Mixing Synthetic and Video Images of an Outdoor Urban Environment

M.-O. Berger B. Wrobel-Dautcourt S. Petitjean G. Simon

First submission: 11 July 1997 - Revised: April 1998

Abstract

Mixing video and computer-generated images is a new and promising area of research for enhancing reality. It can be used in all the situations when a complete simulation would not be easy to implement. Past work on the subject has relied for a large part on human intervention at key moments of the composition.

In this paper, we show that if enough geometric information about the environment are available, then efficient tools developed in the computer vision literature can be used to build a highly automated augmented reality loop. We focus on outdoor urban environments and present an application for the visual assessment of a new lighting project of the bridges of Paris.

We present a fully augmented 300-image sequence of a specific bridge, the Pont Neuf. Emphasis is put on the robust calculation of the camera position. We also detail the techniques used for matching 2D and 3D primitives and for tracking features over the sequence. Our system overcomes two major difficulties. First, it is capable of handling poor quality images, resulting from the fact that images were shot at night since the goal was to simulate a new lighting system. Second, it can deal with important changes in viewpoint position and in appearance along the sequence. Throughout the paper, many results are shown to illustrate the different steps and difficulties encountered.

1 Introduction

Augmented reality is the technique by which real images can be enhanced by addition of computer-generated or real information. At least, this is the definition of augmented reality we shall adopt in this paper, since the term is currently used in different ways by different people, without what could reasonably be considered a consistent definition. Augmented reality shows great promises in fields where a simulation *in situ* would be either impossible, not realistic enough or too expensive. Of particular interest are applications in medical imaging [1, 25], urban design - for instance for visually assessing the incidence of a future building in an already existing urban environment [7] - and manufacturing [5]. Also, this type of integration is gaining importance in the film industry.

However, this area has up-to-now been largely under-explored and the few applications that have been developed rely heavily on the intervention of the user at the different steps of the mixing process. There are two reasons for this. First, the superposition of the real and virtual objects has to be very accurate for a satisfactory visual effect. Jittering effects are highly unpleasing in a video sequence. Second, the camera may make abrupt position changes, in which case it is very difficult to predict beforehand what features can be detected in images and what features will be used to compute the viewpoint and do the tracking in a sequence.

1.1 A Look at the Literature

We focus here on those works in augmented reality where the authors have attempted to use vision tools for the super-imposition of synthetic objects on video images. It should be noted that past research has essentially dealt with very good quality images of indoor scenes.

Perhaps the first work to consider the overlay of computer-generated images onto a background photograph is that of [28]. [18] give manual solutions for evaluating the visual impact of a building on the landscape. A first try at automating the composition process is presented in [19]. The authors explore some of the key issues involved in augmented reality: computation of the shadows cast by real light sources on virtual objects, simulation of weather phenomena, improvement of the quality of the montage using a novel anti-aliasing technique. Their approach suffers however from two drawbacks. The parameters of the camera are determined by matching 2D points with the corresponding 3D points of the model, but these points are determined manually. In addition, the influence of virtual light sources and specular inter-reflections is not taken into account.

In [14], a composition technique is described that works for a full image sequence and a method for mapping an aerial photograph onto a three-dimensional terrain model is presented. This environmental assessment application is however largely simplified by the fact that aerial photo data are considered as textures. Also, viewpoint determination is only briefly addressed. [9] present a general system for combining virtual buildings and real images of the environment, which is capable of realizing computer animations. Viewpoint determination is achieved by assuming that there are points in the image whose world coordinates are known (*passpoints*). These passpoints are then tracked in a sequence of images to estimate the new position of the camera from the one previously computed. Also, the detection of occluding objects of the real scene is done manually.

A framework is proposed in [13] for automating as much as possible the mixing process. Applications presented concern indoor environments (*e.g.*, a virtual ball moving in front of a set of plates). The acquisition system can be controlled with excellent precision, so its position is supposed to be known. The optical characteristics of the camera are computed using a reference background object of the real scene.

The system of [27] uses computer vision techniques for overlaying real images onto very good quality video images. Pose calculation uses Newton's method but is only briefly discussed. Tracking is correlation-based and object registration is reliably achieved by tracking many features of the object. To check whether features have been correctly tracked, two cross-ratios of four areas of the triangles formed by five points are computed. They are projective invariants and should remain the same over images. A low latency vision hardware allows overlay of image to be done in real-time.

Finally, a registration system allowing to track objects in real time is presented in [21, 22]. A user-specified set of model-image correspondences, known camera parameters and a pre-compiled aspect table that associates discrete viewpoints with object features visible from those views are used to initialise the system. An initial pose is computed using the given 2D-3D correspondences. The system then goes on by repeating a three-step loop:

- view indexing - pose information is used to extract visible features from the aspect table;
- tracking - visible features are used as location hypotheses of feature templates in the next image;
- pose computation - the templates are matched in the image and a new pose is computed.

Some of the system characteristics used by these authors match fairly closely our choices: use of robust methods for pose estimation - iterative re-weighting least squares form of M-estimation [17] -, and correlation-based tracking, combined with steerable filters. Combinations of rotations and translations do not seem to be accurately compensated by their tracker however. Also, the aspect table is constructed manually off-line.

1.2 Motivations and General Framework

To be able to efficiently mix video and virtual images, it is necessary to have a robust method for computing the position of the camera in order to insert the virtual objects at their right place in the images. Among existing systems for the composition of images, one can distinguish between two classes: those which make use of markers adequately placed in the scene to facilitate viewpoint determination and those which base the calculation of the camera position on the natural structures of the scene. Since we aim at using our system in outdoor urban environments, we can not hope to get help from markers. We are thus uniquely concerned with works on image composition where registration is achieved using the sole geometric features of the scene. We also intend to bring the number of manual operations to a minimum.

Most of the current research works on image composition deal with manufactured environments - see for instance [27, 22] - filmed in good light conditions and from a close range. By contrast, our work is concerned with complex outdoor environments filmed in poor lighting conditions, since the ultimate goal is to simulate the effect of a new lighting system on the surrounding architectural elements. And even if the different steps leading to the pose estimation remain the same, *i.e.*, feature detection and extraction, tracking, viewpoint calculation, our system stands out from previous work on the following grounds:

- **Difficult primitive extraction**. The scene is filmed from far range and the images are noisy. Contour extraction is then a difficult operation and the contour chains obtained are not stable from one image to the other. Contrary to other approaches, we have only few prominent points to extract and in addition they may be poorly localized. Besides, it is out of the question to be able to properly extract junctions.
- **Inaccurate modelling**. Since it was built from architectural maps, the modelling of the architectural scene considered deviates significantly from the truth at certain places. This, along with unreliable contour data, explains why we have improved our calculations by the use of the robust technique of M-estimators.

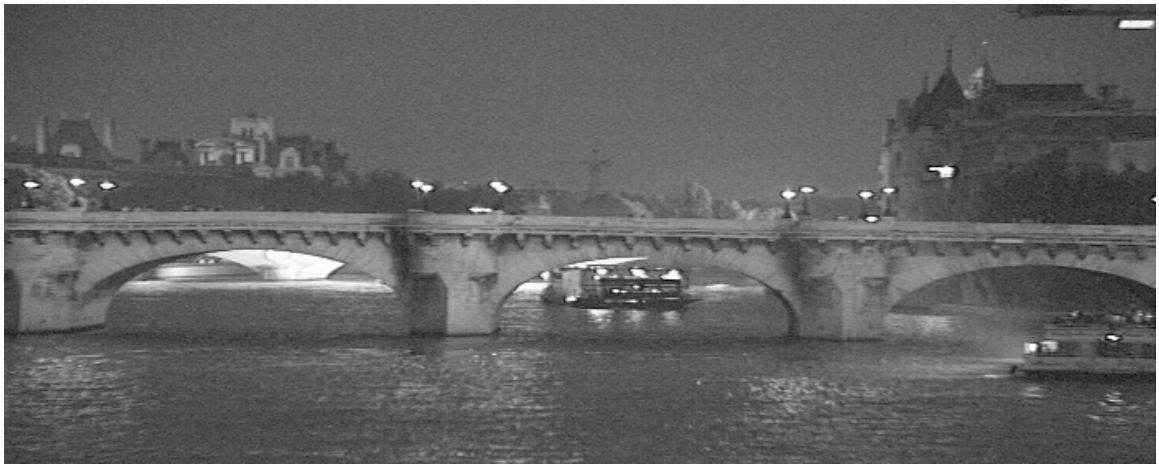


Figure 1: The 160^{th} image of the sequence.

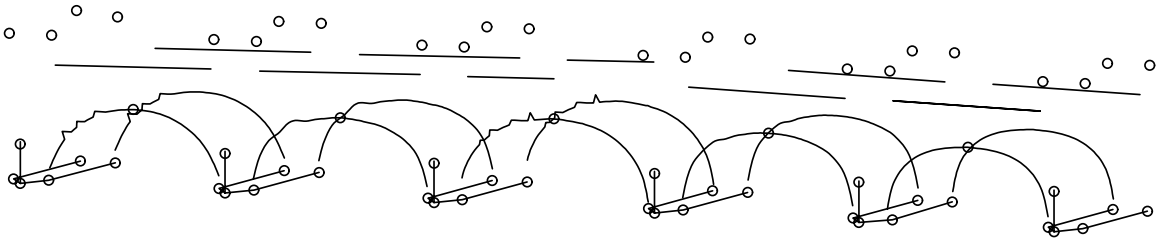


Figure 2: The complete wireframe model of the bridge. The poor quality of this modelling is quite apparent at certain places. Small circles show the location of feature points used for 2D-3D primitive matching (those that seemingly hover over the bridge correspond to street lamps).

- **No structuring of primitives**. In outdoor environments, the primitives extracted are often points, sometimes lines and curves. However, only few more complex features such as corners or junctions are present. Moreover, the features extracted are not topologically structured, making it impossible to use a structural approach to minimise the problems of the initial matching between image and model primitives.
- **Large vision field**. We do not impose any *a priori* either on the distance of observation or on the position of the viewpoint. The "aspect graph" types of approach which propose model points likely to appear in the next image are thus excluded because these points change drastically as a function of the distance of observation.

1.3 Application: The Bridges of Paris

A simulation of the illumination of a large architectural set may be quite difficult to implement *in situ*, not to mention that it would be very expensive. Thus, recent years have seen computer simulations become more and more popular. Reliable global illumination softwares are now available, so trial-and-error lighting experiments can now be done almost routinely (at the cost of large computation times though). Our motivation came from the insertion of a model illuminated synthetically in its real environment. Decision-makers of the Paris city administration were willing to have a new lighting system for a number of bridges of the Seine around the "Ile de la Cité". They wanted to test several candidate illumination projects and be able to choose on computer simulations alone what project was the best. Most importantly, they also wanted to evaluate the influence of this illumination on the surrounding elements.

A 300-image sequence of the Pont Neuf, one image of which is shown in Fig. 1, was shot at dusk time from another bridge. The fact that the images are dark has a strong influence on the quality of the segmentation and consequently on the strategies used for the tracking. It also restricts the number of reliable features that we are able to use. The reader should keep this in mind when visually evaluating the final composition.

It should also be noted that the modelling of the bridge was done mostly manually using information from architectural maps. Fig. 2 shows that it is very unreliable at places, as the staircase effect on some of the arches reveals. Much better results could certainly be obtained with a laser modelling.

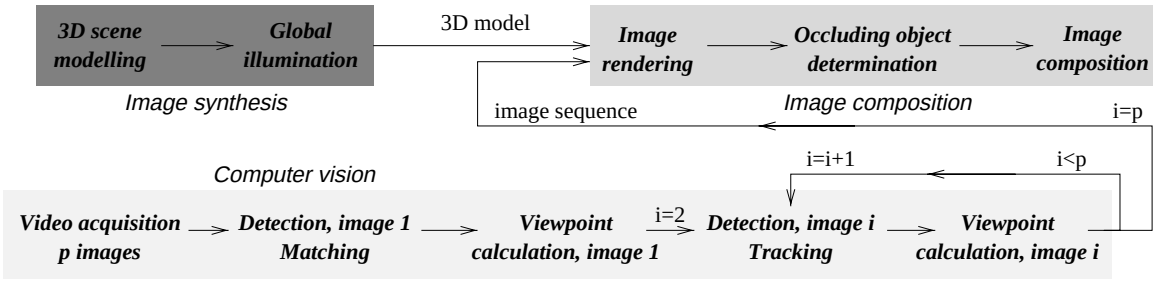


Figure 3: An augmented reality loop.

1.4 Paper Outline

In this paper, we present a framework for enriching a large image sequence of an outdoor urban environment and we apply these solutions to the sequence of the Pont Neuf. We assume no *a priori* knowledge on the extrinsic parameters of the camera system and the detection of features in images is autonomous, except for the first image and at certain key images where it is guided.

Our augmented reality loop works as follows. A set of non-coplanar features is detected in the first image of the sequence and matched semi-automatically to their corresponding features in the 3D model of the virtual object to inlay. These features are used to compute an estimate of the position of the camera with the scaled orthography approach of [8]. This initial guess is then refined by a robust minimisation of the projection errors. Once the viewpoint has been computed, a virtual object, rendered previously by a radiosity algorithm, can be embedded in the image. The features detected in the first image are then tracked over the sequence and matched to 3D features. This way, viewpoint position is determined for each image by applying the above procedure to the features tracked.

We now proceed as follows. In Section 2, the complete augmented reality loop is outlined in more details. The algorithms used for estimating pose are presented in Section 3 and the method for matching 2D image points with 3D geometric primitives in the first image is explained in Section 4. Section 5 presents the techniques we have developed for tracking features over the sequence of images. Results of the composition for the bridge sequence are then presented in Section 6. Finally, areas of current and future research are discussed in Section 7, before concluding. Note that a preliminary version of this work appeared in [3].

2 Augmented Reality Loop

Let us first outline the different building blocks of our augmented reality system. As can be seen on Fig. 3, this system really consists of three distinct parts. The first relies on computer vision tools and is the subject of this paper. The second part originates in image synthesis and in the third computer-generated and real information are combined to enhance the reality and produce a visually satisfying result. For sake of completeness we give a very short presentation of these last two parts here, but the interested reader should refer to [7] for more details on how to properly handle the interactions between real and virtual objects and light sources.

2.1 Image synthesis

The computer graphics part of our system aims at modelling the objects to inlay in the sequence of images and at globally illuminating these objects. Modelling the 3D scene really is a three-fold process: it involves

- modelling the geometry - here, a set of polygonal faces obtained from various architectural plans;
- modelling the properties of the materials assigned to the surfaces of the scene - this is based on *in situ* measures [6];
- modelling the positions and intensities of the virtual light sources, *i.e.*, imitating real source characteristics.

Objects in an urban environment are made of diffuse and specular materials. A radiosity algorithm is used to compute the diffuse inter-reflections between the surfaces [10]. A major feature of radiosity methods is that their computations are viewpoint-independent, so that illumination will not have to be computed for each image of the sequence.

2.2 Image composition

Once the viewpoint is determined for each image, the computer-generated images are rendered using a ray-casting algorithm. The main bottleneck of this step lies in the determination of the real occluding objects that stand between the camera and the virtual objects. The different virtual objects are then superimposed using composition operators (overlay, addition, multiplication) and the coherence of the composition is ensured.

In the image sequence of the Pont Neuf, a boat is stationed in front of the rightmost bridge pile in the last 40 images of the sequence and thus occludes the synthetic bridge. Currently, these occlusions are determined manually roughly every ten images and by interpolation in-between. But since occlusions may be more severe in general, we are currently investigating ways of automating their determination - see Section 7 and [4] for more.

3 Viewpoint Determination

To accurately place virtual objects in a video image, a necessary task is to compute the position of the camera that shot the image. When the computation is done using images of n scene points and their 3D counterparts, the problem is known as the *perspective n-point problem*.

Past work on the subject has seen two different kinds of solutions: closed-form and iterative methods. Closed-form solutions usually deal with very special configurations of scene primitives (four coplanar points, three orthogonal segments, a circle, ...). Iterative solutions use a number of features usually larger than closed-form methods but may be more robust both because measurement errors and image noise usually average out between features and because of the redundancy of information brought by the larger number of features.

We have implemented two methods for estimating pose in our system: the first is closed-form and based on the perspective inversion of four coplanar feature points [11] and the second is iterative and takes as input a set of non-coplanar points [8]. Though looking for coplanar features seemed a natural way to go in an urban environment, computations with such features have however turned out to be unstable and thus difficult to use in our application. We will thus mainly focus on the second method in this section but report a few interesting conclusions we drew after using the first.

3.1 Camera Setup and Calibration

The camera model used is the classical pinhole model. 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 reason why the computation of the intrinsic parameters has been clearly separated from viewpoint determination is that the number of features that we will be able to detect and track over the sequence is usually small (typically, between 10 and 20), and in any case insufficient to perform a complete calibration.

We assume that our camera is centered at point O and that the image plane is located at $z = f$, with f the focal length. A scene with feature points $M_0, \cdots, M_n$ is in the camera field of view, with the coordinate frame (M_0u, M_0v, M_0w) attached to it. The coordinates of the point M_i in the object frame are assumed to be known, as well as the image coordinates (x_i, y_i) of the projection m_i of M_i. Call $\mathcal{W}$ the matrix transforming world coordinates into viewing coordinates,

$$\mathcal{W} = \mathcal{R}\mathcal{T},$$

where $\mathcal{R}$ is the 3×3 rotation matrix and $\mathcal{T}$ is the 3×1 translation matrix. The perspective projection matrix $\mathcal{M}$ is then obtained from $\mathcal{W}$ as follows:

$$\mathcal{M} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \mathcal{W}.$$

Determining viewpoint then consists in computing the six parameters $a_1, \cdots, a_6$ of matrix $\mathcal{W}$, *i.e.*, the three parameters of the rotation matrix $\mathcal{R}$ (for instance the Euler angles) and the three parameters of the translation matrix $\mathcal{T}$.

3.2 The Method of Ferri *et al.*

The first method we have implemented for estimating pose is the exact perspective inversion of four coplanar points, which follows from the invariance of the cross ratio of four collinear points [11]. We thoroughly tested this method of determining viewpoint, but it turned out to be highly unreliable in our experiments:

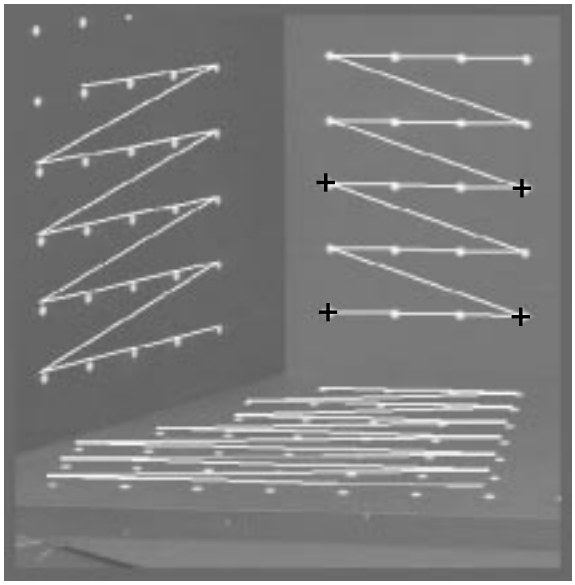


Figure 4: Viewpoint determination with Ferri's method given four perfectly coplanar points on a plane of a calibration pattern and their 2D correspondents.

- When applied to a perfectly modelled calibration pattern as shown in Fig. 4, the exact perspective inversion method gives a viewpoint for which the four coplanar points used are perfectly reprojected. But as one moves away from those points, the projection error grows larger and larger.
- If in addition we consider an environment for which the model is noisy, then the viewpoint computed with Ferri's method can only serve as a starting point for an iterative method that minimises the projection errors. Unfortunately, since the model of our bridge is highly biased, this starting point is so far from the actual viewpoint that convergence may be hard to attain. As an example, Fig. 5.a shows the initial estimate computed with Ferri's method on four coplanar points and Fig. 5.b displays the result after 52 iterations of minimising projection errors. After if the result after minimisation is much closer to the truth, we are still pretty far from the actual viewpoint position, as may be seen for instance by looking at the leftmost pile of the bridge.
- If more than four coplanar points are available, some techniques may be used to select the best 4-point configurations and to properly combine the results obtained for each configuration, as we have described in [3]. The criteria used for this selection are aimed at ensuring the stability of perspective inversion, and were based on considerations on the length of the diagonals and the angle between the diagonals. We also evaluated situations when a large number of features was available, some of them being coplanar, applying Ferri's method to the coplanar points and including the other points in the computation of the projection error. But even when 20 feature points can be used, 10 of them being coplanar, the estimate obtained this way was far from the true viewpoint position, though the results have improved.

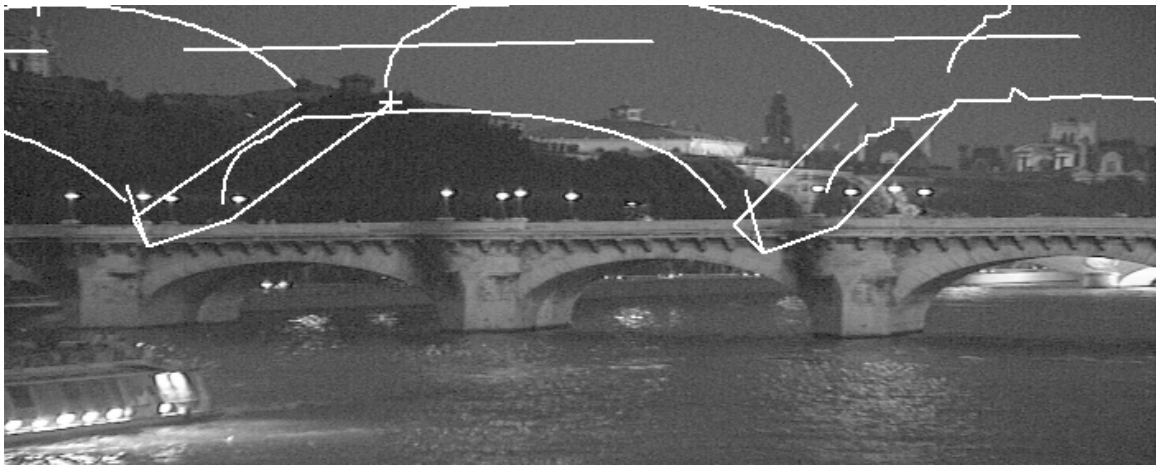
The unperfect initial viewpoint provided by Ferri's method, along with our biased model, has led us to use a different method to determine viewpoint, getting rid of the need for the coplanarity hypothesis.

3.3 The Method of DeMenthon and Davis

Consider the setup of Fig. 6. The rotation matrix $\mathcal{R}$ is the matrix whose rows are the coordinates of the unit vectors $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ of the camera system expressed in the object coordinate system (as usual, $\mathbf{i}$ is along the x axis, $\mathbf{j}$ is along the y axis and $\mathbf{k}$ is along the z axis). To compute the rotation, only the vectors $\mathbf{i}$ and $\mathbf{j}$ are needed, $\mathbf{k}$ being the cross product $\mathbf{i} \times \mathbf{j}$. The translation vector $\mathcal{T}$ is the vector $\overrightarrow{OM_0}$, whose coordinates are X_0, Y_0, Z_0. If point M_0 is a visible feature of the scene with image point m_0, the translation vector is aligned with vector $\overrightarrow{Om_0}$ and equal to $\frac{Z_0}{f}\overrightarrow{Om_0}$. Globally, object pose is thus well-defined once $\mathbf{i}, \mathbf{j}$ and Z_0 have been computed.

The main idea behind the algorithm described in [8] is to first compute an approximate pose under the assumption that the image points have been obtained by a scaled orthographic projection (called Pose from Orthography and Scaling - POS for short). The next iterations consist in shifting the feature points of the object in the pose

a.



b.

Figure 5: a. Initial estimate provided by the exact perspective inversion of four coplanar points. b. Refined estimate after 52 iterations.

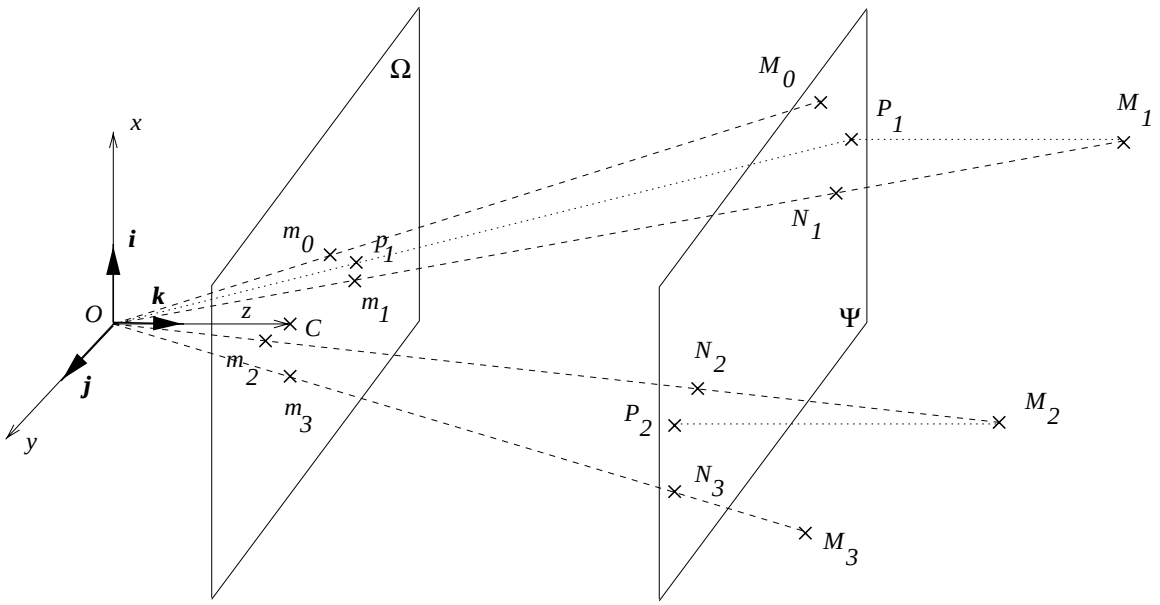


Figure 6: Setup for the method of DeMenthon and Davis.

obtained to the lines of sight, obtain a scaled orthographic projection of these shifted points, and compute an approximate pose using this projection (algorithm called POSIT - POS with iterations). Only a few iterations are needed to converge to an accurate pose.

3.3.1 Perspective Projection

Call Ψ the plane through M_0 and parallel to the image plane. The viewline through M_i intersects Ψ in N_i and M_i projects orthogonally onto Ψ in P_i. Thus

$$\overrightarrow{M_0M_i} = \overrightarrow{M_0N_i} + \overrightarrow{N_iP_i} + \overrightarrow{P_iM_i}.$$

It is clear that

$$\overrightarrow{M_0N_i} = \frac{Z_0}{f}\overrightarrow{m_0m_i}.$$

If C is the intersection of the z axis with the image plane, then the two vectors $\overrightarrow{N_iP_i}$ and $\overrightarrow{Cm_i}$ are proportional:

$$\overrightarrow{N_iP_i} = \frac{\overrightarrow{M_0M_i}.\mathbf{k}}{f}\overrightarrow{Cm_i}.$$

Now we want to take the dot product of $\overrightarrow{M_0M_i}$ with $\mathbf{i}$. The product $\overrightarrow{P_iM_i}.\mathbf{i}$ is zero. The product $\overrightarrow{m_0m_i}.\mathbf{i}$ is $x_i - x_0$ and $\overrightarrow{Cm_i}.\mathbf{i}$ is x_i. We thus end up with the first fundamental equation of perspective projection:

$$\overrightarrow{M_0M_i}.\frac{f}{Z_0}\mathbf{i} = x_i(1+\varepsilon_i) - x_0,$$

where

$$\varepsilon_i = \frac{1}{Z_0}\overrightarrow{M_0M_i}.\mathbf{k}.$$

Similarly,

$$\overrightarrow{M_0M_i}.\frac{f}{Z_0}\mathbf{j} = y_i(1+\varepsilon_i) - y_0.$$

Writing $\mathbf{I} = \frac{f}{Z_0}\mathbf{i}$ and $\mathbf{J} = \frac{f}{Z_0}\mathbf{j}$, the system of equations is then:

$$\begin{cases} \overrightarrow{M_0M_i}.\mathbf{I} &= x_i(1+\varepsilon_i) - x_0, \\ \overrightarrow{M_0M_i}.\mathbf{J} &= y_i(1+\varepsilon_i) - y_0. \end{cases} \tag{1}$$

3.3.2 Scaled Orthographic Projection

Scaled orthography is an approximation to perspective projection, where assumption is made that the depths Z_i of the different points of the scene do not vary much and can be set to the depth of a reference point M_0. The image of a point M_i is thus a point of the image plane Ω with coordinates

$$x'_i = f\frac{X_i}{Z_0}, \quad y'_i = f\frac{Y_i}{Z_0}.$$

Now we may work out a construction somewhat similar to the above. We have:

$$\overrightarrow{M_0M_i} = \overrightarrow{M_0P_i} + \overrightarrow{P_iM_i}.$$

The vector $\overrightarrow{M_0P_i}$ is $\frac{Z_0}{f}\overrightarrow{m_0p_i}$, where p_i is the image of P_i. The dot product of $\overrightarrow{m_0p_i}$ with $\mathbf{i}$ is $x'_i - x_0$ and the dot product $\overrightarrow{P_iM_i}.\mathbf{i}$ is zero. We thus have that:

$$x'_i = x_i(1+\varepsilon_i),$$

and similarly

$$y'_i = y_i(1+\varepsilon_i).$$

3.3.3 Pose Estimation

The basic idea behind the POS algorithm is that if values are given to ε_i, then System (1) is a linear system of equations the unknowns of which are the coordinates of $\mathbf{I}$ and $\mathbf{J}$. Once we have $\mathbf{I}$ and $\mathbf{J}$, $\mathbf{i}$ and $\mathbf{j}$ are obtained by normalising $\mathbf{I}$ and $\mathbf{J}$, and Z_0 is obtained from the norm of one of the vectors: $Z_0 = \frac{f}{\|\mathbf{I}\|}$. See the next section for the scheme used for solving System (1).

As we have seen, solving for pose by assuming that ε_i is given amounts to finding the pose for which the points M_i have as scaled orthographic projections the image points with coordinates $[x_i(1+\varepsilon_i), y_i(1+\varepsilon_i)]$.

Initially, one can set $\varepsilon_i = 0$. The POS algorithm then solves System (1) for $\mathbf{i}, \mathbf{j}$ and Z_0. From these values, vector $\mathbf{k}$ is computed and a new value of ε_i is obtained:

$$\varepsilon_i = \frac{1}{Z_0} \overrightarrow{M_0 M_i}.\mathbf{k}.$$

With this new value, System (1) is solved again. The repetition of the above process is the core of the POSIT algorithm. Several iterations are sufficient to converge to an accurate pose estimation.

3.3.4 Solving the Linear System

Suppose then that the values of ε_i have been computed at the previous step. Writing System (1) for the n points M_i of the scene, we have two matrix equations:

$$\mathcal{A}\mathbf{I} = \mathbf{a}, \quad \mathcal{A}\mathbf{J} = \mathbf{b},$$

where $\mathbf{a}$ (resp. $\mathbf{b}$) is the vector with i-th coordinate $x_i(1+\varepsilon_i)$ (resp. $y_i(1+\varepsilon_i)$) and $\mathcal{A}$ is the matrix of the coordinates of the points M_i in the object frame.

If we have at least three visible points other than M_0, such that these points are not coplanar, then matrix $\mathcal{A}$ has full rank and one can compute its pseudo-inverse $\mathcal{B}$ using a Singular Value Decomposition of $\mathcal{A}$. This justifies what we do in Section 4.

3.4 Refining the First Estimate

Once the method of DeMenthon and Davis has given us a perspective projection matrix that is more or less close to the correct one, the idea now is to use this matrix as a starting point to iteratively converge to the best possible matrix.

The robustness of the above estimation must be improved because errors or outliers in the tracking stage will severely affect the accuracy of the viewpoint location. Such a process is needed for at least two reasons: mismatches may occur during the tracking process (for instance, two neighbouring features may overlap) and will be propagated along the sequence. Besides, the 2D features are not detected accurately due to the bad quality of our images. Hence a robust technique for the viewpoint computation is needed [17] because it is well-known that least-square estimations are very sensitive to noise. The most popular estimators are the M-estimators and the least median of squares. The advantage of the M-estimator is that it can be reduced to a weighted least square problem. It is robust to bad localisation errors but it is not really robust to false matches (outliers). On the other hand, the least median of squares is robust to outliers but it cannot be reduced to a formula. In our application, we mainly have to deal with bad localisations, so we focus on the use of the M-estimator.

3.4.1 Improving Robustness

Suppose that $a_1, \cdots, a_6$ are the six parameters of displacement, *i.e.*, the parameters of matrix $\mathcal{W}$. The optimisation is done only on these six parameters. Then, if (u'_i, v'_i) are the pixel coordinates of the projection of a point M_i (obtained by $\mathcal{M}$) and (u_i, v_i) its pixel coordinates as given by the feature detector, we want to minimise the function:

$$g(a_1, \cdots, a_6) = \sum_{i=1}^{n} \rho(r_i),$$

where n is the number of points considered, $r_i = r_i(a_1, \cdots, a_6) = d((u_i, v_i), (u'_i, v'_i))$ is the residual and d denotes the Euclidean distance. Note that each point could be weighted for instance by a measure of confidence σ_i given by the feature detector, in which case we would minimise the function $\sum_{i=1}^{n} \frac{1}{\sigma_i} \rho(r_i)$.

The function ρ is chosen according to statistical considerations [23]. If Λ is the derivative of ρ, then the above amounts to solving the following system:

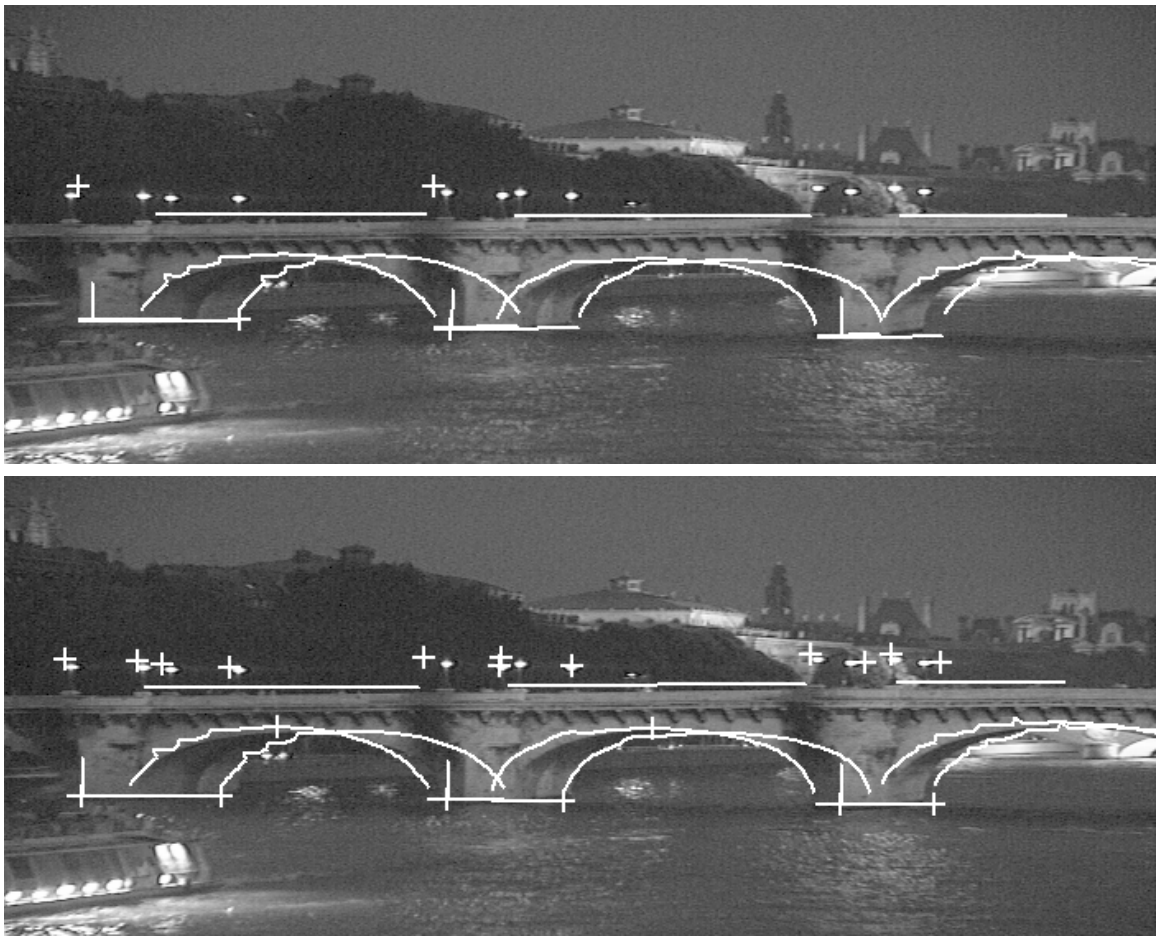
$$\sum_{i=1}^{n} \Lambda(r_i) \frac{\partial r_i}{\partial a_k} = 0, \quad k = 1, \cdots, 6.$$

This shows that Λ may be seen as a weighting or influence function. Different probability distributions can be used (normal, doubly exponential), but an appropriate choice is the Cauchy-Lorentz distribution, with:

$$\rho(z) = \log\left(1 + \frac{z^2}{2}\right), \qquad \Lambda(z) = \frac{z}{1 + \frac{z^2}{2}}.$$

That is, the more the points deviate from the model, the less importance they have in the calculation. The Cauchy-Lorentz distribution thus allows in practice to eliminate the more distant points.

3.4.2 Minimising the Function g



a.

b.

Figure 7: a. Initial estimate with the method of DeMenthon and Davis applied to four non-coplanar points. b. Refined estimate after 23 iterations using Powell's method.

Numerous works have been devoted to the minimisation of a multivariate function starting from an initial position A_0 - see in particular [20] and [26]. This complex problem is generally decomposed into several simpler ones: a set of 1-dimensional minimisations done successively in several directions. The main difference between the different methods lies in the choice of these directions. For instance:

- Powell proposed a very simple algorithm, said of *quadratic convergence*, which converges to a local minimum: the direction used at step i is $A_i - A_{i-p}$, where p is some positive integer.
- The conjugate gradient method minimises at step i in the direction $-\nabla g(A_i)$.

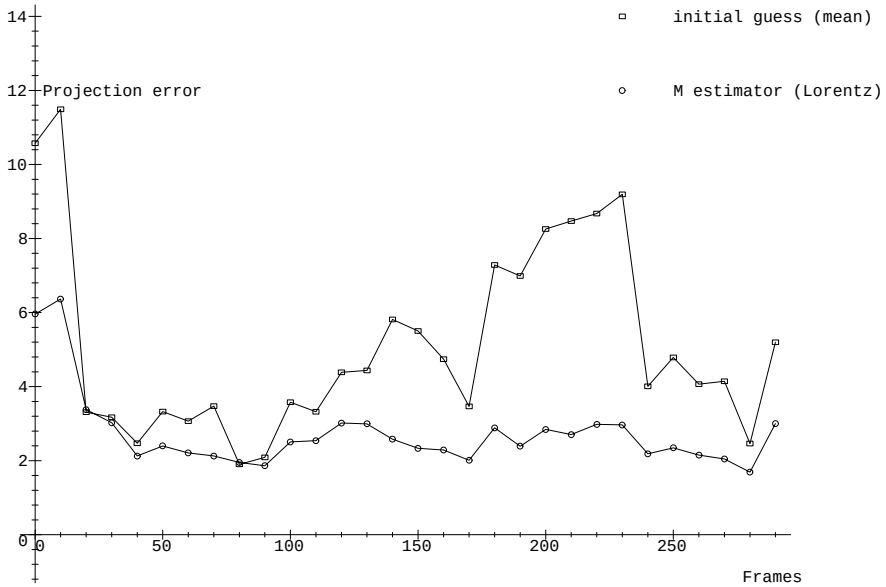


Figure 8: Projection errors in pixels after the initial estimate (squares) and after refinement with Cauchy-Lorentz (circles), using the method of DeMenthon.

Powell's method may be very useful in situations where the gradient of g is unknown. In our case, we know the derivative of ρ, so the gradient of g is also known. Both of these methods have been implemented, but Powell's method turned out to be more efficient in our application, using the Cauchy-Lorentz distribution seen above.

3.4.3 Results

Consider Fig. 7. As can be seen, the initial estimate given by the method of DeMenthon and Davis is much better than the one computed with Ferri's exact perspective inversion (Fig. 5) and is not hindered by the noisy model of the bridge. And it takes less iterations to converge to a viewpoint position accurate enough for our application. To see where refinement with Cauchy-Lorentz has improved the projection, the reader should direct his attention for instance to the bottom of the back arches of the bridge.

When dealing with a new image of the sequence, we may either directly use the iterative pose estimation method above to compute a first estimate of viewpoint position or compute such an estimate by minimisation using the viewpoint calculated in the previous image and the information provided by the tracker. Both methods have proved reliable in our experiments. Generally speaking, the second would however be difficult to apply on a sparse image sequence, with larger rotation angle between two images. In such a situation, tracking would in general be possible but the estimate given by the previous position may not be sufficient for the optimisation to converge. The images displayed in the rest of the paper have been computed using the first strategy.

Figure 8 is a plot of the mean projection error for each image of the sequence, taking into account for each image all the primitives visible in that image. The curve with boxes represents the mean of the projection error after the computation of the initial guess, whereas the curve with circles shows the projection error after the robust estimation of the viewpoint using the Cauchy-Lorentz estimator. One conclusion of this plot is that considering the inaccurate detection and the poor quality of the modelling, we are still able to keep the projection errors at a low and quite acceptable level.

4 2D-3D Primitive Matching

Here is how we achieve the matching between 2D and 3D primitives in the first image of the sequence. A number m of interest points (between 10 and 20) is extracted in this image. Currently, these points are extracted manually. Obtaining them automatically seems to be computationally prohibitive even when the images are of good quality. In the ensuing images of the sequence, interest points are obtained by tracking those found in the previous image, except when the number of such points falls below a minimum or when these points are not well spread over the object. Indeed, the scene is not viewed in its entirety from a single viewpoint, so that as the camera moves, some interest points disappear and some other come into the field of view. There are thus moments when the user is

Figure 9: Contour chains.

requested to point some new feature points in the image. These new 2D points have to be automatically matched with the 3D points of the model.

4.1 Pruning the Hypotheses

To solve the problem of matching 2D and 3D primitives in the first image of the sequence, we have the following at our disposal:

- a wireframe model of the scene (see Fig. 2) or to be more precise a complete wireframe model of an object that may not be fully visible in any image of the sequence.

  We have manually extracted from this model n distinguished feature points (roughly 70): corners, top point of each arch, centres of the lamps... They are shown on Fig. 2.

- m 2D points extracted manually from the first image corresponding to feature points of the model.

Also we suppose that at any moment there are at least four 3D model points that we know are present in the current image.

Now, four non-coplanar 3D feature points and their 2D matches are sufficient to compute the pose of the scene. From these, we can generate $\frac{m!}{(m-4)!}$ potential pose estimates. To prune these estimates and find the best one, we apply successively two heuristics that have proved reliable for the application of the bridges of Paris:

- Heuristic 1 is classical: keep only those pose estimates for which the sum of the projection errors for the four 3D points is smaller than some threshold. We usually keep between 100 and 400 estimates at this stage.

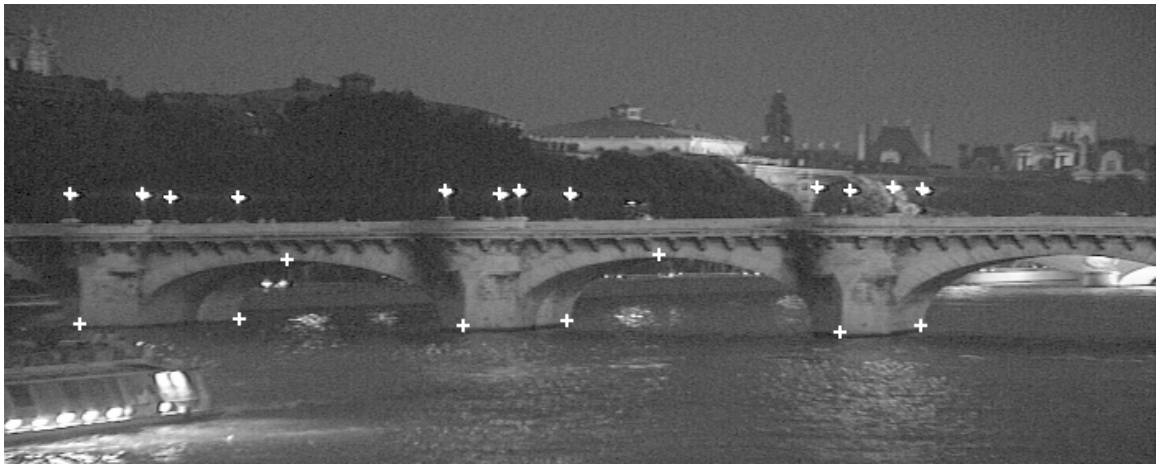
- Heuristic 2 is similar to the first except that the calculation of the projection error is achieved on all feature points of the model. When computing this error, we penalise those model points for which we have not been able to locate a 2D correspondent in a given window. We keep the best estimate after this step.

4.2 A Few Words on the Use of Contour Chains

Initially, we planned to use the contour image of the scene to evaluate the adequation of the projection of the model of the bridge with the real bridge. Unfortunately, since the sequence has been shot at night, the images are not sufficiently contrasted and it is very difficult to obtain a significative gradient image. The contour chains built from these gradients are “unstable”: they are very dependent on viewpoint position and very sensitive to the threshold used - compare Figs. 9 and 11.a. They also lead to poor evaluations when superimposing the image of chains of strong gradient and the projection of the model.

A high threshold produces few chains which do not generally correspond to parts of the model, since the bridge is not sufficiently “contrasted” with respect to the surrounding environment. A low threshold produces many chains for which any viewpoint estimate is equally correct: it is easy to find a chain in the neighbourhood of a contour of the reprojected model. Our images are not good enough to obtain correct contour chains and to allow the evaluation of each viewpoint estimate by comparing the entire edge contours of the object with the image edges as in [12]. To be efficient in our context, this method would need to be coupled with some *a priori* knowledge on the scene, something we wanted to avoid.

a.



b.

Figure 10: Matching 2D and 3D primitives. a. The first image of the sequence, including 20 points extracted manually from the contour map. b. Result of computing viewpoint by automatic 2D-3D primitive matching.

4.3 Result Images

The first image of the scene contains 20 of the n distinguished feature points of the bridge model and is shown in Fig. 10.a. The result of computing the viewpoint by automatic matching of 2D-3D primitives on the first image of the sequence is given on Fig. 10.b.

5 Feature Tracking

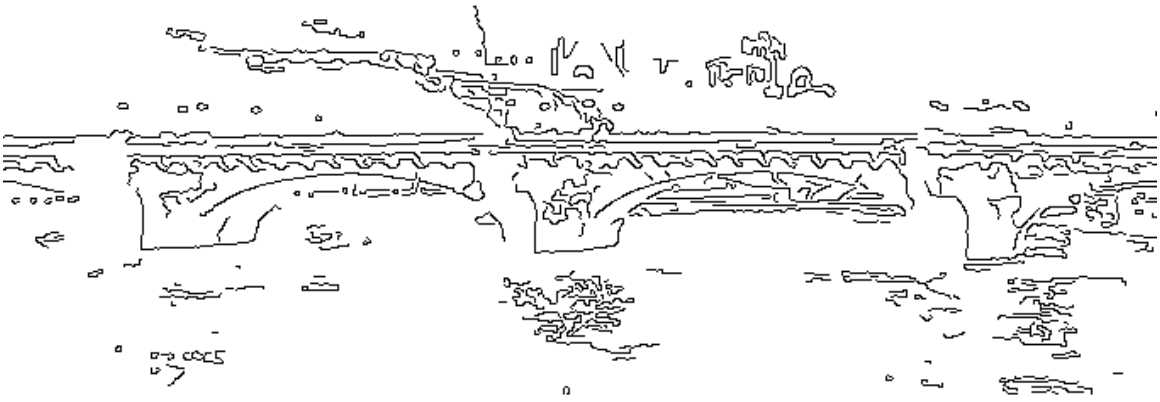
We describe in this section the tracking techniques that we implemented in our system. Two kinds of features will be tracked: points, used to determine the viewpoint, and curve segments (the arches). The arches are not used directly as curved segments for viewpoint determination in the present version of the system, as could be done with the method of [16], but only because they provide interesting feature points (the top of each arch for instance - see point 4 of Fig. 11.c). In fact, we will see that even points are tracked using a curve-based tracking tool.

Features to be tracked are selected before execution in the first image and the corresponding 3D points in the model are retained. Then, they are tracked along the sequence in the autonomous way described below. Because of the length of the sequence, new features to be tracked appear as the sequence evolves. The tracking process is thus stopped momentarily to select these new features.

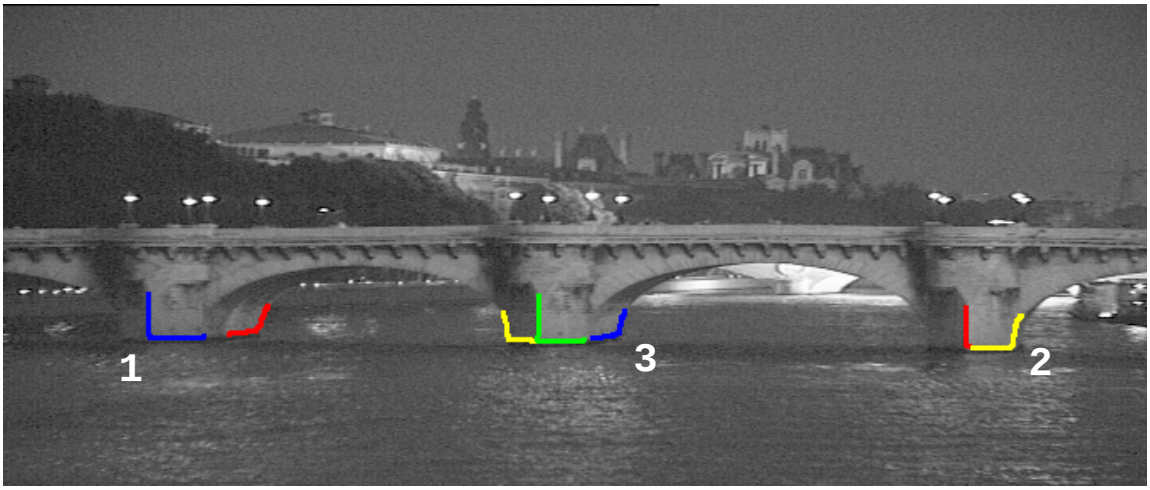
The sequence we consider here was shot in nighttime. Consequences are that the segmentation is very poor and that feature points cannot be tracked using the contour map. Hence, we use a curve-based tracking tool capable of tracking interest curves containing the feature points used for the pose computation. For instance, as shown in Fig. 11, instead of tracking the point at the basis of the bridge pier, we track the curve drawn in red and we infer the point position by computing the corner best fitting the curve. As shown in the examples, sufficiently accurate features can then be obtained.

Other points than the bases of the piers can be of great interest for the pose computation: the street lamps on the

a.



b.



c.

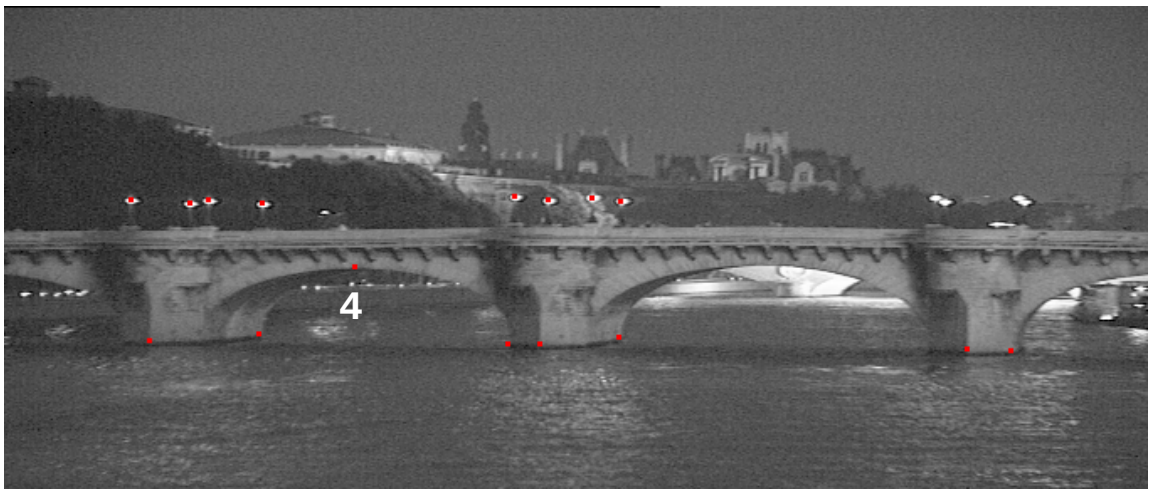


Figure 11: Tracking features. a. The contour map for the 60^{th} image. b. The curves obtained with the snakes techniques. c. The points obtained after approximating these curves with corners.

bridge whose 3D position is known. The light emitted by these lamps always appears as white blobs in the images whatever the viewpoint and can be tracked using correlation methods as described below.

We now describe in details the two tracking techniques used: correlation-based tracking and curve-based tracking.

5.1 Correlation-Based Tracking

As soon as a new street lamp appears, the centre of the light bulb is detected in the following way: the user draws a small polygon enclosing the light bulb; because the light appears as a white blob on the image, the bulb can be outlined using a simple threshold. The centroid of this blob is then considered as the image of the 3D centroid of the bulb.

The lamp is then tracked along the sequence using correlation. Given a small square enclosing the lamp in the current image, the correlation process computes the translation best fitting this window in the next image. The centroid of the white blob in this new window gives the position of the feature in the next image.

5.2 Curve-Based Tracking

Other points than the lamps are tracked using a curve tracker. The tracker we designed is described in earlier works [2]. It is made up of the following two steps:

- a *prediction* step: an approximation of the 2D motion field is computed iteratively from the normal optical flow on the whole curve;
- a *convergence* step: from the predicted curve, an active contour [15] converges towards the nearest contour which is generally the homologous one. It must be noted than some strategies have been developed to best ensure this convergence.

This tool is used to track curved features like the arches of the bridge. In order to detect feature points, we take advantage of the fact that most of them are the corner points of contour lines. Once an interest point has been chosen, a sufficiently contrasted curve containing this point is outlined using for instance a snake technique. This curve is then tracked along the sequence using the tool described above. Then, we search for the angular point of each of these curves: for each point M_i belonging to such a curve, the two lines best fitting $M_{j\{j<i\}}$ and $M_{j\{j\geq i\}}$ are computed. The point giving rise to the best fitting score is retained and the intersection of these two lines is considered as the searched feature point (point 1 on Fig. 11.b). Note that this method is used to track angular points between lines or between curves. In fact, if the curve containing the feature point is sufficiently small, the approximation of the curve with a line is quite satisfactory (points 2 and 3 on Fig. 11.b). Such a method turns out to be very robust to noise since most feature points are not easily detectable in the images.

Nevertheless, some of these angular points do not correspond to 3D points of the model because they are the intersections of a vertex of the bridge with the river surface (points 1, 2, 3 in Fig. 11.b). The elevation of these points can however be roughly estimated because the dimensions of the bridge are known. The elevation of the river surface is then refined in the following way: from the upper estimation h_0, we compute for each elevation h in the $[h_0 - 50 \text{ cm}, h_0 + 50 \text{ cm}]$ range (regularly sampled) the viewpoint corresponding to the tracked set of points, when the elevation of the points on the river surface is set to h. The correct elevation is the one for which the projection error is the smallest.

6 Results and Discussion

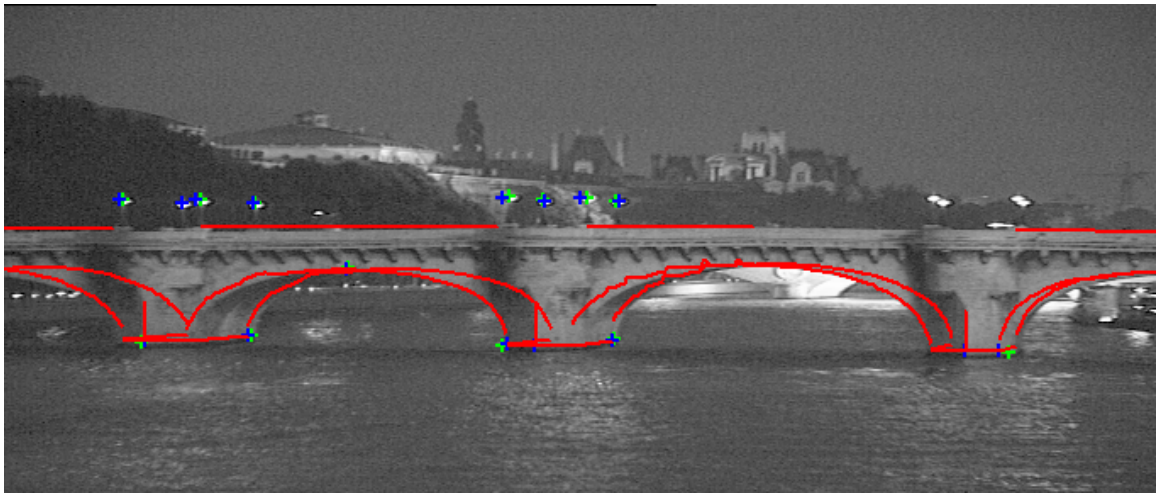
In the previous sections, we have examined the different components of our system separately and discussed the algorithmic choices we made. We now turn our attention to the augmented reality loop as a whole, present results of applying it to the Pont Neuf sequence and discuss its performance.

A quantitative analysis of the performance of the feature tracking and pose estimation algorithms on synthetic data and for various types of features is given elsewhere [24].

6.1 The Augmented Pont Neuf Sequence

The entire 300-image sequence has been augmented with the procedure described in this paper. Fig. 12 describes the composition of the 60^{th} image: Fig. 12.a shows the original image on which we have superimposed the projection of the model using the viewpoint computed and Fig. 12.b shows the final result of the composition. Figure 13 shows similar results for the 100^{th} image.

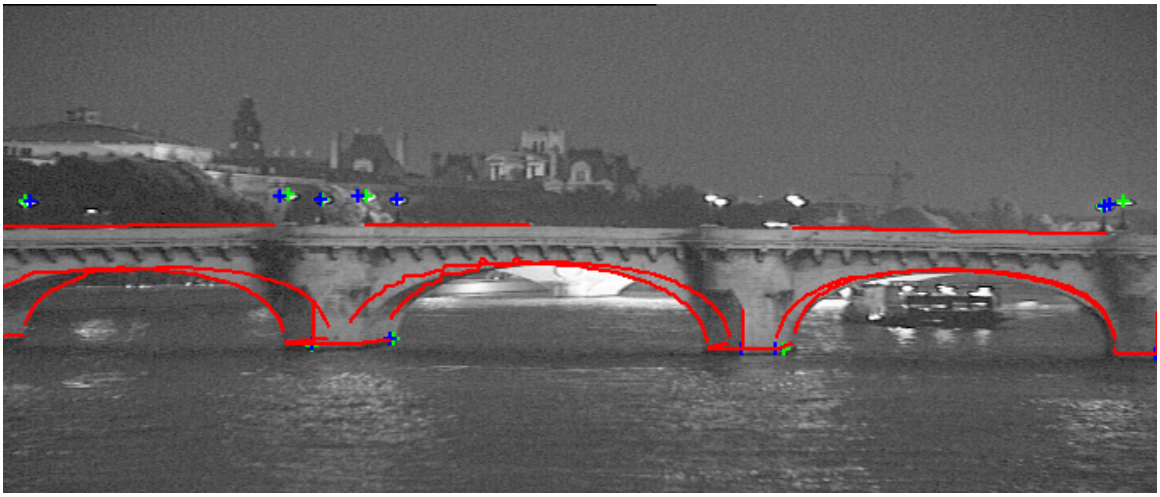
a.



b.

Figure 12: Composition for the 60^{th} image. a. The red curves correspond to the model projected onto the image using the viewpoint computed. The 2D features used for viewpoint determination appear in green and the projection of the corresponding 3D points are in blue. b. Result of the composition.

a.



b.

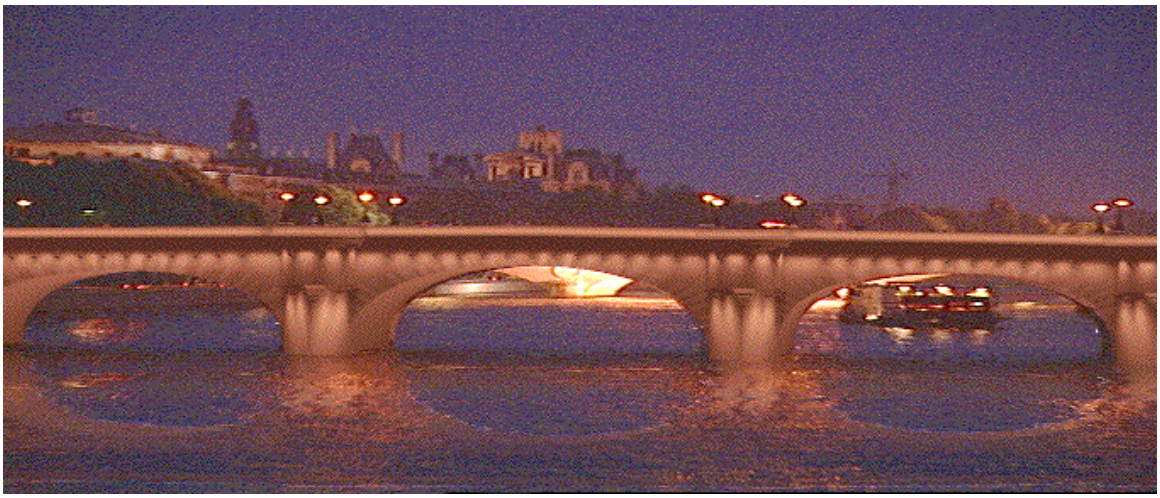


Figure 13: Composition for the 100^{th} image of the sequence.

Several MPEG movies are available on our WWW server at the address:

$$\texttt{http://www.loria.fr/isa/Bridges/}\left\{\begin{array}{c}\texttt{tracking}\\ \texttt{projection}\\ \texttt{augmented}\end{array}\right\}\texttt{.mpg},$$

where `tracking` stands for the sequence showing the features used for the tracking, `projection` for the original sequence with the model superimposed and `augmented` for the fully augmented sequence, with the bridge illuminated synthetically (each is roughly 2.4 MB in size).

6.2 Discussion

Some comments are in order. The above results prove that our system can handle poor quality images and an inaccurate modelling of the object to inlay. In addition, it can account for a large vision field and a rotating motion: in the Pont Neuf application, the motion is purely panoramic and the total camera rotation is roughly 20 degrees (the bridge is 90 meters long and the viewpoint is located 300 meters away from the bridge). Our experiments show that the tracker is able to deal with an apparent motion of the primitives of up to 20 pixels between two consecutive images.

Considering the conditions that we have already mentioned (bad segmentation due to darkness, poor modelling), the overall visual impression is quite satisfactory. The reader who gets a chance to see the entire augmented sequence will however witness a slight jittering effect, that may best be explained with the help of Fig. 14. The world coordinates are such that the z-axis represents the depth of field and the y coordinate is the elevation. In Fig. 14, we drew the evolutions of the x, y and z coordinates of a point P standing on the optical axis of the camera at a distance of one meter of the optical centre C. It appears that the evolutions of the x and z coordinates are very regular, but that of y is highly singular, thus yielding this jittering. Incidentally, our analysis of the modelling of the bridge showed that it is quite correct in the x and z directions, but inconsistent in the y direction. For instance, we were never able to correlate both the lamps on the parapets and the arches at the same time, due to this incoherent modelling in the y direction. Also, it should be noted that the lens of the camera used had a distortion that we were not able to fully correct.

6.3 System Performance

Although latency and execution speed are not central to the present application, they usually are key issues in augmented reality research. We shall thus say a few words here about the computational requirements of our system.

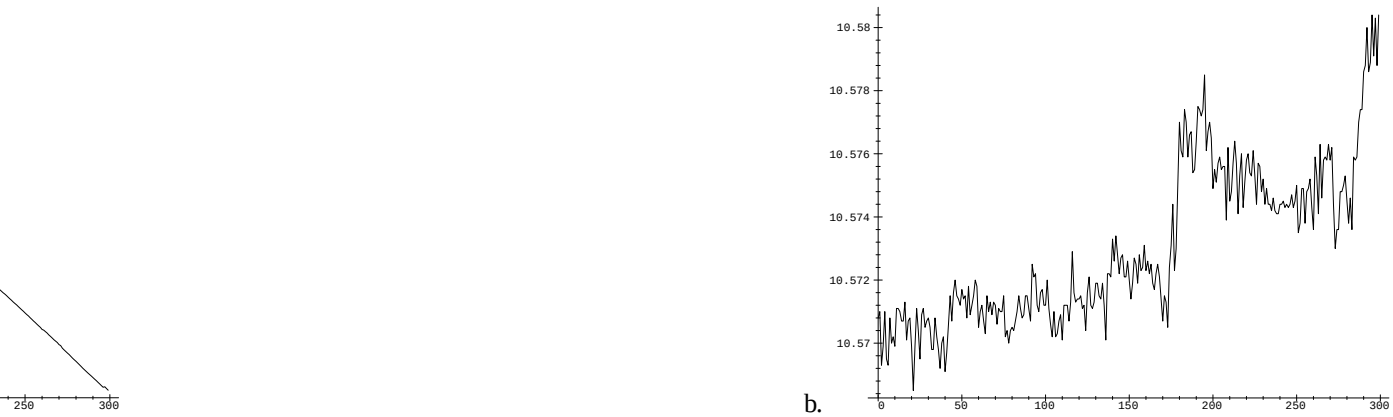
Without any kind of optimisation, the only algorithmic part that does not currently achieve real time is the 2D-3D primitive matching for the first image. On a Sun Ultra 1 Model 10 (the following computation times have all been measured on this machine with Rational Software's Quantify), this matching takes 34.4 seconds for a set of 17 2D points and 70 3D points. The determination of viewpoint with the method of DeMenthon and Davis applied to 20 input points takes less than one millisecond, while optimisation to converge to an accurate viewpoint position takes 92 milliseconds. And tracking a primitive of 30 points takes 97 milliseconds for the prediction step and 70 milliseconds for the convergence step. In other words, apart from the initialisation on the first image of the sequence, the tools presented in this paper are amenable to real-time performance. Of course, global illumination algorithms are known to have large computation times. And indeed, notably because of the high number of light sources involved, more than 50, the rendering of the bridge we inlayed in our image sequence took more than 7 hours to compute.

7 Conclusion and Future Research Directions

We have presented in this paper a complete augmented reality forward loop. An application to the simulation of a lighting project of a bridge in Paris has been explained and results are shown that prove the validity of our approach. Emphasis has been put on the computer vision tools used to determine viewpoint and to track a set of features in a sequence of images of an outdoor urban environment. The method overcomes several weaknesses of previous approaches, especially as far as automatic and robust viewpoint determination is concerned.

We are currently working on several features that would robustify and enhance our augmented reality system:

- **Perspective inversion of curved segments.** If curved arcs are present in the scene, they may be used to further converge to the position of the viewpoint. Except in the case of circles [11], an exact perspective inversion for ellipses is not possible. We may thus use an iterative algorithm developed by [16], taking as initial estimate the perspective projection matrix obtained for a number of (coplanar or non-coplanar)



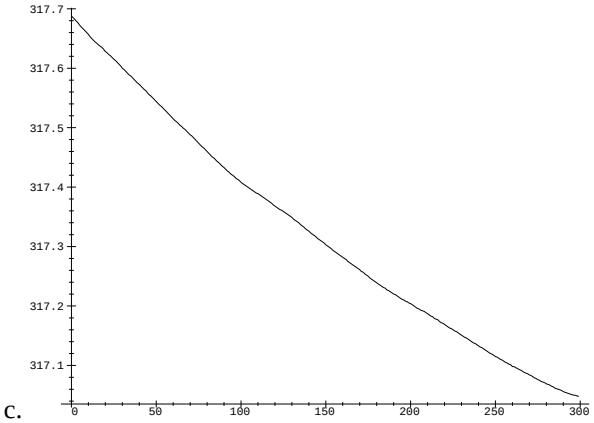


Figure 14: Evolutions of the x, y and z coordinates of a point standing on the optical axis of the camera. Abscissa represents frame number and ordinate is measured in meters.

points. We did not use this feature in the application described because the modelling of the arches was too inconsistent.

- **Smoothing the sequence.** We do not currently ensure any kind of global coherence of the sequence, meaning that it is not smoothed to avoid jittering for instance (tracking provides a local coherence). This could very well be accounted for by a technique based on Kalman filtering for instance. Also, the regularity of the camera motion could be used to improve the prediction of a feature position in the next image.

- **Occlusions.** One of the limitations of the present system is that the mask of the occluding objects must be determined by hand. Hence we are currently investigating an algorithm allowing the static occluding objects to be detected without 3D reconstruction. The underlying idea is to compare for each image point the optical flow of this feature with the theoretical flow generated by the 3D point of the virtual object that projects onto the feature point [4].

References

[1] Bajura, M., Fuchs, H., and Ohbuchi, R. (1992). Merging virtual objects with the real world: seeing ultrasound imagery within the patient. *Computer Graphics Proceedings, Annual Conference Series*, 26:203–210. Proceedings of SIGGRAPH'92.

[2] Berger, M.-O. (1994). How to track efficiently piecewise curved contours with a view to reconstructing 3D objects. In *Proceedings of 12th* ICPR *(International Conference on Pattern Recognition), Jerusalem, Israel*, volume 1, pages 32–36.

[3] Berger, M.-O., Simon, G., Petitjean, S., and Wrobel-Dautcourt, B. (1996). Mixing synthesis and video images of outdoor environments: application to the bridges of Paris. In *Proceedings of 13th* ICPR *(International Conference on Pattern Recognition), Vienna, Austria*, volume 1, pages 90–94.

[4] Berger, M.-O. (1997). Resolving occlusions in augmented reality: a contour-based approach without 3D reconstruction. In *Proceedings of* CVPR *(*IEEE *Conference on Computer Vision and Pattern Recognition), Puerto Rico*.

[5] Caudel, T. and Mizell, D. (1992). Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *Proceedings of Hawaii International Conference on System Sciences*.

[6] Cazier, D., Chamont, D., Deville, P., and Paul, J.-C. (1994). Modeling characteristics of light: a method based on measured data. In *Proceedings of the Second Pacific Conference on Computer Graphics and Applications*, pages 113–128.

[7] Chevrier, C., Belblidia, S., and Paul, J.-C. (1995). Compositing computer-generated images and video films: an application for visual assessment in urban environments. In Earnshaw, R. and Vince, J., editors, *Proceedings of* CGI*'95 (Computer Graphics International)*, pages 115–125. Academic Press.

[8] DeMenthon, D. and Davis, L. (1995). Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15:123–141.

[9] Ertl, G., Mueller-Seelich, H., and Tabatabai, B. (1991). Move-x: a system for combining video films and computer animation. In Purgathofer, W., editor, *Proceedings of Eurographics'91*, pages 305–314.

[10] Fasse, I., Paulo, S. S., Perrin, J.-P., and Paul, J.-C. (1994). Accurate synthesis images for architectural design. In *Proceedings of the Conference on Multimedia for Architecture and Urban Design*, pages 229–243.

[11] Ferri, M., Mangili, F., and Viano, G. (1993). Projective pose estimation of linear and quadratic primitives in monocular computer vision. CVGIP*: Image Understanding*, 58(1):66–84.

[12] Huttenlocher, D. and Ullman, S. (1990). Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212.

[13] Jancène, P., Neyret, F., Provot, X., Tarel, J.-P., Vézien, J.-M., Meilhac, C., and Verroust, A. (1995). RES: computing the Interactions between Real and Virtual Objects in Video Sequences. In *Proceedings of the* IEEE *Workshop on Networked Realities*, Boston, MA, pages 27–40.

[14] Kaneda, K., Kato, F., Nakamae, E., Nishita, T., Tanaka, H., and Noguchi, T. (1989). Three-dimensional terrain modelling and display for environmental assessment. *Computer Graphics Proceedings, Annual Conference Series*, 23:207–214. Proceedings of SIGGRAPH'89.

[15] Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: active contour models. *International Journal of Computer Vision*, 1:321–331.

[16] Kriegman, D. and Ponce, J. (1990). On recognizing and positioning curved 3D objects from image contours. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 12(12):1127–1137.

[17] Kumar, R. and Hanson, A. (1994). Robust methods for estimating pose and a sensitivity analysis. CVGIP*: Image Understanding*, 60(3):313–342.

[18] Maver, T., Purdie, C., and Stearn, D. (1985). Visual impact analysis - modelling and viewing the natural and built environment. *Computer and Graphics*, 9(2):117–124.

[19] Nakamae, E., Harada, K., Ishizaki, T., and Nishita, T. (1986). A montage method: the overlaying of the computer generated images onto a background photograph. *Computer Graphics Proceedings, Annual Conference Series*, 20:207–214. Proceedings of SIGGRAPH'86.

[20] Press, W., Flannery, B., Tenkolsky, S., and Vetterling, W. (1988). *Numerical Recipes in C, The Art of Scientific Computing*. Cambridge University Press.

[21] Ravela, S., Draper, B., Lim, J., and Weiss, R. (1995). Adaptive tracking and model registration across distinct aspects. In *Proceedings of* IROS*'95 (*IEEE/RSJ *Conference on Intelligent Robots and Systems)*, pages 174–180, Pittsburgh.

[22] Ravela, S., Draper, B., Lim, J., and Weiss, R. (1996). Tracking object motion across aspect changes for augmented reality. In *Proceedings of* IUW *(*DARPA *Image Understanding Workshop)*.

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

[24] Simon, G. and Berger, M.-O. (1998). A two-stage robust statistical method for temporal registration from features of various type. *International Journal of Computer Vision*, submitted.

[25] State, A., Livingstone, M., Garett, W., Hirota, G., Whitton, M., and Pisan, E. (1996). Technologies for Augmented Reality Systems: Realizing Ultrasound Guided Needle Biopsies. *Computer Graphics Proceedings, Annual Conference Series*, 30:439–446. Proceedings of SIGGRAPH'96.

[26] Stoer, J. and Bulirsch, R. (1980). *Introduction to Numerical Analysis*. Springer-Verlag, New York.

[27] Uenohara, M. and Kanade, T. (1996). Real-time vision-based object registration for image overlay. *Journal of Computers in Biology and Medicine*. In press.

[28] Uno, S. and Matsuka, H. (1979). A general purpose graphic system for computer-aided design. *Computer Graphics Proceedings, Annual Conference Series*, 13(2):25–32. Proceedings of SIGGRAPH'79.