Discovering Synergies for Robot Manipulation with Multi-Task Reinforcement Learning

Zhanpeng He and Matei Ciocarlie

***Abstract*— Controlling robotic manipulators with high-dimensional action spaces for dexterous tasks is a challenging problem. Inspired by human manipulation, researchers have studied generating and using postural synergies for robot hands to accomplish manipulation tasks, leveraging the lower dimensional nature of synergistic action spaces. However, many of these works require pre-collected data from an existing controller in order to derive such a subspace by means of dimensionality reduction. In this paper, we present a framework that simultaneously discovers a synergy space and a multi-task policy that operates on this low-dimensional action space to accomplish diverse manipulation tasks. We demonstrate that our end-to-end method is able to perform multiple tasks using few synergies, and outperforms sequential methods that apply dimensionality reduction to independently collected data. We also show that deriving synergies using multiple tasks can lead to a subspace that enables robots to efficiently learn new manipulation tasks and interactions with new objects.**

I. INTRODUCTION

Recent advances show promise towards building competent robots hands that are able to achieve complex manipulation tasks. Many of these robots are designed to be versatile for general manipulation tasks and have numerous degrees of freedom. For example, the Shadow Hand can accomplish various in-hand manipulation tasks: finger pivoting, sliding, and gaiting a cube [1]. However, using robots with such high-dimensional control spaces remains a challenge in both model-based control and model-free methods. For model-based methods, such as model-predictive control, it is difficult to derive an accurate model due to the manipulators' high degrees of freedom. Even when an accurate model exists, model-based methods are computationally expensive, also due to high-dimensional state spaces. In the case of model-free methods, even when it is indeed possible to learn robust policies for manipulation tasks, the training process still requires very large amounts of robot experience to accommodate the large action space of highly dexterous hands. Finally, highly actuated hands are also difficult and expensive to manufacture, and can be fragile in use. From all of these perspectives, using robot hands with a very high dimensional control space remains a challenge.

In contrast, humans can achieve many manipulation tasks in a synergistic way. They select hand postures [2][3][4] that allow them to efficiently interact with objects from a small configuration space, even though human hands can reach a large number of postures. Inspired by human manipulation, roboticists have also explored the concept of motor synergies in the context of robotic manipulation, looking to find a low-dimensional subspace of postures that allow for efficient planning for grasping tasks. Such a subspace has generally been extracted by performing dimensionality reduction on ample amounts of task solution data collected via simulations or human demonstrations.

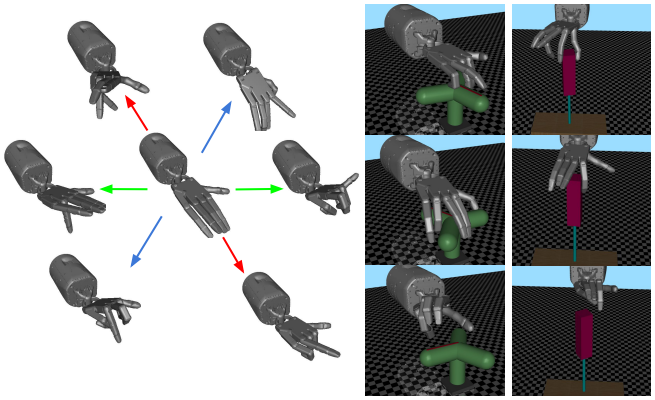


Fig. 1: Example of multi-task synergy discovery. The DiscoSyn algorithm can discover manipulation synergies such as the 3-dimensional linear synergy set shown in (a), while simultaneously deriving control policies that use these synergies for manipulation tasks such as the valve turning task shown in (b). Learned synergies also facilitate subsequent learning of new manipulation tasks, such as the top-down screwing task shown in (c).

However, collecting such data for dimensionality reduction can be slow and expensive since this process requires an expert with domain knowledge of the robotic task to generate solutions. Even when such data exist, the extracted synergies can be biased if the solutions they are learned on lack variety. Furthermore, although human hands inspire many designs of robot manipulators, they have different kinematic structures, and robots do not necessarily accomplish tasks in the same ways as humans. Therefore, synergies generated from human collected data might not be helpful for robot manipulation, and discovering synergies directly using robotic hands remains an essential problem.

While many works on synergies focus on grasping, finding such a subspace can be helpful in many others manipulation tasks, and such a subspace could potentially be shared across multiple tasks. For example, using a screw driver topdown and turning a cylindrical dial can be achieved using similar hand postures to grasp the target object. Learning synergies from a number of tasks that are diverse and exhibit complex dynamics could lead to a subspace that in turn contains diverse hand

Zhanpeng He is with the Department of Computer Science, Columbia University, New York, USA zhanpeng@cs.columbia.edu

Matei Ciocarlie is with the Department of Mechanical Engineering, Columbia University, New York, USA matei.ciocarlie@columbia.edu

postures. Such a subspace can potentially provide efficient exploration while interacting with new objects and hence allow for learning a control policy faster for an unseen task.

Finally, although linear synergies are well studied in the field of grasping, they can have limited applicability since linear models have limited capacity. We would like to discover linear synergies when feasible, but also to allow for the flexibility of learning non-linear synergies which are more capable of expressing diverse hand postures.

In this work, we present DiscoSyn, a framework that simultaneously discovers a synergy space and control policies that leverage this space for manipulation. Instead of extracting hand synergies from pre-collected control data, our method builds on multi-task reinforcement learning, which allow us to learn policies for multiple tasks that can share a synergy space. The results of our framework include both a policy that selects task-specific sequences of low-dimensional actions, and a synergy model shared across all tasks which projects these synergistic actions back to the original high-dimensional action space. We summarize our contributions as follows:

- To the best of our knowledge, we are the first to propose an algorithm that can discover synergies shared across multiple manipulation tasks and simultaneously learn policies that select actions from the synergy space.
- We demonstrate that our framework can learn both linear synergies, which have the advantage of being easily interpretable, and non-linear synergies, which provide a larger space that can be more suitable for learning diverse manipulation tasks.
- We show evidence that the learned synergies allow an agent to explore new environments efficiently and can be used to learn previously unseen tasks.

II. Related Work

Inspired by human-like manipulation, robotics researchers have attempted to leverage synergistic manipulation for a wide range of applications. Several works [4][5][6][7][8][9][10][11] applied dimensionality reduction on hand postures to study this behavior. These works follow the same sequential pipeline: their approaches first gather high-dimensional actions and extract synergies using by applying dimensionality reduction techniques, such as Principal Component Analysis (PCA), and show that a few principal components (PC's) can explain most of the variance. However, these only study linear synergies and most of them only work on grasping tasks. Our work, in contrast, focuses on control synergies of robotic hands that are useful for multiple dexterous manipulation tasks and do not place constraints of the form of synergies. Another key feature of our framework is that it does not rely on pre-collected data. Our agent find solutions for multiple tasks when discovering synergies.

To find solutions for robotic tasks automatically, our framework utilizes reinforcement learning, which has shown promise results in learning different manipulation tasks [12][13][14][15]. Nagabandi et al. [16] purpose a model-based method that learns a dynamics model of the environment and then multiple tasks such as a Baoding ball, weight pulling, and valve turning. OpenAI et al. [1] demonstrate the effectiveness of model-free methods, such as Proximal Policy Optimization (PPO)[17], on learning dexterous in-hand manipulation. However, all of these works learn policies on the full-dimensional action space. Our work leverages these advances of RL algorithms, but also allows an agent to discover a low-dimensional action space that can be used to learn new tasks efficiently.

To discover synergies across different tasks, we place our agents in multiple environments. Researchers have extensively studied learning multiple tasks simultaneously using RL and find that multi-task RL can leverage shared information across different tasks [18][19][20][21]. Hausman et al. [22] proposed a framework that learns continuous task representations and a task-dependent policy. Yu et al. [23] investigate the performance of various multi-task RL algorithms, for example, multi-task soft actor-critic, on several multi-task sets (Meta-world). Sodhani et al. [24] present a method that leverages languages as context to learn representations of a task to facilitate learning of multiple tasks. All of these works show promising results on learning multiple tasks considering extracting prior knowledge to learn unseen tasks. To achieve this, many previous works show the generalization ability of their task representation and try to find an appropriate representation for an unseen task. Our work, focused on robot hands, proposes a synergy model that contains hand configurations that allow for efficient exploration of unseen manipulation tasks and objects.

The closest work to ours is Group Factor Policy Search (GrouPS)[25], which integrates Group Factor Analysis (GFA) with RL. GrouPS designs the policy to be of a particular structure, which can be further interpreted as a linear synergy model, and directly applies policy search using this policy. While it shows promising results on extracting synergies from a two-arm grasping task, it can only use a linear policy in a single task setting. On the other hand, our framework provides the flexibility of choosing the structure of both the task-dependent policy and the synergy model and we are thus able to apply it to more complex manipulation tasks. While GrouPS focus on learning synergies on a single grasping task, our work emphasizes the importance of extracting synergies from multiple manipulation tasks that require more dexterous motor skills.

III. Approach

Our method aims to extract control synergies among diverse tasks which provide a compact subspace for agents to derive a policy efficiently. We employ a general definition of a synergy – a manifold that encodes a subset of high-dimensional control commands. A point $z \in \mathcal{R}^b$ in synergy space represents a low-dimensional action, which can then be projected to a point $a \in \mathcal{R}^d$ in the original high-dimensional action space using a synergy model $p(a|z)$. The parameters ϕ of this model effectively determine the synergy space. For example, if we choose to use a deterministic linear synergy model, then $\phi \in \mathcal{R}^{b \times d}$ encapsulates a matrix, and p represents the multiplication of z by ϕ. However, the general notation

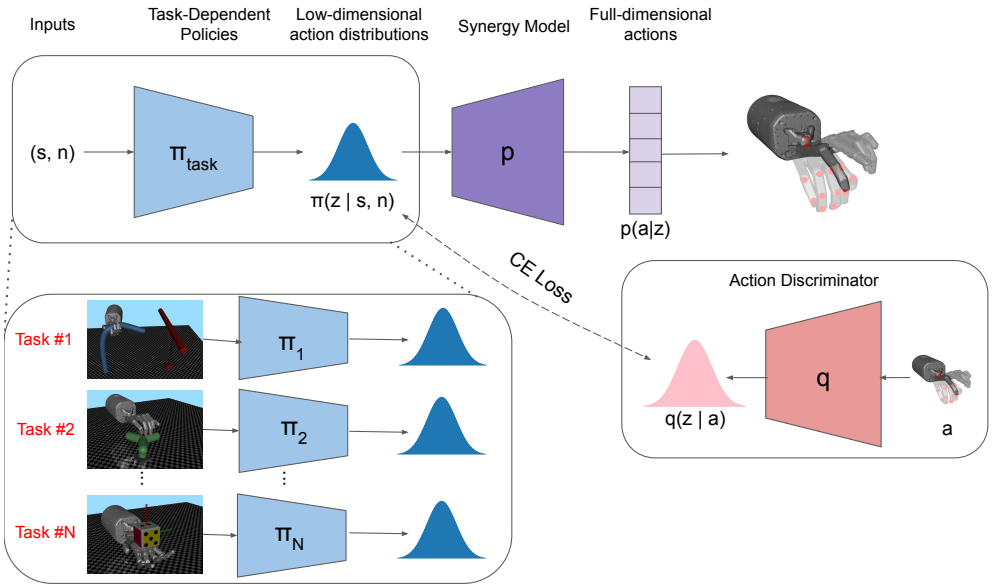


Fig. 2: Overview of DiscoSyn. Observations from each task are fed to the task-dependent policy head π_{task} to derive low dimensional actions z. Then, the synergy model p takes the z as inputs and infer the full dimensional actions a that can be directly applied to the environment.

introduced here allows for more general, potentially non-linear or non-deterministic synergy models. The first key goal of our approach is thus to learn the parameters ϕ that determine the behavior of a synergy model appropriate for a given set of manipulation tasks.

In addition to the parameters of the synergy model $p(a|z)$, the second key goal of our framework is to simultaneously learn task-conditioned policies that find solutions by generating sequences of low-dimensional actions z. Critically, we force our agent to discover shared synergies among different tasks by only using one synergy model to decode these low-dimensional solutions.

In summary, our agent, takes observations from a task, generates an low-dimensional action with a task-conditioned policy, this synergistic action is then projected back into a full-dimensional action using a synergy model shared across all tasks, and finally the full-dimensional action is applied to the environment.

A. Multi-task reinforcement Learning

To achieve this framework, we train our agent via multi-task RL in multiple Markov Decision Processes (MDP's). An MDP is identified by a tuple of $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$, where $\mathcal{S} \in \mathbb{R}^n$ represents the states space and $\mathcal{A} \in \mathbb{R}^m$ denotes the actions space, $\mathcal{R}$ is the reward function $\mathcal{R}(s, a)$ and $\mathcal{P}$ is the probability of transitioning to the next state s': $\mathcal{P}(s'|s, a)$. We formally define our problem of multi-task RL as follows. We assume that we have access to a set of tasks $\mathcal{N} = [1, 2, ..., N]$, which may vary in any aspects of a standard MDP. The goal of multi-task RL is to learn a task-conditioned policy $\pi(a|s, n)$ that maximize the average expected returns across all the tasks: $\mathbb{E}_{n\sim\mathcal{N}}[\mathbb{E}_{\pi}[\sum_{t=0}^{\infty} \gamma_t \mathcal{R}_n(a_t, s_t)]]$, where γ_t is the discounted factor at time step t.

B. Synergy learning with multi-task RL

To integrate synergy learning with multi-task RL, our policy comprises of a tasks-conditioned component π_{task}, whose outputs are low-dimensional actions z, and a synergy model p that takes low-dimensional actions and recovers the full dimensional actions a: $p(a|z)$. Hence, as shown in Fig. 2, an action can be computed from a state and the task identity by $\pi(a|s, n) = \pi_{task}(z|s, n)p(a|z)$.

Our goal is to discover a synergy space that is compact but still contains diverse postures that are efficient for learning different tasks. While learning hand synergies with multiple tasks can potentially lead to such a diverse space, we encourage our policy to learn diverse solutions in the full-dimensional action space of each task by employing maximum-entropy RL. Specifically, instead of optimizing for the general multitask expected return, we optimize for a maximum-entropy expected return: $\max_\pi \mathbb{E}_{\pi, n\in\mathcal{N}}\Big[\sum_{t=0}^{\infty} \gamma^i \big(r^n(s_t, a_t) \;+\; \alpha\mathbb{H}[\pi_\theta(a_t|s_t, n)]\big)\Big]$. This objective function does not provide intuitions about optimizing the synergy space. However, we can derive a lower bound[1] of the entropy term $\mathbb{H}[\pi_\theta(a_t|s_t, n)]$ by applying Jensen's Inequality:

$$
\begin{aligned}
\mathbb{H}[\pi_\theta(a_t|s_t, n)] &= \mathbb{E}_\pi[-\log \pi_\theta(a_t|s_t, n)] \\
&\geq \mathbb{E}_{\pi_\theta(a,z|s,n)}\Big[\log(\frac{q(z|a, s, n)}{\pi(a, z|s, n)})\Big] \\
&= -\mathbb{E}_{\pi_\theta(a,z|s,n)}\Big[-\log q(z|a_t)\Big] \\
&+ \mathbb{H}[\pi_\theta(z_t|s_t, n)] + \mathbb{E}_{\pi_\theta(z|s,n)}\Big[\mathbb{H}[p_\phi(a_t|z_t)]\Big]
\end{aligned}
\tag{1}
$$

Equation 1 gives us an intuition on how to optimize for such a lower action space. The first term suggests

[1]A detailed proof is included in https://roamlab.github.io/discosyn/.

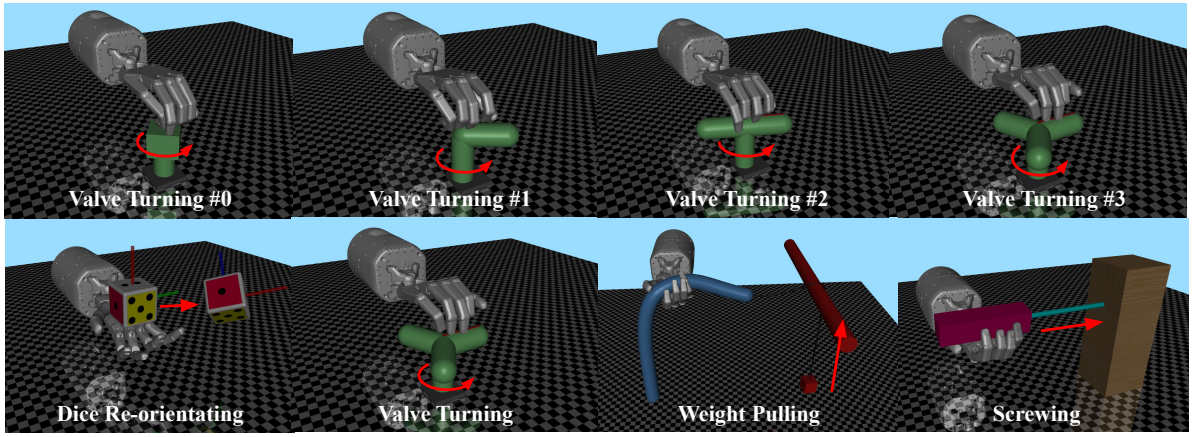


Fig. 3: Two task sets are used to test DiscoSyn. Task set #1: Turning valves with different number of handles. Task set #2: Four tasks that have different goals (represented as different reward functions) and different dynamics (represented as different transition functions).

that the low dimensional actions should be identifiable by the full-dimensional actions. The second term and third term encourage our agent to find diverse solutions for each task. Since the true distribution $q(z|a)$ is not tractable, we approximate it by learning a discriminator $q_\psi(z|a)$ from sampled data. Hence, in summary, we optimize our task-conditioned policies as well as the synergy model using an extended reward:

$$\mathcal{L}(\theta, \phi) = \max_{\pi} \mathbb{E}_{\substack{\pi(a,z|s,n) \\ t \in T}} \left[\sum_{i=0}^{\infty} \gamma^i \hat{r}(s_t, a_t, z_t, n) \right] \tag{2}$$

where

$$\begin{aligned} \hat{r}(s_t, a_t, z_t, n) = r^n(s_t, a_t) &+ \alpha_1 \mathbb{E}_{n \in \mathcal{N}} \left[\mathbb{H} \left(\pi_\theta(z_t|s_t, n) \right) \right] \\ &+ \alpha_2 \log q_\psi(z_t|a_t) \\ &+ \alpha_3 \mathbb{H} \left(p_\phi(a_t|z_t) \right) \end{aligned} \tag{3}$$

Here, α_1, α_2, α_3 are constant. Algorithm 1 shows the procedure of co-optimizing a task-dependent policy and a synergy model. In this work, we use PPO to optimize π and p.

Algorithm 1 DiscoSyn training

1: **while** returns have not converged **do**
2: Sample a batch of tasks $n \in \mathcal{N}$
3: Sample H trajectories using the current policy $\pi_{task}(z|s, n)p(a|z)$ for each sampled task
4: Optimize discriminator q using collected (z, a)
5: Optimize π_{task} and p with Eq. 2
6: **end while**

C. Implementation Details

Task-conditioned policy One naive implementation of the task-conditioned policy π_{task} can be a single model whose inputs combine the state s and task identity n. In this work, we empirically find that using a single model fails to learn all the tasks. This can be caused by conflicts in gradients since the tasks we include in our task set requires an agent to perform diverse motor skills. Hence, we employ a multi-head structure for π_{task}, including N models while training our agent with N tasks. When our agent encounters task n, we pick the nth model and feed the observation input to derive a low dimensional action z.

Synergy model The framework presented so far does not specify the concrete form of the synergy model $p(a|z)$. We have used it in this work to learn two forms of synergies: linear and non-linear.

Linear synergies produce a manifold that can be interpreted analytically and can be useful in robotic hand design since they provide intuitions for underactuation mechanisms. To learn linear synergies, we employ the dimensionality reduction view also used in other works and treat our synergy model as a reversed process of dimensionality reduction. We parameterize our linear synergy models with a matrix $\phi \in \mathcal{R}^{b \times d}$. In the case of using a deterministic synergy model, we can derive a full-dimensional action a by multiplying z by ϕ. In this work, during training, we use stochastic synergy models to encourage exploration. Thus, we treat the product of z and ϕ as parameters of the full-dimensional action distribution. Specifically, in our experiments, we use normal distributions and calculate the mean by $a_{mean} = z \cdot \phi$. Then, we model the standard deviation using a vector in the same shape as a_{mean}. During testing, we remove the stochasticity of our synergy model by only using a_{mean} as our action output a for stable hand behaviors.

On the other hand, although non-linear synergies are difficult to interpret, non-linear models have a larger capacity to store diverse hand postures, which leads to the possibility of being more versatile and applicable to more tasks. To learn non-linear synergies, we can use any non-linear models for $p(a|z)$. Here, we use multi-layer perceptrons (MLP) to learn non-linear synergies.

D. Learning unseen tasks using learned synergy model

DiscoSyn presents a method to learn a hand configuration space for robot manipulation tasks. This space is generally of much smaller dimensionality than the original full dimensional action space, and thus provides an opportunity to speed up learning of new tasks, not comprised in the original set that the synergy model was learned on. To learn an unseen

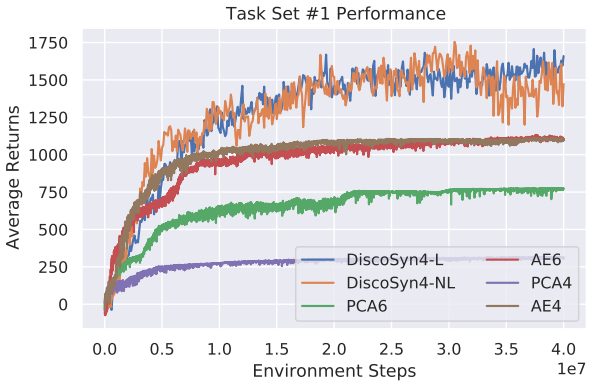


Fig. 4: Training curves for task set #1.

tasks n', where $n' \notin \mathcal{N}$, we take a learned synergy model $p(a|z)$, freeze the parameters of p, and directly optimize a task-dependent policy $\pi(z|s, n')$, whose outputs are lower-dimensional actions z.

IV. Experiments and Results

A. Synergies discovery and baselines

To test if our method can discover synergies for multiple tasks, we apply DiscoSyn on two training task sets. As shown in Fig 3, each task set contains four manipulation tasks and we will discuss the design considerations of them in IV-B. Although these tasks have different observation spaces, as mentioned in III-C, we use a multi-headed structure network for the task-dependent policy $\pi_{task}(z|s, n)$. Hence, the input dimension can be different for each head. We also test our method using different numbers of dimensions of the z space representing the number of synergies we can extract. We evaluate the policy learning performance, as well as the ability of extracting a low dimension representations of the high dimensional actions with the explained variance of the data.

To evaluate DiscoSyn, which simultaneously learns policies and synergies, we compare our method against a sequential baseline that produces synergies via dimensionality reduction on pre-collected solutions of each task. For this baseline, we first generate task solutions by training independent polices for each task, and collect trajectories using the learned RL agents. Then, we apply dimensionality reduction method to these data; we use PCA as a linear method to compare against the linear version of DiscoSyn, and a neural-network-based auto-encoder as a non-linear method to compare against the non-linear version of DiscoSyn. After applying the chosen dimensionality reduction method to extract synergies, we finally then attempt to train a policy that operates on the resulting low dimensional action space to learn each task.

B. Task sets

In this work, We apply our method to learn manipulation tasks using a 20-Dof simulated Shadow hand. We design the two task sets to be different across different characterstics of an MDP and to require the hand to interact with different objects. Here, we describe the two task sets[1] we use.

	Valve0	Valve1	Valve2	Valve3
PCA4	✗	✗	✗	✗
AE4	✓	✗	✓	✓
PCA6	✓	✗	✗	✓
AE6	✓	✗	✓	✓
DiscoSyn3-L	✓	✓	✓	✓
DiscoSyn3-NL	✓	✓	✓	✓
DiscoSyn4-L	✓	✓	✓	✓
DiscoSyn4-NL	✓	✓	✓	✓

TABLE I: Task set #1 results. Each row shows the results of a method and the name of the method represent: the underlying mechanism to extract synergies, the number of synergies and the type of synergy model (L for linear and NL for non-linear). For example, DiscoSyn3-L represent using DiscoSyn with 3 linear synergies.

	Dice	Valve	Weight Pulling	Screwing
PCA4	✗	✗	✗	✓
AE4	✓	✗	✗	✓
PCA6	✓	✗	✗	✓
AE6	✓	✗	✗	✓
DiscoSyn4-L	✓	✓	✗	✗
DiscoSyn4-NL	✓	✓	✓	✓
DiscoSyn6-L	✓	✓	✓	✓
DiscoSyn6-NL	✓	✓	✓	✓

TABLE II: Task set #2 results.

Task set #1: This task set contains four tasks that only vary on the transition model and have the same reward function. Specifically, we ask the hand to turn different types of valves (valve with different number of handles) counter-clockwise. **Task set #2**: This task set contains four tasks that vary on both the transition model and the reward function: dice reorienting, weight pulling, valve turning, and screwing. We design this task set to require the shadow hand to manipulate objects in different degrees of freedom. For example, the dice has 6 DoF while the screw only has a hinge and sliding joints.

C. Performance

As shown in table I, DiscoSyn can learn all the tasks in task set #1 using 3 synergies with both linear and non-linear synergies[2]. On the other hand, for the more challenging task set #2, our method learns all the tasks using 6 synergies but fails to tackle the whole task set using 4 linear synergies. With non-linear synergies, DiscoSyn can find a low-dimensional subspace that can allow for policy learning for all four tasks in task set #2. We attribute this to the larger capacity of synergy model can decode low-dimensional actions to more high dimensional actions.

As shown in Table I, the sequential baseline with linear (PCA) synergies only learns 2 out of 4 tasks from task set #1, even when using 6 synergies. On the other hand, the sequential baseline with non-linear (AE) synergies can perform 3 out of

[2] For more details about polices and synergy models, please find more training curves, detailed task descriptions and videos on: https://roamlab.github.io/discosyn/.

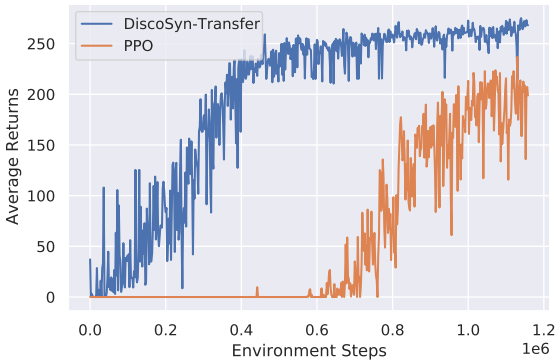


Fig. 5: Training curves of learning the sparse goal-condition valve turning with a pretrained synergy model (in blue) and in full dimension action space with standard RL (in orange).

4 tasks with 4 synergies. For taskset #2, the PCA baseline only learns 2 out of 4 tasks with 6 synergies and only learns one task with 4 synergies. In both task sets, the AE performance does not degrade with the decreased number of dimensions on the latent space since AE models generally have large capacity.

D. Learning unseen tasks with learned synergies

In this experiment, we test if the learned synergies can be leveraged for learning new tasks. We design new test tasks for both sets: 1. turning a valve cylinder; 2. clockwise (CW) valve turning; 3. top-down screwing. We use the cylinder valve to test if the synergy model can be used to interact with novel-shape objects, the CW valve turning task to see if the synergy model can provide rich learning signals for another task with a seen object, and the top-down screwing to test if the synergies help learning task with different dynamics. We find that with learned synergies, an agent can learn all of these tasks efficiently (representative runs of the learned policy can be found in the video accompanying the submission). We attribute this to the learned synergies providing rich interaction between the manipulator and the objects and hence rich reward signals that lead to fast learning of unseen tasks.

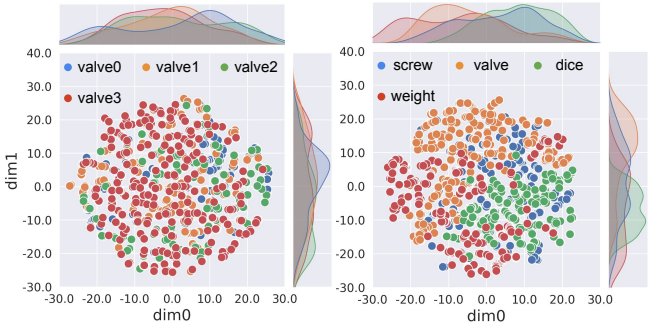
While our synergy models can be transferred to unseen tasks, we further test whether the synergies learned allows for efficient explorations. We design a sparse reward task that requires the hand to turn the valve to a specific goal joint position. The agent is only rewarded by 1 if the distance between the valve joint position and the target is smaller than a threshold σ; otherwise the reward is 0. As shown in Figure 5, an agent using learned synergies observes rewards within a few time steps and starts learning the task while a PPO agent operating on the full dimensional action space requires more than 40,000 steps to see the first reward signals.

V. Discussion

Can DiscoSyn discovers synergies while simultaneously learning the tasks? Our results show that we can extract 3, 4 and 6 synergies and our policies learn to accomplish all tasks in the experimental sets. Furthermore, given the structure of our framework, the lower-dimensional action space always explains 100% of the variance of the high-dimensional actions.

On the other hand, the applying PCA on independently learned policies only achieves explained variance of 62.8%, 50.8% and 41.5% using 6, 4 and 3 respectively for task set #1, and only slightly higher variance for task set #2. This implies that the actions learned from standard RL do not necessarily enforce a linear structure. When forced to learn a policy confined to the already defined lower dimensional space, the sequential baseline fails to solve all tasks.

Figure 1 presents some of the hand postures our framework learns and shows how our learned synergies are used in the training and test tasks. Our policy uses hand postures from the synergy space to interact with the object and also to re-pose the hand to transition to the next gaiting cycle.



(a) Task set #1 (b) Task set #2

Fig. 6: t-SNE visualization of low dimensional actions selected by our policy for task set #1 (a) and task set #2 (b).

How is DiscoSyn affected by the training tasks? Our results show that the diversity of a task set does influence the learning of the synergy model. This is expected: if a task set contains many different tasks, the capacity of z space needs to be larger. Our experimental results show that we cannot learn a linear synergy model with 4 synergies for task set #2 while DiscoSyn can use as few as 3 synergies for task set #1. We also visualize how each task utilizes the z space for both task sets. Since the tasks in task set #1 are similar, they share a large portion of the z space. On the other hand, in task set #2, which contains more diverse tasks, the task-dependent policies tend to occupy different parts of the z space. For example, the valve turning and dice re-orientation only overlap in a small space.

VI. Conclusions

Our results show that DiscoSyn is able to learn synergies that are effective for multiple manipulation tasks, while simultaneously learning policies that operate on this low dimensional action space in an end-to-end manner. We also demonstrate that the learned synergy model can be reused for unseen task and allows for efficient exploration during transfer. Compared to a classic pipeline that extracts synergies on existing control data, our method can use fewer synergies to solve multiple tasks. We note that, in its current stage, DiscoSyn is limited by requiring a pre-defined number of latent dimensions. In the future, we hope to equip DiscoSyn with the ability to discover the dimensionality of the smallest synergy space needed for a set of task, and also to apply its results to the design of underactuated robotic hands.

References

[1] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub W. Pachocki, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. "Learning Dexterous In-Hand Manipulation". In: *CoRR* abs/1808.00177 (2018). arXiv: 1808.00177. URL: http://arxiv.org/abs/1808.00177.

[2] J. Napier. "The prehensile movements of the human hand." In: *The Journal of bone and joint surgery. British volume* 38-B 4 (1956), pp. 902–13.

[3] Thea Iberall. "Human Prehension and Dexterous Robot Hands". In: *The International Journal of Robotics Research* 16.3 (1997), pp. 285–299. DOI: 10.1177/027836499701600302. eprint: https://doi.org/10.1177/027836499701600302. URL: https://doi.org/10.1177/027836499701600302.

[4] Marco Santello, Martha Flanders, and John Soechting. "Postural Hand Synergies for Tool Use". In: *The Journal of neuroscience : the official journal of the Society for Neuroscience* 18 (Dec. 1998), pp. 10105–15. DOI: 10.1523/JNEUROSCI.18-23-10105.1998.

[5] C.R. Mason, J.E. Gomez, and T.J. Ebner. "Hand Synergies During Reach-to-Grasp". In: *Journal of neurophysiology* 86 (Jan. 2002), pp. 2896–910. DOI: 10.1152/jn.2001.86.6.2896.

[6] E. Todorov and Z. Ghahramani. "Analysis of the synergies underlying complex hand manipulation". In: *The 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. Vol. 2. 2004, pp. 4637–4640. DOI: 10.1109/IEMBS.2004.1404285.

[7] Y. Liu, Li Jiang, Dapeng Yang, and Hong Liu. "Analysis of Hand and Wrist Postural Synergies in Tolerance Grasping of Various Objects". In: *PLoS ONE* 11 (2016).

[8] Vrajeshri Patel, P. Thukral, Martin K. Burns, I. Florescu, R. Chandramouli, and R. Vinjamuri. "Hand Grasping Synergies As Biometrics". In: *Frontiers in Bioengineering and Biotechnology* 5 (2017).

[9] Nestor Jarque-Bou, Alessandro Scano, Manfredo Atzori, and Henning Müller. "Kinematic synergies of hand grasps: a comprehensive study on a large publicly available dataset". In: *Journal of NeuroEngineering and Rehabilitation* 16 (May 2019). DOI: 10.1186/s12984-019-0536-6.

[10] Matei Ciocarlie and Peter Allen. "Hand Posture Subspaces for Dexterous Robotic Grasping". In: *I. J. Robotic Res.* 28 (June 2009), pp. 851–867. DOI: 10.1177/0278364909105606.

[11] Raul Suarez, Jan Rosell, and Nestor Garcia. "Using synergies in dual-arm manipulation tasks". In: vol. 2015. June 2015, pp. 5655–5661. DOI: 10.1109/ICRA.2015.7139991.

[12] Oliver Kroemer, Scott Niekum, and George Dimitri Konidaris. "A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms". In: *CoRR* abs/1907.03146 (2019). arXiv: 1907.03146. URL: http://arxiv.org/abs/1907.03146.

[13] Christian Daniel, Gerhard Neumann, Oliver Kroemer, and Jan Peters. "Hierarchical Relative Entropy Policy Search". In: *Journal of Machine Learning Research* 17.93 (2016), pp. 1–50. URL: http://jmlr.org/papers/v17/15-188.html.

[14] Jan Peters and Stefan Schaal. "Reinforcement learning of motor skills with policy gradients". In: *Neural Networks* 21 (June 2008), pp. 682–. DOI: 10.1016/j.neunet.2008.02.003.

[15] E. Theodorou, J. Buchli, and S. Schaal. "Reinforcement learning of motor skills in high dimensions: A path integral approach". In: *2010 IEEE International Conference on Robotics and Automation* (2010), pp. 2397–2403.

[16] Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. "Deep Dynamics Models for Learning Dexterous Manipulation". In: *CoRR* abs/1909.11652 (2019). arXiv: 1909.11652. URL: http://arxiv.org/abs/1909.11652.

[17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. "Proximal Policy Optimization Algorithms". In: *CoRR* abs/1707.06347 (2017). arXiv: 1707.06347. URL: http://arxiv.org/abs/1707.06347.

[18] F. Tanaka and M. Yamamura. "Multitask reinforcement learning on the distribution of MDPs". In: *Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation. Computational Intelligence in Robotics and Automation for the New Millennium (Cat. No.03EX694)*. Vol. 3. 2003, 1108–1113 vol.3. DOI: 10.1109/CIRA.2003.1222152.

[19] Daniele Calandriello, Alessandro Lazaric, and Marcello Restelli. "Sparse Multi-Task Reinforcement Learning". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/94c7bb58efc3b337800875b5d382a072-Paper.pdf.

[20] Diana Borsa, Thore Graepel, and John Shawe-Taylor. "Learning Shared Representations in Multi-task Reinforcement Learning". In: *CoRR* abs/1603.02041 (2016). arXiv: 1603.02041. URL: http://arxiv.org/abs/1603.02041.

[21] Carlo D'Eramo, Davide Tateo, Andrea Bonarini, Marcello Restelli, and Jan Peters. "Sharing Knowledge in Multi-Task Deep Reinforcement Learning". In: *Interna-*

tional Conference on Learning Representations. 2020. URL: https://openreview.net/forum?id=rkgpv2VFvr.

[22] Karol Hausman, Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, and Martin Riedmiller. "Learning an Embedding Space for Transferable Robot Skills". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=rk07ZXZRb.

[23] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. "Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning". In: *Conference on Robot Learning (CoRL)*. 2019. arXiv: 1910.10897 [cs.LG]. URL: https://arxiv.org/abs/1910.10897.

[24] Shagun Sodhani, Amy Zhang, and Joelle Pineau. "Multi-Task Reinforcement Learning with Context-based Representations". In: *CoRR* abs/2102.06177 (2021). arXiv: 2102.06177. URL: https://arxiv.org/abs/2102.06177.

[25] Kevin Sebastian Luck and Heni Ben Amor. "Extracting bimanual synergies with reinforcement learning". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 4805–4812. DOI: 10.1109/IROS.2017.8206356.