Retexturing in the Presence of Complex Illumination and Occlusions

Julien Pilet Vincent Lepetit Pascal Fua

École Polytechnique Fédérale de Lausanne, Switzerland

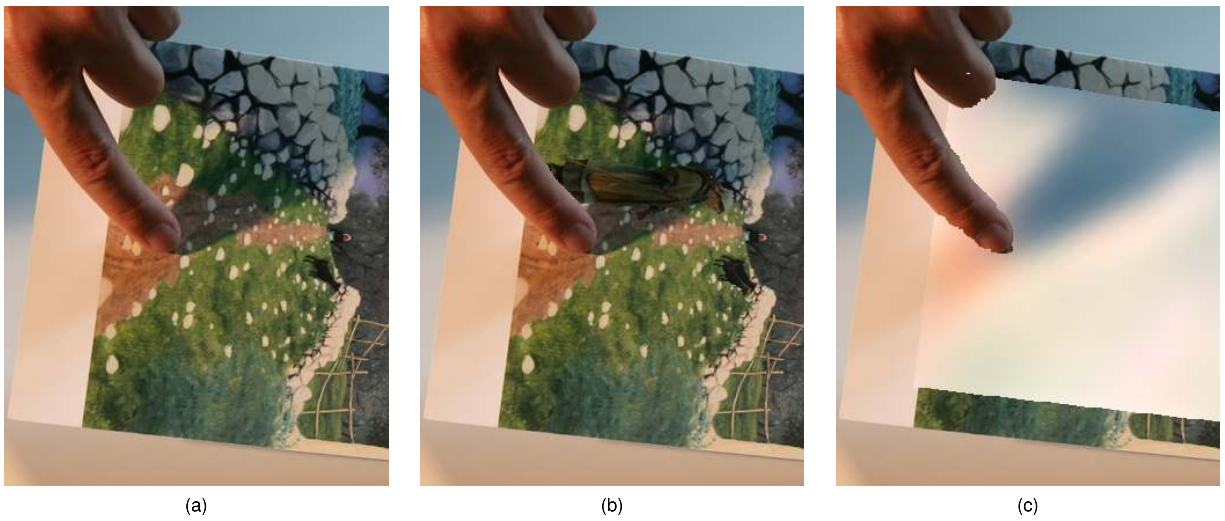


Figure 1: Adding a virtual character into a postcard. (a) The original image is partially hidden by a finger, which casts a shadow on it. (b) The picture of a woman has been added. It is correctly shaded and her feet are not shown since they would have been hidden by the finger, had she appeared in the original postcard. (c) What the postcard would have looked like if its middle part had been white. This can be viewed as *diminished reality*. Note that the shadow cast by the finger is correctly modeled. This figure, as well as most of the others, is best viewed in color.

Abstract

We present a non-rigid registration technique that achieves spatial, photometric, and visibility accuracy. It lets us photo-realistically augment 3D deformable surfaces under complex illumination conditions and in spite of severe occlusions. There are many approaches that address some of these issues but very few that simultaneously handle all of them as we do.

We use triangulated meshes to model the geometry and introduce explicit visibility maps as well as separate illumination parameters for each mesh vertex. We cast our registration problem in an Expectation Maximization framework that allows robust and fully automated operation. It provides explicit illumination and occlusion models that can be used for rendering purposes.

Keywords: Occlusion, Augmented Reality, Shadows

1 Introduction

In recent years, there has been growing interest in modeling generic deformable surfaces from video sequences [3, 2, 11, 6, 19, 16]. For Augmented Reality purposes, it allows *retexturing* of the surfaces, which involves changing their appearance while preserving their shape, as shown in Fig. 1.

However, current methods suffer from a number of limitations. Some require markers [3] or large training databases [6]. Others lack accuracy [11] or artificially limit the number of the colors on the surface [19, 16] or the number of light sources [18].

In this paper, we propose a method that overcomes these limitations and allows automated retexturing under complex illumination conditions and in the presence of occlusions. Our technique starts from a set of wide baseline correspondences between a *model image* of the undeformed surface and the *current image* we are trying to retexture to obtain initial 3D pose and shape estimates. These estimates are then refined together with visibility and lighting maps by matching the texture in the model image against that of the input image.

Our approach is inspired by standard template matching approaches [10, 1], although we explicitely model occlusions and local illumination parameters, which gives us both additional robustness and increased realism when synthesizing the new textures. Our only requirement is that the real surface we are trying to model is textured enough to establish the initial correspondences.

More specifically, the visibility map we use defines whether or not a pixel in the model image is visible or hidden in the input one. The lighting map includes separate illumination parameters for each vertex of the meshes we use to represent the surfaces. It accounts for the fact that illumination may vary across the surface in unpredictable ways, with the only assumption that these variations are piecewise smooth. Our algorithm can therefore handle much more complex combinations of occlusion and lighting patterns than state-of-the-art methods, such as the recent approach [5] that proposes an elegant EM framework to impose the spatial coherence of occlusions. Of course, this flexibility comes at a cost since it makes estimating the shape, illumination, and visibility parameters a potentially ill-conditioned optimization problem with many local minima. The main technical contribution of this paper is therefore an optimization scheme that provides stable and realistic solutions. This allows fully automated operation without manual initialization to produce Augmented Reality results such as those depicted by Fig. 1.

2 Related Work

Effective 3D deformable surface modeling for Augmented Reality purposes must include accurate geometric reconstruction, recovery of the illumination parameters to correctly relight the objects to be added, and correct handling of both self-occlusions and other objects that may hide parts of the surface. As will be discussed in this section, there are many approaches that address some of these issues but very few that handle all of them as we do.

2.1 Non-Rigid Registration

There are two major approaches to registration of non-rigid surfaces. The first relies on image feature matching and can be fully automated [12, 7]. The second involves direct minimization of pixel intensity differences. It tends to be more precise than the former but requires good initial estimates to avoid getting trapped in local minima. It is therefore often restricted to frame-to-frame tracking [1], or used in conjunction with very specific deformation models [4, 13]. This requirement can be relaxed for near-regular textures [9], or when there are only a few colors on the surface [19, 16]. However, this lacks generality.

This is why we chose to combine the best of both worlds by initializing the registration process using wide-baseline matching of feature points [7] and then refining the initial estimates using pixel level minimization. The initialization step does not require illumination or occlusion modeling but is sufficiently accurate to bootstrap it. It also frees us from the need of a training database, which will not be available in general.

2.2 Illumination Models

For our application, we require not only robustness to illumination changes but actual retrieval of illumination parameters for rendering purposes. Image warping approaches have long incorporated polynomial [17] or spline [16] illumination models. Similarly, [5] allows for an affine illumination change. However, none of these models includes enough parameters to represent truly complex lighting effects, such as those caused by arbitrary objects casting shadows as shown in Fig. 1.

Active appearance models [4, 6] and morphable models [13] incorporate more powerful illumination models, which have been learned by performing Principal Component Analysis (PCA) on a training set of views of the target objects under different lighting conditions. This is effective for face tracking because light often comes more or less from above, faces are usually oriented vertically, and strong shadow patterns are uncommon. Thus, using face image databases and PCA to reduce the dimensionality of the illumination space makes perfect sense but is not generally applicable, especially when no training database is available.

By contrast, our approach incorporates very flexible illumination models— those used in this paper have around 600 parameters, in contrast to the 20 or so typically used by active appearance models—that can handle complex shading patterns without prior training. This makes a significant difference in the complexity of the problem we face. Allowing more degrees of freedom for irradiance estimation results in many more local minima in the objective function we minimize because image intensity differences between model and input images can be reduced by changing either the lighting or shape parameters.

2.3 Retrieving Occlusion Masks

A standard approach to deal with occlusions is based on robust estimators that decrease the influence of large error terms [1]. For example, in [6], robust estimation is implemented efficiently in the inverse compositional algorithm in which the Hessian matrices are pre-computed. This is achieved by assuming spatially coherent occlusions, computing the visibility of individual mesh facets, and using them to assemble the matrices.

Note, however, that straightforward robust estimation will not perform well in the presence of strong illumination effects without explicitly taking them into account. Furthermore, as is the case of illumination modeling discussed above, we need not only robustness to occlusions but also explicit models of their locations for rendering purposes, as shown in Fig. 1. In [5], an elegant framework is introduced for imposing the spatial coherence of occlusions by means of an EM algorithm. Its implementation, however, relies on a simple pixel-wise measure to predict occlusions. This is not sufficient when used in conjunction with an illumination model as flexible as ours because occluded model image pixels can be made to look very similar to pixels in the input image by simply varying the illumination parameters. As discussed below, we address this issue by estimating pixel visibility using whole neighborhoods as opposed to individual pixels.

3 Basic Shape and Illumination Recovery

We next introduce our generative model and show how we use it to retrieve surface shape and illumination parameters *without* considering occlusions. Those will be introduced in the following subsection by incorporating this generative model into an EM framework.

3.1 Generative Model

As discussed earlier, we assume we are given a *model image* in which the surface is not deformed and lit by diffuse lighting. Our goal is to match it against an *input image* in which the surface is both deformed and subjected to complex illumination effects.

To this end, we represent the surface as 3D triangular meshes whose geometry is controlled by a vector θ of 6 global orientation parameters and 20 shape parameters. Following the approach introduced in [15] the shape parameters are taken to be weights assigned to deformation modes that were computed by performing PCA on a database of deformed versions of the mesh. Given θ and the model image, we can synthesize S_θ, the image we would see if the surface had been acquired with an identical geometry, but under constant diffuse lighting.

To account for the fact that each part of the mesh may receive different amounts of light, we introduce Λ, a vector that contains one *lighting factor* per vertex and an offset s that is common to all vertices and arises from different camera settings. It is used to generate a second synthetic image $S_{\theta,\Lambda}$ as follows. For a pixel x belonging to a facet whose vertices have lighting factors Λ_a, Λ_b and Λ_c, we take the $S_{\theta,\Lambda}(x)$ the gray level of x to be

$$S_{\theta,\Lambda}(x) = S_\theta(x)(\beta_a\Lambda_a + \beta_b\Lambda_b + \beta_c\Lambda_c) + s \, , \tag{1}$$

where β_a, β_b and β_c are the barycentric coordinates of x with respect to the three vertices. For any given value of θ, by representing all pixel values in a column vector, this can be written in matrix form as

$$S_{\theta,\Lambda} = B_\theta\Lambda \; . \tag{2}$$

This formulation gives us the flexibility required to handle arbitrary shading patterns whose shape cannot be predicted in advance. When working with color images, we use the same formalism but introduce a different lighting factor for each color band.

3.2 MAP Estimation

Estimating the shape and illumination parameters in an input image I amounts to estimating

$$\begin{aligned}(\widehat{\theta},\widehat{\Lambda}) &= \operatorname*{argmax}_{\theta,\Lambda} P(\theta,\Lambda \mid I) \ , \\ &= \operatorname*{argmax}_{\theta,\Lambda} P(I \mid \theta,\Lambda)P(\theta,\Lambda) \ , \\ &= \operatorname*{argmax}_{\theta,\Lambda} P(I \mid \theta,\Lambda)P(\theta)P(\Lambda) \ ,\end{aligned} \tag{3}$$

where the second line of the equation follows from Bayes' rule and the third from assuming independence between the surface shape parameterized by θ and the illumination factors Λ. This is not strictly true because illumination depends on surface orientation. However, it is a reasonable assumption for our purposes since we usually deal with ambient diffuse lighting and because illumination effects which are shape independent, such as the shadows of Fig. 5(a), tend to be dominant.

To enforce the fact that the surface we model does not stretch or shrink, we take $P(\theta)$ to be

$$\begin{aligned}P(\theta) &\propto exp(-E_D) \\ E_D(\theta) &= \sum_{i=1}^{V}\sum_{v_j\in\mathscr{N}(v_i)}(\|v_i-v_j\|-L_{i,j})^2 \ ,\end{aligned} \tag{4}$$

where v_i is a vertex of the mesh deformed by θ, $\mathscr{N}(v_i)$ represents the set of all its neighbors, and $L_{i,j}$ is the distance between v_i and v_j in the undeformed mesh.

We model the piecewise smooth nature of illumination by penalizing large second derivatives in the spatial values of the lighting factors. We write

$$\begin{aligned}P(\Lambda) &\propto exp(-E_\Lambda) \ , \\ E_\Lambda &= \sum_{(a,b,c)\in A}(-\Lambda_a+2\Lambda_b-\Lambda_c)^2 \ , \\ &= \|K\Lambda\|^2 \ ,\end{aligned} \tag{5}$$

where A is the set of aligned, connected, and equidistant vertice triplets in the base mesh and K is a large but very sparse matrix.

Finally, we take $P(I \mid \theta,\Lambda)$ to simply be

$$\begin{aligned}P(I\mid\theta,\Lambda) &\propto exp(-E_I) \ , \\ E_I &= \|I-S_{\theta,\Lambda}\|^2 \ ,\end{aligned} \tag{6}$$

which amounts to saying that once the gray levels of the input image pixels are explained in terms of the model, only independent pixel noise remains.

3.3 Optimization Framework

Given the expressions of $P(I \mid \theta,\Lambda)$, $P(\Lambda)$, and $P(\theta)$ introduced above, finding the MAP estimate of Eq. 3 is equivalent to solving the least-squares problem

$$(\widehat{\theta},\widehat{\Lambda}) = \operatorname*{argmin}_{\theta,\Lambda}\|I-S_{\theta,\Lambda}\|^2+\|K\Lambda\|^2+E_D(\theta)\,. \tag{7}$$

In practice, we automatically initialize θ using a set of correspondences between the model and input images using a wide baseline matching technique, as in [7]. We then alternatively and iteratively solve over Λ and over θ. The latter is standard Levenberg-Marquardt while the former is achieved by solving the linear system

$$\begin{aligned}\operatorname*{argmin}_{\Lambda} &= \|I-S_{\theta,\Lambda}\|^2+\|K\Lambda\|^2 \\ &= \|I-B_\theta\Lambda\|^2+\|K\Lambda\|^2 \\ &= \left\|\begin{bmatrix}B_\theta\\K\end{bmatrix}\Lambda-\begin{bmatrix}I\\0\end{bmatrix}\right\|^2\end{aligned} \tag{8}$$

where B_θ comes form Eq. 2, which accounts for the lighting factors of Eq. 1. Observe that since Λ can be solved in closed form, no initial estimate is required.

4 HANDLING OCCLUSIONS

The method introduced in Section 3 is effective in the absence of occlusions. Here we extend it so that it also works in their presence, not only to achieve robustness but also to compute precise occlusion masks that can be used to augment only the visible parts of the surface and ignore the others, as shown in Figs. 1 and 2.

To this end, we take advantage the EM framework introduced in [5] to estimate visibility while imposing spatial coherence of the occlusion masks, something that standard robust estimation does not do. However, its original implementation only has global illumination parameters and does not account for local lighting effects, such as shadows, which our generative model does. This potentially creates many ambiguities since occlusions and shadows often have similar effects on pixel intensities. We have therefore replaced the very local similarity measures used in [5] by more global ones that involve whole neighborhoods. As a result, our approach can disambiguate local shadows from occlusions.

In the remainder of this section, we briefly summarize the EM approach of [5] and discuss its limitations. We then show how they can be overcome by replacing the local measures it relies on by more global ones.

4.1 Visibility Maps

The starting point of [5] is to rewrite Eq. 3 as

$$(\widehat{\theta},\widehat{\Lambda}) = \operatorname*{argmax}_{\theta,\Lambda}\sum_{v}P(I,v\mid\theta,\Lambda)P(\theta)P(\Lambda)\,, \tag{9}$$

where v is a binary visibility map that signals if individual pixels lie on the target surface or are occluded. Spatial coherence of the occlusions is enforced by assigning to visibility maps a prior probability term $P(v)$ that is inversely proportional to the number of transitions between visible and occluded pixels.

Solving Eq. 9 implies evaluating a sum over all possible maps, and direct maximisation is infeasible. Instead, a mean-field expectation-maximization algorithm is used.

The E-step estimates a probability of visibility $b(x)$ for each pixel. In order to enforce spatial consistency, the optimal set of such probabilities must satisfy the set of coupled non-linear equations

$$b(x)=\sigma\left(\frac{2}{T}\sum_{y\in\Gamma(x)}(2b(y)-1)+\log\frac{f(x)P_f}{g(x)P_g}\right) \ , \tag{10}$$

where $\Gamma(x)$ denotes the 4-neighborhood of x, $\sigma(x)=1/(1+exp(-x))$ is the sigmoid function, T is a parameter that controls the amount of regularization, P_f is the prior probability of visibility, and $P_g=1-P_f$. The $f(.)$ and $g(.)$ functions approximate the probability of pixels being either visible or occluded given their individual gray levels. In our case, we implemented the original approach [5] by defining f and g as

$$\begin{aligned}f(x) &= G(I(x),S_{\theta,\Lambda}(x),\sigma^2) \\ g(x) &= 1/256\end{aligned} \tag{11}$$

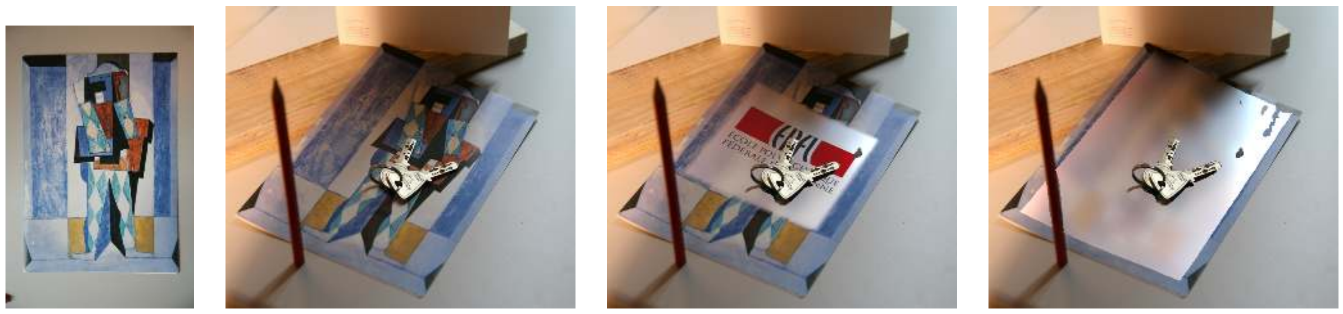

(a) (b) (c) (d)

Figure 2: Adding a virtual logo. (a) The model image. (b) An input image showing the model partially occluded by a set of keys and shaded by a complex lighting environment. (c) The input image is augmented with an EPFL logo on top of which the keys remain visible. (d) Diminished reality version of the surface in which the original texture has been replaced by a shaded version of the surface, assumed to be featureless. The small holes in the upper right corner correspond to pixels that have been erroneously labeled as occluded because they were specular.

where G is a Gaussian centered on $S_{\theta,\Lambda}(x)$ with a variance of σ^2. The system of Eq. 10 is then solved by iterative re-substitution.

The M-step updates the model parameters taking $b(x)$ as expected visibility. In our case, this amounts to using them to weight the terms of eq 7. It can therefore be formulated as finding

$$(\check{\theta},\check{\Lambda}) = \underset{\theta,\Lambda}{\operatorname{argmin}} \left\|MI - MS_{\theta,\Lambda}\right\|^2 + \|K\Lambda\|^2 + E_D(\theta)\,, \tag{12}$$

where M is a diagonal matrix composed of all the $b(x)$ values.

The model as described above assumes conditional independence of pixel gray levels given the model parameters. While this may be reasonable for visible pixels that are correctly explained by the model so that only iid noise remains, this is certainly not true for occluded pixels that are not explained at all by the model. Furthermore, in the presence of complex illumination patterns, observing individual pixel intensities is not a discriminating enough criterion to assess visibility. If the current estimate of the Λ lighting factors is inaccurate over a local area, the individual residuals might be larger than they should and the pixels erroneously labeled as occluded. Conversely, when an occluded object has approximately the same color as the target surface or when local illumination tasks the color of the occluding object, occluded pixels can easily be missed.

4.2 Visibility and Normalized Cross-Correlation

Our goal is to alleviate the problems discussed above by improving the robustness of the visibility update equations of Eq. 10 to local illumination changes, so that potential inaccuracies during the M-step do not result in poor visibility estimates during the E-step. Our proposed solution is to replace the f function of Eq. 11 by a more robust one, which we will denote as d and takes into account neighborhoods as opposed to individual pixels.

To define d, let us first introduce the visibility weighted normalized cross-correlation between two vectors A and B

$$\gamma(A,B,w) = \frac{\sum_i w_i\left(A_i - \overline{A}\right)\left(B_i - \overline{B}\right)}{\sqrt{\sum_i w_i\left(A_i - \overline{A}\right)^2 \sum_i w_i\left(B_i - \overline{B}\right)^2}} \tag{13}$$

where w is a set of weights, and $\overline{A}$ and $\overline{B}$ denote the averages weighted by w over A and B respectively. Let W_x be a neighborhood around x that does not include x. As a shortcut, we write the vector of pixel intensities within W_x as:

$$I(W_x) = [I(y)|y \in W_x]$$

and we define $S_{\theta,\Lambda}(W_x)$ and $b(W_x)$ similarly. Likewise, we write

$$[1-b(W_x)] = [1-b(y)|y \in W_x]\ .$$

Finally, let

$$\begin{aligned} c(W_x, b(W_x)) &= R\ \max\left(0, \gamma\left(S_{\theta,\Lambda}(W_x), I(W_x), b(W_x)\right)\right) \\ R &= exp\left(-\sigma_c\left(\frac{\overline{S_{\theta,\Lambda}(W_x)}}{\overline{I(W_x)}} - 1\right)^2\right) \end{aligned}$$

where σ_c is a weighting constant. In practice, $c(W_x, b(W_x))$ tends to be close to 0 for occluded pixels and to 1 when model and input pixels match up to an affine transformation of their intensities, even if the precise lighting parameters are unknown. The term R penalizes a too large average difference. The $b(W_x)$ term prevents the correlation window W_x from crossing a visibility boundary.

Figure 3 shows a case where the $b(y)$ in x neighbordhood are binary. The part where the $b(y) = 1$ is a window over the visible area, while the rest is occluded. The two correlation values $c(W_x, b(W_x))$ and $c\left(W_x, [1-b(W_x)]\right)$ are expected to take a value close to 1 and 0 respectively, and are two possible clues for the $d(x, b(W_x))$ function we want to design. However, $d(x, b(W_x))$ is not function of $b(x)$, since $x \notin W_x$. To decide on which side x lies and to choose the correct value, we take the correlation measure which is the most compatible with x and define d as

$$d(x, b(W_x)) = \begin{cases} c\left(W_x, b(W_x)\right) & \text{if } \delta^+ < \delta^- \\ c\left(W_x, 1-b(W_x)\right) & \text{otherwise} \end{cases} \tag{14}$$

where

$$\begin{aligned} \delta^+ &= \left(c(W_x, b(W_x)) - c(W_x \cup x, [b(W_x), 1]\right)^2 \\ \delta^- &= \left(c(W_x, 1-b(W_x)) - c(W_x \cup x, [1-b(W_x), 1]\right)^2 \end{aligned}$$

We can now rewrite the visibility update equation of Eq. 10 as

$$b(x) = \sigma\left(\frac{2}{T}\sum_{y\in\Gamma(x)}(2b(y)-1) + \log\frac{d\left(x, b(W_x)\right) f\left(I(x)\right) P_f}{g\left(I(x)\right) P_g}\right)\ . \tag{15}$$

Solving this system using iterative re-substitution quickly converges to a sharp probability field with clearly defined visibility boundaries. Color images are handled by computing $d(.)$ for each channel and by multiplying them together. Compared to Eq. 10, Eq. 15 is more robust to illumination estimation errors.

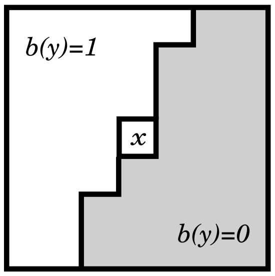


Figure 3: Correlation windows to compute correlation at pixel x.

5 Implementation Issues

The results shown in this paper were obtained using 1092×728 images and we used a correlation window of size 11×11 to perform the visibility computations of Section 4.2. We take P_f to be 0.9 and T to be 0.5 when computing the $b(x)$ probabilities of visibility by solving the coupled system of Eq. 10.

Our current implementation focuses on accuracy rather than speed. We did not spend much effort on computation efficiency: Processing an image takes 6 to 7 minutes on a modern PC. This could however be sped-up by several orders of magnitude as follows.

Jacobian precomputation The most time consuming part is the geometric registration of Section 3.3, in which a synthetic image S_θ is iteratively rendered to minimize its difference with the input image I. This registration is slow because we use straightforward Levenberg-Marquardt to minimize, every iteration of which requires a costly recomputing of the Jacobian. This computation burden could be reduced by using faster template matching techniques [1, 8]. These precompute directions in which to perform the line search given residuals. According to [1], this is at least one order of magnitude faster.

GPU vs CPU The synthetic image S_θ and its spatial derivative is obtained by linearly interpolating two pyramid levels that are sampled with bicubic interpolation. Although it requires 32 texture access for every pixel rendered, this rendering ensures image gradient continuity thanks to bicubic interpolation properties. The texture pyramid avoids sampling artifacts, and a continuous gradient helps the Levenberg-Marquardt algorithm to converge. Our current implementation only uses the CPU. Using the GPU instead could massively accelerate rendering and gradient computation. The dedicated and parallel nature of graphic hardware would provide an acceleration of at least one order of magnitude.

Constrained optimization Geometric constraints of Eq. 5 could be imposed during minimization by first determining the linear subspace locally satisfying them, and then performing within that subspace a Newton or Levenberg-Marquardt step minimizing the pixels' sum of square difference. This would improve algorithm's convergence properties and reduce the required number of iterations by a factor of at least two, according to our initial experiments.

Integral images The weighted cross-correlation of Eq. 13 can be computed with a complexity linear with the number of pixels and independent of the window size. Rearranging Eq. 13 gives:

$$\frac{\sum(w_iA_iB_i) - \overline{B}\sum w_iA_i - \overline{A}\sum w_iB_i + \overline{A}\overline{B}\sum w_i}{\sqrt{\sum w_iA_i^2 - 2\overline{A}\sum w_iA_i + \overline{A}^2\sum w_i}\sqrt{\sum w_iB_i^2 - 2\overline{B}\sum w_iB_i + \overline{B}^2\sum w_i}} . \tag{16}$$

Computing a sum over a rectangular area from an integral image takes a constant constant time. Thus, reducing the complexity of the above formula to constant time only requires to compute integral images of the following 8 images: A, B, w, wA, wB, wAB, wA^2, and wB^2. Computing at the same time $\gamma(A, B, 1-w)$ only requires 3 additional integral images: A^2, B^2 and AB. According to our preliminary tests, this implementation is about 15 times faster than our original one.

Combining all these improvements could potentially yield a speed increase of several orders of magnitude, which would lead to a frame rate of a few Herz on our 1092×728 images, or even faster using smaller images or parallel architectures that are now becoming common. Such an optimized implementation could then be used for interactive Augmented Reality.

6 Results

Our approach requires a model image in which the surface appears undeformed and a parametric model of its potential deformations. To validate it, we used the color postcards of Fig. 1 and 2 that we scanned to produce well-textured model images with homogeneous illumination. We represent the postcards as textured 600-vertex rectangular meshes associated with deformation modes we compute using the technique described in [14]. We use these models to demonstrate the robustness of our approach to occlusions and illumination changes.

6.1 Occlusion Handling

Figs. 1(b), 2(c), 4(b) demonstrate the importance of the occlusion mask to only render additional objects where they are expected to be visible, which is critical for augmentation purposes. In Fig. 1(c), we render the pixels that have been labeled as occluded using the original image texture and the others using the illumination model.

Note that the shape of the occluding finger is recovered almost everywhere to a ± 1 pixel accuracy. In Fig. 5, we compare our visibility map to the one we obtain when we use the original approach of [5] as described in Section 4.1, which turns out be much less reliable.

In our experience, these results are typical. Errors may occur when a large uniform surface occludes a uniform one of a similar color. Our method can also misclassify pixels that are subject to phenomena that the generative model of Section 3 does not take into account, such as motion blur, defocus, or specular reflections. The latter is what produces the tiny holes in the visibility map of Fig. 2. However, such mistakes are not frequent enough to prevent realistic augmentation. In fact, as shown in Fig. 6, our technique can even be used to remove limited motion blur from video images. Including explicit motion blur modeling into our generative model should allow us to handle far more.

6.2 Illumination Handling

To demonstrate our algorithm's ability to simultaneously handle changing lighting conditions and surface deformations, we acquired a video sequence of the postcard of Fig. 1 lit by both daylight coming through a window and an incandescent light nearby. In that configuration, simply rotating the postcard towards the window or the lamp changes the illuminant color. As shown in Fig. 7, the algorithm nevertheless continues to return accurate deformation and illumination parameters that are sufficiently accurate to realistically embed the virtual character into the original texture, even though a hand casts a shadow across the surface.

In this section we have decoupled the presentation of visibility computation and lighting parameters estimation. Nevertheless, as shown in Fig. 8, the two are intimately coupled. It is because we can compute good visibility masks that we can also recover meaningful illumination parameters.

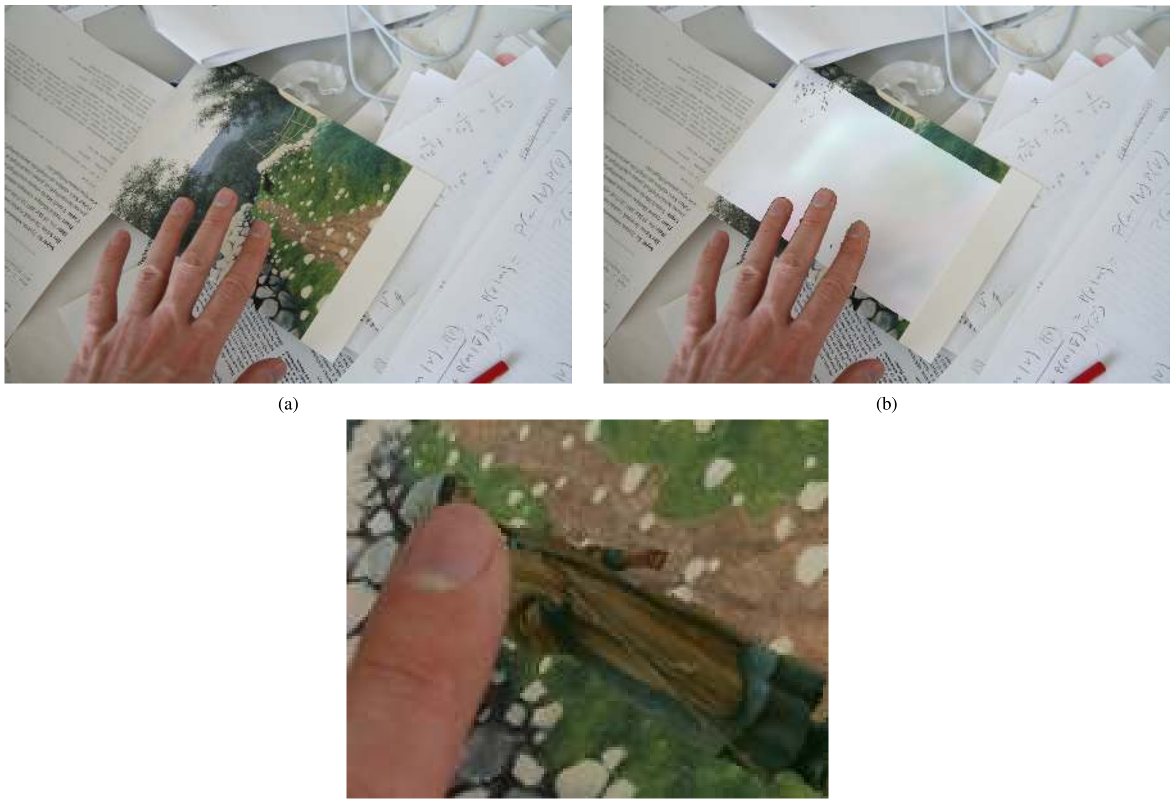

(a) (b) (c)

Figure 4: Handling a large occlusion. (a) The postcard of Fig. 1 is occluded by several fingers. (b) The diminished reality version of the postcard. (c) A detail of the augmented postcard. The visibility map is computed accurately enough so that rendering the virtual character under the finger appears realistic.

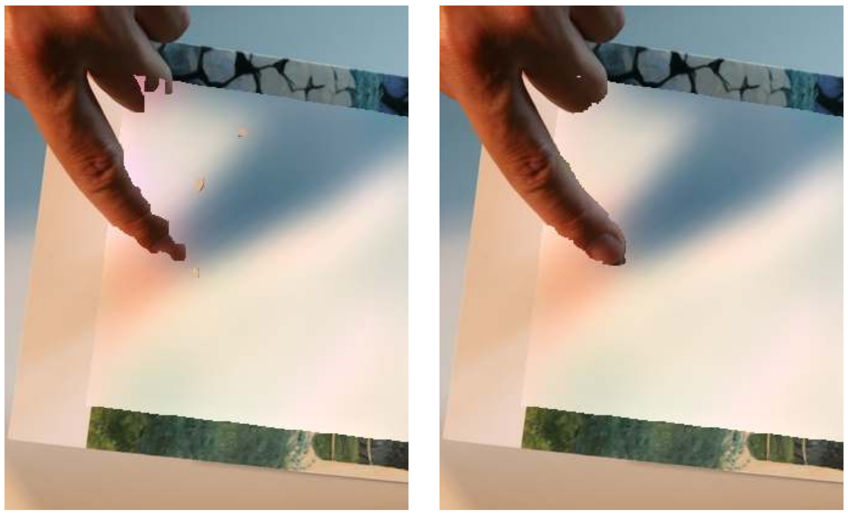
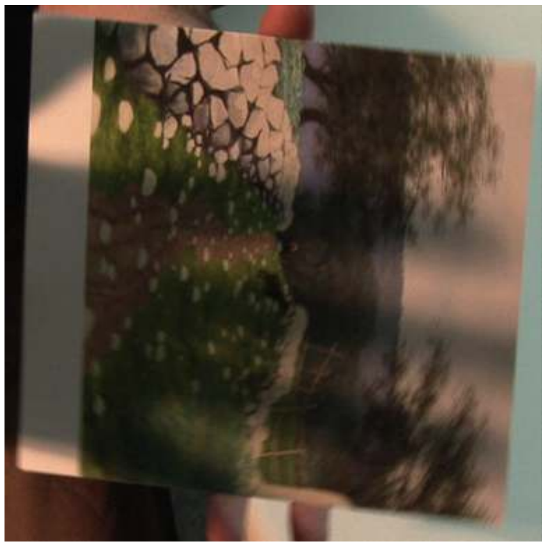

(a) (b)

Figure 5: (a) The visibility map of Fig. 1 computed using the original technique of [5] (b) Visibility map as computed by the algorithm of section 4.2.


(a)

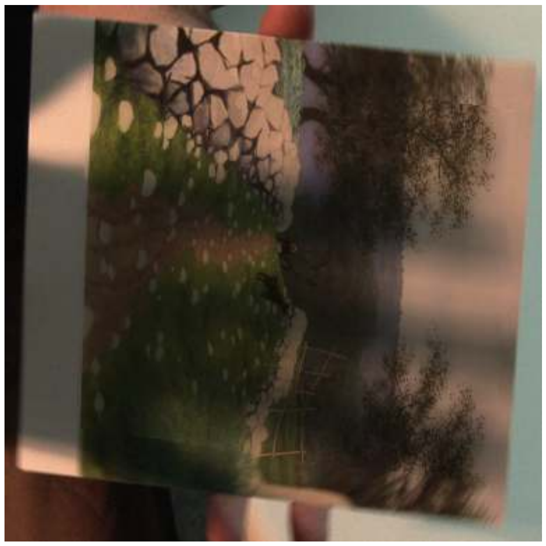

(b)

Figure 6: Removing motion blur. (a) Input image from a video sequence of the deforming postcard, with some motion blur and a complex shadow. (b) The model image is warped using the recovered motion and illumination parameters. The motion blur has disappeared, which is particularly visible on the tree, while the shadows are correctly reproduced.

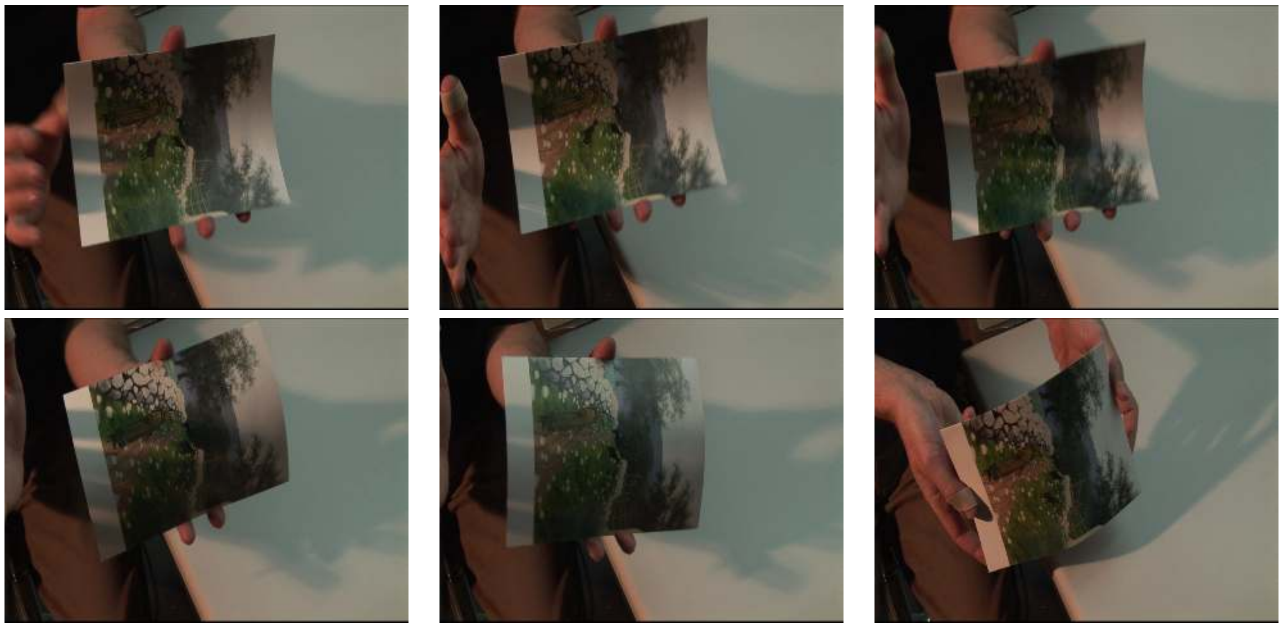


Figure 7: Simultaneously handling changing illumination, cast shadows, and surface deformations. The postcard of Fig. 1 is lit by both daylight and incandescent light while being rotated and deformed. The virtual character still blends smoothly into the real postcard even though the changes in orientation results in a change of lighting color and a hand casts a shadow on the surface.

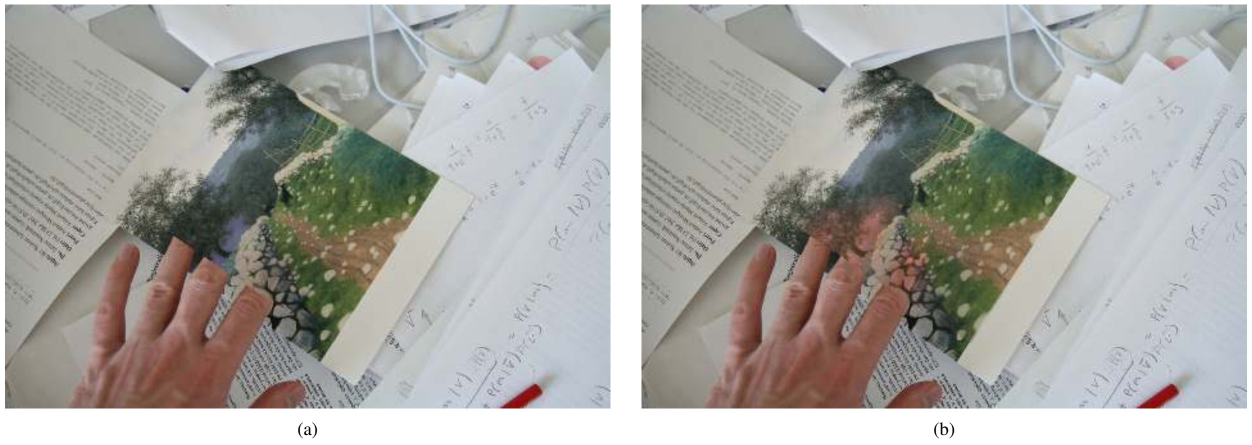


Figure 8: Interplay of the visibility computation and recovery of the illumination parameters. (a) Image synthesized by warping the model image of Fig. 1 using the pose and illumination parameters recovered from the image of Fig. 4(a) using our complete method. (b) The model image is warped using the same geometric pose. However, occlusion detection has been switched off and the recovered illumination parameters are corrupted by the occluding fingers, which results in an unrealistic rendering.

7 Conclusion

We have proposed a method for registering deformable surfaces that is robust both to occlusions and complex illumination effects. It returns a lighting model that can be used to relight the objects to be added to the scene and a visibility map that lets us draw them only at appropriate locations. The key components of our approach are a lighting model that is flexible enough to handle arbitrary lighting patterns and an algorithm for visibility estimation that takes into account whole neighborhoods instead of individual pixels.

One weakness of the current approach is that it requires relatively textured surfaces to produce accurate results. However, since we recover local illumination models, we should be able to take advantage of shading information in untextured parts of the surface, thereby increasing the range of applicability of our method. This will be the subject of future work.

References

[1] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, pages 221–255, March 2004.

[2] A. Bartoli and A. Zisserman. Direct Estimation of Non-Rigid Registration. In *British Machine Vision Conference*, Kingston, UK, September 2004.

[3] D. Bradley and G. Roth. Augmenting Non-Rigid Objects with Realistic Lighting. Technical Report NRC/ERB-1116, Oct. 2004.

[4] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active Appearance Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6), jun 2001.

[5] Rik Fransens, Christoph Strecha, and Luc Van Gool. A mean field em-algorithm for coherent occlusion handling in map-estimation prob. In *Conference on Computer Vision and Pattern Recognition*, 2006.

[6] Ralph Gross, Iain Matthews, and Simon Baker. Active appearance models with occlusion. *Image and Vision Computing*, 24(6):593–604, 2006.

[7] J.Pilet, V. Lepetit, and P. Fua. Fast non-rigid surface detection, registration and realistic augmentation. *International Journal of Computer Vision*, January 2007. In press.

[8] F. Jurie and M. Dhome. A simple and efficient template matching algorithm. In *International Conference on Computer Vision*, Vancouver, Canada, July 2001.

[9] Wen-Chieh Lin and Yanxi Liu. Tracking dynamic near-regular textures under occlusion and rapid movements. In *European Conference on Computer Vision*, May 2006.

[10] B. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[11] J. Pilet, V. Lepetit, and P. Fua. Augmenting Deformable Objects in Real-Time. In *International Symposium on Mixed and Augmented Reality*, Vienna, October 2005.

[12] D. Pritchard and W. Heidrich. Cloth motion capture. In *Eurographics*, volume 22(3), pages 263–271, September 2003.

[13] S. Romdhani and T. Vetter. Efficient, robust and accurate fitting of a 3d morphable model. In *International Conference on Computer Vision*, Nice, France, 2003.

[14] M. Salzmann, S. Ilić, and P. Fua. Physically Valid Shape Parameterization for Monocular 3–D Deformable Surface Tracking. In *British Machine Vision Conference*, Oxford, UK, September 2005.

[15] M. Salzmann, J. Pilet, S. Ilić, and P. Fua. Surface Deformation Models for Non-Rigid 3–D Shape Recovery. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007. In press.

[16] Volker Scholz and Marcus Magnor. Texture replacement of garments in monocular video sequences. In *Eurographics Symposium on Rendering*, 2006.

[17] Stan Sclaroff and John Isidoro. Active blobs: region-based, deformable appearance models. *Computer Vision and Image Understanding*, 89(2-3), 2003.

[18] R. White and D.A. Forsyth. Combining cues: Shape from shading and texture. In *Conference on Computer Vision and Pattern Recognition*, 2006.

[19] R. White and D.A. Forsyth. Retexturing single views using texture and shading. In *European Conference on Computer Vision*, volume LNCS 3954, pages 70–81, 2006.