Handling Occlusion in Augmented Reality Systems: A Semi-Automatic Method

Vincent Lepetit and Marie-Odile Berger
LORIA/INRIA Lorraine
BP 101, 54602 Villers les Nancy, France
{lepetit,berger}@loria.fr

Abstract

We present a semi-automatic approach to solve occlusion in AR systems. Once the occluding objects have been segmented by hand in selected views called key-frames, the occluding boundary is computed automatically in the intermediate views. To do that, the 3D reconstruction of the occluding boundary is achieved from the outlined silhouettes. This allows us to recover a good prediction of the 2D occluding boundary which is refined using region-based tracking and active contour models. As a result, we get an accurate estimation of the occluding objects.

Various results are presented demonstrating occlusion resolution on real video sequences. Results and videos are available at the URL: http://www.loria.fr/~lepetit/Occlusions.

1. Introduction

The objective of augmented reality (AR) is to add virtual objects to real video sequences, allowing computer-generated objects to be overlaid on the video in such a manner as to appear part of the viewed 3D scene. Applications include computer-aided surgery, tele-operations, and special effects for the film and the broadcast industries. This paper concentrates on the particular application of video post-production.

Realistic image composition requires that the augmented patterns be correctly occluded by foreground objects. However, solving the occlusion problem for AR is challenging when little is known about the real world we wish to augment. Theoretically, resolving occlusion amounts to compare the depth of the virtual objects to that of the real scene. However, computing dense and accurate depth maps from images is difficult [9]. This explains why the accuracy of the obtained occluding boundary is generally poor. Moreover, in most AR applications, the interframe motion is not *a priori* known but must be computed. Inacurate motion estimation thus results in possibly large reconstruction errors.

In order to overcome problems stemming from possibly large reconstruction errors, Ong [6] proposed a semi interactive approach to solve occlusion: the occluding objects are segmented by hand in selected views called key-frames. These silhouettes are used to build the 3D model of the occluding object. The 2D occluding boundary is then obtained by projecting the 3D shape in the intermediate frames. However, due to the uncertainty on the computed interframe motion, the recovered 3D shape do not project exactly onto the occluding objects in the key-frames nor in the intermediate frames.

In this paper, we also use the concept of key-views but we do not attempt to build the 3D model of the occluding objects from all the key-frames. The novelty in this paper is twofold: (i) we do not attempt to recover the 3D model of the occluding objects from all the key-views. We only compute the 3D occluding boundary from two consecutive key views. The projection of this 3D curve is a good prediction of the actual 2D occluding boundary in the intermediate frames. (ii) we recover the actual occluding boundary with a good accuracy using deformable region-based tracking followed by an adjustment stage based on *snakes*. This allows us to compensate easily for the interframe motion error. We then obtain an accurate estimation of the occluding boundary over the sequence.

2. Overview of the system

Theoretically, the 3D shape of the occluding object can be computed from its silhouettes detected in an image sequence. For AR applications however, the interframe camera motion is computed from image/model correspondences or with 2D/2D correspondences over time [4, 7]. The errors resulting from this inaccurate registration makes the 3D reconstruction untractable. That is the reason why we only attempt to recover the 3D occluding boundary from two consecutive key-frames instead of recovering the 3D shape of the occluding object from the whole sequence. Fig. 1 explains the way we compute a first estimation of the 2D occluding boundary in each frame of the sequence. First,

the user points out key-frames which correspond to views where aspect changes of the occluding object occur. These key-frames are framed in black in Fig. 1. The user also outlines the occluding object on these key-frames (in white). It is well known that the 3D occluding boundary depends on the camera viewpoint. However, the starting point for our method is to build a good approximation of the 3D occluding boundary which will be used for all the frames between two kew-views. This 3D curve is built using stereotriangulation from the two silhouettes outlined by the user provided that the translation between the two frames is not null (Fig. 1.a and b). The projection of this approximated occluding boundary on the intermediate frames thus provides a fair estimation of the 2D occluding boundary (Fig. 1.c and 1.d).

Due to the uncertainty on the computed interframe motion, this prediction can be relatively far from the actual occluding boundary for at least two reasons (see for instance Fig. 7.a): (i) the computed 3D occluding boundary is only an approximation of the real one because stereotriangulation is performed from two occluding contours. (ii) more importantly, errors on the camera parameters induce reconstruction errors on the 3D curve and consequently errors on its projection in the considered frame.

One of the main contributions of this paper is to show that the error on the computed camera parameters can be estimated. The uncertainty on the 3D occluding boundary can then be deduced. This allows us to define a region of interest around the predicted contour which is likely to contain the actual occluding boundary (section 3). The refinement stage (section 4) is then carried out within this region: region-based tracking is first used to recover the region whose size and texture only differ from the predicted shape with an affine transformation. Finally, active contour models are used to adjust the occluding boundary.

3. Reconstructing the 3D occluding boundary

3.1. Computing the camera parameters

In this section we first briefly recall how we compute the camera motion over the sequence. Our approach to motion computation takes advantage of 3D knowledge on the scene as well as 2D/2D correspondences over time [7]. Given the viewpoint $[\mathbf{R}_k, \mathbf{t}_k]$ computed in a given frame k, we compute the viewpoint p in the next frame $k+1$ using the 3D model points M_i whose projections are detected in frame $k+1$. In addition, we use interest points [5] $(q_k^i, q_{k+1}^i)_{1 \leq i \leq m}$ that are automatically extracted and matched between frames k and $k+1$. The quality of the viewpoint can be assessed by the distance between q_{k+1}^i and the epipolar line $ep_{k+1}(q_k^i)$. The viewpoint is therefore recovered by minimizing:

$$\Phi(p) = \frac{1}{n}\sum_{i=1}^{n} dist^2(m_i, proj(M_i) + \frac{\lambda}{2m}\sum_{i=1}^{m} dist^2(q_{k+1}^i, ep_{k+1}(q_k^i)) + dist^2(q_k^i, ep_k(q_{k+1}^i)) \quad (1)$$

3.2. 3D reconstruction

We will now take some time to examine the 3D reconstruction process of the occluding boundary in a little detail. Let C_1 and C_2 be the silhouettes detected in the key-frames. In order to reconstruct thei r corresponding 3D curve, we first have to match the points of C_1 and C_2. Let c_1 a point of C_1. To find its correspondant c_2, we determine the points $\{c_2^1 \dots c_2^n\}$ of C_2 which lie on the epipolar line associated to c_1, and the points $\{c_1^1 \dots c_1^m\}$ of C_1 which lie on the epipolar line passing by c_1 (see figure 2). If $n \neq m$, something gone wrong (due to epipolar geometry imprecision) and we don't attribute any correspondant to c_1. If $n = m$, the constraint order along the epipolar lines says that there is an index i such as $c_1^i = c_1$ and $c_2^i = c_2$. Then, we can recover the 3D point which reprojects on c_1 and c_2.

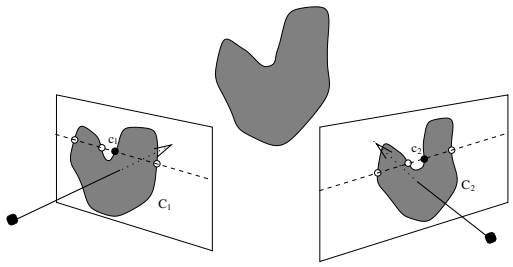


Figure 2. Matching of C_1 and C_2

As some parts of C_1 do not heave 3D counterpart (if $n \neq m$), we still have to estimate the corresponding 3D curve by interpolation. Let C' be a set of points that do not have 3D corresponding points in the reconstruction process, and let c_{11} and c_{22} be its extremities, previously reconstructed as C_{11} and C_{12}. Estimating the corresponding 3D curve by the segment $[C_{11}C_{12}]$ would not be a good idea, because the reprojection of $[C_{11}C_{12}]$ is not generally C'. A better estimation is shown in figure 3: the corresponding 3D point I_c of a point c on C' is computed as the nearest point to the segment $[C_{11}C_{12}]$ which belongs to the line $[Oc_1)$ (O the center of the camera). This way the estimated curve reprojection is C'.

Note that if the key views correspond to very different aspects of the occluding object, the reconstruction may fail because the number of intersections of the epipolar line with the two outlines shapes are different for numerous points on the curves. That is the reason why the user has to choose carefully the key-views in order to avoid this kind of problems. For an example, consider the case of the *cow sequence* in section 5.2.

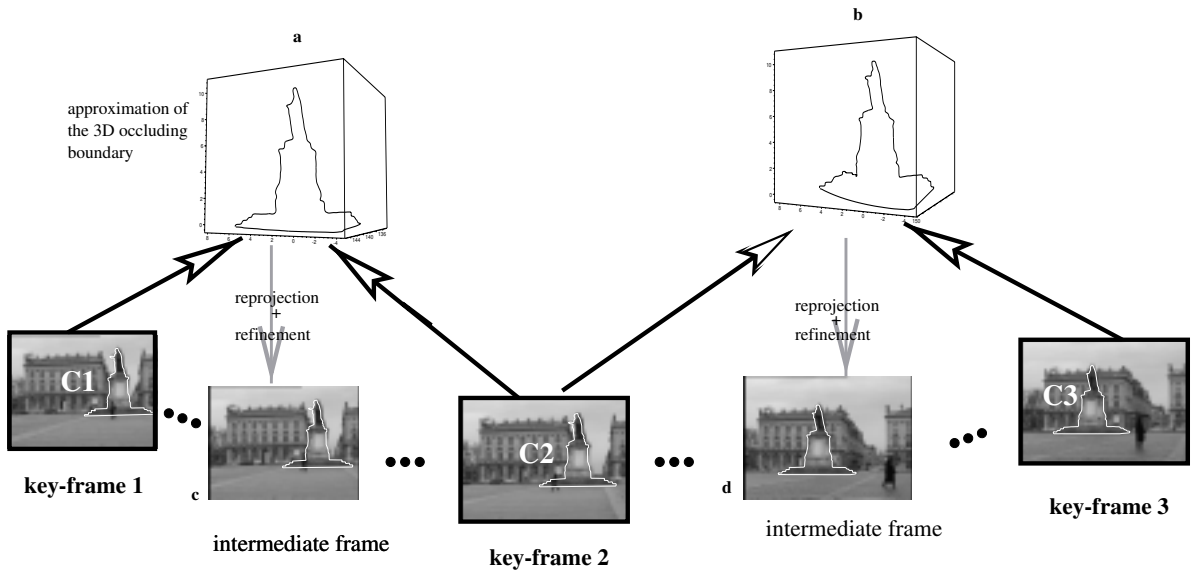


Figure 1. Overview of the system.

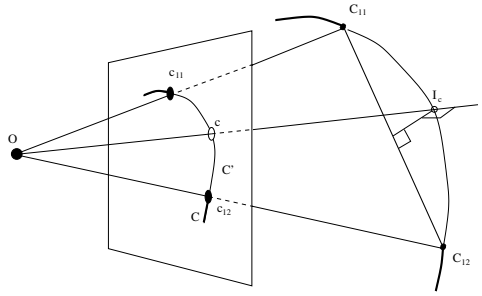


Figure 3. Estimation of the unreconstructed parts of the 3D contour

3.3. Taking into account the error on the estimated motion

The critical role of motion error in scene reconstruction has been pointed out in [8]. In this paper, we use the ϵ indifference region [1] to investigate the reliability of the estimated camera parameters and to deduce the uncertainty on the 3D occluding boundary.

The fact that we have elected to minimize a function $\Phi(p)$ means that we set some store by obtaining a low value of this function. It is reasonable to suppose that values of Φ almost as low as Φ^* would satisfy us almost as much as Φ^*. This gives rise to an ϵ indifference region in p space described by the equation:

$$\epsilon_{region} = \{p \; such \; that | \Phi(p) - \Phi(p^*)| \leq \epsilon\}$$

In a sufficiently small neighborhood of p^* we may approximate Φ by means of its Taylor equation:

$$\Phi(p) \approx \Phi(p^*) + \nabla\Phi(p^*)^t \delta p + \frac{1}{2}\delta p^t H(p^*)\delta p \quad (2)$$

where H^* is the hessian of Φ computed at $p = p^*$. More details on the computation of H^* are given in Annex A.

As p^* is the minimum of Φ, the gradient is null at the optimum $\nabla\Phi(p^*) = 0$ and equation (2) becomes

$$\Phi(p) \approx \Phi(p^*) + \frac{1}{2}\delta p^t H(p^*)\delta p$$

The ϵ indifference region is then defined by:

$$|\delta p^t H(p^*)\delta p| \leq 2\epsilon$$

which is the equation of a 6-dimensional ellipsoid.

Fig.4 shows these indifference regions computed along the Stanislas sequence (we use $\epsilon = 1$). The building in the background is the 3D model used for registration. For each frame of the sequence, we drew the ϵ indifference region for the translation parameters.

We can now compute the reconstruction error on the occluding boundary from these indifference regions. If point correspondences were available, the reconstruction error could be recovered in an analytical way from viewpoint uncertainties [8]. Unfortunately, as we only have curve correspondences, the matched points depends on the viewpoint and are computed as the intersection of the epipolar line of the point with C_2. We therefore resort to an exhaustive approach. We consider the *extremal viewpoints*, that are the vertices of the 6-dimensional indifference ellipsoid. Let $\{p_1^1, ..., p_1^{12}\}$ (resp $\{p_2^1, ..., p_2^{12}\}$) the extremal viewpoints in

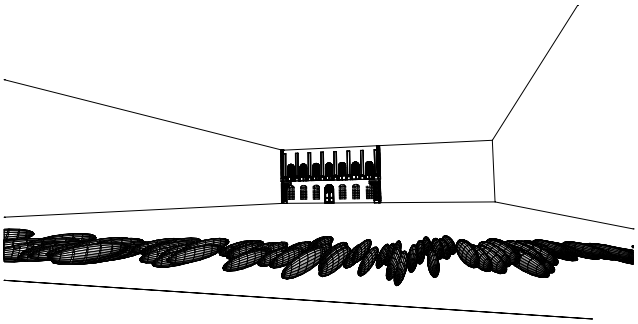


Figure 4. The indifference regions for the translation parameters over the Stanislas sequence.

the two key-views. Let m_1 be a point on C_1. Given an extremal viewpoint p_1, we can compute the 12 possible reconstructions of m_1 with the 12 extremal views in key-frame 2. Using the 12 extremal viewpoints in key-frame 1, we then obtain 12^2 extremal reconstructions of m_1 according to the uncertainty computed on the two key-views. The convex hull of these 144 points is a good approximation of the 3D reconstruction error on m_1.

We can now predict the position of the 2D occluding boundary in the in-between frames by simply reprojecting the 3D occluding boundary. To estimate the 2D uncertainty on the projected boundary C, we have to take into account the 3D reconstruction error and the uncertainty on the considered viewpoint. We again resort to an exhaustive method: for each point m_i on C, the 12^2 possible extremal reconstructions are projected onto the current frame using the 12 extremal viewpoints of this frame. We define the spatial uncertainty on the predicted occluding boundary associated to m_i as the convex hull of these 12^3 image points. This area is denoted Λ_i in the following.

The main stages for computing the 2D uncertainty on the predicted occluding boundary are illustrated in Fig. 5: Fig. 5.a exhibits a point on the predicted boundary and Fig. 5.b shows the projection of the corresponding 3D extremal points using the extremal viewpoints and the convex hull Λ_i. Finally, Fig. 7.a shows the 2D uncertainty computed for each point of the predicted boundary (dotted line). The points are drawn with black circles or crosses and the uncertainty is drawn in white. The reader can notice that some points on the steps have no associated spatial uncertainty. Indeed, because the key silhouettes do not match exactly, the epipolar line computed with some extremal viewpoints does not always intersect C_2. If more than 50% of the epipolar lines computed with the 12^2 extremal viewpoints do not intersect C_2, the spatial uncertainty is not defined at this point.

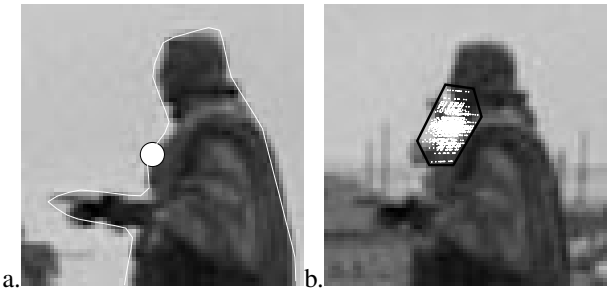


Figure 5. Computation of the spatial uncertainty on the predicted occluding boundary.

4. Refining the occluding boundary

As a result of the prediction stage we get an estimate of the occluding boundary along with its 2D uncertainty in the considered frame. In addition we compute not only the boundary but also the texture of the occluding object so as to get a *predicted template* of the occluding object. The texture $I_{template}$ is computed from the nearest key-view by using 2d local image transformation.

We still have to determine the occluding object from the *predicted template*. Due to the error on the computed motion and also because reconstruction is achieved from occluding contours, the template boundary can be relatively far from the actual occluding object and their shapes can also differ (see for instance Fig. 7.a). However, it is important to note that the actual boundary belongs to the computed uncertainty region. Following previous works on deformable structures [2] we use a hierarchical algorithm; we first compute a global estimation of the shape deformation between the key-frame and the current frame. Then we use a fine tuning deformation to adjust the details. As affine transformations seem to be appropriate to describe shape variations due to motion uncertainties, the affine motion that best matches the occluding template on the considered image is searched for:

$$transf_a(m) = \begin{cases} a_1 m_x + a_2 m_y + a_3 \\ a_4 m_x + a_5 m_y + a_6 \end{cases}$$

The optimal parameter a is defined as the one that yields the best fit between the predicted template $I_{template}$ and the current image I. The best match is defined as the minimum of the correlation measure:

$$\Psi(a) = \sum_i \psi_a(i) \tag{3}$$

$$\psi_a(i) = \sum_{\substack{d^x, d^y = -W \\ (m_i + d) \in R_{\mathcal{C}}}}^{d^x, d^y = W} (I_{template}(m_i + d) - I(transf_a(m_i + d)))^2$$

where the predicted curve C is defined by the set of vertices $\{m_i\}_{1\leq i\leq n}$, $d = (d^x, d^y)$, W is the size of the correlation window and R_C is the region inside C. Note that only the points which are inside the occluding objects are considered in the estimation. This way, points belonging to the changing background do no affect the matching process.

In addition, we have slightly modified the correlation measure in order to take into account the 2D uncertainty on the predicted curve. A penalty term is used to ensure that the matched point belongs to Λ_i. The penalty has the form αW^2 where α is a constant value. The function to be minimized is therefore defined as:

$$\psi_a(i) = \begin{cases} \sum_d (I_{template}(m_i + d) - I(transf_a(m_i + d)))^2 \\ \quad if\ transf_a(m_i) \in \Lambda_i, \\ \alpha W^2\ otherwise. \end{cases}$$

Note that if Λ_i is not defined, or equivalently if $\Lambda_i = \infty$, the first item of ψ_a is used because the assumption $transf_a(m_i) \in \Lambda_i$ is fullfiled. These points are therefore considered in the correlation function without further constraints. Finally, fine tuning adjustement of the occluding boundary is performed with snakes from $transf_a(C)$.

5. Results and discussion

The effectiveness of our approach is demonstrated on three sequences: the Stanislas sequence, the cow sequence and the Loria sequence. Each of these sequence demonstrates the hability of our algorithm to handle occlusions in various situations. We want to prove that our algorithm is efficient even in some cases which are well known to be difficult both for viewpoint recovery and 3D reconstruction. In the considered examples we especially adress the case of camera motions along the optical axes which are difficult for the tracking task and the 3D reconstruction (see the cow sequence and the Loria sequence). We also consider in the Loria sequence a camera path which goes towards the occluding object and goes beyond it. Note that the original and the augmented videos can be seen at out URL http://www.loria.fr/~lepetit/Occlusions

5.1. The Stanislas Sequence

The Stanislas sequence was shot from a car which turned around the square. Our aim is to incrust a virtual plane passing behind the statue. Here, the 3D model of the opera is used for registration (the building in the back of the scene) while the 3D model of the statue is unknown. The three keyframes chosen by the user are shown in Fig. 6 (frames 66, 118, 150). Fig. 6 exhibits the recovered occluding boundary in the frames 15, 66 and 130. The overall visual impression is very good though the predicted boundary is sometimes relatively far from the actual one.

Fig. 7 clearly proves the efficiency of incorporating motion error into our process. The uncertainty on the predicted curve is drawn in white. The points m_i that are outside the uncertainty region Λ_i after the region based tracking are shown black crosses, whereas the points inside the region are drawn with black circles. For both images, the predicted 2D curve is shown in dotted lines. If the 2D uncertainty is not considered (Fig. 7.a), the recovered boundary is erroneous, especially near the steps. On the contrary, if points are constrained to be in the uncertainty region, the occluding boundary is successfully recovered (Fig. 7.b).

5.2. The cow sequence

This sequence consists of 120 frames. The camera undergoes various motions: translation along the optical axis and also rotating motions. In this sequence, we want to add a brown cow just behind the black and white cow. The key views used to recover the 3D occluding boundaries as well as the outlined boundaries are shown in Fig. 8. Note that some key views are very close (frames 30, 31, 40, 41). This is because the aspect graph of the occluding object is very complicated especially due to the legs of the cow: in frame 30, only 3 legs are visible whereas the four legs are visible in frame 31. Also between frame 40 and 41, three legs are visible (two of them are pressed) whereas the four legs are visible in frame 41. Fours legs are visible whereas two of them are pressed in the next frame. The topology of the occluding boundary is then different between these two frames. This leads us to define two key-views in order that the 3D reconstruction and especially the matching stage succeeds. The 3D occluding boundaries built from the key-views are shown in Fig. 9 (first row). Also shown in the figure are zooms on the computed 2D occluding boundary for frames 20, 35 and 110 so that the user can appreciate the accuracy of the occluding boundary. Finally, some snapshots of the augmented scene are shown in Fig. 10. When looking at the full video, the reader can notice that the composition is very stable and realistic, even on the foreground of the scene, and that the added objects really appear part of the 3D scene.

5.3. The Loria sequence

This sequence consists of 500 frames and was shot around our laboratory, the LORIA. The dominant motion of the camera is a translation along the optical axis. Such a motion is known to be difficult both for motion recovery and for 3D reconstruction. Indeed, the line of sight of 3D points which lie in front of the camera are nearly parallel and small localization error on the corresponding 2D points may lead to large errors on the reconstructed point. Besides this, another difficulty of this sequence originates

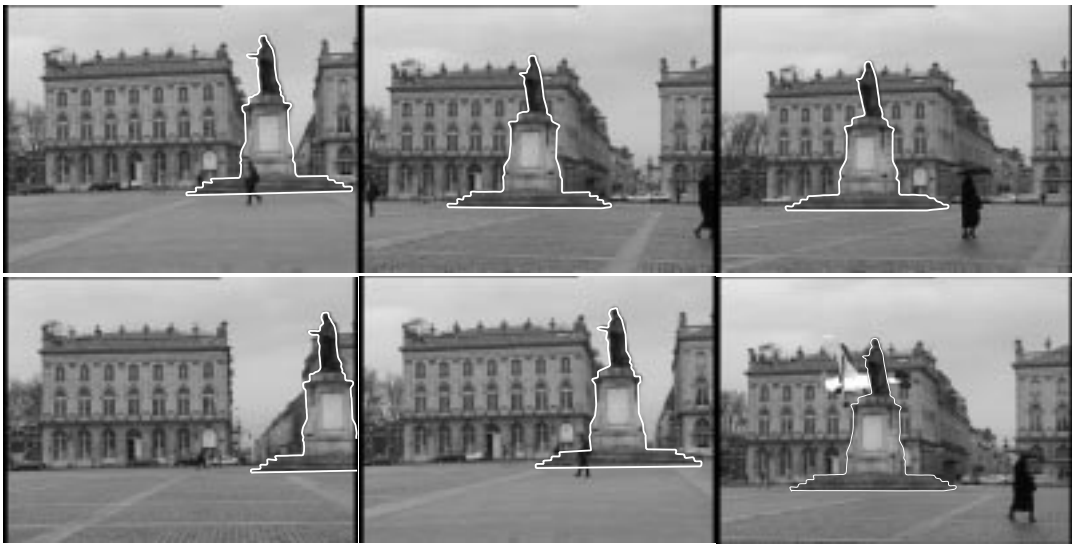
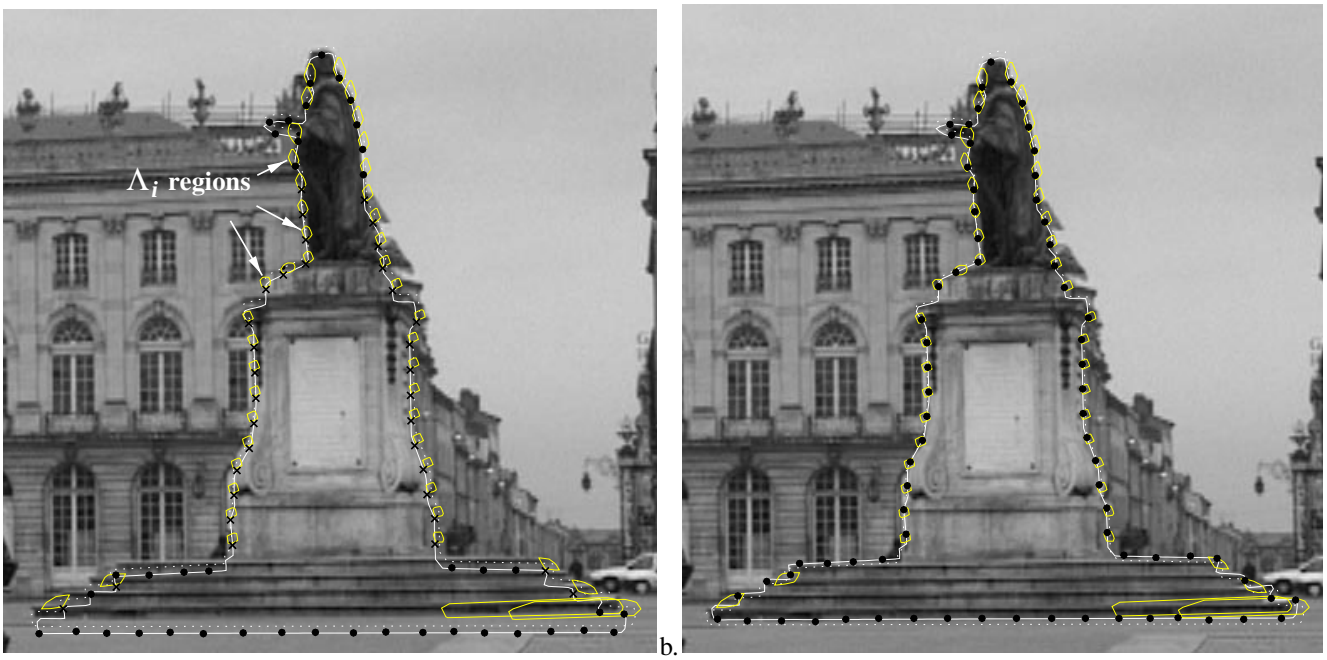


Figure 6. (first row) : The key-views along with the user-defined silhouette: frame 60, 118 and 150. (second row): The recovered occluding boundary in the frames 15, 66,130 and the augmented scene.



a. b.

Figure 7. The recovered occluding boundary without (a) and with (b) the use of the 2D uncertainty. The predicted curve is shown with dotted lines. The points that belong to the uncertainty region Λ_i are shown with black circles, whereas the points outside Λ_i are drawn with black crosses.

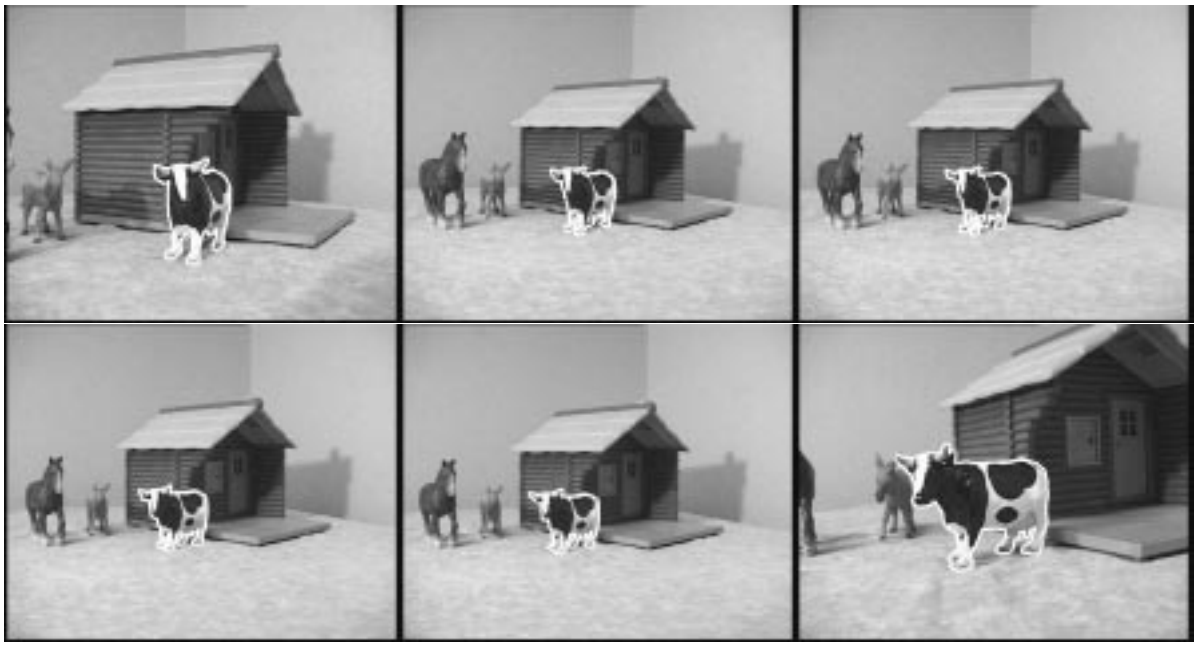


Figure 8. The key-frames for the cow sequence: frames 0, 30, 31, 40, 41, 120.

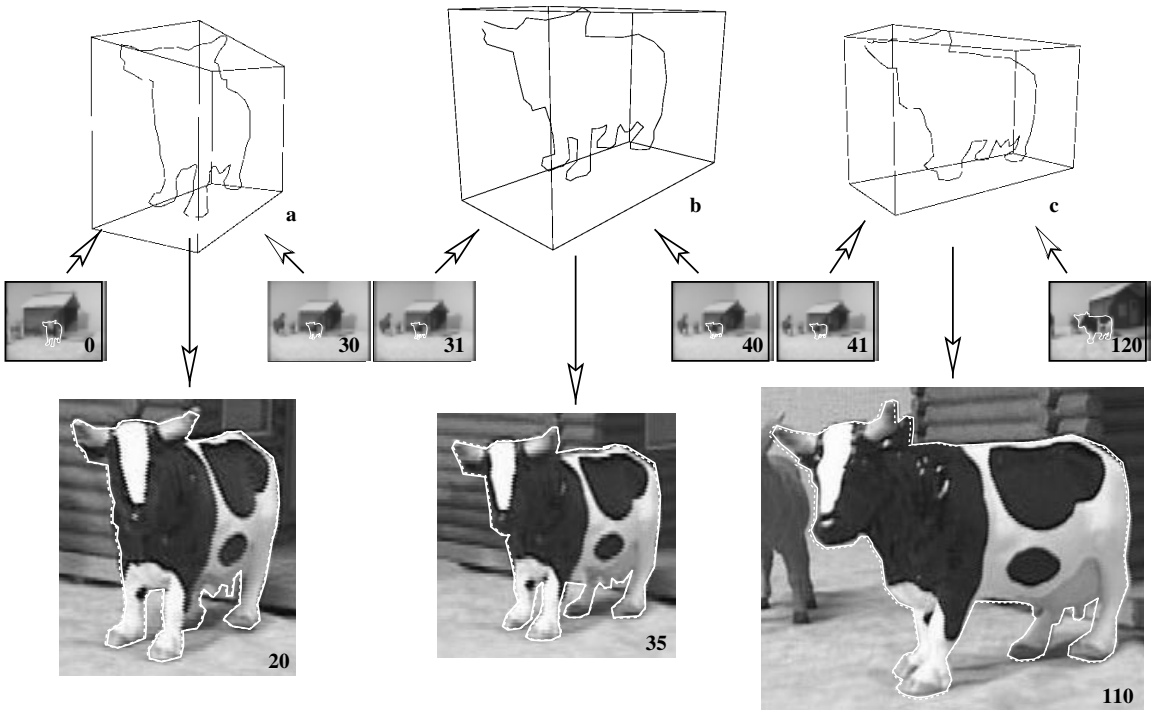


Figure 9. The cow sequence:
(first row) : The 3D occluding boundary built from the key views: (a) from frames 0 and 30, (b) from frames 31 and 40, (c) from frames 41 and 120.
(last row) : The computed occlunding boundary (bold lines) in frames 20, 35 and 110 superimposed on the original images.

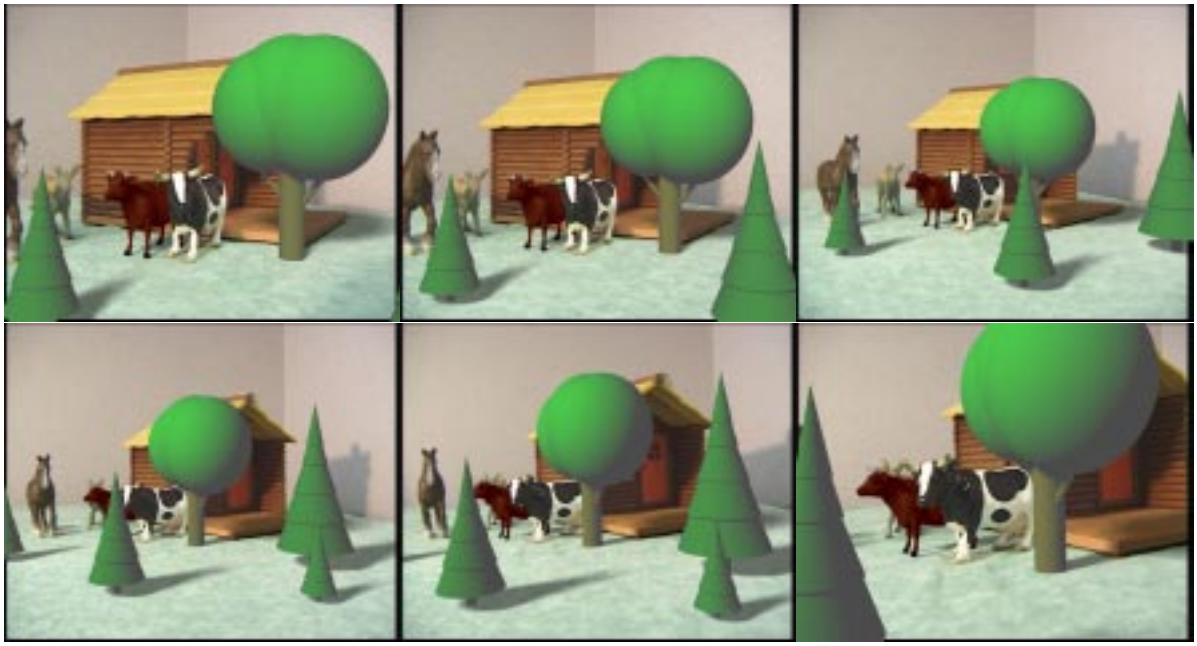


Figure 10. Snapshots of the augmented scene : frames 5, 15, 35, 60, 80, 110.

in the place of the virtual object in the scene. The virtual object stands near the camera path and is occluded by the white post. As the camera moves, the size of the occluding object increases. This can cause trouble both on the reconstruction process and on the region based refinement stage. In addition, the occluding boundary must be outlined with a good acuracy because the added object lies in the foreground of the scene and small errors are easily detected by the human vision.

In this sequence, we only use two key-views (frames 0 and 327) because the aspect of the occluding object does not change very much. Though the motion is a translation along the optical axis, the viewpoint is correctely recovered and the 3D reconstruction of the white post is quite good (Fig. 11). The last row of Fig. 11 shows the result of the refinement stage for several frames. The predicted occluded boundary is drawn with dashed lined whereas the results of the region based refinment stage is shown with bold lines. Although some prediction is sometimes relatively far from the actual occluding boundary, the refinement stage succeed in recovering the actual boundary in nearly all cases. However, some problems arise at the end of the sequence when the light post is going to leave the image. This is because small errors on the viewpoint sometimes results in large reprojection errors on the object and the prediction can be far form the actual boundary (see Fig. 11.e).

Finally, the scene has been augmented with a *porsche* which is parked in front of the building (Fig. 12).

6. Conclusion

We have presented a new approach for resolving occlusion for AR tasks. The key concept is that fine detection of occluding boundary can be achieved with moderate user interaction. One of the main strengths of our algorithm concerns its ability to handle uncertainties on the computed motion between two frames. Through judicious choice of key-frames, our approach seems to be more convenient and more accurate than most existing approaches.

Annex A: Computing the Hessian H^*

The computation of H^* originates in [3],

H^* is the value of the Hessian $H = \frac{\partial}{\partial z}(\frac{\partial \Phi}{\partial z})^t$ computed at the minimum p^* of Φ. Φ is defined as $\Phi(p) = \frac{1}{n}\sum_{i=1}^{n} {r_i}^2 + \frac{\lambda}{2m}\sum_{i=1}^{m} {v_i}^2$ where

$$
\begin{aligned}
{r_i}^2 &= dist^2(m_i, proj(M_i)) \\
{v_i}^2 &= dist^2\left(q^i_{k+1}, ep_{k+1}(q^i_k)\right) + dist^2\left(q^i_k, ep_k(q^i_{k+1})\right)
\end{aligned}
$$

${r_i}^2$ and ${v_i}^2$ can be expressed as an analytical function of the 6-dimensional vector $p = (\alpha, \beta, \gamma, t_x, t_y, t_z)$ using the fundamental matrix. Because the analytic expression of the second derivatives of ${v_i}^2$ with respect to p are really untractable, we use an approximation to the first order: $H \approx 2\sum \frac{1}{n}\left(\frac{\partial r_i}{\partial p}\right)^t\left(\frac{\partial r_i}{\partial p}\right) + \frac{\lambda}{2m}\sum\left(\frac{\partial v_i}{\partial p}\right)^t\left(\frac{\partial v_i}{\partial p}\right)$.

References

[1] Y. Bard. *Nonlinear Parametric Estimation*. Academic Press, 1974.

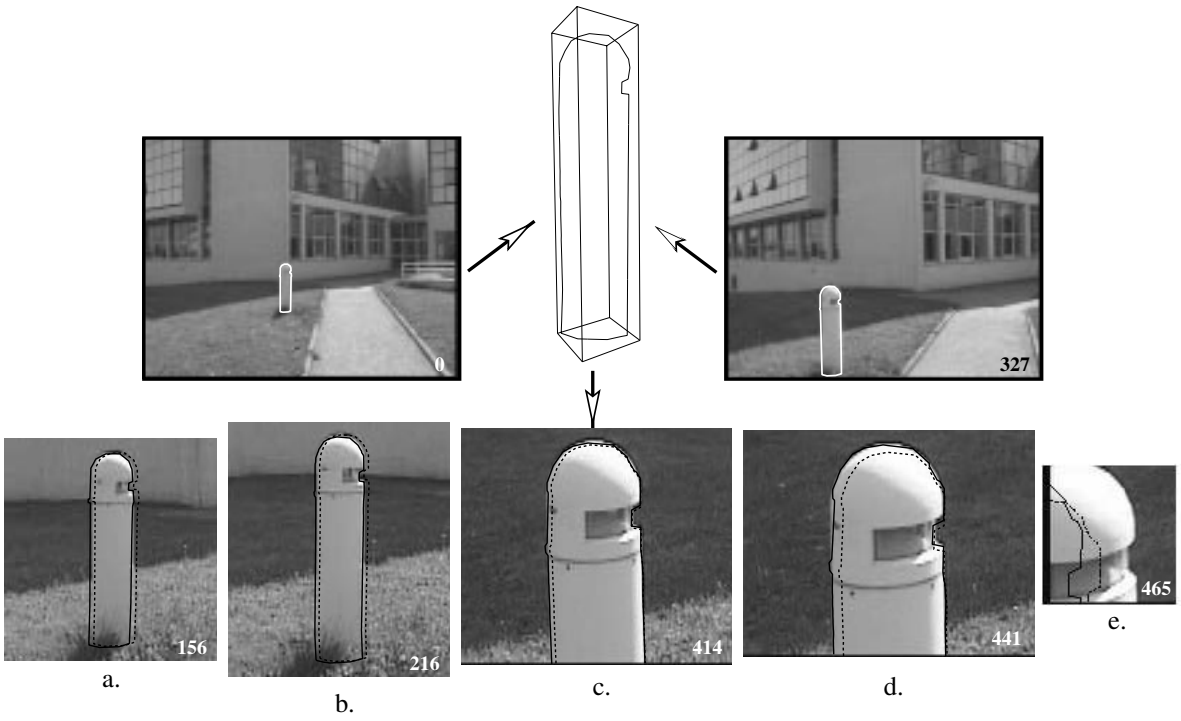


Figure 11. The Loria sequence:
(first row) : The key-views (frame 0 and 327) and the 3D occluded boundary.
(second row) : Zoom on the predicted occluding boundary (dashed lines) and the recovered occluding boundary (bold lines) for frames 156, 216, 414, 441, 465.

Figure 12. The augmented scene for the Loria sequence (frames 0, 100, 300, 400).

[2] B. Bascle and R. Deriche. Stereo Matching Reconstruction and Refinement of 3D curves Using Deformable Contours. In *Proceedings of 4th International Conference on Computer Vision, Berlin (Germany)*, pages 421–430, 1993.

[3] Csurka, C. Zeller, Z. Zhang, and O. Faugeras. Characterizing the Uncertainty of the Fundamental matrix. *Computer Vision and Image Understanding*, 68(1):18–36, May 1997.

[4] A. Fitzgibbon and A. Zisserman. Automatic Camera Recovery for Closed or Open Images Sequences. In *Proceedings of 5th European Conference on Computer Vision, University of Freiburg (Germany)*, pages 311–326, June 1998.

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

[6] K. C. Ong, H. C. Teh, and T. S. Tan. Resolving occlusion in image sequence made easy. *The Visual Computer*, 14:153–165, 1998.

[7] G. Simon, V. Lepetit, and M.-O. Berger. Computer Vision Methods for Registration: Mixing 3D Knowledge and 2D Correspondences for Accurate Image Composition. In *First International Workshop on Augmented Reality, San Francisco*, Nov. 1998.

[8] J. Thomas, A. Hanson, and J. Oliensis. Understanding Noise: The Critical Role of Motion Error in Scene Reconstruction. In *Proceedings of 4th International Conference on Computer Vision, Berlin (Germany)*, 1993.

[9] M. Wloka and B. Anderson. Resolving Occlusions in Augmented Reality. In *Symposium on Interactive 3D Graphics Proceedings, (New York)*, pages 5–12, Aug. 1995.