Zeroth-Order Optimizer Benchmarking for 3D Performance Capture

Alexandros Doumanoglou*, Petros Drakoulis, Kyriaki Christaki, Nikolaos Zioulis, Vladimiros Sterzentsenko, Antonis Karakottas, Dimitrios Zarpalas, Petros Daras

Abstract
In the field of 3D Human Performance Capture, a high quality 3D scan of the performer is rigged and skinned to an animatable 3D template mesh that is subsequently fitted to the captured performance's RGB-D data. Template fitting is accomplished via solving for the template's pose parameters that better explain the performance data at each recorded frame. In this paper, we challenge open implementations of zeroth-order optimizers to solve the template fitting problem in a human performance capture dataset. The objective function that we employ approximates, the otherwise costly to evaluate, 3D RMS hausdorff distance between the animated template and the 3D mesh reconstructed from the depth data (target mesh) at an individual recorded frame. We distinguish and benchmark the optimizers, in three different real-world scenarios, two of which are based on the geometric proximity of the template to the target in individual frames, while in the third one we fit the template sequentially to all target frames of the recorded sequence. Conclusions of this work can serve as a reference for future optimizer implementations and our findings can server as a baseline for future multi-objective optimization approaches. To facilitate this, we make part of our benchmark and experiment setup publicly available (`https://github.com/VCL3D/nevergrad`, `https://github.com/VCL3D/PerformanceCapture/releases/`).

Keywords
Benchmarking — Open Source — Optimization — Performance Capture — Nevergrad

Centre for Research and Technology HELLAS, Information Technologies Institute
***Corresponding author**: aldoum@it.gr

Contents

1 **Introduction** 1
2 **Algorithms** 2
3 **Objective Function** 3
4 **Experimental Setup** 4
4.1 Local fitting 6
4.2 Global fitting 6
4.3 Sequence fitting 6
5 **Results and Analysis** 7
5.1 Local fitting 7
5.2 Global fitting 8
5.3 Sequence fitting 9
5.4 Supplementary material 9
6 **Conclusion** 9
7 **Acknowledgement** 9
References 9

1. Introduction

Performance capture refers to the technology that enables the digitization of human performances with the goal of re-using them in various contexts like films and video games. For capturing the human performance, lower cost marker-less multi-view performance capture systems, approach this challenge with minimal restrictions (*i.e.* no markers) often employing color and/or depth (RGB+D) commodity sensors at sparse placement configurations. In a special-case of performance capture systems, a pre-digitized articulated (animatable) template $3D$ mesh of the performer is fit into processed data coming from the sensing devices, acquired during the human performance capture.

While any performance may be naively digitized by constructing separately a $3D$ mesh per each-frame of the captured performance, fitting a template mesh to the captured data has two major benefits: **a)** the quality of the template $3D$ mesh which is acquired by capturing a human in a T or A-like pose is free of self-occlusions and thus more faithful to the overall human's geometry. Contrariwise, constructing a mesh of a human standing an arbitrary pose may lack proper details due to self-occlusions. **b)** Exporting the digitized performance as an asset that can be natively integrated to $3D$ authoring tools like Blender, Maya, 3D Studio Max and Game Engines like Unity or Unreal becomes a lot easier, due to leveraging the skinning/articulation properties of the template mesh.

Template fitting is typically achieved by solving for the pose parameters of the template mesh which better explain the captured performance data. The quality of the fit is directly or indirectly related to the quality of the template mesh geometry, the errors introduced in its articulation process and the depth quality of the sensors used to capture the performance. In the case when the template mesh is acquired by a low-cost capturing setup, one more challenge arises, due to the noise inherently found in the depth-maps provided by the depth sensors, manifesting in both the template mesh (noisy template) and the captured performance data (noisy data). Furthermore, despite proper pre-processing, the automatic articulation process of a template mesh captured by a low-cost setup is typically prone to introducing additional errors in the location of the joints and consequently to the mesh's skinning weights. This is because common articulation (i.e. skinning) algorithms like [1] are usually designed to work on $3D$ meshes authored by artists in specialized mesh authoring tools (i.e. noise-free) and not on $3D$ meshes coming from sensing devices (i.e. noisy).

Most existing approaches [2, 3, 4] solving the template fitting problem in 3D performance capture, rely on gradient-based solvers and objective functions formulated by landmark correspondence establishment between the articulated template and pre-processed sensor data. Further, template fitting is either processed frame-by-frame, where the solution to the last frame becomes the initial point of the next frame's optimization, or (post-)processed globally in the temporal domain, by almost always relying on a small-motion prior assumption. Despite its general success, landmark correspondence establishment often fails in the case of fast subject motion, directly affecting the quality of the solution obtained by the gradient-based solvers. Apart from this potential failure, the previously mentioned strategy of utilizing gradient-based solvers on par with landmark correspondence establishment algorithms, cannot be used in the case where the template mesh, at its initial pose, is geometrically distant from the first frame of the captured performance, again due to the violation of the small motion assumption prior.

Solving the problem of template mesh fitting to performance captured data in its generic case (i.e. dropping the small motion assumption prior) requires excluding any strategy relying on non-robust landmark correspondence-establishment. The most straight-forward approach is to employ an objective function which measures the geometric distance between the template at any arbitrary pose and the data of the performance. However, depending on the case, such an objective function may be either non differentiable or highly non-linear (i.e. suffer from multiple local minima), with respect to the template's pose parameters. Thus, in theory, solving for such an objective should be suitable for zeroth-order optimization algorithms.

In this paper, we challenge popular, publicly available, implementations [5] of zeroth-order optimizers to solve the real-world problem of noisy template mesh fitting to noisy performance capture data in three different settings. Fitting the template mesh to performance capture data with: **a)** a small geometric distance, essentially covering the small to medium subject motion case (*local fitting*), **b)** a larger geometric distance, covering the case of larger movements or initial template fit to the first frame of the captured performance (*global fitting*) and **c)** fitting the template mesh sequentially to all frames of a captured performance (*sequence fitting*). The performance capture system that we used captures a noisy point cloud of the performance at each time-frame by spatially fusing [6] temporally synchronized multi-view depth maps [7] [1]. The template mesh is reconstructed at an initial frame of the sequence, using Screen Poisson Surface Reconstruction [8]. The objective function that we use consists of a weighted combination of $3D$, projective $2D$ and pose prior terms. Quantitative evaluation of the template fit is done using RMS Hausdorff distance between the template mesh at the converged pose and a mesh that is created from the performance capture data at the given frame.

Apart from the study and the analysis of the benchmarked algorithms in the three previously mentioned settings, we release a public version of our performance capture benchmark server [2] (serving objective function evaluations at any arbitrary pose of the template mesh), the dataset [3] that we used in this paper and an open source performance capture benchmark Python client, with the latter integrated to the Nevergrad [5] platform [4].

2. Algorithms

In this benchmark we choose to assess 9 open-source, widely used optimization algorithms available in the Nevergrad toolbox. The algorithm selection was based on their popularity (*i.e.* their established position in the literature) and their performance on the template fitting task during some preliminary empirical evaluation, with the most popular and successful optimizers finally included in the study. Additionally, in the *local fitting* case of our experimentation, which covers the small to medium subject motion, we've chosen to include the Random Search (RS) [9] algorithm to serve as an additional baseline. In detail, our benchmark comprises the following algorithms:

1. Differential Evolution (DE) [10], which generates new candidate solutions by combining existing ones over a maintained population of candidates.

2. Real-Space Particle Swarm Optimization (RS-PSO) [11], which mimics the well-informed swarming behavior of social species.

[1] `https://github.com/VCL3D/VolumetricCapture`
[2] `https://github.com/VCL3D/PerformanceCapture/releases/tag/1.0`
[3] `https://github.com/VCL3D/PerformanceCapture/releases/tag/dataset_1.0`
[4] `https://github.com/VCL3D/nevergrad`

3. Discrete One Plus One (Disc 1+1) Evolution Strategy [12] , which constitutes an improvement over the Random Search algorithm with adaptive step size. At each iteration, One Plus One, in its Discrete version, mutates only one variable on average.

4. Covariance Matrix Adaptation (CMA) Evolution Strategy [13], which adapts the mutation distribution of each generation based on the evaluation of the previous one.

5. Test-Based Population-Size Adaptation (TBPSA) [14], an algorithm designed for noisy optimization applications which increases the population size when noise related behaviour is detected.

6. Powell [15], which minimizes the objective function by tweaking one parameter at a time.

7. A chain combination of CMA and Powell algorithms, (CMA+Powel)

8. NGO [5]

9. Shiwa [16]

3. Objective Function

Fitting the template mesh to a specific frame of the performance is done via optimizing its pose parameters and is driven by an objective function that measures alignment between the animatable template and the performance data.

We optimize for the template's pose parameters $\mathbf{p} := \{\mathbf{t} \in \mathbb{R}^3,\ \theta \in \mathbb{R}^{J\times 3}\}$, with $\mathbf{t}$ the translation of the root joint, and θ the rotation parameters of all J joints, including root. These are expressed using the exponential map rotation parameterization [17] and used to animate the template model through dual quaternion skinning [18], after estimating the per vertex skinning weights using [1].

When fitting the template to a target frame of the performance, an implicit surface of the captured performance's point cloud data is estimated using a parallel GPU implementation [19] of [20] and $3D$ reconstructed into a watertight $3D$ mesh using the Marching Cubes algorithm [21]. Apart from the watertight trait that reduces the subsequent calculation complexity (as the number of vertices is greatly reduced compared to the point cloud size), the process also helps to filter out noise as it implicitly acts as a spatial low-pass filter.

Let $\mathcal{M}_T$ denote the animatable template mesh at its neutral pose and $\mathcal{M}_P$ denote the mesh created from the performance's point cloud data. Let also $\mathcal{A}_\mathbf{p}$ denote the animated template mesh at pose $\mathbf{p}$ (*i.e* $\mathcal{M}_T$ with all its vertices and normals transformed based on pose parameters $\mathbf{p}$). In order to capture the user's performance, we seek to solve for $\mathbf{p}$ that minimizes the geometric distance between $\mathcal{A}_\mathbf{p}$ and $\mathcal{M}_P$ at each frame. Let $dist(\mathcal{M}_A,\mathcal{M}_B)$ denote the Root Mean Squared Error (RMSE) distance from mesh $\mathcal{M}_A$ to mesh $\mathcal{M}_B$. Similar to [22], we define:

$$dist(\mathcal{M}_A,\mathcal{M}_B) = \sqrt{\frac{1}{N}\sum_{\mathbf{v}\in\mathcal{M}_A}\min_{\mathbf{u}\in\mathcal{M}_B}||\mathbf{v}-\mathbf{u}||_2^2} \tag{1}$$

with N denoting the number of vertices in $\mathcal{M}_A$ and $\mathbf{v}$, $\mathbf{u}$ denoting the vertices of $\mathcal{M}_A$ and $\mathcal{M}_B$, respectively. Strictly speaking, fitting $\mathcal{A}_\mathbf{p}$ to $\mathcal{M}_P$, requires minimizing the Hausdorff-like RMSE distance between $\mathcal{A}_\mathbf{p}$ and $\mathcal{M}_P$ which is defined as

$$H(\mathcal{A}_\mathbf{p},\mathcal{M}_P) = \max(dist(\mathcal{A}_\mathbf{p},\mathcal{M}_P), dist(\mathcal{M}_P,\mathcal{A}_\mathbf{p})) \tag{2}$$

Computing $dist(\mathcal{A}_\mathbf{p},\mathcal{M}_P)$ for an arbitrary $\mathbf{p}$ can be accelerated by building the Squared Euclidean Distance Transform (DT) of $\mathcal{M}_P$ once and then use the pre-computed distance field to quickly evaluate distances between $\mathcal{A}_\mathbf{p}$ and $\mathcal{M}_P$ for any $\mathbf{p}$. While constructing the DT for $\mathcal{M}_P$ once can serve all subsequent $dist(\mathcal{A}_\mathbf{p},\mathcal{M}_P)$ queries for any $\mathbf{p}$, adopting the same strategy for $\mathcal{A}_\mathbf{p}$ is computationally expensive [23] as querying the DT of $\mathcal{A}_\mathbf{p}$ at the points of $\mathcal{M}_P$ requires re-computing the DT of the $\mathcal{A}_\mathbf{p}$ for each $\mathbf{p}$. To overcome this complexity, in the objective function that we employ, we approximate $dist(\mathcal{M}_P,\mathcal{A}_\mathbf{p})$ via a $2D$ projective error term based on a rendering operation, as in [23].

In more detail, the objective function we use comprises 5 error terms and is given below:

$$E(\mathbf{p}) = \underbrace{\overbrace{\lambda_J E_J(\mathbf{p})}^{2D} + \overbrace{\lambda_D E_D(\mathbf{p}) + \lambda_S E_S(\mathbf{p})}^{3D}}_{data\ fitting} + \underbrace{\overbrace{\lambda_P E_P(\mathbf{p}) + \lambda_A E_A(\mathbf{p})}^{prior}}_{constraints} \tag{3}$$

In (3), E_D is a direct computation of $dist(\mathcal{A}_\mathbf{p},\mathcal{M}_P)$, E_J a proxy for $dist(\mathcal{M}_P,\mathcal{A}_\mathbf{p})$ through $2D$ rendered projections, E_S a surface alignment term based on surface normals that has been found to help overcome local minima, while E_P and E_A are prior terms penalizing template self intersections and irregular human poses. The error terms of our objective function are chosen such that, under proper conditions, minimizing (2) also minimizes (3).

Following our previous discussion, computation of E_D relies on the Squared Euclidean Distance Transform (DT) $\mathcal{S}_\mathbf{P}(\mathbf{u}) \in \mathbb{R}$ of $\mathcal{M}_P$ with $\mathbf{u} \in \mathbb{R}^3$ a sampling position within the voxel grid that $\mathcal{S}$ is defined in. Its domain's spatial extent is dictated by a slightly scaled up bounding box of the performance's point cloud. The same bounding box is used to $3D$ reconstruct $\mathcal{M}_P$. For the implementation of DT, we use a custom parallel-friendly GPU algorithm based on the separable algorithm of [24]. The $3D$ distance error is thus defined as:

$$E_D(\mathbf{p}) = \sqrt{\frac{1}{V_A}\sum_{\mathbf{v}\in\mathcal{A}_\mathbf{p}}\mathcal{S}_\mathbf{P}(\lfloor\mathbf{v}\rceil) + ||\mathbf{v}-\lfloor\mathbf{v}\rceil||_2^2} \tag{4}$$

with $\lfloor\cdot\rceil$ being a clamping function bounding the spatial vertex coordinates to the bounding box and V_A the total number of vertices of the animated mesh. The sampling locations $\mathbf{v}$ are the animated vertex positions of the template's vertices as articulated by the optimized pose parameters $\mathbf{p}$. Apart from averaging the samples of the squared distance transform, we also add an approximate extrapolation term $||\mathbf{v} - \lfloor\mathbf{v}\rfloor||_2^2$ to extend the domain of E_D outside the domain of $\mathcal{S}$. A constant or clamped error at out of boundary cases would hinder the convergence of zeroth order optimizers.

As already mentioned previously, in order to aid the optimizers to converge towards the global optimum, we additionally use a surface alignment error term E_S that measures the surface orientation distance between $\mathcal{A}_\mathbf{p}$ and $\mathcal{M}_P$ and is an adaptation of a similar term found in [25]:

$$E_S(\mathbf{p}) = \frac{1}{V_A} \sum_{(\mathbf{v},\mathbf{n}) \in \mathcal{A}_\mathbf{p}} 1 - \langle \nabla \mathcal{S}_\mathbf{P}(\lfloor\mathbf{v}\rfloor), \mathbf{n} \rangle^2, \tag{5}$$

where $\mathbf{n}$ is the normal of $\mathbf{v} \in \mathcal{A}_\mathbf{p}$ which is also rotated based on $\mathbf{p}$. This error term is similarly constrained in the voxel grid that $\mathcal{S}$ is defined in, with out-of-bounds sampling returning the maximum error value (1.0).

These $3D$ error terms suffer from many local minima and smooth error landscape valleys due to the closest surface distance formulation and the high degrees of freedom that the articulation entails. A common issue is fitting limbs of $\mathcal{A}_\mathbf{p}$ close to the torso, whereas the actual limb location of $\mathcal{M}_P$ may be far away (*i.e.* raised arms). This mainly stems from the fact that minimizing $dist(\mathcal{A}_\mathbf{p}, \mathcal{M}_P)$, which is represented by E_D, does not necessarily minimize $dist(\mathcal{M}_P, \mathcal{A}_\mathbf{p})$.

To overcome the previous limitation, the use of a symmetric Hausdorff-like metric would be ideal. To avoid the otherwise costly function evaluations of the Hausdorff-like metric, we utilize a projective $2D$ silhouette error to account for the distance from $\mathcal{M}_P$ to $\mathcal{A}_\mathbf{p}$. This is done by picking k virtual viewpoints which we align in position and orientation with the k viewpoints of the multi-view performance capture system and extracting the silhouette masks for each one of the $\mathcal{M}_P$ and $\mathcal{A}_\mathbf{p}$ via a rendering function $\mathcal{R}_k$, producing $\mathbf{M}_k^P, \mathbf{M}_k^A \in \Omega, k \in [1, K]$, with K being the number of viewpoints, $\Omega \in \mathbb{R}^{W \times H}$ the image domain of width W and height H, and the superscripts denoting correspondence with the meshes $\mathcal{M}_P$ and the animated template $\mathcal{A}_\mathbf{p}$. As the geometry of the animated mesh is a function of the pose parameters $\mathbf{p}$, so are the masks $\mathbf{M}_k^A = \mathcal{R}_k(\mathcal{A}_\mathbf{p})$. The $2D$ error term relies on the Jaccard index [26] (or otherwise *intersection-over-union*) and is formulated as:

$$E_J(\mathbf{p}) = \frac{1}{K} \sum_{k}^{K} 1 - \frac{\mathbf{M}_k^P \cap \mathbf{M}_k^A}{\mathbf{M}_k^P \cup \mathbf{M}_k^A}, \tag{6}$$

defined on the image domain Ω. While this error term implicitly captures $dist(\mathcal{M}_P, \mathcal{A}_\mathbf{p})$, its landscape lacks adequate granularity and $3D$ accuracy due to being projective. This is partly addressed by the averaging across the K viewpoints, but with sparse configurations (small K) this countering effect can be unreliable and camera-pose dependent.

In addition to the data-fitting terms, a set of constraints are added to respect our prior knowledge about human articulation and posing. These comprise a prior E_A for the set of plausible human poses and a self-penetration term E_P in order to prevent the interpenetration of different body parts of the animated template. The former is defined based on the anthropomorphic constraints of the human body with respect to semantic inter-joint angles (*i.e.* flexion, abduction, etc.):

$$E_A(\mathbf{p}) = \sum_{j \in \mathbf{J}} \alpha_j(\theta_j), \tag{7}$$

where $\mathbf{J}$ is a subset of the joints that will be considered, and α_j is a per joint binary function that tests the swing and twist components of each joint's rotation parameters θ_j for lying inside the valid semantic angle ranges.

Finally, E_P is based on a coarse proxy template constructed by fitting $3D$ ellipsoids on each body segment of the template mesh, as segmented by the most significant skinning coefficient. Then, each animated vertex is tested for intersection with the remaining segments, providing the penetration error term:

$$E_P(\mathbf{p}) = \sum_{\mathbf{v} \in \mathcal{A}_\mathbf{p}} \sum_{j \in \mathbf{J}} \rho_j(\mathbf{v}), \tag{8}$$

where ρ_j is the intersecting function between a vertex $\mathbf{v}$ and the $3D$ ellipsoid of joint (*i.e.* bone) j. This error term prevents articulations that result in non-plausible meshes, acting as the second prior constraint of our objective function. Varying the error term weights $\lambda \in \mathbb{R}^+$ yields a plethora of complex error landscapes. In our experiments though, we have fixed all λ coefficients to specific values based on empirical evaluation, while a short ablation on λ_J is given in supplementary material.

4. Experimental Setup

Software. To conduct this benchmark we developed two software components. The first one, the *Performance Capture Application* (PCA), is the core component able to load RGB+D multi-view recordings, captured using the volumetric capture platform of [7], and integrates workflows for the construction, skinning and animation of $\mathcal{M}_T$, as well as per-frame creation of $\mathcal{M}_P$. For template fitting, PCA incorporates its own optimization engine pipeline based on optimizer "ask"-"tell" interface. Given a specific optimizer instance, the optimization engine first "asks" the optimizer for the next function evaluation query point $\mathbf{p}$, animates $\mathcal{M}_T$ to produce $\mathcal{A}_\mathbf{p}$ (based on $\mathbf{p}$), evaluates the objective function (3) and "tells" the result back to the optimizer, before starting the next "ask-tell" iteration cycle. PCA is written in C++, utilizes NVIDIA CUDA for accelerating the evaluation of the error terms of (3) and OpenGL to accelerated the renderings of $\mathbf{M}_k^P$ and $\mathbf{M}_k^A$. To utilize the implementation of the zero-th order optimizers found in Nevergrad [5] a second component was written in Python, namely the "Nevergrad Optimization

Service" (NGOService) which basically constitutes a wrapper around Nevegrad's Application Programming Interface (API), exposed with a messaging interface over RabbitMQ [27] for inter-process communication with PCA.

Dataset: The data sequences used in this study consist of 12 distinct in-lab captured performances of volunteers in a variety of poses and scenarios (athletic, narration, performance) that are turned into point clouds in a way described in Section 1. The sequences are acquired using a camera setup of 4-6 commodity RGB-D sensors. All performances start with the subject standing in an A-like pose. The 3D animatable template mesh is reconstructed at a selected frame, typically the initial one, using Poisson Surface Reconstruction [8] and rigged and skinned using [28]. The result of the animatable template creation process is the embedding of a skeleton with 18 joints to the mesh's 3D geometry, accompanied with per vertex skinning weights associating vertices with the respective skeleton joints. The names of all skeleton joints are: *Root, Middle Torso, Lower Torso, Head, R.Hip, R. Knee, R. Ankle, R. Foot, R. Shoulder, R. Elbow, R. Hand, L.Hip, L. Knee, L. Ankle, L. Foot, L. Shoulder, L. Elbow, L. Hand*, with *R* and *L* denoting right and left, respectively. See Fig. 1c for a visualization of template mesh articulation.

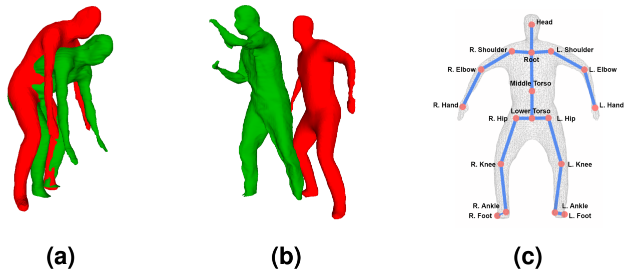

Figure 1. Experimental Setup of the local fitting case (a), and the global fitting case (b). The initial pose of the template mesh is depicted in red, while the target pose of the live mesh is depicted in green. The parameterization of the articulation is also presented (c).

Experiment definition. As an experiment we define the problem of fitting the subject's template 3D mesh to a specific target frame of a captured performance (for *local* and *global fitting*), or to a sequence of target frames in *sequence fitting*. Each experiment instance is associated with a subset of the sensed 3D data (either a single frame or a sequence of frames) captured during the performance recording of a single human and a constructed animatable template of the performer ($\mathcal{M}_T$). Furthermore, an experiment is bound to a specific objective function parameterization and to a specific optimizer under a pre-defined budget. Objective function parameterization consists of defining a subset of template's animation variables to optimize against and specifying weights of individual error terms (i.e $\lambda_J, \lambda_D, \lambda_S, \lambda_P, \lambda_A$). Further, each experiment has a predefined variable mutation variance configuration (which – the mutation variance – is also mutable itself), depending on whether it is a *local fitting*, *global fitting*, or *sequential fitting* experiment, with *local* and *sequential fitting* sharing lower mutation variance values compared to *global fitting*. Moreover, each experiment's variable is given specific bounds, effectively reducing the search space for *local fitting* and *sequential fitting*, while keeping it more wide for *global fitting*. Finally, any optimizer-specific parameter values were left to the Nevergrad's defaults.

Optimization parameters. Based on the exponential map parameterization for the joint rotation parameters (i.e variables), which consists of 3 degrees of freedom (DOFs), and accounting 3 extra DOFs for the global translation of the mesh, the dimensionality of the template fitting problem is 57 (i.e $3 \cdot 18 + 3$). However, to slightly simplify the problem without loosing any important accuracy we exclude all 3 DOFs of *R.Foot* and *L.Foot* and 2 DOFs from each one of *L.Ankle*, *R.Ankle*, *R.Hand*, and *L.Hand*, leading to a problem dimensionality of 43. All joints' rotation parameters are expressed with respect to their *parent* joint.

Experiment cases In our benchmark we consider 3 distinctive experiment cases: **Local fitting** deals with small geometric distance between the template mesh $\mathcal{M}_T$ and the mesh reconstructed at the target frame $\mathcal{M}_P$ (see Fig. 1a). This case corresponds to small inter-frame motion, for example slow movements or *near by* frames. In this scenario, the template mesh is created from a past frame near the target frame. Local fitting is considered the least challenging task in the study, as the parameter search space is narrowed down, to better exploit the small geometric distance between $\mathcal{M}_T$ and $\mathcal{M}_P$.

Global fitting, deals with larger geometric distances between $\mathcal{M}_T$ and $\mathcal{M}_P$ (see Fig. 1b), corresponding to faster/more dynamic subject movements or frames with larger temporal difference. In *Global fitting*, the template mesh is created from a frame near the start of the captured performance where the subject stands in a *A*-like pose, while the target frame is several seconds apart. This case is considered the most demanding one, as the parameter search space is significantly larger, and it has proven to be especially challenging for the optimization algorithms to escape local minima, explore efficiently, and locate the optimal solution.

Finally, **Sequence fitting** refers to sequentially fitting the template mesh to all frames, or a selected frame span, of a captured performance. This case has the same dimensionality and constrained parameter search space as the local fitting case. In this setting, the solution to frame at time t, f_t, becomes the starting point for fitting the template to frame f_{t+1}. Ideally, in case the fitted solution for frame f_t is optimal, the geometric distance between the template $\mathcal{M}_T$ at f_t and the target mesh $\mathcal{M}_P$ at f_{t+1} is kept small with more chances for fitting to succeed in the subsequent frames. However, in case the the optimizer fails to fit the template for frame f_t due to not escaping a local minima, or a bad exploration strategy, it becomes very challenging to escape the local minimum and re-converge at later frames. As a result, the error might end up cumulative.

4.1 Local fitting

In the *local fitting* setting, we benchmark the 10 optimization algorithms mentioned in Section 2, using 3 different function evaluation budgets: 2000, 4000 and 7000, on 10 captured performances. All variable mutation variances were set to 0.1 meters for the root translation and 3°angle for the root and joints' rotations. The search space is bounded in a range of 0.8 meters for the root translation and 30°angle for the root and joints's rotations, to account for the small motion assumption in near-by frames. For the objective function term weights, we used the following empirical values: $\lambda_J = 0.1, \lambda_D = 1.0, \lambda_S = 0.01, \lambda_P = 1.0, \lambda_A = 1.0$. Finally, the total number of repetitions for each experiment was 5.

Algorithm 1: *Global fitting* Group Optimization Algorithm

```
Input: Optimization budgets $b_{G_i}$, Objective function $E$
// initialize parameter group variables to zero
$\theta_{p_{G_i}} \leftarrow \mathbf{0}, \forall i \in \{0,..,N_G - 1\}$;
$\theta \leftarrow \bigcup_{i \in \{0,1,...,N_G-1\}} \{\theta_{p_{G_i}}\}$;
for Group $G_i = 0, 1, ..., N_G - 1$ do
    // optimize for parameter group $\theta'_{p_{G_i}}$ keeping other variables constant
    $\theta'_{p_{G_i}} \leftarrow \text{optimize}_{\theta_{p_{G_i}}} E(\theta_{p_{G_i}}; \theta - \theta_{p_{G_i}})$;
    // update $\theta$ with the converged values
    $\theta \leftarrow (\theta - \theta_{p_{G_i}}) \bigcup \theta'_{p_{G_i}}$
end
return $\theta$
```

Table 1. *Global fitting:* Variable bounds for each joint's rotation (in degrees). N/A values indicate removed DOF.

Joint	Swing #1 (X)		Swing #2 (Y)		Twist (Z)	
	Min	Max	Min	Max	Min	Max
Root Rotation	-50.0	50.0	-30.0	100.0	-50.0	50.0
Middle Torso	-50.0	50.0	N/A	N/A	N/A	N/A
Down Torso	-50.0	50.0	-25.0	25.0	-50.0	50.0
Head	-25.0	25.0	-25.0	25.0	-25.0	25.0
L/R Hip	-45.0	45.0	-60.0	60.0	-100.0	100.0
L/R Knee	-45.0	45.0	-100.0	100.0	-20.0	20.0
L/R Ankle	-100.0	10.0	N/A	N/A	N/A	N/A
L/R Shoulder	-60.0	60.0	-60.0	60.0	-60.0	60.0
L/R Elbow	-90.0	90.0	-90.0	90.0	-90.0	90.0
L/R Hand	-90.0	90.0	N/A	N/A	N/A	N/A

4.2 Global fitting

In the global fitting setting, we used 8 captured performances to test the 9 optimization algorithms discussed in Section 2.

In order to facilitate the challenging global fitting task, instead of optimizing all parameters at once, which was empirically proven to fail by a far margin under considerable budget numbers, we separated the variables into $N_G = 5$ groups and optimized for the variables of each group jointly, while each group was processed sequentially. Parameter groups were chosen based on their functional proximity, mimicking the hierarchical structure of the human body. Specifically, the optimization parameters of Group 0 denoted as p_{G_0} include rotation variables for all *Root, Middle Torso, Lower Torso* and *Head* joints, as well as the variables for *Root* translation. The rest four groups consist of the extremities' parameters. p_{G_1} contains the variables corresponding to the joints of the left leg (i.e. *L.Hip, L. Knee, L. Ankle*), p_{G_2} the variables related the joints of the right leg, p_{G_3} the variables corresponding to the joints of the left arm (i.e. *L. Shoulder, L. Elbow, L. Hand*) and p_{G_4} the variables corresponding to the joints of the right arm.

The optimization process begins with the Group 0 and ends with the Group 4. We should note that, due to the hierarchical nature of the limbs' connectivity and the sequential proceeding of the groups, the p_{G_0} optimization sub-problem is the most crucial since it's failure makes it almost impossible for the rest to converge.

As in the *local fitting* case, we used 3 different function evaluation budgets, namely: 2000, 4000 and 7000. The total optimization budget B, is distributed across the parameter groups. Since the first group G_0 is the most prominent and contains a larger number of parameters to tune, it is assigned a budget b_{G_0} equal to the $3/11$ of the total budget B, with the rest of the groups assigned $2/11$ each ($b_{G_i}, i = \{1,2,3,4\}$). Pseudo-code of the group optimization strategy is provided in Algorithm 1. Further, to ease, the otherwise challenging task, we additionally removed 2 DOFs from *Middle Torso*, effectively reducing the problem dimensionality to 41. Preliminary empirical experimentation showcased that the modular group strategy along with the reduction of the problem's dimensionality to 41, as described previously, is essential to tackle the vast global search space effectively and results to better optimization outcomes. In *global fitting*, we set all term weights λ equal to the same values as in the *local fitting* case. Each experiment was repeated 5 times, as in *local fitting*, while variable bounds for translation were set to $[-1.25, 1.25]$ for all X,Y,Z axis and for rotation were fine tuned per joint as depicted in Table 1.

4.3 Sequence fitting

In the *sequence fitting* setting we conducted experiments on 6 captured performances. Since the captured performances can be quite long (in the context of frame by frame optimization) and exceed 1000 frames in length, we decided to limit the optimization frame span in most of the cases to up to 450 frames; although in some cases we conducted the sequence fitting experiments for up to 800 sequential frames. Experiments in this setting take a considerable amount of time and to accommodate that, we've chosen to benchmark the 5 top performing optimizers in the *global* and *local fitting* settings: RS-PSO, Disc 1+1, CMA, NGO and Shiwa in a fixed budget configuration of 4000 function evaluations and for a single repetition. The dimensionality of the problem, the parameter search space and the objective function term weights are all kept the same as in the *local fitting* case.

5. Results and Analysis

5.1 Local fitting

Table 2. *Local fitting:* Average Loss for experiments with the same budget and optimizer, aggregating across experiment repetitions and performances. Optimizers are sorted with respect to the average achieved loss. The relative performance (RP) column indicates relative performance with respect to the best performing optimizer for each budget.

B=2000			B=4000			B=7000		
Optimizer	**RP**	**Loss**	**Optimizer**	**RP**	**Loss**	**Optimizer**	**RP**	**Loss**
RS - PSO	+0.00%	0.03540	**RS - PSO**	+0.00%	0.02916	**CMA**	+0.00%	0.02818
Disc 1+1	+3.99%	0.03681	**Disc 1+1**	+17.54%	0.03428	**RS - PSO**	+0.51%	0.02833
NGO	+35.07%	0.04781	**CMA**	+19.24%	0.03477	**NGO**	+9.65%	0.03090
Shiwa	+38.08%	0.04888	**Shiwa**	+20.41%	0.03512	**CMA+Powell**	+12.07%	0.03158
CMA	+40.57%	0.04976	**NGO**	+22.12%	0.03561	**Shiwa**	+13.32%	0.03194
Powell	+40.97%	0.04990	**Powell**	+22.87%	0.03583	**Disc 1+1**	+13.41%	0.03196
CMA+Powell	+43.12%	0.05067	**CMA+Powell**	+34.38%	0.03919	**Powell**	+14.93%	0.03239
DE	+56.98%	0.05557	**DE**	+72.39%	0.05028	**DE**	+64.55%	0.04638
TBPSA	+86.71%	0.06610	**TBPSA**	+105.06%	0.05980	**TBPSA**	+87.89%	0.05295
RS	+123.95%	0.07928	**RS**	+159.60%	0.07571	**RS**	+161.56%	0.07372

Table 3. Wilcoxon test for the *Local fitting case*, at a confidence level of 0.05. When the optimizer at cell's row performs better (worse/equal) than the optimizer at cell's column, the cell is marked with '+' ('-'/'≈').

	B = 2000									
	CMA+Powell	**CMA**	**DE**	**Disc 1+1**	**NGO**	**Powell**	**RS**	**RS-PSO**	**Shiwa**	**TBPSA**
CMA+Powell	N/A	≈	+	-	-	≈	+	-	≈	+
CMA	≈	N/A	+	-	≈	≈	+	-	≈	+
DE	-	-	N/A	-	-	-	+	-	-	+
Disc 1+1	+	+	+	N/A	+	+	+	≈	+	+
NGO	+	≈	+	-	N/A	≈	+	-	≈	+
Powell	≈	≈	+	-	≈	N/A	+	-	≈	+
RS	-	-	-	-	-	-	N/A	-	-	-
RS-PSO	+	+	+	≈	+	+	+	N/A	+	+
Shiwa	≈	≈	+	-	≈	≈	+	-	N/A	+
TBPSA	-	-	-	-	-	-	+	-	-	N/A
	B = 4000									
	CMA+Powell	**CMA**	**DE**	**Disc 1+1**	**NGO**	**Powell**	**RS**	**RS-PSO**	**Shiwa**	**TBPSA**
CMA+Powell	N/A	-	+	-	-	≈	+	-	-	+
CMA	+	N/A	+	≈	≈	≈	+	-	≈	+
DE	-	-	N/A	-	-	-	+	-	-	+
Disc 1+1	+	≈	+	N/A	≈	≈	+	-	≈	+
NGO	+	≈	+	≈	N/A	≈	+	-	≈	+
Powell	≈	≈	+	≈	≈	N/A	+	-	≈	+
RS	-	-	-	-	-	-	N/A	-	-	-
RS-PSO	+	+	+	+	+	+	+	N/A	+	+
Shiwa	+	≈	+	≈	≈	≈	+	-	N/A	+
TBPSA	-	-	-	-	-	-	+	-	-	N/A
	B = 7000									
	CMA+Powell	**CMA**	**DE**	**Disc 1+1**	**NGO**	**Powell**	**RS**	**RS-PSO**	**Shiwa**	**TBPSA**
CMA+Powell	N/A	-	+	≈	≈	≈	+	-	≈	+
CMA	+	N/A	+	+	+	+	+	≈	+	+
DE	-	-	N/A	-	-	-	+	-	-	+
Disc 1+1	≈	-	+	N/A	≈	≈	+	-	≈	+
NGO	≈	-	+	≈	N/A	≈	+	-	≈	+
Powell	≈	-	+	≈	≈	N/A	+	-	≈	+
RS	-	-	-	-	-	-	N/A	-	-	-
RS-PSO	+	≈	+	+	+	+	+	N/A	+	+
Shiwa	≈	-	+	≈	≈	≈	+	-	N/A	+
TBPSA	-	-	-	-	-	-	+	-	-	N/A

The *local fitting* scenario corresponds to a case where the meshes $\mathcal{M}_T$ and $\mathcal{M}_P$ are expected to have a small geometric difference. This is attributed to the small temporal difference between the two examined frames in the recorded sequence. As a consequence, optimizers that behave well in local exploration should have an edge over innately exploratory ones, especially under lower budgets.

Table 2 depicts the average loss values achieved, aggregated across different performances and experiment repetitions, categorized per budget and optimizer. Additionally, following the guidelines from [29], for each recorded sequence out of the total 10 (i.e. for each problem instance), we compute the average loss value achieved by each optimizer across the 5 repetitions and use this value as a representative value for the performance of the optimizer to the specific problem instance. We use these representative values to conduct a Wilcoxon signed rank test among all pairs of optimizers, at a confidence level of 0.05, evaluating whether there is a significant statistical difference on the optimizer performances. The results are depicted in Table 3.

Table 4. *Local fitting:* RMS Hausdorff distance, w.r.t the mesh bounding box, for experiments with the same budget and optimizer, aggregating across experiment repetitions and performances. Optimizers are sorted w.r.t the RMS hausdorff distance. The relative performance (RP) column indicates relative performance w.r.t the best performing optimizer for each budget.

B=2000			B=4000			B=7000		
Optimizer	**RP**	**Haus**	**Optimizer**	**RP**	**Haus**	**Optimizer**	**RP**	**Haus**
RS - PSO	+0.00%	0.00833	**RS - PSO**	+0.00%	0.00706	**CMA**	+0.00%	0.00610
Disc 1+1	+7.86%	0.00898	**Disc 1+1**	+16.67%	0.00824	**RS - PSO**	+3.56%	0.00631
NGO	+48.07%	0.01233	**Shiwa**	+17.61%	0.00831	**NGO**	+16.39%	0.00709
Shiwa	+48.61%	0.01238	**CMA**	+18.65%	0.00838	**Disc 1+1**	+23.01%	0.00750
CMA+Powell	+56.51%	0.01304	**NGO**	+21.52%	0.00858	**Powell**	+28.69%	0.00784
Powell	+60.06%	0.01333	**Powell**	+27.11%	0.00898	**CMA+Powell**	+30.53%	0.00796
CMA	+61.79%	0.01348	**CMA+Powell**	+53.61%	0.01085	**Shiwa**	+39.64%	0.00851
DE	+69.95%	0.01416	**DE**	+77.33%	0.01252	**DE**	+85.35%	0.01130
TBPSA	+116.29%	0.01802	**TBPSA**	+123.14%	0.01576	**TBPSA**	+125.82%	0.01377
RS	+170.79%	0.02256	**RS**	+201.43%	0.02129	**RS**	+233.29%	0.02032

According to Table 2, for budgets 2000 and 4000, RS-PSO achieves top performance while for budget 7000 it comes second, performing on-par with the top performing algorithm CMA. Wilcoxon signed rank test (Table 3) also testifies to the same conclusion. In the lower budget case $B = 2000$, Disc 1+1 performs close to RS-PSO which according to the Wilcoxon test, there is no significant statistical difference on the performance of the two optimizers under this budget. However, for $B = 4000$, while Disc 1+1 is the second best performing algorithm, the margin between its performance and RS-PSO's is significantly increased with RS-PSO outperforming Disc 1+1 significantly, as corroborated by the Wilcoxon test. As budget increases, the performance difference between NGO, CMA+Powell, Shiwa, Disc 1+1 and Powell is consistently decreasing, with the Wilcoxon test further supporting this result. We should note that the CMA algorithm is known in bibliography to perform on-par with the state-of-the-art for local optimization and large budgets, something apparent in our experiments as well, where it managed to reach top for the $B = 7000$ scenario and practically tie with RS-PSO. Finally, DE and TBPSA are ranked among the worst performing optimizers as they are consistently ranked 8th and 9th under all budgets, while, with no surprise, RS performs worst, regardless the budget.

As a post optimization process we calculated the average, across all experiments, RMS Haussdorf distance between the performance's point cloud data $\mathcal{M}_P$ and the animated template mesh $\mathcal{A}_\mathbf{p}$ at the converged pose. The results are depicted in Table 4. In general, the optimizers' performance in the RMS

Table 5. *Global fitting:* Average Loss for experiments with the same budget and optimizer, aggregating across experiment repetitions and performances. Optimizers are sorted with respect to the average achieved loss. The relative performance (RP) column indicates incremental relative performance with respect to the best performing optimizer for each budget. Results in this table are for $\lambda_J = 0.1$.

B=2000			B=4000			B=7000		
Optimizer	**RP**	**Loss**	**Optimizer**	**RP**	**Loss**	**Optimizer**	**RP**	**Loss**
Shiwa	+0.00%	0.06763	**NGO**	+0.00%	0.05565	**NGO**	+0.00%	0.04882
NGO	+1.46%	0.06861	**CMA**	+2.27%	0.05691	**Shiwa**	+3.54%	0.05055
Disc 1+1	+6.68%	0.07214	**Shiwa**	+2.40%	0.05699	**CMA+Powell**	+5.52%	0.05151
CMA	+8.46%	0.07335	**CMA+Powell**	+10.26%	0.06136	**CMA**	+6.84%	0.05216
RS - PSO	+14.55%	0.07747	**Disc 1+1**	+16.30%	0.06472	**RS - PSO**	+13.41%	0.05537
TBPSA	+23.56%	0.08356	**RS - PSO**	+18.52%	0.06596	**Disc 1+1**	+26.51%	0.06176
CMA+Powell	+26.59%	0.08561	**TBPSA**	+28.41%	0.07146	**TBPSA**	+33.91%	0.06537
DE	+32.08%	0.08932	**DE**	+36.31%	0.07586	**DE**	+35.92%	0.06635
Powell	+97.87%	0.13381	**Powell**	+64.57%	0.09158	**Powell**	+56.20%	0.07625

Hausdorff results is consistent with the performance in the achieved loss values (especially for $B = 2000$ and $B = 4000$) (Table 2) indicating that the proxy objective function (3) is a good approximation of the Hausdorff metric. However, since our objective function is an approximation to the Hausdorff metric, the ranking of the optimizers with respect to the RMS Hausdorff distance does not strictly follow the ranking with respect to loss in all cases. Despite that, across all budgets, the top 2 and worst 2 performing optimizers are the same in the two tables. For $B = 2000$ the 3rd and 4th ranked positions are taken by the same optimizers, regardless the metric (i.e loss/hausdorff). Further, for $B = 4000$, apart from loss, CMA and Shiwa perform on-par in RMS hausdorff terms, as well. However, for $B = 7000$, while in loss terms NGO, CMA+Powell, Shiwa, Disc 1+1 and Powell are considered to perform on-par, their difference in performance in RMS Hausdorff terms is more prominent.

Table 6. *Global fitting:* Average Hausdorff-RMS for experiments with the same budget and optimizer, aggregated across experiment repetitions and performances. Optimizers are sorted w.r.t the average Hausdorff-RMS achieved. The relative performance (RP) column indicates relative performance w.r.t the best performing optimizer for each budget.

B=2000			B=4000			B=7000		
Optimizer	**RP**	**Haus**	**Optimizer**	**RP**	**Haus**	**Optimizer**	**RP**	**Haus**
Shiwa	+0.00%	0.02629	**NGO**	+0.00%	0.02185	**Shiwa**	+0.00%	0.02087
NGO	+5.79%	0.02781	**CMA**	+2.93%	0.02249	**NGO**	+1.14%	0.02111
CMA	+14.22%	0.03002	**Shiwa**	+6.44%	0.02326	**CMA+Powell**	+4.55%	0.02182
Disc 1+1	+14.51%	0.03010	**CMA+Powell**	+14.27%	0.02497	**CMA**	+5.19%	0.02196
RS - PSO	+15.56%	0.03038	**TBPSA**	+18.39%	0.02587	**RS - PSO**	+5.90%	0.02211
TBPSA	+18.13%	0.03105	**RS - PSO**	+19.51%	0.02611	**TBPSA**	+16.48%	0.02431
CMA+Powell	+22.91%	0.03230	**Disc 1+1**	+27.94%	0.02796	**DE**	+21.14%	0.02529
DE	+35.72%	0.03568	**DE**	+28.52%	0.02808	**Disc 1+1**	+22.18%	0.02550
Powell	+122.05%	0.05837	**Powell**	+100.68%	0.04385	**Powell**	+92.38%	0.04016

Overall, RS-PSO can be seen as a good candidate for solving the *local fitting* case, as it is always ranked first except for the higher budget case in which, however, it is ranked second with a very small performance difference (+0.51% in loss terms and +3.56% in Hausdorff distance terms) with respect to the best performing optimizer (CMA). This finding, may also apply to other real-world problems as well, where the optimal solution is known to lie in the local neighborhood of a known initial point. Last, as budget increases, CMA is the only algorithm that consistently improves its ranked position in the benchmark, with no other optimizer exhibiting the same behaviour.

5.2 Global fitting

Table 7. Wilcoxon test for the *Global fitting case*, at a confidence level of 0.05. When the optimizer at cell's row performs better (worse/equal) than the optimizer at cell's column, the cell is marked with '+' ('-'/'≈').

	B = 2000								
	CMA+Powell	**CMA**	**DE**	**Disc 1+1**	**NGO**	**Powell**	**RS-PSO**	**Shiwa**	**TBPSA**
CMA+Powell	N/A	-	≈	-	-	+	-	-	≈
CMA	+	N/A	+	≈	≈	+	≈	≈	+
DE	≈	-	N/A	-	-	≈	-	-	≈
Disc 1+1	+	≈	+	N/A	≈	+	≈	≈	+
NGO	+	≈	+	≈	N/A	+	+	≈	+
Powell	-	-	≈	-	-	N/A	-	-	-
RS-PSO	+	≈	+	≈	-	+	N/A	-	≈
Shiwa	+	≈	+	≈	≈	+	+	N/A	+
TBPSA	≈	-	≈	-	-	+	≈	-	N/A
	B = 4000								
	CMA+Powell	**CMA**	**DE**	**Disc 1+1**	**NGO**	**Powell**	**RS-PSO**	**Shiwa**	**TBPSA**
CMA+Powell	N/A	≈	+	≈	≈	+	+	≈	+
CMA	≈	N/A	+	≈	≈	+	+	≈	+
DE	-	-	N/A	-	-	+	-	-	≈
Disc 1+1	≈	≈	+	N/A	-	+	≈	≈	+
NGO	≈	≈	+	+	N/A	+	+	≈	+
Powell	-	-	-	-	-	N/A	-	-	-
RS-PSO	-	-	+	≈	-	+	N/A	-	≈
Shiwa	≈	≈	+	≈	≈	+	+	N/A	+
TBPSA	-	-	≈	-	-	+	≈	-	N/A
	B = 7000								
	CMA+Powell	**CMA**	**DE**	**Disc 1+1**	**NGO**	**Powell**	**RS-PSO**	**Shiwa**	**TBPSA**
CMA+Powell	N/A	≈	+	+	≈	+	≈	≈	+
CMA	≈	N/A	+	+	≈	+	≈	≈	+
DE	-	-	N/A	≈	-	≈	-	-	≈
Disc 1+1	-	-	≈	N/A	-	+	≈	-	≈
NGO	≈	≈	+	+	N/A	+	+	≈	+
Powell	-	-	≈	-	-	N/A	-	-	≈
RS-PSO	≈	≈	+	≈	-	+	N/A	≈	≈
Shiwa	≈	≈	+	+	≈	+	≈	N/A	≈
TBPSA	-	-	≈	≈	-	≈	≈	≈	N/A

In contrast to *local fitting*, in the *global fitting* scenario, the optimizers expected to perform best are the ones which better explore the search space beyond the locality of the initial point. Similar to the *local fitting* case, Table 5 illustrates the average achieved loss values aggregated across different performances and experiment repetitions, categorized per budget and optimizer.Further, we perform the Wilcoxon signed rank test exactly the same way as it was described for the *local fitting* scenario (Table 7).

In loss terms, regardless the budget, Powell is consistently ranked in the last position of the benchmark, Shiwa is always ranked among the first 3 places and TBPSA and DE perform on-par with Powell for $B = 7000$. For $B = 2000$ Shiwa performs best, on-par with NGO, Disc 1+1 and CMA, as testified by the Wilcoxon test. In the fourth place comes RS-PSO, followed by TBPSA, CMA+Powell and DE which perform similarly. For $B = 4000$ and $B = 7000$, the top performing optimizer is NGO, with CMA, Shiwa and CMA+Powell exhibiting similar performance. Disc 1+1 performs on-par with RS-PSO for $B = 4000$. Finally, TBPSA and DE perform similarly and are ranked 7th and 8th respectively. Last, regarding $B = 7000$, RS-PSO and Disc 1+1 perform on-par and are ranked 5th and 6th.

Regarding the geometric error measured by RMS hausdorff distance between the performance's point cloud data $\mathcal{M}_P$ and the animated template mesh $\mathcal{A}_\mathbf{p}$ at the converged pose, the ranking on the performance of the optimizers mostly follows the same patterns, as depicted in Table 6.

Overall, regardless budget, NGO, is the best candidate to solve the *Global fitting* problem, followed by Shiwa. As budget increases, the only optimizers that are able to *consistently* improve their ranked position are NGO (in loss terms) and CMA+Powell (in loss and RMS Hausdorff terms), while Disc 1+1 is consistently outperformed by more optimizers as budget increases (i.e its ranked position is worsened across budgets). RS-PSO which was found to perform best in the *local fitting* case, in this scenario is outperformed by other optimizers by a considerable margin, especially at lower budgets.

Table 8. *Sequence fitting:* Average loss and RMS Hausdorff for $B = 4000$, aggregated across frames and sequences. The relative performance (RP) column indicates relative performance with respect to the best performing optimizer.

Optimizer	RP	Loss	Optimizer	RP	Hauss
RS-PSO	+0.00%	0.03092	**RS-PSO**	+0.00%	0.012379
Disc 1+1	+21.54%	0.03758	**NGO**	+24.03%	0.015354
NGO	+28.72%	0.03980	**Shiwa**	+28.73%	0.015936
Shiwa	+30.69%	0.04041	**Disc 1+1**	+29.52%	0.016034
CMA	+31.24%	0.04058	**CMA**	+32.12%	0.016356

5.3 Sequence fitting

The *sequence fitting* scenario can be considered as an extension to the *local fitting* case, essentially evaluating the optimizers in aggregated like *local fitting* experiments, expanding the *local fitting* dataset to a large number of frames. As errors tend to accumulate in this setting, optimizers with more aggressive exploration strategies may be able to overcome error accumulation. Numerical results regarding loss and RMS hausdorff distance between $\mathcal{M}_P$ and $\mathcal{A}_\mathbf{p}$ are given in Table 8. Interestingly, RS-PSO is ranked first both in loss and RMS Hausdorff terms, just like in *local fitting*, outperforming other optimizers considerably. While Disc 1+1 is ranked 2nd in loss terms, under the RMS Hausdorff metric is ranked 4th. NGO and Shiwa are ranked second and third in RMS Hausdorff terms, even though they are ranked 3rd and 4th, respectively, under the loss criterion. Similar to *local fitting*, both NGO and Shiwa perform similarly in *sequence fitting*, as well. Finally CMA is the worse performing optimizer with respect to both loss and RMS Hausdorff metrics. Overall, compared to the corresponding *local fitting* case with $B = 4000$, apart from RS-PSO performing top, which is common in both cases, the rankings of the other optimizers slightly vary. This can be explained by the Wilcoxon signed rank test which for the *local fitting* case indicated that all optimizers apart from RS-PSO perform on-par.

5.4 Supplementary material

Additional qualitative figures depicting the animated template at the converged poses along with animatable exploration strategy visualizations for all optimizers for some experiments are given in supplementary material (`https://github.com/VCL3D/nevergrad/blob/pb-material/supplementary pdf`).

6. Conclusion

In this paper, we thoroughly benchmarked public open implementations of zeroth-order optimizers to the real-word problem of template mesh fitting to 3D captured performances. Three different fitting cases were considered, essentially covering three different real-world scenarios. Overall, RS-PSO was found to perform best in the *local* and *sequence fitting* cases, while for the *global fitting* scenario the best performing optimizer was NGO. Individual experiment result plots and tables, as well as an ablation study for different objective function error term weightings and qualitative samples are given in supplementary material. We hope that this work can become a future reference for practitioners in 3D performance capture and additionally serve as a baseline for future works approaching the same problem from the scope of multi-objective optimization.

7. Acknowledgement

This work has been supported by the EU funded H2020 Program: INFINITY, Grant Agreement 883293.

References

[1] Ilya Baran and Jovan Popović. Automatic rigging and animation of 3d characters. *ACM Transactions on graphics (TOG)*, 26(3):72–es, 2007.

[2] Weipeng Xu, Avishek Chatterjee, Michael Zollhöfer, Helge Rhodin, Dushyant Mehta, Hans-Peter Seidel, and Christian Theobalt. Monoperfcap: Human performance capture from monocular video. *ACM Transactions on Graphics (ToG)*, 37(2):1–15, 2018.

[3] Marc Habermann, Weipeng Xu, Michael Zollhoefer, Gerard Pons-Moll, and Christian Theobalt. Livecap: Real-time human performance capture from monocular video. *ACM Transactions on Graphics (TOG)*, 38(2):1–17, 2019.

[4] Edilson De Aguiar, Carsten Stoll, Christian Theobalt, Naveed Ahmed, Hans-Peter Seidel, and Sebastian Thrun. Performance capture from sparse multi-view video. In *ACM SIGGRAPH 2008 papers*, pages 1–10. 2008.

[5] J. Rapin and O. Teytaud. Nevergrad - A gradient-free optimization platform. `https://GitHub.com/FacebookResearch/Nevergrad`, 2018.

[6] Vladimiros Sterzentsenko, Alexandros Doumanoglou, Spyridon Thermos, Nikolaos Zioulis, Dimitrios Zarpalas, and Petros Daras. Deep soft procrustes for markerless volumetric sensor alignment. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 818–827. IEEE, 2020.

[7] Vladimiros Sterzentsenko, Antonis Karakottas, Alexandros Papachristou, Nikolaos Zioulis, Alexandros Doumanoglou, Dimitrios Zarpalas, and Petros Daras. A low-cost, flexible and portable volumetric capturing system. In *2018 14th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pages 200–207. IEEE, 2018.

[8] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013.

[9] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(null):281–305, February 2012.

[10] Rainer Storn and Kenneth Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, Dec 1997.

[11] M. Zambrano-Bigiarini, M. Clerc, and R. Rojas. Standard particle swarm optimisation 2011 at cec-2013: A baseline for future pso improvements. In *2013 IEEE Congress on Evolutionary Computation*, pages 2337–2344, 2013.

[12] M. Schumer and K. Steiglitz. Adaptive step size random search. *IEEE Transactions on Automatic Control*, 13(3):270–276, 1968.

[13] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[14] Michael Hellwig and Hans-Georg Beyer. Evolution under strong noise: A self-adaptive evolution strategy can reach the lower performance bound - the pccmsa-es. In Julia Handl, Emma Hart, Peter R. Lewis, Manuel López-Ibáñez, Gabriela Ochoa, and Ben Paechter, editors, *Parallel Problem Solving from Nature – PPSN XIV*, pages 26–36, Cham, 2016. Springer International Publishing.

[15] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, 01 1964.

[16] Jialin Liu, Antoine Moreau, Mike Preuss, Jeremy Rapin, Baptiste Roziere, Fabien Teytaud, and Olivier Teytaud. Versatile black-box optimization. GECCO '20, page 620–628, New York, NY, USA, 2020. Association for Computing Machinery.

[17] F Sebastian Grassia. Practical parameterization of rotations using the exponential map. *Journal of graphics tools*, 3(3):29–48, 1998.

[18] Ladislav Kavan, Steven Collins, Jiří Žára, and Carol O'Sullivan. Skinning with dual quaternions. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, pages 39–46, 2007.

[19] Dimitrios S Alexiadis, Anargyros Chatzitofis, Nikolaos Zioulis, Olga Zoidi, Georgios Louizis, Dimitrios Zarpalas, and Petros Daras. An integrated platform for live 3d human reconstruction and motion capturing. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(4):798–813, 2016.

[20] Michael Kazhdan. Reconstruction of solid models from oriented point sets. In *Proceedings of the third Eurographics symposium on Geometry processing*, pages 73–es, 2005.

[21] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987.

[22] A. Doumanoglou, D. S. Alexiadis, D. Zarpalas, and P. Daras. Toward real-time and efficient compression of human time-varying meshes. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(12):2099–2116, 2014.

[23] Dimitrios S Alexiadis, Nikolaos Zioulis, Dimitrios Zarpalas, and Petros Daras. Fast deformable model-based human performance capture and fvv using consumer-grade rgb-d sensors. *Pattern Recognition*, 79:260–278, 2018.

[24] David Coeurjolly and Annick Montanvert. Optimal separable algorithms to compute the reverse euclidean distance transformation and discrete medial axis in arbitrary dimension. *IEEE transactions on pattern analysis and machine intelligence*, 29(3):437–448, 2007.

[25] Dmitriy Smirnov, Matthew Fisher, Vladimir G Kim, Richard Zhang, and Justin Solomon. Deep parametric shape predictions using distance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 561–570, 2020.

[26] Paul Jaccard. Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. *Bull Soc Vaudoise Sci Nat*, 37:241–272, 1901.

[27] RabbitMQ - Messaging that just works. `https://www.rabbitmq.com/`, 2021.

[28] Ilya Baran and Jovan Popović. Automatic rigging and animation of 3d characters. *ACM Trans. Graph.*, 26(3):72–es, July 2007.

[29] Joaquín Derrac, Salvador García, Daniel Molina, and Francisco Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3 – 18, 2011.