Robust Monocular Visual Odometry for a Ground Vehicle in Undulating Terrain

Ji Zhang, Sanjiv Singh, and George Kantor

Abstract Here we present a robust method for monocular visual odometry capable of accurate position estimation even when operating in undulating terrain. Our algorithm uses a steering model to separately recover rotation and translation. Robot 3DOF orientation is recovered by minimizing image projection error, while, robot translation is recovered by solving an NP-hard optimization problem through an approximation. The decoupled estimation ensures a low computational cost. The proposed method handles undulating terrain by approximating ground patches as locally flat but not necessarily level, and recovers the inclination angle of the local ground in motion estimation. Also, it can automatically detect when the assumption is violated by analysis of the residuals. If the imaged terrain cannot be sufficiently approximated by locally flat patches, wheel odometry is used to provide robust estimation. Our field experiments show a mean relative error of less than 1%.

1 Introduction

The task of visual odometry is to estimate motion of a camera, and by association the vehicle it is attached to, using a sequence of camera images. Typically, visual odometry is used in those cases where GPS is not available (eg. in planetary environments), or is too heavy carry (eg. on a small air vehicle), or, is insufficiently accurate at a low cost (eg. in agricultural applications). In ground vehicle applications, visual odometry can provide an alternative or compliment to wheel odometry since it is not prone to problems such as wheel slippage that can cause serious errors. Recent developments show significant progress in visual odometry and it now possible to estimate 6DOF motion using stereo cameras [1–3]. Stereo cameras help provide scale and some constraints to help recovery of motion but their use comes at a cost. Accuracy is dependant on inter-camera calibration which can be hard to ensure if the cameras are separated significantly. The use of stereo cameras also reduces the

Ji Zhang (zhangji@andrew.cmu.edu), Sanjiv Singh (ssingh@cmu.edu), and George Kantor (kantor@cmu.edu) are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA

Fig. 1 (a) An example of the type of terrain over which our ground vehicle based visual odometry is intended to work (b) An example of the type of scene that can be imaged by the visual odometry system. Monocular visual odometry systems that assume a flat environments fail in such a case.

field of view because only features that lie in the intersection of the field of view of two cameras can be used. Finally, cost in components, interfacing, synchronization, and computing are higher for stereo cameras compared to a monocular camera.

While it is impossible to recover scale in translation for arbitrary camera motion in 6DOF when using monocular imaging, it is possible to recover scale when some additional information such as the distance and attitude of camera from the ground plane, such as is reasonably constant on a ground vehicle, is available. Recent work shows that under the assumption that the imaged areas are flat and level, it is possible to use visual odometry with monocular imaging [4–6]. This is a significant constraint in that such methods fail if the imaged areas are not guaranteed to be flat.

Here we report on relaxing the constraint such that visual odometry coupled with wheel odometry can be viable in undulating and even in severely 3D settings (Fig. 1) using monocular vision. We do this in two ways. First, our formulation of visual odometry only requires the imaged areas to be locally flat but not necessarily level. Our method recovers the ground inclination angle by finding coplanar features tracked on the ground. Second, the method can automatically determine when the imaged areas are not well approximated by locally flat patches and uses wheel odometry. The result is a monocular system that recovers differential motion with non-holonomic constraint in 3DOF rotation and 1DOF translation. When used on a ground vehicle, our experiments indicate an accuracy comparable to that from state-of-the-art stereo systems even the vehicle is tested in undulating terrain.

To estimate motion from imagery, the standard way is formulating visual odometry into a bundle adjustment problem and solves numerically through iteration. Alternatively, by using a steering model, the proposed method decouples the problems of estimating rotation and translation. In the first step, we estimate robot orientation using QR factorization [7] applied to a RANSAC algorithm [8] that minimizes the image reprojection error. In the second step, we use the same set of inlier features found by the RANSAC algorithm and solve an optimization problem that recovers translation together with the ground inclination angles. Since the full blown problem is believed to be NP hard, we utilize an approximation that ensures computational feasibility. The proposed two-step estimation algorithm is able to run with very low computational cost. Further, if the ground patches cannot be approximated as locally flat, the second step estimation becomes inaccurate. Then, wheel odometry is used to compute translation, and visual odometry is only for recovering rotation.

The rest of this paper is organized as follows. In section 2, we present related work. In section 3, we define our problem. The problem is mathematically solved in Section 4 with implementation details provided. Experimental results are shown in Section 5 and conclusions are made in Section 6.

2 Related Work

Today, it is commonly possible to estimate camera motion using visual odometry, that is through the tracking of features in an image sequence. [2, 3]. Typically, the camera motion is assumed to be unconstrained in the 3D space. For stereo systems [9–11], the baseline between the two cameras functions as a reference from which the scale of motion can be recovered. For example Paz, et al's method estimates the motion of stereo hand-hold cameras where scale is solved using features close to the cameras [12]. Konolige, at al's stereo visual odometry recovers 6DOF camera motion from bundle adjustment [1]. The method is integrated with an IMU that handles the orientational drift of visual odometry. It is able to work for lone distance navigation in off-road environments. For monocular systems [13–15], if camera motion is unconstrained, scale ambiguity is unsolvable. Using a monocular camera, Civera, et al formulate the motion estimation and camera calibration into one problem [16]. The approach recovers camera intrinsic parameters and 6DOF motion up to scale.

When a monocular system is used in such a way that the camera motion is constrained to a surface, recovering scale is possible. For example, Kitt, et al's method solves scale ambiguity using Ackermann steering model and assumes the vehicle drives on a planar road surface [5]. Nourani Vatani and Borges use Ackermann steering model along with a downward facing camera to estimate the planar motion of a vehicle [6]. Since the method only recovers the vehicle planar motion, an INS system is used to obtain vehicle pitch and roll angles. Scaramuzza, et al's approach adopts a single omnidirectional camera [4], where Ackermann steering model and steering encoder readings are used as constrains. This approach can recover motion at a low computational cost with a single feature point, and shows significantly improved accuracy compared to unconstrained cases. Scaramuzza also shows that a monocular camera placed with an offset to the vehicle rotation center can recover scale when the vehicle is turning [17]. In straight driving, however, the formulation degenerates and the scale is no longer recoverable.

In [4–6, 17], the methods all assume a planar ground model. However, violation of the assumption can make motion estimation fail. Compared to the existing work, our method does not require the imaged terrain to be flat and level. Our method simultaneously estimates the inclination angle of the ground while recovering motion. Further, our method combines wheel odometry to deal with the case where the system automatically determines if the terrain cannot be well approximated by a local flat patch. Here, we summarize our theoretical analysis of the motion estimation due to space limitations. A more complete analysis will be published in the future.

3 Problem Definition

We assume that the vehicle uses Ackermann steering [18] which limits the steering to be perpendicular to the axles of the robots. We also assume that the camera is well modeled as a pinhole camera [7] in which the intrinsic and extrinsic parameters are calibrated.

3.1 Notations and Coordinate Systems

As a convention in this paper, we use right uppercase superscription to indicate the coordinate systems, and right subscription k, $k \in Z^+$ to indicate the image frames. We use $\mathscr{I}$ to denote the set of feature points in the image frames.

- Camera coordinate system $\{C\}$ is a 3D coordinate system. As shown in Fig. 2, the origin of $\{C\}$ is at the camera optical center with the z-axis coinciding with the camera principal axis. The $x-y$ plane is parallel to the camera image sensor with the x-axis parallel to the horizontal direction of the image pointing to the left. A point i, $i \in \mathscr{I}$, in $\{C_k\}$ is denoted as $\boldsymbol{X}^C_{(k,i)}$.
- Vehicle coordinate system $\{V\}$ is a 3D coordinate system. The origin of $\{V\}$ is coinciding with the origin of $\{C\}$, the x-axis is parallel to the robot axles pointing to the robot left hand side, the y-axis is pointing upward, and the z-axis is pointing forward. A point i, $i \in \mathscr{I}$, in $\{V_k\}$ is denoted as $\boldsymbol{X}^V_{(k,i)}$.
- Image coordinate system $\{I\}$ is a 2D coordinate system with its origin at the right bottom corner of the image. The u- and v- axes in $\{I\}$ are pointing to the same directions as the x- and y- axes in $\{C\}$. A point i, $i \in \mathscr{I}$, in $\{I_k\}$ is $\boldsymbol{X}^I_{(k,i)}$.

3.2 Problem Description

Since our robot remains on the ground and follows the Ackermann steering model, the translation is limited to the z-direction in $\{V\}$. Let Δz be robot translation between frames $k-1$ and k, Δz is in the $\{V_{k-1}\}$ coordinates. In this paper, we treat the features on the ground in the near front of the robot as coplanar. As shown in Fig. 3,

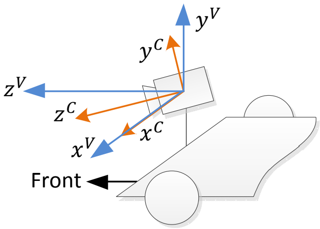


Fig. 2 Illustration of the vehicle coordinate system $\{V\}$ and the camera coordinate system $\{C\}$.

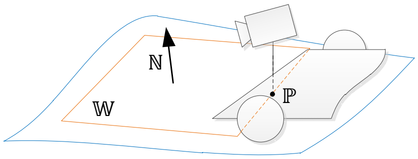


Fig. 3 Modeling the ground. The blue colored curve represents the ground, ℙ is the projection of the camera center, and 𝕎 is the plane representing the ground in the near front of the robot. 𝕎 has pitch and roll DOFs around ℙ.

let $\mathbb{W}$ indicate the plane. Let d_0 be the height of the camera above the ground, d_0 is set as a known constant. Let $\mathbb{P}$ be the projection of the camera center. We model $\mathbb{W}$ with 2 rotational DOFs around $\mathbb{P}$. Let $\mathbb{N}$ be the normal of $\mathbb{W}$, and let t_k and r_k be the Euler angles of $\mathbb{N}$ around the x- and z- axes in $\{V_k\}$, respectively. t_k and r_k represent the pitch and roll inclination angles of the ground. Let Δp, Δt, and Δr be robot rotation angles around the y-, x-, and z- axes of $\{V_{k-1}\}$ between frames $k-1$ and k, we have $\Delta t = t_k - t_{k-1}$ and $\Delta r = r_k - r_{k-1}$. In this paper, we want to measure the robot motion between consecutive frames. Our visual odometry problem can be defined as

Problem 1 *Given a set of image frames k, $k \in Z^+$, and the camera height d_0, compute Δp, Δt, Δr, and Δz for each frame k.*

4 Visual Odometry Algorithm

4.1 Rotation Estimation

In this section, we recover the 3DOF robot orientation. We will show that by using the Ackermann steering model, robot orientation can be recovered regardless of translation. From the pin-hole camera model, we have the following relationship between $\{I\}$ and $\{C\}$,

$$\varsigma \boldsymbol{X}^{I}_{(k,i)} = \mathbf{K}\boldsymbol{X}^{C}_{(k,i)}, \tag{1}$$

where ς_k is a scale factor, and $\mathbf{K}$ is the camera intrinsic matrix, which is known from the pre-calibration [7].

The relationship between $\{C\}$ and $\{V\}$ is expressed as

$$\boldsymbol{X}^{C}_{(k,i)} = \mathbf{R}_z(r_0)\mathbf{R}_x(t_0)\mathbf{R}_y(p_0)\boldsymbol{X}^{V}_{(k,i)}, \tag{2}$$

where $\mathbf{R}_x(\cdot)$, $\mathbf{R}_y(\cdot)$, and $\mathbf{R}_z(\cdot)$ are rotation matrices around the x-, y-, and z- axes in $\{V\}$, respectively, and p_0, t_0, and r_0 are corresponding rotation angles from $\{V\}$ to $\{C\}$. Here, note that p_0, t_0, and r_0 are the camera extrinsic parameters, which are known from the pre-calibration [7].

Let $\tilde{\boldsymbol{X}}^{V}_{(k,i)}$ be the normalized term of $\boldsymbol{X}^{V}_{(k,i)}$, we have

$$\tilde{\boldsymbol{X}}^{V}_{(k,i)} = \boldsymbol{X}^{V}_{(k,i)} / z^{V}_{(k,i)}. \tag{3}$$

where $z^V_{(k,i)}$ is the 3rd entry of $\boldsymbol{X}^V_{(k,i)}$. $\tilde{\boldsymbol{X}}^V_{(k,i)}$ can be computed by substituting (2) into (1) and scaling $\boldsymbol{X}^V_{(k,i)}$ such that the 3rd entry becomes one.

From the robot motion, we can establish a relationship between $\{V_{k-1}\}$ and $\{V_k\}$ as follows,

$$\boldsymbol{X}^V_{(k,i)} = \mathbf{R}_z(\Delta r)\mathbf{R}_x(\Delta t)\mathbf{R}_y(\Delta p)\boldsymbol{X}^V_{(k-1,i)} + [0,\ 0,\ \Delta z]^T\,, \tag{4}$$

where $\mathbf{R}_x(\cdot)$, $\mathbf{R}_y(\cdot)$, and $\mathbf{R}_z(\cdot)$ are the same rotation matrices as in (2).

Substituting (3) into (4) for frame $k-1$ and k, and since Δp, Δt, and Δr are small angles in practice, we perform linearization to obtain the following equations,

$$c_i \tilde{x}^V_{(k,i)} = \tilde{x}^V_{(k-1,i)} + \Delta p + \tilde{y}^V_{(k-1,i)}\Delta r, \tag{5}$$

$$c_i \tilde{y}^V_{(k,i)} = \tilde{y}^V_{(k-1,i)} + \Delta t - \tilde{x}^V_{(k-1,i)}\Delta r, \tag{6}$$

$$c_i = 1 - \tilde{x}^V_{(k-1,i)}\Delta p - \tilde{y}^V_{(k-1,i)}\Delta t + \Delta z / z^V_{(k-1,i)}, \tag{7}$$

where $\tilde{x}^V_{(l,i)}$ and $\tilde{y}^V_{(l,i)}$, $l = k-1, k$, are the 1st and the 2nd entries of $\tilde{\boldsymbol{X}}^V_{(l,i)}$, respectively, $z^V_{(l,i)}$ is the 3rd entry of $\boldsymbol{X}^V_{(l,i)}$, and c_i is a scale factor, $c_i = z^V_{(k,i)} / z^V_{(k-1,i)}$.

Eq. (5) and (6) describe a relationship of Δp, Δt, and Δr without interfering with Δz. This indicates that by using the Ackermann steering model, we can decouple the estimation problem and recover Δp, Δt, and Δr separately from Δz. Stacking (5) and (6) for different features, we have

$$\mathbf{A}\boldsymbol{X} = \boldsymbol{b}, \tag{8}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \tilde{y}^V_{(k-1,1)} & -\tilde{x}^V_{(k,1)} & 0 & 0 & \dots \\ 0 & 1 & -\tilde{x}^V_{(k-1,1)} & -\tilde{y}^V_{(k,1)} & 0 & 0 & \dots \\ 1 & 0 & \tilde{y}^V_{(k-1,2)} & 0 & -\tilde{x}^V_{(k,2)} & 0 & \dots \\ 0 & 1 & -\tilde{x}^V_{(k-1,2)} & 0 & -\tilde{y}^V_{(k,2)} & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix},$$

$$\boldsymbol{b} = -\left[\tilde{x}^V_{(k-1,1)},\ \tilde{y}^V_{(k-1,1)},\ \tilde{x}^V_{(k-1,2)},\ \tilde{y}^V_{(k-1,2)},\ \dots\right]^T,$$

$$\boldsymbol{X} = [\Delta p,\ \Delta t,\ \Delta r,\ c_1,\ c_2,\ \dots]^T.$$

Eq. (8) can be solved using the QR factorization method. Since $\mathbf{A}$ is a sparse matrix, the QR factorization can be implemented very efficiently. Let $\tilde{x}'^V_{(k-1,i)}$ and $\tilde{y}'^V_{(k-1,i)}$ be the reprojected coordinates of $\tilde{x}^V_{(k,i)}$ and $\tilde{y}^V_{(k,i)}$ in $\{V_{k-1}\}$. The QR factorization minimizes the image reprojection error,

$$\min_{\substack{\Delta p, \Delta t, \Delta r, \\ c_i,\, i \in \mathscr{I}}} \sum_{i \in \mathscr{I}} (\tilde{x}^V_{(k-1,i)} - \tilde{x}'^V_{(k-1,i)})^2 + (\tilde{y}^V_{(k-1,i)} - \tilde{y}'^V_{(k-1,i)})^2. \tag{9}$$

With (8) solved, let $e^x_{(k-1,i)} = \tilde{x}^V_{(k-1,i)} - \tilde{x}'^V_{(k-1,i)}$ and $e^y_{(k-1,i)} = \tilde{y}^V_{(k-1,i)} - \tilde{y}'^V_{(k-1,i)}$, $e^x_{(k-1,i)}$ and $e^y_{(k-1,i)}$ represent the reprojection errors of feature i, $i \in \mathscr{I}$, in $\{V_{k-1}\}$. Using (5) and (6), we can compute $e^x_{(k-1,i)}$ and $e^y_{(k-1,i)}$ as,

$$e^x_{(k-1,i)} = \tilde{x}^V_{(k-1,i)} + \Delta p + \tilde{y}^V_{(k-1,i)} \Delta r - c_i \tilde{x}^V_{(k,i)}, \tag{10}$$

$$e^y_{(k-1,i)} = \tilde{y}^V_{(k-1,i)} + \Delta t - \tilde{x}^V_{(k-1,i)} \Delta r - c_i \tilde{y}^V_{(k,i)}. \tag{11}$$

Similarly, let $e^x_{(k,i)}$ and $e^y_{(k,i)}$ be the reprojection errors in $\{V_k\}$, $e^x_{(k,i)}$ and $e^y_{(k,i)}$ can be obtained as

$$e^x_{(k,i)} = e^x_{(k-1,i)}/c_i, \; e^y_{(k,i)} = e^y_{(k-1,i)}/c_i. \tag{12}$$

Define $\Sigma_{(l,i)}$, $l \in \{k-1, k\}$, as a 2×2 matrix,

$$\Sigma_{(l,i)} = \mathrm{diag}\left[(e^x_{(l,i)})^2, (e^y_{(l,i)})^2 \right], \; l \in \{k-1, k\}. \tag{13}$$

$\Sigma_{(l,i)}$ contains the covariance of $\tilde{\boldsymbol{X}}^V_{(l,i)}$ measured from the image reprojection error, which will be useful in the following sections.

4.2 Robot Translation

With robot orientation recovered, we derive the expression of translation in this section. The task of recovering translation is formulated into an optimization problem in the next section, and solved in the same section. An shown in Fig. 3, recall that $\mathbb{W}$ is the plane representing the local ground in the near front of the robot, and t_k and r_k are the pitch and roll angles of $\mathbb{W}$. For a feature i, $i \in \mathscr{I}$, on $\mathbb{W}$, the following relationship holds from geometric relationship,

$$-z^V_{(k,i)} (\tilde{y}^V_{(k,i)} - \tan r_k \tilde{x}^V_{(k,i)} + \tan t_k) = d_0, \tag{14}$$

where d_0 is the height of the camera above the ground.

Since t_k and r_k are small angles in practice, we approximate $\tan t_k \approx t_k$ and $\tan r_k \approx r_k$. Then, by substituting (14) into (7) for frames $k-1$ and k, we can derive

$$\alpha t_k + \beta r_k = \gamma, \tag{15}$$

where

$$\begin{aligned}
\alpha &= -\tilde{x}^V_{(k-1,i)} \Delta p + \tilde{y}^V_{(k-1,i)} \Delta t - c_i + 1, \\
\beta &= (\alpha + 1) \tilde{x}^V_{(k,i)} - \tilde{x}^V_{(k-1,i)}, \\
\gamma &= -(\alpha + 1)(\Delta t + \tilde{x}^V_{(k,i)} \Delta r - \tilde{y}^V_{(k,i)}) - \tilde{y}^V_{(k-1,i)}.
\end{aligned}$$

Eq. (15) contains two unknown parameters, t_k and r_k, which indicates that we can solve the function by using two features. Let (i, j) be a pair of features, $i, j \in \mathscr{I}$, here we use (i,j) to solve (15). Then, let $\Delta z_{(i,j)}$ be the translation computed from feature pair (i,j). From (14), we can derive

$$\Delta z_{(i,j)} = \frac{1}{2}(T_{(k,i)} + T_{(k,j)} - T_{(k-1,i)} - T_{(k-1,j)}), \tag{16}$$

where

$$T_{(l,h)} = d/(\tilde{y}^V_{(l,h)} + \tilde{x}^V_{(l,h)} r_k + t_k),\ l \in \{k-1,k\},\ h \in \{i,j\},$$

Now, let $\sigma_{(i,j)}$ be the standard deviation of $\Delta z_{(i,j)}$ measured from the image reprojection error, $\sigma_{(i,j)}$ will be useful in the next section. From (16), it indicates that $\Delta z_{(i,j)}$ is a function of $\tilde{\boldsymbol{X}}^V_{(l,h)}$, $l \in \{k-1,k\},\ h \in \{i,j\}$. Let $\mathbf{J}_{(l,h)}$ be the Jacobian matrix of that function with respect to $\tilde{\boldsymbol{X}}^V_{(l,h)}$, $\mathbf{J}_{(l,h)} = \partial \Delta z_{(i,j)} / \partial \tilde{\boldsymbol{X}}^V_{(l,h)}$, we can compute

$$\sigma^2_{(i,j)} = \sum_{l \in \{k-1,k\}} \sum_{h \in \{i,j\}} \mathbf{J}_{(l,h)} \Sigma_{(l,h)} \mathbf{J}^T_{(l,h)}, \tag{17}$$

4.3 Translation Recovery by Optimization

In the above section, we showed that the translation can be recovered using a pair of features. In this section, we want to estimate translation using multiple features, by solving an optimization problem that minimizes the error variance of translation estimation. Suppose we have a total number of n features, $n \in Z^+$, combination of any two features can provide $n(n-1)/2$ feature pairs. Let $\mathscr{J}$ be a set of feature pairs, $1 \leq |\mathscr{J}| \leq n(n-1)/2$. Here, we use the feature pairs in $\mathscr{J}$ to compute the translation Δz. Define Δz as the weighted sum of $\Delta z_{(i,j)}$, $(i,j) \in \mathscr{J}$,

$$\Delta z = \sum_{(i,j) \in \mathscr{J}} w_{(i,j)} \Delta z_{(i,j)}, \tag{18}$$

where $w_{(i,j)}$ is the weight for feature pair (i,j), such that

$$\sum_{(i,j) \in \mathscr{J}} w_{(i,j)} = 1, \text{ and } w_{(i,j)} \geq 0,\ (i,j) \in \mathscr{J}. \tag{19}$$

Define σ as the standard deviation of Δz measured from the image reprojection error. Here, we want to compute Δz such that σ is minimized. We start with our first question. For a given set of feature pairs $\mathscr{J}$, how to assign the weights $w_{(i,j)}$, $(i,j) \in \mathscr{J}$, such that σ is the minimum? Mathematically, the problem can be expressed as,

Problem 2 *Given $\sigma_{(i,j)}$, $(i,j) \in \mathscr{J}$, compute*

$$\{w_{(i,j)},\ (i,j) \in \mathscr{J}\} = \arg\min_{w_{(i,j)}} \sigma^2, \tag{20}$$

subject to the constraints in (19).

To solve this problem, we can prove that if each feature i, $i \in \mathscr{I}$ belongs to at most one feature pair in $\mathscr{J}$, then Problem 2 is analytically solvable using the Lagrange multiplier method [19]. However, if a feature exists in multiple feature pairs, the problem becomes a convex optimization problem that has to be solved numerically [20]. Here, we directly give the solution for Problem 2,

$$\min_{w_{(i,j)}} \sigma^2 = \sum_{(i,j) \in \mathscr{J}} w_{(i,j)}^2 \sigma_{(i,j)}^2, \tag{21}$$

where

$$w_{(i,j)} = \frac{1/\sigma_{(i,j)}^2}{\sum_{(p,q) \in \mathscr{J}} 1/\sigma_{(p,q)}^2}, \ (i,j) \in \mathscr{J}. \tag{22}$$

With Problem 2 solved, we come to our second question. How to select the feature pairs in $\mathscr{J}$ such that σ is the minimum? Mathematically, the problem is

Problem 3 *Given $\mathscr{I}$ and $\sigma_{(i,j)}$, $i,j \in \mathscr{I}$, determine*

$$\{\mathscr{J} = \{(i,j)\}, \ i,j \in \mathscr{I}\} = \arg\min_{\mathscr{J}} (\min_{w_{(i,j)}} \sigma^2), \tag{23}$$

such that each feature i, $i \in \mathscr{I}$ belongs to at most one feature pair in $\mathscr{J}$.

Problem 3 can be reformulated into a balanced graph partition problem [21], which is believed to be NP-hard [22]. Here, we focus on an approximation algorithm. The following two inequalities help us to construct the approximation algorithm. First, we find a sufficient condition for selecting the feature pairs. For feature pair (i,j), $i,j \in \mathscr{I}$, if the following inequality is satisfied, then $(i,j) \in \mathscr{J}$,

$$\frac{1}{\sigma_{(i,j)}^2} > \frac{1}{\sigma_{(i,q)}^2} + \frac{1}{\sigma_{(p,j)}^2}, \ \forall p,q \in \mathscr{I}, \ p,q \neq i,j. \tag{24}$$

Second, we find that if we select the feature pairs (i,j), $i,j \in \mathscr{I}$ in the increasing order of $\sigma_{(i,j)}$, we can obtain a set of feature pairs, let it be $\tilde{\mathscr{J}}$, and let $\tilde{\sigma}$ be the standard deviation of Δz computed using feature pairs in $\tilde{\mathscr{J}}$. Let σ_* be the standard deviation of solving Problem 3 without approximation, we can prove that,

$$\tilde{\sigma}^2 \leq 2\sigma_*^2. \tag{25}$$

Eq. (25) indicates that we can solve Problem 3 with an approximation factor of 2. Consequently, the feature pair selection algorithm is shown in Algorithm 1. In Line 5, we first sort the feature pairs in the increasing order of $\sigma_{(i,j)}$, $i,j \in \mathscr{I}$. Then in Lines 6-14, we go through each feature pair and check if (24) is satisfied. If yes, the feature pair is selected. Then, in Lines 15-19, we select the rest of the feature pairs in the increasing order of $\sigma_{(i,j)}$, $i,j \in \mathscr{I}$. The algorithm returns $\mathscr{J}$ in Line 20.

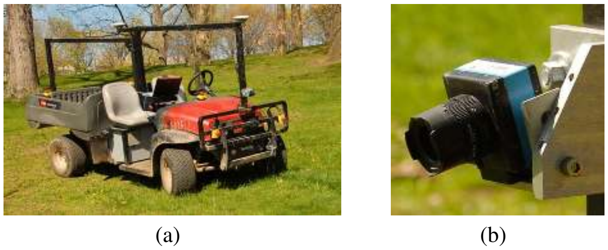
Algorithm 1: Feature Pair Selection

```
input : $\mathscr{I}$ and $\sigma_{(i,j)}$, $i, j \in \mathscr{I}$
output : $\mathscr{J}$
begin
    $\mathscr{J} = \emptyset$;
    Sort $\sigma_{(i,j)}$, $i, j \in \mathscr{I}$ in increasing order;
    Create a variable $\sigma_i$ for each $i \in \mathscr{I}$;
    for the decreasing order of $\sigma_{(i,j)}$, $i, j \in \mathscr{I}$ do
        $\sigma_i = \sigma_{(i,j)}$, $\sigma_j = \sigma_{(i,j)}$;
    end
    for each $i, j \in \mathscr{I}$ do
        if $1/\sigma^2_{(i,j)} > 1/\sigma^2_i + 1/\sigma^2_j$ then
            Put $(i, j)$ in $\mathscr{J}$, then delete $i, j$ from $\mathscr{I}$;
        end
    end
    for the increasing order of $\sigma_{(i,j)}$, $i, j \in \mathscr{I}$ do
        if $i, j \in \mathscr{I}$ then
            Put $(i, j)$ in $\mathscr{J}$, then delete $i, j$ from $\mathscr{I}$;
        end
    end
    Return $\mathscr{J}$.
end
```

4.4 Implementation and Hybrid with Wheel Odometry

To implement the algorithm, we select a number of ”good features” with the local maximum eigenvalues using the openCV library, and track the feature points between consecutive frames using the Lucas Kanade Tomasi (LKT) method [23]. To estimate robot rotation, we solve (8) using QR factorization method. The QR factorization is applied to a RANSAC algorithm that iteratively selects a subset of the tracked features as inliers, and uses the inliers to recover the 3DOF rotation, namely Δp, Δt, and Δr. After recovering the rotation, we also obtain the error covariance for each feature point from (13). Using the inliers selected by the RANSAC algorithm and the corresponding error covariance, we can select the feature pairs based on Algorithm 1 and recover robot translation Δz based on (18), (16), and (22).

In the two-step estimation process, the translation estimation requires the ground patches to be locally flat, while the rotation estimation does not rely on such requirement. Therefore, when this requirement is violated, the translation estimation becomes inaccurate. To deal with this case, a checking mechanism is implemented. If the error variance σ^2, the ground inclination angle t_k or r_k is larger than a corresponding threshold, a hybrid odometry system is used. The wheel odometry is used for computing translation, and the visual odometry is for recovering rotation. This strategy allows the system to work robustly even when the camera field of view is blocked by obstacles.



(a) (b)

Fig. 4 (a) Our robot, and (b) A monocular camera attached in front of the robot.

5 Experiments

We conduct experiments using an electrical vehicle as shown in Fig. 4(a). The vehicle measures 3.04m in length and 1.50m in width. The wheelbase of the vehicle is 2.11m. The vehicle is embedded with wheel encoders that measures the driving speed. An Imagingsource DFK 21BUC03 camera is attached in front of the vehicle, as shown in Fig. 4(b). The camera resolution is set at 640×480 pixels and the focal length is 4mm (horizontal filed of view 64°). The vehicle is equipped with a high accuracy INS/GPS system (Applanix Pos-LV), accurate to better than 10 cm for ground truth acquisition.

5.1 Computation Time

We first show computation time of the proposed visual odometry algorithm. The algorithm is tested on a laptop computer with Quad 2.5GHz CPUs and 6G RAM. We track 300 features at each frame. As shown in Table 1, the feature tracking takes 38ms and consumes an entire core. The state estimation takes 5ms and runs on another core. The proposed algorithm is able to run at 26Hz on average.

5.2 Accuracy of Test Results

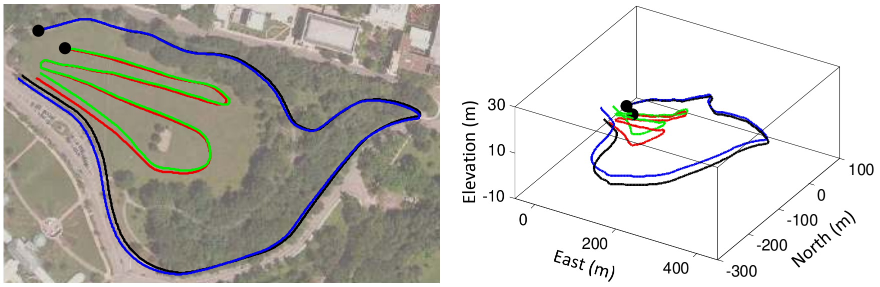
To demonstrate the accuracy of the proposed visual odometry algorithm, we conduct experiments with relatively long driving distance. The test configuration is shown in Table 2. The experiments are conducted with different elevation change and ground material. The overall driving distance for the 6 tests is about 5km. The mean relative error of the visual odometry is 0.83%. Specifically, the trajectories of Test 1-2 are presented in Fig. 5.

Table 1 Computation time of the visual odometry algorithm using 300 features.

Feature Tracking	State Estimation	Overall
38ms	5ms	43ms

Table 2 Accuracy test configuration and relative error computed from 3D coordinates.

Test	Configuration			Relative
No.	Driving Distance	Elevation Change	Ground Material	Error
1	903m	18m	Grass	0.71%
2	1117m	27m	Asphalt	0.87%
3	674m	15m	Grass+Soil	0.74%
4	713m	17m	Concrete	1.13%
5	576m	21m	Asphalt	0.76%
6	983m	13m	Soil	0.81%



(a) Planar view (b) 3D view

Fig. 5 (a) Planar view and (b) 3D view of the robot trajectories in accuracy test 1-2 (Table 1). The black colored dots are the starting points. The green colored curve is the visual odometry output for Test 1, and the red colored curve is the corresponding ground truth. The blue colored curve is the visual odometry output for Test 2, and the black colored curve is the ground truth. Ground truth is measured by a high accuracy INS/GPS system.

5.3 Experimental Results

To test the robustness of the proposed method, we conduct experiments with obstacles on the driving path. When the camera field of view is blocked by an obstacle, the requirement on local flatness of the ground pathes is violated. In this case, a hybrid odometry system is used. The translation is measured by wheel odometry and the rotation is estimated by visual odometry. As shown in Table 3, the robustness tests are conducted with different number of obstacles. By using the hybrid odometry system, the relative error is kept much lower than using the visual odometry only. Specifically, the trajectories and obstacles of Test 1 are shown in Fig. 6.

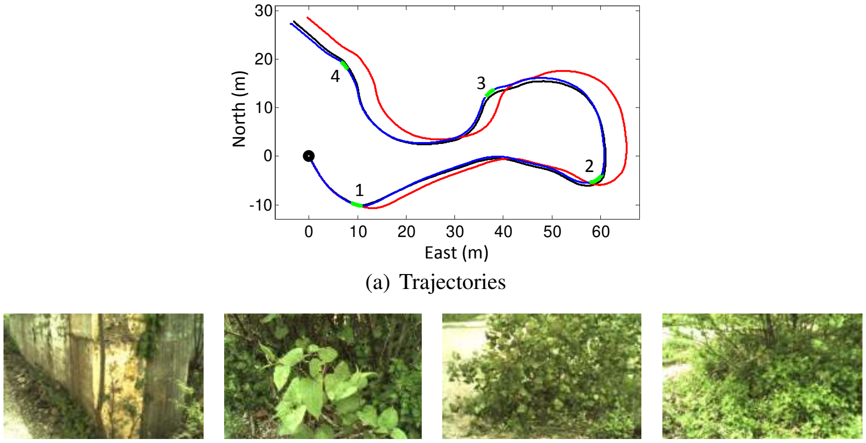
5.4 Analysis of Optimization

Finally, we analyze the effectiveness of the optimization procedure in Section 4.3. We compare three different versions of visual odometry algorithms as follows.

1. Visual Odometry (**VO**): The proposed visual odometry algorithm of this paper.

Table 3 Robustness test configuration and relative error computed from 3D coordinates.

Test	Configuration		Relative Error	
	Driving	Obstacle	Visual+Wheel	Visual
No.	Distance	No.	Odometry	Odometry
1	167m	4	0.43%	1.83%
2	124m	3	0.39%	2.46%
3	182m	4	0.54%	1.54%
4	263m	6	0.61%	4.13%
5	106m	3	0.47%	2.76%
6	137m	5	0.41%	3.81%



(a) Trajectories

(b) Obstacle 1 (c) Obstacle 2 (d) Obstacle 3 (e) Obstacle 4

Fig. 6 (a) Robot trajectories for robustness test 1 (Table 2). The test includes 4 obstacles labeled with numbers. The corresponding obstacles are shown in (b)-(e). The black colored dot is the starting point. The blue-green colored curve is measured by the hybrid odometry system, the blue colored segments are measured by visual odometry and the green colored segments are measured by visual odometry for rotation and wheel odometry for translation, the red colored curve is measures by visual odometry only, and the black colored curve is the ground truth.

2. Visual Odometry Random Pair Selection (**VORPS**): In this version, we turn off the feature pair selection and use randomly selected the feature pairs. By using this algorithm, we can inspect the effect of Problem 3.
3. Visual Odometry Equal Weight (**VOEW**): In this version, we completely turn off the optimization and use equal weights instead of optimized weights in (18). By doing this, we can inspect the effect of Problem 2.

For comparison, we define two evaluation metrics. Let $\bar{\sigma}$ as the mean standard deviation of the one-step translation Δz, and let $\bar{\varepsilon}$ be the mean relative error of the visual odometry output, $\bar{\sigma}$ and $\bar{\varepsilon}$ are computed using combination of the data in Table 2. Comparison of the results is presented in Fig. 7. Since $\bar{\sigma}$ of VOEW is significantly larger than that of VO or VORPS, we have to show the full scaled

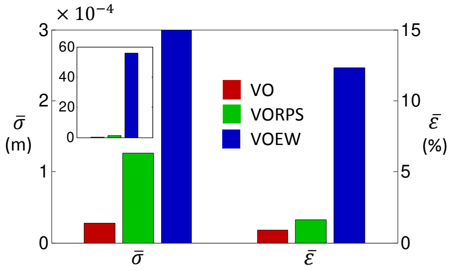


Fig. 7 Comparison of 3 different versions of the visual odometry. VO is the proposed visual odometry algorithm of this paper. VORPS is another version without the feature pair selection, randomly selected feature pairs are used. VOEW uses equal weights instead of optimized weights in (18). $\bar{\sigma}$ is the mean standard deviation of the one-step translation Δz. A full scaled comparison of $\bar{\sigma}$ is shown in the small thumbnail at the left-top corner. $\bar{\varepsilon}$ is the mean relative error of the visual odometry. The results are obtained using combination of the data in Table 2.

comparison in a small thumbnail at the left-top corner of the figure. From Fig. 7, it is obvious that the errors of VOEW and VORPS are larger then those of VO, especially the errors of VOEW are significantly larger. This result indicates that the optimization functions effectively, while using the optimized weights (Problem 2) plays a more important role than using the selected feature pairs (Problem 3) for reducing the visual odometry error.

6 Conclusion and Future Work

Estimation of camera motion by tracking visual features is difficult because it depends on the shape of the terrain which is generally unknown. The estimation problem is furthermore difficult when a monocular system is used because scale of the translation component cannot be recovered. Our method succeeds in two ways. First, it simultaneously estimates a planar patch in front of the camera along with camera motion, and second recovers scale by taking advantage of the fixed distance from the camera to the ground. In some cases, approximating the terrain in front of the vehicle as a planar patch cannot be justified. Our method automatically detects these cases and uses a hybrid odometry system in which rotation is estimated from visual odometry and translation is recovered by wheel odometry.

Since this paper relies on a kinematical vehicle steering model, lateral wheel slip is not considered. For the future work, we are considering a revision to the vehicle motion model such that the algorithm can handle more complicated ground conditions where lateral wheel slip is noticeable.

References

1. K. Konolige, M. Agrawal, and J. Sol, “Large-scale visual odometry for rough terrain,” *Robotics Research*, vol. 66, p. 201212, 2011.

2. M. Maimone, Y. Cheng, and L. Matthies, “Two years of visual odometry on the mars exploration rovers,” *Journal of Field Robotics*, vol. 24, no. 2, pp. 169–186, 2007.
3. D. Nister, O. Naroditsky, and J. Bergen, “Visual odometry for ground vechicle applications,” *Journal of Field Robotics*, vol. 23, no. 1, pp. 3–20, 2006.
4. D. Scaramuzza, “1-point-ransac structure from motion for vehicle-mounted cameras by exploiting non-holonomic constraints,” *International Journal of Computer Vision*, vol. 95, p. 7485, 2011.
5. B. Kitt, J. Rehder, A. Chambers, and et al., “Monocular visual odometry using a planar road model to solve scale ambiguity,” in *Proc. European Conference on Mobile Robots*, September 2011.
6. N. Nourani-Vatani and P. Borges, “Correlation-based visual odometry for ground vehicles,” *Journal of Field Robotics*, vol. 28, no. 5, 2011.
7. R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. New York, Cambridge University Press, 2004.
8. M. Fischler and R. 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.
9. A. Howard, “Real-time stereo visual odometry for autonomous ground vehicles,” in *IEEE International Conference on Intelligent Robots and Systems*, Nice, France, Sept 2008.
10. D. Dansereau, I. Mahon, O. Pizarro, and et al., “Plenoptic flow: Closed-form visual odometry for light field cameras,” in *International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, CA, Sept. 2011.
11. P. Corke, D. Strelow, and S. Singh, “Omnidirectional visual odometry for a planetary rover,” in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, Sept. 2004, pp. 149–171.
12. L. Paz, P. Pinies, and J. Tardos, “Large-scale 6-DOF SLAM with stereo-in-hand,” *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 946–957, 2008.
13. B. Williams and I. Reid, “On combining visual slam and visual odometry,” in *IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, May 2010.
14. M. Wongphati, N. Niparnan, and A. Sudsang, “Bearing only fast SLAM using vertical line information from an omnidirectional camera,” in *Proc. of the IEEE International Conference on Robotics and Biomimetics*, Bangkok, Thailand, Feb. 2009, pp. 494–501.
15. A. Pretto, E. Menegatti, M. Bennewitz, and et al., “A visual odometry framework robust to motion blur,” in *IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009.
16. J. Civera, D. Bueno, A. Davison, and J. Montiel, “Camera self-calibraction for sequential bayesian structure form motion,” in *Proc. of the IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009, pp. 130–134.
17. D. Scaramuzza, “Absolute scale in structure from motion from a single vehicle mounted camera by exploiting nonholonomic constraints,” in *IEEE International Conference on Computer Vision*, Kyoto, Japan, Sept. 2009.
18. T. Gillespie, *Fundamentals of Vehicle Dynamics*. SAE International, 1992.
19. D. Bertsekas, *Nonlinear Programming*. Cambridge, MA, 1999.
20. J. Zhang and D. Song, “On the error analysis of vertical line pair-based monocular visual odometry in urban area,” in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, Oct. 2009, pp. 187–191.
21. R. Krauthgamer, J. Naory, and R. Schwartzz, “Partitioning graphs into balanced components,” in *The Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, New York, NY, Jan. 2009.
22. K. Andreev and H. Racke, “Balanced graph partitioning,” *Theory Comput. Systems*, vol. 39, 2006.
23. B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proceedings of Imaging Understanding Workshop*, 1981, pp. 121–130.