Randomly Sparsified Synthesis for Model-based Deformation Analysis

Stefan Reinhold, Andreas Jordt, Reinhard Koch

Department of Computer Science, University of Kiel, Germany

Abstract. The tracking of deformation is one of the current challenges in computer vision. Analysis by Synthesis (AbS) based deformation tracking provides a way to fuse color and depth data into a single optimization problem very naturally. Previous work has shown that this can be done very efficiently using sparse synthesis. Although sparse synthesis allows AbS-based tracking to perform in real-time, it requires a great amount of problem specific customization and is limited to certain scenarios. This article introduces a new way of randomized adaptive sparsification of the reference model that adjusts the sparsification during the optimization process according to the required accuracy of the current optimization step. It will be shown that the efficiency of AbS can be increased significantly using the proposed method.

1 Introduction

Pose estimation and 3D object tracking are major research areas in computer vision for more than 40 years. Traditionally a two-stage approach is applied, where first images are analyzed using edges, features, or optical flow, and next the features are used to calculate a result. The benefit of this two-stage approach is a mathematically sound formulation since the problem is reduced to aligning corresponding feature locations within the image.

However, if the reconstruction task is extended to the non-rigid case, the fundamental constraint allowing such a sound formulation is often removed. The continuous improvement of depth cameras in the last 20 years and especially the appearance of the Microsoft Kinect in 2010 has enriched the family of reconstruction algorithms by depth-image based approaches, which in many cases differ significantly from color image based reconstruction. Although for the rigid case mature reconstruction [12] and people tracking [23] techniques are available, generic deformation reconstruction still remains subject to research. Since depth images provide dense distance information, and color images allow to deduce lateral movements, a combination of both sensor types yields a powerful tool to allow ambiguity free deformation reconstruction.

Color and depth information are not easily combined, hence a joint problem formulation is difficult when following the traditional two-stage reconstruction using features. A class of algorithms bypassing the fusion problem is Analysis by Synthesis [14] (AbS, sometimes also referred to as "direct methods"). Instead of deducing a solution in multiple stages from the input image, the optimization problem is formulated as finding parameters for a scene model that generates artificial sensor images, which fit the real input images best. A lot of inherent reconstruction problems are circumvented this way, including the fusion of various input entities. As long as input data of a sensor can be synthesized, the joint problem formulation is straight forward. A disadvantage of AbS is the increased computational work, since the evaluation of the objective function implies a rendering process for every sensor involved and a global search, in many cases without available dertivatives[1], whereas feature correspondences allow local optimization on a function with known derivatives. The key to efficient AbS is the combination of a fast synthesis, based on partial, 'sparse' synthesis [13], and an efficient global optimization scheme. Such a system can even track deforming objects in real-time [24].

In this article, a new way of synthesis sparsification for AbS system is introduced that circumvents the need for manual reference object sparsification, provides automatic adjustment of the synthesis implicitly regarding the reference object complexity, and improves the overall AbS performance significantly. Experiments show a potential speed up of factor 8 and more compared to the established sparse synthesis.

1.1 Related Work

Deformation tracking in combined depth and color (RGB-D) video is a research area that has not been investigated as much as deformation tracking in color video or depth video alone. A lot of research has been carried out on color-based deformation tracking using Non-Rigid Structure from Motion (NRSfM), which was introduced by Bregler et al. [3] as an extension to Structure from Motion (SfM) [25] by a set of deformation spaces, and it has been extended over the years [5,6,26]. Like the original SfM, NRSfM utilizes 2D feature movements to formulate a convex minimization problem which is solved using local (least-squares) optimization methods. This allows to simultaneously recover camera position, object shape, and deformation. If the initial object shape is already known, Salzmann et al. [21] proposed a method that utilizes the object geometry as a deformable triangle mesh, rather than relying on linear deformation spaces. For the multi-camera case, Cagniar et al. [4] introduced a method that is able to track complex human deformation spaces if the background can be subtracted in the input footage. Similar to the multi-camera tracking of Rosenhahn [20], it utilizes the object contour in the input data as well as feature movements to align the tracking target with the input image.

The problem of deformation tracking in depth data is handled in a very different way by most algorithms, as depth images provide less information that can be used to generate correspondences, such as features. Hence, algorithms relying on depth features do only provide good results for small deformations

[1] One exception can be found in [7], where the derivatives of the reprojection error are calculated for a multi-view stereo setting.

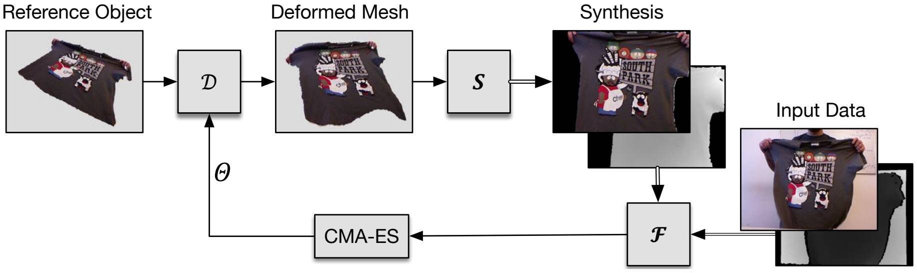

Fig. 1. Overview of the Analysis by Synthesis approach. A reference object is deformed by a deformation function $\mathcal{D}$ using a estimated parameter vector Θ. The sensor set $\mathcal{S}$ synthesizes artificial images which are compared to the input images by a fitness function $\mathcal{F}$. The fitness value is used by the optimizer (CMA-ES) to generate a refined Θ.

[17]. The main advantage of range data over color is its ability to reconstruct a 3D model of the observed scene for every frame, allowing it to directly track deformations in 3D space. Algorithms like [18] perform local, ICP-like [2] shape fitting combined with global smoothness enforcement. Tracking human poses in a depth camera is already well understood and can be solved by machine learning approaches [23]. For many specialized tasks, e.g., skeleton based tracking [10], hand tracking [19,22] real-time solutions are available, as well as for generic models and deformations if they are small enough such that a local, hierarchical optimization can be applied [28]. Xu et al. [27] expressed non-rigid object tracking in RGB-D video as an inference problem in a Conditional Markov Random Field using deformable patches to represent the observed surface. This way they removed the requirement of the tracking target to be completely visible in the first frame. In [11] Innmann et al. combine color feature tracking with a depth-based constraint formulation to dynamically reconstruct geometric shapes from a single RGB-D sensor at real-time.

The remainder of this article is organized as follows: section 2 summarizes the AbS tracking approach and the sparse synthesis as introduced in [13,14], section 3 proposes the new adaptive sparsification method and introduces an approximation algorithm to estimate the inaccuracy induced by sparsification. Section 4 evaluates the proposed method on real data with known ground-truth and section 5 concludes this article.

2 Analysis by Synthesis

The AbS concept is a generative, model-driven approach that differs from traditional reconstruction problem formulations in many key aspects. Input images are not analyzed and processed, instead, a parameter space is established that allows to express the solution as a parameter vector within this search space. A solution guess (model parameters) can be evaluated by synthesizing the input images given the solution candidate and comparing it to the real input (see Fig. 1). This approach can be efficient, if the synthesis is not performed by standard rendering, but in a sparse way [24]. An in-depth introduction to AbS used for deformation reconstruction can be found in [14].

Let M be a textured triangle mesh serving as a reference object for the AbS tracking. Sparse synthesis reduces the triangle mesh information given by M to the indexed set of vertices $V = \{v_i\} \subset \mathbb{R}^3$ and their corresponding texture color $C = \{c_i\} \subset [0, 255]^3$. A deformation parameter $\mathit{\Theta}$ is evaluated as follows:

Given a set of Sensors $\mathcal{S}$, let the input image be given as a function $I_c^{in} : \mathbb{R}^2 \to [0, 255]$ and $I_d^{in} : \mathbb{R}^2 \to \mathbb{R}$ for each color sensor $c \in \mathcal{S}$ and depth sensor $d \in \mathcal{S}$, respectively. Let further $\mathcal{P}_s : \mathbb{R}^3 \to \mathbb{R}^2$ denote the projection function for sensor s, mapping 3D points to the image plane. Let $\mathcal{D}_{\boldsymbol{\theta}} : \mathbb{R}^3 \to \mathbb{R}^3$ be a deformation function parameterized by $\boldsymbol{\theta}$. Then the RMS error for a color sensor s_c is

$$e_{s_c}(\boldsymbol{\theta}) = \left(|V|^{-1} \sum_{(\boldsymbol{v},\boldsymbol{c}) \in V \times C} \|(I_{s_c}^{in} \circ \mathcal{P}_{s_c} \circ \mathcal{D}_{\boldsymbol{\theta}})(\boldsymbol{v}) - \boldsymbol{c}\|_2^2 \right)^{-\frac{1}{2}}, \tag{1}$$

where $\circ$ denotes function composition. The depth error can be defined accordingly. But since using an absolute depth error would make the error measure dependent on the scene scale, we define the relative RMS depth error for a depth sensor s_d as follows:

$$e_{s_d}(\boldsymbol{\theta}) = \left(|V|^{-1} \sum_{\boldsymbol{v} \in V} \frac{\left((I_{s_d}^{in} \circ \mathcal{P}_{s_d} \circ \mathcal{D}_{\boldsymbol{\theta}})(\boldsymbol{v}) - \|\mathcal{D}_{\boldsymbol{\theta}}(\boldsymbol{v}) - \mathcal{O}_{s_d}\|_2\right)^2}{\|\mathcal{D}_{\boldsymbol{\theta}}(\boldsymbol{v}) - \mathcal{O}_{s_d}\|_2^2} \right)^{-\frac{1}{2}}, \tag{2}$$

where $\mathcal{O}_{s_d}$ is the camera center of sensor s_d. In addition to providing invariance to scene scale, this error formulation implicitly regards the accuracy of depth sensors, which usually decreases with distance. A more sophisticated error formulation, exploiting an explicit noise model can be found in [15].

The error sum of all sensors can be defined as a fitness function

$$\mathcal{F}(\boldsymbol{\theta}) = \sum_{s \in \mathcal{S}} \lambda_s e_s(\boldsymbol{\theta}) \tag{3}$$

using weights $\lambda_s \in \mathbb{R}_{>0}$ for each sensor s, which can either be found manually, by automatic adjustment [14], or by deriving them from the probability distribution of the measurements of each sensor [16]. Additional constraints or regularization terms can be added to the fitness function depending on the task at hand.

The tracking can now be formulated as an optimization problem: $\boldsymbol{\theta}^{\star} = \operatorname{argmin} \mathcal{F}(\boldsymbol{\theta})$, which is solved using the Covariance Matrix Adaptation - Evolution Strategy (CMA-ES) optimization scheme [9]. It proved to be very suitable for this class of problems, since it is coordinate system independent, derivation free, very economical with respect to the number of necessary fitness function evaluations, while still being able to find global optima in noisy and non-convex environments [1]. The optimization scheme falls into the class of distribution-based optimizers. For every iteration, a set of samples (parameter vectors $\boldsymbol{\theta}$) is distributed with respect to a multivariate Gaussian distribution and evaluated by the fitness function $\mathcal{F}$. Using the fitness of each parameter vector $\mathcal{F}(\boldsymbol{\theta})$, the Gaussian distribution is updated such that it maximizes the probability of producing higher fitness values in the next iteration.

The combination of a fast implementation of $\mathcal{F}$ and the efficient CMA-ES optimization allows to track deformations in real-time. A crucial parameter when tuning the AbS framework for a real-time tracking task is the degree of sparsification, as the evaluation time of each iteration has a linear relationship to the number of vertices used in the sparse synthesis. Too few vertices will cause the optimization to loose accuracy, while too many vertices cause long optimization times.

In each iteration i of the CMA-ES, a new population of parameter vectors $\{\boldsymbol{\theta}_{i,j}\}_{j=0,\dots,\lambda-1}$ is sampled, so that $\boldsymbol{\theta}_{i,j} \sim \mathcal{N}(\overline{\boldsymbol{\theta}}_{i-1}, \sigma_{i-1}^2 C_{i-1})$, for all $j = 0,\dots,\lambda-1$, where $\overline{\boldsymbol{\theta}}_{i-1}$, σ_{i-1}^2 and C_{i-1} are the population mean, variance and normalized covariance matrix (i.e. $\det C = 1$) of the last iteration, respectively and $\lambda \in \mathbb{N}_{>1}$ is the population size. The fitness function is then evaluated for each individual and the parameter vectors are sorted with respect to their fitness values. For easier notation (w.l.o.g), let the parameter vector indices match their fitness rank, that is $\mathcal{F}(\boldsymbol{\theta}_{i,0}) \geq \mathcal{F}(\boldsymbol{\theta}_{i,1}) \geq \cdots \geq \mathcal{F}(\boldsymbol{\theta}_{i,\lambda-1})$. To calculate a new mean, the sorted parameter vectors are weighted by a weight vector $\boldsymbol{w} \in [0,1]^\lambda$, with $w_0 \leq w_1 \leq \cdots \leq w_{\lambda-1}$ and $\|\boldsymbol{w}\|_1 = 1$. Evolutionary selection is implemented by setting the first $\mu \in \mathbb{N}$ weights to zero. The selection index μ is usually set to $\mu = \frac{\lambda}{2}$ and the non-zero weights are chosen in a linear or logarithmic fashion [8]. The new population mean is then computed as $\overline{\boldsymbol{\theta}}_i = \sum_j w_j \boldsymbol{\theta}_{i-1,j}$. The new covariance matrix C_i is compiled from the old covariance C_{i-1}, the sample covariance of the current distribution and the distribution C_i^1 along the recent path of movements in the parameter space. For further details refer to [9].

3 Adaptive Sparsification

The idea of adaptive sparsification is to exploit an important property of the CMA-ES: The distribution at iteration i around a mean $\overline{\boldsymbol{\theta}}_i$ is completely calculated from ordered sample positions and the update $\overline{\boldsymbol{\theta}}_{i-1} \rightarrow \overline{\boldsymbol{\theta}}_i$ and therefore solely depends on the order induced by the fitness function $\mathcal{F}$ and not by the fitness values themselves.

Now let $\mathcal{F}_s$ be a sparsified version of $\mathcal{F}$, in which only a random subset V_s of V is regarded, with $|V_s| = s|V|$, $s \in (0,1]$. The larger the sparsification rate s^{-1}, the likelier it is that $\mathcal{F}_s$ deviates from $\mathcal{F}$ introducing a sparsification error. But if s is chosen in a way that induces a sparsification error small enough such that it does not change the order of the samples induced by $\mathcal{F}_s$ in each iteration, the sparsification does not affect the optimization at all. Although in practice the missing knowledge about $\mathcal{F}$ and the random selection of V_s does not allow to

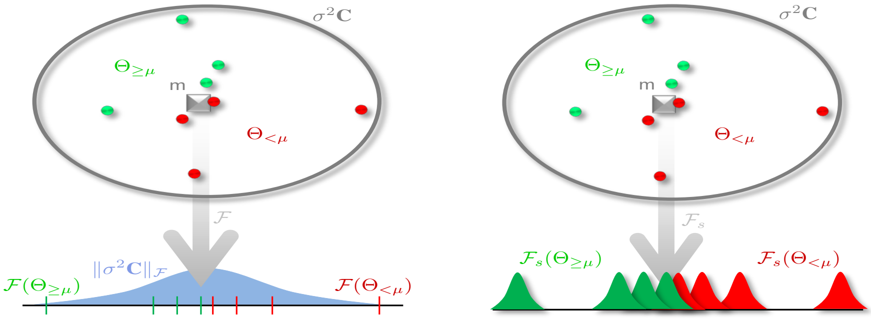
$\sigma^2\mathbf{C}$
$\Theta_{\geq\mu}$
m
$\Theta_{<\mu}$
$\mathcal{F}$
$\|\sigma^2\mathbf{C}\|_{\mathcal{F}}$
$\mathcal{F}(\Theta_{\geq\mu})$
$\mathcal{F}(\Theta_{<\mu})$
$\sigma^2\mathbf{C}$
$\Theta_{\geq\mu}$
m
$\Theta_{<\mu}$
$\mathcal{F}_s$
$\mathcal{F}_s(\Theta_{\geq\mu})$
$\mathcal{F}_s(\Theta_{<\mu})$

Fig. 2. Left: The fitness function $\mathcal{F}$ calculates a scalar fitness value $\mathcal{F}(\boldsymbol{\Theta})$ (lower part) for every individual $\boldsymbol{\Theta} \sim \mathcal{N}(\boldsymbol{m}, \sigma^2 C)$ (upper part). The image $\|\sigma^2 C\|_{\mathcal{F}}$ of the distribution in parameter space is visualized in blue. Right: In contrast to $\mathcal{F}$ (left), the sparsified fitness function $\mathcal{F}_s$ introduces an uncertainty to the results, which is modeled as a Gaussian distribution (lower part) for every $\mathcal{F}_s(\boldsymbol{\Theta})$.

provide a guaranteed preservation of order between $\mathcal{F}$ and $\mathcal{F}_s$, it is expedient to investigate the probability of an order change happening and how to maintaining a low probability throughout the optimization.

For the following derivation we assume that the images of the $\mathcal{N}(\overline{\boldsymbol{\theta}}, \sigma^2 C)$ distributed parameter vectors in $\mathcal{F}$ also follow a gaussian distribution (cf. Fig. 2 left). In general this assumption does not hold since $\mathcal{F}$ depends on the input images and is often neither monotone nor continuos. However, since we are only interested in investigating the probability of an order change of two parameter vectors $\boldsymbol{\theta}_i, \boldsymbol{\theta}_{i+1}$, with $\mathcal{F}(\boldsymbol{\theta}_i) > \mathcal{F}(\boldsymbol{\theta}_{i+1})$, it is sufficient to investigate $\mathcal{F}$ in a local neighborhood of $\boldsymbol{\theta}_i$, in which we can approximate $\mathcal{F}$ linearly. Therefore, let $\|\sigma^2 C\|^2_{\mathcal{F}}$ denote the covariance of the image distribution in $\mathcal{F}$.

For sake of simplicity, let $\lambda = 2$ and $\mu = 1$, so that $\boldsymbol{\theta}_0, \boldsymbol{\theta}_1$, with $\mathcal{F}(\boldsymbol{\theta}_0) > \mathcal{F}(\boldsymbol{\theta}_1)$, are the only individuals of the population. Let $d = \mathcal{F}(\boldsymbol{\theta}_0) - \mathcal{F}(\boldsymbol{\theta}_1)$ and $d_s = \mathcal{F}_s(\boldsymbol{\theta}_0) - \mathcal{F}_s(\boldsymbol{\theta}_1)$ be the distances of the projections under the original and the sparse fitness function, respectively. If $\mathcal{F}(\boldsymbol{\theta})$ and $\mathcal{F}_s(\boldsymbol{\theta})$ are gaussian, then d and d_s are also gaussian with $d \sim \mathcal{N}(0, \|\sigma^2 C\|^2_{\mathcal{F}})$ and $d_s \sim \mathcal{N}(0, \|\sigma^2 C\|^2_{\mathcal{F}_s})$. It can be seen easily, that the probability p of an order change between $\mathcal{F}(\boldsymbol{\theta}_0)$, $\mathcal{F}(\boldsymbol{\theta}_1)$ and $\mathcal{F}_s(\boldsymbol{\theta}_0)$, $\mathcal{F}_s(\boldsymbol{\theta}_1)$ is equal to the probability that d and d_s have different signs. This can be visualized by the intersecting Integrals of the probability density functions (Fig. 2 right). To calculate the integral, it is convenient to rewrite d_s as $d_s = d + \Delta_s$ with $\Delta_s \sim \mathcal{N}(0, 2\sigma_s^2)$. With this we can write

$$
\begin{aligned}
p &= P(|\Delta_s| > |d|)P(\mathrm{sign}(\Delta_s) \neq \mathrm{sign}(d)) && (4)\\
&= \frac{1}{2} P\left(\frac{|\Delta_s|}{|d|} > 1\right) && (5)\\
&= \frac{1}{2} \int_1^\infty \mathcal{N}\left(0, \frac{\sigma_s^2}{\|\sigma^2 C\|^2_{\mathcal{F}}}\right) , && (6)
\end{aligned}
$$

where $P(A)$ denotes the probability of the event A to occur. Although the integral in equation 6 cannot be calculated directly, known discrete evaluations (e.g., σ-bounds) of the distribution integral can be used as upper limits for an estimation (e.g. $p < 0.01$). The 2σ-bound of the normal distribution yields a probability of about 0.03. Using

$$0.03 \approx 1 - \int_{-2}^{2} \mathcal{N}(0,1) = 2 \int_{0}^{\infty} \mathcal{N}(0,1), \tag{7}$$

the requested probability $p < 0.01$ is given if $\sigma_s^2 \|\sigma^2 C\|_{\mathcal{F}}^{-2} = \frac{1}{2}$. During the optimization $\|\sigma^2 C\|_{\mathcal{F}}^2$ is provided by the variance of the calculated fitness values in every iteration, therefore to suffice $p < 0.01$, s should be choose such that σ_s^2 satisfies

$$\sigma_s^2 = \frac{\|\sigma^2 C\|_{\mathcal{F}}^2}{2}\,. \tag{8}$$

For CMA-ES to work properly, more than two individuals are required. To account for that, the probability of an order change has to be evaluated for every pair of individuals, except for the combinations in which both individuals are weighted zero. As the probabilities are equal for each combination of individuals, the extensions to more than two individuals can be done by making use of the number of pairwise comparisons

$$n = \left(\sum_{j=1}^{\lambda-1} j - \sum_{j=1}^{\mu-1} j \right) = \sum_{j=\mu}^{\lambda-1} j, \tag{9}$$

which leads to an overall probability of

$$p_n = \sum_{k=0}^{n-1} (1-p)^k p^k, \tag{10}$$

that can be approximated by $p_n \approx np$ for small p. For example with a population of size eight, the probability of an order change is approximately 6% when choosing σ_s^2 according to (8). To change this probability, the σ-bounds need to be chosen accordingly.

The relationship between s and σ_s^2 depends on the input images and therefore is very complex. Since evaluating σ_s^2 by discrete probes in every iteration to estimate s is computational very expensive, we developed a heuristic to adapt s to the current requirement of the optimization by choosing s such that $s \propto \frac{1}{\sigma}$. In the next section we show that we can obtain a speedup of eight and higher using this heuristic as compared to a static sparse synthesis.

4 Evaluation

To analyze the potential of adaptive sparsification, practical tests have been performed using a publicly available 313 frame RGB-D input sequence[2] with

[2] RGB-D sequence available at http://cvlab.epfl.ch/data/dsr

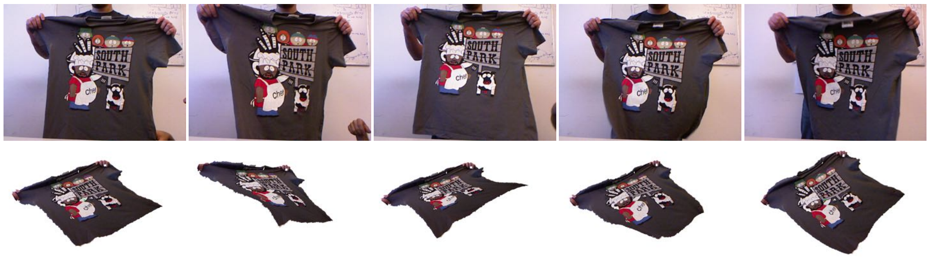

Fig. 3. Color input data of the deformation tracking method (top row) along with the deformed referenced model from a rotated point of view (bottom row).

ground-truth data (see Fig. 3). The input sequence comes with a set of a few hundreds 3D-to-2D correspondences between points inside a reference mesh and the input image for each frame. By deforming the reference mesh using the deformation function resulting from the tracking and by using the 2D image coordinates to obtain the true camera measurements, the absolute tracking error in mm can be computed. A NURBS-based deformation function was used as described in [14] such that the parameter space of $\mathcal{D}$ contains the control point positions in mm.

4.1 Induced Inaccuracy

When the adaptive sparsification is applied in practice, the rate of induced order changes is not necessarily a good way to analyze the possible benefits, also because low order change probabilities can only be obtained for high σ or a large number of individuals, which in turn increases the workload.

The far more interesting question is, to what extend does a sparsification induced order change influence the overall optimization process. Figure 4 (left) depicts the fitness values during the optimization, performed using different sparsification rates, averaged over all frames of the input sequence. It shows that even high sparsification rates (i.e. small s) do not influence the optimization process in the early and middle stages much. However, in the final optimization process the lack of accuracy clearly prevents the sparse fitness function from reaching a precise optimization goal (see Fig. 4 left, zoomed area).

The benefit of sparsification becomes more visible when the optimization process is plotted over equivalent computational work rather than iteration counts. As a fitness function evaluation of $\mathcal{F}_{0.01}$ only takes 1% of the computational work compared to $\mathcal{F}$, this sparse optimization can perform 100 iterations in the same time as one full synthesis iteration is performed. Figure 4 (right) depicts the same data but plotted over actual computational work, equivalent to one iteration with the full vertex set. Hence, to exploit high precision and efficient evaluation, the sparsification rate should be adapted during the optimization process. Equation (8) motivated the linear coupling of the sparsification rate s to σ. Figure 4 (dashed orange) shows the fitness function for the case that

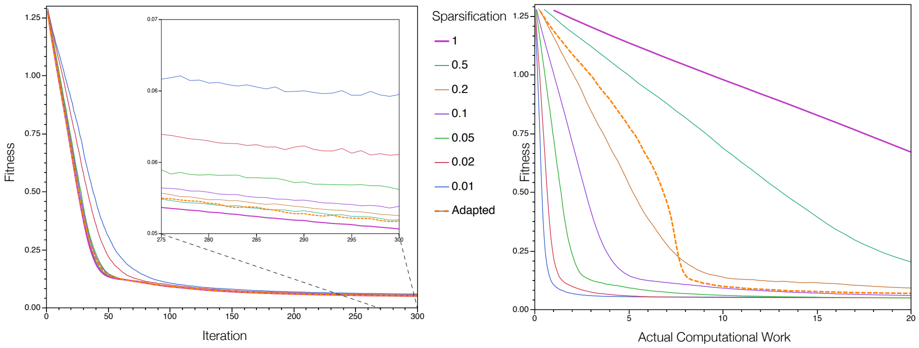

Fig. 4. The optimization process averaged over the complete 313 frame sequence depicted in Fig. 3. Left: the fitness value for the sparse vertex set during the optimization for various sparsification rates, plotted over the number of iterations. The last 25 iterations are zoomed to better visualize the differences between the sparsification rates at the late optimization stage. Right: the same fitness values as on the left but plotted over the actual computational work instead of iterations.

the sparsification rate is chosen such that $s \propto \frac{1}{\sigma}$. This online adaptation of the sparsification rate s provides an optimization result as accurate as the synthesis using the full vertex set, but using less than 10% of the computational work.

Besides from lowering the number of vertices that are evaluated in each iteration, low sparsification rates also act as a damping factor for the search range σ: In each iteration a new subset of vertices is randomly chosen. Therefore, if CMA-ES discovers a local minimum of the fitness function in one iteration, it might not be a minimum in the following iteration, where a different subset is evaluated. If in one iteration of CMA-ES two or more individuals with non-neglectable distances in the parameter space have equally low fitness values, the population search range σ increases. Therefore, an increasing σ indicates that the optimization process hit several local minima, whereas a decreasing σ indicates that it converges into a single minimum. Figure 5 (right) depicts the progress of σ over the optimization process for various sparsification rates, averaged over all frames of the input sequence. The search range rapidly increases in the first optimization stage for all sparsification rates and then converges until the end of the optimization. The peak for low sparsification rates is significantly lower than for the higher ones, resulting in a narrower search space. As a consequence the global minimum is not found and therefore the optimization process becomes less accurate (cf. Fig. 4 left). For the adaptive sparsification this is not the case. The coupling $s \propto \frac{1}{\sigma}$ prevents the optimization to converge prematurely to a local minimum.

4.2 Results

Figure 4 (right) already shows the effectiveness of adaptive sparsification during the optimization on real data. Figure 5 (left) illustrates the tracking error

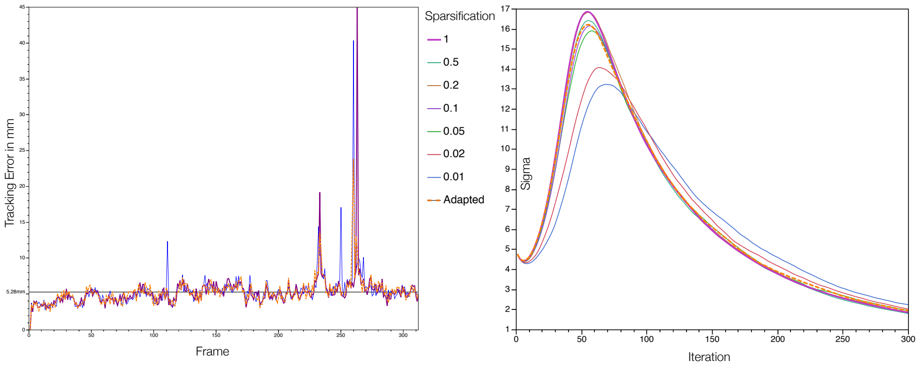

Fig. 5. Left: Tracking error for each frame in the input sequence in mm for 100%, 1% and adaptive sparsification. Right: σ_s during the optimization process, averaged over all frames, depicted for various sparsification rates.

for each frame of the input sequence, for the full vertex set synthesis, for a 1% sparsification and for the adaptive sparsification. In average all three variants perform equally well: (full vertex set: $5.23mm \pm 2.67mm$, 1% sparsification: $5.37mm \pm 2.55mm$, adaptive sparsification: $5.28mm \pm 1.83mm$), but the adaptive sparsification has the lowest standard deviation. To compare our results with the approach in [27], we also computed the depth and the 2D-displacement[3] errors separately[4]. The mean depth errors are $3.38mm \pm 0.091$, $3.48mm \pm 1.03mm$ and $3.42mm \pm 0.91mm$, for the full vertex set, for the 1% sparsification and for the adaptive sparsification, respectively. The 2D-displacement errors are $3.28mm \pm 2.63mm$, $3.35mm \pm 2.40mm$ and $3.29mm \pm 1.71mm$. As a comparison[5] Xu et al. in [27] achieved higher mean depth and displacement errors of over $5mm$.

5 Conclusion

This article introduced a novel approach to increase the effectiveness of the sparse synthesis used in AbS deformation tracking. The method uses random adaptive sparsification to adjust the number of reference model vertices to the optimization status of the tracking algorithm. By coupling the sparsification rate to the distribution size, the accuracy of the synthesis is adapted to the current requirement of the optimizer. Tests on publicly available real data have been presented, showing large speed up capabilities (factor 8 and more) while maintaining the accuracy of a full synthesis and also increasing the robustness.

[3] Called registration error in [27].

[4] Plots are available as supplementary materials.

[5] Since there were no definite numbers in [27] available, we roughly estimated a lower bound based on the given figures 5 and 6.

References

1. Auger, A., Brockhoff, D., Hansen, N.: Benchmarking the (1,4)-cma-es with mirrored sampling and sequential selection on the noisy bbob-2010 testbed. the 12th annual conference comp p. 1625–1632 (Jul 2010)
2. Besl, P.J., McKay, N.D.: A method for registration of 3-d shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(22), 239–256 (Feb 1992)
3. Bregler, C., Hertzmann, A., Biermann, H.: Recovering non-rigid 3d shape from image streams. CVPR 2, 2690–2696 (Jan 2000)
4. Cagniart, C., Boyer, E., Ilic, S.: Iterative mesh deformation for dense surface tracking. 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops p. 1465–1472 (Jan 2009)
5. Del Bue, A., Agapito, L.: Non-rigid stereo factorization. International Journal of Computer Vision 66(22), 193–207 (Jan 2006)
6. Fayad, J.a., de Agapito, L., Del Bue, A.: Piecewise quadratic reconstruction of non-rigid surfaces from monocular sequences. ECCV 6314, 297–310 (Jan 2010)
7. Gargallo, P., Prados, E., Sturm, P.: Minimizing the Reprojection Error in Surface Reconstruction from Images. In: 2007 IEEE 11th International Conference on Computer Vision. pp. 1–8. IEEE (2007)
8. Hansen, N.: The cma evolution strategy: A comparing review. Towards a New Evolutionary Computation 192, 75–102 (Jan 2006)
9. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation () 9(22), 159–195 (Jan 2001)
10. Helten, T., Baak, A., Bharaj, G., Müller, M., Seidel, H.P., Theobalt, C.: Personalization and evaluation of a real-time depth-based full body tracker. 3DV p. 279–286 (Jan 2013)
11. Innmann, M., Zollhöfer, M., Nieß ner, M., Theobalt, C., Stamminger, M.: Volumedeform: Real-time volumetric non-rigid reconstruction. CoRR abs/1506.02753 cs.CV (Jan 2016)
12. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R.A., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A.J., et al.: Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. UIST p. 559–568 (Jan 2011)
13. Jordt, A., Koch, R.: Fast tracking of deformable objects in depth and colour video. BMVC p. 1–11 (Jan 2011)
14. Jordt, A., Koch, R.: Direct model-based tracking of 3d object deformations in depth and color video. International Journal of Computer Vision 102(11), 239–255 (Sep 2012)
15. Jordt, A., Koch, R.: Reconstruction of Deformation from Depth and Color Video with Explicit Noise Models. Time-of-Flight and Depth Imaging 8200(Chapter 7), 128–146 (2013)
16. Jordt, A., Koch, R.: Reconstruction of deformation from depth and color video with explicit noise models. Time-of-Flight and Depth Imaging 8200, 128–146 (Jan 2013)
17. Kambhamettu, C., Goldgof, D., He, M., Laskov, P.: 3d nonrigid motion analysis under small deformations. Image and Vision Computing 21(33), 229–245 (Mar 2003)
18. Li, H., Adams, B., Guibas, L.J., Pauly, M.: Robust single-view geometry and motion reconstruction. ACM Trans. Graph. () :1-175:10 28(55), 175 (Jan 2009)

19. Qian, C., 0001, X.S., Wei, Y., Tang, X., 0001, J.S.: Realtime and robust hand tracking from depth. CVPR p. 1106–1113 (Jan 2014)
20. Rosenhahn, B., Kersting, U.G., Powell, K., Klette, R., Klette, G., Seidel, H.P.: A system for articulated tracking incorporating a clothing model. Mach. Vis. Appl. () 18(11), 25–40 (Jan 2007)
21. Salzmann, M., Hartley, R.I., Fua, P.: Convex optimization for deformable surface 3-d tracking. ICCV p. 1–8 (Jan 2007)
22. Sharp, T., Keskin, C., Robertson, D., Taylor, J., Shotton, J.: Accurate, robust, and flexible real-time hand tracking. Proc CHI (Jan 2015)
23. Shotton, J., Fitzgibbon, A.W., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., 0001, A.B.: Real-time human pose recognition in parts from single depth images. CVPR p. 1297–1304 (Jan 2011)
24. Steimle, J., Jordt, A., Maes, P.: Flexpad: Highly flexible bending interactions for projected handheld displays. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems p. 237–246 (Jan 2013)
25. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: a factorization method. International Journal of Computer Vision 9(22), 137–154 (Jan 1992)
26. Torresani, L., Hertzmann, A.: Learning non-rigid 3d shape from 2d motion. Advances in Neural . . . (Jan 2003)
27. Xu, W., Salzmann, M., Wang, Y., Liu, Y.: Nonrigid surface registration and completion from rgbd images. Computer Vision –ECCV 2014 8690, 64–79 (Jan 2014)
28. Zollhöfer, M., Theobalt, C., Stamminger, M., Nieß ner, M., Izadi, S., Rehmann, C., Zach, C., Fisher, M., Wu, C., Fitzgibbon, A., et al.: Real-time non-rigid reconstruction using an rgb-d camera. ACM Transactions on Graphics 33(44), 1–12 (Jul 2014)