Flow Separation for Fast and Robust Stereo Odometry

Michael Kaess, Kai Ni and Frank Dellaert

***Abstract*— Separating sparse flow provides fast and robust stereo visual odometry that deals with nearly degenerate situations that often arise in practical applications. We make use of the fact that in outdoor situations different constraints are provided by close and far structure, where the notion of close depends on the vehicle speed. The motion of distant features determines the rotational component that we recover with a robust two-point algorithm. Once the rotation is known, we recover the translational component from close features using a robust one-point algorithm. The overall algorithm is faster than estimating the motion in one step by a standard RANSAC-based three-point algorithm. And in contrast to other visual odometry work, we avoid the problem of nearly degenerate data, under which RANSAC is known to return inconsistent results. We confirm our claims on data from an outdoor robot equipped with a stereo rig.**

I. Introduction

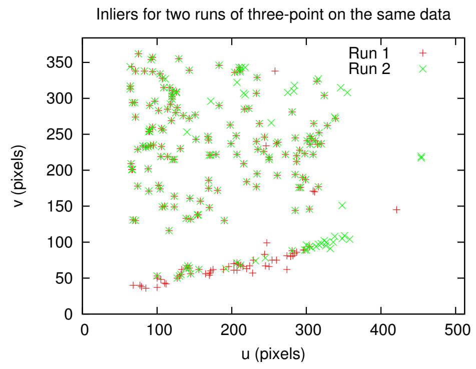
Visual odometry is a technique that estimates the ego-motion from images perceived by moving cameras. A typical use is autonomous navigation for mobile robots, where getting accurate pose estimates is a crucial capability in many settings. Visual odometry does not require sensors other than cameras, which are cheap, passive and have low power consumption. Therefore, there are many interesting applications ranging from search and rescue over reconnaissance to commercial products such as entertainment and household robots. Furthermore, visual odometry lays the foundation for visual SLAM, which improves large-scale accuracy by taking into account long-range constraints including loop closing.

Visual odometry is at its heart a camera pose estimation [1], and has seen considerable renewed attention in recent years. Olson [2] uses visual odometry and incorporates an absolute orientation sensor to prevent drift over time. Nister *et al.* [3] present a real-time system using a three-point algorithm, which works in both monocular and stereo settings. Levin and Szeliski [4] use loopy belief propagation to calculate visual odometry based on map correlation in an off-line system. Some systems [5], [6] also use omni-directional sensors to increase the field of view. Ni and Dellaert [7] present an image-based approach that has high computational requirements and is not suitable for high frame rates. Large-scale visual odometry in challenging outdoor environments is presented by Konolige *et al.* [8]–[10] and Nister *et al.* [11], but the problem of degenerate data is not addressed. Recent work by Zhu *et al.* [12] remembers landmarks to improve the accuracy of visual odometry.

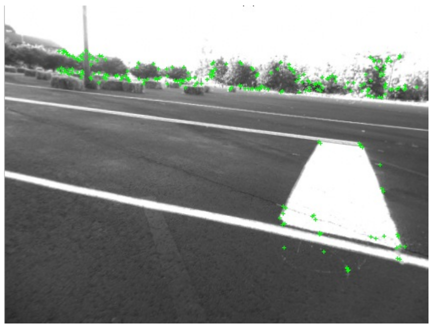
This work was partially supported by DARPA grant FA8650-04-C-7131.

M. Kaess is with CSAIL, Massachusetts Institute of Technology, Cambridge, MA 02139, USA `kaess@mit.edu`

K. Ni and F. Dellaert are with the School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA `{nikai,dellaert}@cc.gatech.edu`



(a) Two runs of standard visual odometry on the same data yield different sets of inliers in degenerate situations.



(b) Ground surface with low texture resulting in degenerate data.

Fig. 1: Examples of nearly degenerate data based on low availability of features close to the camera, which are needed to estimate the translation. (a) An example in which the inliers for two different runs of RANSAC significantly differ, as shown by the red plus signs and green crosses. In this case, patches of strong light in otherwise dark areas of the image lead to overexposure resulting in only a small number of close features along a line. (b) A different scenario that causes a standard three-point RANSAC approach to fail because most of the structure is too distant to provide translational constraints, but a high inlier ratio can be achieved from the features in the background.

Almost all visual odometry work use the random sample consensus (RANSAC) algorithm [13] for robust model estimation, and are therefore susceptible to problems arising from nearly degenerate situations. The expression “degenerate data” refers to data that is insufficient for constraining a certain estimation problem. Nearly degenerate data means that there are only a few data points without which the remaining data is degenerate. It is well known that RANSAC fails when directly applied to nearly degenerate data [14], [15], but no visual odometry work addressed this so far. In visual odometry nearly degenerate data occurs for a variety of reasons, such a ground surfaces with low texture (see Fig. 1(b)), bad lighting conditions that result in overexposure (which resulted in the example in Fig. 1(a)), and motion blur. The consequence is that multiple runs of RANSAC on the same data yield different results, as the example in

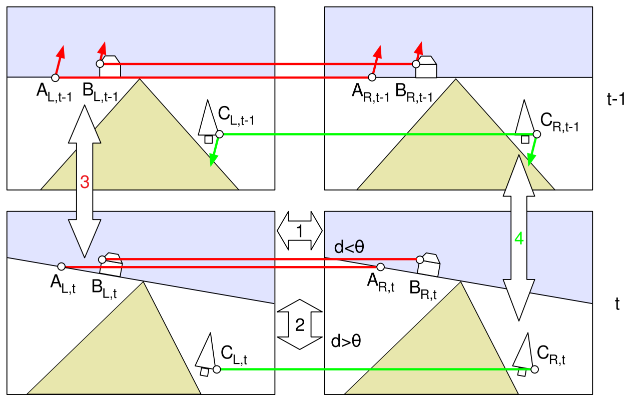


Fig. 2: Our approach consists of four steps that are performed for each new frame: (1) Sparse stereo and putative matching. (2) Separate the sparse flow by disparity based on the threshold θ that adapts to the vehicle speed. (3) Recover the rotational component of the camera motion from two distant points A_t and B_t using RANSAC to robustly deal with outliers. (4) Recover the translational component of the camera motion from one close point C_t again using RANSAC.

Fig. 1(a) shows. While no previous visual odometry work has dealt with the problem of degeneracy, Frahm and Pollefeys [15] present a general method called QDEGSAC for robust estimation in such cases. However, their general method is not suitable here because of the hard real-time constraints of visual odometry.

We present a novel approach to visual odometry that is fast and robustly deals with nearly degenerate data. An overview of the approach is shown in Fig. 2. Based on extracted image features we first establish stereo correspondences as well as putative matches between two successive frames. We then separate the stereo features based on their disparity and the vehicle speed into two sets. The set with low disparity is used to recover the camera rotation with a two-point algorithm. Based on the recovered rotations, we then use the second set of features to recover the vehicle translation with a one-point algorithm. We have implemented our algorithm as well as a reference three-point algorithm on an outdoor robot with a stereo rig. We show that our algorithm is faster than the reference system, and that it successfully deals with nearly degenerate data.

II. Stereo Odometry by Sparse Flow Separation

As shown in Fig. 2 our algorithm performs the following four steps on each new stereo pair:

1) Perform sparse stereo and putative matching.
2) Separate features based on disparity.
3) Recover rotation with two-point RANSAC.
4) Recover translation with one-point RANSAC.

Below we describe these steps in detail. We assume that we have rectified images with equal calibration parameters for both cameras of the stereo pair, in particular focal length f and principal point (u_0, v_0). We define the reference camera to be the one whose pose is tracked. The other view is defined by the baseline b of the stereo pair. Camera poses are represented by a translation vector $\mathbf{t}$, and the three Euler angles yaw ϕ, pitch θ and roll ψ, or alternatively the corresponding rotation matrix R.

A. *Sparse Stereo And Putative Matches*

We extract features in the current frame and establish stereo correspondences between the left and right image of the stereo pair. For a feature in one image, the matching feature in the other is searched for along the same scan line, with the search region limited by a maximum disparity. As there are often multiple possible matches, appearance is typically used and the candidate with lowest difference in a small neighborhood accepted, resulting in the set of stereo features $\mathcal{F} = \{(u_i, v_i, u'_i)\}_i$, where (u, v) is the location of a feature in the reference frame and (u', v) the corresponding feature in the other frame.

Based on the stereo features $\mathcal{F}_t$ from the current frame and the features $\mathcal{F}_{t-1}$ from the previous frame we establish *putative matches*. For a feature in the previous frame, we predict its location in the current frame by creating a 3D point using disparity and projecting it back. For this reprojection we need to have a prediction of the vehicle motion, which is obtained in one of the following ways:

- Odometry: If wheel odometry or IMU are available.
- Filter: Predict camera motion based on previous motion.
- Stationary assumption: At high frame rate we obtain a small enough motion to approximate by a stationary camera.

As the predicted feature locations are not exact in any of these cases, we select the best of multiple hypotheses. We use the approximate nearest neighbors (ANN) algorithm [16] to efficiently obtain a small set of features within a fixed radius of the predicted location. The best candidate based on template matching is accepted as a putative match. We denote the set of putative matches with $\mathcal{M}$. As some putative matches will still be wrong, we use a robust estimation method below to filter out incorrect matches.

B. *Separate Features*

We separate the stereo features based on their usefulness in establishing the rotational and the translational components of the stereo odometry. The key idea is that small changes in the camera translation do not visibly influence points that are far away. While points at infinity are not influenced by translation and are therefore suitable to recover the rotation of the camera, there might only be a small number or even no such features visible due to occlusion, for example in a forest or brush environment. However, as the camera cannot translate far in the short time between two frames (0.067 seconds for our 15 frames per second system), we can also use points that have disparities somewhat larger than 0. Even if the camera translation is small, however, if a point is close enough to the camera its projection will be influenced by this translation.

We find the threshold θ on the disparity of a point for which the influence of the camera translation can be neglected. The threshold is based on the maximum allowed pixel error given by the constants Δu and Δv, for which values in the range of 0.1 to 0.5 seem reasonable. It also depends on the camera translation $\mathbf{t} = (t_x, t_y, t_z)$ that can again be based on odometry measurements, a motion filter, or a maximum value provided by physical constraints of the motion. Considering only the center pixel of the camera as an approximation, we obtain the disparity threshold

$$\theta = \max\left\{ \frac{b}{\frac{t_x}{\Delta u} - \frac{t_z}{f}}, \frac{b}{\frac{t_y}{\Delta v} - \frac{t_z}{f}} \right\} \tag{1}$$

We separate the putative matches into the set $\mathcal{M}_{\text{rot}} = \{\mathcal{M} | \text{disparity} \leq \theta\}$ that is useful for estimating the rotation, and the set $\mathcal{M}_{\text{trans}} = \{\mathcal{M} | \text{disparity} > \theta\}$ that is useful for estimating the translation. Note that we always have enough putative matches in $\mathcal{M}_{\text{rot}}$ even if the robot is close to a view obstructing obstacle, due to physical constraints. As the robot gets close to an obstacle, its speed has to be decreased in order to avoid a collision, therefore increasing the threshold θ, which allows closer points to be used for the rotation estimation. On the other hand, it is possible that all putatives have disparities below the threshold θ, in particular for $\mathbf{t} = 0$. In that case we still have to use some of the close putatives for calculating the translation, as we do not know if the translational speed of the camera is exactly 0 or just very small. We therefore always use a minimum number of the closest putative matches for translation estimation, even if their disparities fall below θ.

C. Rotation: Two-point RANSAC

We recover the rotational component R of the motion based on the set of putative matches $\mathcal{M}_{\text{rot}}$ that are not influenced by translation. For points at infinity it is straightforward to recover rotation based on their direction. Even if points are close to the camera such that reliable depth information is available, but the camera performs a pure rotational motion, the points can be treated as being at infinity, as their depths cannot be determined from the camera motion itself. Even though the camera's translation is not necessarily 0 in our case, we have chosen the threshold θ so that the resulting putative matches $\mathcal{M}_{\text{rot}}$ can be treated as points at infinity for the purpose of rotation estimation. We therefore take a monocular approach to rotation recovery.

While the set of putative matches contains outliers, let us for a moment assume that the matches $(\mathbf{z}^R_{i,t}, \mathbf{z}^R_{i,t-1}) \in \mathcal{M}_{\text{rot}}$ with $\mathbf{z}^R_{i,t} = (u^R_{i,t}, v^R_{i,t})$ and $\mathbf{z}^R_{i,t-1} = (u^R_{i,t-1}, v^R_{i,t-1})$ are correct and therefore correspond to the homogeneous directions (ie. $w^R_i = 0$)

$$X^R_i = \begin{bmatrix} x^R_i & y^R_i & z^R_i & 0 \end{bmatrix}^T \tag{2}$$

Two such matches are necessary to determine the rotation of the camera for either of the following two methods:

- We estimate the rotation R together with n directions X^R_i (2 degrees of freedom, because X^R_i is homogeneous with $w^R_i = 0$), yielding $3{+}2n$ degrees of freedom (DOF). Each match yields 4 constraints, therefore $n = 2$ is the minimum number of correspondences needed to constrain the rotation.
- We estimate only the rotation R, by using the features from the previous time $t-1$ to obtain the direction of the points. This yields 3 DOF, with only 2 remaining constraints per match, again yielding $n = 2$.

For pure rotation ($\mathbf{t} = 0$) the reprojection error E is

$$E \quad = \quad \left\| \mathbf{z}^R_i - \nu^R(R, \begin{bmatrix} x^R_i \\ y^R_i \\ z^R_i \end{bmatrix}) \right\|^2 \tag{3}$$

where $(u, v) = \nu^R(R, X)$ is the monocular projection of point X into the camera at pose $R, \mathbf{t} = 0$. We numerically obtain an estimate for the rotation R and optionally the point directions by minimizing the non-linear error term

$$R_t = \underset{R_t}{\operatorname{argmin}} \sum_{i,\tau \in \{t,t-1\}} \left\| \mathbf{z}^R_{i,\tau} - \nu^R \left(R_\tau, \begin{bmatrix} x^R_i \\ y^R_i \\ z^R_i \end{bmatrix} \right) \right\|^2 \tag{4}$$

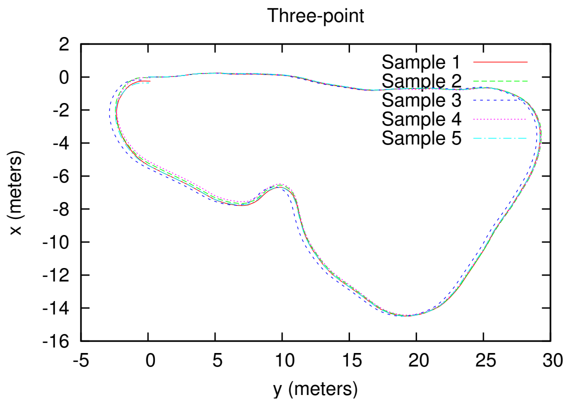
where $R_{t-1} = I$ and therefore R_t the rotational component of the visual odometry. Note that we also need to enforce $\left\| \begin{bmatrix} x^R_i & y^R_i & z^R_i \end{bmatrix}^T \right\|^2 = 1$ to restrict the extra degree of freedom provided by the homogeneous parametrization.

While we have assumed correct matches so far, the putative matches in $\mathcal{M}_{\text{rot}}$ are in fact noisy and contain outliers. We therefore use the random sample consensus (RANSAC) algorithm [13] to robustly fit a model. The sample size is two, as two putative matches fully determine the camera rotation, as discussed above. RANSAC repeatedly samples two points from the set of putatives and finds the corresponding rotation. Other putatives are accepted as inliers if they agree with the model based on thresholding the reprojection error E from (3). Sampling continues until the correct solution is found with some fixed probability. A better rotation estimate is then determined based on all inliers. Finally, this improved estimate is used to identify inliers from all putatives, which are then used to calculate the final rotation estimate $\hat{R}$.

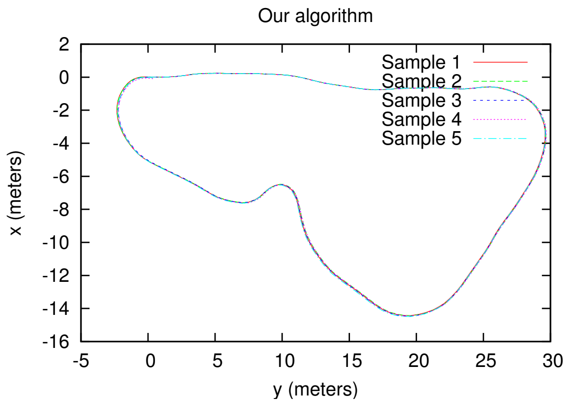
D. Translation: One-point RANSAC

Based on the now known camera rotation, we recover the translation from the close putative matches $\mathcal{M}_{\text{trans}}$. We denote a putative match as $\mathbf{z}^{\mathbf{t}}_{i,t}$ and the corresponding 3D points as $X^{\mathbf{t}}_i$. Each measurement imposes $2 \times 3 = 6$ constraints, i.e. $\mathbf{z}^{\mathbf{t}}_{i,t} = (u^{\mathbf{t}}_{i,t}, v^{\mathbf{t}}_{i,t}, u'^{\mathbf{t}}_{i,t})$ and $\mathbf{z}^{\mathbf{t}}_{i,t-1} = (u^{\mathbf{t}}_{i,t-1}, v^{\mathbf{t}}_{i,t-1}, u'^{\mathbf{t}}_{i,t-1})$, which now includes stereo information in contrast to Section II-C. Intuitively we can recover the translation from a single putative match, as each of the two stereo frames defines a 3D point and the difference between the points is just the camera translation. Practically we again have two different approaches:

1) We estimate both the translational component $\mathbf{t}$ with 3 DOF and the 3D points $\left\{X^{\mathbf{t}}_i\right\}_i$ with 3 DOF each. Each measurement contributes 6 constraints, therefore a single match will make the system determinable.
2) We estimate only the translation $\mathbf{t}$ yielding 3 DOF, by using the previous stereo feature to generate the 3D point. Each measurement then only contributes 3 constraints, again requiring only a single match to constrain the camera translation.



(a) Multiple runs yield different trajectories for three-point.



(b) Our algorithm returns nearly stable results.

(c) Camera images.

Fig. 3: Trajectories from multiple runs on the San Antonio data set of (a) the reference three-point implementation and (b) our algorithm. Some images of the environment are shown in (c). It is apparent that the three-point algorithm returns inconsistent trajectories, while our algorithm provides almost the same result in each run.

Similar to Section II-C, the translation is recovered by optimizing over the translation and optionally the 3D points:

$$\mathbf{t}_t, \{X_i^{\mathbf{t}}\}_i = \underset{\mathbf{t}_t, \{X_i^{\mathbf{t}}\}_i}{\operatorname{argmin}} \sum_{i,\tau \in \{t,t-1\}} \left\| \mathbf{z}_{i,\tau}^{\mathbf{t}} - \nu^{\mathbf{t}} \left(\hat{R}_\tau, \mathbf{t}_\tau, X_i^{\mathbf{t}} \right) \right\|^2 \tag{5}$$

where $(u, v, u') = \nu^{\mathbf{t}}(R, t, X)$ is the stereo projection function, and we choose $R_{t-1}, \mathbf{t}_{t-1}$ to be a camera at the origin, and $R_t = \hat{R}$ is the rotation recovered by the two-point algorithm. Consequently, $\mathbf{t}_t$ is the translation of the camera that we are interest in. Again, we use RANSAC to robustly deal with outliers, where each sample defines a translation according to (5) and the final model is also determined by (5) using all inliers.

III. Experiments and Discussion

We compare both efficiency and accuracy of our algorithm with a reference three-point algorithm based on multiple data sets from an outdoor robot equipped with a stereo camera.

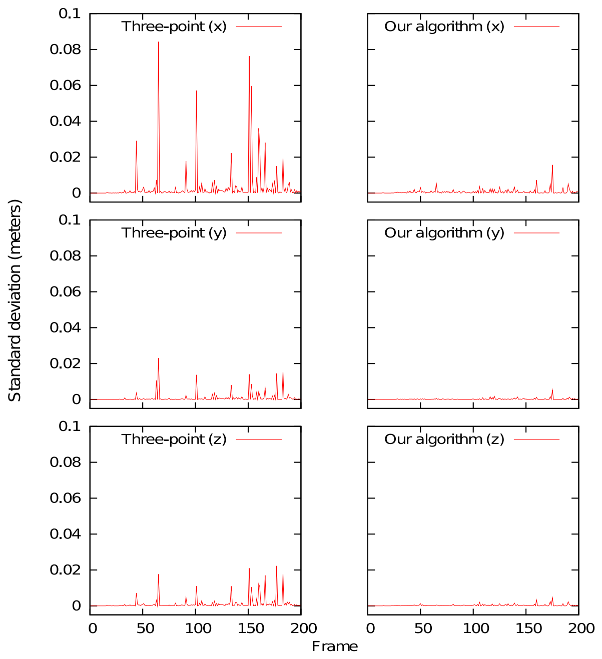


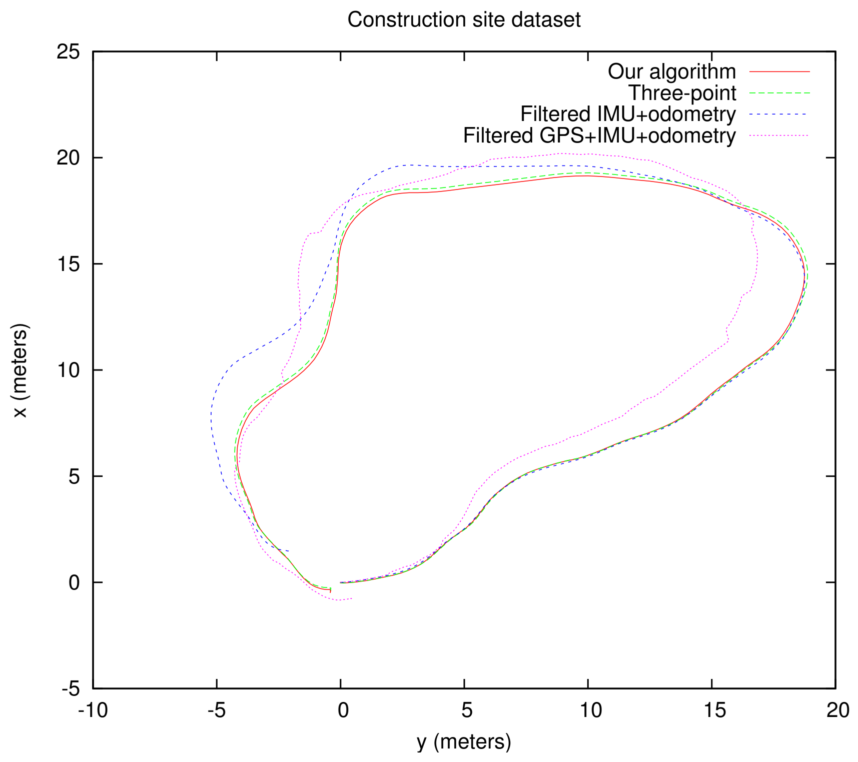
Fig. 4: Standard deviations for the x, y and z components of 15 runs over the first 200 frames of the San Antonio sequence for both the reference three-point algorithm (left column) and our algorithm (right column). While the three-point algorithm shows significant variation on some frames, our algorithm yields almost stable results, in agreement with Fig. 3.

For our reference three-point algorithm we again use non-linear optimization, but this time to simultaneously recover rotation, translation and optionally the 3D points by

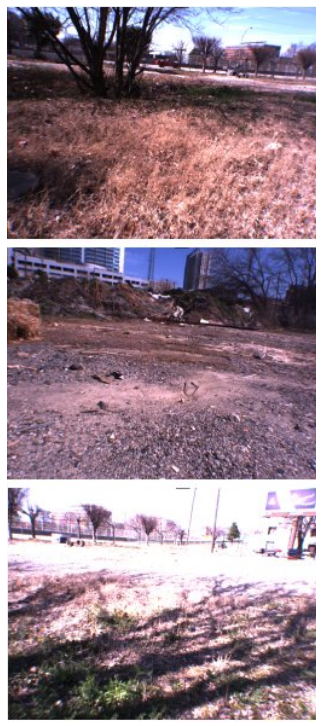
$$R, \mathbf{t}, \{X_i\}_i = \underset{R,\mathbf{t},\{X_i^{\mathbf{t}}\}_i}{\operatorname{argmin}} \sum_i \left\| \mathbf{z}_i - \nu \left(K \begin{bmatrix} R & \mathbf{t} \end{bmatrix} X_i \right) \right\|^2 \tag{6}$$

where we use the measurements $\mathbf{z}_i$ and corresponding points X_i from all putative matches $\mathcal{M}_{rot} \cup \mathcal{M}_{trans}$. The implementation is shared between the two algorithms to make comparison as fair as possible.

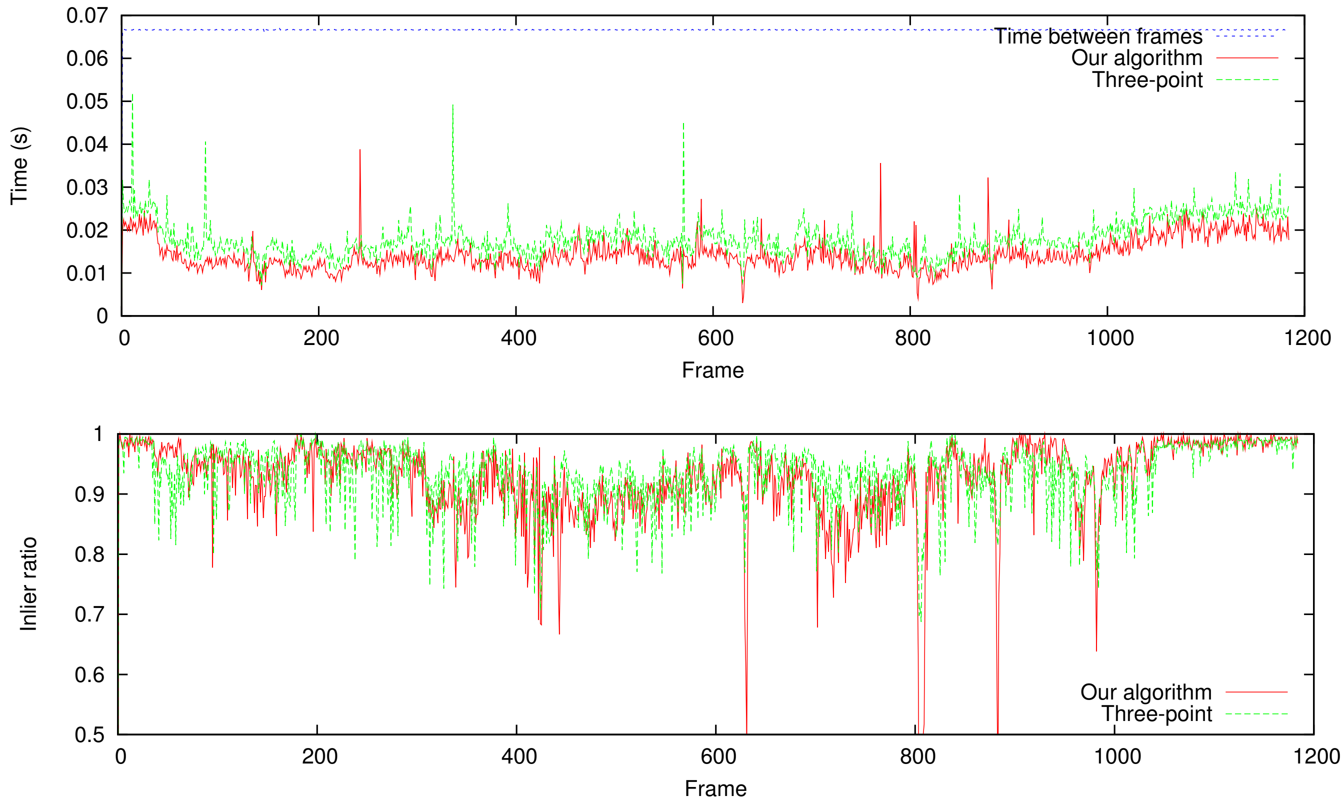
In our implementation, we use the Harris corner detector [17] to obtain features in the reference frame and use dense stereo information to establish stereo correspondences for those features. Based on images with a resolution of 512×384 pixels, we extract a maximum of 600 features per image, by detecting a maximum of 50 features in each region of a 3×4 grid to ensure that the features are well spread out over the image. We use the Levenberg-Marquardt algorithm for all non-linear optimizations. For RANSAC, we use a 99% confidence, with a maximum number of 1000 iterations to guarantee real-time performance even for frames where no consistent model can be found. We do not estimate 3D points and directions, but instead generate them based on the previous frame to speed up the non-linear optimization. For sparse stereo flow separation we use the constant threshold of $\theta = 8$ pixels determined from (1) with a baseline of $0.12m$ and based on the maximum speed of the mobile robot platform. Timing results are obtained on a Core 2 Duo 2.2GHz laptop computer.



(a) Trajectories based on multiple algorithms and sensors.



(b) Camera images.



(c) Timing and inlier ratios comparing three-point with our algorithm.

Fig. 5: Results from our construction site data set, that compare our algorithm with the reference three-point algorithm, showing increased efficiency under comparable quality. (a) The start and end points of the actual trajectory are at the origin. (b) The robot was driven over grass and gravel under difficult lighting conditions. (c) Top: Time required for RANSAC for our approach (one-point and two-point together) and the reference three-point implementation. Bottom: Inlier ratios for the two-point component of our algorithm and the reference three-point algorithm. The inlier ratio is comparable for all, but RANSAC converges faster for a smaller sample size, and therefore our algorithm is faster.

A. Nearly Degenerate Data and Robustness

We show that our algorithm deals well with nearly degenerate data, which is often encountered in outdoor visual odometry settings, but has not been addressed by any visual odometry work so far. Fig. 3(a) shows how degeneracy influences the recovered trajectory by returning inconsistent results. This San Antonio sequence consists of putative matches (without images) recorded at 15 frames per second during a live demo of our work. As can be seen, each run on the same data yields a different trajectory. Fig. 3(b) shows that our algorithm returns nearly consistent results over multiple runs. Fig. 4 analyzes the problem in more detail for the first 200 frames of this sequence. While our algorithm suffers only minor variations (right column), the three-point algorithm suffers significant variations for some frames (left column). The variations are mainly in the forward (x) direction that is most difficult to constrain, but also in the other two directions. Fig. 1(b) shows an example of two different sets of inliers for one frame, which results in significantly different translation estimates. As discussed earlier, the problem is that RANSAC quickly finds a large number of inliers, most of which constrain the rotation, while only a few matches provide a possibly wrong translation estimate. Based on the large number of inliers, RANSAC severely underestimates the number of samples needed to achieve a certain confidence in the results. Our algorithm instead splits the problem and therefore avoids the degeneracy as can be seen from the results presented here.

B. Speed and Accuracy

We show that our algorithm performs faster than the standard three-point algorithm, while producing at least comparable results. Fig. 5 shows results from our construction site data set, for which the images have been recorded at 15 frames per second. Fig. 5(a) shows the recovered robot trajectory together with the combination of wheel odometry and inertial measurement unit (IMU), as well as combined with GPS. Fig. 5(b) shows sample images from this challenging sequence, which features vastly changing lighting conditions. In Fig. 5(c) we show comparisons for the execution times and inlier ratios. Our algorithm performs faster than the three-point algorithm, even though we perform RANSAC twice, once for rotation and once for translation. The faster execution is explained by the smaller sample size for each case as compared to three-point. Therefore it is more likely to randomly select a good sample and consequently RANSAC needs less iterations, assuming that both have the same inlier ratio. The inlier ratios for the two-point component of our algorithm as well as for the three-point algorithm are very similar as shown in the bottom diagram.

IV. Conclusion

We have presented a novel approach to stereo visual odometry that successfully deals with nearly degenerate data and is faster than a standard three-point algorithm. The key component is splitting the translation and rotation recovery into two separate estimation problems, and using appropriate features for each one based on disparity and camera speed. We implemented this system and compared it to a reference three-point implementation, which is the most common approach underlying visual odometry systems today. In addition to running the system on log files, we have presented our system live on a mobile robot in the context of the DARPA LAGR program, from which the San Antonio data set was obtained.

While accuracy of the presented system can further be improved by applying fixed-lag smoothing [11], by using key frames to prevent drift [10], and by using landmarks to provide longer range constraints [12], these methods are orthogonal to the presented work, and were not the focus of this publication. Note that fixed-lag smoothing does not solve the problem of degeneracy, as only frame-to-frame correspondences are used. As those frame-to-frame correspondences are affected by a failure of RANSAC to find correct matches that constrain the translation, the same error will also be present after fixed-lag smoothing as no new matches will be added by fixed-lag smoothing.

References

[1] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[2] C. Olson, "Stereo ego-motion improvements for robust rover navigation," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2001, pp. 1099–1104.

[3] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, Jun 2004, pp. 652–659.

[4] A. Levin and R. Szeliski, "Visual odometry and map correlation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[5] R. Bunschoten and B. Kröse, "Visual odometry from an omnidirectional vision system," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, vol. 1, 2003, pp. 577–583.

[6] P. Corke, D. Strelow, and S. Singh, "Omnidirectional visual odometry for a planetary rover," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[7] K. Ni and F. Dellaert, "Stereo tracking and three-point/one-point algorithms - a robust approach in visual odometry," in *Intl. Conf. on Image Processing (ICIP)*, 2006.

[8] K. Konolige, M. Agrawal, R. Bolles, C. Cowan, M. Fischler, and B. Gerkey, "Outdoor mapping and navigation using stereo vision," in *Intl. Sym. on Experimental Robotics (ISER)*, Jul 2006.

[9] K. Konolige and M. Agrawal, "Frame-frame matching for realtime consistent visual mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Apr 2007, pp. 2803–2810.

[10] K. Konolige, M. Agrawal, and J. Sola, "Large scale visual odometry for rough terrain," in *IROS visual SLAM workshop*, Oct 2007.

[11] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry for ground vehicle applications," *J. of Field Robotics*, vol. 23, no. 1, Jan 2006.

[12] Z. Zhu, T. Oskiper, S. Samarasekera, R. Kumar, and H. Sawhney, "Ten-fold improvement in visual odometry using landmark matching," in *Intl. Conf. on Computer Vision (ICCV)*, Oct 2007.

[13] R. Bolles and M. Fischler, "A RANSAC-based approach to model fitting and its application to finding cylinders in range data," in *Intl. Joint Conf. on AI (IJCAI)*, Vancouver, BC, Canada, 1981, pp. 637–643.

[14] O. Chum, T. Werner, and J. Matas, "Two-view geometry estimation unaffected by a dominant plane," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[15] J. Frahm and M. Pollefeys, "RANSAC for (quasi-)degenerate data (QDEGSAC)," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[16] S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu, "An optimal algorithm for approximate nearest neighbor searching," *Journal of the ACM*, vol. 45, no. 6, pp. 891–923, 1998.

[17] C. Harris and M. Stephens, "A combined corner and edge detector," *Proceedings of the 4th Alvey Vision Conference*, pp. 147–151, 1988.