Camera motion estimation by tracking contour deformation: Precision analysis ⋆

G. Alenyà *, C. Torras

Institut de Robòtica i Informàtica Industrial, CSIC-UPC
Llorens i Artigas 4-6, 08028 Barcelona

Abstract

An algorithm to estimate camera motion from the progressive deformation of a tracked contour in the acquired video stream has been previously proposed. It relies on the fact that two views of a plane are related by an affinity, whose 6 parameters can be used to derive the 6 degrees-of-freedom of camera motion between the two views. In this paper we evaluate the accuracy of the algorithm. Monte Carlo simulations show that translations parallel to the image plane and rotations about the optical axis are better recovered than translations along this axis, which in turn are more accurate than rotations out of the plane. Concerning covariances, only the three less precise degrees-of-freedom appear to be correlated. In order to obtain means and covariances of 3D motions quickly on a working robot system, we resort to the Unscented Transformation (UT) requiring only 13 samples per view, after validating its usage through the previous Monte Carlo simulations. Two sets of experiments have been performed: short-range motion recovery has been tested using a Staübli robot arm in a controlled lab setting, while the precision of the algorithm when facing long translations has been assessed by means of a vehicle-mounted camera in a factory floor. In the latter more unfavourable case, the obtained errors are around 3%, which seems accurate enough for transferring operations.

Key words: Egomotion estimation, active contours, precision analysis, unscented transformation

⋆ This work is partially funded by the EU PACO-PLUS project FP6-2004-IST-4-27657, the Consolider Ingenio 2010 project CSD2007-00018, and the Catalan Research Commission through the Robotics consolidated group. Guillem Alenyà was supported by CSIC under a JAE-Doc fellowship.

* Corresponding author.
Email addresses: galenya@iri.upc.edu (G. Alenyà), torras@iri.upc.edu (C. Torras).

1 Introduction

The importance conferred to noise in computer vision has progressively increased along the years. While the first visual geometry algorithms focused on the minimum number of points required to obtain a desired information, later input redundancy was incorporated into these algorithms to cope with real noisy images, and nowadays error propagation and uncertainty estimation techniques are being applied as a necessary step to then actively try to reduce uncertainty.

Algorithms to recover epipolar geometry and egomotion have followed this general trend. Although eight point matches are known to be sufficient to derive the fundamental matrix [1, 2], redundant matchings lead to a more useful estimation in practice [3, 4]. A first step to explicitly deal with errors is to detect outliers. The two most popular algorithms are: Least Mean Squares (LMedS), of which Zhang [5] gives a detailed description, and Random Sample Consensus (RANSAC), proposed by Fischler and Bolles [6]. Torr and Murray [7] provide a comparative study of them.

The next step is to model input noise in order to analyse how it propagates to the output. Good introductions with examples applied to real cases can be found in [8] and [9]. In the absence of a priori knowledge, input uncertainty is often assumed to obey a Gaussian distribution [4]. Variable interdependence, when available, is usually represented by means of a covariance matrix [10, 11]. Some studies [12, 13] show that using covariances to characterize uncertainty in the location of point features within algorithms based on point correspondences may lead to improvement over algorithms that do not take uncertainty into account. Moreover, uncertainty in the estimation of the covariance matrix also affects the quality of the outputs.

Once input noise is modelled, uncertainty propagation can be studied either analytically or statistically. Analytic studies often entail deriving the Jacobian of the input-output relation which, for nonlinear functions, requires resorting to linear approximations. Gonçalves and Araújo [14] carry out such an analysis for egomotion recovery from stereo sequences. Other egomotion estimation algorithms entail a singular value decomposition (SVD) and specific procedures have been developed to analyse how uncertainty propagates to singular vectors and values [15, 16].

Uncertainty propagation can also be studied from a statistical viewpoint. Monte Carlo simulation is a powerful and simple tool [17] that entails sampling the input space densely and executing the algorithm for each sample. Thus, it is only affordable when few results are needed or computing time is not a concern. To speed up the process, Julier and Uhlmann proposed the

Unscented Transformation (UT), which attempts to find the smallest sample set that captures the statistical distribution of the data. The appropriateness of UT for a particular problem can be tested through Monte Carlo simulation and, if validated, this transformation leads to considerable time savings.

The goal of this paper is to analyse the accuracy of an algorithm we previously proposed [18, 19] to estimate camera motion from the progressive deformation of a tracked contour in the acquired images. Section 2 briefly reviews how the affinity linking two views is extracted from contour deformation, and how this affinity is then used to derive 3D camera motion in our algorithm. Next, in Section 3, the precision of this algorithm is analysed using Monte Carlo simulation.

A second aim of this work is to obtain the covariance matrix of the six degrees-of-freedom of egomotion in real-time, in order to use it in robotics applications. For this, the Unscented Transformation becomes of great use, as explained in Section 4. By allowing to propagate covariances, this transformation permits analysing correlations between the recovered translations and rotations. Finally, some conclusions and future prospects are presented in Section 5. Additionally, Appendix A contains the complete results of the covariance propagation performed in Section 4.

2 Mapping contour deformations to camera motions

Under weak-perspective conditions (when the depth variation of the viewed object is small compared to its distance to the camera, and the object is close to the principal ray), the change in the image projection of an object in two different views can be parameterised as an affine deformation in the image plane, which can be expressed as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{M} \begin{bmatrix} x \\ y \end{bmatrix} + \mathbf{t} \tag{1}$$

where (x, y) is a point in the first view, (x', y') is a point in the second view, and $\mathbf{M} = [M_{i,j}]$ and $\mathbf{t} = (t_x, t_y)$ are, respectively, the matrix and vector defining the affinity in the plane.

Assuming restricted weak-perspective imaging conditions instead of the more general perspective case is advantageous when perspective effects are not present or are minor [20]. The parameterisation of motion as an affine image deformation has been used before for active contour tracking [21], qualitative robot pose estimation [18] and visual servoing [22].

The affinity relating two views is usually computed from a set of point matches [23, 24]. However, point matching algorithms require richly textured objects, or scenes, which often are not available. In these situations, object contours may be easier to find [9]. In this work an active contour [21] fitted to a target object is used as an alternative to point matching. The active contour is coded as a B-Spline [25] and, accordingly, a small vector of control points is enough to represent the whole contour.

Let $\mathbf{Q}$ be the vector formed with the coordinates of N_Q control points, first all the x-coordinates, and then all the y-coordinates:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q^x} \\ \mathbf{Q^y} \end{bmatrix} . \tag{2}$$

It has been formerly demonstrated [18, 21] that the difference between two views of a contour in terms of control points, $\mathbf{Q'} - \mathbf{Q}$, can be written as

$$\mathbf{Q'} - \mathbf{Q} = \mathbf{WS} \tag{3}$$

where $\mathbf{W}$ is the *shape matrix*

$$\mathbf{W} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{Q^x} & \mathbf{0} & \mathbf{0} & \mathbf{Q^y} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{Q^y} & \mathbf{Q^x} & \mathbf{0} \end{bmatrix} \tag{4}$$

composed of $\mathbf{Q^x}$, $\mathbf{Q^y}$, and the N_Q-dimensional vectors $\mathbf{0} = (0, 0, \ldots, 0)^T$ and $\mathbf{1} = (1, 1, \ldots, 1)^T$, and where

$$\mathbf{S} = (t_x, t_y, M_{11} - 1, M_{22} - 1, M_{21}, M_{12}) \tag{5}$$

is the 6-dimensional *shape vector* that in fact encodes the image deformation from the first to the second view.

In our implementation, the contour is tracked along the image sequence with a Kalman filter [21] and, for each frame, the shape vector and its associated covariance matrix are updated. The affinity coded by the shape vector relates to the 3D camera motion in the following way [18, 21]:

$$\mathbf{M} = \frac{Z_0}{Z_0 + T_z} \begin{bmatrix} R_{11} & R_{21} \\ R_{21} & R_{22} \end{bmatrix} , \tag{6}$$

$$\mathbf{t} = \frac{1}{Z_0 + T_z} \begin{bmatrix} T_x \\ T_y \end{bmatrix} , \tag{7}$$

where R_{ij} are the elements of the 3D rotation matrix $\mathbf{R}$, T_i are the elements of the 3D translation vector $\mathbf{T}$, and Z_0 is the distance from the viewed object to the camera in the initial position.

We will see next how the 3D rotation and translation are obtained from the $\mathbf{M} = [M_{i,j}]$ and $\mathbf{t} = (t_x, t_y)$ defining the affinity. Representing the rotation matrix in Euler angles form,

$$\mathbf{R} = \mathbf{R_z}(\phi)\mathbf{R_x}(\theta)\mathbf{R_z}(\psi), \tag{8}$$

where $\mathbf{R_z}(\phi)$ is the matrix encoding a rotation of ϕ about the Z axis, $\mathbf{R_x}(\theta)$ is the matrix encoding a rotation of θ about the X axis, and $\mathbf{R_z}(\psi)$ is the matrix encoding a rotation of ψ about the Z axis. Note that due to the particular arrangement of those rotation matrices, here the upper 2×2 principal sub-matrix $\mathbf{R}|_\mathbf{2}$ of $\mathbf{R}$ is equal to the product of the corresponding upper 2×2 principal sub-matrices of the decomposed rotation

$$\mathbf{R}|_\mathbf{2} = \mathbf{R_z}|_\mathbf{2}(\phi)\mathbf{R_x}|_2(\theta)\mathbf{R_z}|_2(\psi)\,. \tag{9}$$

Combining (6) and (9) we obtain

$$\begin{aligned}\mathbf{M} =& \frac{Z_0}{Z_0 + T_z}\mathbf{R_z}|_\mathbf{2}(\phi)\mathbf{R_x}|_\mathbf{2}(\theta)\mathbf{R_z}|_\mathbf{2}(\psi) = \\ &= \frac{Z_0}{Z_0 + T_z}\mathbf{R_z}|_\mathbf{2}(\phi)\begin{bmatrix}1 & 0\\ 0 & cos\theta\end{bmatrix}\mathbf{R_z}|_\mathbf{2}(\psi)\,.\end{aligned} \tag{10}$$

Then,

$$\mathbf{M}\mathbf{M}^T = \mathbf{R_z}|_\mathbf{2}(\phi)\begin{bmatrix}L & 0\\ 0 & Lcos^2\theta\end{bmatrix}\mathbf{R_z}|_\mathbf{2}{}^{-1}(\phi) \tag{11}$$

where

$$L = \left(\frac{Z_0}{Z_0 + T_z}\right)^2.$$

This last equation shows that θ can be calculated from the eigenvalues of the matrix $\mathbf{M}\mathbf{M}^T$, which we will name (λ_1, λ_2):

$$cos\theta = \sqrt{\frac{\lambda_2}{\lambda_1}}, \tag{12}$$

where λ_1 is the largest eigenvalue. The angle ϕ can be extracted from the eigenvectors of $\mathbf{M}\mathbf{M}^T$; the eigenvector $\mathbf{v_1}$ with larger value corresponds to the

first column of $\mathbf{R_z}|_{\mathbf{2}}(\phi)$:

$$\mathbf{v_1} = \begin{bmatrix} cos\phi \\ sin\phi \end{bmatrix}. \tag{13}$$

Isolating $\mathbf{R_z}|_{\mathbf{2}}(\psi)$ from Equation (10),

$$\mathbf{R_z}|_{\mathbf{2}}(\psi) = (1 + \tfrac{T_z}{Z_0}) \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{cos\theta} \end{bmatrix} \mathbf{R_z}|_{\mathbf{2}}(-\phi)\mathbf{M}, \tag{14}$$

and observing, in Equation (11), that

$$1 + \frac{T_z}{Z_0} = \frac{1}{\sqrt{\lambda_1}},$$

$sin\psi$ can be found, and then ψ.

Once the angles θ, ϕ, ψ are known, the rotation matrix $\mathbf{R}$ can be derived from Equation (8).

The scaled translation in direction Z is calculated as

$$\frac{T_z}{Z_0} = \frac{1}{\sqrt{\lambda_1}} - 1. \tag{15}$$

The rest of components of the 3D translation can be derived from $\mathbf{t}$ and $\mathbf{R}$ using Equation (7):

$$\frac{T_x}{Z_0} = \frac{t_x}{\sqrt{\lambda_1}}, \tag{16}$$

$$\frac{T_y}{Z_0} = \frac{t_y}{\sqrt{\lambda_1}}. \tag{17}$$

Using the equations above, the deformation of the contour parameterized as a planar affinity permits deriving the camera motion in 3D space. Note that, to simplify the derivation, the reference system has been assumed to be centered on the object.

3 Assessing the precision of the obtained motion components

3.1 Rotation representation and systematic error

As shown in Equation (8), rotation is codified as a sequence of Euler angles $\mathbf{R} = \mathbf{R_z}(\phi)\mathbf{R_x}(\theta)\mathbf{R_z}(\psi)$. Typically, this representation has the problem of the Gimbal lock: when two axes are aligned there is a problem of indetermination. In a noisy scenario, this happens when the second rotation $\mathbf{R_x}(\theta)$ is near the null rotation. As a result, small variations in the camera pose due to noise in the contour acquisition process do not lead to continuous values in the rotation representation (see $\mathbf{R_z}(\phi)$ and $\mathbf{R_z}(\psi)$ in Fig. 1(a)). Using this representation, means and covariances cannot be coherently computed. In our system this could happen frequently, for example at the beginning of any motion, or when the robot is moving towards the target object with small rotations.

We propose to change the representation to a *roll-pitch-yaw* codification. It is frequently used in the navigation field, it being also called *heading-attitude-bank* [26]. We use the form

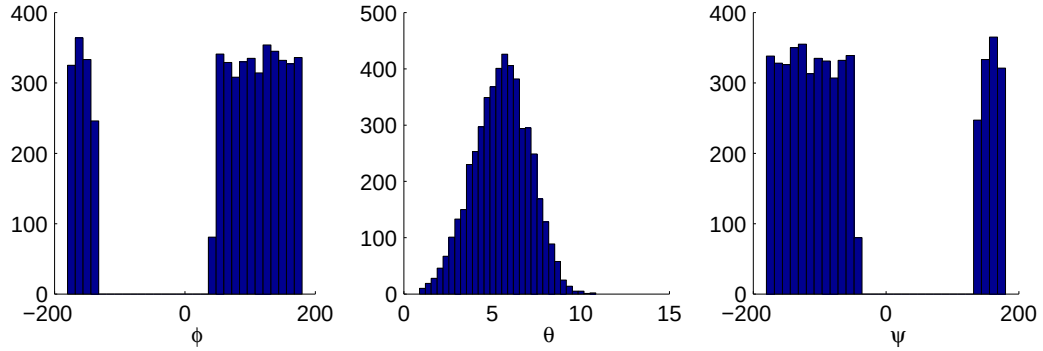
$$\mathbf{R} = \mathbf{R_z}(\psi)\mathbf{R_y}(\theta)\mathbf{R_x}(\phi) = \begin{bmatrix} c_\psi c_\theta & s_\psi c_\phi + c_\psi s_\theta s_\phi & s_\psi s_\phi - c_\psi s_\theta c_\phi \\ -s_\psi c_\theta & c_\psi c_\phi - s_\psi s_\theta s_\phi & c_\psi s_\phi + s_\psi s_\theta c_\phi \\ s_\theta & -c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}, \quad (18)$$

where s_ψ and c_ψ denote the sinus and cosinus of ψ, respectively. The inverse solution is:

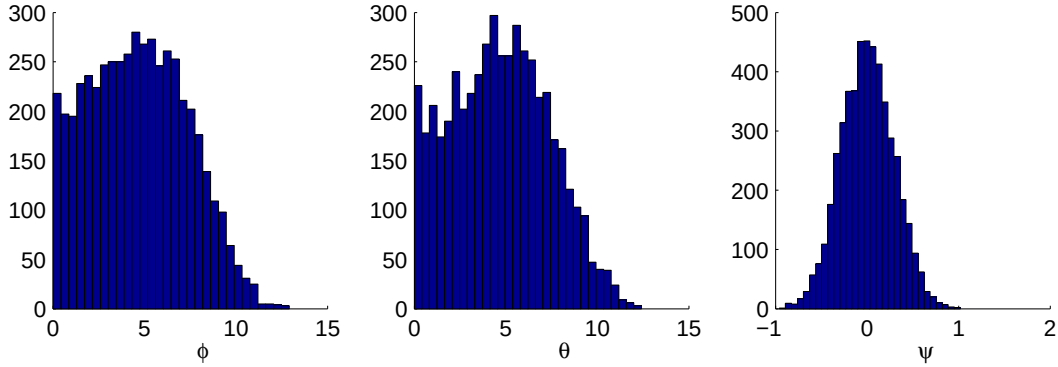
$$\phi = \arctan 2(R_{32}, R_{33}) \quad (19)$$
$$\theta = \arcsin(-R_{31}) \quad (20)$$
$$\psi = \arctan 2(R_{21}, R_{11}). \quad (21)$$

Typically, in order to represent all the rotation space the elemental rotations should be restricted to lie in the $[0..2\pi]$ rad range for ψ and ϕ, and in $[0..\pi]$ rad for θ.

Indeed, tracking a planar object by rotating the camera about X or Y further than $\pi/2$ radians has no sense, as in such position all control points lie on a single line and the shape information is lost. Also, due to the *Necker reversal* ambiguity, it is not possible to determine the sign of the rotations about these axes. Consequently, without loss of generality, we can restrict the range of the rotations around the X and Y axis, ϕ and θ respectively, to lie in the range



(a) Rotation matrix is obtained by composing $\mathbf{R} = \mathbf{R_z}(\phi)\mathbf{R_x}(\theta)\mathbf{R_z}(\psi)$ as proposed before in [21]

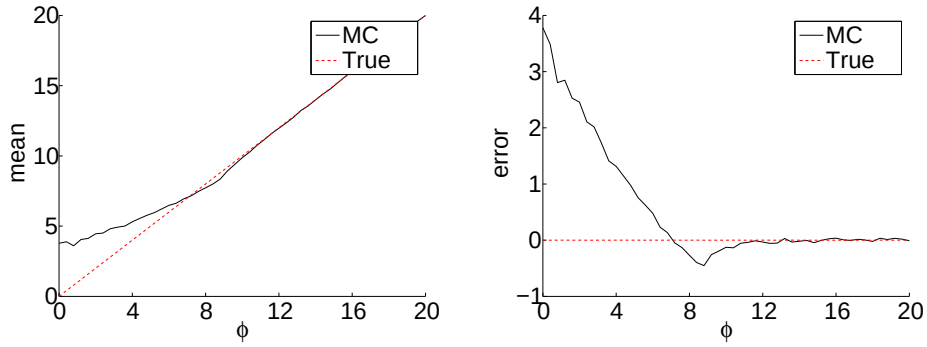


(b) Rotation matrix is obtained by composing $\mathbf{R} = \mathbf{R_z}(\psi)\mathbf{R_y}(\theta)\mathbf{R_x}(\phi)$ as proposed here in (18)

Figure 1. Histogram of the computed rotation values for 5000 trials adding Gaussian noise with $\sigma = 0.5$ pixels to the contour control points. (a) In the ZXZ representation, small variations of the pose correspond to discontinuous values in the rotation components $R_z(\phi)$ and $R_z(\psi)$. (b) In contrast, the same rotations in the ZYX representation yield continuous values.

$[0..\frac{\pi}{2})$ radians and let $\mathbf{R_z}(\psi)$ in $[0..2\pi]$ radians. With this representation, the Gimbal lock has been displaced to $\cos(\theta) = 0$, but $\theta = \pi/2$ is out of the range in our application.

With the above-mentioned sign elimination, a bias is introduced for small $\mathbf{R_x}(\phi)$ and $\mathbf{R_y}(\theta)$ rotations. In the presence of noise and when the performed camera rotation is small, negative rotations will be estimated positive. Thus, the computation of a mean pose, as presented in the next Section, will be biased. Figure 2(a) plots the results of an experiment where the camera performs a rotation from 0 to 20° about the X axis of a coordinate system located at the target. Clearly, the values $\mathbf{R_x}(\phi)$ computed by the Monte Carlo simulation are closer to the true ones as the amount of rotation increases. Figure 2(b) summarizes the resulting errors. This permits evaluating the amount of systematic error introduced by the rotation representation.



(a) Rotation values (b) Error

Figure 2. Systematic error in the R_x component. Continuous line for values obtained with Monte Carlo simulation and dotted line for true values. The same is applicable to the R_y component.

In sum, the proposed rotation space is significantly reduced, but we have shown that it is enough to represent all possible real situations. Also, with this representation the Gimbal lock is avoided in the range of all possible data. As can be seen in Figure 1(b), small variations in the pose lead to small variations in the rotation components. Consequently, means and covariances can be coherently computed with Monte Carlo estimation. A bias is introduced when small rotations about X and Y are performed, which disappears when the rotations become more significant. As will be seen later, this is not a shortcoming in real applications.

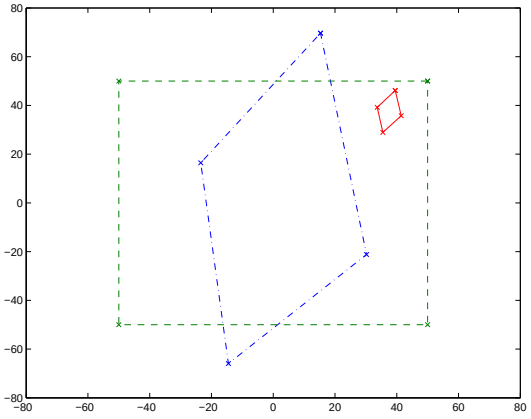


Figure 3. Original contour projection (dotted line), contour projection after the combined motion detailed in Section 3.2.2 (continous line), and contour projection with the combined motion excluding the translation (point-dotted line) to better appreciate the extreme deformation reached with rotations in the experiments, including some perspective effects.

3.2 Uncertainty propagation to each motion component

The synthetic experiments are designed as follows. A set of control points on the 3D planar object is chosen defining the B-Spline parameterisation of its contour. The control points of the B-Spline are projected using a perspective camera model yielding the control points in the image plane (Fig. 3). Although the projection is performed with a complete perspective camera model, the recovery algorithm assumes a weak-perspective camera. Therefore, the perspective effects show up in the projected points (like in a real situation) but the affinity is not able to model them (only approximates the set of points as well as possible), so perspective effects are modelled as affine deformations introducing some error in the recovered motion. For these experiments the camera is placed at 5000 mm and the focal distance is set to 50 mm.

Several different motions are applied to the camera depending on the experiment. Once the camera is moved, Gaussian noise with zero mean and $\sigma = 0.5$ is added to the new projected control points to simulate camera acquisition noise. We use the algorithm presented in Section 2 to obtain an estimate of the $3D$ pose for each perturbed contour in the Monte Carlo simulation. 5000 perturbed samples are taken. Next, the statistics are calculated from the obtained set of pose estimations.

3.2.1 Effect of noise on the recovery of a single translation or rotation

Here we would like to determine experimentally the performance (mean error and uncertainty) of the pose recovery algorithm for each camera component motion, that is, translations T_x, T_y and T_z, and rotations R_x, R_y and R_z. The first two experiments involve lateral camera translations parallel to the X or Y axes. With the chosen camera configuration, the lateral translation of the camera up to 300 mm takes the projection of the target from the image center to the image bound. The errors in the estimations are presented in Figure 4(a) and 4(c), and as expected are the same for both translations. Observe that while the camera is moving away from the initial position, the error in the recovered translation increases, as well as the corresponding uncertainty. The explanation is that the weak-perspective assumptions are less satisfied when the target is not centered. However, the maximum error in the mean is about 0.2%, and the worst standard deviation is 0.6%, therefore lateral translations are quite correctly recovered. As shown in [27], the sign of the error depends on the target shape and the orientation of the axis of rotation.

The third experiment involves a translation along the optical axis Z. From the initial distance $Z_0 = 5000$ the camera is translated to $Z = 1500$, that is a translation of -3500 mm. With this translation the effects of approximating

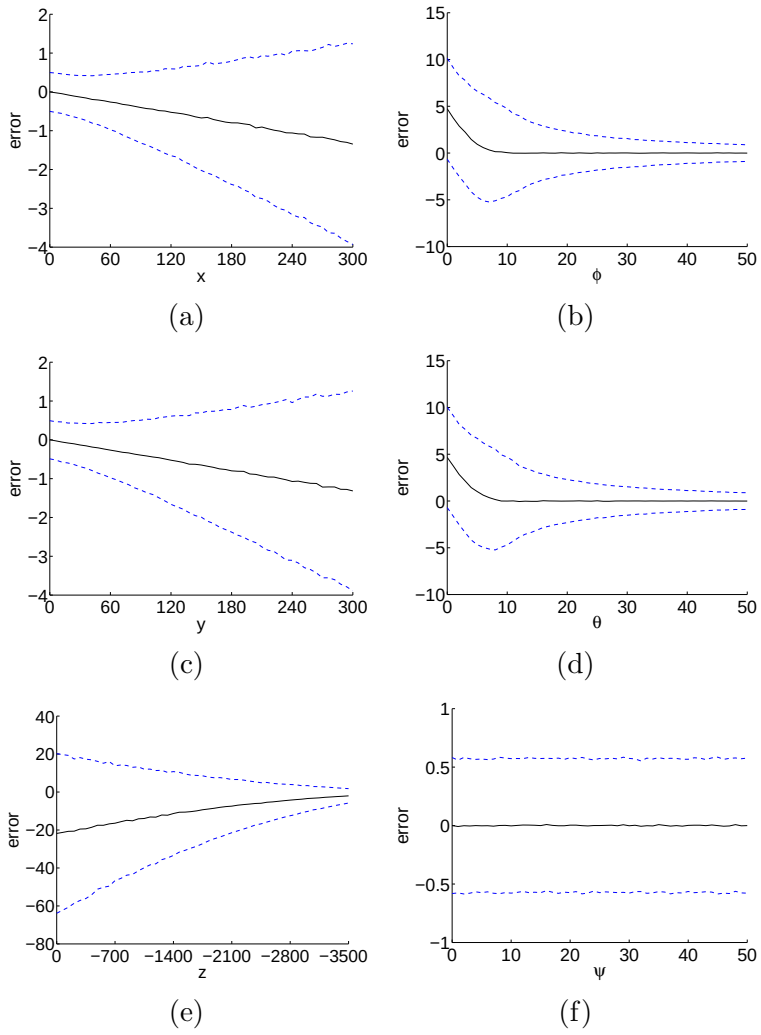


Figure 4. Mean error (solid lines) and 2σ deviation (dashed lines) for pure motions along and about the 3 coordinate axes of a camera placed at 5000 mm and focal length 50 mm. Errors in T_x and T_y translations are equivalent, small while centered and increasing while uncentered, and translation is worst recovered for T_z (although it gets better while approximating). Errors for small R_x and R_y rotations are large, as contour deformation in the image is small, while for large transformations errors are less significant. The error in R_z rotations is negligible.

can be clearly appreciated. The errors and the confidence values are shown in Figure 4(e). As the camera approaches the target, the mean error and its standard deviation decrease. This is in accordance with how the projection works [1]. As expected, the precision of the translation estimates is worse for this axis than for X and Y.

[1] The resolution in millimeters corresponding to a pixel depends on the distance of the object to the camera. When the target is near the camera, small variations in depth are easily sensed. Otherwise, when the target is far from the camera, larger motions are required to be sensed by the camera.

The next two experiments involve rotations of the camera about the target. In the first, the camera is rotated about the X and Y axes of a coordinate system located at the target. Figure 4(b) and 4(d) show the results. As expected, the obtained results are similar for these two experiments. We use the alternative rotation representation presented in Section 3.1, so the values R_x and R_y are restricted. As detailed there, all recovered rotations are estimated in the same side of the null rotation, thus introducing a bias. This is not a limitation in practice since, as will be shown in experiments with real images, the noise present in the tracking step masks these small rotations, and the algorithm is unable to distinguish rotations of less than about 10° anyway.

The last experiment in this Section involves rotations of the camera about Z. As expected, the computed errors (Fig. 4(f)) show that this component is accurately recovered, as the errors in the mean are negligible and the corresponding standard deviation keeps also close to zero.

3.2.2 Effect of noise on the recovery of a composite motion

In the next experiment a trajectory is performed combining all component motions along and about the 3 coordinate axes. With respect the last experiment, the T_z motion has been reversed to go from 5000 mm to 8500. This is because the approach up to 1500 mm performed in the preceding experiment, combined with the lateral translation, would take the target out of the image.

Obviously, the lateral translation is not sensed in the same way when the camera is approaching (Figure 4(a)) as when the camera is receding. At the end of the camera motion the target projection is almost centered and, as can be observed in Figures 5(a) and 5(c), the error in the lateral translation recovery keeps close to 0 as the projection is almost centered in all the sequence. Congruent with receding motion, the uncertainty grows as the camera gets farther from the target. Comparing Figure 5(a) with Figure 4(a), observe that uncertainty grows in both cases, but it is caused by different reasons.

Depth translation recovery error is shown in Figure 5(e). It is nearly null along all the motion, except at the beginning of the sequence, when the camera has not moved. We will show later the this is due to the bias introduced by the rotation representation together with a correlation between the recovered motions. As soon as rotations R_x and R_y are correctly recovered, the T_z translation is also. As expected, while receding, uncertainty increases. In Figure 4(e), there is a bias error all along the trajectory, because rotations are always null. Instead, in this experiment (Fig. 5(e)), there is a bias error only at the beginning.

Results for rotations R_x and R_y are very similar to those in the preceding experiment. The uncertainty at the end of the sequence is slightly larger due

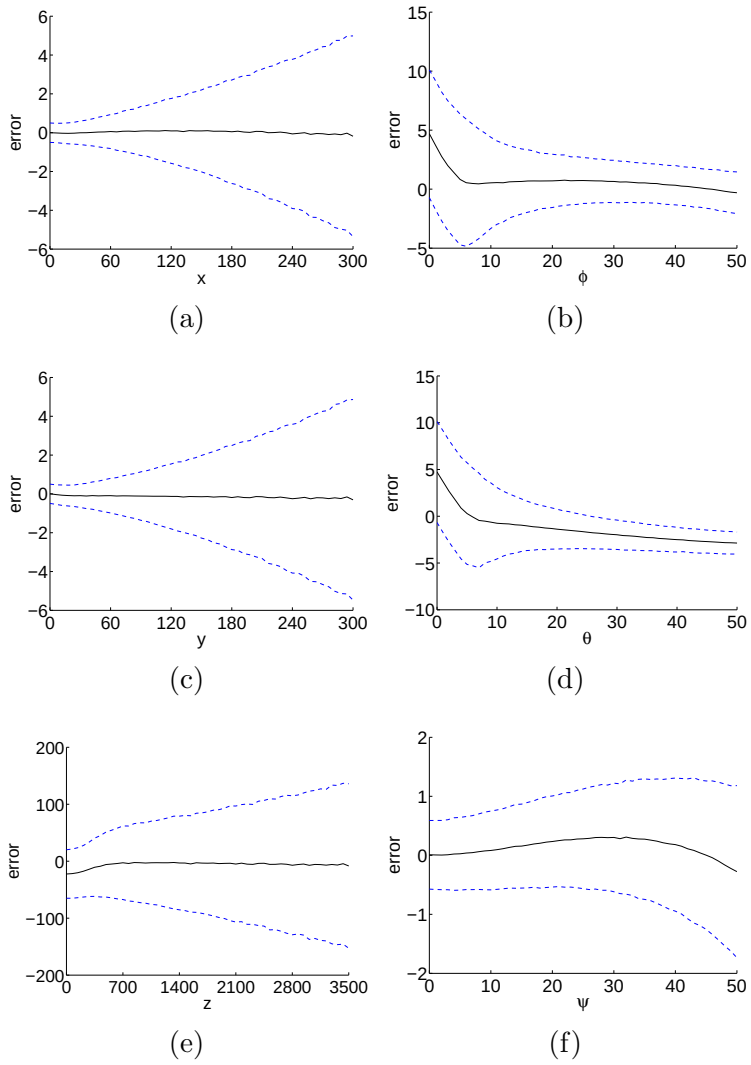


Figure 5. Mean error (solid lines) and 2σ deviation (dashed lines) for a combined motion along and about the 6 coordinate axes of a camera placed at 5000 mm and focal length 50 mm.

to the increment in the distance between camera and target. The same reason is applicable to the uncertainty computed for the R_z rotation (Fig. 5(f)), which also increases. On the other hand, due to the extreme rotation at the end of the sequence ($R_x = 50°$ and $R_y = 50°$), a negligible error in the estimation of the R_z rotation appears.

3.2.3 Sensitivity to the amount of input noise for a composite motion

We would like to compute the estimated uncertainty of the recovered motion as a function of the amount of noise added to the projected contour control points. The experiment setup is the same presented in last Section. The camera motion is defined by a translation of 100 mm along each coordinate axis, and

σ	T_x	T_y	T_z	R_x	R_y	R_z
0.1	0.1105	0.1100	5.2330	0.1982	0.1939	0.0602
0.2	0.2195	0.2185	10.3813	0.3939	0.3850	0.1193
0.3	0.3292	0.3289	15.6153	0.5925	0.5794	0.1791
0.4	0.4390	0.4377	20.7633	0.7910	0.7710	0.2383
0.5	0.5464	0.5465	25.8702	0.9855	0.9616	0.2968
0.6	0.6589	0.6576	31.1632	1.1824	1.1513	0.3612
0.7	0.7681	0.7663	36.3466	1.3787	1.3463	0.4193
0.8	0.8800	0.8786	41.6336	1.5787	1.5415	0.4810
0.9	0.9944	0.9927	47.0746	1.7858	1.7412	0.5449
1.0	1.0991	1.0979	52.0709	1.9856	1.9338	0.6007

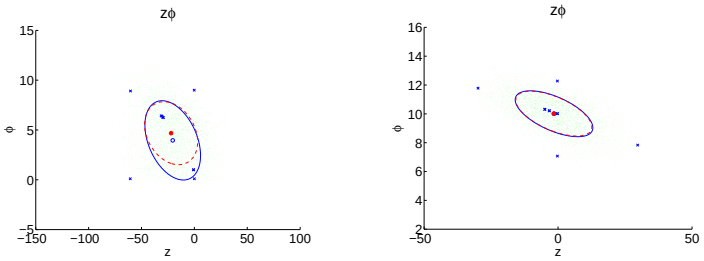
Table 1
Standard deviations of the six component motions for increasing levels of noise added to the contour control points.

a rotation of 30° about an axis centered on the target defined by the vector $(1, 1, 1)$. Gaussian noise with zero mean and standard deviation from $\sigma = 0.1$ to $\sigma = 1.0$ in steps of 0.1 is repeatedly added to the contour control points, yielding a set of shape vectors. From these shape vectors, we calculate the mean shape vector and the covariance in shape space. The results are summarized in Table 1, where the standard deviations are shown.

As expected, as the noise increases the uncertainty also increases. Note that in all motion components the uncertainty increases in the same proportion. Noise of $\sigma = 1.0$ implies a perturbation in the projected control points of ± 2 pixels, which is a considerable one. In this situation, uncertainties in the T_x, T_y and R_z components -i.e., motions within the frontoparallel plane- are very small. Uncertainties in R_x and R_y components are larger. The worstly recovered component is T_z. Remember that, as has been previously shown, the performance in this component depends on the initial distance. Here it was $Z_0 = 5000$ mm. From the point of view of precision, this uncertainty will be only about 1%.

3.2.4 Relative precision of the different motion components

The results obtained are congruent with intuition. Lateral camera translations T_x and T_y produce greater changes in pixels, so they are better recovered than the translation T_z along the optical axis. Rotations R_z about the optical axis cause large changes in the image, and are better recovered than the other two pure rotations, R_x and R_y. Estimated variances differ largely for the various



(a) (b)

Figure 6. Graphical representation of the 2×2 covariance submatrices by means of the 50% error ellipse. Small points are the projected outputs (recovered motions) of input samples after Monte Carlo simulation, a bigger point is drawn for the mean, and a dotted ellipse for the uncertainty. Superimposed are the results computed using the Unscented Transformation, which will be explained in Section 4.1. Crosses stand for the transformed sigma points and a continuous ellipse for the uncertainty.

motions. The largest errors and variances occur when the contour projection is not centered in the image, as weak-perspective assumptions are violated. If the distance to the target is small, more precision is attained due to increased resolution, but perspective effects appear. Small rotations out of the plane are badly estimated, but as the rotation increases the error and the variance diminish. Rotations in the plane are correctly recovered with small variance.

3.3 Covariance of the recovered motion

To obtain covariance estimates some experiments have been carried out, including null camera motion and, analogous to the preceding Section, motion along and about the 3 coordinate axes. An exhaustive account of the results can be found in Appendix A. Nearly all experiment outcomes are coincident in that no correlations appear between motion components. There are only two exceptions, which will be presented next.

Figure 6(a) corresponds to an experiment without any camera motion, where noise has been added to the contour control points. As can be observed, the recovered motion components R_x and T_z are correlated. This can be explained as follows. When the rotation is slightly overestimated the target projection should be smaller than it really is. To compensate for this effect, an underestimation of the camera translation is obtained where the camera is assumed to be closer to the target. When the recovered rotation value increases, the correlation also exists, but its effect is less visible as the rotation value is better recovered. This can be seen in Figure 6(b), which corresponds to an experiment where the camera has turned 10° with the coordinate system centered on the target. Here, since the rotation is larger, the translation is correctly recovered. The same observations are applicable to the R_y and T_z motion components.

The above cross relation explains the underestimation of the translation T_z presented before in Figure 4(e), which appears because, near the null rotation, R_x and R_y are overestimated.

Figure 7(a) shows the second source of error detected with the covariance propagation study. When a camera rotation is performed about the X axis, a slight translation is computed along the Y axis. This can be explained by analysing the projection process assumed in the weak-perspective camera model. When the camera rotates, some 3D target points become closer to the camera than others. Figure 7(b) illustrates this fact. For simplicity, it is easier to represent target rotation than camera rotation, but both situations are equivalent. Farther points project slightly closer to the projection axis than nearby points. Consequently, a small T_y translation is computed. Analogously, if the rotation is about the Y axis, the translation is then computed along the X axis. The maximum amount of translation occurs at $R_x = 45°$, but we can observe that these errors keep always very small (in real experiments we will see that these errors are under millimetric). This is not a correlation between variables, but an effect due to the differences between the weak-perspective model assumed and the perspective model really used to find the projections of the contour control points [2].

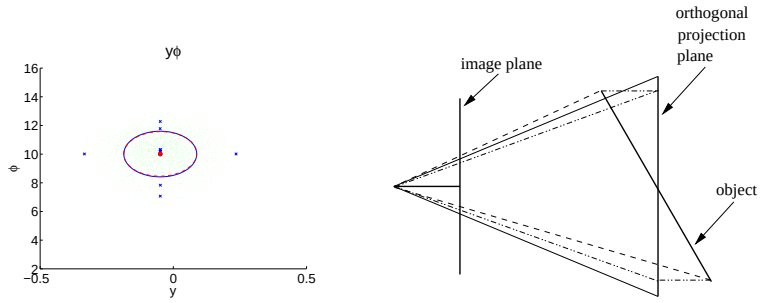
These are the only two correlations between recovered motion components detected in the covariance study performed using Monte Carlo simulation. As mentioned before, the complete set of experimental results, including all motions along and about the coordinate axes and all graphical representations of 2×2 covariance submatrices, can be found in Appendix A.

4 Experimentation with real images

The Monte Carlo simulations used before are a simple and powerful tool. However, they can only be applied when few results are needed or when the calculation time is not a restriction. We would like to find the uncertainty propagated to each camera pose at frame rate, which is 20 fps in our case. We cannot use Monte Carlo simulation since, for each pose, we should sample the entry space densely and execute the algorithm of motion estimation for each sample, which is obviously highly time consuming.

The alternative we have adopted, the so-called Unscented Transformation,

[2] If the initial distance is larger (less depth relief, as demanded by affine camera models) this effect is smaller, as the affine camera model fits better the perspective one, and negligible lateral translations are recovered, as will be seen in the Section on real experimentation.



(a) (b)

Figure 7. (a) Graphical representation of the 2×2 covariance submatrix by means of the 50% error ellipse relating rotation ϕ about the X axis and translation along the Y axis. Only a camera rotation is performed, but a small translation is also recovered. (b) Effect of rotating a planar object (which is equivalent to rotating a camera around a target). The weak-perspective model assumes that image projections should be where dotted rays meet the image plane, but they are really where the dashed lines are projected. Consequently, a lateral translation is recovered when only a rotation has been performed. Note that its amount depends on the initial distance.

attempts to find the minimum number of samples that represent the statistical distribution of the data, thus reducing considerably the computing time required for uncertainty propagation.

4.1 The Unscented Transformation (UT)

The Unscented Transformation (UT) was proposed by Julier and Uhlmann [28, 29]. It is a method for propagating the statistics of a general nonlinear transformation. The UT is not restricted to assuming that noise sources follow a Gaussian distribution. Briefly, it works as follows: first, a set of input points (called *sigma points*) is deterministically selected. This set of points is expected to capture the statistics of the input distribution. Second, the nonlinear transformation is applied to these points. Finally, the statistics of the transformed points are calculated to obtain the estimation of the output statistics for the given input distribution. Note that there is no need to calculate the partial derivatives of the nonlinear transformation.

UT has been often used whithin a Kalman Filter paradigm to perform the recursive prediction and state update, leading to the so-called Unscented Kalman Filter (UKF) [29]. The complexity of the resulting algorithm is the same as that of the EKF. Julier and Uhlmann [28] demonstrated the benefits of the UKF in the context of state-estimation for nonlinear control, and Wan and Van Der Merwe [30] showed its application in parameter estimation problems. They also developed a formulation where the square root of the covariance matrix is propagated instead of the matrix itself [31]. With this formulation,

the UT algorithm has better numerical properties (mainly in the UKF framework), and its complexity decreases for parameter estimation problems. An extension of the concept of the sigma points to work with particle filters and sums of Gaussians was also developed [32]. Lefebvre et al. [33] proposed an alternative interpretation of the UT as statistical linear regression, which is useful to justify the benefits over linearisation.

Different strategies for the choice of the sigma points have been developed. We will use the originally proposed solution, based on symmetric sigma points [29]. It consists of selecting $2N_x + 1$ sigma points, where N_x is the dimensionality of the input random variable. One sigma point is placed at the mean, and the others are placed at the mean plus or minus one standard deviation in each dimension. It can be seen as placing one sigma point at the center of each face of a hypercube. This is sometimes called the second-order UT, because it guarantees that the mean and covariance (the first two moments) are preserved through the transformation.

The N_x-dimensional random variable $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance matrix $\Sigma_{\mathbf{x}}$ is approximated by the set of points:

$$\begin{aligned} \mathbf{x}^0 &= \bar{\mathbf{x}} \\ \mathbf{x}^i &= \bar{\mathbf{x}} + \left(\sqrt{\frac{N_x}{1-w^0}\Sigma_{\mathbf{x}}}\right)_i \quad \text{for } i = 1, \ldots, N_x \\ \mathbf{x}^{i+N_x} &= \bar{\mathbf{x}} - \left(\sqrt{\frac{N_x}{1-w^0}\Sigma_{\mathbf{x}}}\right)_i \quad \text{for } i = 1, \ldots, N_x \end{aligned} \tag{22}$$

with the weights

$$\begin{aligned} w^0 & \\ w^i &= \frac{1-w^0}{2N_x} \text{ for } i = 1, \ldots, N_x \\ w^{i+N_x} &= \frac{1-w^0}{2N_x} \text{ for } i = 1, \ldots, N_x \end{aligned} \tag{23}$$

where $(\sqrt{N_x\Sigma_{\mathbf{x}}})_i$ is the ith row or column [3] of the matrix square root of $N_x\Sigma_{\mathbf{x}}$, and w^i is the weight associated with the ith sigma point. The weights must satisfy the condition $\sum w^i = 1$.

By convention the first sigma point $\mathbf{x}^0$ corresponds to the point located at the mean. The weight w^0 assigned to this point controls where the others will be

[3] Depending on how the matrix Σ is formed, columns or rows are used. If $\Sigma = AA^T$, then the sigma points are formed from the rows of A. However, if $\Sigma = A^TA$, the sigma points are formed from the columns of A.

located. If w^0 is positive, the remaining points tend to move farther from the origin, thus preserving the covariance value. If w^0 is negative, the points tend to be closer to the origin [34]. This is a fine tuning mechanism to approximate the higher-order moments of the distribution.

The mean and covariance of the output variable $\mathbf{y}$ can be estimated from the mapped sigma points $\mathbf{y}^i = f(\mathbf{x}^i)$ according to

$$\begin{aligned} \bar{\mathbf{y}} &= \sum_{i=0}^{2n} w^i \mathbf{y}^i \\ \Sigma_{\mathbf{y}} &= \sum_{i=0}^{2n} w^i \left\{\mathbf{y}^i - \bar{\mathbf{y}}\right\} \left\{\mathbf{y}^i - \bar{\mathbf{y}}\right\}^T . \end{aligned} \tag{24}$$

4.2 Using UT to estimate mean egomotion and covariance

In our egomotion estimation algorithm, the input space is the 6-dimensional shape vector (Eq. 5), which is transformed through equations 12-17 into the three translational and the three rotational motion components. To propagate covariances using UT, $2d+1 = 13$ symmetric sigma points are selected, $d = 6$ being the dimension of the input space.

The procedure to estimate covariances using UT is the following. First, an active contour is manually initialized by specifying some control points. This defines a shape matrix $\mathbf{W}$ according to Equation 5. The estimation algorithm, outlined in Algorithm 1, then proceeds as follows. In each iteration, one new image is acquired. A Kalman filter estimates the affine deformation of the current contour with respect to the initial contour, coded as a shape vector, as well as an associated covariance. Based on them, a UT is applied by selecting 13 sigma points in the shape space to which the nonlinear transformation is applied to find the estimated 3D motion and its estimated covariance.

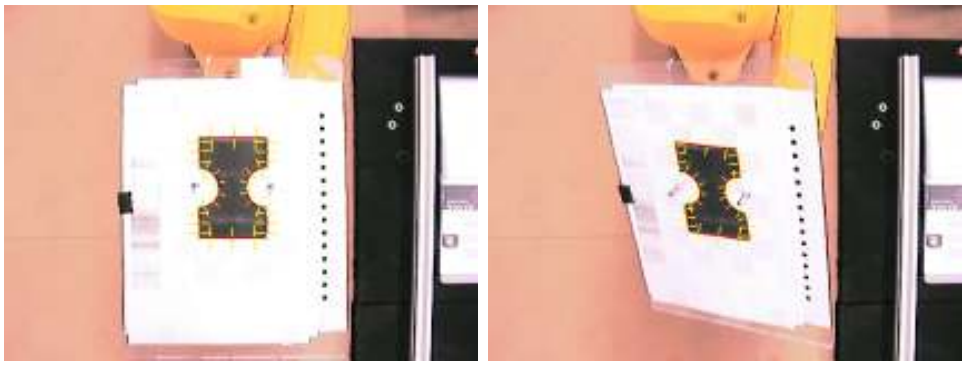
To validate the derivation of covariances using UT for our egomotion estimation algorithm, the same synthetic experiments reported for Monte Carlo simulation in Section 3.3 and Appendix A were carried out again using UT, and the results are included in Figures 6, 7, and A.1 to A.6. In all the experiments the covariance estimated with UT is very similar to the one obtained with Monte Carlo simulation and, therefore, we conclude that UT can be used to estimate the covariance of the recovered motion.

Two sets of experiments entailing real robot motions have been performed. In the first one, a target handled by a Stabli robot arm rotates in front of a still camera. In the second one, a camera is mounted on a Still EGV-10 forklift vehicle, which follows a slightly oscillating path towards a target, resulting in

Input: Inverse of the shape matrix $\mathbf{W}^{-1}$, initial control points $\mathbf{Q}_{template}$
Output: Egomotion $\mathbf{RT}$ and covariance $\mathbf{\Sigma_{RT}}$
Acquire a new image;
Iterating the Kalman filter, estimate the control point locations $\mathbf{Q}$ of the projected contour, and estimate the corresponding shape vector using $\mathbf{S} = \mathbf{W}^{-1}(\mathbf{Q} - \mathbf{Q}_{template})$;
Find the 13 symmetric sigma points $\mathbf{x}^i$ in shape space (Eq. 22) and their corresponding weights w^i (Eq. 23) from the shape vector $\mathbf{S}$ and the covariance $\mathbf{\Sigma_S}$ estimated by the Kalman filter;
for *i=1* **to** *13* **do**
    Calculate the recovered motion corresponding to the sigma point $\mathbf{x}^i$;
end
Calculate the egomotion $\mathbf{RT}$ and the covariance $\mathbf{\Sigma_{RT}}$ by applying Equation (24);

Algorithm 1: One iteration of the egomotion and covariance estimation algorithm using UT.

large translational motion.



(a) $frame_0$ (b) $frame_{80}$

Figure 8. Initial image and maximally rotated image for the rotation experiment.

4.2.1 Rotation experiments using a robot arm

In the first experiment, we estimate motion uncertainty in a setup that we used before [27]. A Staübli robot arm handling an artificial target rotates it 40° (stopping at each 10°) about an axis placed on the target at 45°, and then returns to the initial position. Figure 8 shows the target at the initial position and at the point of maximum inclination. Observe that this is equivalent to rotating the camera −45° about the same axis. In Figure 9 the evolution of the 6 motion components along the sequence of acquired frames is plotted, showing both the UT estimation and the transformation of the shape vector (named as "direct").

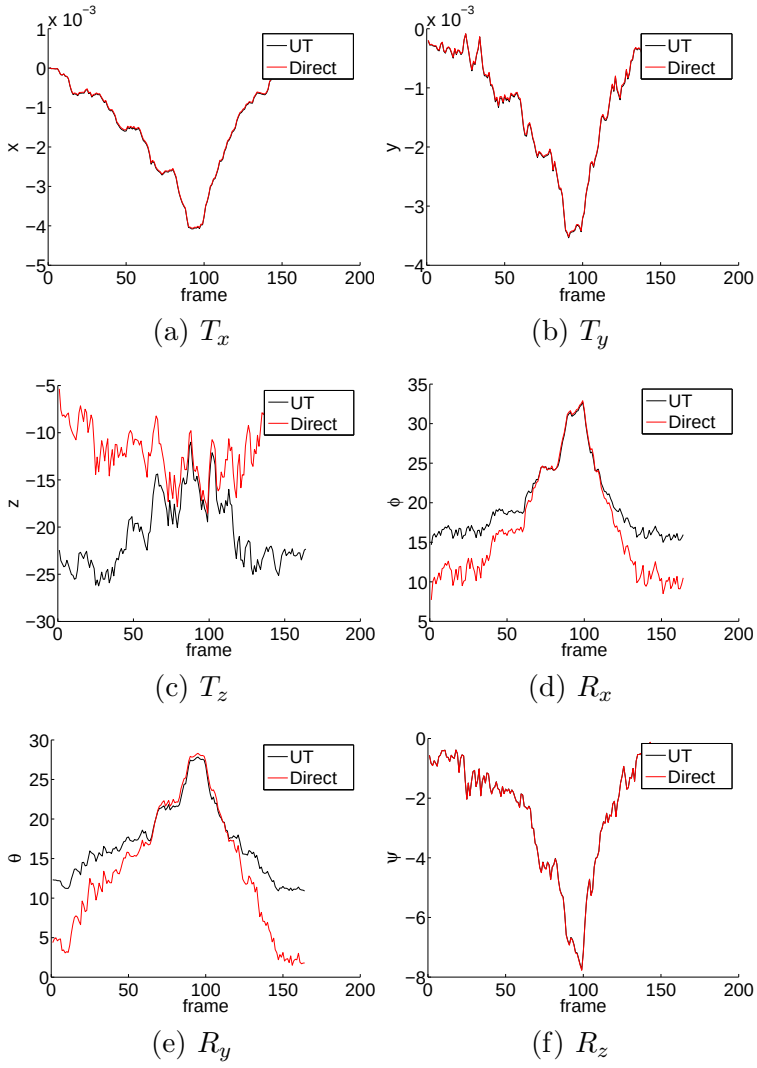
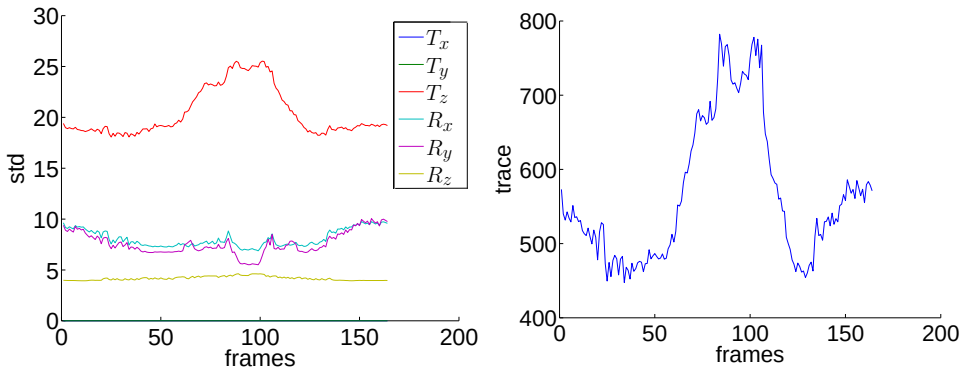


Figure 9. Component motions recovered in the real rotation experiment, consisting of a rotation of 40° about a 45° inclined axis placed frontoparallel to the camera and centered on the object, and later a rotation of −40° about the same axis. In red the results obtained with the original algorithm, and in black the values of the mean calculated with the algorithm using UT.

Congruent with synthetic results (see Fig. 7(b), where a slight T_y translation is recovered when only R_x rotation is performed), small T_x and T_y translations (Fig. 9(a) and 9(b)) are recovered while they are not really performed. Thanks to the calibration performed, these results can be expressed in millimeters and we can conclude that the computed translation errors (of at most 0.004 mm) are negligible. As expected, T_z translations are recovered with more error. The calibration process estimated an initial distance from camera to target of $Z_0 = 500$ mm, so the precision in the recovery of this translation is between 1% and 3%, which is also concordant with simulation results.

Rotations R_x and R_y are not correctly recovered under 15°, because of noise



(a) deviations (b) trace

Figure 10. (a) Standard deviations computed with UT for the real rotation experiment, and (b) trace of the resulting covariance matrix.

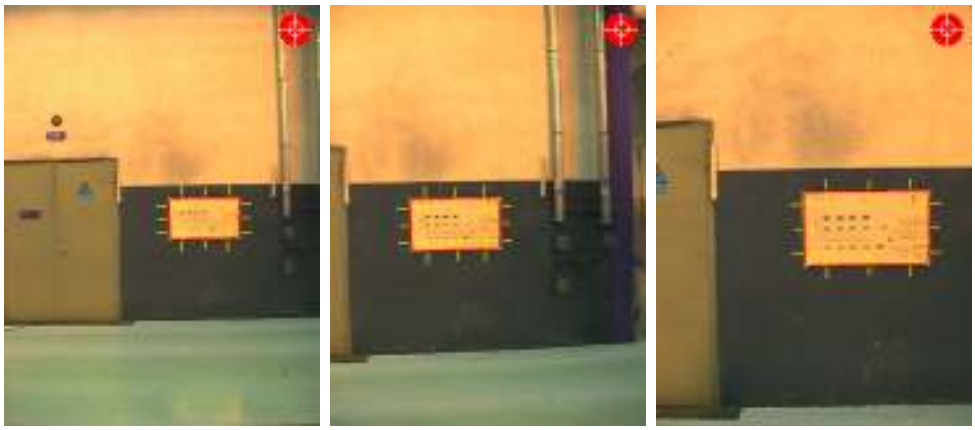
in the acquisition and tracking processes. The bias due to the rotation representation contributes also to this initial error. Between frames 50 and 100 we can observe clearly the pauses at every 10°, and how the rotation computed with the UT and the one computed directly from the shape vector coincide. For the R_z rotation, they coincide along the whole sequence.

In Figure 10(a) the standard deviations estimated for the whole motion sequence are shown. T_x and T_y deviations are nearly null. The most important deviation is obtained for the T_z component. Note that the deviation increases in the middle of the plot, where rotation is larger. This is due to the perspective effect explained before in Figure 7(b). It can be observed that the deviations for R_x and R_y components slightly diminish when rotation values increase, and return to the initial values when the target is rotated backwards to the initial position, where the null rotation should be recovered. As expected, due to the correlation between rotations R_x, R_y and translation T_z, the uncertainty in T_z slightly diminishes also.

Figure 10(b) plots the trace of the covariance matrix. Since it is a rough estimation of the covariance size [35], it will serve us here to illustrate the global uncertainty behavior. In our application, the trace is heavily influenced by the uncertainty in the T_z component. The global uncertainty decreases in the first part of the sequence, when the rotations are better estimated, but in the middle of the sequence the global uncertainty increases because of the uncertainty in T_z.

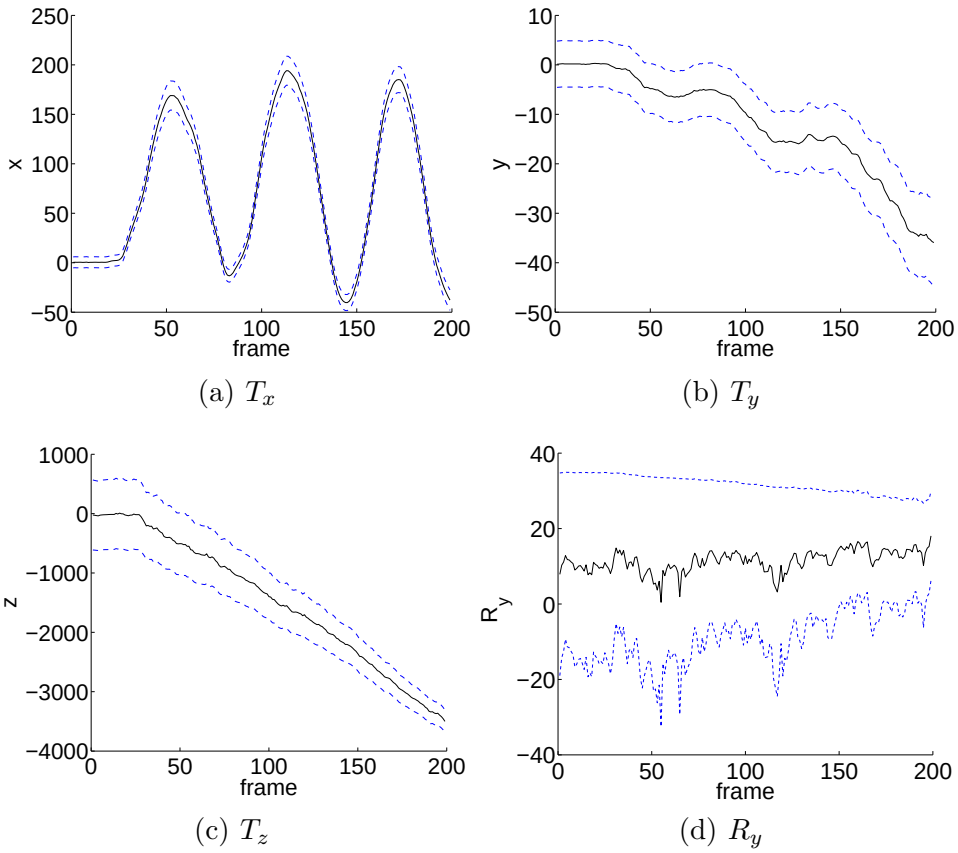
4.2.2 Long-range translation experiment using an autonomous vehicle

The second experiment uses the data that we collected in an experience performed in a real factory environment. The robotized forklift vehicle was equipped with a positioning laser, whose recordings were used as ground truth



(a) $frame_{first}$ (b) $frame_{middle}$ (c) $frame_{last}$

Figure 11. Real experiment entailing a large slightly-oscillatory translation.

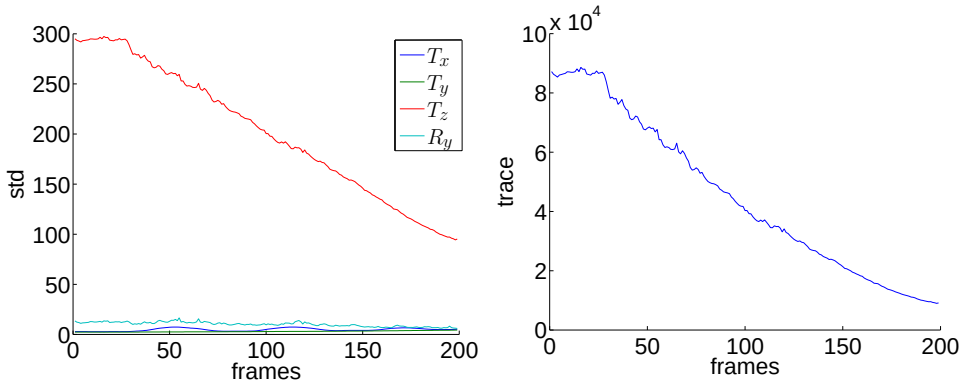


(a) T_x (b) T_y

(c) T_z (d) R_y

Figure 12. Recovered component motions for the experiment with a forklift vehicle. The continuous line represents the motion value and the dashed lines are the 2σ bounds for the whole sequence.

to evaluate the visual egomotion estimations. In order to obtain metric results, we calibrated the camera and estimated the initial distance with the laser. An information board was selected as target, and its initial distance was estimated to be 7700 mm. An approach of about 3500 mm with a slight lateral oscillation



(a) deviations (b) trace

Figure 13. (a) Standard deviations computed with UT, and (b) trace plot.

was performed and three of the acquired frames are displayed in Figure 11: at the initial position, at half sequence and at the final position. The analysis of the recovered motion results was presented in [36], where a reduced 4D shape space was used. Here we will calculate the covariance for each estimation of 4-degrees-of-freedom motion.

Figure 12 shows the obtained results. Observe that the uncertainty in the T_x component increases when the distance of the tracked contour to the image center also increases. This was explained in Section 3.2.1 and it is due to the non satisfaction of the affine camera model assumptions. A translation in the Y direction is also obtained. As explained in [36], this is due to misalignments between the camera and the robot reference frames. Like for the T_x component, uncertainty increases when the distance of the tracked contour to the image center increases, but in this case values are smaller and this effect is not easily seen. As expected, uncertainty in the translation T_z diminishes as the robot gets closer to the target. Unfortunately, in this experiment, rotations R_y were very small and it was not possible to recover them. Consequently, the uncertainty estimated for this component is very large (see Fig. 12(d)).

We plot the standard deviations of the motion components and the trace of the covariance matrix in Figure 13. As expected, the trace is dominated by the T_z uncertainty. Observe how standard deviation of the T_z component varies with distance. Compared to the previous experiment, where the initial distance was 500 mm and standard deviation was between 20 and 25, the accuracy in the estimation of T_z and its deviation are similar, since the deviation values are between 300 and 100 for distances from 7700 mm to 3500 mm.

Summarizing, in this experimental Section we have shown that uncertainty in egomotion recovery can be estimated at frame rate. By using an implementation that exploits this capability, we have tested the precision of our approach in practice, leading to the conclusion that an error of around 3% is obtained for long-range trajectories. Thus, the approach seems promising to be used for

transferring operatins (since it doesn't require pre-setting the environment), in combination with more precise laser positioning for loading and unloading the forklift vehicle [36].

5 Conclusions and future prospects

A method for estimating robot egomotion has been presented, which relies on real-time contour tracking in images acquired with a monocular vision system. Contour deformation due to camera motion is codified as a 6-dimensional affine shape vector, from which the 6 degrees of freedom of 3D motion are recovered.

The precision of the algorithm has been analyzed through Monte Carlo simulation, and the results obtained are congruent with intuition. Lateral camera translations T_x and T_y produce greater changes in pixels, so they are better recovered than the translation T_z along the optical axis. Rotations R_z about the projection axis cause large changes in the image, and are better recovered than the other two pure rotations, R_x and R_y. Estimated variances differ largely for the various motions. The largest errors and variances occur when the contour projection is uncentered in the image, as weak-perspective assumptions are violated. If the distance to the target is small, more precision is attained, but perspective effects appear. Small rotations out of the plane are badly estimated, but as the rotation increases the error and the variance diminish. Rotations in the plane are correctly recovered with small variance.

The Unscented Transformation has been used in real experiments to compute the uncertainty of the estimated robot motion. A real-time implementation of the tracking, egomotion and uncertainty estimation algorithms has been accomplished. A Staübli robotic arm has been used to assess the performance of the approach when facing large rotations. A second set of real experiments, carried out in a brewer warehouse with a forklift vehicle, has been used to validate the motion estimation algorithm in the case of long-range translations. Contrarily to laser estimation procedures, a natural landmark was used and no previous intervention on the environment was needed. In a previous work [36] we calculated the error in motion recovery for this experiment. Here, with the uncertainty estimation algorithm, we have obtained a relative small standard deviation (about 3%) in the most uncertain robot motion component namely T_z which leaves the real translation value within the statistical predicted margins. This supports vision-based egomotion estimation as a promising alternative in situations with relatively low-precision demands.

Future work is clearly oriented by the conclusions reached in this work. On the one hand, synthetic experiments suggest that the target should be centered in the image to keep the weak-perspective assumptions and attain more precision.

On the other hand, real experiments show that the range of applicability of the proposed algorithm is limited as the contour should be kept within the image all along the sequence. One solution is to switch from one target contour to another when the former disappears from the image. Another solution, which we will explore in future work, is to keep the target into the image with the use of a pan-and-tilt camera. This will permit larger robot motions with smaller uncertainty.

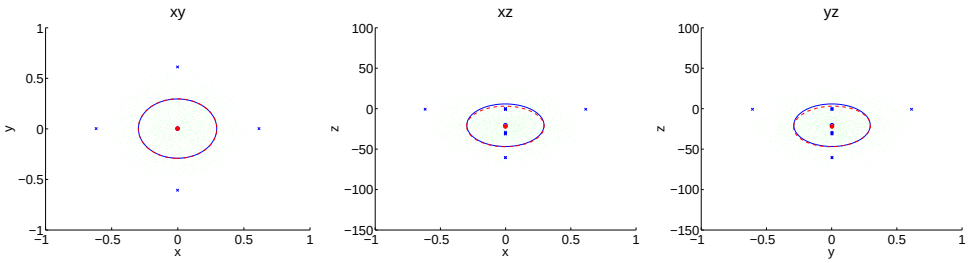
We have also noticed that the size of the target projection in the image should be kept within reasonable margins to be able to track and deduce valid information. For this reason, the approaching translations in the experiments in the warehouse were of at most 5 meters. This is also a limitation. We are currently exploring the use of a zooming camera to maintain the size of the target projection onto the image constant. This presents some challenges as changing the zoom complicates the pan-and-tilt control, since depending on the initial distance (which we assume unknown), different control gains should be applied.

A Complete covariance results

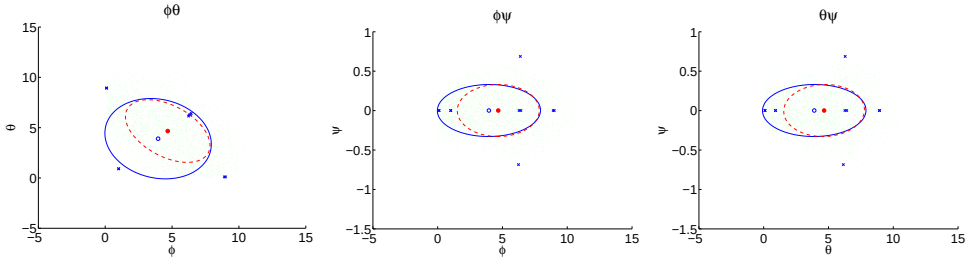
As mentioned in Section 3.3, our goal was to evaluate the covariances resulting from the egomotion recovery algorithm so as to identify possible correlations between motion components. As in the experiments in that Section, Gaussian noise with zero mean and $\sigma = 0.5$ is added to the projected target. To visualize the obtained 6×6 covariance matrices, we represent the motion components pairwise on $2D$ planes and we draw the mean value and the 50% error ellipse. Note that to represent all the possible 2×2 submatrices we need 15 $2D$ combinations. We will also compare Monte Carlo and UT results. The Unscented Transformation uses the covariance obtained in shape space by the Monte Carlo simulation to select the sigma points with the symmetric schema described in Section 4.1.

A.1 Perturbing the contour at the initial position

The first experiment in this Section is performed around the initial position, without any camera motion. Figure A.1(a) shows that no correlations between translation components can be observed. Coherently with results in Section 3.2.1 (Fig. 4), we can see that as the target projection is centered in the image, the T_x and T_y components are precisely recovered with $\overline{x} \approx 0$ and $\overline{y} \approx 0$ and small error. As expected, the computation of the depth translation T_z is less precise. The estimation of the statistics with the Unscented



(a) Covariance between translation components.



(b) Covariance between rotation components.

Figure A.1. Representation of 2×2 covariance submatrices for perturbations around the initial contour position: (a) corresponds to covariances between translation components, and (b) covariances between rotations components. Monte Carlo results (green points for sample projections, filled circle and dotted ellipse for computed mean and covariance), and UT results (crosses for sigma points, empty circle and solid ellipse for mean and covariance) are shown.

Transformation is close to the one obtained with the Monte Carlo simulation.

Figure A.1(b) shows the correlations between the rotation components. As mentioned before in Section 3.1, due to the rotation representation used, the correct values in the R_x and R_y axes when these values are nearly zero cannot be recovered. A bias is introduced near the null rotation in each of these axes, and it has the effect of creating a straight border in the plot for the Monte Carlo simulation, because no negative points are allowed. The R_z rotation is correctly recovered. The covariance estimations obtained with the UT do not exactly match those obtained with the Monte Carlo algorithm due to the mentioned bias, so we should verify later if the UT approximation is valid here.

In Figure A.2 the remaining 2×2 covariance matrices are represented, all involving a translation and a rotation component. We can see the bias effect explained for the zero rotations about X and Y in all figures involving these rotations. As before, the remaining cases are correctly estimated by the Unscented Transformation.

A cross relation can be observed between the estimation of the rotation ϕ about the X axis and the translation along the Z direction, which has been

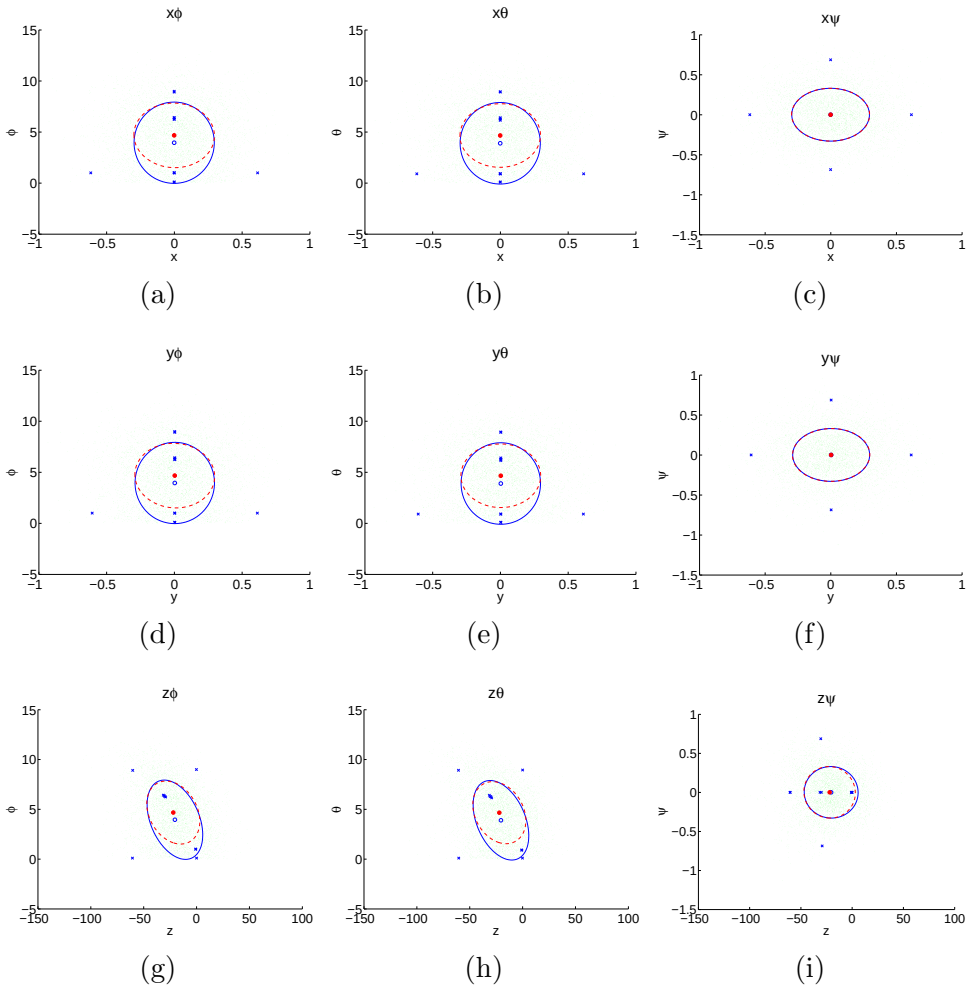


Figure A.2. Representation of 2×2 covariance submatrices of translations and rotations for perturbations around the initial contour position. Monte Carlo results (green points for sample projections, filled circle and dotted ellipse for computed mean and covariance), and UT results (crosses for sigma points, empty circle and solid ellipse for mean and covariance) are shown.

explained before in Section 3.3. A similar cross relation appears between the rotation θ about the Y axis and T_z.

A.2 Perturbing the contour after a single component motion

Now we would like to estimate the covariances in the case of camera motion along and about the coordinate axes. Significant information appears in the axis of motion chosen in each experiment so, from the 15 2×2 submatrices, we will plot only the 6 submatrices involving this axis.

First a translation of 300 mm along the X axis is performed. We can see in

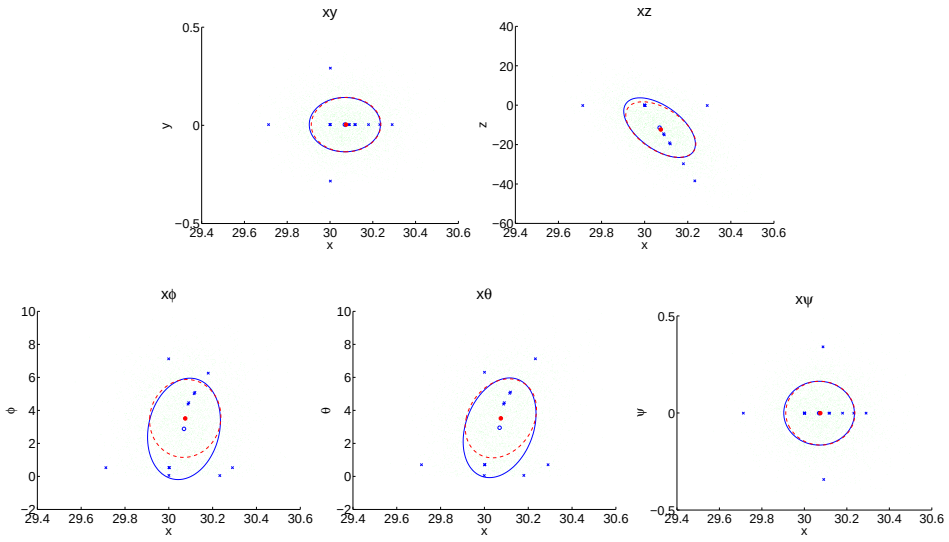


Figure A.3. Representation of 2×2 covariance submatrices of translations and rotations for perturbations around $T_x = 30$ mm. Monte Carlo results (green points for sample projections, filled circle and dotted ellipse for computed mean and covariance), and UT results (crosses for sigma points, empty circle and solid ellipse for statistics) are shown. Results are similar for a $T_y = 30$ mm translation.

Figure A.3 the values obtained. The uncertainty is congruent with the values presented in Figure A.1(a). The translation T_x is correctly recovered with mean $\bar{x} \approx 0$ and small uncertainty. The $T_x - T_z$ plot seems to show a correlation between both variables, but observing the scale of the figure we can deduce that the correlation is spurious. The plots with the R_x and R_y components show the bias effect described in the preceding Section, so no conclusion can be extracted from them. T_x and R_z have no correlation, as can be seen in the last plot. Error of the UT estimation with respect to the Monte Carlo is negligible, except in the R_x and R_y dimensions (this effect can also be observed in the previous Figures A.1(b) and A.2).

The results and considerations presented for T_x motion are also valid in the case of a T_y translation.

When the translation is performed along the Z axis (Fig. A.4), the T_x and T_y translations are correctly estimated, and their uncertainty keeps small, but the value T_z is underestimated. As expected, the bias in the R_x and R_y rotation values is present, as they are close to zero. The correlation observed in the null motion experiment between $T_z - R_x$ and $T_z - R_y$ is also observed here. The Unscented Transformation again fits correctly the Monte Carlo mean and covariance.

The next experiment is performed orbiting the camera around the target about the X axis (Fig. A.5). Congruent with the previous results (compare with Fig. 4(b) when rotation is about 30°), the uncertainty in the recovered R_x

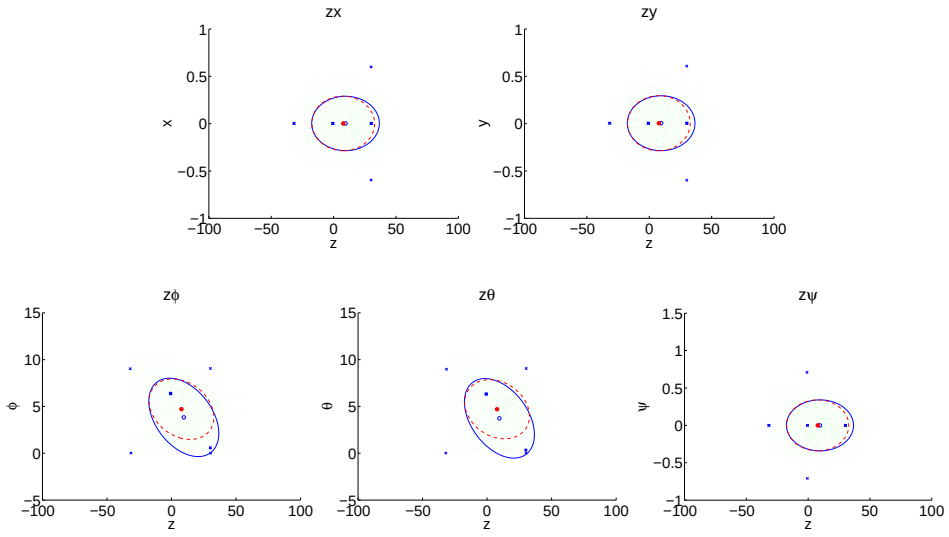


Figure A.4. Representation of 2×2 covariance submatrices of translations and rotations for perturbations around $T_z = 30$ mm. Monte Carlo results (green points for sample projections, filled circle and dotted ellipse for computed mean and covariance), and UT results (crosses for sigma points, empty circle and solid ellipse for statistics) are shown.

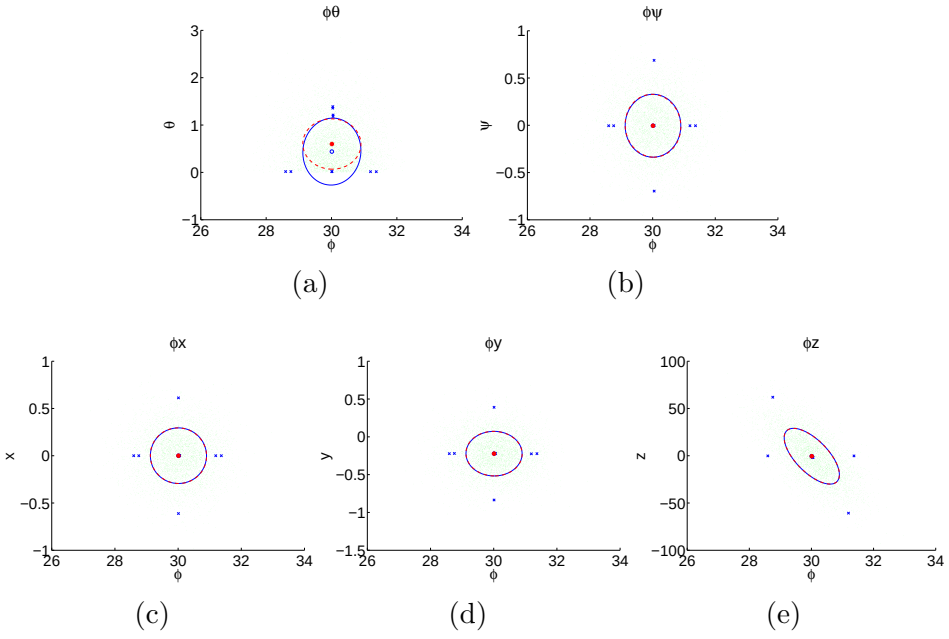


Figure A.5. Representation of 2×2 covariance submatrices of translations and rotations for perturbations around $R_x = 30°$ motion. Monte Carlo results (green points for sample projections, filled circle and dotted ellipse for computed mean and covariance), and UT results (crosses for sigma points, empty circle and solid ellipse for statistics) are shown. Results are similar for a $R_y = 30°$ rotation.

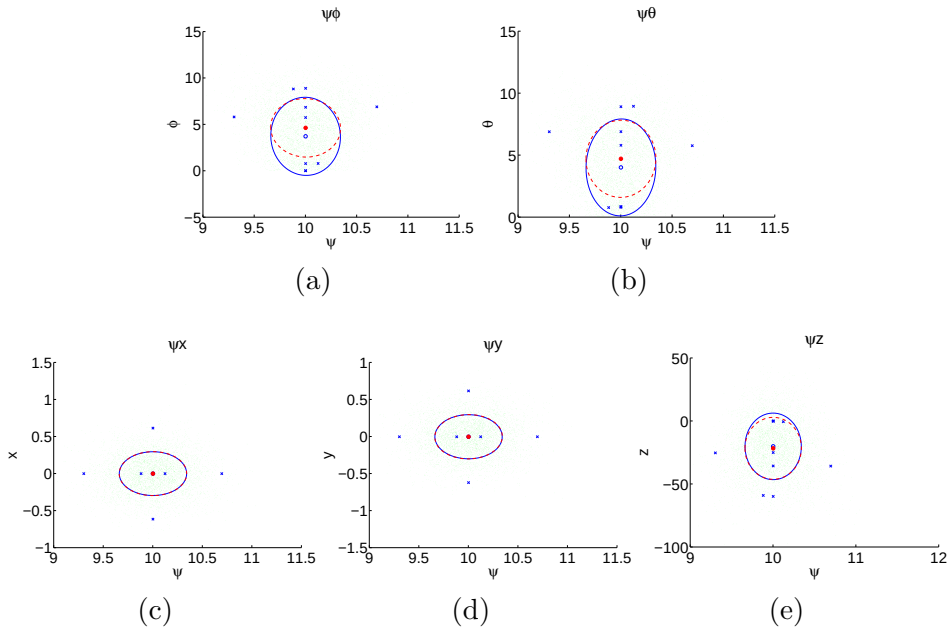


Figure A.6. Representation of 2×2 covariance submatrices of translations and rotations for perturbations around $R_z = 10°$ motion. Monte Carlo results (green points for sample projections, filled circle and dotted ellipse for computed mean and covariance), and UT results (crosses for sigma points, empty circle and solid ellipse for statistics) are shown.

values is smaller than that in the experiments with no rotation (Fig. A.1). The bias in the R_y component is present, and R_z is also well recovered and uncertainty values have not been altered. With respect to the translations, T_x is correctly recovered but T_y is estimated with a slight error. We have explained this effect in Section 3.3 by analysing the projection process (Fig. 7(b)).

The last plot in Figure A.5 shows the correlation between R_x and T_z explained before in Section 3.3. Observe that the bias in T_z introduced when rotations R_x and R_y are overestimated is not present here. This is because large rotations are better estimated and, accordingly, there is no need of T_z to be underestimated to compensate for the errors. As before, the estimation of the statistics with the Unscented Transformation is close to the one obtained with the Monte Carlo simulation.

The considerations above are also applicable to rotations about the R_y axis.

In the last experiment the camera is rotated about the optical axis Z. Refer to Figure A.6. As expected, the value of this rotation is precisely recovered, and the error keeps small. No correlation with R_x or R_y rotations is observed, but the typical bias in these variables is also present, inducing the explained bias in T_z. Translations are recovered as in previous experiments, and no correlations with R_z appear.

In all the experiments we have shown that covariance estimated with UT is very similar to that obtained with Monte Carlo simulation. We can conclude that UT can be used to estimate the covariance of the obtained pose. We have confirmed the correlation between T_z and R_x or R_y variables, and the small translations recovered when some rotations are performed.

References

[1] H. Longuet-Higgins, A computer program for reconstructing a scene from two projections, Nature 293 (11) (1981) 133–135.

[2] R. I. Hartley, In defense of the eight-point algorithm, IEEE Trans. Pattern Anal. Machine Intell. 19 (6) (1997) 580–593.

[3] Y. Liu, T. Huang, O. Faugeras, Determination of camera location from 2d to 3d lines and point correspondences, IEEE Trans. Pattern Anal. Machine Intell. 12 (1) (1990) 28–37.

[4] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, 2nd Edition, Cambridge University Press, 2004.

[5] Z. Zhang, Determining the epipolar geometry and its uncertainty: a review, Int. J. Comput. Vision 27 (2) (1998) 161–195.

[6] M. Fischler, R. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Comm. ACM 24 (1981) 381–385.

[7] P. Torr, D. Murray, Outlier detection and motion segmentation, Sensor Fusion VI, SPIE 2059 (1993) 432–443.

[8] J. Clarke, Modelling uncertainty: A primer, Tech. Rep. 2161/98, University of Oxford. Dept. Engineering science (1998).

[9] A. Criminisi, Accurate visual metrology from single and multiple uncalibrated images., Springer, 2001.

[10] D. D. Morris, K. Kanatani, T. Kanade, Uncertainty modeling for optimal structure from motion, in: Vision Algorithms Theory and Practice, Springer LNCS, 1999.

[11] W. Chojnacki, M. J. Brooks, A. van den Hengel, D. Gawley, On the fitting of surfaces to data with covariances, IEEE Trans. Pattern Anal. Machine Intell. 22 (11) (2000) 1294–1303.

[12] M. Brooks, W. Chojnacki, D. Gawley, A. van den Hengel, What value covariance information in estimating vision parameters?, in: International Conference on Computer Vision, 2001, pp. 302–308.

[13] Y. Kanazawa, K. Kanatani, Do we really have to consider covariance matrices for image features?, in: Proc. IEEE Int. Conf. Comput. Vision, Vancouver, BC, Canada, 2001, pp. 301–306.

[14] N. Gonalves, H. Arajo, Analysis and comparison of two methods for the estimation of 3d motion parameters., Robotics and Autonomous Systems 45 (1) (2003) 23–50.

[15] T. Papadopoulo, I. A. Loukakis, Estimating the jacobian of the svd: theory and applications, Tech. Rep. RR-3961, INRIA (2000).
[16] J. Weng, T. S. Huang, N. Ahuja, Motion and structure from image sequences, Springer-Verlag, 1993.
[17] A. Doucet, N. de Freitas, N. Gordon (Eds.), Sequential Monte Carlo methods in practice, Springer, 2001.
[18] E. Martínez, C. Torras, Qualitative vision for the guidance of legged robots in unstructured environtments, Pattern Recognition 34 (2001) 1585–1599.
[19] G. Alenyà, E. Martínez, C. Torras, Fusing visual and inertial sensing to recover robot egomotion, Journal of Robotic Systems 21 (1) (2004) 23–32.
[20] B. Tordoff, D. Murray, Reactive control of zoom while fixating using perspective and affine cameras, IEEE Trans. Pattern Anal. Machine Intell. 26 (1) (2004) 98–112.
[21] A. Blake, M. Isard, Active contours, Springer, 1998.
[22] T. Drummond, R. Cipolla, Application of lie algebras to visual servoing, Int. J. Comput. Vision 37 (1) (2000) 21–41.
[23] J. Koenderink, A. J. van Doorn, Affine structure from motion, J. Opt. Soc. Am. A 8 (2) (1991) 377–385.
[24] L. S. Shapiro, A. Zisserman, M. Brady, 3D motion recovery via affine epipolar geometry, Int. J. Comput. Vision 16 (2) (1995) 147–182.
[25] J. Foley, A. van Dam, S. Feiner, F. Hughes, Computer Graphics. Principles and Practice, Addison-Wesley Publishing Company, 1996.
[26] L. Sciavicco, B. Siciliano, Modeling and Control of Robot Manipulators, Springer-Verlag, London, 2000.
[27] M. Alberich-Carramiñana, G. Alenyà, J. Andrade-Cetto, E. Martínez, C. Torras, Recovering Epipolar Direction from Two Affine Views of a Planar Object, Comp. Vis. and Image Und. 122 (2) (2008) 195–209.
[28] S. J. Julier, J. K. Uhlmann, Unscented filtering and nonlinear estimation, Proc. IEEE 92 (3) (2004) 401–422.
[29] S. J. Julier, J. K. Uhlmann, A new extension of the Kalman filter to nonlinear systems, in: I. Kadar (Ed.), Proc. 11th SPIE Int. Sym. Aerospace/Defense Sensing, Simulation, Controls, SPIE, Orlando, 1997, pp. 182–193.
[30] E. Wan, R. Van Der Merwe, The unscented kalman filter for nonlinear estimation, in: Adaptive Systems for Signal Processing, Communications, and Control Symposium, 2000, pp. 153–158.
[31] R. van der Merwe, E. Wan, The square-root unscented kalman filter for state and parameter-estimation, in: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Salt Lake City, Utah, 2001.
[32] R. van der Merwe, E. Wan, Gaussian mixture sigma-point particle filters for sequential probabilistic inference in dynamic state-space models, in: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Hong Kong, 2003.

[33] T. Lefebvre, H. Bruyninckx, J. De Schutter, Comment on " A new method for the nonlinear transformation of means and covariances in filters and estimators", IEEE Trans. Automat. Contr. 47 (8) (2002) 1406 –1408.

[34] S. Julier, J. Uhlmann, H. F. Durrant-Whyte, A new method for the nonlinear transformation of means and covariances in filters and estimators, IEEE Trans. Automat. Contr. 45 (3) (2000) 477–482.

[35] R. Sim, N. Roy, Global A-optimal robot exploration in SLAM, in: Proc. IEEE Int. Conf. Robot. Automat., Barcelona, 2005, pp. 673–678.

[36] G. Alenyà, J. Escoda, A.B.Martínez, C. Torras, Using laser and vision to locate a robot in an industrial environment: A practical experience, in: Proc. IEEE Int. Conf. Robot. Automat., Barcelona, 2005, pp. 3539–3544.