2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Visual Tracking of Deforming Objects Using Physics-based Models

Agniva Sengupta[1], Alexandre Krupa[1], Eric Marchand[1]

Abstract— **In this paper, we propose a framework for tracking the deformation of soft objects using a RGB-D camera by utilizing the physically-based model of the considered object. A coarse, 3D template of the object being tracked is the only prior information required by the proposed method. The proposed approach does not rely on the accurate knowledge of the material properties of the object being tracked. In this paper, we integrate computer vision based tracking methodology with physical model based deformation representation without requiring expensive numerical optimization for minimizing non-linear error terms. The proposed approach enables deformation tracking by joint minimization of a geometric error and a direct photometric intensity error while utilizing co-rotational Finite Element Method (FEM) as the underlying deformation model. The proposed method has been validated both on synthetic data (with groundtruth) and real data.**

I. Introduction

Physically based models have been used to analyze deformations of objects of complex shapes since the inception of structural mechanics. Some of the recent computer-vision based approaches have tried to leverage the advantages of the physics based models for deformation tracking. Surprisingly, these physical-model based approaches often fall short of other competing deformation models in terms of accuracy [1]. Many deformation tracking methodologies used by the computer vision and robotics community utilizes geometric regularizers or assumption of isometry or conformality as a deformation model. However, structural mechanics offer many highly evolved methodologies for modelling deformation in complex shaped objects. In this paper, we propose a method of non-rigid object tracking using a RGB-D camera that combines the minimization of a geometric error that penalizes the model to pointcloud distance and a photometric error that penalizes inter-frame photometric intensity difference. A coarse, 3D template of the object being tracked is the only prior input necessary for the proposed approach. The entire observable surface of the deforming object is tracked in 3D using the data from the RGB-D camera. The deformation model used for representing the underlying physics is based on FEM. Thanks to the physics based model, it is possible to track large deformations that do not preserve the volume of the object. The proposed method is robust to occlusion and the tracking accuracy is not dependent on apriori knowledge of physical properties of the deforming object. However, if the physical parameters are known apriori, we demonstrate that the approach can also be utilized for estimating the external contact forces applied on the surface of the deforming object. The following are the main contributions of this paper:

- We propose a new methodology for deformation tracking using joint optimization of a geometric and photometric error defined on the surface model of the deforming object. A non-linear least squares optimization based method with an analytic expression of the Jacobian relating the variation of geometric and visual error to displacement of the vertices of the mechanical mesh is proposed to track the deformation of objects
- The method proposed here has also been utilized to track external forces applied on the surface of deforming objects. Such applications are an important advantage of using physically-based models for deformation tracking.

[1]*Univ Rennes, Inria, IRISA, CNRS, Rennes, France*
e-mail: {agniva.sengupta, alexandre.krupa, eric.marchand}@irisa.fr.

II. Background

A method to track deformable objects by applying virtual forces on the simulation of a physically based model was proposed in [2]. The physics of the deforming object was modelled as a collection of linked rigid bodies or particles and the tracking was formulated as an Expectation-Maximization (EM) problem. The qualitative results in [2] are impressive, but there are a few sequences where the tracking suffers noticeable drift in heavily occluded regions. It is not trivial to modify the approach of [2] to utilize a Newtonian optimization scheme for minimizing the visual error. [3] improved the approach of [2] by considering co-rotational FEM. But, in [3], no attempt was made to tackle the deformation tracking problem using conventional optimization since the gradient estimation of the high dimensional optimization step was considered to be intractable. In this context, it must be noted that physical model free deformation tracking approaches such as [4], [5] depend heavily on the assumption that the deformation is isometric. Consequently, these approaches are not preferable when there is a significant change in the volume of the deforming object. In [6], the authors utilized co-rotational FEM to track deformation by purely geometric matching of pointcloud with the 3D object model, producing highly accurate tracking results at frame-rate. However, the approach of [6] was also sensitive to occlusion, large inter-frame motion and demonstrated a tendency to drift along the tangent of large planar surfaces, which can be intuitively explained as an inherent drawback of tracking solely using point-to-point or point-to-plane correspondences. In [7], the deformation tracking using physically based model has been formulated as a conventional optimization problem, but only using depth information. Moreover, the optimization in [7] requires expensive simulation of object deformation using FEM for computing the gradient of the visual error, which

is unsuitable for deformation tracking at frame-rate. [8] proposes a force tracking methodology based on the approach of [7] which is slow and does not utilize the photometric information for tracking.

III. Methodology

The proposed method tracks deformable objects using a RGB-D camera with the help of a coarse 3D template of the deforming object. The proposed approach can be summarized by the following iterative steps:

- Rigidly track the deforming object w.r.t the camera
- Analyze the residual error after rigid tracking to determine the region which is likely to have deformed
- Minimize a combination of geometric and photometric error terms to track the deformation using a set of selected vertices of the model in those deformed region

We now describe the entire approach in details.

A. Notation

Throughout the rest of the paper, the notation $\mathbf{Q}$ is used to denote the vertices of the volumetric mechanical tetrahedral mesh $\mathcal{K}$, while $\mathbf{P}^{\mathcal{S}} = (\mathrm{P}_{\mathrm{X}}^{\mathcal{S}}, \mathrm{P}_{\mathrm{Y}}^{\mathcal{S}}, \mathrm{P}_{\mathrm{Z}}^{\mathcal{S}})$ denotes the vertices of the surface model $\mathcal{S}$. The mapping $\mathcal{S} \mapsto \mathcal{K}$ is done via barycentric mapping. Following standard convention, the object to camera homogeneous transformation matrix is given as ${}^{C}\mathbf{T}_{O} = \begin{bmatrix} {}^{C}\mathbf{R}_{O} & {}^{C}\mathbf{t}_{O} \\ 0 & 1 \end{bmatrix}$ and this is utilized to align $\mathcal{K}$ and $\mathcal{S}$ to the actual object, where ${}^{C}\mathbf{R}_{O}$ and ${}^{C}\mathbf{t}_{O}$ are the rotation matrix and translation vector respectively. Between two consecutive data frames, the pose of the object in current frame (p) with respect to pose in the previous frame $(p-1)$ is $\mathbf{q} = ({}^{p}\mathbf{t}_{p-1}, \theta\mathbf{u})$, where θ and $\mathbf{u}$ are the angle and axis of rotation ${}^{p}\mathbf{R}_{p-1}$. The time derivative of $\mathbf{q}$ is given as ${}^{p}\mathbf{v}_{p-1} = \delta\mathbf{q}$, where $\mathbf{v} \in \mathrm{se}(3)$ is the velocity screw. The color image for the p-th data frame is given by $\boldsymbol{\mathcal{I}}_p$. The perspective projection function $\boldsymbol{\Pi}(\cdot)$ is used to project the 3D point $\mathbf{P}$ to its corresponding 2D coordinates (u, v) in the image plane, such that $\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{\Pi}\Big(\begin{bmatrix} \mathbf{P} \\ 1 \end{bmatrix} \Big) = \mathbf{K}_p \boldsymbol{\Pi}_p {}^{\mathrm{c}}\mathbf{T}_{\mathrm{o}} \begin{bmatrix} \mathbf{P} \\ 1 \end{bmatrix}$, where $\mathbf{K}_p$ is the camera calibration matrix in the standard notation and $\boldsymbol{\Pi}_p = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \end{bmatrix}$. The function $\boldsymbol{\mathcal{I}}(u, v)$ gives the grayscale intensity at that pixel position and $(\nabla\boldsymbol{\mathcal{I}}_p^u, \nabla\boldsymbol{\mathcal{I}}_p^v)$ gives the corresponding image gradient. (f_x, f_y) are focal lengths in pixel units.

B. Deformation Model

We employ FEM [9] to model the underlying physics of the non-rigid object. Let the four vertices of a particular face of the tetrahedral mechanical mesh be denoted by the vertices $\mathbf{Q}_1$, $\mathbf{Q}_2$, $\mathbf{Q}_3$ and $\mathbf{Q}_4$ and the centroid be a point $\mathbf{Q}_{\mathrm{C}}$. To obtain the displacement of vertices as a result of application of external force on the object, a second order differential equation of the following nature needs to be solved:

$$\mathbf{M}\ddot{\mathbf{Q}} + \mathbf{D}\dot{\mathbf{Q}} + \mathbf{K}\mathbf{Q} = \mathbf{F}_{ext} + \mathbf{F}_e \tag{1}$$

where $\mathbf{M}$ and $\mathbf{D}$ represent the mass and damping matrices of the model respectively, and $\mathbf{K}$ is the global stiffness matrix. $\mathbf{F}_{ext}$ denotes the external forces acting on the vertices and $\mathbf{F}_e$ are the internal elastic force vector acting on the vertices of the tetrahedral elements. A linear solver based on conjugate gradient descent is used to solve Eqn. (1) using [10]. Co-rotational FEM [11] is a modification of linear FEM that allows robust handling of strong deformations in the mesh. If the model undergoes a large deformation, let the centroid of the tetrahedral element at this deformed configuration be denoted by a point $\mathbf{Q}_{\mathrm{C}}^{\mathrm{R}}$. An arbitrary vertex $\mathbf{Q}_x$ of the mechanical mesh in the undeformed frame gets deformed to $\mathbf{Q}$ in the new frame (variables with the $\tilde{\cdot}$ notation are expressed in the corotated frame $\mathbf{Q}_{\mathrm{C}}^{\mathrm{R}}$). With the co-rotational formulation [12], it can be shown that $\tilde{\mathbf{d}} = \mathbf{R}^{\top}\big(\mathbf{U} - \mathbf{Q}_{\mathrm{C}} + \mathbf{Q}_x\big) - \mathbf{Q}_x$ given that $\mathbf{R}$ is the rotation matrix, $\mathbf{U}$ is the displacement of $\mathbf{Q}_x$, expressed in the object frame of $\mathbf{Q}_{\mathrm{C}}$, and $\tilde{\mathbf{d}}$ is the displacement of $\mathbf{Q}_x$ in the corotated frame $\mathbf{Q}_{\mathrm{C}}^{\mathrm{R}}$. Applying similar deformation to the vertices of the tetrahedron, the internal, elastic forces can be represented by:

$$\mathbf{F}_e = \mathbf{R}_e\mathbf{K}_e\tilde{\mathbf{U}}^R \tag{2}$$

Here, $\tilde{\mathbf{U}}^R = (\tilde{\mathbf{Q}}_1, \tilde{\mathbf{Q}}_2, \tilde{\mathbf{Q}}_3, \tilde{\mathbf{Q}}_4)$, $\mathbf{K}_e$ represents the stiffness matrix and $\mathbf{R}_e$ is the 12×12 block diagonal matrix of four 3×3 rotation matrices $\mathbf{R}$ stacked diagonally.

C. Visual Tracking

We track the deforming object by minimizing two error terms defined on the surface of $\mathcal{S}$. The minimization is done using a set of *control handles* $\mathbf{C}$ on $\mathcal{K}$, such that the entire deformation becomes optimizable by regulating the displacement of $\mathbf{C}$. $\mathbf{C}$ are those vertices of $\mathcal{K}$ that shall be controlled to minimize the error terms defined on $\mathcal{S}$. Any displacement of $\mathbf{C}$ gets propagated to all vertices of $\mathcal{K}$ using the deformation model. We are interested in minimizing a geometric error $\mathbf{E}_{\mathrm{depth}}$ and a photometric error $\mathbf{E}_{\mathrm{photo}}$. The combined error term can be represented as:

$$\mathbf{E}_{\mathcal{S}} = \begin{bmatrix} \mathbf{E}_{\mathrm{depth}} & \mathbf{E}_{\mathrm{photo}} \end{bmatrix}^{\top} \tag{3}$$

$\mathbf{E}_{\mathcal{S}}$ is minimized using a set of displacement vector on $\mathbf{C}$. For the sake of simplicity, we describe the proposed mechanism of minimizing $\mathbf{E}_{\mathcal{S}}$ with the help of a single *control handle*, i.e., the u-th *control handle* of $\mathbf{C}$, denoted by $\mathbf{u}_{\mathrm{C}}$. However, to optimize in a non-linear least squares fashion, we need to obtain the gradient of the stacked cost function. Numeric estimation can be computationally expensive as determining the node displacements typically involves multiple iteration of conjugate gradient descent for solving an equation of the nature of Eqn. (1). One of the key contribution of this paper is the proposed mechanism for minimizing $\mathbf{E}_{\mathcal{S}}$ with non-linear least squares optimization along with a strategy for analytically estimating the Jacobian.

Let us analyze the case of an arbitrary 3D point $\mathbf{P}$ of the pointcloud, which lies on or near the triangle $\begin{pmatrix} \mathbf{P}_i^{\mathcal{S}} & \mathbf{P}_{i+1}^{\mathcal{S}} & \mathbf{P}_{i+2}^{\mathcal{S}} \end{pmatrix}$ of $\mathcal{S}$. Let us assume that a small displacement $\Delta\mathbf{u}_{\mathrm{C}} = \begin{bmatrix} \Delta\mathrm{u}_{\mathrm{C_X}} & \Delta\mathrm{u}_{\mathrm{C_Y}} & \Delta\mathrm{u}_{\mathrm{C_Z}} \end{bmatrix}$ produces the displacement $\Delta\mathbf{x}, \Delta\mathbf{y}$ and $\Delta\mathbf{z}$ on the j-th surface

plane/facet of the mesh $\mathcal{S}$ (due to the underlying deformation of the mechanical mesh) comprising of three vertices $\mathbf{P}_i^{\mathcal{S}}, \mathbf{P}_{i+1}^{\mathcal{S}}$ and $\mathbf{P}_{i+2}^{\mathcal{S}}$ respectively. This vector of vertex positions is given by:

$$\boldsymbol{\vartheta}_j = \begin{bmatrix} (\mathbf{P}_i^{\mathcal{S}})^\top & (\mathbf{P}_{i+1}^{\mathcal{S}})^\top & (\mathbf{P}_{i+2}^{\mathcal{S}})^\top \end{bmatrix} \tag{4}$$

Subsequently, the gradient of the error $\mathbf{E}_{\mathcal{S}}(\mathbf{P})$ w.r.t $\mathbf{u}_{\mathrm{C}}$ is given by the Jacobian:

$$\mathbf{J}_{\mathbf{S}}(\mathbf{P}) = \frac{\partial \mathbf{E}_{\mathcal{S}}(\mathbf{P})}{\partial \mathbf{u}_{\mathrm{C}}} = \underset{1\times 9}{\frac{\partial \mathbf{E}_{\mathcal{S}}(\mathbf{P})}{\partial \boldsymbol{\vartheta}_j}} \underset{9\times 3}{\frac{\partial \boldsymbol{\vartheta}_j}{\partial \mathbf{u}_{\mathrm{C}}}} \tag{5}$$

The term $\frac{\partial \boldsymbol{\vartheta}_j}{\partial \mathbf{u}_{\mathrm{C}}}$ is somewhat similar to the classical strain-displacement matrix, as expressed conventionally in FEM literature [13]. The details of how $\frac{\partial \mathbf{E}_{\mathcal{S}}(\mathbf{P})}{\partial \boldsymbol{\vartheta}_j}$ can be derived in our case is described in Sec. III-E.

Computing $\frac{\partial \boldsymbol{\vartheta}_j}{\partial \mathbf{u}_{\mathrm{C}}}$ can be done by estimating the deformation of the mechanical model $\mathcal{K}$, for every new *control handle* explored by our proposed approach, only once. For the j-th *control handle* $\mathbf{C}_j$, this is done by successively displacing the vertex in $\mathcal{K}$ corresponding to $\mathbf{C}_j$ by a small distance along the positive direction of X, Y and Z axis. This estimation step produce three deformed meshes per *control handle*, which can be directly utilized for determining $\frac{\partial \boldsymbol{\vartheta}_j}{\partial \mathbf{u}_{\mathrm{C}}}$ using forward finite differences. Assuming that $\mathbf{u}_{\mathrm{C}}$ points to the c-th vertex of $\mathcal{K}$, the values of $\frac{\partial \mathbf{P}_i^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}}$ can be mapped to a matrix $\boldsymbol{\Gamma}_{\mathrm{O}}$ by a mapping function $\mathscr{C}$ such that $\boldsymbol{\Gamma}_{\mathrm{O}} = \mathscr{C}\big(\frac{\partial \mathbf{P}_i^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}}\big)$ where:

$$\frac{\partial \boldsymbol{\vartheta}_j}{\partial \mathbf{u}_{\mathrm{C}}} = \Big(\frac{\partial \mathbf{P}_i^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}}, \frac{\partial \mathbf{P}_{i+1}^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}}, \frac{\partial \mathbf{P}_{i+2}^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}}\Big) \tag{6}$$

Here, $\mathscr{C}$ represents the mapping $\frac{\partial \mathbf{P}_i^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}} \mapsto \boldsymbol{\Gamma}_{\mathrm{O}}$. Assuming that M denotes the number of vertices in $\mathcal{K}$ and $\mathcal{S}$, $\mathscr{C}$ can be expressed as:

$$\boldsymbol{\Gamma}_{\mathrm{O}} = \mathscr{C}\Big(\frac{\partial \mathbf{P}_i^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}}\Big) = \sum_{i=0}^{\mathrm{M}} \sum_{c=0}^{\mathrm{M}} \frac{\partial \mathbf{P}_i^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}} \otimes \mathfrak{J}_{\mathrm{M}\times\mathrm{M}}^{i,c} \tag{7}$$

where $\otimes$ gives the Kronecker product and $\mathfrak{J}_{\mathrm{M}\times\mathrm{M}}^{i,c}$ denotes the conventional single-valued matrix of dimension $[\mathrm{M}\times\mathrm{M}]$ for the c-th control handle, such that:

$$\big(\mathfrak{J}_{\mathrm{M}\times\mathrm{M}}^{i,c}\big)_{\mathrm{x,y}} = \begin{cases} 1 & \text{if } \mathrm{x} = i \text{ and } \mathrm{y} = c, \forall \mathrm{x}, \mathrm{y} \in \mathrm{M} \\ 0 & \text{else} \end{cases} \tag{8}$$

and the x and y in the subscript of $(\cdot)_{\mathrm{x,y}}$ denotes the row and column index of an element of the matrix. Based on the matrix indices, $\boldsymbol{\Gamma}_{\mathrm{O}}$ can be represented as:

$$\big(\boldsymbol{\Gamma}_{\mathrm{O}}\big)_{a\mathrm{M}+i, b\mathrm{M}+c} = \big(\frac{\partial \mathbf{P}_i^{\mathcal{S}}}{\partial \mathbf{u}_{\mathrm{C}}}\big)_{a,b} \mid \forall i, c \in \mathrm{M} \wedge a, b \in \{0,1,2\} \tag{9}$$

We term the matrix $\boldsymbol{\Gamma}_{\mathrm{O}}$ as *influence matrix* (since it denotes the *influence* of the displacement of a vertex on its neighbors) and it can be conveniently computed offline once per every object model and stored. During tracking, $\boldsymbol{\Gamma}_{\mathrm{O}}$ is loaded from memory when required.

For estimating Eqn. (5), we derive a new matrix $\mathbf{R}^{\Gamma}$ such that:

$$\underset{3\mathrm{M}\times 3\mathrm{M}}{\mathbf{R}^{\Gamma}} = {}^{C}\mathbf{R}_{O} \otimes \mathbf{I}_{\mathrm{M}\times\mathrm{M}} \tag{10}$$

where $\mathbf{I}_{\mathrm{M}\times\mathrm{M}}$ is an identity matrix of dimension $[\mathrm{M}\times\mathrm{M}]$. $\mathbf{R}^{\Gamma}$ allows us to rotate the *influence matrix* to the current camera reference frame. This transformed matrix $\boldsymbol{\Gamma}_{\mathrm{C}}$ is given by $\boldsymbol{\Gamma}_{\mathrm{C}} = \mathbf{R}^{\Gamma}\boldsymbol{\Gamma}_{\mathrm{O}}$. We then derive a new matrix $\boldsymbol{\Gamma}$ by thresholding $\boldsymbol{\Gamma}_{\mathrm{C}}$ such that its value at an arbitrary index m, n associated with the i-th vertex $\mathbf{P}_i^{\mathcal{S}}$ is given by:

$$\boldsymbol{\Gamma}_{m,n} = \begin{cases} 1 & \text{if } \boldsymbol{\varepsilon} > \frac{3\alpha||\Delta\mathbf{u}_C||}{2} \\ 0 & \text{else} \end{cases} \tag{11}$$

given that $\boldsymbol{\varepsilon} = \sum_{a=0}^{2}\sum_{b=0}^{2}\Big(\big(\boldsymbol{\Gamma}_{\mathrm{C}}\big)_{m+a,n+b}\Big)^2$ where $\Delta\mathbf{u}_C$ is the calibrating deformation (implying that $||\Delta\mathbf{u}_C||$ is a constant for a particular $\boldsymbol{\Gamma}_{\mathrm{O}}$). α is a tunable parameter which regulates the area for which the value of $\boldsymbol{\Gamma}$ will be 1.

This thresholding of Eqn. (11) is done because we intend to utilize the point-to-plane distance as the error term $\mathbf{E}_{\mathrm{depth}}$. It is known classically that point-to-plane distance minimization using planar (or nearly planar) surface model can result in the model '*sliding*' over the 3D data [14]. The calibrating deformation $\Delta\mathbf{u}_C$ produces larger deformation in vertices closer to the vertex index c, i.e., $\boldsymbol{\Gamma}_{\mathrm{C}}$ highly prioritizes vertices close to c. This, in turn, converts the Jacobian in Eqn. (5) to effectively consider only a small patch of surface near the vicinity of the c-th vertex, while neglecting the small magnitude of deformation observed in the vertices farther away from the c-th vertex. To overcome this problem, we propose to binarize $\boldsymbol{\Gamma}_{\mathrm{C}}$ to $\boldsymbol{\Gamma}$ for obtaining the gradient of $\mathbf{E}_{\mathrm{depth}}$. However, the Jacobian for $\mathbf{E}_{\mathrm{photo}}$ can be obtained directly from $\boldsymbol{\Gamma}_{\mathrm{C}}$ without modification.

D. Determining the Control Handles

It is possible to estimate suitable positions for the *control handles* by analyzing the cost function. Given the two objective functions $\mathbf{E}_{\mathrm{depth}}$ and $\mathbf{E}_{\mathrm{photo}}$, we define two vectors $\mathbf{Z}^{\mathrm{depth}}$ and $\mathbf{Z}^{\mathrm{photo}}$, such that their values at the index corresponding to the 2D image point $\boldsymbol{\Pi}(\mathbf{P}_l)$ is given by:

$$\begin{aligned} \mathbf{Z}^{\mathrm{depth}}_{\boldsymbol{\Pi}(\mathbf{P}_l)} &= (\mathbf{W}_{\mathrm{depth}})_{l,l}\mathbf{E}_{\mathrm{depth}}(\mathbf{P}_l) \\ \mathbf{Z}^{\mathrm{photo}}_{\boldsymbol{\Pi}(\mathbf{P}_l)} &= (\mathbf{W}_{\mathrm{photo}})_{l,l}\mathbf{E}_{\mathrm{photo}}(\mathbf{P}_l) \end{aligned} \tag{12}$$

for the l-th point $\mathbf{P}_l$ of the pointcloud. $\mathbf{Z}^{\mathrm{depth}}$ and $\mathbf{Z}^{\mathrm{photo}}$ forms a vectorized image from the pixel intensities corresponding to the error values of $\mathbf{E}_{\mathrm{depth}}$ and $\mathbf{E}_{\mathrm{photo}}$ respectively. $\mathbf{W}_{\mathrm{depth}}$ and $\mathbf{W}_{\mathrm{photo}}$ are the diagonal weighting matrix derived from $\mathbf{E}_{\mathrm{depth}}$ and $\mathbf{E}_{\mathrm{photo}}$ using Tukey based m-estimator [15]. The combined error matrix $\mathbf{Z}_{\boldsymbol{\Pi}}$ on the image plane is obtained by:

$$\mathbf{Z}_{\boldsymbol{\Pi}} = \Big\langle \mathrm{vec}^{-1}_{\mathrm{H}\times\mathrm{W}}\Big(\mathbf{Z}^{\mathrm{depth}}\Big)\Big\rangle \odot \Big\langle \mathrm{vec}^{-1}_{\mathrm{H}\times\mathrm{W}}\Big(\mathbf{Z}^{\mathrm{photo}}\Big)\Big\rangle \tag{13}$$

where $\odot$ denotes the Hadamard product and $\langle\cdot\rangle$ gives the normalized matrix. $\mathrm{vec}^{-1}_{\mathrm{H}\times\mathrm{W}}(\cdot)$ denotes the vector to matrix map $\mathbb{R}^{\mathrm{HW}} \mapsto \mathbb{R}^{\mathrm{H}\times\mathrm{W}}$, where H and W are the height and width image respectively. A median blur with 3×3 sized

kernel is applied on $\mathbf{Z_\Pi}$ followed by a linear thresholding and the output of this operation is clustered into multiple clusters. The centroid of each cluster is associated with the nearest projection of the visible vertices of $\mathcal{K}$ on the image plane. These associated vertex indices of $\mathcal{K}$ are identified as the *control handles* for that particular frame.

E. Non-rigid Error Minimization

Assuming that an estimate of ${}^{p}\mathbf{T}_{p-1}$ has already been obtained (using the method described in Sec. III-F), a combination of depth based geometric error and direct photometric error is minimized to track the deforming object. We now define the error term that needs to be minimized for a single point $\mathbf{P}$ of the pointcloud at the p-th data frame. The point-to-plane distance based geometric error is given by:

$$\mathbf{E}_{\text{depth}}(\mathbf{P}^{p}) = \mathbf{n}_j \cdot \mathbf{P}^{p} - d_j \tag{14}$$

assuming that the j-th surface plane of $\mathcal{S}$ (where $\mathbf{n}_j$ is the normal and d_j is distance to origin) corresponds with the point $\mathbf{P}$ on the image plane. We propose a photometric error term defined on the (p-1)-th frame by:

$$\mathbf{E}_{\text{photo}}(\mathbf{P}^{p-1}) = \boldsymbol{\mathcal{I}}_{p}(\mathbf{P}_{e}^{p-1}) - \boldsymbol{\mathcal{I}}_{p-1}(\mathbf{P}^{p-1}) \tag{15}$$

The updated point position $\mathbf{P}_{e}^{p-1}$ is determined by a barycentric map $\mathbf{P}_{e}^{p-1} = \begin{bmatrix}(\mathbf{P}_{i}^{\mathcal{S}})' & (\mathbf{P}_{i+1}^{\mathcal{S}})' & (\mathbf{P}_{i+2}^{\mathcal{S}})'\end{bmatrix}\mathbf{B}$ such that $\mathbf{P}^{p-1}$ corresponded with the triangle $\begin{pmatrix}\mathbf{P}_{i}^{\mathcal{S}} & \mathbf{P}_{i+1}^{\mathcal{S}} & \mathbf{P}_{i+2}^{\mathcal{S}}\end{pmatrix}$ and $(\mathbf{P}_{i}^{\mathcal{S}})'$ gives the updated vertex position of $\mathbf{P}_{i}^{\mathcal{S}}$ when subject to the update $\boldsymbol{\vartheta}$, given that $\mathbf{B}$ is a column vector denoting the barycentric coordinates of $\mathbf{P}^{p-1}$ w.r.t $\mathbf{P}_{i}^{\mathcal{S}}, \mathbf{P}_{i+1}^{\mathcal{S}}$ and $\mathbf{P}_{i+2}^{\mathcal{S}}$.

Eqn. 3 can be slightly modified to:

$$\mathbf{E}_{\mathcal{S}} = \begin{bmatrix}\mathbf{E}_{\text{depth}} & \mu\mathbf{E}_{\text{photo}}\end{bmatrix}^{\top} \tag{16}$$

as the combined cost function, where $\mu = \beta\frac{\|\mathbf{E}_{\text{depth}}\|}{\|\mathbf{E}_{\text{photo}}\|}$ is used for bringing the geometric and photometric error terms to the same scale and $0.9 \leq \beta \leq 1.2$ is a tunable parameter for weighting the relative influence of the photometric cost function.

The combined cost function that we seek to minimize is Eqn. (16) w.r.t $\mathbf{u}_C$, the displacement of the control handle. The Jacobian relating the change of $\mathbf{u}_C$ to the change of $\mathbf{E}_{\mathcal{S}}$ is obtained by utilizing Eqn. (10) and Eqn. (11) in Eqn. (5), such that:

$$\mathbf{J} = \frac{\partial\mathbf{E}_{\mathcal{S}}}{\partial\mathbf{u}_C} = \begin{bmatrix}\frac{\partial\mathbf{E}_{\text{depth}}}{\partial\boldsymbol{\xi}}\boldsymbol{\Gamma} & \mu\frac{\partial\mathbf{E}_{\text{photo}}}{\partial\boldsymbol{\xi}}\boldsymbol{\Gamma}_{\text{C}}\end{bmatrix}^{\top} \tag{17}$$

where:

$$\frac{\partial\mathbf{E}_{\text{depth}}}{\partial\boldsymbol{\xi}} = \begin{bmatrix}-\mathbf{n}_j^{\top} - \mathbf{n}_l^{\top}\big(\mathbf{A}\big[\mathbf{P}_{i+2}^{\mathcal{S}} - \mathbf{P}_{i+1}^{\mathcal{S}}\big]_{\times}\big) \\ -\mathbf{n}_l^{\top}\big(\mathbf{A}\big[\mathbf{P}_{i}^{\mathcal{S}} - \mathbf{P}_{i+2}^{\mathcal{S}}\big]_{\times}\big) \\ -\mathbf{n}_l^{\top}\big(\mathbf{A}\big[\mathbf{P}_{i+1}^{\mathcal{S}} - \mathbf{P}_{i}^{\mathcal{S}}\big]_{\times}\big)\end{bmatrix}^{\top} \tag{18}$$

given that:

$$\mathbf{A} = \frac{1}{\|\overline{\mathbf{n}}_j\|}(\mathbf{I}_{3\times3} - \mathbf{n}_j\mathbf{n}_j^{\top}) \tag{19}$$

provided $(\cdot)_{\times}$ gives the skew-symmetric matrix and:

$$\overline{\mathbf{n}}_j = (\mathbf{P}_{i+2}^{\mathcal{S}} - \mathbf{P}_{i}^{\mathcal{S}}) \times (\mathbf{P}_{i+1}^{\mathcal{S}} - \mathbf{P}_{i}^{\mathcal{S}}) \tag{20}$$

and $\mathbf{n}_l = \mathbf{P}_{l}^{p} - \mathbf{P}_{i}^{\mathcal{S}}$. On the other hand:

$$\frac{\partial\mathbf{E}_{\text{photo}}}{\partial\boldsymbol{\xi}} = \begin{bmatrix}\nabla\boldsymbol{\mathcal{I}}_{p}^{u} \\ \nabla\boldsymbol{\mathcal{I}}_{p}^{v}\end{bmatrix}^{\top}\begin{bmatrix}b_1 & 0 \\ 0 & b_1 \\ b_2 & 0 \\ 0 & b_2 \\ b_3 & 0 \\ 0 & b_3\end{bmatrix}^{\top}\begin{bmatrix}\mathbf{G}_{\mathbf{P}^{\mathcal{S}}}^{i} & & \\ & \mathbf{G}_{\mathbf{P}^{\mathcal{S}}}^{i+1} & \\ & & \mathbf{G}_{\mathbf{P}^{\mathcal{S}}}^{i+2}\end{bmatrix} \tag{21}$$

where:

$$\mathbf{G}_{\mathbf{P}^{\mathcal{S}}}^{q} = \begin{bmatrix}\frac{f_x}{(\mathbf{P}_{q}^{\mathcal{S}})_{\text{Z}}} & 0 & -f_x\frac{(\mathbf{P}_{q}^{\mathcal{S}})_{\text{X}}}{(\mathbf{P}_{q}^{\mathcal{S}})_{\text{Z}}^2} \\ 0 & \frac{f_y}{(\mathbf{P}_{q}^{\mathcal{S}})_{\text{Z}}} & -f_y\frac{(\mathbf{P}_{q})_{\text{Y}}}{(\mathbf{P}_{q}^{\mathcal{S}})_{\text{Z}}^2}\end{bmatrix}\forall q \in i, (i+1), (i+2) \tag{22}$$

and $\mathbf{B} = \big(b_1, b_2, b_3\big)$ are the barycentric coordinates.

The optimization method of our choice is Levenberg-Marquadt (LM) like [16], and the update is given by:

$$\delta\mathbf{u}_{\text{C}} = -(\mathbf{H} + \lambda\mathbf{I}_{\text{N}\times\text{N}})^{-1}\mathbf{J}^{\top}\mathbf{W}^{\top}\mathbf{W}\mathbf{E}_{\mathcal{S}} \tag{23}$$

where $\mathbf{H} = \mathbf{J}^{\top}\mathbf{W}^{\top}\mathbf{W}\mathbf{J}$ is the approximation of the Hessian and $\mathbf{W} = \text{diag}\big(\mathbf{W}_{\text{depth}}, \mathbf{W}_{\text{photo}}^{\mu}\big)$ where $\mathbf{W}_{\text{depth}}$ and $\mathbf{W}_{\text{photo}}^{\mu}$ is obtained using the Tukey m-estimator on $\mathbf{E}_{\text{depth}}$ and $\mu\mathbf{E}_{\text{photo}}$ respectively. λ is a scaling factor and N is the size of the error vector $\mathbf{E}_{\mathcal{S}}$. The new position $\mathbf{u}_{\text{C}}+\delta\mathbf{u}_{\text{C}}$ of $\mathbf{C}_j$ is to be applied on the mechanical model such that it deforms to minimize the errors from (14) and (15). Apart from α in Eqn. (11), β related to Eqn. (16) and λ (and excluding the clustering mechanism of Sec. III-D), there are no parameters involved in the non-rigid visual tracking approach presented here.

F. Initialization

The rigid pose of the object in the camera's reference frame is updated at the beginning of each frame using an approach similar to [17], by jointly minimizing a geometric error and a sparse feature based error, given by:

$$\mathbf{e}^{\text{D}}({}^{p}\mathbf{v}_{p-1}) = \Big(\big({}^{p}\mathbf{R}_{p-1}\mathbf{P}^{p-1} + {}^{p}\mathbf{t}_{p-1}\big)\cdot\mathbf{n}_j\Big) - d_j \tag{24}$$

$$\mathbf{e}^{\text{K}}({}^{p}\mathbf{v}_{p-1}) = \begin{pmatrix}\boldsymbol{\Pi}(\mathbf{P}^{p-1})_x - x^* \\ \boldsymbol{\Pi}(\mathbf{P}^{p-1})_y - y^*\end{pmatrix} \tag{25}$$

where $\mathbf{P}^{p-1}$ represents an arbitrary 3D point from the last data frame, which has been matched to the j-th plane of the object's 3D model, denoted by the normal vector $\mathbf{n}_j = (n_j^{\text{X}}, n_j^{\text{Y}}, n_j^{\text{Z}})$ and distance to origin d_j. $\big(\boldsymbol{\Pi}(\mathbf{P}^{p-1})_x, \boldsymbol{\Pi}(\mathbf{P}^{p-1})_y\big)$ are the projection of the same arbitrary point in the (p-1)-th image while (x^*, y^*) are the same image points matched in the p-th image using Harris corner features. ${}^{p}\mathbf{v}_{p-1} \in \text{se}(3)$ gives the velocity twist between the $(p-1)$-th frame and p-th frame.

G. Force Tracking

Next, the method of Sec. III-C - III-E is used to estimate the deforming forces acting on an object. Two additional information are required for force tracking: the material properties of the object being tracked, i.e., Young's modulus, Poisson's ratio, Rayleigh stiffness etc. and the approximate point of contacts on the object. The force tracking experiments have been performed by tracking the tool applying

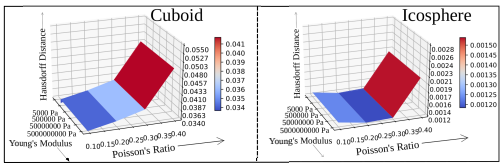

Fig. 1: Variance of $\mathcal{H}$ with Y and σ for the two simulated objects, *cuboid* and *icosphere* (color coded with the value of $\mathcal{H}$)

		Cube	Icosphere
SF	Constant $\mathbf{\Gamma}$	5.76 %	**4.83** %
	Updated $\mathbf{\Gamma}$	5.75 %	5.42 %
CRF	Constant $\mathbf{\Gamma}$	**2.14** %	5.25 %
	Updated $\mathbf{\Gamma}$	**2.14** %	5.34 %
[20]		2.93 %	7.93 %
[7]		3.28 %	6.67 %

TABLE I: Tracking accuracy for two synthetic sequences in terms of Hausdorff distance between tracking output and GT

the force using fiducial markers. Once a point $\mathbf{P}_\mathrm{P}$ have been identified as the point of contact, its nearest mechanical mesh vertex $\mathbf{Q}_\mathrm{P}$ is obtained using a nearest neighbor search. Thereafter, the force applied on $\mathbf{P}_\mathrm{P}$ is given as $\left\|\sum_i \mathbf{F}_{\mathbf{Q}_i}\right\|$, for all indices i such that $\mathbf{Q}_i$ is a neighbor of $\mathbf{Q}_\mathrm{P}$, wherein the force vectors $\mathbf{F}_{\mathbf{Q}_i}$ is derived using (2).

IV. Results

The results can be roughly divided into three categories:

- For the simulated objects with groundtruth, we focus on comparing the fundamental concept of the non-rigid tracking approach proposed here with comparable state-of-the-art methods. We also use challenging simulated sequences to quantitatively validate the robustness of our approach;
- For deformation tracking on real data, we emphasize the capability of our algorithm to accurately track large volumetric deformations in soft objects;
- We provide experimental validation of the force tracking approach proposed in Sec. III-G

A. Simulated Results: Validation with Groundtruth

We base our simulated results on two objects[1], a *cuboid* and an *icosphere*, as shown in Fig. 3 and Fig. 2. The simulated data is generated using the Blender software [18] and the object deformations are generated manually using harmonic coordinates with B-spline basis [19]. Some quantitative results on synthetic sequences are expressed as a percentage of the largest diagonal of the bounding box of the objects (2.884m for the *cuboid* and 3.408m for the *icosphere*), and hence the values are unitless.

First, we demonstrate that the difference in accuracy between maintaining the *influence matrix* $\mathbf{\Gamma}$ constant (C. $\mathbf{\Gamma}$) and re-computing and updating $\mathbf{\Gamma}$ (U. $\mathbf{\Gamma}$) at each frame numerically is not significant. To establish this proposition, we run tests on the synthetic data with and without holding $\mathbf{\Gamma}$ constant for both standard, linear FEM (SF) and co-rotational FEM (CRF). The results, summarized in Table I, are expressed in terms of the Hausdorff distance ($\mathcal{H}$) from groundtruth (GT). It can be clearly seen that the variation in accuracy of tracking is between 0 to 0.59 % for both the sequences tested here. This comes in exchange for a large improvement in runtime (upto $> 53\%$ improvement in per frame time requirement, see Sec. IV-D for time requirements of the implementation) of the entire algorithm, since multiple FEM simulations per iteration of LM is highly expensive. We also use these sequences for comparing the proposed approach to [7] and our re-implementation of [20]. The corotational FEM based version of the proposed approach outperforms [20] and [7] in both the sequences. Fig. 2 shows the visual comparison of all the approaches tested on the synthetic sequences.

Using a specific *control handle* and maintaining all other parameters constant, the variation of the tracking accuracy (in terms of $\mathcal{H}$) of the method proposed in this paper w.r.t the Young's modulus (Y) and Poisson's ratio (σ) is shown in Fig. 1. It is clear that the tracking accuracy do not vary significantly with change in Young's modulus. This observation is in accordance with [7]. However, the tracking accuracy varies significantly with change in Poisson's ratio. This is because materials with Poisson's ratio in the range of $0.35 - 0.45$ are highly ductile in nature and hence significantly harder to track with.

Fig. 3 shows the output of the proposed approach on two challenging deformation sequences on *cuboid* and *icosphere*. The two objects were subjected to large volumetric deformation and were occluded using large floating objects in the synthetic scene. The mean value of $\mathcal{H}$ is $0.0696m$ for the *cuboid* and $0.0836m$ for the *icosphere*, while the value of $\mathcal{H}$ for every frame of the sequence if shown in Fig. 4.

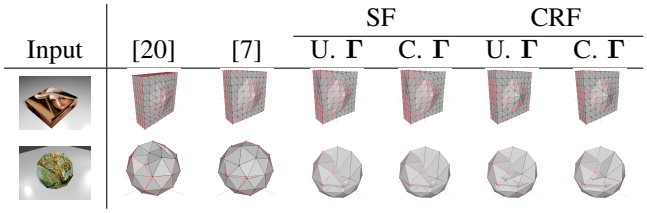

Fig. 2: Comparison between GT and tracking output for the two synthetic sequences *cube* and *icosphere*. The red edges and vertices shows the object model from the GT, the black ones are from the tracking output.

B. Experiments on Real Data

The real data has been captured using an Intel RealSense D435 camera. Three objects were used, a block of *sponge*, a rugby *ball* (cut in half) and a soft *dice*. These objects were strongly deformed from the top. The output of tracking has been demonstrated visually[2] in Fig. 5 while the evolution of point-to-plane distance across the two sequences have been logged in Fig. 6. The *sponge* and *ball* demonstrates highly

[1]All data available at: github.com/icra2021/VisualDeformationTracking

[2]For a detailed video of the results, please visit: youtu.be/ScJnz_j4-cs

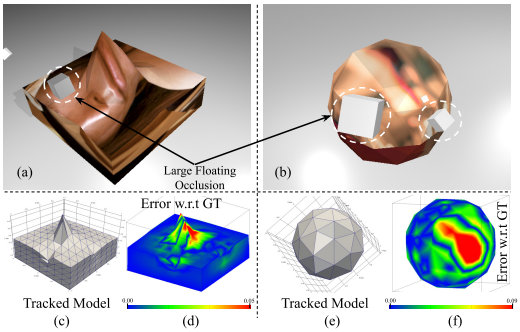

Fig. 3: The *cuboid* and *icosphere* are subjected to large deformation ((a) & (b)) with large, floating objects flying across the scene (as occlusion). The tracking output is shown in (c) and (e) while $\mathbf{E}_{\text{depth}}$ is shown on the surface of $\mathcal{S}$ in (d) and (f)

accurate tracking with a mean geometric error (from Eqn. 14) of -0.81 mm and -0.18 mm respectively, with a standard deviation of 0.47 mm and 0.62 mm. This accuracy is highly robust to occlusion and demonstrates the suitability of the proposed approach to robotic manipulation of soft objects. The *dice* is a slightly smaller deformation, but shows a high mean accuracy of ~ 0.61 mm.

C. Force Tracking

To validate the force tracking methodology, we use a 6-DOF anthropomorphic robot arm (Viper 850 from ADEPT) fitted with a ATI Gamma IP65 force/torque sensor and a 3D-printed stylus as an end-effector distal tool. The robot is used only to utilize its force sensor to obtain a GT of the force. We use the *sponge* and the *ball* to validate the force tracking method. The Young's modulus of the *sponge* and the *ball* is determined by repeated indentation tests and is found to be 460 kPa and 160 kPa respectively. These objects are subjected to a strong deformation using the robot's 3D printed end-effector and the point of contact on the object is tracked using pre-trained markers (fitted to the probe). The results of the force tracking approach, as summarized in Fig. 7, shows an accuracy of $\sim 97\%$ for the *sponge* and of $\sim 90\%$ for the *ball* when the deformation is optimally observable.

D. Implementation

The approach proposed in this paper has been implemented in C++ on an Intel Xeon CPU working at 3.70 GHz. Utilizing only a single core of the computer (without using GPU), the un-optimised code was able to achieve a runtime of 350 ms - 550 ms per frame while tracking deformations using the proposed approach, showing that it can be possible to achieve real-time performance at frame rate.

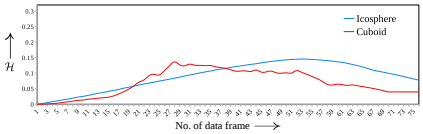

Fig. 4: The value of $\mathcal{H}$ (in m) across all the frames of the two sequences shown in Fig. 3

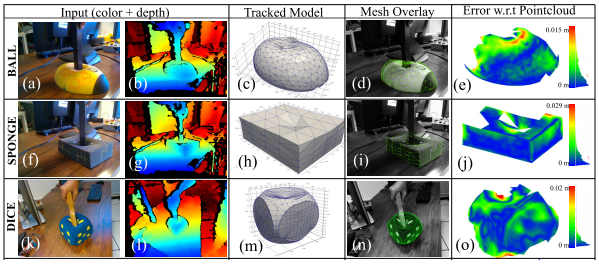

Fig. 5: The color and depth image input for the *ball*, *sponge* and *dice* sequences are shown in (a), (b), (f), (g), (k) & (l). The tracked model are shown in (c), (h) & (m). (d), (i) & (n) shows the object model overlaid on the image, while (e), (j) & (o) shows the geometric error for the visible surfaces, derived using [21]

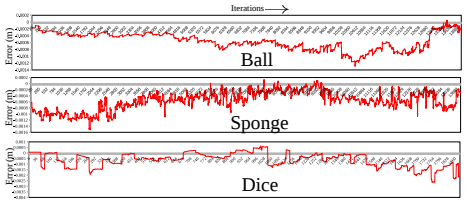

Fig. 6: Value of the weighted geometric error, i.e., $\|\mathbf{W}_{\text{depth}}\mathbf{E}_{\text{depth}}\|$ across the iterations of optimization for the sequences in Fig. 5

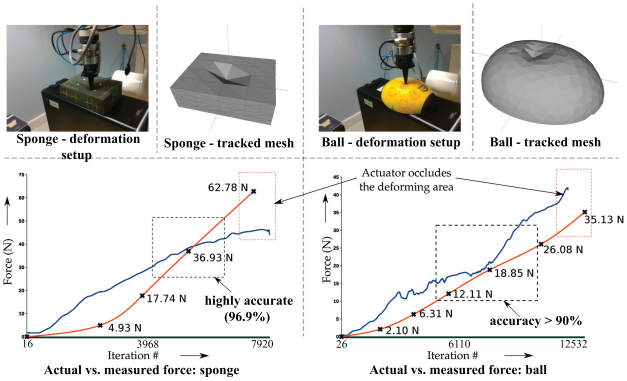

Fig. 7: Setup and results for force tracking. The red plot is the force measurement from the robot's force sensor and the blue plot is the estimated force reported by the proposed approach

V. Conclusion

The paper presented here describes a new method for combining geometric and photometric visual error minimization with FEM to create an accurate and fast deformation tracking method. The algorithm has been tested on synthetic and real data and has been shown to outperform state-of-the-art methods in tracking accuracy for generic deforming objects. The algorithm proposed here can be extended to other, more complex physical models without loss of generalization. The method proposed here has been demonstrated as a reliable visual force tracking system, when the material properties of the object being tracked is available.

Acknowledgement

We gratefully acknowledge the support from The Research Council of Norway through participation in GentleMAN (299757) project.

REFERENCES

[1] Sebastian Hoppe Nesgaard Jensen, Alessio Del Bue, Mads Emil Brix Doest, and Henrik Aanæs. A benchmark and evaluation of non-rigid structure from motion. *arXiv preprint arXiv:1801.08388*, 2018.

[2] John Schulman, Alex Lee, Jonathan Ho, and Pieter Abbeel. Tracking deformable objects with point clouds. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1130–1137, 2013.

[3] Bin Wang, Longhua Wu, KangKang Yin, Uri M Ascher, Libin Liu, and Hui Huang. Deformation capture and modeling of soft objects. *ACM Transactions on Graphics (TOG)*, 34(4):94–1, 2015.

[4] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 343–352, 2015.

[5] Shaifali Parashar, Daniel Pizarro, Adrien Bartoli, and Toby Collins. As-rigid-as-possible volumetric shape-from-template. In *Proceedings of the IEEE International Conference on Computer Vision (CVPR)*, pages 891–899, 2015.

[6] Antoine Petit, Vincenzo Lippiello, and Bruno Siciliano. Real-time tracking of 3d elastic objects with an rgb-d sensor. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3914–3921, 2015.

[7] Agniva Sengupta, Alexandre Krupa, and Eric Marchand. Tracking of non-rigid objects using rgb-d camera. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 3310–3317, 2019.

[8] Agniva Sengupta, Romain Lagneau, Alexandre Krupa, Eric Marchand, and Maud Marchal. Simultaneous tracking and elasticity parameter estimation of deformable objects. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 10038–10044, 2020.

[9] A. Nealen, M. Muller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. *Computer Graphics Forum*, Vol 25:809–836, 2006.

[10] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlik, et al. Sofa: A multi-model framework for interactive physical simulation. In *Soft tissue biomechanical modeling for computer assisted surgery*, pages 283–321. 2012.

[11] M. Muller and M. Gross. Interactive virtual materials. In *Proceedings of Graphics interface*, pages 239–246. Canadian Human-Computer Communications Society, 2004.

[12] Carlos A Felippa. A systematic approach to the element-independent corotational dynamics of finite elements. Technical report, Technical Report CU-CAS-00-03, Center for Aerospace Structures, 2000.

[13] Michael Friswell and John E Mottershead. *Finite element model updating in structural dynamics*, volume 38. Springer Science & Business Media, 2013.

[14] Yang Chen and Gérard G Medioni. Object modeling by registration of multiple range images. In *IEEE International Conference on Robotics and Automation (ICRA)*, 1991.

[15] DA Freedman and Persi Diaconis. On inconsistent m-estimators. *The Annals of Statistics*, pages 454–461, 1982.

[16] Jorge J Moré. The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer, 1978.

[17] Souriya Trinh, Fabien Spindler, Eric Marchand, and François Chaumette. A modular framework for model-based visual tracking using edge, texture and depth features. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 89–96. IEEE, 2018.

[18] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018.

[19] Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic coordinates for character articulation. *ACM Transactions on Graphics (TOG)*, 26(3):71–es, 2007.

[20] Antoine Petit, Vincenzo Lippiello, Giuseppe Andrea Fontanelli, and Bruno Siciliano. Tracking elastic deformable objects with an rgb-d sensor for a pizza chef robot. *Robotics and Autonomous Systems*, 88:187–201, 2017.

[21] Cloudcompare (version 2.11.1) [gpl software]. Retrieved from `http://www.cloudcompare.org/`, 2020.