Interactive Perception for Deformable Object Manipulation

Zehang Weng*,1, Peng Zhou*,†,2, Hang Yin1, Alexander Kravberg1,
Anastasiia Varava1, David Navarro-Alarcon2 and Danica Kragic1

***Abstract*— Interactive perception enables robots to manipulate the environment and objects to bring them into states that benefit the perception process. Deformable objects pose challenges to this due to significant manipulation difficulty and occlusion in vision-based perception. In this work, we address such a problem with a setup involving both an active camera and an object manipulator. Our approach is based on a sequential decision-making framework and explicitly considers the motion regularity and structure in coupling the camera and manipulator. We contribute a method for constructing and computing a subspace, called Dynamic Active Vision Space (DAVS), for effectively utilizing the regularity in motion exploration. The effectiveness of the framework and approach are validated in both a simulation and a real dual-arm robot setup. Our results confirm the necessity of an active camera and coordinative motion in interactive perception for deformable objects.**

I. Introduction

Interactive Perception (IP) exploits various types of forceful interactions with the environment to facilitate perception [1]. Specifically, interaction allows for jointly considering acquired sensor information and actions taken over a time span. Despite the high dimensionality of the augmented space, the causal relation between sensation and action gives rise to structure for perceiving environment properties in a predictive and dynamical manner. Such structure is called Action Perception Regularity [1], and is believed to be the key for IP to reveal richer signals, which is impossible for passive and one-shot perception. IP has been shown effective in exploring and exploiting object and environment properties, with the main focus on rigid and articulated objects [2], [3], [4], [5], [6].

Deformable object manipulation (DOM) adds to the complexity of this identified interactive perception problem (e.g., clothes or bags manipulation). The space where the perception process resides in is further enlarged with much more degrees-of-freedom (DOFs) of the deformable materials. Establishing the predicative relation between action and sensation is challenged by an underactuated system. Moreover, significant occlusions may present due to material flexibility and fail an interaction strategy without an actively controlled camera. An example is shown in Fig. 1, where an active camera called *perceiver* is necessitated for a better observation of an object in a bag while the bag is opened by another manipulator called *actor*. This will again add more DOFs to the problem. Lastly, deformable environments might entail a perception process consisting of multiple steps: the robots need to take actions for better perception which in turn bases a better action decision to take. How to effectively coordinate the perceiver and the actor to shorten this interactive process can thus be a key to efficient perception. To this end, we propose to address interactive perception for deformable objects in a partially observable Markov Decision Process (POMDP) framework. We focus on bag-like objects and a dual-arm agent setup with an active camera similar to Fig. 1. Our aim is to study the feasibility of solving DOM for IP under the POMDP formulation and whether active vision can facilitate the solution. To leverage the regularity in coupling the perceiver and the actor, our approach proposes to dynamically construct a motion subspace, Dynamic Active Vision Space (DAVS), based on object features for an efficient search. Specifically, we introduce a manifold-with-boundary formulation to characterize a compact camera action subspace to constrain the potential next view. We provide detailed algorithms to calculate the manifold and integrate it into a general action search framework such as reinforcement learning (RL) in solving the POMDP. The simulation results highlight the importance of active vision capability from the controlled camera and the superior performance due to DAVS compared to baseline methods. DAVS shows its strength in generalizing to unseen dynamics and shapes. Finally, we show the proposed method is robust for a successful transfer to the real hardware with a moderate fine-tuning on a dual-arm setup.

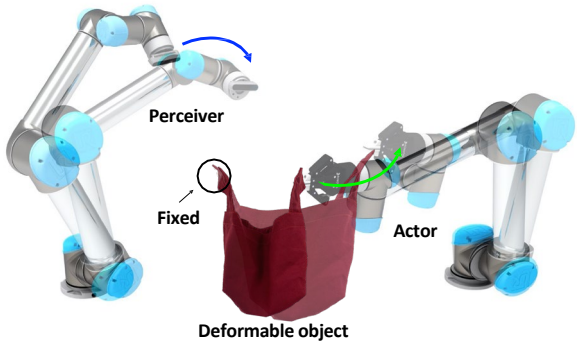

Fig. 1. An example of Interactive Perception. The perceiver (camera) is moved to a new viewpoint while the actor (end-effector) opens the bag for better perception of in-bag object.

In summary, our main contributions are:

- We propose a formulation involving both an active camera and a highly deformable object, which to our knowledge is the first in the context of interactive

*Authors with equal contribution. † Corresponding author.
[1]The authors are with CAS/RPL, KTH, Stocholm, Sweden. {zehang,hyin,okr,varava,dani}@kth.se
[2]The authors are with The Hong Kong Polytechnic University, KLN, Hong Kong. {jeffzhou@hku.hk, dnavar@polyu.edu.hk}

perception.

- We present a novel method that constructs the camera action space as a manifold with boundary (DAVS), based on Structure of Interest (SOI).
- We conduct extensive studies in both simulation and real-world experiments to demonstrate the effectiveness of our methods in challenging DOM scenarios, We show that our method generalizes well to objects with different dynamical properties and unseen shapes.

II. Related Work

Interactive Perception and manipulation of deformable objects [7], [8], [9], [10] are commonly studied as two separate problems in the robotics community. In this work, we aim to develop a framework that allows for a more effective perception in a manipulation task performed on a deformable object. Our work relates to concepts such as hand-eye coordination [11], visual servoing [12], active perception [13], [14], [15], [16], [17], [18] and interactive perception [1], but none of these fully addresses all the three aspects in a single framework: a deformable object, moving camera, and a robot interacting with the object.

Recent work on learning policies in the context of IP focuses mostly on rigid objects and RL [2], [3]. Works considering deformable objects and a moving camera reside commonly to active vision and do not address the interaction aspect [19], [20], [21], [22]. RL is again employed in the recent work of [23] but the focus is kept on resolving the observation for better human pose estimation with no interactive aspect. IP in the context of deformable objects has mainly been considered using a static camera [7]. The most important aspect of these works is to address the dimensionality in terms of perception using a latent representation and then devising action planning in a lower-dimensional space [24], [25], [26]. Notably, advancements in complex 3D deformable bag manipulation have been presented [27], [28], [29], [30], [31]. These works have shown significant progress in the domain but do not incorporate an active movable camera for task achievement. The recent benchmarks that build upon advanced simulation engines again focus primarily on a static camera setup [32], [33].

The work in [11] learns hand-eye coordination for grasping, using deep learning to build a grasp success prediction network, and a continuous servoing mechanism to use this network to continuously control a robotic manipulator. Similarly to our approach, it uses visual servoing to move into a pose that maximizes the probability of success on a given task. However, the application is bin-picking, using a static camera, and no deformable objects are considered. As per technical methods, we also follow a decision-making formulation and reinforcement learning as [2], [3]. Our work differs by contributing a factorization of the policy action space and algorithms that leverage the dependency between the camera and object for efficient search. The proposed structure is shown to be critical to addressing challenges faced by [2] in deformable object scenarios.

In summary, we contribute a formulation of an interactive perception approach that relies on a manifold-with-boundary representation to address the high-dimensionality of the perception-action problem and efficiently encode the coupling between the perceiver, deformable object, and the actor.

III. Methodology

We formulate interactive perception involving deformable objects as a POMDP (S, O, A, T, R, γ), with state $s \in S$ denoting the configuration of robotic end-effector, camera, and object, and corresponding observation as $o \in O$. State transition model $T(s' \mid s, a)$ characterizes the probability of transitioning to state s' from taking action $a \in A$ under state s. The action space is comprised of desired camera and end-effector poses, represented as a Cartesian product $A = A_{cam} \times A_e$. $R(s) \in \mathbf{R}$ is a state reward function, and $\gamma \in [0, 1)$ is the discount factor.

Our core idea to tackle the problem is constructing a subspace of A for efficient action search and maximizing the accumulated reward. An overview of the framework components is depicted in Fig. 2. Specifically, the subspace is built upon certain object features, which are called the structure of interest (SOI). The geometry of this structure is coupled with the end-effector action. The structure is used to generate the temporally varying boundary of a manifold, which is called dynamic active vision space. In the end, the camera actions are projected to the constructed space in a reinforcement learning process. We detail the mathematical representation of the manifold and the development of each component in the following sections.

A. Action Space Factorization and Manifold with Boundary

Given the IP task in Fig. 1, we need to propose the camera action a_t^{cam} and end-effector action a^e given the observation o_{t-1}. A naive solution is to factorize the action space as $\pi(a^{cam}_t, a^e_t | o_{t-1}) = p(a^{cam}_t | o_{t-1}) p(a^e_t | o_{t-1})$ with the assumption of conditional independence, which however results in high dimensionality for action exploration. In this work, we consider that the camera action space should also depend on the previous end-effector action, resulting in $\pi(a^{cam}_t, a^e_t | o_{t-1}) = p(a^{cam}_t | a^e_{t-1}, o_{t-1}) p(a^e_t | o_{t-1})$. This implies the possibility of sampling a^{cam}_t from a subspace depending on a^e_{t-1}, thereby enabling efficient exploration of the original action space. We use the *manifold with boundary* M [34] as a tool to describe this subspace and its construction.

The introduction of M essentially imposes constraints, and thus structure/regularity, on the original camera action space. To this end, we augment the POMDP (S, O, Z, T, R, γ), such that $z \in Z = M \times A^e$ is the constrained action space subjecting to M. The new constrained IP problem can be written as

$$\max_{\theta} \mathrm{E}_{s_t, z^{cam}_t, z^e_t} \left[\sum_{t=0}^{T} \gamma^t R(s_t, z^{cam}_t, z^e_t) \right], \quad s.t. \quad z^{cam}_t \in M(z^e_{t-1}) : \{\mathrm{Int} M, \partial M\} \tag{1}$$

with θ as the learning parameters of model M and $\pi(z^e | o)$.

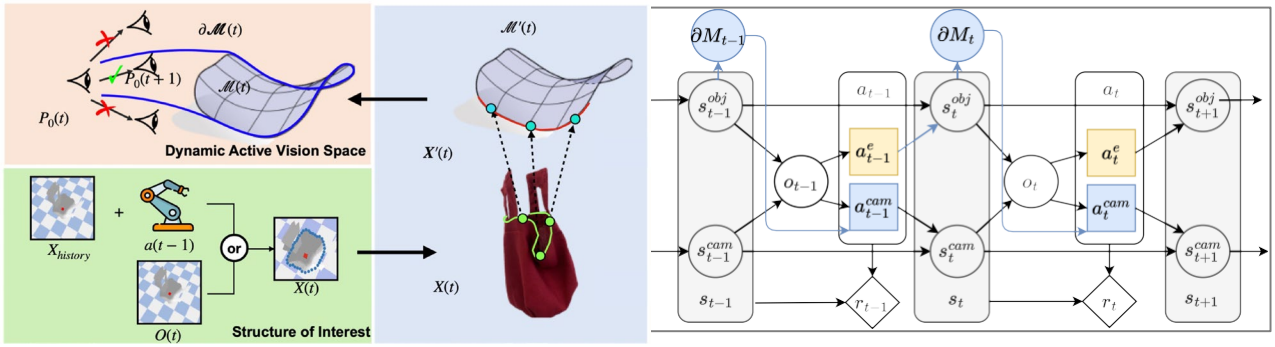

Fig. 2. [Left] Illustration of the proposed framework — a subspace of camera action is constructed and represented with manifold with boundary, accounting for the coupling with end-effector motion via the structure of interest (SOI) on deformable objects. [Right] Illustration of state action transition of the proposed framework.

B. Manifold with Boundary-based DAVS

The manifold with boundary specifies a compact subspace of the original camera action space. We adopt a half spherical space as the original space, which is often assumed in active perception research [35], [36], [23]. Specifically, the spherical space is defined by viewpoint centroid, V_0 and a viewpoint radius r, with both as known parameters: $S^2(r) = \{p \in \mathrm{R}^3 \mid \|p - V_0\| = r, z \geq 0\}$.

We seek to identify a subspace of S^2 to exploit the regularity of camera action space. This subspace is called Dynamic Active Vision Space (DAVS) for its dependency on the observation and end-effector action at each time step. To model such a dependency, we propose to use object features as an intermediary which are in turn influenced by end-effector actions. Specifically, we consider a set of key-points $\mathbf{X}(t) = [X_1(t), X_2(t), \cdots, X_N(t)]^T$ with $X_i(t) \in \mathrm{R}^{1\times 3}$ as a representation of task-dependent features, such as garment opening and handle loop in Fig. 2, which are named as structure of interest (SOI). SOI can be acquired by using key-point extraction techniques [26], [37] or predicative models [38], [27].

To focus on the contributed method, we assume annotated key-points that can be readily retrieved from a model $f_{SOI}(o_t)$. We expect SOI to resemble a single loop structure for the computation in subsequent implementation. This can be satisfied by design in the cases of, for instance, vertices on bag openings. To ensure such a structure, we process raw SOI key-points $\mathbf{X}(t)$ by first projecting them on a middle sphere with a radius $r^* = max(\|X_i(t) - V_0\|)$, with the projected points defined as:

$$\bar{\mathbf{X}}(t) = V_0 + \frac{(V_0 - X_i(t))r^*}{\|X_i(t) - V_0\|} \mid X_i(t) \in \{X_1(t), X_2(t), \ldots, X_N(t)\} = \mathbf{X}(t) \tag{2}$$

Selecting the projected points $\mathbf{X}^*(t)$ on the convex hull of $\bar{\mathbf{X}}(t)$, we obtain a SOI polygon with desired loop structure.

We construct DAVS based on this SOI polygon to compute a subset of the original camera action space S^2. A subset of S^2 at each time step is built by projecting key-points $\mathbf{X}^*(t)$ to $\mathbf{X}'(t)$ with a ray-tracing model, as shown in Fig. 3:

$$\mathbf{X}'_i(t) = V_0 + \frac{(V_0 - X^*_i(t))r}{\|X_i(t) - V_0\|} \mid X^*_i(t) \in \{X^*_1(t), X^*_2(t), \ldots, X^*_n(t)\} = \mathbf{X}^*(t) \tag{3}$$

By connecting the n projected SOI points in their respective order with N geodesic paths, we generate a manifold M' with boundary $\partial M'$ (which is part of S^2) consisting of the piecewise geodesic paths. In our implementation, we represent the geodesic path as a discrete set of path points within a certain density range to accelerate the computation process [25].

Algorithm 1: Action Exploration with DAVS

Input: Observation, o; Camera position, P_0; Viewpoint centroid, V_0; Viewpoint radius, r.
Output: Camera action policy, α^{cam}_k; End-effector action policy, α^e_k ;

```
for each episode do
    Initial feasible state s_0 ;
    Initial parameter setting for active vision ;
    for each time step k do
        SOI Modeling X(t) ← f_SOI(o_{k-1}) ;
        Sample policy end-effector action
          α^e_k ~ π(· | o_{k-1}) ;
        Based on X(t), P_0, V_0, and r, compute DAVS
          using Alg. (2) to get M with its boundary
          ∂M ;
        Compute direction vectors v_1 and v_2 based on
          geodesic paths on γ_{P_0P_1} and γ_{P_0P_2} ;
        Sample policy camera action
          α^cam_k ~ π(· | o_{k-1}, v_1, v_2, v_0, ω)_cam ;
        Apply the control camera action α_k to the
          environment ;
        Construct action tuple α_k = (α^e_k, α^cam_k) ;
        Get next observation o_{k+1} and reward r_k from
          the environment ;
        Provide the transition tuple (o_{k-1}, α_k, o_k, r_k)
          to the RL algorithm ;
```

A DAVS for the current camera pose is constructed by finding a centroid on $M'(t)$, which is calculated as the Karcher mean of the boundary points $\mathbf{X}'(t)$, see Fig. 3. Formally, (M', d) defines a complete metric space and d

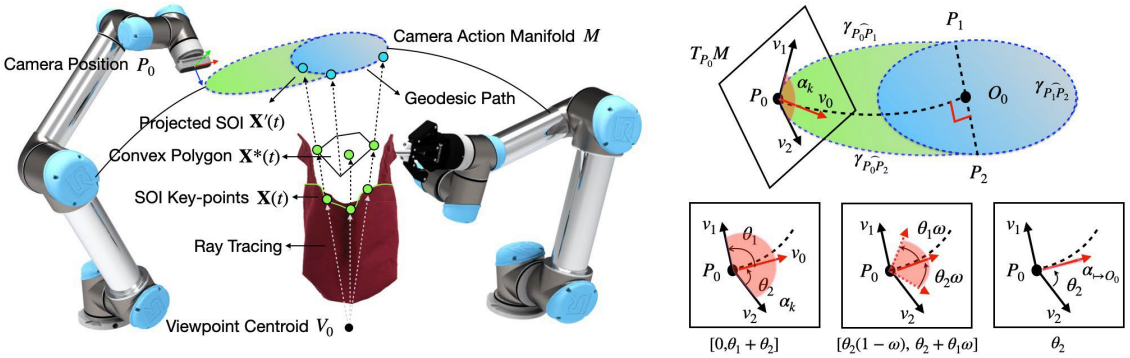

Fig. 3. Illustration of the process of dynamic active vision space (DAVS) generation. [Left] We extract the SOI points at each time step t and convert them to projected 3D SOI (blue) points through ray tracing, on the original camera action manifold. [Right top] The manifold with boundary based on projected SOI and current camera position by Algorithm 2 [Right bottom] Parameterized exploration space.

Algorithm 2: Dynamic Active Vision Space (DAVS) Generation

Input: SOI set (ring vertices), $\mathbf{X}(t)$; Camera position, P_0; Viewpoint centroid, V_0; Viewpoint radius, r.

Output: Manifold M with boundary ∂M representing the reshaped camera action space

1 Compute $\mathbf{X}_i'(t)$ with ray tracing model using Eq. (3) ;
2 Generate refined active vision space, $M'(t)$ and its boundary $\partial M'(t)$;
3 Compute centroid O_0 using the trust-region method with Eq. (5) ;
4 Compute P_1 and P_2 s.t. $Geodesic_{P_1P_2} \perp Geodesic_{P_0O_0}$;
5 Form manifold with boundary M s.t. boundary $\partial M = \{p \mid p \in \gamma_{\widehat{P_0P_1}}, \gamma_{\widehat{P_1P_2}}, \gamma_{\widehat{P_2P_0}}\}$ (each p has a neighborhood homeomorphic to an open subset of *closed n-dimensional upper half-space* H^n.

denotes the geodesic distance in this metric space. For any position p in M', the Fre´chet variance Ψ is defined to be the sum of squared distances from p to the X_i' and is minimized as:

$$f_{O_0}(p) = \underset{p\in M}{\operatorname{argmin}} \Psi(p) = \underset{p\in M}{\operatorname{argmin}} \sum_{i=1}^{N} d^2(p, X_i) \tag{4}$$

To efficiently solve this problem, we follow [39] to transform it into a trust-region subproblem on $T_{p_k}M'$. Given the cost function $f_{O_0} : M' \to \mathrm{R}$ and a current iterate $p_k \in M'$, we use retraction R_{p_k} to locally map the minimization problem for f on M into a minimization problem for its pullback $\hat{f}_{O_0}(p_k) : T_{p_k}M' \to \mathrm{R}$, with which the trust-region problem reads:

$$\begin{aligned} \min_{\eta\in T_{p_k}M} \hat{f}_{p_k}(\eta) = {} & f(p_k) + \langle \operatorname{grad} f(p_k), \eta\rangle \\ & + \frac{1}{2}\langle H_k[\eta], \eta\rangle \ \text{s.t.}\ \langle \eta, \eta\rangle_{p_k} \le \Delta_k^2 \end{aligned} \tag{5}$$

For the symmetric operator H_k we use Hessian $\operatorname{Hess} f(x)[\xi] = \nabla_\xi \operatorname{grad} f(x)$. A trust region is specified by Δ_k for the validity of ambient space approximation.

In addition to the centroid, we find two points P_1 and P_2 on $\partial M'(t)$, such that the geodesic path passing through them is locally perpendicular to the one passing through P_0 and O_0. As a result, the DAVS is obtained by enclosing the three paths, $\gamma_{\widehat{P_0P_1}}$, $\gamma_{\widehat{P_1P_2}}$, and $\gamma_{\widehat{P_0P_2}}$. The full process is summarized as Algorithm 2. Additionally, we introduce a ω to parameterize the exploration space of the camera's action. As illustrated in Fig. 3, consider the tangent space $T_{P0}M$ at the current camera position P_0 to a DAVS. Here, the local geodesic paths $\gamma_{\widehat{P_0P_1}}$, $\gamma_{\widehat{P_0P_2}}$ and $\gamma_{\widehat{P_0O_0}}$ are projected onto this tangent space, resulting in vectors v_1, v_2, and v_0. Let us assume that angles θ_1 and θ_2 are formed between v_1 and v_0, and v_2 and v_0, respectively. By introducing a parameter $\omega \in [0, 1]$, we gain control over the camera action exploration within the range $[\theta_2(1-\omega), \theta_2+\theta_1\omega]$ (v_2 as zero-degree axis). In our manuscript, we select the action space with $\omega = 1$ to provide a broader space for exploring effective policies in complex scenarios. By reducing ω, we can narrow the action sampling space and enforce a preference for alignment with v_0, which corresponds to the direction along the geodesic path $\gamma_{\widehat{P_0O_0}}$. Reducing ω to 0 causes the camera action policy to strictly follow the geodesic path $\gamma_{\widehat{P_0O_0}}$, leading directly to the centroid of DAVS.

C. Exploiting DAVS in Action Exploration

The construction of DAVS provides a subspace that associates promising camera actions to task-relevant object features under the end-effector action. To this end, we propose to exploit this space to bias the search of camera action in IP, e.g., in reinforcement learning. As shown in Fig. 2, we determine two tangent directions, v_1 and v_2, based on the boundary of DAVS at the current camera position. The camera action is sampled with a direction between v_1 and v_2. This biases the action exploration towards the DAVS and as such can boost search efficiency when object features indicate rewarding areas. We detail the entire process as Alg. 1 in the context of action sampling in reinforcement learning.

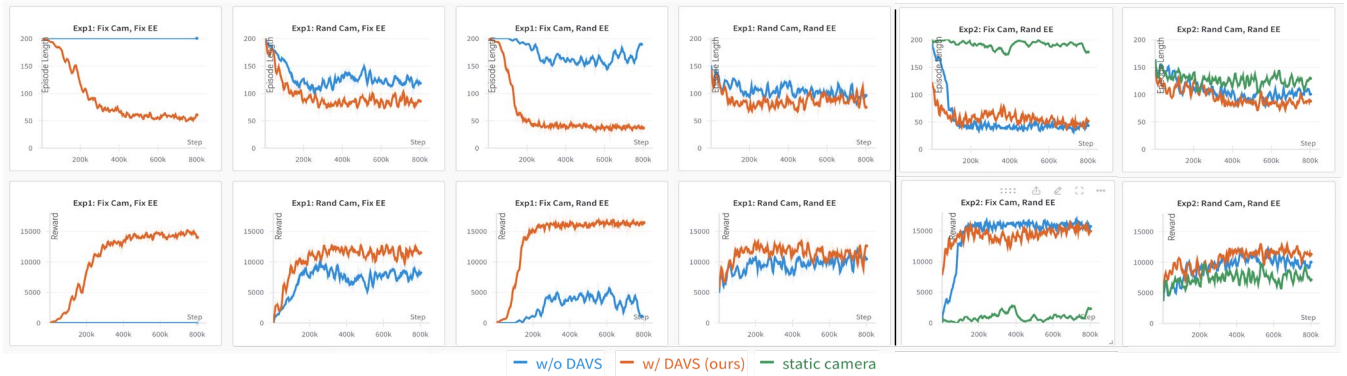

Fig. 4. Left: quantitative evaluation on CubeBagClean. The first row represents the **episode length** (↓) for solving 4 tasks across different combinations of random (Rand) and fixed (Fix) camera (Cam) and end-effector (EE) birth modes. The second row shows the **total discounted rewards** (↑). Right: quantitative evaluation on CubeBagObst subtasks. The first two figures represent the **episode length** (↓) and **reward** (↑) for the scenario with fixed camera (Fix Cam) and random end-effector (Rand EE) birth modes. The third and fourth figures are **episode length** (↓) and **reward** (↑) with random camera (Rand Cam) and random end-effector (Rand EE) birth modes.

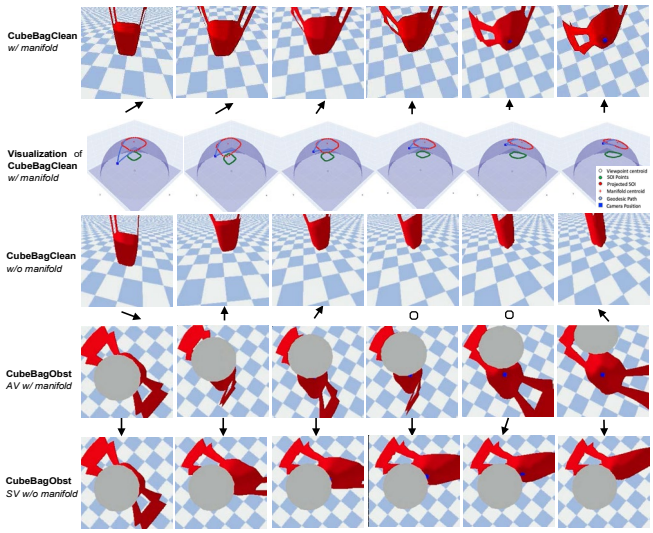

Fig. 5. Visualization of the learned hand-eye policies. In each row except for the second row, we show the images of intermediate frames in an example episode. We also draw arrows ↑ beneath the images to illustrate how the camera moves. In the first and third rows sampled from the **CubeBagClean** case, we compare the performance between IP methods with and without DAVS. We can see that with DAVS, the camera is able to move right and upwards to find the cube in the bag, while the one without DAVS searches more randomly. In the bottom two rows from the **CubeBagObst** case, we compare the methods between IP methods with DAVS and a static vision method. In the IP setting, the camera bypasses the obstacle and finds a feasible solution, while the static vision method doesn't. This reveals the necessity of allowing active vision and active end-effector. The second row is the 3D manifold visualization corresponding to the first row (BlueDot: Camera; Green: 3D SOI; Red: Projected SOI; BlueCurve: Boundary of DAVS).

IV. Simulation Experiments

A. Environment Setup

Our simulation environment is built on the basis of Pybullet [40] which has been previously extended to handle various cloth-like deformable object manipulation tasks [33], [41] under static viewpoint camera. We advance one step further and customize the environment to support the empirical evaluation of the proposed IP framework as shown in Fig. 3.

We evaluate the performance of the proposed framework with DAVS in two environments: (1) CubeBagClean: environment contains a deformable bag with the left handle fixed in the air and a rigid cube placed inside; (2) CubeBagObst: CubeBagClean with an additional circular obstacle above the bag, providing an additional view occlusion. To ensure diversity of the initial position of the cube, it is spawned and released from a certain height above the bag, falling down into the bag freely.

In both simulation environments, we provide three types of observations: a 2D depth image, a 2D SOI heatmap, and the end-effector position. Based on the observations, the camera and end-effector manipulating the bag are allowed to act simultaneously, but restricted to a maximum of 200 action steps in one episode. Specifically, the end-effector can be arbitrarily moved while the active camera is traversing the half-sphere surface, similar to the CMU's panoptic massive camera grid [42] setting. The perceiver agent is controlled by pitch and yaw angles and stares at a constant focus point. To avoid collisions and ensure the simulation stability, we constrain the end-effector action with a bounding box and the camera yaw angle with a range.

B. Evaluation Metrics

We use CubeBagClean and CubeBagObst to assess the efficiency of using the proposed action space, in terms of succeeding in finding the cube. In CubeBagClean, we vary the birth modes of the camera and the end-effector to create 4 subtasks where either the camera or the end-effector can be spawned in a fixed or a random place. In CubeBagObst, we consider 2 subtasks with both fixed and random camera birth modes while the end-effector is always initialized in a random position. The fixed camera position is selected in a way that the bag interior is not immediately visible, so non-trivial camera trajectories are required to solve the task. In each task, our reward function encodes a goal of maximizing the visibility of the rigid cube and the opening ring. We use a potential-based reward formulation [43] to encourage revealing more of the visible cube pixels and the SOI. The

reward function is as follows:

$$r_t = \lambda_r \Delta A_t^{\text{rigid}} + \lambda_d \Delta N_t^{\text{ring}} + \lambda_f \mathbf{I}(A_t^{\text{rigid}}, A^{\text{ring}}, t) \quad (6)$$

where $\Delta A_t^{\text{rigid}} = A_t^{\text{rigid}} - A_{t-1}^{\text{rigid}}$ is the increased visibility of rigid object pixels in the image captured by the active camera. λ_r is a scale compensation factor depending on the relative distance between the object and the camera. $\Delta N_t^{\text{ring}} = N_t^{\text{ring}} - N_{t-1}^{\text{ring}}$ is the increased number of visible deformable SOI vertices in the active camera view. λ_d is a constant scale factor for the deformable object. $\mathbf{I}(\cdot)$ is an indicator function that checks if the IP policy has reached the final goal. Specifically, $\mathbf{I}(\cdot)$ returns 1 only if A_t^{rigid} and A_t^{ring} both reach the preset visibility thresholds before timeout. λ_f is a time-dependent scale factor, defined as $100 \times (T_{\max} - t)$, encouraging the IP policy to finish the task as quickly as possible. To demonstrate the effectiveness of DAVS in comparison to the state-of-the-art, we choose a model-free RL algorithm PPO [44] as the policy search algorithm. We provide quantitative evaluations regarding the finishing episode length and the final reward, and qualitative results illustrating the learned policy behaviors among different methods.

C. Baseline Methods

We choose the PPO [44] and a straightforward Visual Servoing (VS) approach that focuses on maximizing the visibility of the SOI within the camera's view as our baselines. The 2D depth and heatmap images are concatenated, and embedded as a fix-length feature vector through a convolution neural network architecture from [45]. The image feature vector and the end-effector location vector are concatenated and used as PPO input. The output camera and end-effector actions thus depend on both current image features and end-effector position.

D. Result

1) CubeBagClean: Fig. 4 shows the learning process on 4 subtasks, comparing PPO without DAVS to the variant with the proposed DAVS on the metrics of the finished episode length and total discounted rewards. It is evident that the performance of the baseline is improved by a significant margin by introducing DAVS (with manifold), achieving lower episode length and higher reward in all four subtasks. This indicates that DAVS successfully encodes the coordination between the camera and end-effector agents and expedites the finding of better solutions. We also visualize an example of the learned IP policies in Fig. 5. The last row of the figure illustrates the dynamic evolution of the manifold in a CubeBagClean episode in the first row. The camera position, SOI, projected SOI, and DAVS are plotted in different colors. For each subtask, we report the episode length, total reward, and the success rate on 100 random scenes as presented in Table I.

2) CubeBagObst: The additional circular obstacle in this scenario introduces more view occlusion and hence requires more complex maneuver and coordination of camera and end-effectors. In this experiment, besides a default PPO with and without DAVS, we also take a PPO with static vision into account for comparison to show the insufficiency of static vision methods. From Fig. 4, we observe the agents in IP with DAVS outperform the counterparts without DAVS or with the static vision solution, particularly when the camera starts from a fixed but challenging side perspective. The qualitative result is shown in Fig. 5.

3) Generalization to objects with new properties: To evaluate the trained networks' generalizability, we analyzed their performance when interacting with bags exhibiting unseen yet similar dynamics, despite the model being trained with a single-bag dynamic setup. Specifically, we explored the impact of random variations in the bag's spring elastic stiffness and damping stiffness on its dynamics.

For each CubeBagClean subtask, we created 100 scenes with random bag dynamics to evaluate the models' performance, with and without DAVS. As presented in Table I, the IP policy, when utilizing DAVS, not only consistently outperforms its counterpart without DAVS but also demonstrates superior generalizability, underscoring the robustness and effectiveness of our methodology.

Furthermore, we investigated the impact of variations in shape, which were not seen during training. Specifically, we created scenes with 36 bags with shapes adapting from [33]. As shown in Table I, the IP policy with DAVS performs better. We observed that certain tasks with unseen shapes were easier to achieve due to two reasons: 1) some new bags had large openings, making it easier to capture the object inside during manipulation; 2) some new bags were more shallow, causing the contained cube to be closer to the camera. Consequently, the rigid cube area appeared relatively larger than in the original setup in the captured image, making reaching the pre-set cube area threshold easier.

4) Comparison with other baselines: As shown in Table II, with the reduction of ω to shrink the exploration space for camera actions, the camera is more likely to move directly into the manifold centroid. The PPO policy with DAVS achieves shorter episode lengths, higher total rewards, and higher success rates. While $\omega = 0$ significantly improves performance in simpler scenarios (CubeBagClean), higher values of ω ($\text{DAVS}_{\omega=0.5}$ and $\text{DAVS}_{\omega=1}$) are crucial for handling more complex environments (CubeBagObst). The Visual Servoing method (VS) performs better than $\text{DAVS}_{\omega=1}$ in some aspects but is outperformed by DAVS with ω settings on CubeBagObst. In general, $\text{DAVS}_{\omega=1}$ achieves the best overall balance, making it the most robust approach among the tested methods. We also examined the failure cases in the simulation experiments. In our Pybullet simulation setup, we executed the movements of the camera and the actor simultaneously. Most failures occurred when the simulated gripper moved too quickly, causing overstretching of the deformable handles. Additionally, the relatively slow movement of the camera made it more difficult to locate the object, especially in scenes involving obstacles. Furthermore, under the challenging generalization test, the IP method failed particularly when the stiffness differed significantly or with a small bag opening.

Dynamics	Method	Episode Length (↓) / Reward (↑) / Success Rate(%) (↑)			
		Fix Cam, Fix EE	Rand Cam, Fix EE	Fix Cam, Rand EE	Rand Cam, Rand EE
Seen	w/o DAVS	200.0 / 0.3 / 0.0	104.4 / 9561.8 / 66.0	164.7 / 3537.0 / 23.0	110.5 / 8958.9 / 49.0
	w/ DAVS (ours)	**39.0 / 16107.1 / 100.0**	**64.0 / 13600.7 / 84.0**	**35.24 / 16483.2 / 95.0**	**90.1 / 10998.6 / 61.0**
New Dynamics	w/o DAVS	200.0 / 0.2 / 0.0	116.4 / 8359.0 / 59.0	152.0 / 4796.1 / 30.0	114.5 / 8557.1 / 46.0
	w/ DAVS (ours)	**54.4 / 14565.6 / 91.0**	**84.6 / 11541.6 / 73.0**	**39.7 / 16035.9 / 94.0**	**80.7 / 11943.3 / 66.0**
New Shapes	w/o DAVS	188.8 / 1125.0 / 11.9	122.5 / 7750.1 / 50.5	95.6 / 10439.2 / 70.5	101.5 / 9859.9 / 58.8
	w/ DAVS (ours)	**97.9 / 10213.0 / 66.4**	**98.2 / 10184.4 / 63.1**	**38.7 / 16141.4 / 93.4**	**72.7 / 12734.3 / 80.6**

TABLE I

PERFORMANCE ACROSS VARIED CAMERA AND END-EFFECTOR SETTINGS

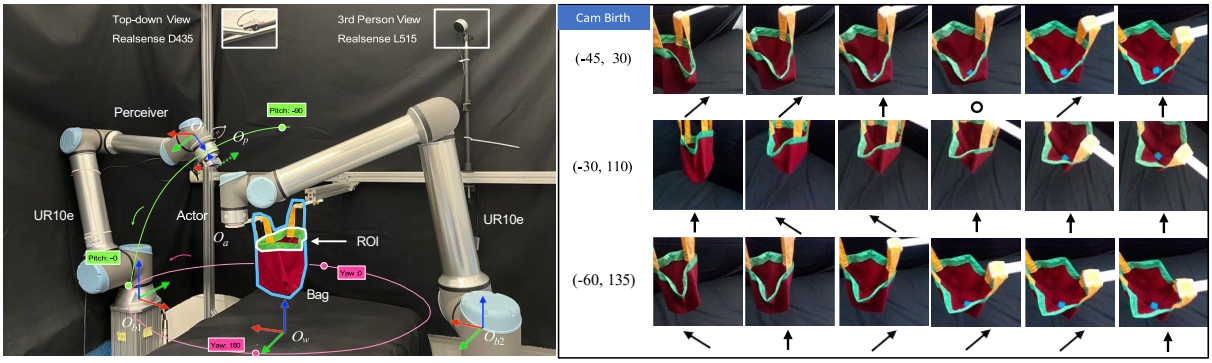

Fig. 6. (Left) The experimental set-up of active perception for deformable object manipulation; (Right) The real-world experimental results with different camera birth modes. Arrows ↑ beneath the images illustrate how the camera moves.

Method	Episode Length (↓) / Reward (↑) / Success Rate(%) (↑)		
	CubeBagClean	CubeBagObst	Total(%) (↑)
w/o DAVS	137.6 / 6248.0 / 36.0	87.2 / 11539.5 / 65.0	50.5
DAVS$_{\omega=0}$	**32.0 / 16824.8 / 99.0**	143.8 / 5739.8 / 36.0	67.5
DAVS$_{\omega=0.5}$	59.6 / 13950.7 / 82.0	90.4 / 10760.6 / 59.0	71.5
DAVS$_{\omega=1}$	62.7 / 13740.9 / 78.0	**68.3 / 13536.2 / 75.0**	**76.5**
VS	35.4 / 16107.7 / 96.0	137.5 / 5348.4 / 34.0	65.0

TABLE II

PERFORMANCE BY DIFFERENT APPROACHES

V. REAL-ROBOT EXPERIMENTS

To assess the effectiveness of the policy trained using DAVS for deformable object manipulation with active perception, we selected a policy fine-tuned within the CubeBagClean simulation environment (featuring random camera and end-effector positions) for real-world experimentation. Figure 6 illustrates our real-world experimental configuration for active perception in deformable object manipulation tasks. Here, a UR10e robotic arm (the actor) manipulates one handle of a fabric bag (with the other handle fixed) using a 3D-printed end-effector. Concurrently, a second UR10e robotic arm (the perceiver) equipped with a RealSense D435 depth camera observes the environment. We manually measured the transformation matrices between the world coordinate frame O_w and the perceiver's base frame O_{b1}, as well as between O_w and the actor's base frame O_{b2}. Mirroring the camera perceiver configuration from our PyBullet training environment, the camera's action space is limited to a hemispherical area, with its orientation constantly aimed at the sphere's center—which also coincides with the origin of the world frame O_w. Given the known central point of the camera's field of view and the sphere's radius, the camera's position and orientation are straightforwardly defined by its pitch and yaw angles. Taking into account the kinematic constraints, these angles are set within the ranges of [-89, -30] degrees for pitch and [45, 135] degrees for yaw. The inputs to our real RL agent comprise the end-effector pose, depth image and a ring-shaped heat map. At each step, the DAVS generates a construction based on the captured green Region of Interest (ROI) of the fabric bag, which is defined by the bag's green opening rim. This green rim is roughly identified by an overhead camera and then transformed into the world frame using a fixed transformation. The outputs generated are the pitch, yaw, and relative motion actions of the actor. Following approximately 5,000 episodes of fine-tuning, the perceiver and actor synergistically cooperate to reliably locate a blue cube within the bag. Figure 6 showcases the successful outcomes of these real-world experiments.

VI. CONCLUSION

In this work, we address deformable object manipulation through Interactive Perception in a system involving an active camera, a robotic end-effector, and a deformable object. We contribute to a novel formulation of the problem with common model assumptions and enable tractable computation through our proposed framework. Then, we perform simulation experiments with different view occlusion situations and demonstrate that our proposed framework with DAVS outperforms the state-of-the-art methods. Finally, we confirm that our method generalizes well to bags featuring previously unseen dynamical properties, as well as their efficacy within an actual real-world interactive perception system. The challenges presented in this work lead to the future directions of improving hand-eye coordination, such as explicitly incorporating dynamics prediction within

the IP framework, and more effective parameterized DAVS constructions with task dependency for deformable object manipulation.

REFERENCES

[1] J. Bohg, K. Hausman, B. Sankaran *et al.*, "Interactive perception: Leveraging action in perception and perception in action," *IEEE Trans. Robot.*, vol. 33, no. 6, pp. 1273–1291, 2017.

[2] R. Cheng, A. Agarwal, and K. Fragkiadaki, "Reinforcement learning of active vision for manipulating objects under occlusions," in *Conf. Rob. Learn.* PMLR, 2018, pp. 422–431.

[3] T. Novkovic, R. Pautrat *et al.*, "Object finding in cluttered scenes using interactive perception," in *IEEE Int. Conf. on Robotics and Automation*. IEEE, 2020, pp. 8338–8344.

[4] R. M. Martin and O. Brock, "Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors," in *IEEE/RSJ Int. Conf. on Robots and Intelligent Systems*. IEEE, 2014, pp. 2494–2501.

[5] D. Katz and O. Brock, "Manipulating articulated objects with interactive perception," in *IEEE Int. Conf. on Robotics and Automation*. IEEE, 2008, pp. 272–277.

[6] D. Katz, A. Orthey, and O. Brock, "Interactive perception of articulated objects," in *Experimental Robotics*. Springer, 2014, pp. 301–315.

[7] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Sci. Robot.*, vol. 6, no. 54, p. eabd8803, 2021.

[8] V. E. Arriola-Rios, P. Guler, F. Ficuciello, D. Kragic, B. Siciliano, and J. L. Wyatt, "Modeling of deformable objects for robotic manipulation: A tutorial and review," *Front. Robot. AI*, vol. 7, p. 82, 2020.

[9] J. Zhu, A. Cherubini, C. Dune, D. Navarro-Alarcon, F. Alambeigi, D. Berenson, F. Ficuciello, K. Harada, X. Li, J. Pan *et al.*, "Challenges and outlook in robotic manipulation of deformable objects," *arXiv preprint arXiv:2105.01767*, 2021.

[10] R. Herguedas, G. Lo´pez-Nicola´s, R. Aragu¨e´s, and C. Sagu¨e´s, "Survey on multi-robot manipulation of deformable objects," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 977–984.

[11] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. J. Robot. Res.*, vol. 37, no. 4-5, pp. 421–436, 2018.

[12] D. Kragic and H. I. Christensen, "Survey on visual servoing for manipulation," COMPUTATIONAL VISION AND ACTIVE PERCEPTION LABORATORY, Tech. Rep., 2002.

[13] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," vol. 1, no. 4, pp. 333–356, 1988.

[14] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos, "Revisiting active perception," *Auton. Robots*, vol. 42, no. 2, pp. 177–196, 2018.

[15] A. L. Yuille and A. Blake, *Active vision*. MIT Press, 1992.

[16] D. H. Ballard, "Animate vision," *Artificial intelligence*, vol. 48, no. 1, pp. 57–86, 1991.

[17] E. Rivlin and H. Rotstein, "Control of a camera for active vision: Foveal vision, smooth tracking and saccade," vol. 39, no. 2, pp. 81–96, 2000.

[18] X. Zhang, D. Wang, S. Han, W. Li, B. Zhao, Z. Wang, X. Duan, C. Fang, X. Li, and J. He, "Affordance-driven next-best-view planning for robotic grasping," *arXiv preprint arXiv:2309.09556*, 2023.

[19] J. Sock, S. Hamidreza Kasaei, L. Seabra Lopes, and T.-K. Kim, "Multi-view 6d object pose estimation and camera motion planning using rgbd images," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 2228–2235.

[20] S. Wenhardt, B. Deutsch *et al.*, "Active visual object reconstruction using d-, e-, and t-optimal next best views," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* IEEE, 2007, pp. 1–7.

[21] P.-P. Va´zquez, M. Feixas, M. Sbert, and W. Heidrich, "Viewpoint selection using viewpoint entropy." in *VMV*, vol. 1. Citeseer, 2001, pp. 273–280.

[22] H. Van Hoof, O. Kroemer *et al.*, "Maximally informative interaction learning for scene exploration," in *IEEE/RSJ Int. Conf. on Robots and Intelligent Systems*. IEEE, 2012, pp. 5152–5158.

[23] E. Ga¨rtner, A. Pirinen, and C. Sminchisescu, "Deep reinforcement learning for active human pose estimation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 07, 2020, pp. 10 835–10 844.

[24] M. Lippi, P. Poklukar, M. C. Welle, A. Varava, H. Yin, A. Marino, and D. Kragic, "Latent space roadmap for visual action planning of deformable and rigid object manipulation," in *IEEE/RSJ Int. Conf. on Robots and Intelligent Systems*. IEEE, 2020, pp. 5619–5626.

[25] P. Zhou, J. Zhu, S. Huo, and D. Navarro-Alarcon, "LaSeSOM: A latent and semantic representation framework for soft object manipulation," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5381–5388, 2021.

[26] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "kpam: Keypoint affordances for category-level robotic manipulation," *arXiv preprint arXiv:1903.06684*, 2019.

[27] Z. Weng, F. Paus, A. Varava, H. Yin, T. Asfour, and D. Kragic, "Graph-based task-specific prediction models for interactions between deformable and rigid objects," in *IEEE/RSJ Int. Conf. on Robots and Intelligent Systems*. IEEE, 2021, pp. 5741–5748.

[28] A. Bahety, S. Jain, H. Ha, N. Hager, B. Burchfiel, E. Cousineau, S. Feng, and S. Song, "Bag all you need: Learning a generalizable bagging strategy for heterogeneous objects," *arXiv preprint arXiv:2210.09997*, 2022.

[29] Z. Xu, C. Chi, B. Burchfiel, E. Cousineau, S. Feng, and S. Song, "Dextairity: Deformable manipulation can be a breeze," *arXiv preprint arXiv:2203.01197*, 2022.

[30] L. Y. Chen, B. Shi *et al.*, "Autobag: Learning to open plastic bags and insert objects," in *IEEE Int. Conf. on Robotics and Automation*. IEEE, 2023, pp. 3918–3925.

[31] P. Zhou, P. Zheng, J. Qi, C. Li, C. Yang, D. Navarro-Alarcon, and J. Pan, "Bimanual deformable bag manipulation using a structure-of-interest based latent dynamics model," *arXiv preprint arXiv:2401.11432*, 2024.

[32] X. Lin, Y. Wang, J. Olkin, and D. Held, "Softgym: Benchmarking deep reinforcement learning for deformable object manipulation," *arXiv preprint arXiv:2011.07215*, 2020.

[33] R. Antonova, P. Shi, H. Yin, Z. Weng, and D. K. Jensfelt, "Dynamic environments with deformable objects," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021.

[34] J. Lee, *Introduction to topological manifolds*. Springer Science & Business Media, 2010, vol. 202.

[35] B. Calli, W. Caarls, M. Wisse, and P. P. Jonker, "Active vision via extremum seeking for robots in unstructured environments: Applications in object recognition and manipulation," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 4, pp. 1810–1822, 2018.

[36] J. A. Gibbs, M. P. Pound, A. P. French, D. M. Wells, E. H. Murchie, and T. P. Pridmore, "Active vision and surface reconstruction for 3d plant shoot modelling," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 17, no. 6, pp. 1907–1917, 2019.

[37] L. Manuelli, Y. Li, P. Florence, and R. Tedrake, "Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning," *arXiv preprint arXiv:2009.05085*, 2020.

[38] A. Sanchez-Gonzalez, J. Godwin *et al.*, "Learning to simulate complex physics with graph networks," in *Proc. Int. Conf. Mach. Learn.* PMLR, 2020, pp. 8459–8468.

[39] J. Townsend, N. Koep, and S. Weichwald, "Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation," *J. Mach. Learn. Res.*, vol. 17, no. 137, p. 1–5, 2016.

[40] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[41] D. Seita, P. Florence, J. Tompson, E. Coumans, V. Sindhwani, K. Goldberg, and A. Zeng, "Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks," in *IEEE Int. Conf. on Robotics and Automation*. IEEE, 2021, pp. 4568–4575.

[42] H. Joo, H. Liu, L. Tan, L. Gui, B. Nabbe, I. Matthews, T. Kanade, S. Nobuhara, and Y. Sheikh, "Panoptic studio: A massively multiview system for social motion capture," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 3334–3342.

[43] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Icml*, vol. 99, 1999, pp. 278–287.

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

[45] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.