Monocular Visual Odometry using a Planar Road Model to Solve Scale Ambiguity

Bernd Kitt* Jörn Rehder‡ Andrew Chambers† Miriam Schönbein* Henning Lategahn* Sanjiv Singh†
*Department of Measurement and Control Systems, Karlsruhe Institute of Technology, Karlsruhe, Germany
†The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA
‡Hamburg University of Technology, Hamburg, Germany

***Abstract*— Precise knowledge of a robots's ego-motion is a crucial requirement for higher level tasks like autonomous navigation. Bundle adjustment based monocular visual odometry has proven to successfully estimate the motion of a robot for short sequences, but it suffers from an ambiguity in scale. Hence, approaches that only optimize locally are prone to drift in scale for sequences that span hundreds of frames.**

In this paper we present an approach to monocular visual odometry that compensates for drift in scale by applying constraints imposed by the known camera mounting and assumptions about the environment. To this end, we employ a continuously updated point cloud to estimate the camera poses based on 2d-3d-correspondences. Within this set of camera poses, we identify keyframes which are combined into a sliding window and refined by bundle adjustment. Subsequently, we update the scale based on robustly tracked features on the road surface. Results on real datasets demonstrate a significant increase in accuracy compared to the non-scaled scheme.

***Index Terms*— Localization, Navigation, Robot Vision**

I. Introduction

Ego-motion estimation is an important prerequisite in robotic's applications. Many higher level tasks like obstacle detection, collision avoidance or autonomous navigation rely on an accurate localization of the robot. All of these applications make use of the relative pose of the current camera with respect to the previous camera frame or a static world reference frame. Usually, this localization task is performed using GPS or wheel speed sensors. In recent years, camera systems became cheaper and the performance of computing hardware increased dramatically. Hence, high resolution images can be processed in real-time on standard hardware. It has been proven, that the information given by a camera system is sufficient to estimate the motion of a moving camera in a static environment, called *visual odometry* [16].

Compared to the abovementioned sensors, camera systems have different advantages. It is well known, that the accuracy of the GPS-localization depends on the number of satellites used. This number drops down in urban environments with large buildings on either side of the road. The accuracy of wheel speed sensors depends mainly on the slip between wheel and road, which can be high depending on the terrain. Obviously, the localization results based on these sensors are highly affected by the environment. Further drawbacks of GPS or inertial measurement units (IMUs) are the low accuracy and the high cost, respectively. The local drift rates for visual odometry are mostly smaller than the drift rates of IMUs, except for expensive high accuracy integrated navigation systems which fuse inertial measurements with GPS data [10]. As for all incremental motion estimation techniques, long term drift can only be mitigated by applying loop-closure on (visual) place recognition (e.g. [20]) or by fusing absolute localization data.

In this work, we make use of a fully calibrated monocular camera to estimate the pose of the current camera with respect to a global world reference frame. The used datasets were captured in urban environments using a vehicle moving at a speed of approximately $15m/s$ on average. The six degrees of freedom (6 DoF) motion of the vehicle is estimated merely on the visual information. No additional sensor measurements such as GPS- or IMU-data are used in contrast to [1] or [5].

The remainder of this paper is organized as follows: The following section describes work already done in the field of vision-based motion estimation. In Section III, the monocular motion estimation approach is described, which is extended in section IV to cope with drift in scale. We close the paper with experimental results in section V, a short conclusion and an outlook on future work.

II. Related Work

In recent years, many algorithms have been developed that estimate the ego-motion of a robot. These algorithms can roughly be classified into two main categories, namely algorithms using monocular camera systems (e.g. [24], [18]) and binocular approaches (e.g. [15], [12]). A further subdivision is possible into algorithms using only feature matches between consecutive frames (e.g. [10], [21]) and algorithms using feature tracks over a couple of images (e.g. [11], [17]).

Each class of algorithms has different benefits and drawbacks. Monocular algorithms suffer from the scale ambiguity in the translational camera movement which is usually resolved using measurements from IMUs (e.g. [5]) or a combination of wheel speed sensors and GPS as in Agrawal et al. (e.g. [1], [2]). Compared to algorithms which use feature tracks, algorithms making use only of feature matches usually suffer from higher drift rates, since the used information incorporates only two images. The entire trajectory is then computed by accumulating the relative camera motions between two consecutive frames. This drift can be reduced using feature tracks over a sequence of images combined with a *bundle adjustment* scheme [23]. The drawback of bundle adjustment is the computational burden of the optimization process. To relax this, most algorithms are based on a bundle adjustment which performs the optimization only over a limited number of images, i.e. a sliding window. Other algorithms make use of additional sensors, like IMU- or GPS-systems (e.g. [1]) to increase the accuracy of the estimation. Obviously, the use of GPS-information reduces drift significantly because of the global nature of the system. Furthermore, approaches applying assumptions about the observer's motion have been developed. Scaramuzza et al. [18] make use of a planar motion model and the non-holonomic constraints of wheeled vehicles to reduce the parameter space and increase the accuracy.

Good localization results have also been achieved using visual SLAM techniques (e.g. [4]) which simultaneously estimate a map of the environment jointly with the camera pose inside this map. Besides the computational complexity of these approaches, most of the monocular visual SLAM techniques perform only well in well structured environments and at low speed. Hence, these approaches are mainly applicable to indoor environments. Recently, Strasdat et al. [20] developed a monocular SLAM algorithm applicable for large-scale environments, which resolves the drift in scale when loop-closure occurs. Since we focus on open-loop scenarios, where a robot is travelling from A to B, this approach is not suitable in our case.

Compared to the abovementioned approaches which combine monocular vision with additional sensors, our algorithm uses only visual inputs. To solve translational scale drift, we make use of some assumptions about the mounting pose of the camera and the planarity of the road surface in the vicinity of the vehicle. Given these reasonable assumptions, we can continuously resolve the ambiguity in scale and reduce the scale drift significantly. We prefer to use a monocular camera setup because the effort used in the calibration process is much less than in the binocular case. Furthermore, calibration errors directly affect the motion estimation process. Hence, the use of a monocular camera mitigates the effect of an erroneous calibration onto the motion estimation.

Compared to Scaramuzza et al. [18] our assumptions are less restrictive since we assume only a locally planar surface but estimate the robot's movement in 6 DoF.

III. Monocular Visual Odometry

The proposed algorithm for performing monocular visual odometry assumes a fully calibrated camera with known and fixed intrinsic calibration parameters $\boldsymbol{K}$. Given a sequence of images, the goal is to estimate the camera pose at each time step solely based on these images. To this end, we track salient image features $\boldsymbol{x}_j^k$ (e.g. corners [7], [19]) over a series of frames. Here, j describes the index of the feature track and k denotes the index of the frame, respectively. Based on these feature tracks, we reconstruct the poses of the moving camera with respect to a predefined world reference frame and the locations of the scene points corresponding with the tracks.

In the following sections we will give a detailed description of the different steps of the pose estimation algorithm. Since the feature tracks are short, the (arbitrary) scale of the estimated camera poses drifts over time. In section IV we propose a technique to reduce the scale drift and recover scale from a moving monocular camera.

A. Pose Initialization

The pose initialization step is based on a set of three preselected keyframes $\mathcal{K} = \{K_1, K_2, K_3\}$ and corresponding feature points visible in all of these images. Based on the feature points, we compute the epipolar geometry between $K_1 \to K_2$ and $K_1 \to K_3$, which describes the relative pose of the keyframes with respect to the world reference frame. Note, that the world reference frame coincides with the camera coordinate frame of K_1. To this end, we use the normalized eight-point-algorithm [9] wrapped in a RANSAC framework [6] to reject outliers caused by independently moving objects or false feature matches. Outliers are detected, using the pairwise Euclidean distance between the observed features and their corresponding epipolarlines. All features with a distance larger than a predefined threshold are classified as outlier.

Based on the essential matrix $\boldsymbol{E}^{K_1 \to K_3}$, we recover the pose $\left\{\boldsymbol{R}^{K_3}, \boldsymbol{t}^{K_3}\right\}$ of K_3 with respect to the world reference frame [8]. Here, $\boldsymbol{R}$ and $\boldsymbol{t}$ describe the orientation and translation of the camera respectively. Note, that $\boldsymbol{t}^{K_3}$ can only be recovered up to an arbitrary scale factor. Using the recovered pose of the third keyframe and the feature correspondences, we compute the corresponding scene points for all features which are inliers in both pairwise epipolar geometry estimations.

Based on the triangulated scene points and their corresponding features in K_2 we can compute its pose with respect to the world reference frame using the algorithm proposed in [13]. Afterwards, we perform a bundle adjustment [23] over these three keyframes to get the best estimation for the camera poses and the scene points. The bundle adjustment minimizes the reprojection error, i.e.

$$\min_{\boldsymbol{P}^k, \boldsymbol{X}_j} \sum_{j,k} d\left(\boldsymbol{P}^k \boldsymbol{X}_j, \boldsymbol{x}_j^k\right)^2, \tag{1}$$

making it the maximum likelihood estimate, assuming a Gaussian distribution in the measurement errors [23]. Here, $\boldsymbol{P}^k = \boldsymbol{K} \cdot \left[\boldsymbol{R}^k | \boldsymbol{t}^k\right]$ denotes the 3×4 projection matrix of camera k and $d\left(.\right)$ denotes the geometric distance between two image points.

B. Pose Estimation

Using the previously initialized scene structure, the estimation of the camera pose $\left\{\boldsymbol{R}^k, \boldsymbol{t}^k\right\}$ corresponding to image I^k, is based on already initialized scene points, visible in the current image, i.e. a set of 2d-3d-correspondences. Given a set of N already known scene points $\mathcal{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_j, \ldots \boldsymbol{X}_N\}$ and feature tracks in the current image I^k, we use all feature tracks $\boldsymbol{x}_j^k$ in the current image with an already associated scene point $\boldsymbol{X}_j$ to estimate the current camera pose in the global coordinate frame, using the algorithm proposed in [13].

Since we would like to reject features lying on independently moving objects or false feature matches, we use a RANSAC scheme to get a robust pose estimation. Hence, we use randomly chosen subsets of the 2d-3d-correspondences and create a pose hypothesis for each subset. Given the pose hypothesis and the set of visible scene points, we can compute the expected image points. Subsequently, we evaluate the quality of each hypothesis using the Euclidean reprojection error between the expected image points based on the current hypothesis and the measured image features. All correspondences with an error larger than a predefined threshold are classified as outlier. The final estimation uses only the correspondences of the largest set of inliers.

C. Keyframe Selection and Pose Refinement

Since the pose estimation scheme proposed in the previous section does not update the scene structure, the number of visible scene points decreases over time. Hence, if none of the initialized scene points is visible in the current camera image, no pose estimation is possible. To this end, new keyframes are selected when the number of visible scene points in the current image drops below a threshold.

Whenever a new frame I^k is selected as keyframe K_{M+1}, where M denotes the number of already initialized keyframes, a bundle adjustment is performed. To keep the computational complexity moderate, we perform the bundle adjustment only over a sliding window of L keyframes. Furthermore, we optimize only the m camera poses of the most recent keyframes inside this window ($m < L$), keeping the remaining camera poses fixed. This is reasonable, since the other keyframes have been optimized multiple times in previous optimization steps. Note that the reprojection error in the fixed camera poses is taken into consideration during the optimization process. We use the well established implementation of a sparse bundle adjustment proposed in Louriakis et al. [14].

After the bundle adjustment, we use the most recent keyframe K_{M+1} and the two previous keyframes $\{K_{M-1}, K_M\}$ to triangulate all feature points visible in these keyframes using a three-view triangulation scheme [3]. Note that only those features are triangulated that do not correspond to a previously initialized scene point. To reject outliers, we check the reprojection error of the triangulated scene points in all keyframes used for the triangulation. If the Euclidean reprojection error in any of these keyframes exceeds a certain threshold, the triangulated scene point is rejected.

The following section describes the approach to recover scale from a monocular image sequence based on assumptions about the camera mounting and the road environment. Furthermore, the proposed approach allows for a reduction in the drift of the scale. This part is the main contribution of the paper and consists of a robust triangulation scheme for feature points lying on the less textured asphalt as well as the recovery of the translational scale from monocular imagery.

IV. Recovering Scale

A. Planar Assumption

In automotive applications, cameras are often rigidly mounted in a fixed position and at constant orientation. Furthermore, streets in urban environments may be reasonably assumed to be approximately planar in the vicinity of the vehicle (see figure 2). Additionally, we assume that the roll and pitch movement of the vehicle is negligible, since high dynamic maneuvers are not in the scope of this contribution. Nevertheless, roll and pitch of the camera can be taken into account based on the estimated pose of the vehicle. In this case, only the initial pose of the camera w.r.t. the ground plane has to be known. Employing these assumptions provides a clue that may be exploited to upgrade the monocular visual odometry to a metric reconstruction and to compensate for the inevitable drift in scale. For this purpose, we estimate the ground plane based on correspondences of features that may safely be assumed to lie on this plane. The motion of the camera is then scaled in a way that recovers its known height above ground.

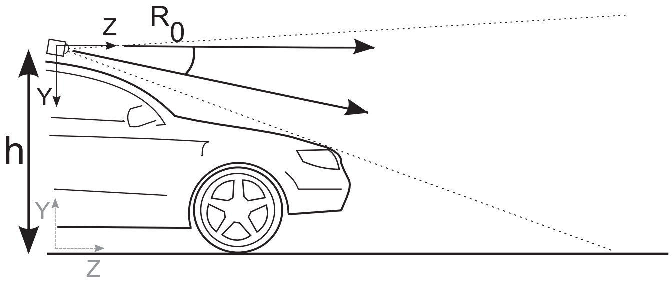

Fig. 1: This figure illustrates the mounting of the camera. h denotes the height of the camera above ground and $\boldsymbol{R}_0$ denotes the orientation of the camera with respect to the ground plane.

B. Patch Matching

While we employ corner-like features (e.g. [19]) in visual odometry, this approach fails to robustly detect features on asphalt due to the lack of strong texture. Furthermore, the images of patches on the street plane close to the camera undergo severe perspective distortions in successive frames caused by the camera movement, which makes it particularly challenging to match them by means of simple block-matching algorithms. However, the known mounting position and orientation of the camera as well as the assumption about the planarity of the area in front of the vehicle may be exploited to overcome these challenges and robustly match features on the street.

Let $\boldsymbol{R}_0$ denote a rotation matrix that accounts for the pitch and roll angle of the camera as depicted in figure 1 and $\boldsymbol{t} = -\boldsymbol{R}_0\,(0, h, 0)^{\mathsf{T}}$ denotes the translation in its reference frame, where h is the height above the road. Further, let the road plane be defined by $Y = 0$ as shown by the gray coordinate frame in figure 1, and a point on this plane by $\boldsymbol{X}' = (X, Z, 1)^{\mathsf{T}}$ in

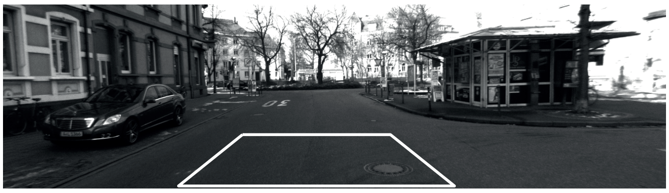

Fig. 2: This figure depicts the region of interest (ROI) in the image that is assumed to be approximately planar.

projective coordinates, then image points $\boldsymbol{x} = (x, y, 1)^{\mathsf{T}}$ are related to points $\boldsymbol{X}'$ on the road plane through a projective transformation $\boldsymbol{x} = \boldsymbol{H}\boldsymbol{X}'$. This transformation is calculated as (see also [8])

$$\begin{aligned} \boldsymbol{x} &= \boldsymbol{K} \cdot [\boldsymbol{R}_0 | \boldsymbol{t}] \, (X, 0, Z, 1)^{\mathsf{T}} && (2) \\ &= \boldsymbol{K} \cdot [\boldsymbol{r}_1 \ \boldsymbol{r}_3 \ \boldsymbol{t}] \, (X, Z, 1)^{\mathsf{T}} && (3) \\ &= \boldsymbol{H}\boldsymbol{X}', && (4) \end{aligned}$$

where $\boldsymbol{r}_1$ and $\boldsymbol{r}_3$ denote the first and third column of $\boldsymbol{R}_0$. Our approach uses the inverse mapping $\boldsymbol{H}^{-1}$ to generate a synthetic fronto-parallel view of a region of interest (ROI) on the street plane, as depicted in figure 3.

As the ROI is in general evenly textured, it is difficult to identify significant features that are well suited for tracking. Instead, a predefined set of square patches is used in frame k, located at the edge of the rectangular ROI closest to the vehicle. This choice ensures that a region with high resolution is used for matching as well as the maximum flow for triangulation. The patches are matched against the ROI at $k-1$, using the sum of absolute differences (SAD) and choosing the minimum value as match. In most cases, the matching exhibits a distinct minimum. In the fronto-parallel projection, patches in successive frames are related by an Euclidean transformation which improves the robustness of patch based matching schemes significantly. In our experiments, the rotational component of this transformation was moderate and thus did not affect the performance of the SAD based matching negatively.

Using the projective transformation $\boldsymbol{H}$, the correspondences are mapped onto image points as

$$\begin{aligned} \boldsymbol{x}_j^k &= \boldsymbol{H}\boldsymbol{X}_j'^{k} \qquad \text{and} && (5) \\ \boldsymbol{x}_j^{k-1} &= \boldsymbol{H}\boldsymbol{X}_j'^{k-1}. && (6) \end{aligned}$$

Note that the purpose of all steps described in this section is to robustly match features on the road surface. Only the positions $\boldsymbol{x}_j^{k-1}$ and $\boldsymbol{x}_j^k$ of these features in I^{k-1} and I^k will be used in the following.

C. Scale Update

In order to relate the height of the camera above the estimated ground plane to the distance of translation, we triangulate the points based on the relative positions of the cameras at frame $k-1$ and frame k as estimated by the visual odometry. Let

$$\widehat{\boldsymbol{P}}^{k-1} = \boldsymbol{K} \cdot [\boldsymbol{R}_0 | \boldsymbol{0}] \tag{7}$$

$$\widehat{\boldsymbol{P}}^{k} = \boldsymbol{K} \cdot \left[\boldsymbol{R}^k \left(\boldsymbol{R}^{k-1}\right)^{-1} \boldsymbol{R}_0 | \boldsymbol{t}^k - \boldsymbol{R}^k \left(\boldsymbol{R}^{k-1}\right)^{-1} \boldsymbol{t}^{k-1} \right] \tag{8}$$

be projection matrices, where $\boldsymbol{R}_0$ accounts for tilting as depicted in figure 1. $\widehat{\boldsymbol{P}}^k$ compensates for the poses of both frames in the global world reference frame, i.e. $\widehat{\boldsymbol{P}}^k$ expresses the relative pose of frame k w.r.t. frame $k-1$. Since we compensate for the mounting orientation of the camera at frame $k-1$, the X-Z-plane of the camera coordinate frame is parallel to the ground plane and its height coincides with the mounting height h of the camera. Then the points $\widehat{\boldsymbol{X}}_j'$, triangulated with $\widehat{\boldsymbol{P}}^{k-1}$ and $\widehat{\boldsymbol{P}}^k$ and the patch correspondences $\boldsymbol{x}_j^{k-1}$ and $\boldsymbol{x}_j^k$ should roughly have the same Y-coordinate (expressed in the coordinate frame of image $k-1$), which corresponds to the height above the road plane in the current scale, i.e. the estimated height of these points depends on the actual translation between the frames. Since we know the height of the camera h and the height of these points, we can rescale the translation vector such that the camera height and the height of the points coincide.

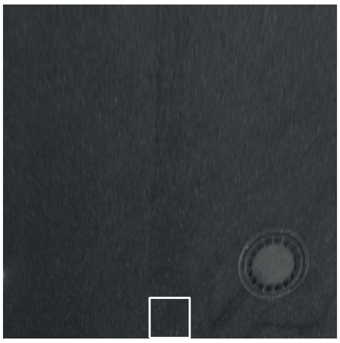

Fig. 3: This figure depicts the synthetic fronto-parallel view of the ROI. The white square marks a single patch that is matched in the preceeding frame.

To robustly estimate the scale factor, we implemented an outlier rejection based on the reprojection error $|\boldsymbol{x}_j^k - \widehat{\boldsymbol{P}}^k \widehat{\boldsymbol{X}}_j|$ to cope with mismatches as well as a simple planarity check to discard estimates of the plane that exceed a threshold in the variance of the Y-coordinates of the triangulated points. The scale factor s^k is determined by

$$s^k = (1 - \alpha) s^{k-1} + \alpha h / \bar{Y}_j^k, \tag{9}$$

with $\alpha \in [0, 1]$ and $\bar{Y}_j^k$ being the mean over all Y-coordinates of $\widehat{\boldsymbol{X}}_j$ triangulated from the image pair k and $k-1$. The factor incorporates the previous scale factor for temporal smoothing of the scale as well as spatial averaging. This smoothing seems reasonable as the drift in scale occurs on a large time-scale, thus making noise rejection a priority over a fast response.

The scale factor s^k is used to scale all camera positions i within the current window as well as the structure observed by the frames in the window. The window for which the scaling is applied consists of all frames $i \in \{S, k\}$, where S is the index corresponding to the oldest keyframe inside the sliding window. As the scale of the motion only affects the translations $\boldsymbol{t}^i$, the scaled translations may be calculated as $\widetilde{\boldsymbol{t}}^i = s^k \left(\boldsymbol{t}^i - \boldsymbol{t}^S\right) + \boldsymbol{t}^S$, where $\boldsymbol{t}^S$ is the translation of the oldest camera pose in the optimization window which is held constant to ensure a smooth trajectory. Analogously, the point cloud is scaled as

$$\widetilde{\boldsymbol{X}}_j = s^k \left(\boldsymbol{X}_j + \left(\boldsymbol{R}^i\right)^{-1} \boldsymbol{t}^i \right) - \left(\boldsymbol{R}^i\right)^{-1} \boldsymbol{t}^i. \tag{10}$$

The proposed scaling of both, the translational camera movements as well as the scene points guarantees a consistently estimated trajectory of the camera as well as the structure of the scene. Obviously, the structure of the scene is too sparse

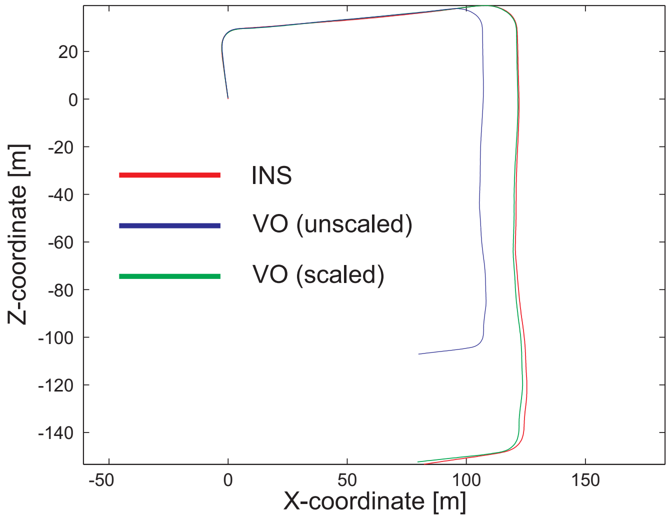

Fig. 4: This figure depicts the results of the proposed algorithm. The red trajectory is the ground truth given by the INS system. The green and blue trajectories are estimated using the visual odometry algorithm. Blue depicts the trajectory with a single scaling at the beginning of the sequence, green is the trajectory with continuously updated scale.

for obstacle avoidance or path planning purposes but sufficient for a robust motion estimation.

V. Experimental Results

For our experiments, we used different datasets captured in real environments. To this end, we used our experimental vehicle, which is equipped with a stereo-camera-rig[1] and a high accuracy integrated navigation system (INS), which combines intertial measurements with a GPS-receiver and wheel speed sensors for measuring motion, pose and orientation of the vehicle. Therefore, the INS yields accurate measurements for the motion of the vehicle along its roll axis and the yaw rate. In the following, these measurements are used as ground truth for our experiments.

The used camera is mounted on top of the car and yields images at a resolution of 1344×391 pixels with $10Hz$. As features, we used Harris corners [7] in combination with a block matching on the image derivatives to get feature correspondences between two adjacent frames. These feature matches are accumulated, to get feature tracks over a series of images.

The results for two challenging datasets with different length and speed can be seen in figure 4 and 5. The first sequence (figure 4) includes 600 frames and was captured with an average speed of approximately $10m/s$. The second dataset (figure 5) with a loopy trajectory consists of 1405 frames, the speed was approximately $15m/s$ on average.

The demonstrated results were computed offline on a standard PC, using an implementation in MATLAB. The length of the sliding windows was set to $L = 10$, optimizing only the $m = 5$ most recent camera poses inside this window. These parameters have been chosen to account for the average track length of approximately four frames. The number of used scene points for the bundle adjustment over the sliding window is approximately 900. This and the good initialization of both, the scene points and the camera poses yield a fast convergence of the bundle adjustment. Per frame, five patches of size 50×50 pixels were matched on the road plane. For the vast majority of patches, the SAD exhibited a distinct minimum which gave rise to accurately triangulated points on the plane. The resulting scale was smoothed with a factor of $\alpha = 0.8$.

Figure 4 displays the results of the proposed approach compared to the ground truth trajectory given by the high accuracy INS (red). Compared to the trajectory which has been scaled only once at the beginning of the sequence (blue), the trajectory using the proposed approach (green) exhibits no significant drift in scale. Obviously, there is an increasing deviation from the ground truth position due to the local nature of the algorithm. Since the drift in scale does not affect the rotational component of the motion estimate, the angular error of the scaled and unscaled trajectories is similar. Only the length of the trajectory is affected by the scaling. Note that the trajectories are manually translated and rotated to align them with the ground truth trajectory.

Currently, the proposed algorithm (without feature tracking) takes approximately $1.7s/frame$ due to the implementation in MATLAB. An implementation in C++, which is planned for future work, should increase the framerate significantly.

VI. Conclusion and Future Work

In this paper we presented an approach for estimating the 6 DoF ego-motion of a vehicle solely from monocular image sequences. By exploiting constraints induced by the known camera mounting and a reasonable assumption about the planarity of the road surface in the vicinity of the vehicle, the algorithm is capable of continuously recovering the scale of the motion. Furthermore we can significantly reduce the drift in scale.

Our experiments have shown, that the scale ambiguity in monocular approaches leads to a large drift in scale. As a result, distant sections of the same estimated trajectory are scaled differently, causing a distortion of the path.

Based on feature tracks over a series of images, we estimate the motion of the camera. To this end, we use a continuously updated scene structure to estimate the pose of the camera based on 2d-3d-correspondences between scene points and image features. Whenever the number of visible scene points in the current image drops below a threshold, we select a new keyframe and refine the previous scene structure and the camera poses in a sliding window based on bundle adjustment. Subsequently, we update the scene structure based on newly triangulated feature tracks. The proposed keyframe selection criterion has proven to be sufficient for the proposed algorithm. Nevertheless, more sophisticated approaches for keyframe selection as proposed e.g. in [22] may be used.

The algorithm employs knowledge about the mounting of the camera and a planar assumption to generate a synthetic

[1] For this work, we use only the left camera of the stereo-camera-rig.

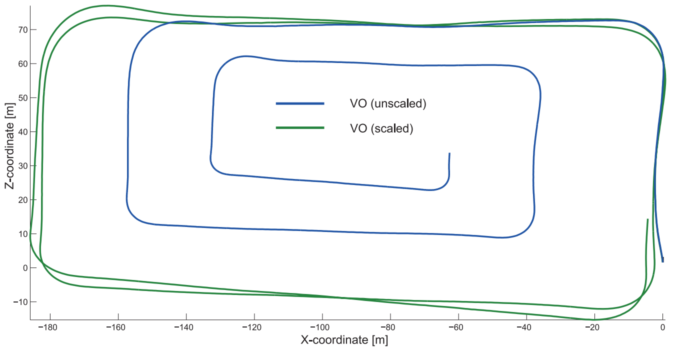

Fig. 5: This figure illustrates the results of the proposed algorithm (green) in comparison to the unscaled version (blue) for a sequence consisting of 1405 frames. As can be seen, the unscaled version suffers from a significant drift in scale. The trajectory exploiting knowledge about the camera mounting exhibits a slight drift due to the local nature of the algorithm. Nevertheless, there is no significant drift in scale.

fronto-parallel view of the road. Despite the lack of strong features on asphalt, patches in successive frames can be matched robustly in this view. Using these matches and the motion hypothesis of the visual odometry, we triangulate points to acquire an estimate of the distance to the ground plane as measured by the visual odometry. Based on the deviation of this measurement from the known mounting height, the algorithm continuously scales the motion and structure of the current window.

The experimental results suggest that the proposed algorithm is capable of accurately reconstructing the trajectory of the vehicle solely based on visual inputs. Furthermore, continuously correcting the scale results in a significant improvement of the accuracy.

Due to the use of only a single camera, the algorithm fails in the presence of dominant independent motion. This may be the case, when moving traffic accounts for a significant share of the image. To improve the robustness of the approach, we are working on a more reliable outlier rejection scheme. The planar assumption is sufficiently fulfilled in most urban scenarios. Nevertheless, large changes in the slope of the road cause our approach to estimate a wrong scaling factor. Although it will recover from such perturbations, the resulting trajectory will be incorrectly scaled around this slope. Another case where the algorithm fails to correctly compute the scale factor is the presence of objects inside the ROI. Future work will include the detection and handling of these conditions as well as a real-time implementation.

Acknowledgements

The first author gratefully acknowledges the support from Karlsruhe House of Young Scientists.

References

[1] Motilal Agrawal and Kurt Konolige. Real-time localization in outdoor environments using stereo vision and inexpensive gps. In *Proceedings of the International Conference on Pattern Recognition*, pages 1063 – 1068, 2006.

[2] Motilal Agrawal and Kurt Konolige. Rough terrain visual odometry. In *Proceedings of the International Conference on Advanced Robotics*, August 2007.

[3] Martin Byröd, Klas Josephson, and Kalle Aström. Fast optimal three view triangulation. In *Proceedings of the Asian Conference on Computer Vision*, pages 549 – 559, 2007.

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1 – 16, June 2007.

[5] Christian Dornhege and Alexander Kleiner. Visual odometry for tracked vehicles. In *Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR 2006)*, 2006.

[6] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381 – 395, 1981.

[7] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proceedings of the Alvey Vision Converence*, pages 147 – 151, 1988.

[8] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in computer vision*. Cambridge University Press, second edition edition, 2008.

[9] Richard I. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580 – 593, June 1997.

[10] Andrew Howard. Real-time stereo visual odometry for autonomous ground vehicles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008)*, pages 3946 – 3952, September 2008.

[11] Andrew Edie Johnson, Steven B. Goldberg, Yang Cheng, and Larry H. Matthies. Robust and efficient stereo feature tracking for visual odometry. In *IEEE International Conference on Robotics and Automation*, pages 39 – 46, May 2008.

[12] Bernd Kitt, Frank Moosmann, and Christoph Stiller. Moving on to dynamic environments: Visual odometry using feature classification. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5551 – 5556, Taipei, Taiwan, October 2010.

[13] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155 – 166, February 2009.

[14] Manolis I. A. Louriakis and Antonis A. Argyros. Sba: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software*, 36(1):1 – 30, March 2009.

[15] Annalisa Milella and Roland Siegwart. Stereo-based ego-motion estimation using pixel-tracking and iterative closest point. In *Proceedings of the IEEE International Conference on Computer Vision Systems*, 2006.

[16] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *IEEE Computer Society Conference Computer Vision and Pattern Recognition*, volume 1, pages 652 – 659, 2004.

[17] Frank Pagel. Robust monocular egomotion estimation based on an iekf. In *Canadian Conference on Computer and Robot Vision*, pages 213 – 220, Kelowna, BC, Canada, May 2009.

[18] Davide Scaramuzza, Friedrich Fraundorfer, and Roland Siegwart. Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In *Proceedings of the IEEE Conference on Robotics and Automation*, May 2009.

[19] Jianbo Shi and Carlo Tomasi. Good features to track. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 1994)*, pages 593 – 600, June 1994.

[20] Hauke Stradsdat, J. M. M. Montiel, and Andrew J. Davison. Scale drift-aware large scale monocular slam. In *Proceedings of the Robotics: Science and Systems Conference*, Zaragoza, Spain, June 2010.

[21] Ashit Talukder and Larry Matthies. Real-time detection of moving objects from moving vehicles using dense stereo and optical flow. In *IEEE International Conference on Intelligent Robots and Systems (IROS 2004)*, volume 4, pages 3718 – 3725, September 2004.

[22] Thorsten Thormählen, Hellward Broszio, and Axel Weissenfeld. Keyframe selection for camera motion and structure estimation from multiple views. In *Proceedings of the European Conference on Computer Vision*, pages 523 – 535, Prague, Czech Republic, May 2004.

[23] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. Bundle adjustment – a modern synthesis. *Lecture Notes in Compute Sciene*, 1883:298 – 372, 2000.

[24] Koichiro Yamaguchi, Takeo Kato, and Yoshiki Ninomiya. Vehicle ego-motion estimation and moving object detection using a monocular camera. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR 2006)*, pages 610 – 613, 2006.