Project Report

Visual Odometry Pipeline

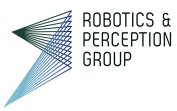
Marius Grimm, Amadeus Oertel, Toni Rosiñol

Titus Cieslewski, Henri Rebecq
Adviser

Prof. Dr. Davide Scaramuzza
Professor

Robotics and Perception Group
University of Zurich & Swiss Federal Institute of Technology Zurich (ETH)

2017-01



Contents

List of Figures

Figure 1: 2D-2D Correspondences with KLT.

1 Introduction

2 Initialisation

2.1 Monocular Initialisation

The monocular initialisation is a key module in the Visual-Odometry pipeline. It is ordered in the follwoing way:

1. Find the 2D-2D correspondences between the the chosen first two images
2. Apply the 8-point Algorithm to estimate the Fundamental matrix combined with running RANSAC.
3. Check the validity of the estimated Fundamental matrix by either comparing the reprojection error or the distance to the epipolar line.
4. RANSAC return a set of inlier of the current 2D-2D correspondences.
5. With the new set of inlier we can re-evaluate the final Essential Matrix E estimate.
6. The Essential Matrix E can then be decomposed into two rotation and two translation hypotheses for the pose of the first camera frame. This gives in total a set of four camera motion possibilities.
7. These hypotheses have to be disambiguated to choose the right camera rotation and translation.

We have implemented two different approaches for finding the 2D-2D correspondences:

1. Exercise implementation of Harris descriptor and detector.
2. Implementation of the Kanade-Lucas Tracker (KLT).

An examplary output of the 2D-2D module can be seen in Figure **??**. The correspondences between two images are indicated.

Due to efficiency reasons we decided to use the epipolar line distance as a measure to validate the estimated Fundamental Matrices. In theory the reprojection error would have yielded better results (sacrificing efficiency); however, in our case the differences were neglectable.

3 Continuous Operation

3.1 KLT

4 Bonus Features

4.1 Plotting

The source code provides extensive plottings to help debug the pipeline. These are enabled by using the *debug_with_figures* flags on the different modules of the pipeline. Moreover, in order to properly visualize the logic of the pipeline, we show a Figure with the main output of the odometry estimation. At the top left of the mentioned Figure, it is shown the current frame that is being processed. On top of it we plot the keypoints triangulated in green, while in yellow we plot the ones that have not yet been triangulated but are potentially going to be triangulated. The figures below the image present both the number of landmarks that are being tracked (left image) and the global trajectory estimated by the algorithm. Finally, the figure on the right shows the last poses of the camera together with the last triangulated landmarks.

4.2 Automatic Selection of Frames for Initialisation

We have further implemented the Bonus Feature "Automatic Selection of Frames for Initialisation". The initialisation will try to find correspondences between different frames in the beginning of the datasets. If a pair of initialisation candidates has enough correspondence inliers the automatic selection tries to find the best pair which minimizes:

1. The average reprojection error and
2. the average epipolar line distance.

Once the minimizing pair was found, the monocular initialisation is recomputed with the new initialisation set of images.

4.3 Relocalisation

4.4 Full Bundle Adjustment

4.5 Calibrated Smartphone Camera and Own Dataset

In order to assess the robustness of our VO pipeline, we decided to record our own dataset. For this endeavour, we printed a checkerboard to calibrate the camera of an smartphone. We then used the calibrated camera to record the dataset. In order to achieve good results for the calibration, we made sure that we followed the following procedure:

1) Set focus and exposure mode of the smartphone to fixed. 2) Ensure that the checkerboard is over a planar surface and presents no light reflections. 3) Take frames of the checkerboard from sufficiently different angles and distances.

Once the calibration dataset recorded, we proceeded to find the intrinsics of the camera. As a first approach we used the Matlab Toolbox. Unfortunately, we could not get good calibration results. This was in part due to the fact that the interface requires you to click on the corners of the checkerboard in the image, and therefore limits the amount of images you can process in a given time. We also used OpenCV to retrieve the calibration matrix of the camera. Nonetheless, we found that the most user-friendly calibration tool was the one offered as a Matlab app named Camera Calibrator.

Using this last tool, we managed to retrieve approximately the same intrinsic parameters independently of the calibration images. Once calibrated, we processed the images to get a gray-scaled, resized and undistorted set of images. Then we fed the new dataset to our VO pipeline and checked the results with our ground truth. The ground truth was taken approximately with no special instrumentation other than a meter.

We encountered many problems while recording the dataset, to name a few: motion blur when moving, changes of illumination on the scene, maintaining a constant focus, transferring images, having enough features on the scene, etc.

In the end, we found that monocular initialisation was having difficulties to output a sufficient amount of 2D-3D correspondences. This led the continuous operation part to stop after two frames on average. The reprojection errors on the triangulated keypoints were of the order of magnitude of tens of pixels, depending on parameters and bootstrap_frames.

In the end, we learned about different tools available for camera calibration and about the difficulties of recording a satisfiable dataset for visual odometry. Or seen from another perspective, we learned about the limits of our VO pipeline.

Our calibration files and images of the dataset are provided under the folder smartphone.

4.6 Quantitative Analysis

4.6.1 Keypoint Tracking via Block Matching versus KLT

4.6.2 Analysis of performance of different ways to extract Harris features

Extracting keypoints from the images is one of the most important parts of a visual odometry pipeline. For that reason, we decided to do a quantitative analysis of the performance of the way we used the Harris detector. In our implementation we added four different methods to apply Harris. The first consists in an implementation that we believed could improve the performance of the keypoints extraction. While running the pipeline we noticed that the keypoints were not uniformly detected over the image. In order to get a more uniform set of keypoints over the image we decided to divide the frame into sub-images. We will refer to them as bins. In each of these bins we run the Matlab implementation of "detectHarrisFeatures".

Figure 2: Uniform Harris Implementation.

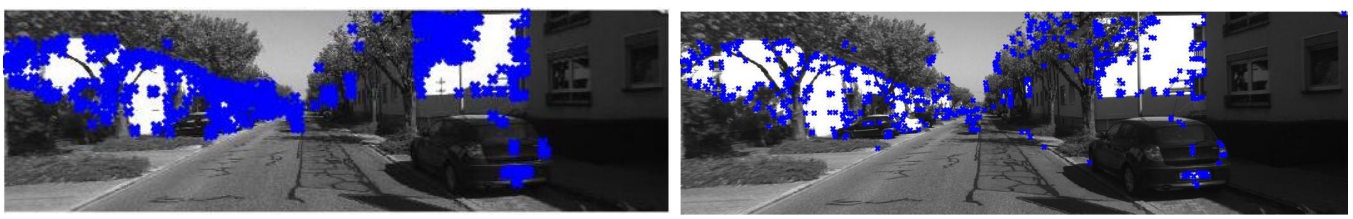
This provides us with localy optimal corners in the bin. Applying this approach recursevely over the bins gives us an homogeneous extraction of features of the image. Figure 3(b) presents the result of this implementation on the first frame of the Kitti dataset.

The other two algorithms that we may use consist in running the Harris corner detector over the whole image, as it is usually done. They differ between them in that one selects the strongest detected keypoints while the other selects them more uniformly. As it can be seen in figure 3(a) and figure **??**, the uniform selection of keypoints performed by Matlab is still not homogeneous over the image. This should not be a problem if the quality of the corners is good. Nonetheless, we experienced in practice that a mapping of keypoints on different patches of the image gives overall better results and is more robust to occlusions.

4.7 Monocular Initialisation Harris Detector via Matlab versus Exercise Harris Detector

5 Conclusion

Main conclusions and final remarks.



(a) Strongest Harris Features.

(b) Uniform Harris Features.

Figure 3: Harris Features.

References

[1] J.-L. Blanco, F.-A. Moreno, and J. González-Jiménez. The málaga urban dataset: High-rate stereo and lidars in a realistic urban scenario. *International Journal of Robotics Research*, 33(2):207–214, 2014.

[2] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[3] D. Scaramuzza and F. Fraundorfer. Visual odometry: Part i - the first 30 years and fundamentals. *IEEE Robotics and Automation Magazine*, 18(4), 2011.

[4] D. Scaramuzza and F. Fraundorfer. Visual odometry: Part ii - matching, robustness, optimization, and applications. *IEEE Robotics and Automation Magazine*, 19(2), 2012.

A Appendix

Additional material such as long mathematical derivations.