# Closed Form Solution for the Scale Ambiguity Problem in Monocular Visual Odometry

Isaac Esteban, Leo Dorst, and Judith Dijk

University of Amsterdam and TNO, The Netherlands

**Abstract.** This paper presents a fast monocular visual odometry algorithm. We propose a closed form solution for the computation of the unknown scale ratio between two consecutive image pairs. Our method requires only 1 2D-3D correspondence. A least square solution can also be found in closed form when more correspondences are available. Additionally we provide a first order analysis on the propagation of the error from the noise in the image features to the computation of the scale. We show by means of simulated and real data that our method is more robust and accurate than standard techniques. We demonstrate that our visual odometry algorithm is well suited for the task of 3D reconstruction in urban areas.

## 1 Introduction

Accurate estimation of the ego-motion of a vehicle is an essential step for autonomous operation. This estimation can be performed using a wide range of sensors such as GPS, wheel encoders, laser scanners or cameras. The use of visual information is in particular interesting due to the flexiblility of the cameras, their reduced cost and the possibility of integration with complementary systems such as road lane warnings, pedestrian detectors or obstacle avoidance techniques.

Early work on motion estimation is based on stereo vision systems [14]. In this setup two cameras are located parallel to each other while recording the scene. Given the knowledge of the baseline and relative orientation of the cameras, one can obtain accurate results when estimating the ego-motion of the camera [5] [17]. More recent work is focussed on monocular vision systems. The problem of estimating the motion of a camera and reconstructing the 3D structure is called "structure from motion" [3] [10]. Successful results have been obtained using both omnidirectional and perspective cameras [20] [19] [17] [18].

Related to the ego-motion estimation is the process of Simultaneous Localization and Mapping, or SLAM [6]. It aims at estimating both the motion of the moving vehicle and the surrounding map. SLAM methods usually involve the use of several sensors, though recent developments have been made that use only cameras [8]. In this work we develop a robust motion estimation algorithm for monocular cameras that can be integrated with complementary mapping techniques such as SLAM.

H. Liu et al. (Eds.): ICIRA 2010, Part I, LNAI 6424, pp. 665–679, 2010.

The nature of monocular systems induces a scale ambiguity in the estimation of the motion and the reconstructed 3D scene (see Figure 1). The global scale cannot be recovered unless information about the real world is introduced. This is achieved, for instance, by gathering information from a GPS or about the distance travelled by the camera. Additionally, if the motion is estimated on a frame-to-frame basis, considering only the image features, there is a scale ambiguity between the estimated translation vectors. We call this the local scale problem. There are a number of solutions in the literature to solve the scale implicitly. Scarammuzza et al. [20] use information about the camera's height to the ground and position with respect to the axis of the vehicle to obtain the global scale factor. Also, in [19] a direct/average observation of the speed of the vehicle is used to account for the distance travelled by the camera. These techniques, used extensively in the literature, suffer from a number of disadvantages. When introducing the speed of the camera, or the distance travelled, we are forced to use external devices such as a GPS or a speedometer. If the speed is averaged, then the motion of the camera is limited to a steady rate.

An alternative common technique, used in [17] [18], is to solve the camera pose using 2D-3D correspondences instead of image only 2D-2D correspondences. This is the so called PnP (Pose from n Points) problem, for which a number of solutions exist [4] [1] [15] [2]. When using PnP methods to estimate the camera pose, the scale is implicitly estimated. These methods however suffer from severe error propagation. Due to the fact that the full camera motion is estimated using only 2D-3D correspondences, the error is propagated as follows: first, from the image features in cameras $C_i$ and $C_{i_+1}$ to the estimation of camera pose at $C_{i+1}$. Second, the error is also propagated to the triangulation of the 3D points. Finally, the error is propagated to the estimation of the camera pose at $C_{i+2}$. This effect is propagated to subsequent camera poses and accumulates over time.

In this paper we focus on the local scale problem for which we derive an explicit closed form solution. We propose an alternative technique with the same computational complexity as the linear P6P solution, or related methods, but with considerable improvement in accuracy. Additionally, we provide a first order analysis on the propagation of the error and present a set of experiments with both simulated and real outdoors data.

This paper is organized as follows. In section 2 we present the notation and definitions. Section 3 presents our closed form solution for the computation of the local scale. In Section 4 we provide a first order analysis on the propagation of the error from the image noise and camera poses to the computation of the scale. In Section 5 we present our results on both simulated and real data. Finally, in Section 6 we draw some conclusions and point out future work.

## 2 Definitions and Notation

We employ the pinhole camera model, though the techniques we present can be accommodated to single view omnidirectional cameras. We begin with the definition of the estimated image features, 3D points and camera poses as a

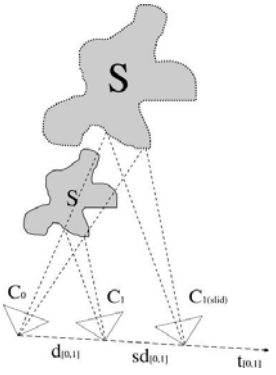


**Fig. 1.** Scale problem in monocular vision. Estimating the camera pose $C_1$ is intrinsically scale free. By sliding the camera over an unknown amount along the translation direction $t_{[0,1]}$, both structures $S$ become indistinguishable reconstructions, only the scale is different.

disturbance of the true values. These definitions are useful to understand the derivation of the closed form solution for the local scale and later for the analysis on the error propagation. Note that at this point we do not model the error and only reason about how the error is propagated.

### 2.1 Disturbed Image Features

We define the disturbed images features $\mathbf{x}_\alpha = \bar{\mathbf{x}}_\alpha + \Delta\mathbf{x}_\alpha$ in homogeneous coordinates (denoted by $H$) as:

$$\mathbf{x}_{\alpha H} + \Delta\mathbf{x}_{\alpha H} = \begin{bmatrix} \mathbf{x}_\alpha \\ 1 \end{bmatrix} + \begin{bmatrix} \Delta\mathbf{x}_\alpha \\ 0 \end{bmatrix} = \begin{bmatrix} u_\alpha + \Delta u_\alpha \\ v_\alpha + \Delta v_\alpha \\ 1 \end{bmatrix}, \tag{1}$$

where $\mathbf{x}_\alpha$ is the estimated location of the true image feature $\bar{\mathbf{x}}_\alpha$ and $\Delta\mathbf{x}_\alpha$ is the disturbance.

### 2.2 Disturbed Point in Space

Our method for the computation of the scale requires at least 1 2D-3D correspondence. We define a disturbed point in 3D space $\mathbf{X}_\alpha = \bar{\mathbf{X}}_\alpha + \Delta\mathbf{X}_\alpha$, or in homogeneous coordinates:

$$\mathbf{X}_{\alpha H} + \Delta\mathbf{X}_{\alpha H} = \begin{bmatrix} \mathbf{X}_\alpha \\ 1 \end{bmatrix} + \begin{bmatrix} \Delta\mathbf{X}_\alpha \\ 0 \end{bmatrix} = \begin{bmatrix} a_{x_\alpha} + \Delta a_{x_\alpha} \\ a_{y_\alpha} + \Delta a_{y_\alpha} \\ a_{z_\alpha} + \Delta a_{z_\alpha} \\ 1 \end{bmatrix} \tag{2}$$

Note that the disturbance represented here will depend on the error in the motion estimation, the feature location and the method for triangulation.

### 2.3 Disturbed Translation

We represent the translation direction with a unitary vector $\mathbf{t}$ with the constraint $||\mathbf{t}|| = 1$, or equivalently $\mathbf{t}^T\mathbf{t} = 1$. We impose this constraint in the disturbed translation $\mathbf{t} = \bar{\mathbf{t}} + \Delta\mathbf{t}$:

$$1 = (\bar{\mathbf{t}} + \Delta\mathbf{t})^T(\bar{\mathbf{t}} + \Delta\mathbf{t}) \tag{3}$$
$$1 = 1 + \bar{\mathbf{t}}^T\Delta\mathbf{t} + \Delta\mathbf{t}^T\bar{\mathbf{t}} + \Delta\mathbf{t}^T\Delta\mathbf{t} \tag{4}$$

Neglecting the second order terms ($\Delta\mathbf{t}^T\Delta\mathbf{t} \approx 0$) we obtain that $\Delta\mathbf{t}^T\bar{\mathbf{t}} = 0$. This indicates that the disturbance $\Delta\mathbf{t}$ is perpendicular to the unitary translation direction $\bar{\mathbf{t}}$. Expressed in coordinates:

$$\bar{\mathbf{t}}_\alpha = \begin{bmatrix} t_x + \Delta t_x \\ t_y + \Delta t_y \\ t_z + \Delta t_z \end{bmatrix}, \tag{5}$$

with ${\Delta t_x}^T t_x + {\Delta t_y}^T t_y + {\Delta t_z}^T t_z = 0$.

### 2.4 Disturbed Rotation

Given that most algorithms use a rotation representation based on a $3 \times 3$ matrix $R$, we represent the error as a relative disturbance over the true rotation [12]:

$$R = \bar{R}(I + \Delta R) \tag{6}$$

A rotation matrix $R$ is an orthogonal matrix ($R^{-1} = R^T$, or equivalently $I = R^T R$). The disturbed rotation must be also an orthogonal matrix, therefore we can obtain, to first order:

$$I = (\bar{R}(I + \Delta R))^T(\bar{R}(I + \Delta R)) \tag{7}$$
$$I = \bar{R}^T\bar{R} + \bar{R}^T\bar{R}\Delta R + \Delta R^T\bar{R}^T\bar{R} + \Delta R^T\bar{R}^T\bar{R}\Delta R \tag{8}$$
$$I \approx I + \Delta R + \Delta R^T \tag{9}$$
$$\Delta R^T = -\Delta R \tag{10}$$

Then $\Delta R$ is a skew symmetric matrix of the form:

$$\Delta R = \begin{bmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{bmatrix}, \tag{11}$$

where $\omega$ is an axis-angle [12] vector representation for a small rotation (the disturbance of the rotation). For shortness we define the rotation matrix with row vectors:

$$R = \begin{bmatrix} {\mathbf{r}_1}^T \\ {\mathbf{r}_2}^T \\ {\mathbf{r}_3}^T \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{12}$$

The absolute disturbance $\bar{R}\Delta R$[1] is also represented with with row vectors $\Delta\mathbf{r}_1{}^T$, $\Delta\mathbf{r}_2{}^T$ and $\Delta\mathbf{r}_3{}^T$.

So far we have described our notation with a first order analysis on the disturbances over image features, points in space, translation directions and rotations. Based on these definitions, we now describe our scale consistent motion estimation and then analyze the propagation of the error.

## 3 Robust and Scale Consistent Motion Estimation

We propose an algorithm for robust and scale consistent estimation of the motion of a camera through space. We present the algorithm in two parts. The first part deals with robust frame-to-frame motion estimation. This estimation is scale free. The second part deals with the estimation of the scale, which we solve in closed form.

### 3.1 Robust Frame-to-Frame Motion

The first part of the algorithm deals with robustly estimating the frame-to-frame motion. This is a well known problem and there are a large number of solutions. Among the most used is the 5-point algorithm [16] and normalized 8-point algorithm [10]. In our work we choose to use the latter given its simplicity, though our contribution on the computation of the scale can be combined with any frame-to-frame motion estimation algorithms. Additionally, we employ a local iterative refinement step in order to improve the estimate without a large computational burden.

We summarize the steps of the algorithm:

- Extract SIFT [13] features using VLfeat [22].
- Match features using nearest neighbors in the SIFT descriptor space.
- Reject outliers using RANSAC [9].
- Estimate the essential matrix defining the frame-to-frame motion using the normalized 8-point algorithm [10].
- Obtain a rotation $R$ and unitary translation $t$ using the method by Horn [11].
- Refine the estimate by minimizing the reprojection error using Levenberg-Marquardt. In this setup the internal calibration parameters (focal length, pixel ratio and focal point) are also optimized.

In general no more than 4 iterations of the refinement step are needed for convergence. The optimization of the rotation is performed in the normalized quaternion space. Note that this optimization step can cope with some of the shortcomings of the 8-point algorithm such as degeneracies. Given a good set of inliers and a slightly larger number of iterations, the procedure can converge to a good solution even when all points lie in a plane. This motion estimation procedure yields a rotation matrix $R$ and a unitary translation direction $t$.

[1] Given that the true rotation is unknown, the absolute disturbance can be approximated by multiplying the estimated rotation $R$ with the relative disturbance $\Delta R$.

### 3.2 Computation of the Local Scale

Using the above algorithm we can robustly estimate the motion of the camera through space. However, the translation ratios between different image pairs is still unknown. In this section we derive a closed form solution for the computation of the local scale.

We know [10], given the camera model, that the relation between the image features and the reconstructed 3D points in homogeneous coordinates (denoted by $H$) is defined by:

$$m\mathbf{x}_{\alpha H} = PX_{\alpha H}, \tag{13}$$

where $P$ is the scale free camera motion ($P = [R|\mathbf{t}]$ with $||\mathbf{t}|| = 1$) and $m$ is the scaling necessary to obtain image points and not image plane points. Given that the camera motion, obtained as described in section 3.1, is only calculated up to scale, we introduce the scaling factor $s$:

$$m\mathbf{x}_{\alpha H} = [R|s\mathbf{t}]X_{\alpha H}, \tag{14}$$

or equivalently:

$$\begin{bmatrix} mu_\alpha \\ mv_\alpha \\ m \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & st_x \\ R_{21} & R_{22} & R_{23} & st_y \\ R_{31} & R_{32} & R_{33} & st_z \end{bmatrix} \begin{bmatrix} a_{x_\alpha} \\ a_{y_\alpha} \\ a_{z_\alpha} \\ 1 \end{bmatrix} \tag{15}$$

For every 2D-3D feature match $\alpha$, we obtain three equations:

$$mu_\alpha = \mathbf{r}_1{}^T\mathbf{X}_\alpha + st_x \tag{16}$$

$$mv_\alpha = \mathbf{r}_2{}^T\mathbf{X}_\alpha + st_y \tag{17}$$

$$m = \mathbf{r}_3{}^T\mathbf{X}_\alpha + st_z \tag{18}$$

**Note:** *At this point we can solve this system for s in different ways. However, let us continue our derivation and we refer the reader to Section 3.3 for a detailed motivation.*

In order to solve for $s$, we substitute equation 18 in equation 16 to remove the image scale factor $m$. Manipulating the terms we single out the scale parameter $s$:

$$(\mathbf{r}_3{}^T\mathbf{X}_\alpha + st_z)u_\alpha = \mathbf{r}_1{}^T\mathbf{X}_\alpha + st_x \tag{19}$$

$$(t_z u_\alpha - t_x)s = (\mathbf{r}_1{}^T - \mathbf{r}_3{}^T u_\alpha)\mathbf{X}_\alpha \tag{20}$$

This is precisely of the form $\mathbf{A}s = \mathbf{b}$, where:

$$\mathbf{A} = [t_z u_\alpha - t_x] \tag{21}$$

$$\mathbf{b} = [(\mathbf{r}_1{}^T - \mathbf{r}_3{}^T u_\alpha)\mathbf{X}_\alpha], \tag{22}$$

We can construct the vectors $\mathbf{A}$ and $\mathbf{b}$ using one row for every 2D-3D correspondence $\alpha$, and solve for $s$ using SVD (Singular Value Decomposition), obtaining a solution in the least square sense. Given that $\mathbf{A}$ and $\mathbf{b}$ are vectors, we can also obtain a closed form solution. We want to obtain the scale $s$ in the least

square sense, which means minimizing $||\mathbf{A}s-\mathbf{b}||^2$. We do so obtaining the partial derivative w.r.t. $s$ and setting it to zero:

$$||\mathbf{A}s - \mathbf{b}||^2 = \mathbf{A}^T\mathbf{A}s^2 - 2\mathbf{A}^T\mathbf{b}s + \mathbf{b}^T\mathbf{b} \tag{23}$$

$$\frac{\partial(||\mathbf{A}s - \mathbf{b}||^2)}{\partial s} = 2\mathbf{A}^T\mathbf{A}s - 2\mathbf{A}^T\mathbf{b} = 0 \tag{24}$$

$$s = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} \tag{25}$$

Hence we obtain a linear solution in the least square sense for the scale parameter $s$. In order to simplify, we express the solution of equation 25 as:

$$s = \mathbf{A}^+\mathbf{b} \tag{26}$$

where:

$$\mathbf{A}^+ = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T \tag{27}$$

### 3.3 A Note on Alternative Methods

Given the set of 3 equations in 16, 17 and 18, there are 4 different ways to solve the system for $s$.

- **Method 1:** solve using equations 16 and 18 as we do. This implies using $u_\alpha$ and $t_x$.
- **Method 2:** alternatively, the system can be solved using equations 17 and 18. This implies using $v_\alpha$ and $t_y$.
- **Method 3:** combining both, the system can be solved using methods 1 and 2, stacking the solutions into vectors $\mathbf{A}$ and $\mathbf{b}$ (resulting in vectors that are twice as long).
- **Method 4:** The last possibility is using equations 16 and 17, combining $u_\alpha$, $v_\alpha$, $t_x$ and $t_y$ in a single solution. This solution is similar to methods 1 and 2 but using the ratio $\frac{u_\alpha}{v_\alpha}$ (or the inverse).

It is at first sight difficult to chose one particular method. In an urban environment where the camera moves along a street, most of the motion occurs in the $x$ direction, while the motion on the $y$ axis is mostly negligible. In this case, we can motivate our decision by suggesting that the motion in the direction $y$ will not contribute to the solution $s$. Rather it will disturb it since the motion will be the result of noise. Naturally, if the camera moves mainly in the $y$ direction, the alternative method 2 should be employed. However our derivation and subsequent analysis holds by simply substituting $t_x$ by $t_y$ and $\mathbf{r}_1{}^T$ by $\mathbf{r}_2{}^T$ (along with the corresponding errors). Furthermore, our analysis also holds (or can be easily adjusted) for methods 3 and 4. For method 3, the resulting error will be result of stacking errors computed for method 1 and 2. For method 4, the error on the ratio $\frac{u_\alpha}{v_\alpha}$ needs to be used instead of $\Delta u_\alpha$. In order to strengthen our motivation, we simulate a camera moving in an urban environment. In our setup the motion of the camera occurs mainly in the $x$ direction. We implement all 4 methods and compare the accuracy in the estimation of the scale. Figure 3 shows the results.

## 4 Error Propagation in Scale Computation

Now that we have a closed form solution to compute the local scale, we wish to analyze how the error is propagated. We begin with the first order error propagation in the computation of $s$. To obtain a full picture of the propagation, the subsequent error analysis is performed on the computation of $\mathbf{A}^+$, $\mathbf{A}$ and $\mathbf{b}$ independently. To first order, the error in equation 26 is:

$$\Delta s = \mathbf{A}^+ \Delta \mathbf{b} + \Delta \mathbf{A}^+ \mathbf{b} \tag{28}$$

This is however expressed in terms of $\Delta \mathbf{A}^+$ and $\Delta \mathbf{b}$. We need to obtain those expressions as a function of the original disturbances in image features, points in space and camera motion.

### 4.1 Error Propagation in the Computation of $\mathbf{A}^+$

We obtain an expression for the computation of $\mathbf{A}^+$ in equation 27. We now analyze how the error is propagated in its computation. We begin by introducing the disturbances:

$$\mathbf{A}^+ + \Delta \mathbf{A}^+ = ((\mathbf{A} + \Delta \mathbf{A})^T (\mathbf{A} + \Delta \mathbf{A}))^{-1} (\mathbf{A} + \Delta \mathbf{A})^T \tag{29}$$

$$= (\mathbf{A}^T \mathbf{A} + \mathbf{A}^T \Delta \mathbf{A} + \Delta \mathbf{A}^T \mathbf{A} + \Delta \mathbf{A}^T \Delta \mathbf{A})^{-1} (\mathbf{A} + \Delta \mathbf{A})^T \tag{30}$$

By first order Taylor, we may derive a first order approximation for the disturbance in the computation of $A^+$:

$$\Delta \mathbf{A}^+ = -(\mathbf{A}^T \mathbf{A})^{-1} \left( (\mathbf{A}^T \Delta \mathbf{A} + \Delta \mathbf{A}^T \mathbf{A})(\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T + \Delta \mathbf{A}^T \right) \tag{31}$$

**Proof**
Neglecting some second order terms in equation 30, we obtain:

$$\mathbf{A}^+ + \Delta \mathbf{A}^+ \approx (\mathbf{A}^T \mathbf{A} + \mathbf{A}^T \Delta \mathbf{A} + \Delta \mathbf{A}^T \mathbf{A})^{-1} (\mathbf{A} + \Delta \mathbf{A})^T \tag{32}$$

We define:

$$Z = (\mathbf{A}^T \mathbf{A} + \mathbf{A}^T \Delta \mathbf{A} + \Delta \mathbf{A}^T \mathbf{A})^{-1} \tag{33}$$

Now, we need to transform $Z$ in the form $Z = (\mathbf{A}^T \mathbf{A})^{-1} + C$ so we can obtain a description of the disturbance $\Delta \mathbf{A}^+$. To accomplish this we employ a first order approximation using Taylor series. We first define:

$$a = \mathbf{A}^T \mathbf{A} \tag{34}$$

$$n = \mathbf{A}^T \Delta \mathbf{A} + \Delta \mathbf{A}^T \mathbf{A}. \tag{35}$$

We use these in equation 33 and transform it to obtain:

$$Z = (a + n)^{-1} = \left( a(1 + a^{-1} n) \right)^{-1} = (1 + a^{-1} n)^{-1} a^{-1}. \tag{36}$$

We need to transform $Z$ according to:

$$(a + n)^{-1} = a^{-1} + C \tag{37}$$

The first part $((1 + a^{-1}n)^{-1})$ is approximated using a first order taylor series:

$$Z \approx (1 - a^{-1}n)a^{-1} = a^{-1} - a^{-1}na^{-1} = a^{-1} + (-a^{-1}na^{-1}), \tag{38}$$

were we obtain $C = -a^{-1}na^{-1}$. This provides a first order approximation of the error propagation in the computation of $\mathbf{A}^+$:

$$\begin{aligned}
\mathbf{A}^+ + \Delta\mathbf{A}^+ &= ((\mathbf{A}^T\mathbf{A})^{-1} + C)(\mathbf{A} + \Delta\mathbf{A})^T \\
&= ((\mathbf{A}^T\mathbf{A})^{-1} - (\mathbf{A}^T\mathbf{A})^{-1}n(\mathbf{A}^T\mathbf{A})^{-1})(\mathbf{A} + \Delta\mathbf{A})^T && (39)\\
&\approx (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T - (\mathbf{A}^T\mathbf{A})^{-1}\left(n(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T + \Delta\mathbf{A}^T\right) && (40)\\
&= \mathbf{A}^+ - (\mathbf{A}^T\mathbf{A})^{-1}\left((\mathbf{A}^T\Delta\mathbf{A} + \Delta\mathbf{A}^T\mathbf{A})(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T + \Delta\mathbf{A}^T\right) && (41)
\end{aligned}$$

Therefore, the disturbance to first order that is propagated through the computation of $\mathbf{A}^+$ is:

$$\Delta\mathbf{A}^+ = -(\mathbf{A}^T\mathbf{A})^{-1}\left((\mathbf{A}^T\Delta\mathbf{A} + \Delta\mathbf{A}^T\mathbf{A})(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T + \Delta\mathbf{A}^T\right) \tag{42}$$

### 4.2 Error Propagation in Computation of A

In the previous section we derive an expression for $\Delta\mathbf{A}^+$ in terms of $\Delta\mathbf{A}$, now we obtain a first order expression for $\Delta\mathbf{A}$:

$$\begin{aligned}
\mathbf{A} + \Delta\mathbf{A} &= (t_z + \Delta t_z)(u_\alpha + \Delta u_\alpha) - (t_x + \Delta t_x) && (43)\\
\Delta\mathbf{A} &= t_z\Delta u_\alpha + \Delta t_z u_\alpha - \Delta t_x && (44)
\end{aligned}$$

### 4.3 Error Propagation in Computation of b

Finally, we need to obtain an expression for the error propagation in the computation of $\mathbf{b}$. We follow the same procedure as before, introducing the disturbances in equation 22 and propagating to first order:

$$\begin{aligned}
\mathbf{b} + \Delta\mathbf{b} &= \left[\left(\mathbf{r}_1{}^T + \Delta\mathbf{r}_1{}^T - (\mathbf{r}_3{}^T + \Delta\mathbf{r}_3{}^T)(u_\alpha + \Delta u_\alpha)\right)(\mathbf{X}_\alpha + \Delta\mathbf{X}_\alpha)\right] && (45)\\
&\approx \left[\left(\mathbf{r}_1{}^T + \Delta\mathbf{r}_1{}^T - \mathbf{r}_3{}^T u_\alpha - \mathbf{r}_3{}^T\Delta u_\alpha - \Delta\mathbf{r}_3{}^T u_\alpha\right)(\mathbf{X}_\alpha + \Delta\mathbf{X}_\alpha)\right] && (46)\\
&\approx \left[(\mathbf{r}_1{}^T - \mathbf{r}_3{}^T u_\alpha)\mathbf{X}_\alpha + (\mathbf{r}_1{}^T - \mathbf{r}_3{}^T u_\alpha)\Delta\mathbf{X}_\alpha + \left(\Delta\mathbf{r}_1{}^T - \mathbf{r}_3{}^T\Delta u_\alpha - \Delta\mathbf{r}_3{}^T u_\alpha\right)\mathbf{X}_\alpha\right] && (47)\\
\Delta\mathbf{b} &= \left[(\mathbf{r}_1{}^T - \mathbf{r}_3{}^T u_\alpha)\Delta\mathbf{X}_\alpha + \left(\Delta\mathbf{r}_1{}^T - \mathbf{r}_3{}^T\Delta u_\alpha - \Delta\mathbf{r}_3{}^T u_\alpha\right)\mathbf{X}_\alpha\right] && (48)
\end{aligned}$$

### 4.4 Conclusions

In Section 4, in the same spirit as [12], we present a first order analysis on the error propagation when computing the local scale (see Section 3.2). We obtain a closed form expression for the propagated error (equation 28). In order to make use of the solution, we performed the analysis on the error propagation on the computation of $\mathbf{A}^+$ (equation 42), $\mathbf{A}$ (equation 44) and $\mathbf{b}$ (equation 48). Getting all four equations together yields a first order closed form representation of the propagation of the error in the computation of the scale using any number of

2D-3D correspondences. This result can be used, for instance, to estimate the confidence on the computation of the scale. This, in turn, can be introduced in a SLAM (or any probabilistic mapping) approach where the covariance of the estimate can now be computed without the need of computationally expensive Monte Carlos simulations. Given that we provide an expression for error in the scale, the error in the 3D triangulation and the frame-to-frame motion needs to be provided. This however can be estimated using either Monte Carlo simulations or, given that the motion is scale independent, can be learn based on noise in the image features. Alternatively, the error propagation on the 8-point algorihtm [21] can be employed together with our derivation.

## 5 Experiments and Results

### 5.1 Comparison with P6P

In this section we present our experimental results with a fully simulated scene of which 3 synthetic images are recorded. 30 3D points are projected onto the 3 cameras located at different positions. The internal parameters of the cameras are considered perfect. Gaussian noise is introduced only in the image feature location. We perform 2000 runs in which a different 3D scene is created and the cameras are randomly positioned. For each run, we estimate the motion of camera 3 using P6P [2] and our novel approach. We increment the noise in the image feature locations from 0.005 to 0.18 pixels of standard deviation. We then compare the estimated camera poses (rotation and translation) with the ground truth camera poses. The results are shown in figure 2. The plots show that our method outperforms the P6P reference algorithm by a large factor. For these experiments no global optimization is used. The improvement is particularly significant in the estimation of the translation, where our approach behaves more linearly. In this case, the P6P algorithm becomes almost unusable as the noise in the feature location increases. In order to reach a comparable accuracy, 4 times more 2D-3D correspondences are neccessary. This cannot be solved by just increasing the frame rate as the number of images required to obtain the same accuracy becomes too large and so does the computational costs.

### 5.2 Outdoors Visual Odometry

We also perform two experiments on real outdoors data. In both experiments images are recorded with a hand held digital reflex camera (DSLR). We use a wide angle lens at a resolution of 1728x1152 pixels. For the first set, 60 images (see figure 4) are recorded covering a total distance of 100 meters of an urban area, following a straight trajectory along one street. Results are shown in figure 5. For this experiment no drift can be perceived. This is because of our accurate

[2] The P6P reference algorithm is a linear algorithm based on the Direct Linear Transform method commonly used for comparison. We choose this algorithm given that, as our proposed method, it is linear and solves in a least square sense.

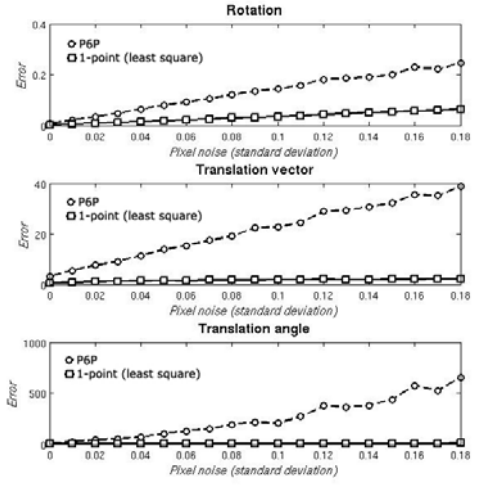


**Fig. 2.** LEFT: Error on the estimation of rotation and translation of a third camera pose given 2 previous camera poses and the reconstructed structure using linear triangulation (2000 runs). Top-left: error in the rotation calculated in the quaternion space. Middle-left: error in the translation vector calculated as the distance of the difference vector between ground truth and estimation. Bottom-left: error in the angle estimation of the translation direction.

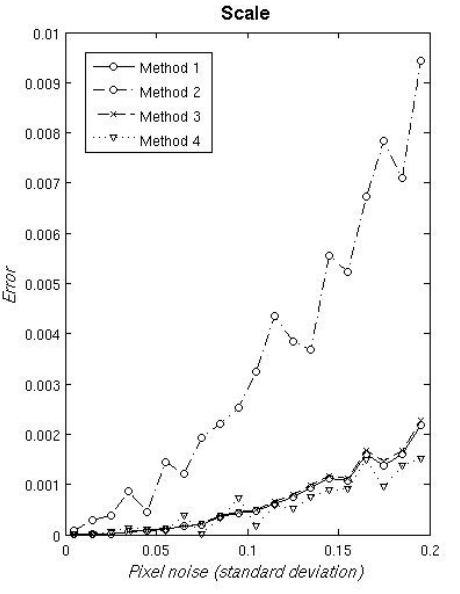


**Fig. 3.** RIGHT: Error on the estimation of the scale using the 4 proposed methods (2000 runs). Method 2 is the worst given that the motion on the $y$ direction does not contribute to the computation of $s$. Methods 1 and 3 offer similar performance, with a slight preference for method 1. Method 4 is slightly better than the rest. Our decision to use method 1 instead of 4 is motivated by the fact that method 4 can suffer from division by zero and its behavior is more erratic with a lower number of runs.

method and the fact that no Bundle Adjustment is used, which is a well known source for drift. The accurate results on the estimation of the camera motion are used to obtain an accurate reconstruction using [7] For the second set, a total 174 images (see figure 6) are recorded covering a distance of 200+ meters. In this case the camera moves around a full block of houses. The same image is used for first and last position to ensure that the true last camera pose is exactly the same as where the first image is recorded. The trajectory is also recovered accurately without the aid of Bundle Adjustment. Given that the loop is closed, we can also measure the accuracy. The distance between the estimated position of the last image and the first image is below 1 meter. This represents an error of less that 0.5%.

**Fig. 4.** Sample images of first dataset

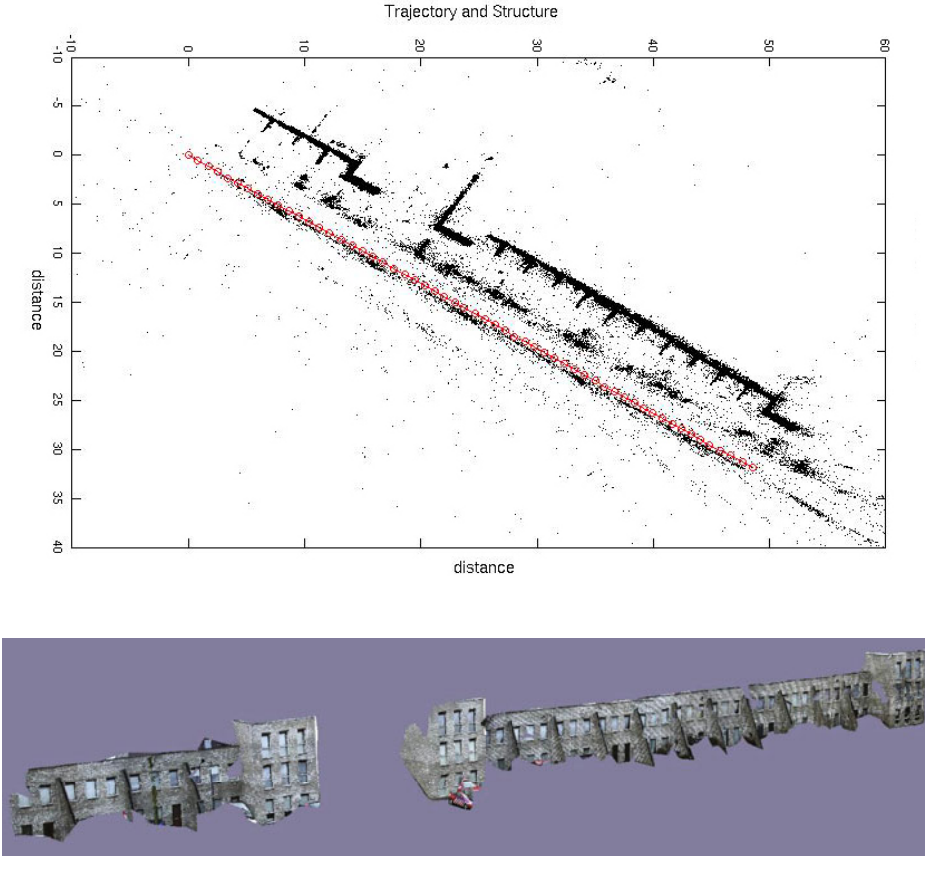


**Fig. 5.** Top: Estimated trajectory (red circles) and recovered point cloud for first set. Bottom: reconstructed 3D model. No global Bundle Adjustment was necessary.

**Fig. 6.** Sample images of second dataset

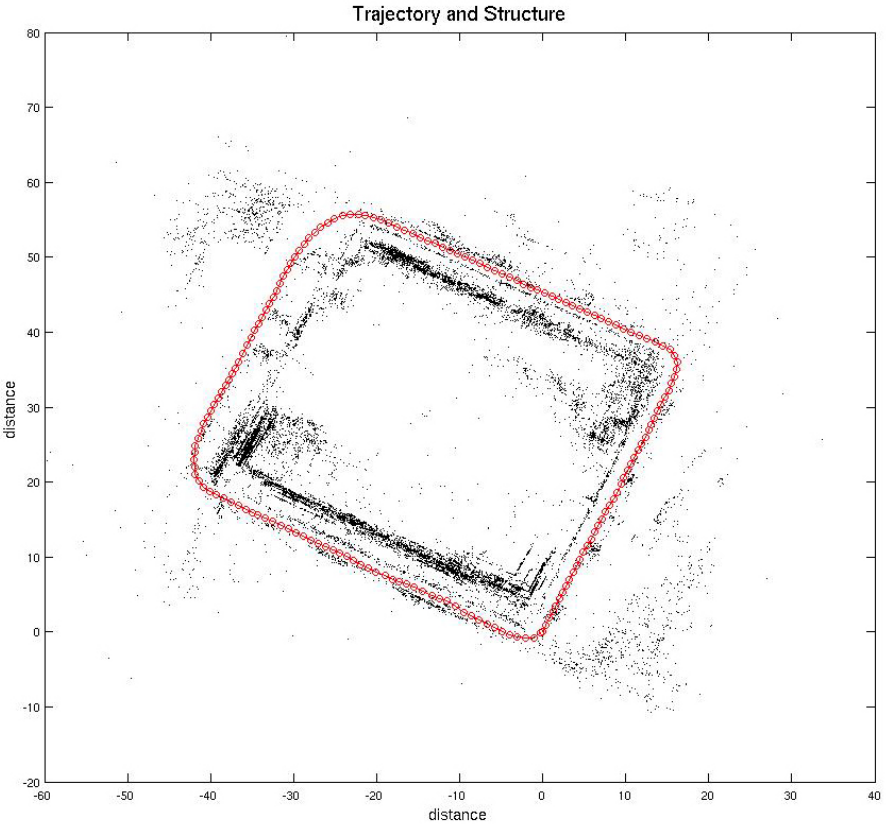


**Fig. 7.** Estimated trajectory (red circles) and recovered point cloud for second set. The camera started at the right-bottom corner moving towards the right side and ending in the same position. This was ensured using the same image for first and last position. No global Bundle Adjustment was necessary.

## 6 Conclusions

In this paper we present a closed form solution for the computation of the scale ratio between the translation directions of two consecutive image pairs. Our method requires only 1 2D-3D correspondence, though a least square solution is presented also in closed form. Combined with the 8-point algorithm and a local iterative refinement step we achieve accurate results with little drift over large trajectories. Our visual odometry algorithm is sufficiently accurate to be used for urban 3D reconstruction. We obtain 3D a reconstruction of a street without the need for global Bundle Adjustment. We also compare our method with a reference linear technique for the computation of a third camera pose using 6 2D-3D correspondences. Our method outperforms by a factor of 3 in

the computation of the rotation and a factor of 10 in the computation of the scale consistent translation. Additionally we provide a first order analysis of the propagation of the error, from the noise in the image features to the computation of the scale. Compared to alternative methods our approach takes full advantage of the larger number of image feature matches across consecutive frames. The 3-frame error propagation to the computation of the rotation and the translation direction is then avoided, effectively reducing the accumulation of error over time. The proposed algorithm is successfully applied to images recorded in a real urban environment for which we obtain accurate 3D reconstructions. In the future we plan to extend our work to take advantage of the error propagation in order to obtain a better estimate than a least square solution. This work and more will be available online at www.fit3d.org.

## References

1. Ameller, M.-A., Triggs, B., Quan, L.: Camera pose revisited - new linear algorithms. Rapport Interne - Equipe MOVI (2000)
2. Bujnak, M., Kukelova, Z., Pajdla, T.: A general solution to the p4p problem for camera with unknown focal length. In: CVPR (2008)
3. Dellaert, F., Seitz, S., Thorpe, C., Thrun, S.: Structure from motion without correspondence. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2000 ) (June 2000)
4. DeMenthon, D., Davis, L.S.: Exact and approximate solutions of the perspective-three-point problem. IEEE Trans. Pattern Anal. Mach. Intell. 14(11), 1100–1105 (1992)
5. Dubbelman, G., van der Mark, W., Groen, F.C.A.: Accurate and robust ego-motion estimation using expectation maximization. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2008)
6. Durrant-Whyte, H., Bailey, T.: Simultaneous localisation and mapping (slam): Part i the essential algorithms. Robotics and Automation Magazine (2006)
7. Esteban, I., Dijk, J., Groen, F.: Fit3d toolbox: multiple view geometry and 3d reconstruction for matlab. In: International Symposium on Security and Defence Europe, SPIE (2010)
8. Eustice, R.M., Pizarro, O., Singh, H.: Visually augmented navigation for autonomous underwater vehicles. IEEE J. Oceanic Eng. (2007) (accepted, to appear)
9. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. pp. 726–740 (1987)
10. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2000), ISBN: 0521623049
11. Horn, B.: Recovering baseline and orientation from essential matrix (1990)
12. Kanatani, K.: Statistical Optimization for Geometric Computation. Dover, New York (1996)
13. Lowe, D.: Distinctive image features from scale-invariant keypoints. Int. J. of Computer Vision (2004)
14. Moravec, H.: Obstacle avoidance and navigation in the real world by a seeing robot rover (1980)
15. Moreno Noguer, F., Lepetit, V., Fua, P.: Accurate non-iterative o(n) solution to the pnp problem. In: ICCV 2007, pp. 1–8 (2007)

16. Nister, D.: An efficient solution to the five-point relative pose problem. In: CVPR (2003)
17. Nister, D., Naroditsky, O., Bergen, J.: Visual odometry for ground vehicle applications. Journal of Field Robotics 23(1) (2006)
18. Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R.: Visual modeling with a hand-held camera. Int. J. Comput. Vision 59(3), 207–232 (2004)
19. Scaramuzza, D., Fraundorfer, F., Siegwart, R.: Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In: IEEE International Conference on Robotics and Automation (2009)
20. Scaramuzza, D., Siegwart, R.: Appearance guided monocular omnidirectional visual odometry for outdoor ground vehicles. IEEE Transactions on Robotics, Special Issue on Visual SLAM. Guest editors: José Neira, Andrew Davison, John J. Leonard (October 2008) (publication date)
21. Sur, F., Noury, N., Berger, M.-O.: Computing the uncertainty of the 8 point algorithm for fundamental matrix estimation. In: Everingham, M., Needham, C., Fraile, R. (eds.) 19th British Machine Vision Conference - BMVC 2008 Proceedings of the British Machine Vision Conference 2008, Leeds, Leeds United Kingdom (September 2008), http://www.comp.leeds.ac.uk/bmvc2008/proceedings/papers/269.pdf
22. Vedaldi, A.: An open implementation of the SIFT detector and descriptor. Technical Report 070012, UCLA CSD (2007)