Feature-based Monocular Visual Odometry

Jay Li

***Abstract*—This project investigates monocular visual odometry, which estimates the trajectory of a camera based only on the video it takes. Generally, there exist three classes of methods for solving this problem, which are feature-based methods, direct methods, and hybrids of the two. This project implements a feature-based method, and discusses the issues involved in scale estimation and the relation to step size.**

I. Introduction

Visual odometry (VO) concerns the problem of incrementally estimating the trajectory of the camera based on the images it takes consecutively. The term was first introduced by Nister in an eponymous paper in 2004 [1], and bears the similarity to an older term *wheel odometry*, which estimates the trajectory of a robot from the number of turns of the wheels. In the case of ground vehicles, an advantage of visual odometry over wheel odometry is that it avoids the problem of wheel slip in uneven terrains, which was an early motivator for its development for NASA Mars exploration program [2]. Other ways of estimating the trajectory include using global positioning system (GPS), inertial measurement units (IMUs), and laser scans.

In the case of visual odometry, it is usually separated into two types, the stereo VO, and monocular VO. Stereo VO uses multiple, typically two, cameras, fixed relative to each other. The 3D positions of the points in the image can be triangulated from the multiple cameras, and thereby giving a direct measurement. Whereas for monocular VO, which uses a single camera, only the directions of the points are measured. Due to this reason, monocular VO can estimate trajectory only up to a scale factor. However, when the scene is far away, stereo VO will degenerate to monocular VO because distance to the points can no longer be measured reliably.

Approaches for tackling monocular VO are commonly classified into three categories, feature-based, direct, and hybrid methods. Feature-based methods use distinctive feature points in the images, and match them between image frames to obtain the relative pose information. Direct methods does this directly through the intensities of the entire, or sub-regions of, image and their gradients between frames [3]. The hybrid methods involve a combination of the two, such as in [4].

This paper focuses on the feature-based methods for monocular VO. In the methods section, the pipeline of them will be presented, and the experiments will then be presented in the results section.

II. Methods

This section presents the steps for a feature-based monocular visual odometry system.

A. Feature Detection and Matching

A key difference of feature-based methods from direct methods is their use of feature points. The correspondences of same points between images encode the relative pose information between the camera frames. Feature detection and matching is in essence the attempt to find and match same world points in different images.

Many point-feature detectors have been proposed in literature. Some prominent ones include SIFT [5], SURF [6], CENSURE [7] and FAST [8]. A good feature detection should be distinctive, repeatable, efficient, and invariant under illumination and geometric changes.

Once the features have been detected across multiple images, they need to be matched against each other, for which a feature descriptor is commonly used. The job of the feature descriptor is to the provide a representation of a feature point such that only the same points will have very similar representations. Some of the notable descriptors are SIFT [5], SURF [6], BRIEF [9], ORB [10] and BRISK [11].

B. Motion Estimation

Given two images with feature point matches, the relative motion between the frames that take these images can then be computed. Concatenating the motions for each consecutive pair of images will then give the entire trajectory.

1) Estimating Essential Matrix: Essential matrix E encodes the relative motion between camera frames up to a unknown scale factor in translation.

$$E = [t]_\times R$$

where

$$[t]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

which is the skew symmetric matrix form of the translation vector t, and R is the rotation matrix.

Geometrically speaking, the essential matrix maps a normalized, i.e. corrected for camera intrinsics, point $\tilde{p}$ in one image to the corresponding epipolar line $E\tilde{p}$ in the second image. For a normalized point $\tilde{p}'$ in the second image that corresponds to the same world point, it must lie on the epipolar line. Figure 1 illustrates the epipolar constraint.

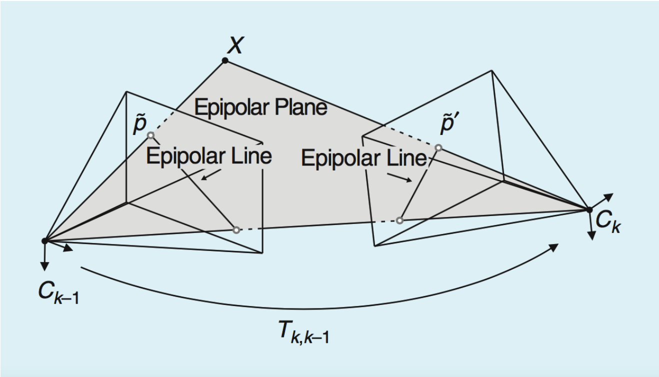

Fig. 1. Epipolar constraint. Image taken from [12].

Mathematically, the epipolar constraint can be written as:

$$\tilde{p}' E \tilde{p} = 0$$

for every pair of corresponding $\tilde{p}$ and $\tilde{p}'$.

Since the essential matrix has five degrees of freedom (three for rotaion, and two for translation up to a scale), in principle, only five correspondences are required to estimate it, for which Nister provided an efficient five-point algorithm [13]. Another algorithm that can accommodate $n \geq 8$ point correspondences is presented in [14].

2) Extracting Relative Pose: Given an essential matrix, there exist four possibilities of the rotation and translation pair [15]. Fortunately, only for one of them, the world point will be in front of both cameras. This is called Cheirality constraint.

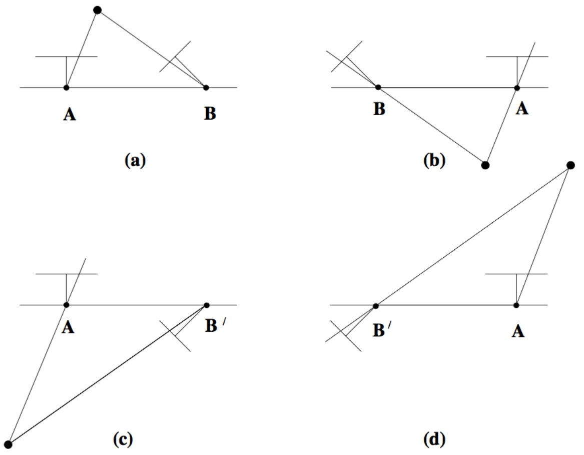

Fig. 2. Illustration of four rotation and translation pairs, and the Cheirality constraint. Image taken from [15].

Theoretically, checking this constraint for one point correspondence is sufficient. However, due to noise in feature point detection and matching and essential matrix estimation, a voting scheme is used.

3) Estimating Relative Scale: As discussed previously, the essential matrix encodes the translation up to an unknown scale. Therefore, we would not be able to obtain the true absolute scale of the camera trajectory. However, we can compute the relative scales between consecutive essential matrices, and thereby ensuring that the translations in the estimated trajectory are of a single, though unknown, scale.

One way to do it is to first find feature matches across three consecutive frames. Then we triangulate the points using the first and second frame, to obtain 3D world points, each denoted as $X_{k-1,i}$. Next, we triangulate the same points using the second and third frame, again obtaining world points, each denoted as $X_{k,i}$. When the estimated essential matrices are of the same scale, the distance between two triangulated world points should be the same, when computed in either way. Therefore, we can compute the relative scale as

$$r = \frac{\|X_{k-1,i} - X_{k-1,j}\|}{\|X_{k,i} - X_{k,j}\|}$$

and scale the second essential matrix accordingly. To account for noise, the relative scale is computed for multiple point pairs, and the mean or median is taken as the estimate.

C. Outlier Rejection

The pipeline presented so far assumes that the feature matching in step 1 is perfect. However, it is common for features to be matched incorrectly, and these wrong correspondences are the outliers. Estimating the essential matrix with outlier correspondences will give dramatically wrong result. Therefore, it is paramount to reject the outliers when estimating the motion. An established algorithm to do it is RANSAC [16].

RANSAC randomly selects at minimum five point correspondences, and computes an essential matrix from it. The rest of the point correspondences are measured against this model, the metric of which could be the distance of point $\tilde{p}'$ to the estimated epipolar line $E\tilde{p}$. Correspondences within a threshold are considered as inliers for this model. Repeat this step many times, and the largest inlier set is chosen as the solution. The essential matrix is then computed with this inlier set.

The number of iterations N required to find a correct solution with probability p can be computed as

$$N = \frac{\log(1-p)}{\log(1-(1-\epsilon)^s)}$$

where s is the number of points used to compute the model in each iteration, and ϵ is the percentage of outliers.

III. Results

To provide a quantitative measure of the accuracy of the visual odometry, we use the following error metric:

$$e = \frac{1}{N} \sum_{k=1}^{N} \|t_{k,estimate} - t_{k,true}\|$$

where $t_{k,estimate}$ is the estimated pose's translation component in world frame at frame k, and $t_{k,true}$ is the corresponding ground-truth translation. To enable reasonable comparison with ground-truth, the first relative pose estimate's scale would be set at its true value.

The experiments are performed with the ICL-NUIM dataset [17].

A. *Scale Estimation Issues*

It turns out that computing relative scales between estimates is tricky, for the following three reasons:

1) Computing relative scales by the discussed method requires feature correspondences across at least three images. When the three images span a relatively long trajectory, they might not have enough correspondences.
2) When the three images span a short trajectory, the triangulation of points would be inaccurate, which would result in inaccurate estimate of scale.
3) Since each scale, apart from the first one, is computed relative to the previous one, if one of the scale estimate is wrong, all the following scale estimates would be wrong, even if the relative scales are computed accurately.

The following estimated trajectory shows an example of the first issue:

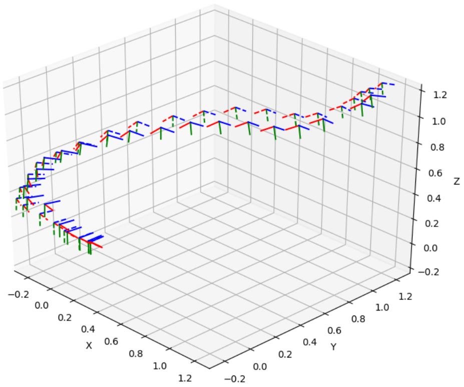

Fig. 3. Trajectory estimated with Living Room 'lr kt2' dataset with step size of 15. The solid frames are the estimates, while the dashed frames are the ground-truth.

It can be seen that the estimates align fairly closely with the ground-truth, and its error as defined previously is 0.123. However, this is not the entire trajectory of the dataset, because at some frame not enough feature correspondences were detected, and the algorithm fell short of the entire path.

The second and third issue are demonstrated in the following figure.

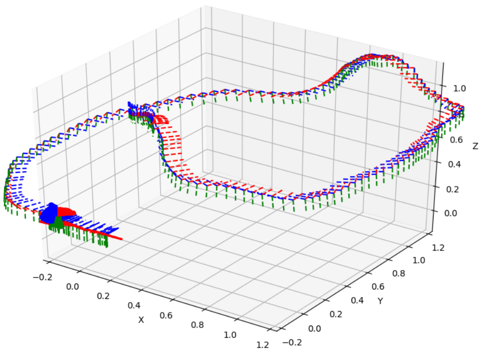

Fig. 4. Trajectory estimated with Living Room 'lr kt2' dataset with step size of 5.

The difference from the previous experiment is that now we set the step size to be 5, which means that the consecutive frames are now closer to each other. Enough correspondences can now be discovered for every three consecutive frames, which enable the whole trajectory to be estimated. However, as can be seen in the figure, the estimated trajectory is now clustered near the beginning of the trajectory, which is caused by a too small scale estimate near the beginning. The remaining trajectory estimate would then be shrinked accordingly. For comparison, the error for this trajectory is 1.404.

B. *Step Size Effect*

The following experiments compare the performance of the algorithm with different step sizes, i.e. the number of frames skipped between each frame pair for estimation. Due to the discussed issues with scale estimation, the following experiments are performed with the scales set to ground-truth.

We vary the step size from 2 to 30, and plot the errors, where each error is averaged across multiple runs.

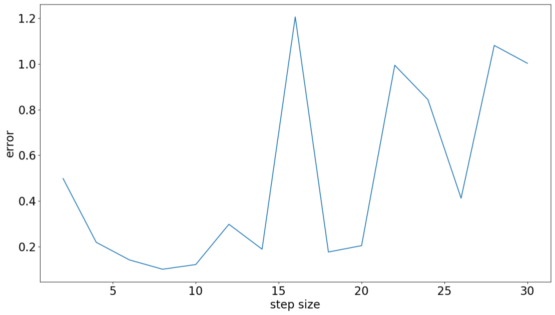

Fig. 5. Error plot against step size, averaged across 5 runs.

The figure shows that in general, too small and too large step size both negatively affect the error metric. However, the dependency on step size is noisy, as shown by the spikes even after averaging. It indicates that the effect of step size could be highly dependent on the specific image sequences, where some pairs of images would not give accurate essential matrix estimation.

A trajectory estimated with step size of 10, which corresponds to the minimum in the error plot is attached for illustration.

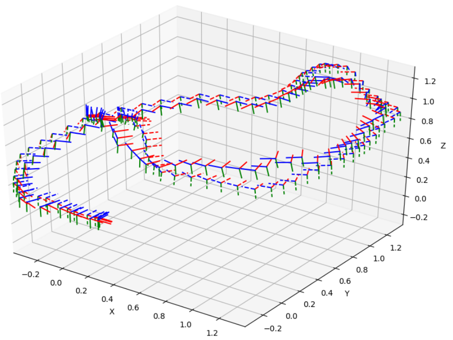

Fig. 6. Trajectory estimated with Living Room 'lr kt2' dataset with step size of 10, where scale is set to ground-truth.

IV. Conclusion

This project presents a pipeline for feature-based monocular visual odometry. It then illustrates the issues with scale estimation by triangulation method. Other ways of estimating relative scales include trifocal constraints [15] and windowed bundle adjustment [18]. The trajectory estimate's relation to step size is then illustrated, which shows that the specific image pairs for essential matrix estimation could affect accuracy significantly. An adaptive method where step size is not fixed could potentially tackle this problem.

Acknowledgment

The author would like to thank Prof. Todd Zickler for teaching the computer vision course CS283, and for the advice outside classroom. The author would also like to express gratitude for the help from Dor Verbin and Mia Polansky during office hours and by emails.

References

[1] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 1, pp. I–I, Ieee, 2004.

[2] H. P. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover.," tech. rep., Stanford Univ CA Dept of Computer Science, 1980.

[3] M. Irani and P. Anandan, "About direct methods," in *Workshop on Vision Algorithms*, 1999.

[4] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, pp. 404–417, Springer, 2006.

[7] M. Agrawal, K. Konolige, and M. R. Blas, "Censure: Center surround extremas for realtime feature detection and matching," in *European Conference on Computer Vision*, pp. 102–115, Springer, 2008.

[8] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*, pp. 430–443, Springer, 2006.

[9] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*, pp. 778–792, Springer, 2010.

[10] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, "Orb: An efficient alternative to sift or surf.," in *ICCV*, vol. 11, p. 2, Citeseer, 2011.

[11] S. Leutenegger, M. Chli, and R. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 IEEE international conference on computer vision (ICCV)*, pp. 2548–2555, Ieee, 2011.

[12] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part i: The first 30 years and fundamentals," *IEEE Robotics and Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[13] D. Nister, "An efficient solution to the five-point relative pose problem," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 2, pp. II–195, IEEE, 2003.

[14] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 5828, p. 133, 1981.

[15] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[16] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[17] A. Handa, T. Whelan, J. McDonald, and A. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," in *IEEE Intl. Conf. on Robotics and Automation, ICRA*, (Hong Kong, China), May 2014.

[18] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," in *International workshop on vision algorithms*, pp. 298–372, Springer, 1999.