Convex Optimization for Deformable Surface 3–D Tracking

Mathieu Salzmann
CVLab
EPFL, Switzerland
mathieu.salzmann@epfl.ch

Richard Hartley
RSISE
ANU, Australia
richard.hartley@anu.edu.au

Pascal Fua
CVLab
EPFL, Switzerland
pascal.fua@epfl.ch

Abstract

3–D shape recovery of non-rigid surfaces from 3–D to 2–D correspondences is an under-constrained problem that requires prior knowledge of the possible deformations. State-of-the-art solutions involve enforcing smoothness constraints that limit their applicability and prevent the recovery of sharply folding and creasing surfaces.

Here, we propose a method that does not require such smoothness constraints. Instead, we represent surfaces as triangulated meshes and, assuming the pose in the first frame to be known, disallow large changes of edge orientation between consecutive frames, which is a generally applicable constraint when tracking surfaces in a 25 frames-per-second video sequence. We will show that tracking under these constraints can be formulated as a Second Order Cone Programming feasibility problem. This yields a convex optimization problem with stable solutions for a wide range of surfaces with very different physical properties.

1. Introduction

Using point correspondences to recover the 3–D shape of deformable surfaces from a single video is a severely under-constrained problem: Surfaces with very different shapes can look very similar under perspective projection.

This problem has been addressed by introducing *a priori* shape models. Structure from motion techniques [18, 10, 22] and physics-based models [13, 14, 3, 5, 12] have been proposed to retrieve the deformations of non-rigid surfaces. However, these approaches typically incorporate regularization terms or make linearity assumptions that constrain the surface to deform smoothly. Machine learning methods [2, 15, 11] present an attractive alternative, but most of them assume a linear mapping from a low-dimensional manifold to the high-dimensional data space, which again imposes smoothness constraints and is not always accurate. Non-linear dimensionality reduction techniques would avoid such problems, however they require training data which often is not available.

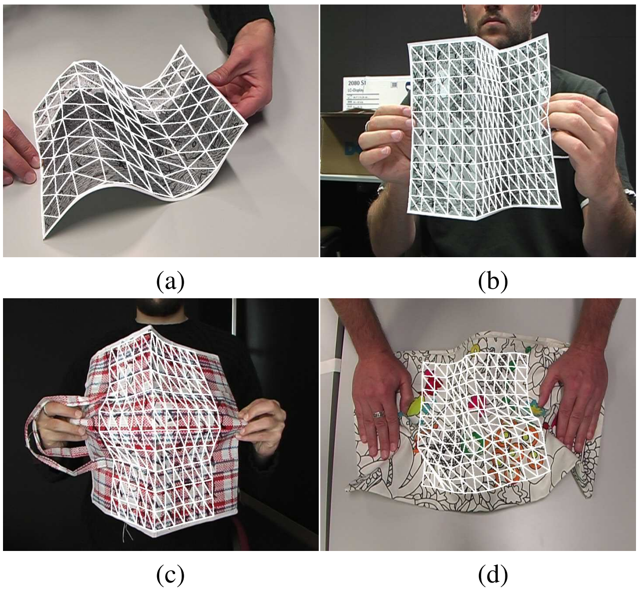


Figure 1. Reconstructing highly deformable surfaces from single video sequences. Because it does not impose smoothness constraints, our algorithm can use the same parameters to handle surfaces with very different physical properties. (a) Smoothly deforming paper. (b) Paper with two well marked creases. (c) Plastic bag. (d) Piece of cloth.

Existing techniques are therefore not adapted to track 3–D surfaces such as those of Fig. 1 that can fold or produce sharp creases at unpredictable locations. In this paper, we follow work on rigid objects [6, 8, 16] and show that recovering 3–D surface shape from point correspondences can also be described as a Second Order Cone Programming (SOCP) problem. This yields a convex formulation with a unique minimum and lets us handle highly-deformable surfaces without adding unwarranted smoothness constraints.

The central element of our approach is first to formulate the constraints imposed by the correspondences as SOCP constraints and second to introduce appropriate additional constraints that force the solution of the SOCP problem to be physically plausible. In practice, we simply disallow overly large edge orientation changes from one frame to the next and excessive edge stretching. This is generally applicable when tracking in a 25 frames-per-second video sequence. It has therefore proved effective at handling surfaces as different as the smoothly deforming piece of paper, the same piece paper with sharp creases in it, and the much more flexible plastic bag and piece cloth of Fig. 1.

2. Related Work

3–D shape recovery from a monocular video sequence is an underconstrained problem. Thus, over the years, people have proposed many approaches to introducing *a priori* knowledge and resolving the ambiguities, most of which make very strong and restrictive assumptions about the object of interest.

In this section, we briefly review these methods. We then discuss the Second Order Cone Programming framework that has recently been introduced in our field to improve the solvability of a number of classical problems and serves as the basis for our own approach.

2.1. Deformable Shape Recovery

Structure from motion methods that rely on tracked feature points have proved to be effective. However, modeling deformations as a linear combination of constant basis vectors [10, 18] greatly oversimplifies the general behavior of a surface. Furthermore, describing the problem as several rigid objects moving with respect to one another [22] is not adapted to model the deformations of materials such as cloth or thin plastic sheets.

Similarly, physics-based models have been a very popular way to introduce *a priori* knowledge. The original 2–D models were first applied to shape recovery [7]. They have been used for 2–D surface registration [1] and were rapidly extended to various 3–D formulations [3, 13, 12]. To reduce the dimensionality of the models, linearity assumptions have also been made through modal analysis [14, 3, 5]. Even though the physics-based approach has been extremely successful, it implies some knowledge of the pseudo-physical properties of the surface, which may not be available. Furthermore, the complexity of modeling a true nonlinear behavior tends to restrict these methods to cases where nonlinearities are small. In fact, modeling a true nonlinear behavior is a complex task subject to convergence instabilities well-known in the mechanical engineering community.

The complexity of modelling the true physical properties of surfaces has made statistical learning techniques attractive. Training data is used to build a deformation model to apply to new data. However, most existing techniques consider the sub-space defined by the training data to be linear [4, 11, 2, 15], which is a restrictive assumption. Nonlinear learning has been demonstrated for human body tracking [19] but involves a complex objective function that may be difficult to optimize.

Another approach to disambiguating the surface recovery problem is to use richer sources of information. For example, it was recently shown that texture and shading information could be combined to retrieve the shape of a deformable surface [21]. However, very strong assumptions on the lighting environment must be made, and therefore the method lacks generality.

2.2. Convex Optimization

It was recently shown that several computer vision problems such as triangulation, camera resectioning and homography estimation can be formulated as Second Order Cone Programming feasibility problems [6, 8]. These are particular types of convex optimization problems where no function is minimized. Instead, one looks for a vector $\mathbf{X}$ that satisfies the m constraints

$$\|\mathbf{A_i}\mathbf{X} + \mathbf{b_i}\|_2 \leq (\mathbf{c_i}^{\mathrm{T}}\mathbf{X} + d_i) \ , \text{ for } i = 1, ..., m \ . \quad (1)$$

Such problems can be solved very effectively using available packages such as SeDuMi [17]. The drawback of these initial formulations is their sensitivity to outliers, but a solution to this weakness was proposed in [16].

However, such formulation has only been demonstrated for rigid objects. Here, we extend these approaches to deformable 3–D surfaces.

3. SOCP for Non-Rigid Meshes

In this section, we show that recovering the 3–D shape of a deformable surface from a single video sequence can be formulated as an SOCP problem, as described by Eq. 1.

The surface is represented as a triangulated mesh whose vertices positions we want to retrieve in each frame of the sequence. We automatically establish 3–D to 2–D correspondences between the first frame, where the 3–D pose is assumed to be known, and the others by first tracking the surface in 2–D using normalized cross-correlation. We then compute correspondences by picking random samples in each facet and looking in each frame in an area limited by the 2–D tracking result for 2–D points matching their projections in the first frame. To this end we use standard cross-correlation, which gives us a set of potentially noisy 3–D to 2–D correspondences between surface points $\mathbf{x_i}$ and image points $(\hat{u}_i, \hat{v}_i)$, which we will use to estimate the 3–D coordinates of the mesh vertices at each time t as described below. We also assume that the camera projection matrix $\mathbf{P}$ is known and remains constant. This does not mean that the camera cannot move, but that we can only recover a relative motion of the surface with respect to it.

3.1. Correspondences as SOCP Constraints

Let $\mathbf{x_i}$ be the 3–D coordinates of a surface point that we express in terms of its barycentric coordinates of the facet j inside which it lies as

$$\mathbf{x_i} = a_i\mathbf{v_{j,1}} + b_i\mathbf{v_{j,2}} + c_i\mathbf{v_{j,3}} \ , \tag{2}$$

where $\mathbf{v_{j,k}}$ is the k^{th} vertex of facet j. The projection of $\mathbf{h_i} = (\mathbf{x_i}^{\mathrm{T}}\ 1)^{\mathrm{T}}$ given the camera projection matrix $\mathbf{P}$ is

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} \frac{\mathbf{P_1h_i}}{\mathbf{P_3h_i}} \\ \frac{\mathbf{P_2h_i}}{\mathbf{P_3h_i}} \end{pmatrix} \ ,$$

where $\mathbf{P_k}$ refers to the k^{th} row of the projection matrix. We define the reprojection error with respect to an image measurement $(\hat{u}_i\ \hat{v}_i)^{\mathrm{T}}$ as

$$\|\frac{\mathbf{P_1h_i}}{\mathbf{P_3h_i}} - \hat{u}_i, \frac{\mathbf{P_2h_i}}{\mathbf{P_3h_i}} - \hat{v}_i\| = \frac{\|(\mathbf{P_1} - \hat{u}_i\mathbf{P_3})\mathbf{h_i}, (\mathbf{P_2} - \hat{v}_i\mathbf{P_3})\mathbf{h_i}\|}{\mathbf{P_3h_i}} \ . \tag{3}$$

Ideally, we would want the reprojection error to be zero for all $\mathbf{x_i}$, $1 \leq i \leq m$ for which we have found a corresponding $(\hat{u}_i, \hat{v}_i)$ image point. In practice, due to noise, this is never possible. Therefore, as in [6], we introduce an additional variable γ and write our problem as

$$\begin{aligned} \min_{\gamma,\mathbf{V}} \gamma \ \ & \text{subject to } \gamma \geq 0 \text{ and} \\ & \|(\mathbf{P_1} - \hat{u}_i\mathbf{P_3})\mathbf{h_i}, (\mathbf{P_2} - \hat{v}_i\mathbf{P_3})\mathbf{h_i}\| \leq \gamma\mathbf{P_3h_i} \ , \\ & \text{for } i = 1, ..., m \ . \end{aligned} \tag{4}$$

where $\mathbf{V}$ is the concatenation of the three coordinates of all the mesh vertices. Intuitively, γ represents the radius on the image plane of the cone centered in the camera and whose axis goes through the image measurement.

For a fixed value of γ, because the $\mathbf{x_i}$ of Eq. 2 and therefore also the $\mathbf{h_i}$ are linear combinations of the vertex coordinates, Eq. 4 defines an SOCP problem of the form described by Eq. 1. We can then find the minimal γ using a bisection algorithm at each step of which we solve the corresponding SOCP feasibility problem.

Note that the fact that the 3–D points are on the surface of the mesh plays a critical role. Without this constraint, they could move independently from each other. Since nothing would then prevent them to match their 2–D projection within a zero-radius cone, this would result in a perfect but meaningless solution. However, forcing them to remain on the surface of the deformable mesh avoids that problem, since the barycentric coordinates that define the 3–D points impose a natural coherence between them.

A common criticism of SOCP formulations of the correspondence problem is that they are very sensitive to outliers. Indeed, the minimal γ will take the value that is a function of the worst correspondence, therefore allowing the reprojection error of correct matches to be worse than it should. However, Sim et al. [16] proposed a method to remove the outliers and get the correct pose of a rigid object using an SOCP approach. They showed that, at the end of the bisection algorithm, the set of matches whose reprojection error equals the minimal γ contains outliers. Therefore, removing these points and re-optimizing in the same manner as before yields a better pose. In our implementation, we apply the same idea and iterate the bisection algorithm with the correspondences having a reprojection error less than the previous minimal γ, until we reach a maximal reprojection error of 2 pixels. In practice, this only implies running the bisection algorithm at most 5 times.

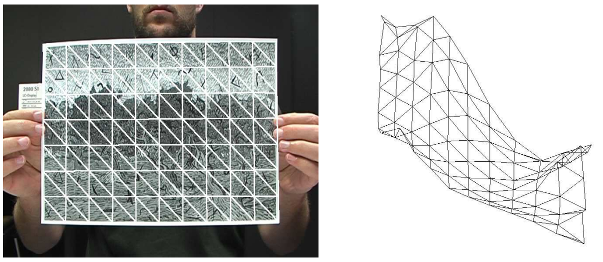

(a) (b)

Figure 2. Reconstructing a piece of paper using only the correspondences constraints of Section 3.1 but not the deformation constraints of Section 3.2. (a) The reprojection of the mesh is correct. (b) However, the 3–D shape as seen from a side view is completely wrong because the depth ambiguities are not properly resolved.

3.2. Additional Constraints

In general, solving the minimization problem of Eq. 4 without additional constraints yields a surface whose points project at the right place but whose overall shape may nevertheless be wrong, as shown in Fig. 2. The global scale of the surface can vary without affecting the reprojection error, and, more damagingly, given noisy data, the depth of the vertices is hard to precisely estimate because many different shapes can yield very similar projections.

Penalty functions have often been used to address this problem. They are usually designed either to prevent the mesh from folding sharply or to stop it from expanding or shrinking. The former results in a loss of generality as surfaces that crease cannot be modeled properly, while the latter typically involves a non-convex term to force the edges of the mesh to retain their original length. Here, we introduce a weaker and more generic constraint that fits into our SOCP framework: As shown in Fig. 3, we avoid the orientation of the edges to change irrationally between two consecutive frames. In the meantime, our constraints also ensure that an edge will not stretch or compress too much, thus partially solving the global scale ambiguity. Independently of the surface's curvature, this is generally applicable when tracking it in a 25 frames-per-second video sequence.

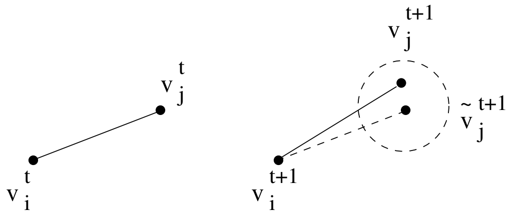


Figure 3. We predict that the orientation of the edge between $\mathbf{v_i}$ and $\mathbf{v_j}$ at time $t+1$ will be the same as at time t. We then constrain the distance between vertex $\mathbf{v_j^{t+1}}$ and its prediction $\tilde{\mathbf{v}}_\mathbf{j}^\mathbf{t+1}$ to be less than some specified value.

Furthermore, it can be expressed as a convex constraint as follows.

Let us assume that we know the shape of the mesh at time t and let us consider an edge linking vertices $\mathbf{v_i^t}$ and $\mathbf{v_j^t}$ in this configuration. Assuming that the orientation of this edge will be similar at time $t+1$, if the position $\mathbf{v_i^{t+1}}$ of vertex i at that time were known, we could predict that of vertex j to be close to

$$\tilde{\mathbf{v}}_\mathbf{j}^\mathbf{t+1} = \mathbf{v_i^{t+1}} + L_{i,j}\frac{\mathbf{v_j^t} - \mathbf{v_i^t}}{\|\mathbf{v_j^t} - \mathbf{v_i^t}\|} \ , \tag{5}$$

where $L_{i,j}$ is the original length of the edge. In practice we do not know $\mathbf{v_i^{t+1}}$ but we can nevertheless require that

$$\|\mathbf{v_j^{t+1}} - \tilde{\mathbf{v}}_\mathbf{j}^\mathbf{t+1}\| \leq \lambda L_{i,j} \ , \tag{6}$$

where $\tilde{\mathbf{v}}_\mathbf{j}^\mathbf{t+1}$ is defined in Eq. 5. This constraint fits perfectly within our SOCP framework and we can add one for each edge to those of Eq. 4. Note that these additional constraints allow the mesh to expand or shrink, but only within an amount controlled by the value of λ. Given a mesh that satisfies the SOCP constraints, we handle the remaining scale ambiguity by rescaling it so that its area remains the same as in the initial position. As will be shown in Section 4, this results in a system that is now sufficiently constrained to yield good results.

4. Experimental Results

In the previous section, we showed how convex optimization can be applied to the problem of recovering the 3–D shape of a surface from a single video sequence. We presented constraints that do not prevent the surface from folding sharply. Our method relies on 3–D to 2–D correspondences and only requires the pose in the first frame of the sequence to be known. This might be seen as a limitation, however automated initialization of deformable surfaces from single images is a severely under-constrained problem that requires much stronger deformation models than ours. Such models could be learned from training data if it were available, which is rarely the case. Since our method makes few assumptions on the physical behavior of the surface, it is ideal to acquire such data.

We first validate our approach using synthetic data. We then use ordinary videos to demonstrate that it produces good results for very different kinds of materials. In all our experiments, both for synthetic and real data, the value of λ in Eq. 6 was set to 0.1, independently of the properties of the surface and of its deformations.

4.1. Synthetic Data

As a first experiment, we synthetically deformed the 88-vertex mesh shown in Fig. 4 by applying forces to randomly chosen vertices and strongly penalizing stretching of its edges. This produced a sequence of 50 different shapes, from which we could obtain correspondences by projecting 3–D points defined by their randomly chosen barycentric coordinates using a perspective projection matrix. We then added gaussian noise with mean zero and variance one and two to their image locations. Fig. 4 shows the reconstruction results for variance two from a different perspective. The differences are very small even though the surface folds very sharply in some frames. The largest errors are in the depth direction, as could be expected since motion in that direction is hard to measure using point correspondences. The deformation constraints of Section 3.2 resolve most of the resulting ambiguities but still leave some uncertainty.

To compare the performance of SOCP against another powerful optimization technique, we reimplemented our tracking algorithm using CFSQP [9], which provides C functions to solve constrained minimization problems using Sequential Quadratic Programming. We reformulated our problem as the minimization of the sum of squared reprojection errors under the deformation constraints of Section 3.2. The top row of Fig. 5 shows the median vertex-to-ground-truth-surface distances for noise variances 1 and 2 for each one of the frames in the synthetic sequence. The distances are of the order of 0.1cm for a mesh of size 10cm×7cm. Both the SOCP and CFSQP implementations produced roughly comparable errors. However, even though SOCP was coded in `Matlab` whereas CFSQP was coded in C, SOCP was about 50 times faster than CFSQP: It took only 15 minutes against 12 hours to process the whole sequence on the same 3.0 GHz PC. The second row of Fig. 5 shows the median reprojection errors over the correspondences. CFSQP yields slightly higher accuracy, but the errors still remain under one pixel for SOCP. This can be explained by the fact that SOCP does not minimize the reprojection errors, but finds a solution such that these errors are smaller than a given value.

Since CFSQP can handle non-convex constraints, we replaced the deformation constraints of Section 3.2 by constraints that prevent the mesh edges from changing their length. In theory, this should be more appropriate when

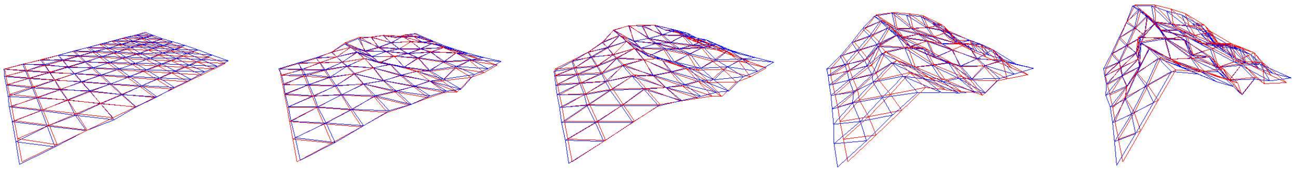
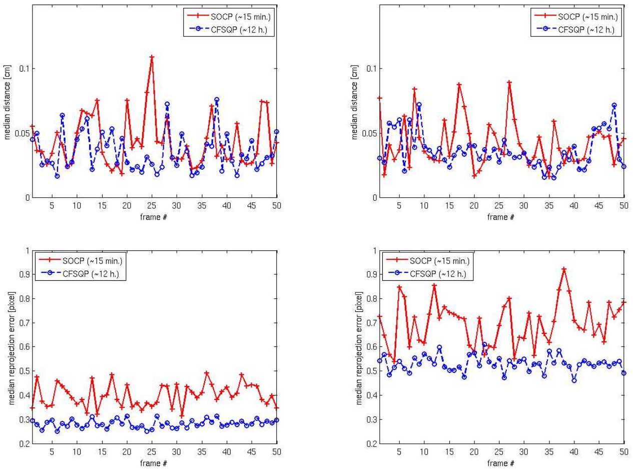
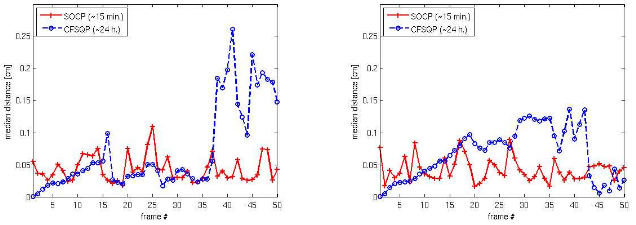


Figure 4. Reconstructing an 88-vertex mesh with sharp folds using perfect correspondences that were corrupted using zero-mean Gaussian noise with variance two. The shape of the reconstructed mesh (blue) corresponds very closely to the original one (red). The meshes are seen from a different perspective than the one used to retrieve the shapes in order to highlight the differences.



$\sigma = 1$ $\sigma = 2$

Figure 5. We compare the results of our SOCP formulation (solid red) against those obtained using CFSQP, a constrained nonlinear least-squares minimization (dashed blue) for the 50 frames of the synthetic sequence of Fig. 4, for noise variance $\sigma = 1$ and 2. In the top row, we show the distance between the original mesh and its reconstruction. Both methods give similar results but SOCP is about 50 times faster. In the second row, we give the median reprojection errors. For both methods, they are less than one pixel, even though CFSQP performs slightly better. Recall, however, that SOCP does not precisely minimze the reprojection errors, but enforces the reprojections to lie in a cone of a given radius.



$\sigma = 1$ $\sigma = 2$

Figure 6. Introducing non-convex inextensibility constraints, for noise variance $\sigma = 1$ and 2. Since CFSQP can handle such constraints, we introduce them into our CFSQP formulation and, as in Fig. 5, compare the results (in blue) against those of SOCP (in red). In addition to being much slower, CFSQP gives unstable results and fails to converge in some frames after 2000 iterations.

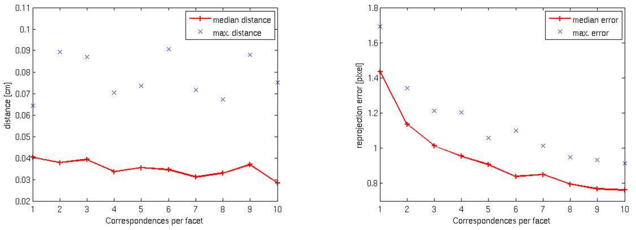


Figure 7. Influence of the number of correspondences in each facet on the reconstruction for noise variance $\sigma = 2$. We decreased the number of correspondences per facet from 10 to 1, and display the median (red line) and maximum (blue crosses) values of the same errors as in Fig. 5. We show the 3–D distance errors in the left image, and the reprojection errors in the right one. Note that the 3–D vertex-to-surface distance is little affected by the correspondences, whereas the reprojection error decreases in a more noticeable manner.

tracking inextensible surfaces. In practice, as shown in Fig. 6, even though CFSQP performs better in some frames, it is less stable than SOCP. This is particularly visible towards the end of the sequence. In some frames, CFSQP failed to converge even after 2000 iterations, which explains why it is even slower than before and highlights the complexity of the problem when non-convex constraints are used.

Finally, in Fig. 7, we show the influence of the number of correspondences on the quality of our reconstruction in the case of a variance 2 gaussian noise. We decreased the number of correspondences in each facet from 10 to 1 and tracked the surface throughout the 50 frames of the sequence. For each frame, we computed the median vertex-to-ground-truth distance and median reprojection error. For each number of correspondences per facet, we display on the left image, the median and maximum values of such 3–D distances over the sequence, and on the right image, the median and maximum values of such reprojection errors over the sequence. The number of correspondences has little influence on the 3–D distances, since the vertices can slide along the true surface without changing these measures. The reprojection errors are more strongly affected, but note that, from 4 correspondences per facet, they drop below one pixel. Of course, this still assumes at least one correspondence in each facet. With such a weak deforma-

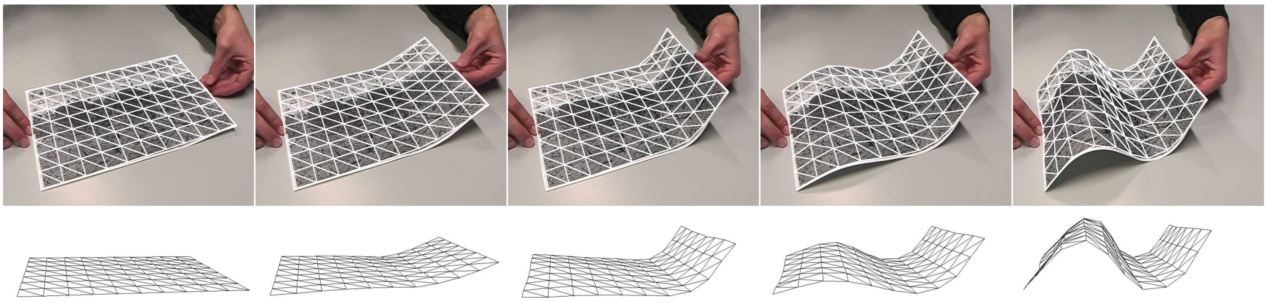


Figure 8. Reconstructing a deforming sheet of paper from a 116 frames video. The mesh is reprojected in the image in the top row and seen from a different perspective in the bottom one. Even though no smoothness constraint was enforced, the algorithm correctly recovered smooth deformations.

tion model, if some facets did not contain any, the reconstruction would inevitably degrade. This would especially be the case for facets on the boundary of the surface, since their vertices are constrained by fewer neighbors than the ones in the middle.

4.2. Real Data

We now show reconstruction results of real deformable surfaces made of paper, cloth, and plastic. The video sequences were acquired with an ordinary digital camera. Due to their very different physical properties, the behavior of the surfaces ranges from smooth deformations for the paper to sharp folds and creases for the cloth and plastic. However, no parameter tuning was necessary to obtain these results with our algorithm. Videos of all the examples of the section are submitted as supplementary material.

Smooth Deformations As a first experiment on real data, we considered a sheet of paper that we modeled as an 88-vertex mesh. Fig. 8 shows that we can retrieve the correct shape of a surface that deforms smoothly, even though our formulation involves no explicit penalty term on the curvature of the reconstructed surface.

Sharper Folds Because we do not penalize curvature, nothing stops our method from recovering the correct shape in the presence of folds and creases, as demonstrated by the reconstruction of the pre-folded sheet of paper of Fig. 9, the plastic bag of Figs. 10 and 11, and the piece of cloth of Figs. 12 and 13. All these examples would create problems for approaches to surface reconstruction that impose strong smoothness constraints.

5. Conclusion

In this paper, we have presented an approach to retrieving the shape of a deformable surface from a single video using convex optimization in the form of Second Order Cone Programming. To this end, we have formulated both the constraints imposed by the 3–D to 2–D correspondences and the additional shape constraints required to handle the depth ambiguities as SOCP constraints. These shape constraints restrict the motion from one frame to the next but do not impose unwarranted surface smoothness. This lets us recover sharp folds and creases that would create problems for most standard techniques, and yields much faster convergence than constrained nonlinear least-square optimization for quantitatively equivalent results.

In future work, we intend to show that a wider array of constraints can also be expressed in a convex optimization framework such as SOCP or the more general Semidefinite Programming, so that additional sources of image information can be exploited. We will therefore explore approaches to reparameterizing the reconstruction problem in terms of variables that result in convex formulations of constraints that initially were not, as was done in [20] in the context of image manifold learning under local isometry assumptions. We further intend to build stronger deformation models of lower dimensionality by learning them from training data. Since such data is rarely available, we see the proposed approach as a practical solution to the problem of generic data acquisition.

References

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

[2] V. Blanz and T. Vetter. A Morphable Model for The Synthesis of 3–D Faces. In *ACM SIGGRAPH*, pages 187–194, Los Angeles, CA, August 1999.

[3] L. Cohen and I. Cohen. Deformable models for 3-d medical images using finite elements and balloons. In *Conference on Computer Vision and Pattern Recognition*, pages 592–598, 1992.

[4] T. Cootes, G. Edwards, and C. Taylor. Active Appearance Models. In *European Conference on Computer Vision*, pages 484–498, Freiburg, Germany, June 1998.

[5] H. Delingette, M. Hebert, and K. Ikeuchi. Deformable surfaces: A free-form shape representation. In *SPIE Geometric Methods in Computer Vision*, volume 1570, pages 21–30, 1991.

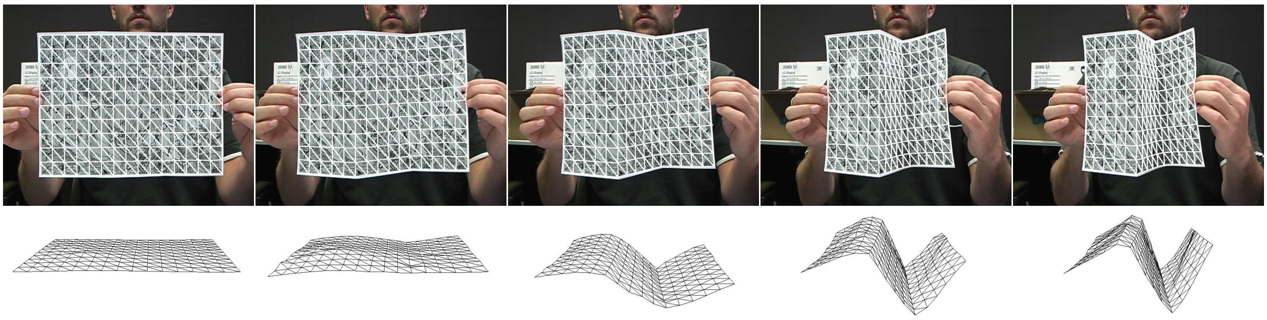


Figure 9. Reconstructing the deformations of a piece of paper with two sharp folds in it, so that that they are no longer smooth. Note that our method correctly recovers the creases.

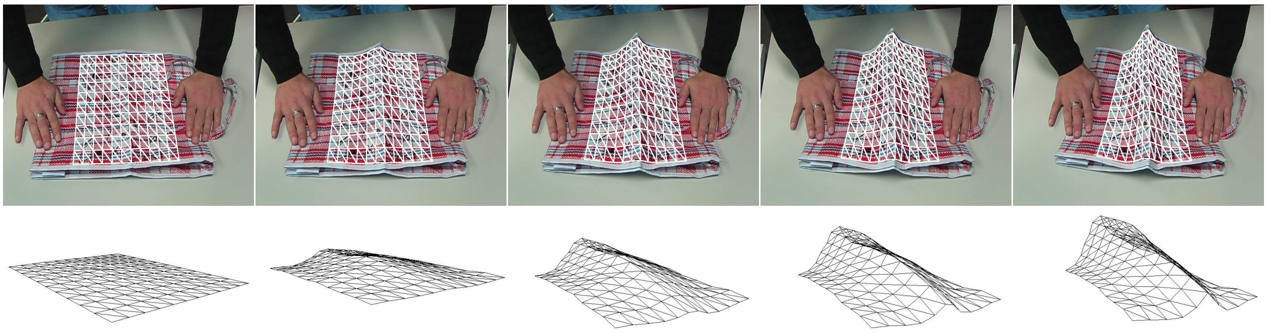


Figure 10. Recovering the deformations of a plastic bag with a sharp crease in it from from an 86 frames video.

[6] F. Kahl. Multiple view geometry and the L_∞-norm. In *International Conference on Computer Vision*, pages 1002–1009, Beijing, China, 2005.

[7] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[8] Q. Ke and T. Kanade. Quasiconvex optimization for robust geometric reconstruction. In *International Conference on Computer Vision*, pages 986–993, 2005.

[9] C. Lawrence, J. L. Zhou, and A. L. Tits. User's guide for cfsqp version 2.5: A c code for solving (large scale) constrained nonlinear (minimax) optimization problems, generating iterates satisfying all inequality constraints.

[10] X. Llado, A. D. Bue, and L. Agapito. Non-rigid 3D Factorization for Projective Reconstruction. In *British Machine Vision Conference*, Oxford, UK, September 2005.

[11] I. Matthews and S. Baker. Active Appearance Models Revisited. *International Journal of Computer Vision*, 60:135–164, November 2004.

[12] T. McInerney and D. Terzopoulos. A Finite Element Model for 3D Shape Reconstruction and Nonrigid Motion Tracking. In *International Conference on Computer Vision*, pages 518–523, Berlin, Germany, 1993.

[13] D. Metaxas and D. Terzopoulos. Constrained deformable superquadrics and nonrigid motion tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):580–591, 1993.

[14] A. Pentland. Automatic extraction of deformable part models. *International Journal of Computer Vision*, 4(2):107–126, 1990.

[15] 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.

[16] K. Sim and R. Hartley. Removing outliers using the L_∞ norm. In *Conference on Computer Vision and Pattern Recognition*, pages 485–494, Washington, DC, 2006.

[17] J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones, 1999.

[18] L. Torresani, A. Hertzmann, and C. Bregler. Learning nonrigid 3d shape from 2d motion. In *Advances in Neural Information Processing Systems*. MIT Press, Cambridge, MA, 2003.

[19] R. Urtasun, D. Fleet, A. Hertzman, and P. Fua. Priors for people tracking from small training sets. In *International Conference on Computer Vision*, Beijing, China, 2005.

[20] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Conference on Computer Vision and Pattern Recognition*, June 2004.

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

[22] J. Xiao and T. Kanade. Uncalibrated perspective reconstruction of deformable structures. In *International Conference on Computer Vision*, 2005.

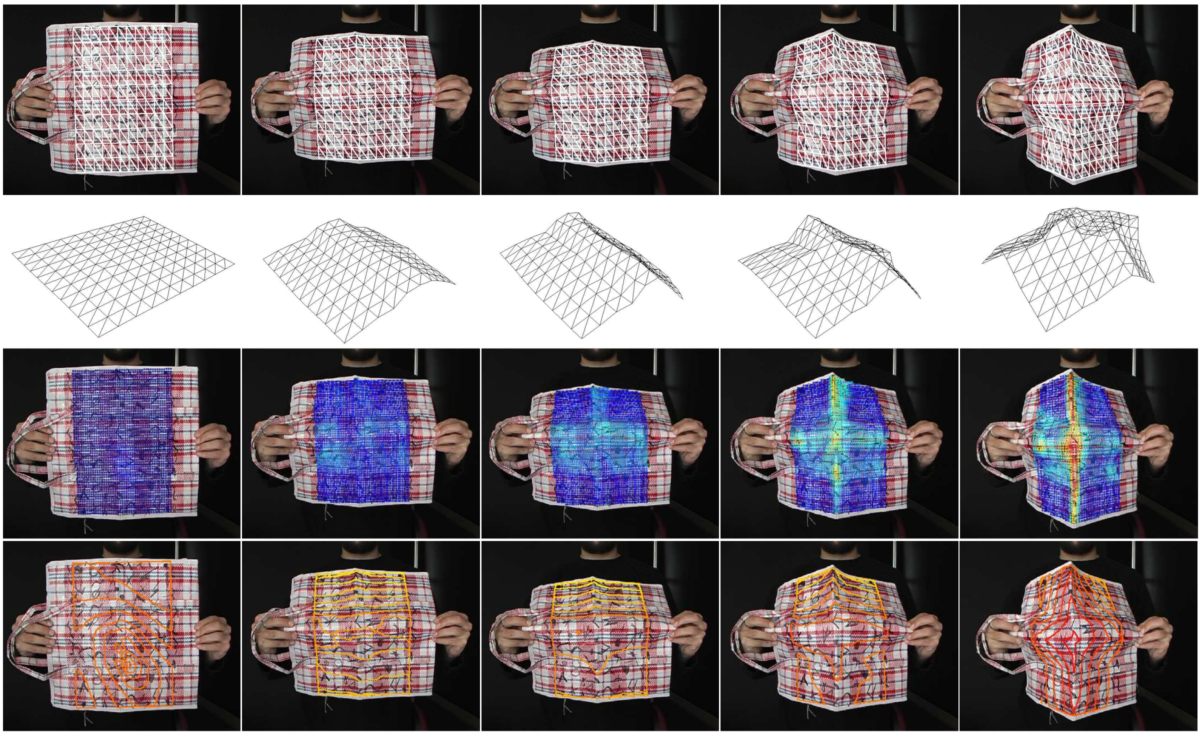


Figure 11. Recovering more complex deformations of the plastic bag. The first two rows depict the reprojection of the mesh into the original images and the mesh seen from a different perspective as before. In the third row, we overlay the mean curvature of the recovered surface on the images. The high curvature areas, shown in red, correspond to the actual creases that can be seen in the top row. In the fourth row, we overlay the level-lines of constant z on the images. We recommend viewing the last two rows in color as they might be difficult to interpret on a greyscale printed copy.

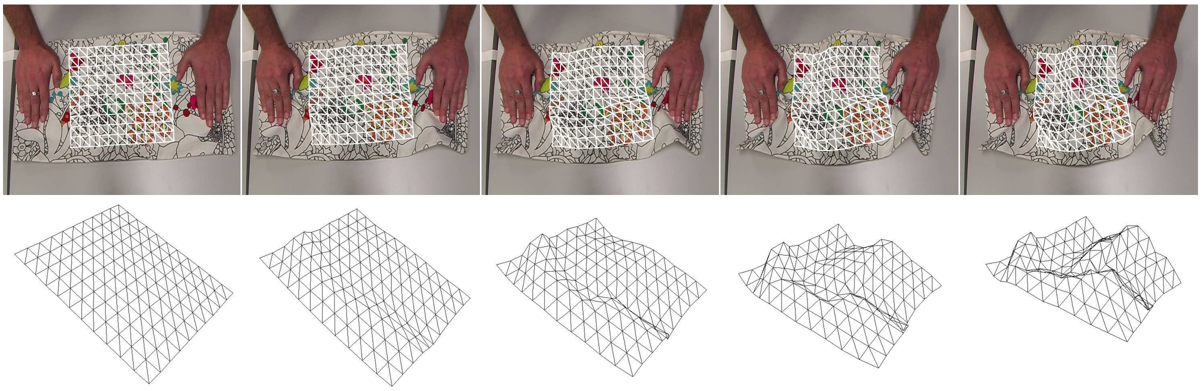


Figure 12. Recovering the deformations of a piece of cloth with several folds.

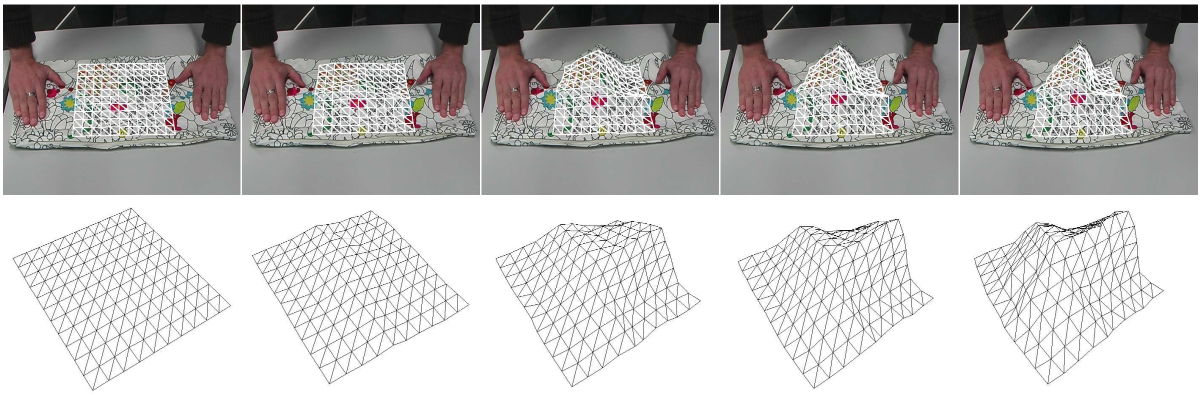


Figure 13. Another example of a different deformation of that same cloth in a 61 frames sequence.