DeRi-Bot: Learning to Collaboratively Manipulate Rigid Objects via Deformable Objects

Zixing Wang* and Ahmed H. Qureshi

***Abstract*— Recent research efforts have yielded significant advancements in manipulating objects under homogeneous settings where the robot is required to either manipulate rigid or deformable (soft) objects. However, the manipulation under heterogeneous setups that involve both deformable and rigid objects remains an unexplored area of research. Such setups are common in various scenarios that involve the transportation of heavy objects via ropes, e.g., on factory floors, at disaster sites, and in forestry. To address this challenge, we introduce DeRi-Bot, the first framework that enables the collaborative manipulation of rigid objects with deformable objects. Our framework comprises an Action Prediction Network (APN) and a Configuration Prediction Network (CPN) to model the complex pattern and stochasticity of soft-rigid body systems. We demonstrate the effectiveness of DeRi-Bot in moving rigid objects to a target position with ropes connected to robotic arms. Furthermore, DeRi-Bot is a distributive method that can accommodate an arbitrary number of robots or human partners without reconfiguration or retraining. We evaluate our framework in both simulated and real-world environments and show that it achieves promising results with strong generalization across different types of objects and multi-agent settings, including human-robot collaboration.**

I. Introduction

The manipulation of heterogeneous soft-rigid systems, which involve connected deformable and rigid bodies, is crucial for various tasks. For instance, in forestry, ropes are often used to transport logs or large tree branches from one place to another, as illustrated in Fig. 1. Similarly, in disaster response scenarios, first responders may need to use ropes to move debris or heavy objects to search for victims or clear the path for rescue vehicles. Therefore, designing robots that can manipulate connected rigid and deformable objects in such scenarios can significantly enhance the efficiency and safety of operations.

Despite various applications of manipulating soft-rigid object systems, the existing works have primarily explored homogeneous settings where robots manipulate either rigid or deformable objects [1], [2]. For instance, recent research has demonstrated impressive capabilities in rigid body manipulation, such as novel objects grasping [3] and tool-based manipulation [4]–[6]. Some studies have also explored deformable object manipulation, with [7] using a manipulator to effectively unfold clothing and [8] learning to wave ropes to vault and knock objects.

In this paper, we propose the first **De**formable-**Ri**gid Ro**bot** (DeRi-Bot) framework that enables a rigid body manipulation with deformable objects. Specifically, moving a rigid object to designated positions by pulling ropes, as shown in Fig. 2. However, designing such framework needs to overcome the following challenges presented by soft-rigid systems. Firstly, deformable bodies possess significantly more degrees of freedom than rigid bodies, which makes explicit and numerical modeling of their movement considerably more complex and expensive. Secondly, the complexity of soft-rigid systems prohibits direct scaling of the connected rigid bodies' size and shape to the system control signal, leading to generalization problems. Additionally, the inherent deformability makes soft bodies' movement behavior highly unpredictable and introduces multi-solution problems, which can be roughly viewed as a stochastic system. Finally, when dragging rigid objects via ropes, multiple agents, such as humans or robots, often collaborate to move the object, as

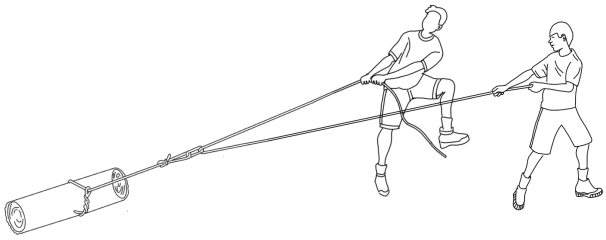

Fig. 1: A real-world scenario of moving a rigid object (tree trunk) via soft objects (ropes).

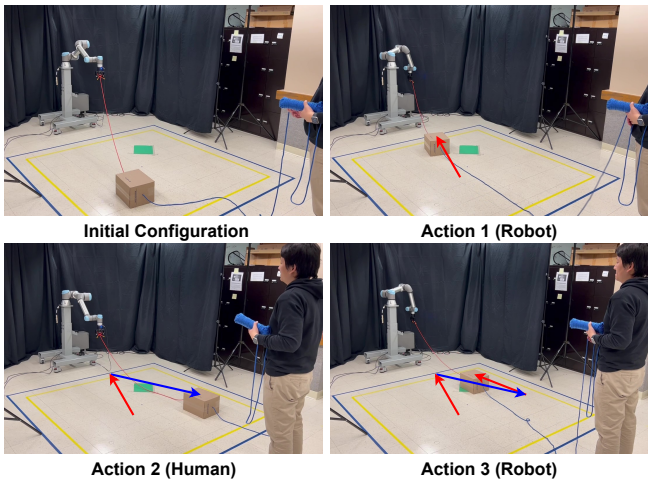

Fig. 2: A demonstration of DeRi-Bot collaborating with a human in the real world in moving the rigid brown box to its target position given by a green marker.

Zixing Wang and Ahmed H. Qureshi are with the Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA. Emails: {wang5389, ahqureshi}@purdue.edu
* indicates corresponding author

illustrated in Figs. 1-2. Therefore, the framework must be able to operate in a team with any number of agents to achieve efficient manipulation of soft-rigid systems.

These challenges necessitate the following components for the design of DeRi-Bot. (1) **Action Prediction Network** (APN). It predicts the required robot action and implicitly models the latent pattern between the soft-rigid body system and robot control commands for manipulating arbitrary-sized rigid bodies via ropes, thus tackling the first two challenges mentioned above. (2) **Configuration Prediction Network** (CPN). Because of the high stochasticity of soft bodies (third challenge), we intend to explore solutions in a wider range. Thus, we improvise by sampling commands from a Gaussian distribution centered around the output of APN. Given the sampled and APN generated action commands and the current environment state, CPN predicts the corresponding outcome configuration of the soft-rigid body, allowing the selection of the best action for execution that leads to the given goal state. (3) Lastly, to overcome the problem introduced by the need for a collaborative system, our framework decouples the synchronous task into an asynchronous independent task, i.e., only one agent acts at a time, and we leverage our CPN to select the agent whose action leads the rigid object closer to its target. This allows multiple agents to collaborate asynchronously to achieve the target. We integrate all the above-mentioned components into a unified framework, DeRi-Bot, and evaluate it in simulated and real-world environments. The results indicate that the proposed framework achieves all our objectives and generalizes to various system setups, including multiagent and human-robot collaboration settings, without needing any reconfiguration or retraining of the proposed framework.

To summarize, the main contributions of this work are as follows:

- We propose DeRi-Bot, the first framework that enables collaborative manipulation of rigid bodies using deformable bodies.
- We develop APN to implicitly model the underlying pattern governing the relationship between soft-rigid body systems and robot action commands.
- We design CPN to obtain visual foresight and overcome the challenge of rope stochasticity. This results in a robust framework capable of predicting the soft-rigid object configurations from different robot actions and selecting the best action for execution to reach the target position.
- A strategy to train the DeRi-Bot system asynchronously such that the proposed solution can scale to an arbitrary number of agents manipulating the rigid object via ropes.

II. Related Works

Since DeRi-Bot is the first work focusing on heterogeneous soft-rigid body manipulation, we present the existing works solving the homogeneous task of manipulating rigid or soft objects and draw their relevance and use cases to our proposed framework.

A. Rigid Body Manipulation

Plenty of work has been done in rigid body manipulation [4]–[6], [9]–[12], but none of these approaches consider the task of using soft objects such as ropes for manipulating rigid bodies. The use of rigid objects to manipulate other rigid objects has also been widely explored. For instance, [12] proposes a Task and Motion Planning (TAMP) technique for using rigid objects to interact with other rigid objects, such as using a hook to drag another rigid object closer to the robot. Another method introduces a Deep Affordance Foresight [5] that predicts the long-term effects of various actions, including tool use for moving other rigid objects. Likewise, [10] performs the poking action with a stick to move rigid objects and develop a visual foresight of the environment. Similar to these approaches, we also build a visual foresight module, but our approach considers a deformable object, i.e., rope, to manipulate a rigid object, which introduces the complexity of dealing with soft-object physics and its impact on the rigid objects' motion.

B. Soft Body Manipulation

Deformable object manipulation is a challenging problem due to its high degree of freedom. Plenty of work has been introduced ranging from imitation learning to reinforcement learning to model deformable object dynamics [13], [14]. For example, [13] optimizes learning from the demonstration for generalization to the novel deformable object manipulation task. [14] uses self-supervised and imitation learning to manipulate the rope to place it in the given target configuration. Other than that, demonstration-free methods such as [15] leverage model-free reinforcement learning to manipulate 1 to 2-dimensional deformable objects, and [8] use a self-supervised framework to plan for high-speed rope manipulation. The application of modeling deformable object physics has been demonstrated in various tasks such as knot-tying with a rope [16]–[18] and rope untangling [19]–[21]. Although the methods in modeling deformable object dynamics may be helpful in our approach to predict desired rope configurations for moving the attached rigid object, we follow a model-free approach that directly trains a robot action policy and implicitly learns the deformable object physics along with their relation to rigid-object and robot action dynamics. Perhaps a more relevant approach to our method is the iterative residual method [22]. Similar to CPN, it leverages the Delta Dynamics Network to predict the outcome rope tip trajectory after adjusting the previous whipping action. By iteratively applying the best adjustments to the last action, the rope tip eventually reaches the target position. Similar to their approach, our method also forecasts object states given different actions, but we evaluate actions supported by neural networks rather than random residual actions and consider a heterogeneous setup that combines both rigid and soft objects.

III. Methods

In this section, we present our methodology, DeRi-Bot, to achieve our objectives of designing a system that can work

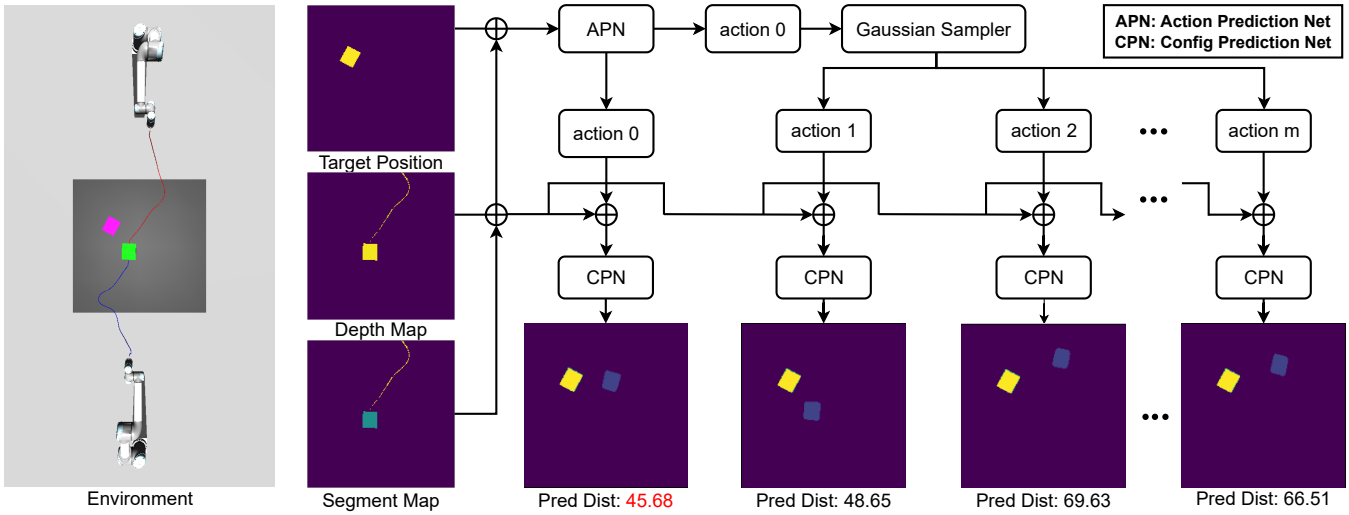

Fig. 3: An iteration of DeRi-Bot workflow. The APN model takes as input the target position map, depth map, and segmentation map to predict an action command. The Gaussian Sampler generates more actions around the output of APN. The CPN model takes as input the depth map, the segmentation map, and all the generated actions to predict the corresponding next states of the rigid object. The framework picks the best action that yields the minimum distance between the rigid object and its given target position. This figure shows such a process for the robot on the top. However, by design, all the robots' proposed actions will be compared together for the selection of the robot and its best action. Furthermore, note that we add the target to the visualization of CPN outputs to illustrate the effect of different actions. The raw outputs of CPN do not have the yellow target blocks.

with an arbitrary number of agents to collaborate in moving rigid objects to their goals via rope manipulation. To account for multiagent settings, we formalize a decoupling strategy that allows our framework trained on a single agent to generalize to multiagent scenarios asynchronously. Specifically, given the environment's current state, only one of the agents can execute its proposed action at a given time step. The agent for executing the action is selected heuristically or by a human's decision. The process is repeated iteratively, modifying the object's position to minimize the distance to the given target position. Due to decoupling strategy, DeRi-Bot possesses the following advantages:

- It can accommodate any number of robots of the same type without retraining or reconfiguration.
- It has the flexibility to work with human actors whose actions are hard to predict and estimate.

In the remainder of this section, we describe our problem statement and DeRi-Bot's components including it's neural models, workflow, and training strategy.

A. Problem Statement

We formulate the task with our proposed asynchronous action primitive as follows. An instance of the experimental environment contains one object, one target indicator, and an arbitrary number of agents connected to the object via ropes (one rope per agent). We use the following notation to represent their states. $d \in D$ denotes the orthographic depth map of the task environment, where $D \subset \mathbb{R}^{n \times n}$ and n indicates the map dimension. We regularize D by setting the ground plane as 0 meters. Similarly, $s \in S$ denotes the segmentation map of the environment, where $S \subset \{0,1,2\}^{n \times n}$, in which 0, 1, and 2 represent the ground plane, the rope, and the object. A variant of S that is without the encoding of rope is represented as $\hat{s} \in S$. We use a binary map $b \in B$ to encode the target position, where $B \subset \{0,1\}^{n \times n}$, and pixels belonging to the target are indicated by 1's. Both S and B are defined in the same domain as D. Let each agent be denoted by $c \in C$. Each agent's desired end-effector position is indicated by $a \in A$, where $A \subset \mathbb{R}^3$ represents the 3D workspace.

At each time step t, our system observes the environment state $(s_t,\ \hat{s}_t,\ d_t,\ b)$ and outputs a_t for the given agent c. The agent c moves to the given end-effector position, a_t. Thus, it pulls the rope, which moves the rigid object attached to it toward the target. Hence, DeRi-Bot aims to generate a series of actions $\{a_0,\ a_1,\ \ldots,\ a_T\}$, from $t = 0$ to $t = T$, for each agent such that they minimize the Euclidean distance, indicated as l, between the centroid of the object's current position and the given target position.

B. Action Prediction Network (APN)

The objective of APN for a given agent is to predict the next action that reduces the distance between the object's current and the target's position given the environment state. Specifically, at step t, the input for APN is a 3-channel 2D array (depth map d_t, segmentation map s_t and binary target map b):

$$a_t \leftarrow \text{APN}(d_t,\ s_t,\ b)$$

All three channels are spatially pixel-aligned, which has been proved to help the deep neural network interpret the spatial relation among channels encoding different information [23], [24]. Note that the b will not change over the runtime for a given target. Furthermore, these channels are rotated so that the corresponding robot always resides on the top side, which also applies to the inputs to our CPN network. The output of APN $a_t = (x,\ y,\ z)$ represents a position that the end-effector of the corresponding robot will move to, where x, y, and z is in the frame defined by the robot's operational space controller. The network is supervised by the Mean Square Error (MSE) between the network output and the ground truth command. The ground truth commands are generated by our random exploration strategy defined in Section III-E.

The model architecture of APN employs a Convolutional Neural Network (CNN) [25] to encode and interpret the 2D inputs. The intermediate output from the CNN is hereby processed by a Multi-Layer Perceptron (MLP) to output the desired 3D vector. In this work, the APN is backboned by a ResNet-style skip connected CNN [26]. The MLP module is a one-hidden layer fully connected network.

As we mentioned in Section I, DeRi-Bot improvises based on the output of APN to ameliorate the stochastic problems induced by the deformability of soft bodies to improve the chances of success in reaching the target. Therefore, we model by a multivariate Gaussian distribution $X \sim \mathcal{N}(\mu, \Sigma)$, where $\mu = a_t^0$ and $\Sigma = \sigma^2 * I$. We choose $\sigma = 0.5$ and generate various m extra action samples, $\{a_t^1,\ \cdots,\ a_t^m\}$, from that distribution which are then evaluated based on CPN predictions to select the best action a_t^* for execution. The a_t^0 action is set to be a_t generated originally by the APN.

C. Configuration Prediction Network (CPN)

CPN takes as input the depth map d_t, the segmentation map s_t and the action command $a_t^i \in \{a_t^0,\ a_t^1,\ \cdots,\ a_t^m\}$ to predict the outcome environment state segmentation map $\hat{s}_{t+1}^i$ after executing the action a_t^i:

$$\hat{s}_{t+1}^i \leftarrow \text{CPN}(d_t,\ s_t,\ a_t^i)$$

The third channel a_t^i is produced by broadcasting and zero-padding a_t^i to the same size as other channels. We employ the deeplabv3+ [27], an advanced image segmentation method, to serve as the network structure of CPN. The network is trained using the supervision of the Binary Cross Entropy (BCE) loss between the network output and the ground truth segmentation map without rope encoding.

Although DeRi-Bot can work without CPN by simply iteratively executing the output of APN of each robot, the system's performance can be improved by the synergy of APN and CPN. Furthermore, the advantage of CPN is twofold. First, it allows the selection of the best action a_t^* from a given set of actions, $\{a_t^0,\ a_t^1,\ \cdots,\ a_t^m\}$, for a given agent, i.e.,

$$a_t^* \leftarrow \arg\min_i\ l(\text{CPN}(d_t,\ s_t,\ a_t^i),\ b)\ \ \forall i \in [0,\ m]$$

Second, it also allows the selection of an agent during multiagent operations, i.e., the agent whose best action results in a minimum distance of the object to the goal among all agents.

D. Workflow

The workflow of DeRi-Bot is presented in Algorithm 1 and illustrated in Fig. 3. For a given task, at each step, APN proposes an action command for each of the K agents given d, s, and b. A Gaussian sampler is employed to sample user-defined m extra commands for each proposed action, yielding total $K \times (m+1)$ commands. Then, the generated commands will be sent to CPN to predict their corresponding outcome configurations. Given the outputs, we pick the action predicted to yield the shortest l and execute it with its corresponding robot. At the end of each step, the system can repeat the process or terminate if goal conditions are met, as described in Section IV-B. However, in the case of a human-robot collaboration setting, the human participant gets the preference to decide if they want to act or let the robot executes their action. Furthermore, humans can also terminate the iterative process if they consider goal conditions have been met or the maximum number of interactions steps are reached without achieving the goal.

Algorithm 1 DeRi-Bot Framework

```
Init(Object, Target)   ▷ Randomly place object and target
for t = 0 ... ∞ do
    for j = (1 ... k) do          ▷ For each of the k robots
        d_t, s_t, b ← Observe(c_j)          ▷ Get Env config
        a_t^0 ← APN(d_t, s_t, b)
        ŝ_{t+1}^0 ← CPN(d_t, s_t, a_t^0)
        l_0^j ← Dist(ŝ_{t+1}^0, b)
        for i = (1 ... m) do      ▷ Sample m extra actions
            a_t^i ← Sample(a_t^0, σ)
            ŝ_{t+1}^i ← CPN(d_t, s_t, a_t^i)      ▷ Predict outcome
            l_i^j ← Dist(ŝ_{t+1}^i, b)            ▷ Calculate offset
        end for
    end for
    î, ĵ ← argmin(l_0^1 ... l_m^k)      ▷ Find best action index
    Execute(a_t^î, c_ĵ)                 ▷ Execute the best action
    if Term. Cond. then break           ▷ Break condition check
    end if
end for
```

E. Data Generation

Our dataset was generated using a dual-bot setup, as illustrated in Fig. 4. In this setup, two robots take turns executing randomly sampled action commands. For each move, we randomly sample an a within the valid space, and the robot moves its end-effector from its home position (shown as the top-left of Fig. 2) to the sampled position, following the linear path between them. After each action, we collect depth maps before and after the movement and use color and depth information to produce corresponding segmentation maps. A dataset instance consists of the command, the depth, and the segmentation maps of the environment before and after

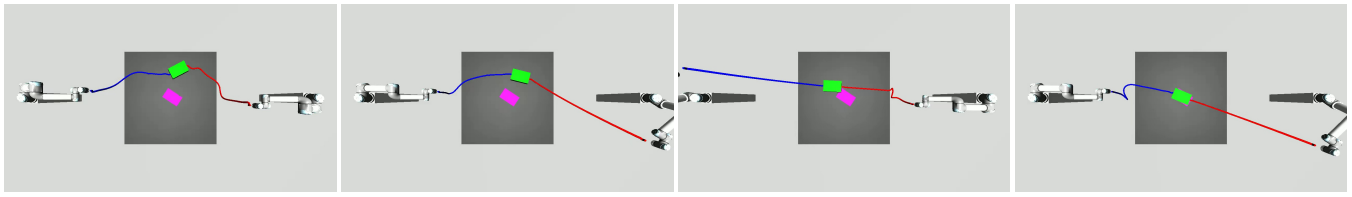

Fig. 4: An example of a dual-bot setup in moving the green box to its target position indicated by the purple marker.

TABLE I: Evaluation results on a dual-robot simulation setup

	DeRi-Bot (APN + CPN)	Random Sampling	Informed Sampling	APN without CPN
Shortest Offset	**0.306 ± 0.198**	0.393 ± 0.196	0.422 ± 0.272	0.317 ± 0.188
Final Offset	**0.315 ± 0.207**	0.654 ± 0.268	0.562 ± 0.290	0.422 ± 0.235
Step Cost	5.29 ± 1.75	4.53 ± 1.11	**4.32 ± 1.27**	4.70 ± 1.14

the action, i.e., d_t, s_t, d_{t+1}, s_{t+1}, and a_t. Furthermore, in each instance of interaction, the box size (length, width, and height) was randomized within the allowed range for sim2real generalization. We expect that a large scale of interactive data points can help our neural networks interpret the underlying pattern governing the relationship between the environment configurations and actions.

To train the APN, as indicated in Section III-B, the input data is d_t, s_t and b while the label is a_t. On the other hand, the CPN dataset instance has input data d_t, s_t and a_t with the ground truth of next state without rope information, i.e., $\hat{s}_{t+1}$. We collected 67,000 instances and divided them into training, validation, and testing splits with a ratio of 0.7 : 0.15 : 0.15.

IV. Experiments

We conducted a series of experiments in simulated environments for the dual-robot and the quad-robot setup and in the real world for the human-robot setup to evaluate the performance and generalization ability of our proposed DeRi-Bot framework. We comprehensively report the details of the setup and results of the experiments, along with the training and testing details of the individual neural models.

A. Training and Evaluation of Neural Models

We constructed and trained APN and CPN based on PyTorch-Lightning [28], respectively, with a batch size of 100 using Adam Weight Decay (AdamW) optimizer [29]. The training process cost 1.53 and 0.73 hours for APN and CPN with a Nvidia RTX 3090 GPU. The final MSE loss of APN on the training and the test set is 0.0201 and 0.0189 meters. For the CPN, the final BCE loss on the training and the test set is 0.0060 and 0.0061, respectively. This validates that our neural model design does not overfit the training data and generalize to novel scenarios of the testing dataset.

B. Workflow Evaluation

We build a 2 meters × 2 meters square arena using MuJoCo [30]. In this area, create two test settings. First, the dual-robot setting contains two UR5e robots residing on the left and right sides of the arena. Second, the quad-robot setting contains four UR5e robots placed on each side of the arena. The object to be manipulated is connected to each robot by a rope. We create 100 test scenarios in each dual and quad robot setting. Each scenario contains a different-sized rigid object with its randomly sampled start and goal configurations. In these scenarios, we evaluate various baselines along with the DeRi-Bot. The terminal condition in each scenario is defined in terms of the shortest distance of the rigid object to its target. If more than three times the offset of the rigid object to the target position is greater than the history's shortest distance, the experiment is terminated, and corresponding evaluation metrics are recorded.

C. Baselines

We evaluate our method along with other baselines to measure the relative performance of DeRi-Bot. The following methods are evaluated and presented in our sim-world experiments:

- **Random Sampling**: This method uniformly samples actions from the action space.
- **Informed Sampling**: This method samples actions from the action space based on the vector from the centroid of the object's current position to the centroid of the target position.
- **APN without CPN**: This method alternately executes commands generated by APN of each robot without sampling and CPN.
- **DeRi-Bot**: Our proposed framework incorporating the APN, sampling from a Gaussian Distribution, and CPN's visual foresight. We sample 3 extra actions from Gaussian Distribution around the output action from APN.

D. Metrics

The following metrics are employed to measure the performance of DeRi-Bot and the baselines quantitatively.

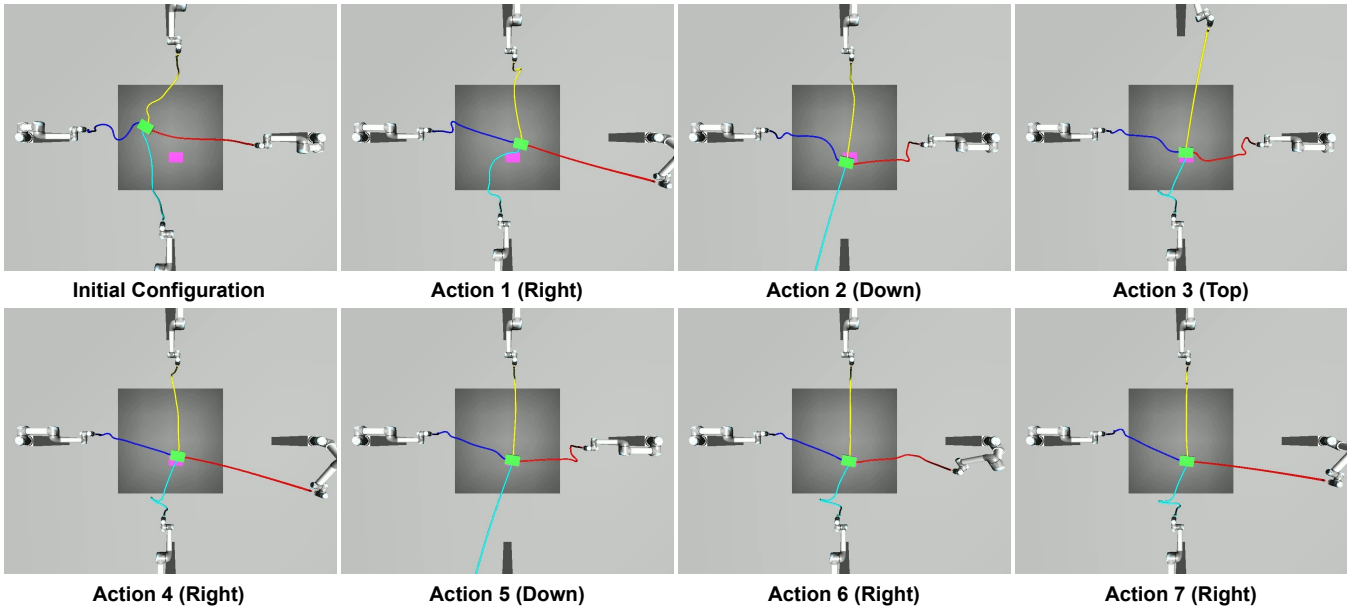

Fig. 5: An example of the quad-bot collaboration process in moving the green box to its purple target. All robots propose their actions, and the robot yielding the minimum distance to the target is selected based on the CPN visual foresight.

TABLE II: Generalization results of DeRi-Bot to quad-robot setup. It can be seen that compared to the dual-robot setting, the quad-robot setup yields better results as they have a higher degree of reachability to various directions over the arena.

	DeRi-Bot (4 Movable)	DeRi-Bot (2 Movable)
Shortest Offset	**0.117 ± 0.083**	0.330 ± 0.164
Final Offset	**0.127 ± 0.103**	0.347 ± 0.176
Step Cost	5.25 ± 1.53	**4.52 ± 1.51**

Furthermore, we compute the mean and standard deviation across all test scenarios of these metrics:

- **Shortest Offset**: This metric represents the shortest distance in the trajectory between the object position and the target configuration. The unit of this metric is also meters.
- **Final Offset**: This represents the distance between the centroid of the target configuration and the object position after termination. The unit of this metric is in meters.
- **Step Cost**: The total step cost, i.e., the number of actions executed in each experiment.

E. Sim-world Results and Analysis

The experimental results of the standard dual-robot setup evaluation are presented in Table I, while an example of the task's top-down view is illustrated in Fig. 4. As Table I indicates, DeRi-Bot performs best in the experiments. Regarding the Shortest Offset metric, the APN-related methods, DeRi-Bot and APN w/o CPN, exhibit significant advantages over other baselines, indicating the robustness and effectiveness of the APN network. Furthermore, the better performance of DeRi-Bot compared to APN without CPN indicates that our sampling module with CPN foresight helps further reduce the offset and obtain the best performance among all methods. In terms of the Final Offset, the trend is the same as the Shortest Offset. However, the advantage of DeRi-Bot over other baselines is even larger. As we use a threshold-based termination strategy, we can see DeRi-Bot's framework is able to generate more viable commands to maintain the progress achieved, which strongly supports the feasibility and efficiency of our proposed framework pipeline. Finally, in terms of step cost, though DeRi-Bot consumed the most step budget, given its advantages in the Final and Shortest Offset, we believe they are necessary and tolerable costs to accomplish the given task. In summary, our experiments demonstrate that DeRi-Bot is a robust and effective framework pipeline for the dual-robot setup task. The results validate the capacity of the APN network and the usefulness of the CPN-supported sampling module in manipulating soft-rigid object systems.

F. Sim-world Generalization Experiment

Our dataset and standard evaluation are conducted in the dual-robot setup as shown in Fig. 4. To evaluate the generalization ability of the DeRi-Bot, we built a quad-robot environment to manipulate ropes to move the attached rigid object. In these settings, we also report dual robot performance by disabling the two robots and keeping the robots on the left and right sides of the arena. According to the quantitative result, our conclusion is twofold. Firstly, DeRi-Bot can generalize to multiagent setups as expected without any retraining. This is due to our decoupling strategy and spatially aligning and rotating the network's input maps such that the robot is always residing on the top side of them

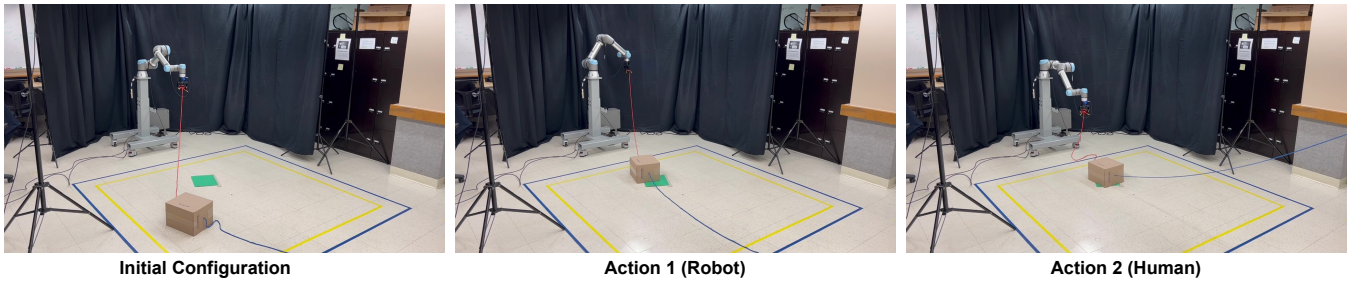

Fig. 6: An example of short-horizon real-world human-robot collaboration. It takes two steps to move the object to the target position.

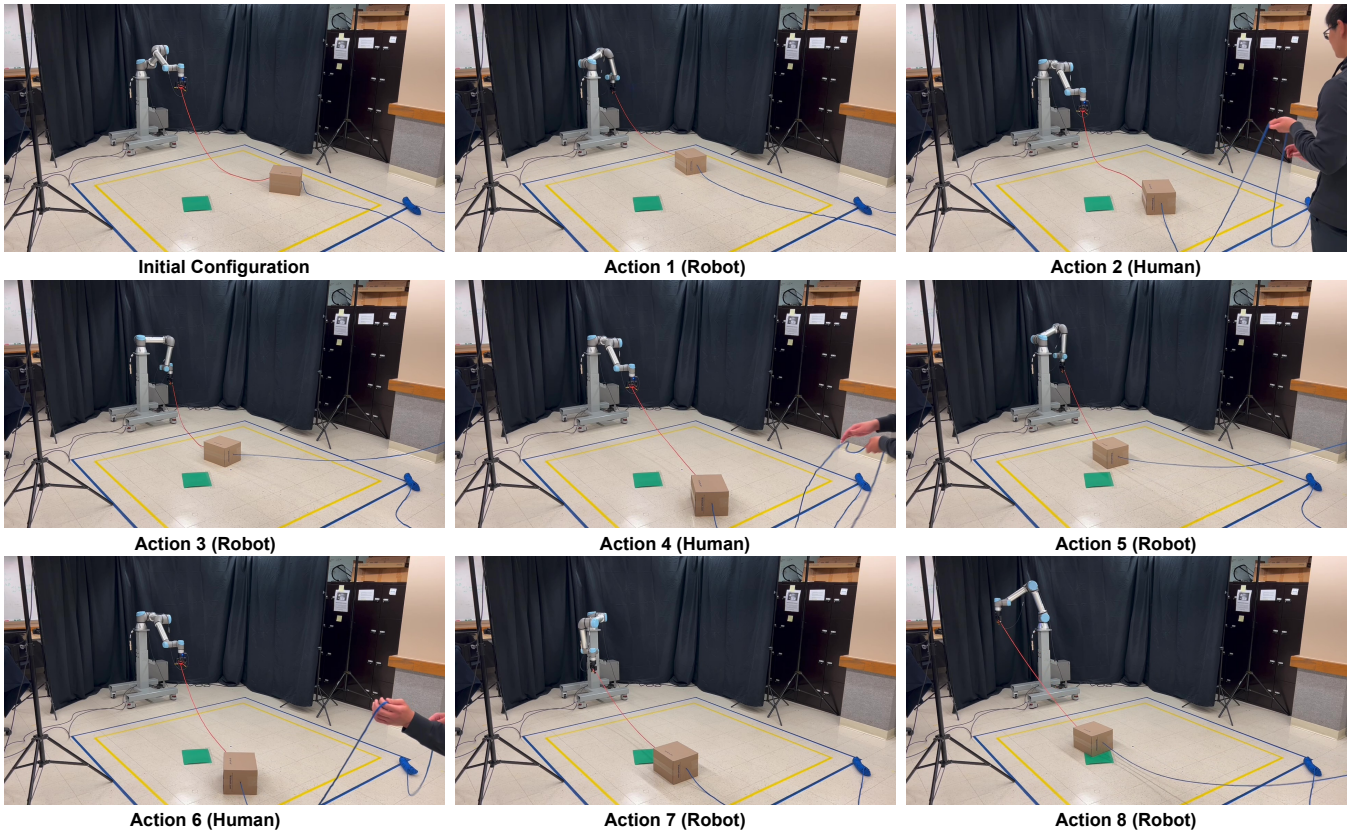

Fig. 7: An example of long-horizon real-world human-robot collaboration. It takes eight steps to move the object to the target position.

during prediction. Secondly, Table II indicates the results are much better than the performance in a dual-robot setup. It aligns the intuition as more robots at different positions provide more moving directions. For example, Fig. 5 shows vertical movements, which in the dual-robot case won't be possible, leading to more interaction steps to reach the target. It is also worth noting that in Fig. 5, action 6 does not affect the object's position, but action 7 moves it closer to the target. This is because the object's position after action 5 is almost perfect, so in action 6, the generated action yield no movement. However, in the next attempt, i.e., in action 7, CPN determined a sampled action could further minimize the offset, so the framework had the right robot to execute that command. This case strongly necessitates the CPN-supported sampling module to achieve the targets with minimum errors.

G. *Real-world Generalization Experiment*

To test our proposed framework's sim-to-real transfer and human-robot setup generalization ability, we conducted a following real-world experiment. We created three testing scenarios with different goal configurations and invited three volunteers to collaborate with our DeRi-Bot in moving the rigid object to given targets via ropes. Furthermore, the human and robot iteratively took steps to move the object. The human participant could do one of the following actions whenever the robot finished its action. Manipulate the rope

to move the object, skip and let the robot conduct the manipulation again, or terminate the experiment. It is worth noting that, unlike stationary robots, human operators can walk to the desired position and pull the rope, yielding larger action space.

The average final and shortest offset of all the experiments are 0.261 and 0.194 meters, and the average step cost is 6.67. We demonstrate two example trails in Fig. 2, 6 and 7 for short and long-horizon operations. We observed that our framework accomplished the task objective with human operators in most situations, as also evident from the results. This validates that our decoupling design of DeRi-Bot empowers it to collaborate with humans without any modification or neural model retraining. Furthermore, it also validates that our data generation strategy allows direct sim2real transfer.

V. Conclusions and Future work

In this work, we propose a DeRi-Bot framework to manipulate rigid objects via soft objects to transport them from one place to another. Our method comprises various neural models that, together with our algorithmic pipeline, achieve high performance in moving the objects to their target positions. Furthermore, our decoupling and training strategy allows the generalization of our framework to multi-agent settings and to real-world human-robot collaboration tasks. In future work, we aim to extend our framework to complex surfaces where the DeRi-Bot must account for resistive forces due to complex surface landscapes and ground frictional forces. Another possible avenue of research is to tackle the manipulation of unknown rigid objects with orientation constraints. This would also require the system to reason about complex rigid object geometry and the high degree of freedom rope dynamics.

References

[1] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, 2019.

[2] J. Sanchez, J. A. Corrales Ramon, B. C. BOUZGARROU, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: A survey," *The International Journal of Robotics Research*, vol. 37, pp. 688 – 716, 06 2018.

[3] A. K. Keshari, H. Ren, and A. H. Qureshi, "Cograsp: 6-dof grasp generation for human-robot collaboration," *arXiv preprint arXiv:2210.03173*, 2022.

[4] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese, "Keto: Learning keypoint representations for tool manipulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 7278–7285.

[5] D. Xu, A. Mandlekar, R. Martín-Martín, Y. Zhu, S. Savarese, and L. Fei-Fei, "Deep affordance foresight: Planning through what can be done in the future," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6206–6213.

[6] A. Xie, F. Ebert, S. Levine, and C. Finn, "Improvisation through physical understanding: Using novel objects as tools with visual foresight," *arXiv preprint arXiv:1904.05538*, 2019.

[7] H. Ha and S. Song, "Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding," in *Conference on Robotic Learning (CoRL)*, 2021.

[8] H. Zhang, J. Ichnowski, D. Seita, J. Wang, H. Huang, and K. Goldberg, "Robots of the lost arc: Self-supervised learning to dynamically manipulate fixed-endpoint cables," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4560–4567.

[9] Z. Zhang, Z. Jiao, W. Wang, Y. Zhu, S.-C. Zhu, and H. Liu, "Understanding physical effects for effective tool-use," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9469–9476, 2022.

[10] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," *Advances in neural information processing systems*, vol. 29, 2016.

[11] A. Stoytchev, "Behavior-grounded representation of tool affordances," *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 3060–3065, 2005.

[12] M. A. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum, "Differentiable physics and stable modes for tool-use and manipulation planning," 2018.

[13] A. X. Lee, S. H. Huang, D. Hadfield-Menell, E. Tzeng, and P. Abbeel, "Unifying scene registration and trajectory optimization for learning from demonstrations with application to manipulation of deformable objects," *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4402–4407, 2014.

[14] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine, "Combining self-supervised learning and imitation for vision-based rope manipulation," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2146–2153.

[15] Y. Wu, W. Yan, T. Kurutach, L. Pinto, and P. Abbeel, "Learning to manipulate deformable objects without demonstrations," 07 2020.

[16] J. E. Hopcroft, J. K. Kearney, and D. B. Krafft, "A case study of flexible object manipulation," *The International Journal of Robotics Research*, vol. 10, pp. 41 – 50, 1991.

[17] W. Wang and D. Balkcom, "Tying knot precisely," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3639–3646.

[18] T. Morita, J. Takamatsu, K. Ogawara, H. Kimura, and K. Ikeuchi, "Knot planning from observation," vol. 3, 10 2003, pp. 3887 – 3892 vol.3.

[19] W. H. Lui and A. Saxena, "Tangled: Learning to untangle ropes with rgb-d perception," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 837–844.

[20] J. Grannen, P. Sundaresan, B. Thananjeyan, J. Ichnowski, A. Balakrishna, M. Hwang, V. Viswanath, M. Laskey, J. Gonzalez, and K. Goldberg, "Untangling dense knots by learning task-relevant keypoints," in *Conference on Robot Learning*, 2020.

[21] Y. She, S. Wang, S. Dong, N. Sunil, A. Rodriguez, and E. Adelson, "Cable manipulation with a tactile-reactive gripper," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1385–1401, 2021.

[22] C. Chi, B. Burchfiel, E. Cousineau, S. Feng, and S. Song, "Iterative residual policy for goal-conditioned dynamic manipulation of deformable objects," in *Proceedings of Robotics: Science and Systems (RSS)*, 2022.

[23] Z. Wang and N. Papanikolopoulos, "Spatial action maps augmented with visit frequency maps for exploration tasks," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3175–3181.

[24] J. Wu, X. Sun, A. Zeng, S. Song, J. Lee, S. Rusinkiewicz, and T. Funkhouser, "Spatial action maps for mobile manipulation," in *Proceedings of Robotics: Science and Systems (RSS)*, 2020.

[25] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[27] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[28] W. Falcon et al., "Pytorch lightning," *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, vol. 3, 2019.

[29] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2017.

[30] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.