Complex & Intelligent Systems
https://doi.org/10.1007/s40747-024-01453-9

ORIGINAL ARTICLE

Improved shape-from-template method with perspective space constraints for disappearing features

Dongliang Tan[1] · Huamin Yang[1] · Zhengang Jiang[1] · Weili Shi[1] · Jun Qin[1] · Feng Qu[1]

Received: 24 December 2023 / Accepted: 20 April 2024

Abstract

Inferring the 3D surface shape of a known template from 2D images captured by a monocular camera is a challenging problem. Due to the severely underconstrained nature of the problem, inferring shape accurately becomes particularly challenging when the template exhibits high curvature, resulting in the disappearance of feature points and significant differences between the inferred and actual deformations. To address this problem, this paper proposes a concise and innovative approach that utilizes a physical simulator incorporating the object's material properties and deformation law. We utilize a view frustum space constructed from the contours of a monocular camera image to effectively restrict the physically-based free motion of the template. Additionally, we employ mesh denoising techniques to ensure the smoothness of the surface following deformation. To evaluate our shape inference results, we utilize a ground truth 3D point cloud generated from multiple viewpoint images. The results demonstrate the superior performance of our approach compared to other methods in accurately inferring deformations, particularly in scenarios where feature points are unobservable. This method carries significant practical implications across diverse domains, including virtual reality, digital modeling, and medical surgery training.

Keywords Shape-from-Template · Computer Vision · Virtual Reality · Digital Modeling

Introduction

Monocular non-rigid 3D reconstruction from single 2D images possesses significant applications in augmented reality [1], robot vision [2], and computer-assisted surgery [3, 4]. However, solutions in this domain are yet to be fully developed. This is largely attributable to the inherent difficulty in recovering the shape's depth from a 2D image. One well-studied method, known as Shape-from-Template (SfT) [5], demonstrates promising outcomes in resolving the depth of isometrically deformed objects. SfT utilizes a range of priors and constraints, such as the object's 3D rest shape, texture map, and camera intrinsics, to infer the 3D shape of deformable objects from a single input image.

The conventional SfT method is principally composed of two main components: registration and 3D shape inference. One approach involves assuming that the template experiences solely isometric or conformal deformations [6, 7], which facilitates an analytical solution to SfT through the resolution of the corresponding PDEs.

Deep learning, a subset of machine learning techniques focused on learning data representations, has recently extended its applications beyond computer vision and image processing to encompass control and optimization. Wang et al. [8] introduced a Q-learning based fault estimation (FE) and fault tolerant control (FTC) scheme within an iterative learning control (ILC) framework. Song et al. [9] presented a switching-like event-triggered strategy (SETS) to address intermittent Denial-of-Service (DoS) attacks. As an optimizable approach, learning-based SfT methods have been introduced. Navami et al. [10] introduced ϕ-SfT, incorporating a physics-based rendering loss term to refine the dynamic process of a template characterized by physical attributes. David et al. [11] substituted the conventional physics-based simulation in ϕ-SfT with neural surrogate models, drastically diminishing the computational time required for the optimization process from several hours to only a few minutes per scene.

Subsequently, existing SfT methods are classified into three distinct categories: shape inference methods, integrated methods, and Deep Neural Network (DNN) based SfT meth-

✉ Huamin Yang
yhm@cust.edu.cn

[1] School of Computer Science and Technology, Changchun University of Science and Technology, Changchun, China

Published online: 08 May 2024

ods. Most shape inference methods feature a wide baseline yet seldom run in real-time due to the absence of registration between the static template and the warped image. In contrast, integrated methods, which require an initialization close to target deformation, cover both the registration and shape inference parts. This characteristic makes them suitable for short-baseline scenarios , yet it requires more computational resources. Moreover, conventional SfT primarily depends on the extraction and matching of feature points. Consequently, significant deformation or bending of the template may render shape inference susceptible to severe distortion. Compared to other methods, learning-based SfT methods exhibit a wide baseline, yet their real-time performance is contingent upon the scale of the deep network employed. Furthermore, Learning-based SfT methods are object-specific, necessitating substantial training data and adequate computational resources, thus limiting their applicability as a general solution when the deformable object varies. Recently, a model unknown matching network model [12] has demonstrated significant potential in few-shot learning, targeting the resolution of issues related to sparse fault samples and cross-domain between data sets in real industrial situations.

This paper introduces an improved Shape-from-Template (SfT) approach aimed at addressing particular shortcomings of existing state-of-the-art methods, thereby significantly enhancing the accuracy of monocular non-rigid 3D reconstruction. We employ a conventional SfT method to tackle the challenge of tracking the 3D shape of deforming objects. Our approach takes a 640 × 480 image, captured using standard consumer hardware, as input and produces an output that represents the consistent 3D shape of the object depicted in the image. In contrast to previous methods [6, 7, 13, 14], the non-convex SfT problem is not addressed through initialization or refinement in this work. Based on actual observations, a constrained physical simulator has been incorporated. The simulator supplants geometric techniques in traditional SfT workflows and rectifies errors arising from iterative resolutions through non-physical position-based dynamics methods, as employed in [14]. By integrating a simulator characterized by parameters of tangible physical significance, we not only enhance shape inference accuracy but also improve the convergence rate of numerical solutions and scalability across various deformable objects. We are not the first to incorporate physical priors into the SfT problem. Works like ϕ-SfT [10] and Physics-guided Shape-from-Template [11] optimize the SfT solution by using differentiable rendering techniques to optimize physical parameters. In contrast, we introduce perspective space constraints for 3D objects, derived from contour information in the input image, to constrain the template's range of motion. An increased amount of constraint information on the input [15] facilitates convergence rates and enhances the precision of solutions to optimization problems. These constraints significantly reduce shape inference errors arising when feature points become unobservable.

The main contributions of this study are summarized as follows:

1. A novel approach is proposed to address SfT, characterized by the integration of a physical simulator. This approach enhances shape inference precision and accelerates numerical solution convergence by integrating material properties into the deformable template and executing dynamic simulations.
2. Perspective space constraints, derived from the contours of a monocular camera image, are introduced to restrict the template's physics-based free motion. This significantly reduce deformation errors attributable to the absence of feature points during extensive bending.

Related work

This section provides a comprehensive review of methods employed in monocular non-rigid 3D reconstruction of isometrically deforming objects. These methods differ with respect to their input requirements, prior knowledge, and deformation models. They are categorized into three groups: shape inference methods, integrated methods, and DNN-based SfT methods. For each category, the underlying assumptions, key characteristics, and limitations associated with the methods are described.

Shape inference methods are predicated on the assumption that the registration between the template and the input image is precomputed. The accuracy of this registration step is crucial for the subsequent shape inference. These methods frequently utilize keypoint matches and employ mismatch removal techniques, leveraging the known templates and textures present in the image. Bartoli et al. [16] were the first to study the problem known as Shape-from-Template and introduced a novel class of methods termed first-order methods. Pizarro et al. [17] addressed the issue of outliers and stability improvement by assuming local smoothness of the surface to be detected. Chhatkuli et al. [18] solved SfT by estimating depth gradients or surface normals and integrating them to obtain the shape. Their method surpassed previous uninitialized methods in terms of accuracy. Özgür et al. [14] introduced Particle-SfT, a novel Shape-from-Template (SfT) algorithm capable of handling both isometric and non-isometric deformations. This method leverages position based dynamics and sight-line constraints to estimate the depth information of 3D objects. It has been demonstrated to exhibit high accuracy and convergence. Casillas-Perez et al. [6] established a theoretical framework for equiareal

SfT and demonstrated the feasibility of reconstructing a surface exactly with a much weaker prior than isometry. Salzmann et al. [19] demonstrated that reconstruction could be accomplished by resolving a system of quadratic equations, representing distance constraints between adjacent vertices of a mesh. Moreover, Salzmann et al. subsequently established that incorporating inequality constraints instead of equality constraints in earlier methods not only results in more accurate representations but also yields convex formulations for reconstruction problems [20]. Famouri et al. [21] utilized the estimation of affine transformations, employing the nearest neighbors of keypoint pairs. They then estimated the depth of each keypoint in the deformed image based on its associated affine transformation. Aranda et al. [22] proposed a template-based shape servoing scheme. The template enables both the inference of the object's shape using an enhanced Shape-from-Template algorithm and the steering of the object's deformation through the robots' movements. Brunet et al. [23] addressed two significant limitations of the current state-of-the-art methods. Firstly, they introduced convex methods capable of handling noise in both the template and image points. Second, they proposed a non-convex method that incorporates "true" isometric constraints. Malti et al. [24] introduced a stretching energy formulation that incorporates the Poisson ratio parameter of the surface. This formulation unifies both geometric and mechanical constraints into a single energy term. The deformation is then addressed by optimizing this energy term.

Integrated methods encompass both registration and shape inference. Ngo et al. [25] utilized corresponding feature points from the reference and input images to reformulate the shape inference problem into an image recovery problem. Collins et al. [26] introduced a real-time Shape-from-Template (SfT) framework that aimed to address two key sub-problems: robust registration and 3D shape inference. They proposed the Deformable Render-based Block Matching (DRBM) method, designed to globally search for matching pixels between template textures and input images. The pixel matching relationship is subsequently transformed into a spatial coordinate constraint problem, with shape inference being achieved through the efficient solution of a sparse linear system. However, our experiments revealed that DRBM exhibits instability under varying lighting conditions. This method was subsequently applied to 3D organ tracking in laparoscopic videos [27]. Ostlund [28] introduced a novel approach for parameterizing the vertex coordinates of a mesh as a linear combination of a subset of them. Casillas-Perez et al. [7] proposed isowarp, marking a theoretical and practical breakthrough in SfT that imposes 3D geometric constraints on the warps, thereby outperforming the best existing reconstruction methods by using the analytic direct depth solution.

DNN-based SfT methods typically consist of object detection and shape inference components. The object detection module is commonly employed to extract the shape mask of the template object from the input image. However, numerous existing shape inference methods demonstrated limited adaptability when dealing with non-rigid deformations. Fuentes et al. [29] proposed the first texture-generic deep learning SfT method, which adapts to new texture maps at runtime without the need for texture-specific fine tuning. Subsequently, they integrated statistical learning with physics-based reasoning [30], achieving commendable generalization performance under conditions such as wide baseline, occlusion, illumination variations, weak texture, and blurring. Golyanik et al. [31] proposed HDM-Net, a network trained with various non-rigid deformation structures. HDM-Net proved effective in handling small and moderate isometric deformations. However, it exhibited significant errors when confronted with large deformations. Pumarola et al. [32] proposed Geometry-Aware Network that is made of three main branches: the 2D detection branch, the depth branch, and the shape branch. Due to the allocation of different stages of SfT to different sub-networks in this model, the 2D shape and texture features of the template can be effectively decoupled, making the network more generalized. Shimada et al. [33] proposed the Isometry-Aware Monocular Generative Adversarial Network (IsMo-GAN). This method fully utilizes 2D convolution to efficiently achieve 3D point cloud deformation prediction, but there are still significant errors in the reconstruction of complex deformation scenes.

Methods

We proposed a coarse-to-fine method for deformation inference when feature points become unavailable or disappear. The method comprises two key components: registration and inference. The pipeline of the proposed SfT method is illustrated in Fig. 1. During the registration phase, ROBUSfT is employed to carry out preliminary feature point extraction and mismatch removal [13]. In the deformation inference section, a physical simulator was designed based on perspective space constraints derived from the contours of a monocular camera image, enabling more accurate tracking of real physical deformations compared to the position-based dynamics (PBD) method [14, 34]. Furthermore, the complete pipeline incorporated Bicubic B-Spline (BBS) image warps [35] and bilateral mesh denoising [36]. Section "Background and notation" begins with a brief review of the relevant technologies and symbolic representations utilized in the method. Section "Constrained physics simulator" comprehensively presents the proposed physics-based simulator for deformation inference when feature points become unobservable. The

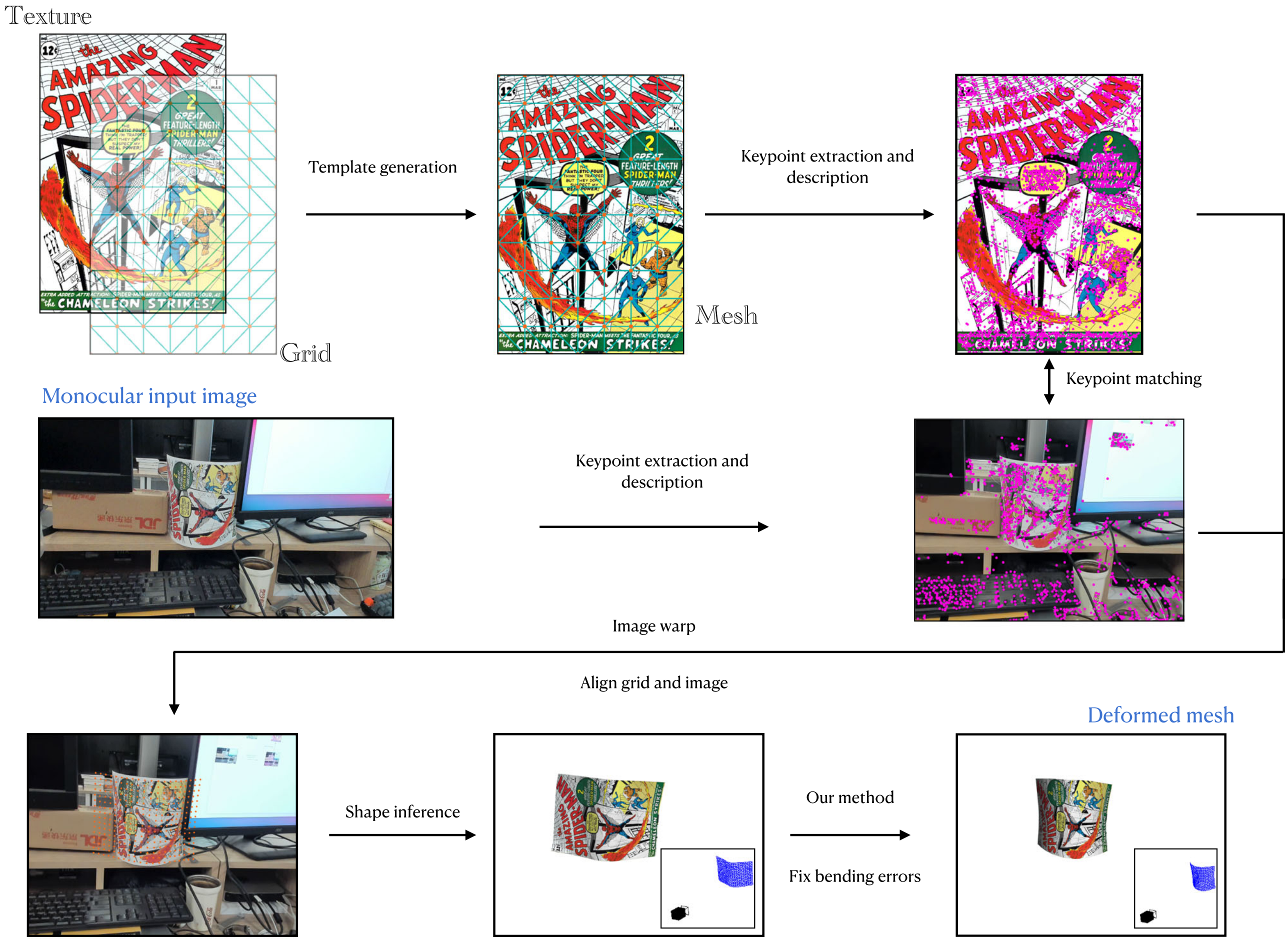


Fig. 1 Pipeline of the SfT method

system setup and implementation details are then described in section "Implementation".

Background and notation

Object segment We assume that the monocular camera remains static and that the segmentation masks for both the foreground object and the background of the captured image are accessible. In practice, the Segment Anything Model (SAM) [37] is utilized to segment the object from template images captured by the camera. Particularly in scenarios with a static background, the difference between the current image and the background image can be leveraged to achieve faster segmentation. In the proposed template deformation inference method, continuous contour information proves crucial. Consequently, fitting the collected discrete contour data based on pixels becomes necessary. The observed image captured by a perspective camera is denoted as I. The segmentation mask image derived from I is denoted as I_b, with pixel values set to one for pixels belonging to the template and zero otherwise. The centroid $(\bar{x}, \bar{y})$ of the contour region is calculated as follows:

$$\bar{x} = \frac{M_{10}}{M_{00}}, \quad \bar{y} = \frac{M_{01}}{M_{00}}, \tag{1}$$

where

$$M_{ij} = \sum_x \sum_y x^i y^j I_b(x, y). \tag{2}$$

A polar coordinate system is established with the pixel coordinates of the centroid as the origin. To convert the pixel coordinates (x, y) to polar coordinates (ρ, θ), a transformation is employed:

$$(\rho, \theta) = cartToPolar(x - \bar{x}, y - \bar{y}) \tag{3}$$

Contour fitting To interpolate and smooth the polar coordinate of the contour, the Moving Least Squares (MLS) method, as proposed by Lancaster and Salkauskas [38], is employed. The fundamental idea behind Moving Least Squares involves performing weighted least squares fitting across the entire parameter domain [39]. For any given θ, the objective is to minimize

$$\min \sum_i w(\theta - \theta_i) \| f(\theta_i) - \rho_i \|^2, \tag{4}$$

where w denotes a weighted function defined through Euclidean distance, and f represents a polynomial interpolation function, which can be expressed as

$$f(\theta) = b(\theta)^T c = b(\theta) \cdot c, \tag{5}$$

where $b(\theta) = [b_1(\theta), \ldots, b_k(\theta)]^T$ is the polynomial basis vector, and $c = [c_1, \ldots, c_k]^T$ is the vector of unknown coefficients. Once $f(\theta)$ is solved, the contour can be fitted by computing the polar diameter $\rho = f(\theta)$ at any given θ angle.

Image warp A $n_x \times n_y$ mesh of control points, denoted $\Phi_{i,j}$ with uniform spacing δ, is utilized to cover the template texture image I_s whose domain is $\Omega = \{(x, y) \mid 0 \leq x < X,\ 0 \leq y < Y\}$. Subsequently, the feature point $(x, y)^T$ in the image I_s is mapped to point $(\hat{x}, \hat{y})^T = (x+\Delta x, y+\Delta y)^T$ in the image I_t, captured by a monocular camera under the elastic deformation, and the offset $(\Delta x, \Delta y)^T$ can be expressed as the 2-D tensor product of the familiar 1-D Bicubic B-Spline

$$(\Delta x, \Delta y)^T = \sum_{l=0}^{3} \sum_{m=0}^{3} B_l(u) B_m(v) \phi_{i+l, j+m} \tag{6}$$

where $i = \lfloor x/\delta \rfloor - 1$, $j = \lfloor y/\delta \rfloor - 1$, $u = x/\delta - \lfloor x/\delta \rfloor$, $v = y/\delta - \lfloor y/\delta \rfloor$, and B_l denotes the lth basis function of the B-spline [40, 41]

$$\begin{aligned} B_0(u) &= (1-u)^3/6 \\ B_1(u) &= (3u^3 - 6u^2 + 4)/6 \\ B_2(u) &= (-3u^3 + 3u^2 + 3u + 1)/6 \\ B_3(u) &= u^3/6. \end{aligned} \tag{7}$$

In contrast to thin-plate splines [42], B-splines provide enhanced computational efficiency, particularly with a large number of control points.

Constrained physics simulator

The 3D template is discretized into a mesh comprising m vertices, aligning the vertices with the control points used for image warp, and the meshed template is modeled as a mechanical system, specifically a mass-spring system [43, 44]. In this system, the mass points correspond to the vertices of the mesh, and the springs represent the connections between the vertices. The position is defined as a function of time, $x = x(t)$, and velocity is subsequently defined as the derivative of position with respect to time, i.e., $v = v(t) = \dot{x}(t)$. Utilizing Taylor's series expansion to expand $x(t_n)$ around t_{n+1} enables the derivation

$$x(t_n) = x(t_{n+1}) - h \frac{\partial x(t_{n+1})}{\partial t_{n+1}} + \frac{h^2}{2} \frac{\partial^2 x(\xi_n)}{\partial \xi_n^2} \tag{8}$$

where $h = t_{n+1} - t_n$ is the time step size. Subsequently, by neglecting higher-order terms in Eq. (8), the following equation is derived:

$$x(t_{n+1}) = x(t_n) + h\dot{x}(t_{n+1}) \tag{9}$$

which is widely recognized as the implicit backward Euler method [43] within the realm of numerical integration techniques. Let $x_n \in \mathbb{R}^{3m}$ and $v_n \in \mathbb{R}^{3m}$ denote the system configuration vector representing the positions and velocities of all mass points at time t_n. Forces are assumed to be conservative, i.e., $f = -\nabla E$, where $f \in \mathbb{R}^{3m}$ represents the vector of internal and external forces acting on all mass points, and $E : \mathbb{R}^{3m} \rightarrow \mathbb{R}$ is a potential energy function. For a mass point x^i, connected to q springs, the energy and force are represented as follows:

$$\begin{aligned} E &= \sum_{e=1}^{q} \left(\frac{1}{2} k (\|x^i - x^e\| - L_e)^2 \right) \\ f_i(x) &= -\nabla_i E = \sum_{e=1}^{q} \left(-k (\|x^i - x^e\| - L_e) \frac{x^i - x^e}{\|x^i - x^e\|} \right) \end{aligned} \tag{10}$$

where k is the spring coefficient, and L_e represents the rest length of the spring connecting two vertices x^i and x^e. The system states that $x_1, x_2, \ldots, x_{n+1}$ can then be calculated and discretized as follows:

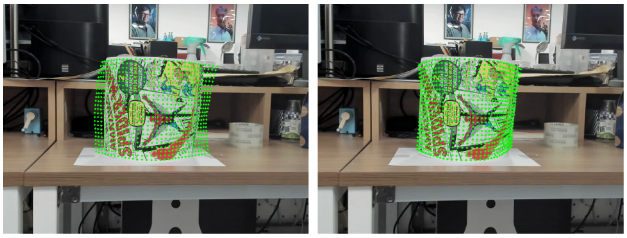
$$\begin{aligned} x_{n+1} &= x(t_n) + h\dot{x}(t_{n+1}) = x_n + h v_{n+1} \\ v_{n+1} &= v(t_n) + h\dot{v}(t_{n+1}) = v_n + h M^{-1} f(x_{n+1}) \end{aligned} \tag{11}$$

where $M \in \mathbb{R}^{3m \times 3m}$ is positive diagonal mass matrix. Subsequently, velocities v_{n+1} are eliminated from Eq. (11):

$$x_{n+1} = x_n + \Delta t v_n + \Delta t^2 M^{-1} f(x_{n+1}) \tag{12}$$

To solve the nonlinear system in Eq. (12), it is converted into an optimization problem, and the target function is constructed

$$F(x) = \frac{1}{2\Delta t^2} (x - x_n - \Delta t v_n)^T M (x - x_n - \Delta t v_n) + E(x), \tag{13}$$



(a) Before registration (b) After registration

Fig. 2 Registration of grids and contours

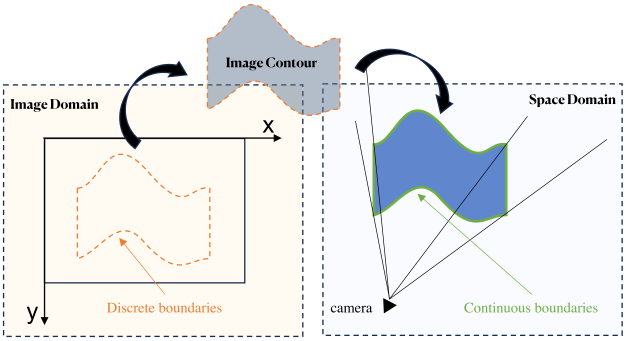


Fig. 3 The diagram shows the process of creating perspective space constraints from image contours

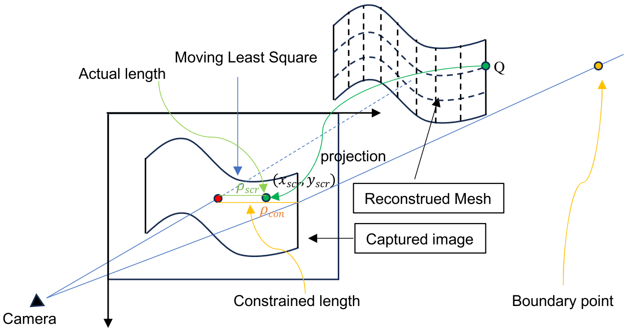


Fig. 4 The constraint position for each grid point

then solving Eq. (12) becomes equivalent to finding the extreme point of F in Eq. (13). To ensure computational efficiency, the fast mass-spring model [45] is employed as the fundamental physical simulator. The construction of constraints for physical simulators primarily comprises three steps.

Step 1: Register Grid and Contour. The initial step involves registering the projection of the grid $\mathcal{G}$ along the template's edge in image space with contour $\mathcal{C}$ of the image. Extracting precise feature points from the texture becomes challenging when the surface of the tracked template undergoes substantial bending deformation. Consequently, significant discrepancies may arise between the inferred deformation results and the input reference image. To address this, we employ BBS to transform the grid's projection into the contour range of the reference image. To compute the transformation parameters, peripheral grid points are utilized as the reference origin, and a ray is emitted towards the centroid of the image. The intersection point between the ray and the image contour serves as the essential matching point for the transformation, as illustrated in Fig. 2.

Step 2: Perspective Space Constraints. Following the initial registration transformation of the template's grid points in *step 1*, their accuracy in deformation inference is impacted. Depending on the types of errors that occur, they are primarily categorized into two distinct groups. The group located near the image's center, obtained through feature point deformation, exhibits higher accuracy. Conversely, the group near the image's edge, interpolated from the deformation displacement field parameterized by BBS, yields relatively lower accuracy. The grid points in these two groups exhibit distinct deformation accuracy characteristics. The grid's edge portion is relatively sparse, resulting in larger actual deformation errors. Conversely, the grid points located in the center of the image are densely distributed, resulting in smaller errors. In contrast to the previous methods, a novel physics simulator is introduced, relying on perspective space constraints derived from projected contours, as illustrated in Fig. 3. The perspective space constraints adapt dynamically in real-time to accommodate the deformation of the tracked surface. Deformed targets are identified and extracted from the camera-captured images, subsequently achieving pixel-level accuracy in contour extraction for these targets. The previously mentioned Moving Least Squares method is utilized to fit the discrete contour data. The fitting function is expressed as

$$fit\_contour(\theta) = mls(\boldsymbol{\theta}, \boldsymbol{\rho}, \theta), \qquad (14)$$

where $(\boldsymbol{\theta}, \boldsymbol{\rho}) = \{(\theta_i, \rho_i) | \theta_i, \rho_i \in \mathcal{C}\}$. To establish the constraint space, we initiate the process by starting from the fitted contour and connecting each point on the contour to the camera. Collectively, these rays form an irregular surface space that delineates the desired constraint boundaries. In practical scenarios, the deformation motion range for real templates remains uncertain. This uncertainty makes determining the exact constraint space range challenging, preventing the pre-saving of effective points constituting the boundary conditions.

To address this issue, real-time computations are performed to determine each grid point's constraint position, as shown in Fig. 4. For any given point $Q(X, Y, Z)$ on the grid, the corresponding pixel coordinates projected onto the imaging plane are calculated as follows:

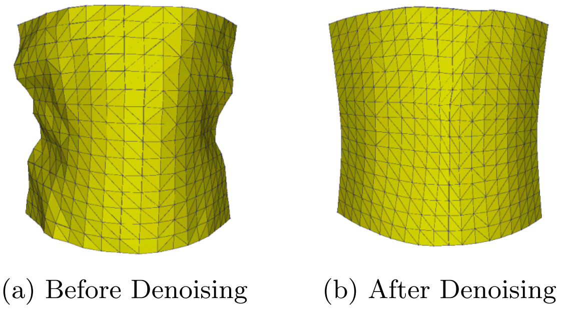

(a) Before Denoising (b) After Denoising

Fig. 5 Bilateral mesh denoising

$$\begin{aligned} x_{scr} &= f_x \cdot \frac{X}{Z} + c_x, \\ y_{scr} &= f_y \cdot \frac{Y}{Z} + c_y, \end{aligned} \tag{15}$$

where f_x and f_y denote the camera's focal lengths in the horizontal and vertical directions, respectively, following calibration, and c_x and c_y represent the offsets of the projection screen's coordinate centers. Substituting Eq. (15) into Eq. (3) yields the polar coordinates $(\rho_{scr}, \theta_{scr})$ of the point Q and determines the constraint range of Q's polar diameter: $\rho_{con} = fit\_contour(\theta_{scr})$. As the motion range of the mesh is confined within the perspective constraint space, Dirichlet boundary conditions are employed in the physical simulator to ensure $\rho_{scr} \leq \rho_{con}$. The method can be conceptualized as operating a physics-based simulator within a transparent, smooth, and irregular container, with fixed constraints established at specific positions. In brief, the method is represented as an optimization problem for a physical simulator utilizing implicit time integration under perspective space constraints:

$$\begin{aligned} &\min F(x) = \frac{1}{2\Delta t^2}(x - x_n - \Delta t v_n)^T \\ &M(x - x_n - \Delta t v_n) + E(x), \\ \\ &s.t. \quad \rho^i_{scr} - fit\_contour(\theta^i_{scr}) \leq 0, \\ &\quad i \in \{1, 2, \ldots, |\mathcal{G}|\}, \end{aligned} \tag{16}$$

where $(\rho^i_{scr}, \theta^i_{scr}) = cartToPolar(x^{Q_i}_{scr} - \bar{x}, y^{Q_i}_{scr} - \bar{y})$. Practical experience indicates that the iterative Limited-memory BFGS (L-BFGS) [46] algorithm efficiently resolves Eq. (16) with rapidity.

Step 3: Bilateral Mesh Denoising. The method relies significantly on the extraction and matching of feature points, rendering it susceptible to environmental factors and resulting in unstable positions of feature points on the image. The shaking of feature points is considered a form of noise, which subsequently influences the positions of template grid points in three-dimensional space. To address this issue, grid filtering was adopted to eliminate the noise, achieving good results, as illustrated in Fig. 5.

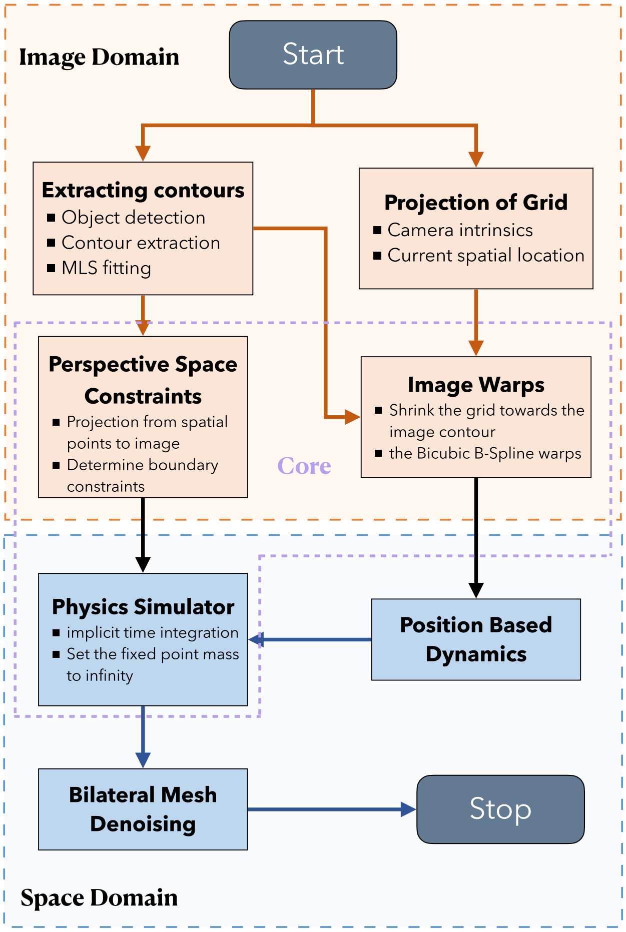


Image Domain
Start
Extracting contours
■ Object detection
■ Contour extraction
■ MLS fitting
Projection of Grid
■ Camera intrinsics
■ Current spatial location
Perspective Space Constraints
■ Projection from spatial points to image
■ Determine boundary constraints
Core
Image Warps
■ Shrink the grid towards the image contour
■ the Bicubic B-Spline warps
Physics Simulator
■ implicit time integration
■ Set the fixed point mass to infinity
Position Based Dynamics
Bilateral Mesh Denoising
Stop
Space Domain

Fig. 6 The flowchart of our method with the perspective space constraint

Implementation

We write the backbone of our scheme in C++ and use Eigen for matrix and vector operations. The Delaunay method [47] is used for triangulating the template, and OpenGL is employed for rendering, user interaction, and 3D visualization. Video acquisition, camera calibration, and image processing are achieved through OpenCV. The code runs on a Dell laptop with an Intel Xeon Silver 2.20 GHz CPU and a Quadro RTX8000 GPU.The flowchart of the method is shown in Fig. 6.

Experiments and results

Our method is characterized by its training-free nature, which facilitates the replacement of 3D tracking objects. To evaluate the accuracy and robustness of these methods across

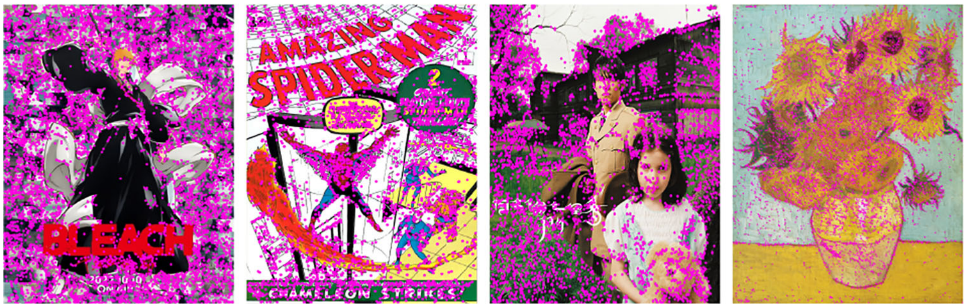


Fig. 7 Texture images covered by feature points

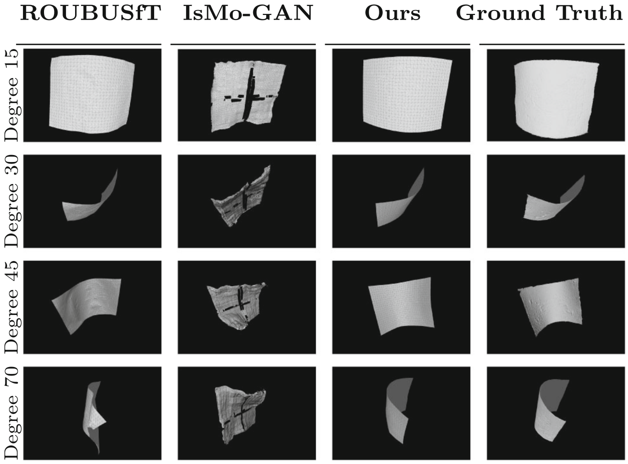


Fig. 8 Qualitative comparisons of deformation effects on templates with varying bending degrees using different methods. For each row, the degree of bending on the template increases gradually from top to bottom

different bending degrees and deformation modes with real data, four texture images featuring distinct feature point distributions were meticulously selected, as illustrated in Fig. 7. Images captured by a monocular RGB camera were utilized as input for the experiments. We acquire high-resolution images of the template's deformation from multiple perspectives and employ these images to perform 3D reconstruction. The reconstructed model is then employed as the ground truth within the framework of our comparative experimentation.

Comparison with existing methods

Our approach is compared with state-of-the-art methods such as ROUBUSfT [13], recognized for its robustness, and IsMo-GAN [33], noted for its deep learning-based universality. Considering IsMo-GAN's claim of good generalization and the absence of original texture images, we directly assessed its reasoning ability in template deformation using the pre-trained model provided by the authors.

The primary contribution of our method lies in its capacity to accurately infer the actual deformation of the template under significant bending and deformation conditions. To demonstrate the superiority of this method, four distinct bending states were selected, and the 3D surfaces reconstructed by various methods were compared. This visual comparison facilitates an intuitive recognition of this method's effectiveness. We evaluated the capacity of different methods to infer the deformation of templates with varying degrees of bending, as shown in Fig. 8. At low levels of bending, various methods are capable of accurately inferring the deformation of the template, consistent with the actual outcomes. As the degree of bending increases, however, local areas of the template may become invisible, causing relevant feature points to disappear. Consequently, the reconstruction error of ROUBUSfT significantly increases, particularly at the edge of the template, where accurate inference becomes impossible. Despite its claimed universality, the IsMo-GAN method exhibits poor deformation inference performance across various states, failing to demonstrate its potential. In addition to its inability to infer significant deformation states, the 3D shapes reconstructed by IsMo-GAN contain significant noise, resulting in an unsmooth surface. Our method demonstrated excellent universality, indicating its effective applicability to diverse tracking targets. Extensive experiments were conducted to evaluate the adaptability of this method for inferring deformations on templates with various texture features and different bending degrees. The results of these experiments are presented in Fig. 9. It is evident that this comprehensive approach effectively infers deformations across different states. However, it is essential to acknowledge that the accuracy of feature point matching and contour edge extraction play a crucial role in this method. Consequently, localized errors and variations may occur in the process of local and edge detection.

We selected a stationary state of the template and captured the shape inference results at four different time points, as depicted in Fig. 10. The results obtained with the ROUBUSfT method exhibited noticeable shaking, whereas our method demonstrated excellent stability.

To further quantify the error in deformation inference, the inferred point cloud is designated as $\mathcal{S} = \{s_i \in \mathbb{R}^3\}_i$ and the reconstructed ground truth point cloud as $\mathcal{G} = \{g_j \in \mathbb{R}^3\}_j$. Given that the point cloud $\mathcal{S}$ is obtained through inference and the point cloud $\mathcal{G}$ is reconstructed from images captured from multiple perspectives, they are inherently defined in different

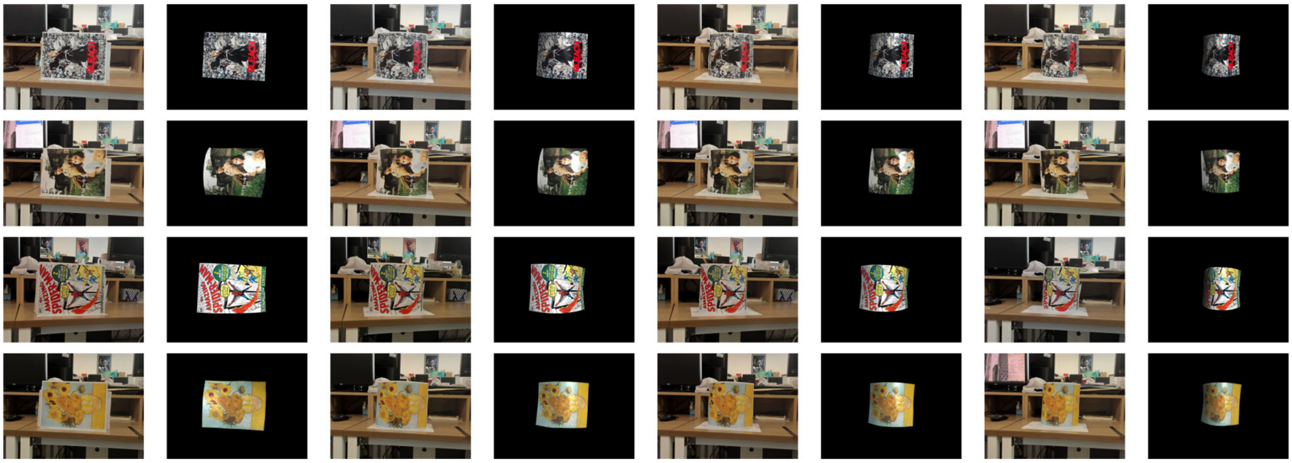
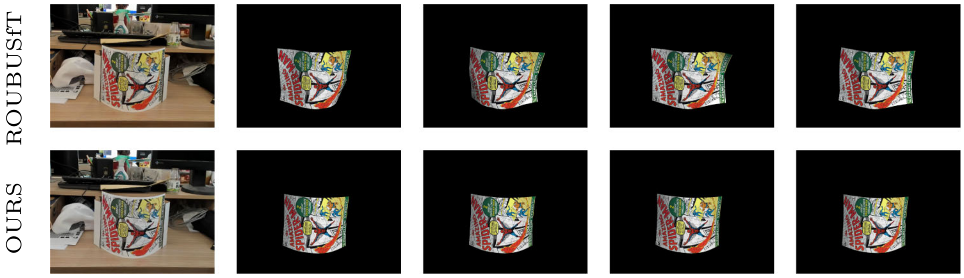
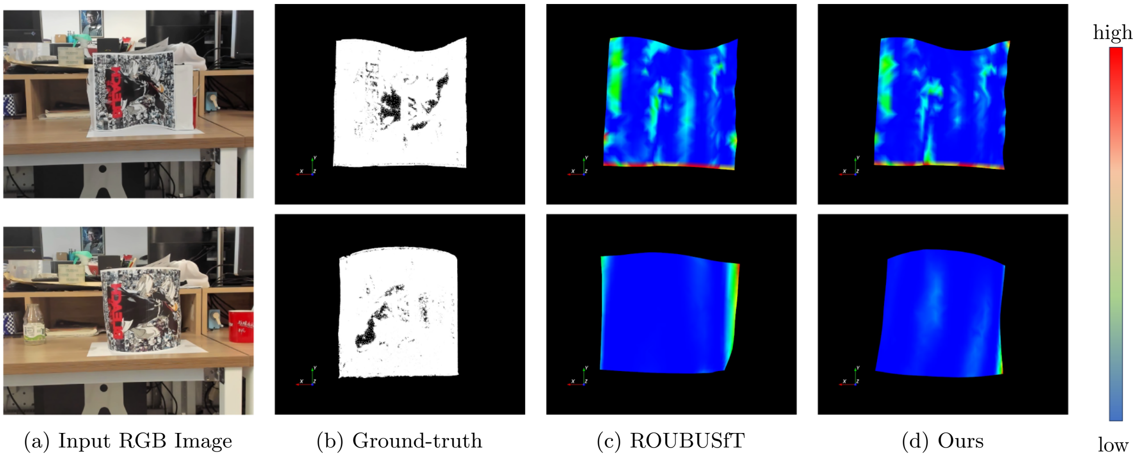


Fig. 9 Shape inference of templates in real scenes. From top to bottom, each row represents the same texture. From left to right, each pair of columns forms a group, where the left column represents images captured from a monocular camera, and the right column displays the results of shape inference. The level of template curvature varies across different groups

Fig. 10 Shape inference results of stationary templates at different time





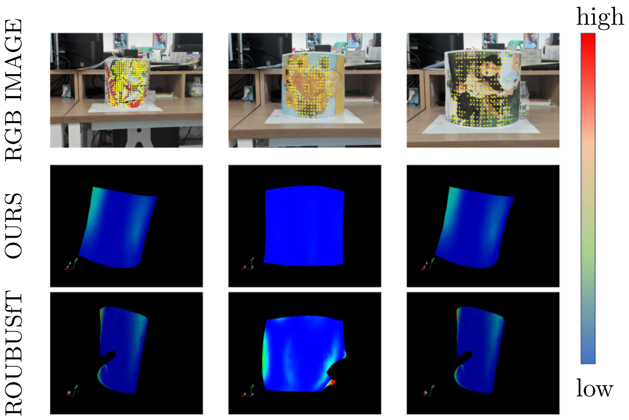
(a) Input RGB Image (b) Ground-truth (c) ROUBUSfT (d) Ours

Fig. 11 We show quantitative results as colour-coded error maps. For **a** the given RGB image, **b** the ground-truth shape, **c** reconstructed shape by ROUBUSfT, **d** our reconstructed shape

coordinate systems. For convenience, coordinate system calibration is not performed. Instead, the Iterative Closest Point (ICP) [48] algorithm is employed to register and align the two sets of point clouds. This approach mitigates disparities arising from the differing coordinate systems and facilitates a more accurate comparison between $\mathcal{S}$ and $\mathcal{G}$. Specifically, the inference error between point cloud $\mathcal{S}$ and point cloud $\mathcal{G}$ is evaluated by the Euclidean distance:

$$Err_{\mathcal{S}^{ICP},\mathcal{G}} = \frac{1}{|\mathcal{S}^{ICP}|}\sum_{s\in\mathcal{S}^{ICP}} min_{g\in\mathcal{G}}\|s-g\|_2^2 \tag{17}$$

where $\mathcal{S}^{ICP}$ represents the registered point cloud of $\mathcal{S}$ and $\mathcal{G}$ using the ICP algorithm.



(a) Texture 1 (b) Texture 2 (c) Texture 3

Fig. 12 We quantitatively compared the shape inference errors under different textures

We conducted error analysis on two distinct deformation states. In the case of minor deformations, it was observed that almost all local feature points of the template could be accurately captured. Figure 11 (upper row) shows that the standard deviation error for ROUBUSfT was 1.84 mm, whereas the corresponding error for the proposed method was 1.99 mm. Overall, both methods exhibited a high level of consistency. Conversely, in scenarios with significant deformations where local feature points are obscured, the proposed method demonstrated marked superiority over the ROUBUSfT method. As illustrated in the lower row of Fig. 11, the proposed method achieves an error of 4.18mm compared to the ground truth, whereas the ROUBUSfT method exhibits an error of 7.60 mm. This shows the considerable advantage of the proposed method in accurately estimating deformations under such conditions.

The deformation error of the proposed method across various textures was assessed, as shown in Fig. 12. Compared to the ROUBUSfT method, the proposed approach not only exhibits a comparatively lower error but also remarkable stability. Table 1 presents the maximum and standard deviation of shape inference errors for templates with various textures in different states. In *Texture 1 (T1)*, the error generated by the grid points is the largest among the three cases, reaching 50.797 mm, primarily due to the significant distance between the target and the camera. *Texture 3 (T3)* exhibits a higher relative error compared to Texture 2, mainly due to the inability to detect feature points around the image. However, the proposed method consistently outperforms other methods in both maximum error and standard deviation across all cases. Specifically, the proposed method achieves a maximum error reduction of sixfold compared to the ROUBUSfT method and a threefold reduction in maximum standard deviation.

Table 1 Inference errors for templates with different textures

Type	Method	T 1	T 2	T 3
Max	Ours	**22.331**	**4.697**	**8.189**
	ROUBUSfT	50.797	26.907	22.056
SD	Ours	**9.726**	**2.268**	**2.80**
	ROUBUSfT	11.182	6.301	4.569

Bold highlights the better result in the comparative experiment

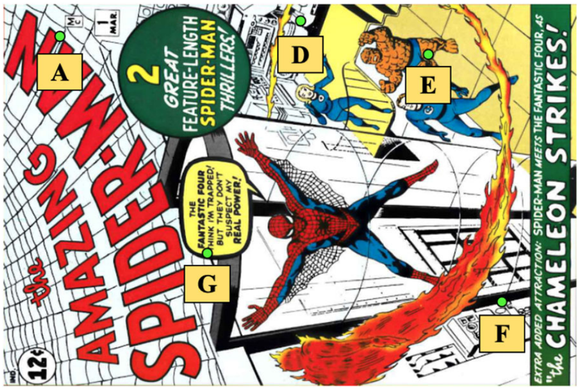

A
D
E
F
G
THE AMAZING SPIDER-MAN
2 GREAT FEATURE-LENGTH SPIDER-MAN THRILLERS!
THE CHAMELEON STRIKES!
EXTRA ADDED ATTRACTION: SPIDER-MAN MEETS THE FANTASTIC FOUR, AS
THE FANTASTIC FOUR THINK I'M TRAPPED! BUT THEY DON'T SUSPECT MY REAL POWER!
12¢

Fig. 13 The five marker points *(A, D, E, F, G)* are employed for tracking in both the true spatial coordinates and our deformation inference

Evaluation and performance

An optical tracking device (NDI Polaris Vega), featuring an inaccuracy of only 0.2 mm, was employed to determine the true spatial coordinates (NDI coordinates) of the deformable template and compare them to our deformation inference. For convenience, five marker points (A, D, E, F, G) on the template were selected, as shown in Fig. 13, and their coordinates were recorded in both NDI coordinates and the deformation inference (camera coordinates) for two different deformation states, as detailed in Table 2. The goal is to quantify the deformation errors between the actual deformation in NDI coordinates and the reconstructed results in camera coordinates. There is no necessity to align the two coordinate systems. We provide the 3D error metric

$$\varepsilon = \frac{1}{|R|} \sum_{(P,Q)\in R} \|P - Q\|, \tag{18}$$

where R represents a set of point pairs with P in camera coordinates and Q as its corresponding ground truth 3D point in NDI coordinates. For the five markers specified, the errors across two distinct deformation states were measured as $\varepsilon_1 = 5.96$ mm and $\varepsilon_2 = 4.21$ mm, respectively. The segments AG, FG, and DE were selected and rendered using Blender, as illustrated in Fig. 14. The computing efficiency of this method has been impacted by the implementation of numerous enhanced strategies. Fifty iterations

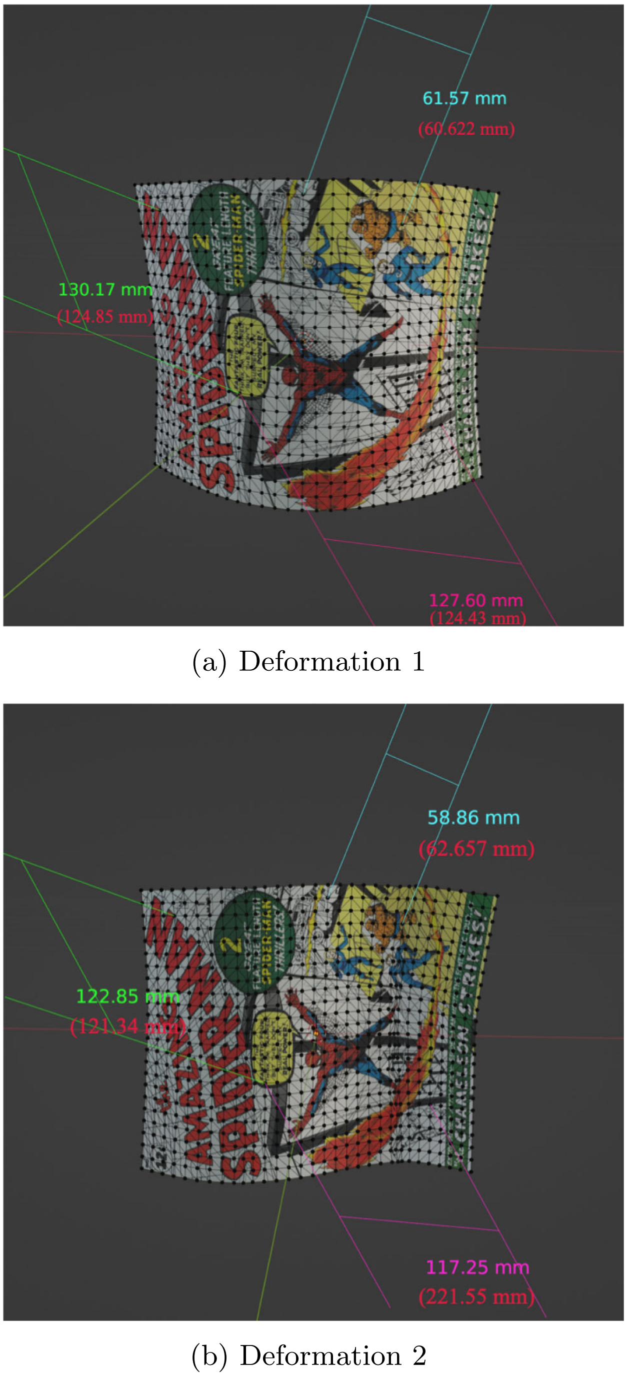
Table 2 The coordinates of the marker points are in two different deformation states, both in the NDI coordinate system and the camera coordinate system

Point	Deformation 1		Deformation 2	
	NDI	Camera	NDI	Camera
A	(123.66, 144.92, − 1229.05)	(− 95.73, − 77.8, 19.837)	(116.37, 117.44, − 1305.35)	(− 90.028, − 76.312, − 10.184)
D	(119.11, 240.87, − 1176.87)	(3.6115, − 87.277, − 31.247)	(103.33, 209.22, − 1340.66)	(6.3932, − 89.089, − 3.3607)
E	(131.81, 292.03, − 1206.81)	(58.274, − 71.34, − 7.8084)	(115.77, 259.30, − 1376.20)	(56.374, − 72.288, 22.803)
F	(246.17, 310.19, − 1242.77)	(84.334, 51.942, 18.866)	(234.28, 290.97, − 1376.56)	(77.781, 49.95, 16.323)
G	(229.00, 199.04, − 1189.53)	(− 34.587, 29.411, − 21.539)	(218.65, 180.14, − 1323.53)	(− 32.088, 30.497, − 19.695)

were conducted on a 24 × 32 meshed template, approximating the size of A4 paper, resulting in an average processing time of 4 s per deformation inference. The recently introduced method required approximately 3 s, constituting 75 percent of the total processing time. To further understand the performance bottleneck of our method, we divided it into four main components: image segmentation, contour fitting, physics simulation, and bilateral mesh denoising. Physics simulation encompasses the constrained physics simulator, as outlined in section "Constrained physics simulator", and position-based dynamics, as shown in Fig. 6. Different mesh sizes for the template were employed, ranging from 8 × 16 to 32 × 40, with seven incremental steps. For each dataset, 50 deformation inferences were conducted, and the average execution time was calculated, as illustrated in Fig. 15. With the increase in mesh size, there is a corresponding rise in the overall processing time. Contour fitting and physics simulation, key components of this method, exhibit a nearly linear increase, accounting for only about 20 percent of the overall processing time. Conversely, image segmentation and bilateral mesh denoising, key components of the comprehensive solution, constitute over 70 percent of the processing time, highlighting a critical issue impacting the performance of our approach. The deformation error for various mesh sizes was also evaluated, as illustrated in Fig. 16. Upon reaching a mesh size of 24 × 32, both the mean and median deformation errors of the grid points reach their minimum values. With the continued increase in mesh size, a higher amount of computational resources is required. However, the accuracy of the deformation inference remains relatively unchanged. The determination of mesh size should be based on the actual physical dimensions and texture characteristics of the deformable template, rather than on the assumption that larger sizes are inherently superior.

Discussion and future directions

We employ existing deep learning models to perform object detection and segmentation on images captured by monocular cameras, significantly reducing the real-time operational efficiency of the system. For a sparsely populated grid measuring 8x16, this component accounts for nearly 50 percent of the total execution time. To improve runtime efficiency, exploring alternatives like utilizing smaller and more efficient image segmentation models or devising targeted algorithms suited to specific application scenarios is beneficial. Such approaches hold the potential to significantly enhance overall performance. During the preliminary refinement of feature point positions based on image contours, a simple ray intersection method is employed due to its computational efficiency and robustness. However, the newly adjusted positions of the feature points exhibit significant errors, which considerably impact the subsequent deformation estimation. Dynamically adjusting the positions of global feature points while correcting local features on the edges can be considered. This approach aids in preventing issues related to dense local features and sparse internal features, potentially complicating the optimization process. By modifying local and global features simultaneously, a more balanced distribution of feature points can be achieved, thus facilitating the optimization process. In the core stage of deformation estimation, our method, based on perspective space constraints, facilitates the shape inference of real template deformations with notable speed and precision. The edges of the template image captured by the camera are extracted. These edges are subsequently transformed from the pixel coordinate system to the world coordinate system. This transformation process enables the determination of the actual template's motion range in three-dimensional space. However, in regions lacking visibility and feature points, this method tends to introduce errors. We believe that this is due to the lack of constraints in the process of free deformation and is also influenced by the iteration count of the physical simulator, as convergence is not achieved. Grid size significantly influences the speed and accuracy of this method. Generally, grid size determination should consider the template's physical dimensions and texture characteristics. With the expansion of the template and the increase in feature points, a larger grid becomes necessary. Having a suitably dense grid that precisely represents the geometric shape of



(a) Deformation 1

(b) Deformation 2

Fig. 14 Distance evaluation in different deformation states. Different colors are used to represent the straight-line distances between various marker points of interest. The red ones with brackets are the ground-truth

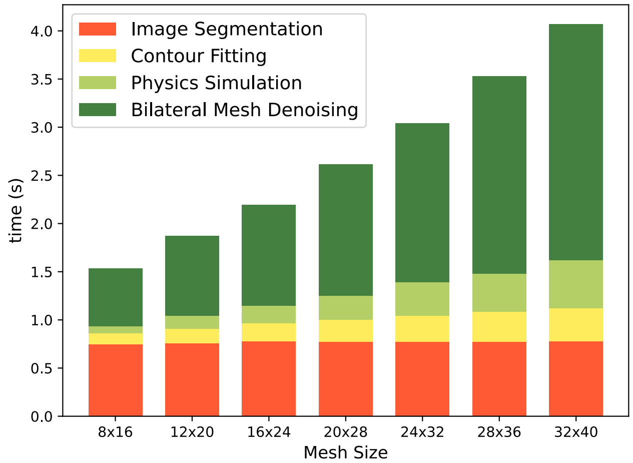


Fig. 15 The computational time of different components within one deformation inference varies by mesh size

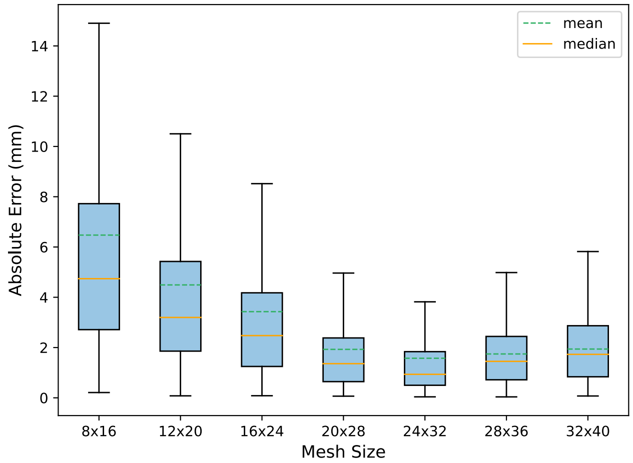


Fig. 16 Estimation of deformation inference errors for different mesh sizes

the actual template and encompasses all necessary feature points is crucial for deformation inference. The primary contribution of this work is integrating a constrained physical simulator into the SfT method. We have demonstrated the impact of physics-based simulation on reconstructing 3D surfaces. This method compensates effectively for the limitations of visual information, yielding more realistic and precise reconstruction outcomes. It is noteworthy that different simulators suit different material types, and simulation results may vary depending on the materials' physical properties. In this method, a mass-spring model is employed, which, while simple and efficient, may exhibit slightly lower accuracy compared to more sophisticated techniques such as finite element analysis. A crucial parameter within the mass-spring model is the springs' stiffness coefficient. Assigning different stiffness coefficients to each spring in the system allows for a better approximation of the simulated objects' physical properties. Lloyd et al. [44] proposed a parameter identification method that achieves accuracy comparable to finite element methods within the spring-mass framework. However, desirable results have been achieved in this implementation without the extensive fine-tuning of these parameters, leading to the decision to forgo further exploration in this area. In future work, it is essential for us to explore efficient and rapid object detection and segmentation methods tailored to specific template shapes, thereby enhancing the overall operational efficiency of the system. At the deformation inference stage, detecting a broader range of prior features from various perspectives is crucial to accurately determining the type and extent of template deformations. This will facilitate the optimization of deformation estimation methods and improve the accuracy of deformation estimation. Within the domain of learning-based approaches, a variety of models have already demonstrated significant potential. With an adequate volume of data samples, these models can provide initial estimates for traditional methods, thus enhancing the convergence speed of deformation estimation.

Conclusions

We have introduced an improved Shape from Template (SfT) method that effectively addresses the issue of shape inference errors arising from the invisibility of feature points in cases of significant deformations, which traditional methods struggle to handle. Utilizing images captured by a monocular RGB camera, this method extracts the contour of the target template, serving as the basis for establishing perspective space constraints. Ultimately,a physics-based simulator is employed to facilitate free motion and infer the 3D shape from 2D images. To evaluate this method's performance, experiments were conducted on templates with varying texture features, bending degrees, and deformation states. Particularly, we focused on scenarios where feature points were unobservable and quantified the experimental errors accordingly. The results indicated that our method generally outperforms the comparison method, with more noticeable improvements observed in cases involving substantial curvature. Our method effectively mitigates the bending inference errors associated with feature point based SfT methods and holds significant application value in the computer vision and virtual reality domains.

Acknowledgements This work was supported by Jilin Provincial Natural Science Foundation (YDZJ202101ZYTS050).

Data availability The datasets generated and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Pilet J, Lepetit V, Fua P (2008) Fast non-rigid surface detection, registration and realistic augmentation. Int J Comput Vis 76(2):109–122. https://doi.org/10.1007/s11263-006-0017-9
2. Aranda M, Antonio Corrales Ramon J, Mezouar Y, Bartoli A, Özgür E (2020) Monocular visual shape tracking and servoing for isometrically deforming objects. In: 2020 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 7542–7549. https://doi.org/10.1109/IROS45743.2020.9341646
3. Lamarca J, Parashar S, Bartoli A, Montiel JMM (2021) Defslam: tracking and mapping of deforming scenes from monocular sequences. IEEE Trans Robot 37(1):291–303. https://doi.org/10.1109/TRO.2020.3020739
4. Li P, Tang M, Ding K, Wu X, Liu Y (2021) Monocular tissue reconstruction via remote center motion for robot-assisted minimally invasive surgery. Complex Intell Syst 8:2923–2936
5. Bartoli A, Gérard Y, Chadebecq F, Collins T, Pizarro D (2015) Shape-from-template. IEEE Trans Pattern Anal Mach Intell 37(10):2099–2118. https://doi.org/10.1109/TPAMI.2015.2392759
6. Casillas-Perez D, Pizarro D, Fuentes-Jimenez D, Mazo M, Bartoli A (2019) Equiareal shape-from-template. J Math Imaging Vis 61:607–626
7. Casillas-Perez D, Pizarro D, Fuentes-Jimenez D, Mazo M, Bartoli A (2021) The isowarp: the template-based visual geometry of isometric surfaces. Int J Comput Vis 129(7):2194–2222
8. Wang R, Zhuang Z, Tao H, Paszke W, Stojanovic V (2023) Q-learning based fault estimation and fault tolerant iterative learning control for mimo systems. ISA Trans 142:123–135
9. Song X, Wu N, Song S, Stojanovic V (2023) Switching-like event-triggered state estimation for reaction-diffusion neural networks against dos attacks. Neural Process Lett 55:8997–9018
10. Kairanda N, Tretschk E, Elgharib M, Theobalt C, Golyanik V (2022) ϕ-sft: shape-from-template with a physics-based deformation model. In: Computer vision and pattern recognition (CVPR)
11. Stotko D, Wandel N, Klein R (2023) Physics-guided shape-from-template: monocular video perception through neural surrogate models
12. Tao H, Cheng L, Qiu J, Stojanovic V (2022) Few shot cross equipment fault diagnosis method based on parameter optimization and feature metric. Meas Sci Technol 33(11):115005
13. Shetab-Bushehri M, Aranda M, Mezouar Y, Bartoli A, Ozgur E (2023) Robusft: robust real-time shape-from-template, a c++ library. arXiv preprint arXiv:2301.04037
14. Özgür E, Bartoli A (2017) Particle-sft: a provably-convergent, fast shape-from-template algorithm. Int J Comput Vis 123:184–205
15. Zhuang Z, Tao H, Chen Y, Stojanovic V, Paszke W (2022) An optimal iterative learning control approach for linear systems with nonuniform trial lengths under input constraints. IEEE Trans Syst Man Cybern Syst 1:1. https://doi.org/10.1109/TSMC.2022.3225381
16. Bartoli A, Gérard Y, Chadebecq F, Collins T, Pizarro D (2015) Shape-from-template. IEEE Trans Pattern Analy Mach Intell 37(10):2099–2118

17. Pizarro D, Bartoli A (2012) Feature-based deformable surface detection with self-occlusion reasoning. Int J Comput Vis 97:54–70
18. Chhatkuli A, Pizarro D, Bartoli A, Collins T (2016) A stable analytical framework for isometric shape-from-template by surface integration. IEEE Trans Pattern Anal Mach Intell 39(5):833–850
19. Salzmann M, Moreno-Noguer F, Lepetit V, Fua P (2008) Closed-form solution to non-rigid 3d surface registration. In: European conference on computer vision. Springer, Berlin, pp 581–594
20. Salzmann M, Fua P (2009) Reconstructing sharply folding surfaces: a convex formulation. In: 2009 IEEE conference on computer vision and pattern recognition. IEEE, pp 1054–1061
21. Famouri M, Bartoli A, Azimifar Z (2018) Fast shape-from-template using local features. Mach Vis Appl 29:73–93
22. Aranda M, Ramon JAC, Mezouar Y, Bartoli A, Özgür E (2020) Monocular visual shape tracking and servoing for isometrically deforming objects. In: 2020 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 7542–7549
23. Brunet F, Bartoli A, Hartley RI (2014) Monocular template-based 3D surface reconstruction: convex inextensible and nonconvex isometric methods. Comput Vis Image Underst 125:138–154
24. Malti A, Hartley R, Bartoli A, Kim J-H (2013) Monocular template-based 3D reconstruction of extensible surfaces with local linear elasticity. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1522–1529
25. Ngo DT, Östlund J, Fua P (2015) Template-based monocular 3D shape recovery using Laplacian meshes. IEEE Trans Pattern Anal Mach Intell 38(1):172–187
26. Collins T, Bartoli A (2015) [poster] realtime shape-from-template: System and applications. In: 2015 IEEE International symposium on mixed and augmented reality. IEEE, pp 116–119
27. Collins T, Bartoli A, Bourdel N, Canis M (2016) Robust, real-time, dense and deformable 3d organ tracking in laparoscopic videos. In: International conference on medical image computing and computer-assisted intervention. Springer, Berlin, pp 404–412
28. Östlund J, Varol A, Ngo DT, Fua P (2012) Laplacian meshes for monocular 3d shape recovery. In: Computer Vision–ECCV 2012: 12th European conference on computer vision, Florence, October 7–13, 2012, Proceedings, Part III 12. Springer, Berlin, pp 412–425
29. Fuentes-Jimenez D, Pizarro D, Casillas-Perez D, Collins T, Bartoli A (2021) Texture-generic deep shape-from-template. IEEE Access 9:75211–75230
30. Fuentes-Jimenez D, Pizarro D, Casillas-Pérez D, Collins T, Bartoli A (2022) Deep shape-from-template: single-image quasi-isometric deformable registration and reconstruction. Image Vis Comput 127:104531
31. Golyanik V, Shimada S, Varanasi K, Stricker D (2018) Hdm-net: monocular non-rigid 3D reconstruction with learned deformation model. In: Virtual reality and augmented reality: 15th EuroVR international conference, EuroVR 2018, London, October 22–23, 2018, Proceedings 15. Springer, Berlin, pp 51–72
32. Pumarola A, Agudo A, Porzi L, Sanfeliu A, Lepetit V, Moreno-Noguer F (2018) Geometry-aware network for non-rigid shape prediction from a single view. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4681–4690
33. Shimada S, Golyanik V, Theobalt C, Stricker D (2019) Ismo-gan: adversarial learning for monocular non-rigid 3D reconstruction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops
34. Müller M, Heidelberger B, Hennix M, Ratcliff J (2007) Position based dynamics. J Vis Commun Image Represent 18(2):109–118
35. Rueckert D, Sonoda LI, Hayes C, Hill DL, Leach MO, Hawkes DJ (1999) Nonrigid registration using free-form deformations: application to breast MR images. IEEE Trans Med Imaging 18(8):712–721
36. Fleishman S, Drori I, Cohen-Or D (2003) Bilateral mesh denoising. In: ACM SIGGRAPH 2003 papers, pp 950–953
37. Kirillov A, Mintun E, Ravi N, Mao H, Rolland C, Gustafson L, Xiao T, Whitehead S, Berg AC, Lo W-Y et al (2023) Segment anything. arXiv preprint arXiv:2304.02643
38. Lancaster P, Salkauskas K (1981) Surfaces generated by moving least squares methods. Math Comput 37(155):141–158
39. Nealen A (2004) An as-short-as-possible introduction to the least squares, weighted least squares and moving least squares methods for scattered data approximation and interpolation. 130(150), 25. http://www.nealen.com/projects
40. Lee S, Wolberg G, Shin SY (1997) Scattered data interpolation with multilevel b-splines. IEEE Trans Vis Comput Graph 3(3):228–244. https://doi.org/10.1109/2945.620490
41. Saini D, Kumar S, Singh MK, Ali M (2021) Two view nurbs reconstruction based on gaco model. Complex Intell Syst 7(5):2329–2346
42. Bookstein FL (1989) Principal warps: thin-plate splines and the decomposition of deformations. IEEE Trans Pattern Anal Mach Intell 11(6):567–585. https://doi.org/10.1109/34.24792
43. Baraff D, Witkin A (2001) Large steps in cloth simulation. In: Proceedings of SIGGRAPH, vol 98. https://doi.org/10.1145/280814.280821
44. Lloyd B, Székely G, Harders M (2007) Identification of spring parameters for deformable object simulation. IEEE Trans Vis Comput Graph 13(5):1081–1094
45. Liu T, Bargteil AW, O'Brien JF, Kavan L (2013) Fast simulation of mass-spring systems. ACM Trans Graph 32(6):1–7. https://doi.org/10.1145/2508363.2508406
46. Liu DC, Nocedal J (1989) On the limited memory BFGS method for large scale optimization. Math Program 45(1–3):503–528. https://doi.org/10.1007/bf01589116
47. Su P, Drysdale RLS (1997) A comparison of sequential Delaunay triangulation algorithms. Comput Geom 7(5–6):361–385. https://doi.org/10.1016/s0925-7721(96)00025-9
48. Besl PJ, McKay ND (1992) Method for registration of 3-D shapes. In: Sensor fusion IV: control paradigms and data structures, vol 1611. SPIE, pp 586–606