Computer Vision Methods for Registration: Mixing 3D Knowledge and 2D Correspondences for Accurate Image Composition

G. Simon, V. Lepetit and M.-O. Berger
LORIA, BP 239,
54506 Vandoeuvre-les-Nancy, France
e-mail: {gsimon, lepetit, berger}@loria.fr

Abstract

We focus in this paper on the problem of adding computer-generated objects in video sequences. A two-stage robust statistical method is used for computing the pose from model-image correspondences of tracked curves. This method is able to give a correct estimate of the pose even when tracking errors occur. However, if we want to add virtual objects in a scene area which does not contain (or contains few) model features, the reprojection error in this area is likely to be large. In order to improve the accuracy of the viewpoint, we use 2D keypoints that can be easily matched in two consecutive images. As the relationship between two matched points is a function of the camera motion, the viewpoint can be improved by minimizing a cost function which encompasses the reprojection error as well as the matching error between two frames. The reliability of the system is shown on an encrustation of a virtual car in a sequence of the Stanislas square.

1 Introduction

Augmented Reality (AR) is an effective mean for utilizing and exploiting the potential of computer-based information and databases. In augmented reality, the computer provides additional information that enhance or augment the real world, rather than replacing it with a completely virtual environment. In contrast to virtual reality, where the user is immersed in a completely computer-generated world, AR allows the user to interact with the real world in a natural way. This explains why interest in AR has substantially increased in the past few years and medical, manufacturing or urban planning applications have been developed [3, 5, 23].

We focus in this paper on the problem of adding computer-generated objects (also called virtual objects) in video sequences. This is one of the key points for numerous AR applications: for instance, suppose we want to assess the potential impact of a new construction in its final setting; visualizing the architectural project on a video of the environment allows designers to test several architectural projects on computer simulations alone. Special effects in movies also require such a composition process.

In order to make AR systems effective, the computer generated objects and the real scene must be combined seamlessly so that the virtual objects align well with the real ones. It is therefore essential to determine accurately the location and the optical properties of the cameras. The registration task must be achieved with special care because the human visual system is very good at detecting even small misregistrations. Realistic merging of virtual and real objects also requires that objects behave in a physical plausible manner in the environment: they can be occluded by objects in the scene, they are shadowed by other objects ...

In this paper, we only focus on the registration problem because it is one of the most basic challenge in augmented reality. But we have proposed some preliminary solutions to the occlusion problem in [2].

Registration problems can be solved by using either algorithmic solutions or sensor-based solutions. For instance, position sensors (as Polhemus sensors) can be used to locate the camera (or the viewer) [21]. Easily detectable landmarks placed in the scene can also be used to make the registration process easier [5]. However, instrumenting the real world is not always possible, especially for vast or outdoor environments. Thus, vision-based object registration is an interesting and cheaper approach that leaves the environment unmodified.

In our approach, we assume that the camera has been calibrated previously. This means that the internal characteristics of the camera (focal length and optical centre) have been previously computed using a calibration target [7]. Thus, the registration problem amounts to compute the location (or the viewpoint) of the camera.

Pose recovery has been extensively studied in the past few years. Two broad classes of methods can be distinguished: the most classical one uses object-based registration; this means that 3D knowledge is needed to compute the pose from image/model correspondences. Such methods minimize the reprojection error of the models features in the image. Hence, one of the limitations of this method originates in the spatial distribution of the model points: the reprojection error is likely to be large far from the 3D features used for the viewpoint computation.

The other alternative is basically 2D: if the projection of a sufficient number of points is observed from different positions, the camera pose can be recovered as well as the 3D location of the points up to a scale factor [6, 22]. Unfortunately, these approaches turn out to be very sensitive to inaccuracies in 2D feature measurements.

Our approach encompasses the strength of these two methods: the classical viewpoint computation from 3D-2D correspondences is utilized to compute a first approximation of the viewpoint. Then, automatic extracting and matching of key-points between two consecutive views allow us to refine the viewpoint computation. As a result, viewpoint computation can be achieved through the sequence at the expense of minimal 3D knowledge on the scene.

Our system stands out from previous works on the following ground:

- **Minimal interaction with the user**: the user is only requested to point out four points and their 3D counterparts in the first image of the sequence. The 2D features corresponding to the visible model features are then automatically determined and tracked in the sequence (the tracking tool we use is described in [1]). The pose is then computed from their correspondences with the 3D model.
- **Model features**: Since the world around us is not piecewise-planar, the significant features that can be extracted in an image are often curved contours, especially if we consider outdoor environments. One of the strength of our pose algorithm is to handle points, lines and curves correspondences for pose computation.
- **Robustness**: It is well known that false 3D-2D matches (outliers) can have a large effect on the resulting pose. We have therefore developed 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 or curved features.
- **Accounting for interest points correspondences** Mixing 3D-2D correspondences in an image and 2D-2D correspondences between two consecutive frames allows us to compute the viewpoint accurately. Unlike model-based methods which require a large number of 3D features to give precise results, our mixing method only needs little 3D knowledge on the scene.

The paper begins with the explanation of our robust algorithm for viewpoint computation from 3D-2D correspondences, including results on various augmented reality applications. Section 3 describes how the viewpoint computation can be dramatically improved using key-points correspondences. Finally, we show results demonstrating the accuracy of pose estimation.

2 Robust pose computation from various features

Once the model/image correspondences have been established in the first frame, they are generally 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. While numerous papers are dedicated to pose estimation from points or lines [4, 9, 14], only few papers have been devoted to the 3D-2D registration of curves [8, 13]. However, these papers are not concerned with possible matching errors. The details of our robust pose computation algorithm (RPC) are given in this section. Emphasis is put on the robustness of the computation. First we address the problem of points correspondences; the much more difficult case of curves correspondences is then considered.

2.1 Pose computation from point correspondences

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 [7]), 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 points $M_{i\{1\leq i\leq n\}}$ and their corresponding points $m_{i\{1\leq i\leq n\}}$ in the image. Computing the viewpoint amounts to find $\mathbf{R}, \mathbf{t}$ which minimize the re-projection error:

$$\sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} Dist(m_i, Proj(M_i)).$$

Unfortunately the least-square estimator is not robust against false matches, because the larger the residual r_i is, the larger is its influence on the final estimate.

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, 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, 24].

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

$$\min_{\mathbf{p}} \operatorname{med}_i r_i^2, \tag{1}$$

where $\operatorname{med}_i x_i = x_{[n/2]}$ if $x_1 \leq x_2 \leq x_n$.

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. Moreover, such an estimation is cost expensive because LMS cannot be reduced to a straightforward formula.

We therefore prefer to use the *M-estimation* technique, developed by Huber [12], which minimizes the sum of a function of the residuals:

$$\min_{\mathbf{p}} \sum_{i=1}^{n} \rho(r_i), \tag{2}$$

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 (2). 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).

Minimization (2) 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).

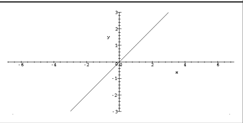
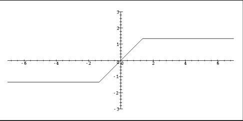
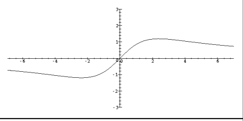
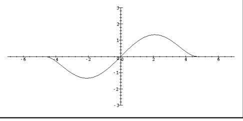
type	$\rho(x)$	$\psi(x)$	graph of $\psi(x)$
Meansquares	$x^2/2$	x	
Huber $\begin{cases} \text{if } \lvert x\rvert \leq c \\ \text{if } \lvert x\rvert > c \end{cases}$	$\begin{cases} x^2/2 \\ c(\lvert x\rvert - c/2) \end{cases}$	$\begin{cases} x \\ c * \operatorname{sgn}(x) \end{cases}$	
Cauchy	$\frac{c^2}{2}\log\left(1+\left(\frac{x}{c}\right)^2\right)$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

Table 1: A few commonly used M-estimators.

2.2 Computing viewpoint from curve correspondences

This problem is much more difficult because point-to-point correspondences between the 3D curves and the corresponding 2D curves are not available. 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

$$\min_{\mathbf{p}} \sum_{i,j} \rho(d_{i,j}) \tag{3}$$

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.

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 1.c), 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 1.c).

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 $\{d_{i,j}\}_{1 \leq j \leq l'_i}$. 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

$$r_i^2 = \frac{1}{l'_i} \sum_{j=1}^{l'_i} \rho(d_{i,j}). \tag{4}$$

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

The global stage

Once a robust residual has been computed for each feature, the viewpoint is computed in a robust way by minimizing

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

where ρ is the Tukey function.

Discarding Feature Outliers

The detection of feature outliers can now 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 $r_i > 2.5 \overset{\wedge}{\sigma}$ (where $\overset{\wedge}{\sigma}$ is computed in a robust way [14]), then the feature is discarded.

In order to refine the pose, we can now perform a least-square estimation on the retained features (r_i being still given by equation (4)).

The problem of maintaining registration over time (replacing the feature outliers, integrating the new appearing features…) is detailed in [20].

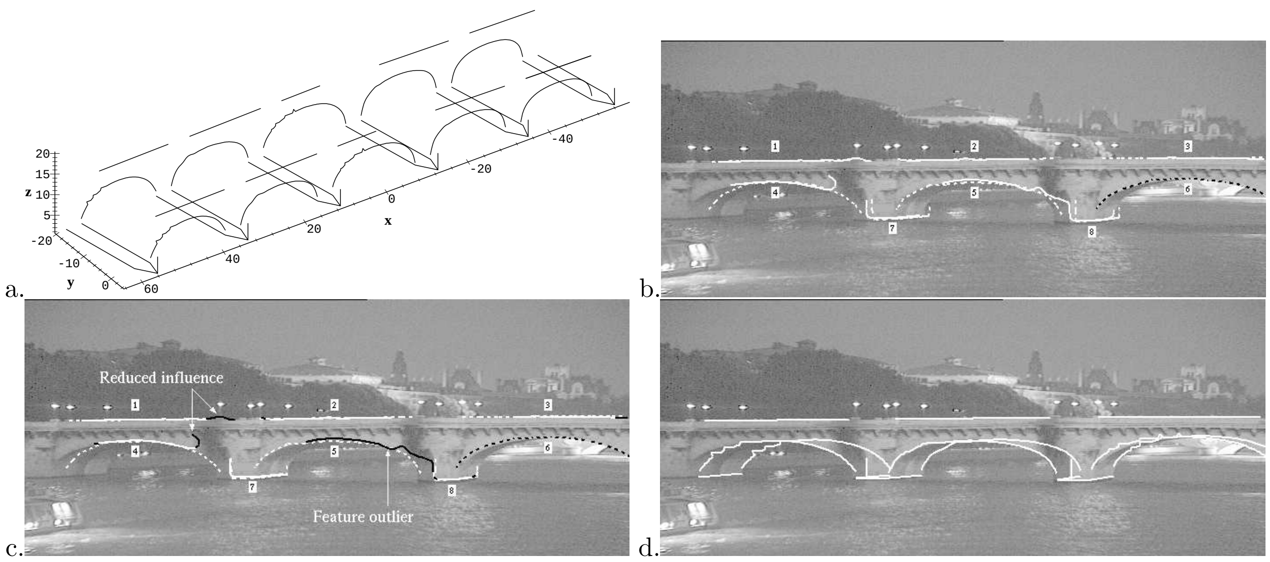


Figure 1: Temporal registration.
(a): Wireframe model of the object to be registered (the bridge) (b): Tracking of image features. White lines correspond to the tracked features, white dashed lines to the projection of their 3D correspondents in the previous frame and black dashed lines to the features not (yet) used. (c): Robust pose computation. Sections for which residuals are greater than c and feature outliers are drawn in black. (d): Re-projection of the model.

3 Improving the viewpoint computation

Computing the viewpoint from 2D/3D correspondences allows us to minimize the reprojection error for the model features. Unfortunately, if we want to add virtual objects in a scene area which does not contain (or contains few) model points, the reprojection error in this area is likely to be large. As an example, consider the scene depicted in Fig. 2 in which we want to add a virtual car passing behind the statue (on the right). The viewpoint has been computed from 3D features belonging to the opera (the building in the back of the scene). The epipolar lines for the points drawn in Fig. 2.a are shown in Fig. 2.b: they pass far from the corresponding points. This proves that the viewpoint is not computed with sufficient accuracy. As a consequence, the projection of the virtual object in the scene is likely to be somewhat incorrect. Moreover, occluding objects which stand before the virtual ones will be erroneously reconstructed. Therefore, the occlusions between the virtual objects and the real scene cannot be solved properly which reduces dramatically the realism of the image composition.

In order to improve the accuracy of the viewpoint, we use key-points that can be easily matched in two consecutive images. As the relationship between two matched points is a function of the camera motion and of the intrinsic parameters, the viewpoint can be improved by minimizing a cost function which encompasses the reprojection error as well as the matching error between two frames. Since the key-points do not generally correspond to the model points, the viewpoint computation will be improved through these 2D correspondences.

Section 3.1 describes the way to extract key-points. Section 3.2 presents the cost function we use to improve the viewpoint. Significant results are shown in section 4.

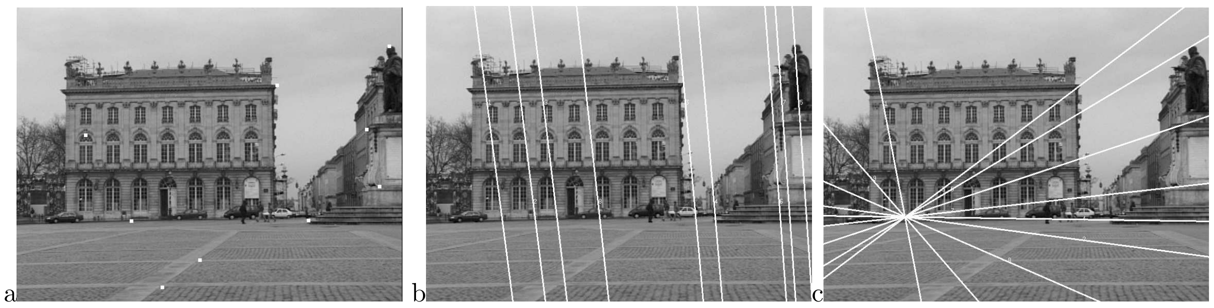


Figure 2: Epipolar lines corresponding to the points drawn in (a): with the RPC algorithm (b) with the mixing method (c)

3.1 Extracting and matching key-points

Key-points (or interest points) are locations in the image where the signal changes two dimensionally: corner, T-junctions, locations where the texture varies significantly... Approaches for detecting key-points can be broadly divided in two groups: the first group involves first extracting edges and then searching for points having maximum curvature; the second group consists of approaches that work directly on the grey-level image. As the edges are already used in the viewpoint computation, we resort to the second approach that provides us with interesting texture points which are not yet used. We use the approach developed by Harris and Stephens [11]: they use the autocorrelation function of the image to compute a measure which indicates the presence of an interest point. More precisely, the eigenvalues of the matrix

$$\begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}$$

are the principal curvatures of the auto-correlation function. If these values are high, a key-point is declared.

We still have to match these key-points between two consecutive images. Numerous works use correlation techniques to achieve this task [24]. These methods are well-suited when the motion in the image is roughly a translation but they are unable to cope with rotations and scale changes. That is the reason why we prefer to use the matching approach developed in [19]: each key-point is characterized locally by a vector of differential invariants under the group of displacements. For example, the vector of differential invariants up to second order is

$$\begin{bmatrix} I \\ I_x^2 + I_y^2 \\ I_{xx}I_x^2 + 2I_{xy}I_xI_y + I_{yy}I_y^2 \\ I_{xx} + I_{yy} \\ I_{xx}^2 + 2I_{xy}I_{yx} + I_{yy}^2 \end{bmatrix}.$$

The invariance of the vector makes the matching stage easier even in case of important geometric transformations. Moderated scale changes can also be considered. The interested reader can find further details on the computation of the local invariants in [19]. The key-points are then matched according to a measure of similarity between the invariant vectors. Neighbouring constraints are also used to make the matching easier (close key-points must have a similar disparity).

3.2 Mixing 3D knowledge and points correspondences

Given the viewpoint $[\mathbf{R}_k, \mathbf{t}_k]$ computed in a given frame k, we now explain how we compute the viewpoint in the next frame $k+1$ using the 3D model as well as the matched keypoints $(q_1^i, q_2^i)_{1 \leq i \leq m}$. Before describing the cost function to be minimized, we first recall in a little detail the relationships between two matched key-points m_1, m_2 and the two viewpoints $[\mathbf{R}_k, \mathbf{t}_k]$ and $[\mathbf{R}_{k+1}, \mathbf{t}_{k+1}]$. Let $\mathbf{\Delta R}, \mathbf{\Delta t}$ be the relative displacement of the camera between frame k and $k+1$. Let A be the intrinsic matrix of the camera:

$$A = \begin{bmatrix} k_uf & 0 & u_0 \\ 0 & k_vf & v_0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Let q_1 and q_2 be the images of a 3D point M on the cameras. Their homogeneous coordinates are denoted $\tilde{q_1}$ and $\tilde{q_2}$. We then have the fundamental equation (see appendix A for a proof of this result)

$$\tilde{q_2}^t A^{-1^t} \mathbf{\Delta T} \mathbf{\Delta R} A^{-1} \tilde{q_1} = 0,$$

where $\mathbf{\Delta T}$ is an antisymmetric matrix such that $\mathbf{\Delta T}x = \mathbf{\Delta t} \wedge x$ for all x. $F = A^{-1^t} \mathbf{\Delta T} \mathbf{\Delta R} A^{-1}$ is called the fundamental matrix.

Then, a simple way to improve the viewpoint computation using the interest points is to minimize

$$min_{\mathbf{R}_{k+1}, \mathbf{t}_{k+1}} \left(\frac{1}{n} \sum_{i=1}^{n} \rho_1(r_i) + \frac{\lambda}{m} \sum_{i=1}^{m} \rho_2(vi) \right), \tag{5}$$

where

- r_i is the distance in frame $k+1$ between the tracked features and the projection of the model features,
- $v_i = \sqrt{\frac{1}{(F\tilde{q_1^i})_1^2 + (F\tilde{q_1^i})_2^2} + \frac{1}{(F^t\tilde{q_2^i})_1^2 + (F^t\tilde{q_2^i})_2^2}} |\tilde{q_2^i}^t F \tilde{q_1^i}|$ measures the quality of the matching between q_i and q_i' [15],

- ρ_1 and ρ_2 are M-estimators. Note that the use of a M-estimator for the key-points correspondences is not essential: as the key-points are significant points in the image, false matches are unusual.

The λ parameter controls the compromise between the closeness to the available 3D data and the quality of the 2D correspondences between the key-points. We use $\lambda = 2$ in our practical experiments.

The minimum of equation 5 is computed by using an iterative algorithm for minimization such as Powell's algorithm.

4 Results

Fig. 3 shows the result of using the mixing algorithm to incrust a virtual car in a video sequence. A video of the Stanislas Square, city of Nancy, France, has been shot from a car running around the square. Our aim is to incrust a virtual car passing on the square. It is worth noting here that the 3D data available on the scene only concern the opera. On the other hand, the 3D model of the statue is not available. The virtual car is encrusted in the foreground of the scene, on the ground near the statue.

Fig.3.a shows the projection of the opera model in the image by using the RPC algorithm alone with the car incrustation. The projection error seems quite correct but the car seems to hover! This result is not really surprising as the viewpoint computation has been performed with almost coplanar points in the background of the scene. The key-points extracted in the scene and the matching arrows are shown in Fig.3.b. The result of the mixing algorithm is shown in Fig.3.c. The car and the scene are combined seamlessly and the realism of the composition is very good.

Fig. 4 exhibits the viewpoint evolution for the RPC algorithm and the mixing method. In the considered sequence, the camera is passing on the opposite side of the opera. The motion of the camera is then a translation along the x-axis. Fig. 4 proves the efficiency of the mixing method: the t_y and t_z coordinates are nearly constant whereas the t_x, α, β and γ parameters evolve slowly.

Other significant results on video image sequences can be seen at URL http://www.loria.fr/ gsimon/videos.html.

5 Conclusion

To conclude, we have presented a robust and accurate registration method which allows us to combine the real and the virtual worlds seamlessly. One of the main advantages of our approach is to perform pose computation over the sequence in a completely autonomous manner. The accuracy of the pose computation is due to the combined use of 3D-2D correspondences in an image and 2D-2D correspondences in two consecutive frames: indeed the use of 2D-2D correspondences allows us to bring some kind of spatial information on the scene in areas where 3D model features are missing. As a result, our method only requires a limited number of 3D features to be effective.

Currently, the time needed to process one frame is around 25 s (for 5 3D curves and 100 key-points on an UltraSparc 143MHz). However, we can greatly improve the speed of our algorithm by processing both the pose computation and the key-points extraction in a parallel way. In this way, we think that our algorithm is amenable to real time.

Future works will concern the automatic determination of the intrinsic camera parameters. Indeed, these parameters are currently computed off line before shooting the sequence. As the zoom of the camera may change during shooting, it would be interesting to compute dynamically the intrinsic camera parameters.

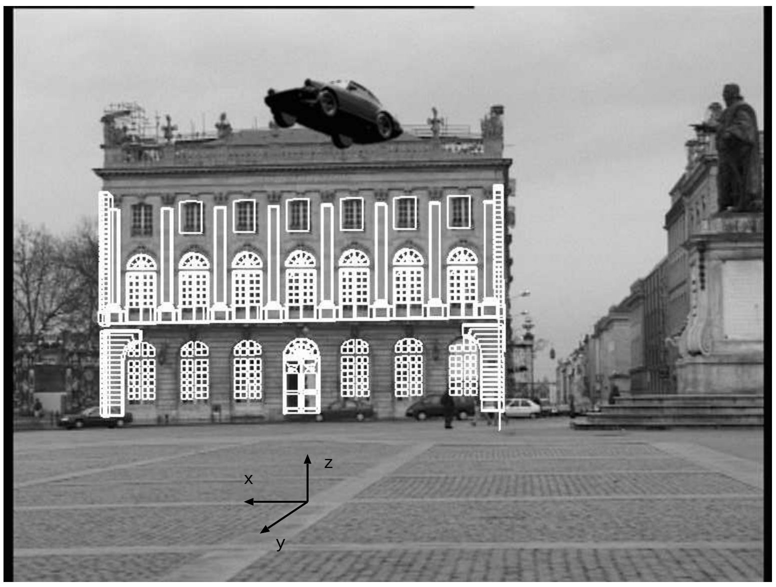
Appendix A: the fundamental matrix

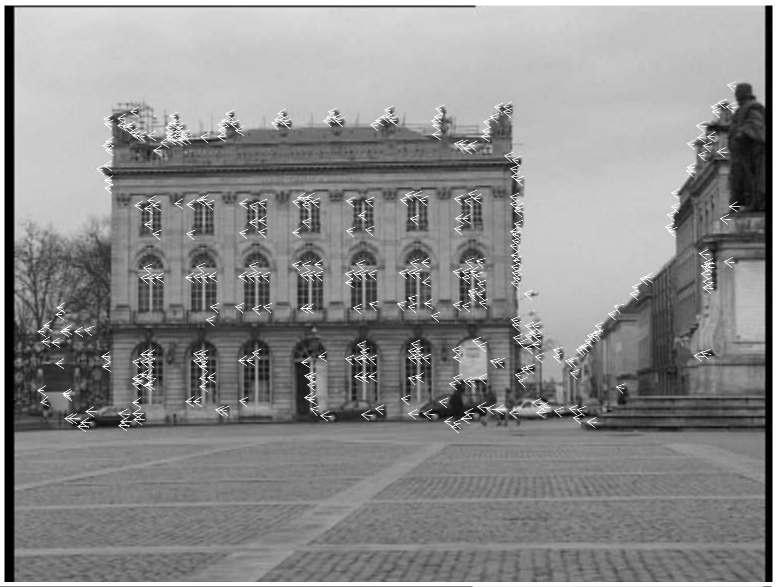
Consider the case of 2 cameras as shown in Fig. 5 where C_1 and C_2 are the optical centres of the cameras. Let the displacement from the first to the second be $[\mathbf{R}, \mathbf{t}]$. Let m_1 and m_2 be the image of a 3D point M. The three vectors $\vec{C_1m_1}$, $\vec{C_2m_2}$ and t are coplanar. Without loss of generality, we assume that M is expressed in the coordinate frame of the first camera. As the coordinates of C_2m_2 in the first frame are RC_2m_2, the coplanarity of the three vectors can be expressed by the equation:

$$(\mathbf{t} \wedge \vec{C_1m_1}).\vec{C_2m_2} = 0. \tag{6}$$

As $\mathbf{t} \wedge x = \mathbf{T}x$ with $\mathbf{T} = \left[\begin{array}{ccc} 0 & -t_z & ty \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{array} \right]$, we can write equation (6) as

$$\vec{C_2m_2}^t \mathbf{T}\mathbf{R}\vec{C_1m_1} = 0.$$

a. 

b. 

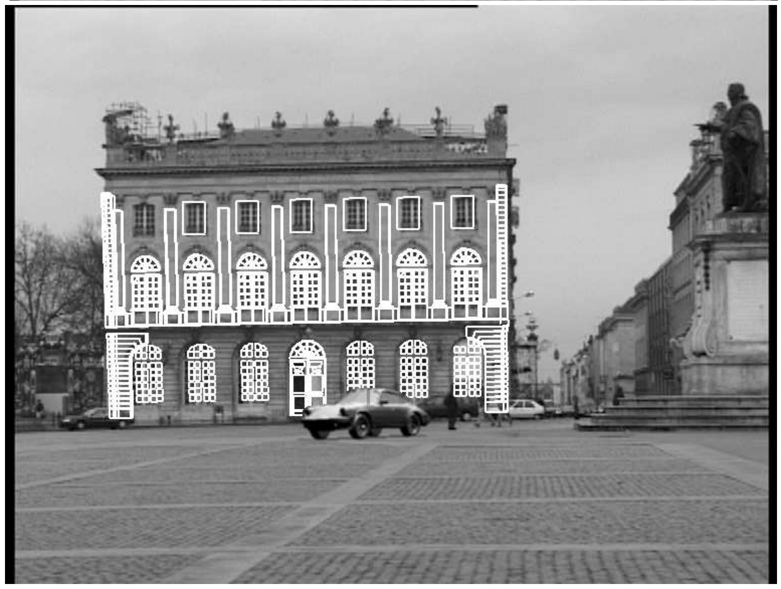
c. 

Figure 3: Result of the mixing algorithm: (a) the image composition using the viewpoint computed with the RPC algorithm (b) the extracted keypoints (c) image composition with the result given by the mixing algorithm.

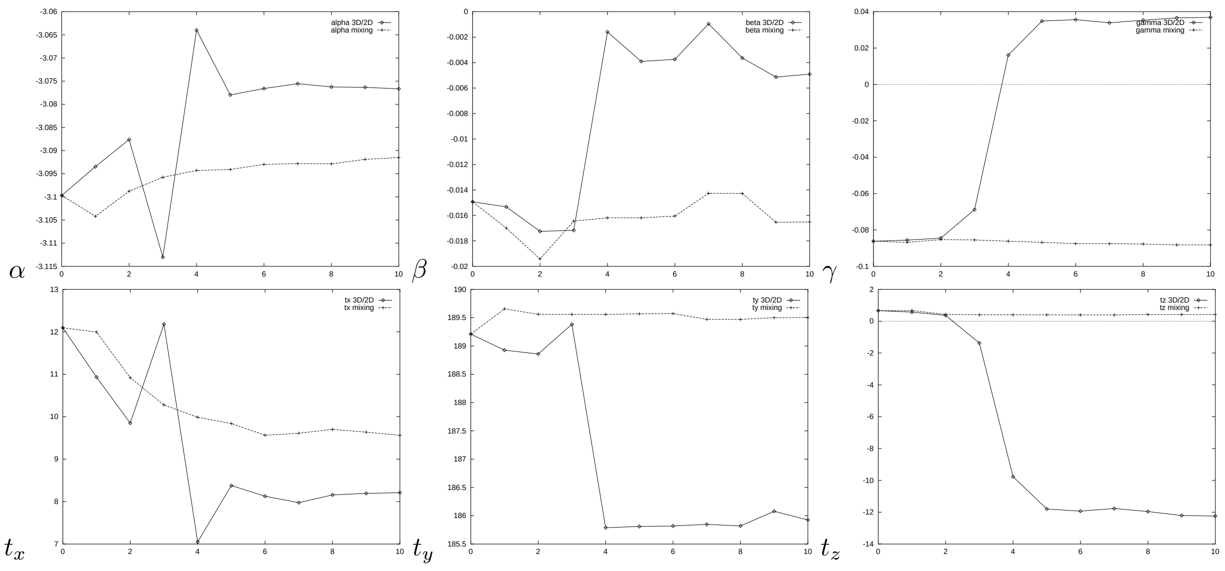


Figure 4: Evolution of the motion parameters (Euler angles and translation) for the RPC and the mixing algorithm.

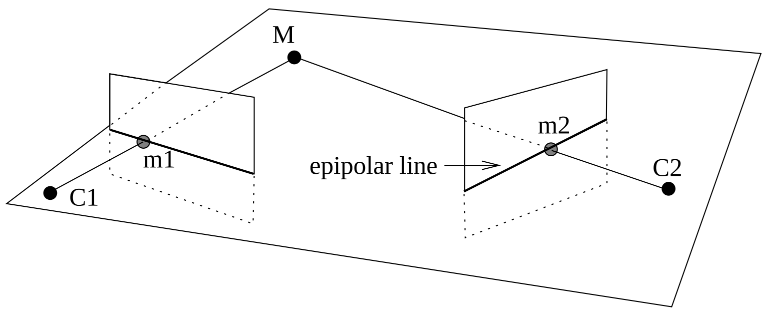


Figure 5: Geometry of two cameras

The mapping between the pixel coordinates $m(u,v)$ and the metric coordinates (X,Y,Z) in the camera frame is given by

$$\tilde{m} = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} k_u f & 0 & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = A \begin{bmatrix} X \\ Y \\ Z \end{bmatrix},$$

where k_u, k_v are the pixel sizes, f is the focal length, and u_0, v_0 are the coordinates of intersection of the optical axis with the image plane.

Hence, we have

$$\tilde{m_2}^t (A^{-1})^t \mathbf{TR} A^{-1} \tilde{m_1} = 0.$$

Let F be $F = (A^{-1})^t \mathbf{TR} A^{-1}$; then two corresponding points m_1 m_2 satisfy the equation

$$\tilde{m_2}^t F \tilde{m_1} = 0.$$

References

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

[2] M.-O. Berger. Resolving occlusion in augmented reality: a contour-based approach without 3d reconstruction. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Puerto Rico (USA)*, pages 91–96, June 1997.

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

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

[5] G. Ertl, H. Müller-Seelich, and B. Tabatabai. MOVE-X: A System for Combining Video Films and Computer Animation. In *Eurographics*, pages 305–313, 1991.

[6] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint.* Artificial Intelligence. MIT Press, 1993.

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

[8] 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.

[9] 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.

[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] C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Proceedings of 4th Alvey Conference*, Cambridge, August 1988.

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

[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] Q.-T. Luong, R. Deriche, O. Faugeras, and T. Papadopoulo. On Determining the Fundamental Matrix: Analysis of Different Methods and Experimental Results. Rapport de recherche 1894, INRIA, 1993.

[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] C. Schmid and R. Mohr. Local Grayvalue Invariants for Image Retrieval. *IEEE Transactions on PAMI*, 19(5):530–535, August 1997.

[20] G. Simon and M.-O. Berger. A Two-stage Robust Statistical Method for Temporal Registration from Features of Various Type. In *Proceedings of 6th International Conference on Computer Vision, Bombay (India)*, pages 261–266, January 1998.

[21] A. State, G. Hirota, D. Chen, W. garett, and M. Livingston. Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking. In *Computer Graphics (Proceedings Siggraph New Orleans)*, pages 429–438, 1996.

[22] C. Tomasi and T. Kanade. Shape and Motion from Image Streams under Orthography: A Factorization Method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

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

[24] 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.