Learning Visual Shape Control of Novel 3D Deformable Objects from Partial-View Point Clouds

Bao Thach[1], Brian Y. Cho[1], Alan Kuntz[1], Tucker Hermans[1,2]

***Abstract*— If robots could reliably manipulate the shape of 3D deformable objects, they could find applications in fields ranging from home care to warehouse fulfillment to surgical assistance. Analytic models of elastic, 3D deformable objects require numerous parameters to describe the potentially infinite degrees of freedom present in determining the object's shape. Previous attempts at performing 3D shape control rely on hand-crafted features to represent the object shape and require training of object-specific control models. We overcome these issues through the use of our novel *DeformerNet* neural network architecture, which operates on a partial-view point cloud of the object being manipulated and a point cloud of the goal shape to learn a low-dimensional representation of the object shape. This shape embedding enables the robot to learn to define a visual servo controller that provides Cartesian pose changes to the robot end-effector causing the object to deform towards its target shape. Crucially, we demonstrate both in simulation and on a physical robot that *DeformerNet* reliably generalizes to object shapes and material stiffness not seen during training and outperforms comparison methods for both the generic shape control and the surgical task of retraction.**

I. Introduction

Manipulation of 3D deformable objects stands at the heart of many tasks we wish to assign to autonomous robots. For example, home-assistance robots must be able to manipulate objects such as sponges, mops, bedding, and food to help people with day-to-day life. Robots operating in warehouses should safely handle deformable containers such as bags and boxes in order to package outgoing orders. Factory robots benefit from the ability to remove deformable objects from containers. Most critically, surgical assistive robots are required to safely and precisely manipulate deformable tissue and organs.

However, 3D deformable object manipulation presents many challenges [1]. The shape of deformable objects require a potentially infinite number of degrees of freedom (DOF) to describe, compared to only 6 DOF for rigid objects. As a result, deriving low-dimensional but accurate and expressive state representations for deformable objects is difficult. An additional challenge compared with simpler linear deformable objects such as ropes and cloth arises as elastic 3D deformable objects cannot be released without returning to their initial configuration. Further, deformable objects frequently have complex dynamics [2], making the process of deriving a model laborious and potentially computationally intensive. These issues all present themselves in the specific problem we examine in this work: 3D deformable object shape control. The shape control problem requires a robot to manipulate the internal DOF of a 3D deformable object to reach a desired shape.

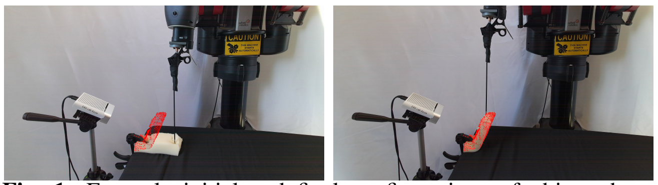
Fig. 1: Example initial and final configurations of object shape control via shape servo with *DeformerNet* on a physical robot using a laparoscopic tool. We visualize in red the goal point clouds given to the controller.

While rigid-body manipulation has received a large amount of study [3], due to the challenges listed above, autonomous 3D deformable object manipulation currently still remains an under-researched area [1, 4]—despite its potential relevance and need. Existing work for 3D deformable shape control leverages hard-coded feature vectors to describe deformable object state [5], which struggles to represent large sets of shapes. While learning-based methods show great promise in both rigid [6, 7] and deformable object manipulation [4, 8], these methods require a large amount of training data. Due to the difficulty of accurately simulating deformable objects, existing methods for shape control rely on data gathered via real-world setups, limiting the efficacy of learning-based approaches. Further, the ability to successfully manipulate deformable material is heavily dependent on where the robot grasps an object, however current works do not provide methods for selecting grasping points conditioned on the desired post-grasp manipulation.

In this work, we take steps toward addressing each of these gaps in the context of 3D deformable shape control. Our method takes as input a partial-view point cloud representation of a 3D deformable object and a desired goal shape. We build our method around a novel neural-network architecture, *DeformerNet*, which is trained on a large amount of data gathered via a recently-developed high-fidelity deformable object simulator, Isaac Gym [4, 9, 10].

Our method first reasons over the initial and target shape to select a manipulation point. Following selection of this grasp point, *DeformerNet* takes as input the current and target point clouds of the object, embeds the shape into a low-dimensional latent space representation, and computes a change in end-effector position that moves the object closer to the goal shape. The robot executes this motion and proceeds in a closed-loop fashion generating commands from *DeformerNet* until reaching the desired goal shape.

[1]Robotics Center and School of Computing, University of Utah, Salt Lake City, UT 84112, USA; [2]NVIDIA Corporation, Seattle, WA, USA; {bao.thach, brian.cho, alan.kuntz, tucker.hermans}@utah.edu B.T. was supported in part by NSF Award #2024778.

Figure 1 shows the initial and final configurations from an example manipulation using *DeformerNet* on a physical robot. Our results provide the first empirical demonstration of the importance of manipulation point selection for 3D shape control.

We focus our evaluation on the surgical robotics domain. We task a robot with manipulating three classes of object primitives into a variety of goal shapes using a laparoscopic tool. Unlike the preliminary results presented in our previous workshop paper [11] we vary the dimensions and the stiffness properties of the objects. We demonstrate effective manipulation on test objects both in simulation and on a physical robot. Importantly we show that our method can manipulate objects that fall both inside and outside the distributions of object shape and stiffness seen in training. We show that our DeformerNet outperforms both a sampling-based strategy and a model-free reinforcement learning approach on the shape control task.

We additionally present a strategy for applying our method to the common surgical task of retraction where we simplify the need of a target shape to only specifying a plane which the deformable tissue needs to be on one side of. We demonstrate successful retraction both in simulation and on the physical robot. We make available all code and data associated with this paper at `https://sites.google.com/view/deformernet/home`.

II. Related Work

Many approaches leverage machine learning with point cloud sensing to manipulate 3D rigid objects [6, 7, 12–15]. Authors have proposed various neural network architectures to encode object shape to achieve varying tasks such as grasp planing [6, 7, 14, 15], collision checking [12], shape completion [15], and object pose estimation [13]. In this work, we build upon these concepts to apply a learning-based approach which reasons over point cloud sensing with learned feature vectors to manipulate 3D deformable objects.

Solutions to 3D deformable object shape control [1] can be categorized into learning-based and learning-free approaches. Among the learning-free methods, a series of papers [16–18] define a set of geometric features on the object as the state representation. The authors use this representation to perform visual servoing with adaptive linear controller. These methods only work for known objects with distinct texture and cannot generalize to a diverse set of objects. This formulation controls the displacements of individual points which does not fully reflect the 3D shape of the object. For precise control, one must use a large number of feature points, making control highly susceptible to noise and occlusion. Other learning-free works [19–21] represent the object shape using 2D image contours; this severely limits the space of controllable 3D deformations.

Among learning-based 3D shape control methods, Hu *et al.* [5] represents the current state-of-the-art work in 3D shape control. Specifically, they use extended FPFH [22] to extract a feature vector from an input point cloud and learn to predict deformation actions via a neural network to control objects to desired shapes. However, we show that this architecture over-simplifies the complex dynamics of 3D deformable objects and thus struggles to learn to control to a diverse set of target shapes [11].

There has also been work on shape control of deformable objects that exhibit lower dimensional behavior, e.g., 1D objects such as rope, and 2D objects, such as cloth [2, 8, 23–25]. These methods typically either directly learn a policy using model-free RL that map RGB images of the object to robot actions [2, 23] or learn predictive models of the object under robot actions [8, 24–26]. These 1D and 2D works do not scale to the 3D deformable object shape control problem, either because they leverage lower dimensional object or sensing (e.g. RGB images) representation or the inherent physical differences between 1D, 2D, and 3D objects (e.g. 3D elastic tissue will return to its initial shape after released).

With respect to surgical robotics, several learning-based approaches have been applied to other surgical tasks including suturing [27, 28], cutting [29, 30], tissue tracking [31], and simulation [32]. In this work we apply our method to surgical retraction. Attanasio et al. [33] propose the use of surgeon-derived heuristic motion primitives to move tissue flaps identified by a vision system. In [34], a grasp location and planar retraction trajectory is computed with a linearized potential energy model leveraging online simulation. In [35], a logic-based task planner is leveraged which guarantees interpretability, however this work focuses on manipulating a single thin tissue sheet and does not show shape or material property generalization or validation on a physical robot. Nagy et al. [36] propose the use of stereo vison accompanied by multiple control methods, however the method assumes a thin tissue layer and a clear view of two tissue layers. Pore et al. [37] introduce a model-free reinforcement learning method which learns safe motions for a robot's end effector during retraction, however it does not explicitly reason over the deformation of the tissue. We compare against a similar approach, using the same model-free reinforcement learning algorithm, but adapted to our task to explicitly reason over the tissue state.

III. Problem Formulation

We address the problem of robotically manipulating a 3D deformable object from an initial shape to a goal shape. In this context, *3D* refers to *triparametric* or *volumetric* objects [1] which have no dimension significantly smaller than the other two, unlike *uniparametric* (e.g., rope) and *biparametric* objects (e.g., cloth).

We define the shape of the 3D volumetric object to be manipulated as $\mathcal{O} \subset \mathbb{R}^3$, noting that it will change over time as the robot manipulates it and the object interacts with the environment. As typical robots cannot directly sense $\mathcal{O}$, we consider a partial-view point cloud $\mathcal{P} \subset \mathcal{O}$ as a subset of the points on the surface of $\mathcal{O}$, due to the prevalence of sensors that produce point clouds. We define the point cloud representing the initial shape of the object as $\mathcal{P}_{\text{i}}$, the goal shape for the object as $\mathcal{P}_{\text{g}}$, and the shape of the object at a given intermediate point in time $\mathcal{P}_{\text{c}}$.

We note that the successful manipulation of a deformable object depends on the point on the object the robot grasps, i.e., the manipulation point (see Fig. 2). As such, we present the first problem as the selection of a manipulation point, which we define as $\mathbf{p}_{\mathrm{m}} = [x, y, z] \in \mathcal{O}$.

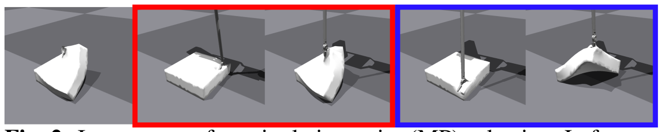
Fig. 2: Importance of manipulation point (MP) selection. Leftmost: goal shape; Red box: successful MP; Blue box: failed MP.

Having grasped the object, the robot can change that object's shape by moving its end-effector and in turn moving the manipulation point of the object. We define a manipulation action $\mathcal{A}$ as a change in the manipulation point, formally $\mathcal{A} \in \mathbb{R}^3, \mathcal{A} = \Delta\mathbf{x} = [\Delta x, \Delta y, \Delta z]$. The resulting problem then becomes to define a policy $\pi : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}^3$, which maps the point cloud representing the object shape and the goal point cloud to an action vector describing the change in manipulation point that drives the object toward the goal shape, i.e., $\pi(\mathcal{P}_{\mathrm{c}}, \mathcal{P}_{\mathrm{g}}) = \mathcal{A}$. The repeated application of a successful policy π results in a manipulation point trajectory, which when executed by the robot, results in transforming the object from its initial shape to a goal shape.

IV. Method

In this section we explain the details of our proposed approach. We first explain our *shape servo* [20] approach to create a feedback policy for 3D deformable object shape control. Following this we give details of the *DeformerNet* network architecture at the heart of our shape servo policy. Finally in this section we present our approach to selecting a manipulation point, conditioned on the goal configuration, used by the robot while performing shape control.

A. *Shape Servo Control with* DeformerNet

The shape servo formulation [5, 20] uses visual feedback, here in the form of partial-view point clouds of the object being manipulated, as input to a policy that computes a robot action that attempts to instantaneously bring the current shape, $\mathcal{P}_{\mathrm{c}}$ closer to the target shape, $\mathcal{P}_{\mathrm{g}}$.

Following the notation from Sec. III we seek to construct a shape servo policy of the form $\pi_{\mathrm{s}}(\mathcal{P}_{\mathrm{c}}, \mathcal{P}_{\mathrm{g}}) = \mathcal{A}$. We decompose our policy into two stages: (1) a feature extraction stage and (2) a deformation controller (c.f. Fig. 3 top).

The feature extractor $g(\mathcal{P}) = \psi$ takes a point cloud as input and outputs a shape feature vector we define as ψ. We use two parallel feature extraction channels taking as input $\mathcal{P}_{\mathrm{c}}$ and $\mathcal{P}_{\mathrm{g}}$ and generating feature vectors ψ_{c} and ψ_{g} respectively. We then take the difference of these two to define the feature displacements: $\Delta\psi = \psi_{\mathrm{c}} - \psi_{\mathrm{g}}$.

Our deformation control function, F, takes this feature displacement as input and outputs the desired instantaneous change in end-effector position, hence: $\mathcal{A} = F(\Delta\psi)$.

The composite shape servo policy thus takes the form $\pi_{\mathrm{s}}(\mathcal{P}_{\mathrm{c}}, \mathcal{P}_{\mathrm{g}}) = F(g(\mathcal{P}_{\mathrm{c}}) - g(\mathcal{P}_{\mathrm{g}})) = \mathcal{A}$. We then use a resolved rate controller to compute the desired joint velocities following the desired end-effector displacement output by our shape servo policy π_{s}.

Training this model takes a straightforward supervised approach. We simply record the robot manipulating an object of interest, set the terminal object point cloud as $\mathcal{P}_{\mathrm{g}}$, select any previous point cloud from the trajectory as $\mathcal{P}_{\mathrm{c}}$ and the associated end-effector displacement between the two configurations as $\mathcal{A}$. We give further details of this training procedure in Sec. V.

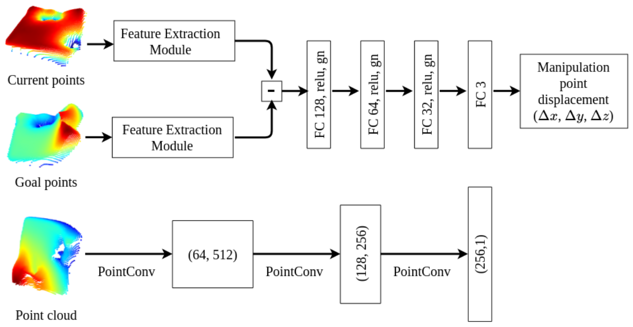

Fig. 3: (Top) Architecture of *DeformerNet*; (Bottom) architecture of the feature extraction module.

B. DeformerNet *Architecture Details*

As described previously, *DeformerNet* consists of two stages: feature extraction and deformation control. Our feature extractor uses three successive PointConv [38] convolutional layers that successively output clouds of dimension (64, 512), (128, 256) and ultimately a 256-dimension vector that acts as the shape feature. We downsample the input current, $\mathcal{P}_{\mathrm{c}}$, and goal point clouds, $\mathcal{P}_{\mathrm{g}}$, to 1024 points using the furthest point sampling method from [39] before inputting them into the network. We provide full details of the architecture in the bottom of Fig. 3.

The deformation control stage takes this 256-dimension *differential feature vector* and passes it through a series of fully-connected layers (128, 64, and 32 neural units, respectively). The fully-connected output layer produces the desired 3D displacement. We use ReLU activation function and group normalization [40] for all convolutional and fully-connected layers except for the linear output layer.

We use the standard mean squared error loss function for training *DeformerNet*. We adopt the Adam optimizer and a decaying learning rate which starts at 10^{-3} and decreases by 1/10 every 50 epochs.

C. *Manipulation point prediction*

As discussed above and shown in Fig. 2 the location at which the robot grasps the object greatly influences whether the robot can reach a target shape. As such we present here an approach to selecting an appropriate manipulation point prior to performing the shape control task. Recall we wish to find a manipulation point on the surface of the object, $\mathbf{p}_{\mathrm{m}} \in \mathcal{O}$. However, we must infer this location given the initial $\mathcal{P}_{\mathrm{i}}$ and target point clouds, $\mathcal{P}_{\mathrm{g}}$ prior to acting. We propose the use of a keypoint-based heuristic to select the manipulation point. Our preliminary work [11] showed this heuristic slightly outperformed a regression-based approach.

Our heuristic follows a simple idea, points that move more should generally be closer to the manipulation point. Assume we have a set of K keypoint matches $\mathcal{M} = \{(u_j, v_j) | u_j \in \mathcal{P}_\mathrm{i}, v_j \in \mathcal{P}_\mathrm{g}\}_{j=1:K}$ between the initial and goal point cloud. We define the associated keypoint displacements as $\delta_k = \{\|u_j - v_j\|\}_{j=1:K}$. We then estimate the manipulation point as the location defined by the displacement-weighted average of the M keypoints with largest displacement.

We use an unsupervised keypoint detection algorithm based on the Transporter Network of [41]. The original Transporter network defines an unsupervised reconstruction loss between source and target image pairs from a video sequence. To adapt transporters to our 3D manipulation point prediction problem, we leverage pairs of source-target point clouds collected in simulation to train the model. We convert the point cloud data to an organized, array-like point cloud format to make them compatible with the original Transporter network architecture.

V. Experiments and results

We evaluate our method in both simulation, via the Isaac Gym environment [9], and on a real robot. For both simulation and real robot experiments, training data for the learned models are generated in Isaac Gym. In Isaac Gym, we use a simulation of a patient-side manipulator of the daVinci research kit (dVRK) [42] robot to manipulate objects (see Fig. 4 (right)). For the real robot experiments, we use a Baxter research robot with a laparoscopic tool attached to its end effector and an Azure Kinect camera generating point clouds of the deformable object (see Fig. 1). In both cases, we affix one end of the deformable object to the environment and task the robot with manipulating it via one grasp point.

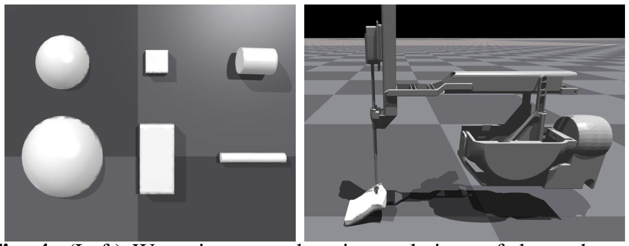
Fig. 4: (Left) We train on random interpolations of these shapes. (Right) Experimental setup showing a patient-side manipulator of the dVRK in Issac gym.

A. Goal-Oriented Shape Servoing

We evaluate our method's ability to deform the object to the goal point cloud. In our previous workshop paper [11], we reported the performance of our method when the model was trained and tested on one object geometry and Young modulus and demonstrated that our method outperforms a current state-of-the-art method for learning-based 3D shape servoing by Hu *et al.* [5].

1) Training Data Generation: We expand on this evaluation in this work by first evaluating our method's ability to control the shape of a variety of 3D deformable object shape primitives, including hemispheres, rectangular boxes, and cylinders (see Fig. 4). For each primitive, we investigate three different stiffness values (represented by Young's modulus): 1 kPa, 5 kPa, and 10kPa, which represent stiffness properties similar to those seen across different biological tissues [43, 44]. The three shape primitives, each with three stiffness values result in a total of nine object types for evaluation.

For each of the nine object types, we create a training dataset of objects with geometries sampled uniformly at random from interpolations between the sizes of the shapes in Fig. 4. In addition, each object for training is assigned a Young modulus sampled from a Gaussian distribution with means and standard deviations of (1kPa, 0.2kPa), (5kPa, 1kPa), and (10kPa, 1kPa) for the 1 kPa, 5 kPa, and 10 kPa test scenarios, respectively. We train a separate model for each of the nine objects, using the same *DeformerNet* architecture.

We generate each training dataset by randomly sampling 300 pairs of initial object configurations and manipulation points. For each pair, the robot deforms the object to 10 random shapes for a total of 3000 random trajectories. We record partial-view point clouds of the object and the robot's end-effector positions at multiple checkpoints during the execution of this trajectory using the depth camera available inside the Issac gym environment. We form supervised data input-output pairs for training *DeformerNet*. The input, $(\mathcal{P}_t, \mathcal{P}_\mathrm{g})$ consists of a point cloud along the trajectory at any arbitrary time t, as well as the point cloud at the end of this trajectory. We compute the output, $\Delta \mathbf{x}_t$, as the displacement between the end-effector position at time t and the end of trajectory. We sample 10,000 such pairs of data points for training our model.

2) Generalization Performance: We are interested in evaluating the performance of our method on test scenarios that are both inside and outside the training distributions in simulation. To generate test scenarios outside the training distribution, we sample objects with random dimensions smaller than the minimum and larger than the maximum of each of our primitive-shaped objects. We additionally sample Young moduli with values 2-4 standard deviations from the mean of the training distribution moduli. For each test scenario we select 10 random objects from inside the training distribution and 10 from outside the training distribution. We then sample 10 random goal shapes for each of the 20 test objects. We select the manipulation point for testing using our keypoint-based heuristic with $K = 200$ keypoints.

We use Chamfer distance as our primary evaluation metric to describe how close the final manipulated object's point cloud is to the goal point cloud. Chamfer distance computes the average distance of each point in one point cloud to the closest point in the other point cloud, $d_c = \frac{1}{|\mathcal{P}_1|} \sum_{x \in \mathcal{P}_1} \min_{y \in \mathcal{P}_2} ||x - y||^2$. Fig. 5 visualizes the result of each object type with a boxplot recorded over the 20 test objects with 10 goal shapes each. The box represents the quartiles, the center line the median, and the whiskers represent min and max final Chamfer distance. For visualization purpose, we also provide a sample snapshot of the robot performing shape servoing to a goal shape in Fig. 7.

The experiment results show that our method is capable of generalizing what it learns from training to adapt to

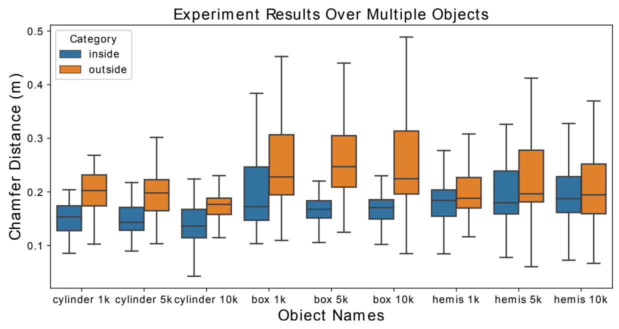

Fig. 5: Distribution of Chamfer distance after shape servoing. "Inside" reports for object inside the training distribution and "outside" for random objects outside the training distribution.

geometries, material properties, and goal shapes it has never seen before, both inside and outside the training distribution, although predictably with some fall off in performance outside the training distribution. We observe that the primary common cause of failure comes from the heuristic manipulation point predictor selecting a grasp location unable to achieve the goal shape.

3) Baseline Comparisons: We also compare the performance of our method against Rapidly-exploring Random Tree (RRT) [45] and model-free Reinforcement Learning (RL) for the 3D shape servo problem. Here we restrict the task to be trained and tested on a single box object as described in [11] and use only one manipulation point throughout training and testing.

For the RRT implementation, we define the configuration space as the joint angles of the dVRK manipulator. We define a goal region as any object point cloud that has Chamfer distance less than some tolerance from the goal point cloud. We use the finite element analysis model [10] in the Isaac Gym [9] simulator to derive the forward model for RRT.

We use proximal policy optimization (PPO) [46] (as in [37]) with hindsight experience replay (HER) [47] for model-free RL. We use our *DeformerNet* architecture for the actor and critic network except for the critic output being set to single scalar to encode the value function. Each episode we condition the policy on a newly sampled goal shape. We train the RL agent with 100,000 samples—10 times the amount of data provided to *DeformerNet*.

We evaluate DeformerNet, RRT, and model-free RL with 10 random goal shapes. Fig. 6 shows the success rate of the three methods at different levels of goal tolerance. We clearly see that even with 10 times the training data compared to our method, the model-free RL agent achieves a significantly lower success rate compared to the other two methods. We also note that while RRT succeeds comparably to our method at looser goal tolerances, at tighter goal tolerances RRT fails more often. Further, unlike our method, RRT does not incorporate feedback during execution. As such RRT will not be able to recover if the object shape deviates from the plan. While one might think to perform replanning, we note that RRT requires several orders of magnitude more computation time required than our shape servoing approach. For instance, at a tolerance of 0.4 (where both our method and RRT achieve 100% success), over the 10 test goal shapes, the lowest computation time required by RRT was 3.3 minutes,

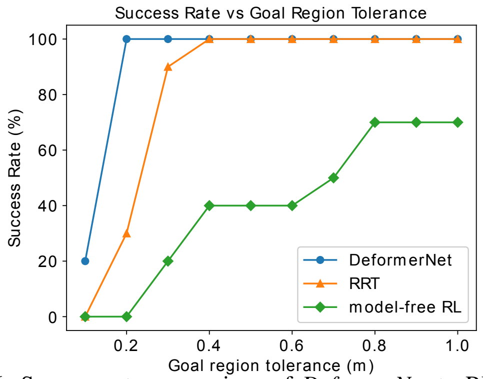

Fig. 6: Success rate comparison of *DeformerNet* to RRT and RL baselines for varying levels of goal tolerance.

the highest was 121.6 minutes, mean was 38.7 minutes, and standard deviation was 40.1 minutes. Our DeformerNet, however, only requires a pass through the neural network which takes minimal time. As a result, for this task, we note a significant success rate improvement for our method over model-free RL, a success rate improvement at strict goal tolerance values over RRT, and a significant computation time improvement over RRT in all cases.

4) Physical Robot Goal-Oriented Shape Servoing: We next evaluate our method's ability to perform shape servoing on the real robot, while having been trained entirely in simulation. The experimental setup (shown in Fig. 1) leverages a foam box affixed on one side to a table. We segment the object's point cloud out from the rest of the scene by fitting a plane to the table with RANSAC [48] and selecting the points above this planes. We filter out the black table clamp and the laparoscopic tool using pixel intensity.

We generate three distinct goal shapes (Fig. 10 (left)) by manually moving the object to random shapes with the laparoscopic tool and recording the resulting point cloud. Figure 9 describes the success rate of the 15 trials over different goal tolerance levels. Figure 7 visualizes a typical manipulation sequence.

To showcase our method's robustness, we additionally evaluate on 3 goal point clouds obtained entirely from the simulator (Fig. 10 (right)). Figure 9 visualizes the success rate of the 15 trials over different goal tolerance levels. A sample visualization is provided in Fig. 7. Overall we note a slight drop in quantitative performance in the real world compared to simulation, while qualitatively still succeeding.

B. Surgical Retraction

We next evaluate our method's ability to perform a mock surgical retraction task, in which a thin layer of tissue is positioned on top of a kidney. We task the robot with grasping the tissue layer and lifting it up to expose the underlying area. Figure 8 (top, left) shows the simulation environment composed of a kidney model with a deformable tissue layer placed over it and fixed to the kidney on one side. We train *DeformerNet* on a box object similar in dimensions to the tissue layer, but without the kidney present.

Instead of requiring the operator (e.g. surgeon) to provide an explicit shape for the robot to servo the tissue to, we instead just require them to define a plane which the tissue

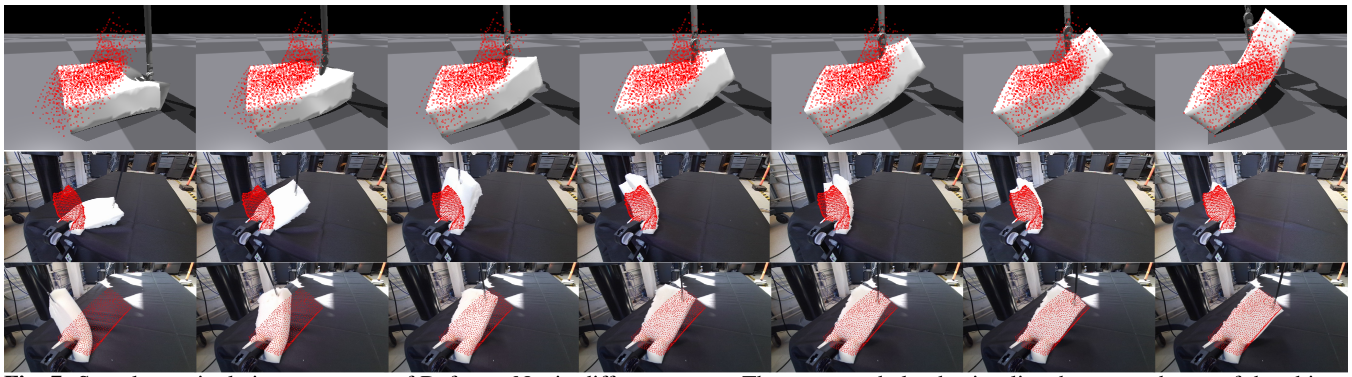
Fig. 7: Sample manipulation sequences of DeformerNet in different setups. The sparse red clouds visualize the target shapes of the object. First row: with simulated dVRK in Isaac Gym (0.18 m final Chamfer dist.). Second row: with physical robot and real goal point clouds (0.30m final Chamfer dist.). Third row: with physical robot and simulated goal point clouds (0.39m final Chamfer dist.).

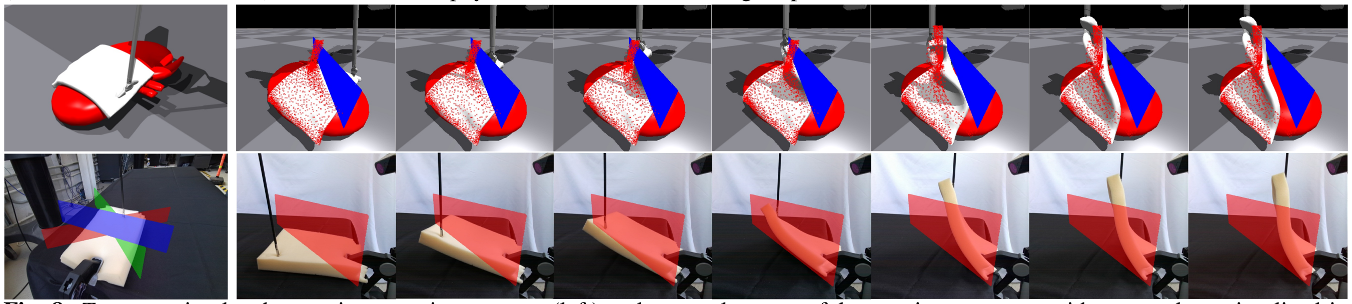
Fig. 8: *Top row*: simulated retraction experiment setup (left) and a sample successful retraction sequence with target plane visualized in blue. *Bottom row*: visualization of target planes for physical robot retraction (left) and a successful sequence with target plane in red.

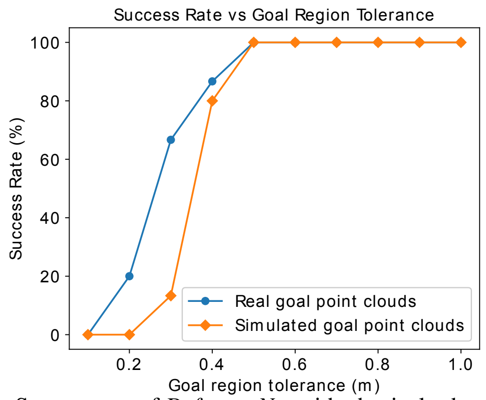

Fig. 9: Success rate of *DeformerNet* with physical robot when given real goal point clouds and simulated goal point clouds.

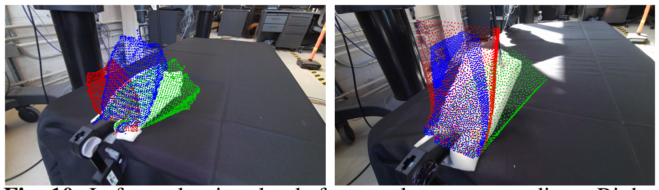
Fig. 10: Left: goal point clouds from real sensor recordings. Right: goal point clouds generated in simulation.

should be folded to one side of. An example plane can be seen in Fig. 8. We use a simple algorithm to infer a goal point cloud for the object based on this target plane. We use RANSAC [48] to find a dominant plane in the object cloud and then find the minimum rotation to align this plane with the target plane. We then apply this estimated transformation to any points not lying on the correct side of the plane and set this as the target cloud along with the points currently satisfying the goal. If after reaching the goal point cloud any part of the object still resides on the wrong side of the plane, we shift the target plane further into the goal region along the plane's normal vector and repeat the entire process.

To evaluate we sample 100 random planes with differing orientations in simulation and task the method with moving the tissue layer beyond the plane. Our approach reveals the kidney underneath with a success rate of 95%.

We also evaluate retraction on the physical robot. We affix a thin layer of foam to the table and task the robot with moving the object via the laparoscopic tool beyond a target plane. We evaluate on 3 different planes (see Fig. 8), and for each plane conduct 5 trials. We observe a 100% success rate across the 15 trials. We provide visualizations of representative retraction experiments in Fig. 8.

VI. Conclusions

In this paper we presented a novel-approach to closed-loop 3D deformable object shape control. Crucially we demonstrate through rigorous simulated and physical-robot experiments that shape servoing with *DeformerNet* can manipulate objects with novel material properties or shape while only requiring a partial-view 3D point cloud as input. We further demonstrate how our shape servoing approach can be adapted to the task of surgical retraction, where a much simpler goal representation in the form of a separating plane needs only be provided. Our future work aims to extend our manipulation approach to more surgical tasks, to manipulation of plastic materials [49], and manipulating deformable 3D objects common to homes and warehouses. Finally, we wish to move beyond our greedy, visual servoing approach to provide more explicit planning for longer-horizon tasks.

References

[1] J. Sanchez, J. A. C. Ramon, B.-C. Bouzgarrou, and Y. Mezouar, "Robotic Manipulation and Sensing of Deformable Objects in Domestic and Industrial Applications: A Survey," *Intl. Journal of Robotics Research*, vol. 37, no. 7, pp. 688–716, 2018. [Online]. Available: https://journals.sagepub.com/doi/abs/10.1177/0278364918779698

[2] Y. Wu, W. Yan, T. Kurutach, L. Pinto, and P. Abbeel, "Learning to manipulate deformable objects without demonstrations," *Robotics: Science and Systems*, 2020. [Online]. Available: https://arxiv.org/abs/1910.13439

[3] M. T. Mason, "Toward Robotic Manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 1–28, 2018.

[4] I. Huang, Y. Narang, C. Eppner, B. Sundaralingam, M. Macklin, T. Hermans, and D. Fox, "DefGraspSim: Simulation-based grasping of 3D deformable objects," in *RSS Workshop on Deformable Object Simulation in Robotics (DO-Sim)*, 2021. [Online]. Available: https://arxiv.org/abs/2107.05778

[5] Z. Hu, T. Han, P. Sun, J. Pan, and D. Manocha, "3-D Deformable Object Manipulation Using Deep Neural Networks," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4255–4261, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8769898

[6] Q. Lu, M. V. der Merwe, B. Sundaralingam, and T. Hermans, "Multi-Fingered Grasp Planning via Inference in Deep Neural Networks," *IEEE Robotics & Automation Magazine (Special Issue on Deep Learning and Machine Learning in Robotics)*, vol. 27, no. 2, pp. 55–65, 2020. [Online]. Available: https://arxiv.org/abs/2001.09242

[7] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," *Intl. Conf. on Computer Vision*, pp. 2901–2910, 2019.

[8] W. Yan, A. Vangipuram, P. Abbeel, , and L. Pinto, "Learning predictive representations for deformable objects using contrastive estimation," in *Conference on Robot Learning*, 2020. [Online]. Available: https://arxiv.org/abs/2003.05436

[9] J. Liang, V. Makoviychuk, A. Handa, N. Chentanez, M. Macklin, and D. Fox, "GPU-Accelerated Robotic Simulation for Distributed Reinforcement Learning," *arXiv:1810.05762*, 2018. [Online]. Available: https://arxiv.org/abs/1810.05762

[10] M. Macklin, K. Erleben, M. Müller, N. Chentanez, S. Jeschke, and V. Makoviychuk, "Non-Smooth Newton Methods for Deformable Multi-Body Dynamics," *ACM Trans. on Graphics*, 2019.

[11] B. Thach, A. Kuntz, and T. Hermans, "DeformerNet: A Deep Learning Approach to 3D Deformable Object Manipulation," in *RSS Workshop on Deformable Object Simulation in Robotics (DO-Sim)*, 2021. [Online]. Available: https://drive.google.com/file/d/1Alhv27gwcPA1lzJu4u0SR61UgbJKeKJA

[12] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox, "6-DOF Grasping for Target-driven Object Manipulation in Clutter," *IEEE Intl. Conf. on Robotics and Automation*, pp. 6232–6238, 2020.

[13] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox, "Self-supervised 6D Object Pose Estimation for Robot Manipulation," *IEEE Intl. Conf. on Robotics and Automation*, pp. 3665–3671, 2020.

[14] Q. Lu, M. V. der Merwe, and T. Hermans, "Multi-Fingered Active Grasp Learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. [Online]. Available: https://arxiv.org/abs/2006.05264

[15] M. V. der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, "Learning Continuous 3D Reconstructions for Geometrically Aware Grasping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020. [Online]. Available: https://sites.google.com/view/reconstruction-grasp/home

[16] D. Navarro-Alarcon, Y. Liu, J. G. Romero, and P. Li, "Visually servoed deformation control by robot manipulators," *IEEE Intl. Conf. on Robotics and Automation*, p. 5259–5264, 2013. [Online]. Available: https://ieeexplore.ieee.org/document/6581888

[17] D. Navarro-Alarcon, Y.-h. Liu, J. G. Romero, and P. Li, "On the visual deformation servoing of compliant objects: Uncalibrated control methods and experiments," *Intl. Journal of Robotics Research*, vol. 33, no. 11, pp. 1462–1480, 2014.

[18] D. Navarro-Alarcon, H. M. Yip, Z. Wang, Y.-H. Liu, F. Zhong, T. Zhang, and P. Li, "Automatic 3-D Manipulation of Soft Objects by Robotic Arms With an Adaptive Deformation Model," *IEEE Trans. on Robotics*, vol. 32, no. 2, pp. 429–441, 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7429768

[19] J. Qi, G. Ma, J. Zhu, P. Zhou, Y. Lyu, H. Zhang, and D. Navarro-Alarcon, "Contour Moments Based Manipulation of Composite Rigid-Deformable Objects with Finite Time Model Estimation and Shape/Position Control," *arXiv:2106.02424*, 2021. [Online]. Available: https://arxiv.org/abs/2106.02424

[20] D. Navarro-Alarcon and Y.-H. Liu, "Fourier-Based Shape Servoing: A New Feedback Method to Actively Deform Soft Objects into Desired 2-D Image Contour," *IEEE Trans. on Robotics*, vol. 34, no. 1, pp. 272–1279, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8106734

[21] J. Zhu, D. Navarro-Alarcon, R. Passama, and A. Cherubini, "Vision-based manipulation of deformable and rigid objects using subspace projections of 2d contours," *Robotics and Autonomous Systems*, vol. 142, p. 103798, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092188902100083X

[22] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D recognition and pose using the Viewpoint Feature Histogram," *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 2155–2162, 2010. [Online]. Available: https://ieeexplore.ieee.org/document/5651280

[23] J. Matas, S. James, , and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation," *Conference on Robot Learning*, pp. 734—-743, 2018.

[24] X. Ma, S. Chen, D. Hsu, and W. S. Lee, "Contrastive variational model-based reinforcement learning for complex observations," *Conference on Robot Learning*, 2020.

[25] D. McConachie and D. Berenson, "Bandit-Based Model Selection for Deformable Object Manipulation," *arXiv:1703.10254*, 2017.

[26] X. Ma, D. Hsu, and W. S. Lee, "Learning Latent Graph Dynamics for Deformable Object Manipulation," *arxiv*, 2021. [Online]. Available: https://arxiv.org/abs/2104.12149

[27] J. van den Berg, S. Miller, D. Duckworth, H. Hu, A. Wan, K. Goldberg, and P. Abbeel, "Superhuman Performance of Surgical Tasks by Robots Using Iterative Learning from Human-Guided Demonstrations," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, May 2010, pp. 2074–2081.

[28] Z.-Y. Chiu, F. Richter, E. K. Funk, R. K. Orosco, and M. C. Yip, "Bimanual Regrasping for Suture Needles using Reinforcement Learning for Rapid Motion Planning," *IEEE Intl. Conf. on Robotics and Automation*, 2021.

[29] B. Thananjeyan, A. Garg, S. Krishnan, C. Chen, L. Miller, and K. Goldberg, "Multilateral surgical pattern cutting in 2d orthotropic gauze with deep reinforcement learning policies for tensioning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2371–2378.

[30] A. Murali, S. Sen, B. Kehoe, A. Garg, S. McFarland, S. Patil, W. D. Boyd, S. Lim, P. Abbeel, and K. Goldberg, "Learning by Observation for Surgical Subtasks: Multilateral Cutting of 3D Viscoelastic and 2D Orthotropic Tissue Phantoms," in *IEEE Intl. Conf. on Robotics and Automation*, May 2015, pp. 1202–1209.

[31] J. Lu, A. Jayakumari, F. Richter, Y. Li, and M. C. Yip, "SuPer Deep: A Surgical Perception Framework for Robotic Tissue Manipulation using Deep Learning for Feature Extraction," *IEEE Intl. Conf. on Robotics and Automation*, 2021.

[32] J. Xu, B. Li, Y.-H. L. Bo Lu, Q. Dou, and P.-A. Heng, "SurRoL: An Open-source Reinforcement Learning Centered and dVRK Compatible Platform for Surgical Robot Learning," *arXiv:2108.13035*, 2021.

[33] A. Attanasio, B. Scaglioni, M. Leonetti, A. F. Frangi, W. Cross, C. S. Biyani, and P. Valdastri, "Autonomous Tissue Retraction in Robotic Assisted Minimally Invasive Surgery – A Feasibility Study," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6528–6535, Oct. 2020.

[34] R. Jansen, K. Hauser, N. Chentanez, F. Van Der Stappen, and K. Goldberg, "Surgical retraction of non-uniform deformable layers of tissue: 2D robot grasping and path planning," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 4092–4097.

[35] D. Meli, E. Tagliabue, D. Dall'Alba, and P. Fiorini, "Autonomous tissue retraction with a biomechanically informed logic based framework," *arXiv preprint arXiv:2109.02316*, 2021.

[36] T. D. Nagy, M. Takács, I. J. Rudas, and T. Haidegger, "Surgical subtask automation—Soft tissue retraction," in *2018 IEEE 16th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2018, pp. 000 055–000 060.

[37] A. Pore, D. Corsi, E. Marchesini, D. Dall'Alba, A. Casals, A. Farinelli, and P. Fiorini, "Safe Reinforcement Learning using Formal Verifica-

tion for Tissue Retraction in Autonomous Robotic-Assisted Surgery," *arXiv:2109.02323*, 2021.

[38] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep Convolutional Networks on 3D Point Clouds," *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 9613–9622, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8954200

[39] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," *IEEE Conf. on Computer Vision and Pattern Recognition*, 2017. [Online]. Available: https://arxiv.org/abs/1612.00593

[40] Y. Wu and K. He, "Group normalization," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[41] T. D. Kulkarni, A. Gupta, C. Ionescu, S. Borgeaud, M. Reynolds, A. Zisserman, and V. Mnih, "Unsupervised learning of object keypoints for perception and control," *Advances in Neural Information Processing Systems*, vol. 32, pp. 10 724–10 734, 2019. [Online]. Available: https://arxiv.org/abs/1906.11883

[42] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da Vinci® Surgical System," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, May 2014, pp. 6434–6439.

[43] B. Hinz, "Mechanical aspects of lung fibrosis: a spotlight on the myofibroblast," *Proc Am Thorac Soc*, vol. 9, no. 3, pp. 137–47, 2012. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/22802288/

[44] A. M. Handorf, Y. Zhou, M. A. Halanski, and W.-J. Li, "Tissue stiffness dictates development, homeostasis, and disease progression," *Organogenesis*, vol. 11, no. 1, pp. 1–15, 2015. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/25915734/

[45] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *Intl. Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[47] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," *arXiv preprint arXiv:1707.01495*, 2017.

[48] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[49] Z. Huang, Y. Hu, T. Du, S. Zhou, H. Su, J. B. Tenenbaum, and C. Gan, "Plasticinelab: A soft-body manipulation benchmark with differentiable physics," in *International Conference on Learning Representations*, 2021.