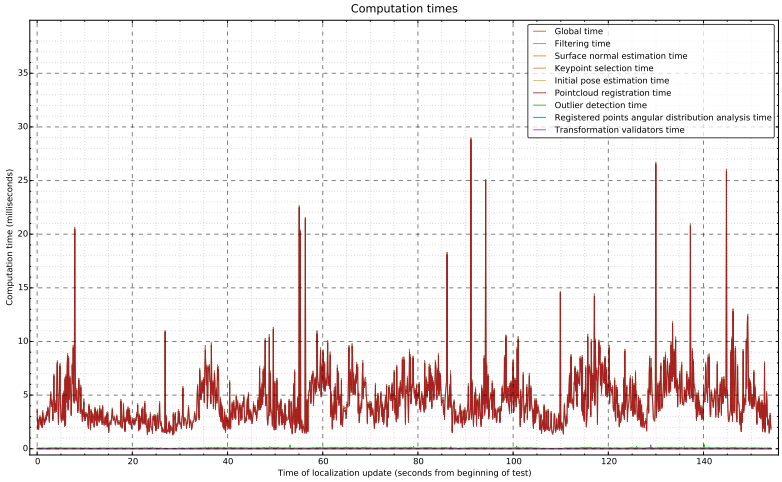


Figure A.2.2.46: Localization system computation times

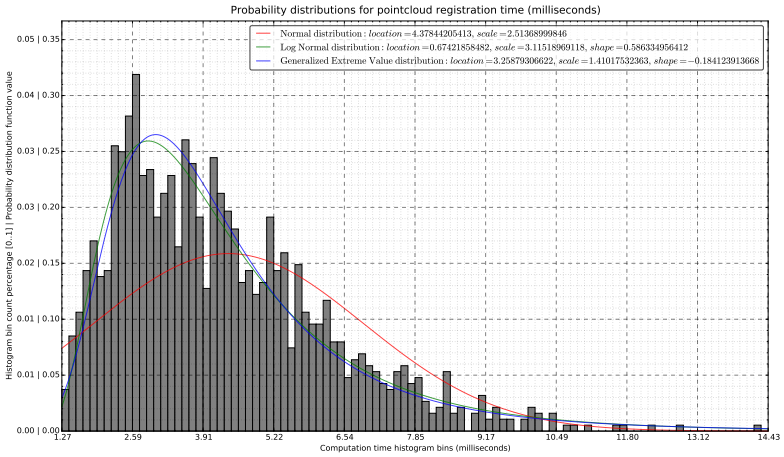


Figure A.2.2.47: Probability distributions for the point cloud registration time

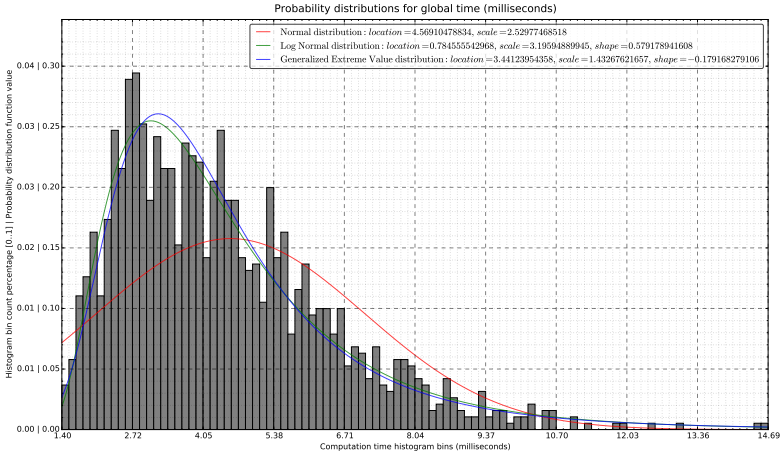


Figure A.2.2.48: Probability distributions for the localization system global computation time

A.2.3 Second SLAM path using the Pioneer robot at 19 cm/s

The figures in this section show the detailed results of the test performed with the Pioneer robot with a 25 mm resolution map in the industrial hall following the second SLAM path and moving at 19 cm/s.

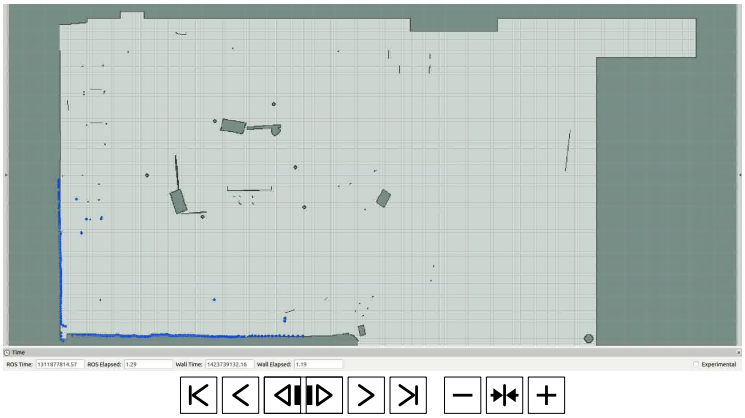


Figure A.2.3.1: Animation of laser scans using the ground truth poses

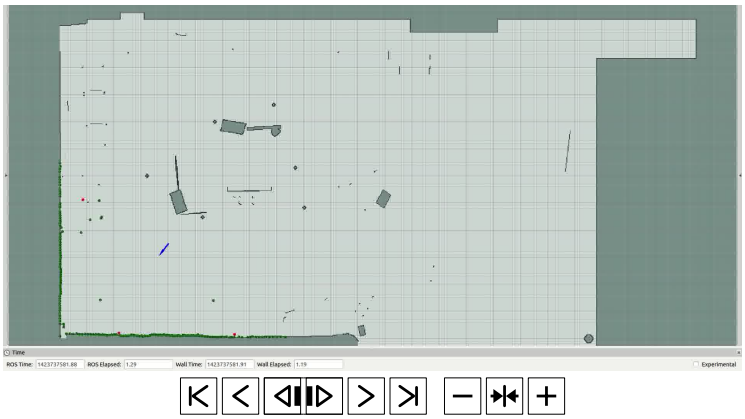


Figure A.2.3.2: Animation of laser scans using the localization system poses

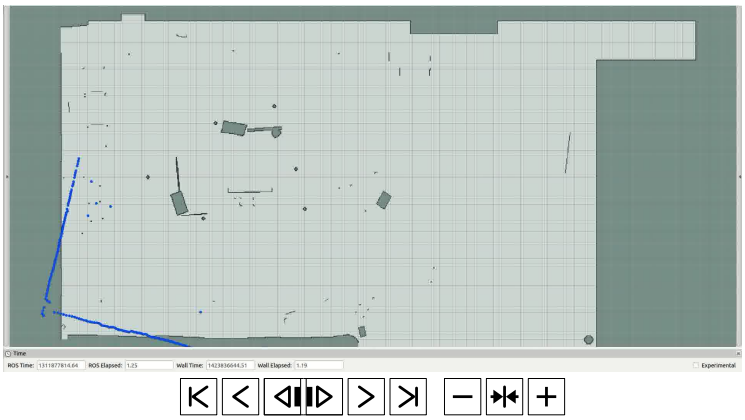


Figure A.2.3.3: Animation of laser scans using the AMCL poses

Figure A.2.3.4: Laser scans assembled on top of the map using the ground truth poses

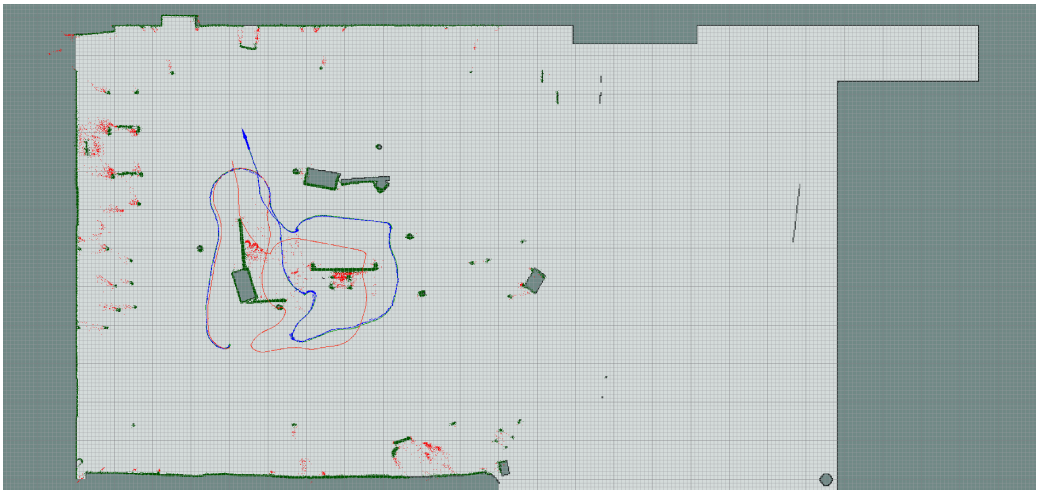


Figure A.2.3.5: Laser scans assembled on top of the map using the localization system poses

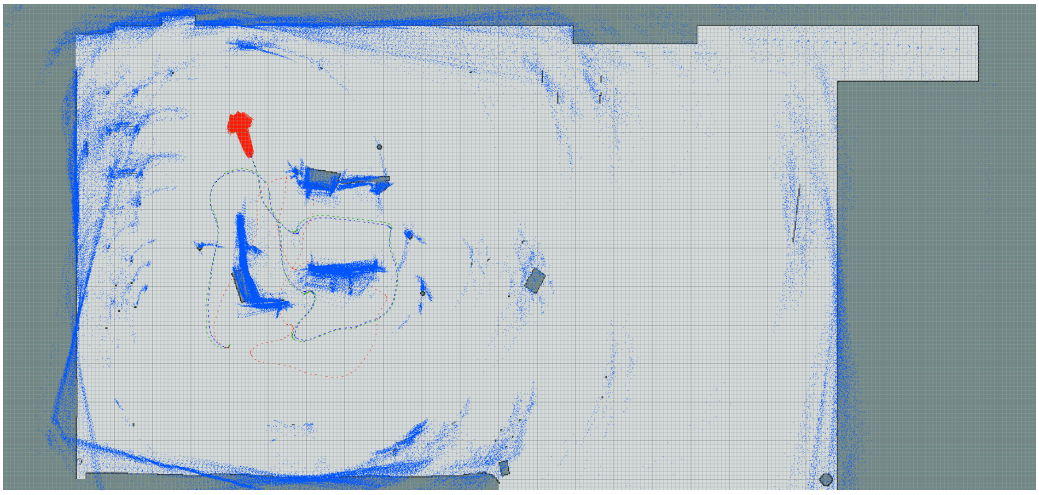


Figure A.2.3.6: Laser scans assembled on top of the map using the AMCL poses

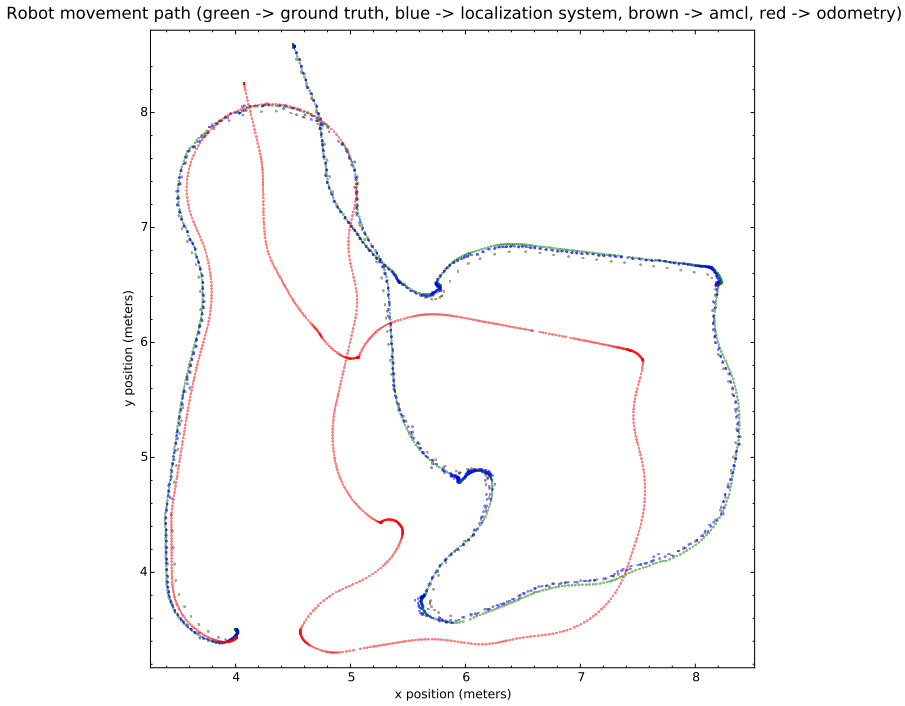


Figure A.2.3.7: Poses estimated by the ground truth, localization system, AMCL and odometry

Figure A.2.3.8: Evolution of x position over time

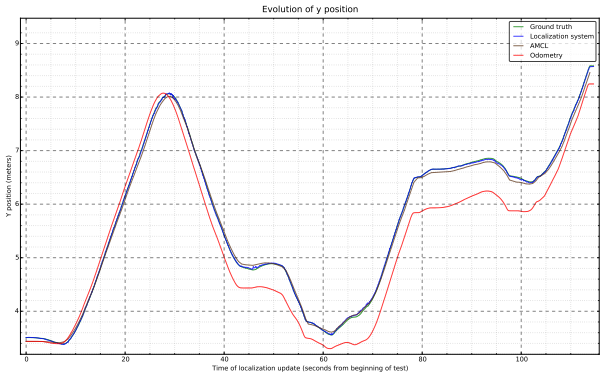


Figure A.2.3.9: Evolution of y position over time

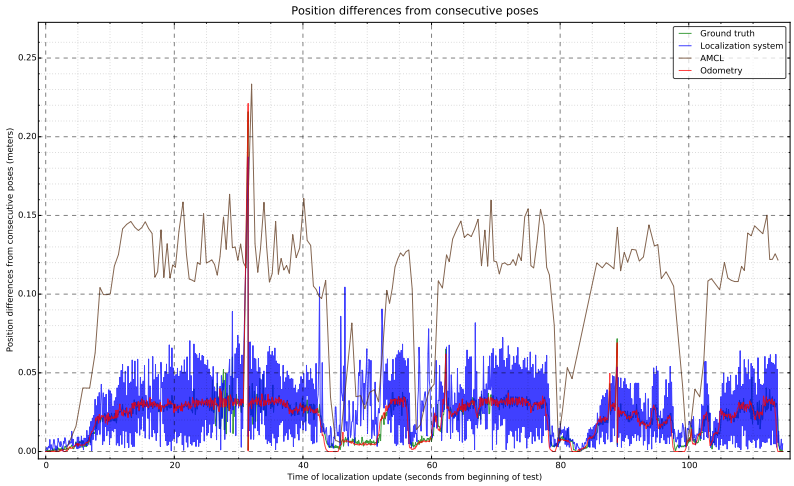


Figure A.2.3.10: Distance traveled between consecutive pose estimations

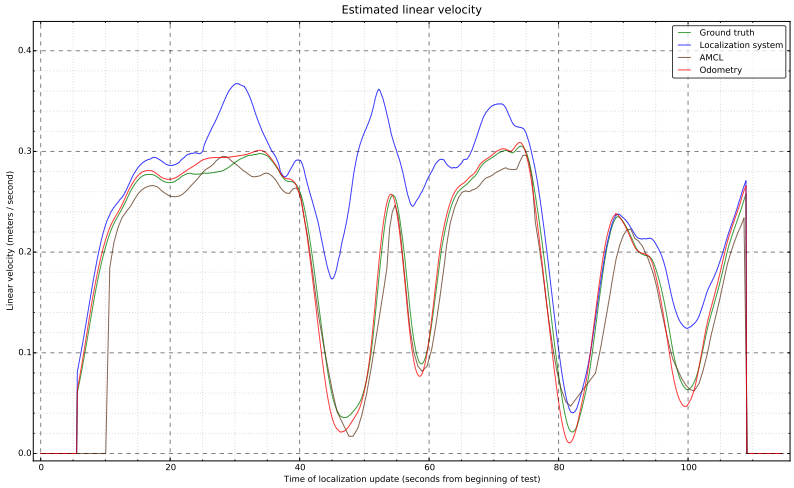


Figure A.2.3.11: Estimated linear velocity

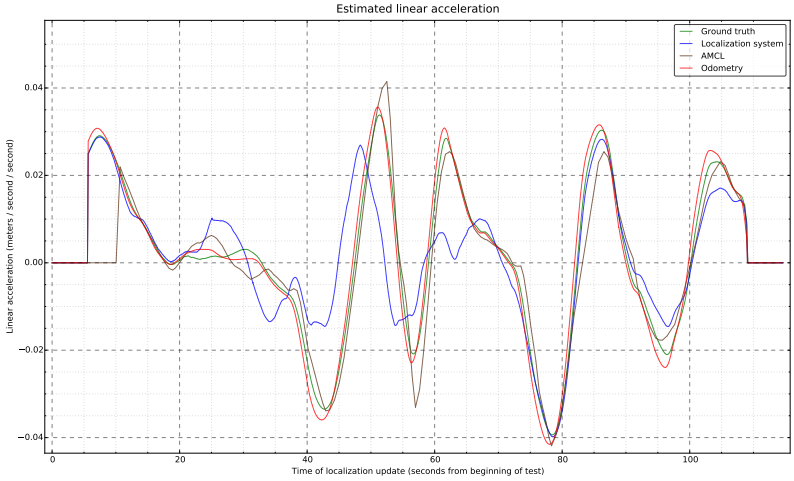


Figure A.2.3.12: Estimated linear acceleration

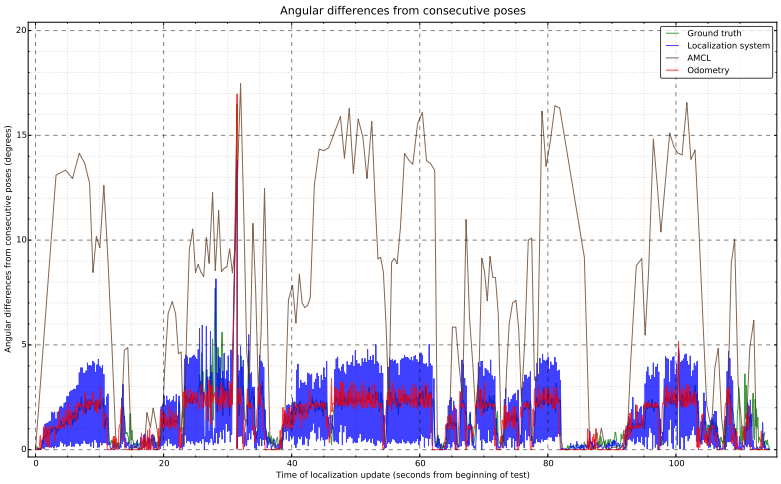


Figure A.2.3.13: Angular differences between consecutive pose estimations

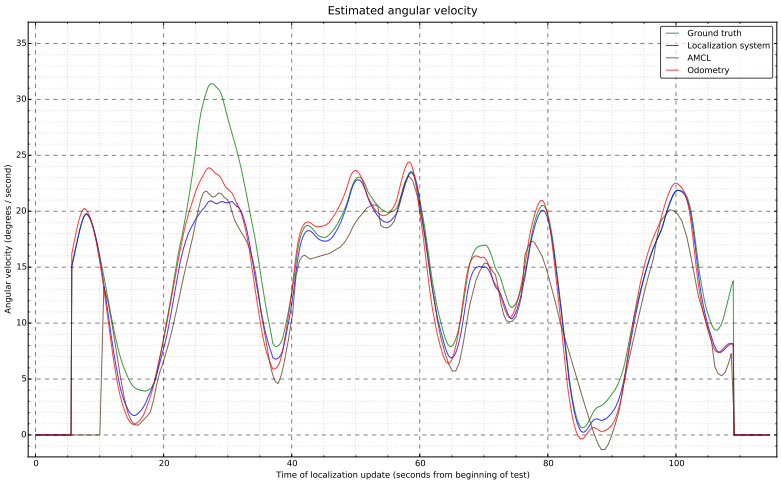


Figure A.2.3.14: Estimated angular velocity

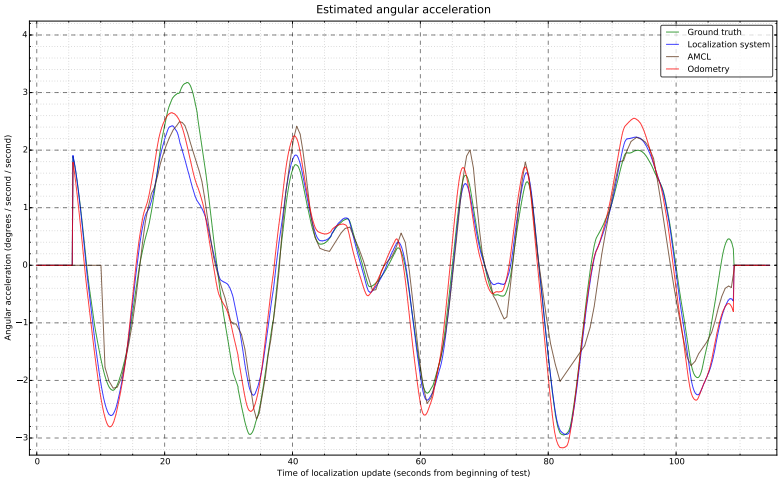


Figure A.2.3.15: Estimated angular acceleration

Figure A.2.3.16: Odometry translation errors

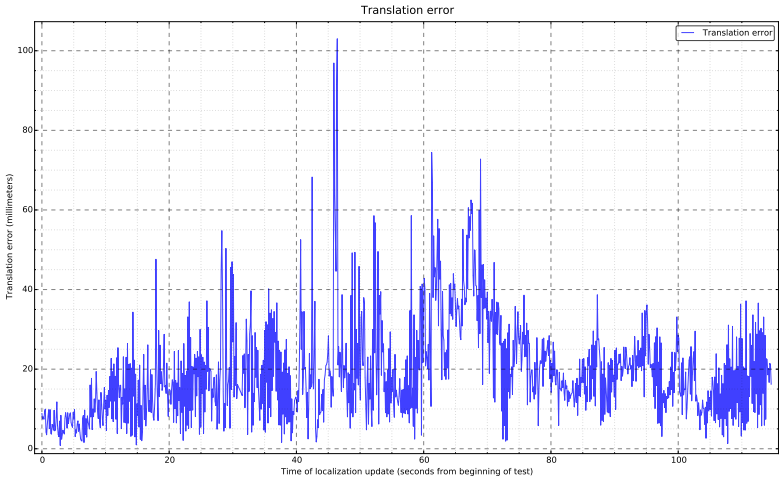


Figure A.2.3.17: Localization system translation errors

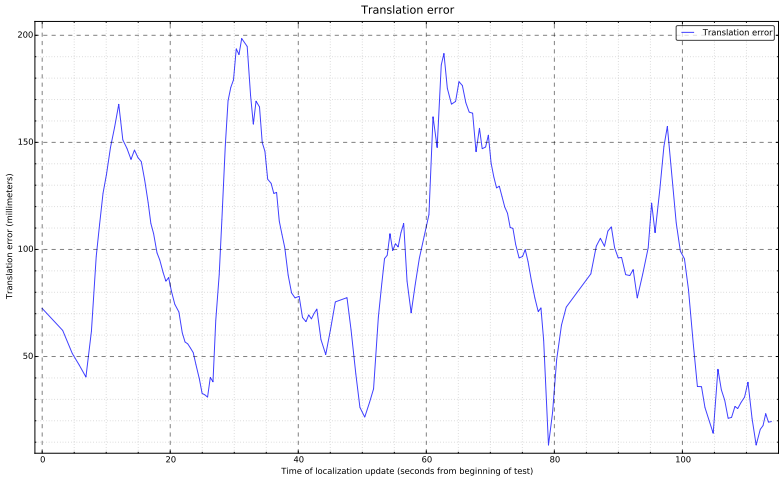


Figure A.2.3.18: AMCL translation errors

Figure A.2.3.19: Odometry translation errors components

Figure A.2.3.20: Localization system translation errors components

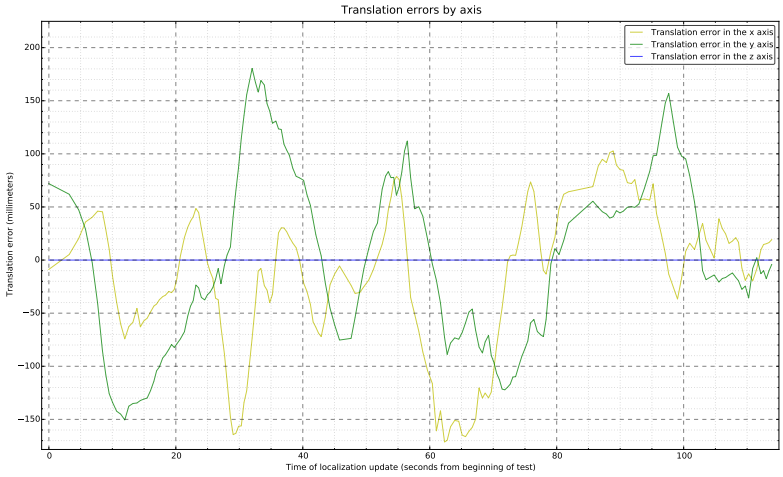


Figure A.2.3.21: AMCL translation errors components

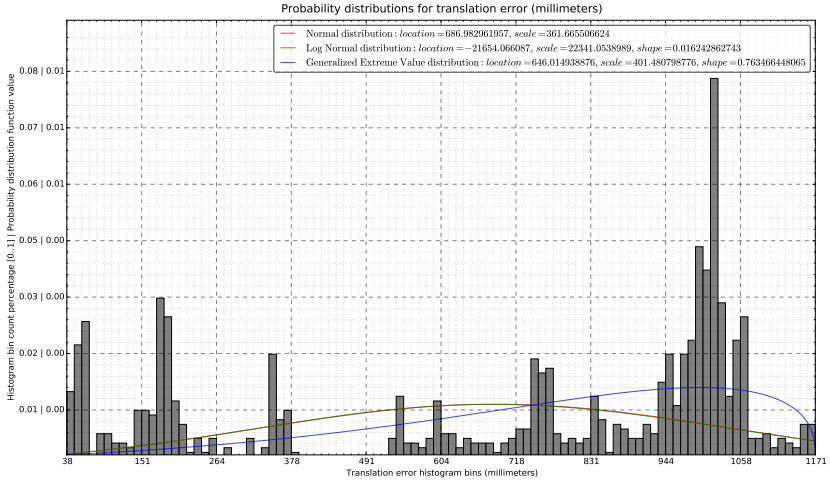


Figure A.2.3.22: Probability distributions for the odometry translation errors

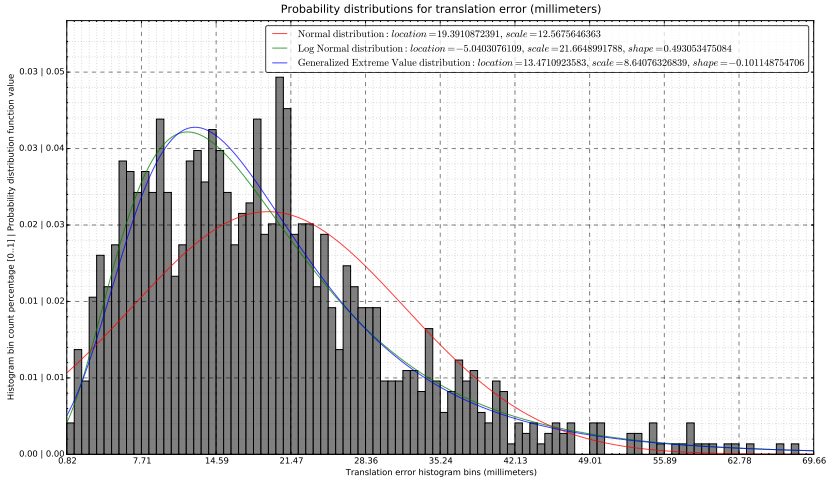


Figure A.2.3.23: Probability distributions for the localization system translation errors

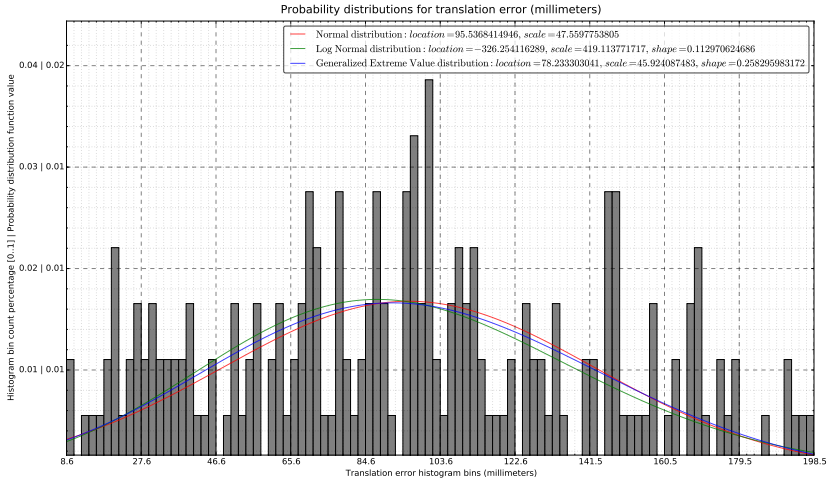


Figure A.2.3.24: Probability distributions for the AMCL translation errors

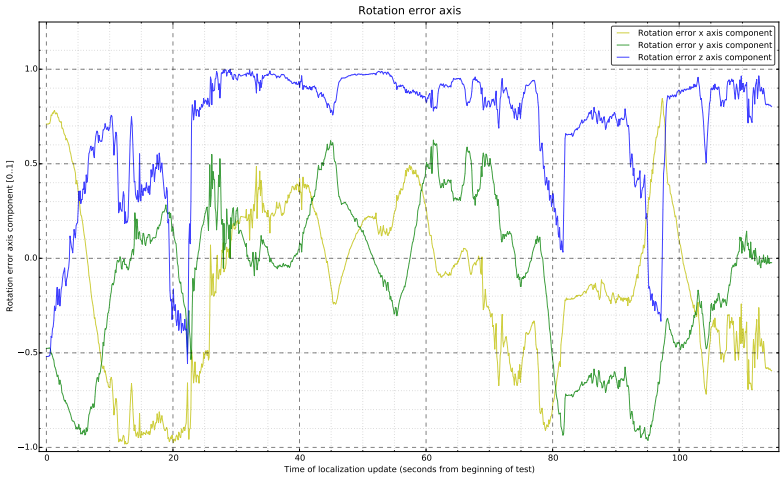


Figure A.2.3.25: Odometry rotation errors axis

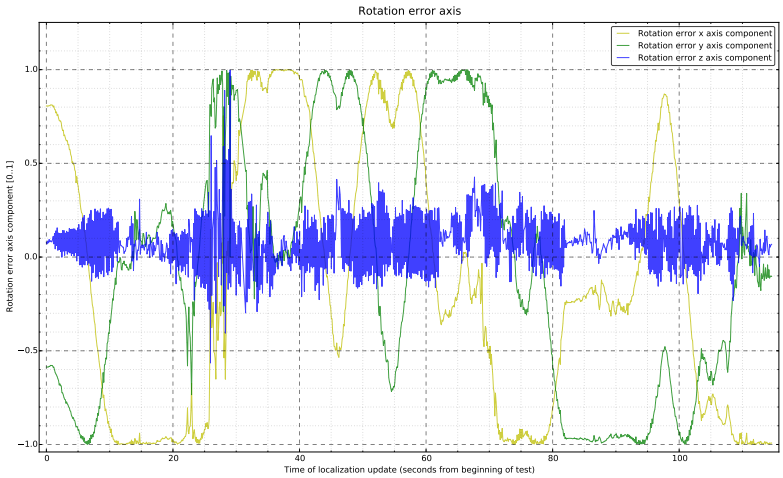


Figure A.2.3.26: Localization system rotation errors axis

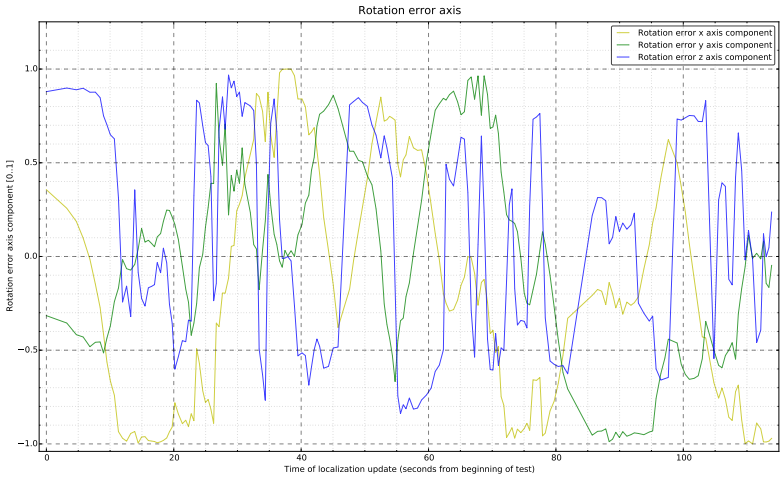


Figure A.2.3.27: AMCL rotation errors axis

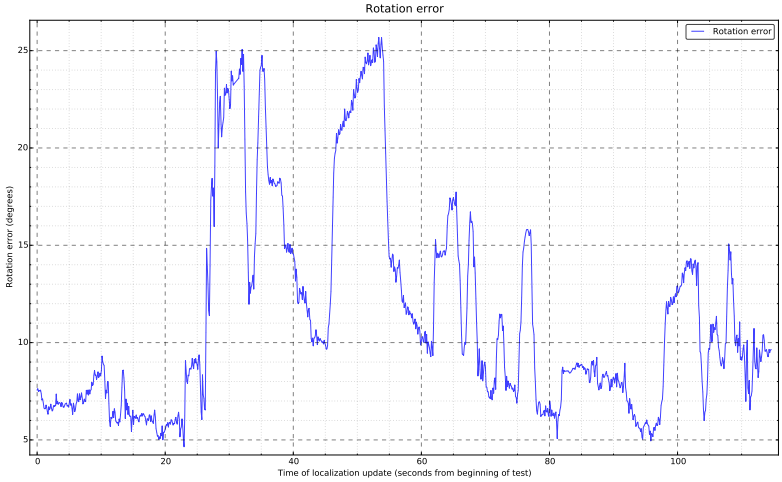


Figure A.2.3.28: Odometry rotation errors

Figure A.2.3.29: Localization system rotation errors

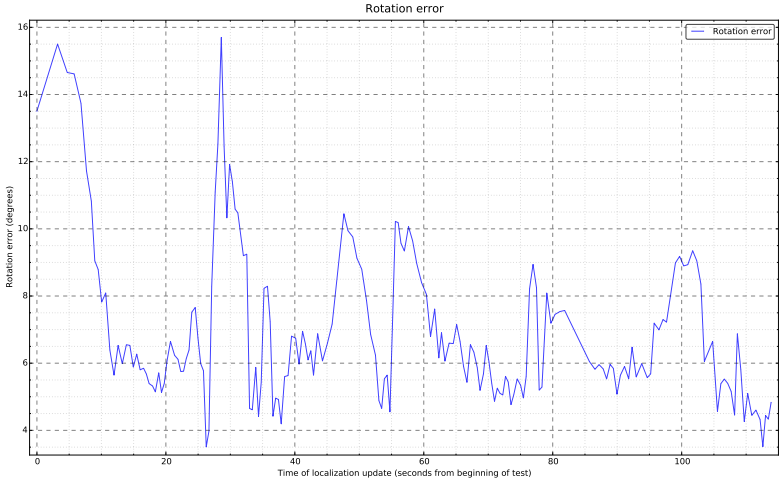


Figure A.2.3.30: AMCL rotation errors

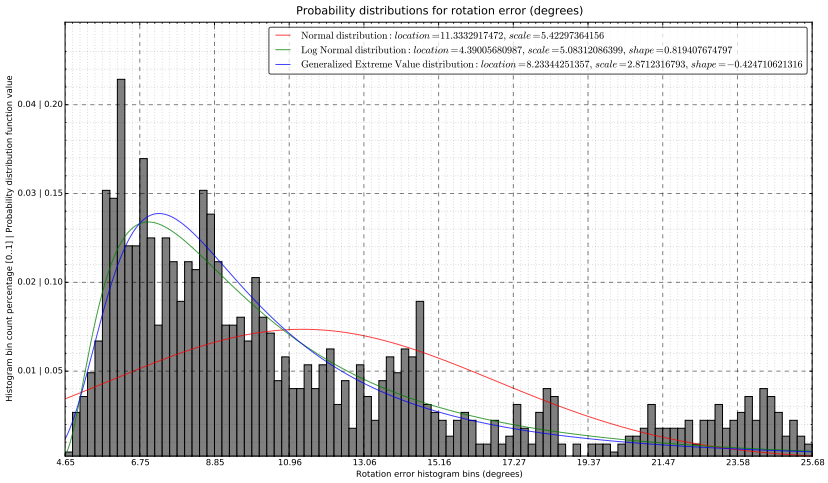


Figure A.2.3.31: Probability distributions for the odometry rotation errors

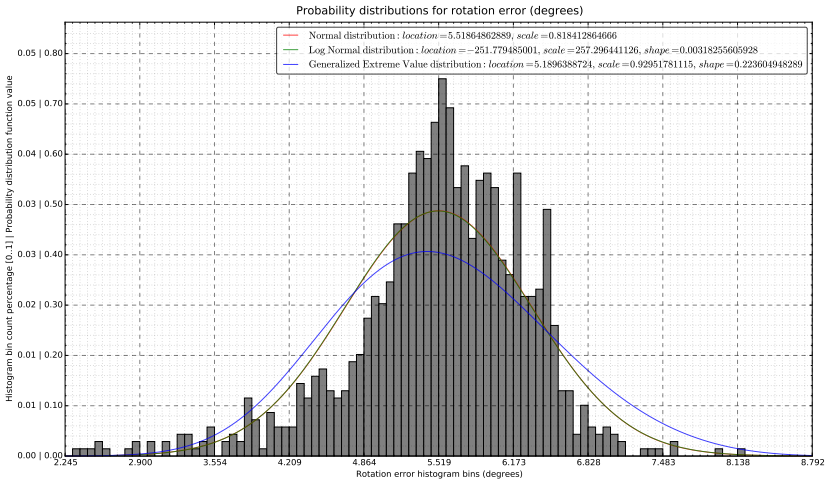


Figure A.2.3.32: Probability distributions for the localization system rotation errors

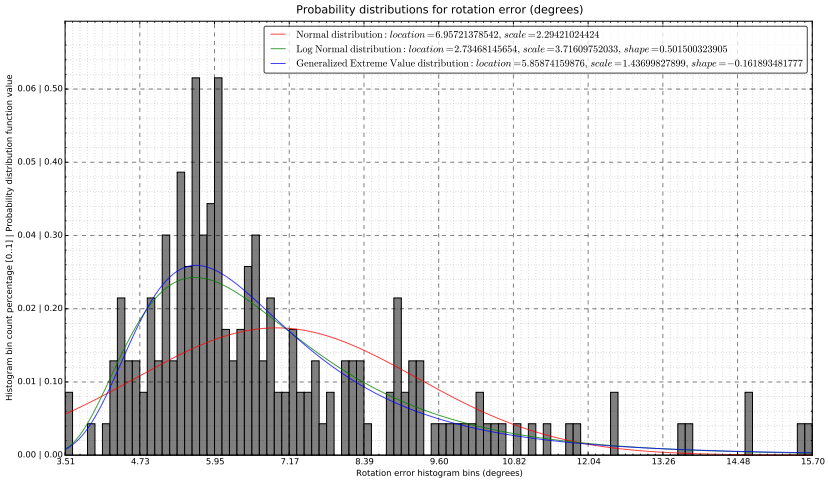


Figure A.2.3.33: Probability distributions for the AMCL rotation errors

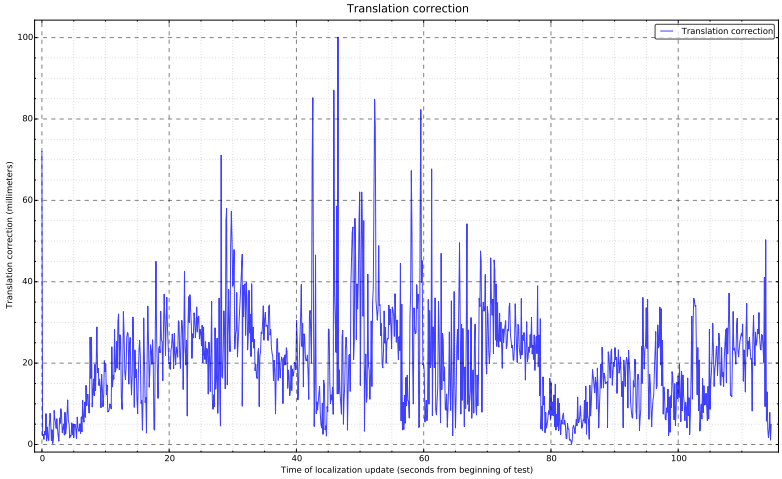


Figure A.2.3.34: Translation corrections performed by the localization system

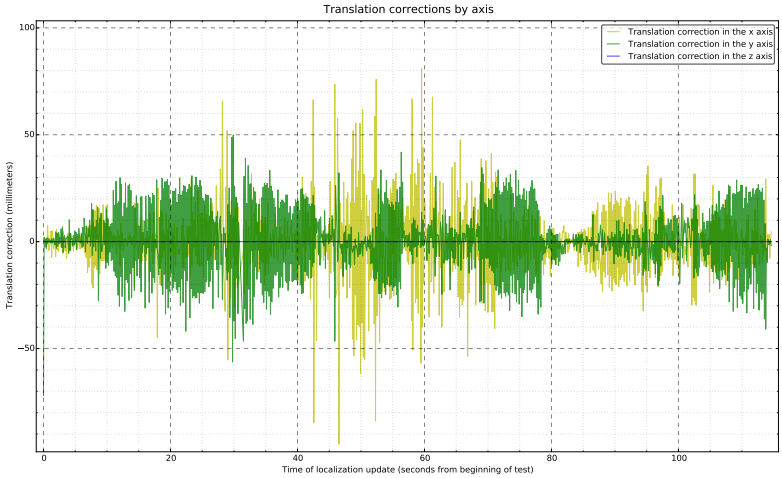


Figure A.2.3.35: Translation corrections components performed by the localization system

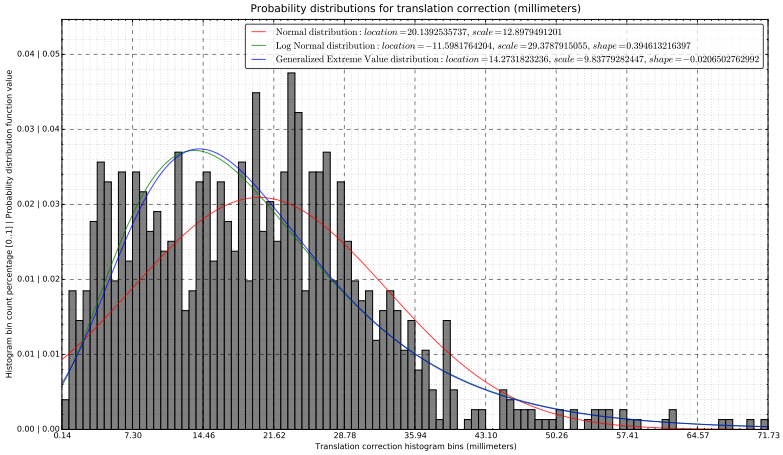


Figure A.2.3.36: Probability distributions for the translation corrections performed by the localization system

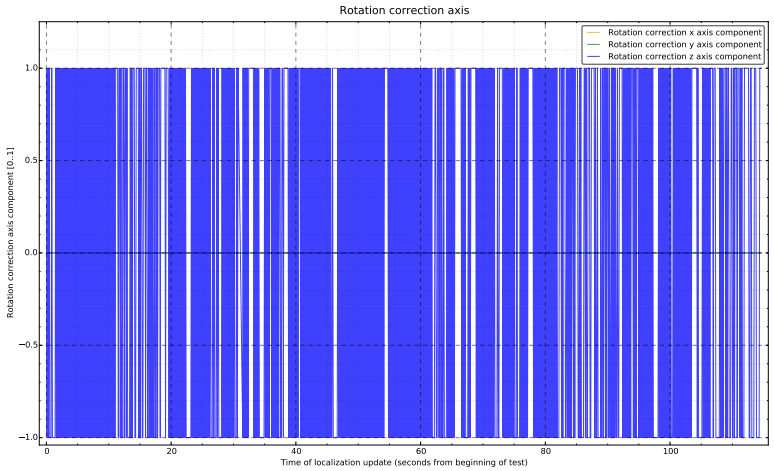


Figure A.2.3.37: Rotation corrections (axis) performed by the localization system

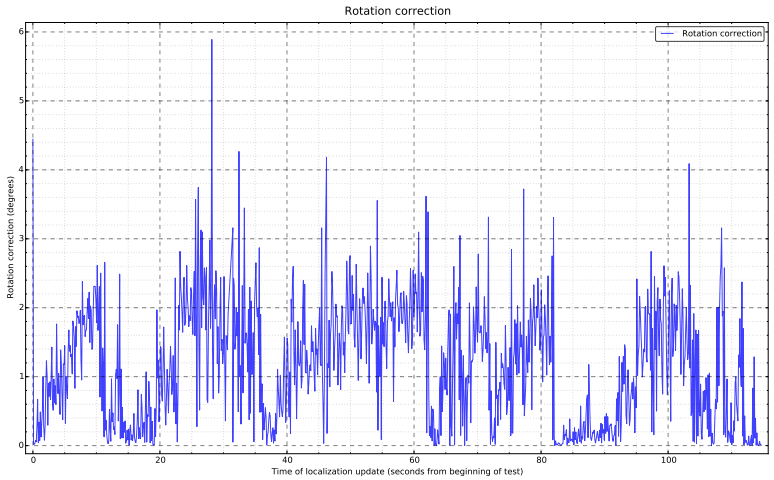


Figure A.2.3.38: Rotation corrections performed by the localization system

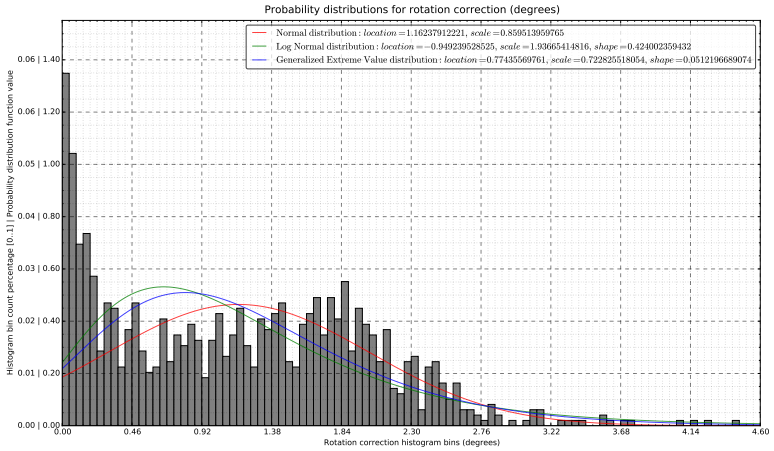


Figure A.2.3.39: Probability distributions for the rotation corrections performed by the localization system

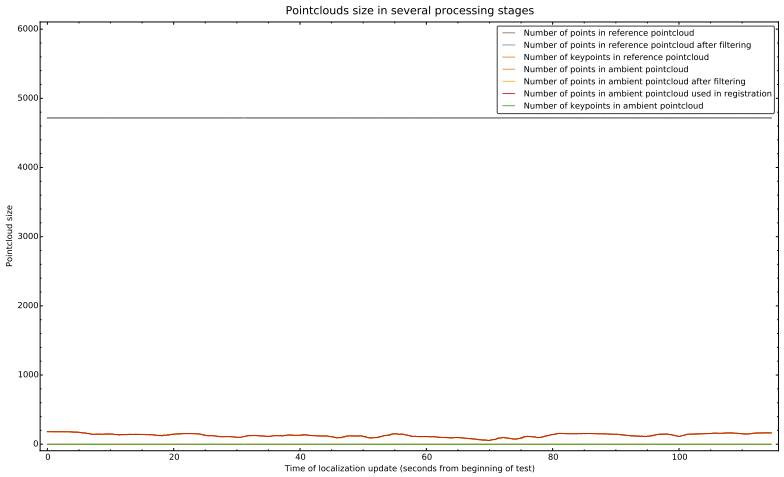


Figure A.2.3.40: Point clouds size in several of the localization system processing stages

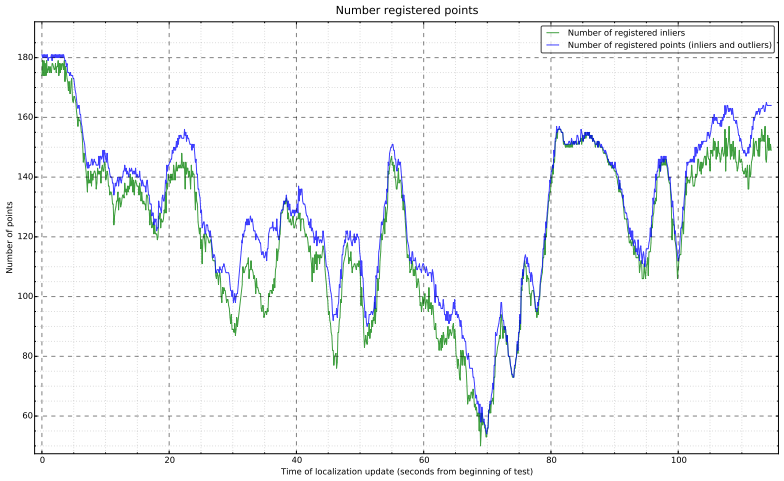


Figure A.2.3.41: Number of registered points

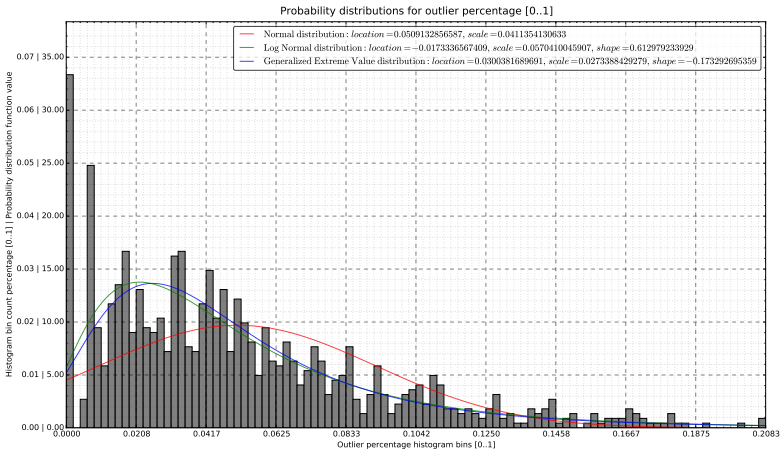


Figure A.2.3.42: Probability distributions for the ambient point cloud outlier percentage

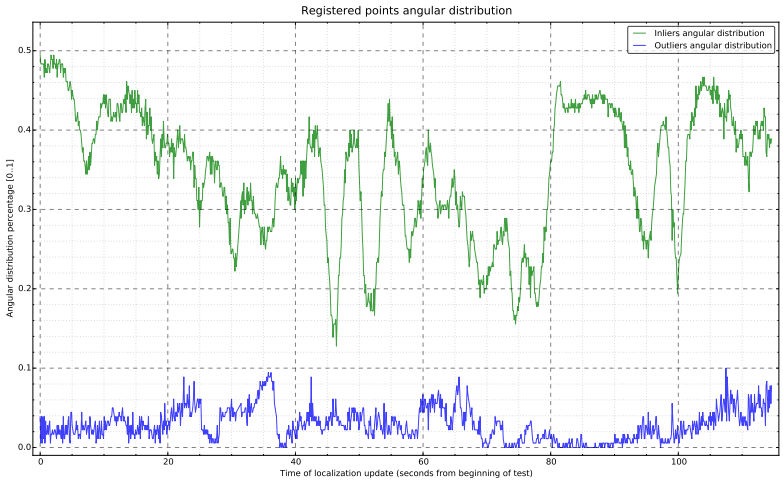


Figure A.2.3.43: Angular distribution percentage of registered points

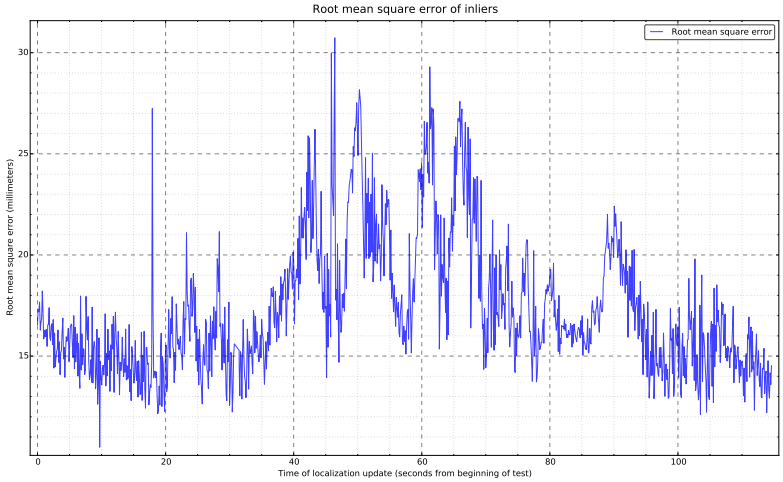


Figure A.2.3.44: Root Mean Square Error of the inliers

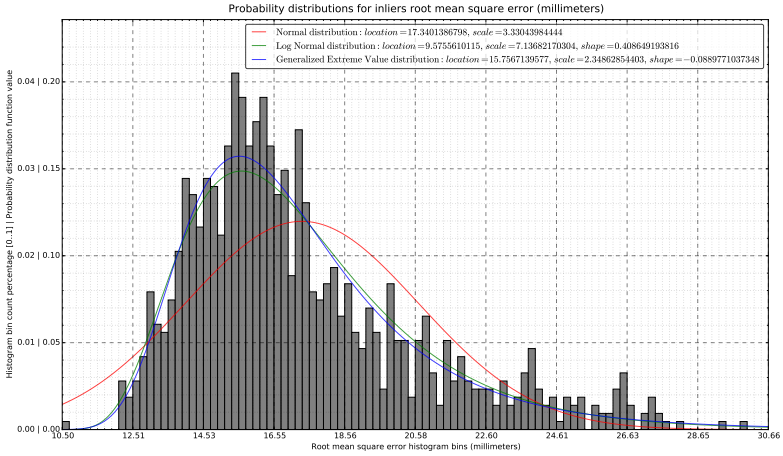


Figure A.2.3.45: Probability distributions for the Root Mean Square Error of the inliers

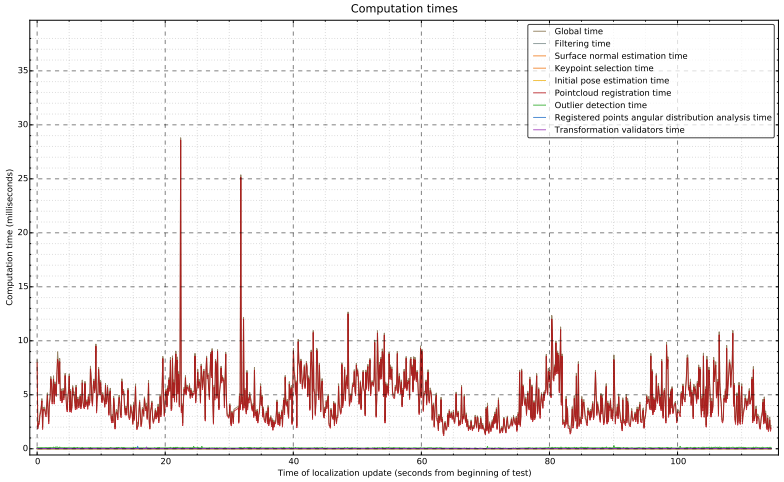


Figure A.2.3.46: Localization system computation times

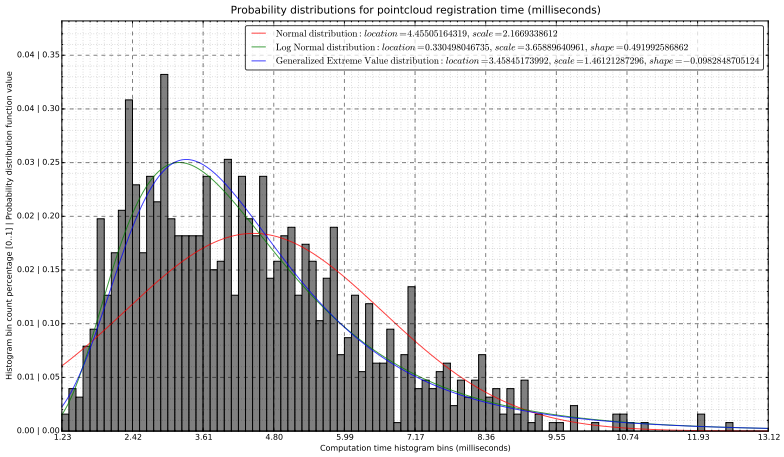


Figure A.2.3.47: Probability distributions for the point cloud registration time

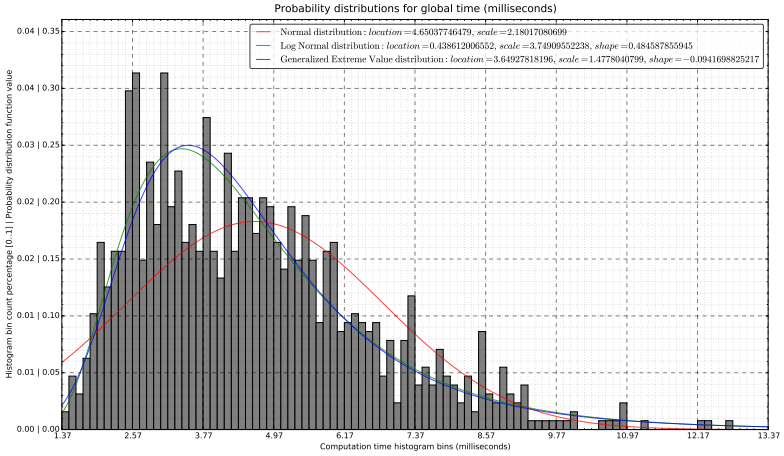


Figure A.2.3.48: Probability distributions for the localization system global computation time

A.2.4 Third SLAM path using the Pioneer robot at 16 cm/s

The figures in this section show the detailed results of the test performed with the Pioneer robot with a 25 mm resolution map in the industrial hall following the third SLAM path and moving at 16 cm/s.

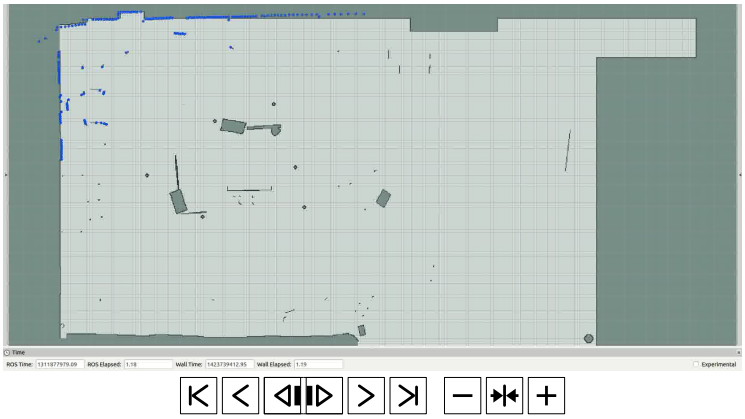


Figure A.2.4.1: Animation of laser scans using the ground truth poses

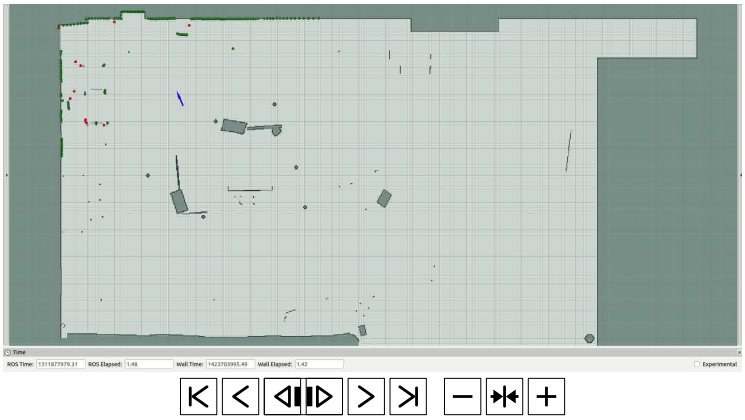


Figure A.2.4.2: Animation of laser scans using the localization system poses

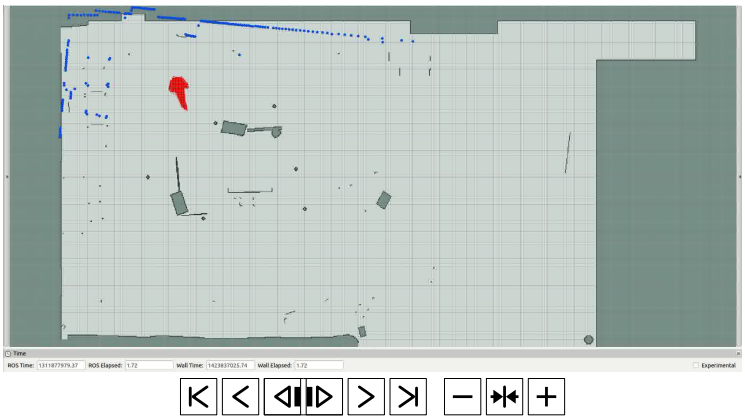


Figure A.2.4.3: Animation of laser scans using the AMCL poses

Figure A.2.4.4: Laser scans assembled on top of the map using the ground truth poses

Figure A.2.4.5: Laser scans assembled on top of the map using the localization system poses

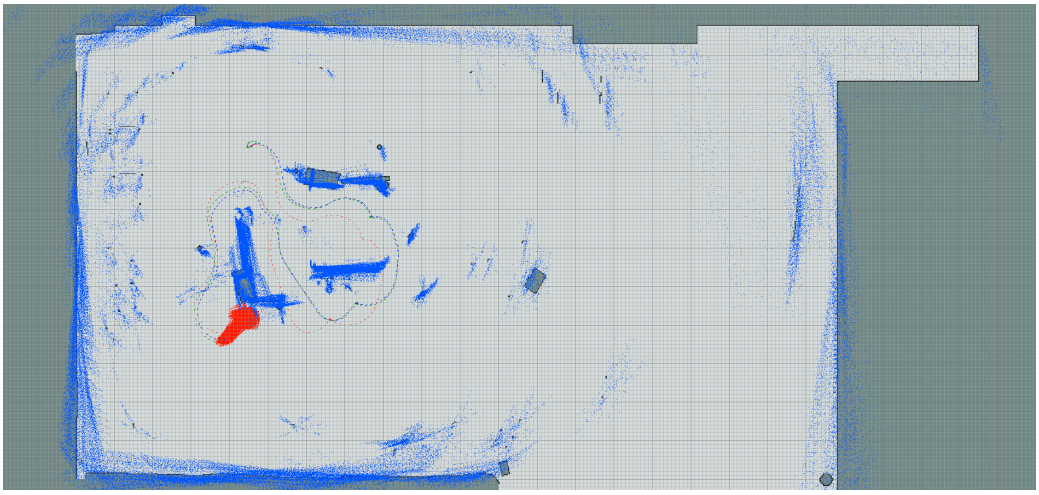


Figure A.2.4.6: Laser scans assembled on top of the map using the AMCL poses

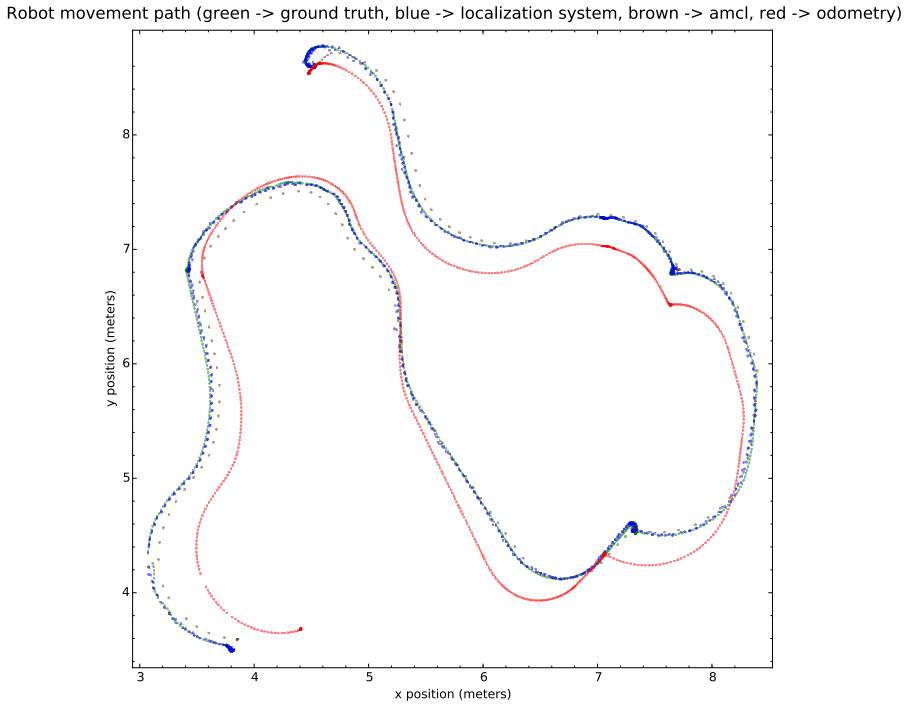


Figure A.2.4.7: Poses estimated by the ground truth, localization system, AMCL and odometry

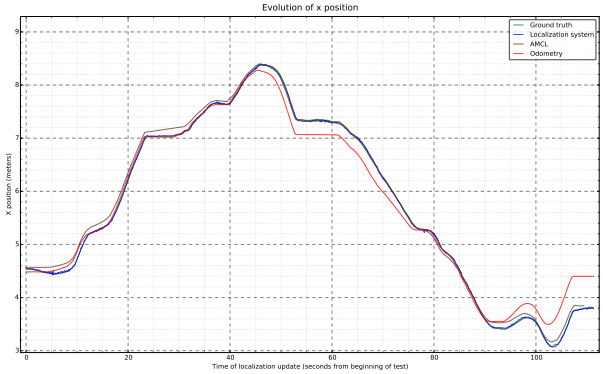


Figure A.2.4.8: Evolution of x position over time

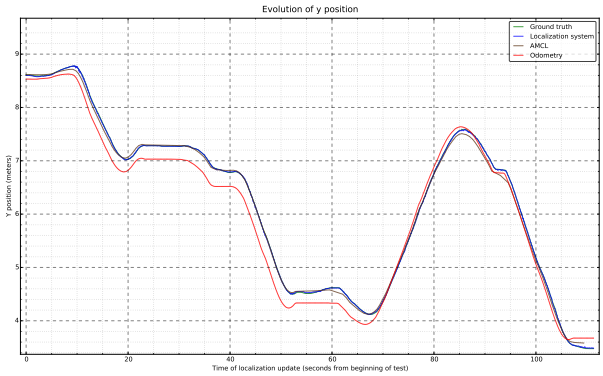


Figure A.2.4.9: Evolution of y position over time

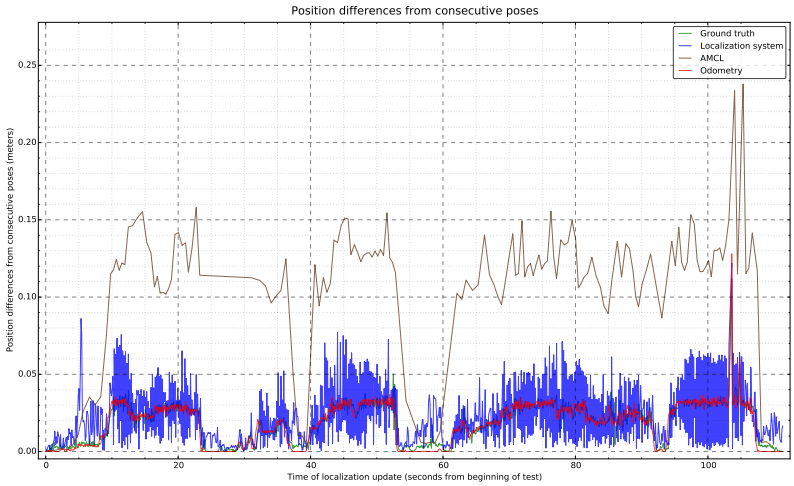


Figure A.2.4.10: Distance traveled between consecutive pose estimations

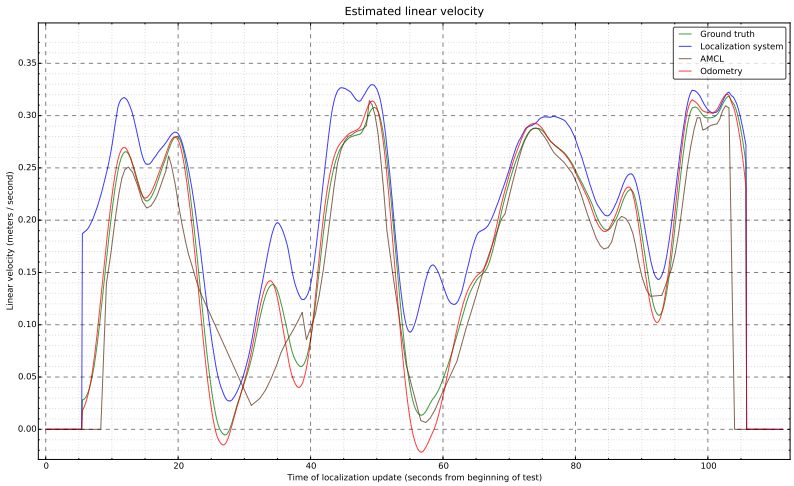


Figure A.2.4.11: Estimated linear velocity

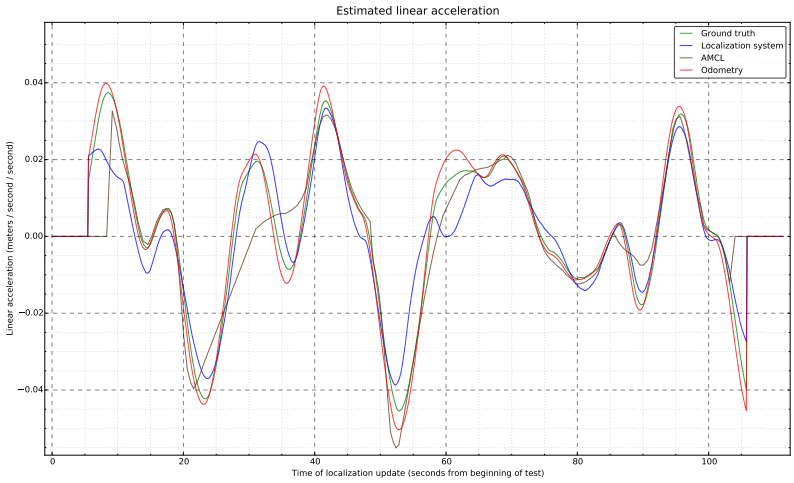


Figure A.2.4.12: Estimated linear acceleration

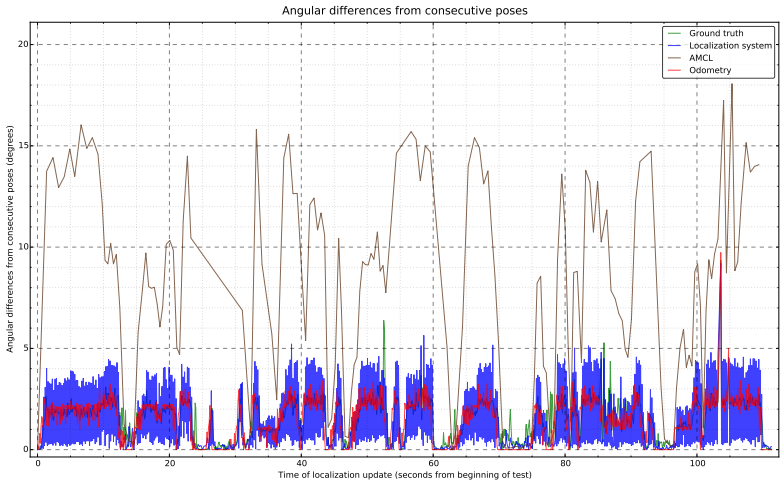


Figure A.2.4.13: Angular differences between consecutive pose estimations

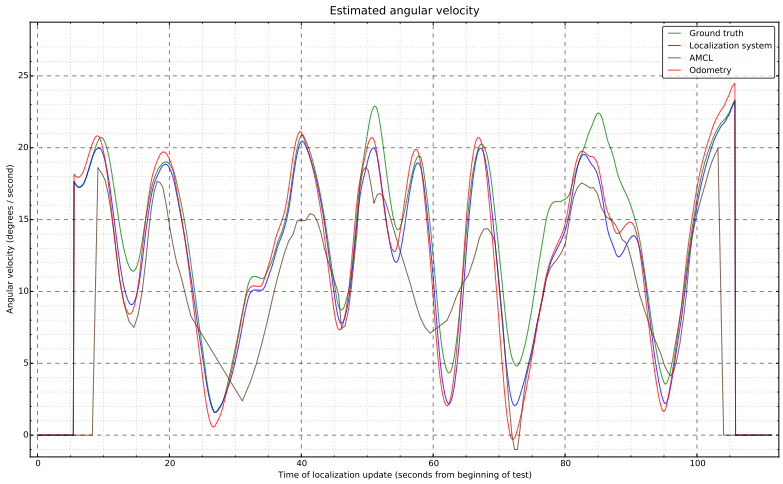


Figure A.2.4.14: Estimated angular velocity

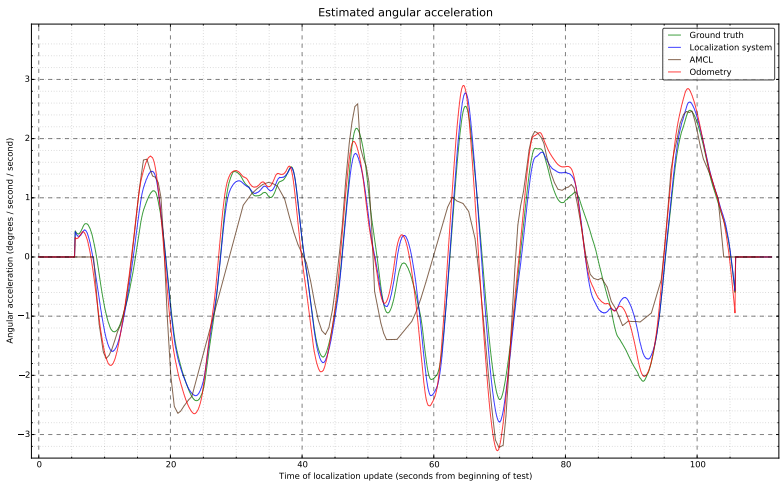


Figure A.2.4.15: Estimated angular acceleration

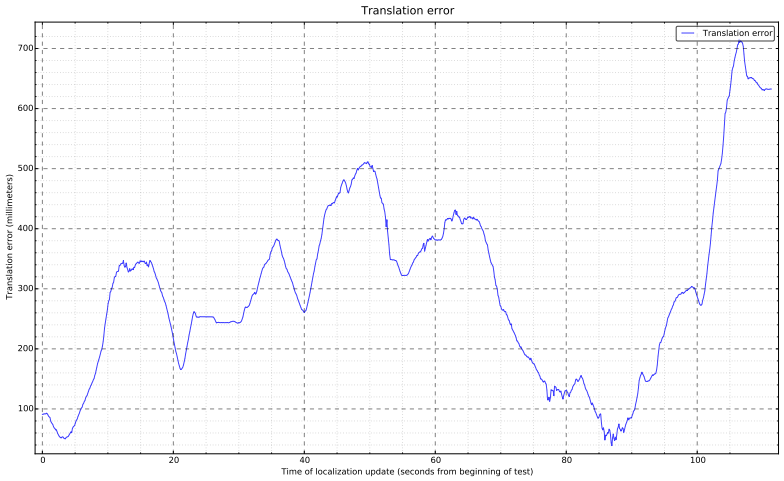


Figure A.2.4.16: Odometry translation errors

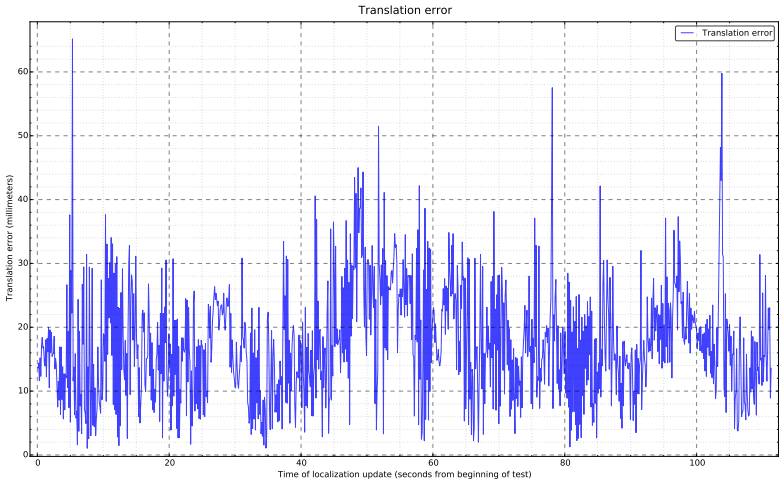


Figure A.2.4.17: Localization system translation errors

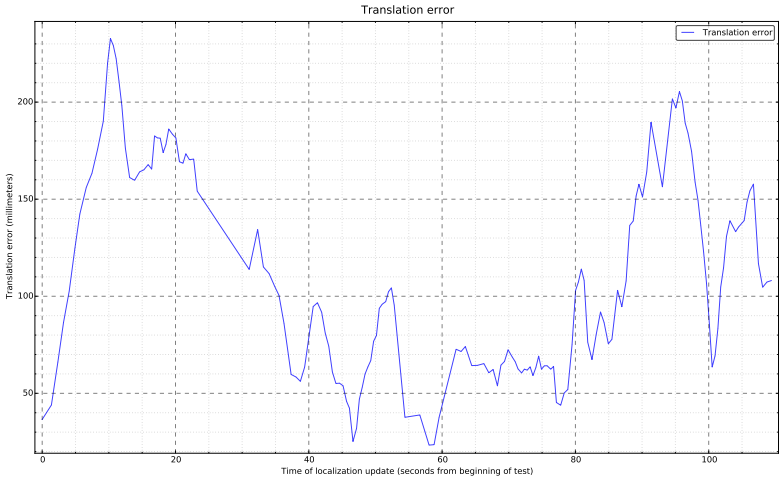


Figure A.2.4.18: AMCL translation errors

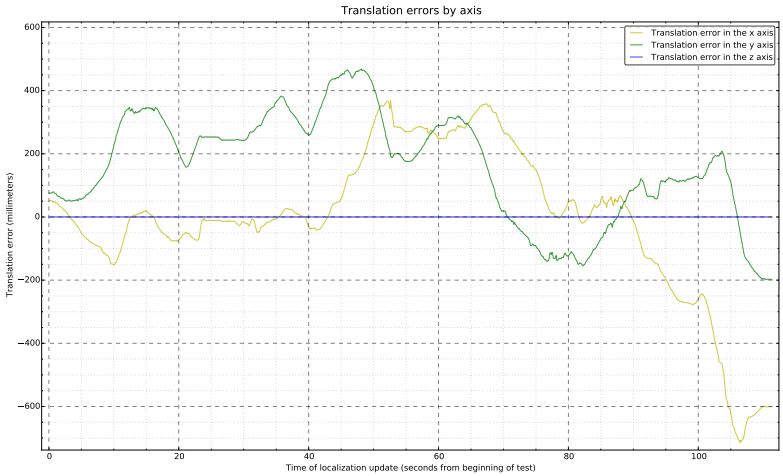


Figure A.2.4.19: Odometry translation errors components

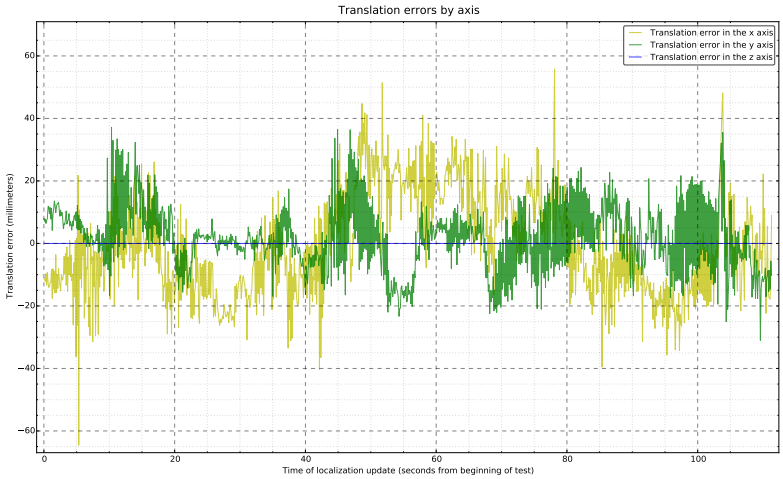


Figure A.2.4.20: Localization system translation errors components

Figure A.2.4.21: AMCL translation errors components

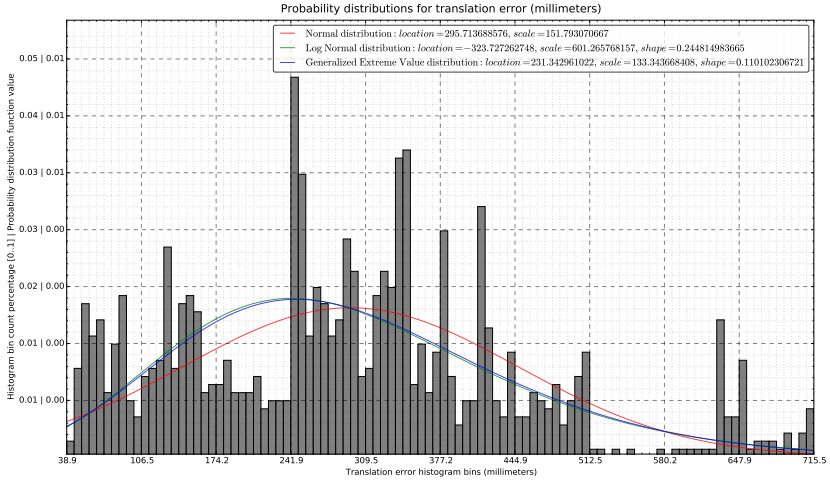


Figure A.2.4.22: Probability distributions for the odometry translation errors

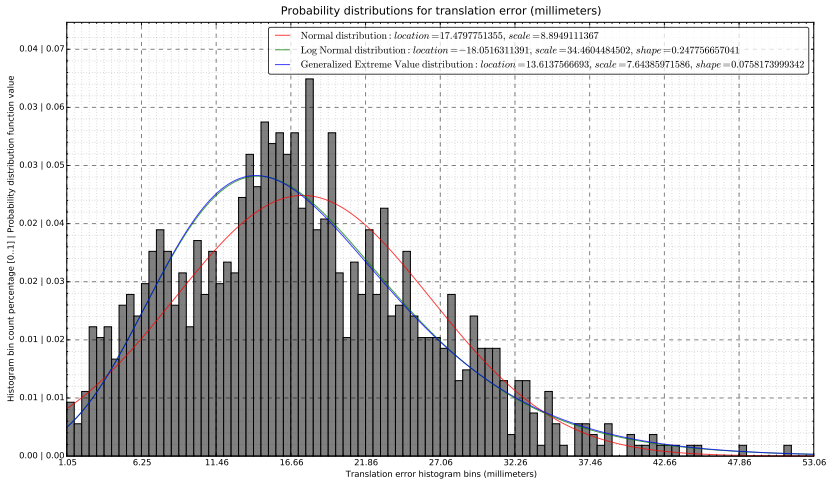


Figure A.2.4.23: Probability distributions for the localization system translation errors

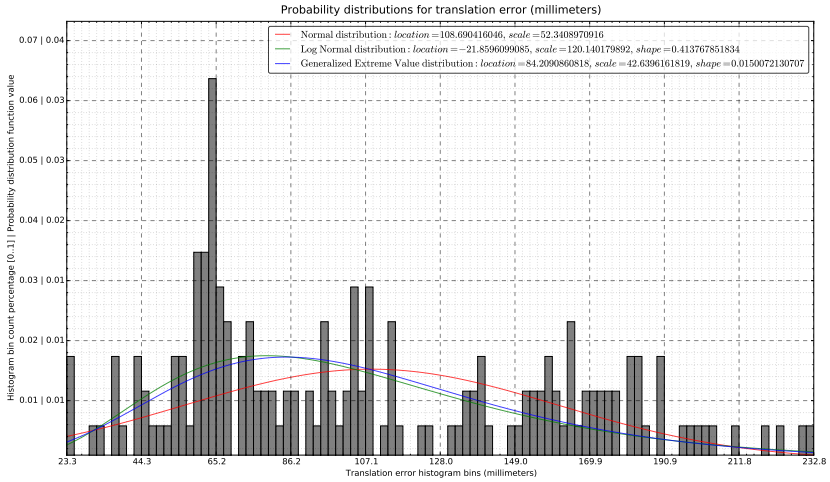


Figure A.2.4.24: Probability distributions for the AMCL translation errors

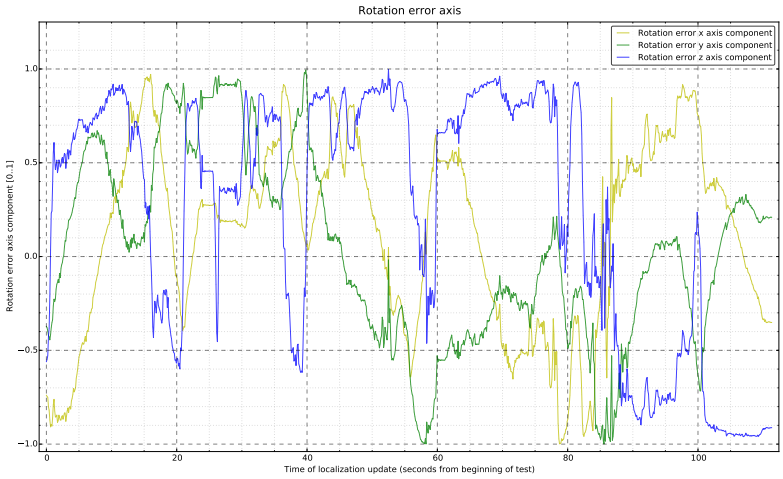


Figure A.2.4.25: Odometry rotation errors axis

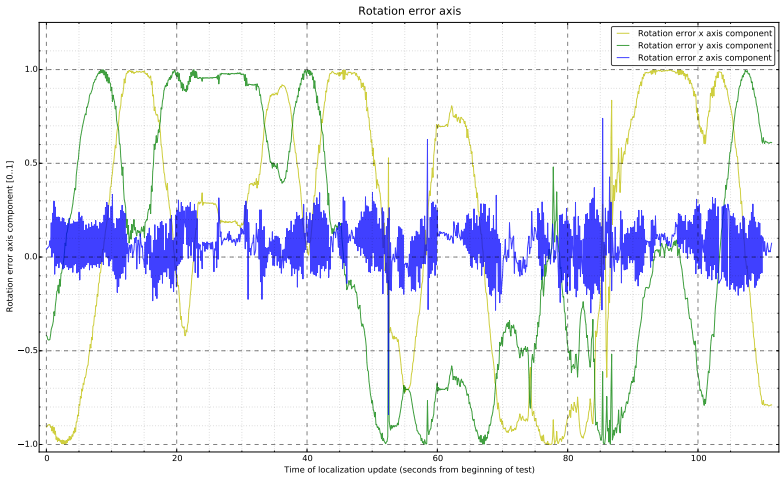


Figure A.2.4.26: Localization system rotation errors axis

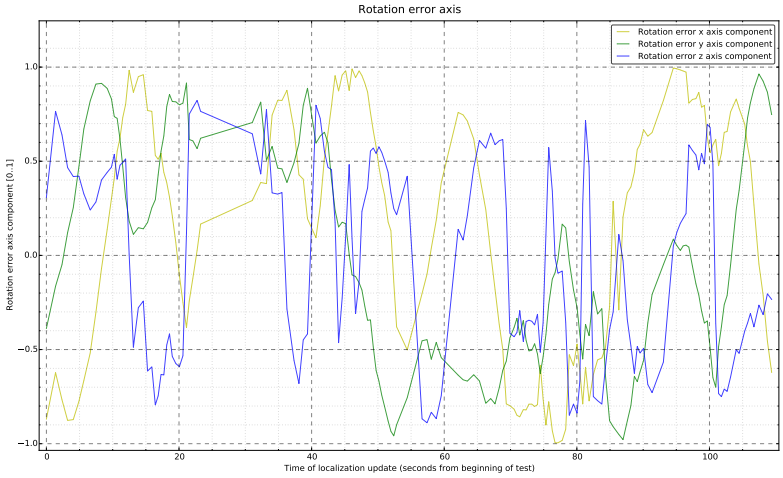


Figure A.2.4.27: AMCL rotation errors axis

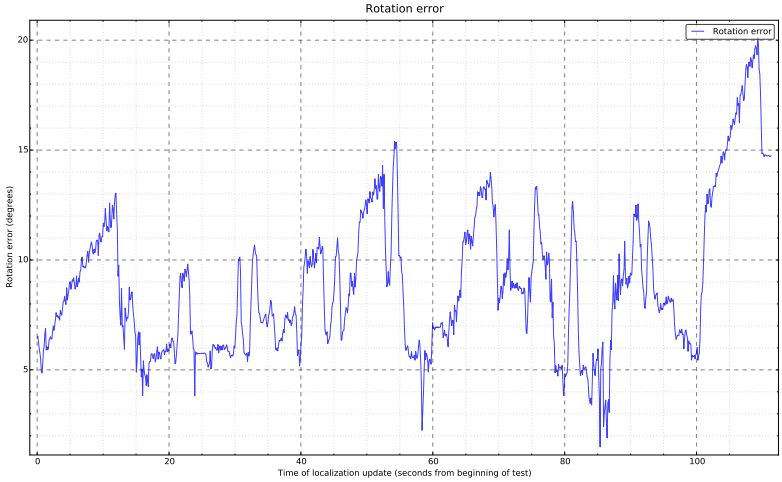


Figure A.2.4.28: Odometry rotation errors

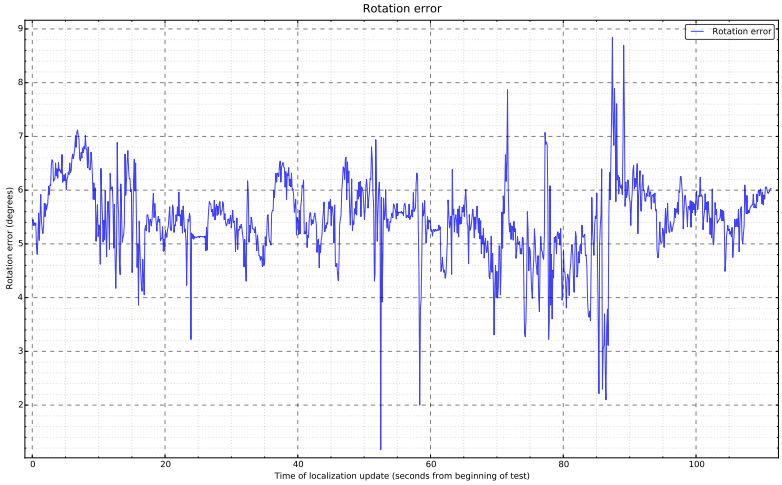


Figure A.2.4.29: Localization system rotation errors

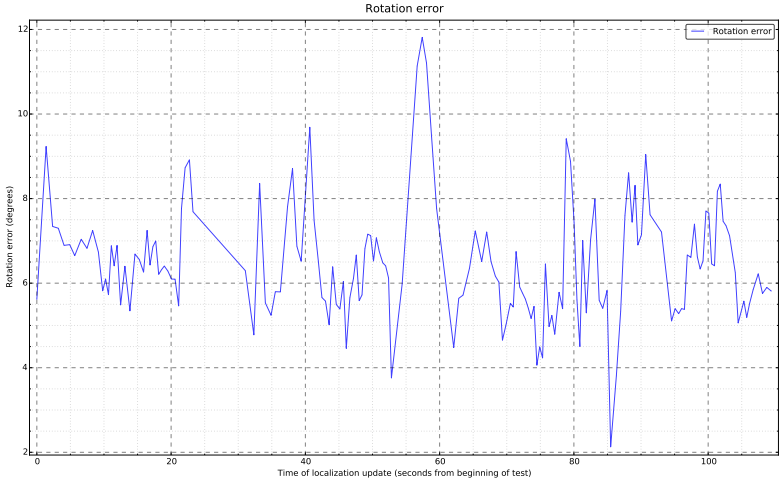


Figure A.2.4.30: AMCL rotation errors

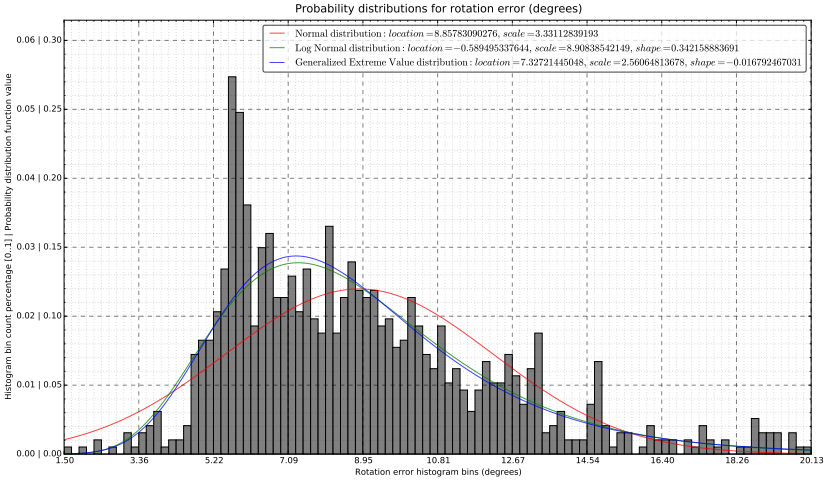


Figure A.2.4.31: Probability distributions for the odometry rotation errors

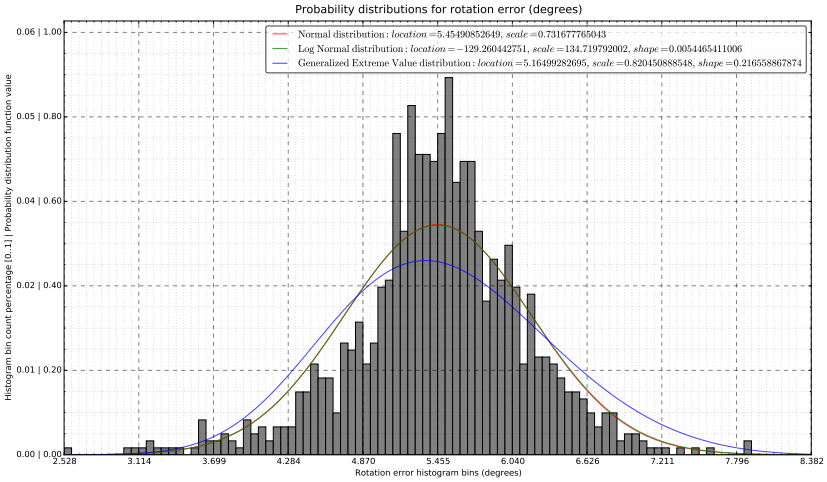


Figure A.2.4.32: Probability distributions for the localization system rotation errors

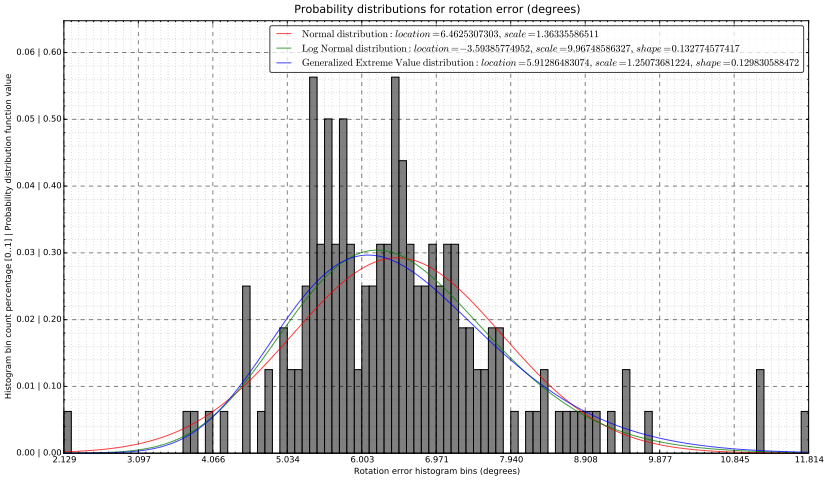


Figure A.2.4.33: Probability distributions for the AMCL rotation errors

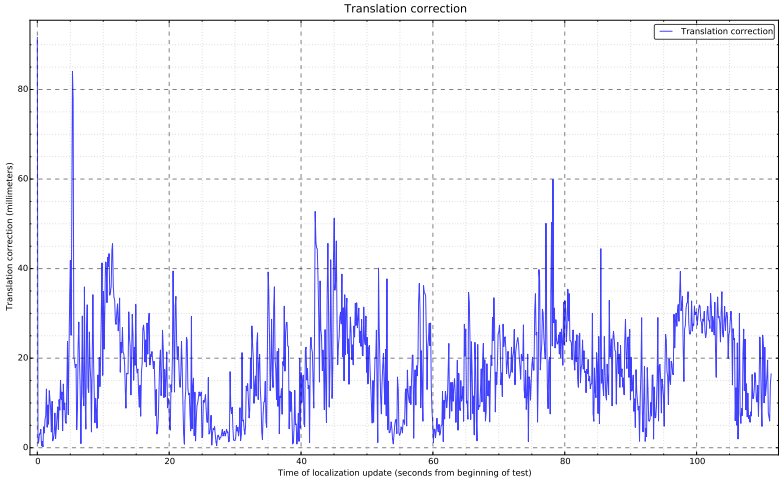


Figure A.2.4.34: Translation corrections performed by the localization system

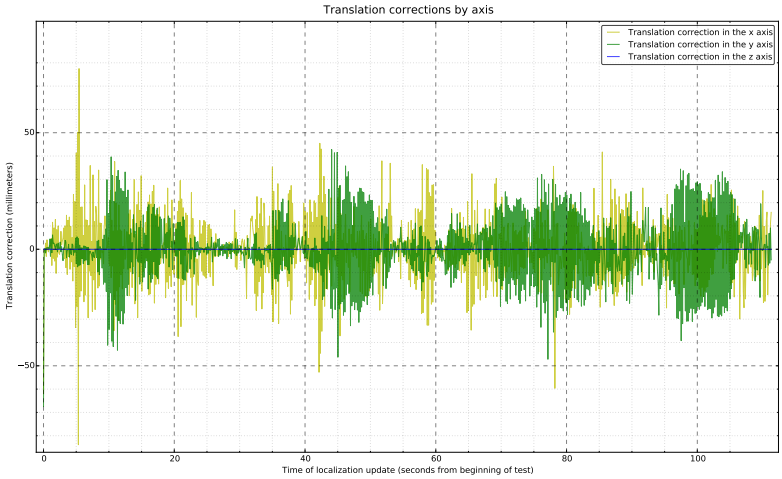


Figure A.2.4.35: Translation corrections components performed by the localization system

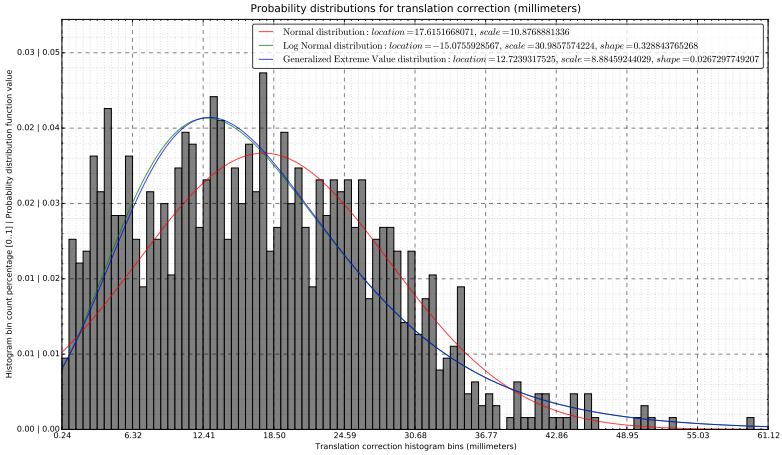


Figure A.2.4.36: Probability distributions for the translation corrections performed by the localization system

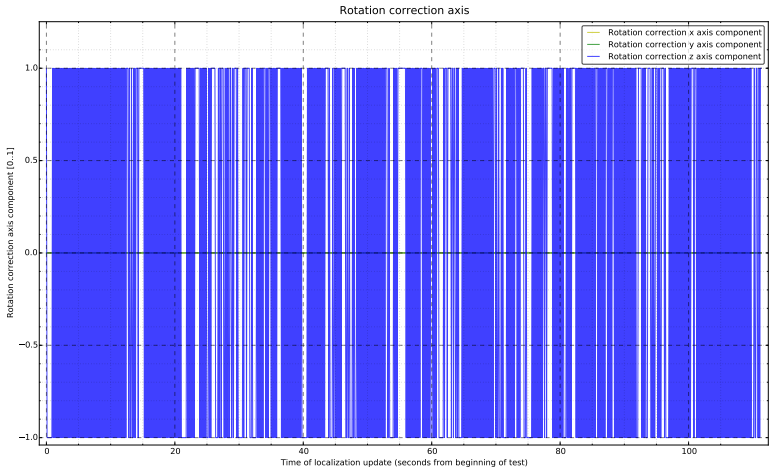


Figure A.2.4.37: Rotation corrections (axis) performed by the localization system

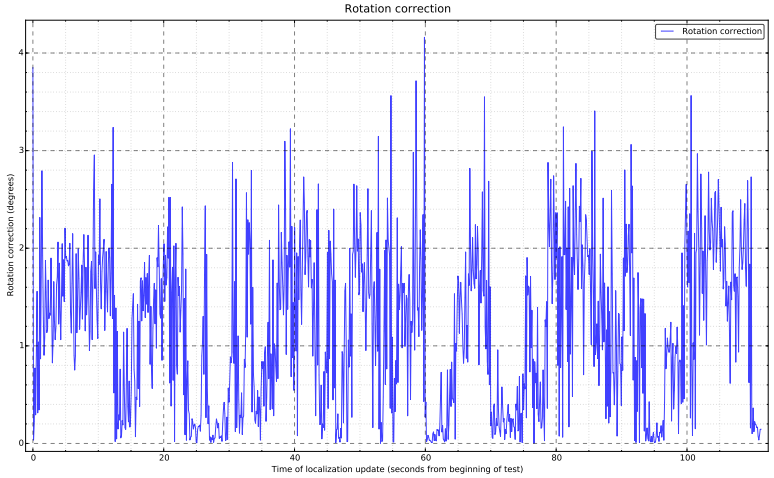


Figure A.2.4.38: Rotation corrections performed by the localization system

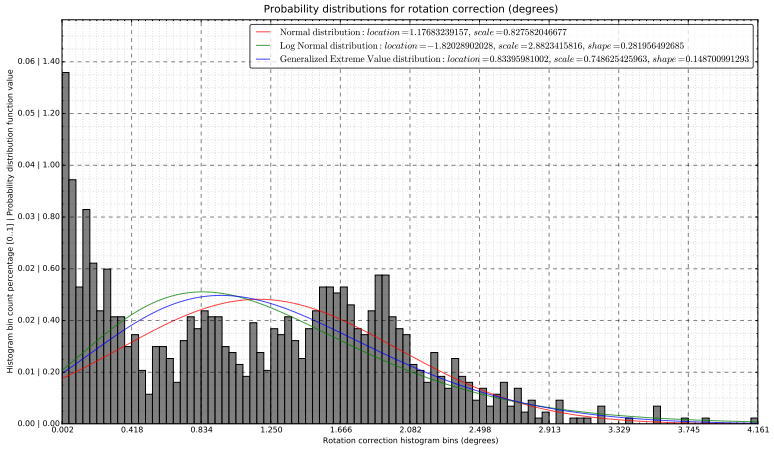


Figure A.2.4.39: Probability distributions for the rotation corrections performed by the localization system

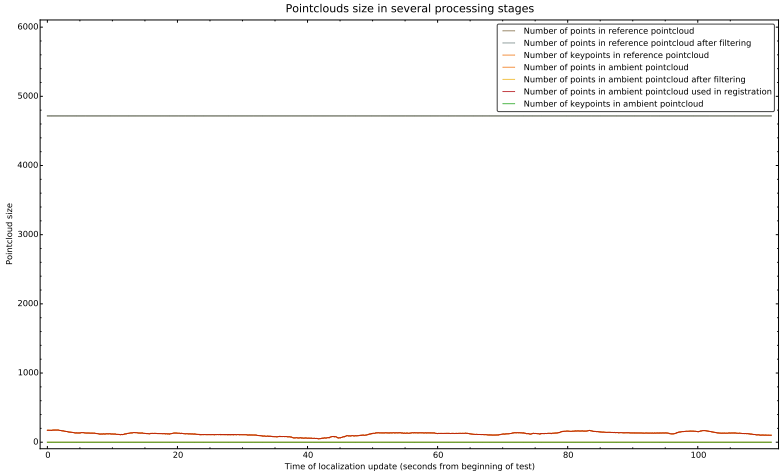


Figure A.2.4.40: Point clouds size in several of the localization system processing stages

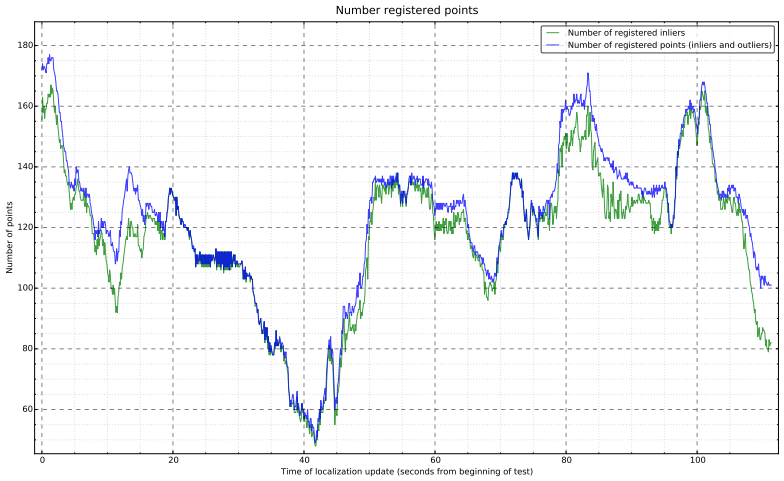


Figure A.2.4.41: Number of registered points

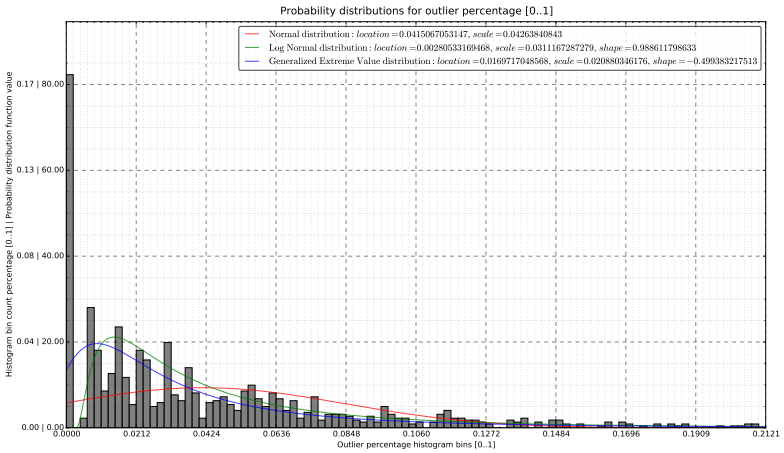


Figure A.2.4.42: Probability distributions for the ambient point cloud outlier percentage

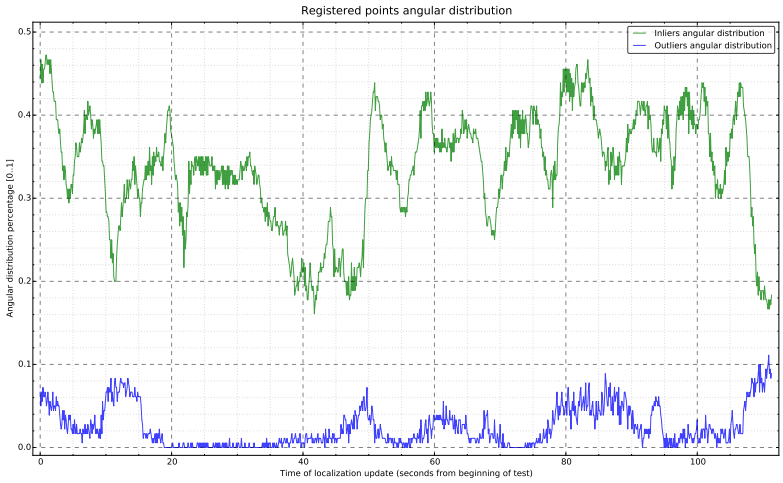


Figure A.2.4.43: Angular distribution percentage of registered points

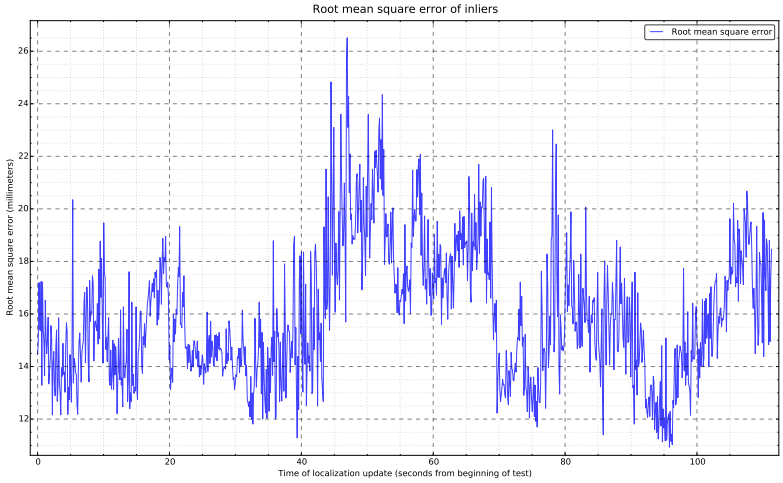


Figure A.2.4.44: Root Mean Square Error of the inliers

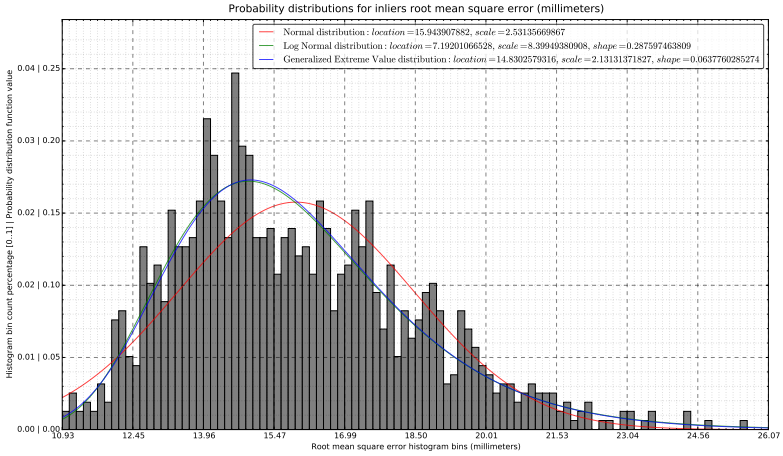


Figure A.2.4.45: Probability distributions for the Root Mean Square Error of the inliers

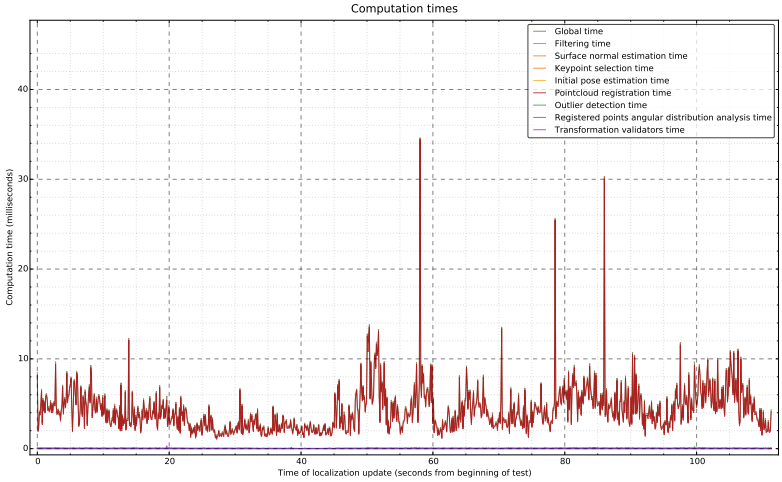


Figure A.2.4.46: Localization system computation times

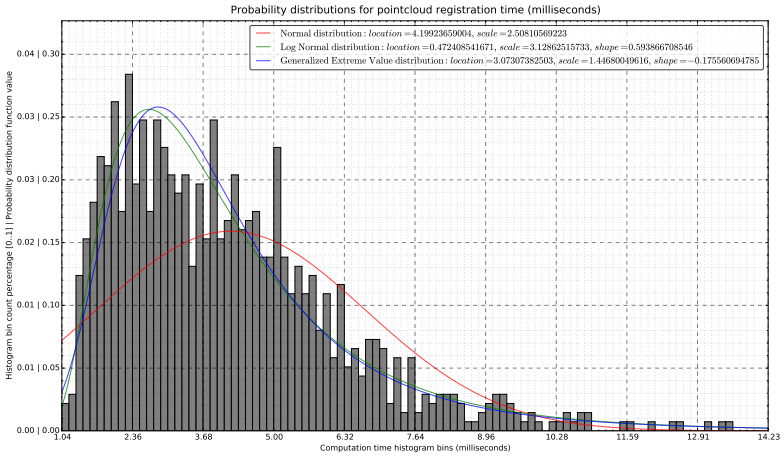


Figure A.2.4.47: Probability distributions for the point cloud registration time

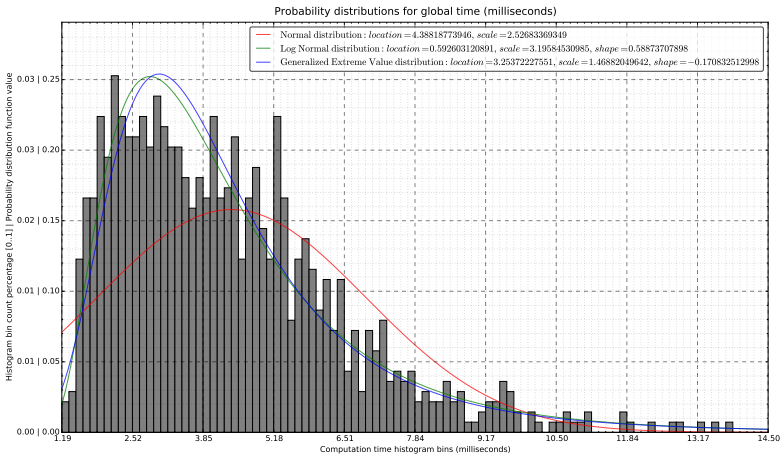


Figure A.2.4.48: Probability distributions for the localization system global computation time

A.3 Guardian simulator tests

The next subsections show some of the most relevant results retrieved with the Guardian simulator (Gazebo 1.9) in the structured indoor environment.

A.3.1 Wall following path using the simulated Guardian robot at 30 cm/s in the static environment

The figures in this section show the detailed results of the test performed with the simulated Guardian robot with a 2 mm resolution map in the static structured indoor environment following the lower and right wall at 30 cm/s.

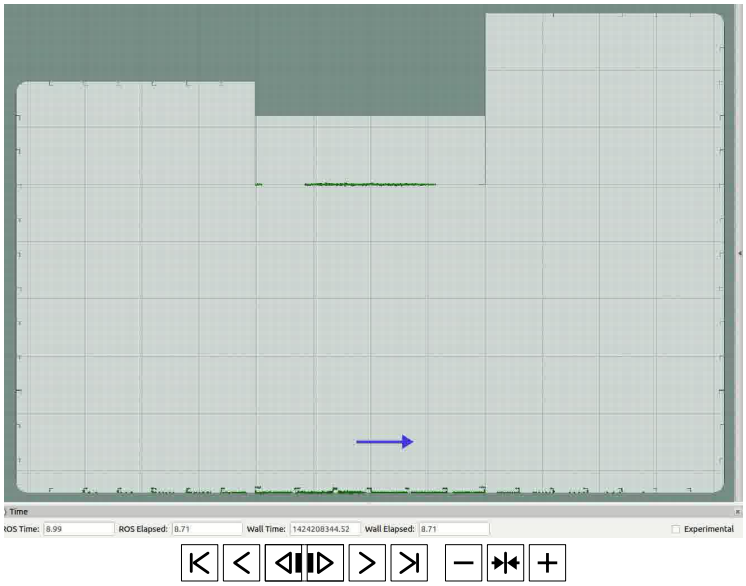


Figure A.3.1.1: Animation of laser scans using the localization system poses

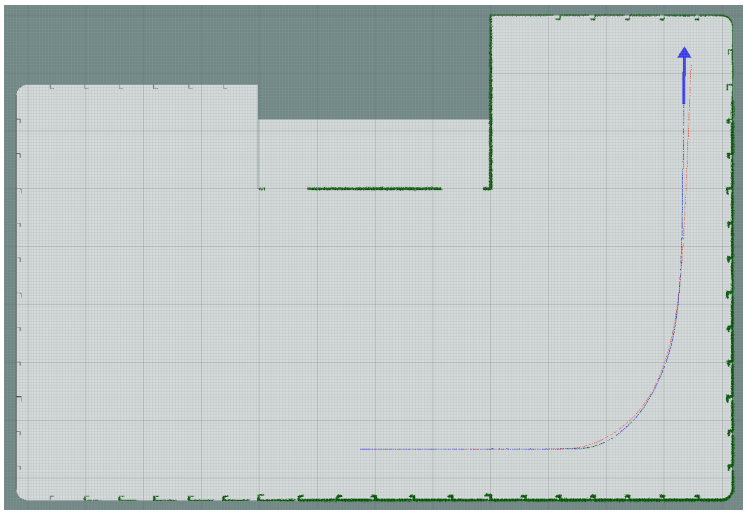


Figure A.3.1.2: Laser scans assembled on top of the map using the localization system poses

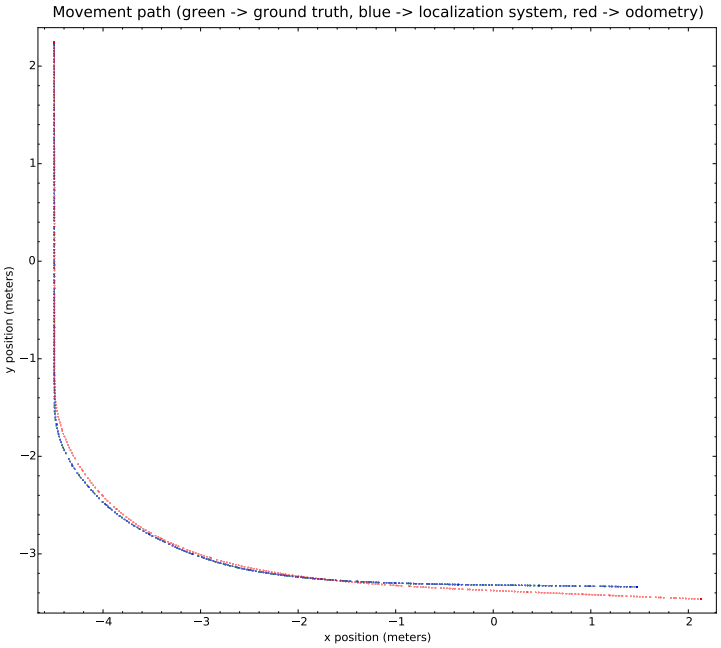


Figure A.3.1.3: Poses estimated by the ground truth, localization system and odometry

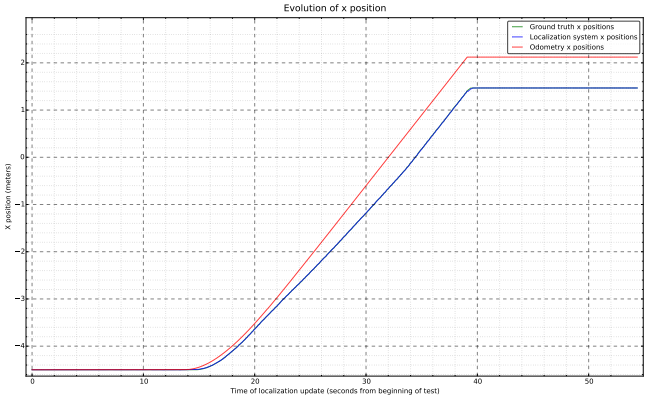


Figure A.3.1.4: Evolution of x position over time

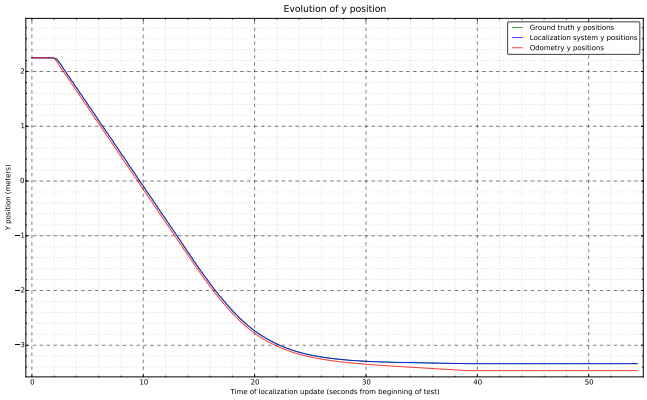


Figure A.3.1.5: Evolution of y position over time

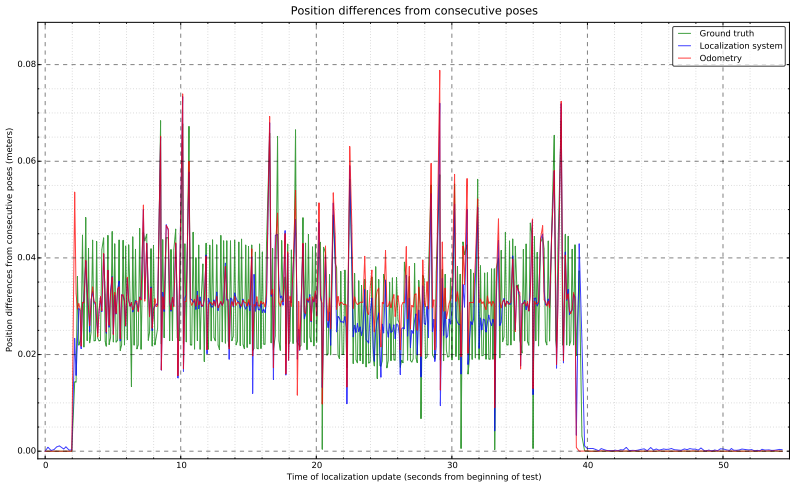


Figure A.3.1.6: Distance traveled between consecutive pose estimations

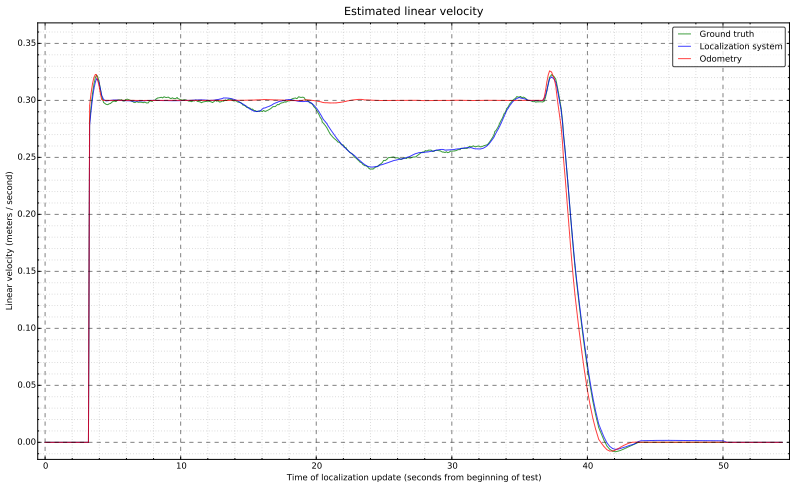


Figure A.3.1.7: Estimated linear velocity

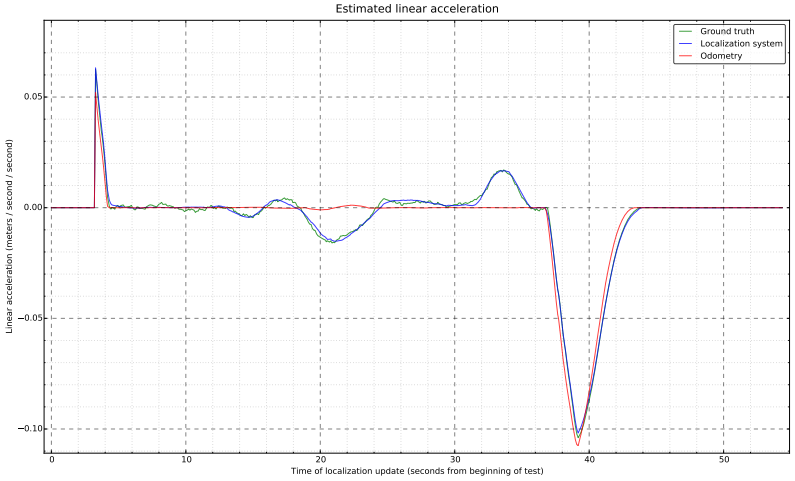


Figure A.3.1.8: Estimated linear acceleration

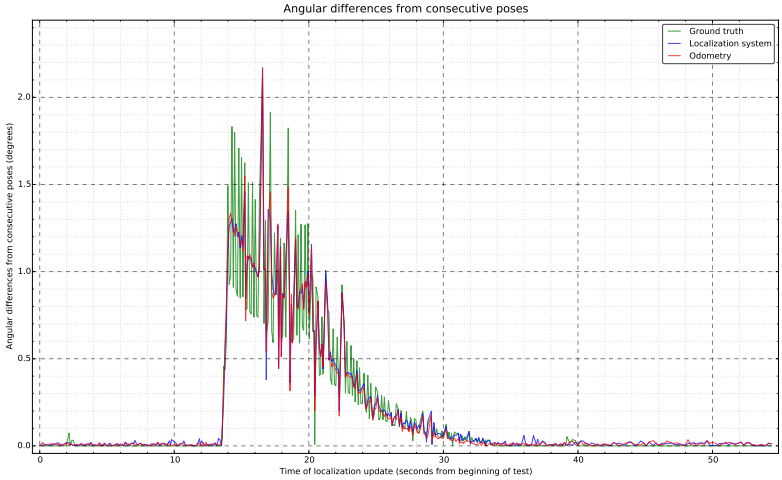


Figure A.3.1.9: Angular differences between consecutive pose estimations

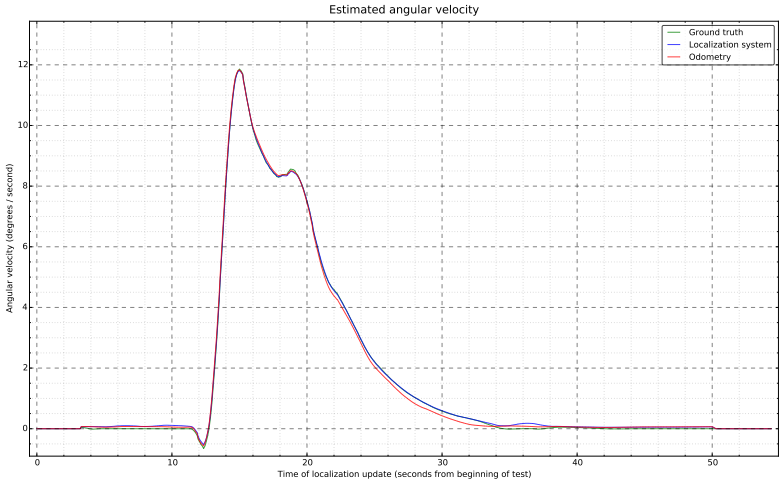


Figure A.3.1.10: Estimated angular velocity

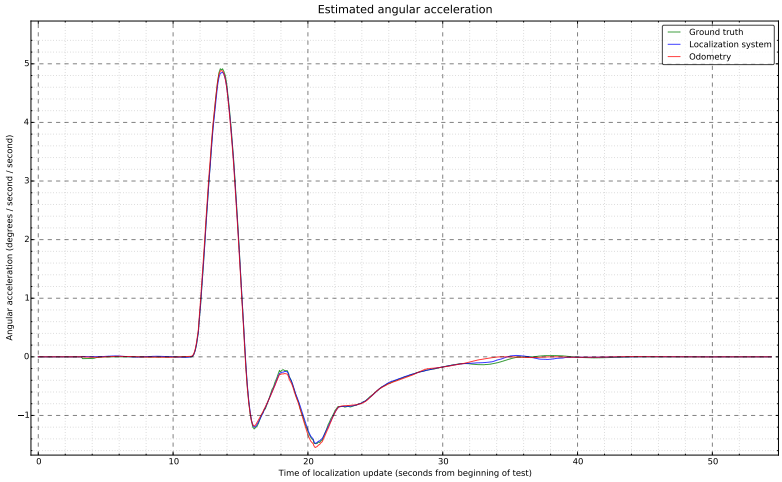


Figure A.3.1.11: Estimated angular acceleration

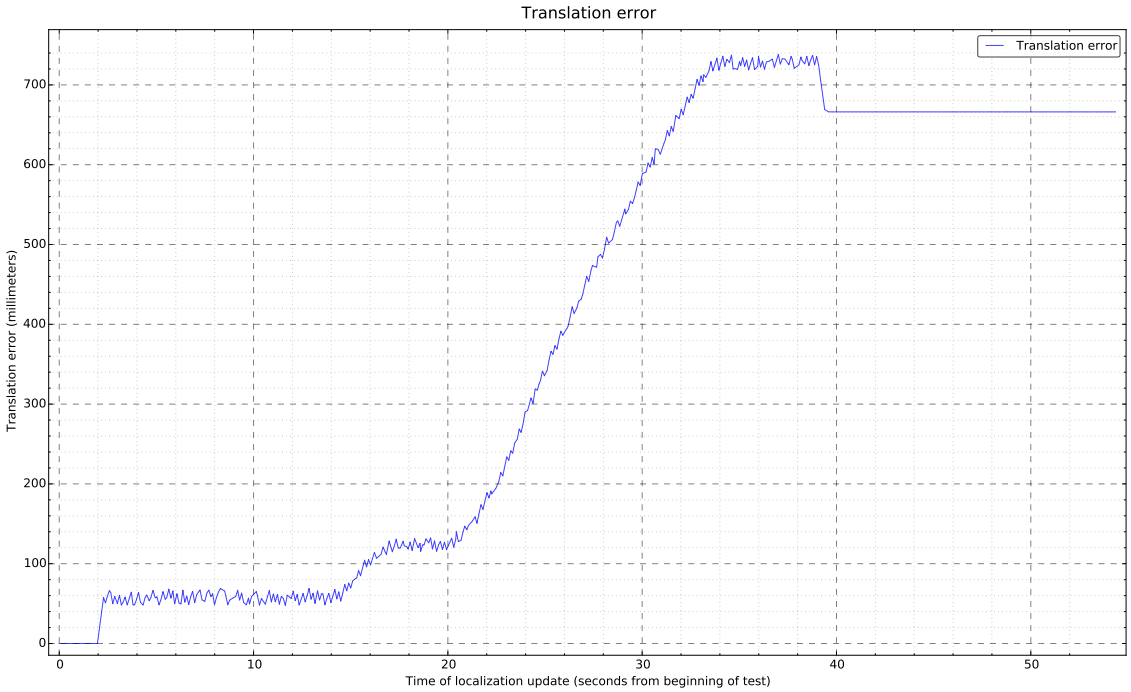


Figure A.3.1.12: Odometry translation errors

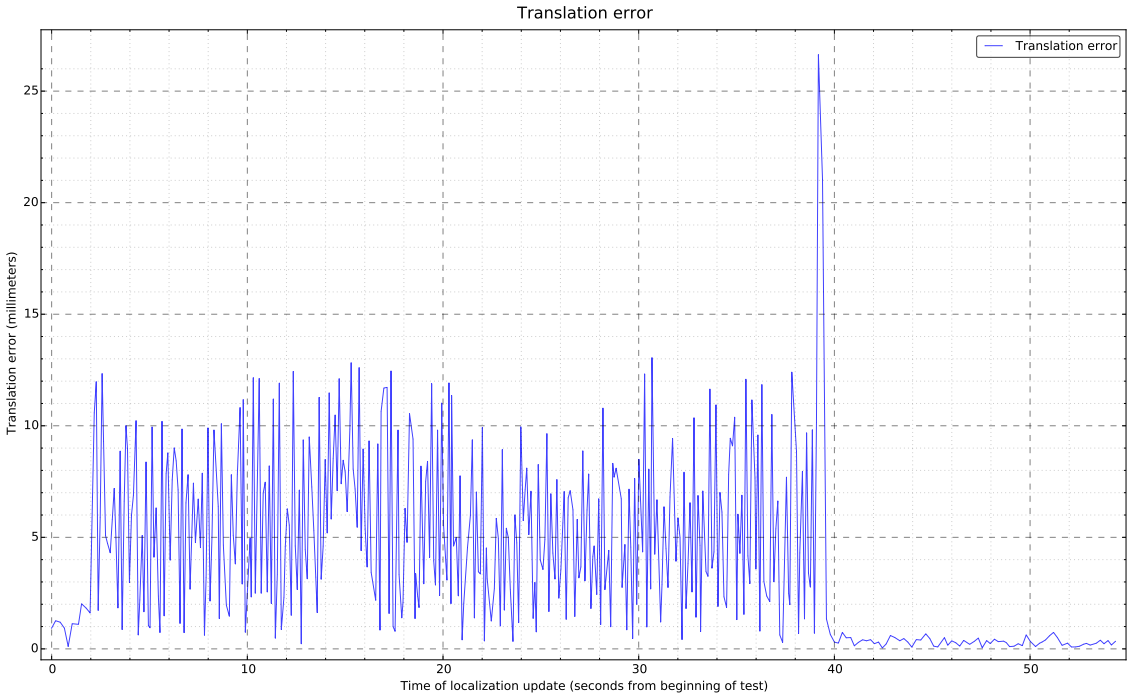


Figure A.3.1.13: Localization system translation errors

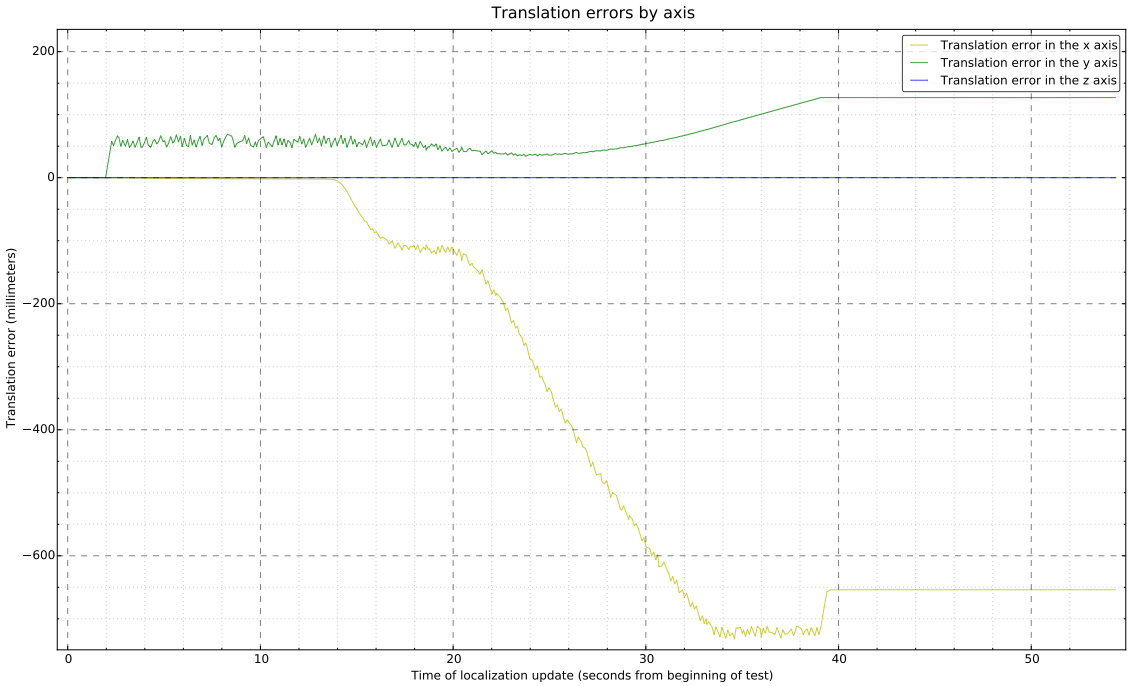


Figure A.3.1.14: Odometry translation errors components

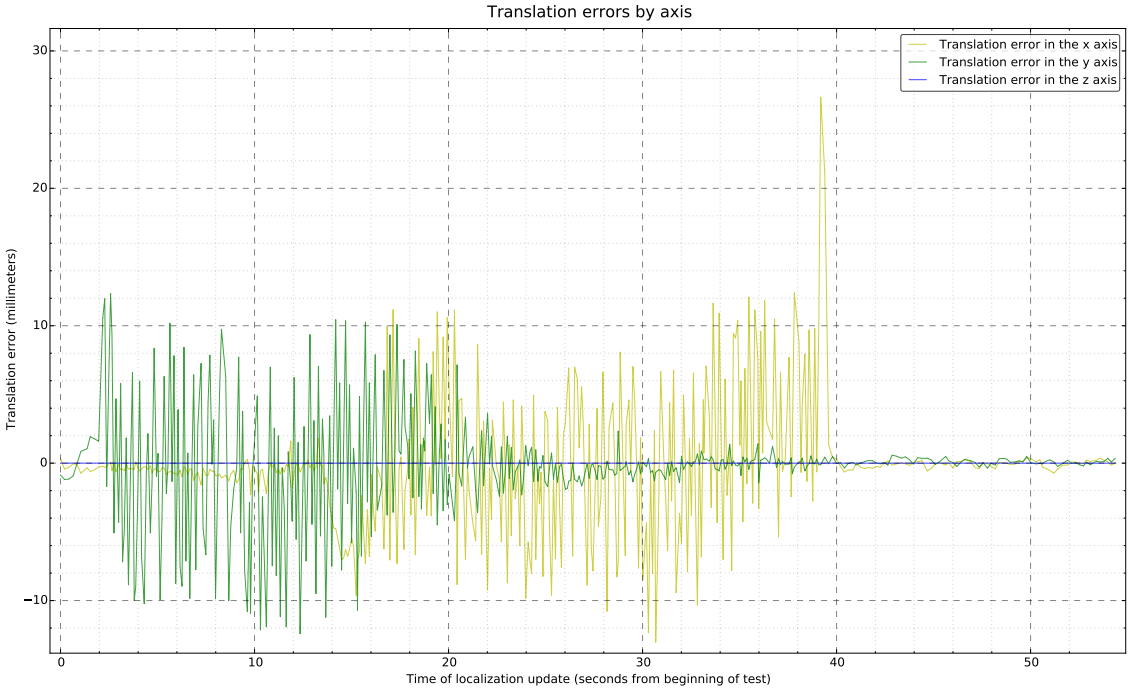


Figure A.3.1.15: Localization system translation errors components

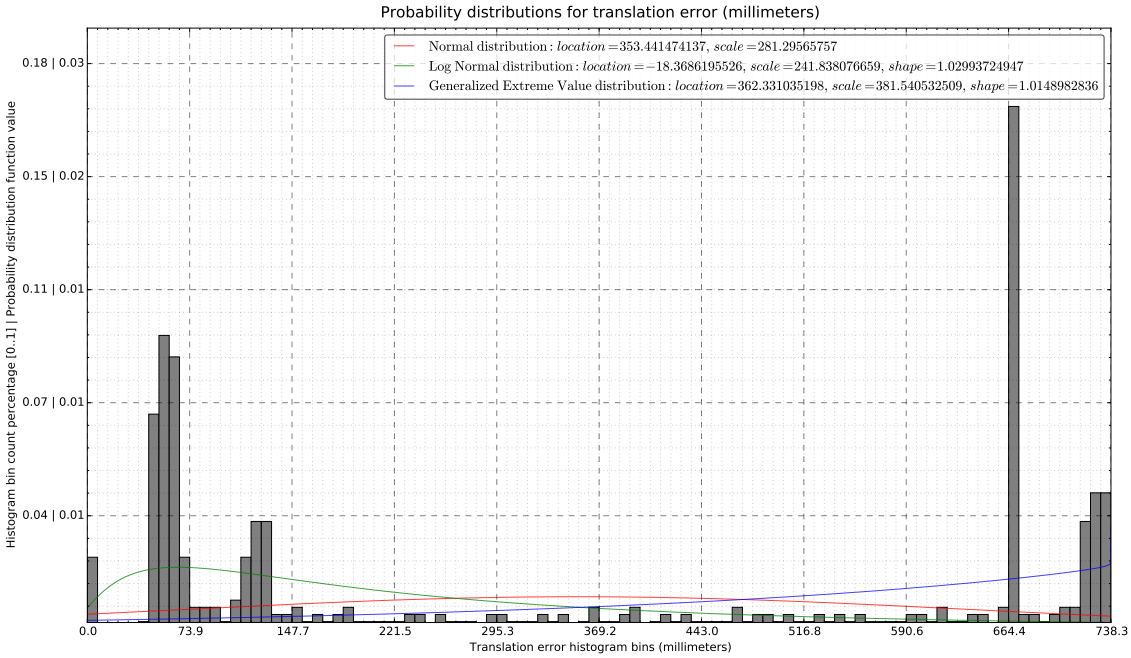


Figure A.3.1.16: Probability distributions for the odometry translation errors

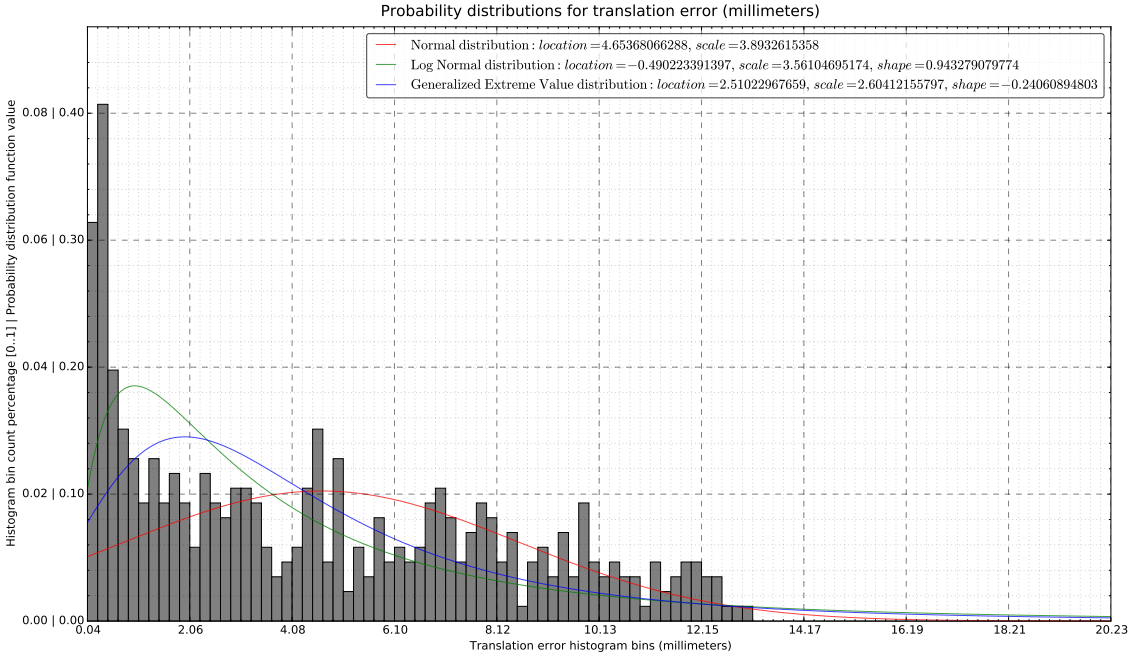


Figure A.3.1.17: Probability distributions for the localization system translation errors

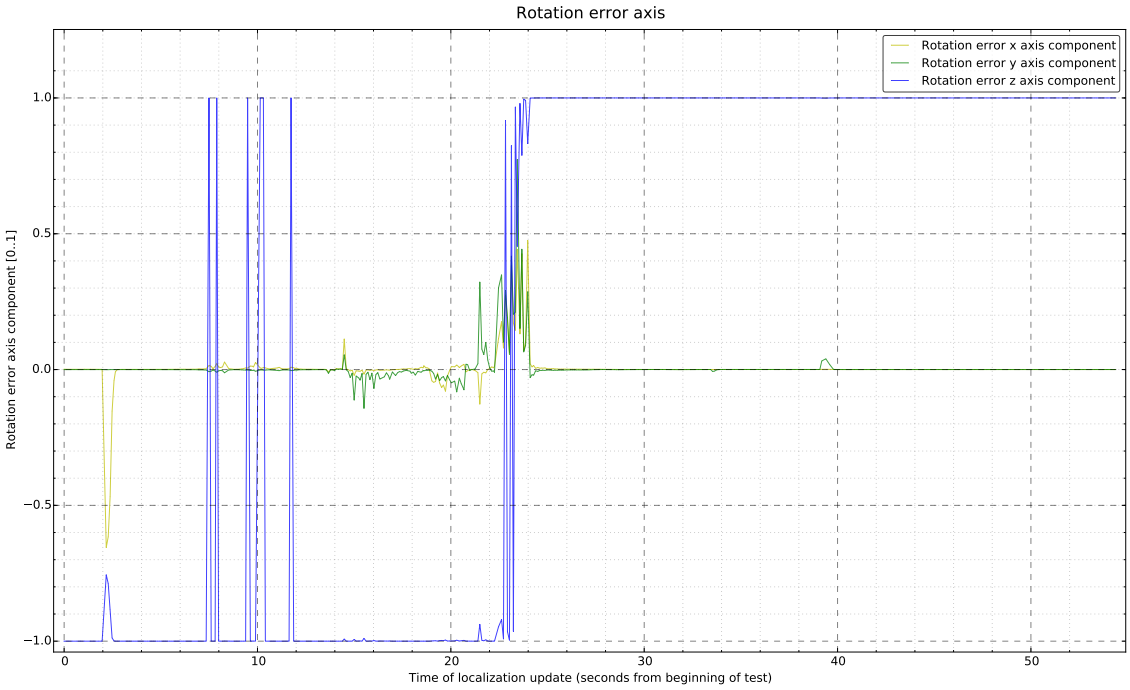


Figure A.3.1.18: Odometry rotation errors axis

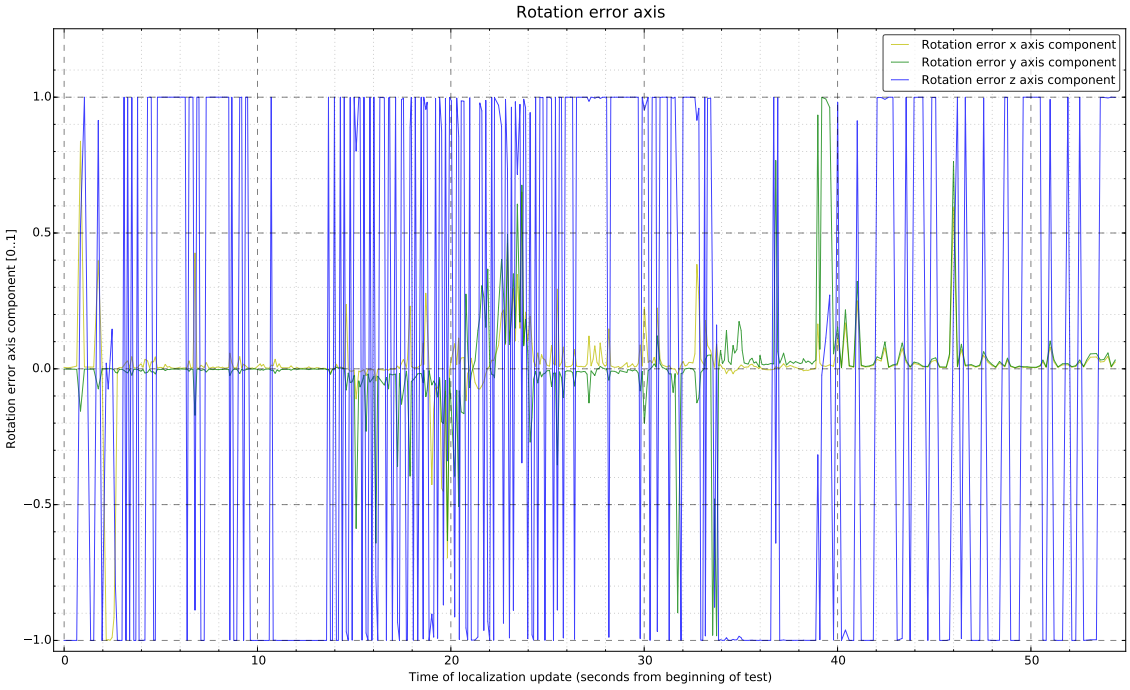


Figure A.3.1.19: Localization system rotation errors axis

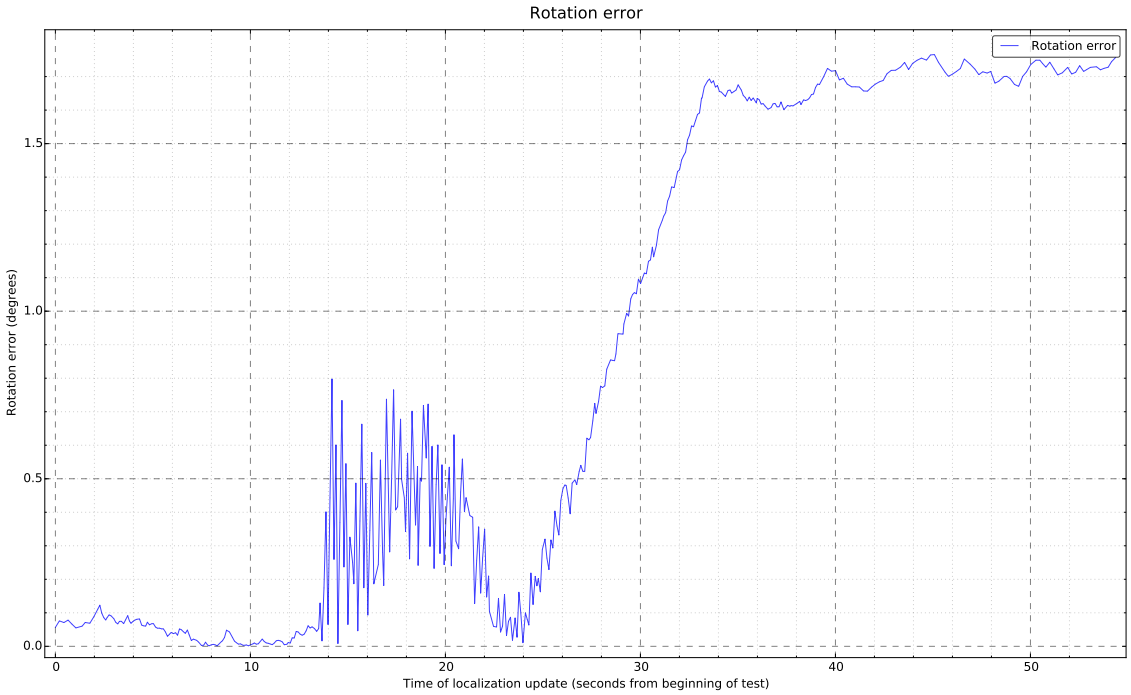


Figure A.3.1.20: Odometry rotation errors

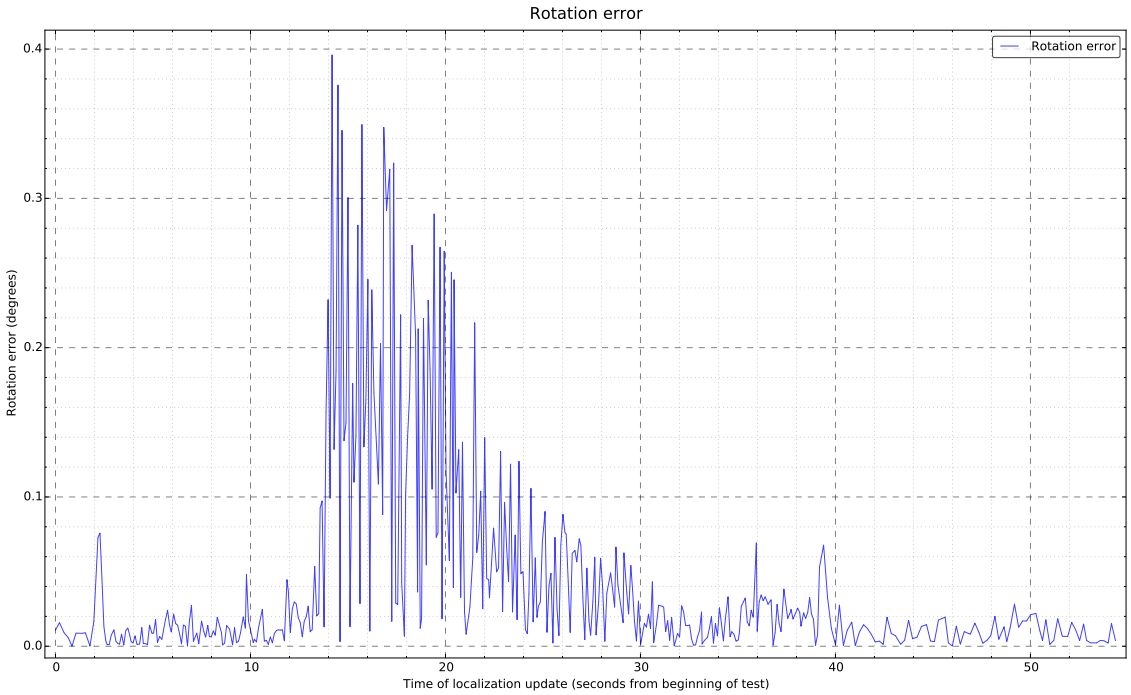


Figure A.3.1.21: Localization system rotation errors

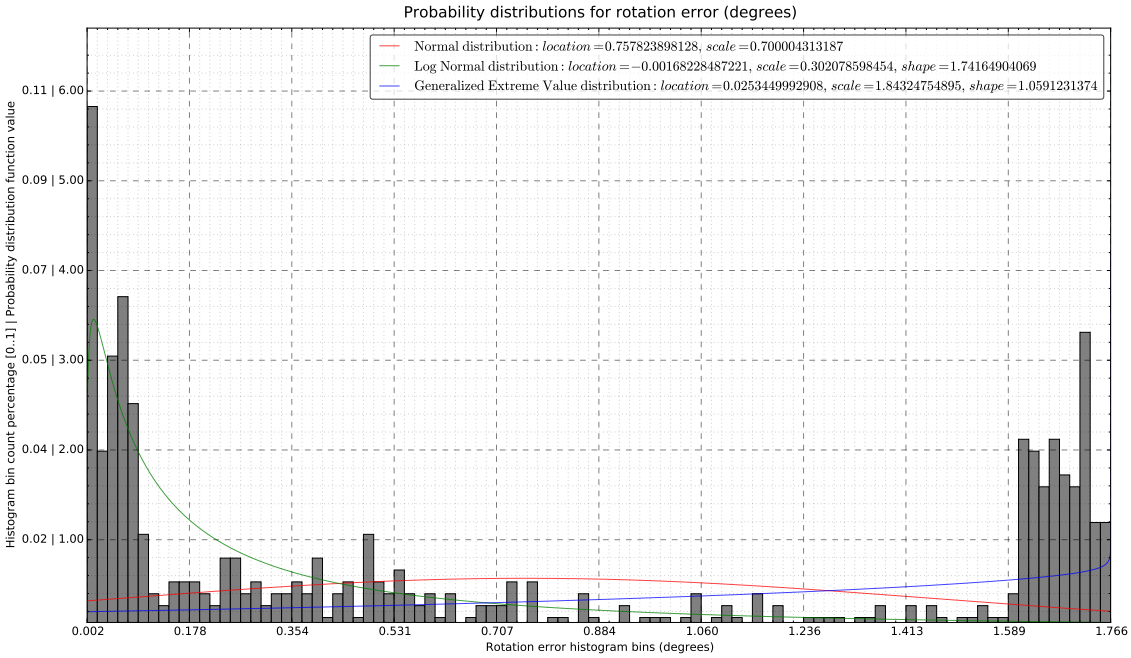


Figure A.3.1.22: Probability distributions for the odometry rotation errors

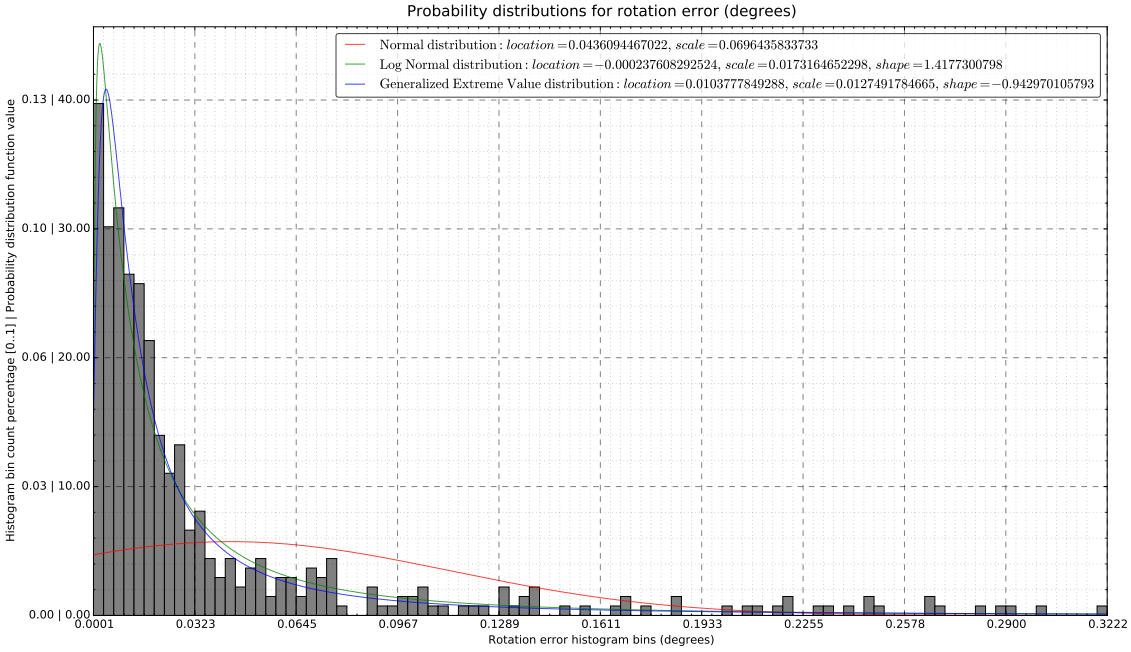


Figure A.3.1.23: Probability distributions for the localization system rotation errors

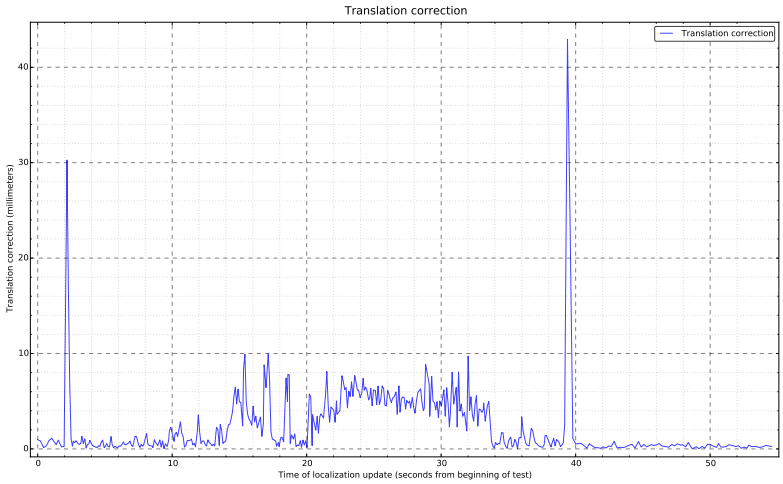


Figure A.3.1.24: Translation corrections performed by the localization system

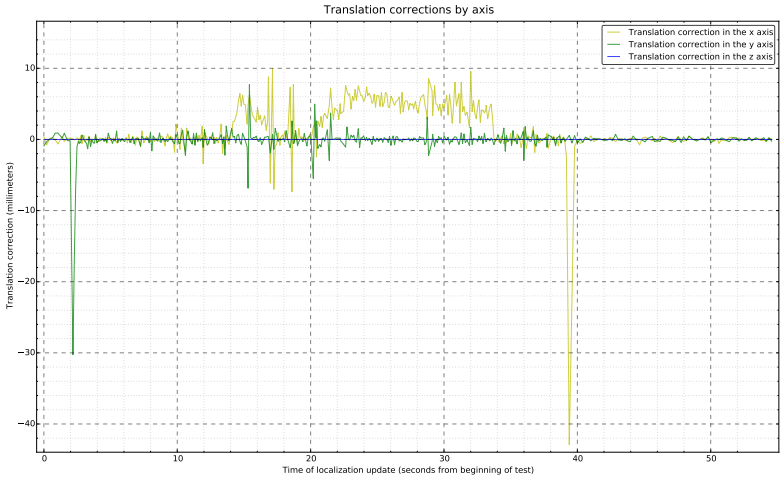


Figure A.3.1.25: Translation corrections components performed by the localization system

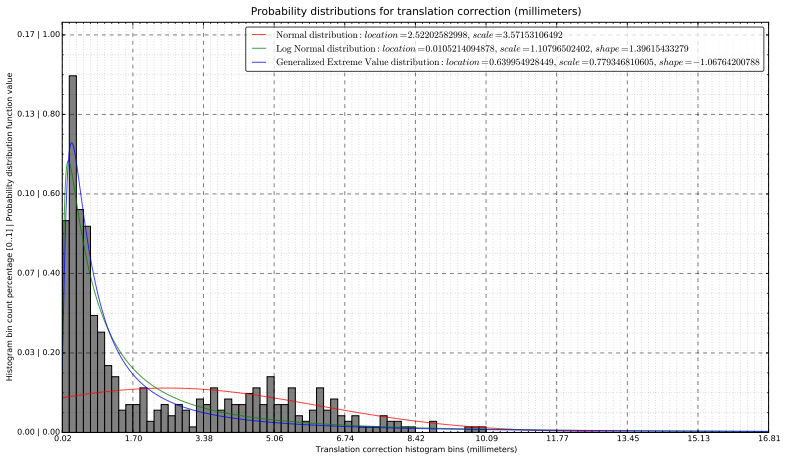


Figure A.3.1.26: Probability distributions for the translation corrections performed by the localization system

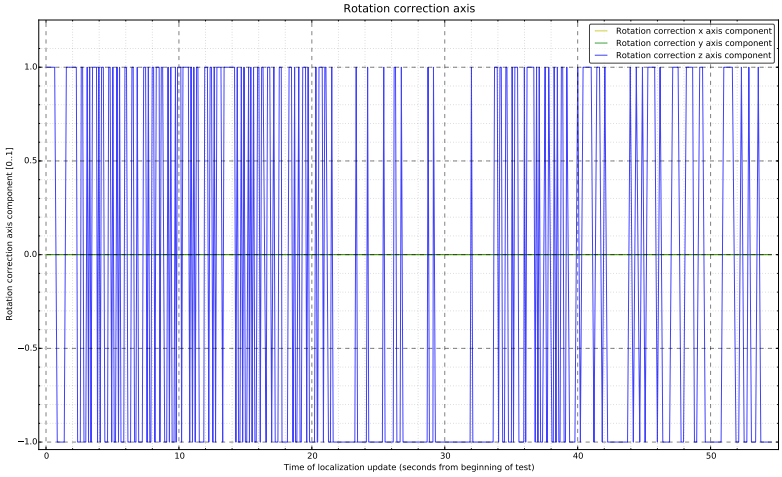


Figure A.3.1.27: Rotation corrections (axis) performed by the localization system

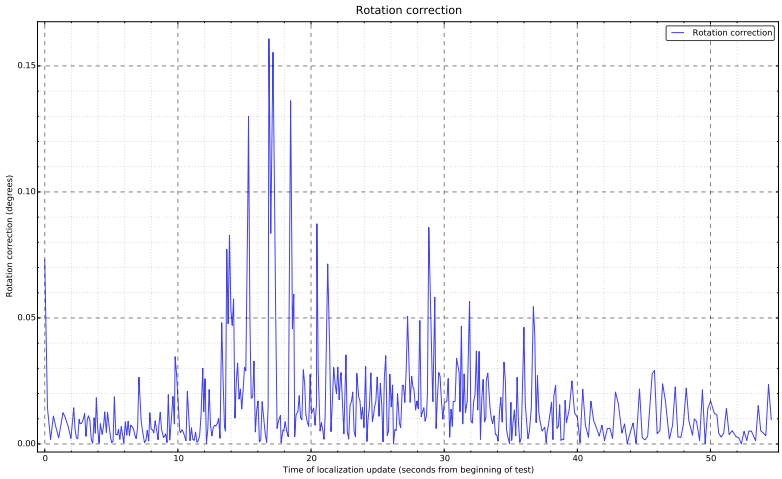


Figure A.3.1.28: Rotation corrections performed by the localization system

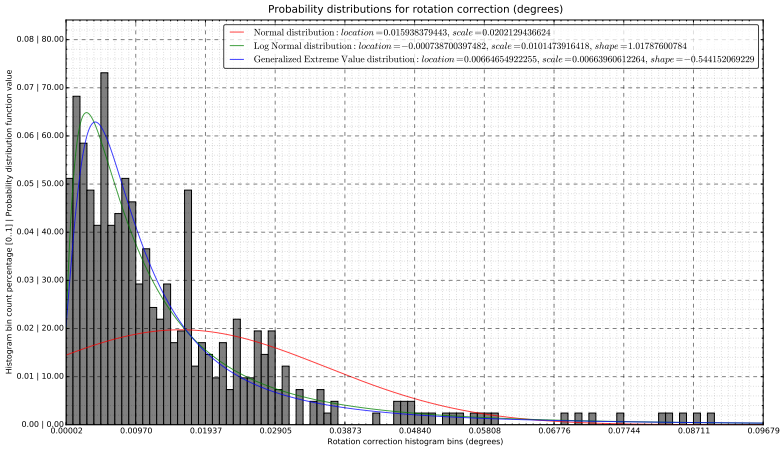


Figure A.3.1.29: Probability distributions for the rotation corrections performed by the localization system

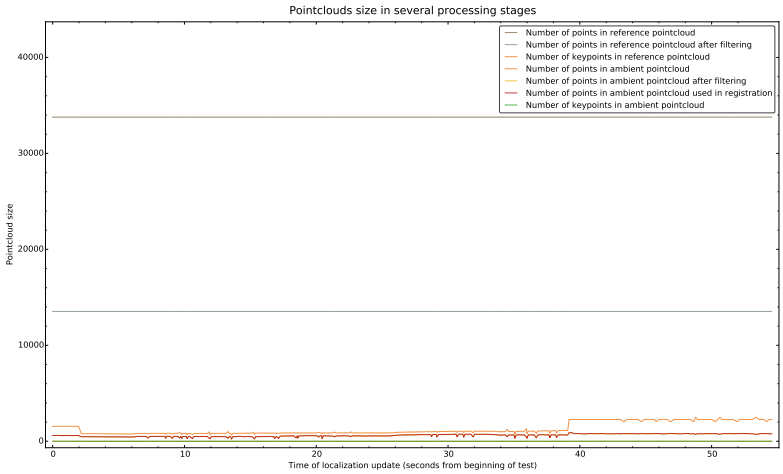


Figure A.3.1.30: Point clouds size in several of the localization system processing stages

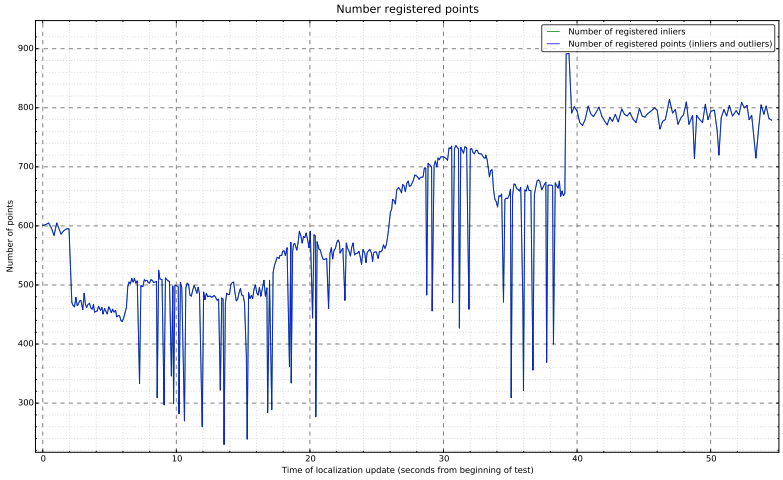


Figure A.3.1.31: Number of registered points

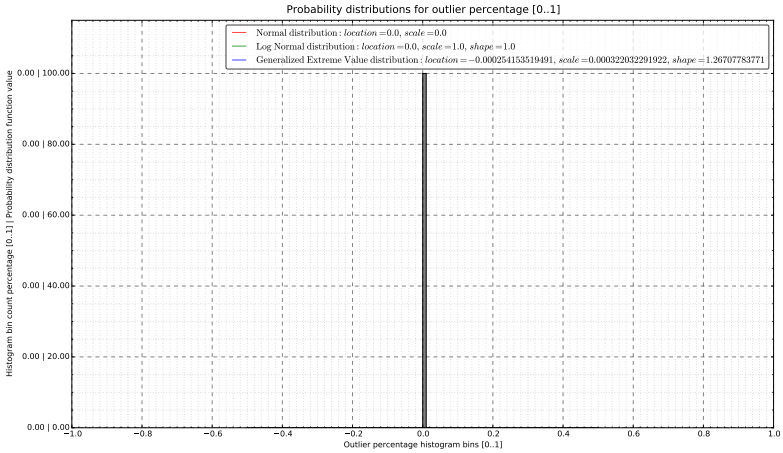


Figure A.3.1.32: Probability distributions for the ambient point cloud outlier percentage

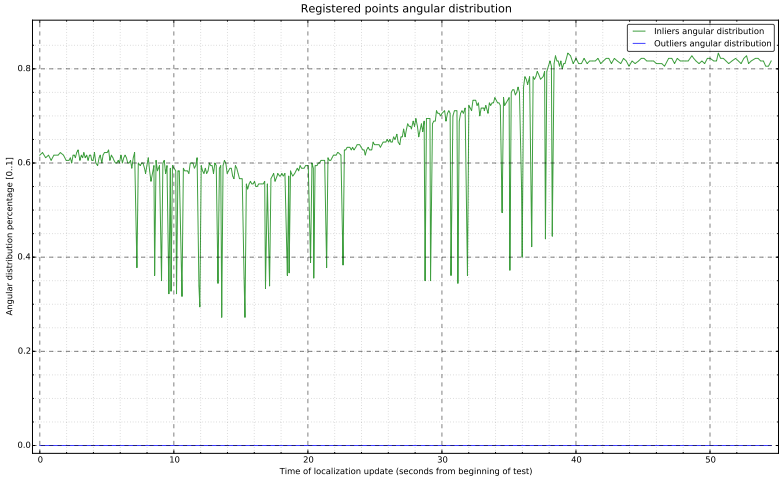


Figure A.3.1.33: Angular distribution percentage of registered points

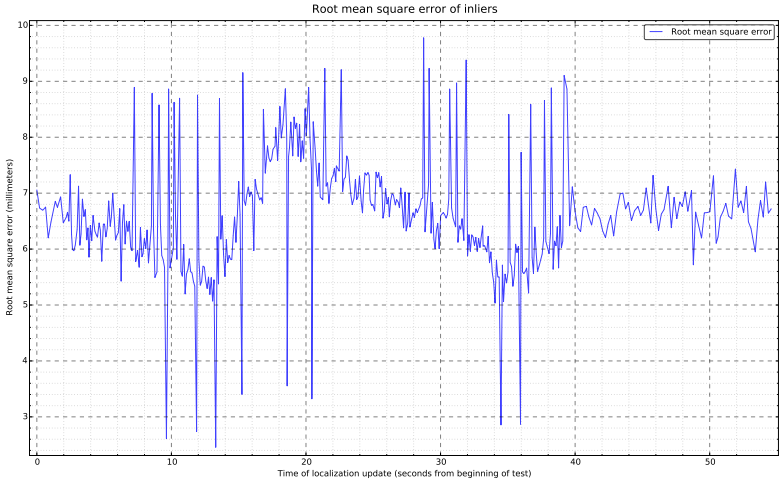


Figure A.3.1.34: Root Mean Square Error of the inliers

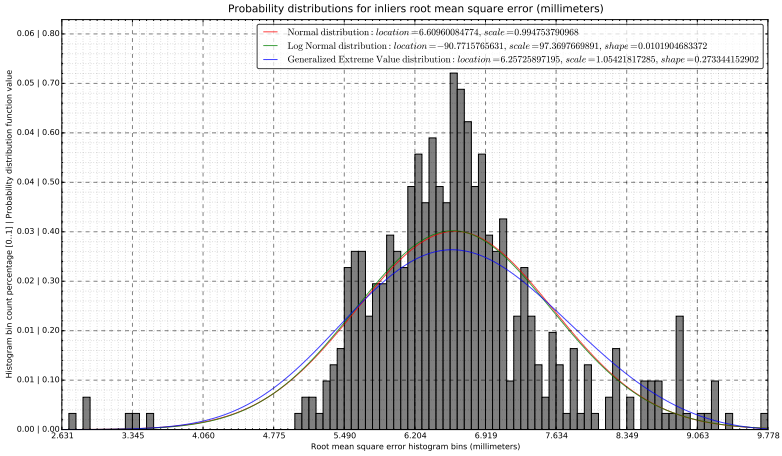


Figure A.3.1.35: Probability distributions for the Root Mean Square Error of the inliers

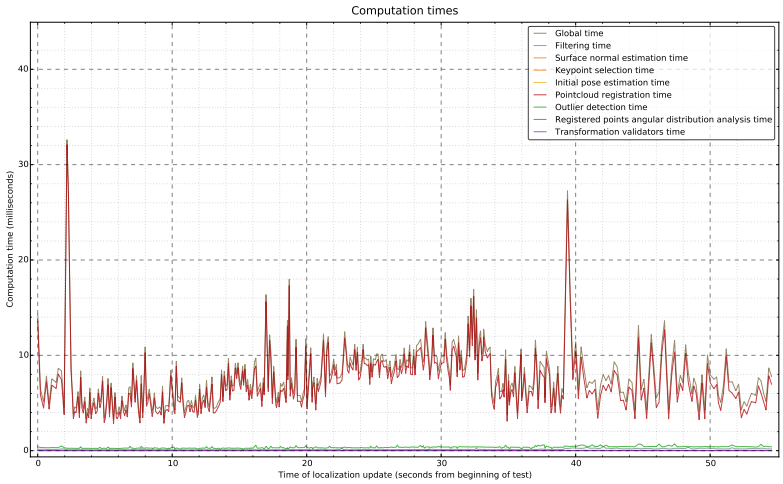


Figure A.3.1.36: Localization system computation times

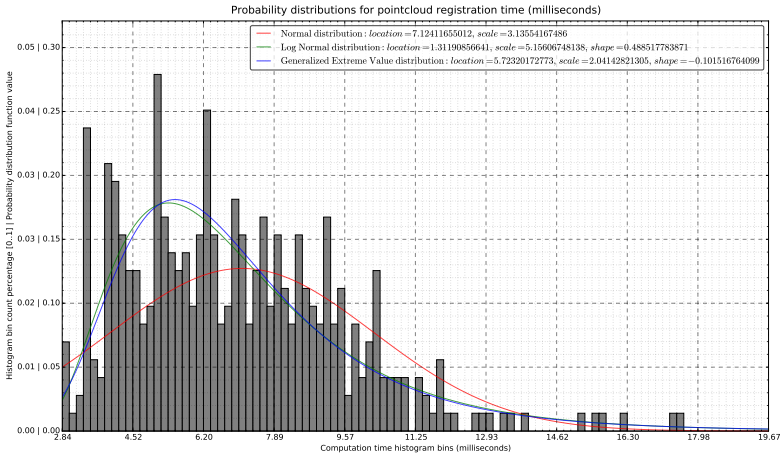


Figure A.3.1.37: Probability distributions for the point cloud registration time

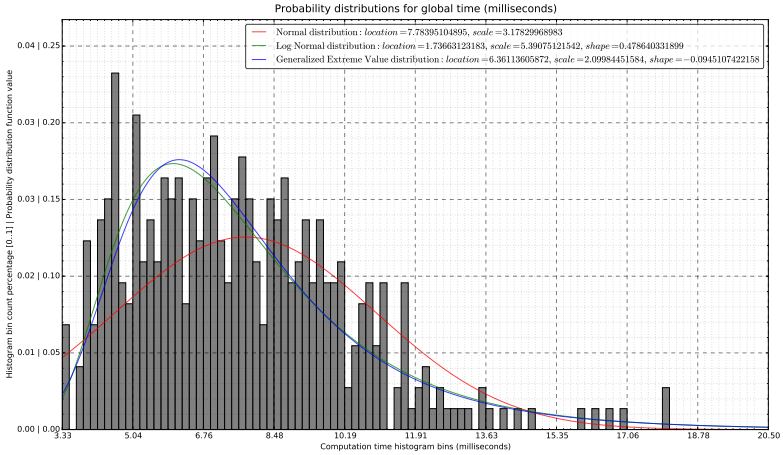


Figure A.3.1.38: Probability distributions for the localization system global computation time

A.3.2 Wall following path using the simulated Guardian robot at 30 cm/s in the static cluttered environment

The figures in this section show the detailed results of the test performed with the simulated Guardian robot with a 2 mm resolution map in the static cluttered indoor environment following the lower and right wall at 30 cm/s.

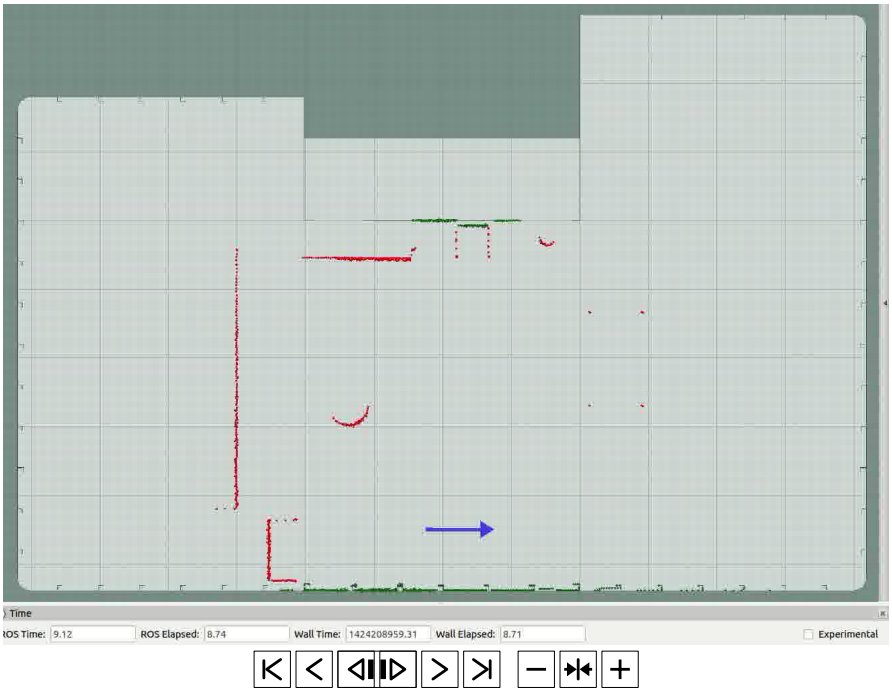


Figure A.3.2.1: Animation of laser scans using the localization system poses

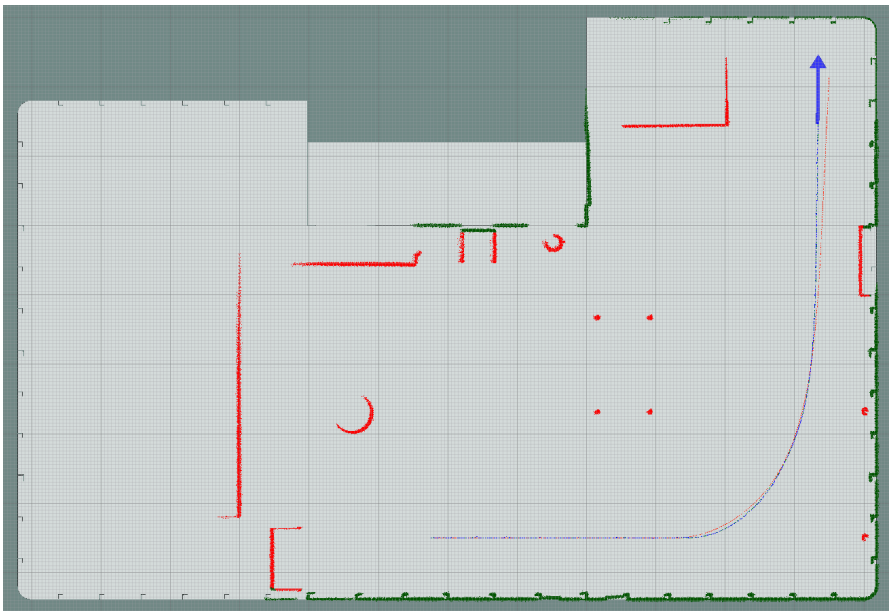


Figure A.3.2.2: Laser scans assembled on top of the map using the localization system poses

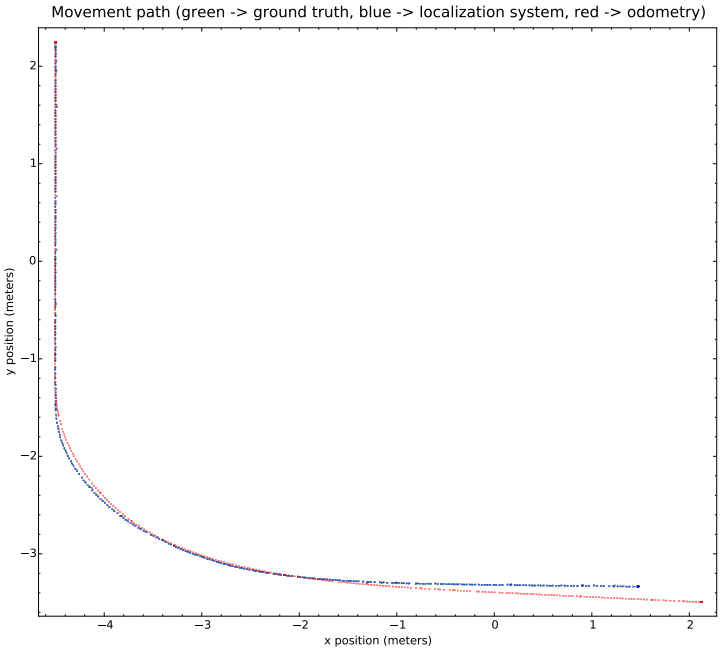


Figure A.3.2.3: Poses estimated by the ground truth, localization system and odometry

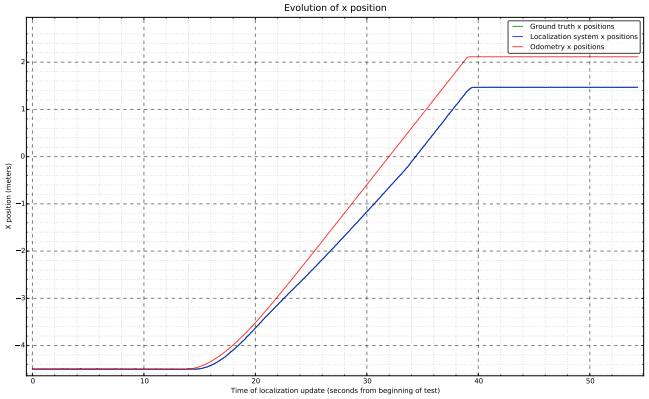


Figure A.3.2.4: Evolution of x position over time

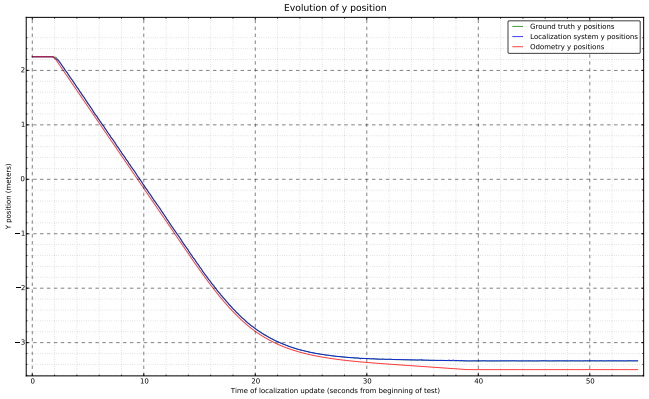


Figure A.3.2.5: Evolution of y position over time

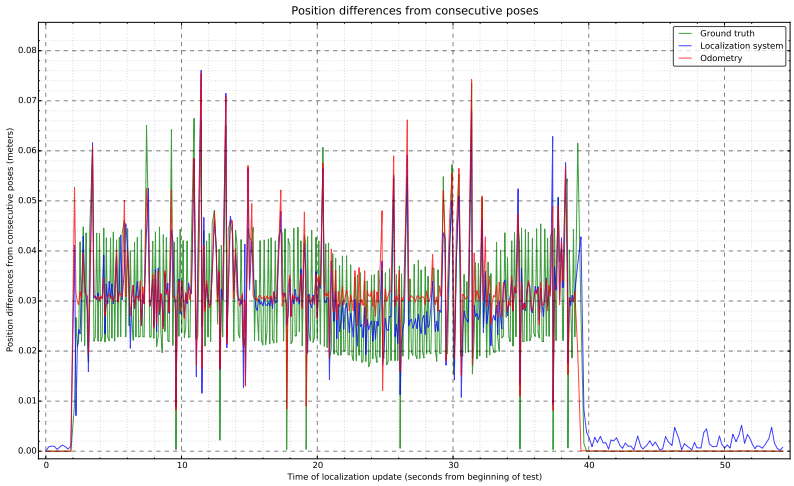


Figure A.3.2.6: Distance traveled between consecutive pose estimations

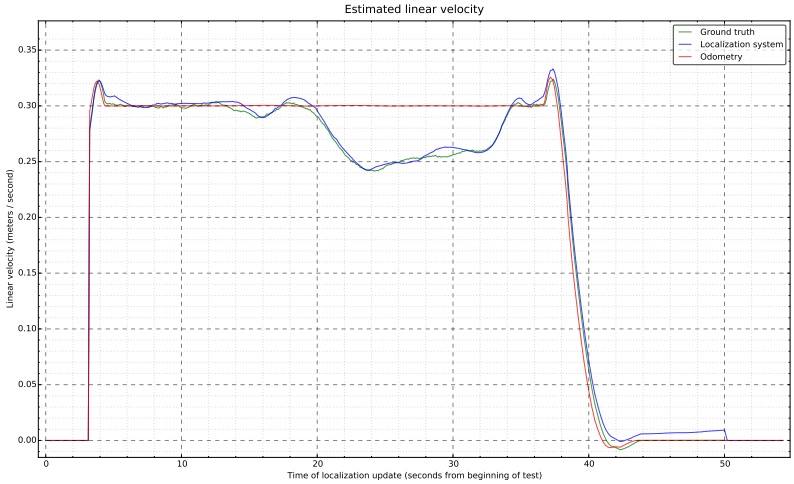


Figure A.3.2.7: Estimated linear velocity

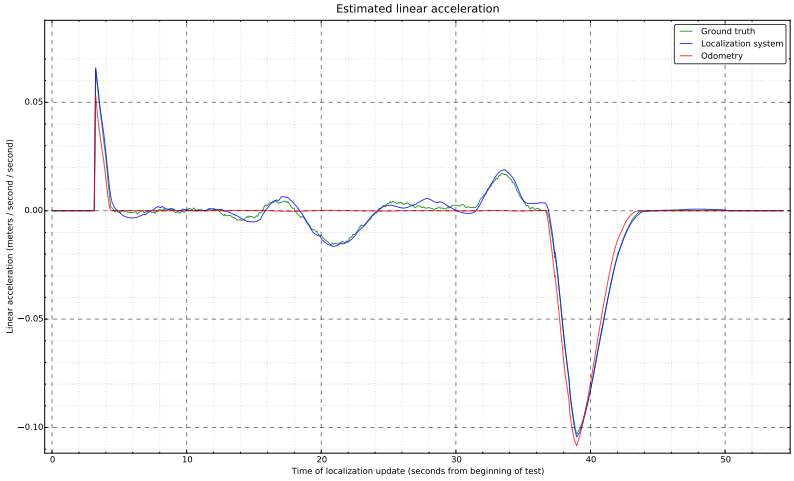


Figure A.3.2.8: Estimated linear acceleration

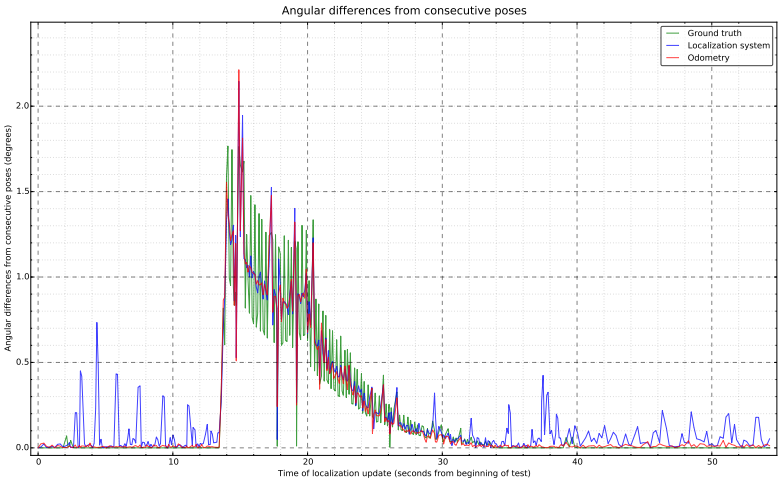


Figure A.3.2.9: Angular differences between consecutive pose estimations

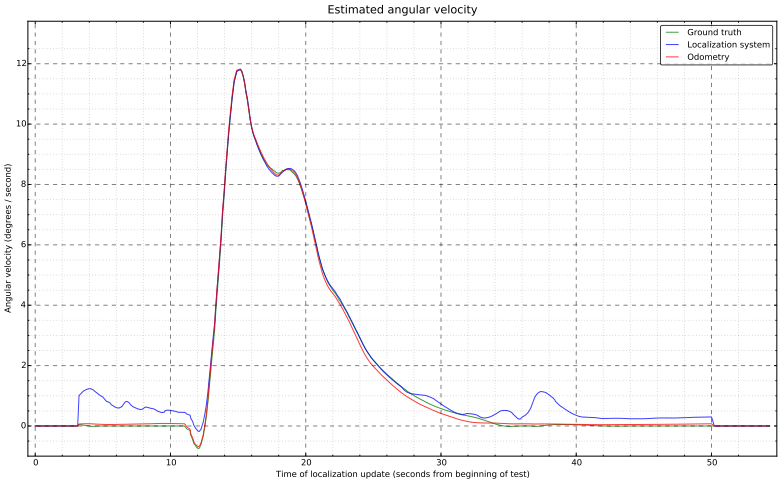


Figure A.3.2.10: Estimated angular velocity

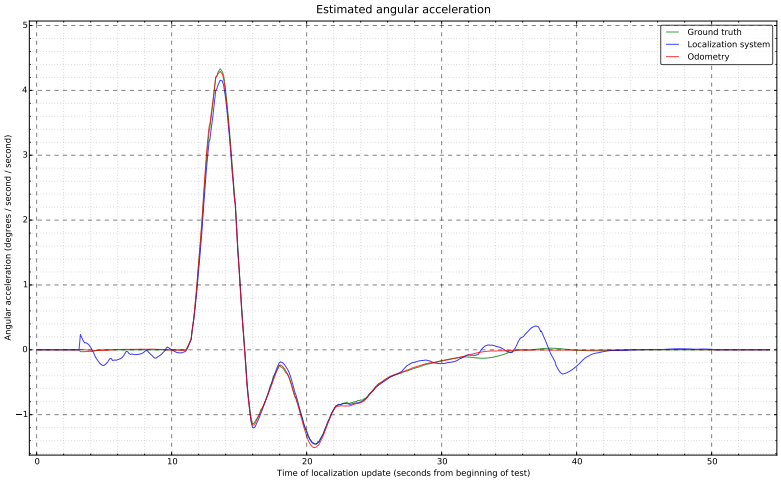


Figure A.3.2.11: Estimated angular acceleration

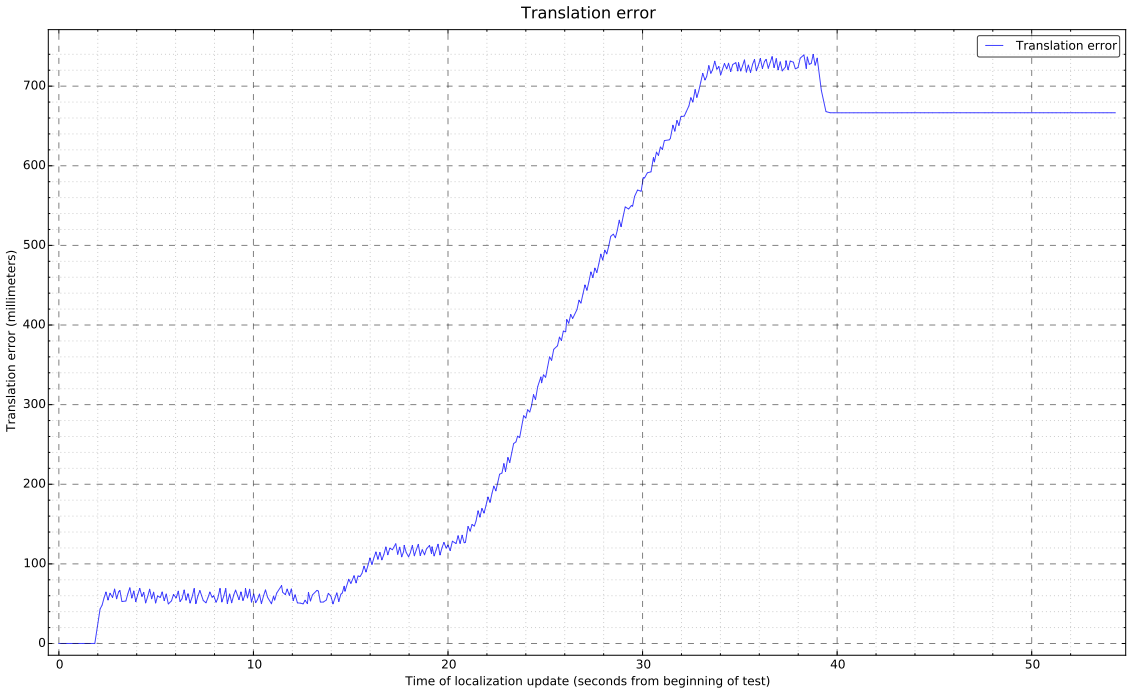


Figure A.3.2.12: Odometry translation errors

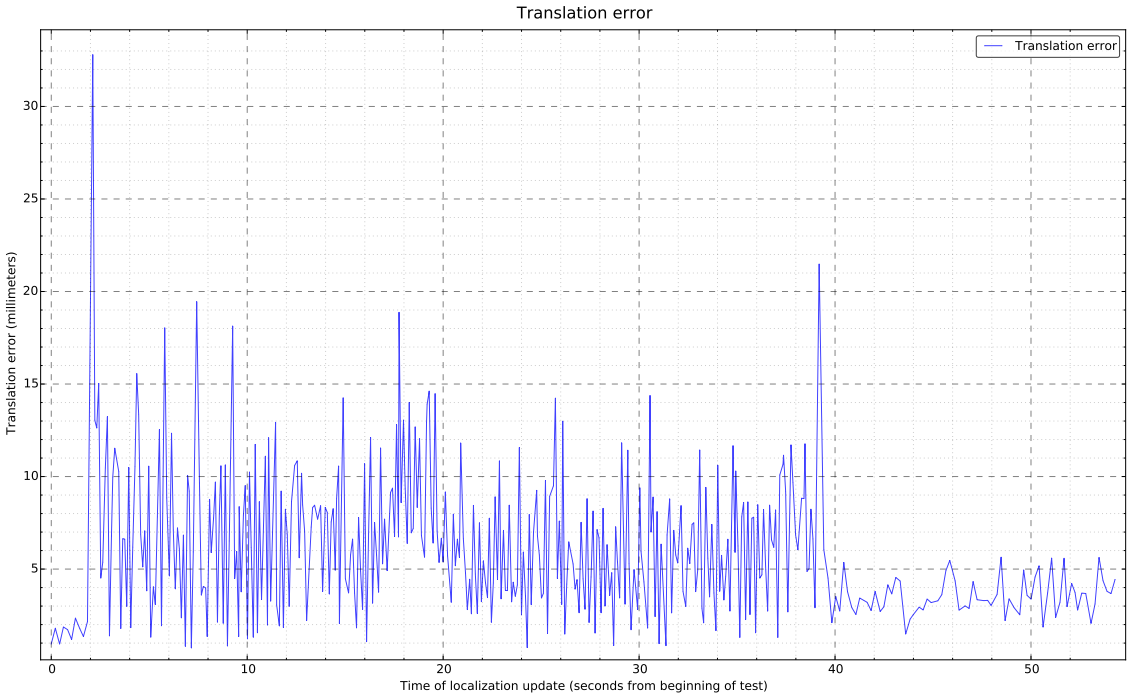


Figure A.3.2.13: Localization system translation errors

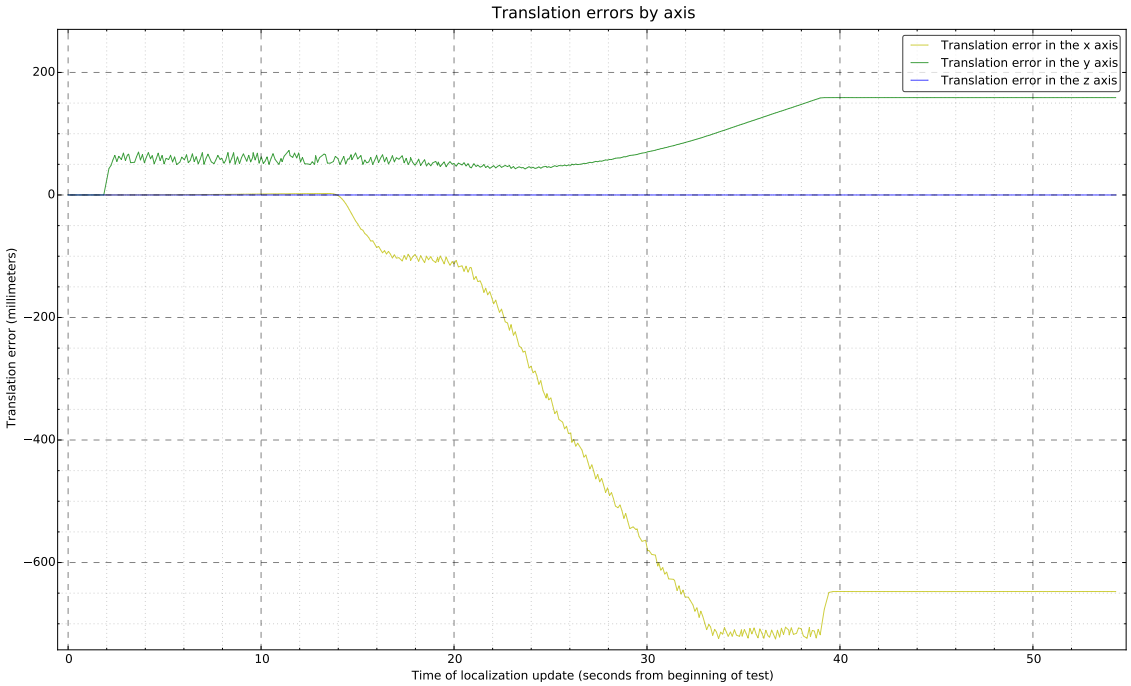


Figure A.3.2.14: Odometry translation errors components

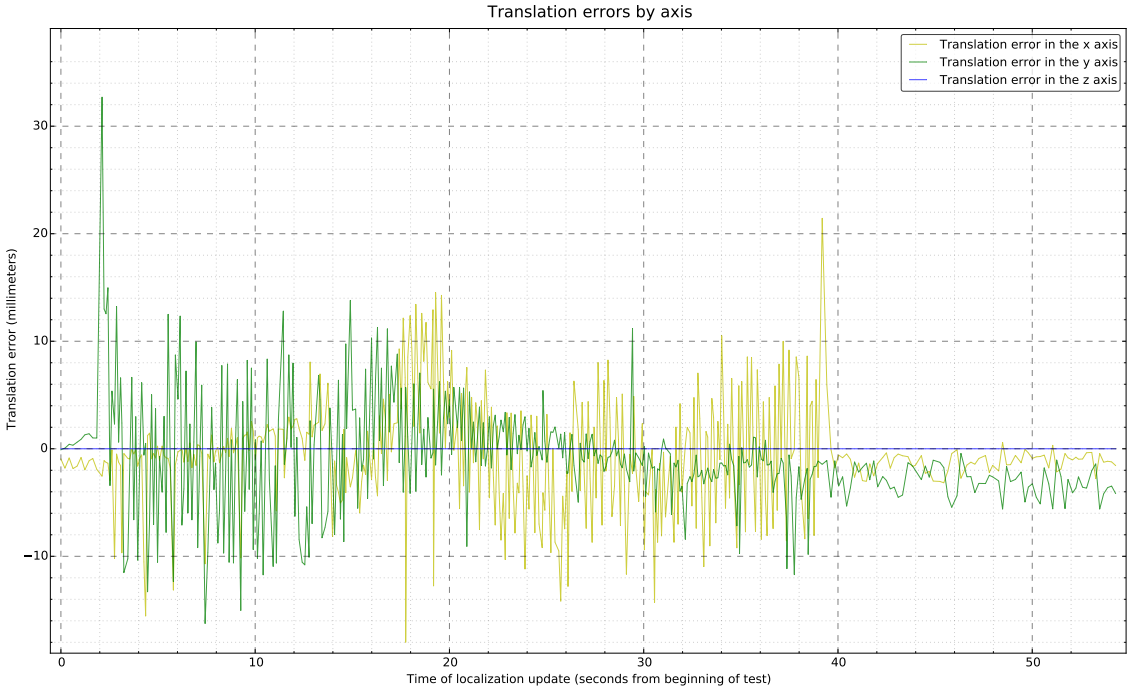


Figure A.3.2.15: Localization system translation errors components

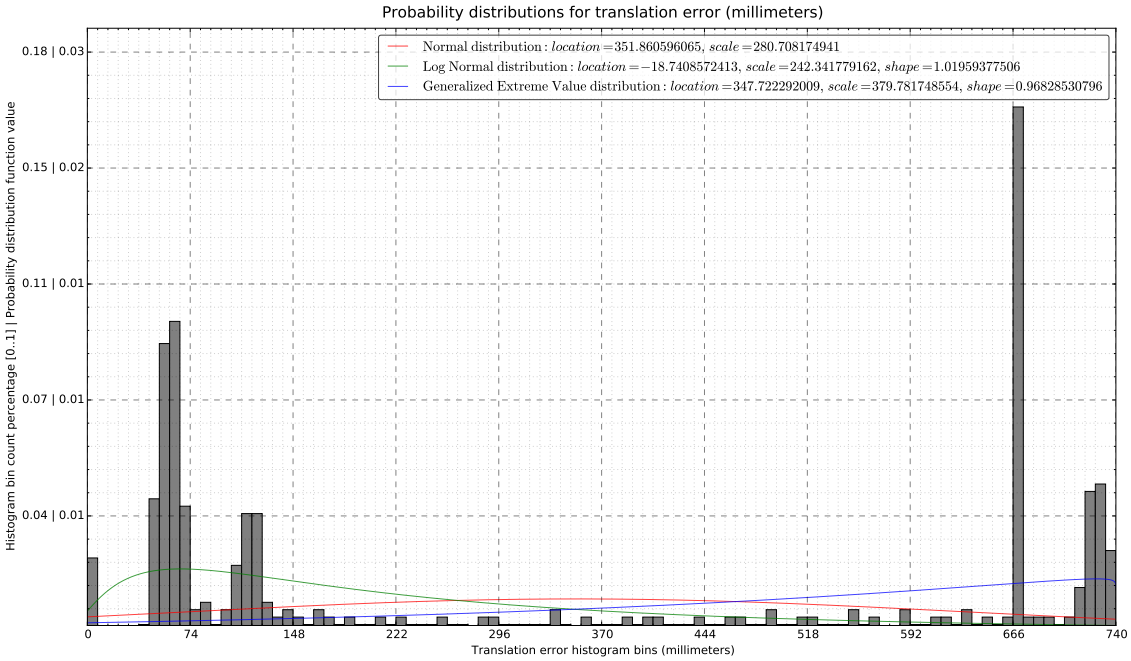


Figure A.3.2.16: Probability distributions for the odometry translation errors

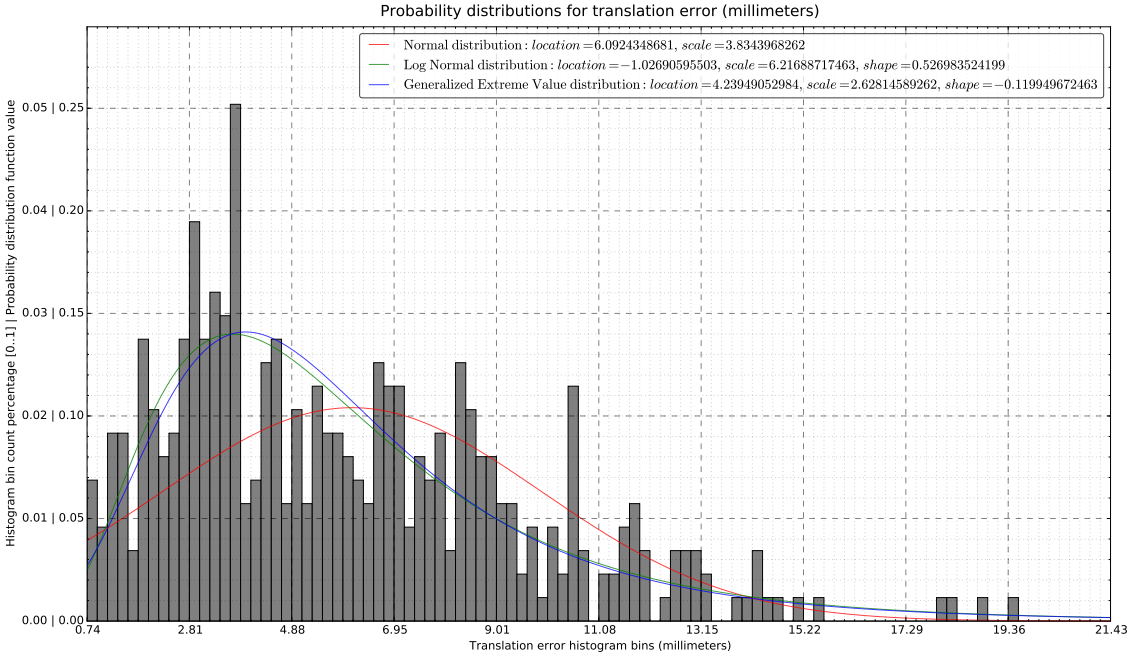


Figure A.3.2.17: Probability distributions for the localization system translation errors

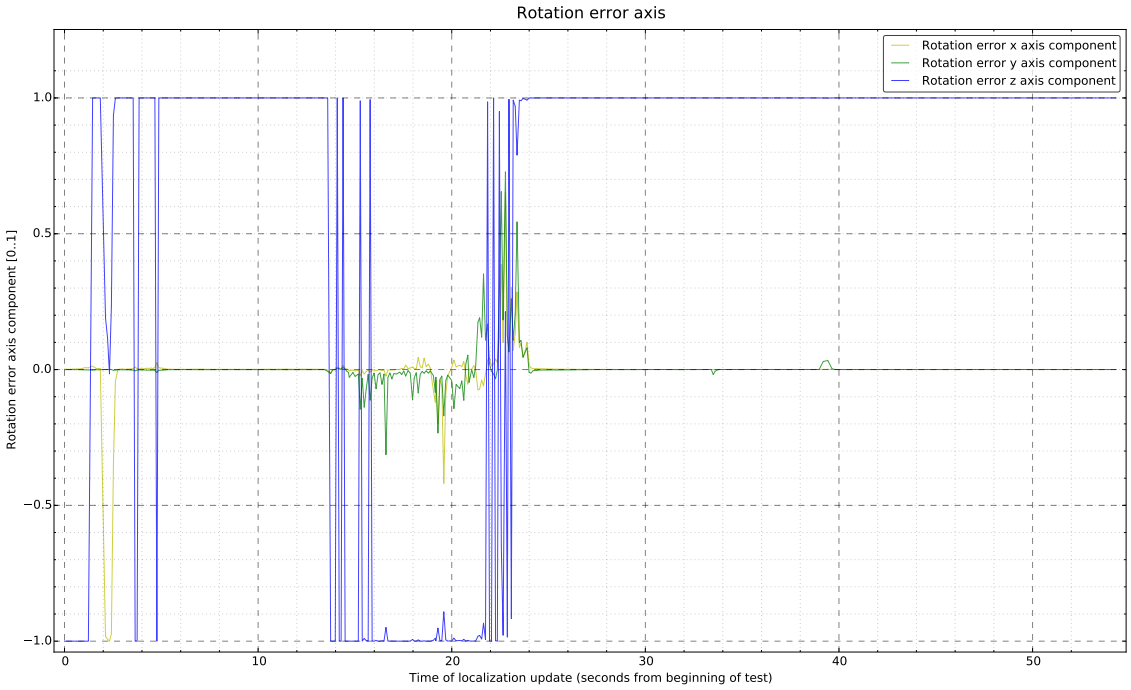


Figure A.3.2.18: Odometry rotation errors axis

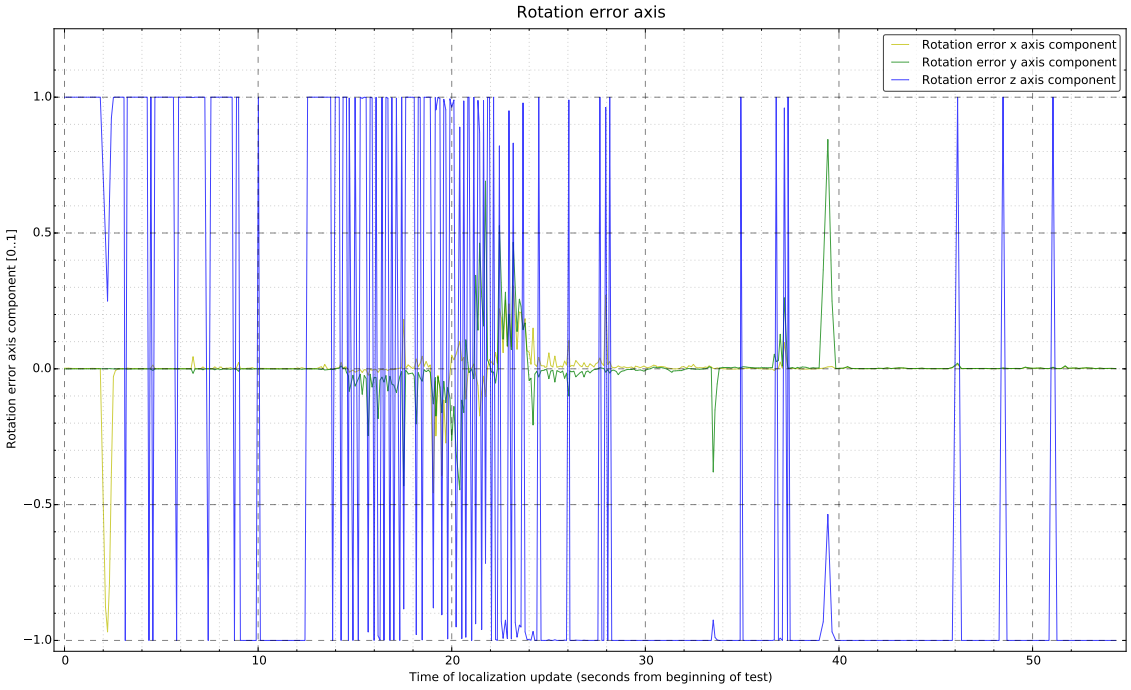


Figure A.3.2.19: Localization system rotation errors axis

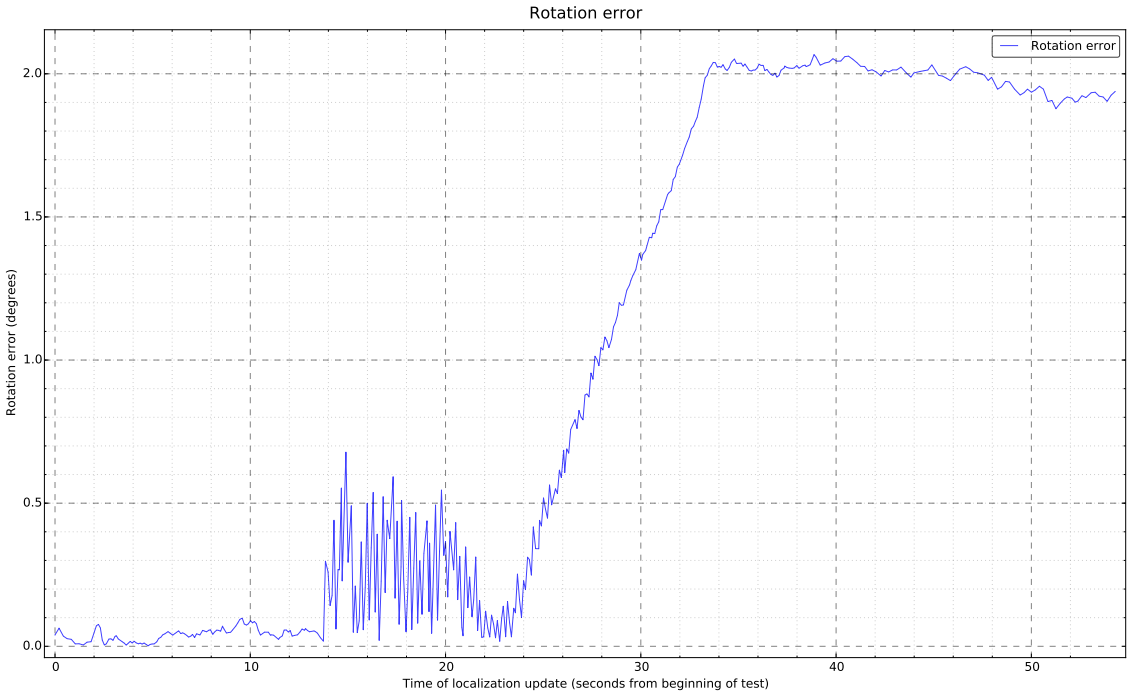


Figure A.3.2.20: Odometry rotation errors

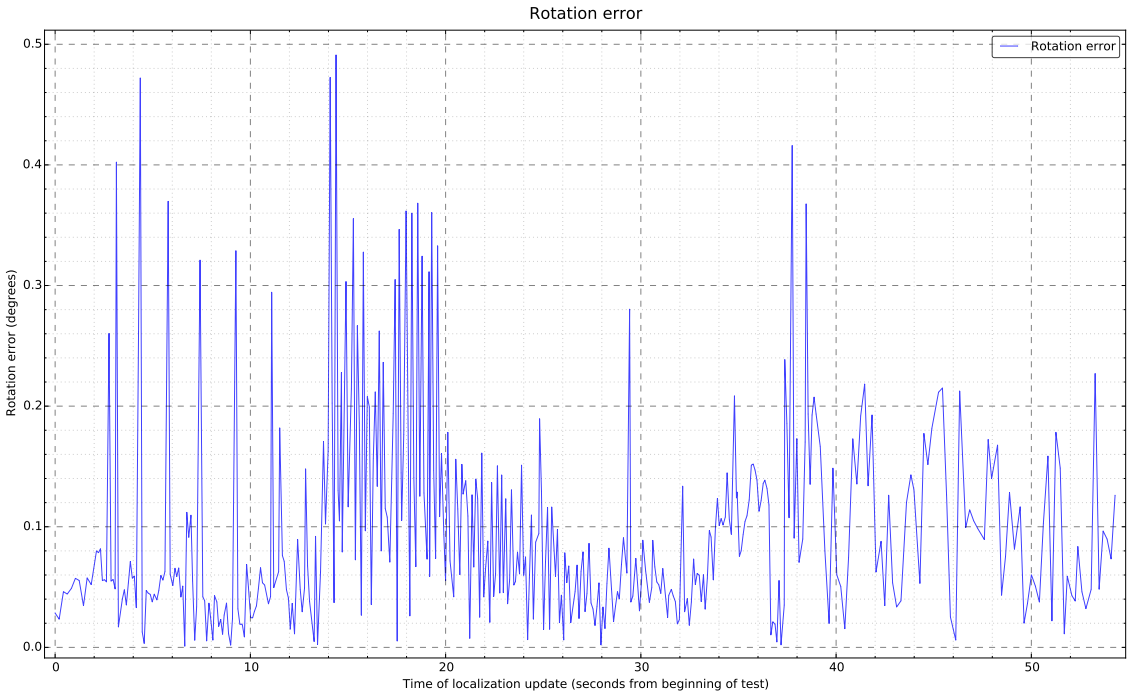


Figure A.3.2.21: Localization system rotation errors

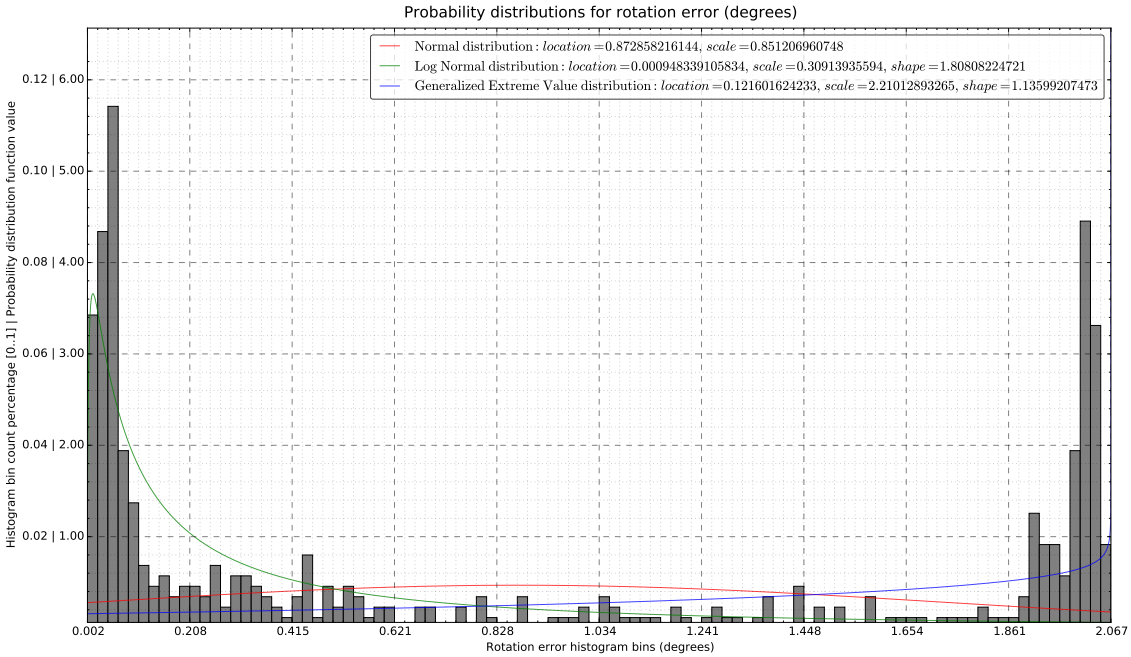


Figure A.3.2.22: Probability distributions for the odometry rotation errors

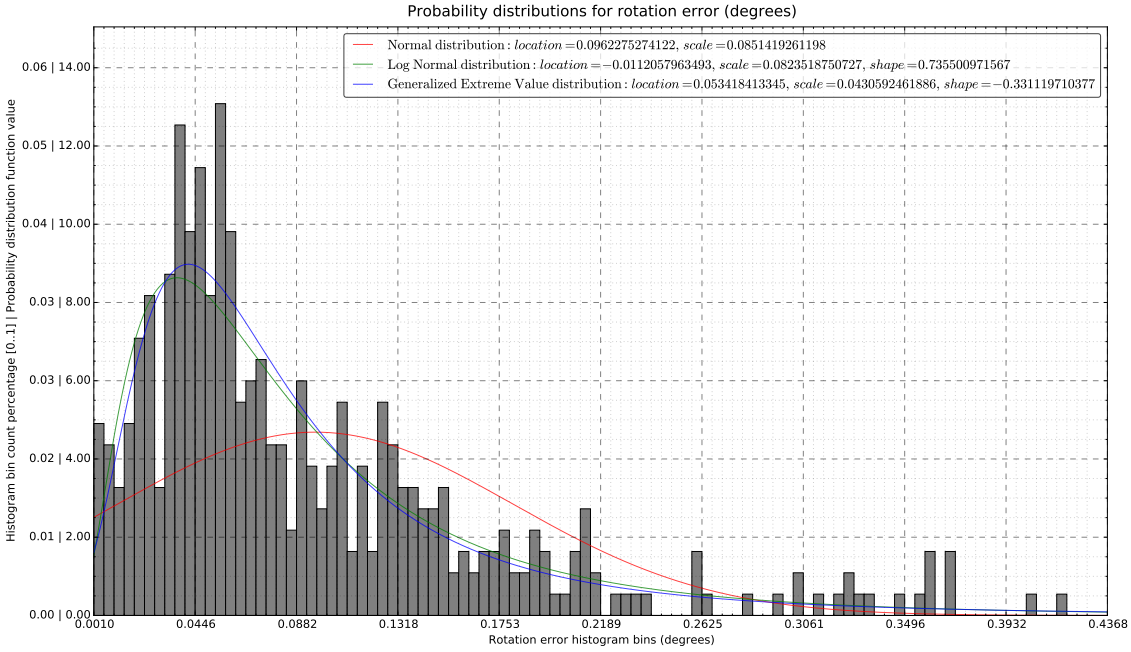


Figure A.3.2.23: Probability distributions for the localization system rotation errors

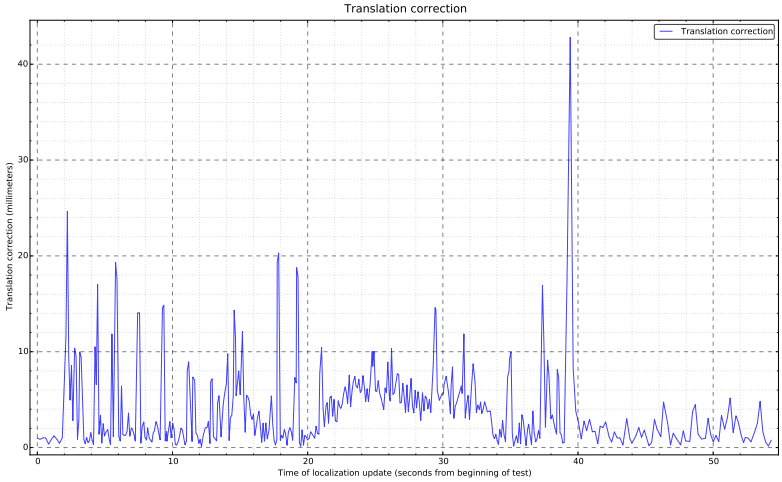


Figure A.3.2.24: Translation corrections performed by the localization system

Figure A.3.2.25: Translation corrections components performed by the localization system

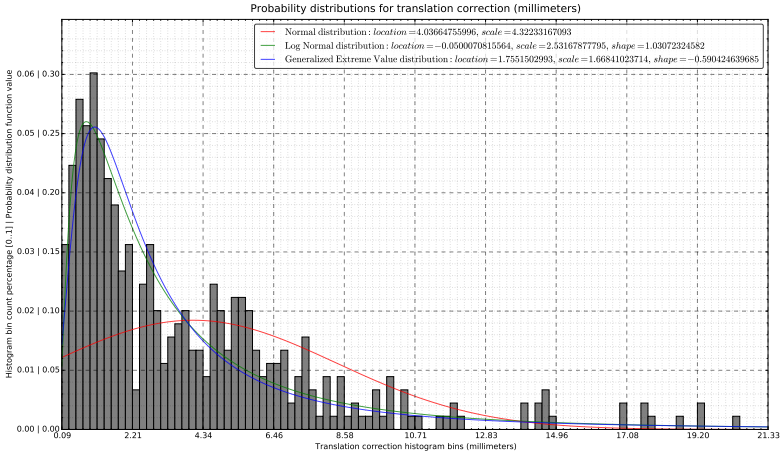


Figure A.3.2.26: Probability distributions for the translation corrections performed by the localization system

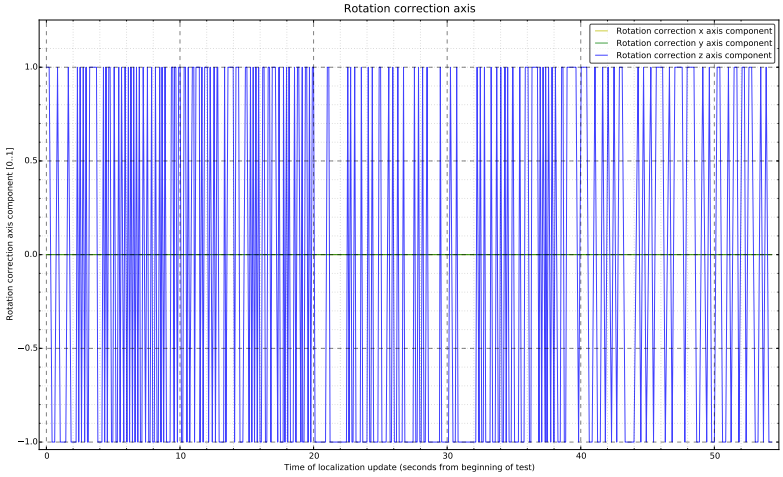


Figure A.3.2.27: Rotation corrections (axis) performed by the localization system

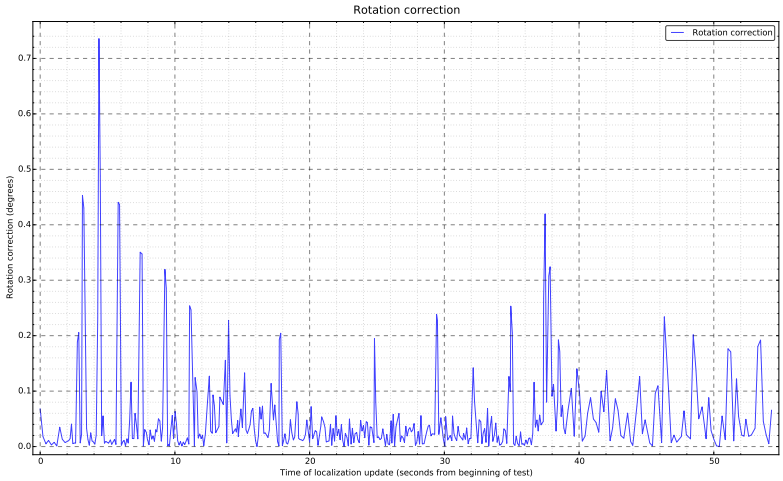


Figure A.3.2.28: Rotation corrections performed by the localization system

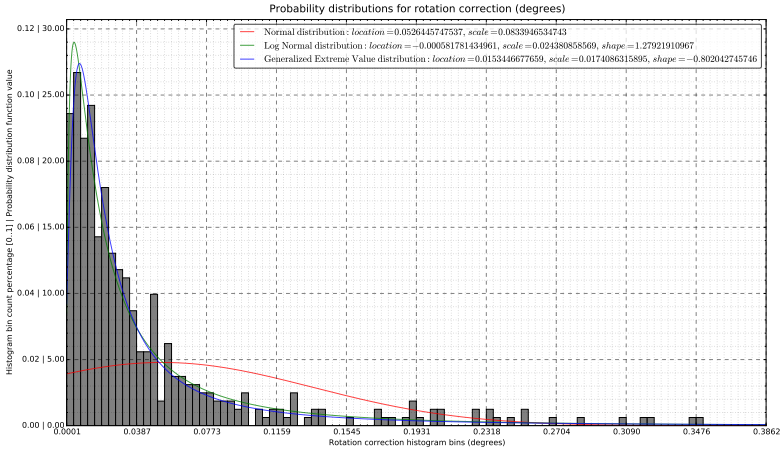


Figure A.3.2.29: Probability distributions for the rotation corrections performed by the localization system

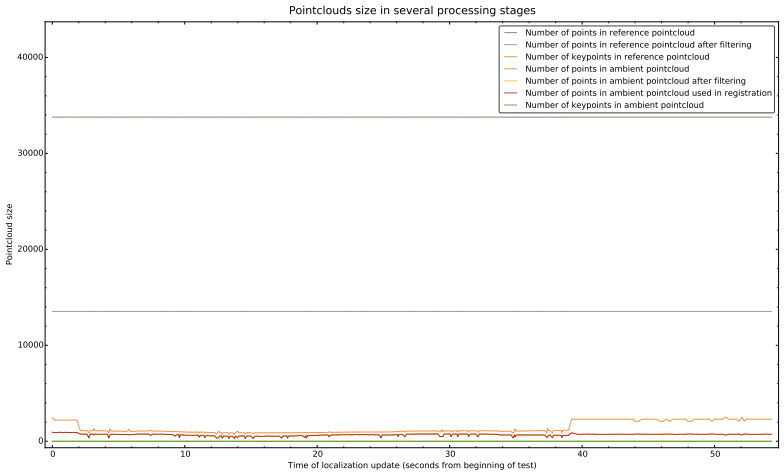


Figure A.3.2.30: Point clouds size in several of the localization system processing stages

Figure A.3.2.31: Number of registered points

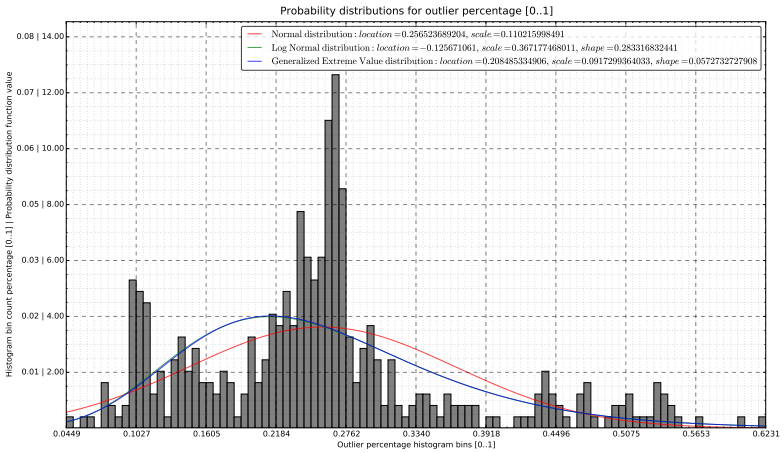


Figure A.3.2.32: Probability distributions for the ambient point cloud outlier percentage

Figure A.3.2.33: Angular distribution percentage of registered points

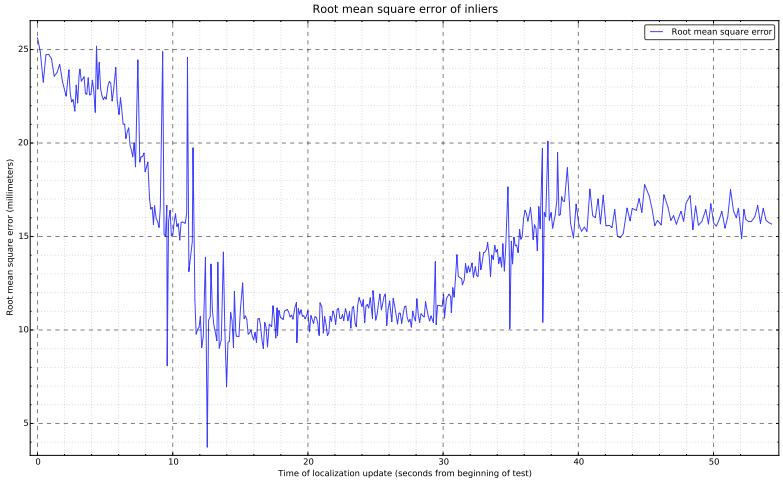


Figure A.3.2.34: Root Mean Square Error of the inliers

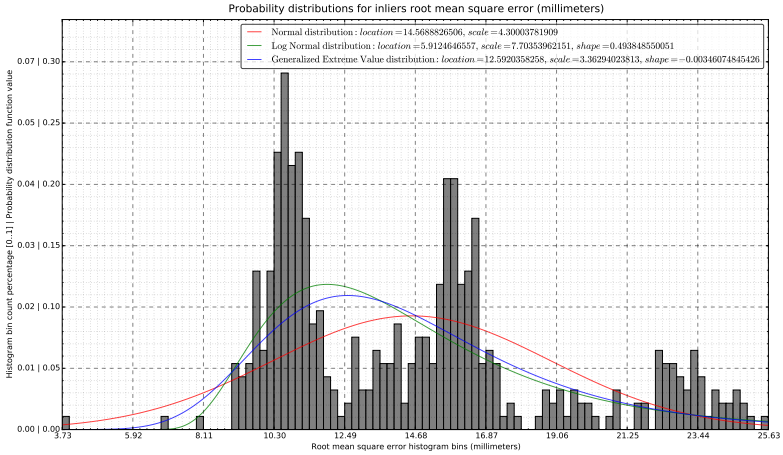


Figure A.3.2.35: Probability distributions for the Root Mean Square Error of the inliers

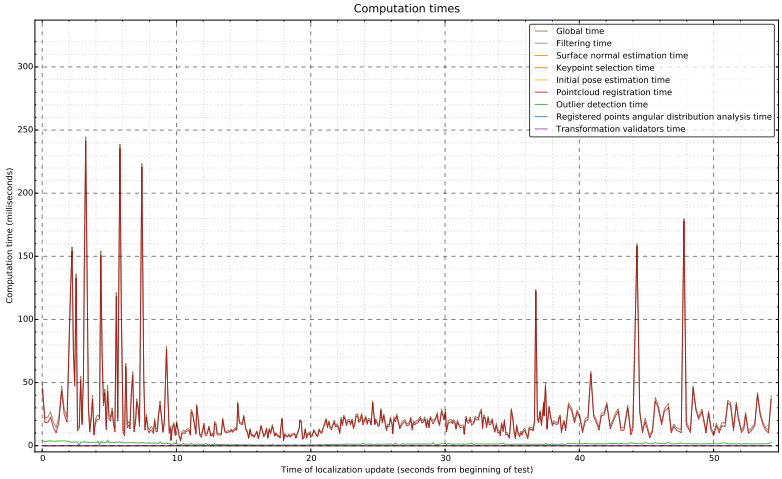


Figure A.3.2.36: Localization system computation times

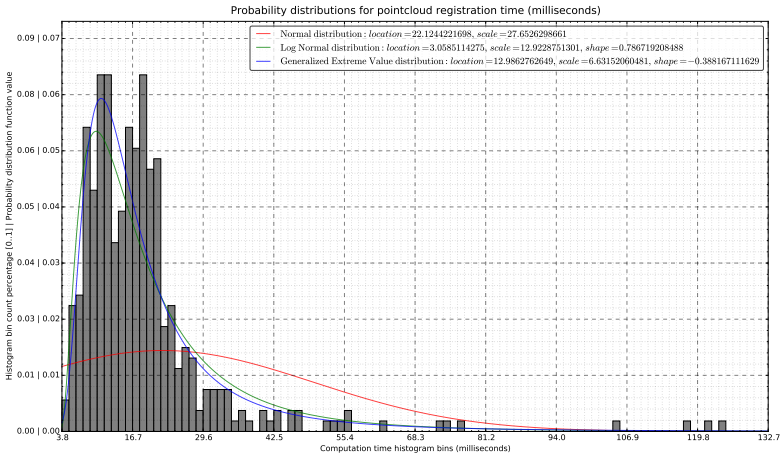


Figure A.3.2.37: Probability distributions for the point cloud registration time

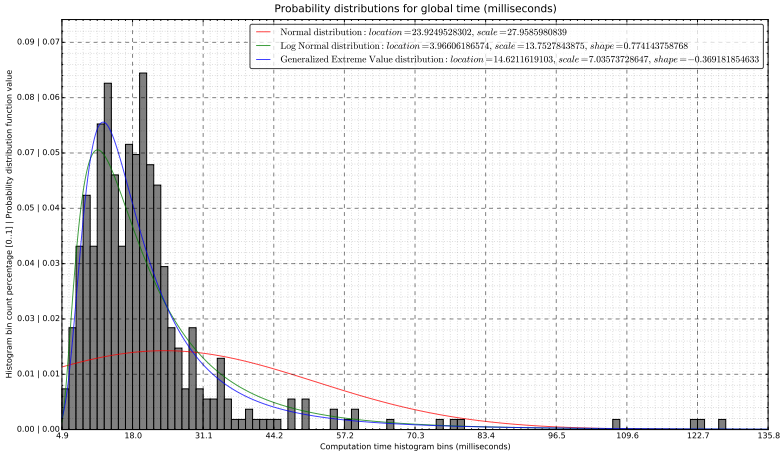


Figure A.3.2.38: Probability distributions for the localization system global computation time

A.3.3 Wall following path using the simulated Guardian robot at 30 cm/s in the dynamic cluttered environment

The figures in this section show the detailed results of the test performed with the simulated Guardian robot with a 2 mm resolution map in the dynamic cluttered indoor environment following the lower and right wall at 30 cm/s.

Figure A.3.3.1: Animation of laser scans using the localization system poses

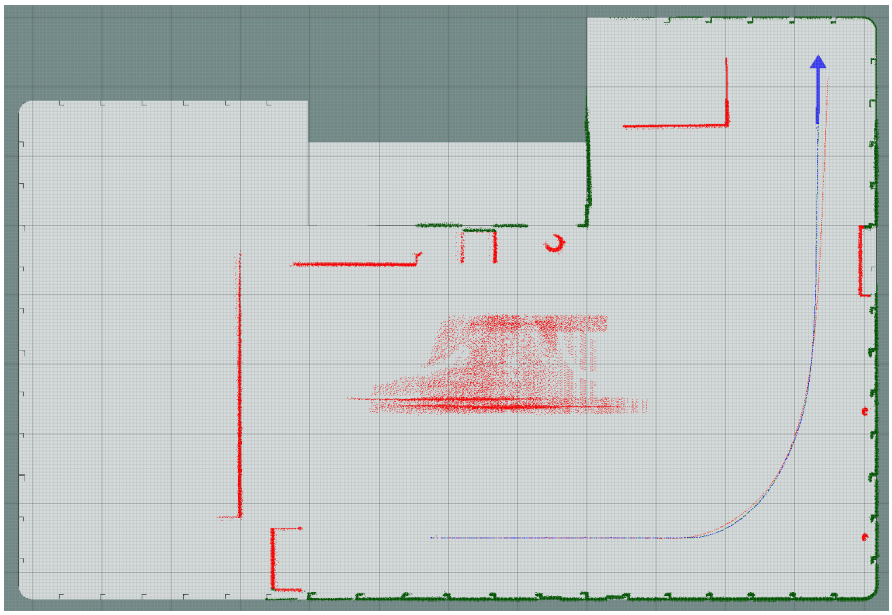


Figure A.3.3.2: Laser scans assembled on top of the map using the localization system poses

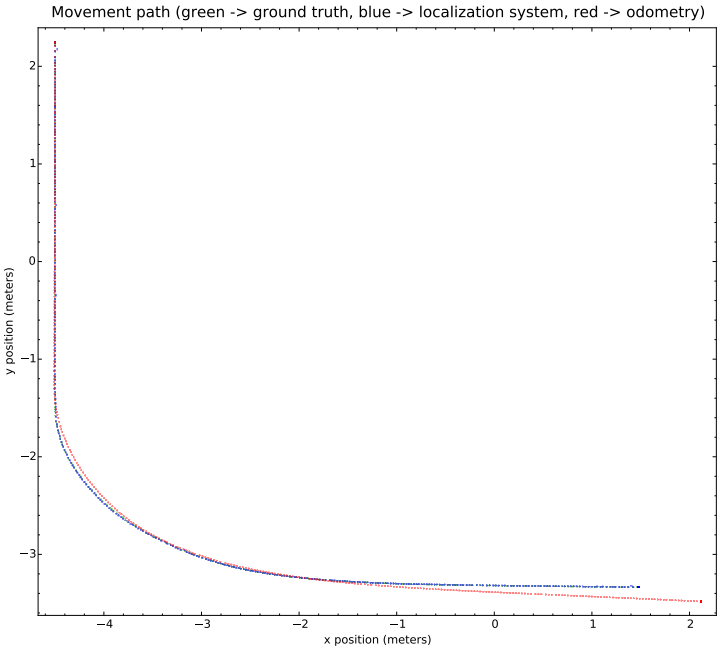


Figure A.3.3.3: Poses estimated by the ground truth, localization system and odometry

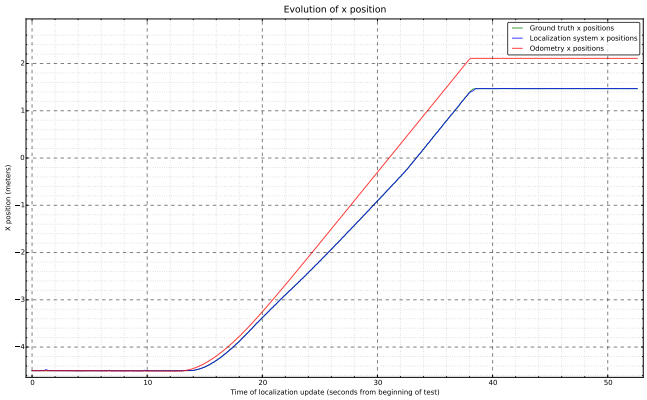


Figure A.3.3.4: Evolution of x position over time

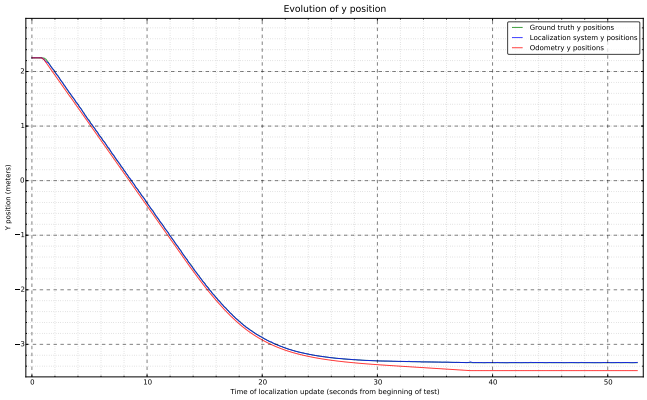


Figure A.3.3.5: Evolution of y position over time

Figure A.3.3.6: Distance traveled between consecutive pose estimations

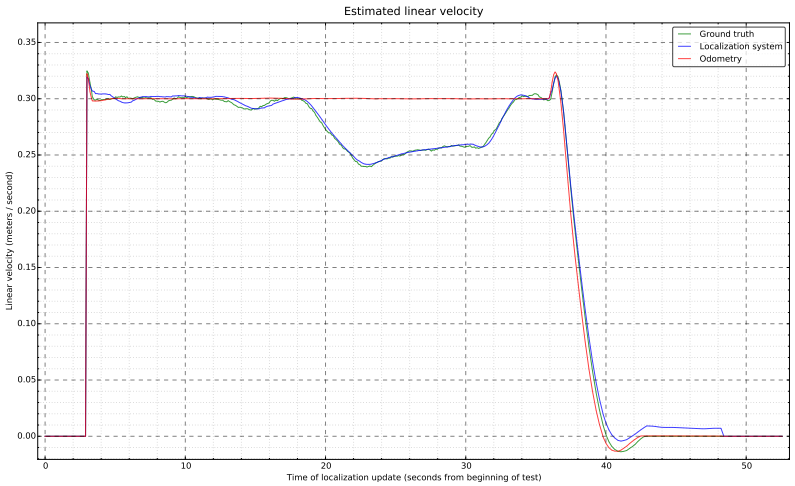


Figure A.3.3.7: Estimated linear velocity

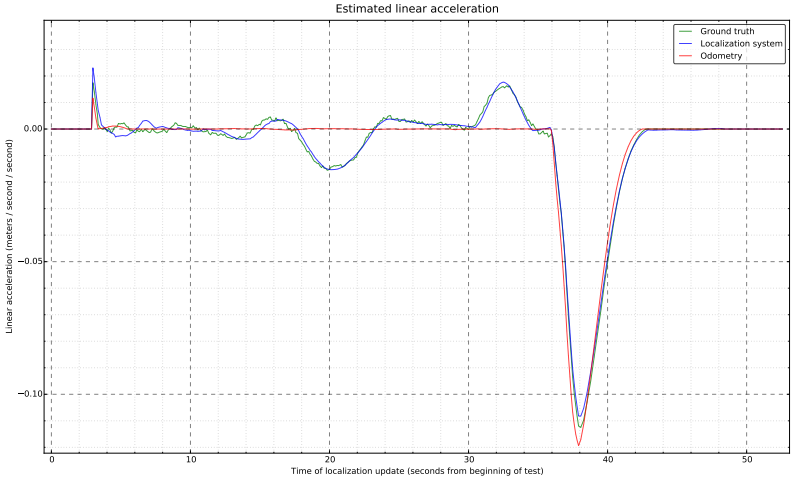


Figure A.3.3.8: Estimated linear acceleration

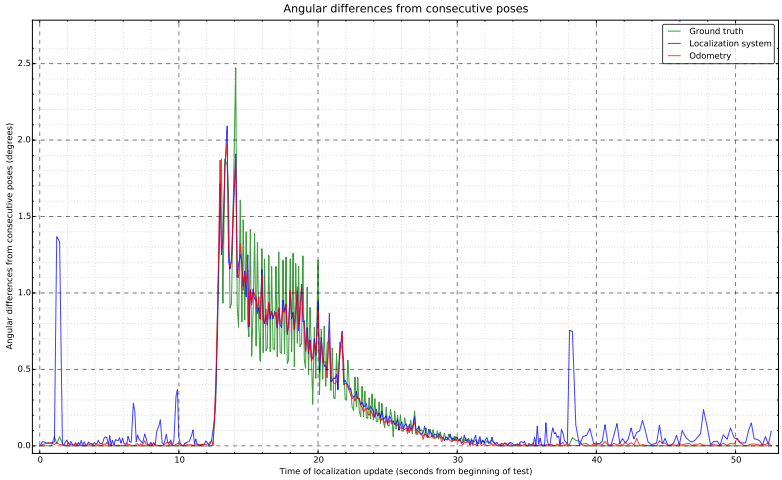


Figure A.3.3.9: Angular differences between consecutive pose estimations

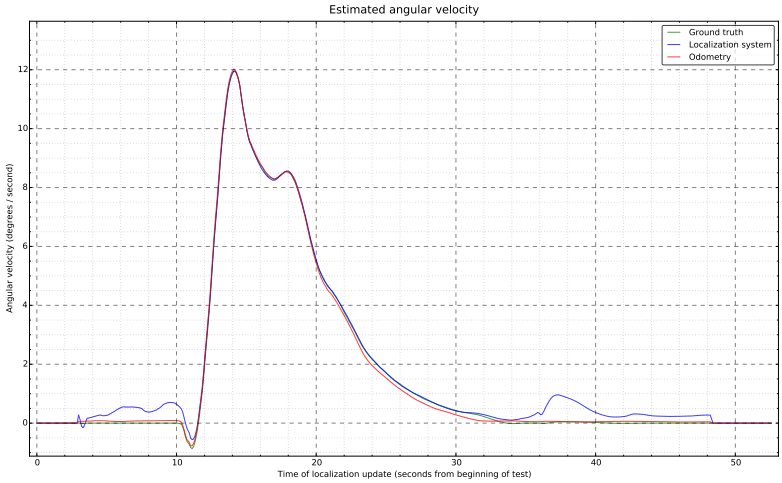


Figure A.3.3.10: Estimated angular velocity

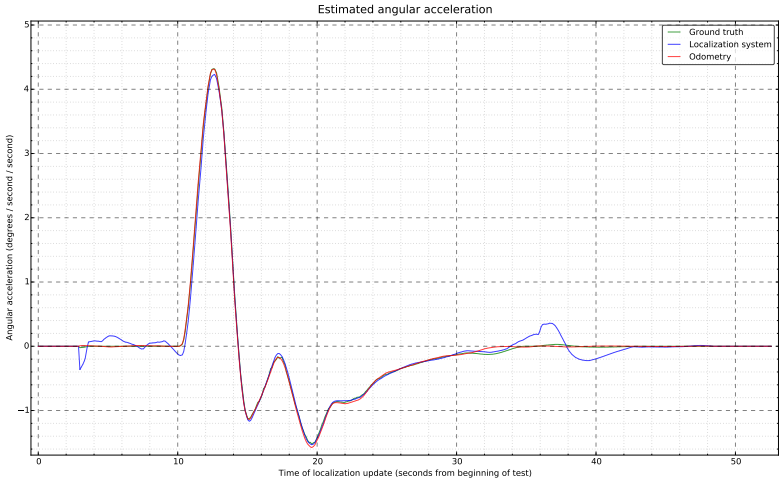


Figure A.3.3.11: Estimated angular acceleration

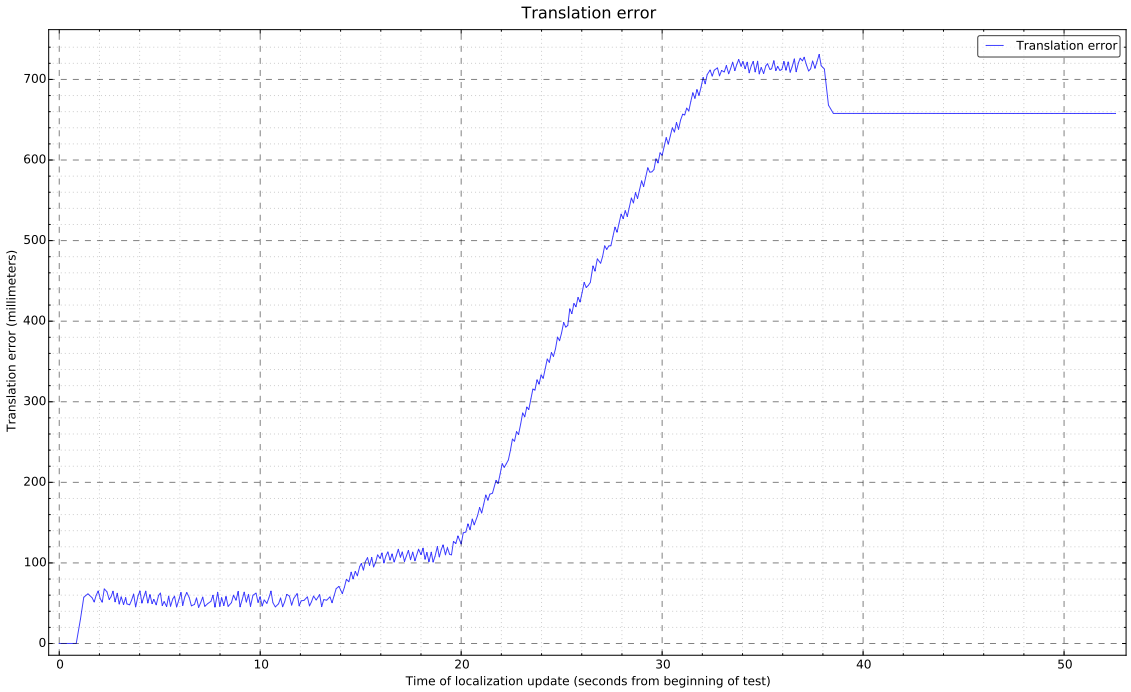


Figure A.3.3.12: Odometry translation errors

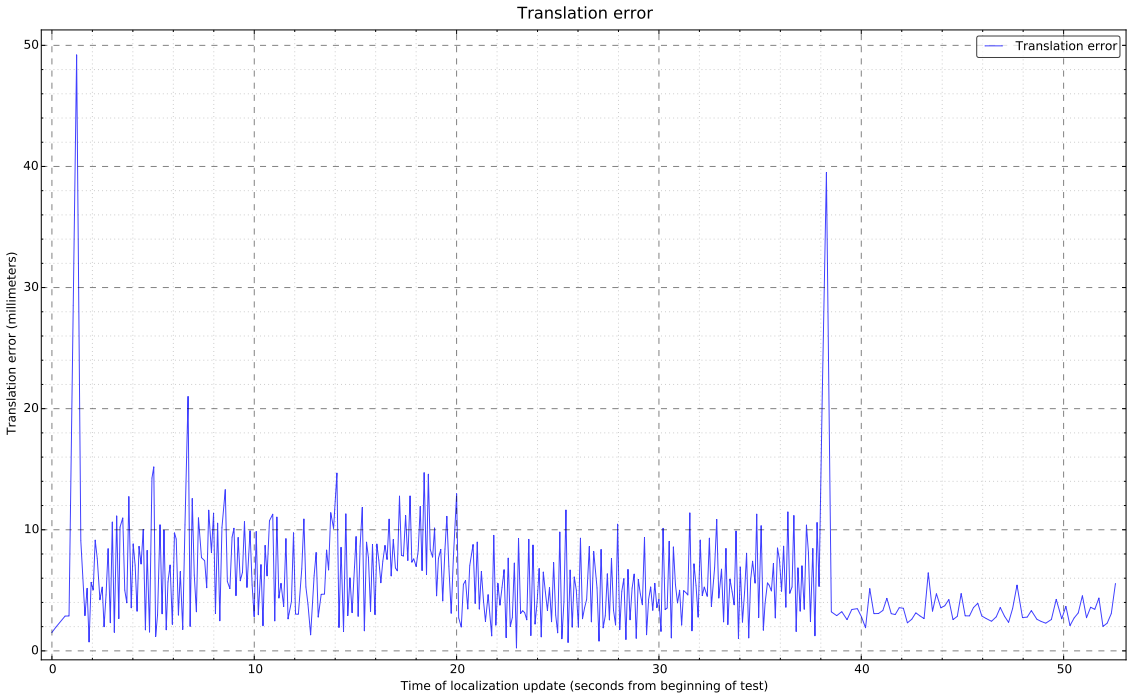


Figure A.3.3.13: Localization system translation errors

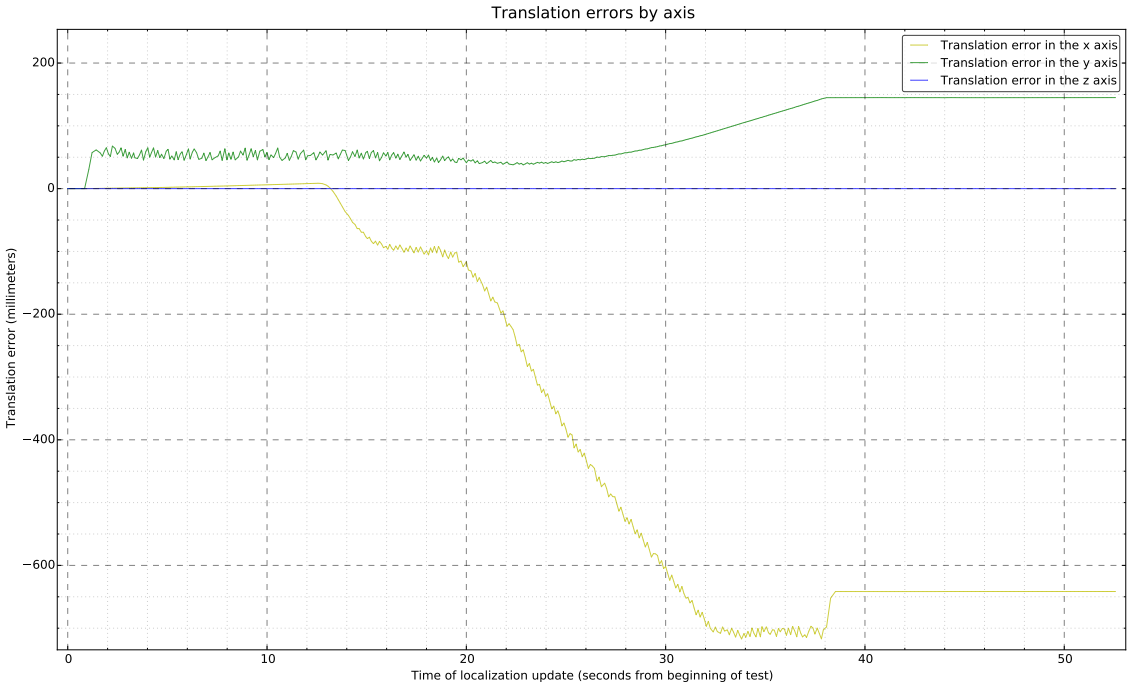


Figure A.3.3.14: Odometry translation errors components

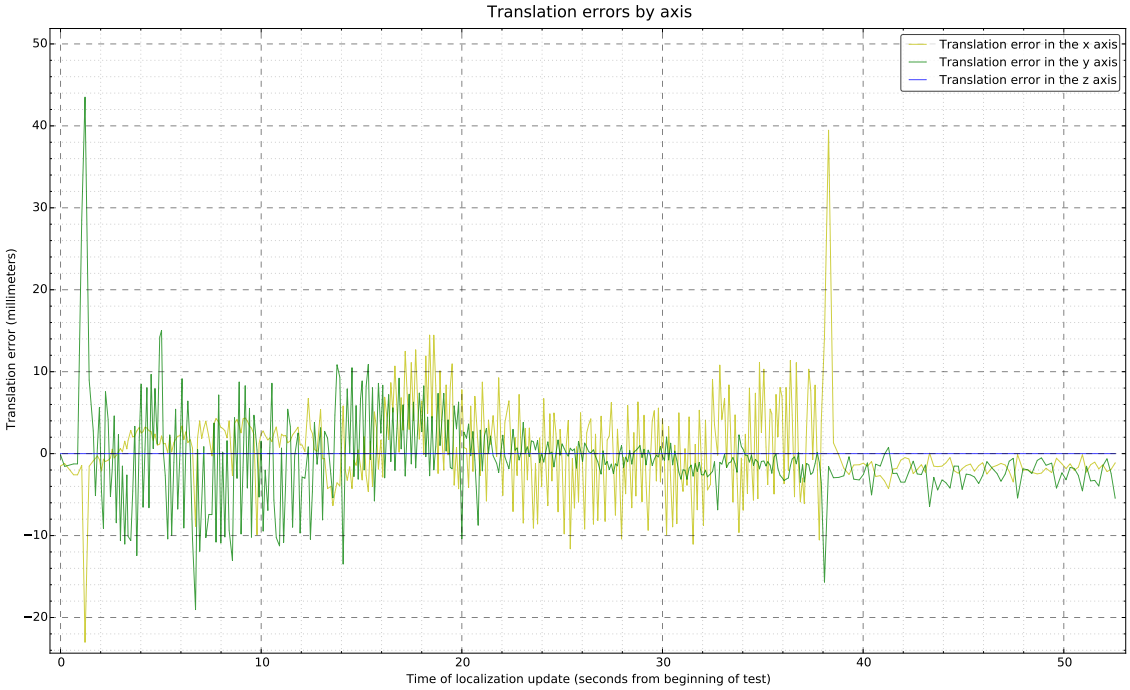


Figure A.3.3.15: Localization system translation errors components

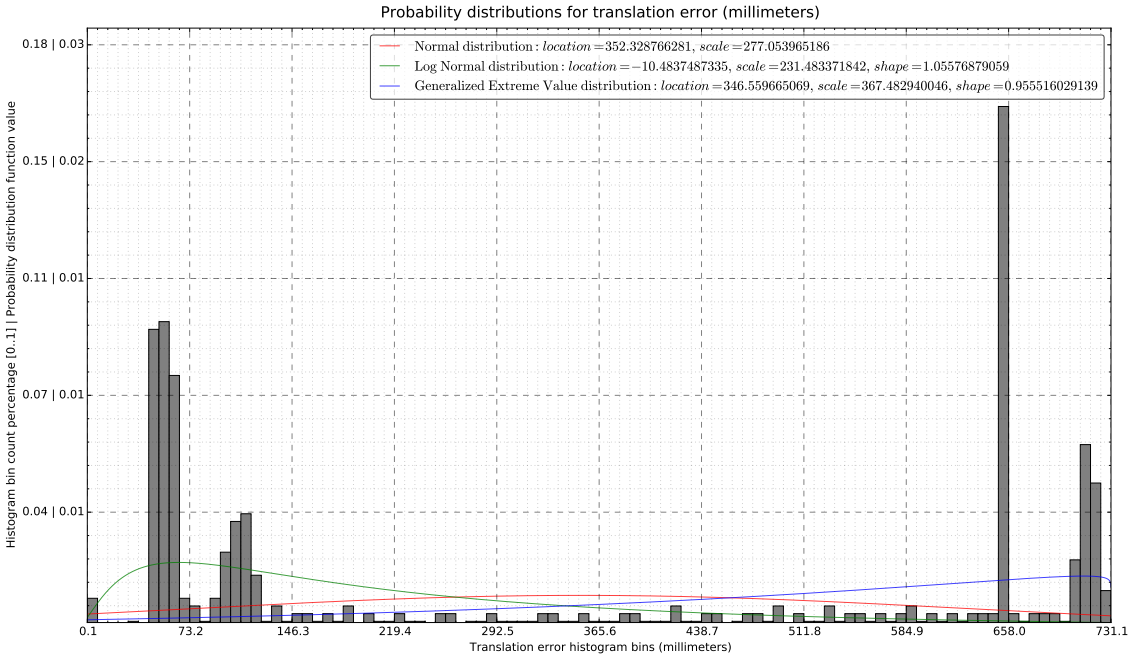


Figure A.3.3.16: Probability distributions for the odometry translation errors

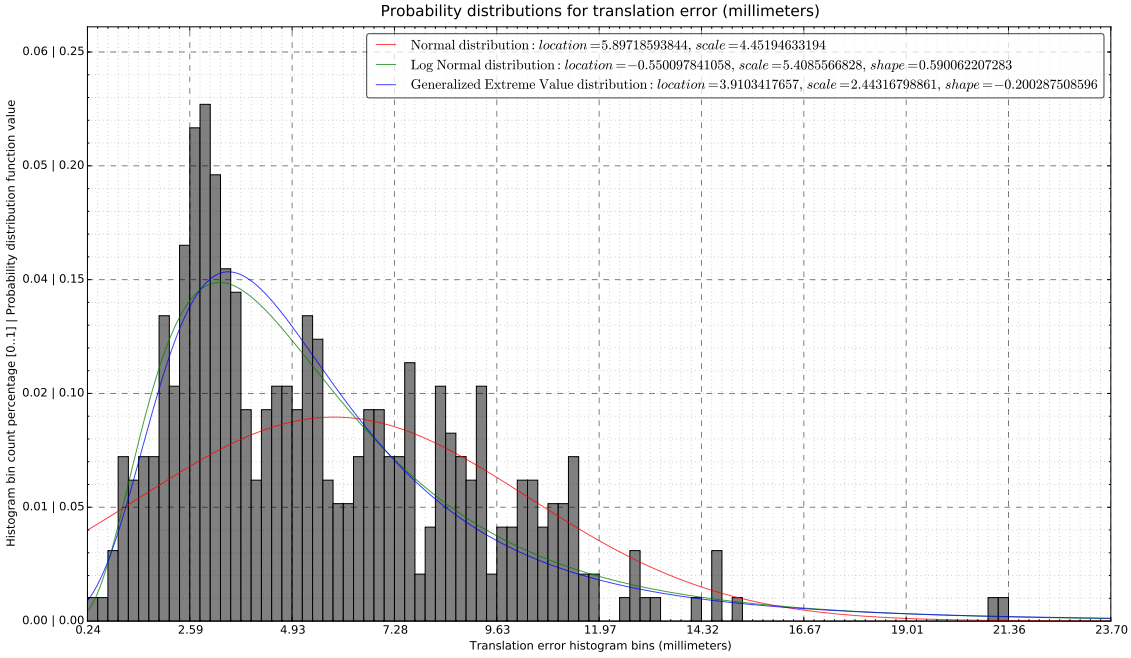


Figure A.3.3.17: Probability distributions for the localization system translation errors

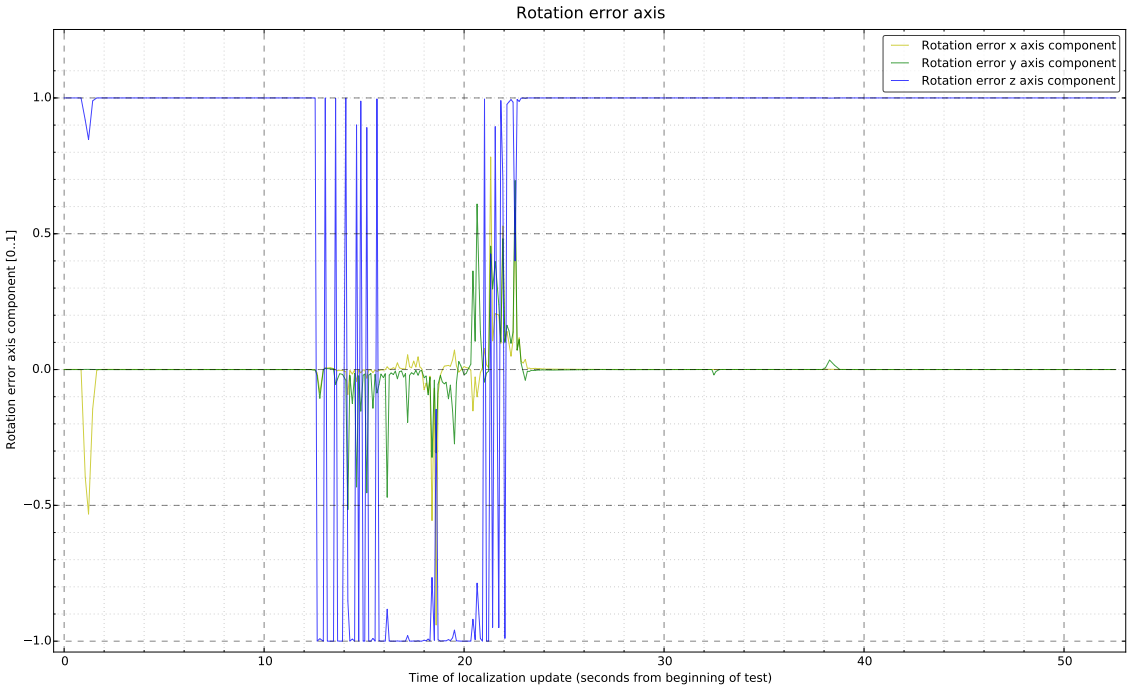


Figure A.3.3.18: Odometry rotation errors axis

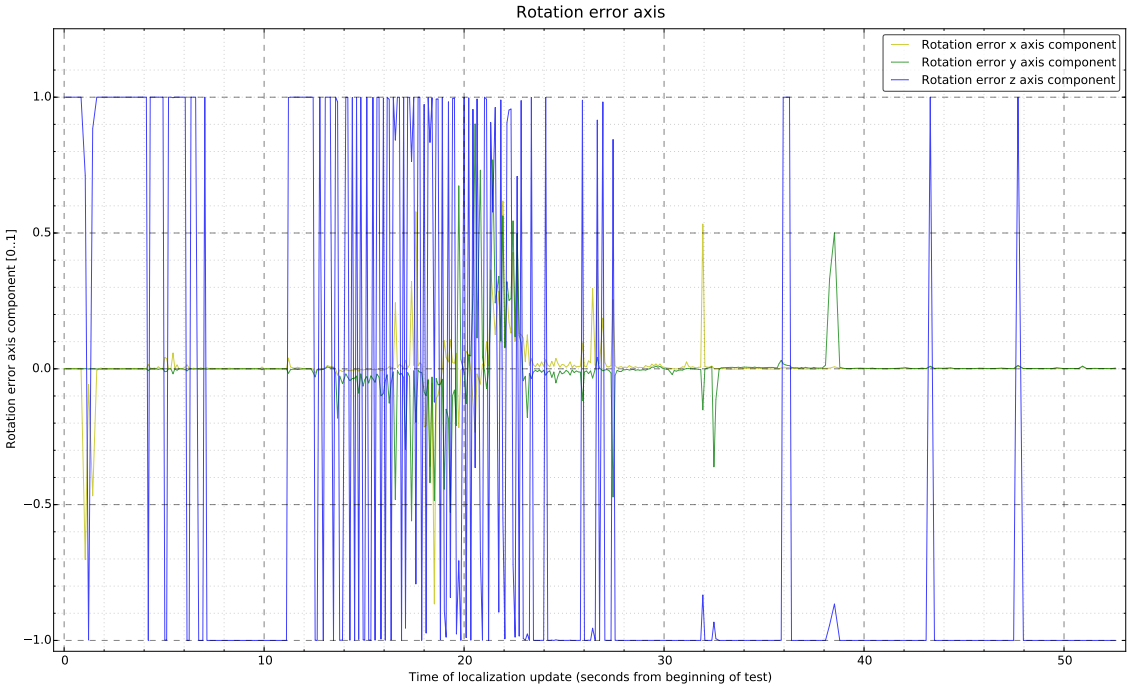


Figure A.3.3.19: Localization system rotation errors axis

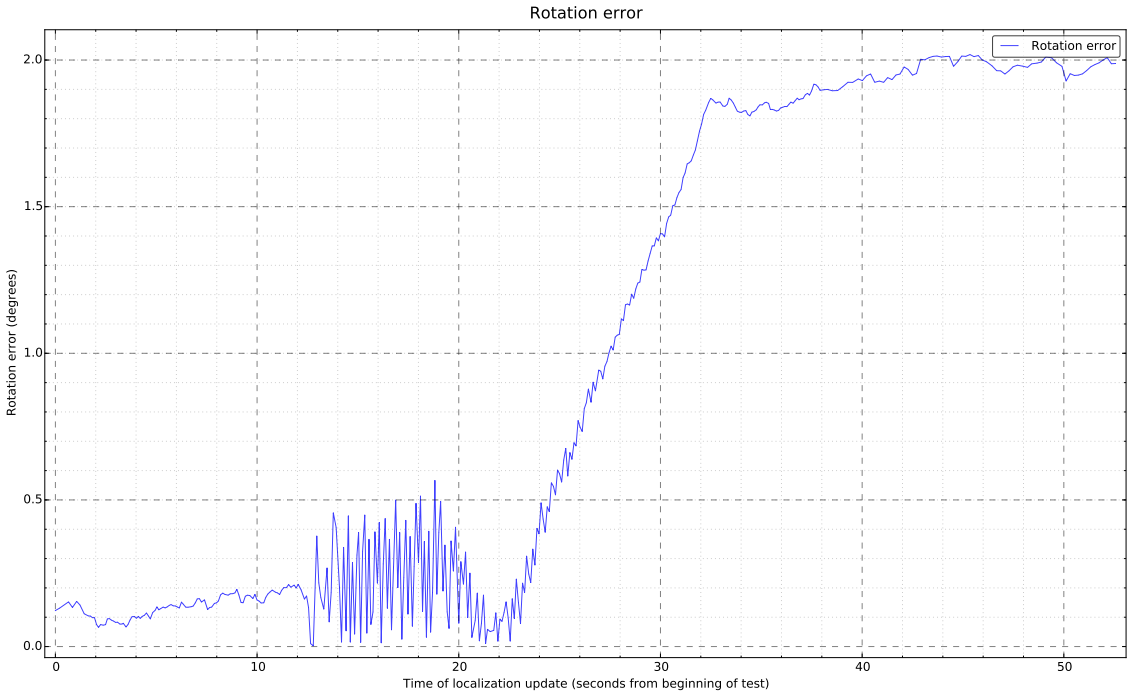


Figure A.3.3.20: Odometry rotation errors

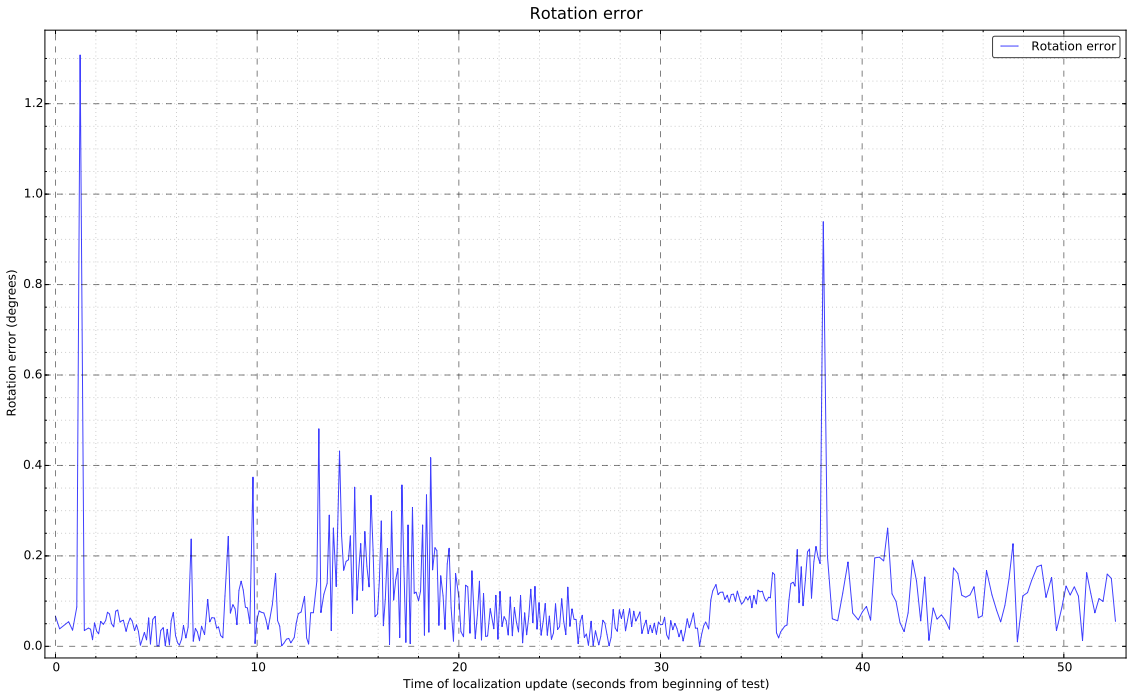


Figure A.3.3.21: Localization system rotation errors

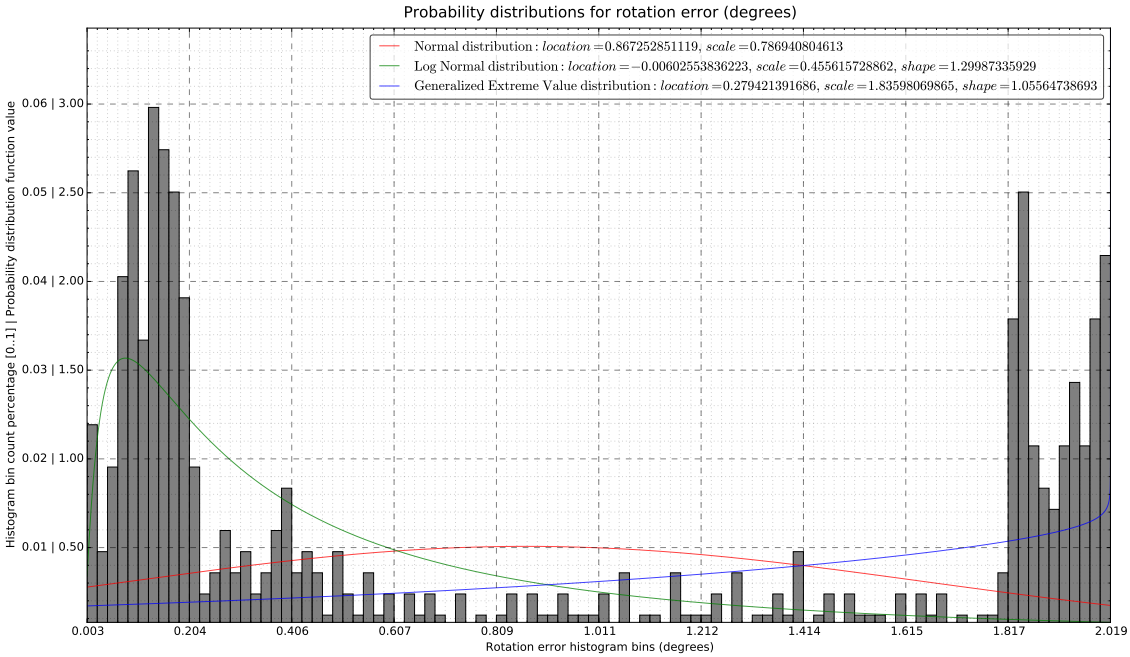


Figure A.3.3.22: Probability distributions for the odometry rotation errors

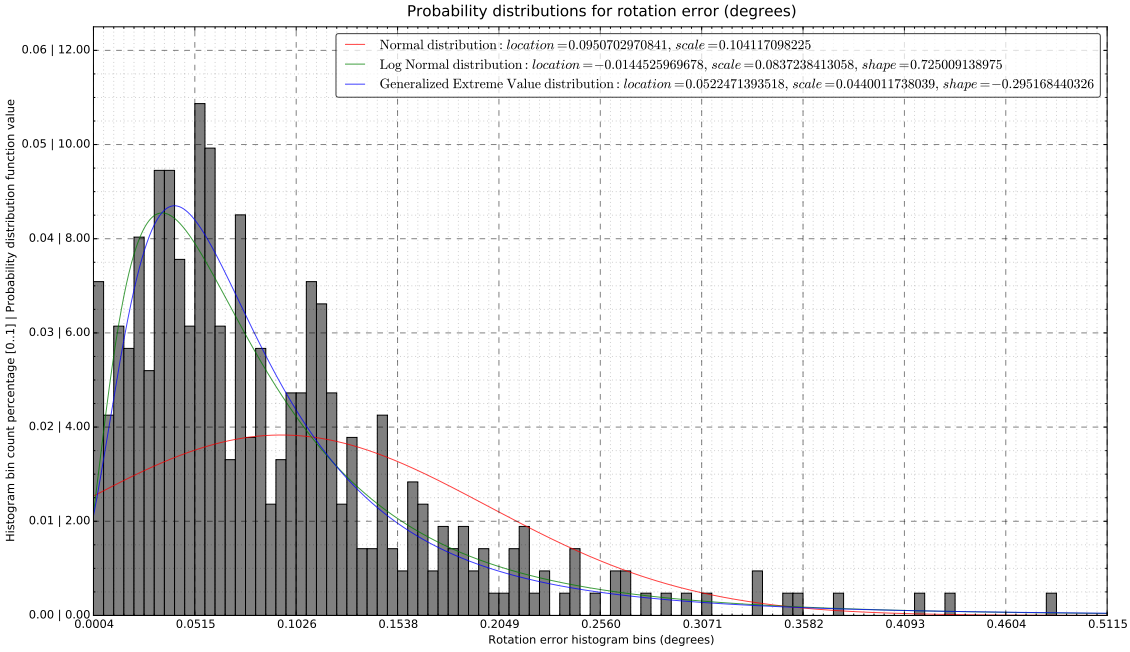


Figure A.3.3.23: Probability distributions for the localization system rotation errors

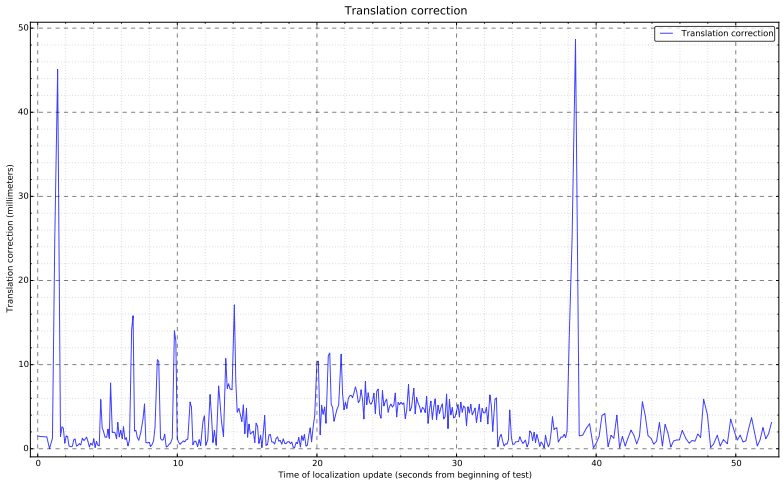


Figure A.3.3.24: Translation corrections performed by the localization system

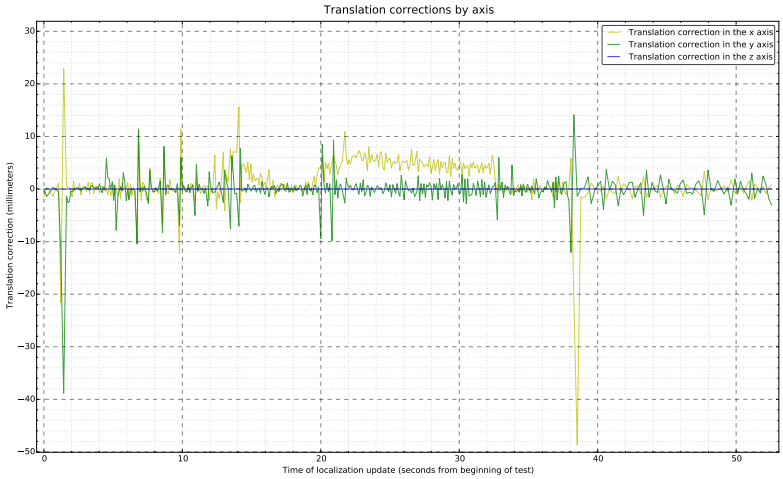


Figure A.3.3.25: Translation corrections components performed by the localization system

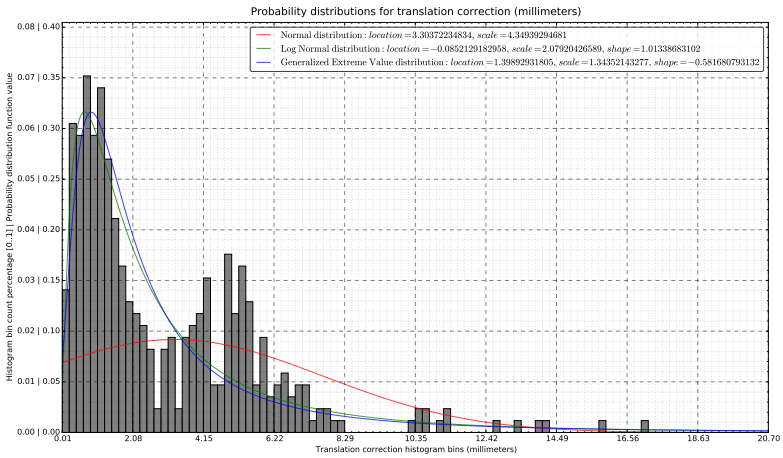


Figure A.3.3.26: Probability distributions for the translation corrections performed by the localization system

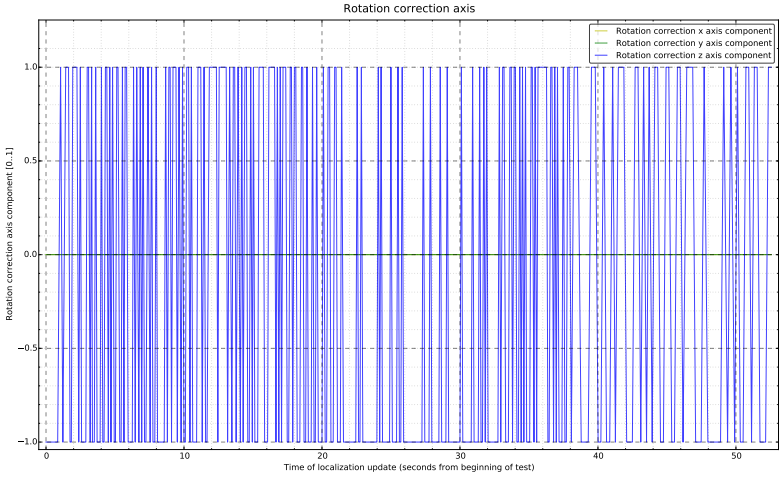


Figure A.3.3.27: Rotation corrections (axis) performed by the localization system

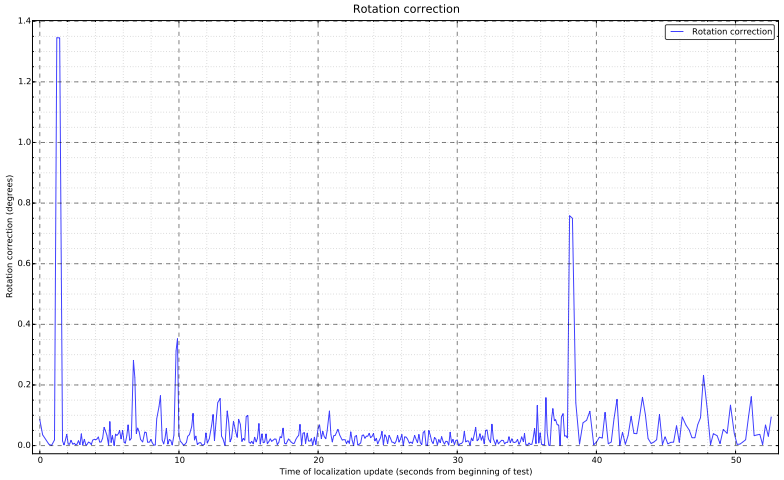


Figure A.3.3.28: Rotation corrections performed by the localization system

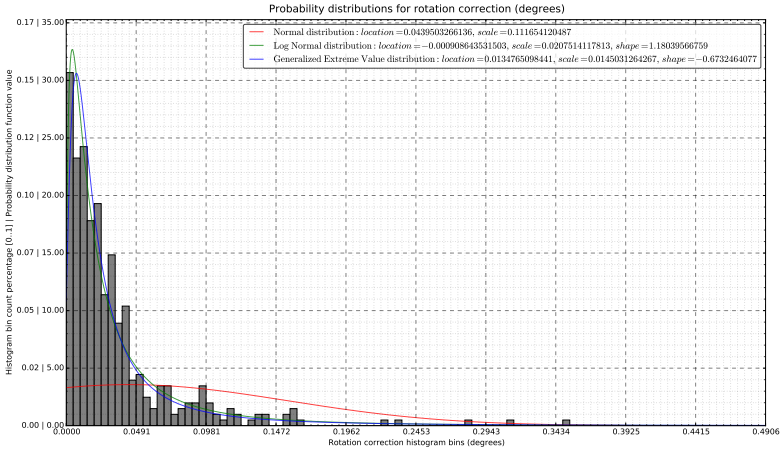


Figure A.3.3.29: Probability distributions for the rotation corrections performed by the localization system

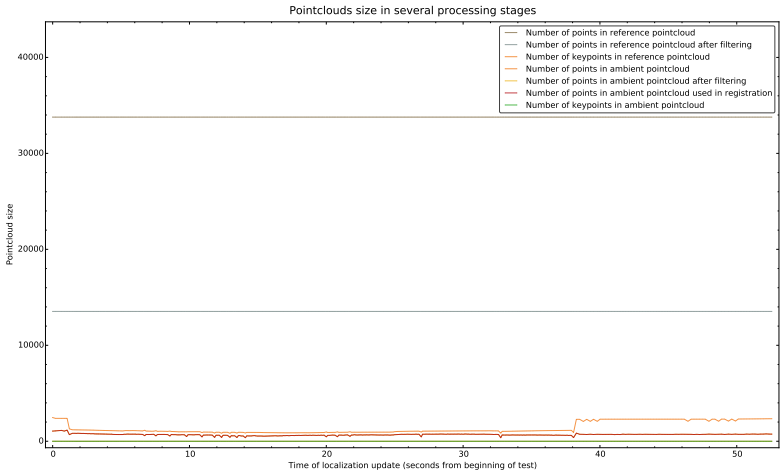


Figure A.3.3.30: Point clouds size in several of the localization system processing stages

Figure A.3.3.31: Number of registered points

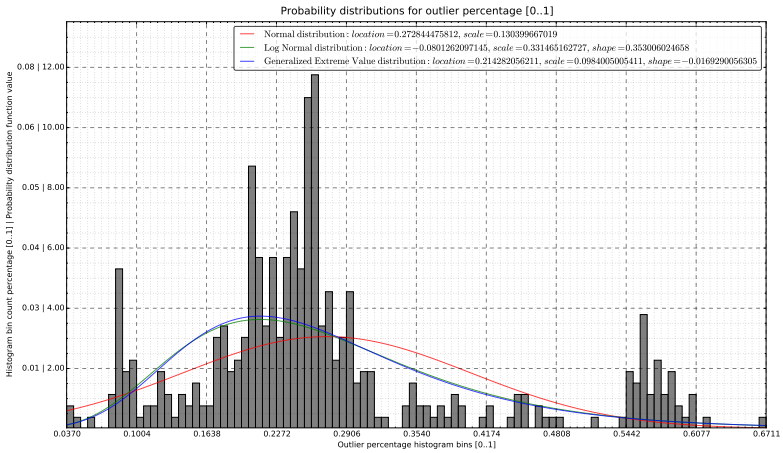


Figure A.3.3.32: Probability distributions for the ambient point cloud outlier percentage

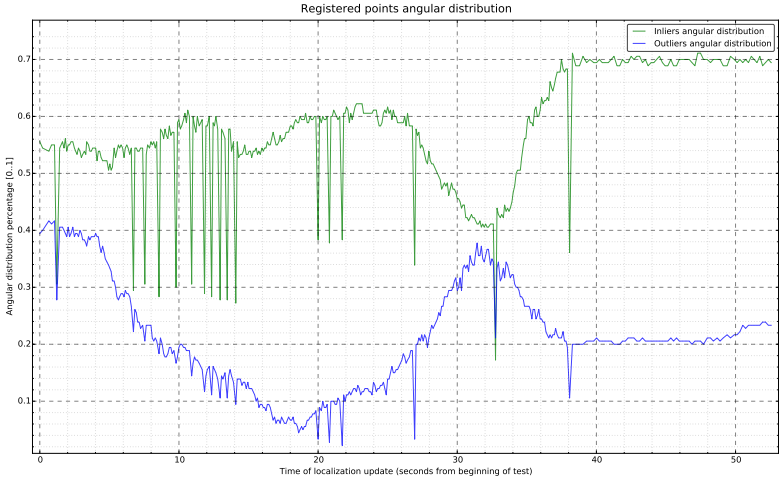


Figure A.3.3.33: Angular distribution percentage of registered points

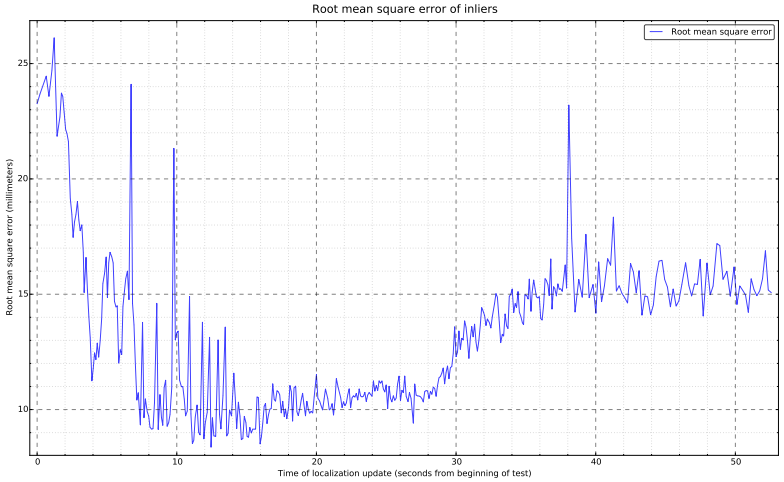


Figure A.3.3.34: Root Mean Square Error of the inliers

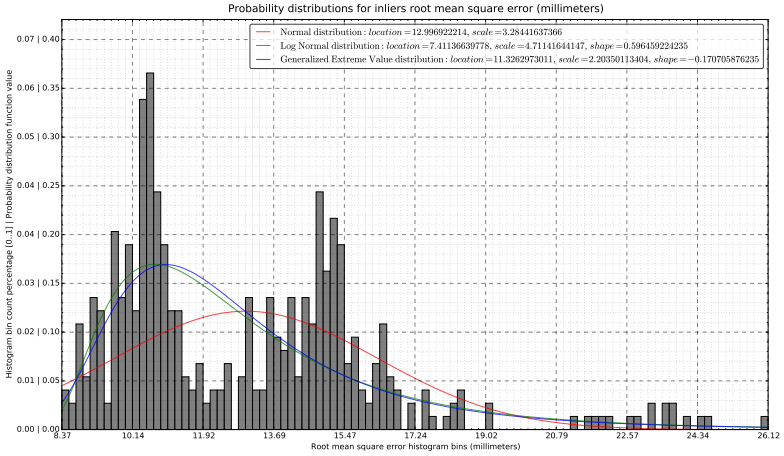


Figure A.3.3.35: Probability distributions for the Root Mean Square Error of the inliers

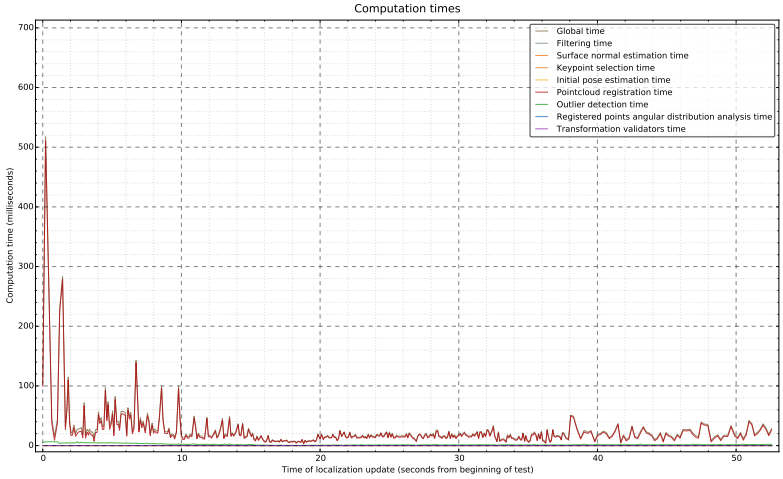


Figure A.3.3.36: Localization system computation times

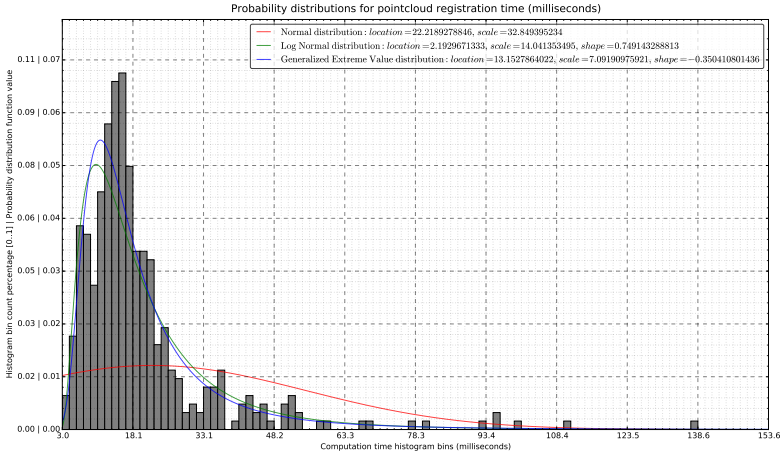


Figure A.3.3.37: Probability distributions for the point cloud registration time

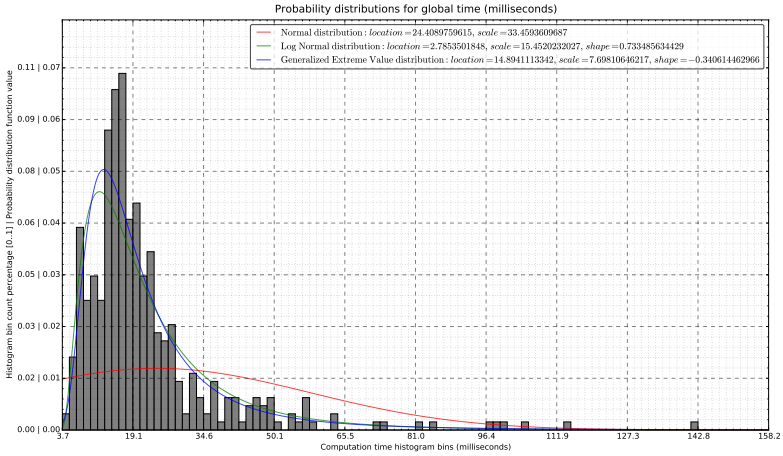


Figure A.3.3.38: Probability distributions for the localization system global computation time

Appendix B

6 DoF localization system results

The next sections present in detail the results achieved with the localization system configured to 6 DoF.

Each test starts by presenting an interactive sequence of images in order to give an overview of the sensor data retrieved along the movement path of the Kinect. To allow a more broad analysis of the point cloud registration quality achieved with the proposed localization system, it is also provided figures with all the Kinect point clouds assembled using the estimated poses. For a more visual inspection, the synchronized poses from the localization system (blue arrows) and ground truth (green arrows) are also presented. To have a much more detailed description of the Kinect movement path, several more graphs are shown with information about the evolution of the x, y and z position, along with the distance that the Kinect traveled between pose estimations and the associated linear / angular velocities / accelerations. Later on, the translation and rotation errors and their respective probability distributions are shown. In the end, it is presented some graphs with information about the point clouds size in each of the main processing stages of the localization system pipeline, along with the number of registered points, inliers analysis (percentage and RMSE) and finally some more graphs showing how much time the localization system required for each pose estimation.

B.1 Kinect tests

The next subsections show some of the most relevant results achieved with the ETHZ-ASL Kinect dataset [PMC+11].

B.1.1 Kinect with overview flying motion at 30 cm/s

The figures in this section show the detailed results of test performed with the Kinect sensor with a 20 mm resolution map in the flying arena following a smooth path and moving at 30 cm/s.

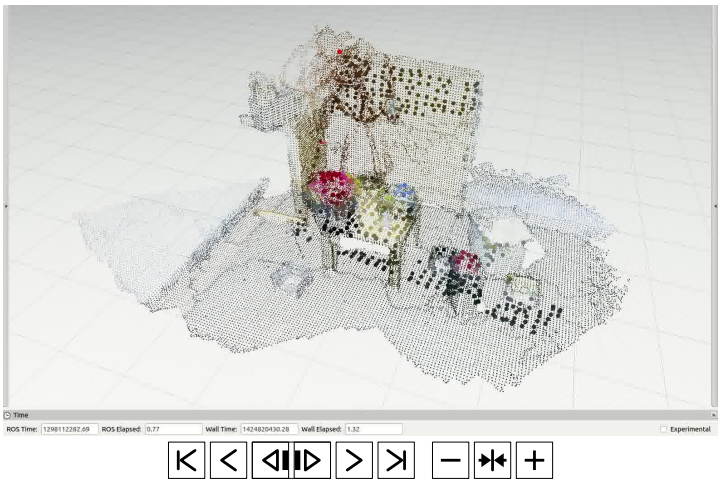


Figure B.1.1.1: Animation of point clouds using the localization system poses

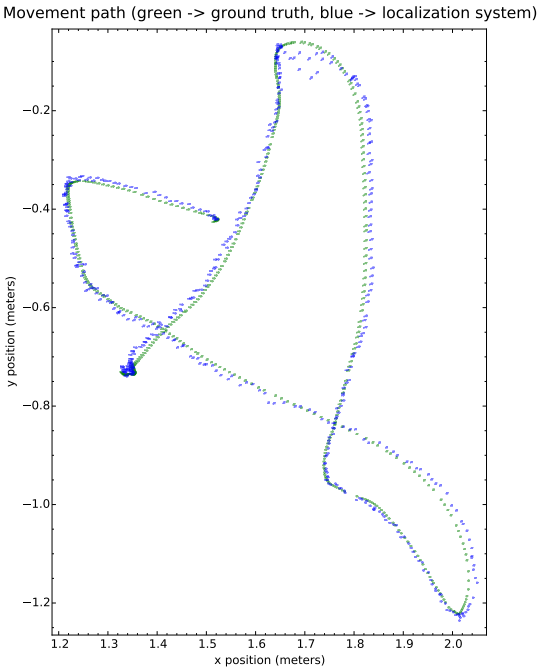


Figure B.1.1.2: Poses estimated by the ground truth and localization system

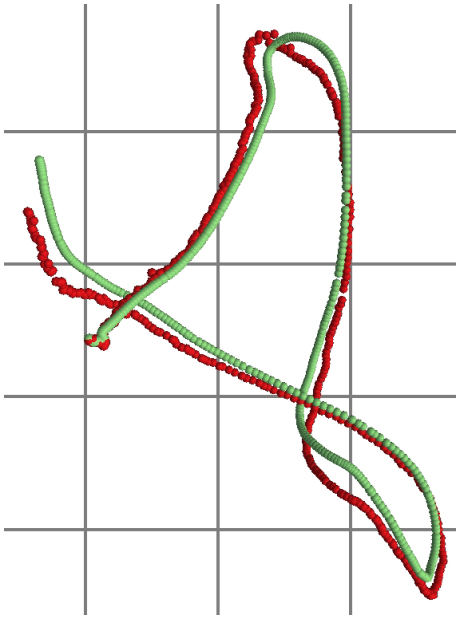


Figure B.1.1.3: Poses estimated by the *ethzasl_icp_mapper* package [PMC+11]

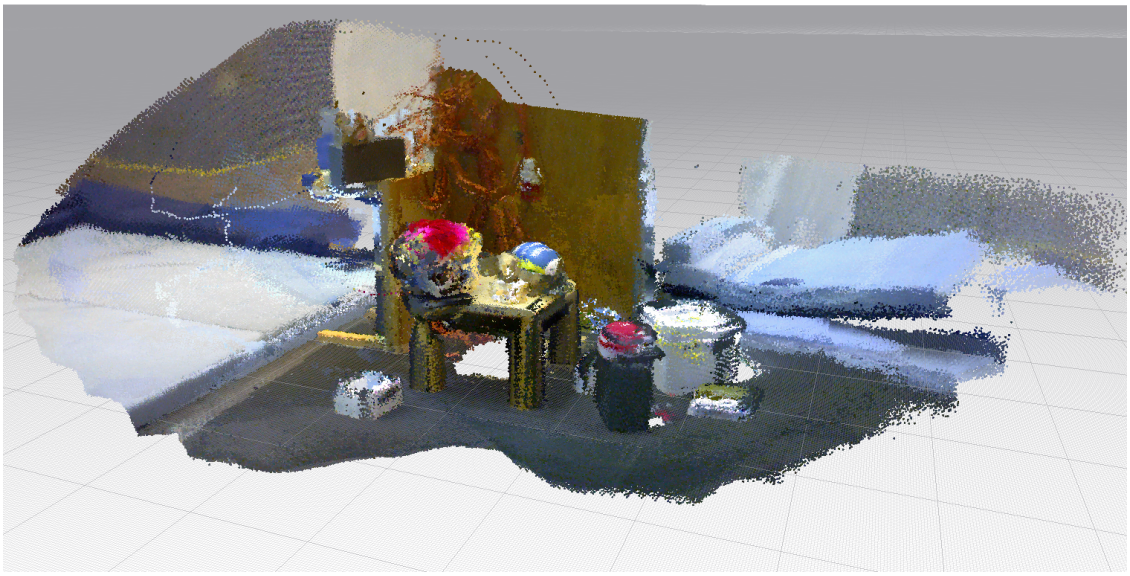


Figure B.1.1.4: Point clouds assembled on top of the map using the ground truth poses

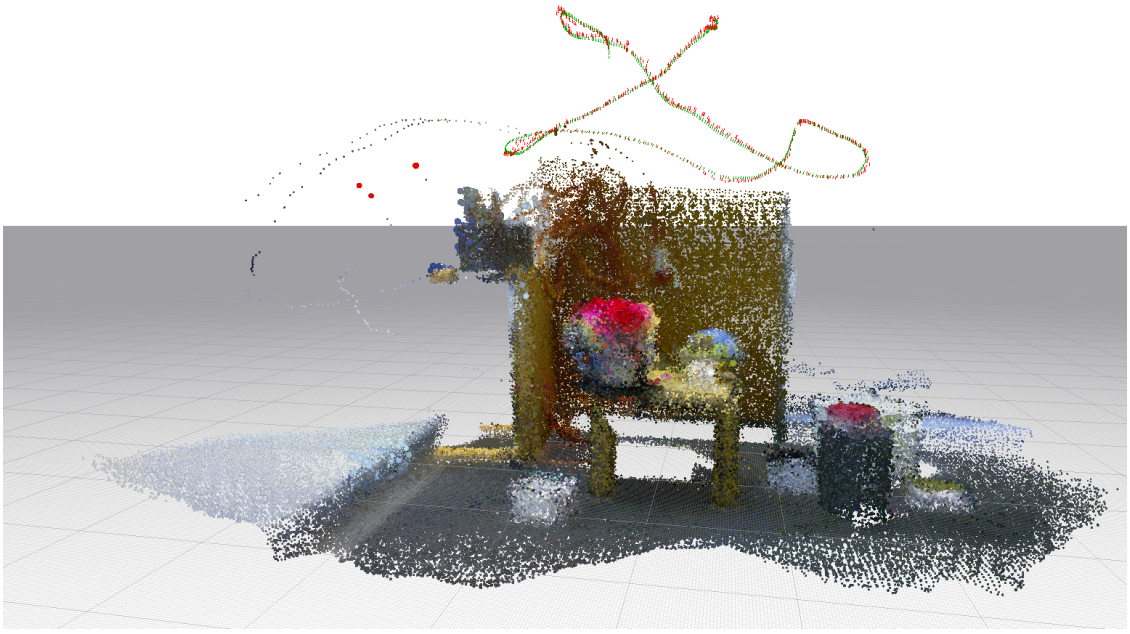


Figure B.1.1.5: Point clouds assembled on top of the map using the localization system poses

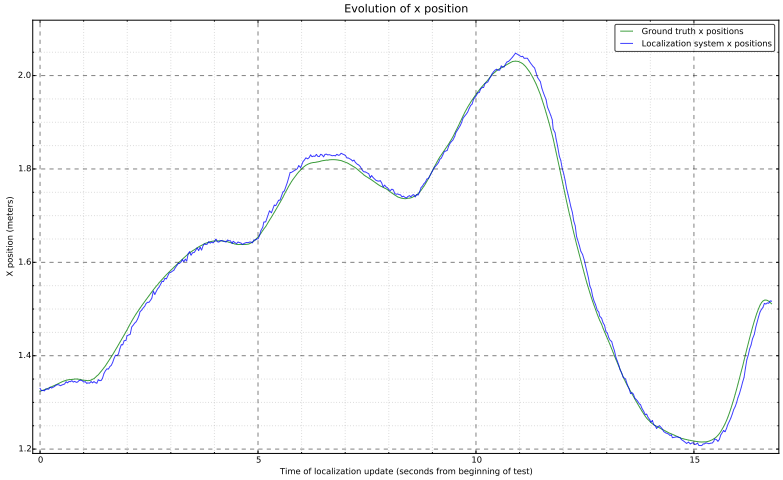


Figure B.1.1.6: Evolution of x position over time

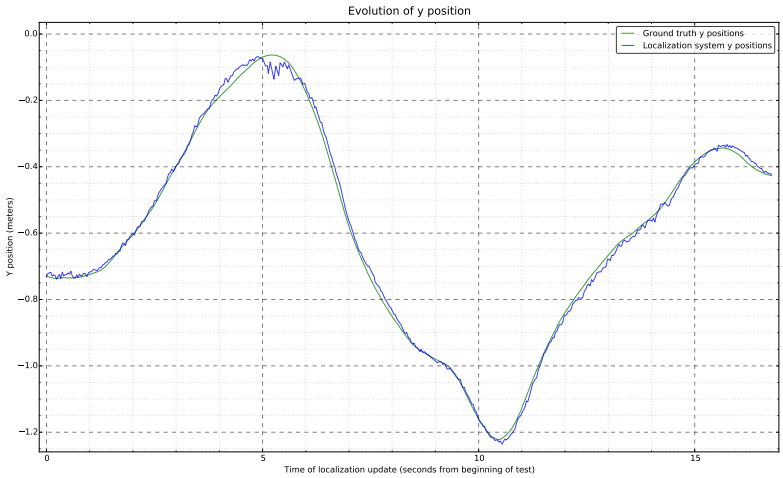


Figure B.1.1.7: Evolution of y position over time

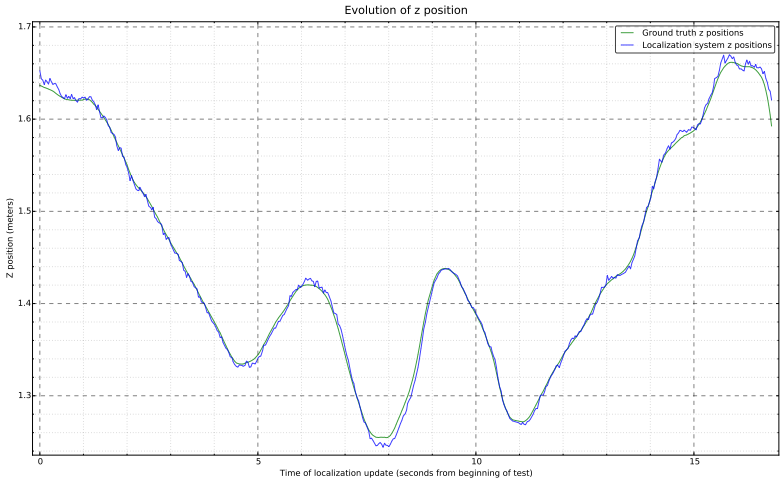


Figure B.1.1.8: Evolution of z position over time

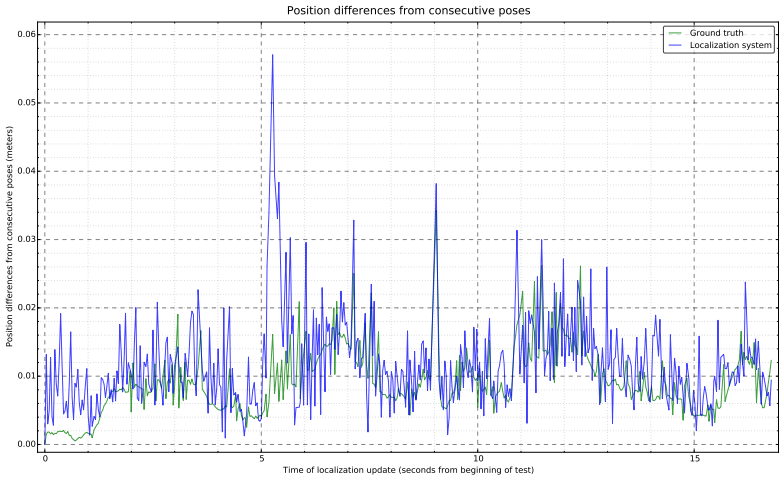


Figure B.1.1.9: Distance traveled between consecutive pose estimations

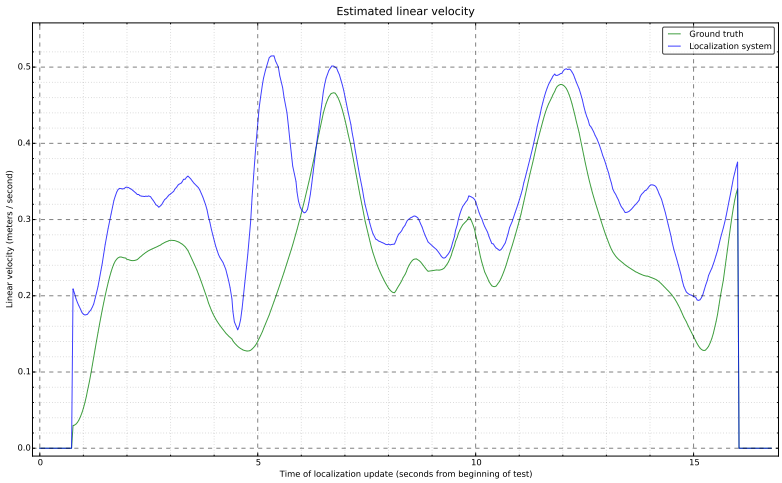


Figure B.1.1.10: Estimated linear velocity

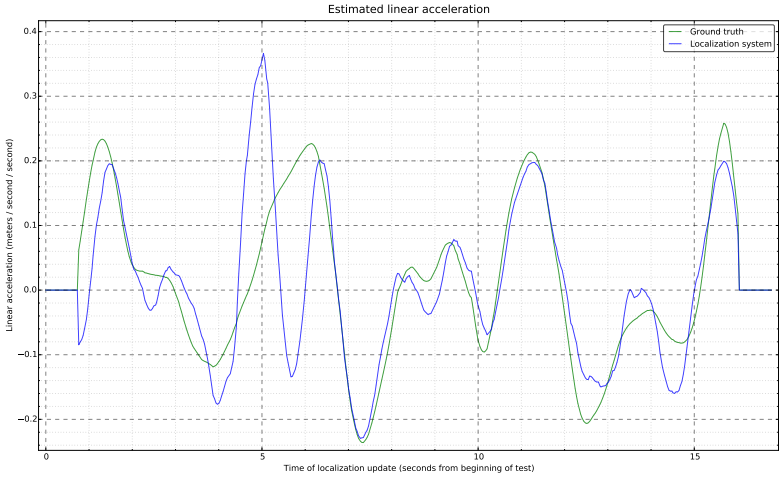


Figure B.1.1.11: Estimated linear acceleration

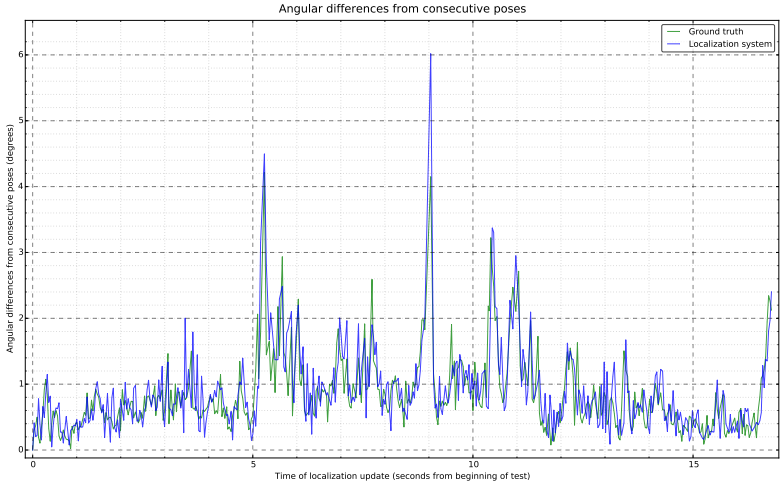


Figure B.1.1.12: Angular differences between consecutive pose estimations

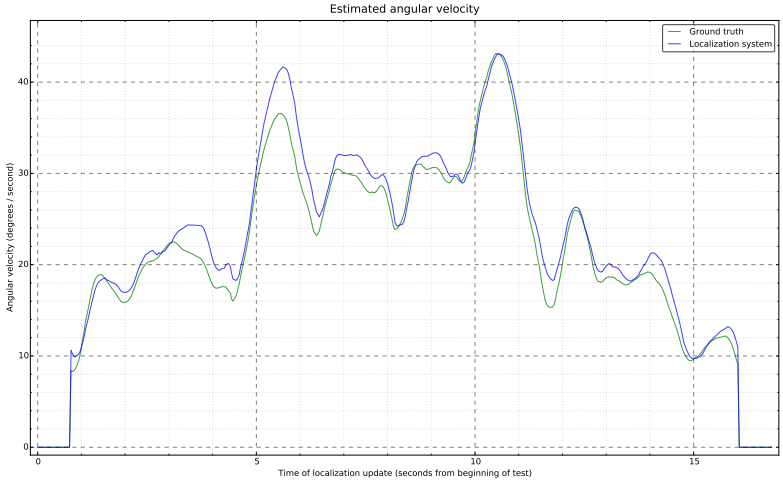


Figure B.1.1.13: Estimated angular velocity

Figure B.1.1.14: Estimated angular acceleration

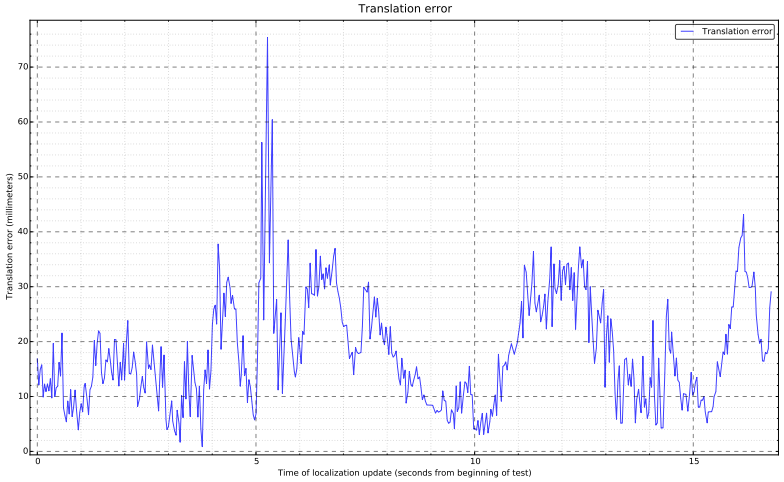


Figure B.1.1.15: Localization system translation errors

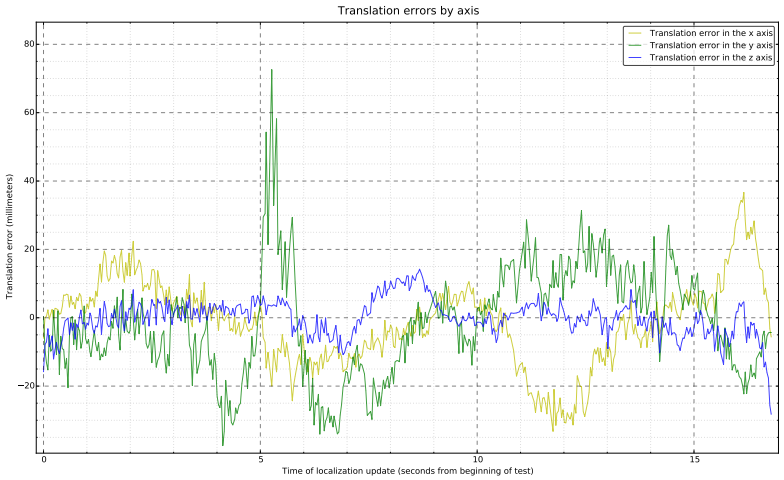


Figure B.1.1.16: Localization system translation errors components

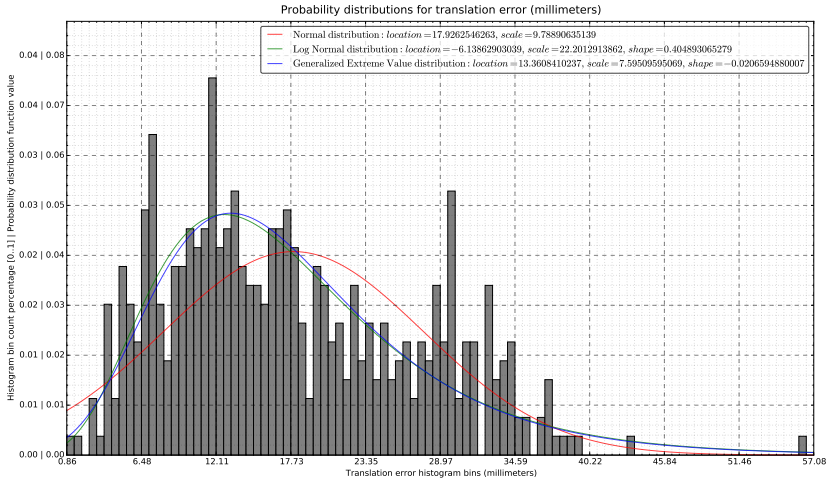


Figure B.1.1.17: Probability distributions for the localization system translation errors

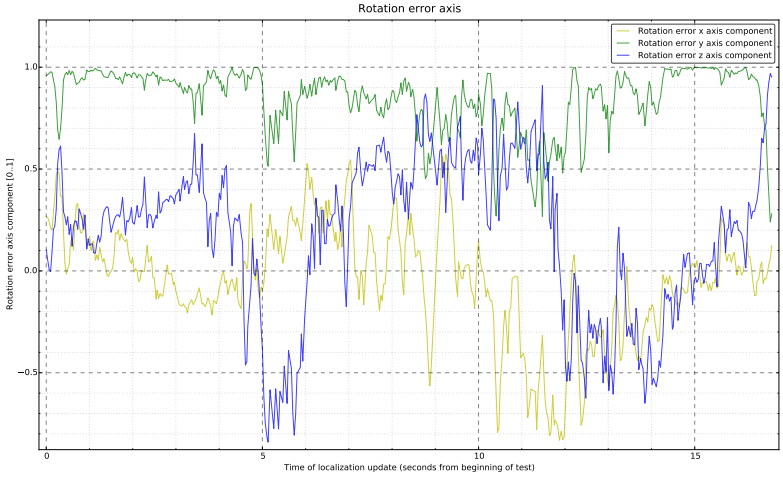


Figure B.1.1.18: Localization system rotation errors axis

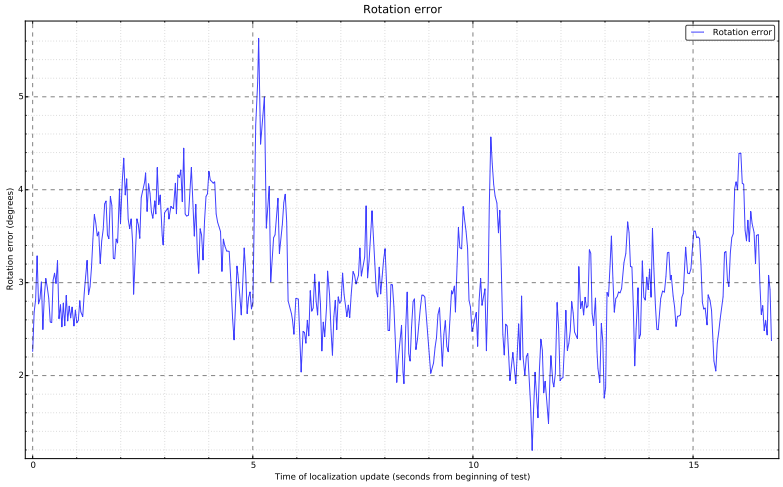


Figure B.1.1.19: Localization system rotation errors

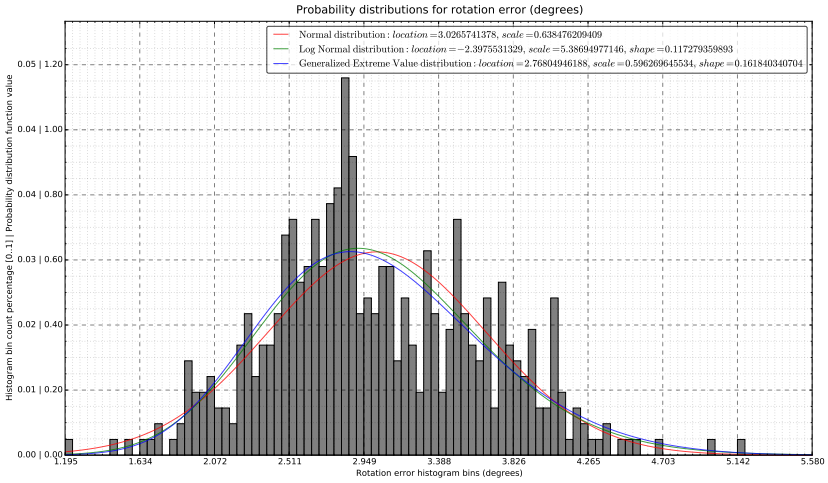


Figure B.1.1.20: Probability distributions for the localization system rotation errors

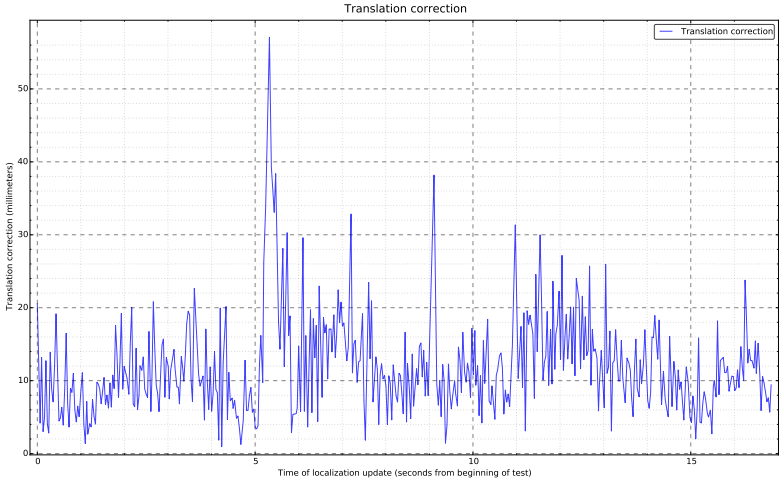


Figure B.1.1.21: Translation corrections performed by the localization system

Figure B.1.1.22: Translation corrections components performed by the localization system

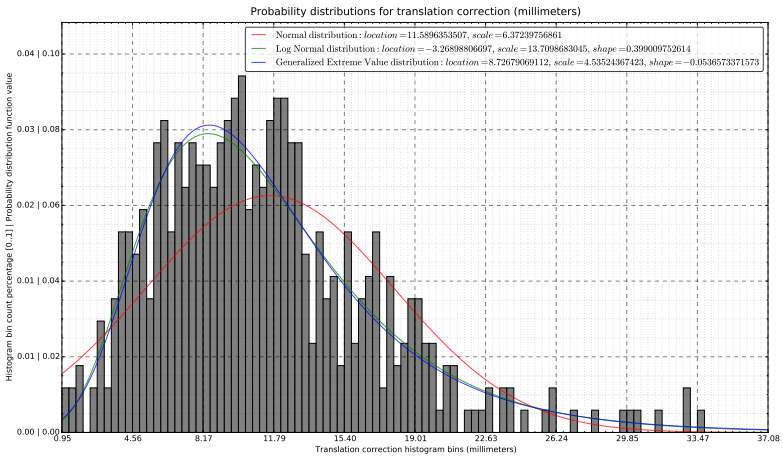


Figure B.1.1.23: Probability distributions for the translation corrections performed by the localization system

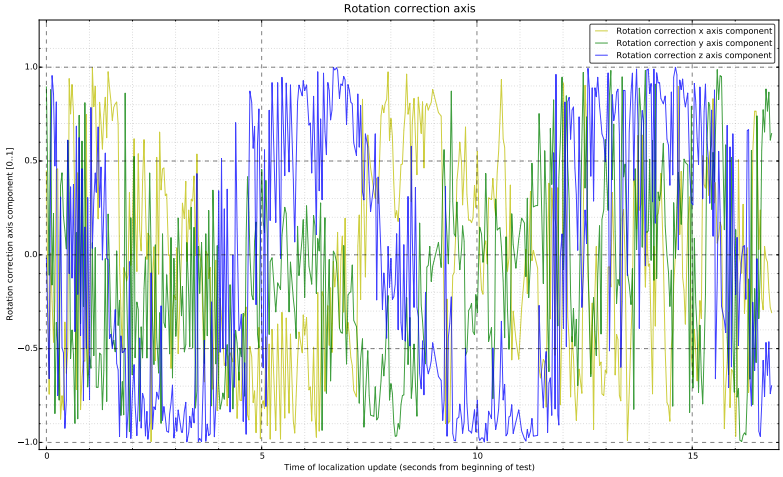


Figure B.1.1.24: Rotation corrections (axis) performed by the localization system

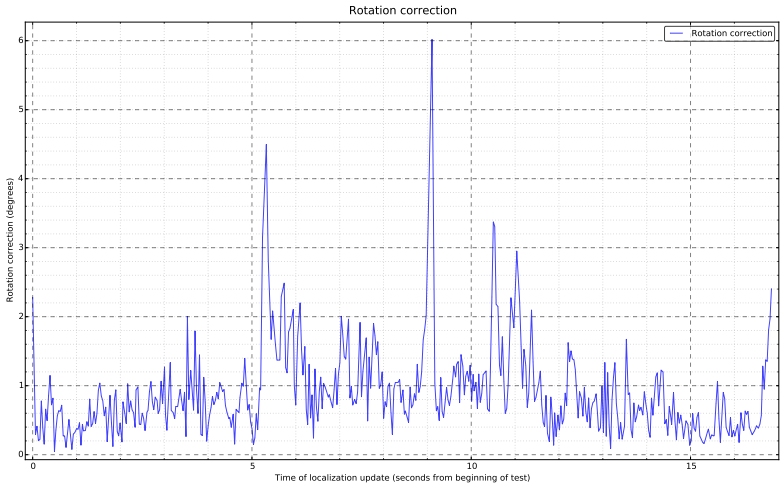


Figure B.1.1.25: Rotation corrections performed by the localization system

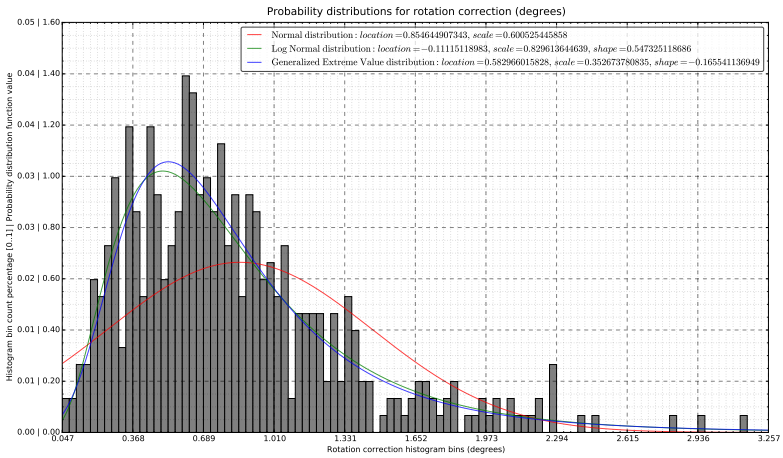


Figure B.1.1.26: Probability distributions for the rotation corrections performed by the localization system

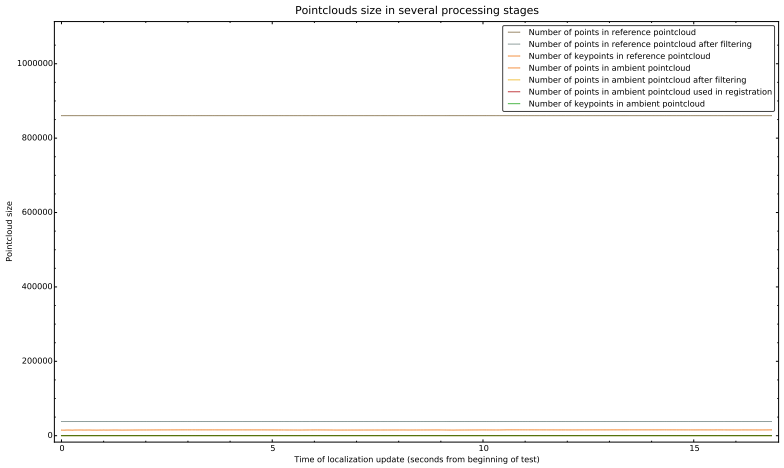


Figure B.1.1.27: Point clouds size in several of the localization system processing stages

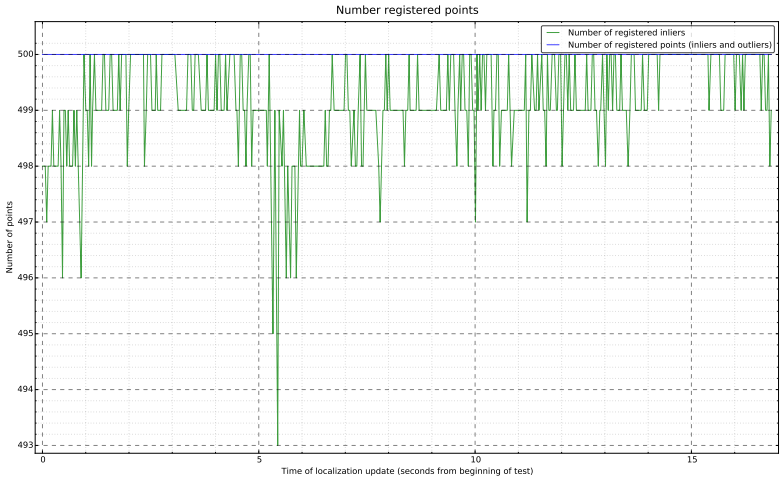


Figure B.1.1.28: Number of registered points

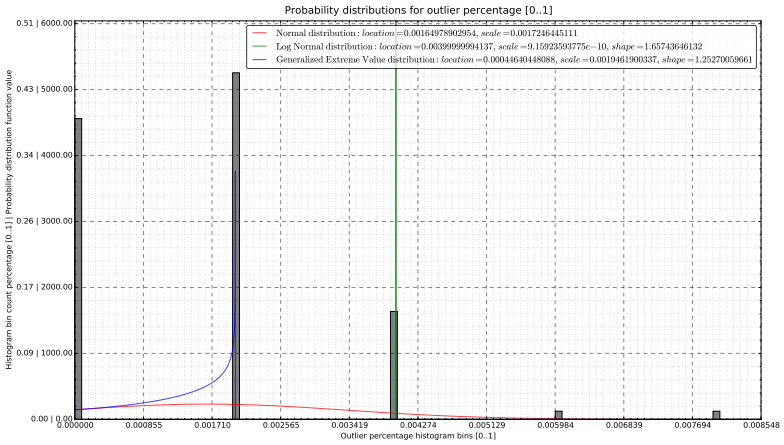


Figure B.1.1.29: Probability distributions for the ambient point cloud outlier percentage

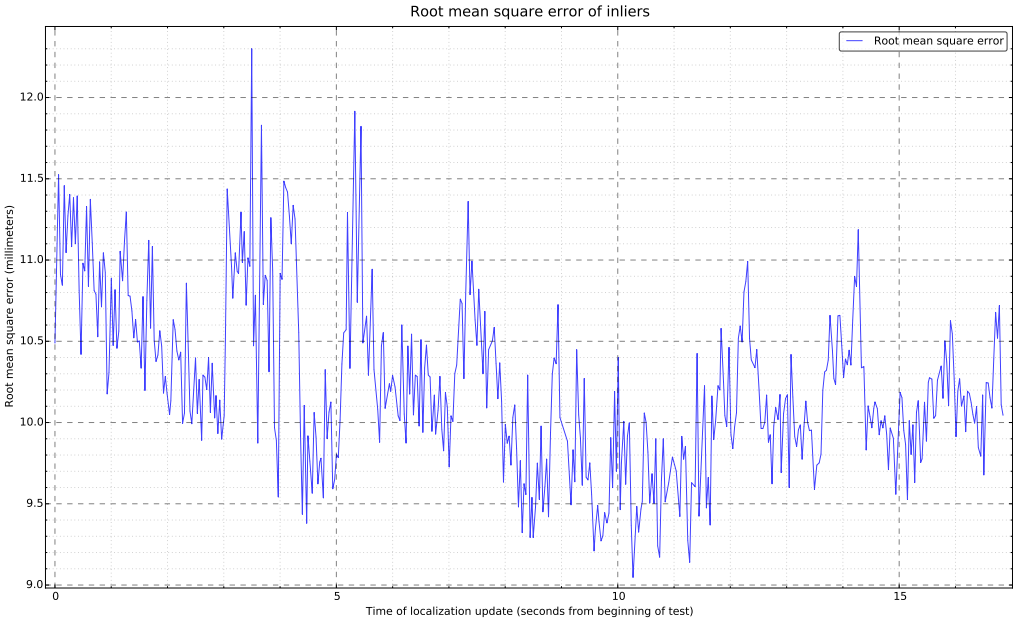


Figure B.1.1.30: Root Mean Square Error of the inliers

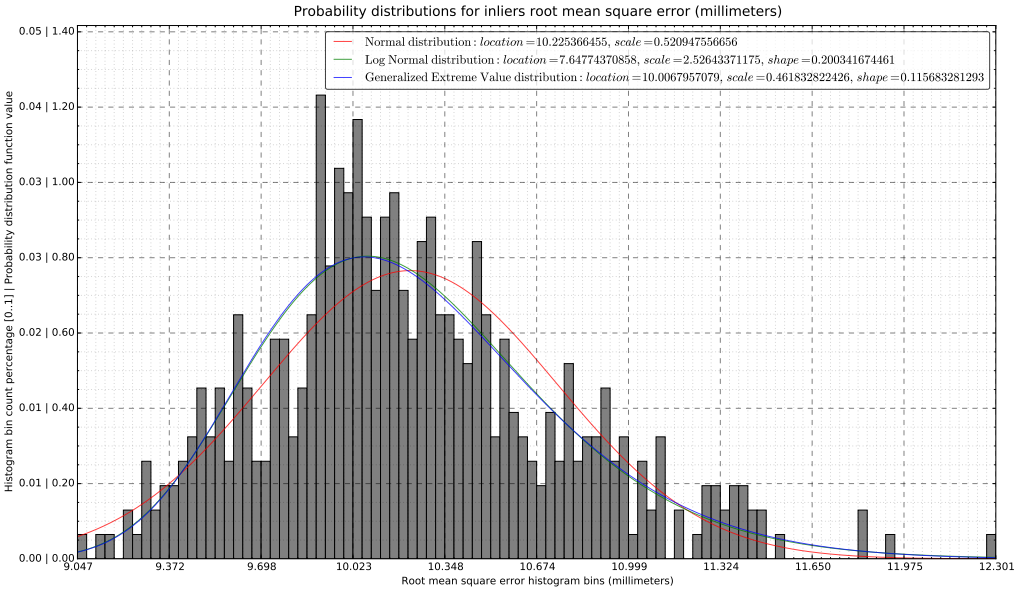


Figure B.1.1.31: Probability distributions for the Root Mean Square Error of the inliers

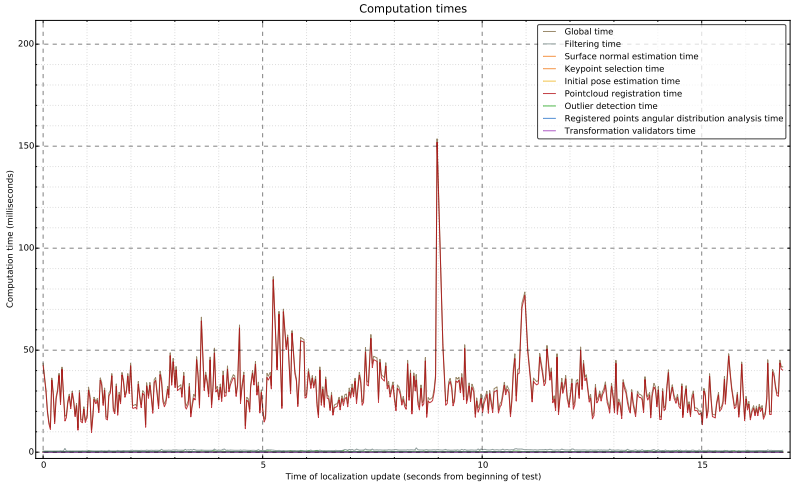


Figure B.1.1.32: Localization system computation times

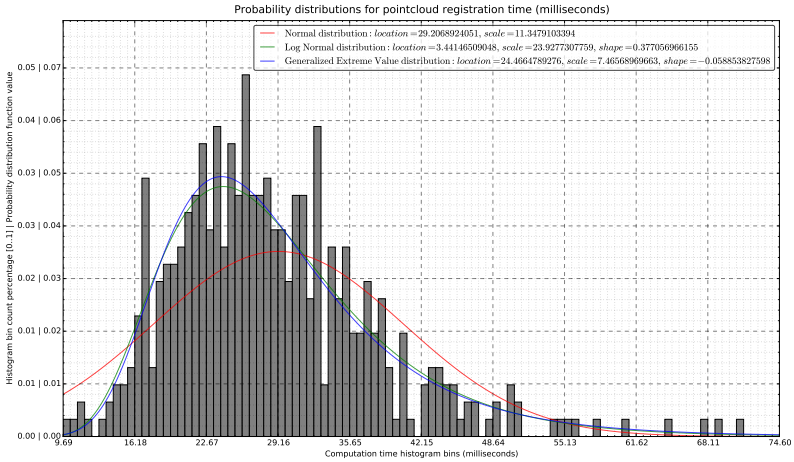


Figure B.1.1.33: Probability distributions for the point cloud registration time

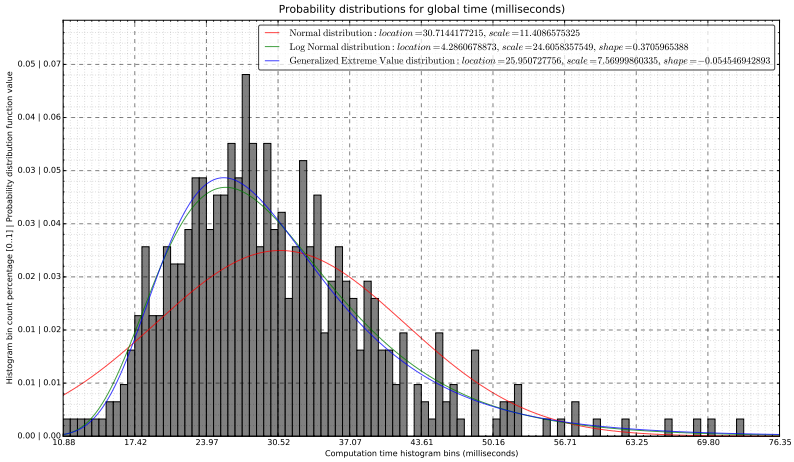


Figure B.1.1.34: Probability distributions for the localization system global computation time

B.1.2 Kinect with translation motions at 20 cm/s

The figures in this section show the detailed results of test performed with the Kinect sensor with a 20 mm resolution map in the flying arena performing several translations in the air and moving at 20 cm/s.

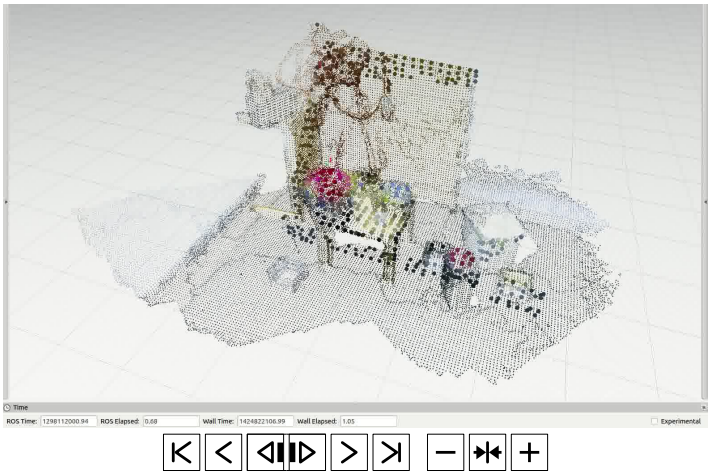


Figure B.1.2.1: Animation of point clouds using the localization system poses

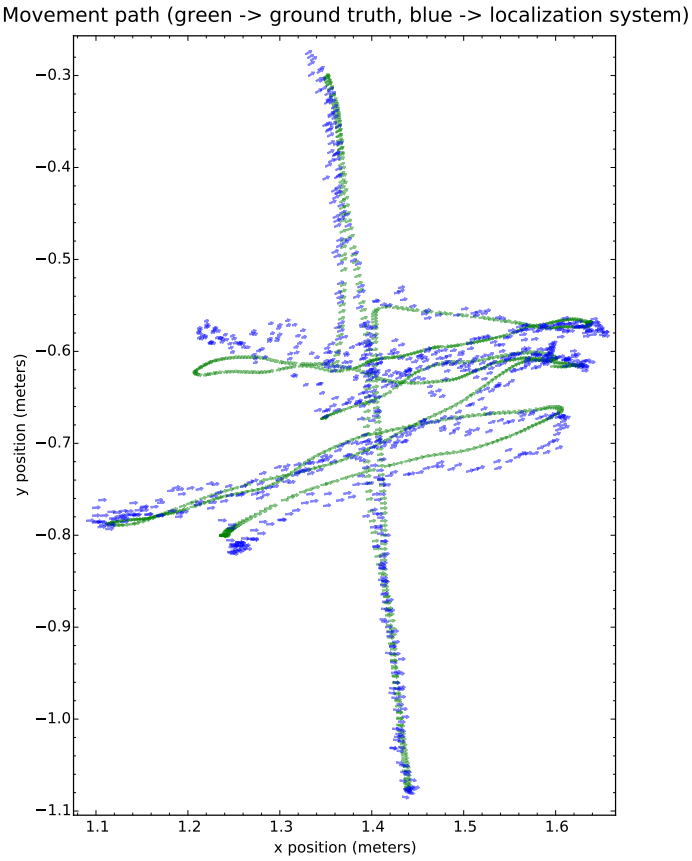


Figure B.1.2.2: Poses estimated by the ground truth and localization system

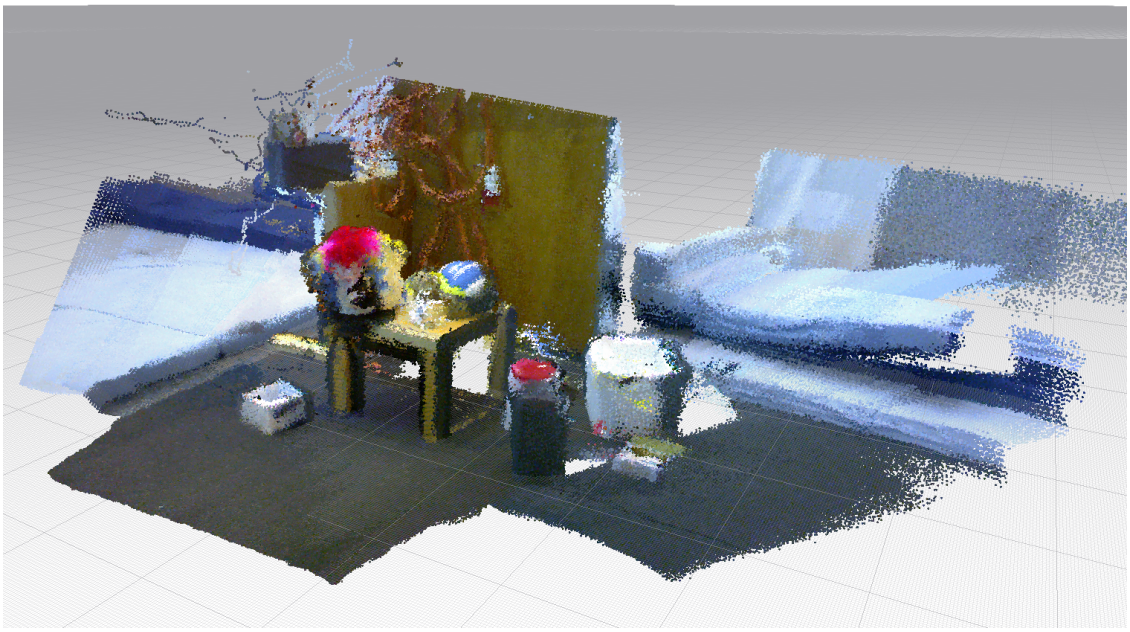


Figure B.1.2.3: Point clouds assembled on top of the map using the ground truth poses

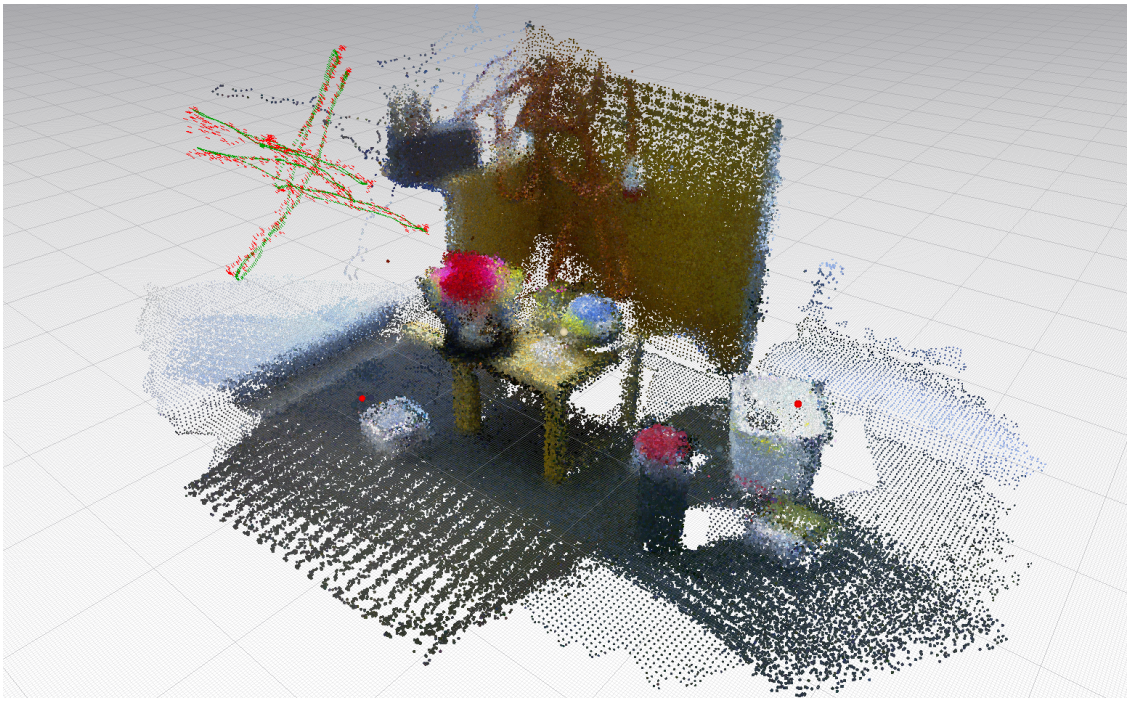


Figure B.1.2.4: Point clouds assembled on top of the map using the localization system poses

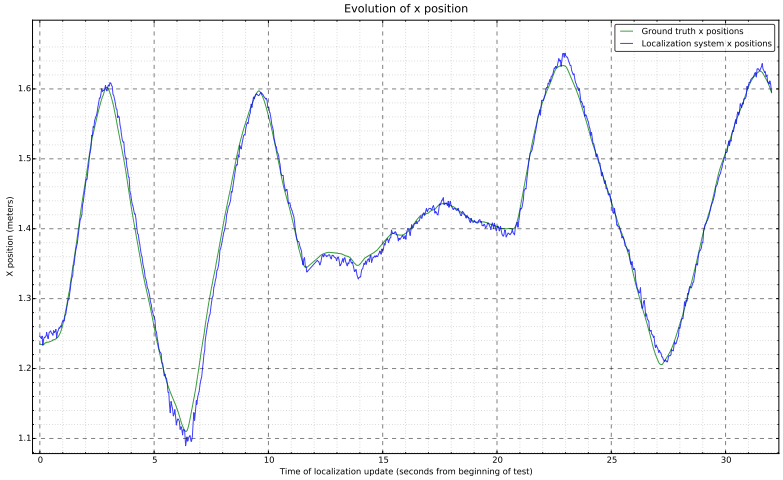


Figure B.1.2.5: Evolution of x position over time

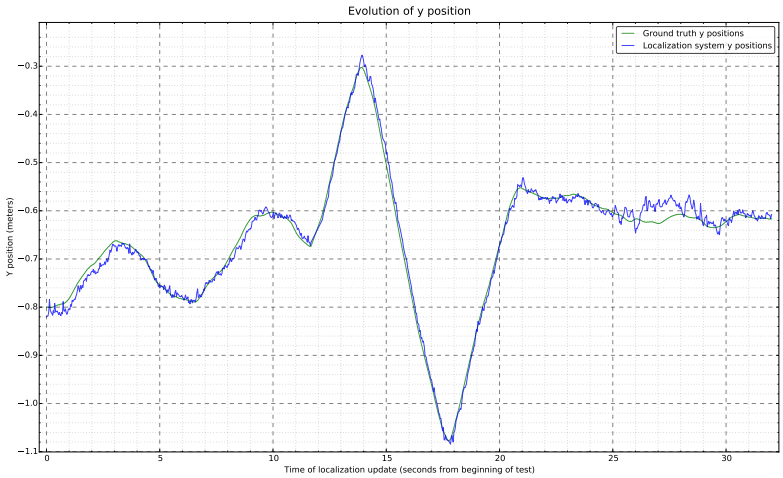


Figure B.1.2.6: Evolution of y position over time

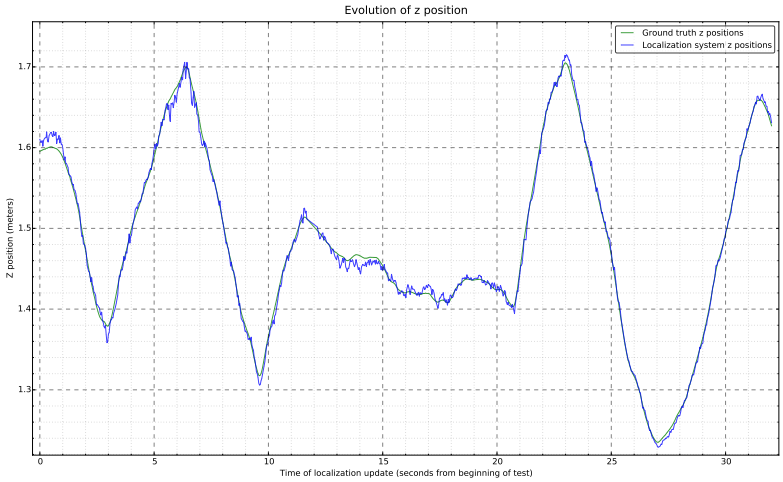


Figure B.1.2.7: Evolution of z position over time

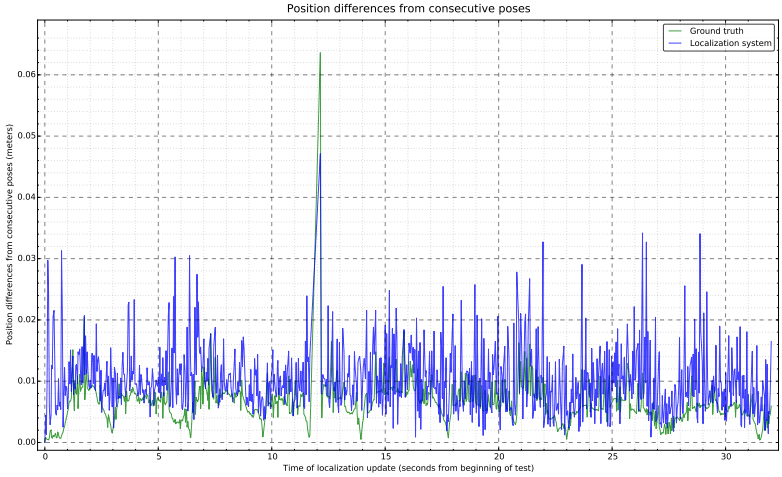


Figure B.1.2.8: Distance traveled between consecutive pose estimations

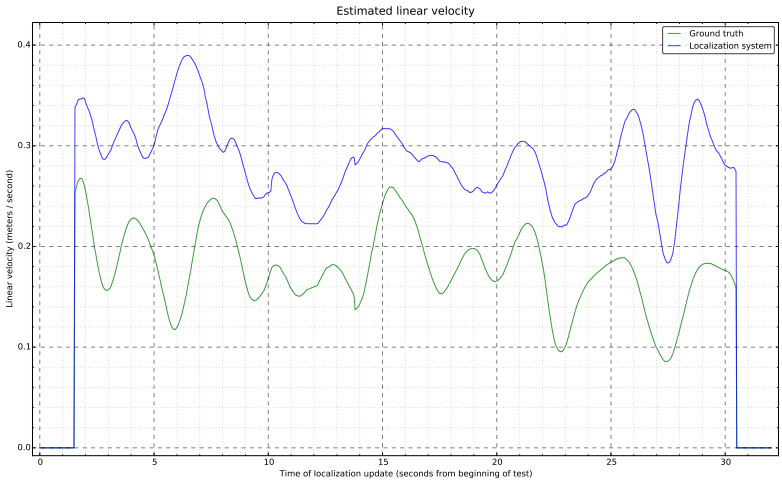


Figure B.1.2.9: Estimated linear velocity

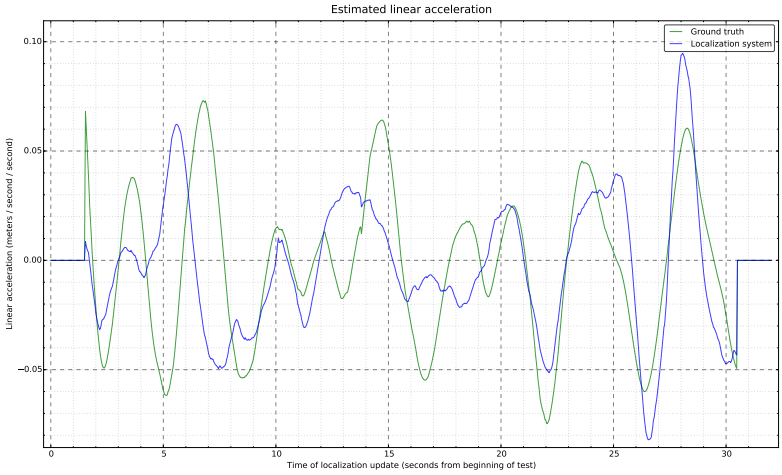


Figure B.1.2.10: Estimated linear acceleration

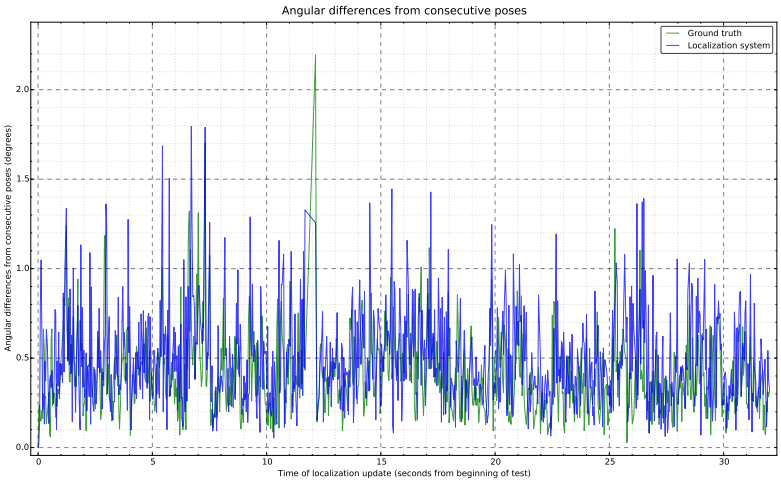


Figure B.1.2.11: Angular differences between consecutive pose estimations

Figure B.1.2.12: Estimated angular velocity

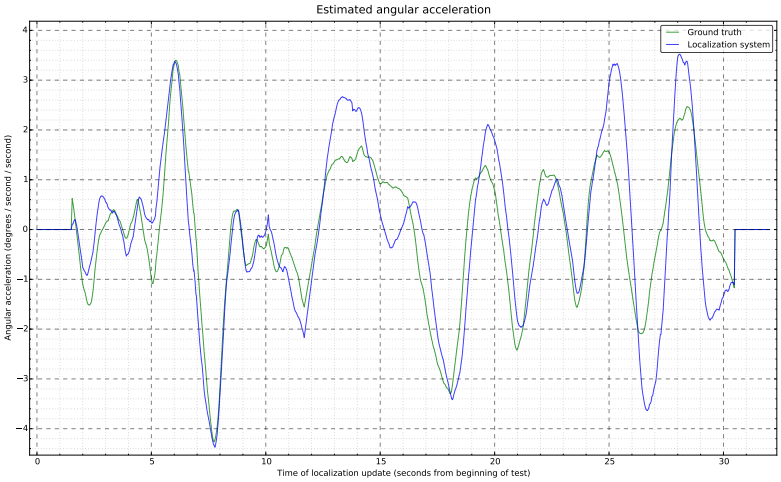


Figure B.1.2.13: Estimated angular acceleration

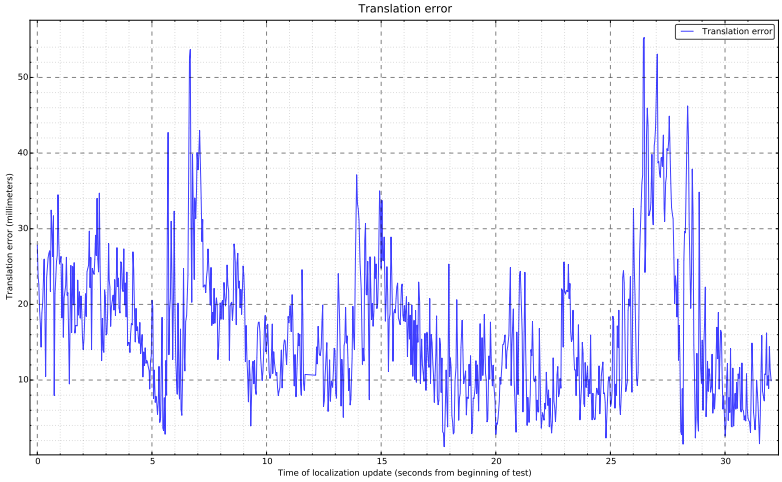


Figure B.1.2.14: Localization system translation errors

Figure B.1.2.15: Localization system translation errors components

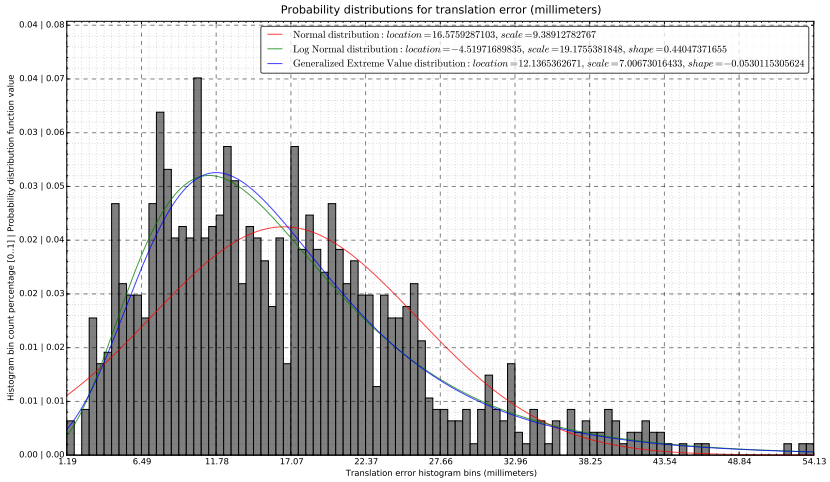


Figure B.1.2.16: Probability distributions for the localization system translation errors

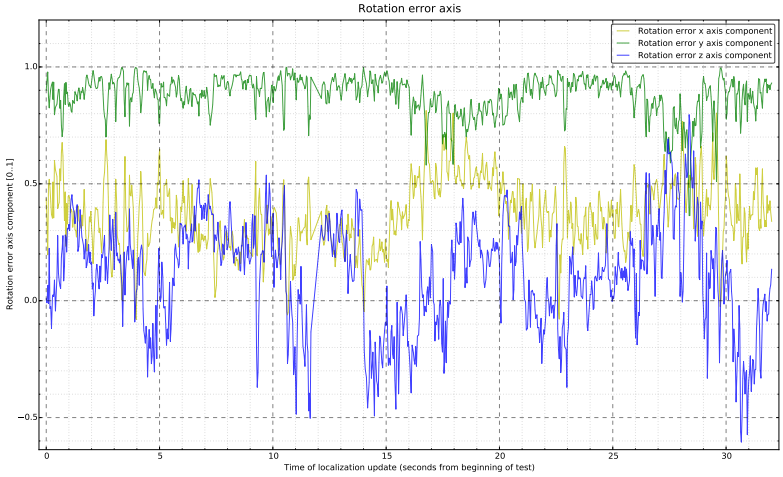


Figure B.1.2.17: Localization system rotation errors axis

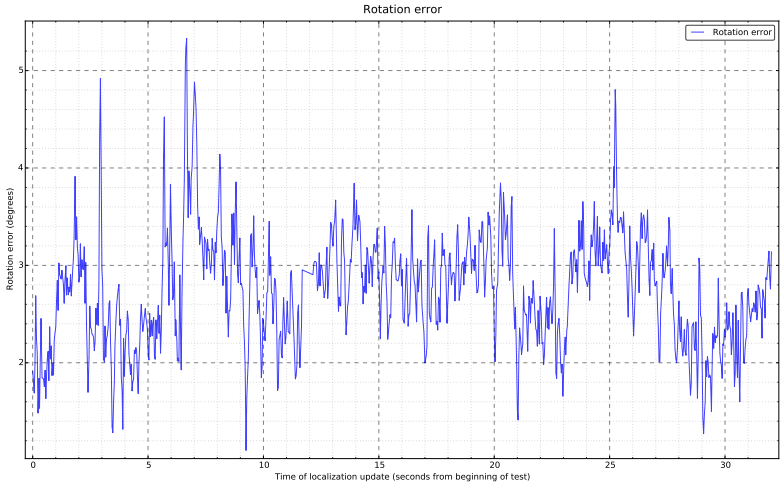


Figure B.1.2.18: Localization system rotation errors

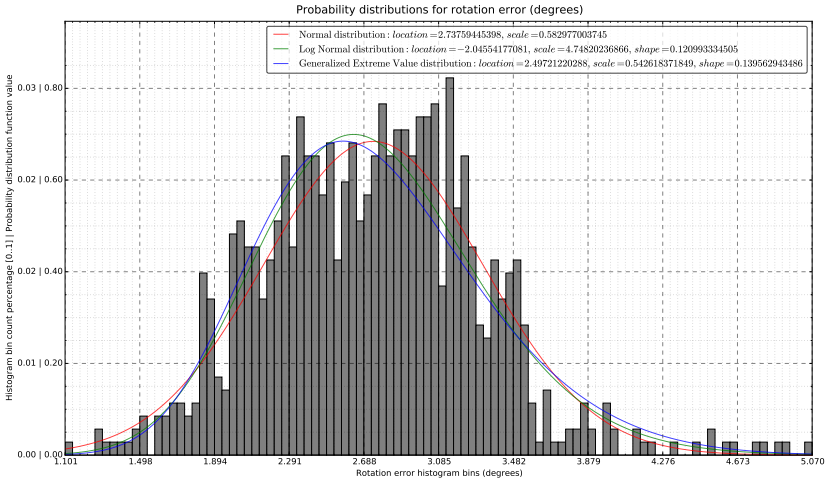


Figure B.1.2.19: Probability distributions for the localization system rotation errors

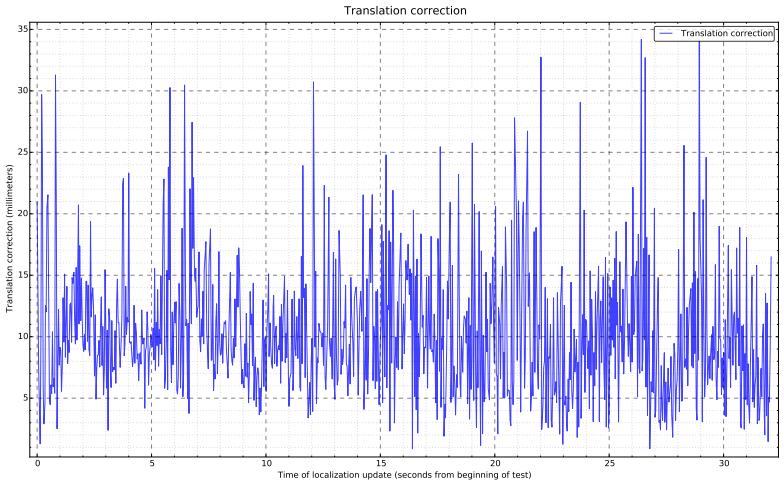


Figure B.1.2.20: Translation corrections performed by the localization system

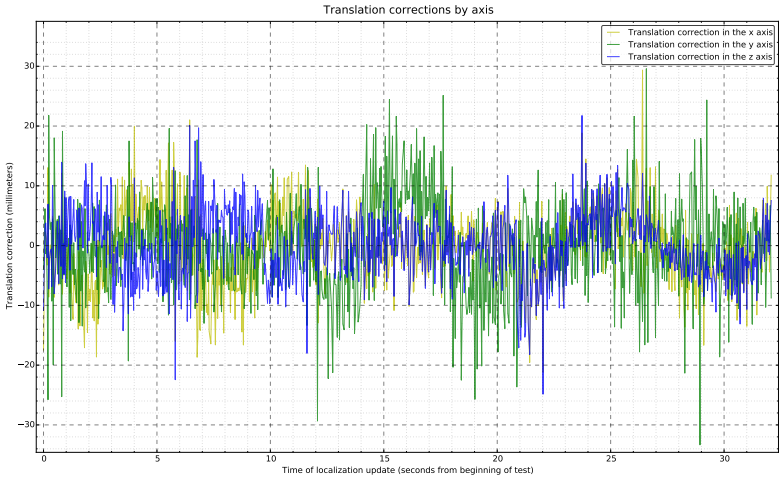


Figure B.1.2.21: Translation corrections components performed by the localization system

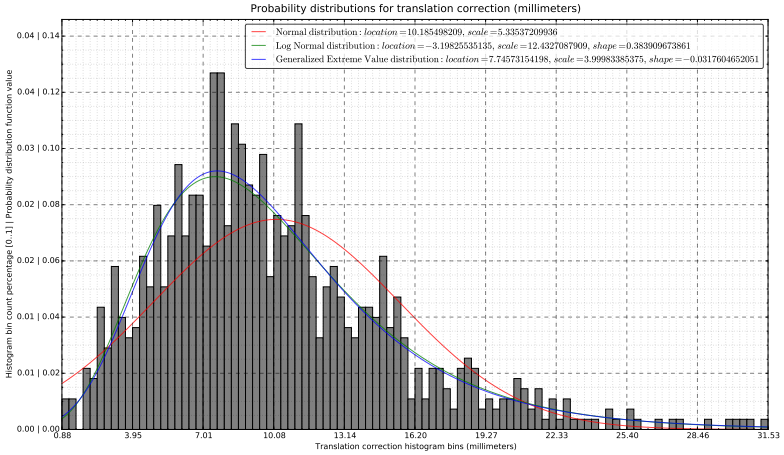


Figure B.1.2.22: Probability distributions for the translation corrections performed by the localization system

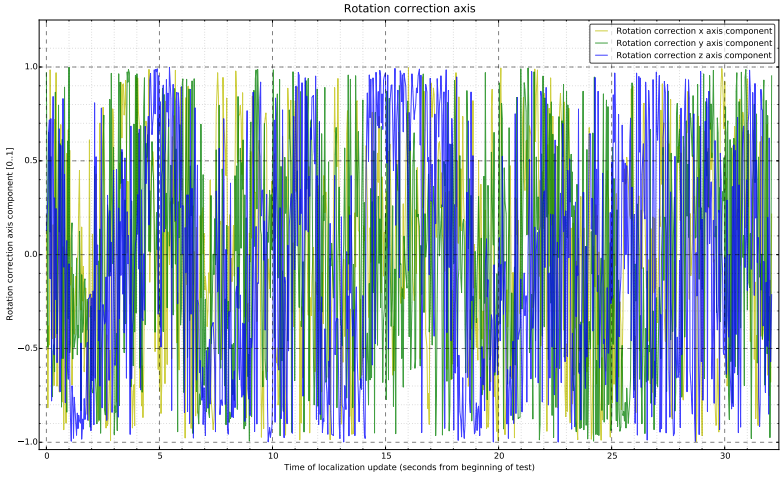


Figure B.1.2.23: Rotation corrections (axis) performed by the localization system

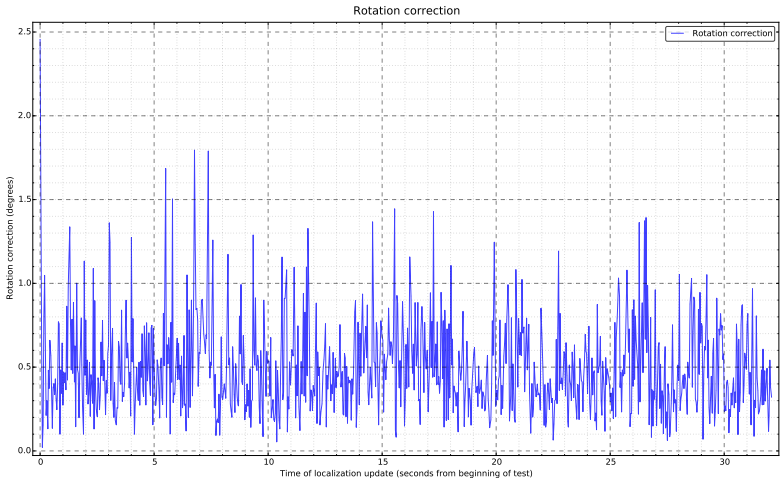


Figure B.1.2.24: Rotation corrections performed by the localization system

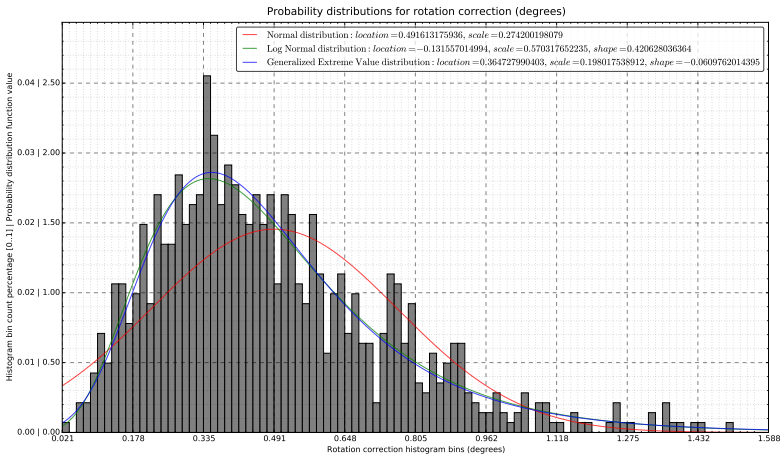


Figure B.1.2.25: Probability distributions for the rotation corrections performed by the localization system

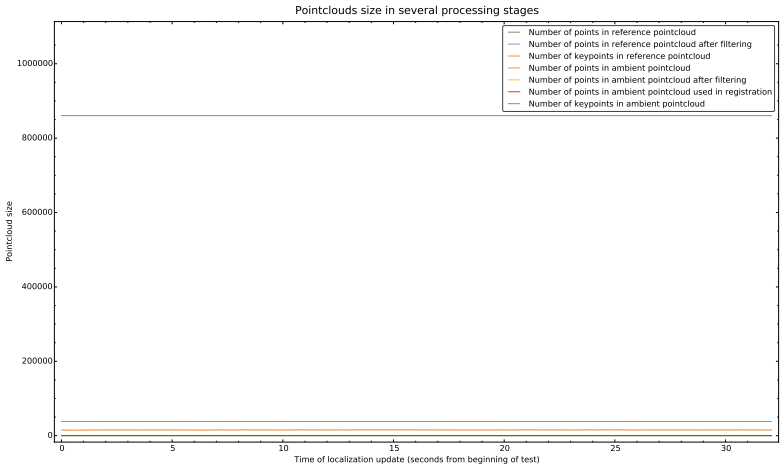


Figure B.1.2.26: Point clouds size in several of the localization system processing stages

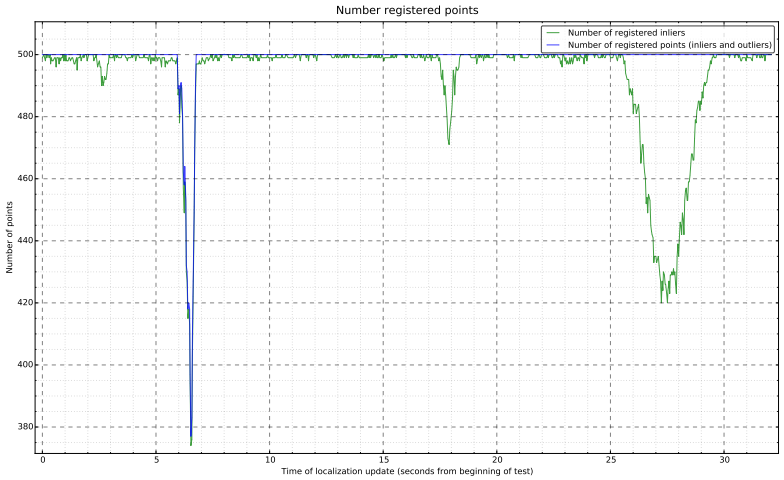


Figure B.1.2.27: Number of registered points

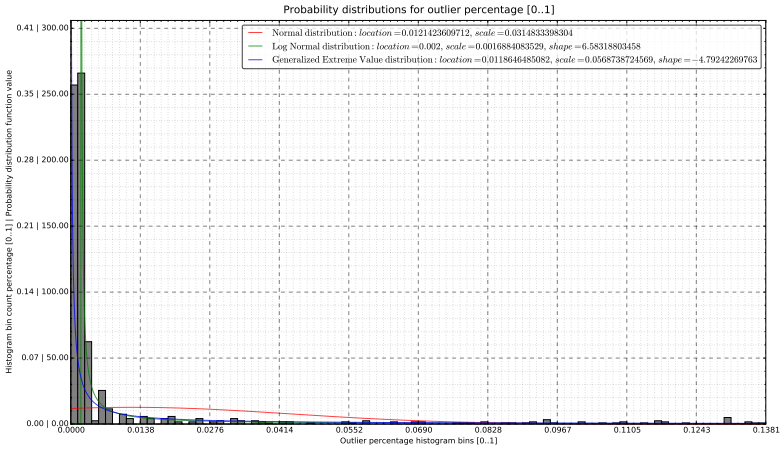


Figure B.1.2.28: Probability distributions for the ambient point cloud outlier percentage

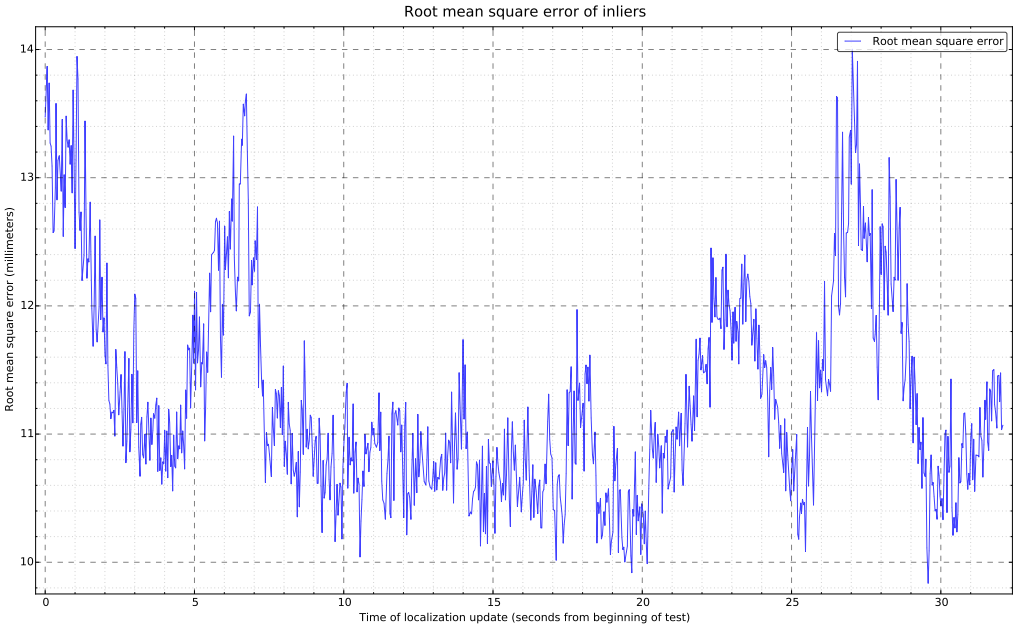


Figure B.1.2.29: Root Mean Square Error of the inliers

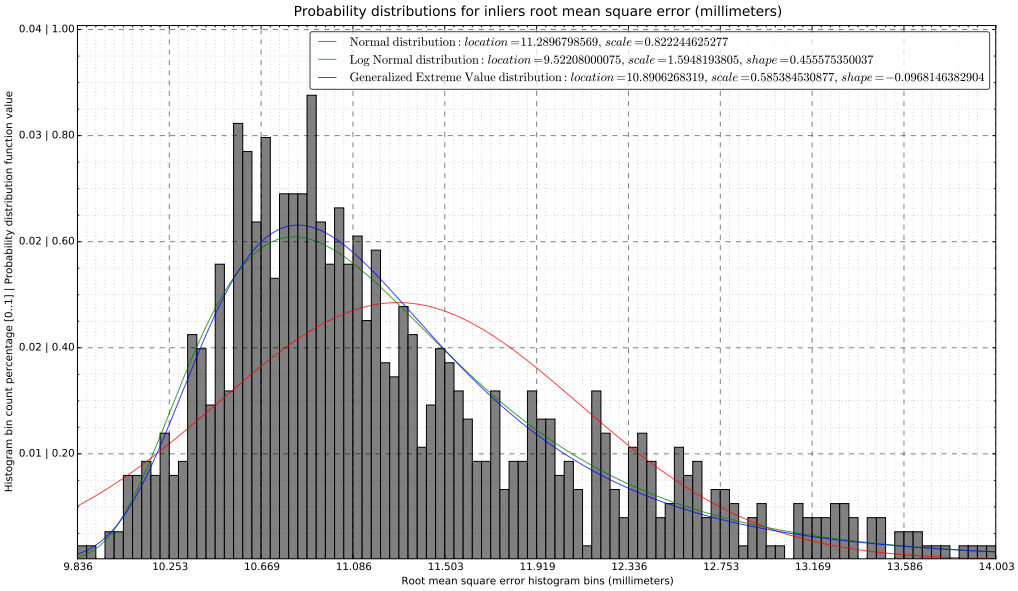


Figure B.1.2.30: Probability distributions for the Root Mean Square Error of the inliers

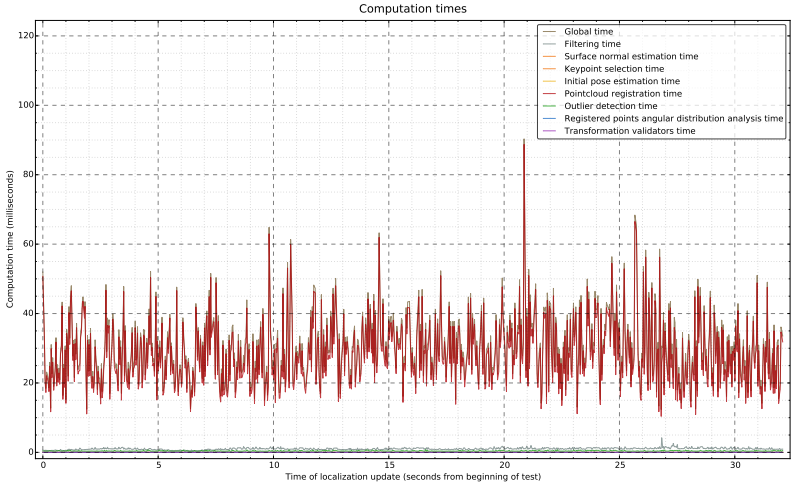


Figure B.1.2.31: Localization system computation times

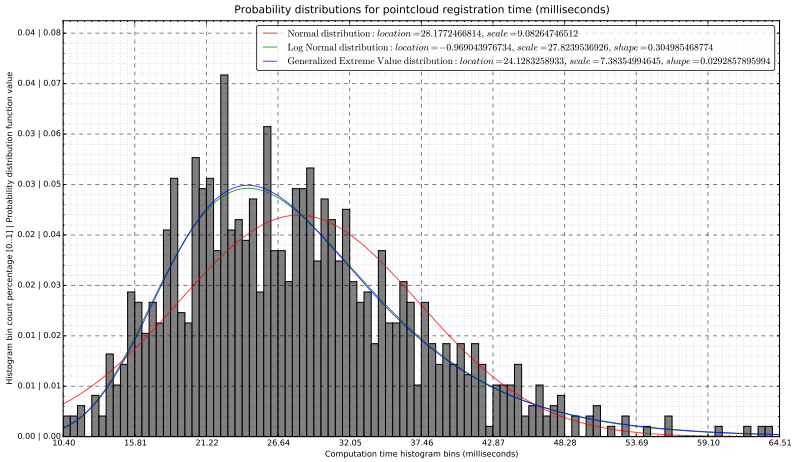


Figure B.1.2.32: Probability distributions for the point cloud registration time

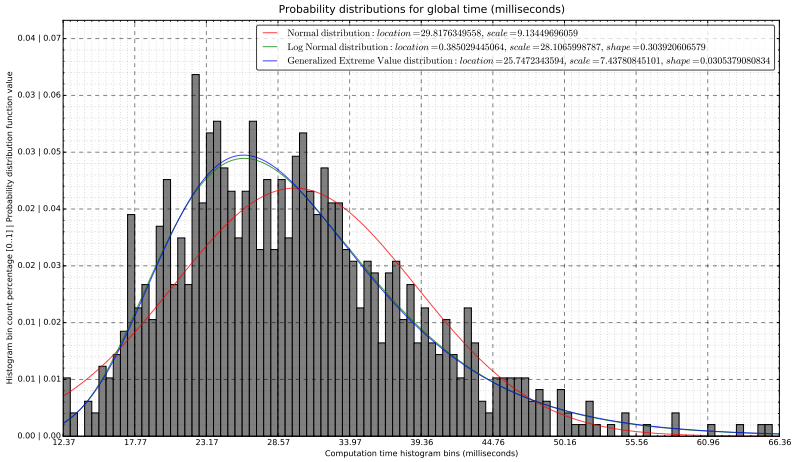


Figure B.1.2.33: Probability distributions for the localization system global computation time

B.1.3 Kinect with rotations motion at 10 cm/s

The figures in this section show the detailed results of test performed with the Kinect sensor with a 20 mm resolution map in the flying arena performing several rotations in the air and moving at 10 cm/s.

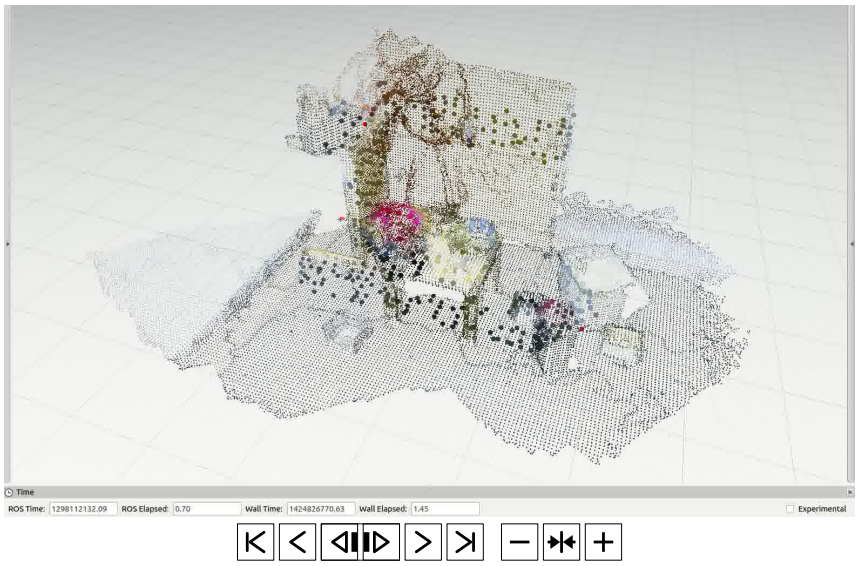


Figure B.1.3.1: Animation of point clouds using the localization system poses

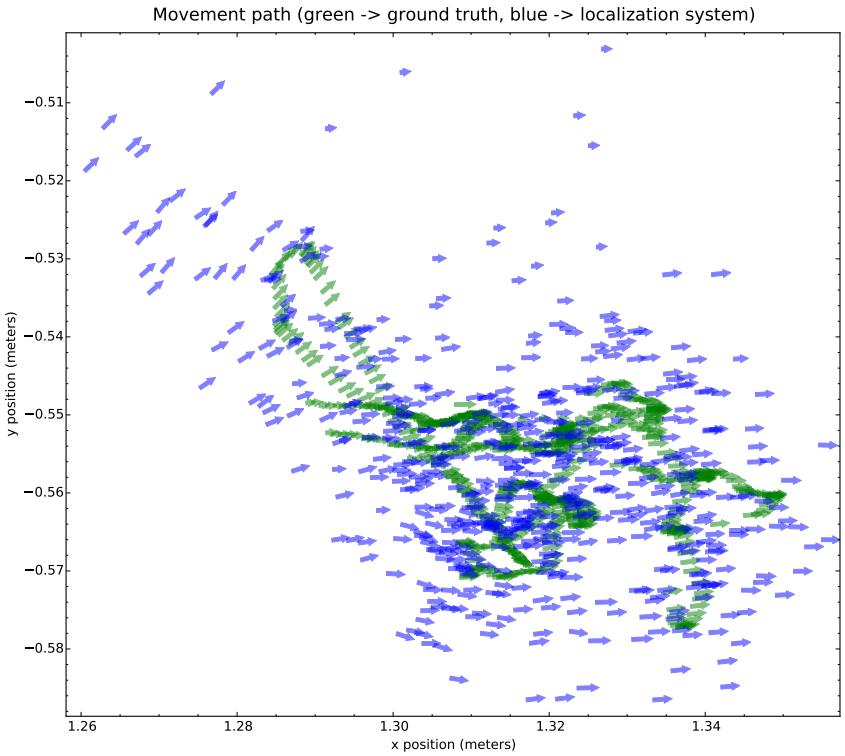


Figure B.1.3.2: Poses estimated by the ground truth and localization system

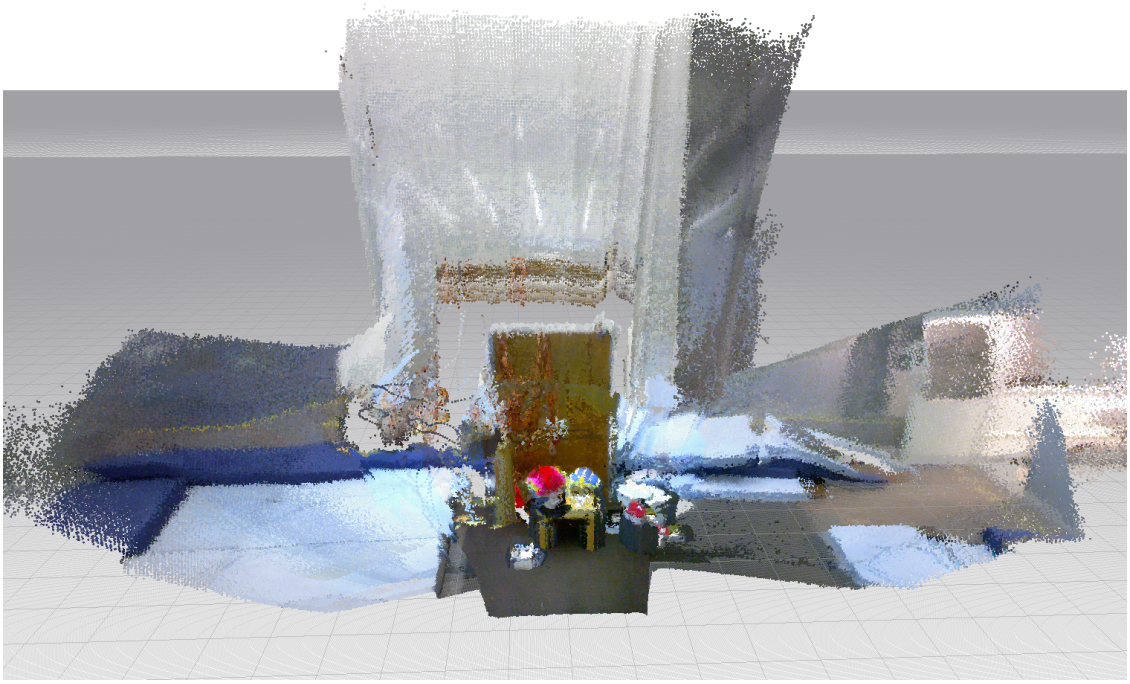


Figure B.1.3.3: Point clouds assembled on top of the map using the ground truth poses

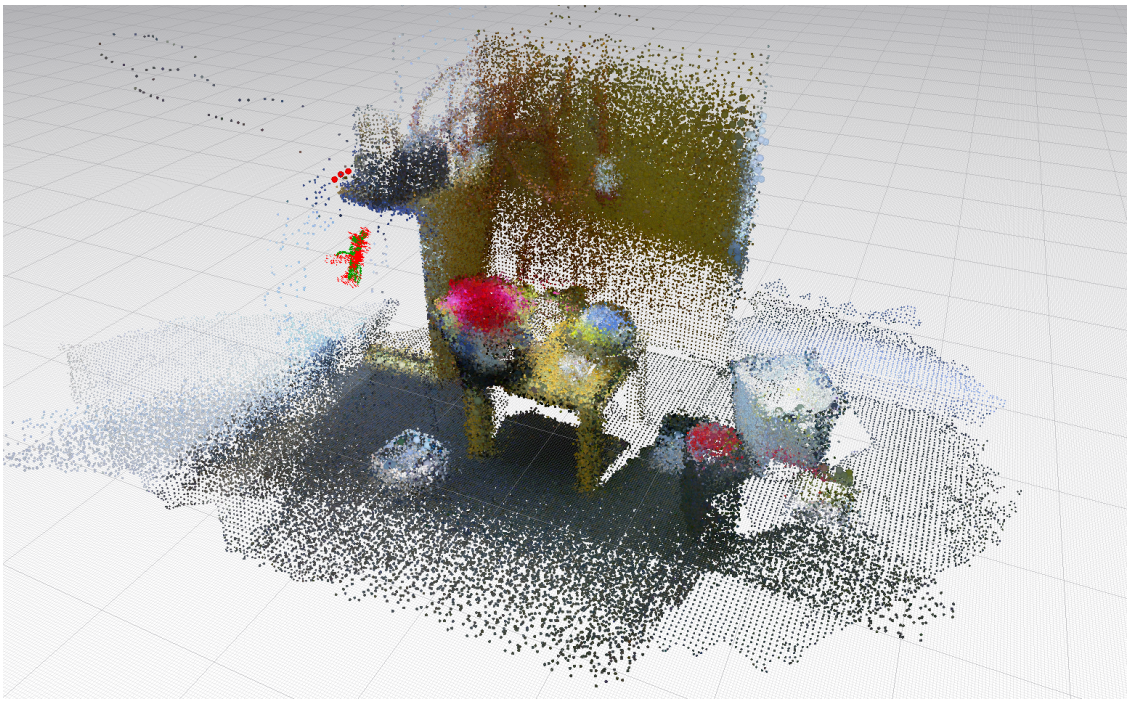


Figure B.1.3.4: Point clouds assembled on top of the map using the localization system poses

Figure B.1.3.5: Evolution of x position over time

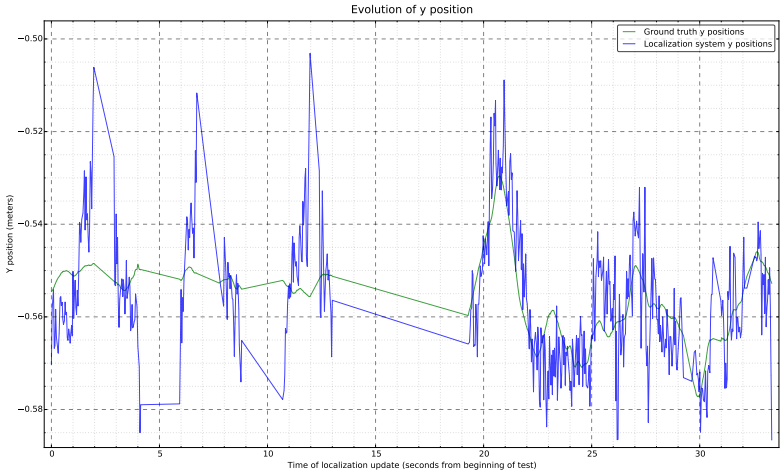


Figure B.1.3.6: Evolution of y position over time

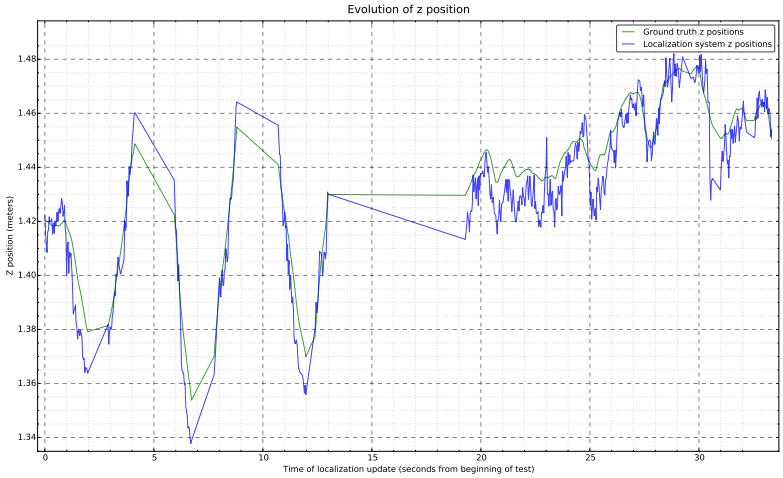


Figure B.1.3.7: Evolution of z position over time

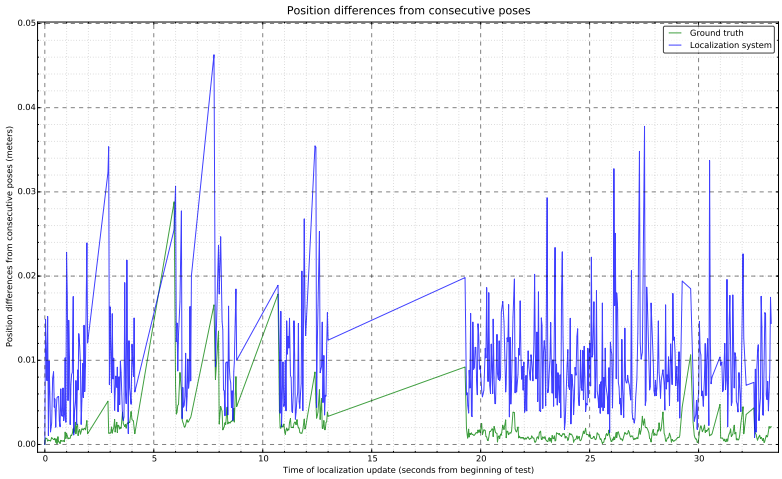


Figure B.1.3.8: Distance traveled between consecutive pose estimations

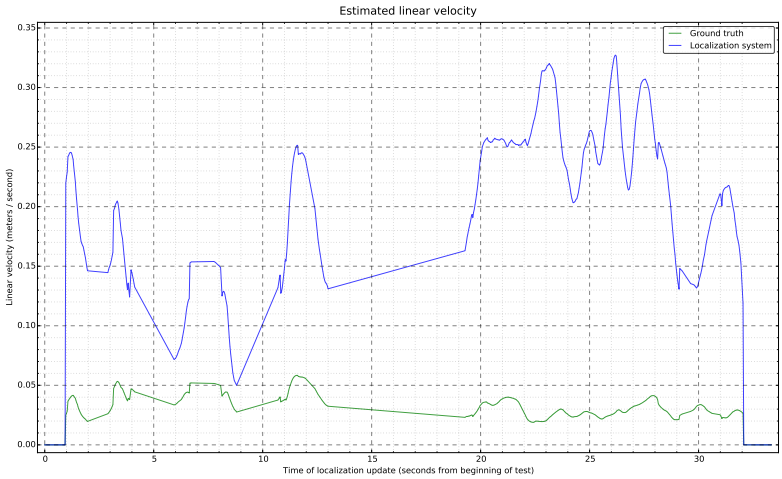


Figure B.1.3.9: Estimated linear velocity

Figure B.1.3.10: Estimated linear acceleration

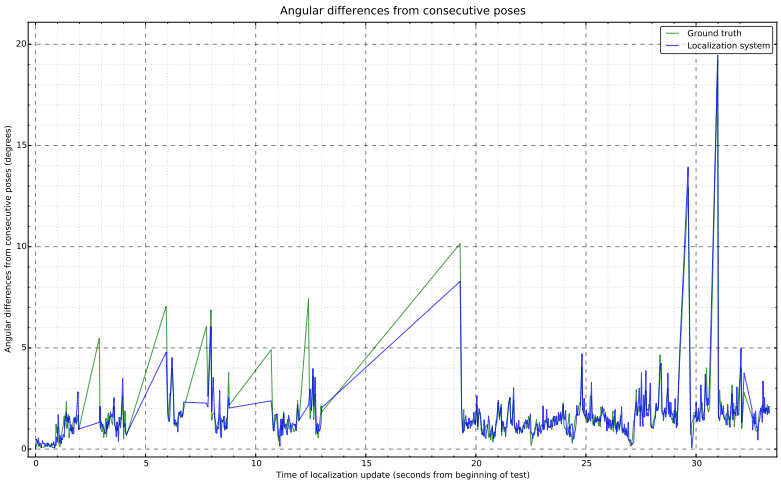


Figure B.1.3.11: Angular differences between consecutive pose estimations

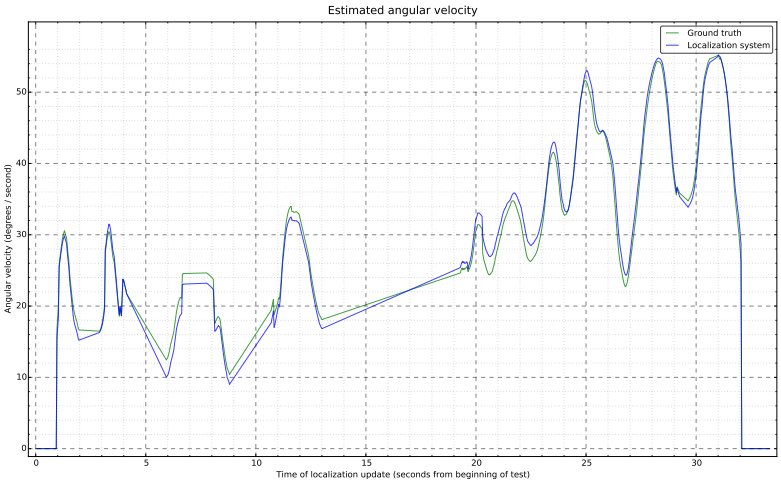


Figure B.1.3.12: Estimated angular velocity

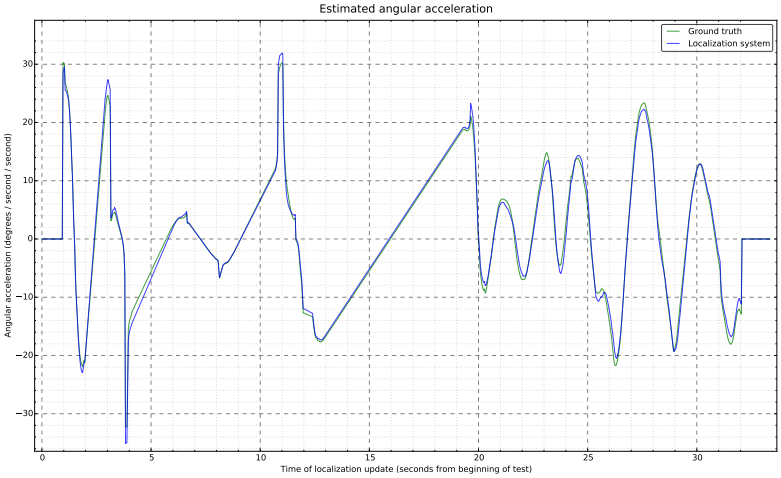


Figure B.1.3.13: Estimated angular acceleration

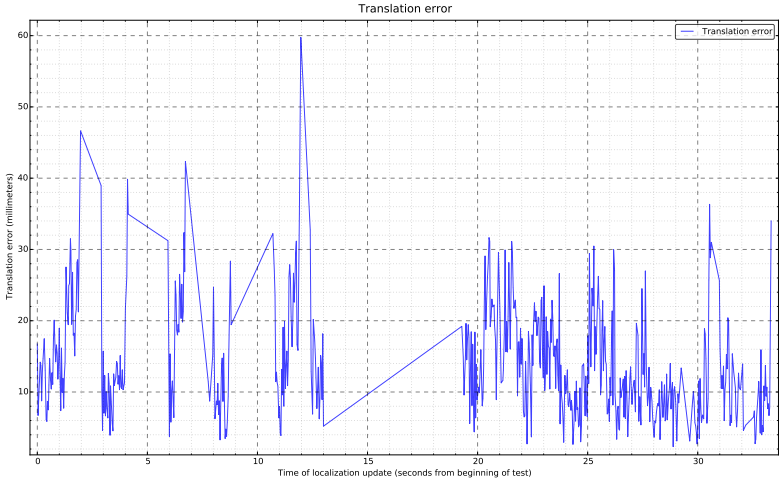


Figure B.1.3.14: Localization system translation errors

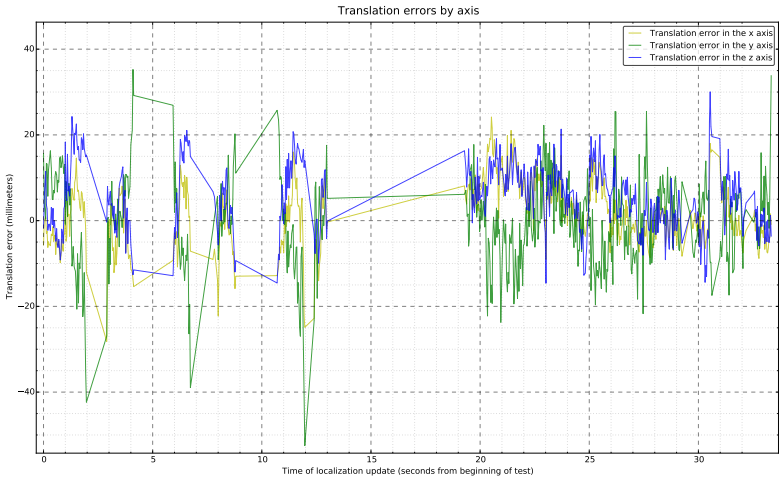


Figure B.1.3.15: Localization system translation errors components

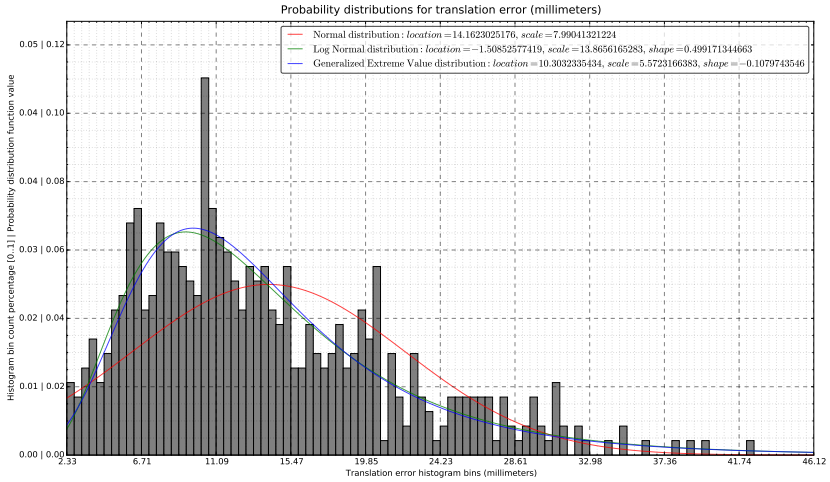


Figure B.1.3.16: Probability distributions for the localization system translation errors

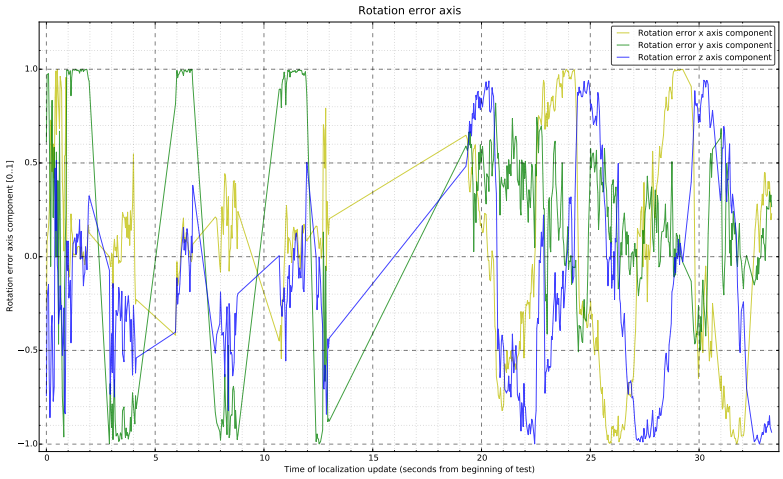


Figure B.1.3.17: Localization system rotation errors axis

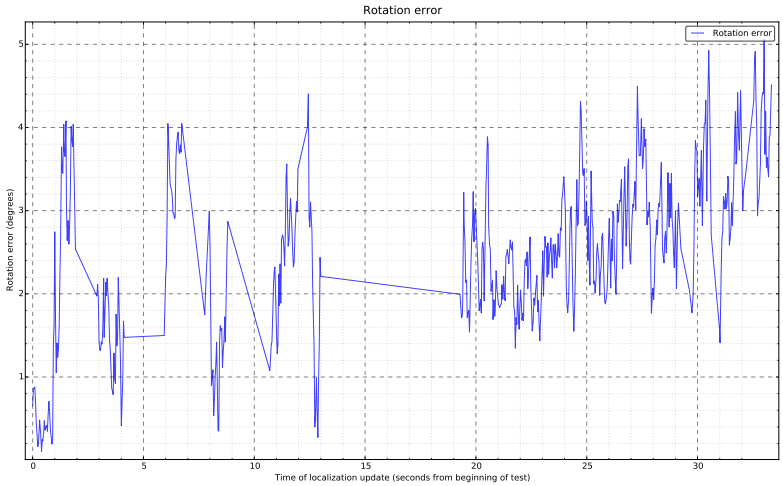


Figure B.1.3.18: Localization system rotation errors

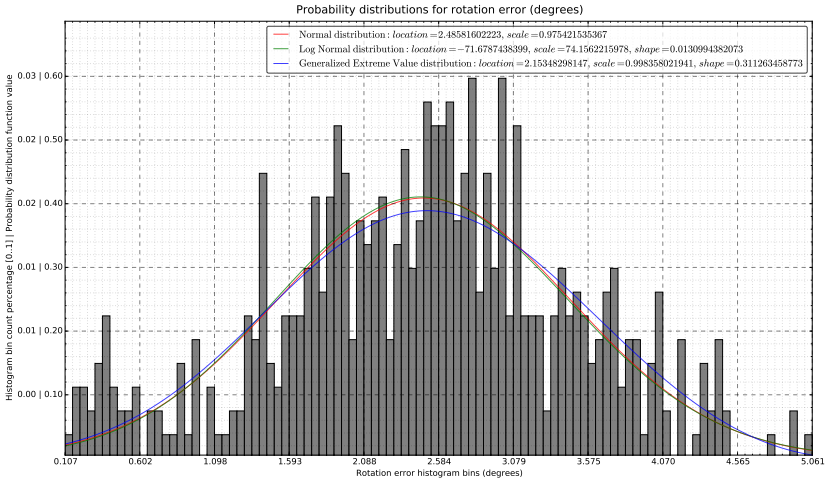


Figure B.1.3.19: Probability distributions for the localization system rotation errors

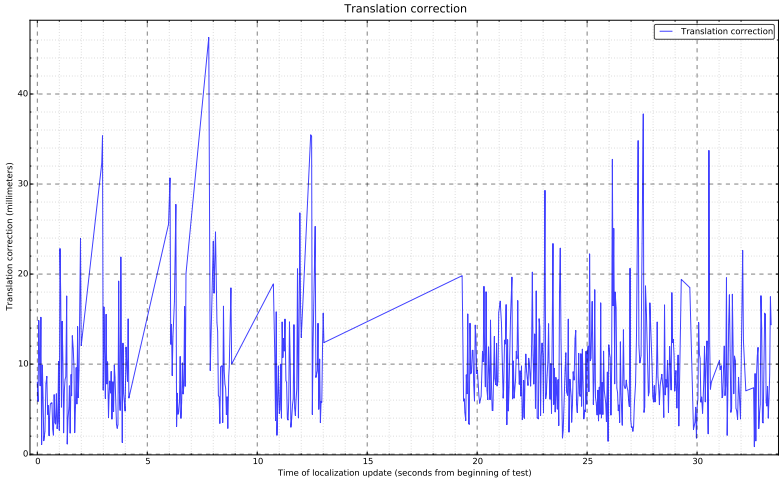


Figure B.1.3.20: Translation corrections performed by the localization system

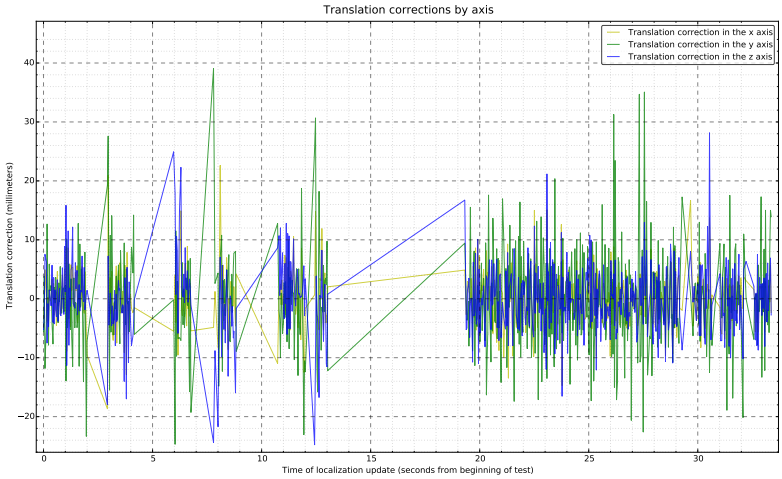


Figure B.1.3.21: Translation corrections components performed by the localization system

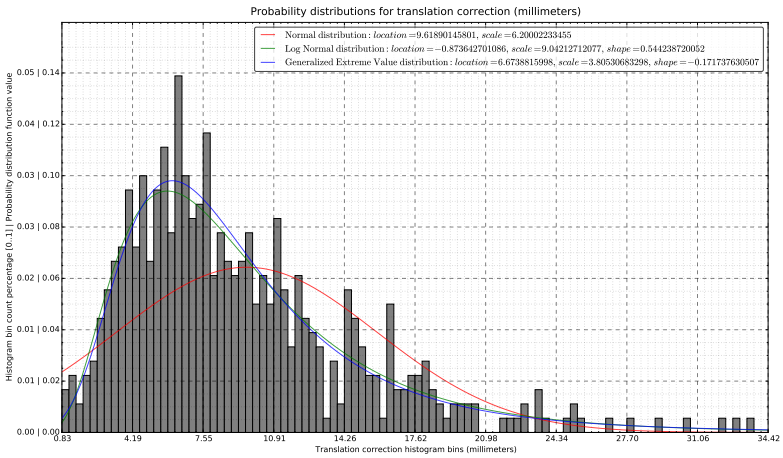


Figure B.1.3.22: Probability distributions for the translation corrections performed by the localization system

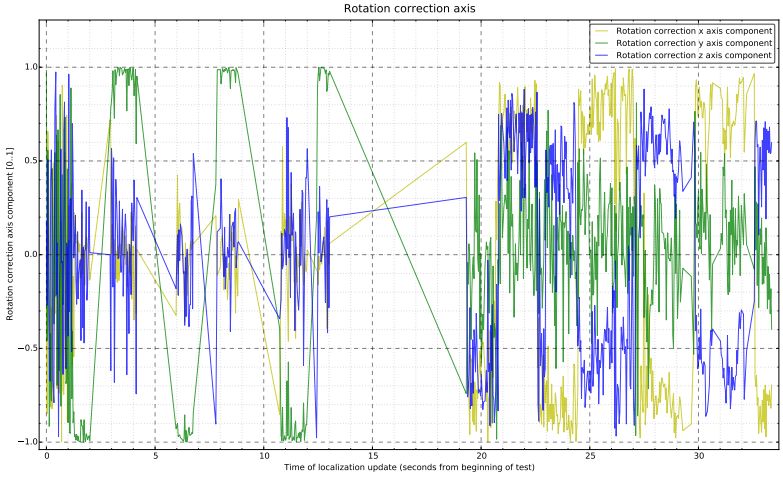


Figure B.1.3.23: Rotation corrections (axis) performed by the localization system

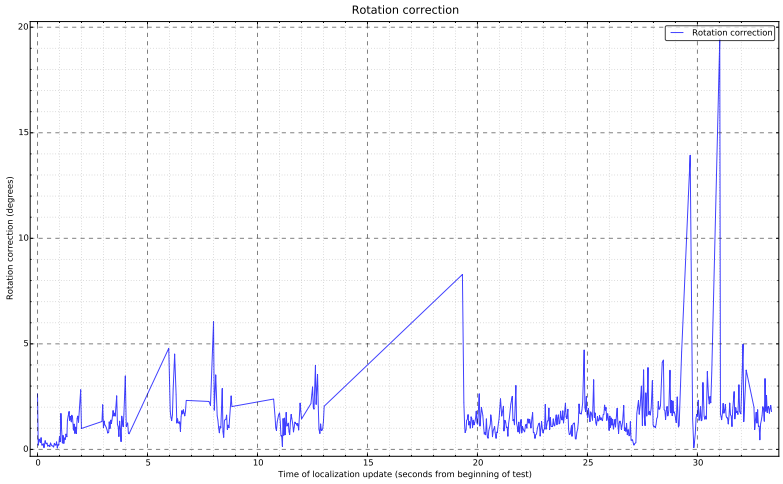


Figure B.1.3.24: Rotation corrections performed by the localization system

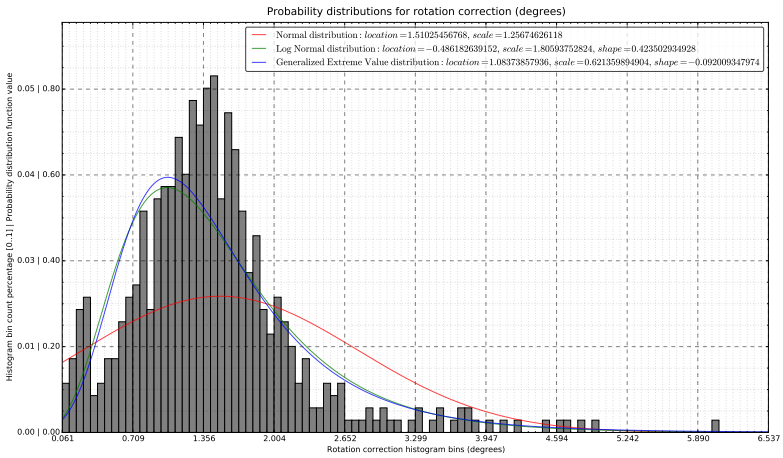


Figure B.1.3.25: Probability distributions for the rotation corrections performed by the localization system

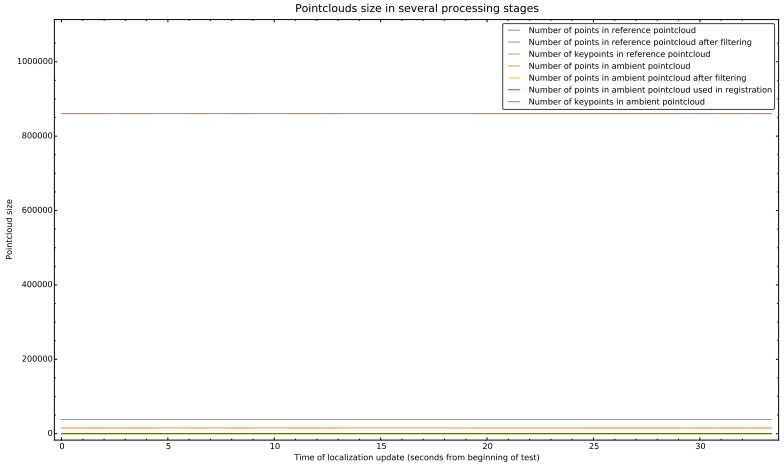


Figure B.1.3.26: Point clouds size in several of the localization system processing stages

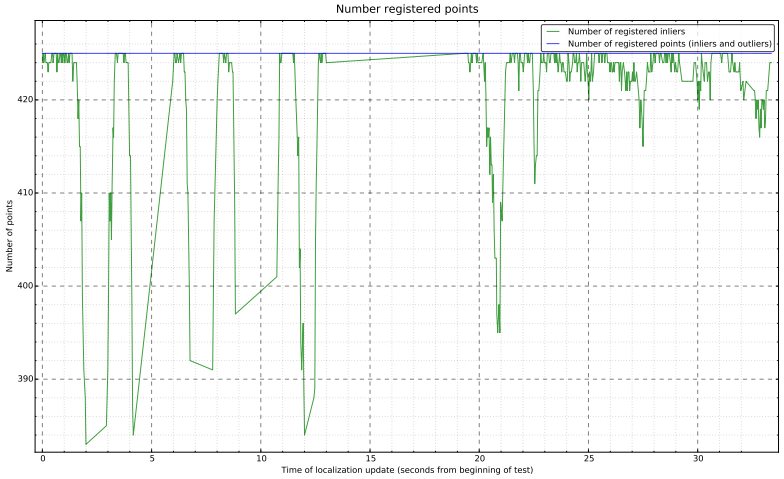


Figure B.1.3.27: Number of registered points

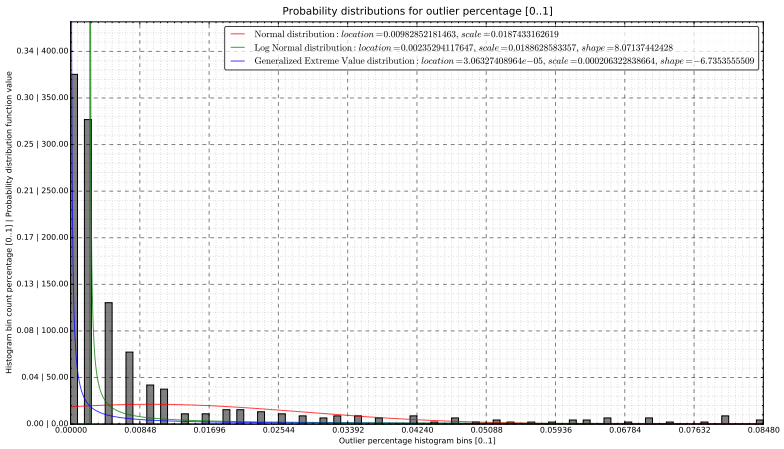


Figure B.1.3.28: Probability distributions for the ambient point cloud outlier percentage

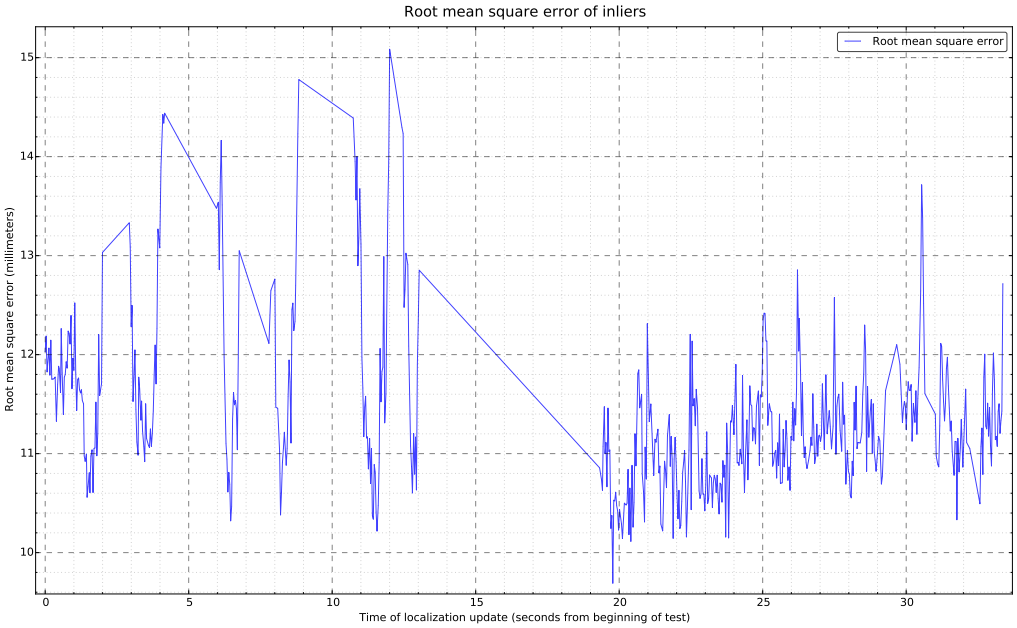


Figure B.1.3.29: Root Mean Square Error of the inliers

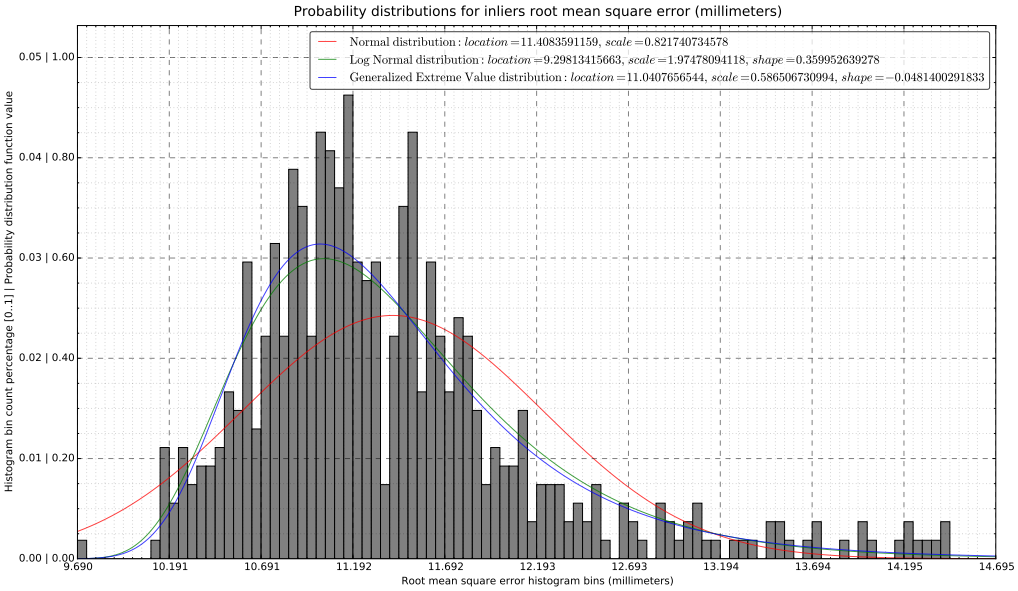


Figure B.1.3.30: Probability distributions for the Root Mean Square Error of the inliers

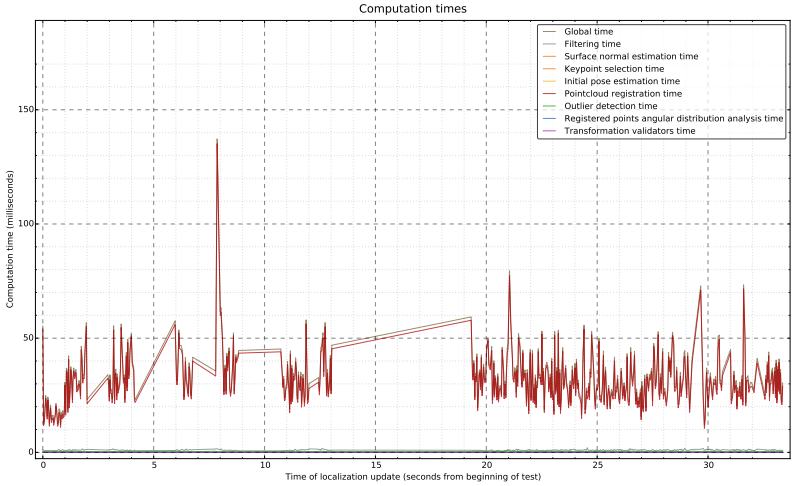


Figure B.1.3.31: Localization system computation times

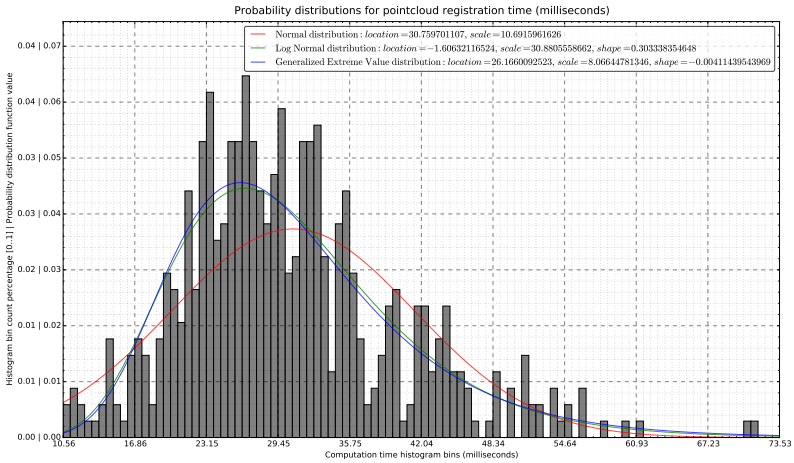


Figure B.1.3.32: Probability distributions for the point cloud registration time

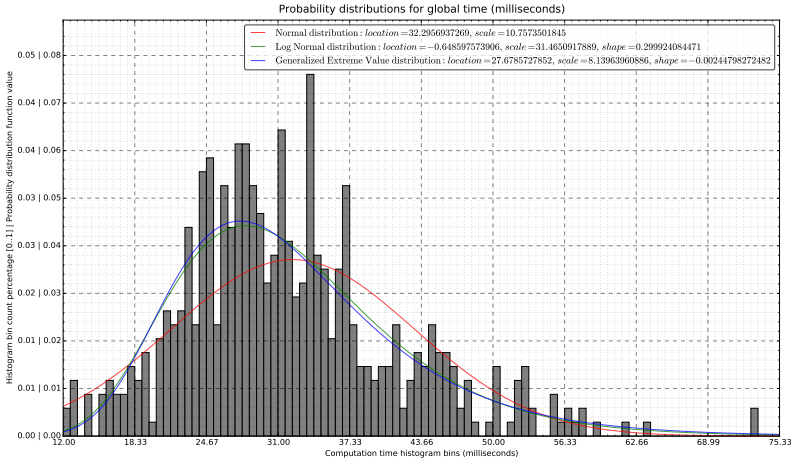


Figure B.1.3.33: Probability distributions for the localization system global computation time

References

[ABCO+03] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3–15, January 2003.

[AMGC02] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[BGFM10] J.-L. Blanco, J. Gonzalez, and J.-a. Fernandez-Madrigal. Optimal Filtering for Non-parametric Observation Models: Applications to Localization and SLAM. *The International Journal of Robotics Research*, 29(14):1726–1742, May 2010.

[BM92] Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[BOG+10] Mussa Bshara, Umut Orguner, Fredrik Gustafsson, Leo Van Biesen, and Student Member. Fingerprinting Localization in Wireless Networks Based on Received-Signal-Strength Measurements : A Case Study on WiMAX Networks Fingerprinting Localization in Wireless Networks Based on Received-Signal-Strength Measurements : A Case Study on WiMAX Networ. ... *IEEE Transactions on*, 59(1):283–294, 2010.

[BTP13] Sofien Bouaziz, Andrea Tagliasacchi, and Mark Pauly. Sparse iterative closest point. *Computer Graphics Forum*, 32(5):113–123, August 2013.

[Che03] Z H E Chen. Bayesian Filtering : From Kalman Filters to Particle Filters , and Beyond VI Sequential Monte Carlo Estimation : Particle Filters. *Statistics*, 182(1):1–69, 2003.

[CK02] Yongguang Chen Yongguang Chen and H. Kobayashi. Signal strength based indoor geolocation. *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333)*, 1(1):436–439, 2002.

[CSSK02] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek. The Trimmed Iterative Closest Point algorithm. *Object recognition supported by user interaction for service robots*, 3:545–548, 2002.

[DKU+07] Zhen Dai, Stefan Knedlik, Pakorn Ubolkosold, Junchuan Zhou, and Otmar Loffeld. Integrated GPS navigation for civilian vehicles in challenging environment. *2007 IEEE International Conference on Vehicular Electronics and Safety, ICVES*, pages 1–6, December 2007.

[DZA+13] M. Djehaich, H. Ziane, N. Achour, R. Tiar, and N. Ouadah. SLAM-ICP with a Boolean method applied on a car-like robot. *Proceedings of the 2013 11th International Symposium on Programming and Systems, ISPS 2013*, pages 116–121, 2013.

[EW99] Garry a. Einicke and Langford B. White. Robust extended Kalman filtering. *IEEE Transactions on Signal Processing*, 47(9):2596–2599, 1999.

[FA14] S. Filipe and L. a. Alexandre. A Comparative Evaluation of 3D Keypoint Detectors in a RGB-D Object Dataset. *9th International Conference on Computer Vision Theory and Applications*, 2014.

[FB81] Martin a Fischler and Robert C Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applicatlons to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381 – 395, 1981.

[FBDT99] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. *American Association for Artificial Intelligence*, (Handschin 1970):1–6, 1999.

[FHK+04] Andrea Frome, Daniel Huber, Ravi Kolluri, Thomas Bülow, and Jitendra Malik. Recognizing Objects in Range Data Using Regional Point Descriptors. *Computer Vision - ECCV 2004*, 3023:224–237, 2004.

[GIRL03] N. Gelfand, L. Ikemoto, S. Rusinkiewicz, and M. Levoy. Geometrically stable sampling for the ICP algorithm. *Fourth International Conference on 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings.*, pages 260–267, 2003.

[GSK12] Giorgio Grisetti, Hauke Strasdat, and Kurt Konolige. g 2 o: A general Framework for (Hyper) Graph Optimization. (1):1–19, 2012.

[GWG13] Francesco Guarato, James F C Windmill, and Anthony Gachagan. A new sonar localization strategy using receiver beam characteristics. *IEEE International Ultrasonics Symposium, IUS*, pages 445–448, July 2013.

[HWB+13] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, February 2013.

[Ibr10] Modar Ibraheem. Gyroscope-enhanced dead reckoning localization system for an intelligent walker. In *ICINA 2010 - 2010 International Conference on Information, Networking and Automation, Proceedings*, volume 1, pages 67–72, 2010.

[IKP10] Edouard Ivanjko, Andreja Kitanov, and I Petrovic. Model based Kalman Filter Mobile Robot Self-Localization. *Robot Localization and Map . . .* , (1996):59–91, 2010.

[Jez08] O Jez. Evaluation of a factorized ICP based method for 3D mapping and localization. *Atpjournal.Sk*, pages 29–37, 2008.

[Jol02] Jolliffe. *Principal component analysis*, volume 2. 2002.

[JU97] Simon J Julier and Je K Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. *System Identification*, 3:3–2, 1997.

[Kac13] Witold Kaczurba. FPGA-Based System Combines Two Video Streams to Provide 3D Video. *Analog Dialogue*, pages 1–5, 2013.

[Kal60] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(Series D):35, 1960.

[KKOO07] Yong Shik Kim, Bong Keun Kim, Kohtaro Ohba, and Akihisa Ohya. Bias estimation of DGPS multi-path data for localization of mobile robot. *Proceedings of the SICE Annual Conference*, pages 1501–1505, September 2007.

[KL04] Scale-invariant Keypoints and David G Lowe. Distinctive Image Features from. *International journal of computer vision*, 2004.

[Lab14] Mathieu Labb. Online Global Loop Closure Detection for Large-Scale Multi-Session Graph-Based SLAM. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

[LKP06] Sukhan Lee Sukhan Lee, Eunyoung Kim Eunyoung Kim, and Yeonchool Park Yeonchool Park. 3D object recognition using multiple features for robotic manipulation. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, (May):3768–3774, 2006.

[LLRM06] Martin Lauer, Sascha Lange, Martin Riedmiller, and Martin Riedmiller Martin Lauer, Sascha Lange. Calculating the perfect match: an efficient and accurate approach for robot self-localization. *Robocup 2005: Robot soccer world cup ...*, 4020(c):142–153, 2006.

[LYL11] Jianshen Liu, Baoyong Yin, and Xinxing Liao. Robot self-localization with optimized error minimizing for soccer contest. *Journal of Computers*, 6(7):1485–1492, July 2011.

[Mag09] Martin Magnusson. *The Three-Dimensional Normal-Distributions Transform — an Efficient Representation for Registration, Surface Analysis, and Loop Detection*. PhD thesis, Örebro University, 2009.

[MCB10] Julien Moras, Véronique Cherfaoui, and Phillipe Bonnifait. A lidar perception scheme for intelligent vehicle navigation. In *11th International Conference on Control, Automation, Robotics and Vision, ICARCV 2010*, number December, pages 1809–1814, 2010.

[PM63] Miguel Pinto and Ap Moreira. Novel 3D matching self-localization algorithm. *International Journal of ...*, 5(1):1–12, 1963.

[PMC+11] François Pomerleau, Stéphane Magnenat, Francis Colas, Ming Liu, and Roland Siegwart. Tracking a depth camera: Parameter exploration for fast ICP. *IEEE International Conference on Intelligent Robots and Systems*, pages 3824–3829, September 2011.

[Pom13] François Pomerleau. *Methodology and Tools for ICP-like Algorithms*. PhD thesis, ETH Zürich, 2013.

[QCG+09] Morgan Quigley, Ken Conley, Brian Gerkey, Josh FAust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Mg. ROS: an open-source Robot Operating System. *Icra*, 3:5, 2009.

[R.48] W. R. Introduction to Wireless. *Nature*, 162:911–911, April 1948.

[RBB09] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast Point Feature Histograms (FPFH) for 3D registration. *2009 IEEE International Conference on Robotics and Automation*, pages 3212–3217, May 2009.

[RBMB08] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pages 3384–3391, September 2008.

[RC11] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1–4, May 2011.

[Rib04] Maria Isabel Ribeiro. Kalman and Extended Kalman Filters : Concept , Derivation and Properties. *Institute for Systems and Robotics Lisboa Portugal*, (February):42, 2004.

[RKZ13] Michal Reinstein, Vladimir Kubelka, and Karel Zimmermann. Terrain adaptive odometry for mobile skid-steer robots. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 4706–4711, 2013.

[RL01] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on. IEEE*, pages 145–152. IEEE Comput. Soc, 2001.

[Rus10] Radu Bogdan Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Technische Universität München, August 2010.

[San13] Filipe Baptista Neves dos Santos Santos. *A collaborative, non-invasive hybrid semantic localization and mapping system*. PhD thesis, 2013.

[SEE+12] Jrgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. *IEEE International Conference on Intelligent Robots and Systems*, pages 573–580, October 2012.

[SHT09] a Segal, D Haehnel, and S Thrun. Generalized-ICP. *Robotics: Science and Systems*, 2009.

[Sot06] S Sotoodeh. Outlier Detection in Laser Scanner Point Clouds. *Iaprs*, (September):297–302, 2006.

[SS09] Ptm Saito and Rj Sabatine. An analysis of parallel approaches for a mobile robotic self-localization algorithm. *International Journal of Future Generation Comunication and Networking*, 2(4):49–64, 2009.

[STD09] Giovanna Sansoni, Marco Trebeschi, and Franco Docchio. State-of-the-art and applications of 3D imaging sensors in industry, cultural heritage, medicine, and criminal investigation. *Sensors*, 9:568–601, 2009.

[TARK10] Toru Tamaki, Miho Abe, Bisser Raytchev, and Kazufumi Kaneda. Softassign and EM-ICP on GPU. *Proceedings - 2010 1st International Conference on Networking and Computing, ICNC 2010*, pages 179–183, November 2010.

[TGCV12] Gurkan Tuna, Kayhan Gulez, V. Cagri Gungor, and T. Veli Mumcu. Evaluations of different Simultaneous Localization and Mapping (SLAM) algorithms. *IECON Proceedings (Industrial Electronics Conference)*, pages 2693–2698, 2012.

[Thr02] Sebastian Thrun. *Probabilistic robotics*, volume 45. The MIT Press, 2002.

[TSD10] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique shape context for 3d data description. *Proceedings of the ACM workshop on 3D object retrieval - 3DOR '10*, page 57, 2010.

[TSD11] Federico Tombari, Samuele Salti, and Luigi Di Stefano. A combined texture-shape descriptor for enhanced 3D feature matching. *Proceedings - International Conference on Image Processing, ICIP*, pages 809–812, 2011.

[Vit84] Jeffrey Scott Vitter. Faster methods for random sampling. *Communications of the ACM*, 27(7):703–718, 1984.

[WM00] E.a. Wan and R. Van Der Merwe. The unscented Kalman filter for nonlinear estimation. *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, 7:153–158, 2000.

[WV11] Walter Wohlkinger and Markus Vincze. Ensemble of shape functions for 3D object classification. *2011 IEEE International Conference on Robotics and Biomimetics*, pages 2987–2992, December 2011.

[WY07] Shun-xi Wu Shun-xi Wu and Ming Yang Ming Yang. Landmark Pair based Localization for Intelligent Vehicles using Laser Radar. *2007 IEEE Intelligent Vehicles Symposium*, pages 209–214, June 2007.

[WYY+13] Bin Wu, Bailang Yu, Wenhui Yue, Song Shu, Wenqi Tan, Chunling Hu, Yan Huang, Jianping Wu, and Hongxing Liu. A voxel-based method for automated identification and morphological parameters estimation of individual street trees from mobile laser scanning data. *Remote Sensing*, 5(2):584–611, January 2013.

[YPH+14] Hong-seok Yang, Ji-man Park, Jung-suk Han, Jai-bong Lee, Sung-hun Kim, and Insung Yeo. Measuring abutment convergence angles using stereovision dental image processing system. pages 259–265, 2014.

[Yu09] Zhong Yu. Intrinsic shape signatures: A shape descriptor for 3D object recognition. *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops 2009*, pages 689–696, September 2009.

[YXL11] Peng Yang, Li Xie, and Jilin Liu. Angular velocity current statistical model based real-time celestial navigation for lunar rover. *Proceedings of the 30th Chinese Control Conference*, pages 4011–4016, 2011.

[Zha94] Z Zhang. Iterative point matching for registration of free form curves and surfaces. *International Journal of Computer Vision*, 12:119–152, 1994.

[ZL11] Jia Heng Zhou and Huei Yung Lin. A self-localization and path planning technique for mobile robot navigation. *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, pages 694–699, 2011.

[ZMH10] Yang Zhang Yang Zhang, N. Meratnia, and P. Havinga. Outlier Detection Techniques for Wireless Sensor Networks: A Survey. *IEEE Communications Surveys & Tutorials*, 12:159–170, 2010.