# Visual Odometry Dataset

Intermediate report

Alex Kreimer, Vadim Indelman

## 1 TOC

1 TOC
2 Introduction
  2.1 Related Work
  2.2 Summary of our work
3 Data Acquisition
  3.1 Overview
  3.2 Physical Setup
    3.2.1 The cameras
    3.2.2 The GPS
  3.3 Software
  3.4 Experiment Day
4 Post-processing
  4.1 Image sequences
  4.2 GPS data
  4.3 Missing GPS Data
5 Example Sequence
  5.1 Compare VO vs. DGPS data
6 Appendix A: Example Images
7 Appendix B: Example GPS sequence compared to VO data
8 References

## 2 Introduction

Visual odometry refers to the problem of recovering camera motion based on the images taken by it. This problem naturally occurs in robotics, wearable computing, augmented reality and automotive.

Wheel odometry, recovers the motion of the vehicle by examining and integrating the wheel turns over time. In similar manner, visual odometry operates by estimating relative motion of the camera between subsequent images by observing changes in them. Later, these estimates are combined into a single trajectory. Just as wheel odometry, visual odometry is subject to error accumulation over time. Contrary to wheel odometry, visual odometry is not affected by wheel slip in rough terrain. Visual odometry is able to produce motion estimates with errors that are lower than those of the wheel odometry. Another advantage of visual odometry is that cameras are low cost and low weight sensors. All these make visual odometry a viable supplement to other motion recovery methods such as global positioning systems (GPS) and inertial measurement units (IMUs).

Visual odometry becomes a harder problem as the amount of detail in the images diminishes. The images should have sufficient overlap and the scene needs to be illuminated. In stereo setup, the scene must be static or the images taken at the same time. Also, video processing incurs computational burden.

## 2.1 Related Work

An essential tool for VO algorithm evaluation is a comprehensive dataset. There is a number of publically available datasets in the literature (e.g., KITTI [1], Malaga [2]). Each one has its own strengths and weaknesses. KITTI is a de-facto standard, e.g., there is no recent visual odometry work that does not provide its results on the KITTI dataset.

KITTI is a great quality dataset. Said that, it is a solved problem in some sense. The errors of the state of the art algorithms on this dataset are low and it may happen that the algorithms are over-fit to this specific dataset. On the other hand, there are cases that are not covered by KITTI. No traffic scenes, where significant part of the field of view is obscured by another vehicle, the lighting conditions are perfect and the scenes are texture rich.

## 2.2 Summary of our work

In this work we build a dataset that contains a number of sequences that may be used to benchmark visual odometry algorithms. We install synchronized stereo pair along a (D)GPS receiver on a car that travels in an urban as well as rural areas. Our goal is to cover some challenging conditions that happen in day-to-day driving, e.g., field of view occlusions, poor lighting conditions and low texture areas.

Note, that budget constraints led us to a simplified system that is capable of producing 3DOF data (i.e., location only) being unable to track vehicle orientation.

# 3 Data Acquisition

## 3.1 Overview

Our system consist of a synchronized stereo pair and a GPS receiver synced to the cameras. During the recording process GPS device produces a clock signal of 10 Hz that triggers the cameras. The data is written to PC that is connected to both the cameras and the GPS.

## 3.2 Physical Setup

### 3.2.1 The cameras

We use a pair of 3Mp Ethernet IDS-Imaging cameras UI-5240SE. Hardware synchronization is made by external vendor and uses built-in camera-flash sync option. It allows us to capture images at about the same time (empirically measured time delta is around single digit in ms). Low skew is essential for the stereo algorithms. If the time delta is significant, static scene assumption breaks rendering stereo pair useless. The sync quality depends on the vehicle velocity as well. We built a system suitable for regular driver usecase.

Another rig parameter is the baseline (i.e., distance between the cameras). We tend to be close to KITTI setup in this sense.

Another area of concern is the amount of data such stereo rig produces. We shoot at 10 FPS which makes it about 3*2*10 = 60 Mb/s. On top of this the system needs to work at a moving vehicle, which rules out non-SSD hard drives.

The cameras are fixed focus and auto-exposed. Auto exposure is critical for outside scenes, especially on a high-contrast sunny day.

### 3.2.2 The GPS

Consumer grade GPS measurements expected accuracy are about 3-5 m. This is not the same order of magnitude as state of the art visual odometry algorithms. On the other hand VO suffers from dead-reckoning (e.g. error accumulation) while GPS error is absolute and constant. In our work we use Differential GPS correction to improve the expected accuracy of the GPS measurements. An enabler for such correction is the raw GPS data (e.g., pseudoranges, satellites availability, signal strengths, etc.)

After evaluation we chose UBLOX Neo-7P as an inexpensive device that suites our needs.

It is able to both produce the raw measurement data and to trigger the cameras.  The schematic drawing of the system is presented in Figure 2.  UBLOX is connected to the PC through USB and to the cameras by means of electric wire.  It triggers the cameras at 10 Hz and produces GPS measurements synced to the images at 1 Hz.

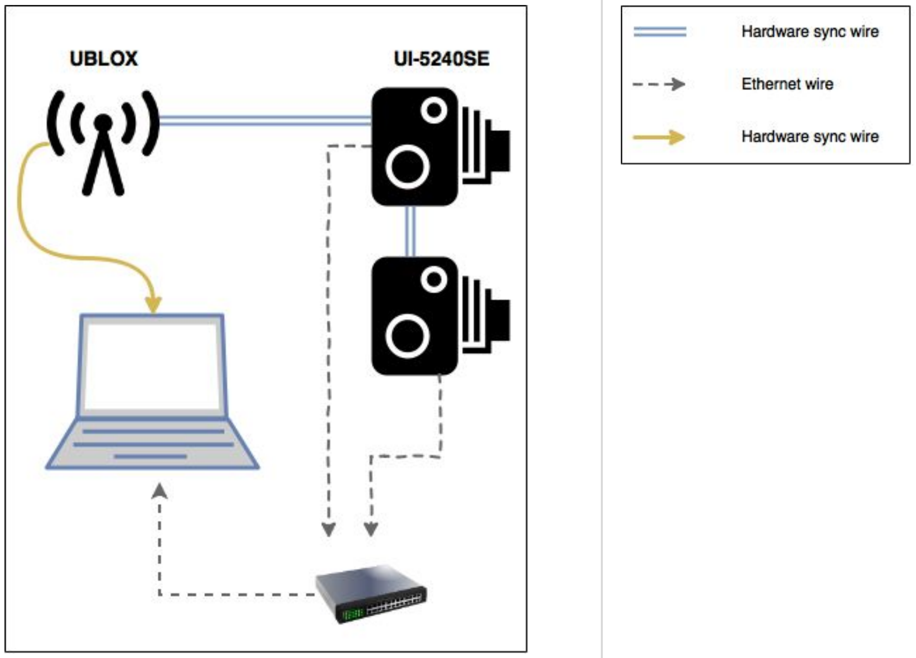


Figure 2.  Diagram of physical setup.

## 3.3 Software

Our system uses a number of software packages for data recording and post-processing.

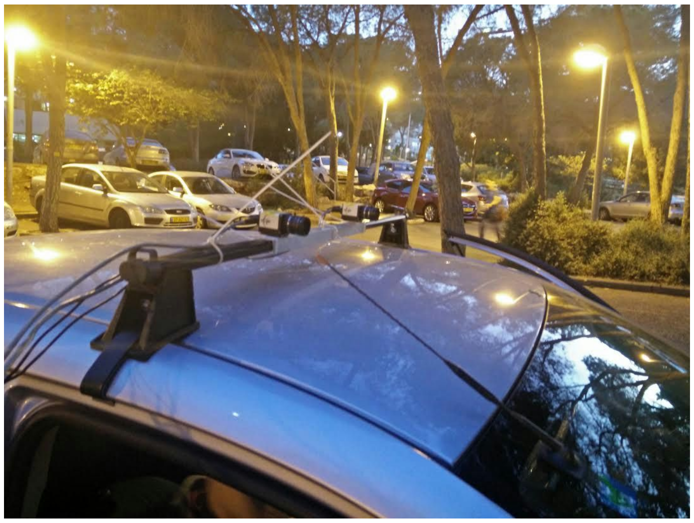
We use IDS Imaging software suite to record the image sequences.  The sequences recorded as video files at 10 FPS.  This software allows us to track the quality of the recording, e.g., if there are lost frames, which improves the quality of the data.

We use goGPS open source positioning software to record GPS data.  Our setup is a bit unusual and thus it is rather hard to find out of the box software that is capable to synchronize the GPS and the cameras.  We use MATLAB version of the software. We modified out of the box version to communicate with UBLOX device.  The modification we made allows us to tell UBLOX to issue a sync pulse to the cameras, which is essential to us.

## 3.4 Experiment Day

We install the camera rig and the GPS antenna on the roof our lab vehicle.  The whole system is powered by the vehicle.  At the beginning of each day we re-calibrate the stereo rig to make sure the calibration data is correct.

We usually record a few minute length video tracks along with the GPS tracks.  The data is stored on the PC for further post-processing.



# 4 Post-processing

## 4.1 Image sequences

At this stage we:

1. Compute the calibration data from the calibration board images
2. Split video files to produce image sequences
3. Rectify image sequences.

We use OpenCV calibration toolbox for these tasks. We bundle this code with the project.

## 4.2 GPS data

First we performat DGPS correction. The idea of this correction is simple: use a known-location base-station to correct the vehicle position measurement. While the idea is simple, there are a lot of technicalities (e.g., which satellites should be taken into consideration and what the weight of their pseudo-ranges should be in the computation). This may be done in real-time or offline. Budget constraint did not allow us to obtain equipment that is capable of real-time correction, thus we do it offline.

We obtain Israel Mapping Center data for the experiment day. There is a base-station on Technion campus. This is important, since the reliability of the correction decays as the distance to the base-station. It is common wisdom, that the correction is reliable for <10 Km baselines (e.g. the distance between the vehicle and the base-station). We tried to respect this rule when doing our experiments.

All the correction heavy-lifting is done by the RTKLib software package. RTKLib correction outputs measurement timestamp, WGS84 location and its covariance, along with some additional data. The example is in the figure below:

```
% (x/y/z-ecef=WGS84,Q=1:fix,2:float,3:sbas,4:dgps,5:single,6:ppp,ns=# of satellites)
GPST_date GPST_time     x-ecef(m)      y-ecef(m)         z-ecef(m)     Q  ns  sdx(m)  sdy(m)  sdz(m)  sdxy(m) sdyz(m) sdzx(m)
2015/12/07 10:06:04.000  4396368.9557  3080094.7655  3433532.7184  2  9  1.5245  1.4641  1.0787  1.4039  1.1152  1.1351
2015/12/07 10:06:05.000  4396368.9519  3080094.9203  3433532.9827  2  9  1.0839  1.0418  0.7670  0.9984  0.7932  0.8070
2015/12/07 10:06:06.000  4396368.8692  3080094.8948  3433532.5777  2  9  0.8863  0.8523  0.6272  0.8165  0.6486  0.6598
```

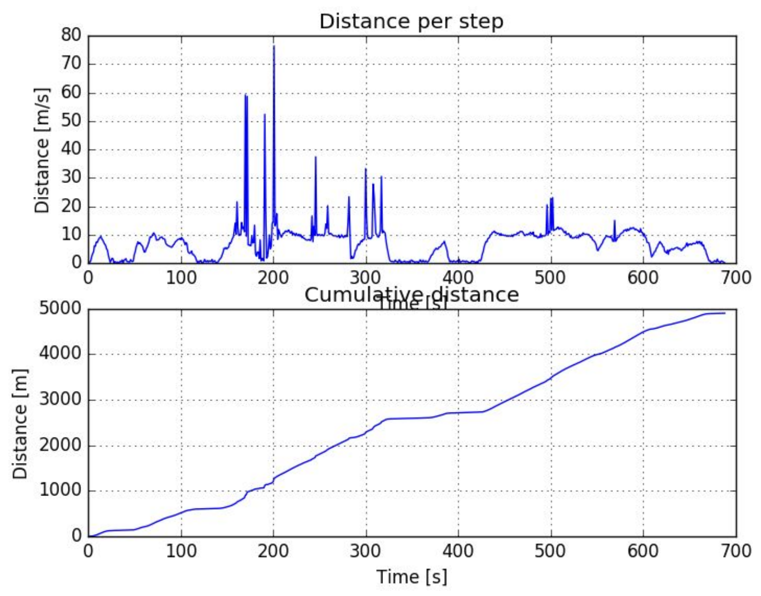
## 4.3 Missing GPS Data

RTKlib may disregard a certain measurement because of its quality (e.g., if RTKLib is sure that the measurement is wrong). In this case the file will have a gap in time stamps. We provide missing data information as a part of the final dataset.

## 5 Example Sequence

Below we provide graphs of a sample sequence from our dataset. The track's length is 4901.99 m and it took 691 s to make (i.e., about 11.52 minutes), which is an average of 7.1 m/s or 25.56 km/h.

As one may see, despite the correction, there are some noisy measurements. Kalman Filter in RTKLib should have fixed these issues, we are investigating them.



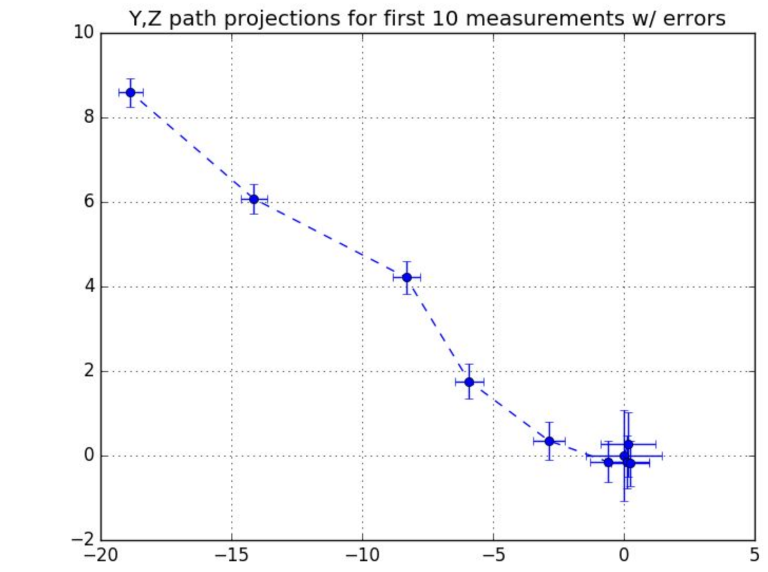


Figure: GPS measurements with corresponding uncertainties. One may see that the first points have higher uncertainty, probably due to the bootstrap of the GPS receiver.

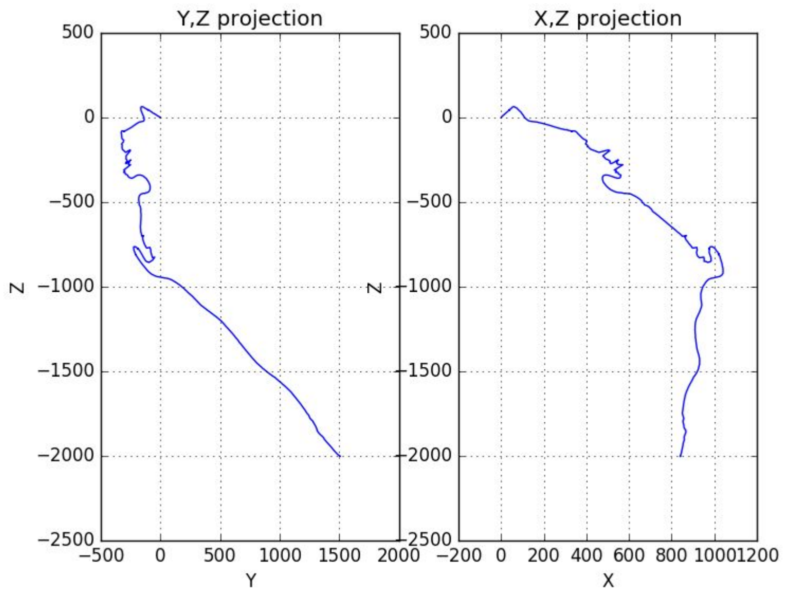


Figure: Vehicle path projected to the X,Z and Y,Z planes

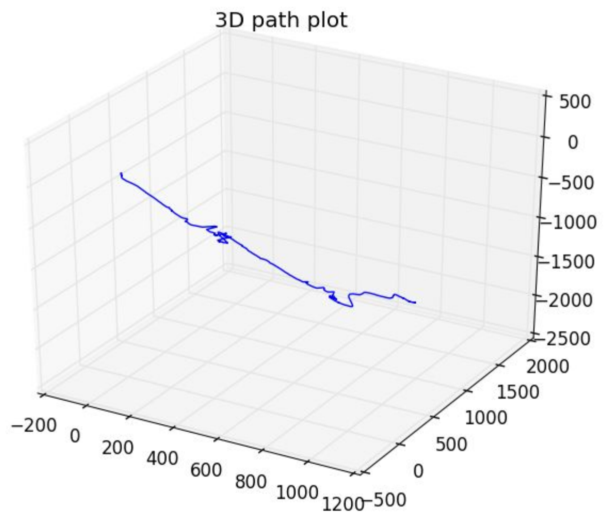


Figure: Vehicle path plotted in 3d

## 5.1 Compare VO vs. DGPS data

There is a number of ways to compare VO and DGPS data, but all of the assume that both the GPS track and the VO track are in the same reference frame, which is not the case with our data.

While GPS locations are provided in Earth Centered WGS84 coordinate frame, Visual Odometry data is usually computed relative to the initial position/orientation of the camera. Thus, before the results may be compared to the ground-truth they need to be represented in the same coordinate frame.

To represent both tracks in the same reference frame we use the following procedure:

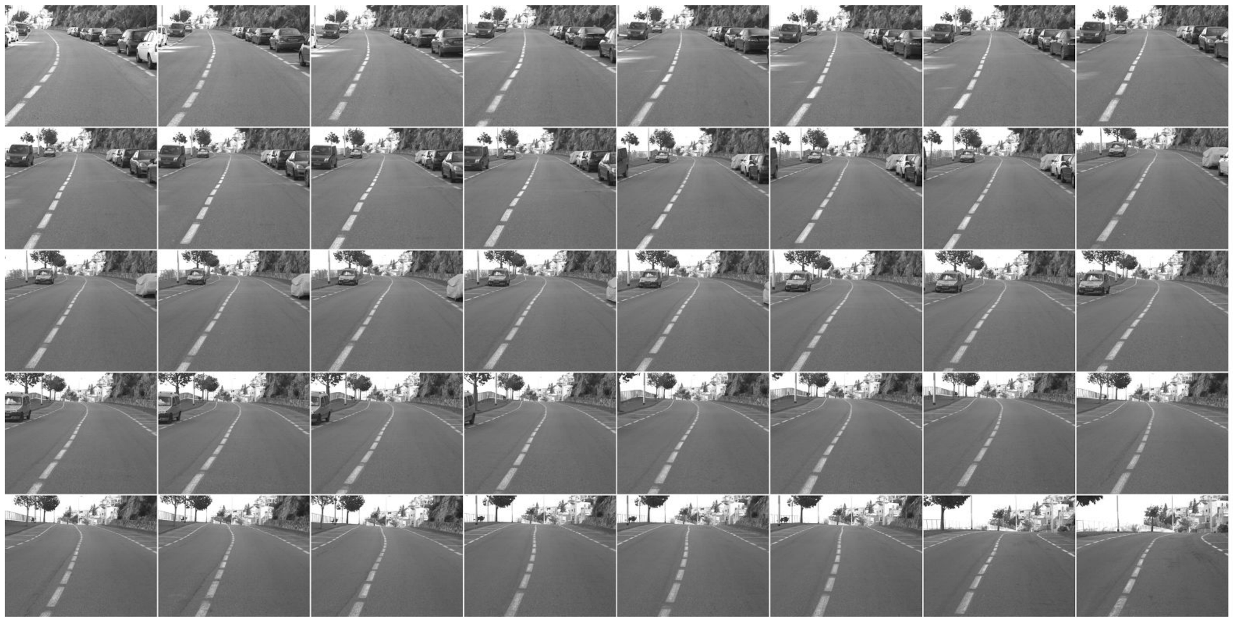
1. Subtract the location of first GPS measurement from all GPS measurements. This effectively brings the origin of the GPS measurements into the location of the first camera instead of being earth centered.
2. Rotate the VO track to coincide with the GPS data.

To compute the rotation we propose the following procedure: let $A$ be the matrix with VO measurements stacked as its rows, and let $B$ be the matrices with GPS measurements stacked as its rows. We search for the rotation matrix $R$ , s.t.

$$argmin \sum_i \|RA(:,i) - B(:,i)\|^2$$

This optimization problem is called absolute orientation problem in the literature and has an SVD and quaternion based solutions (see e.g., [1] for reference).

## 6 Appendix A: Example Images



## 7 Appendix B: Example GPS sequence compared to VO data

We run one off-the shelf VO algorithm (StereoScan) on our dataset and compare the results. Please find the graphs below:

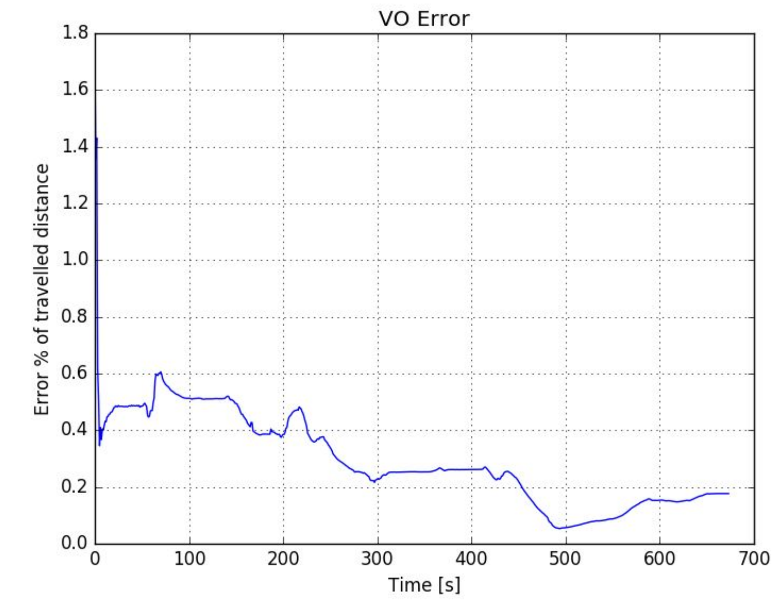


Figure: VO vs. GPS data error. The error is computed on a per-measurement basis as a relative part of path travelled. The errors are significantly higher than those on the KITTI dataset.

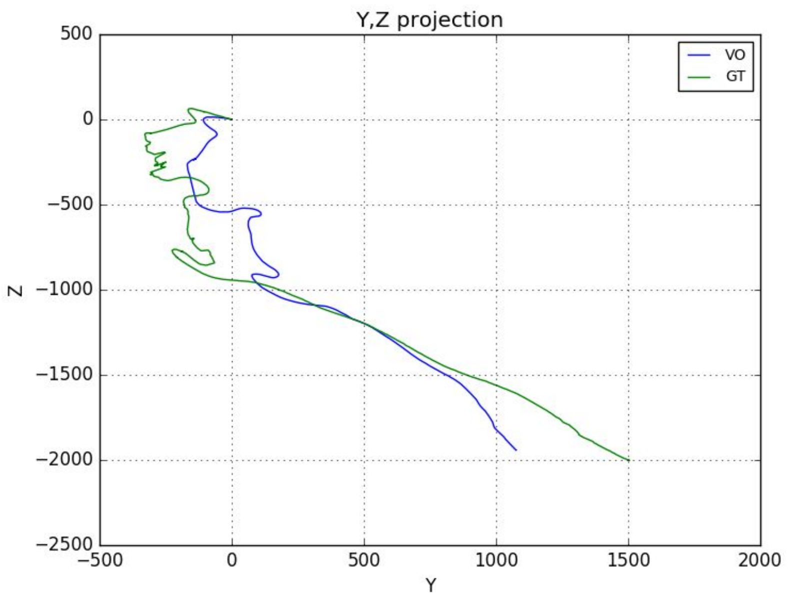


Figure: GPS vs. VO paths overlaid on the same graph

As the graphs suggests the errors of the VO are significantly higher than those that are reported on KITTI. There may be a number of reasons for this behavior:

1. The datasets are different and the algorithm that performs well on KITTI may perform poor on different DATA

2. The error measurement method is different.  KITTI measures pose error for subsequences of various lengths, which lessens the effect of error timing.
3. Our dataset GPS data have erroneous measurements

## 8 References

| | |
|---|---|
| [1] | A. Geiger, P. Lenz and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in Conference on Computer Vision and Pattern Recognition (CVPR), 2012. |
| [2] | J.-L. F.-A. M. a. J. G. Blanco, "A collection of outdoor robotic datasets with centimeter-accuracy ground truth," in Autonomous Robots , 2009, pp. 327-351. |
| [3] | Horn, Berthold KP. "Closed-form solution of absolute orientation using unit quaternions." *JOSA A* 4.4 (1987): 629-642. |