An efficient unified approach to direct visual tracking of rigid and deformable surfaces

Ezio MALIS

***Abstract*— Image-based deformations are generally used for visual tracking of deformable objects moving in the 3D space. For the visual tracking of deformable objects, this assumption has shown to give good results. However it is not satisfying for the visual tracking of 3D rigid objects as the underlying structure cannot be directly estimated. The general belief is that obtaining the 3D structure directly is difficult. In this article, we propose a parameterization that is well adapted either to track deformable objects *or* to recover the structure of 3D objects. Furthermore, the formulation leads to an efficient implementation that can considerably reduce the computational load and it is therefore more adapted to real-time robotic applications. Experiments with simulated and real data validate the approach for deformable object visual tracking and 3D structure estimation. The computational efficiency is also compared to standard methods.**

I. INTRODUCTION

Visual tracking of rigid and deformable surfaces is an active field of research and has many applications for example in medical imagery, augmented reality or robotics. In this article, we focus on iterative methods that are potentially real-time since we are interested in robotic applications such visual servoing. We also focus on methods that do not rely on any off-line learning step as for example [9], [7], [5], [12], [8]. Visual tracking methods can be roughly classified between feature-based and direct methods. Features-based visual tracking generally consists in extracting features in images and then finding the correspondences based on descriptors. These associations can then be used to recover the deformation of the underlying object. The strength of features-based methods reside in the possibility of working on large displacements. The difficulty is to ensure correct associations at a low computational cost. Direct approaches [11] minimize a similarity measure between a reference template and a region of the image to track warped with appropriate geometric and photometric parameters. The underlying assumption is that the deformations between two views of the surface are small. This will typically be the case in video sequences or after an initialization by feature matching. The main advantage of dense visual tracking is accuracy. Initial work mainly focused on the visual tracking of planar rigid surfaces [13], [16] with the iterative Gauss-Newton minimization of the sum of squared differences (SSD) between a reference template and a template in a new image. The same optimization approach can be used for the direct visual tracking of deformable surfaces [3]. The contribution of this article is in the field of direct parametric methods. We propose a parameterization that is well adapted either to deformable object tracking or to the visual tracking of complex 3D surfaces with the direct recovery of the 3D structure. We also focus on the problem of the efficiency of the visual tracking. One important step towards real-time applications with fast frame rate has been improving the efficiency of the Gauss-Newton optimization method. Two approaches are possible for building efficient algorithms. The first one is to keep the same convergence rate (the number of iterations needed to obtain the minimum of the similarity measure) while reducing the computational cost per iteration. This can be achieved by pre-computing partially [10] or completely [1] the Jacobian used in the minimization. The main limitation of these approaches is that, contrarily to [13], they can only be applied to certain warps. Furthermore, Baker et. al. [2] have shown that in the case of surfaces in the 3D Cartesian space the convergence rate of the inverse compositional algorithm is not equivalent to the convergence rate of [13]. An alternative approach for building efficient algorithms is to keep the same computational cost per iteration while increasing the convergence rate. This can be achieved for example by using an efficient second-order minimization method [4]. This approach has been applied to the estimation of a homography for the visual tracking of planar surfaces. In this paper, we investigate how to extend this approach to the visual tracking of continuous surfaces. We propose a flexible and efficient algorithm that can be used for the visual tracking of rigid and deformable surfaces. Compared to existing techniques, a great efficiency is obtained by reducing the number of iterations needed to converge to the minimum of the similarity measure. This leads to a visual tracking algorithm that is more adapted to real-time robotic applications. Experiments with simulated and real data validate the technique for the efficient visual tracking of 3D surfaces.

Ezio Malis is with INRIA. 2004, route des Lucioles. 06902 Sophia Antipolis, France. Email: Ezio.Malis@sophia.inria.fr

II. MODELING

A. Camera and transformation models

We consider a pinhole camera. A 3D point $\mathbf{m} = (x, y, z, 1)$ projects onto the image point $\mathbf{p} = (u, v, 1)$ as follows:

$$\mathbf{p} \propto \left[\begin{array}{cc} \mathbf{K} & 0 \end{array} \right] \mathbf{m} \tag{1}$$

where $\mathbf{K}$ is the upper triangular matrix containing the camera intrinsic parameters. The camera displacement is represented by a (4×4) matrix $\mathbf{T}$ homeomorphic to $\mathbb{SO}(3) \times \mathbb{R}^3$ and containing the rotation matrix $\mathbf{R} \in \mathbb{SO}3$ and the translation vector $\mathbf{t} \in \mathbb{R}^3$. The 3D point $\mathbf{m}$ is eventually deformed to

the point $\mathbf{m}^{'}$ in the same reference frame. The new 3D point projects into a new image point $\mathbf{p}^{'} = (u^{'}, v^{'}, 1)$:

$$\mathbf{p}^{'} \propto \left[\begin{array}{cc} \mathbf{K}^{'} & 0 \end{array} \right] \mathbf{T}\, \mathbf{m}^{'} \tag{2}$$

The camera intrinsic parameters $\mathbf{K}^{'}$ of the new image may be different from the parameters of the reference image. The deformation of the 3D point is divided into two parts:

$$\mathbf{m}^{'} = \frac{1}{\alpha}\mathbf{m} + \boldsymbol{\beta} \tag{3}$$

where, in the first part, $\alpha \in \mathbb{R}^{+}$ is a scale factor that take into account only deformations that change the 3D structure of the object in the reference frame but do not change the reference image and in the second part, $\boldsymbol{\beta} = (\beta_x, \beta_y, 0, 0) \in \mathbb{R}^2$ is a vector that take into account the remaining deformations. If the surface is rigid we obviously have $\alpha = 1$, $\boldsymbol{\beta} = 0$ and $\mathbf{m}^{'} = \mathbf{m}$. Plugging equations (1) and (3) into equation (2) we obtain:

$$\mathbf{p}^{'} \propto \mathbf{H}\,(\mathbf{p} + \boldsymbol{\delta}) + \rho\, \mathbf{e} \tag{4}$$

where $\mathbf{H} = \mathbf{K}^{'}\mathbf{R}\mathbf{K}^{-1}$ is the homography of the plane at infinity, $\mathbf{e} = \mathbf{K}^{'}\mathbf{t}$ is the epipole in the reference image, $\rho = \alpha/z$ is a projective depth and $\boldsymbol{\delta} = (\delta_u, \delta_v, 0)$ is an image coordinates deformation vector. The homography matrix is invertible ($\det(\mathbf{H}) \neq 0$) and it is defined up to a scale factor. Thus, we can normalize the matrix such that $\mathbf{H} \in \mathbb{SL}(3)$ ($\det(\mathbf{H}) = 1$). When the camera is not calibrated, we cannot measure directly the transformation matrix $\mathbf{T}$ but the following (4×4) matrix $\mathbf{Q}$ homeomorphic to $\mathbb{SL}(3) \times \mathbb{R}^3$:

$$\mathbf{Q} = \left[\begin{array}{cc} \mathbf{H} & \mathbf{e} \\ 0 & 1 \end{array} \right] \tag{5}$$

When the 3D surface is rigid $\boldsymbol{\delta} = 0$ and ρ is constant. However, the values of ρ are a function of the image coordinates $\mathbf{p}$. Since ρ is unknown, we can let it vary with respect to time and this allows to compute several deformations without any additional cost. When the deformation of the 3D surface is any we add the unknowns $\boldsymbol{\delta}$. Again, the values of $\boldsymbol{\delta}$ are functions of the image coordinates $\mathbf{p}$. Finally, let us remark that when the object stops to deform we can fix ρ and $\boldsymbol{\delta}$ once and for all and the matrix $\mathbf{Q}$ becomes the only unknown.

B. Surface and deformation models

We suppose that the camera observe a continuous textured surface of the 3D Cartesian space. In some cases we may have a simple parametric model of the 3D surface. For example, a planar surface can be simply modeled with three parameters (the scaled normal vector to the plane). We can model the projective depths of a surface point corresponding to each pixel $\mathbf{p}$ with the following function:

$$\rho = f(\mathbf{p}; \mathbf{s}) \tag{6}$$

where $\mathbf{s}$ is a vector containing the parameters defining ρ. When this model is unknown we propose two methods to approximate it. In both cases, the first step is to select q image points $\mathbf{c}_k$ ($k = \{1, ..., q\}$) called centers. The number of centers q depends on the complexity of the surface we want to approximate. Centers can be dynamically added or removed. In the first method we suppose that our unknown parameters in the vector $\mathbf{s}$ are the projective depths of the surface points that projects onto the centers: $\mathbf{s} = (\rho_1, \rho_2, ..., \rho_q)$. The remaining unknown projective depths for all pixels in the area of interest are computed by interpolation. A second method we propose for approximating the surface ρ is to use Radial Basis Functions [6]. We chose for example thin-plate splines and a first degree polynomial. In this case, the projective depth of point $\mathbf{p}$ is computed as follows

$$f(\mathbf{p}; \mathbf{s}) = \boldsymbol{\gamma}^{\top}\mathbf{p} + \sum_{k=1}^{q} \lambda_k \phi(\|\mathbf{p} - \mathbf{c}_k\|) \tag{7}$$

where $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \gamma_3)$ and $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_q)$ are unknown parameters and $\phi(r) = r^2 \log(r)$. To compute the parameters of the RBF we need to impose the side conditions ans the interpolation conditions [6]. Let $\mathbf{C}^{\top} = [\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_q]$ the (3×q) full rank matrix containing all the coordinates of the centers. The side conditions to be added to equation (7) are $\mathbf{C}^{\top}\boldsymbol{\lambda} = 0$. This implies $\boldsymbol{\lambda} \in \ker(\mathbf{C}^{\top})$. Let $\mathbf{v}_j$ ($j \in \{4, ..., q\}$) be a vector basis of $\ker(\mathbf{C}^{\top})$ which can be obtained from the SVD (Singular Value decomposition) of $\mathbf{C}^{\top}$. Then we have $\boldsymbol{\lambda} = \sum_{j=4}^{q} \gamma_j \mathbf{v}_j$. In this case, the vector $\mathbf{s}$ contains the parameters defining the RBF: $\mathbf{s} = (\gamma_1, \gamma_2, ..., \gamma_q)$. The interpolation conditions $f(\mathbf{c}_k; \mathbf{s}) = \rho_k$ can be imposed directly if we can measure ρ_k or indirectly by minimizing a similarity error in the image (see sectionIII). When the 3D surface is deformable we need also to approximate the two functions f_u and f_v defining the deformations of the image coordinates in the reference frame:

$$\boldsymbol{\delta} = \left[\begin{array}{c} f_u(\mathbf{p}, \mathbf{s}_u) \\ f_v(\mathbf{p}, \mathbf{s}_v) \\ 0 \end{array} \right] \tag{8}$$

where $\mathbf{s}_u$ and $\mathbf{s}_v$ are the corresponding parameters. The two approximation methods described above can be used to approximate the functions but the RBFs provide better results thanks to their regularization properties. In the experiments, we have tested both approximation methods obtaining similar results for rigid surfaces. We have chosen arbitrarily to select the centers either on a regular grid or centered on interest points. When the interest points were well distributed in the area of interest we obtain similar results.

C. Unified warping function model

We define a set of parameters $\boldsymbol{\eta}$ (composed of the camera and surface parameters) needed to align the two images of the surface. From equation (4) we define a warping function $\mathbf{w}(\bullet; \boldsymbol{\eta})$ as a function $\mathbb{P}^2 \mapsto \mathbb{P}^2$:

$$\mathbf{p}^{'} = \mathbf{w}(\mathbf{p}; \boldsymbol{\eta}) \tag{9}$$

We let $\mathbf{H}$ be any homography matrix and we consider ρ as a projective depth. Even if it cannot be computed explicitly, we suppose that the inverse warping function $\mathbf{p} = \mathbf{w}^{-1}(\mathbf{p}^{'}; \boldsymbol{\eta})$ exists. For rigid objects, we set $\boldsymbol{\delta} = 0$ and this warping function is able to capture all the possible projective deformations. For example if the surface is a

plane then we can set $\rho = 0$ and the warp reduces to the standard homographic warp. For deformable surfaces we can capture all projective deformations by letting the parameters $\mathbf{s}$ vary with respect to time. The generalization of the warping to deal with any deformation is thus straightforward by considering $\boldsymbol{\delta}$ varying with time. In the particular case when the surface is rigid and the cameras are calibrated, i.e. $\mathbf{K}$ and $\mathbf{K}^{'}$ are known, we can set $\boldsymbol{\eta} = (\mathbf{T}, \mathbf{s})$ homeomorphic to the Lie group $\mathbb{SO}(3) \times \mathbb{R}^q$. In the general case, we set $\boldsymbol{\eta} = (\mathbf{Q}, \mathbf{s})$ homeomorphic to the Lie group $\mathbb{SL}(3) \times \mathbb{R}^{3q}$. The proposed unified warping function is well adapted for the efficient visual tracking of both rigid and deformable surfaces. On the other hand, standard image-based warping maps like for example the one used in [3] are not adapted to the efficient visual tracking of rigid surfaces. Since they use a warping function that is composed by an affine transformation part and a non-linear one, much more parameters are needed to capture all projective transformations. Consider for example the case of the simple visual tracking of a planar surface that can be done by estimating 8 parameters only (a homography matrix). The affine part (6 parameters) cannot capture the projective deformations, thus the non-linear part is used. More than 2 parameters are needed to capture the remaining deformations. On the other hand, our warping map is able to capture all possible projective deformation using the 8 homography parameters and the non-linear part of the RBF is not needed.

III. Visual Tracking

Let us assume that we have acquired two images I and $I^{'}$ of the same surface. We make the standard assumption that the changes in intensity are only due to camera (or surface) motion. The standard "brightness constancy assumption" can be reformulated as follows. There exists an optimal $\overline{\boldsymbol{\eta}}$ such that the image $I^{'}$ can be warped back to exactly match the reference image:

$$\mathcal{I}^{'}(\mathbf{w}(\mathbf{p}; \overline{\boldsymbol{\eta}})) = \mathcal{I}(\mathbf{p}) \tag{10}$$

We suppose to have a prediction $\widehat{\boldsymbol{\eta}}$ of $\overline{\boldsymbol{\eta}}$. In visual tracking applications the prediction can be provided by a filter or simply the parameters computed at the previous image. We search for the unknown $\widetilde{\boldsymbol{\eta}}$ such that $\widehat{\boldsymbol{\eta}} \circ \widetilde{\boldsymbol{\eta}} = \overline{\boldsymbol{\eta}}$. The composition law depends on the group we are considering (see section II-C). We suppose that the increment $\widetilde{\boldsymbol{\eta}}$ is close to the identity element e of the simply connected Lie group. We can thus parametrize $\widetilde{\boldsymbol{\eta}}$ via the exponential map $\widetilde{\boldsymbol{\eta}} = \boldsymbol{\eta}(\widetilde{\mathbf{x}})$, where $\widetilde{\mathbf{x}}$ are coordinates in the Lie algebra. Finally, we need to solve the following non-linear equations (one equation per pixel):

$$\mathcal{I}^{'}(\mathbf{w}(\mathbf{p}; \widehat{\boldsymbol{\eta}} \circ \boldsymbol{\eta}(\mathbf{x}))) = \mathcal{I}(\mathbf{p}) \tag{11}$$

This non-linear system of equation is generally over-constrained and it may not have an exact solution in the presence of measure and modeling errors. Instead, a least-squares problem is iteratively solved by extending the efficient second-order method proposed by Benhimane and Malis [4] to 3D surfaces.

A. Efficient second-order method

We stack all the n equations (11) in a $(n \times 1)$ vector $\mathbf{y}(\mathbf{x})$. The second-order Taylor expansion of $\mathbf{y}(\mathbf{x})$ about $\mathbf{x} = 0$ is:

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}(0) + \mathbf{J}(0)\mathbf{x} + \frac{1}{2}\mathbf{M}(\mathbf{x})\mathbf{x} + \mathbf{r}_T(\|\mathbf{x}\|^3) \tag{12}$$

where $\mathbf{M}(\mathbf{x}) = (\mathbf{x}^\top \mathbf{H}_1, \mathbf{x}^\top \mathbf{H}_2, ..., \mathbf{x}^\top \mathbf{H}_n)$ is a $(n \times m)$ matrix containing the symmetric Hessians matrices and $\mathbf{r}_T(\|\mathbf{x}\|^3)$ is a third-order Taylor remainder. Similarly, the first-order Taylor expansion of $\mathbf{J}(\mathbf{x})$ about $\mathbf{x} = 0$ is:

$$\mathbf{J}(\mathbf{x}) = \mathbf{J}(0) + \mathbf{M}(\mathbf{x}) + \mathbf{R}_T(\|\mathbf{x}\|^2) \tag{13}$$

where $\mathbf{R}_T(\|\mathbf{x}\|^2)$ is a second-order $(n \times m)$ remainder matrix. Plugging equation (13) into equation (12) we obtain an exact third order expansion of $\mathbf{y}(\mathbf{x})$ without the second order terms:

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}(0) + \frac{1}{2}(\mathbf{J}(0) + \mathbf{J}(\mathbf{x}))\,\mathbf{x} + \mathbf{r}(\|\mathbf{x}\|^3) \tag{14}$$

indeed, $\mathbf{r}(\|\mathbf{x}\|^3) = \mathbf{r}_T(\|\mathbf{x}\|^3) - \mathbf{R}_T(\|\mathbf{x}\|^2)\mathbf{x}/2$ is a third-order remainder. Thus, a second-order approximation of $\mathbf{y}(\widetilde{\mathbf{x}})$ is:

$$\mathbf{y}(\widetilde{\mathbf{x}}) \approx \mathbf{y}(0) + \frac{1}{2}(\mathbf{J}(0) + \mathbf{J}(\widetilde{\mathbf{x}}))\,\widetilde{\mathbf{x}} \tag{15}$$

Since the system of equation is over-constrained, we solve the following least squares problem:

$$\min \|\mathbf{y}(0) + \frac{1}{2}(\mathbf{J}(0) + \mathbf{J}(\widetilde{\mathbf{x}}))\,\widetilde{\mathbf{x}}\|^2 \tag{16}$$

The Jacobian $\mathbf{J}(0)$ can be completely computed from image data. The problem is that in the general case $\widetilde{\mathbf{x}}$ is needed to compute $\mathbf{J}(\widetilde{\mathbf{x}})$. In [4] the authors use the following property $\mathbf{J}(\widetilde{\mathbf{x}})\,\widetilde{\mathbf{x}} = \mathbf{J}^{'}\,\widetilde{\mathbf{x}}$ where the Jacobian $\mathbf{J}^{'}$ can be measured from image data directly. However, this property is valid only if the warping map defines a group action on $\mathbb{P}^2$. When estimating the 3D surface structure, this is not the case. In section III-C, we show that only a part of the Jacobian needs to be approximated. We found experimentally that this approximation is worthwhile as it improves greatly the convergence rate over any first-order approach. Let $\widehat{\mathbf{J}} \approx \mathbf{J}(\widetilde{\mathbf{x}})$, the solution of the problem (16) provides the increment:

$$\widetilde{\mathbf{x}} = -2\left(\mathbf{J}(0) + \widehat{\mathbf{J}}\right)^{+}\mathbf{y}(0) \tag{17}$$

where $^+$ denotes the matrix pseudo-inverse. The update of the estimated parameters is $\widehat{\boldsymbol{\eta}} \leftarrow \widehat{\boldsymbol{\eta}} \circ \boldsymbol{\eta}(\widetilde{\mathbf{x}})$.

B. Computation of $\mathbf{J}(0)$

The Jacobian $\mathbf{J}(0)$ is defined as follows:

$$\mathbf{J}(0) = \nabla_{\mathbf{x}} \mathcal{I}(\mathbf{w}(\mathbf{p}; \widehat{\boldsymbol{\eta}} \circ \boldsymbol{\eta}(\mathbf{x}))|_0 \tag{18}$$

It can be written as the product of 3 Jacobians using the chain derivation rule:

$$\mathbf{J}(0) = \mathbf{J}_{I'}\mathbf{J}_w(\widehat{\boldsymbol{\eta}})\mathbf{J}_x(0) \tag{19}$$

The first Jacobian contains the spatial derivatives of the image warped with $\widehat{\boldsymbol{\eta}}$:

$$\mathbf{J}_{I'} = \nabla_{\mathbf{q}} \mathcal{I}(\mathbf{w}(\mathbf{q}; \widehat{\boldsymbol{\eta}}))|_{\mathbf{p}} \tag{20}$$

The second and third Jacobians are:

$$\mathbf{J}_w(\widehat{\boldsymbol{\eta}}) = \left(\nabla_{\mathbf{q}} \mathbf{w}(\mathbf{q}; \widehat{\boldsymbol{\eta}})|_{\mathbf{p}} \right)^{-1} \nabla_{\boldsymbol{\eta}} \mathbf{w}(\mathbf{p}; \widehat{\boldsymbol{\eta}} \circ \boldsymbol{\eta})|_e \quad (21)$$

$$\mathbf{J}_x(0) = \nabla_{\mathbf{x}} \boldsymbol{\eta}(\mathbf{x})|_0 \quad (22)$$

C. Computation of $\mathbf{J}(\widetilde{\mathbf{x}})$

The Jacobian $\mathbf{J}(\widetilde{\mathbf{x}})$ is defined as follows:

$$\mathbf{J}(\widetilde{\mathbf{x}}) = \nabla_{\mathbf{x}} \mathcal{I}(\mathbf{w}(\mathbf{p}; \widehat{\boldsymbol{\eta}} \circ \boldsymbol{\eta}(\mathbf{x})))|_{\widetilde{\mathbf{x}}} \quad (23)$$

It can be written as the product of 3 Jacobians using chain derivation rule:

$$\mathbf{J}(\widetilde{\mathbf{x}}) = \mathbf{J}_I \mathbf{J}_w(\overline{\boldsymbol{\eta}}) \mathbf{J}_x(\widetilde{\mathbf{x}}) \quad (24)$$

The first Jacobian contains the spatial derivatives of the image warped with $\overline{\boldsymbol{\eta}}$ (i.e. the reference image):

$$\mathbf{J}_I = \nabla_{\mathbf{q}} \mathcal{I}(\mathbf{w}(\mathbf{q}; \overline{\boldsymbol{\eta}}))|_{\mathbf{p}} \quad (25)$$

The second and third Jacobians are:

$$\mathbf{J}_w(\overline{\boldsymbol{\eta}}) = \left(\nabla_{\mathbf{q}} \mathbf{w}(\mathbf{q}; \overline{\boldsymbol{\eta}})|_{\mathbf{p}} \right)^{-1} \nabla_{\boldsymbol{\eta}} \mathbf{w}(\mathbf{p}; \overline{\boldsymbol{\eta}} \circ \boldsymbol{\eta})|_e \quad (26)$$

$$\mathbf{J}_x(\widetilde{\mathbf{x}}) = \nabla_{\mathbf{x}} \widetilde{\boldsymbol{\eta}}^{-1} \circ \boldsymbol{\eta}(\mathbf{x}) \Big|_{\widetilde{\mathbf{x}}} \quad (27)$$

The Jacobian $\mathbf{J}_w(\overline{\boldsymbol{\eta}})$ cannot be exactly computed since $\overline{\boldsymbol{\eta}}$ is unknown. However, $\widehat{\boldsymbol{\eta}} \approx \overline{\boldsymbol{\eta}}$ and we can use the Jacobian $\mathbf{J}_w(\widehat{\boldsymbol{\eta}})$ instead. The last Jacobian $\mathbf{J}_x(\widetilde{\mathbf{x}})$ verifies the following property $\mathbf{J}_x(\widetilde{\mathbf{x}})\widetilde{\mathbf{x}} = \mathbf{J}_x(0)\widetilde{\mathbf{x}}$ from the Lie algebra parameterization [4]. Finally, $\widehat{\mathbf{J}} = \mathbf{J}_I \mathbf{J}_w(\widehat{\boldsymbol{\eta}}) \mathbf{J}_x(0)$ and the second-order increment (17) can be written:

$$\widetilde{\mathbf{x}} = -2 \left((\mathbf{J}_I + \mathbf{J}_{I'}) \mathbf{J}_w(\widehat{\boldsymbol{\eta}}) \mathbf{J}_x(0) \right)^+ \mathbf{y}(0) \quad (28)$$

The computational cost of the second-order approximation is equivalent to the cost of the Gauss-Newton method (the average of $\mathbf{J}_I$ and $\mathbf{J}_{I'}$ is negligible with respect to the computation of the pseudo-inverse).

IV. Experiments

We applied the proposed visual tracking to rigid and deformable surfaces. We select a template in the image $\mathcal{I}_0$ and then we align $\mathcal{I}_1$. To initialize the minimization we suppose initially that the observed surface is a 3D plane parallel to the image plane. The parameters estimated in the alignment of images $\mathcal{I}_0$ and $\mathcal{I}_k$ are used as a starting point for the alignment of images $\mathcal{I}_0$ and $\mathcal{I}_{k+1}$. We can obviously use a filter for prediction and smoothing. When the surface is rigid, the parameters $\mathbf{s}$ are constant and a simple Kalman filter with constant position should work well. For deformable surfaces an appropriate motion model for the parameters $\mathbf{s}$ should be selected. We have successfully tested our algorithm on several video sequences. The image sequences are available on the ESM visual tracking website [14]. The brightness constancy assumption is often violated in the sequences. We found it sufficient to normalize the images at each iteration in order to take into account ambient light changes. However, more complex photometric models could easily be included to our approach. Due to the lack of space, only few experiments are presented in this paper. Additional experimental results can be found in [15] and the corresponding videos can be downloaded from the ESM visual tracking website [14].

A. Comparison with the ground truth

The proposed approach is tested with a sequence with known ground truth. A video sequence has been simulated by warping an image onto a sphere. The sphere has a radius of 30 cm and its center is initially at 1 meter in front of the camera. In the simulation, the camera is calibrated and we can use $\boldsymbol{\eta} = (\mathbf{T}, \mathbf{s})$ homeomorphic to the Lie group $\mathbb{SO}(3) \times \mathbb{R}^q$. We compare the second-order method with the Gauss-Newton method. A (400×400) template is selected (see Figure 1). The centers for the surface approximation are placed on a regular (5×5) grid. To simulate a real-time experiment we fixed the number of iterations of each algorithm to 5. With our unoptimized Matlab code this corresponds to a fixed time of 5 seconds per image.

Fig. 1. Case study for testing the proposed approach and comparing the convergence rate of minimization algorithms. The image on the left shows the regular grid defining the area of interest in the first image. The image on the right shows the corresponding template.

Despite the simple spherical structure of the surface, the standard Gauss-Newton fails to register the images since it does not have enough iterations to converge (with 10 iterations/image the minimization works fine). On the other hand, using the efficient second-order minimization the images are correctly registered. The average RMS (Root Mean Square) error for the registration of the 40 images is 4.9 gray-levels (over 256). Figure 2 shows the visual tracking results after 40 images. The last registered area of interest is correctly aligned with the reference template.

Fig. 2. Visual tracking with the second-order minimization. The image on the left shows the transformation of the regular grid after 40 images of the sequence. The image on the right shows that the warped area of interest in the reference frame is equal to the reference template.

Thus, we are able to directly recover the surface up to a scale factor. Obviously, the precision of the reconstruction depends on the translation made by the camera. Indeed, if the camera motion is a pure rotation the structure is not observable. However, the visual tracking is correctly performed. Figure 3 displays two 3D views of the reconstructed sphere after computing the unknown scale factor from the ground truth. The mean error on the depths corresponding to all the pixels of the template is 1.7 mm while the standard deviation is 1.4 mm.

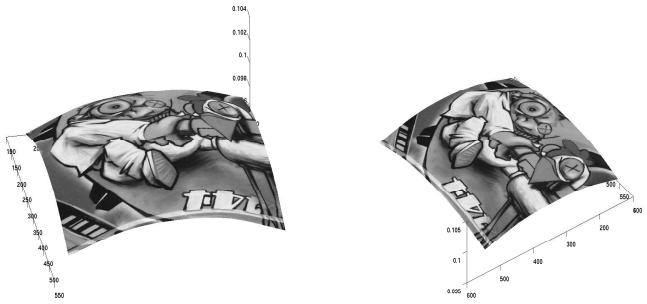

Fig. 3. Two 3D views of the reconstructed surface. The surface lie on a sphere of 300 mm radius.

B. Visual tracking of rigid surfaces

In this experiment we track a vase in a sequence of 450 images acquired with an uncalibrated camera. We select the centers on a regular (6×6) grid and we use again RBFs for surface approximation. Figure 4 shows a selection of five images from the total results. The registration is correctly performed for all images. The average RMS error is 13.8 gray-levels (over 256) but there were again several changes of illumination. These changes are visible in the second row of Figure 4 where the area of interest reprojected in the reference frame appears brighter at the end of the sequence. The average number of iterations per image is 6.

C. Visual tracking of deformable surfaces

In this experiment we track a balloon in a sequence of 1082 images. We select a (262×262) template and the centers are placed on a 4×4 grid. In this experiment we do not use RBFs for surface approximations. We use bi-cubic interpolation to approximate the surface given the centers. We found it sufficient to estimate projective deformations only (i.e. we set $\boldsymbol{\delta} = 0$). Thus, we estimate 27 parameters which leads to a fixed time of 3.5 seconds per image (5 iterations/image) with a Matlab code. The right column of figure 5 shows that all the images of the sequence have been correctly aligned with the reference template despite the strong change in size of the balloon and its deformation. The average intensity error over all the sequence is 5.6. The deformation of the balloon surface is shown in the left column of figure 5 by the deformation of the regular grid.

V. Conclusion

In this paper, we have proposed an efficient method for the visual tracking of surfaces in the 3D Cartesian space. The same approach can be used both for rigid and deformable surfaces. For rigid surfaces, in the case of a calibrated camera we directly obtain an approximation of the structure of the surfaces. In the uncalibrated case, the registration can be used to self-calibrate the camera and obtain the 3D structure. We have tested two methods for surface approximation but we find that the RBFs provide more stable results in the presence of low-textured and/or deformable surfaces. The main improvement over standard algorithms based on Gauss-Newton method is efficiency in registering large deformations between two images. The efficient second order method achieve a faster convergence. The algorithm needs fewer iterations to converge while the computational cost per iteration is equivalent to a Gauss-Newton method. The proposed algorithm is thus more suitable for fast real-time robotic applications. Further improvements on the computation speed can be achieved by selecting dynamically the centers in order to reduce the number of unknowns.

References

[1] S. Baker and I. Matthews, "Equivalence and efficiency of image alignment algorithms," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, December 2001, pp. 1090–1097.

[2] S. Baker, R. Patil, K. Cheung, and I. Matthews, "Lucas-kanade 20 years on: Part 5," Robotics Institute, Carnegie Mellon University, Tech. Rep. CMU-RI-TR-04-64, November 2004.

[3] A. Bartoli and A. Zisserman, "Direct estimation of non-rigid registrations," in *British machine vision conference*, vol. 2, 2004, pp. 899–908.

[4] S. Benhimane and E. Malis, "Real-time image-based tracking of planes using efficient second-order minimization," in *IEEE/RSJ International Conference on Intelligent Robots Systems*, vol. 1, Sendai, Japan, September-October 2004, pp. 943–948.

[5] M. Black and A. Jepson, "Eigen-tracking: robust matching and tracking of articulated objects using a view-based reprsentation," *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.

[6] J. Carr, W. Fright, and R. Beatson, "Surface interpolation with radial basis functions for medical imaging," *IEEE Transactions Med. Imag.*, vol. 16, no. 1, February 1997.

[7] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearence models," in *European Conference on Computer Vision*, vol. 2, 1998, pp. 484–498.

[8] D. J. Fleet, M. J. Black, Y. Yacoob, and A. D. Jepson, "Design and use of linear models for image motion analysis," *Int. J. Comput. Vision*, vol. 36, no. 3, pp. 171–193, 2000.

[9] M. Gleicher, "Projective registration with difference decomposition," in *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, 1997, pp. 331–337.

[10] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 10, pp. 1025–1039, 1998.

[11] M. Irani and P. Anandan, "All about direct methods," in *In W. Triggs, A. Zisserman, and R. Szeliski, editors, Vision Algorithms: Theory and practice. Springer-Verlag, 1999.*, 1999, pp. 267–277.

[12] M. La Cascia, S. Sclaroff, and V. Athitsos, "Fast, reliable head tracking under varying illumination: an approach based on registration of textured-mapped 3D models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 4, pp. 322–336, 2000.

[13] B. Lucas and T. Kanade, "An iterative image registration technique with application to stereo vision," in *International Joint Conference on Artificial Intelligence*, 1981, pp. 674–679.

[14] E. Malis, "Esm visual tracking web site," http://esm.gforge.inria.fr.

[15] ——, "An efficient unified approach to direct image registration of rigid and deformable surfaces," INRIA, Research Report 6089, January 2007. [Online]. Available: https://hal.inria.fr/inria-00123105

[16] H. Y. Shum and R. Szeliski, "Construction of panoramic image mosaics with global and local alignment," *International Journal of Computer Vision*, vol. 16, no. 1, pp. 63–84, 2000.

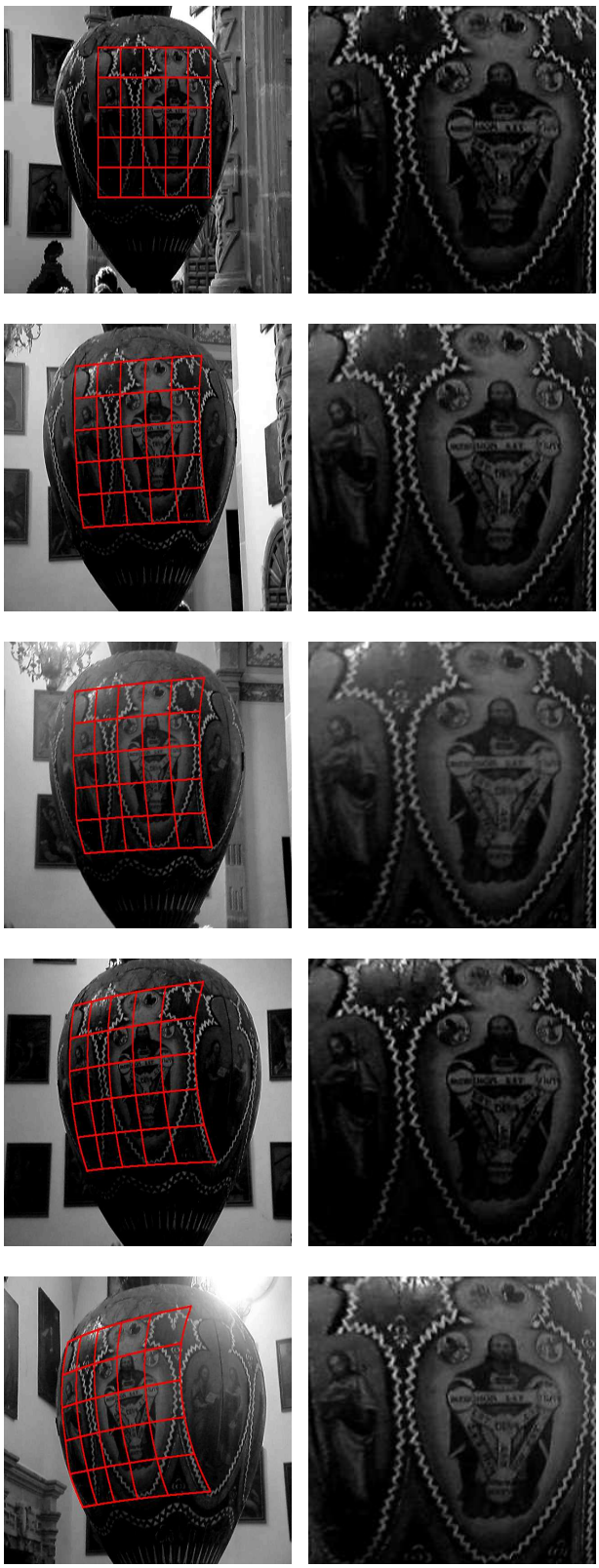

Fig. 4. Visual tracking of a vase in a sequence of 450 images acquired with an uncalibrated camera. The left column shows the (6×6) regular grid used to track the area of interest in the sequence. The right column shows the area of interest registered with respect to the template. Due to the illumination changes, the average RMS error is 13.8 gray-levels (over 256). The average number of iterations per image is 6.

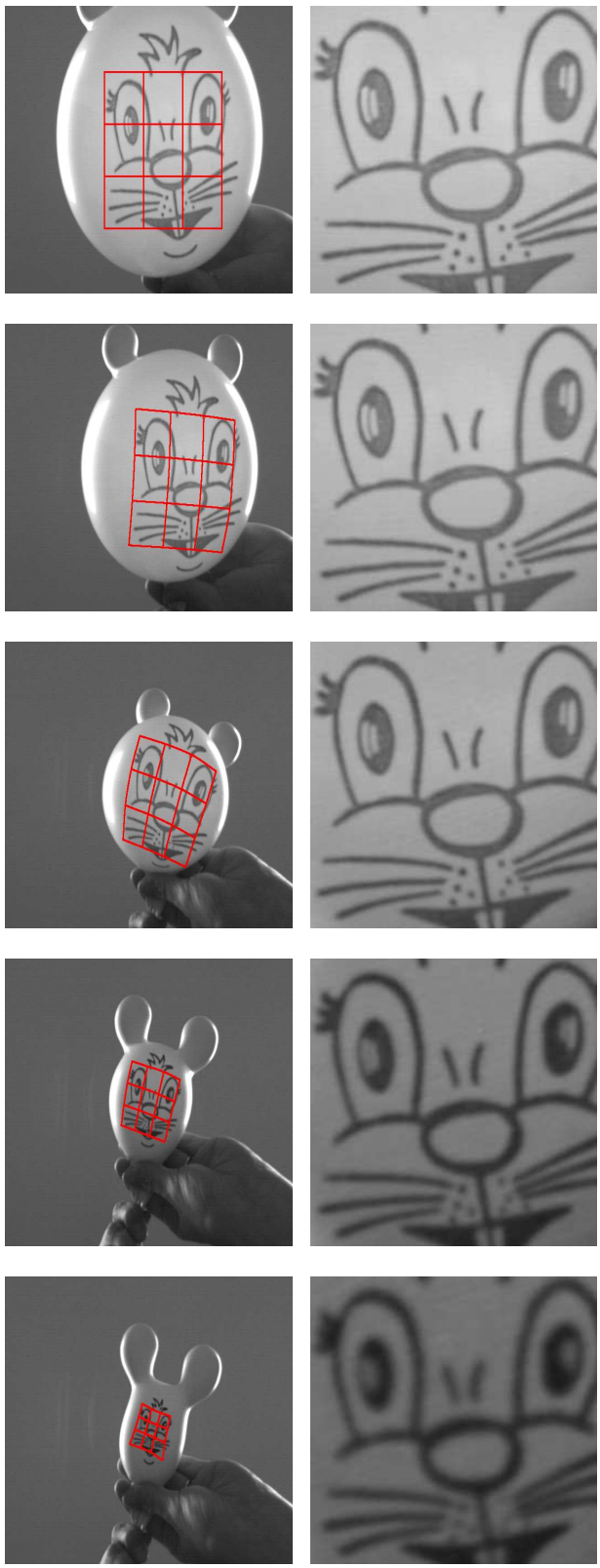

Fig. 5. Visual tracking of a deformable surface in a sequence of 1082 images acquired with an uncalibrated camera. The left column shows the (4×4) regular grid used to track the area of interest in the sequence. The right column shows the area of interest registered with respect to the template. The average intensity error over all the sequence is around 5.6. The average number of iterations per image is 6.