13

Image-based Tracking of Deformable Surfaces

Anna Hilsmann, David C. Schneider and Peter Eisert
Fraunhofer Institute for Telecommunications Heinrich-Hertz-Institute
Humboldt Universität zu Berlin
Germany

1. Introduction

Deformable objects and surfaces are ubiquitous in modern computer vision applications, including medical imaging [Metaxas (1996)], object-based video compression [Ostermann (1994)] and augmented reality [Hilsmann et al. (2010); Pilet et al. (2008)]. In this chapter, we are particularly interested in tracking surfaces whose deformations are difficult to describe, such as drapery of textiles, facial expressions or complex organ motion in medical image sequences. Capturing non-rigid motion of such objects in 2D images is a challenging task because the deformation field complexity is unknown a priori. Therefore, the assumed model has to cope with various types of deformations.

Our framework makes use of image-based deformation models and formulates a robust cost function that can be minimized with common optimization methods, like Gauss-Newton or Levenberg-Marquardt. It is highly modular and can therefore be applied to a broad variety of motion estimation problems, as the motion model can individually be formulated according to the given problem. To avoid error accumulation over the image sequence, we use an analysis-by-synthesis approach, where the error minimization is always carried out between a synthesized reference image built from previous parameter estimates and the actual camera frame. However, in real scenes, tracking is often influenced by varying lighting. If intensity changes due to changes in the lighting conditions in the scene are not considered, the intensity difference between the synthesized reference image and the current frame increases and causes errors in the motion estimation. To avoid these errors, it is essential to also model intensity changes between the images and warp the reference image not only spatially but also photometrically. We will present results from different fields of applications, such as real-time augmented reality (Figure 1), facial expression analysis (Figure 11), and medical applications (Figure 10).

This chapter is structured as follows. Section 2 will briefly review existing literature in the field of deformable tracking and discuss the contribution of our work. Section 3 will give an overview of the mathematical notation used in this chapter. Following, we present different types of deformable motion models in section 4 before we explain our approach to image-based optimization of the model parameters in section 5. Section 6 will present results achieved with our approach for different types of surfaces, such as cloth, faces, and medical images.

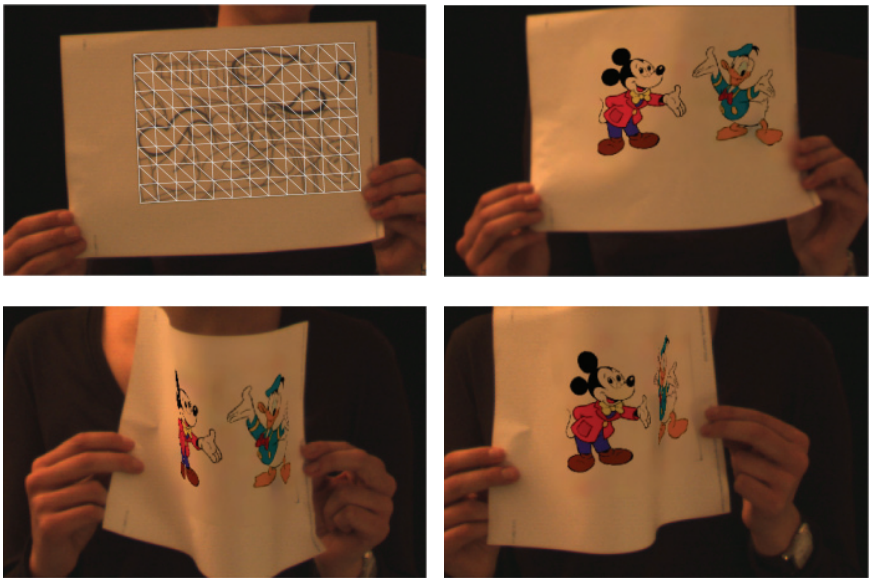


Fig. 1. One application we are targeting: Tracking and retexturing a piece of paper in a video sequence (original frame with overlaid deformation mesh and retextured frames).

2. Related work

Motion between two frames is commonly described by a dense displacement vector field which links the location of each pixel in a given frame to its location in the next frame. Registering a model to an image and tracking of deformable objects is a large field of research and there exists a multitude of methods. Basically, the literature on deformable surface tracking from monocular image sequences distinguishes between marker-based [Scholz & Magnor (2006); White & Forsyth (2006)], feature-based [Pilet et al. (2008)], and image-based [Bartoli & Zisserman (2004); Lim & Yang (2005); Torresani et al. (2001)] methods. As markers are not always available in real scenes, the assumption of such a-priori knowledge limits the applicability of these methods to very special cases. Feature-based methods minimize a distance between a few corresponding feature points, whereas direct methods minimize an error measure that is based on image information of all pixels in the image.

Feature-based methods determine correspondences between distinct feature points and use them to estimate the best transformation between these correspondences. Various features have been described in the literature. Image-based feature points are e.g. local curvature extrema or saddle points, edges or corners [Harris & Stephens (1988); Thirion (1996)] or SIFT [Lowe (2003)] and SURF [Bay et al. (2006)]features. Popular choices of shape-features are the shape context [Belongie et al. (2002)] or statistical moments of the shape [Chen (1993)]. Feature- based methods are mostly used to find rigid or affine transformation. If the set of points is large enough, also more complex transformations can be determined using e.g. radial basis functions [Bookstein (1989)]. The type of the RBF determines overall characteristics of the transformation such as the smoothness or the locality. [Pilet et al. (2008)] proposed a feature-based real-time method for deformable object detection and tracking that uses a wide

baseline matching algorithm and deformable meshes. They estimate the irradiance of the surface separately by warping the reference image to the current frame and estimating the luminance ratio between both images.

Generally, image-based methods yield more accurate results in non-rigid deformation estimation than feature-based methods because they exploit the entire image instead of distinct points. [Bartoli & Zisserman (2004)] presented an optical flow based approach that uses radial basis functions to regularize the flow field. They iteratively insert new center-points for the radial basis functions based on examination of the error image after each iteration. The number of centers grows until the algorithm converges. Recently, [Gay Bellile et al. (2007)] proposed a direct method to estimate deformable motion under self-occlusions by establishing occlusion maps and penalizing a variation in the spatial warp derivative along some direction to prevent mesh-foldings and cope with self-occlusions. [Hilsmann & Eisert (2008)] utilized the idea of mesh-shrinking in an optical flow-based approach by exploiting topological relationships of a mesh-based warp to force the mesh to shrink instead of fold at the occlusion boundary.

While, in general, marker- or feature-based methods are more robust against illumination changes – assuming markers or features are illumination invariant – direct methods usually minimize an error function based on the intensity differences between the aligned images. Therefore, these methods are sensitive to illumination variations. [Gennert & Negahdaripour (1987)] were among the first to propose a direct method robust to illumination variations. They assume that the brightness at time $t + \delta t$ is related to the brightness at time t through a set of parameters that can be estimated from the image sequence. Several other researchers [Bartoli (2008); Haussecker & Fleet (2001); Nastar et al. (1996); Silveira & Malis (2007)] have exploited their ideas to make tracking more robust against lighting changes. [Bartoli (2008)] proposed a dual inverse compositional algorithm to estimate a homography and an affine photometric registration. The registration results are improved by the photometric registration but only global changes are modeled and thus specular reflections or local shadows are not taken into consideration. [Silveira & Malis (2007)] model local illumination changes to make a homography estimation more robust against generic illumination changes. [Pizarro & Bartoli (2007)] proposed to transform images into a 1D shadow invariant space to achieve direct image registration in the presence of even sharp shadows. [Hilsmann & Eisert (2009)] used an extended optical flow constraint and mesh-based models not only as a correction factor for spatial registration but also to actually retrieve local photometric parameters for convincing retexturing purposes in augmented reality applications.

Besides the optimization scheme, state-of-the-art methods differ in the type of the assumed motion model. Tracking arbitrary deformable surfaces without any knowledge about the type of deformation in monocular video sequences is an ill-posed problem as the deformation yields too many ambiguities. Therefore, if no 3D information is required, one approach is to track deformations in the image plane. Some researchers use 2-dimensional mesh-based models [Pilet et al. (2008), Gay Bellile et al. (2007), Hilsmann & Eisert (2009)], others radial basis functions [Bartoli & Zisserman (2004), Bookstein (1989)] to model deformations in 2D. If the type of deformation or a 3D shape model is known a priori, also more specific deformation models can be used in the 3D space. Often, deformation is modeled by weighted superposition of basis functions like superquadrics [Terzopoulos & Metaxas (1991)] or PCA-based models [Salzmann et al. (2007)]. An important application for deformable surface analysis is face tracking [Metaxas (1996), Eisert & Girod (1998)], where specific face models are used to constrain the parameters to model facial deformations.

In this chapter we will present a general framework to deformable surface tracking. In our formulation we separate the motion model from the optimization framework and the regularization term. This makes the framework highly modular and thus adaptive to the given problem.

3. Notation

In general, we denote scalar valued variables with lower case roman letters, e.g. x, vector valued variables with bold lower case letters, e.g. $\mathbf{x}$, and matrices with bold upper case letters, e.g. $\mathbf{X}$. We denote a function f of a pixel position $\mathbf{x}$ parameterized by the parameter vector $\boldsymbol{\theta}$ by $f(\mathbf{x};\boldsymbol{\theta})$. The following list briefly introduces the mathematical notation used in this chapter.

$\mathbf{A}$	Adjacency matrix of a mesh	β	Barycentric coordinates
$\boldsymbol{\mathcal{D}}$	Displacement field	$\delta\boldsymbol{\theta}$	Parameter update
η	Bilinear coordinates	$\mathcal{E}$	Cost function
$\mathcal{E}_D$	Data term of the cost function	$\mathcal{E}_S$	Smoothness term of the cost function
f_x, f_y	scaled focal length	$\mathbf{g}_f$	$1 \times n$ gradient vector
$\mathbf{H}_f$	$n \times n$ Hessian matrix	$\mathcal{I}(\mathbf{x})$	Image intensity at location $\mathbf{x}$
$\mathbf{J_f}$	$m \times n$ Jacobian matrix	K	Number of vertices
$\mathbf{L}$	Laplacian matrix	$\mathbf{l}$	Direction of light source
$\mathbf{n}$	Surface normal	N	Number of parameters
$\mathcal{N}_k$	Neighborhood of a vertex $\mathbf{v}_k$	ψ_g	Geometric warp function
ψ_p	Photometric warp function	ψ_d^3	Warp function for 3D deformation
$\mathbf{p}$	3D object point	$\boldsymbol{\theta}$	Parameter vector
$\hat{\boldsymbol{\theta}}$	Estimated parameter vector	$\boldsymbol{\theta}_g$	Geometric parameter vector
$\boldsymbol{\theta}_p$	Photometric parameter vector	$\boldsymbol{\theta}_d$	Parameters for 3D deformation
$\mathcal{R}$	Region	$\mathbf{R}$	rotation matrix
$\mathbf{t}$	3d translation vector	$\mathbf{V}$	Diagonal matrix of vertex valences
$\mathbf{v}$	vertex	$\mathbf{x}$	Pixel coordinate

4. Deformable models

In our framework, the spatial deformation and motion of a deformable surface in an image are described by a geometric warp function

$$\psi_g(\mathbf{x};\boldsymbol{\theta}) = \mathbf{x} + \boldsymbol{\mathcal{D}}(\mathbf{x};\boldsymbol{\theta}) \tag{1}$$

of the pixel coordinates. $\boldsymbol{\mathcal{D}}(\mathbf{x};\boldsymbol{\theta})$ is a dense 2-dimensional pixel displacement field defined at each image pixel $\mathbf{x}$ and parameterized by a parameter vector $\boldsymbol{\theta}$. Usually, two successive frames in a natural video sequence do not only differ geometrically, but also the intensity of a scene point can vary due to changes in the scene lighting, shading properties etc. For deformable surfaces, the intensity of a scene point varies when the surface normals change. If not handled correctly, varying lighting influences the geometric tracking result, especially for tracking approaches that are based on a brightness constancy assumptions (see section 5.2). Hence, we model intensity changes in an additional photometric warp function

$$\psi_p(\mathcal{I}(\mathbf{x});\boldsymbol{\theta}) = \psi_p(\mathbf{x};\boldsymbol{\theta}) \cdot \mathcal{I}(\mathbf{x}) \tag{2}$$

which is applied multiplicatively to the image intensities. Hence, the parameter vector $\boldsymbol{\theta}$ consists of a geometric part $\boldsymbol{\theta}_g$ and a photometric part $\boldsymbol{\theta}_p$ such that the resulting parameter vector in our framework is given by

$$\boldsymbol{\theta} = \left[\boldsymbol{\theta}_g^T \ \boldsymbol{\theta}_p^T\right]^T .$$

Based on the given tracking problem (type of motion, required accuracy of the deformation field, real-time constraints etc.), different types of parameterization can be chosen. In the following, we will present different motion models used for different types of application.

4.1 2D models

The simplest 2-dimensional model is a dense displacement field, i.e. one displacement vector for each pixel. This *dense model* holds two parameters per pixel. To reduce the number of parameters for e.g. real-time applications or to introduce prior knowledge on the deformation field, we can also model the warp using a 2-dimensional *mesh-based model* with K vertices $\mathbf{v}_k$. Each vertex is associated with two parameters, i.e. its displacements in x- and y-direction $\mathbf{d}_k = [d_{kx}\ d_{ky}]^T$. The warps $\boldsymbol{\psi}_g(\mathbf{x};\boldsymbol{\theta})$ and $\psi_p(\mathbf{x};\boldsymbol{\theta})$ can then be parameterized by arbitrary basis functions defining the deformation field. Different parameterizations will be presented below. In each case, the geometric parameter vector is given in the following form by concatenating the x- and y-coordinates of the displacements:

$$\boldsymbol{\theta}_g = \left[d_{1x}\ \ldots\ d_{Kx}\ d_{1y}\ \ldots\ d_{Ky}\right]^T . \tag{3}$$

The photometric warp is parameterized similarly to the geometric warp by K photometric parameters

$$\boldsymbol{\theta}_p = \left[\rho_1\ \ldots\ \rho_K\right]^T . \tag{4}$$

Having defined the parameter vector, we introduce a matrix notation of the warps:

$$\begin{aligned} \boldsymbol{\psi}_g(\mathbf{x}_i;\boldsymbol{\theta}) &= \mathbf{x}_i + \mathbf{M}_g^{\mathbf{x}_i} \cdot \boldsymbol{\theta} \\ \psi_p\,(\mathbf{x}_i;\boldsymbol{\theta}) &= \mathbf{m}_p^{\mathbf{x}_i} \cdot \boldsymbol{\theta} \end{aligned} \tag{5}$$

where $\mathbf{M}_g^{\mathbf{x}_i}$ and $\mathbf{m}_p^{\mathbf{x}_i}$ are $2 \times N$ and $1 \times N$ matrices defining the parameterization. The superscript $^{\mathbf{x}_i}$ denotes that these matrices differ for each pixel. They depend on the type of parameterization as well as pixel position and will be defined below. We can now easily determine the warp Jacobians, which are required for optimization:

$$\begin{aligned} \mathbf{J}_{\boldsymbol{\psi}_g}(\mathbf{x}_i;\boldsymbol{\theta}) &= \mathbf{M}_g^{\mathbf{x}_i} \\ \mathbf{J}_{\psi_p}(\mathbf{x}_i;\boldsymbol{\theta}) &= \mathbf{m}_p^{\mathbf{x}_i} \end{aligned} \tag{6}$$

4.1.1 Dense model

The dense model holds one displacement vector per pixel, such that the number of parameters is $N = 2p$ for a region of interest with p pixels. The matrices $\mathbf{M}_g^{\mathbf{x}_i}$ and $\mathbf{m}_p^{\mathbf{x}_i}$ are zero matrices

with only one entry in the i^{th}, the$(i+p)^{th}$ and the $(i+2p)^{th}$ column:

$$\begin{aligned}\mathbf{M}_g^{\mathbf{x}_i} &= \underbrace{\begin{bmatrix}0... & 1... & 0... \\ 0... & 0... & 0...\end{bmatrix}}_{(2\times K)} \underbrace{\begin{bmatrix}0... & 0... & 0... \\ 0... & 1... & 0...\end{bmatrix}}_{(2\times K)} \underbrace{\begin{bmatrix}0... & 0... & 0... \\ 0... & 0... & 0...\end{bmatrix}}_{(2\times K)} \\ \mathbf{m}_p^{\mathbf{x}_i} &= \underbrace{\begin{bmatrix}0... & 0... & 0...\end{bmatrix}}_{(1\times K)} \underbrace{\begin{bmatrix}0... & 0... & 0...\end{bmatrix}}_{(1\times K)} \underbrace{\begin{bmatrix}0... & 1... & 0...\end{bmatrix}}_{(1\times K)}\end{aligned} \tag{7}$$

This model is equivalent to the classic optical flow. Due to the aperture problem, the normal equations that arise from the matrices $\mathbf{M}_g^{\mathbf{x}_i}$ during optimization are rank deficient and regularization is a necessity. The Laplacian smoothness term addressed in section 4.3 is one possible choice for regularization.

4.1.2 Affine mesh-based model

In case of a mesh-based model, the number of parameters is determined by the number of vertices in the mesh, such that $N = 2K$ where K is the number of vertices. If a pixel $\mathbf{x}_i$ is surrounded by a triangle consisting of the three mesh vertices $\mathbf{v}_a, \mathbf{v}_b, \mathbf{v}_c$ with indices a, b, c and $\beta_a, \beta_b, \beta_c$ are the three corresponding Barycentric coordinates, it can be represented by the weighted sum of its enclosing vertices:

$$\mathbf{x}_i = \sum_{j\in\{a,b,c\}} \beta_j \mathbf{v}_j \qquad\qquad \beta_a + \beta_b + \beta_c = 1,\ 0 \leq \beta_{a,b,c} \leq 1 \tag{8}$$

A warp with piecewise affine interpolation between the respective three surrounding vertex positions keeps the Barycentric coordinates constant, such that the warps $\psi_g(\mathbf{x};\boldsymbol{\theta})$ and $\psi_p(\mathbf{x};\boldsymbol{\theta})$ can then be parameterized similarly by

$$\begin{aligned}\psi_g(\mathbf{x}_i;\boldsymbol{\theta}) &= \mathbf{x}_i + \boldsymbol{\mathcal{D}}(\mathbf{x}_i;\boldsymbol{\theta}) = \mathbf{x}_i + \sum_{j\in\{a,b,c\}} \beta_j \mathbf{d}_j \\ \psi_p(\mathbf{x}_i;\boldsymbol{\theta}) &= \sum_{j\in\{a,b,c\}} \beta_j \rho_j\end{aligned} \tag{9}$$

This can be formulated in matrix notation as in equation (5) with the following matrices

$$\begin{aligned}\mathbf{M}_g^{\mathbf{x}_i} &= \underbrace{\begin{bmatrix}\beta_a... & \beta_b... & \beta_c... \\ 0... & 0... & 0...\end{bmatrix}}_{(2\times K)} \underbrace{\begin{bmatrix}0... & 0... & 0... \\ \beta_a... & \beta_b... & \beta_c...\end{bmatrix}}_{(2\times K)} \underbrace{\begin{bmatrix}0... & 0... & 0... \\ 0... & 0... & 0...\end{bmatrix}}_{(2\times K)} \\ \mathbf{m}_p^{\mathbf{x}_i} &= \underbrace{\begin{bmatrix}0... & 0... & 0...\end{bmatrix}}_{(1\times K)} \underbrace{\begin{bmatrix}0... & 0... & 0...\end{bmatrix}}_{(1\times K)} \underbrace{\begin{bmatrix}\beta_a... & \beta_b... & \beta_c...\end{bmatrix}}_{(1\times K)}\end{aligned} \tag{10}$$

4.1.3 Bilinear mesh-based model

Similar to the affine parameterization, we can use a bilinear parameterization between the respective four surrounding vertex positions

$$\begin{aligned} \psi_g(\mathbf{x}_i;\boldsymbol{\theta}) &= \mathbf{x}_i + \mathcal{D}(\mathbf{x}_i;\boldsymbol{\theta}) = \mathbf{x}_i + \sum_{j\in\{a,b,c,d\}} \eta_j \mathbf{d}_j \\ \psi_p(\mathbf{x}_i;\boldsymbol{\theta}) &= \sum_{j\in\{a,b,c,d\}} \eta_j \rho_j \\ \eta_a = (1-s)(1-t),\ \eta_b &= s(1-t),\ \eta_c = t(1-s),\ \eta_d = st \end{aligned} \tag{11}$$

$\mathbf{M}_g^{\mathbf{x}_i}$ and $\mathbf{m}_p^{\mathbf{x}_i}$ are matrices composed similarly as for the affine parameterization above except that each row now has four entries $(\eta_a, \eta_b, \eta_c, \eta_d)$ in the corresponding columns.

4.2 3D models

Rather than modelling the motion directly in the 2D image plane, the displacement field (1) can be regarded as the projection of an object's 3D motion and deformation into the 2D domain. Similar to the 2D case, we can specify the deformation of a 3D object, given by its 3D surface points $\mathbf{p}_i$, as a function of deformation parameters $\boldsymbol{\theta}_d$

$$\psi_d^3(\mathbf{p};\boldsymbol{\theta}_d) = \mathbf{p} + \mathcal{D}^3(\mathbf{p};\boldsymbol{\theta}_d). \tag{12}$$

For linear deformation fields, the geometrical warp can then be expressed by a deformation matrix $\mathbf{D}_d^{\mathbf{p}_i}$

$$\psi_d^3(\mathbf{p}_i;\boldsymbol{\theta}_d) = \mathbf{p}_i + \mathbf{D}_d^{\mathbf{p}_i} \cdot \boldsymbol{\theta}_d. \tag{13}$$

This deformation matrix can contain any basis functions that define the deviations of object points from a neutral shape $\mathbf{p}_i$. The amount of deformation performed in the local object coordinate system is controlled by the deformation parameter vector $\boldsymbol{\theta}_d$. Similarly, 3D deformation can be modelled by PCA analysis, with $\mathbf{p}_i$ being a mean shape and $\mathbf{D}_d^{\mathbf{p}_i}$ holding the Eigenvectors of the covariance matrix formed by several sample shapes.

A rigid body transform, specified by rotation matrix $\mathbf{R}_0$ and translation vector $\mathbf{t}_0$, can be used to position and orient the 3D object in a world coordinate system. In addition to deformation, the object is allowed to move which can be described by an update of rotation $\mathbf{R}$ with Euler angles R_x, R_y, R_z and translation $\mathbf{t} = [t_x, t_y, t_z]^T$. The entire geometrical warp in the world coordinate system is then given as

$$\psi_g^3(\mathbf{p}_i;\boldsymbol{\theta}_d) = \mathbf{R}\cdot\mathbf{R}_0(\mathbf{p}_i + \mathbf{D}_d^{\mathbf{p}_i}\cdot\boldsymbol{\theta}_d) + \mathbf{t} + \mathbf{t}_0. \tag{14}$$

For small changes of the parameter vector $\boldsymbol{\theta}_d$ this can be approximated linearly by approximating small rotations by a deformation along the object points' tangents, resulting in

$$\psi_g^3(\mathbf{p}_i;\boldsymbol{\theta}_g) \approx \mathbf{p}_i + \mathbf{D}_g^{\mathbf{p}_i}\cdot\boldsymbol{\theta}_g \tag{15}$$

with the extended parameter vector $\boldsymbol{\theta}_g$.

$$\boldsymbol{\theta}_g = [\boldsymbol{\theta}_d^T, R_x, R_y, R_z, t_x, t_y, t_z]^T \tag{16}$$

holding both deformation and pose parameters. The new matrix $\mathbf{D}_g^{\mathbf{p}_i}$ now covers information from both deformation and rigid body transform

$$\mathbf{D}_g^{\mathbf{p}_i} = \left[\mathbf{R}_0 \cdot \mathbf{D}_d^{\mathbf{p}_i}, \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \times (\mathbf{R}_0 \mathbf{p}_i), \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \times (\mathbf{R}_0 \mathbf{p}_i), \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \times (\mathbf{R}_0 \mathbf{p}_i), \mathbf{I}_3 \right]. \tag{17}$$

The geometrical warp in 3D space given by (14) or (15) can be related with the 2D geometric warp $\boldsymbol{\psi}_g$ via the camera projection. For perspective projection

$$\begin{aligned} x_i &= -f_x \cdot \frac{p_{x,i}}{p_{z,i}} \\ y_i &= -f_y \cdot \frac{p_{y,i}}{p_{z,i}} \end{aligned} \tag{18}$$

with scaled focal lengths f_x and f_y as camera parameters, the linearized displacement field is given by [Eisert & Girod (1998)]

$$\boldsymbol{\psi}_g(\mathbf{x}_i, \boldsymbol{\theta}) \approx \mathbf{x}_i + \underbrace{\begin{bmatrix} -\frac{1}{z(\mathbf{x}_i)}(f_x \mathbf{D}_{g,x}^{\mathbf{p}_i} + x_i \mathbf{D}_{g,z}^{\mathbf{p}_i}) \\ -\frac{1}{z(\mathbf{x}_i)}(f_y \mathbf{D}_{g,y}^{\mathbf{p}_i} + y_i \mathbf{D}_{g,z}^{\mathbf{p}_i}) \end{bmatrix}}_{J_{\boldsymbol{\psi}_g}} \cdot \boldsymbol{\theta}_g \tag{19}$$

with $\mathbf{D}_{g,x,y,z}^{\mathbf{p}_i}$ being the $x-$, $y-$, and $z-$ component of the deformation matrix $\mathbf{D}_g^{\mathbf{p}_i}$ and $z(\mathbf{x}_i)$ the distance of the object point to the camera at pixel $\mathbf{x}_i$.
The geometric warp in the 3D case is attached in the local object coordinate system allowing a semantically meaningful modelling of surface deformation. Similarly, the photometric scaling can be described by compact illumination and reflection models, since geometry and surface normal information can be derived. For a simple shading model consisting of ambient and diffuse light with Lambertian reflection, the photometric warp ψ_p in (2) is given as

$$\psi_p(\mathbf{x}_i; \boldsymbol{\theta}) = c_{amb} + c_{diff} \max\{\mathbf{n}(\mathbf{x}_i) \cdot \mathbf{l}(\mathbf{x}_i), 0\}. \tag{20}$$

Here, $\mathbf{n}$ is the normalized surface normal of the object at a certain pixel position, while $\mathbf{l}$ is a normalized direction vector pointing from the surface point to the light source. The maximum function ensures that surface points facing away from the light source are not illuminated. For analysis of shading parameters, the setup of equations can be limited to surface points being known as illuminated or in shadow, thus making the non-linear maximum function unnecessary. The resulting photometric parameters are then given as

$$\boldsymbol{\theta}_p = [c_{amb}, c_{diff} n_x l_x, c_{diff} n_y l_y, c_{diff} n_z l_z]^T \tag{21}$$

and the individual components of $c_{diff} n_x l_x$ can be derived exploiting that both $\mathbf{n}$ and $\mathbf{l}$ have unit length. In the same way, higher order reflection models can be considered or even PCA spaces modelling pre-computed variations of a lightmap attached to the object surface as described in [Eisert (2003)].

4.3 Model smoothness

It is a reasonable a priori assumption for tracking deformable objects like cloth or tissue that the shape and its motion and deformation is smooth and continuous rather than rough and erratic. This assumption can be expressed by a *smoothness term*, i.e. a function that will associate a cost with undesired, non-smooth model states. Including this function in the optimization later on will lead to a preference of smooth results over rough ones. Algebraically, a smoothness term can also be interpreted as a regularization, especially in the case of the dense model where the data term leads to rank deficiency in the normal equations due to the aperture problem.

Often, smoothness is associated with a vanishing second derivative of a function. Hence, to force a model to be smooth, one can penalize the discrete second derivative of the motion parameters by applying a discrete Laplace operator [Wardetzky et al. (2007)]. The Laplace Matrix of a 2-dimensional grid is often defined as

$$\mathbf{L} = \mathbf{V} - \mathbf{A}, \tag{22}$$

where $\mathbf{V}$ is a diagonal matrix of the vertex valences and $\mathbf{A}$ is the adjacency matrix of the grid. Hence, the Laplace Matrix is based on the neighborhood definition in a grid and for a mesh with K vertices it is a $K \times K$ matrix with one row and one column per vertex. $\mathbf{L}_{ij} = -1$ if vertices $\mathbf{v}_i$ and $\mathbf{v}_j$ are connected and $\mathbf{L}_{ii} = |\mathcal{N}_i|$, where the first subscript denotes the row and the second the column, respectively. $|\mathcal{N}_i|$ denotes the number of vertices in the neighborhood $\mathcal{N}_i$ of vertex $\mathbf{v}_i$. There exist several versions of a scaled Laplace matrix with entries $\mathbf{L}_{ij} = w_{ij}$ if vertices $\mathbf{v}_k$ and $\mathbf{v}_l$ are connected and $\mathbf{L}_{ii} = -1$ with different weighting schemes and different definitions of the neighborhood which influence the *smoothing behavior* of the Laplacian [Taubin (1995)]. The simplest one is a uniform scaling with

$$w_{ij} = \frac{1}{|\mathcal{N}_i|} \tag{23}$$

where $|\mathcal{N}_i|$ denotes the number of vertices in the neighborhood $\mathcal{N}_i$. In order to give closer neighbors a higher influence on vertex $\mathbf{v}_i$ than neighbors with a larger distance, we can also weight each neighbor according to its distance to vertex $\mathbf{v}_i$. In this case the weight is

$$w_{ij} = \frac{1/D_{ij}^{\text{Euc}}}{\sum_{n \in \mathcal{N}_i} 1/D_{in}^{\text{Euc}}} \tag{24}$$

where D_{ij}^{Euc} denotes the Euclidian distance between the two vertex positions $\mathbf{v}_i$ and $\mathbf{v}_j$.

The Laplacian can be applied to any parameter $\boldsymbol{\theta}$ associated with the mesh vertices. The distribution of these parameters over the mesh is perfectly smooth in the aforementioned sense if

$$\nabla\boldsymbol{\theta} = \mathbf{L} \cdot \boldsymbol{\theta} = \mathbf{0}$$

where $\boldsymbol{\theta}$ denotes the set of parameters $\theta_1 ... \theta_K$.

4.3.1 Laplacians and mesh boundaries

The Laplacian matrix explained above does not treat the mesh borders differently from the vertices inside the mesh. For example, the product of the Laplacian and the mesh

vertex coordinates of an undeformed 2-dimensional mesh is not zero at the borders, $\|\mathbf{L} \cdot \mathbf{v}_x\| > 0, \|\mathbf{L} \cdot \mathbf{v}_y\| > 0$. This is due to the asymmetric vertex neighborhood at the mesh boundaries.

One solution is to build *Directional Laplacian matrices* $\mathbf{L}^d$ of the mesh which penalize the second derivatives in different directions, i.e. vertical and horizontal, separately. For each direction of vertex connections one Laplacian is built, with one row for each vertex that has two neighbors in the same direction. For example $\mathbf{L}^d_{ii} = 2$ and $\mathbf{L}^d_{ij} = -1$ if vertices $\mathbf{v}_k$ and $\mathbf{v}_l$ are connected in the direction d. Each row in $\mathbf{L}^d$ corresponds to one vertical vertex triple. The complete Laplacian is build by concatenating the *Directional Laplacian matrices* $\mathbf{L} = \left[\mathbf{L}^{d1} \ ... \ \mathbf{L}^{dn}\right]^T$. This type of Laplacians has several advantages. First, the product of this Laplacian with the mesh vertices of an undeformed mesh is zero as only symmetric neighborhoods, i.e. neighbors which have a *directional counterpart*, are taken into account. Second, by building separate Laplacians for each direction, we provide more constraints at the border vertices than one Laplacian for the full neighborhood.

5. Image-based optimization of the model parameters

Independent of the individual parameterization, the parameters $\boldsymbol{\theta}$ of the underlying model are estimated by minimizing a cost function:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} (\mathcal{E}_D(\boldsymbol{\theta}) + \lambda^2 \mathcal{E}_S(\boldsymbol{\theta})) \tag{25}$$

where $\mathcal{E}_D(\boldsymbol{\theta})$ is the *data term* and $\mathcal{E}_S(\boldsymbol{\theta})$ represents prior assumptions and is often called the *smoothness term*. We formulate both terms in a robust non-linear least-squares sense. This allows to minimize the cost function with common optimization algorithms such as Gauss-Newton (GN) or Levenberg-Marquardt (LM), i.e. by iteratively solving for a parameter update $\delta\hat{\boldsymbol{\theta}}$ and updating the parameter vector $\hat{\boldsymbol{\theta}} \leftarrow \hat{\boldsymbol{\theta}} + \delta\hat{\boldsymbol{\theta}}$. The optimization is performed hierarchically on an image pyramid where each level yields a more accurate parameter estimate. This hierarchical framework has several advantages. It speeds up the iteration time on lower and coarser levels and also allows us to cope with large displacements.
This section will concentrate on the formulation of the data and smoothness terms for different scenarios. For the data term, we exploit a relaxed version of a brightness constancy assumption accounting not only for geometric but also intensity differences between two images. The smoothness term in our framework makes use of a discrete formulation of the Laplacian defined on a 2-dimensional grid as formulated in section 4.3. We will describe the characteristics of different formulations of the Laplacian. Furthermore, we will cover numerical aspects for the estimation of the dense model.

5.1 Optimization framework

Both data term and smoothness term of equation (25) can be expressed in the following form:

$$\mathcal{E}(\boldsymbol{\theta}) = \sum_{i=1}^{m} \rho(r_i(\boldsymbol{\theta})) \tag{26}$$

$\rho(r_i(\boldsymbol{\theta}))$ is a *norm-like function* [McCullagh & Nelder (1998); Wedderburn (1974)], i.e. a symmetric, positive-definite function with a unique minimum at zero, which is chosen to be

less increasing than square. The r_i are the residuals of the particular term. The minimization is computed by an iteratively reweighted least-squares (IRLS) method which is known to be a Gauss-Newton type method for minimizing a sum of norm-like functions of the residuals [Huber (1981)]. In this scheme, the gradient and the (approximated) Hessian of the error function are given by

$$\begin{aligned} \mathbf{g}_{\mathcal{E}} &= \mathbf{r}^T \mathbf{W} \mathbf{J}_\mathbf{r} \\ \mathbf{H}_{\mathcal{E}} &\approx \mathbf{J}_\mathbf{r}{}^T \mathbf{W} \mathbf{J}_\mathbf{r} \end{aligned} \tag{27}$$

where $\mathbf{r} = \mathbf{r}(\boldsymbol{\theta})$, $\mathbf{J}_\mathbf{r} = \mathbf{J}_\mathbf{r}(\boldsymbol{\theta})$ and $\mathbf{W} = diag(w(r_i(\boldsymbol{\theta})))$ is a weight matrix computed in each iteration with

$$w(r_i) = \frac{1}{r_i} \frac{\partial \rho(r_i)}{\partial r_i} \tag{28}$$

An efficient and well known norm-like function is the Huber estimator [Huber (1981)]

$$\rho(r_i) = \begin{cases} \frac{1}{2} r_i^2 & if\ |r_i| \leq \upsilon \\ \upsilon_{\mathrm{H}} |r_i| - \frac{1}{2} \upsilon & \text{otherwise} \end{cases} \tag{29}$$

which is a parabola in the vicinity of zero, and increases linearly at a given level $|r_i| > \upsilon$. The weight function for the Huber kernel is given by

$$w(r_i) = \begin{cases} 1 & if\ |r_i| \leq \upsilon_{\mathrm{H}} \\ \frac{\upsilon_{\mathrm{H}}}{|r_i|} & \text{otherwise} \end{cases} \tag{30}$$

For $\rho(r_i) = \frac{1}{2} r_i^2$, the weight matrix is the identity and the method is equivalent to the (non-robust) Gauss Newton algorithm for non-linear least squares problems. In each iteration a parameter update is estimated by solving the *normal equations*

$$\delta \hat{\boldsymbol{\theta}} = - \left(\mathbf{J}_\mathbf{r}{}^T \mathbf{W} \mathbf{J}_\mathbf{r} \right)^{-1} \mathbf{r}^T \mathbf{W} \mathbf{J}_\mathbf{r} \tag{31}$$

The parameter is then updated by $\hat{\boldsymbol{\theta}} \leftarrow \hat{\boldsymbol{\theta}} + \delta \hat{\boldsymbol{\theta}}$.
For deformable surface tracking, we use the Huber kernel as a robust estimator for the data term and the two norm for the smoothness term. The matrices and vectors of our normal equations therefore are

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_\mathbf{r}(\hat{\boldsymbol{\theta}}) \\ \lambda \mathbf{J}_\mathbf{s}(\hat{\boldsymbol{\theta}}) \end{bmatrix}^T \quad \mathbf{b} = \begin{bmatrix} \mathbf{r}(\hat{\boldsymbol{\theta}}) \\ \mathbf{s}(\hat{\boldsymbol{\theta}}) \end{bmatrix}^T \quad \tilde{\mathbf{W}} = \begin{bmatrix} \mathbf{W} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}. \tag{32}$$

where $\mathbf{r}$ and $\mathbf{s}$ are the vectors of the residuals of the data and the smoothness term, respectively, which will be explained in detail in the following sections.

5.2 Data Term

The data term is based on a relaxed brightness constancy assumption. Methods exploiting the brightness constancy constraint assume that an image pixel $\mathbf{x}$ representing an object point does not change its brightness value between two successive frames $\mathcal{I}_{n-1}$ and $\mathcal{I}_n$:

$$\mathcal{I}_{n-1} \left(\psi_g(\mathbf{x}; \boldsymbol{\theta}_n) \right) = \mathcal{I}_n \left(\mathbf{x} \right) \tag{33}$$

However, this assumption is almost never valid for natural scenes. For this reason, we relax the optical flow constraint equation allowing for multiplicative deviations from brightness constancy:

$$\psi_p(\mathbf{x};\boldsymbol{\theta}_n)\cdot\mathcal{I}_{n-1}\left(\boldsymbol{\psi}_g(\mathbf{x};\boldsymbol{\theta}_n)\right)=\mathcal{I}_n\left(\mathbf{x}\right) \tag{34}$$

with a photometric warp $\psi_p(\mathbf{x};\boldsymbol{\theta}_n)$ and a geometric warp $\boldsymbol{\psi}_g(\mathbf{x};\boldsymbol{\theta}_n)$ as given in equation (5). Hence, the data term is formulated via the differences between the original frame $\mathcal{I}_n$ and the spatially and photometrically warped previous frame $\mathcal{I}_{n-1}$:

$$\mathcal{E}_D(\boldsymbol{\theta})=\sum_{\mathbf{x}_i\in\mathcal{R}}\rho(\underbrace{\psi_p(\mathbf{x}_i;\boldsymbol{\theta}_{n-1})\cdot\mathcal{I}_{n-1}\left(\boldsymbol{\psi}_g(\mathbf{x}_i;\boldsymbol{\theta}_{n-1})\right)-\mathcal{I}_n(\mathbf{x}_i)}_{r_i}) \tag{35}$$

where $\rho(r_i)$ is a norm-like function. Both warp functions are parameterized by the parameter vector $\boldsymbol{\theta}$ which comprises a geometric part $\boldsymbol{\theta}_g$ and a photometric part $\boldsymbol{\theta}_p$ as in equation (4). The Jacobian $\mathbf{J_r}$ of the residuals r_i is a $p\times N$ matrix (with p being the number of pixels in the region of interest) with the following i^{th} row:

$$\frac{\partial r_i(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}=\psi_p(\mathbf{x}_i;\boldsymbol{\theta})\cdot\nabla\mathcal{I}_{n-1}(\boldsymbol{\psi}_g(\mathbf{x}_i;\boldsymbol{\theta}))\cdot\mathbf{J}_{\boldsymbol{\psi}_g}(\mathbf{x}_i;\boldsymbol{\theta})+\mathcal{I}_{n-1}(\boldsymbol{\psi}_g(\mathbf{x}_i;\hat{\boldsymbol{\theta}}))\cdot\mathbf{J}_{\psi_p}(\mathbf{x}_i;\boldsymbol{\theta}) \tag{36}$$

with $\nabla\mathcal{I}=\left[\frac{\partial\mathcal{I}}{\partial x}\ \frac{\partial\mathcal{I}}{\partial y}\right]$. The Jacobians of the warps $\mathbf{J}_{\boldsymbol{\psi}_g}(\mathbf{x}_i;\boldsymbol{\theta})$ and $\mathbf{J}_{\boldsymbol{\psi}_g}(\mathbf{x}_i;\boldsymbol{\theta})$ depend on the warp parameterization. In general, it is a $2\times N$ matrix where N is the number of parameters in the parameters vector θ as given in equation (6) or (19).

5.3 Smoothness term

The smoothness term is defined to regularize all parameters, geometric and photometric ones, using some suitably chosen Tikhonov regularizer

$$\mathcal{E}_S(\boldsymbol{\theta})=\|\boldsymbol{\Gamma}\cdot\boldsymbol{\theta}\|^2. \tag{37}$$

In the simplest case (zeroth-order Tikhonov regularization) $\boldsymbol{\Gamma}$ is the identity matrix $\boldsymbol{\Gamma}=\mathbf{I}$, giving preference to solutions with smaller norms. Other possible regularizations include first-order or second-order Tikhonov regularizations, where Γ approximates first- or second-order derivatives of the parameter, favoring *flat* or *smooth* results.

For the 2-dimensional models, the Laplacian matrix for 2-dimensional grids explained in section 4.3 define such an approximation of the second derivative of the mesh. As the parameter vector consists of three parts which should be regularized independently, the Tikhonov matrix is given by

$$\boldsymbol{\Gamma}=\begin{bmatrix}\mathbf{L} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{L} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \lambda_p\mathbf{L}\end{bmatrix} \tag{38}$$

A similar regularization matrix can be formulated for the 3-dimensional models. λ_p weights the smoothing of the geometric parameters (vertex displacements in $x-$ and y-direction) against the smoothing of the photometric scale. This is necessary due to the different scaling of the pixel displacement and the photometric parameter, the former being additive and the latter multiplicative. The Jacobian of the smoothness term is $\mathbf{J_s}(\boldsymbol{\theta})=\boldsymbol{\Gamma}$.

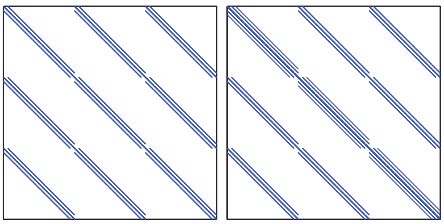


Fig. 2. Example structure of $\mathbf{H_r} \approx \mathbf{J_r}^T\mathbf{J_r}$ and $\mathbf{H} \approx \mathbf{J}^T\mathbf{J}$ for an affine mesh-based model.

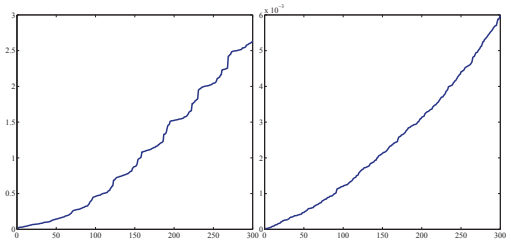


Fig. 3. Eigenvalues of the matrix with and without the smoothness term. Without the smoothness term the matrix is ill-conditioned.

By using a regularizer as given in equation (38), $\mathcal{E}_S(\boldsymbol{\theta})$ penalizes the discrete second derivative of the mesh and regularizes the flow field in addition to the motion model itself, especially in case of insufficient information in the data due to e.g. homogeneous regions with low image gradient. One important advantage of using direct image information instead of features is, that a lack of image information, i.e. image gradients of small magnitude in regions of little texture, automatically lead to a higher local weighting of the smoothness constraint in these regions. Also, the function dominates for vertices detected as outliers when using e.g. the Huber function for robust estimation.

5.4 Numerical issues

For all models presented in section 4 the approximation of the Hessian $\mathbf{H} = \mathbf{J}^T\mathbf{J}$ is a sparse banded matrix (see Figure 2). The sparsity can be exploited to solve the resulting normal equations in each iteration, especially for the dense model where the matrix becomes very large but also very sparse: There are two unknowns per pixel, but there is no coupling of unknowns in the data term. This is in contrast to the affine warp, where all pixels under a certain mesh triangle simultaneously contribute to the six unknowns associated with the triangle's three vertices.

To solve the normal equations for the dense warp in an efficient way, multigrid algorithms can be used. The multigrid scheme exploits the fact that certain iterative solvers, e.g. the computationally inexpensive Gauss-Seidel method, reduce high frequency errors quickly despite of overall slow convergence. Therefore, these algorithms can be used as *smoothers*. After a few smoothing iterations the problem is subsampled to a lower resolution pixel grid and the process is restarted. Only at the bottom stage, the problem, which is now significantly reduced in size, is solved exactly. The exact solution is then propagated upwards to the higher resolution levels. The whole process is called a *V-cycle* in the multigrid literature.

5.5 Analysis by synthesis

Generally, intensity-based differential techniques, which estimate the motion only between two successive frames, often suffer from drift because they accumulate errors indefinitely. This limits their effectiveness when dealing with long video sequences. To avoid error accumulation we make use of an analysis-by-synthesis approach, where the error minimization is always carried out between a synthesized reference image and the actual camera frame. We use the previous parameter sets $\{\hat{\boldsymbol{\theta}}_1, ...\hat{\boldsymbol{\theta}}_{n-1}\}$ to generate a synthetic version of the previous frame $\mathcal{I}_{n-1}$ from a model image $\mathcal{I}_0$. The new parameters $\hat{\boldsymbol{\theta}}_n$ are then estimated from this synthetic previous frame $\hat{\mathcal{I}}_{n-1}$ to the current camera frame $\mathcal{I}_n$ (see Figure 4). This

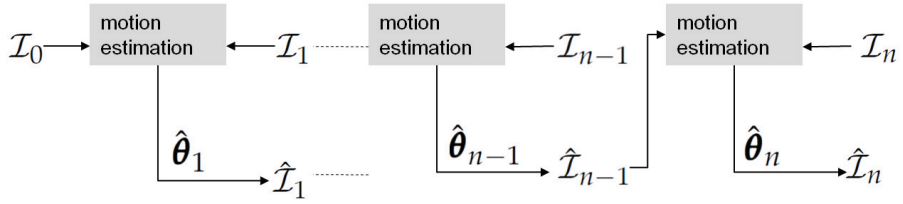


Fig. 4. Analysis by Synthesis framework

way, the model frame serves as reference frame and we assure that no misalignment of the model and the previous frame occurs. Thereby, we allow for recovery from small inaccuracies during parameter estimation.

6. Experimental results

Our approach has been applied to several tracking problems, such as tracking cloth in augmented reality applications, medical image analysis, and facial expression analysis. The different applications will be explained in detail in section 6.3. Section 6.1 and 6.2 focus on experiments on the registration accuracy achieved with our framework and a mesh-based 2-dimensional model as well as on choosing the regularization parameter in the optimization framework.

6.1 Registration accuracy

We evaluated the registration results of real image sequences based on the Root Mean Squared Error (RMSE) between the synthetic image $\hat{\mathcal{I}}_n$ generated from the parameter estimates $\hat{\boldsymbol{\theta}}_n$ and the original current frame $\mathcal{I}_n$ computed over all image pixels in the mesh region $\mathcal{R}$ for several video sequences and compared our approach with the classical optical flow approach. With classical optical flow approach we refer to the original optical flow constraint that does not account for illumination changes, the geometric deformation model and optimization method are equal. Experiments with nine sequences showed that taking illumination parameters into account significantly improves the spatial tracking results. The table in Figure 5 sums up the mean RSME values over the entire sequence for these nine sequences. It shows that taking illumination parameters into account significantly reduces the mean RMSE over the entire sequence by up to 74%. The right image shows the RSME plot of one of the sequences and compares the results achieved with our approach (black solid line) to the classical optical flow method (dashed red line). Additionally, we manually labeled prominent feature points in every 50th frame of two test sequences which serve as ground truth points. We then warped the ground truth points of the reference frame with the geometric deformation parameters of these frames. The mean difference between the estimated positions and the manually labeled ground truth position describes the geometric registration error. This additional evaluation approach is chosen to evaluate geometric registration accuracy separately from photometric registration. We can reduce the mean distance between the estimated and the ground truth position by approximately 40% when taking illumination into account. In all these experiments we used a 2-dimensional mesh-based affine motion model and a regularization parameter $\lambda = 0.5$.

Sequence	classical OF	our approach	%
Picasso	0.0306	0.0204	33.33%
Flower1	0.1364	0.0415	69.57%
Flower2	0.1245	0.0376	69.80%
Art	0.1016	0.0258	74.61%
Flower3	0.1286	0.0385	70.06%
Shirt	0.0630	0.0405	35.71%
Pattern	0.0632	0.0213	66.30%
Cloth1	0.0877	0.0443	49.49%
Cloth2	0.0976	0.0513	47.44%

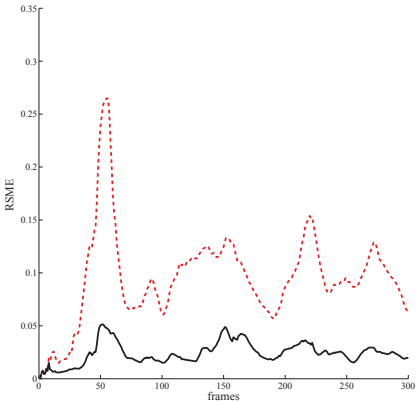


Fig. 5. Left: Comparison of the average RMSE over the entire video sequence with our approach and classical optical flow. Right: Plots of the RMSE between the synthetic frame $\hat{\mathcal{I}}_n$ and the original current frame $\mathcal{I}_n$ with classical optical flow (dashed red) and our method (solid black) for an example sequence.

6.2 Choosing the regularization parameter

Choosing the regularization parameter λ in the optimization framework is not an easy task. The choice of λ describes a trade-off between fitting to the data term –which may be corrupted by noise– and the *smoothness* of the result. Hence, there is no correct λ as the trade-off depends on the considered problem. We can only define a *reasonable* λ, i.e. a λ which represents the best balance between both sides. The influence of the regularization parameter λ can be analyzed using the L-curve [Hansen (1992)], which plots $\mathcal{E}_D(\hat{\boldsymbol{\theta}})$ against $\mathcal{E}_S(\hat{\boldsymbol{\theta}})$ for different values of λ. The L-curve (see Figure 7) is basically made up of two parts which correspond to *oversmoothed* and *undersmoothed* solutions. The more horizontal part corresponds to the solution where the regularization parameter is very large and the solution is dominated by the regularization errors. The more vertical part corresponds to solutions where the regularization parameter is very small and the solution is dominated by the data error. Often, a *reasonable* λ, i.e. a good trade-off between fitting to the data and smoothness term, is considered to be the point of maximal curvature of the L-shaped curve or the point nearest to the origin. Figure 6 shows the influence of different choices for λ. It shows examples of difference images before optimization (left) and for different values of λ. If λ is chosen too small, the result is dominated by the (noise corrupted) data term and if λ is chosen too small, the result is too smooth to be fitted to the data term. Figure 7 shows a typical L-Curve and associated values for the data and smoothness terms for different values for λ.

6.3 Applications

We applied the proposed framework to different fields of applications, presented in the following.

Augmented reality

The proposed method was used for augmented reality applications where a deformable surface, like a piece of paper or cloth, is retextured in a monocular image sequence. The

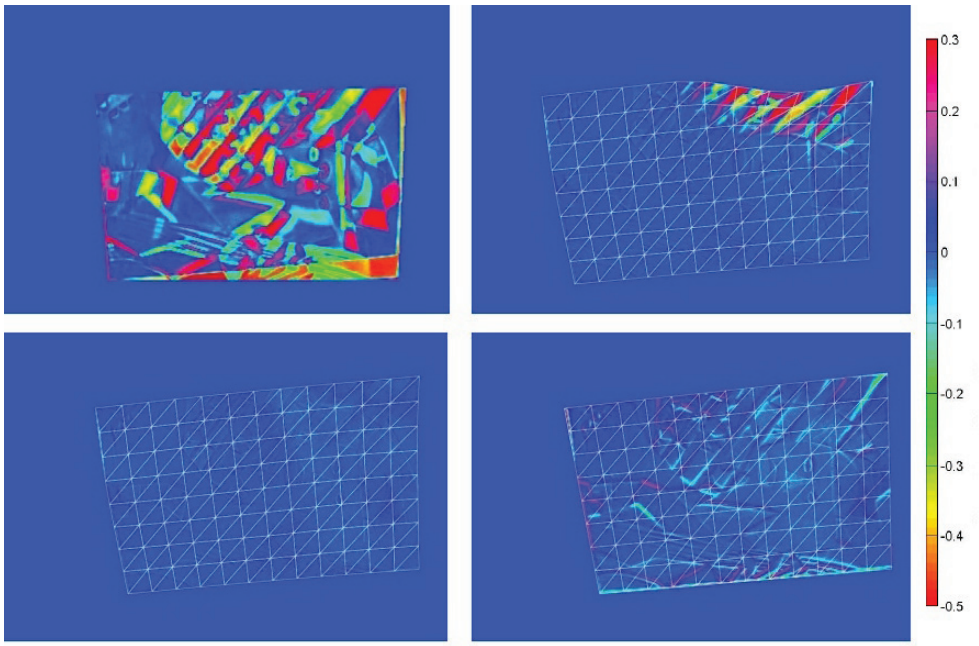
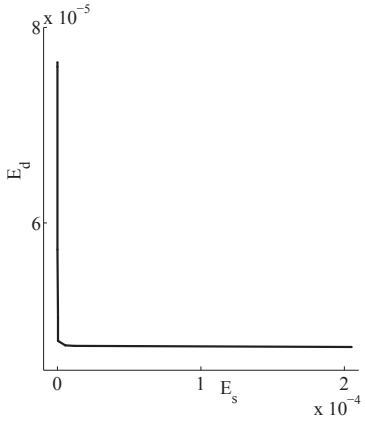


Fig. 6. Difference image between the model frame and an example current frame before motion estimation (left). Difference images after optimization with very small λ (undersmoothed), *reasonable* λ and very large λ (oversmoothed).



λ	$\mathcal{E}_D/10^{-5}$	$\mathcal{E}_S/10^{-5}$
0.01	4.7357	20.535
0.1	4.7449	1.1588
0.2	4.7494	0.5137
1	4.7957	0.0285
5	5.0331	0.0106
10	5.7310	0.0043
100	7.5944	0.0000
1000	7.6385	0.0000

Fig. 7. A typical L-curve and associated values for $\mathcal{E}_D$ and $\mathcal{E}_S$ for different values for λ

intention is to blend a virtual texture into a real video sequence such that its deformation as well as lighting conditions and shading in the final image remain the same (see e.g. Figure 1). We recover geometric and photometric parameters that describe the 2-dimensional

Fig. 8. Tracking and synthesizing different kinds of cloth. The left example shows thick cloth with very smooth deformations while the right example shows cloth that produces small crinkles and creases.

Fig. 9. Tracking and retexturing clothes in an augmented reality application (left: original frame with mesh)

deformation and shading properties of the projected surface in the image plane with the proposed method. The virtual texture is then deformed with the estimated deformation parameters and blended into the original video sequence. A shading map estimated from the photometric parameters is applied to the virtual texture increasing the realistic impression of the augmented video sequence.

Figure 1, 8, and 9 show augmentation results under different lighting conditions using the estimated illumination parameters to establish a shading map. These examples demonstrate how crucial illumination recovery is for convincing texture augmentation of deforming surfaces. The addition of realistic lighting increases the perception of spatial relations between the real and virtual objects. Note, that spatial deformation is purely 2D and the 3-dimensional impression comes from shading. The results from Figure 9 are images from a real-time augmented reality application which we built. In this Virtual Mirror a user is captured with a camera and a rectangular pattern on his shirt is retextured with a user-defined logo or image with correct deformation and illumination. The new logo follows the users movements as if attached to his shirt. The system runs at 25 fps and the images have a resolution of 768×1024 pixels. To this end we use 4 levels of resolution and experiments with synthetic

image sequences with this hierarchical scheme showed that it is able to estimate displacements of up to 25 pixels between two frames with a mean error of 0.2 pixels. On a 2.4 GHz Pentium 4 based system the parameter estimation takes about 40 ms, leading to a frame rate of about 25 fps.

Medical imaging

Our framework is used to stabilize kymographic images. Kymographic imaging is a method for visualizing and analyzing motion. In our use case, it is the vibration of the human vocal folds that is to be visualized. The source material are endoscopic video recordings showing the vocal folds while the patient tries to produce a sound at a certain frequency. Instead of analyzing video images of the whole moving vocal folds, a single line of each frame in the recorded image sequence is extracted. The lines are combined to create a kymogram which is then analysed by the physician. A kymogram is a *time slice* of the vibratory behaviour of the vocal folds, i.e. an X-t-image rather than an X-Y image. Camera movement and motion of the patient as a whole disturb the creation of the kymographic images which relies on the assumption that a certain scanline of the frames displays roughly the same part of the vocal fold throughout the entire endoscopic video sequence. Therefore, kymographic imaging can be greatly improved by compensating camera motion.

We use the methods described in the chapter to compute a deformation field between successive frames in the endoscopic video sequence. The field is computed with the mesh-based affine model. The stiffness, i.e. the weighting of the Laplacian smoothness term, is chosen such that the deformation field adapts to global image motion due to camera movement as well as to the parallax motion induced by different depth layers in the scene. It is set too stiff, however, to compensate for the motion of the vocal folds, which must not be altered by the image stabilization. From the deformation fields, the global motion of a region of interest, which is annotated manually in the first frame, is tracked throughout the sequence. A stabilizing transformation for each frame is found by estimating a rigid transformation that maps the region of interest to its correspondence in the first frame. Figure 10 illustrates the vast increase in kymogram quality achieved by the motion compensation.

Our dense, warp-based approach to image stabilization has several advantages over the classic approach based on feature tracking. Our approach allows us to optimize for a dense motion field using all available image information. This is particularly effective for images of low quality. The endoscopic video we stabilize, for example, suffers from artifacts due to interlacing, image noise, occasional color artifacts and motion blur. We found simple feature detectors such as Harris corners virtually unusable on this material. Furthermore, the optimization approach gives a complete and accurate estimate of nonrigid image motion at a certain scale as the result of a single optimization process. Outliers in the data term are handled during the optimization by the robust error metric (in this case, the Huber function) and there is no need to perform RANSAC or other data selection schemes. Finally, the density of the mesh and the weight of the stiffness term allows us to regulate precisely to which scale of nonrigid motion the algorithm adapts.

Facial Expression Analysis

The human face is a well known example for a deformable object. Head pose can be described by rigid body transforms whereas facial expressions lead to local surface deformations. These deformations, however, are constrained and depend on face muscle and soft tissue properties. Analyses have shown [Ekman & Friesen (1978)], that there are only about 50 groups of muscles in the face, that can be controlled independently when performing facial expressions.

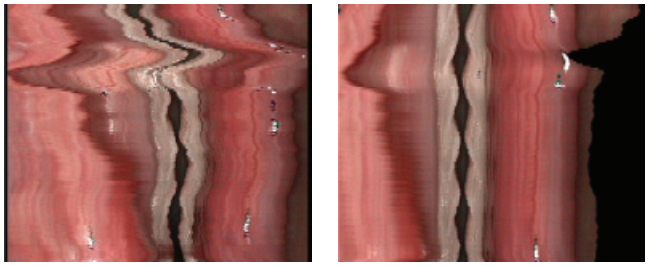


Fig. 10. Kymography: uncompensated and compensated kymographic images.

Therefore, surface deformations as specified in (12) can compactly be described by a limited number of facial expression parameters.
We have applied the framework described in this chapter to the estimation of facial expression parameters [Eisert (2003); Eisert & Girod (1998)]. The geometry of a head is represented by a parameterized generic triangle mesh model, which is individualized by applying surface deformations represented by shape parameters to a standard face geometry. The model is fit to the first frame of a video sequence, which is also projected onto the 3D model as texture. For the description of facial expressions, we have adopted a subset of the MPEG-4 Facial Animation Parameters (FAPs according to [MPG (1999)]), which describe facial expressions by a superposition of simple local surface deformations related to movements of lips, chin, cheeks, etc. In our experiments, we use 21 facial expression parameters $\boldsymbol{\theta}_d$ together with 6 pose parameters to specify surface deformation and orientation according to (14). The mapping from FAPs to arbitrary surface points deformation given by $\mathbf{D}_d$ is modelled using triangular B-splines.
Given the three-dimensional model description, the facial expression parameters are jointly estimated with the global pose parameters in an analysis by synthesis frame work as described in section 5.5. The per-pixel depth information required in (19) as well as the surface normal data are rendered and read back from the graphics card. Smoothness terms penalize FAP differences between the left and right side of the face, favoring symmetric facial expressions. In addition to the geometric deformations, parameters related to illumination are estimated to compensate for lighting changes between the model created for the first frame and the current scene. A simple model with ambient and diffuse colored light is used to describe the photometric changes. Figure 11 gives examples for the facial expression analysis of the proposed framework. From the original frames in the leftmost column, surface deformations are estimated. The deformed model is shown by means of a wireframe and textured representation in the middle columns. The facial animation parameters can also be applied to other face models, implementing expression cloning as illustrated in the rightmost columns of figure 11.

7. Conclusion

We presented a complete framework for tracking of deformable surfaces using image-based optimization methods. We use a relaxed brightness constancy assumption and model both geometrical as well as photometrical changes in an image sequence. In our framework formulation we separated the motion model from the optimization method which makes

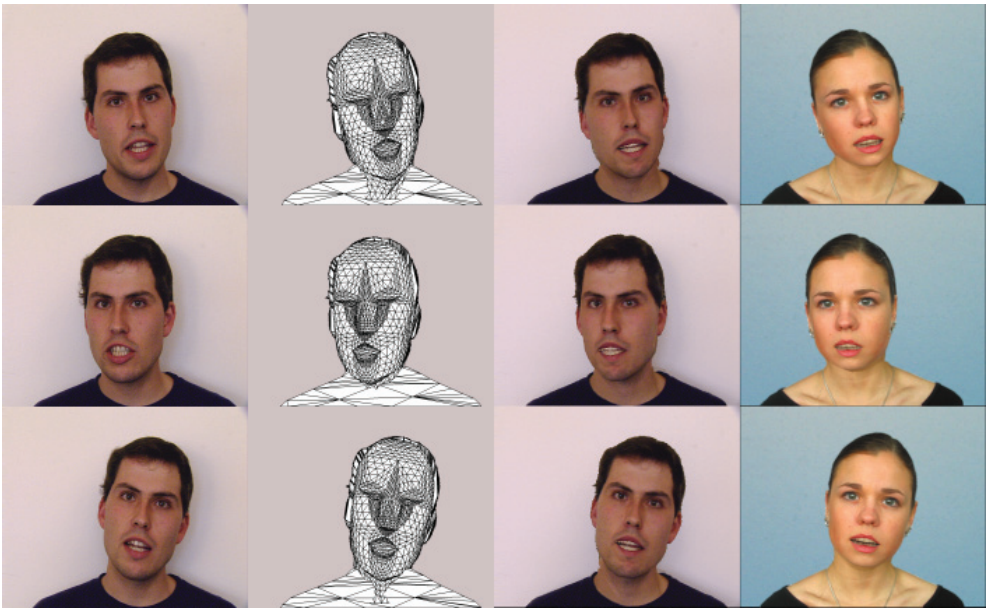


Fig. 11. 3D facial expression analysis. The left colum shows example frames of the original sequence. Facial expression parameters of a deformable face model are estimated. The second column shows the animated mesh, while the third column depicts the textured model. The rightmost column refers to expression cloning where the deformation parameters are mapped onto a different person's model.

it highly adaptive to a broad variety of motion estimation problems, as the motion model can be formulated individuallyfor the given problem. This is shown by a broad variety of applications where we applied the proposed framework.

8. References

Bartoli, A. (2008). Groupwise geometric and photometric direct image registration, *IEEE Trans. on Pattern Analysis and Machine Intelligence* 30(12): 1098–2108.

Bartoli, A. & Zisserman, A. (2004). Direct estimation of non-rigid registrations, *Proc. British Machine Vision Conf. (BMVC 2004)*, London, UK.

Bay, H., Tuytelaars, T. & Gool, L. V. (2006). Surf: Speeded up robust features, *In ECCV*, pp. 404–417.

Belongie, S., Malik, J. & Puzicha, J. (2002). Shape matching and object recognition using shape contexts, *IEEE Trans. Pattern Analysis and Machine Intelligence* 24(4): 509–522.

Bookstein, F. L. (1989). Principal warps: Thin-plate splines and the decomposition of deformations, *IEEE Trans. Pattern Analysis and Machine Intelligence* 11(6): 567–585.

Chen, C. (1993). Improved Moments Invariants for Shape Discrimination, *Pattern Recognition* 26(5): 683–686.

Eisert, P. (2003). MPEG-4 facial animation in video analysis and synthesis, *International Journal of Imaging Systems and Technology* 13(5): 245–256.

Eisert, P. & Girod, B. (1998). Analyzing facial expressions for virtual conferencing, *IEEE Computer Graphics and Applications* 18(5): 70–78.

Ekman, P. & Friesen, W. V. (1978). *Facial Action Coding System*, Consulting Psychologists Press, Inc., Palo Alto.

Gay Bellile, V., Bartoli, A. & Sayd, P. (2007). Direct estimation of non-rigid registrations with image-based self-occlusion reasoning, *Proc. Int. Conf on Computer Vision (ICCV 2007)*, pp. 1–6.

Gennert, M. A. & Negahdaripour, S. (1987). Relaxing the brightness constancy assumption in computing optical flow, *Technical report*, Cambridge, MA, USA.

Hansen, P. (1992). The use of the l-curve in the regularization of discrete ill-posed problems, *SIAM Journ. Sci. Comp.* 34: 561–580.

Harris, C. & Stephens, M. (1988). A combined corner and edge detection, *Proc. 4th Alvey Vision Conference*, pp. 147–151.

Haussecker, H. & Fleet, D. (2001). Computing optical flow with physical models of brightness variation, Vol. 23, Washington, DC, USA, pp. 661–673.

Hilsmann, A. & Eisert, P. (2008). Tracking deformable surfaces with optical flow in the presence of self occlusions in monocular image sequences, *CVPR Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment*, Anchorage, USA.

Hilsmann, A. & Eisert, P. (2009). Joint estimation of deformable motion and photometric parameters in single view video, *ICCV 2009 Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment*, Kyoto, Japan, pp. 1–6.

Hilsmann, A., Schneider, D. & Eisert, P. (2010). Realistic cloth augmentation in single view video under occlusions, *Computers & Graphics* .

Huber, P. (1981). *Robust Statistics*, John Wiley & Sons,.

Lim, J. & Yang, M.-H. (2005). A direct method for modeling non-rigid motion with thin plate spline, *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR 2005)*, pp. 1196–1202.

Lowe, D. (2003). Distinctive Image Features from Scale-Invariant Keypoints, *Int. Journal of Computer Vision* 60(2): 91–110.

McCullagh, P. & Nelder, J. A. (1998). *Generalized linear models*, Chapman & Hall, London.

Metaxas, D. N. (1996). *Physics-Based Deformable Models: Applications to Computer Vision, Graphics, and Medical Imaging*, Kluwer Academic Publishers, Norwell, MA, USA.

MPG (1999). *ISO/IEC 14496-2: Coding of audio-visual objects - Part 2: Visual, (MPEG-4 visual)*.

Nastar, C., Moghaddam, B. & Pentland, A. (1996). Generalized image matching: Statistical learning of physically based deformations, pp. 589–598.

Ostermann, J. (1994). Object-oriented analysis-synthesis Coding (OOASC) based on the source model of moving flexible 3D-objects, *IEEE Trans. on Image Processing* 3(5).

Pilet, J., Lepetit, V. & Fua, P. (2008). Fast non-rigid surface detection, registration and realistic augmentation, *Int. Journal of Computer Vision* 76(2): 109–122.

Pizarro, D. & Bartoli, A. (2007). Shadow resistant direct image registration, *Proc. of the 15th Scandinavian Conference on Image Analysis (SCIA 2007)*, pp. 928–937.

Salzmann, M., Pilet, J., Ilic, S. & Fua, P. (2007). Surface deformation models for nonrigid 3d shape recovery, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 29(8): 1481–1487.

Scholz, V. & Magnor, M. (2006). Texture replacement of garments in monocular video sequences, *Rendering Techniques 2006: Eurographics Symposium on Rendering*, pp. 305–312.

Silveira, G. & Malis, E. (2007). Real-time visual tracking under arbitrary illumination changes, *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR 2007)*, pp. 1–6.

Taubin, G. (1995). A signal processing approach to fair surface design, *Proc. of ACM SIGGRAPH 1995*, pp. 351–358.

Terzopoulos, D. & Metaxas, D. (1991). Dynamic 3d models with local and global deformations: deformable superquadrics, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 13(7): 703–714.

Thirion, J. (1996). New Feature Points Based on Geometric Invariants for 3D Image Registration, *Int. Journal of Computer Vision* 18(2): 121–137.

Torresani, L., Yang, D., Alexander, E. & Bregler, C. (2001). Tracking and modeling non-rigid objects with rank constraints, *Proc. of Int. Conf on Computer Vision and Pattern Recognition, CVPR*.

Wardetzky, M., Mathur, S., Kälberer, F. & Grinspun, E. (2007). Discrete laplace operators: No free lunch, *Proc. of 5th Eurographics Symposium on Geometry Processing*, Aire-la-Ville, Switzerland, pp. 33–37.

Wedderburn, R. W. M. (1974). Quasi-likelihood functions, generalized linear models, and the gauss-newton method, *Biometrika* 61(3): 439–447.

White, R. & Forsyth, D. A. (2006). Retexturing single views using texture and shading., *Proc. European Conf. on Computer Vision (ECCV 2006)*, pp. 70–81.