A two-stage robust statistical method for temporal registration from features of various type

G. Simon and M.-O. Berger
Crin-Cnrs & Inria Lorraine, Bâtiment Loria, BP 239,
54506 Vandœuvre-lès-Nancy cedex, FRANCE
{gsimon,berger}@loria.fr

Abstract

A model registration system capable of tracking an object, the model of which is known, in an image sequence is presented. It integrates tracking, pose determination and updating of the visible features. The heart of our system is the pose computation method, which handles various features (points, lines and free-form curves) in a very robust way and is able to give a correct estimate of the pose even when tracking errors occur. The reliability of the system is shown on an augmented reality project.

1 Introduction

Our aim is to build a model registration system capable of tracking an object, the model of which is known, in an image sequence. Such systems are of great interest for augmented reality applications, in particular when virtual objects must be overlaid on an image sequence or when an object in the scene must be replaced with another one [2, 17, 21].

The registration system must be able to automatically track features in the images and to compute the pose from their correspondences with the 3D model. Since the world around us is not piecewise planar, significant features that can be extracted in an image are often curved contours, especially if we consider outdoor environments. Thus, the features we consider for pose computation are points, lines and curves.

It is generally assumed that correspondences are maintained during tracking. Unfortunately, tracking errors will sometimes result in a model feature being matched to an erroneous image feature. Even a single such outlier can have a large effect on the resulting pose. For point features, robust approaches allow the point to be categorized as outlier or not [10]. When curved features are considered, the problem is not so simple, as some parts of the 2D curves can perfectly match the 3D model whereas other parts can be erroneously matched. To make real progress, it is then necessary to devise a robust algorithm capable of extracting the parts of the features that match the 3D model.

Besides the pose computation problem, another problem must be solved for maintaining registration over time: as the camera undergoes motion with respect to the object, new features may appear while old ones disappear. Hence, the set of model features that are tracked in the sequence must be dynamically updated.

1.1 Previous Work

Computing the pose from 3D-2D correspondences often consists in the following steps. First, matching hypotheses are generated. Then, the pose is computed using these correspondences. In the tracking context, a rough estimate of the pose is available (the pose computed for the previous frame). Hence, the correspondent can be searched for in a limited neighbourhood of the projected feature.

Since a single outlier can have a large effect on the resulting pose, special care is often taken in the matching process, to reduce possible false matches. For instance, Lowe [15] uses a probabilistic criterion to guide the search for correct correspondences, before using a weighted least squares including *a priori* constraints for stabilization. Other methods [9, 12] use a velocity model and a Kalman filter to better predict the position of the image feature. Unfortunately the use of a velocity model imposes regularity constraints on the camera motion; this can be inappropriate for augmented reality applications for which the scene is often shot by a moving observer.

We therefore advocate a less constraining approach. Instead of attempting to refine the matching process, we prefer to use a robust statistical method to compute the pose from the matching induced by the tracking process. Our approach has some common points with [17]. They also use robust methods for pose computation after the tracking stage. But they only consider point features, whereas we consider points, lines

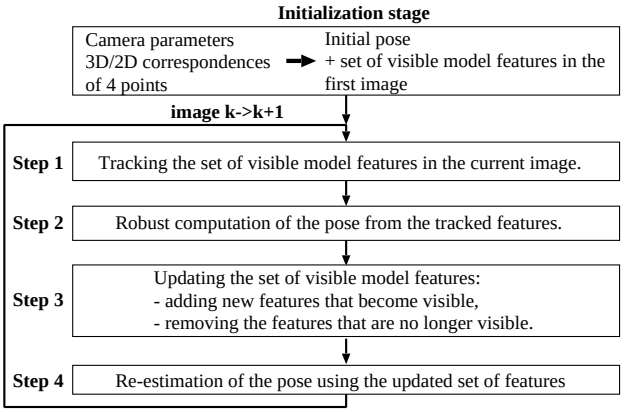

Figure 1: The temporal registration system.

or curved features. Moreover, their tracking strategy based on correlation cannot be extended to complex features.

2 System Overview

Our temporal registration system is initialized with known camera parameters and a user specified set of 3D features that give rise to easily detectable 2D features. At the beginning of the process, the pose in first image is computed from four 3D-2D corresponding points pointed out by the user (with method of Dementhon and Davis [4]), and the 2D features corresponding to the visible model features are automatically determined by the updating algorithm presented below. Once initialized, the system follows a four step loop (Figure 1):

Step 1: Tracking the features. The set of features is tracked in the current image using a curve-based tracker that we have previously developed [1]. We have shown that this method is reliable, fast and that it is able to account for sufficiently large motions. Nevertheless, the tracking may sometimes fail if the snake is attracted by some local edge maxima: this often results in a curve that is only partially well localized (see features 1 and 4 on Figure 3.d). If the prediction step failed (because of the instability of the flow computation), the position reached is generally completely erroneous (see feature 5 on Figure 3.d). Hence, among the set of tracked curves, a small number may be mis-detected or completely erroneous.

Step 2: Computing the pose. This step is the heart of our system. It allows the pose to be computed from various features despite possible tracking errors. We use some kind of *Iterative Closest Point* (ICP) algorithm to achieve this task (see section 3). This algorithm stands out from previous works by the use of robust estimators in a two-stage process: a *local stage*, which computes a robust residual for each feature, and a *global stage*, which minimizes a robust function of these residuals. The interest of our algorithm is two-fold: first it computes the pose in a robust manner, second it allows us to detect and discard the erroneous features. The choice of the robust estimators that are used for the local and the global stage is discussed in section 3.

Step 3: Updating the visible model features. When the camera undergoes large motion, it becomes essential to update the set $\mathcal{S}$ of tracked features; otherwise, the number of features that are visible decreases (even becomes null) and the pose cannot be computed with sufficient reliability. Given a robustly computed pose, we are able to determine which part of the model features projects in the image. When a new feature appears, we have to determine the corresponding 2D feature that will be tracked in the subsequent frames. To perform this task, we first realize a classical Canny edge detection [3] and consider all the contours that are sufficiently close to the projection of the model feature. For each contour c, we compute the pose corresponding to $\mathcal{S} \cup c$ using the pose algorithm of step 2. As a result of our algorithm, we can see if the contour c has been discarded from the computation. If not, this means that c is likely to be the projection of the model feature and is added to $\mathcal{S}$ (if more than one contour is retained, we try to concatenate them and keep the longest obtained contour). Precisions about this method can be found in [20]. It is also used to update the features that are discarded in step 2.

Step 4: Re-estimation of the pose. Finally, the pose is re-estimated by taking into account the new set of model features.

3 Pose Computation

Most pose estimation algorithms use simple features: points [10, 4], lines [5, 19, 14] or circles [8]. But only few papers have been devoted to the 3D-2D registration of curves. Kriegman [13] proposed an algebraic method to compute the pose of a curved object from the observation of its occluding contours (it can be easily applied to perform 3D-2D registration). Unfortunately, this method only deals with surfaces or curves that can be described by a collection of parametric patches (ratio of polynomials). Moreover, the use of elimination theory to compute the pose turns out to be very expensive. The 3D-2D registration problem has also been considered in [7], for medical tasks. Starting from a gross estimation of the pose, an ICP algorithm is used to perform registration. An

extended Kalman filter is used to discard outliers by performing χ^2 tests. Nevertheless, since the results depend on the order of processing of the measurements, the estimation process may be trapped into a local minimum, especially if the initial estimate is not very accurate. Such a process is therefore not really robust against gross errors.

3.1 The Problem

The problem consists in finding the rotation $\mathbf{R}$ and the translation $\mathbf{t}$ which map the world coordinate system to the camera coordinate system. Therefore, if the intrinsic parameters of the camera are known (they can be determined by a calibration process [6]), we have to determine 6 parameters (three for $\mathbf{R}$ and three for $\mathbf{t}$), denoted by vector $\mathbf{p}$. We suppose we know the 3D-2D matching of n various features described by chains of points. Let:

- C_i be a 3D curve, described by the chain of 3D points $\{M_{i,j}\}_{1\leq j\leq l_i}$ (note that C_i can be any 3D feature, including points and lines),
- c_i be the projection of C_i in the image plane, described by the chain of 2D points $\{m_{i,j}\}_{1\leq j\leq l_i}$, where $m_{i,j} = Proj(\mathbf{R}M_{i,j} + \mathbf{t})$ ($Proj$ denoting the projection of a 3D point in the image plane),
- c'_i be the detected curve (tracked curve) corresponding to C_i, described by the chain of 2D points $\{m'_{i,j}\}_{1\leq j\leq l'_i}$.

A simple solution would be to perform a one stage minimization

$$\min \sum_{i,j} \rho(d_{i,j}) \tag{1}$$

where $d_{i,j} = Dist(m'_{i,j}, c_i)$ ($Dist$ being a function which approximates the Euclidean distance from a point to a contour) and ρ is a positive, symmetric function, used to reduce the influence of erroneous parts (see below).

Unfortunately, this method is unsatisfactory because it merges all the features into a set of points, and makes no distinction between local errors (when a feature is only partially well localized), and gross errors (when the position of a feature is completely erroneous). However, these two kinds of errors are not identical, and not treating them separately induces a great loss of robustness and accuracy.

By contrast, we propose to perform a robust estimation in a two-stage process: a *local stage*, which computes a robust residual for each feature, and a *global stage* which minimizes a robust function of these residuals. The local stage reduces the influence of erroneous sections of the contours (features 1 and 4 on Figure 3.d), whereas the global stage discards the *feature outliers*, *i.e.* contours which are completely erroneous, or which contain too large a portion of erroneous points (feature 5 on Figure 3.d).

3.2 The Global Stage

This stage fits the tracked 2D features to the projection of the 3D features, by minimizing the residuals r_i which are computed for each couple of 3D/2D features (see the local stage). In order to reduce the influence of the feature outliers, that is features whose residual is relatively large when the correct pose has been found, we have to perform a robust optimization (using a classical *least squares* - LS - technique, that is minimizing $\sum_{i=1}^{n} r_i^2$, would increase the influence of the features linearly with their residual).

Statisticians have suggested many different robust estimators. Among them, the two most popular are the *M-estimators* and the *least median of squares* (LMS) method, which have been used in many computer vision problems [10, 14, 22].

The LMS technique, suggested by Rousseeuw and Leroy [18], consists of minimizing the median of the squared residuals:

$$\min_{\mathbf{p}} \text{med}_i r_i^2. \tag{2}$$

Minimizing the median ignores the errors of the largest ranked half of the data elements. Thus, this method is able to handle data sets which contain less than 50% outliers. However, as only a part of the data is considered, the LMS is not very accurate. That's why we often perform a *reduced least squares* (RLS) after a LMS, that is a LS which includes all the data whose residual is lower than 2.5 $\hat{\sigma}$, where $\hat{\sigma}$ is the standard deviation of the residuals. Factor $\hat{\sigma}$ has itself to be estimated in a robust way: we take

$$\hat{\sigma} = 2.6477 \sqrt{\frac{1}{h}\sum_{i=1}^{h} r_{o(i)}^2}, \tag{3}$$

where h is equal to $n - [n/2]$ and $r_{o(1)}^2 \leq ... \leq r_{o(n)}^2$ are the ordered squared residuals. Like LMS, the second factor of equation (3) only considers the first ranked half of the data. The first factor is introduced because $\frac{1}{h}\sum_{i=1}^{h} r_{o(i)}^2$ is estimated to be approximately $1/2.6477^2$ when the residuals are random numbers sampled from the gaussian normal distribution $N(0, 1)$ (see [18] for more details). Nevertheless, this method often leads to a local minimum. Indeed, as LMS only

minimizes the residuals of half of the features, it can easily happen that these residuals are much smaller than the ones of the other features, which are hence not taken into account when performing RLS.

We therefore prefer to use the *M-estimation* technique, developed by Huber [11], which minimizes the sum of a function of the residuals:

$$\min_{\mathbf{p}} \sum_{i=1}^{n} \rho(r_i), \tag{4}$$

where ρ is a continuous, symmetric function with minimum value at zero. Its derivative $\psi(x)$ is called the *influence function* because it occurs as a weighting function in optimization (4). These functions are very efficient, but are not suited to cases where the presence of outliers in the data is too large (experimentally, it must be kept below approximately 20%). Table 1 lists three commonly used ρ functions and their derivative. Among these estimators, some are more restrictive than others: when Tukey's influence function is null for residuals larger than a threshold c, Cauchy's remains larger than zero while decreasing, whereas Huber's remains constant, equal to c. We therefore prefer to use Tukey's function, which is restrictive enough to suppress the influence of outliers, but which takes all the data into consideration (by contrast with LMS). The threshold c is taken equal to kS, where k is a constant (we take $k = 4$ for Tukey), and S a scale factor equal to $\hat{\sigma}$.

Minimization (4) can be performed by standard techniques using an initial estimate of $\mathbf{p}$: a very simple approach like Powell's method [16] proved to be sufficient in our case, and relatively fast to compute (for temporal registration, the initial estimate of $\mathbf{p}$ is the pose computed for the previous frame).

Type	$\rho(x)$	$\psi(x)$
Huber $\begin{cases} \text{if } \lvert x\rvert \le c \\ \text{if } \lvert x\rvert > c \end{cases}$	$\begin{cases} x^2/2 \\ c(\lvert x\rvert - c/2) \end{cases}$	$\begin{cases} x \\ c * \operatorname{sgn}(x) \end{cases}$
Cauchy	$\frac{c^2}{2}\log\left(1+\left(\frac{x}{c}\right)^2\right)$	$\frac{x}{1+\left(\frac{x}{c}\right)^2}$
Tukey $\begin{cases} \text{if } \lvert x\rvert \le c \\ \text{if } \lvert x\rvert > c \end{cases}$	$\begin{cases} \frac{c^2}{6}\left[1-\left(1-\left(\frac{x}{c}\right)^2\right)^3\right] \\ c^2/6 \end{cases}$	$\begin{cases} x\left(1-\left(\frac{x}{c}\right)^2\right)^2 \\ 0 \end{cases}$

Table 1: Three commonly used M-estimators.

3.3 The Local Stage

The aim of this stage is to reduce the influence of erroneous sections of the features: to perform this task, the residual error r_i of curve C_i is computed by a robust function of the distances $\{d_{i,j}\}_{1\le j\le l'_i}$. We could take r_i equal to the median of the $\{d_{i,j}\}$, but once again, this method can lead to a local minimum, where only a part of a projected feature is superimposed on the corresponding 2D feature. That's why we prefer to use the M-estimation technique by taking

$$r_i^2 = \frac{1}{l'_i}\sum_{j=1}^{l'_i} \rho(d_{i,j}). \tag{5}$$

This estimate must not be too restrictive, for the reason just evoked. We have hence chosen Huber's function for the local stage (with k set to 2), which has proved to be a good choice in our experiments.

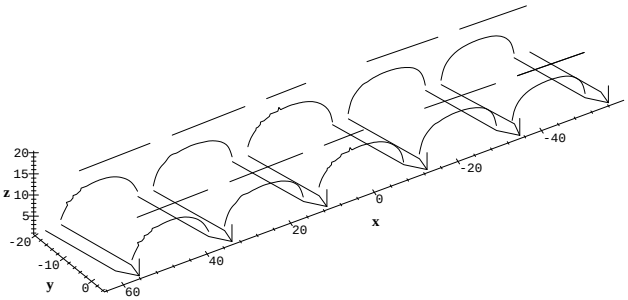

Figure 2: The complete wire-frame model of the bridge.

3.4 Discarding of Feature Outliers

Once a first estimation of the pose has been done using the previous method, the detection of feature outliers can be performed easily: as they should not have influenced the estimation, their residual must be much larger than the other ones. We therefore only have to compare them with the standard deviation of all the residuals: if $r_i > 2.5\ \hat{\sigma}$ (where $\hat{\sigma}$ is given by equation (3)), then the feature discarded.

In order to refine the pose, we can now perform a LS estimation on the retained features (r_i being still given by equation (5)).

4 An Augmented Reality Application

We present in this section an application of our method to an augmented reality application: the illumination of the bridges of Paris. Our motivation came from the encrustation of a model illuminated synthetically in its real environment. The aim was to test several candidate illumination projects for a number of bridges of the Seine, and be able to choose on computer simulations alone what project was the best. Most importantly, we wanted to evaluate the influence of this illumination on the surrounding elements.

A 300-image panoramic sequence of the Pont Neuf was shot at dusk time from another bridge. The fact that the images are dark and noisy has a strong influence on the quality of the segmentation (see Figure 3.a) and restricts the number of reliable features that we are able to use (typically, between 6 and 8 curves). It should also be noted that the modelling of the bridge was done mostly manually using information from architectural maps. It is very unreliable at places (*e.g.* the staircase effect on some of the arches - see Figure 2). The intrinsic parameters of the camera are obtained by calibration on a reference object close to the observer (in the 3 meters range) while the scene under consideration is far from the camera (in the 300 meters range), which induces a lot of noise on these parameters. The reader should keep these facts in mind when visually evaluating the final composition.

Figure 3.b shows the set of tracked features for the 10^{th} image. The solid lines correspond to the tracked 2D features, whereas the dashed lines correspond to the projection of the corresponding model features (black is used for the features which are not - yet - used). The reader may notice that feature 4 is not fully correct. This is mainly due to the snake process, that makes the snake attracted by high gradients. However, as a great part of the feature correctly matches the arch, it is retained in the set of tracked features. The result of the tracking in the 12^{th} image is shown in Figure 3.c. Except for feature 5, the other primitives are well tracked. The error on feature 5 is due to the failure of the prediction step in the tracking process due to noise in the image. Hence, the snake converges towards an erroneous contour.

Figure 3.d shows the re-projection of the model features after the robust pose computation. Despite the bad accuracy of the model, the result is visually convincing. In order the reader to be aware of the parts of the curve which are less taken into account in the computation, we have drawn in black the points for which the residual is greater than c (c is defined in Table 1). Roughly speaking, these points are the ones for which the weight in the computation is decreased because their residual is too large. It must also be noticed that feature 5 is considered as an outlier and is removed from the set of tracked features (discarded features are drawn in black).

In Figure 3.e, a new feature has been added to the set of tracked features (5). Note that in this case, this new feature replaces the one which has been mistracked. Using this new set of features, the pose is re-estimated (Figure 3.f). Figure 4 shows the result of the composition of the computer-generated and the video image for the 60^{th} image. Precisions about this composition may be found in [2].

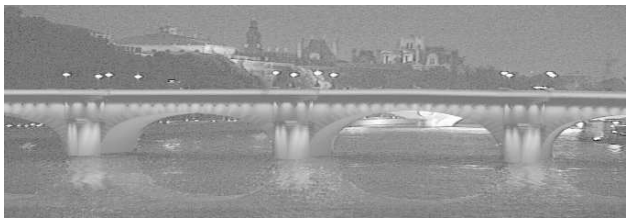

Figure 4: Final composition for the 60^{th} image.

5 Conclusion

We have presented an autonomous model registration system capable of tracking an object in an image sequence. Emphasis has been put on the pose computation method, which handles various features in a very robust way and is also used to update the set of visible features along the sequence. An application to the simulation of a lighting project of a bridge in Paris has proven the validity of our approach.

Several directions for further investigation are suggested: in order to use our method for real time applications, we could greatly improve the speed of our algorithms by processing both the tracking task and the pose estimation in a parallel way. As our pose estimation method admits points, we could also track them along the sequence by using a corner tracker or other kind of point tracking. Finally, we work on an automatic way of finding the more pertinent tracked 3D features.

References

[1] M.-O. Berger. How to Track Efficiently Piecewise Curved Contours with a View to Reconstructing 3D Objects. In *Proceedings of the 12th ICPR, Jerusalem (Israel)*, volume 1, pages 32–36, 1994.

[2] M.-O. Berger, C. Chevrier, and G. Simon. Compositing Computer and Video Image Sequences: Robust Algorithms for the Reconstruction of the Camera Parameters. In *Eurographics'96, Poitiers, France*, volume 15, pages 23–32, August 1996.

[3] J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on PAMI*, 8(6):679–698, 1986.

[4] D. Dementhon and L. Davis. Model Based Object Pose in 25 Lines of Code. *IJCV*, 15:123–141, 1995.

[5] M. Dhome, M. Richetin, J.T. Lapresté, and G. Rives. Determination of the Attitude of 3-D Objects from a Single Perspective View. *IEEE Transactions on PAMI*, 11(12):1265–1278, 1989.

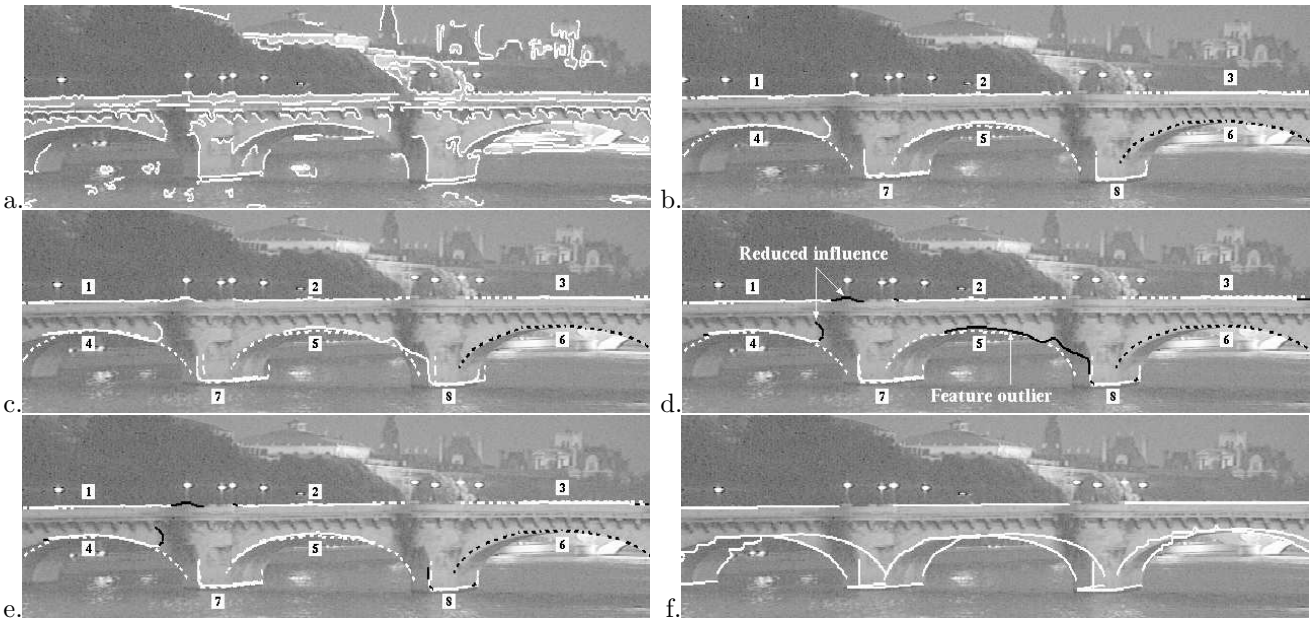

Figure 3: Temporal registration for the 12^{th} image. (a) Edge map. (b) 2D features before tracking (image 10). (c) Tracking in image 12 (projections of the 3D features are those of image 10). (d) Robust pose computation. (e) Updating of tracked features. (f) Re-projection of the model after updating.

[6] O. D. Faugeras and G. Toscani. The Calibration Problem for Stereo. In *Proceedings of IEEE Conference on CVPR, Miami, FL (USA)*, pages 15–20, 1986.

[7] J. Feldmar, N. Ayache, and F. Betting. 3D-2D projective registration of free form curves and surfaces. *Computer Vision and Image Understanding*, 65(3):403–424, 1997.

[8] M. Ferri, F. Mangili, and G. Viano. Projective Pose Estimation of Linear and Quadratic Primitives in Monocular Computer Vision. *CVGIP: Image Understanding*, 58(1):66–84, July 1993.

[9] D. Gennery. Visual Tracking of Known Three Dimensional objects. *IJCV*, 7(3):243–270, 1992.

[10] R. M. Haralick, H. Joo, C. N. Lee, X. Zhuang, V.G. Vaidya, and M. B. Kim. Pose estimation from corresponding point data. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6), 1989.

[11] P. J. Huber. *Robust Statistics*. Wiley, New York, 1981.

[12] D. Koller, K. Daniilidis, and H. H. Nagel. Model-Based Object Tracking in Traffic Scenes. In *Proceedings of Second ECCV, Santa Margherita Ligure (Italy)*, volume 588 of *Lecture Notes in Computer Science*, pages 437–452, 1992.

[13] D. Kriegman and J. Ponce. On Recognizing and Positioning Curved 3D objects from Image Contours. *IEEE Transactions on PAMI*, 12(12):1127–1137, December 1990.

[14] R. Kumar and A. Hanson. Robust Methods for Estimating Pose and a Sensitivity Analysis. *CVGIP: Image Understanding*, 60(3):313–342, 1994.

[15] D. Lowe. Robust Model based Motion Tracking Trough the Integration of Search and Estimation. *IJCV*, 8(2):113–122, 1992.

[16] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C, The Art of Scientific Computing*. Cambridge University Press, 1988.

[17] S. Ravela, B. Draper, J. Lim, and R. Weiss. Tracking Object Motion Across Aspect Changes for Augmented Reality . In *ARPA Image Undertanding Worshop, Palm Spring (USA)*, August 1996.

[18] P. Rousseeuw and A. Leroy. *Robust Regression and Outlier Detection*. Wiley Series in Probability and Mathematical Statistics. Wiley, 1987.

[19] T. Shakunaga. Robust Line Based Pose Enumeration From a Single Image. In *Proceedings of 4th ICCV, Berlin (Germany)*, pages 545–550, 1993.

[20] G. Simon and M.-O. Berger. A two-stage robust statistical method for temporal registration from features of various type. Rapport de recherche 3235, INRIA, August 1997.

[21] M. Uenohara and T. Kanade. Vision based object registration for real time image overlay. *Journal of Computers in Biology and Medecine*, 1996.

[22] Z. Zhang, R. Deriche, O. Faugeras, and Q. Luong. A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry. *Artificial Intelligence*, 78:87–119, October 1995.