Large scale semantic 3D modeling of the urban landscape

Esteban Lopez, I.

Citation for published version (APA):
Optimal Local Scale in Camera Motion

We discussed in Chapter 4 the problem of estimating the motion of a camera and reconstructing the 3D structure. In this setup a single camera records the environment. This technique is common given the flexibility and wide availability of consumer cameras. Both Image-to-Image and Image-to-World approaches can be used to obtain an estimation of the camera motion. Furthermore, optimization of the estimation is a common practice, though local optimization (where only a handful of parameters are refined) is often desired due to the smaller computational burden.

We also discussed in Chapters 2 and 4 the nature of monocular systems and the scale ambiguity in the estimation of the motion and the reconstructed 3D scene (see Figure 5.1). The global scale, which describes the real size of things, cannot be recovered unless information about the world is introduced. This is achieved, for instance, by gathering information from a GPS about the distance travelled by the camera.

Additionally, if the motion is estimated on a frame-to-frame basis, considering only the image features and therefore using Image-to-Image approaches, there is a scale ambiguity between the estimated translation vectors (see Figure 5.1). This means that the estimation of the position of two cameras do not share a common scale. We call this the local scale problem.

The local scale accounts for the ratio between the scale of different camera translation vectors. When the local scale is estimated for every camera, all translation vectors can be placed in the same scale frame where distances can be measured in generic units.
Figure 5.1: The 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. If three cameras are considered, the local scale accounts for the ratio between the scales of the two pairs of cameras (there are three pairs of cameras, though only 2 of the scale ratios are linearly independent). The global scale accounts for the scale of real life objects.

When the global scale is introduced, those units can be translated into real life units such as meters, obtaining the real size of the reconstructed scene.

In this chapter we focus on the local scale problem for which we derive an explicit optimal closed form solution. We first develop the theoretical 1-point method hinted by [48]. Additionally we provide a Least Squares estimate of the scale when more correspondences are available. Then we derive the first order error propagation on the computation of the scale and finally we present an optimal closed form solution. Our proposed method has the same computational complexity as the state-of-the-art methods (see Section 5.2), but with considerable improvement in accuracy.

This chapter is organized as follows. In section 5.2 we present the notation and definitions. Section 5.3 presents a Least Squares closed form solution for the computation of the local scale. In Section 5.4 we provide a first order analysis on the propagation of
5.1 Related Work

There are a number of solutions in the literature to resolve the local or global scale implicitly. Scaramuzza et al. [49] use information about the height of the camera to the ground and position with respect to the axis of the vehicle to obtain the global scale. Also, in [50] a direct/average observation of the speed of the vehicle (reading only the speedometer without any additional aid such as GPS, INS, etc) 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 technique, used in [51] [52] [20], is to solve for the camera pose using 2D-3D correspondences instead of image-only 2D-2D correspondences, therefore employing Image-to-World approaches. This is the so called PnP (Pose from n Points) problem, for which a number of solutions exist [37] [53] [54] [40]. A common linear solution, called P6P, was discussed in Chapter 4. P6P is based on the Direct Linear Transform algorithm and works very similarly to the 8-point algorithm. There are however certain disadvantages when using this range of solutions.

When using PnP methods to estimate the camera pose, the scale is implicitly estimated. These methods suffer from 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 $C_{i+2}$. These effects propagate to subsequent camera poses and accumulate over time.

One of the most advanced algorithms for solving the Pose from n Points problem is the EPnP algorithm by Moreno-Noguer et. al. [38]. They use more than 4 2D-3D correspondences to obtain an optimal closed form solution. Experiments and results are presented in Section 5.6. Finally we draw some conclusions and point out future work in Section 5.7.
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

correspondences and solve the camera pose non-iteratively. The authors claim that their technique is at least as good as any state-of-the-art method and is faster and more robust. Tardif et. al. [48] propose a 1.5-point algorithm for solving the scale consistent translation direction while fixing the rotation. The authors claim that this is more robust than using a theoretical 1-point algorithm, though no comparison is provided.

5.2 Definitions and Notation

In the development of our work we assume the pinhole camera model, though the techniques we present can be also be adapted to single view omnidirectional cameras. We begin with the definition of the estimated image features, 3D points and camera poses as a disturbance of the true values. These definitions are useful to understand the derivation of the optimal solution for the local scale and the analysis on the error propagation.

5.2.1 Disturbed Image Features

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

$$x_{\alpha H} + \Delta x_{\alpha H} = \begin{bmatrix} x_{\alpha} \\ 1 \end{bmatrix} + \begin{bmatrix} \Delta x_{\alpha} \\ 0 \end{bmatrix} = \begin{bmatrix} u_{\alpha} + \Delta u_{\alpha} \\ v_{\alpha} + \Delta v_{\alpha} \\ 1 \end{bmatrix}, \quad (5.1)$$

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

5.2.2 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 $X_\alpha = \bar{X}_\alpha + \Delta X_\alpha$, or in homogeneous coordinates:

$$X_{\alpha H} + \Delta X_{\alpha H} = \begin{bmatrix} X_{\alpha} \\ 1 \end{bmatrix} + \begin{bmatrix} \Delta X_{\alpha} \\ 0 \end{bmatrix} = \begin{bmatrix} U_{\alpha} + \Delta U_{\alpha} \\ V_{\alpha} + \Delta V_{\alpha} \\ W_{\alpha} + \Delta W_{\alpha} \\ 1 \end{bmatrix} \quad (5.2)$$
Note that the disturbance represented here will depend on the error in the motion estimation, the feature location and the method for triangulation.

5.2.3 Disturbed Translation

We represent the translation direction by a unit vector \( \mathbf{t} \) with the constraint \( ||\mathbf{t}|| = 1 \), or equivalently \( \mathbf{t}^T \mathbf{t} = 1 \). We impose this constraint on the disturbed translation \( \mathbf{t} = \bar{\mathbf{t}} + \Delta \mathbf{t} \):

\[
1 = (\bar{\mathbf{t}} + \Delta \mathbf{t})^T(\bar{\mathbf{t}} + \Delta \mathbf{t})
1 = 1 + \bar{\mathbf{t}}^T \Delta \mathbf{t} + \Delta \mathbf{t}^T \bar{\mathbf{t}} + \Delta \mathbf{t}^T \Delta \mathbf{t}
\] (5.3)

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 unit translation direction \( \bar{\mathbf{t}} \).

Expressed in coordinates:

\[
\bar{\mathbf{t}} = \begin{bmatrix}
t_U + \Delta t_U \\
t_V + \Delta t_V \\
t_W + \Delta t_W
\end{bmatrix},
\] (5.4)

with \( \Delta t_U t_U + \Delta t_V t_V + \Delta t_W t_W = 0 \).

5.2.4 Disturbed Rotation

Most algorithms use a rotation representation based on a \( 3 \times 3 \) matrix \( \mathbf{R} \), therefore we represent the error as a relative disturbance over the true rotation \( \bar{\mathbf{R}} \):

\[
\mathbf{R} = \bar{\mathbf{R}}(I + \Delta \mathbf{R})
\] (5.5)

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

\[
I = (\bar{\mathbf{R}}(I + \Delta \mathbf{R}))^T(\bar{\mathbf{R}}(I + \Delta \mathbf{R}))
I = \bar{\mathbf{R}}^T \bar{\mathbf{R}} + \bar{\mathbf{R}}^T \Delta \mathbf{R} \bar{\mathbf{R}} + \Delta \mathbf{R}^T \bar{\mathbf{R}}^T \bar{\mathbf{R}} \Delta \mathbf{R}
I \approx I + \Delta \mathbf{R} + \Delta \mathbf{R}^T
\]

\[
\Delta \mathbf{R}^T = -\Delta \mathbf{R}
\] (5.6)
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

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

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

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

$$
R = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}
$$

The absolute disturbance $R\Delta R$ is also represented by its row vectors $\Delta r_1^T$, $\Delta r_2^T$ and $\Delta r_3^T$. Given that the true rotation is unknown, the absolute disturbance can be approximated to first order by multiplying the estimated rotation $R$ with the relative disturbance $\Delta R$.

5.3 Robust and Scale Consistent Motion Estimation

We propose an algorithm for robust 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 local scale, which we solve in closed form.

5.3.1 Robust Frame-to-Frame Motion

The first part of the algorithm deals with robustly estimating the frame-to-frame motion between two cameras. This is a well known problem and there are a large number of solutions. Among the most used is the 5-point algorithm \[34\], discussed in Section 4.3.1.3, and normalized 8-point algorithm \[I\], discussed in Section 4.3.1.1. 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 algorithm. Additionally, we employ a local iterative refinement step in order to improve the estimate without a large computational burden. Image feature points are calculated
5.3 Robust and Scale Consistent Motion Estimation

and used as input to obtain a projection matrix $P$ consisting of a rotation $R$ and scale free translation vector $t$.

We summarize the steps of the algorithm:

- Match features using nearest neighbors in the SIFT descriptor space (see Chapter 2).
- Reject outliers using RANSAC [35] (see Chapter 3).
- Estimate the essential matrix defining the frame-to-frame motion using the normalized 8-point algorithm [1] (see Chapter 4).
- Obtain a rotation $R$ and unitary translation $t$ using the method by Horn [2] (see Chapter 2).
- 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. In Chapter 4 we discussed how 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 $t$.

5.3.2 Computation of the Local Scale - Least Squares

Using the above algorithm we can robustly estimate the motion of the camera through space. However, the translation ratios between different image pairs are still unknown. In this section we derive a closed form solution for the computation of the local scale. As we discussed in Chapter 2, the projective relation between the image features and the 3D points in homogeneous coordinates (denoted by $H$) is defined by:
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

\[ mX_{\alpha H} = PX_{\alpha H}, \]  \hspace{1cm} (5.9)

where \( P \) is the scale free projection matrix (\( P = [R|t] \) with \( ||t|| = 1 \)) and \( m \) is the scaling necessary to obtain image points. Given that the camera motion, obtained as described in section 5.3.1, is only calculated up to scale, we introduce the scaling factor \( s \):

\[ mX_{\alpha H} = [R|st]X_{\alpha H}, \]  \hspace{1cm} (5.10)

or equivalently:

\[
\begin{bmatrix}
mu_{\alpha} \\
mv_{\alpha} \\
m
\end{bmatrix} =
\begin{bmatrix}
R_{11} & R_{12} & R_{13} & stU \\
R_{21} & R_{22} & R_{23} & stV \\
R_{31} & R_{32} & R_{33} & stW
\end{bmatrix}
\begin{bmatrix}
U_{\alpha} \\
V_{\alpha} \\
W_{\alpha} \\
1
\end{bmatrix}
\]  \hspace{1cm} (5.11)

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

\[
mu_{\alpha} = r_1^TX_{\alpha} + stU \] \hspace{1cm} (5.12)
\[
mv_{\alpha} = r_2^TX_{\alpha} + stV \] \hspace{1cm} (5.13)
\[
m = r_3^TX_{\alpha} + stW. \] \hspace{1cm} (5.14)

5.3.3 Solving for the Scale \( s \)

Given the set of 3 equations in 5.12, 5.13 and 5.14, there are at least 4 different ways to solve the system for \( s \):

- **Method 1:** solve using equations 5.12 and 5.14. This implies using \( u_{\alpha} \) and \( t_U \).
- **Method 2:** solve using equations 5.13 and 5.14. This implies using \( v_{\alpha} \) and \( t_V \).
- **Method 3:** substitute equation 5.14 into equations 5.12 and 5.13
- **Method 4:** The last possibility is using equations 5.12 and 5.13 combining \( u_{\alpha} \), \( v_{\alpha} \), \( t_U \) and \( t_V \) in a single solution.

It is at first sight difficult to choose 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. This is particularly evident for
vehicle mounted cameras where the camera will tilt with the inclination of the ground and the image motion will always be the same. In this case, we can motivate our choice of Method 1 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 or small irregularities on the road. Naturally, if the camera moves mainly in the $y$ direction, the alternative method 2 should be employed. Method 3 is the equivalent of computing methods 1 and 2 and stacking the results to obtain a LS solution. This implies solving $s$ employing DLT, by solving a system of equations of the form $As = b$ using SVD. Method 4 is similar to methods 1 and 2 but using the ratio $\frac{u_{\alpha}}{v_{\alpha}}$ (or the inverse).

5.3.3.1 Solving with Method 1

In order to solve for $s$, we substitute equation 5.14 in equation 5.12 to remove the image scale factor $m$. We then obtain a system of equations in the form $As = b$ such that:

$$A = [t_W u_\alpha - t_U]$$
$$b = [(r_1^T - r_3^T u_\alpha)X_\alpha],$$

(5.15)

(5.16)

We can construct the vectors $A$ and $b$ using one row for every 2D-3D correspondence $\alpha$, and solve for $s$ using SVD (Singular Value Decomposition), obtaining the solution in the Least Squares sense:

$$s = A^+ b,$$

(5.17)

where:

$$A^+ = (A^T A)^{-1} A^T.$$  

(5.18)

5.3.3.2 Experimental Motivation

Clearly our derivation and subsequent analysis hold by simply substituting $t_U$ by $t_V$ and $r_1^T$ by $r_2^T$ (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 the 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 $\Delta u_\alpha$.  

69
We implemented all 4 methods and compared the accuracy in the estimation of the scale. Figure 5.2 shows the results. We created 3 simulated cameras in space, moving in an urban environment along the $V$ direction. Artificial image features were introduced with an increasing error in the position of the pixels. Results show the average error over 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.
5.3 Robust and Scale Consistent Motion Estimation

Figure 5.2: 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.
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

5.4 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 \( A^+ \), \( A \) and \( b \) independently. To first order, the error in equation 5.17 is:

\[
\Delta s = A^+ b + \Delta A^+ b. \tag{5.19}
\]

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

5.4.1 Error Propagation in the Computation of \( A^+ \)

We obtain an expression for the computation of \( A^+ \) in equation 5.18. We now analyze how the error in \( A^+ \) of equation 5.18 is propagated in its computation. We begin by introducing the disturbances:

\[
A^+ + \Delta A^+ = ((A + \Delta A)^T(A + \Delta A))^{-1}(A + \Delta A)^T \tag{5.20}
\]

\[
= (A^T A + A^T \Delta A + \Delta A^T A + \Delta A^T \Delta A)^{-1}(A + \Delta A)^T
\]

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

\[
\Delta A^+ = -(A^T A)^{-1}((A^T \Delta A + \Delta A^T A)(A^T A)^{-1}A^T + \Delta A^T) \tag{5.21}
\]

Proof:

Neglecting some second order terms in equation 5.21 we obtain:

\[
A^+ + \Delta A^+ \approx (A^T A + A^T \Delta A + \Delta A^T A)^{-1}(A + \Delta A)^T \tag{5.22}
\]

We define:
5.4 Error Propagation in Scale Computation

\[ Z = (A^T A + A^T \Delta A + \Delta A^T A)^{-1} \]  \hspace{1cm} (5.23)

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

\[ a = A^T A \]  \hspace{1cm} (5.24)

\[ n = A^T \Delta A + \Delta A^T A. \]  \hspace{1cm} (5.25)

We use these in equation (5.23) and transform it to obtain:

\[ Z = (a + n)^{-1} = (a(1 + a^{-1} n))^{-1} = (1 + a^{-1} n)^{-1} a^{-1}. \]  \hspace{1cm} (5.26)

The term \((1 + a^{-1} n)^{-1}\) is approximated using a first order Taylor series:

\[ (1 + a^{-1} n)^{-1} \approx (1 - a^{-1} n)a^{-1} = a^{-1} - a^{-1} na^{-1} = a^{-1} + (-a^{-1} na^{-1}). \]  \hspace{1cm} (5.27)

This provides a first order approximation of the error propagation in the computation of \( A^+ \):

\[ A^+ + \Delta A^+ = ((A^T A)^{-1} - (A^T A)^{-1} n (A^T A)^{-1})(A + \Delta A)^T \]  \hspace{1cm} (5.28)

\[ \approx (A^T A)^{-1} A^T - (A^T A)^{-1} (n (A^T A)^{-1} A^T + \Delta A^T) \]

\[ = A^+ - (A^T A)^{-1} ((A^T \Delta A + \Delta A^T A)(A^T A)^{-1} A^T + \Delta A^T) \]

Therefore, the disturbance to first order that is propagated through the computation of \( A^+ \) is:

\[ \Delta A^+ = -(A^T A)^{-1} ((A^T \Delta A + \Delta A^T A)(A^T A)^{-1} A^T + \Delta A^T) \]  \hspace{1cm} (5.29)
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

5.4.2 Error Propagation in computation of $A$

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

$$A + \Delta A = [(t_W + \Delta t_W)(u_\alpha + \Delta u_\alpha) - (t_U + \Delta t_U)]$$  \hspace{1cm} (5.30)

$$\Delta A \approx [t_W \Delta u_\alpha + \Delta t_W u_\alpha - \Delta t_U]$$  \hspace{1cm} (5.31)

With the expression [5.16] for $A$, $\Delta A^+$ of [5.29] is now computable.

5.4.3 Error Propagation in Computation of $b$

Finally, we need to obtain an expression for the error propagation in the computation of $b$. We follow the same procedure as before, introducing the disturbances in equation [5.16] and propagating to first order:

$$b + \Delta b = [(r_1^T + \Delta r_1^T - (r_3^T + \Delta r_3^T)(u_\alpha + \Delta u_\alpha)) (X_\alpha + \Delta X_\alpha)]$$ \hspace{1cm} (5.32)

$$\Delta b \approx [(r_1^T + \Delta r_1^T - r_3^T u_\alpha - r_3^T \Delta u_\alpha - \Delta r_3^T u_\alpha) (X_\alpha + \Delta X_\alpha)]$$ \hspace{1cm} (5.33)

5.4.4 Summary and Conclusions on Error Propagation

Putting all four equations [5.16, 5.18, 5.33 and 5.29] 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 in 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 Carlo 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 learned based on noise in the image features. Alternatively, the error propagation on the 8-point algorithm [56] can be employed together with our derivation.
5.5 Computing the Local Scale Optimally

So far we obtained a closed form solution for the computation of the local scale with 1 2D-3D correspondence. We also provided a Least Squares solution when more correspondences are available along with a first order error analysis. Given this, we can also obtain an optimal estimation of the scale in closed form by solving a minimization problem.

5.5.1 Considerations

As show in Section 5.3.2 we can obtain for every 2D-3D correspondence an estimation of the local scale. This estimation is affected by noise, for which we also find an expression in equation 5.19 such that:

\[ s_\alpha = \bar{s}_\alpha + \Delta s_\alpha. \]  

(5.34)

Each \( s_\alpha \) has been obtained using a unique 2D-3D correspondence. \( \Delta s \) is described in equation 5.19 as a statically independent random variable with mean 0 and covariance \( V[s_\alpha] = \Delta s_\alpha \Delta s_\alpha^T \). We now want to find an optimal average of \( s \) that satisfies the constraints:

\[ \bar{s}_\alpha = s. \]  

(5.35)

for \( \alpha = 1, ..., N \). This means finding the optimal average for the scale.

5.5.2 Maximum Likelihood Estimation

Following the work in [55] we find the optimal parameter \( s \).

We begin by assuming one particular value for \( s \). Then each \( s_\alpha \) is corrected by minimizing the Mahalanobis distance. This means solving the minimization problem:

\[ J[s] = \sum_{\alpha=1}^{N} \left( (s_\alpha - s) \cdot V[s_\alpha]^{-1}(s_\alpha - s) \right) \rightarrow \min, \]  

subject to the constraint \( \hat{s}_\alpha = s \) for \( \alpha = 1, ..., N \), where \( \hat{s}_\alpha \) is the maximum likelihood estimator and \( V[s_\alpha] \) is the covariance of the measured scale \( s_\alpha \).

The solution to this minimization is given by:
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

\[ \hat{s} = \left( \sum_{\alpha=1}^{N} V[\alpha]^{-1} \right)^{-1} \left( \sum_{\alpha=1}^{N} V[\alpha]^{-1} s_{\alpha} + s \right) \]  \hspace{1cm} (5.37)

And its covariance matrix:

\[ V[\hat{s}] = \left( \sum_{\alpha=1}^{N} V[\alpha]^{-1} \right)^{-1} \]  \hspace{1cm} (5.38)

5.6 Experiments and Results

5.6.1 Comparison with State of the Art Methods

In this section we compare our approach for estimating the scale with 2 state-of-the-art methods. For this we built a fully simulated scene of which 3 synthetic images are recorded. Thirty 3D points were projected onto the 3 cameras located at different positions. The internal parameters of the cameras were considered perfect. Gaussian noise was introduced only in the image feature location. The motion of the first pair of images was calculated using the 8-point algorithm. The required 3D points for estimating the scale and motion of the third frame were obtained through triangulation of the noisy image features.

We performed 2000 runs in which a different 3D scene was created and the cameras were randomly positioned. For each run, we estimated the motion of camera 3 using EPnP \cite{38}, P1.5P \cite{48} and our novel approaches: the Least Squares method P1Pls and the optimal estimation P1Popt. We incremented the noise in the image feature locations from 0 to 0.4 pixels\footnote{0.4 pixels was the standard deviation obtained when calibrating the cameras and performing corner detection over a checkerboard.} of standard deviation. Then we compared the estimated camera poses with the ground truth camera poses. We performed additional experiments with alternative algorithms such as P6P and P4Pf \cite{40} but the resulting accuracy was worse than any of the presented methods and therefore not shown in the plots.

5.6.1.1 EPnP:

Moreno-Noguer et. al. \cite{38} present a non-iterative O(n) algorithm (EPnP) for estimating the scale consistent camera pose. EPnP requires at least 4 3D-2D point correspondences and is reported to be faster and more accurate than any other closed form
5.6 Experiments and Results

Table 5.1: Algorithm comparison

<table>
<thead>
<tr>
<th></th>
<th>P1Pls/P1Popt</th>
<th>P1.5P</th>
<th>EPnP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min. 3D-2D correspondences</td>
<td>1</td>
<td>1.5</td>
<td>4</td>
</tr>
<tr>
<td>Max. 3D-2D correspondences</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Estimation</td>
<td>s</td>
<td>s and t</td>
<td>R, s and t</td>
</tr>
<tr>
<td>Estimation of R</td>
<td>8-point</td>
<td>8-point</td>
<td>–</td>
</tr>
<tr>
<td>Estimation of t</td>
<td>8-point</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Complexity</td>
<td>O(n)</td>
<td>O(n)</td>
<td>O(n)</td>
</tr>
</tbody>
</table>

approach [54]. Additionally, the authors compare with one of the top iterative methods and even though the results are marginally worse, EPnP is more stable and faster. The authors provide a Matlab implementation of the algorithm that we used for our experiments.

5.6.1.2 P1.5P:

Tardif et. al. [48] propose a linear 1.5-point algorithm to solve for the scale adjusted translation while fixing the rotation. The algorithm is basically a Direct Linear Transform [1] algorithm where the 3 parameters of the translation are estimated in a Least Squares approach. In this respect the algorithm is similar to our Least Squares approach, only we do not use SVD for solving the system and rather derive a closed form solution. The authors however do not consider the error propagation nor any form of optimal computation. In this case the rotation is also computed with the 8-point algorithm.

Figure 5.3 shows the results of accuracy in rotation, scale and translation direction. Our method only deals with the scale, however we consider relevant the comparison of accuracy between methods that use 2D-3D correspondences (PnP) and methods that use only 2D-2D (image only) correspondences. For the latter there is usually a larger number of correspondences given that only 2 images are required.

Regarding the rotation, the plot shows that accuracy of the EPnP is only marginally worse than that of the 8-point algorithm.
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

Figure 5.3: 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: error in the rotation calculated as absolute angle of $\Delta R$ in 5.7. The 1-point coincides with 1.5 because the rotation is calculated with the 8-point algorithm. MIDDLE: error in the translation vector calculated as the distance of the difference vector between ground truth and estimation. BOTTOM: error in the angle estimation of the translation direction. NOTE that 1-point (Least Squares) and 1-point (Optimal) are the same for translation and rotation accuracy. Moreover 1-point and P1.5P are the same for rotation accuracy.
With respect to the estimation of the translation, the 8-point algorithm offers a significant advantage over both the EPnP and the P1.5P algorithms. This can be partially explained by the fact that any method that uses 2D-3D correspondences deals with reconstructed 3D features that contain an accumulated error from both the image space and the estimated camera motion. This error is then passed on to the computation of the rotation and translation. For the case of the methods that use only 2D-2D correspondences, the error propagated to the translation originates only in the image space. In this case the advantage of using a 2D-2D method is clear. There are however scenarios in which such a method will fail. In particular the 8-point algorithm is very sensitive to points lying approximately in a plane. In this case another algorithm can be employed such as the 5-point algorithm, although a simple optimization scheme could deal with some of these shortcomings.

The last figure (see figure 5.3 middle) shows the accuracy of all 4 methods in the computation of the local scale. In this case the P1.5P algorithm performs the worst since the scale is computed together with the translation direction as a Least Squares solution and the error is accumulated as described above. The second method in terms of accuracy is the EPnP. It is significantly better than P1.5P, with a 40% increase in accuracy. The third method is our 1 point Least Squares solution (P1Pls). This method performs very similarly to EPnP offering only an improvement of 10%. Our method for optimal estimation of the scale (P1Popt) performs much better than the other methods offering an improvement of 50% over P1Pls.

5.6.2 Outdoors Visual Odometry

We also performed experiments on real outdoors data. In these experiments, images were recorded with a handheld digital reflex camera (DSLR). We used a wide angle lens at a resolution of 1728x1152 pixels. Additionally we tested our method with the well known benchmark dataset P11 [44].

**Outdoors Scene 1 - 60 images:** For the first set, 60 images were recorded covering a total distance of 100 meters of an urban area (see Figure 5.4), following a straight trajectory along one street (see Figure 5.6).

**Outdoors Scene 2 - 174 images:** For the second set, a total 174 images were recorded covering a distance of 200+ meters (see Figure 5.7). In this case the camera moved around a block of houses (see Figure 5.8). The same image was used for first and last
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

position to ensure that the true last camera pose was exactly the same as where the first image was recorded.

The trajectories and triangulated 3D points are shown in Figures 5.6 and 5.8. For the first set, the 3D structure was recovered consistently and the local scale of the map was correctly estimated. Both trajectories were recovered accurately without the aid of any global iterative refinement scheme such as Bundle Adjustment. Given this accuracy, a 3D model was also obtained for the first set (see Figure 5.5) using the methods described in Chapter 4. In terms of accuracy, in the second dataset, the distance between the estimated position of the last image and the first image was below 1 meter. This represents an error of less than 0.5%.

Figure 5.4: Sample images of first dataset.

Figure 5.5: Top: Estimated trajectory (red circles) and recovered point cloud for first set. Bottom: reconstructed 3D model. No global Bundle Adjustment was necessary.
5.6 Experiments and Results

Figure 5.6: Top view of the reconstructed scene (in black) and camera positions (red circles)

Figure 5.7: Sample images of second dataset.
Figure 5.8: 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 applied.
**5.6 Experiments and Results**

**Benchmark Scene P11 - 11 images:** We also tested our approach on the well known dataset P-11 provided by [44]. In this set, 11 pictures (see Figure 5.9) were taken sequentially around a fountain. The dataset is provided along with ground truth camera positions that were obtained using a laser range finder. The camera positions were estimated accurately and the image key features were reconstructed successfully. The structure of the fountain was recovered as shown in Figure 5.9. The mean error on the computation of the scale for the 9 cameras (the first one is assumed to be at the center of coordinates and the next one set to a distance of one unit) was 0.14% with a standard deviation of $1.55 \times 10^{-06}$.

![Figure 5.9: P11 dataset consisting of 11 images.](image)

![Figure 5.10: 3D reconstruction of P11 benchmark dataset. No bundle adjustment was used and the scale between consecutive images was computed using our optimal algorithm.](image)
5. OPTIMAL LOCAL SCALE IN CAMERA MOTION

5.7 Conclusions

In this chapter we have presented a closed form solution for the computation of the scale ratio between the translation directions of two consecutive image pairs. Our method requires only one 2D-3D correspondence, though a Least Squares solution is presented when more matches are available. Additionally we provided 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 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. Furthermore we took advantage of the estimation of the propagated error and computed the scale optimally also in closed form. This yields results that outperform any benchmark algorithm we compared it to.

Combined with the 8-point algorithm and a local iterative refinement step we achieved accurate results with little drift over large trajectories. Our optimal closed-form solution, integrated in the reconstruction pipeline as we described in Chapter 3, allows for accurate reconstruction. We demonstrated this by reconstructing a street in an urban scene without the need for any global optimization.