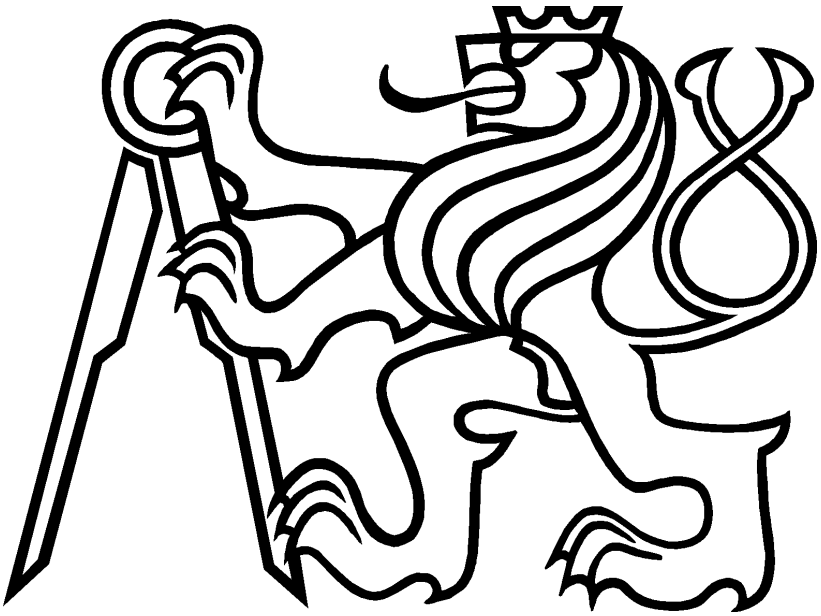
CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering



Jan Vakula

Keypoint localization and matching in difficult scenes for visual odometry

Cybernetic and Robotic

Thesis supervisor: **doc. Ing. Tomáš Svoboda, PhD**

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Bc. Jan V a k u l a

Studijní program: Kybernetika a robotika (magisterský)

Obor: Robotika

Název tématu: Lokalizace klíčových bodů a párování v obtížných scénách pro vizuální odometrii

Pokyny pro vypracování:

Pro robustnost výpočtu pozice robotu z obrazů je rozhodující stabilita lokalizace a párování klíčových bodů v okamžiku, kdy se podoba scény výrazně mění (rotace robota) nebo je obraz poškozen (např. rozmazání). Stávající implementace [1] v těchto momentech selhává. Navrhněte vylepšení existujícího algoritmu . Věnujte zvýšenou pozornost automatické detekci selhání algoritmu. Implementuje navrhnuté řešení jako modul pro Robotic Operating System (ROS). Otestujte a nalezněte meze funkčnosti na reálných datech z různých scén.

Seznam odborné literatury:

[1] Divis, Jiri (2013), 'Visual Odometry from Omnidirectional Camera', Master's thesis, Charles University in Prague, Faculty of Mathematics and Physics.
[2] Jesus, F. & Ventura, R. (2012), Combining monocular and stereo vision in 6D-SLAM for the localization of a tracked wheel robot, in 'Safety, Security, and Rescue Robotics (SSRR), 2012 IEEE International Symposium on'.
[3] Kundu, A.; Krishna, K. M. & Jawahar, C. V. (2011), Realtime Multibody Visual SLAM with a Smoothly Moving Monocular Camera, in 'Computer Vision, 2011. ICCV 2011. IEEE International Conference on'.
[4] Rublee, E.; Rabaud, V.; Konolige, K. & Bradski, G. (2011), ORB: An efficient alternative to SIFT or SURF, in 'Computer Vision (ICCV), 2011 IEEE International Conference on', pp. 2564 -2571.

Vedoucí diplomové práce: doc. Ing. Tomáš Svoboda, Ph.D.

Platnost zadání: do konce letního semestru 2014/2015

L.S.

doc. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 16. 12. 2013

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

DIPLOMA THESIS ASSIGNMENT

Student: Bc. Jan V a k u l a

Study programme: Cybernetics and Robotics

Specialisation: Robotics

Title of Diploma Thesis: Keypoint Localization and Matching in Difficult Scenes for Visual Odometry

Guidelines:

Robustness of keypoint localization, matching and management is a crucial factor in the problem of visual odometry - estimation of robot position from images. The typical difficult situations include sudden changes in scene appearance in case of robot rotation and motion blur. The current implementation [1] is often failing under such circumstances. Propose an improvement and partilarly focus on the problem of automatic failure detection. Implement the propose solution as a node within the ROS (Robot Operating System). Verify the proposed solution on the difficult real-scene data and discuss the possible limitations.

Bibliography/Sources:

[1] Divis, Jiri (2013), 'Visual Odometry from Omnidirectional Camera', Master's thesis, Charles University in Prague, Faculty of Mathematics and Physics.
[2] Jesus, F. & Ventura, R. (2012), Combining monocular and stereo vision in 6D-SLAM for the localization of a tracked wheel robot, in 'Safety, Security, and Rescue Robotics (SSRR), 2012 IEEE International Symposium on'.
[3] Kundu, A.; Krishna, K. M. & Jawahar, C. V. (2011), Realtime Multibody Visual SLAM with a Smoothly Moving Monocular Camera, in 'Computer Vision, 2011. ICCV 2011. IEEE International Conference on'.
[4] Rublee, E.; Rabaud, V.; Konolige, K. & Bradski, G. (2011), ORB: An efficient alternative to SIFT or SURF, in 'Computer Vision (ICCV), 2011 IEEE International Conference on', pp. 2564 -2571.

Diploma Thesis Supervisor: doc. Ing. Tomáš Svoboda, Ph.D.

Valid until: the end of the summer semester of academic year 2014/2015

L.S.

doc. Dr. Ing. Jan Kybic
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, December 16, 2013

Declaration

I hereby declare that I have completed this thesis independently and that I have listed all used information sources in accordance with Methodical instruction about ethical principles in the preparation of university theses.

In Prague on..... .....

Acknowledgements

I would like to thank my supervisor doc. Ing. Tomáš Svoboda, PhD for his support and great advice during this thesis. Many thanks to Ing. Vladimír Kubelka and Ing. Tomáš Petříček for their assistance with the setup of the development environment. I would also like to thank my girlfriend Mgr. Alexandra Karasová for her support and encouragement.

Abstrakt

V této práci jsme otestovali a vylepšili algotritmus založený na práci Jiřího Diviše. Původní algoritmus byl nejprve důkladně otestován a po stanovení hlavních nedostatků také vylepšen. Největší zlepšení se dosáhlo vylepšením párování tzv. key-points. Po úpravách jsme algoritmus otestovali v různých typech prostředí, do kterých se USAR robot může dostat. Nejlépe si algoritmus vedl ve venkovním prostředí, kde bylo ve scéně dostatečné množství výrazných hran. V místnosti jsme narazili na problém s nedostatkem osvětlení, tím došlo ke sbírání rozmazaných obrázků. Proto jsme do algoritmu přidali detekci rozmazaných obrázků, na jejímž základě byly rozmazné obrázky smazány. Otestováním originálního a upraveného algoritmu bylo zjištěno výrazné zlepšení jeho chování.

Abstract

In this thesis we tested and improved an algorithm developed by Jiri Divis. Initially, the previous algorithm was carefully tested and after assessment the main imperfections also improved. The biggest improvements was achieved by a pairing improving so-called key-points. After the modifications we tested the algorithm in various types of environments in which the USAR robot can get. The best results of the algorithm was observed in an outdoor environments where there was sufficient amount of a visible edges in the scene. In an indoor environments we encountered a problem with the lack of lighting, this has led to recording of blurry images. Therefore we added a detection of blurry images to the algorithm under which the blurry images were deleted. By the testing of the previous and the improved algorithms was found out a significant improvement in its behaviour.

Contents

List of Figures

List of Tables

1. Introduction

One of a fundamental function for an autonomous robot is an odometry. It is a method which uses the data from robot sensors to estimate changes in position and rotation over the time. The robot normally has a lot of sensors for measuring its position and rotation, for example:

- rotary encoders on its wheels
- accelerometers, gyroscopes
- cameras, laser range-finders
- GPS

In this thesis I will write about a visual odometry which is process of determining an equivalent odometry information using only sequential camera images[8]. A few types of visual odometry are known, depend on number of cameras camera: monocular (uses only one camera) and stereo (uses two cameras), and we will occupy with monocular omnidirectional visual odometry. This type of visual odometry is crucial for us because we will use only one omnidirectional camera Ladybug3, [12]. The visual odometry algorithm itself is designed to be independent of other sensors and uses only the camera. The ladybug3 is rigidly attached to the robot's body. Because the robot is low, the camera is also placed quite low (approximately 300mm). Therefore the camera view is limited.

1.1 Ladybug 3

The Ladybug3 (Figure 1.1) is a spherical digital video camera system, which has six 2 MP cameras. For our system, the camera looks like would provide only a single composite image. These cameras enable the system to collect video from more than 80% of the full 360° sphere, and an FireWire interface with locking screw connection, that allows JPEG-compressed 12MP resolution images to be streamed to disk at 15 fps. The resolution of streamed images is $1600 \times 800px$. In our application the streamed speed is about 2-3 fps because of hardware limitations of the robot. However, it is not limitation for our algorithm because it is able to process about 2 fps as well.

Figure 1.1: Image of the Ladybug3 camera.

The motivation is to develop robust and precise algorithm of visual odometry which estimates position and rotation of the robot using a camera. The pose, which is combination of position and orientation relative to some coordinate system, could used in INSO. INSO, see in section 1.2 [15], contains Kalman filter. It fuses all data from odometry sensors because the pose estimation is then more accurate than from individual sensor. It can also be used in localization or in SLAM eventually.

1.2 Inertial Navigation System / Odometry

The inertial navigation system aided by odometry (INSO) which we use for comparison fuses inertial and odometry measurements for estimating position and orientation in the space. The inertial measurements - acceleration and angular acceleration are measured by a inertial measurement unit (IMU). The fusion is realized using the extended Kalman filter with a non-linear model with a complementary filter for attitude estimation. This dead reckoning reach 0.8 m RMSE and average of 4% return position error [15]. This accuracy is sufficient for us and we will use INSO in the experiments for comparison.

The algorithm will be tested on a NIFTi robot platform.

1.3 NIFTi robot

The NIFTi robot is skid-steer mobile robot, which can be used for Urban search and rescue (USAR) missions. Robot is equipped with the Point Grey Lady-bug 3 omnicamera from which a spherical projection of the robot's surroundings can be obtained. This robot is shown in Figure 1.2. The position of its camera is not in centre of robot's rotation. Thanks to this the camera is moving although the robot is only rotating. Field of camera's vision is 360° in horizontal plane but disadvantage of the camera is that in vertical plane is 1/3 of view obscured by the robot. The example of spherical image from robot's camera will be shown in next sections.

1.4 ROS

All developed algorithm is implemented in Robotic operating system(ROS)[13]. It is a set of software libraries and tools that help you build the robot applications, moreover all ROS is an open source. The algorithm had been developed for a Fuerte version, but during writing this thesis has been migrated to an Indigo version that also running on NIFTi.

The aim of my Master thesis is to understand, test and improve the original algorithm of visual odometry for mobile robot which was created by Jiri Divis [4]. The existing algorithm was tested inadequately. The algorithm has low stability, sometimes terminates automatically and randomly lost all of the landmarks. The presenting results shows that it is not function at all. It was necessary to identify the main causes of disappointing performance of this algorithm.

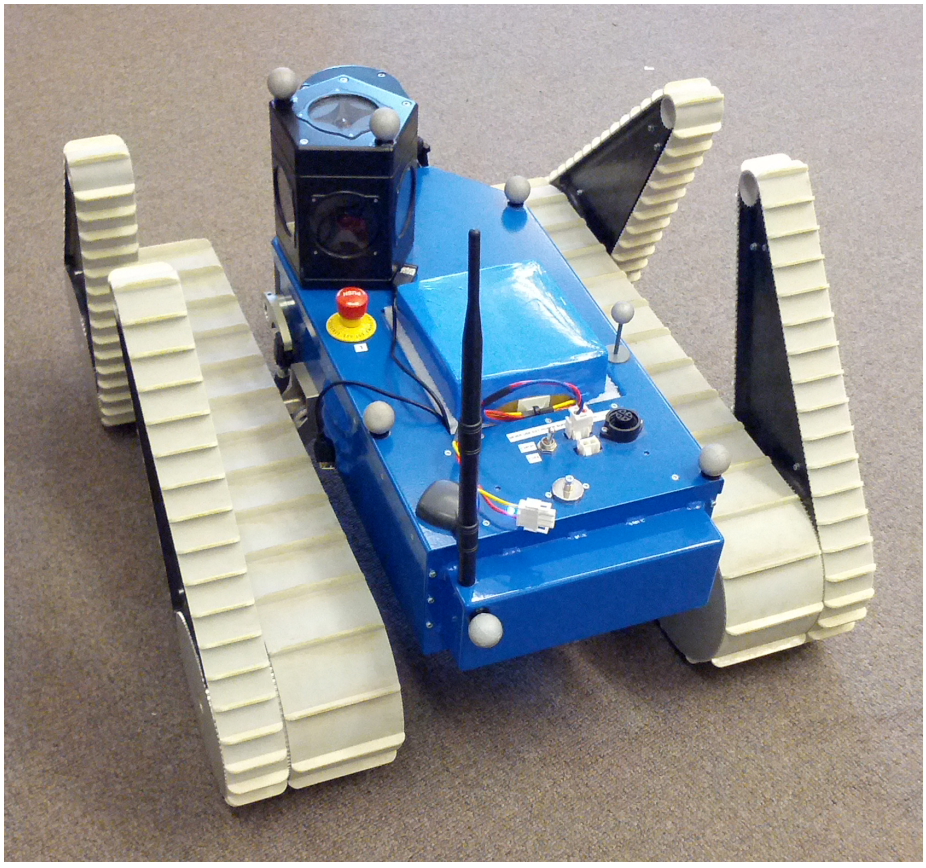


Figure 1.2: In the picture is the NIFTi robot developed for USAR missions, which is located in the ETH in Zürich. At the top front is the black ladybug3 camera. CTU Prag has the camera in black color.

2. Visual odometry

In this section we will discuss an existing algorithm of visual odometry. We will describe important parts of the algorithm as proposed by original author [4].This can help us to understand how the algorithm works and where could be the problems. In view of the fact that the original algorithm was not experimentally evaluated, the new experiments had to be necessarily made. The experiments shows strengths and weaknesses of that program. After this analysis can be proposed and implemented the improvements.

2.1 Feature detector

The feature detection is a method which finds interesting points in the picture. These points are named as keypoints. For our implementation is used ORB which is oriented FAST detector and BRIEF descriptor [14] [11]. This step is very important in the algorithm because the keypoint must be detected invariantly of its observation to rigid body transform. During the experiments was verified that the ORB is good for our application. The ORB is set to find about 1000 keypoints in one image. After finding the points non-maxima suppression (NMS) [7] is applied. After that there are about 420 keypoints remaining in the image. It all depends on type of scene. The example of founding the keypoints is shown in Figure 2.1.

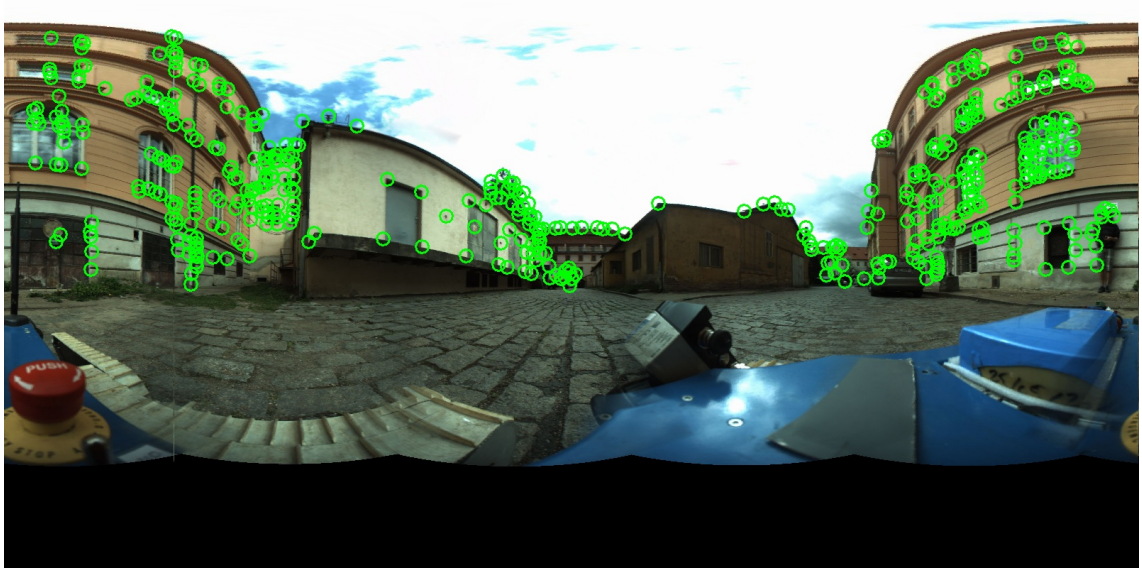


Figure 2.1: The example of founding the keypoints (green circles). NMS is set to 5 pixel.

2.1.1 Non-maxima suppression

The non-Maximum Suppression (NMS) can be definitely formulated as Local Maximum Search, where a local maxi-mum is greater than all its neighbours [7]. Easily, the "weak" points, which are too close to the "strong" point are removed. The strength of the points is measured as Harris corner response/score [5]. The non-maxima suppression decreases the number of detected points and thus the algorithm counts with less points and runs faster.

2.2 Unguided matching

An unguided matching is a method which finds corresponding keypoints in two images. After finding the correspondences, the rigid body transform is estimated between two camera pose. The rigid model contains rotation and translation between camera pose. The translation is computed without scale, this will be further explained. The correspondences between images is computed from a whole images, therefore it is very slowly. For matching is used knn-match and the distance between features must be the nearest to each other.

For example, in one of our experiments 3.1 there are averagely 121 symmetrical matches between two images where 307 keypoints are detected in each image. However, sometimes there are cases where only 50 symmetrical matches were detected. This is relatively low number of matches. For this reason a guided matching could be implemented which allows to increase the matching number.

2.3 Guided matching

The guided matching is designed to increase number of matches between two images. The algorithm is shown in algorithm 1.

Algorithm 1 Guided matching algorithm

1: divide image $I2$ into uniform grid
2: **for all** keypoint $k \in$ in image $I1$ **do**
3: transform k by BRT rotation calculated in unguided matching
4: select cell $c_{i,j}$ where k is transformed
5: get all keypoints from image 2 which are in cell $c_{i,j}$
6: select best match and save it
7: **end for**

There is a potential problem with the transformed keypoint. Original algorithm say, that the geometrically near keypoints are in cell where the keypoint is transformed. But keypoint may be near cell $c1$ border and the nearest keypoint from another image may be in another cell $c2$ which is next to $c1$, see Figure 2.2. The improvements will be discussed in next section 4.4.

The correspondences between two images after unguided matching and guided matching is shown in Figure 2.3.

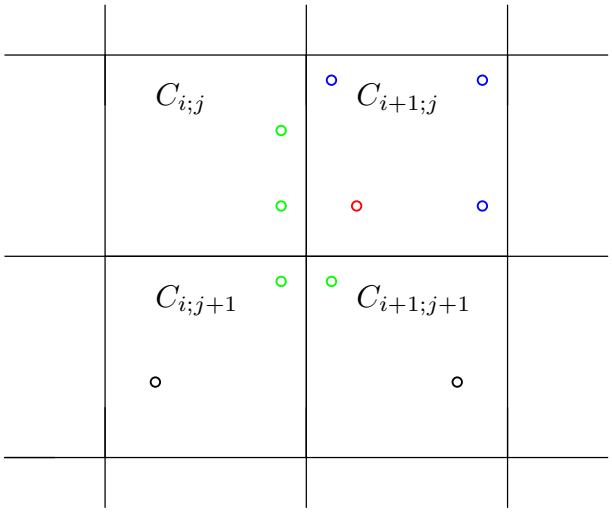


Figure 2.2: The guided matching select geometrically near keypoints. The grid with cells $c_{i,j}$ are precomputed for image. The red circle is transformed keypoint from another image. The blue circles are keypoints which are selected as nearest by original algorithm. The green circles are keypoints which are geometrically near to transformed keypoint.

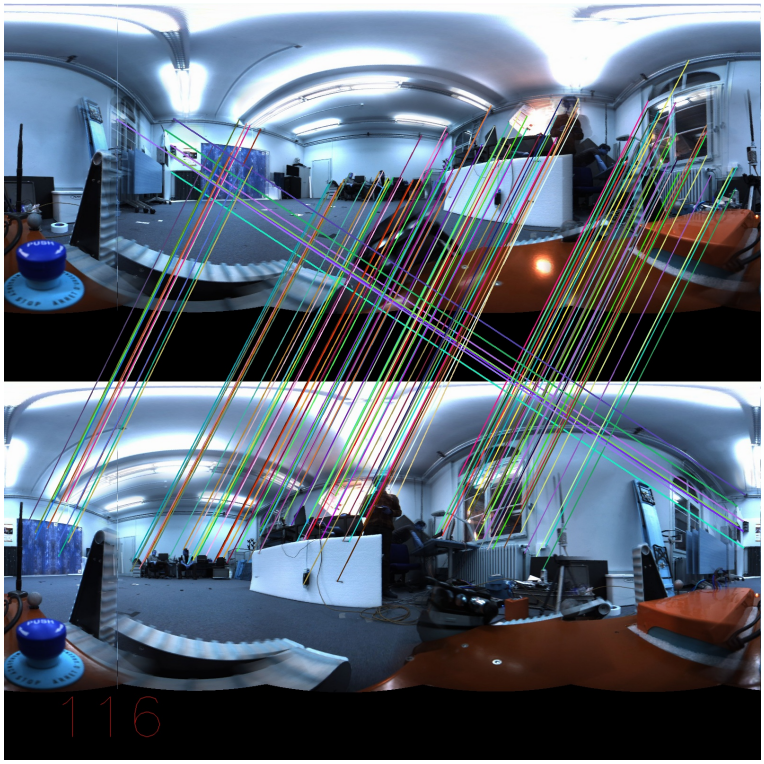


Figure 2.3: Correspondences between two images.

2.4 Scale estimation

The scale can be estimated from a correspondence between keypoints in a image to some 3D point. The 3D point can be landmark (will be explained in next section 2.5) or can be triangulate from two previous frames. Thus, the minimum for scale estimation is correspondences between three images. For better understanding the algorithm see Figure 2.4 or algorithm 2.

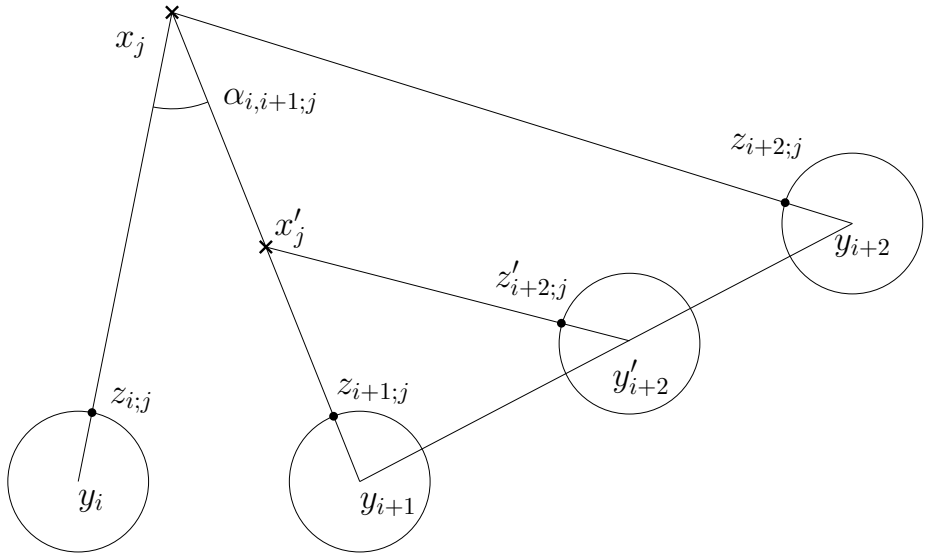


Figure 2.4: The auxiliary sketch to estimate the scale.

Algorithm 2 Scale estimation algorithm. The algorithm estimates scale between pose y_{i+1} and pose y_{i+2}. To get an idea see Figure 2.4

1: Find keypoint which is associated to some 3D point (can be landmark or can be triangulated) in images from pose y_{i+1} and pose y'_{i+2}. The keypoints are named $z_{i+1,j}$,$z_{i+1,j'}$ in pose y_{i+1} and $z_{i+2,j}$ in pose y_{i+2}. The pose y'_{i+2} is not precise positioned in world coordinating frame. It's position is calculated from unguided matching and we do not know scale. It is distance from pose y_{i+1} to pose y'_{i+2}, $|y_{i+1}, y'_{i+2}|$.

2: Triangulate some 3D point x'_j from poses y_{i+1} and y_{i+2}. The 3D point lays on ray from pose y_{i+1} but generally not in exact position as 3D point x_j. We assume that x_j is in correct place in world coordinating frame.

3: Finally the the exact pose y_{i+2} is calculated by similarity of triangles. It is true that

$$\frac{|y_{i+1}, x'_j|}{|y_{i+1}, x_j|} = \frac{|y_{i+1}, y'_{i+1}|}{|y_{i+1}, y_{i+1}|}.$$

The distance $|y_{i+1}, y_{i+1}|$(scale) is now calculated.

2.5 Create Landmarks

The landmark is a point in 3D, which is defined by vector $[x, y, z]$. It presents a distinct part of the scene. The appearance of such landmark does not substantially changed over the time. A lifespan of the landmark has three main phases – formation, maintenance and termination. The landmark is created by a triangulation of three images. In these images is found a correspondence from which the 3D point is calculated by triangulation. The maintenance - there is a adding a new correspondence in a new image to the current three images. These corresponding points improve a estimation of the 3D landmark position. The termination of the landmark occurs when the corresponding point to the landmark is not detectable.

3. Experiments with the original algorithm

Before improving the original algorithm it was necessary to test it. The tests had to be complex and had to cover the most of possible different environments and situations where the robot could appear. Three environments were chosen for the testing - a room, a street and a forest. Each of them represents a typical situation.

3.1 Room

The dataset from this experiment has been captured in a square room in Zurich in March 25, 2013. For this scene has been used robot from Zurich, which has another camera calibration then robot from CTU Prag. The robot started approximately in the center of that room. Then was driven straight near to wall. After that it was driven along the walls and made three square laps. The trajectory is approximate square, where two opposite corners are sharp. In these corners robot had very small turning radius and therefore rotated very quickly. Remaining two corners it curved more smoothly. It is an indoor experiment therefore it brings some problems.

- The indoor scenes are typically less illuminated. Camera have to have a long exposure time to capture enough light. This can cause blurry images during a fast robot motions. This problem also appeared in sharp corners.
- Generally, in a room is typically everything closer to the camera. This way the inaccuracies of spheric cameras approximations are more manifested.

The ground-truth has been attached to this dataset and was measured by VICON 5.0.2. The example of pictures from this dataset is shown in Figure 3.1. The trajectory of the robot was square. Robot made approximately three laps of this trajectory. The calculated angles and positions compared with ground-truth are shown in Figure 3.2. The X-Y graph shows trajectory from the top view, shown in Figure 3.3. There are interesting points A and B in graphs. However, there is a error in estimation position and rotation at the points A and B. In figures 3.2 and 3.3 there are graphs showing number of landmarks during the movement of the robot. The number of landmarks decreases at points A and B. In the graph of position and rotation are also disturbances caused by a small amount of landmarks. This effect may be caused by blurry pictures from camera. An example of a blurry image from position at point A is shown in Figure 3.4. These blurry images were problem for the detector. It causes that the detector found few features and thus the algorithm paired few landmarks as well. From that reason the landmarks probably disappeared.

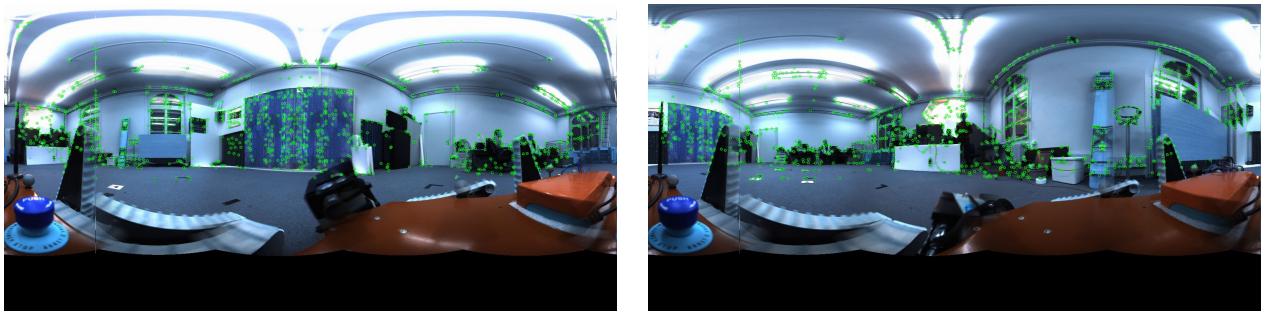


Figure 3.1: Example of pictures from the first dataset. The green circles are the feature points.

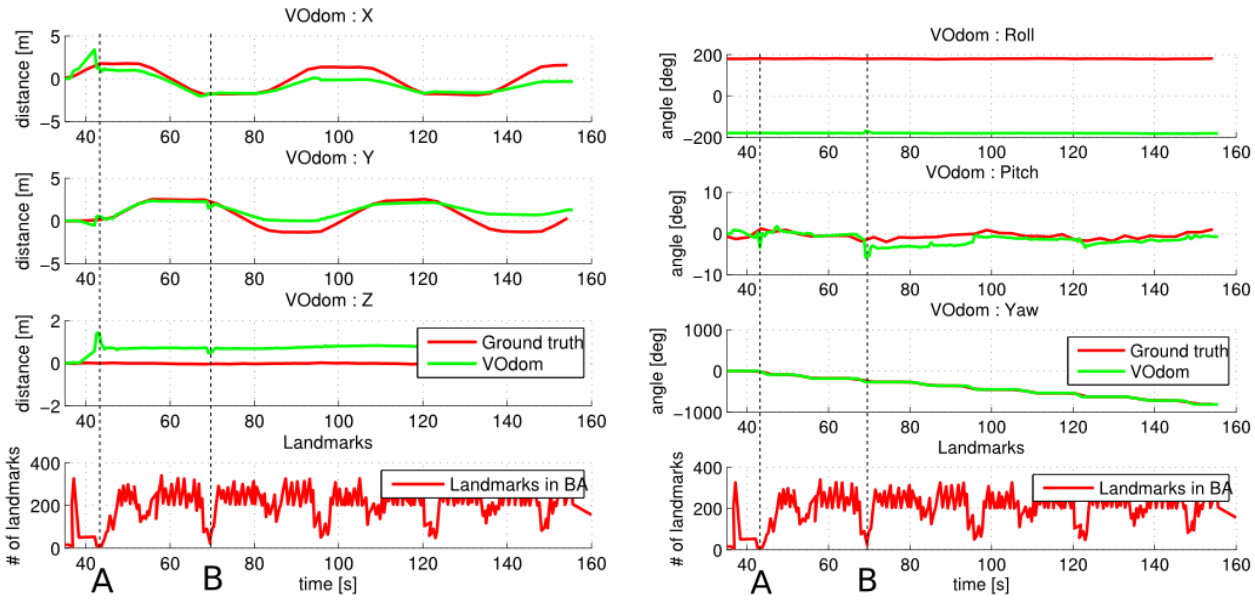


Figure 3.2: Calculated position and rotation compared with the ground-truth.

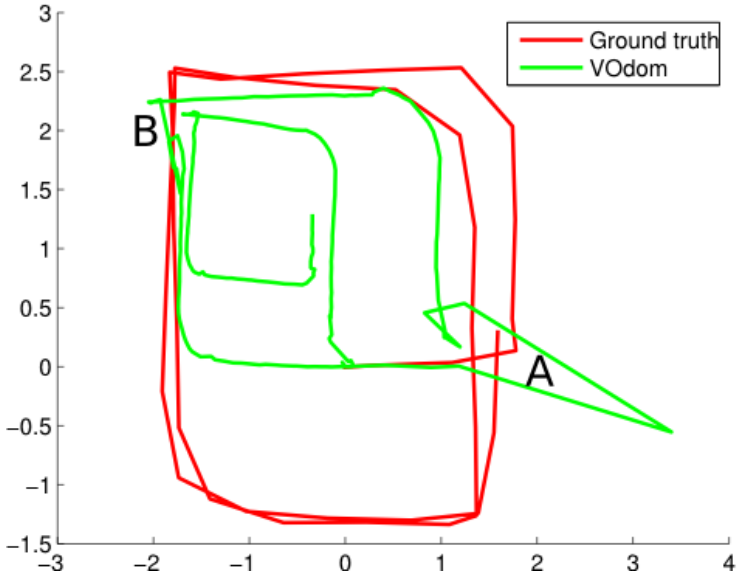


Figure 3.3: X-Y graph shows the trajectory of the robot from the top view.

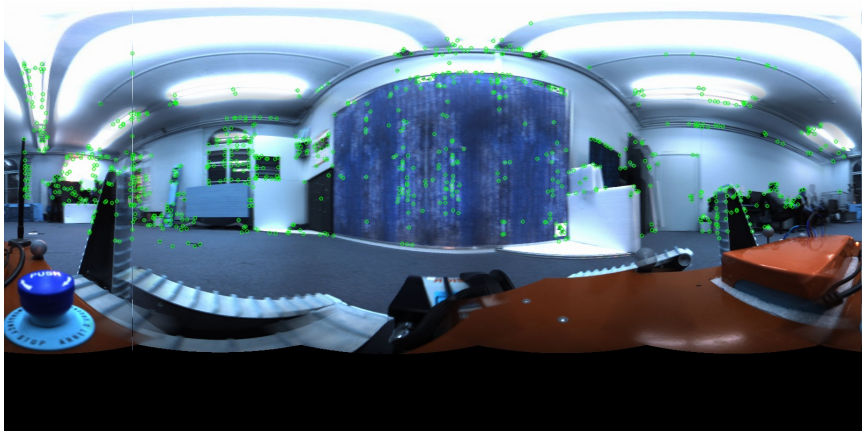


Figure 3.4: An example of a blurry image.

3.2 Street

This dataset has been recorded in Zurich on April 12, 2013. It has been recorded with a robot from ETH Zurich. The example of pictures from this dataset is shown in Figure 3.5 The robot drove along the straight road on one side. After 60 meters it crossed the street and went back on the other side. After that it crossed the street again and was driven to the start position. This created a closed rectangular trajectory of a total length of approximately 135 meters. There were some pedestrians and moving cars. The robot also used the flippers to ride down from a pavement (or ride up as required). Therefore the flippers were not masked during whole drive. That made scene more difficulty. A ground-truth from Laica has been attached to the dataset. The calculated angles and positions compared with the ground-truth and with number of landmarks is shown in Figure 3.6. The X-Y graph is shown in Figure 3.7. The problem in this dataset was during second turn at point C and during third turn at point D. The robot turn probably very fast and the error was in rotation. The problem may occur when the slower PC will be used for computing the algorithm. The slower PC drops images when it is busy. The scale problem is evident in this dataset.

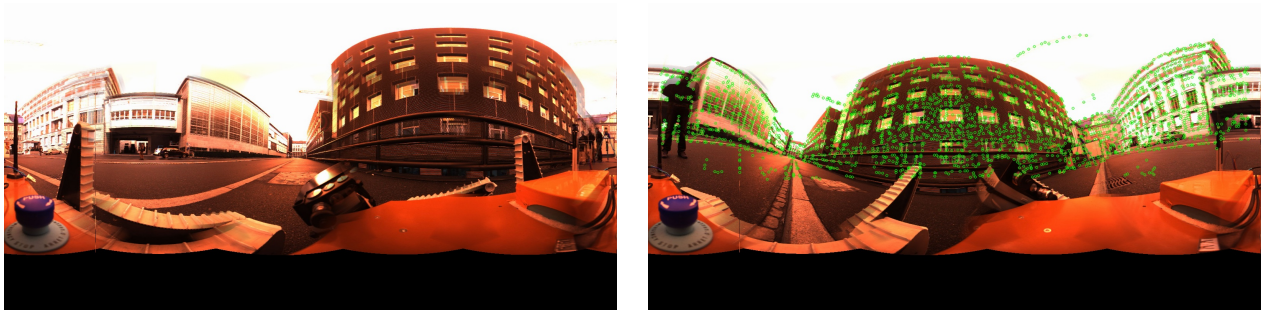


Figure 3.5: Example pictures from the street dataset. The green circles are feature points.

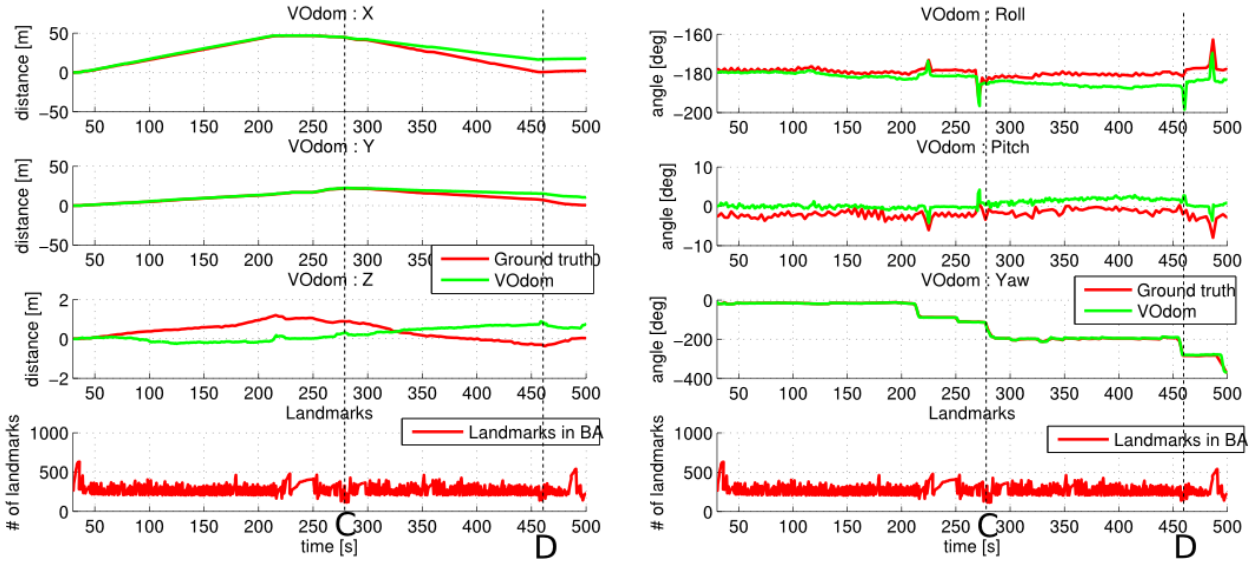


Figure 3.6: Calculated position and rotation compared with the ground-truth.

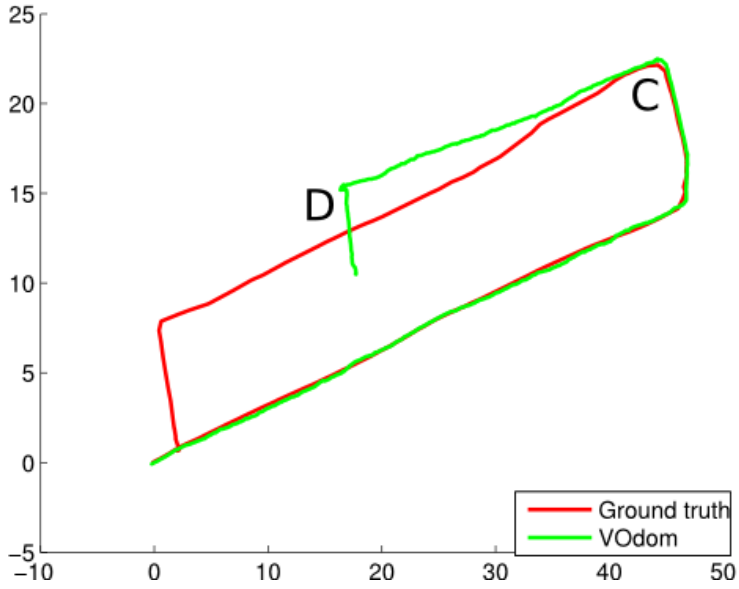


Figure 3.7: X-Y graph which shows trajectory of the robot from the top view.

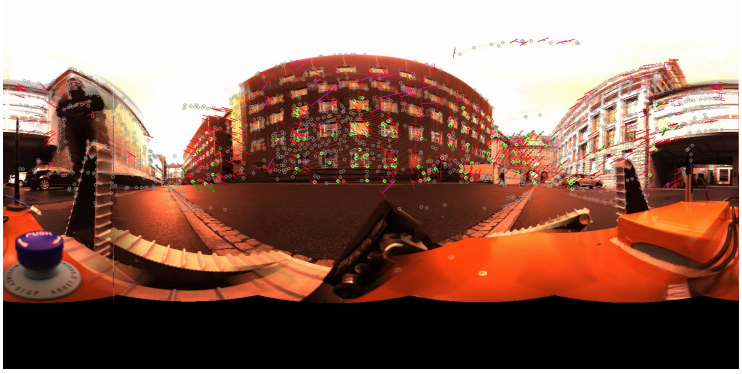


Figure 3.8: The reprojection errors from point D

.

3.3 Forest

The dataset has been recorded in a forest scenery. This scene was the most difficult for the algorithm. The descriptor could not find good features for the matching and the matching algorithm fails. The program terminates with error approximately in middle of scene during scale estimation. The scale estimation need at least one correspond to 3D point. When did not find it, the algorithm terminates a with error. The figures from this are shown in Figure 3.9. The ground-truth was obtained from the ICP (it will be explained in section 5.0.1). The calculated angles and positions are compared with the ground-truth is shown in Figure 3.10. The X-Y graph is shown in Figure 3.11.

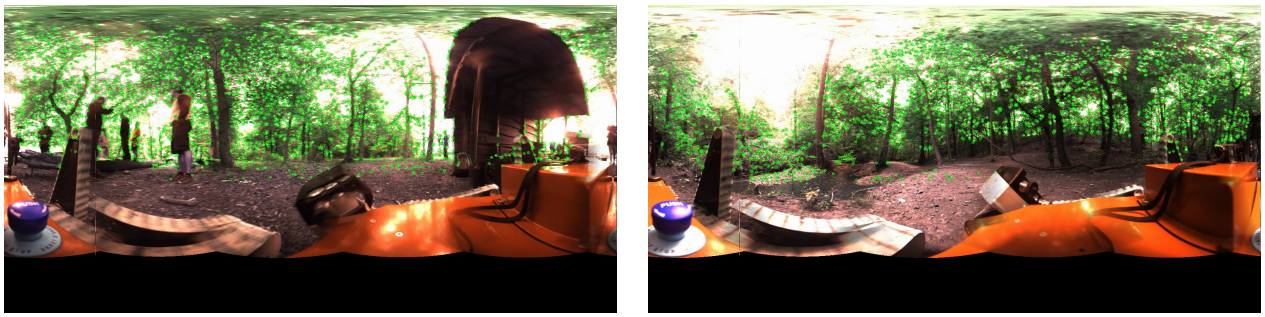


Figure 3.9: Example pictures from forest dataset. The green circles are feature points.

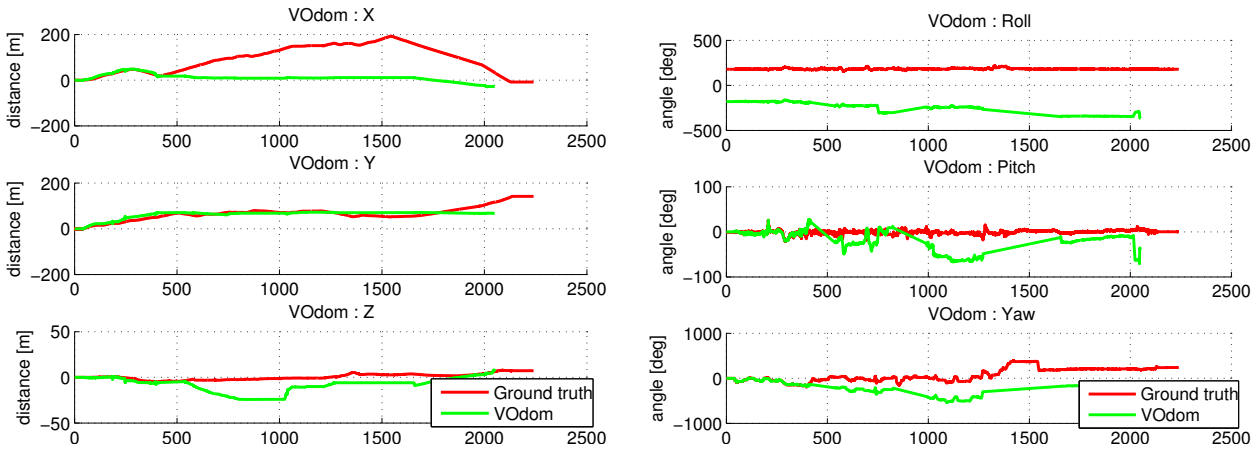


Figure 3.10: Calculated position and rotation compared with the ground-truth.

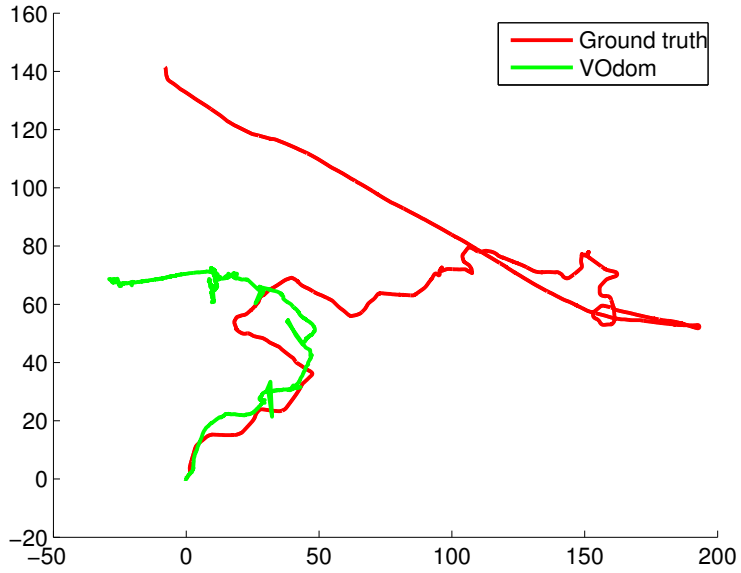


Figure 3.11: X-Y graph which shows trajectory of the robot from the top view.

4. Improvements

In this section is described steps which led to improvements the problems which was detected and described in previous section. At the beginning we dealt with the problem of blurry images. After that we occupied with possibilities of choosing quality landmarks. At the end we improved tracking of key-points from that the landmarks are calculated.

4.1 Blurry image detection

During the experiments has been found that the camera sometimes send a blurred images. We know that exist more types of blurries in the images. [1]. In our case it is a motion blur arising during fast robot's movement in a poorly lit areas. The motion blur appears mainly in horizontal direction because skid-steer mobile robot is able to rotate horizontally very fast. The vertical motion blur may also arise during rolling on the sidewalk, when driving over uneven surfaces etc. It is not so common phenomenon so we will considered only the horizontal motion blur in our improvements.

The problem with blurry images is that there are detected a only few keypoints. The position of detected keypoints is not as precise as in non-blurred images. Inaccurate keypoints position produce inaccurate landmarks which further cause inaccuracy of all the positions estimation. Inaccurate landmarks may be deleted during a bundle adjustment. Experiment 3.1 demonstrated a failure caused by blurry images. The main problem is a insufficient number of keypoints. There are not enough stable correspondences, landmarks

get lost. For our algorithm is better to throw away blurred images than take it into computation.

4.1.1 Algorithm of detection blurry images

To detect blurry images, we suggest to use a Fourier analysis. The Fourier Transform decompose an image into its sinus and cosines components. In other words, it transforms an image from its spatial domain to its frequency domain. The idea is that any function may be exactly approximated exactly with the sum of infinite sinus and cosines functions. The Fourier Transform is a way how to do this [10]. Only a magnitude part of Fourier Transform of the image is used. The magnitude is switched to a logarithmic scale.

Because we want to detect a horizontal blur for that reason is the Fourier transform calculated from rows. In our case is computed in the center of image because the center has more informations than the sides. The useful part of image is shown in Figure 4.1. The final spectrum is computed as a sum of all spectrum from all rows.The spectrum of blurry and normal image is shown in Figure 4.2. The implementation of the algorithm have two phases.

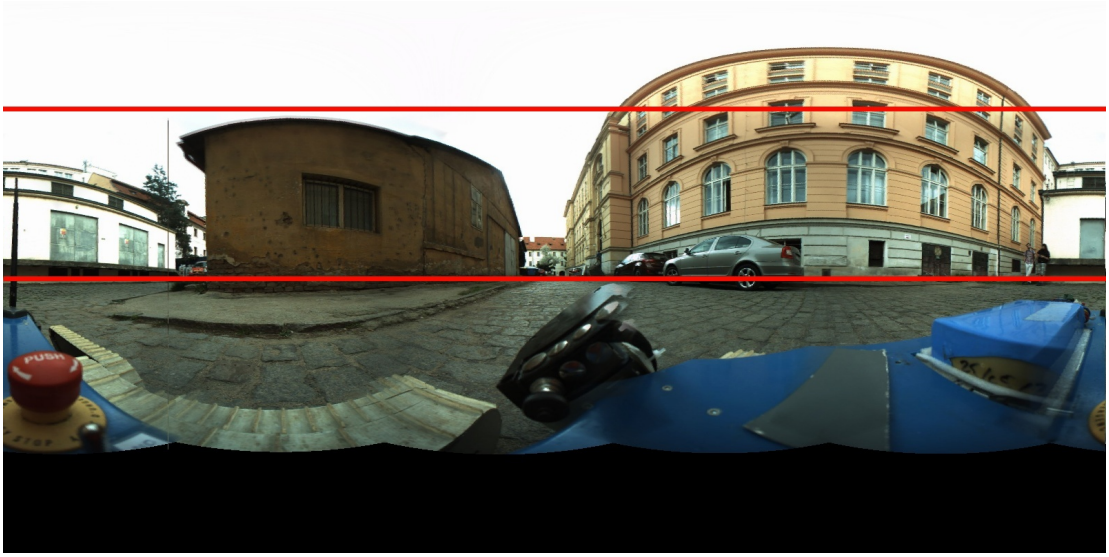


Figure 4.1: Region of interest that is considered in computation is bounded by red rectangle.

The first phase is a learning phase. During the first phase is calculated index of blurriness B_t from the image. B_t is calculated as sum of frequencies. In our case the low frequencies are shown as red line on x axis in Figure 4.2. This first phase is calculated from images only when the robot is not moving. The assumption is that the robot do not capture blurry images when it do not move. The robot do not move when disparity between two consecutive images is small. The second phase is the classification phase. During this phase the index of blurriness B_a is calculated on images which we want to classified. The image is classified as blurry when this condition is true

$$\frac{B_t - B_a}{B_a} > 0.04 \quad .$$

The constant 0.04 was obtained experimentally from datasets which are described in previous sections.

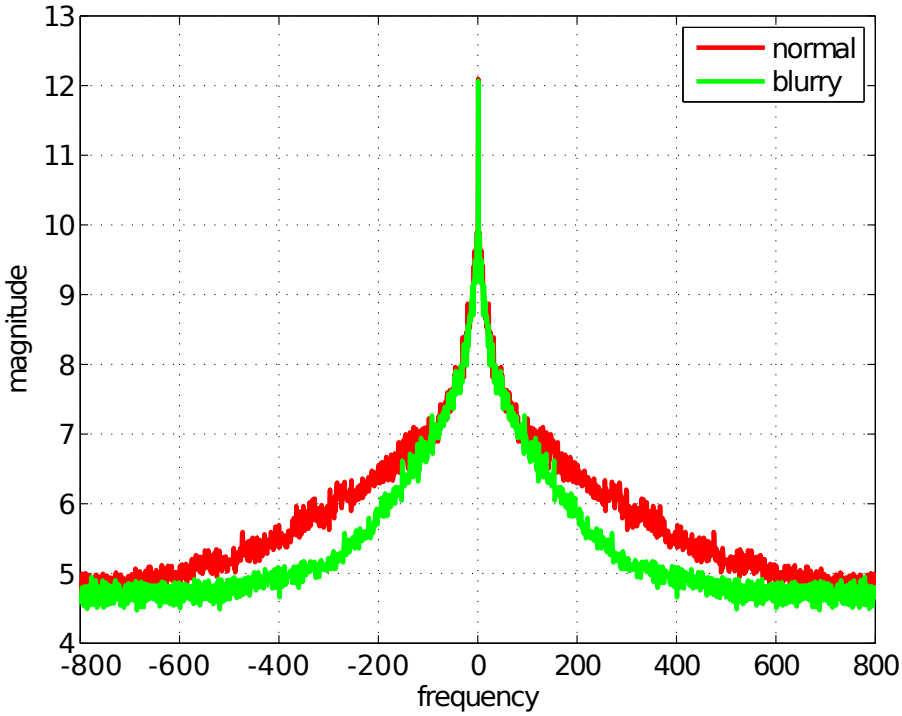


Figure 4.2: Fourier spectrum of the central rows of the image. Note that for the blurry image the higher frequencies attenuate much faster than for the sharp image.

4.2 Inspect of apex angle

During the analysis of the scale estimation a bug in the part of the 3D point computation from three images were detected. This section of the code is used only in the case of few amount of landmarks. The bug was fixed and consequently it was added a controlling of an apex angle in this part of the code. Unless the 3D point is detected at least with apex angle = 1°, so it is ignored. The same is valid in the case of creating the landmarks itself. The apex angle is maximal angel which the observation included by given landmark, see angel alpha in the Figure 4.3.

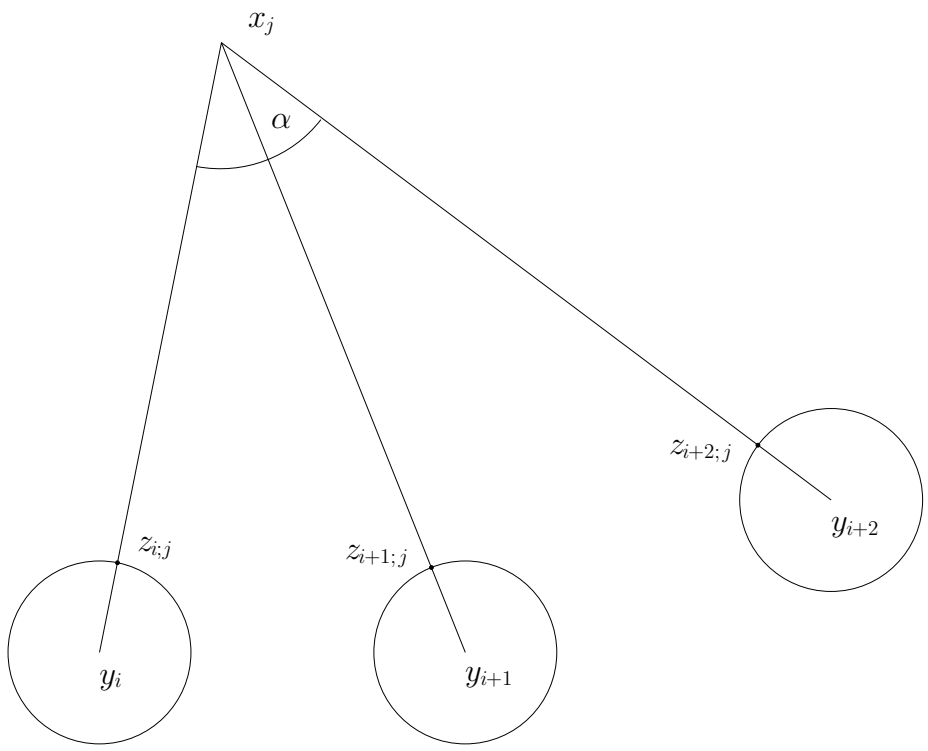


Figure 4.3: Demonstration of apex angle. α is apex angle of landmark x_j.

4.3 Better feature tracking

In section 2.3 was described the guided matching and the potential problems. The problem arises when the keypoint is transformed too close to virtual cell border. All is described in the Figure 2.3. For better feature matching the original algorithm was updated so that the closest keypoint is calculated as Manhattan distance. The picture is not divided into the squares where the nearest point is founding but it is really found the nearest point according to the selected metrics. The results of this change are summarized in table 4.1. The most interesting is row named "bad match - keypoint is not in set of potential match". There we notice the great improvement, which is good for our algorithm. Other bad matches, mentioned in the table are described under the table.

Problem	# in experiment with unchanged algorithm	# in experiment with changed algortithm
good match	42	75
bad match - keypoint is in set of potential matches, but not matched	35	47
bad match - keypoint is not in set of potential matches	52	3
bad match - not keypoint	50	68
bad match - bad mask or robot motion	20	23
bad match - stitching	14	14

Table 4.1: Guided matching inspection. "stitching" - incorrectly composite images from the cameras. "not the keypoint" - in the keyframe there is a point as the keypoint however in the actual image it is not detectable (the orb has not found it). "bad mask or robot motion" - robot thanks to its rotation has covered the place where the keypoint should be found. "keypoint is in set of potential matches, but not matched" - other reason than the aforementioned

4.4 Guided-matching improvements

The next improvement of guide matching is increasing of correspondence number between the images. The previous guided matching algorithm has worked so it was not necessary to create other match pairs between the actual image and the keyframe. It is not bad in principal, however there is a problem that the guided matching supposes any or little translation robot's motion. Therefore we have changed the algorithm - see Figure 4.4 , so the guide matching is now calculated between the actual image and the previous image, there is the more confidence that the translational will be as little as possible. If we know that there is a correspondence between keypoint in actual image ($z_{A,j}$) and keypoint

in previous image ($z_{P,j}$) (purple dashed line) and the correspondence between keypoint in key image ($z_{K,j}$) and keypoint in previous image ($z_{P,j}$) (red dashed line), we then can determine the correspondence between ($z_{K,j}$) and ($z_{A,j}$)(blue dashed line). It is exactly the crucial relation which we find in the algorithm guided matching.

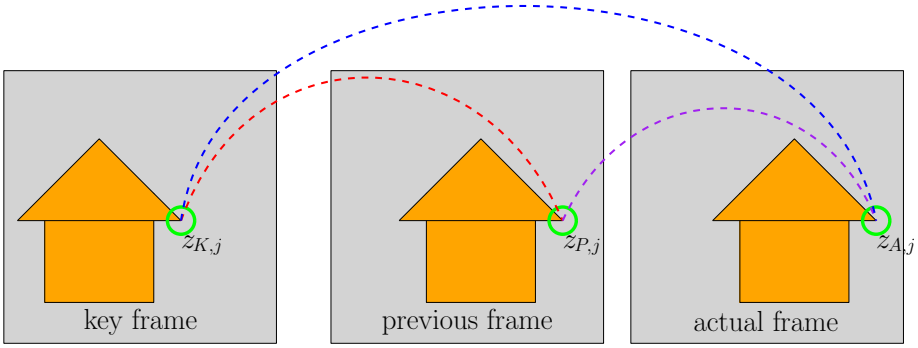


Figure 4.4: Improved guided matching: correspondence between keypoint in actual image ($z_{A,j}$) and keypoint in previous image ($z_{P,j}$) (purple dashed line), correspondence between keypoint in key image ($z_{K,j}$) and keypoint in previous image ($z_{P,j}$) (red dashed line),correspondence between ($z_{K,j}$) and ($z_{A,j}$)(blue dashed line)

5. Experiments

In this section, we analyze the upgrade algorithm performance in various environments. The algorithm is compared with other types of odometry and with also groun-thruth if it will be available. The other types of odometry are the Visual Compass and with the ICP. The ground-thruth have been obtained from VICON and Leica.

5.0.1 ICP

ICP is a method which we use for estimating position and orientation with respect to a global representation called Map. The method is deeply described in paper [6] in section 3.4. It uses rotating laser RangeFinder's data from which is constructed a 3D point cloud. It uses a derivation of the point-to-point ICP algorithm introduced by [2] combined with the trimmed outlier rejection presented by [3]. It is the best position estimation, which we could obtained, if we did not have VICON or Leica. For our application is this comparison sufficient because when the visual odometry fails it is recognizable for us. We can find it out because there is a suddenly big change in position and angle. Furthermore, it is described in following cases.

5.0.2 VICON

For indoor measurements, we use a Vicon motion capture system with nine cameras which cover more than $20m^2$ and giving a few millimeter accuracy at 100 Hz. Vicon can measure the position and the orientation of the robot.

5.0.3 Leica

For outdoor ground-truth estimation, we use a theodolite from Leica Geosystems(TS15). It can track reflective prism to continuously measure robot's position. The precision of position is 3mm. However, Leica cannot measure the orientation of the robot.

5.0.4 Visual Compass

The visual compass is an algorithm, which estimates robot's yaw angle from omnidirectional camera [9]. It is very fast and did not consume much processor time because is alculated using discrete Fourier transform.

5.1 Error function

To evaluate the accuracy of the algorithm we have to set down its error. We use the same method like in the article [6], so we determine the final position error like:

$$e_{rel} = \frac{\|\mathbf{p}_l - \mathbf{p}_{ref,l}\|}{D}$$

where l is the index of the last position sample $\mathbf{p}_l$ with the corresponding reference position $\mathbf{p}_{ref,l}$. The D is total travelled distance.

While this metric is convenient and widely used in the literature, it is however representative only in the end point error regardless of the intermediary results. To express error in dependence on the time we use average position error:

$$e_{avg} = \frac{\sum_{i=1}^{l(t)} \|\mathbf{p}_i - \mathbf{p}_{ref,i}\|}{l(t)}$$

where $l(t)$ simply maps time t to the corresponding sample l. The l is defined as $1 \leq l \leq$ totalnumber of samples.

5.2 Experiments with improved algorithm

In this section the experiments with upgrade algorithm will be evaluated. The three experiments setting (room, street and forest), described in previous section 3, will be used for the comparison of the improvement algorithm with the origin algorithm. Moreover, we added a several new tests, which was created specially for experiments with this algorithm.

5.2.1 The Yard experiment

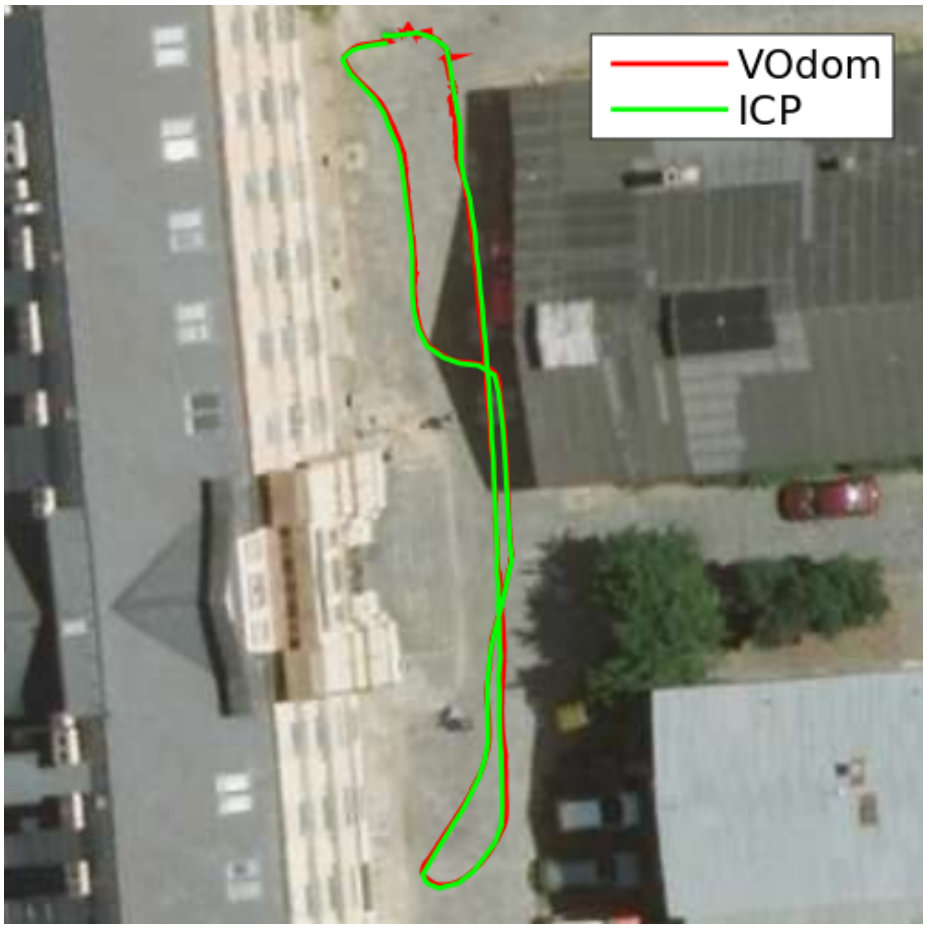


Figure 5.1: The Yard experiment: A robot path computed by proposed VO and referenced overlayed over a satellite image.

This dataset was captured at Charles Square in the center of Prag in the university courtyard. For the recording was used our robot with better camera calibration. The path of the robot was 86.5 meters long. The reference trajectory was calculated from robot's ICP 5.0.1. The start position and the end position of the path were on the same place, on the manhole cover. the starting rotation as well as ending rotation was identical. During the experiment the robot drove about 36 meters along the university building where the ORB could find a lot of keypoints. During the experiment the robot turned back approximately a halfway at side entrance to the university and drove to the start position. The trajectory also with the map[1] as background is shown in Figure 5.1.

The scene of this experiment is for the visual odometry algorithm quite easy. There are a lot of well-marked points on facade houses in the scenery. Therefore is the key-points pairing good and thus it is created sufficient amount of quality landmarks. An example of an image from that scenery is shown in the Figure 5.2

Figure 5.2: The Yard experiment: Picture from the ladybug3 recorded during the experiment. In the picture there are shown some key-points, which belongs to landmark. The red number is number of previous pictures, where exist correspondences to this keypoint. The black number is the landmark's apex angle.

[1]map was obtained from http://www.mapy.cz/

The Figure 5.4 shows the trajectory in the direction of x-axis. The x-axis points in the direction of heading vector of the robot. The z-axis is perpendicular to x-axis and points up. The y-axis is perpendicular to both and points so that the conditions of a right-handed coordinate system were accomplished. The second graph in Figure 5.4 shows average position error in x-axis. The figures 5.5 and 5.6 shows a similar graphs for axis x and axis y. The graphs 5.4, 5.5, 5.6 and 5.1 shows that position estimate is very good. The final error at the end of algorithm is

$$\mathrm{error_x(T)} = 0.29\mathrm{m}, \mathrm{error_y(T)} = 0.18\mathrm{m}, \mathrm{error_y(T)} = 0.23\mathrm{m},$$

where T is time at the end of the experiment. The final position error is

$$\sqrt{\mathrm{error_x(T)}^2 + \mathrm{error_y(T)}^2 + \mathrm{error_z(T)}^2} = 0.41\mathrm{m}.$$

It means that the error in this experiment is 0.5 cm for every meter of the path. The disturbances at the beginning of the algorithm are caused by a small number of high quality landmarks. The quality of landmarks are measured by apex angle 4.2. After few meters the number of high quality landmarks increase and the disturbance disappear. This consequence has a effect on local position estimation however the global position estimation is calculated relatively accurately.

In the Figure 5.3 is a histogram of landmarks lifespan, landmark lifespan is number of images in which the key-point belonging to corresponding landmark appears. According to this assumption the experiment went well with small error.

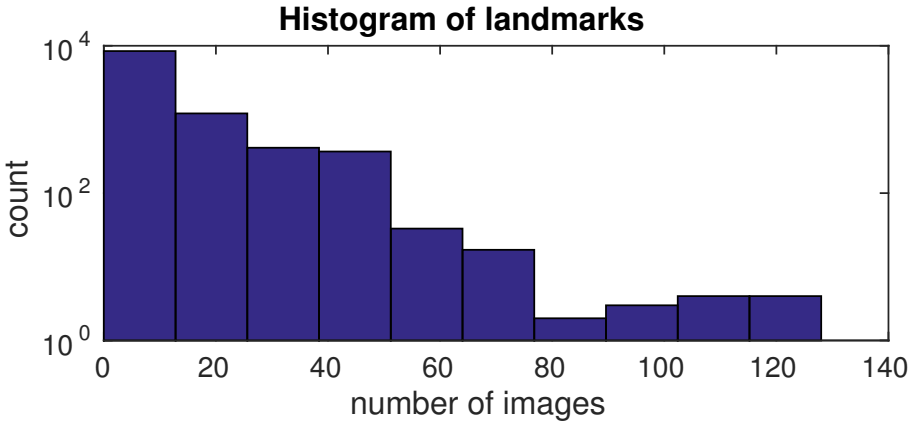


Figure 5.3: The Yard experiment: Histogram of landmarks lifespan, y-axis has logarithmic scale.

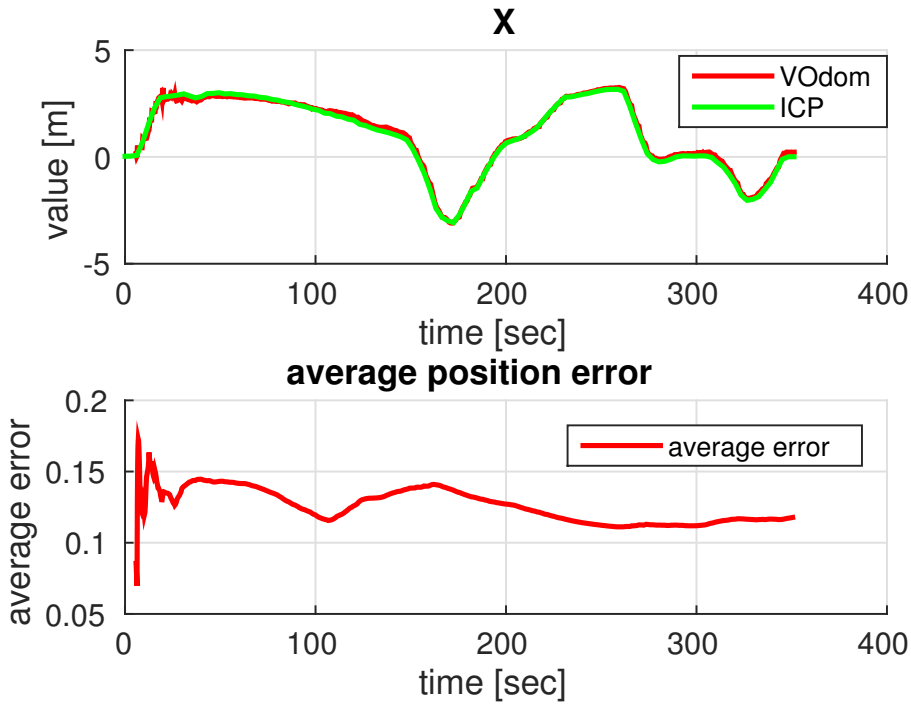


Figure 5.4: The Yard experiment: First graph shows the trajectory in the direction of x-axis in meters. The second graph shows absolute error in meters.

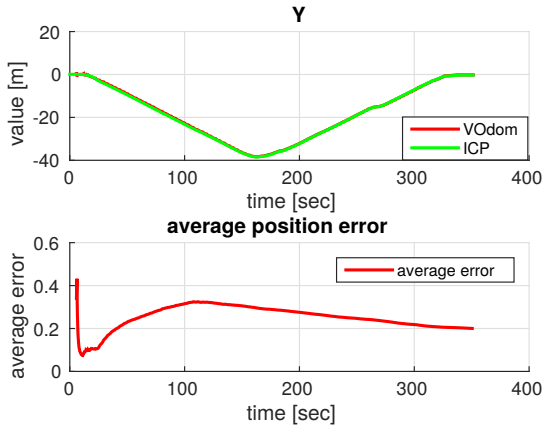


Figure 5.5: The Yard experiment: First graph shows the trajectory in the direction of y-axis in meters. The second graph shows absolute error in meters.

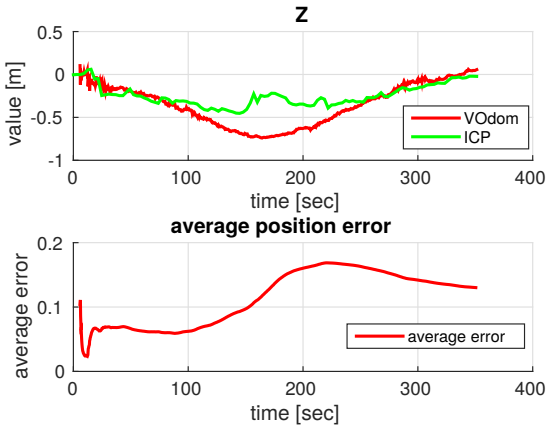


Figure 5.6: The Yard experiment: First graph shows the trajectory in the direction of z-axis in meters. The second graph shows absolute error in meters.

The Figure 5.7 shows the progress of yaw rotation in relation to time. The yaw angle is the most important angle in this experiment. The yaw angle correspond to heading vector of the robot. It is also important because robot moves on an approximately horizontal surface thus roll and pitch are almost 0 degree. The roll and pitch are shown in Figure 5.9 and Figure 5.8. The error in the yaw estimation is shown in second graph in Figure 5.7. The computation of error in angles estimation is the same as computation of error in position estimation. The final errors are

$$\text{error}_{\text{yaw}}(\text{T}) = 1.3°, \text{error}_{\text{pitch}}(\text{T}) = 2.7°, \text{error}_{\text{roll}}(\text{T}) = 2.8°.$$

The angles estimation is very good, as well.

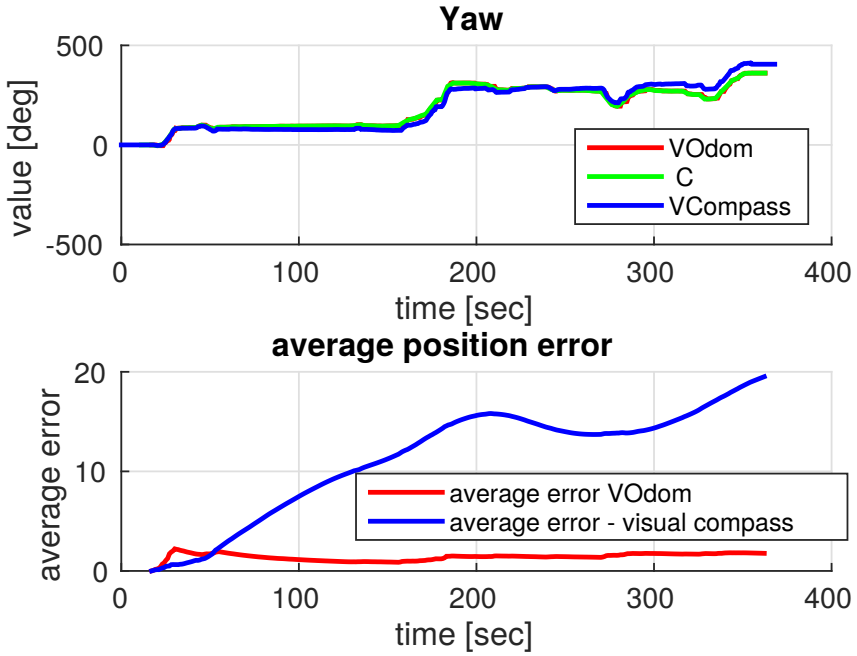


Figure 5.7: The Yard experiment: First graph shows the progress of yaw rotation in relation to time in degrees. The second graph shows the average error in degrees.

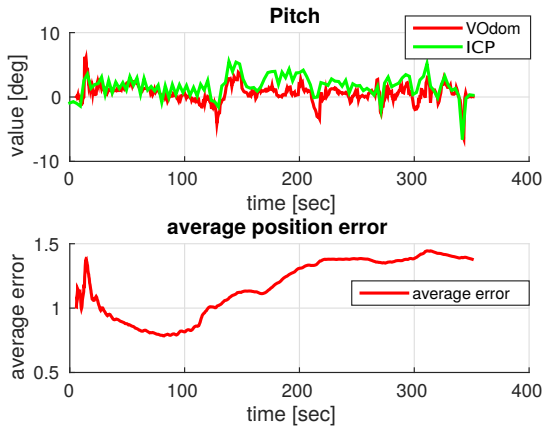


Figure 5.8: The Yard experiment: First graph shows the progress of pitch rotation in relation to time in degrees. The second graph shows the average error in degrees.

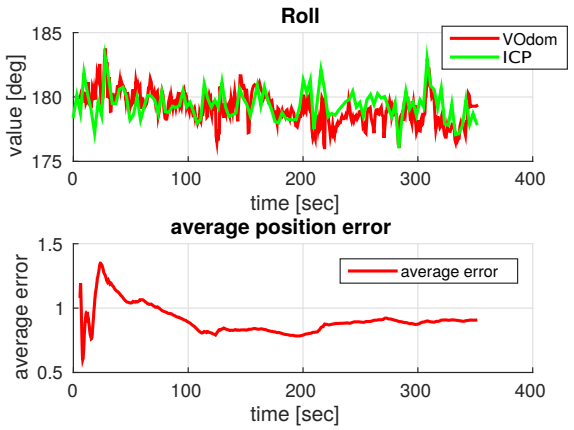


Figure 5.9: The Yard experiment: First graph shows the progress of roll rotation in relation to time in degrees. The second graph shows the average error in degrees.

5.2.2 The Room experiment

The conception and description of this experiment were described above, see chapter 3.1. The data captured in Zurich was tested again however by modified algorithm. The final progresses of computing robot's positions are depicted in the graphs below. It is obviously that the modifying of the original algorithm was useful for better the position estimation and the robot's rotation.

The graph 5.10 compared tested trajectories from the top view. The Vicon trajectory is compared with our improved algorithm and with the old one. The improved algorithm tracks better the ground-thruth from Vicon. It can also be verified using the next graphs. The graph shown in Figure 5.11 compares x-axis and the following graphs 5.12,5.13 compare y-axis and z-axis.

The deleting of blurry images thanks to improvements in section 4.1 had the most positive influence to the algorithm and its behavior. A new model of camera had also an influence to the improvement of the algorithm, then the images were liking better, called stitching. Thus the algorithm can keep better the global scale. Although it is slightly worse in angle Yaw estimation, see Figure 5.14 However, this error is very small: 2° when the robot's rotation during experiment was about 810°. The algorithm behaves better in estimation of other angles, see Figure 5.15 and Figure 5.16.

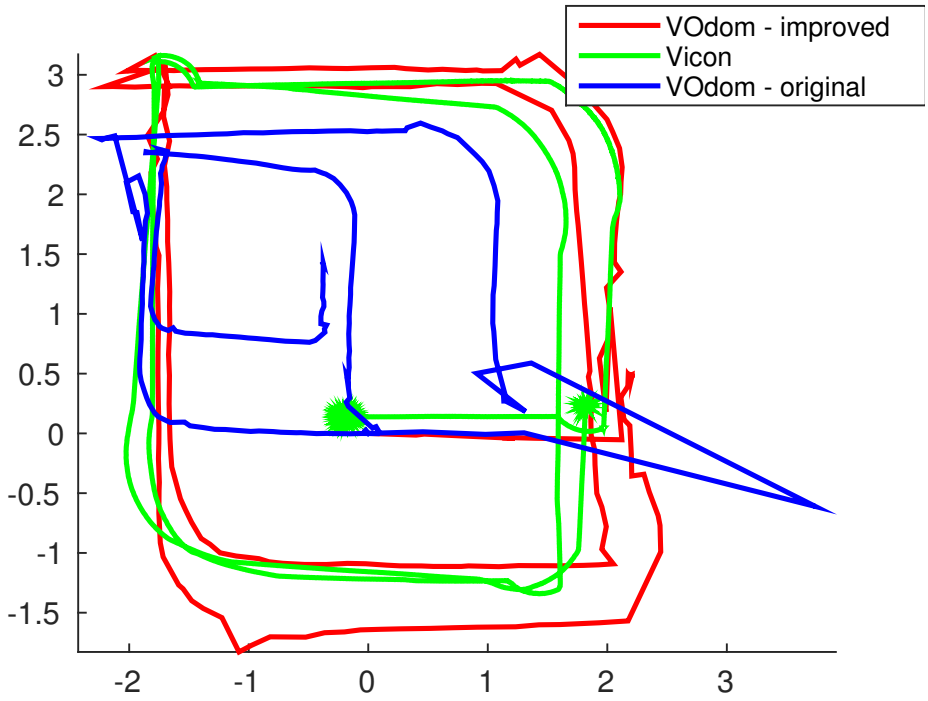


Figure 5.10: The Room experiment: Graph shows the trajectory from the top in meters.

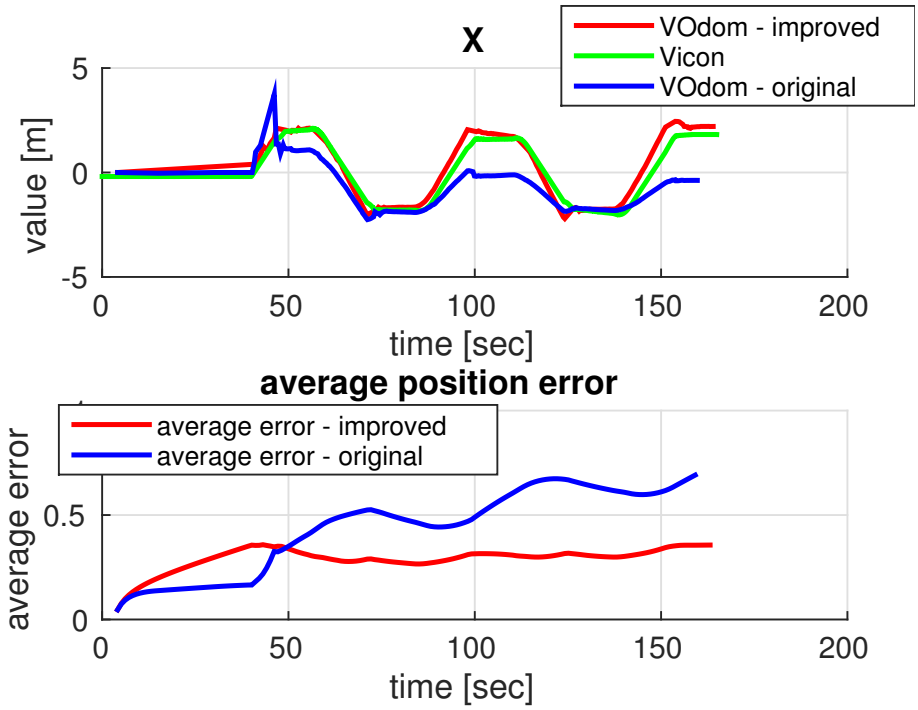


Figure 5.11: The Room experiment: First graph shows the trajectory in the direction of x-axis in meters. The second graph shows the average error in meters.

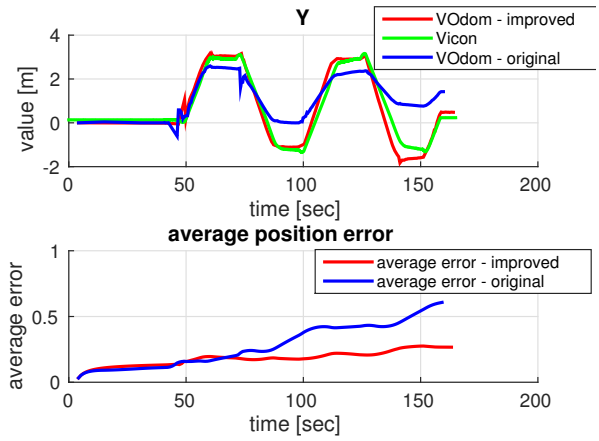


Figure 5.12: The Room experiment: First graph shows the trajectory in the direction of y-axis in meters. The second graph shows the average error in meters.

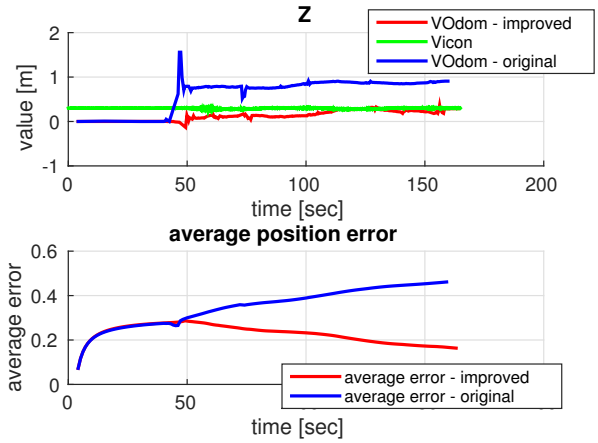


Figure 5.13: The Room experiment: First graph shows the trajectory in the direction of z-axis in meters. The second graph shows the average error in meters.

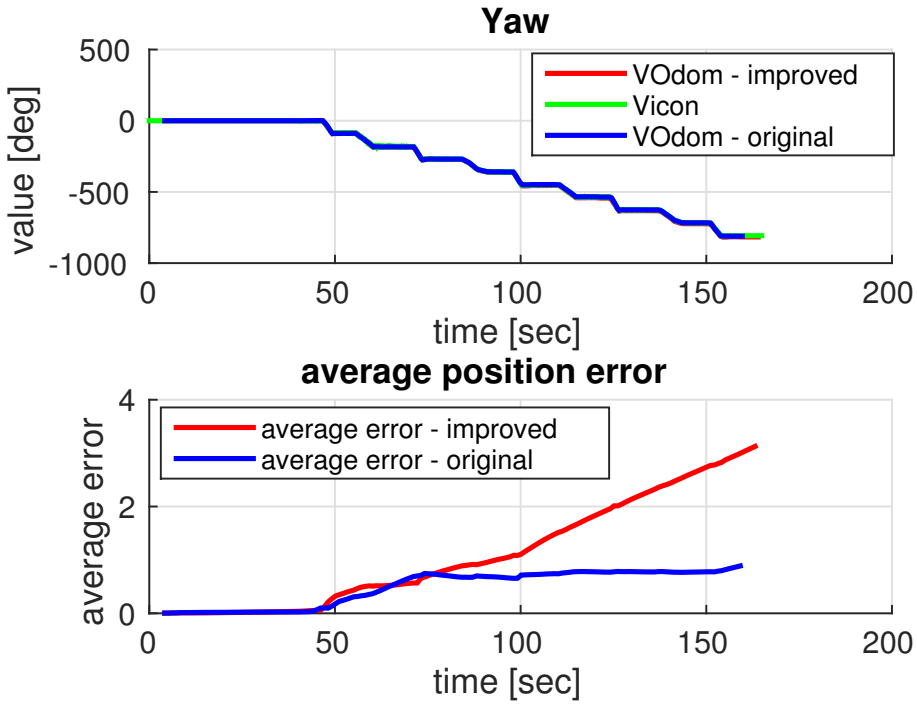


Figure 5.14: The Room experiment: First graph shows the progress of yaw rotation in relation to time in degrees. The second graph shows the average error in degrees.

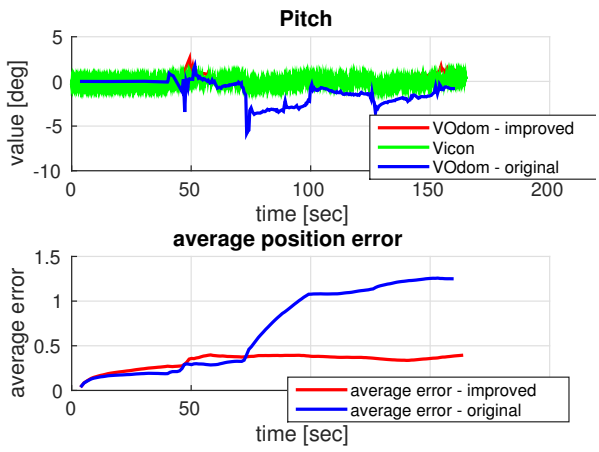


Figure 5.15: The Room experiment: First graph shows the progress of pitch rotation in relation to time in degrees. The second graph shows the average error in degrees.

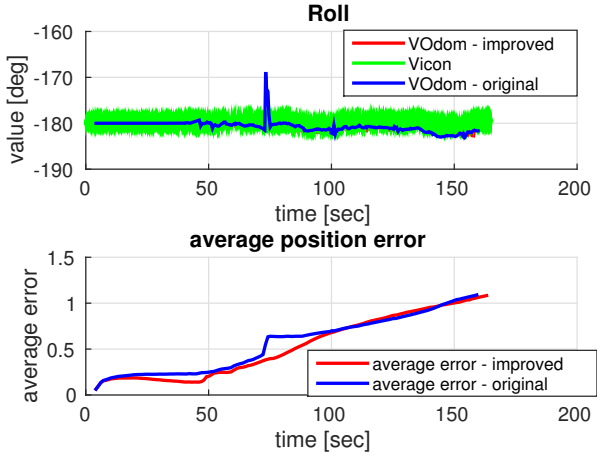


Figure 5.16: The Room experiment: First graph shows the progress of roll rotation in relation to time in degrees. The second graph shows the average error in degrees.

5.2.3 The Street experiment

The scene was described in the section 3.2 above. In this experiment the new virtual camera node has been used. This virtual camera used better parameters and stitched images better. There are no ghost elements in the picture, especially windows, which can caused bad feature matching. In this scene we can observe that landmarks can stay stable for a long time. For example see Figure 5.17. There we can see that landmark near red dot has been on 45 images and his apex angle is 30.9° . This landmark give us very useful information about his location. If we have amount landmarks with this precise location we can estimate robot's location well. See figures 5.18,5.19,5.20 and 5.21, which compare Visual odometry estimation with estimation from Leica system, ICP.

The comparison of the angles with the ICP values is shown in the figures: 5.22, 5.23 and 5.24 This experiment went very well. The algorithm shows a small error in all measured parameters. It also can keep the global scale estimation. This all is caused an overall appearance of scene. There are many interesting points through the scene (mainly window) where many key-points can be found. These key-points are paired stable and therefore it create stable landmarks with great precision.

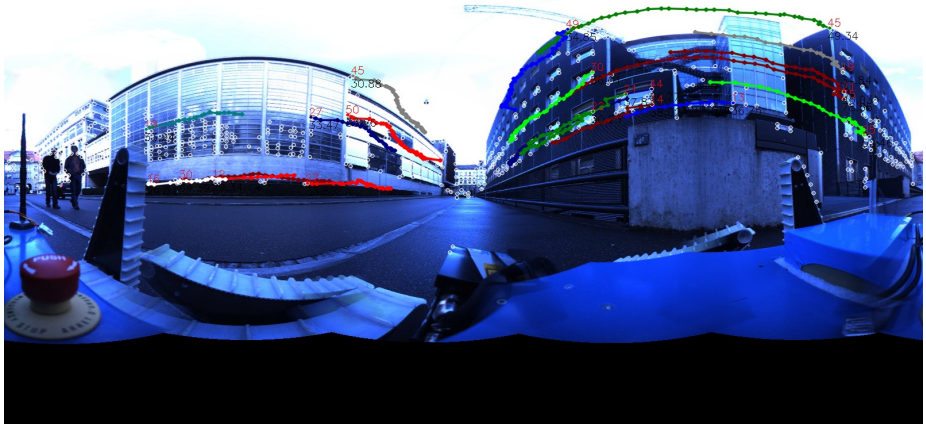


Figure 5.17: The Street experiment: Picture from the ladybug3 recorded during the experiment. In the picture there are shown some key-points, which belongs to the landmark. The red number is number of previous pictures, where exist correspondences to this keypoint. The black number is the landmark's apex angle.

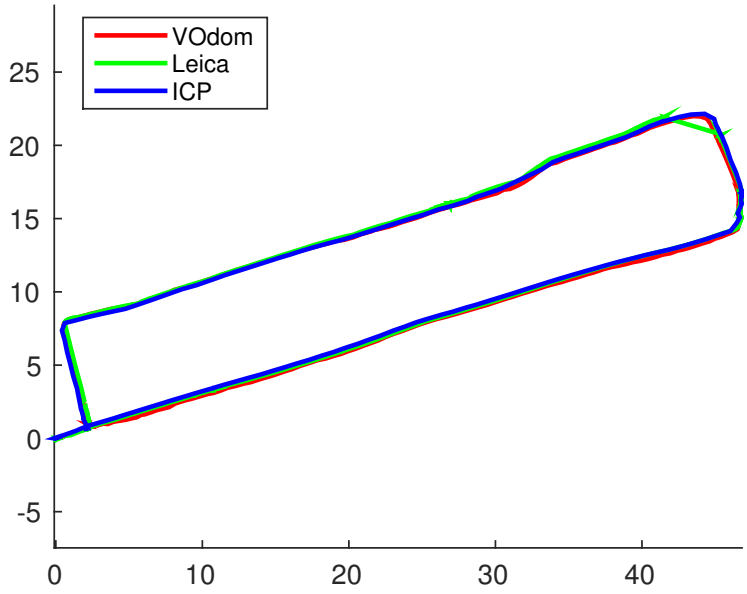


Figure 5.18: The Street experiment: Graph shows the trajectory from the top in meters.

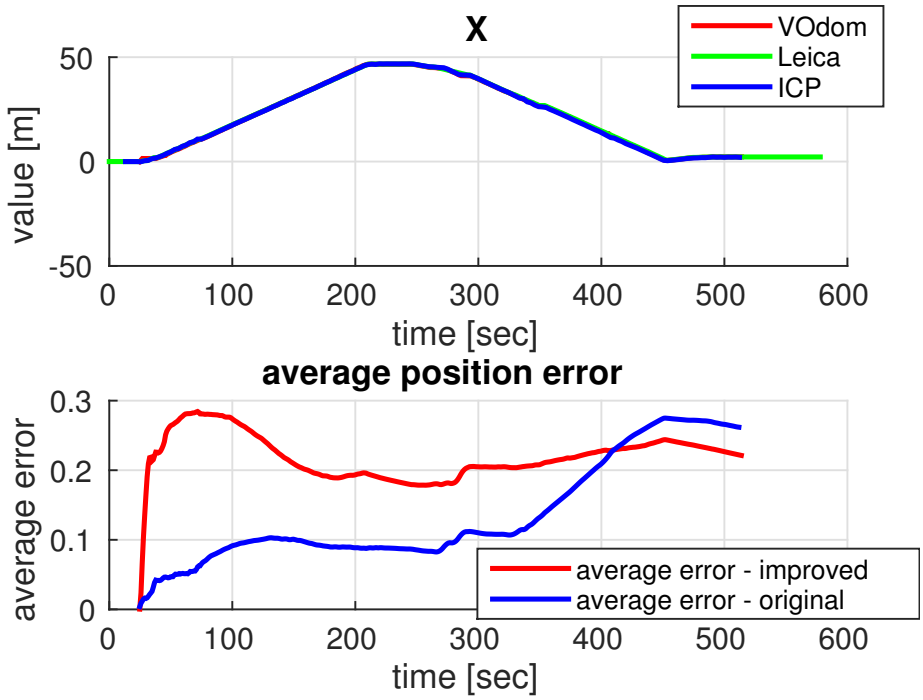


Figure 5.19: The Street experiment: First graph shows the trajectory in the direction of x-axis in meters. The second graph shows the average error in meters.

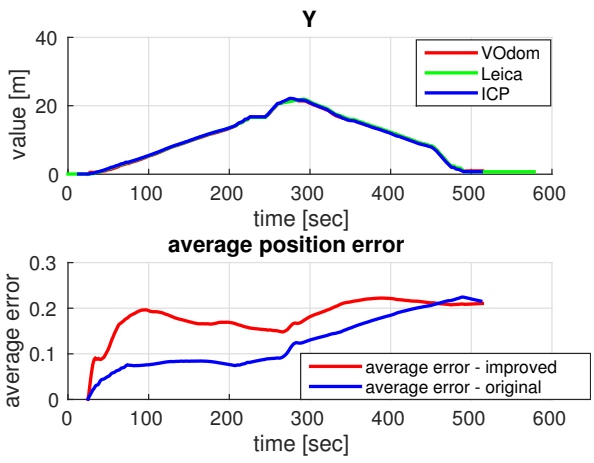


Figure 5.20: The Street experiment: First graph shows the trajectory in the direction of y-axis in meters. The second graph shows the average error in meters.

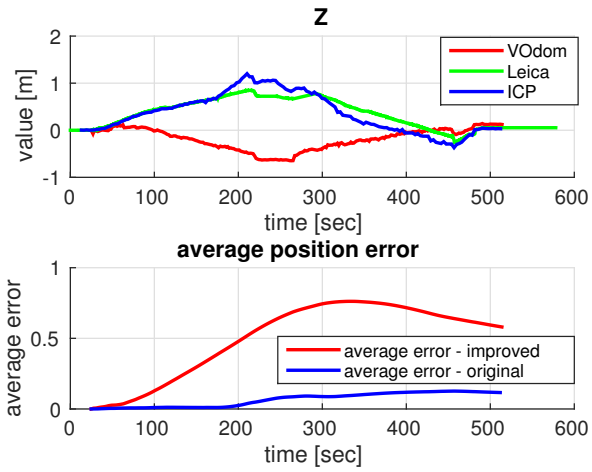


Figure 5.21: The Street experiment: First graph shows the trajectory in the direction of z-axis in meters. The second graph shows the average error in meters.

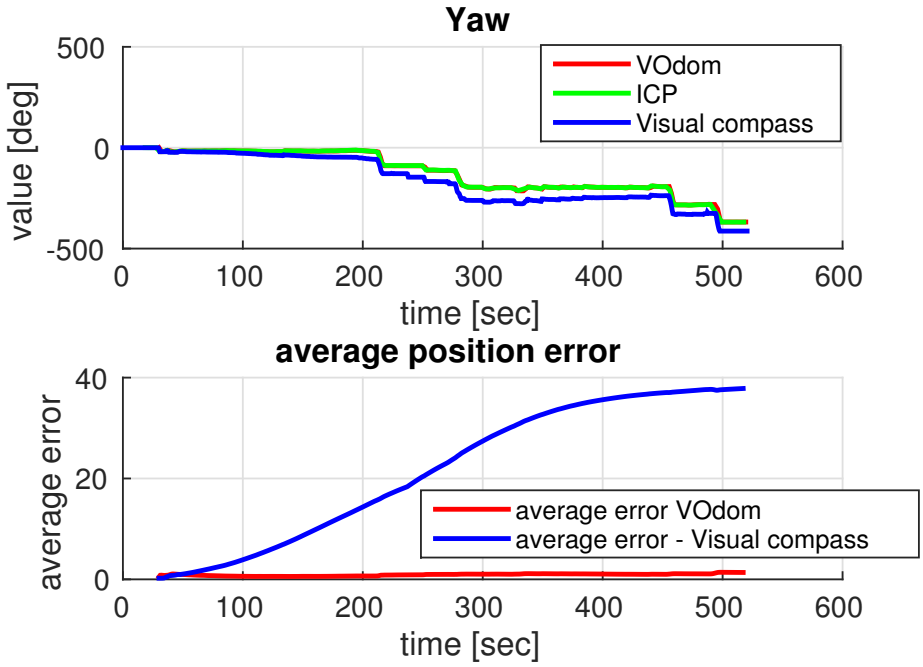


Figure 5.22: The Street experiment: First graph shows the progress of yaw rotation in relation to time in degrees. The second graph shows the average error in degrees.

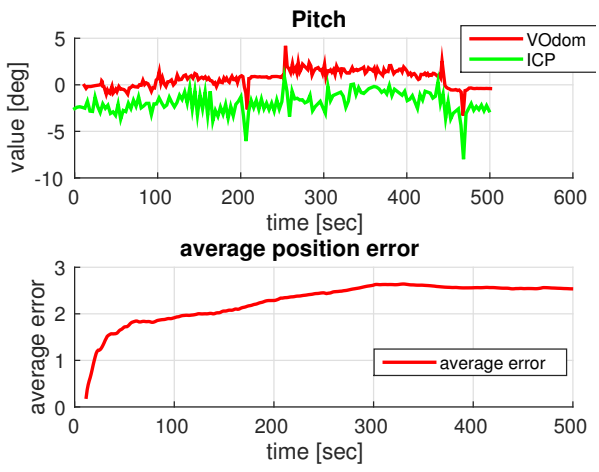


Figure 5.23: The Street experiment: First graph shows the progress of pitch rotation in relation to time in degrees. The second graph shows the average error in degrees.

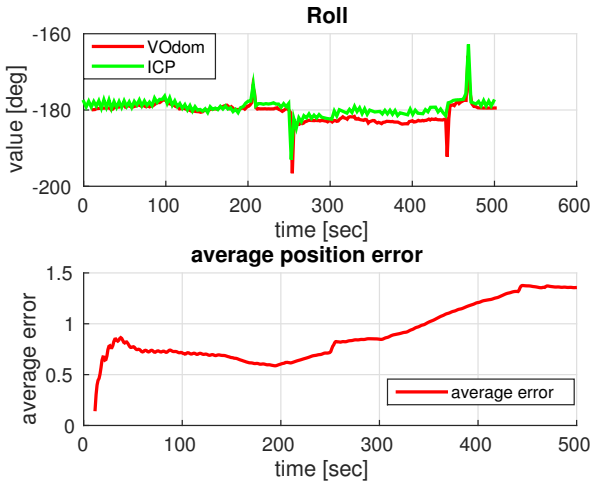


Figure 5.24: The Street experiment: First graph shows the progress of roll rotation in relation to time in degrees. The second graph shows the average error in degrees.

5.2.4 The Forest experiment

The conception and description of this experiment were described above, see chapter 3.3. The data captured in Krc forest was tested again however by modified algorithm. The previous algorithm totally failed in this scene. Whereas the modified algorithm worked better. From the graph is clearly shown that at the beginning it perfectly imitates the reference trajectory. However, approximately in the 400 second the data from camera suddenly stop coming. The data then continue coming in 1200 second. In the graphs it is visible according to the constant function value. The reason of this failure is unknown and it is a possible way of further research. One of the possible reasons is that the date have been already lost in virtual camera. However, we can definitely say that in the time when the data come to the algorithm from the camera, the algorithm works well. Although in all data is a small mistake, the pairing of enough key-points is successful and then the creation of sufficient amount of landmarks is also well done. The course of robot's position and the error is shown in the figures: 5.25, 5.26, 5.27 and 5.28. The course of rotation and its error is shown in the figures: 5.29, 5.30 and 5.31.

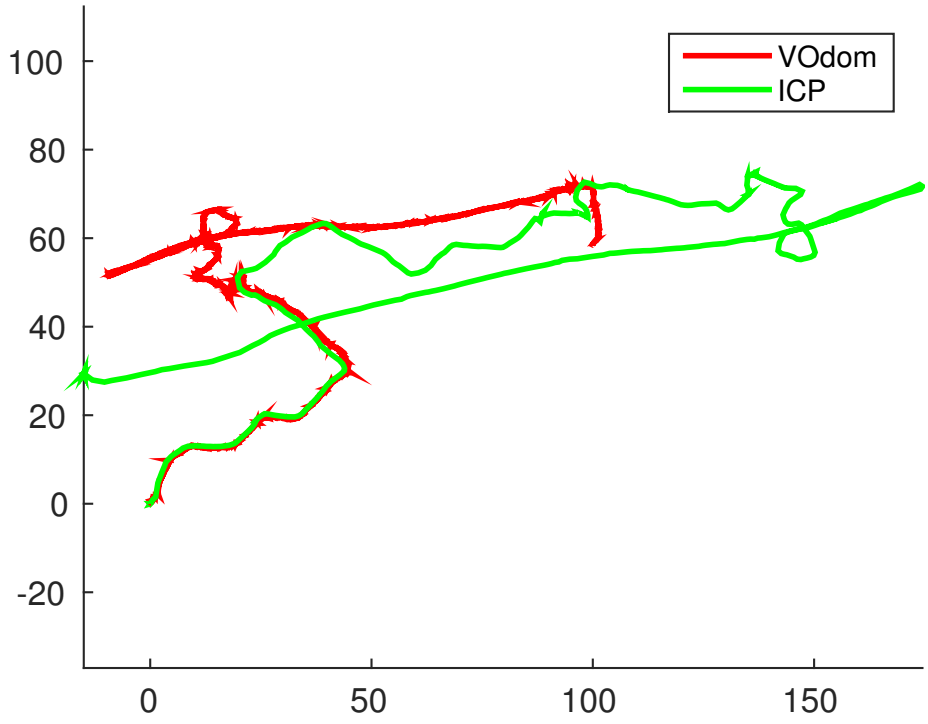


Figure 5.25: The Forest experiment: Graph shows the trajectory from the top in meters.

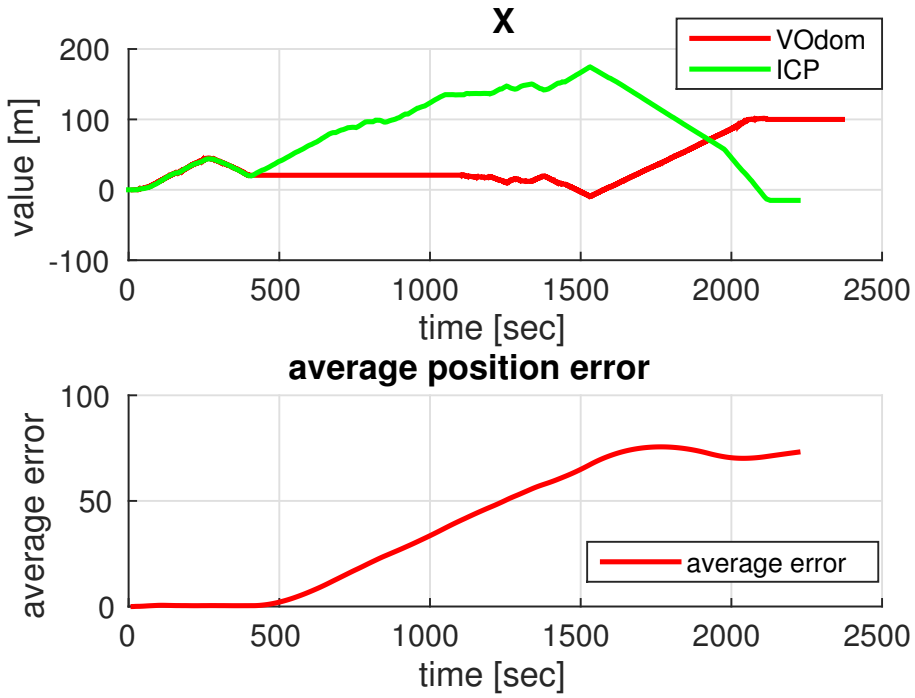


Figure 5.26: The Forest experiment: First graph shows the trajectory in the direction of x-axis in meters. The second graph shows the average error in meters.

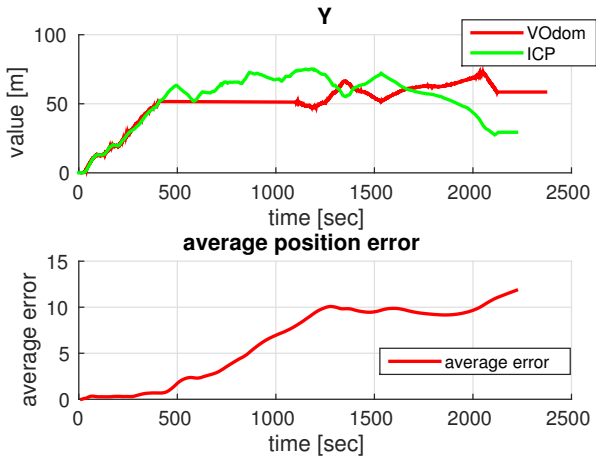


Figure 5.27: The Forest experiment: First graph shows the trajectory in the direction of y-axis in meters. The second graph shows the average error in meters.

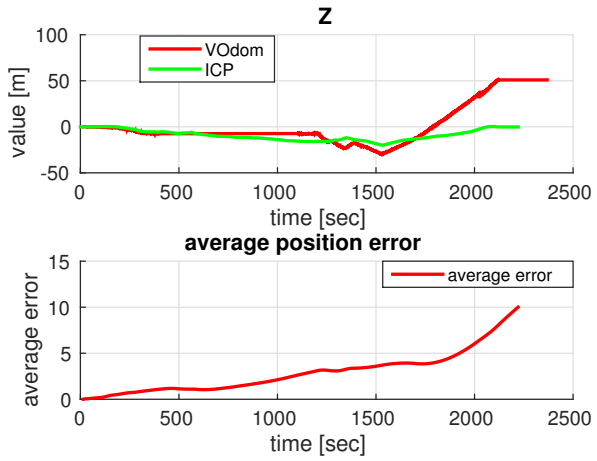


Figure 5.28: The Forest experiment: First graph shows the trajectory in the direction of z-axis in meters. The second graph shows the average error in meters.

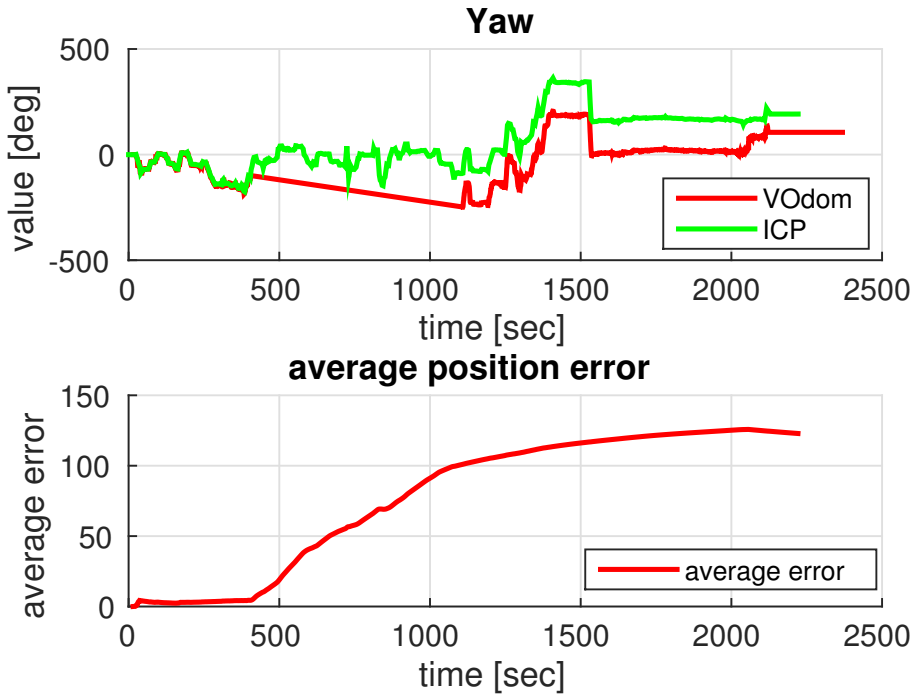


Figure 5.29: The Forest experiment: First graph shows the progress of yaw rotation in relation to time in degrees. The second graph shows the average error in degrees.

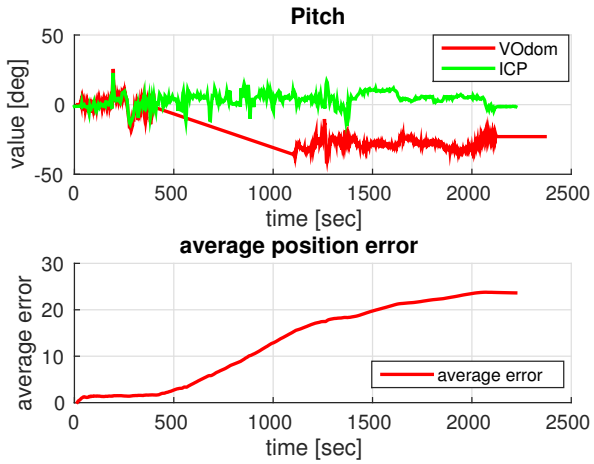


Figure 5.30: The Forest experiment: First graph shows the progress of pitch rotation in relation to time in degrees. The second graph shows the average error in degrees.

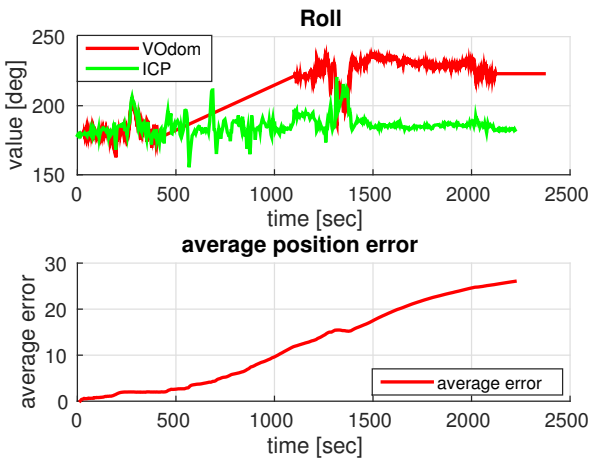


Figure 5.31: The Forest experiment: First graph shows the progress of roll rotation in relation to time in degrees. The second graph shows the average error in degrees.

5.2.5 The Italy experiment

Other new experiments were recorded in Italy at the end of September 2014. This experiment was situated near by a ruined hospital where the robot had driven in debris and thanks to this fact it had had to overcome various obstacles. In contrast with the previous experiments the robot did not drive mostly along the horizontal surface but it was variously tilted or inclined. Generally, it is a very difficult task for visual odometry.

Form the beginning the experiment goes well and there is only small deviation from ICP. In the second third of the experiment the robot goes down a lot. In this moment (see Figure 5.32) the many of stable landmarks, which keep the global scale, are lost. Therefore the estimation of angle is still good but the estimation of position fails. The reason of losing stable landmarks

- Thanks to change in robot configuration for crossing the terrain a lot of correspondences on the robot's body is not found.
- There is not any interesting point on the left side of the image, where the key-points can be found.
- The robot is quite inclined to forward and therefore a lot of important points are in the upper half of the image. Here are the pixels deformed a lot by the spherical camera approximation. Thus there is a big error in estimation of landmark position and therefore the landmark disappears.

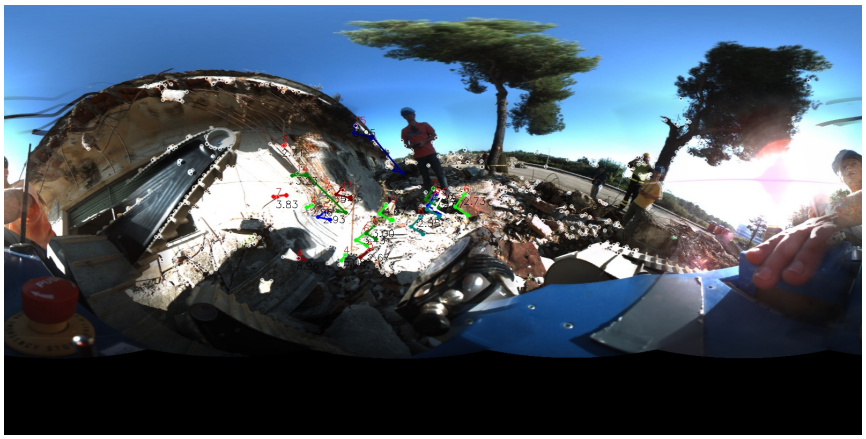


Figure 5.32: The Italy experiment: Picture from the ladybug3 recorded during the experiment. The image is from place where algorithm failed. In the picture there are shown some key-points, which belongs to landmark. The red number is number of previous pictures, where exist correspondences to this keypoint. The black number is the landmark's apex angle.

The course of robot's position and the error is shown in the figures: 5.25, 5.26, 5.27 and 5.28 The course of rotation and its error is shown in the figures: 5.29, 5.30 and 5.31.

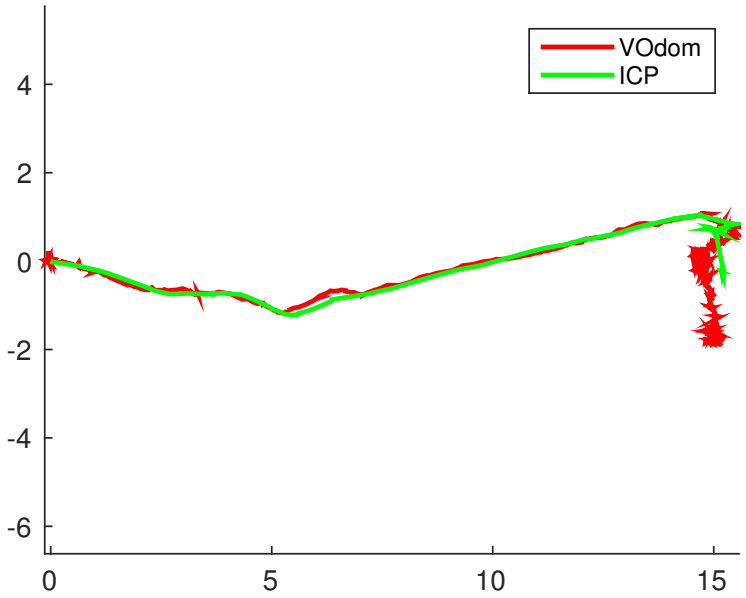


Figure 5.33: The Italy experiment: Graph shows the trajectory from the top in meters.

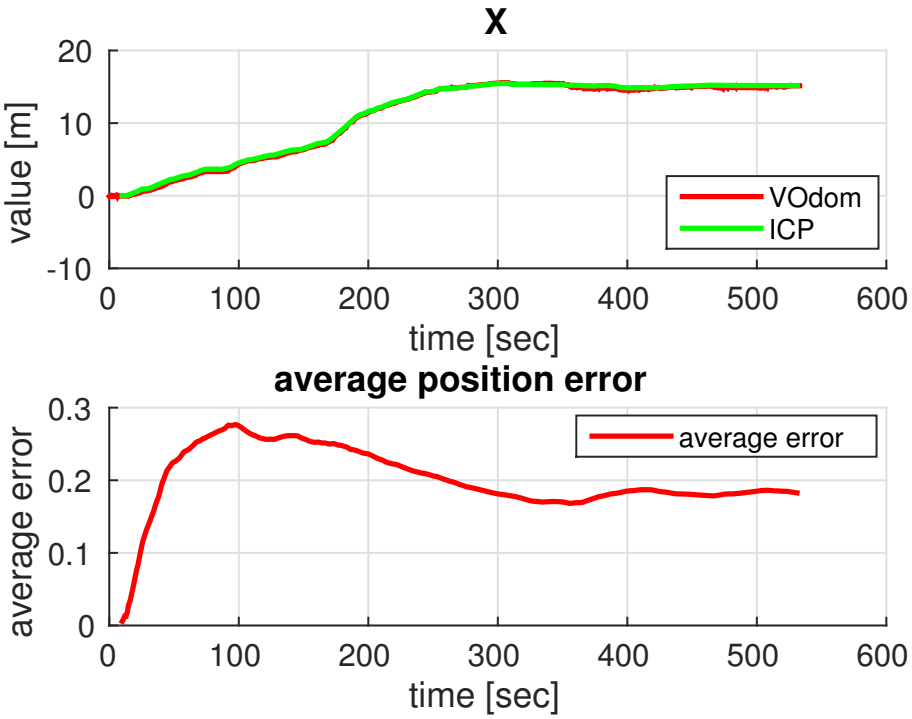


Figure 5.34: The Italy experiment: First graph shows the trajectory in the direction of x-axis in meters. The second graph shows the average error in meters.

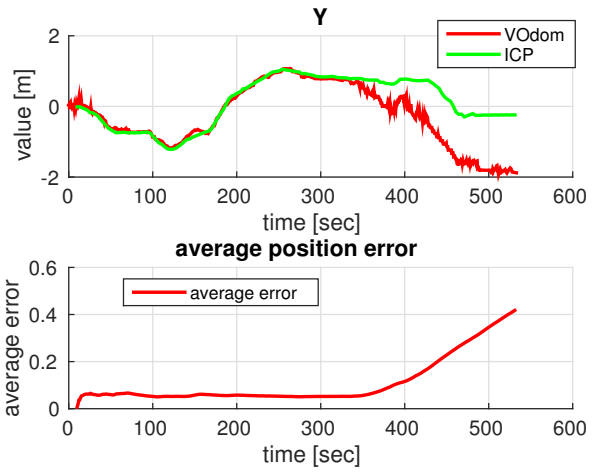


Figure 5.35: The Italy experiment: First graph shows the trajectory in the direction of y-axis in meters. The second graph shows the average error in meters.

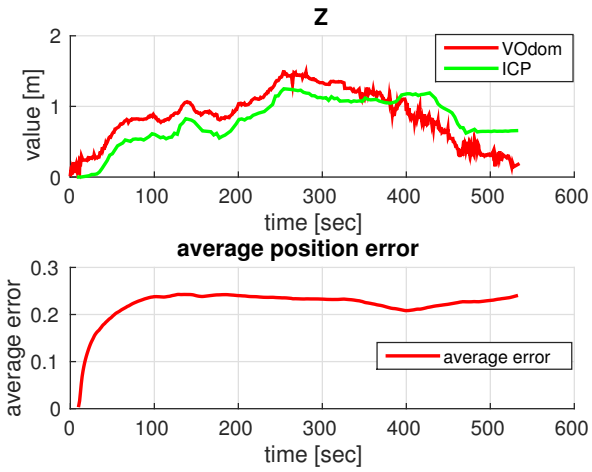


Figure 5.36: The Italy experiment: First graph shows the trajectory in the direction of z-axis in meters. The second graph shows the average error in meters.

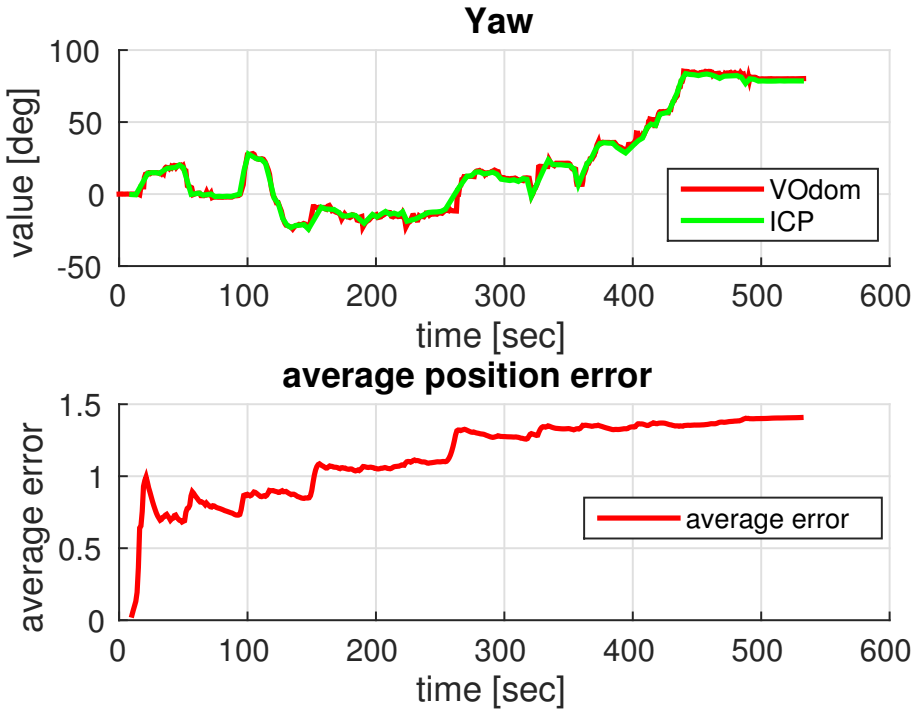


Figure 5.37: The Italy experiment: First graph shows the progress of yaw rotation in relation to time in degrees. The second graph shows the average error in degrees.

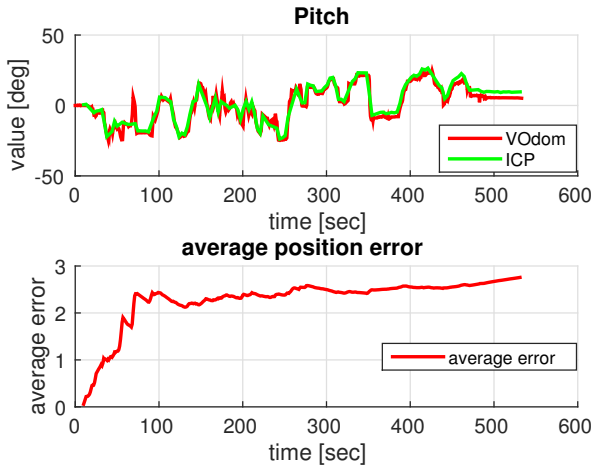


Figure 5.38: The Italy experiment: First graph shows the progress of pitch rotation in relation to time in degrees. The second graph shows the average error in degrees.

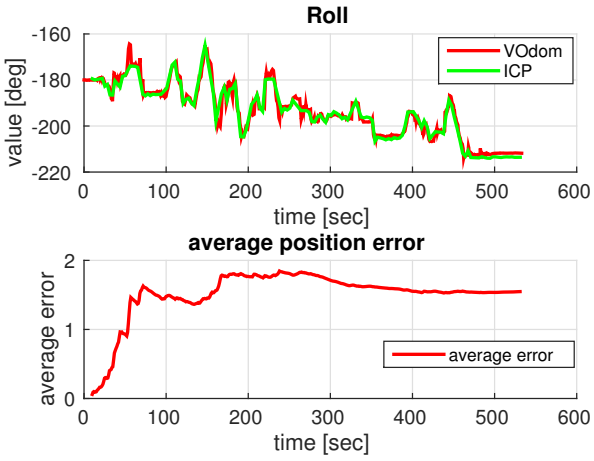


Figure 5.39: The Italy experiment: First graph shows the progress of roll rotation in relation to time in degrees. The second graph shows the average error in degrees.

6. Conclusion

The aim of this thesis was to test and improve the algorithm created by Jiri Divis [4]. At the beginning the original algorithm was intensively tested. This way the main imperfections was uncovered and the algorithm was then improved. The improvements are described in chapter 4. The biggest improvement was achieved by enhancement of key-points pairing which is described in chapter 4.4. In the algorithm was also repaired a several bugs which also had a positive influence on the improved algorithm. After the modifications the algorithm was tested in various types of environments in where the USAR robot can get. The best results of the algorithm was observed in an outdoor environments where there was sufficient amount of a visible edges in the scene. These are environments where is plenty of buildings which has enough well-marked features, e.g. windows, doors or interesting facade. On the buildings can be found enough quality key-points and these can be followed also in other images. In the room we encountered a problem with the lack of lighting, this has led to recording of blurry images. Hence, we added a detection of blurry images to the algorithm under which the blurry images were deleted. Due to the testing in chapter 5 we found out that the improved algorithm works better than the previous one.

Bibliography

[1] Igor Aizenberg, Dmitriy V Paliy, Jacek M Zurada, and Jaakko T Astola. Blur identification by multilayer neural network based on multivalued neurons. *Neural Networks, IEEE Transactions on*, 19(5):883–898, 2008.

[2] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992.

[3] Dmitry Chetverikov, Dmitry Svirko, Dmitry Stepanov, and Pavel Krsek. The trimmed iterative closest point algorithm. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 3, pages 545–548. IEEE, 2002.

[4] Jiri Divis. Visual odometry from omnidirectional camera. Master's thesis, Charles University in Prague, Faculty of Mathematics and Physics, 2013.

[5] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

[6] Vladimír Kubelka, Lorenz Oswald, François Pomerleau, Francis Colas, Tomáš Svoboda, and Michal Reinstein. Robust data fusion of multi-modal sensory information for mobile robots. *Journal of Field Robotics*, 2014.

[7] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 850–855. IEEE, 2006.

[8] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652. IEEE, 2004.

[9] Tomas Nouza. Visual compass for the nifti robot. Technical report, Center for machine perception - CTU FEE, 2013.

[10] openCV. Discrete fourier transform, 2012.

[11] openCV. ORB oriented fast and rotated brief, 2012.

[12] Inc. Point Grey Research. Ladybug3 (1394b), 2012.

[13] ROS. Ros, 2012.

[14] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.

[15] Jakub Simanek, Michal Reinstein, and Vladimir Kubelka. Evaluation of the ekf-based estimation architectures for data fusion in mobile robots. 2014.

CD content

Attached CD contains a text of this master thesis in the PDF format and source codes of the whole text for the LaTeX. The directory tree is in the next table:

Table 1: Directory tree of the attached CD

Directory	Label
`LaTeX`	source codes for the text of the thesis
↪ `src`	directory containing individual chapters
↪ `fig`	directory with used images
↪ `Makefile`	Makefile for building the thesis
`thesis.pdf`	CD version of the thesis