Deformable Template Tracking in 1ms

David Joseph Tan[1]
tanda@in.tum.de
Stefan Holzer[2]
sholzer@fyusion.com
Nassir Navab[1]
navab@in.tum.de
Slobodan Ilic[3]
Slobodan.Ilic@siemens.com

[1] CAMP
Technische Universität München
Munich, Germany

[2] Fyusion, Inc.
San Francisco, USA

[3] Siemens AG
Munich, Germany

Abstract

We address the problem of real-time deformable template tracking. Our approach relies on linear predictors which establish a linear relation between the image intensity differences of a template and the corresponding template transformation parameters. Up to this work, linear predictors have only been used to handle linear transformations such as homographies to track planar surfaces. In this paper, we introduce a method to learn non-linear template transformations that allows us to track surfaces that undergo non-rigid deformations. These deformations are mathematically modelled using 2D Free Form Deformations. Moreover, the simplicity of our approach allows us to track deformable surfaces at extremely high speed of approximately 1 ms per frame that has never been shown before.

To evaluate our algorithm, we perform an extensive analysis of our method's performance on synthetic and real sequences with different types of surface deformations. In addition, we compare our results from the real sequences to the feature-based tracking-by-detection method [20], and show that the tracking precisions are similar but our method performs 100 times faster.

1 Introduction

Template tracking involves the estimation of the transformation parameters that define the rigid motion of the planar template. Typically, this transformation parameters encode linear transformation based on homographies. However, when it comes to tracking surfaces that undergo non-rigid deformations, standard template tracking approaches cannot be applied and deformable transformations have to be used. Related works commonly use image features and strong prior deformation models [22] for tracking surface deformations. Nevertheless, due to the non-rigid deformation a large number of feature points is required that results in a significant number of outliers. Therefore, these methods concentrate on outlier rejection and are rarely concerned by the speed or precision of tracking. There are very few methods that use dense image intensities [18] for tracking deformable surfaces. The main benefit of using dense pixel intensities is that a lack of a large number of feature points is compensated by the dense pixel information and, thus, allows tracking of less textured surfaces such as faces.

One of the methods that have been successful for rigid tracking of planar templates is based on learning linear predictors [13] which map dense image intensity differences to the change of the template transformation parameters. In this paper, we introduce a very simple method that integrates surface deformations into the linear predictors and consequently allows extremely fast tracking of deformable surfaces. Surface deformations are parametrized with a 2D Free Form Deformation model based on cubic B-Splines interpolation. The resulting deformable template tracking algorithm can run at approximately 1 ms per frame.

We evaluate our approach using a dataset of 20 images to compare its performance to the rigid linear predictor in Sec. 5.1. The objective is to compare their performances and explicitly determine if our robustness is affected by the deformation model under linear transforms. It is also used to determine the effects of different parametrization to its robustness and find the optimum parameters that gives the best results. Furthermore, in Sec. 5.2, we evaluate the tracking precision using real video sequences and compare our results to the tracking-by-detection approach of Pilet *et al*. [20]. In terms of precision, we obtained similar tracking performance but our method is more than 100 times faster which makes it more suitable for low-powered devices. Finally, we present several qualitative results in the *Supplementary Material* where we demonstrate tracking through different deformations as well as tracking in low-lighting condition.

2 Related work

Literature regarding deformable template tracking is classified into two well-known categories – feature-based methods [4, 20, 21, 24, 25] where features are used to characterize the images and to estimate the deformation from one image to the other; and, pixel-based methods [1, 7, 12, 18] where the intensity of the images is directly used to find the deformation.

Feature-based methods. In general, the steps in feature-based methods can be summarized into feature correspondence search, outlier rejection mechanism, and finally deformation model estimation. To detect and match features, most of the literature [20, 21, 22, 24, 25] use keypoint descriptors such as SIFT [17], SURF [2] or randomized trees [15]. Furthermore, given the correctly matched features from the reference image to the target image, the thin plate splines of Bookstein [3] is commonly used to estimate the deformation between the matched keypoints because it offers a closed-form solution. Another approach with a closed-form solution is from Salzmann and Fua [22] that reconstructs a 3D model from dense correspondences between the reference template configuration, and the input image relying on the easy to learn local surface deformation models and non-linear constraints between vertices of the local surface parts.

Unlike rigid transforms where we estimate a 2D homography with a maximum of 8 degrees of freedom and easily remove incorrectly matched features by using RANSAC [6], deformable transforms have significantly more degrees of freedom which makes it difficult to identify whether a matched features is an inlier or an outlier. Thus, a number of works [16, 20, 21, 23] have focused on the outlier rejection for deformable transforms. One of the prominent works is from Pizarro *et al*. [21] where they assume that the deformation is locally smooth by using a Delaunay triangulation and iteratively build a set of strong inliers. Another is from Pilet *et al*. [20] where, aside from the quadratic deformation energy that penalizes local surface curvature, they introduce a correspondence energy that includes a ridged shape robust estimator with a decreasing radius of confidence such that, as the radius of confidence decreases, the estimator becomes more selective in choosing inliers. In addition, Tran *et*

al. [23] shows how to utilize a "simple" RANSAC algorithm with Radial Basic Function (RBF).

Therefore, the problem in using feature-based methods is that they depend on the repeatability of the features and require a sufficiently large number of features in the reference image to have enough inliers when matched with the target image. This generates a time-consuming outlier rejection and constraints their application to large templates. As a result, feature-based template matching has reached real-time application at only 10 fps [20].

Pixel-based methods. Less attention has been given to deformable template matching using pixel-based methods to perform in real-time applications. These methods are usually associated with the energy minimization-based image registration, where their energy is composed of the data term that expresses the similarity measure of two images and the smoothing term that defines the rigidity of the deformation. Notable works in this field is the self-occlusion reasoning of Gay-Bellile *et al*. [7] where they added a term called shrinker that shrinks the deformation to prevent it from folding at self-occluded regions.

On one hand, the main advantage of using an energy-based image registration is that it can achieve a subpixel accuracy [12] and can handle self-occlusion [7]; on the other, it requires an extensive amount of time to complete as well as a good initialization to avoid local minima.

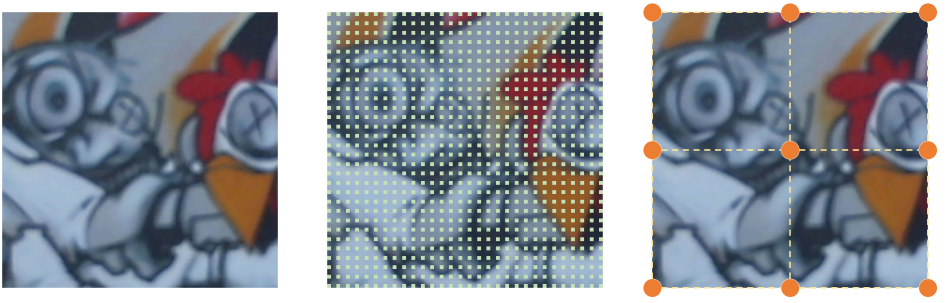
Works that have accomplished real-time application include the Active Appearance Models (AAM) [5, 19]. It relies on applying Principal Component Analysis (PCA) on hundreds of labelled images to find the mean and eigenvectors of the shape, which defines the geometric structure, and appearance, which includes the image intensities. In this way, their model can be represented by the linear combination of the resulting mean and eigenvectors. Another real-time method that exploits dense pixel information and handles deformations is the work of Malis [18]. He mainly concentrates on energy-minimization based template tracking called Efficient Second-order Minimization (ESM) which is designed for tracking rigid motion of planar templates and extends it with thin plate splines to handle surface deformations. However, their method for deformable surfaces runs at 3.5 seconds per frame.

Nonetheless, we introduce a pixel-based method with a learning-based approach that extends the rigid template tracking of Jurie and Dhome [13] to handle non-rigid deformations where we use the Free-Form Deformations (FFD) based on cubic B-Splines [8, 14] as our model. Based on our evaluations, our approach is stable for several types of deformations and can compete against the feature-based method of [20] in terms of precision. However, our method is running at less than 1 ms per frame which makes it the fastest deformable 2D tracking method up-to-date.

3 Deformation Model

Our 2D deformation model is based on the Free-Form Deformations (FFD) using cubic B-Splines. This model is composed of control points that are uniformly arranged in a $K \times L$ grid around a template of size $W \times H$ pixels. Each control point stores a displacement vector that defines the movement of the template. Therefore, the deformation Ψ of a pixel $\mathbf{x} = (x,y)^\top$ is computed as:

$$\Psi \circ \mathbf{x} = \mathbf{x} + \mathcal{D}(\mathbf{x}) \tag{1}$$



(a) Template (b) Sample Points (c) Control Points

Figure 1: Within the template in (a), these succeeding figures demonstrate (b) the location of the sample points that are represented in small squares and (c) the location of the control points that are represented in circles.

such that the displacement vector $\mathcal{D}(\mathbf{x})$ of the pixel $\mathbf{x}$ is interpolated from the displacement vector $\mathbf{d}_{i,j}$ at the control points (i,j) using [8]:

$$\mathcal{D}(\mathbf{x}) = \sum_{l=0}^{3}\sum_{m=0}^{3} B_l(u)\,B_m(v)\,\mathbf{d}_{i+l,j+m} \tag{2}$$

where $i = \lfloor x/\delta x \rfloor - 1$, $j = \lfloor y/\delta y \rfloor - 1$, $u = x/\delta x - \lfloor x/\delta x \rfloor$ and $v = y/\delta y - \lfloor y/\delta y \rfloor$ with the control point spacings $\delta x = W/(K-1)$ and $\delta y = H/(L-1)$, while B_l and B_m are the cubic B-Spline interpolation coefficients [14].

4 Method

Given the frame I_t at time t of a video sequence, we define the reference template on the initial image I_{t_0} using n_s sample points $\{\mathbf{x}_s \in \mathbb{R}^2\}_{n=1}^{n_s}$ on a $W \times H$ rectangular grid as shown in Fig. 1(b). It follows that the intensities on the reference image are described by $\{I_{t_0}(\mathbf{x}_s)\}_{s=1}^{n_s}$.

Initially, the homography $\mathbf{T}_{t_0}$ transforms the location of the template from the template coordinate system, where the template center is the origin, to the image coordinate system. To manage the non-rigid deformation from Sec. 3, we introduce $n_c = K \times L$ control points on the template with a set of displacements $\{\mathbf{d}_c \in \mathbb{R}^2\}_{c=1}^{n_c}$, where $\mathbf{d}_c$ is zero on the initial frame, and transforms the sample points as $\Phi(\mathbf{d}_c) \circ \mathbf{x}_s = \mathbf{T}(\Psi \circ \mathbf{x}_s)$. In vector form, we assign the parameter vector $\mu = [\mathbf{d}_c]_{c=1}^{n_c}$ and the intensity vector $\mathbf{i}(\mu,t) = [I_t(\Phi(\mu) \circ \mathbf{x}_s)]_{s=1}^{n_s}$.

4.1 Objective function

The goal of frame-to-frame tracking is to update the parameters at $t+\tau$ using the given parameters μ_t at time t [9]. Thus, we seek the change in parameters $\delta\mu$ by minimizing:

$$\varepsilon(\delta\mu) = \left\| \mathbf{i}(\mu_t + \delta\mu, t+\tau) - \mathbf{i}(\mu_{t_0}, t_0) \right\|^2. \tag{3}$$

Using Taylor Series Approximation, we have:

$$\mathbf{i}(\mu_t + \delta\mu, t+\tau) \approx \mathbf{i}(\mu_t, t) + \mathbf{J}_\mu(\mu_t, t)\delta\mu + \tau\frac{\partial \mathbf{i}}{\partial t}(\mu_t, t) = \mathbf{i}(\mu_t, t+\tau) + \mathbf{J}_\mu(\mu_t, t)\delta\mu \tag{4}$$

where $\mathbf{J}_{\mu_t} = \mathbf{J}_\mu(\mu_t, t)$ is the Jacobian matrix of $\mathbf{i}$ with respect to μ, and $\frac{\partial \mathbf{i}}{\partial t}(\mu_t, t)$ is estimated using the forward difference approximation. We substitute Eq. 4 into Eq. 3 as:

$$\varepsilon(\delta\mu) \approx \left\| \mathbf{i}(\mu_t, t+\tau) + \mathbf{J}_{\mu_t}\delta\mu - \mathbf{i}(\mu_{t_0}, t_0) \right\|^2. \tag{5}$$

Finally, to find $\delta\mu$ with the minimum ε, we set the $\bigtriangledown_{\delta\mu}\varepsilon = 0$, which implies that:

$$\mathbf{i}(\mu_t, t+\tau) + \mathbf{J}_{\mu_t}\delta\mu - \mathbf{i}(\mu_{t_0}, t_0) = 0 \Rightarrow \delta\mu = -\mathbf{J}^+_{\mu_t}\delta\mathbf{i}(\mu_t, t+\tau) \tag{6}$$

where $\mathbf{J}^+_\mu = \left(\mathbf{J}^\top_\mu \mathbf{J}_\mu\right)^{-1}\mathbf{J}^\top_\mu$ and $\delta\mathbf{i}(\mu, t) = \mathbf{i}(\mu, t) - \mathbf{i}(\mu_{t_0}, t_0)$. Similar to [13], our method aims to replace the time-consuming $-\mathbf{J}^+_{\mu_t}$, that defines the relation between the parameter update $\delta\mu$ and the change in intensity $\delta\mathbf{i}$, by learning a linear predictor matrix $\mathbf{A}$.

4.2 Learning a linear predictor

At $t = t_0$, we create the set $\{(\delta\mathbf{i}_\omega, \delta\mu_\omega)\}_{\omega=1}^{n_\omega}$ where I_ω is a deformed version of the template such that $\Phi(\delta\mu_\omega) \circ \mathbf{x}_s$ is the ground truth location of the sample points on I_ω. For each ω, the parameters in $\delta\mu_\omega$ are generated randomly with values within a radius of τ_μ around the control points.

Looking at $\delta\mathbf{i}_\omega = \mathbf{i}(\mu_{t_0}, \omega) - \mathbf{i}(\mu_{t_0}, t_0)$ more closely, it is noteworthy to mention that the location of the sample points remains constant while the template deforms across all ω. However, deforming the entire template is inefficient and time-consuming. Thus, instead of deforming the template, we transform the location of the sample points using $-\delta\mu_\omega$. In this way, the image I_{t_0} simply becomes a look-up table for $\mathbf{i}(\mu_{t_0}, \omega)$ in $\delta\mathbf{i}_\omega$.

Therefore, we can concatenate the vectors from $\{(\delta\mathbf{i}_\omega, \delta\mu_\omega)\}_{\omega=1}^{n_\omega}$ to construct the matrices $\mathbf{Y} = \left[\delta\mu_1, \delta\mu_2, \cdots, \delta\mu_{n_\omega}\right]$ and $\mathbf{H} = [\delta\mathbf{i}_1, \delta\mathbf{i}_2, \cdots, \delta\mathbf{i}_{n_\omega}]$ with the relation $\mathbf{Y} = \mathbf{AH}$, and learn the linear predictor $\mathbf{A}$ using [13]:

$$\mathbf{A} = \mathbf{Y}\mathbf{H}^\top\left(\mathbf{H}\mathbf{H}^\top\right)^{-1}. \tag{7}$$

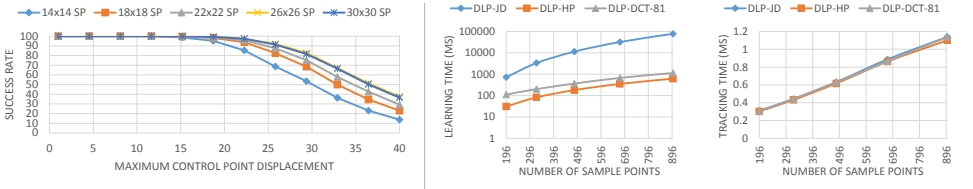
4.3 Tracking with linear predictors

The location of the template is estimated by using a parameter update vector $\delta\mu_{t+\tau}$ that corrects the transformation of $\mathbf{x}_s$ from the frame at t to $t+\tau$. Using the image difference vector $\delta\mathbf{i}_{t+\tau} = \mathbf{i}(\mu_t, t+\tau) - \mathbf{i}(\mu_{t_0}, t_0)$, where $\mathbf{i}(\mu_t, t+\tau)$ is a collection of image intensities in the current frame with the sample point locations of the previous frame, the relation between $\delta\mathbf{i}_{t+\tau}$ and $\delta\mu_{t+\tau}$ in Eq. 3 can be defined using the linear predictor [13] as:

$$\delta\mu_{t+\tau} = \mathbf{A}\delta\mathbf{i}_{t+\tau}. \tag{8}$$

Therefore, to track the reference template, we compute $\delta\mu_{t+\tau}$ using the given vector $\delta\mathbf{i}_{t+\tau}$ and the learned matrix $\mathbf{A}$.

Homography Update. Since the estimated displacements are relative to the coordinate system of the reference template, we keep track of a homography $\mathbf{T}_t$ which approximately maps the current control points from the predicted pose of the template in the current frame to the reference template. $\mathbf{T}_t$ is estimated by computing the homography between the corner point positions in the reference coordinate system and their location obtained in the previous frame. To apply the displacements obtained from the linear predictors, we initially warp all control points back into the reference coordinate system.



(a) Tracking Robustness (b) Learning (c) Tracking

Figure 2: This shows the (a) tracking robustness, (b) learning time and (c) tracking time with respect to the number of sample points $n_s = K \times K$ when using different learning modalities – JD [13], HP [11] and DCT [10]. Using the optimum $26 \times 26 = 676$ sample points arrangement, we can learn the template in 353.38 ms and track in 0.87 ms.

Coarse-to-Fine Strategy. Several linear predictors are learned for a single template that compensate for large to small deformations such that each linear predictor refines the previously estimated parameter vector. The coarse-to-fine approach is applied in a hierarchical way. At the first two levels of hierarchy, linear predictors are learned with the coarse 2×2 control points; while, in the last three, linear predictors are learned with 3×3 control points. Note that between the second and third level, the number of considered control points increases and we linearly interpolate the positions of the additional control points.

Furthermore, each linear predictor is iteratively used to generate better results. In our work, three iterations for each linear predictor are performed.

5 Evaluation

We perform both qualitative and quantitative evaluations of our approach where each template is 150×150 pixels. Learning is parameterized with $n_\omega = 5 \cdot n_s$ training samples where n_s is the number of sample points. All evaluations run on an Intel(R) Core(TM) i7-3820QM CPU with only one core used.

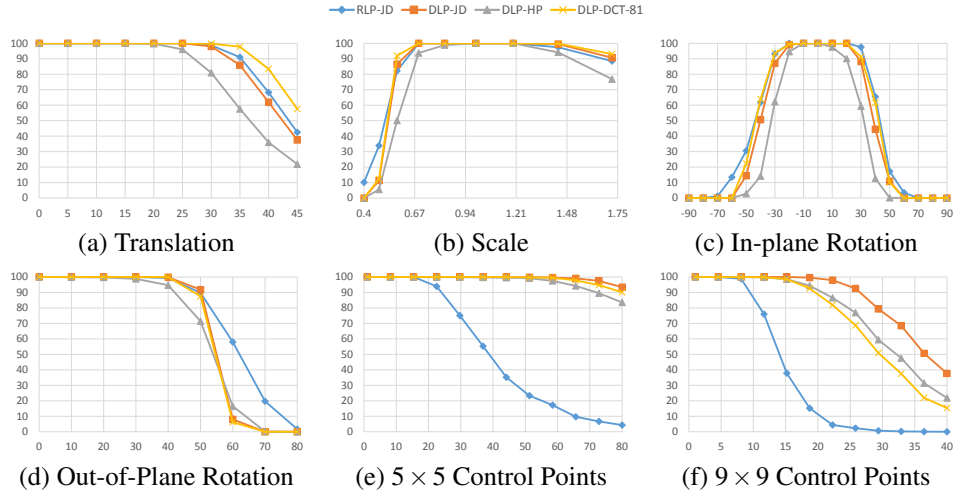
The first quantitative evaluation in Sec. 5.1 uses a dataset of 20 images and evaluates the tracking robustness of our method with respect to rigid and non-rigid transformations. It also includes the comparison with the fast learning methods of Holzer *et al.* [10, 11] where they modify Eq. 7 to avoid the inversion of the large matrix $\mathbf{H}\mathbf{H}^\top$.

Another quantitative evaluation in Sec. 5.2 involves three video sequences that consider problems such as camera noise and motion blur from real deformations. These sequences are used to test the precision of our algorithm where we compute the distance of 10 manually selected points on the template from their tracked positions to the ground truth. Consequently, our results are compared to the ones from the feature-based approach of Pilet *et al.* [20].

Our *Supplementary Material* includes a video that shows quantitative and qualitative results to demonstrate our tracking performance under different deformations and in low-lighting condition.

5.1 Tracking robustness

We randomly warp the 20 images in the dataset as shown in the *Supplementary Material* using rigid transformation, including translation, scale, in-plane rotation and out-of-plane rotation, as well as non-rigid transformations produced from FFD where control points are located on the whole image. This FFD is parameterized using the maximum displacement that a control point can randomly move without overlapping each other.



(a) Translation (b) Scale (c) In-plane Rotation

(d) Out-of-Plane Rotation (e) 5×5 Control Points (f) 9×9 Control Points

Figure 3: Comparison of tracking robustness between Rigid Linear Predictors (RLP-JD) [13], and the Deformable Linear Predictors (DLP) using different learning approaches – JD [13], HP [11] and DCT-81 [10] with $n_r = 81$ coefficients. It evaluates using (a-d) rigid as well as non-rigid transforms using FFD with (e) 5×5 and (f) 9×9 control points on the entire image where the x-axis shows the maximum control point displacement. Note that 5×5 and 9×9 does not refer to the control points of the template but rather to the deformation of the entire image.

The objective is to remove environmental factors such as noise and motion blur; to have full control of the warping parametrization; to determine the range of movements that our approach can handle; and, to find the optimum parameters. Furthermore, we look into the rigid transforms to find out whether using deformable warping parametrization in $\delta\mu$ compromises the linear transformations that [13] can manage.

In these evaluations, two parameters are important – time and robustness. The optimum case is to perform at high-speed with a fast learning procedure without deteriorating the tracking robustness. We define tracking robustness as the percent of successfully recovered template deformations after generating random warps on each image of the dataset. Here, tracking is successful if the average distance of the backwarped sample points of the tracked template to the original sample points on the image is less than 1.5 pixels.

5.1.1 Number of sample points

From the previous section, we know that the sample points subsample the template and, in effect, replaces the template in all computations. Thus, it is crucial to determine the optimum number of sample points such that the time-performance ratio is optimal. We evaluate the tracking robustness in Fig. 2(a) with respect to different number of sample points. It shows that there is no significant improvement between $26 \times 26 = 676$ and $30 \times 30 = 900$ sample points. Thus, the optimum number of sample points for our application is 26×26.

Contrary to the rigid linear predictors [13], our approach requires more sample points because it has more parameters. This is one of the reason why we use a maximum of 3×3 control points as discussed in Sec. 4.3. Another reason is because 3×3 control points are sufficient to give a realistic tracking performance as shown in the *Supplementary Material*.

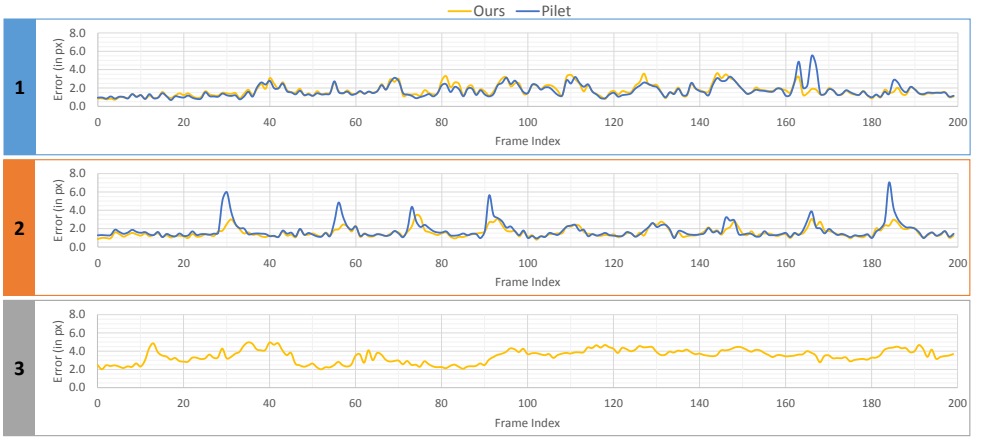


Figure 4: These graphs plot the resulting average distance error (in pixels) of the tracked points to its ground truth from the three sequences.

Furthermore, Fig. 2(b-c) shows the average learning and tracking time while using different learning techniques [10, 11, 13] to replace Eq. 7. For a 26×26 sample point arrangement, we can learn at a minimum of 353.38 ms using [11] and track at approximately 0.87 ms. This illustrates our main contribution and, in comparison to other deformable template tracking algorithms, we claim that our method achieves real-time tracking at the minimum time.

5.1.2 Synthetic evaluation

Using the optimum 26×26 samples points, we compare the performance of our algorithm with the rigid linear predictor of Jurie and Dhome [13] using synthetic rigid and non-rigid deformations of the images of the dataset. The objective is to identify how well our method can handle rigid transformations in comparison to the rigid linear predictor method [13] and how our method perform under non-rigid deformations using different learning methods.

For the rigid transforms in Fig. 3(a-d), the results show that there is no significant difference between [13] and our method. This signifies that our performance does not deteriorate in rigid transformations. On the other hand, it is not surprising that [13] fails in non-rigid deformations in Fig. 3(e-f). Regarding Fig. 3(f), our method starts to fail after 30 pixel movements. This is because, if the displacements of the 9×9 control points are large, the resulting image has a smudge-like deformation which is not a realistic surface deformation.

5.2 Tracking accuracy

We evaluate using three real video sequences with a duration of 200 frames which are taken using a Logitech Webcam C525. The first two sequences consist of a textured template that undergoes a flag-like deformation with different textures, while the last one is a face sequence with relatively low texture undergoing arbitrary facial expressions.

Our aim is to track the location of 10 points that have been manually selected in all sequences. To evaluate these points, we compute the average distance between the tracked points, using our method and the tracking-by-detection method of Pilet *et al*. [20], from the manually provided ground truth point locations. The points are initialized in the sequences as shown in Fig 5. For [20], the authors kindly provided the results for the sequences.

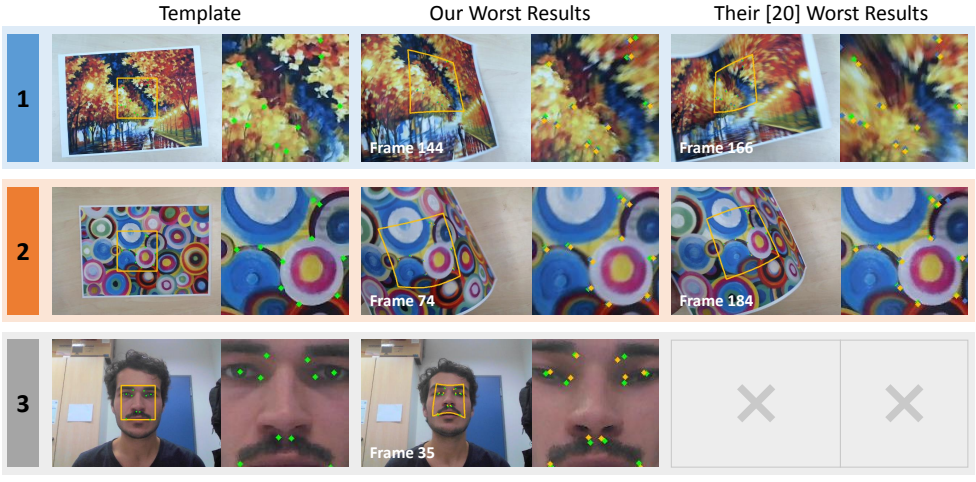


Figure 5: These figures show the learned template and the worst results from [20] as well as our approach. For each pair of images, the rectangular image on the left is a frame from the video sequence while the square image on the right is the backprojected template using our approach. Moreover, the ten points in all these images are labelled with green for ground truth, blue for the results from [20] and yellow for our results.

To generate a realistic flag-like deformation, we induce a fast motion on one corner of the template which results in a motion blur. Throughout the first two sequences, the largest mean error is 3.6 pixels for our approach as shown in Fig. 4 and 7.0 pixels for [20]. Notably, when we backproject the template using the results of our approach, the resulting images are very similar to the template even if frame 166 from Sequence 1 suffers from motion blur. When comparing these two approaches, we can see that the errors in tracking are very similar but our approach is 100 times faster.

On the face sequence, all distance errors are less than 5 pixels as plotted in Fig. 4 while we show in Fig. 5 that the frame with the highest error is the one when the head is tilted up. According to the authors of [20], their work does not work on faces because their geometric model assumes a planar surface bending smoothly without holes.

6 Discussion

Using Eq. 8, linear predictors are learned to track templates by updating the change in parameters ($\delta\mu$) based on the given change in intensity ($\delta\mathbf{i}$). This implies that, for a given $\delta\mathbf{i}$, there is only one unique location where the template can move. Hence, there must be a one-to-one correspondence between $\delta\mathbf{i}$ and $\delta\mu$; otherwise, the movement from $\delta\mu$ becomes ambiguous. This ambiguity is present in all template tracking using linear predictors whether rigid or deformable and causes tracking failures. Therefore, texture is an important factor in linear predictors. For instance, uniformly colored regions or repeating textures (e.g. fine stripped lines) on the template affects the unique relation between $\delta\mathbf{i}$ and $\delta\mu$.

Occlusion is another problem that affects the values in $\delta\mathbf{i}$. However, due to its speed, partial occlusion is handled by using multiple templates as illustrated in Fig. 7 from [10] such that, when some of the templates are occluded, the successfully tracked templates can roughly locate the occluded template for the succeeding frames. Furthermore, another solu-

tion is used in the face tracking application where we pair the linear predictors with a rigid template detector to find the location of the template after occlusion.

Moreover, the range of deformation depends on the model in Sec. 3. Since we are using FFD with cubic splines interpolation, we are limited to smooth deformations which is adequate for a wide range of applications.

Finally, a larger number of control points requires a larger number of sample points due to the exponential increase in degrees of freedom. For example, when using 5×5 instead of 3×3, the degrees of freedom raise from 18 to 50. However, note that a large number of sample points can restrict tracking to large templates only, while our current setup can handle varying template sizes. Thus, we suggest to use multiple templates instead of more control points because it enjoys the benefits of speed and independence in tracking which leads to robustness to partial occlusions, non-lambertian surface and shadows.

References

[1] A. Bartoli and A. Zisserman. Direct estimation of non-rigid registrations. In *British Machine Vision Conference*, 2004.

[2] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *European Conference on Computer Vision*, 2006.

[3] F. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1989.

[4] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 2003.

[5] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active Appearance Models. *PAMI*, 23 (6):681–685, 2001.

[6] M.A. Fischler and R.C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications ACM*, 1981.

[7] V. Gay-Bellile, A. Bartoli, and P. Sayd. Direct estimation of nonrigid registrations with image-based self-occlusion reasoning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[8] B. Glocker, N. Komodakis, G. Tziritas, N. Navab, and N. Paragios. Dense image registration through mrfs and efficient linear programming. *Medical Image Analysis*, 2008.

[9] G.D. Hager and P.N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998.

[10] S. Holzer, S. Ilic, D. Tan, and N. Navab. Efficient learning of linear predictors using dimensionality reduction. In *Asian Conference on Computer Vision*, 2012.

[11] S. Holzer, M. Pollefeys, S. Ilic, D.J. Tan, and N. Navab. Online learning of linear predictors for real-time tracking. In *European Conference on Computer Vision*. 2012.

[12] M. Irani and P. Anandan. About direct methods. *Vision Algorithms: Theory and Practice*, 2000.

[13] F. Jurie and M. Dhome. Hyperplane approximation for template matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[14] S. Lee, G. Wolberg, and S.Y. Shin. Scattered data interpolation with multilevel b-splines. *IEEE Transactions on Visualization and Computer Graphics*, 1997.

[15] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *Conference on Computer Vision and Pattern Recognition*, 2005.

[16] X. Li and Z. Hu. Rejecting mismatches by correspondence function. *International Journal of Computer Vision*, 2010.

[17] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 2004.

[18] E. Malis. An efficient unified approach to direct visual tracking of rigid and deformable surfaces. In *International Conference on Intelligent Robots and Systems*, 2007.

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

[20] J. Pilet, V. Lepetit, and P. Fua. Fast non-rigid surface detection, registration and realistic augmentation. *International Journal of Computer Vision*, 2008.

[21] D. Pizarro and A. Bartoli. Feature-based deformable surface detection with self-occlusion reasoning. *International Journal of Computer Vision*, 2012.

[22] M. Salzmann and P. Fua. Linear local models for monocular reconstruction of deformable surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[23] Q.H. Tran, T.J. Chin, G. Carneiro, M. Brown, and D. Suter. In defence of ransac for outlier rejection in deformable registration. In *European Conference on Computer Vision*. 2012.

[24] J. Zhu and M. Lyu. Progressive finite newton approach to real-time nonrigid surface detection. In *Conference on Computer Vision and Pattern Recognition*, 2007.

[25] J. Zhu, S. Hoi, and M. Lyu. Nonrigid shape recovery by gaussian process regression. In *Conference on Computer Vision and Pattern Recognition*, 2009.