Generalizing to New Tasks via One-Shot Compositional Subgoals

Bian Xihan and Zhang Lianpin and Oscar Mendez and Simon Hadfield

***Abstract*— Generalizing to new tasks with little supervision is a challenge in machine learning and a requirement for future "General AI" agents. Reinforcement and imitation learning is used to adapt to new tasks, but this is difficult for complex tasks that require long-term planning. However, this can be challenging for complex tasks often requiring many timesteps or large numbers of subtasks. This leads to long episodes with long-horizon tasks which are difficult to learn.**

In this work, we attempt to address these issues by training an Imitation Learning agent using in-episode "near future" subgoals. These subgoals are re-calculated at each step using compositional arithmetic in a learned latent representation space. In addition to improving learning efficiency for standard long-term tasks, this approach also makes it possible to perform one-shot generalization to previously unseen tasks, given only a single reference trajectory for the task in a different environment. Our experiments show that the proposed approach consistently outperforms the previous state-of-the-art compositional Imitation Learning approach by 30%. While capable of learning from long episodes where the SOTA fails.

I. INTRODUCTION

As robotics becomes increasingly integrated into society, robots must be capable of performing complex tasks with greater automation and adaptability. These tasks often involve multiple implicit subgoals that vary depending on the environment. As such, it is common for only the target end goal to be specified explicitly. For example, if we ask the robot to bring us a cup of coffee, the robot will need to know where we are, as well as where the kitchen is, the tools, and the procedure for making coffee. The complex composite tasks are often long and difficult to learn, the effort of learning such tasks are enormous. More problematic is the fact that even if we provide explicit subgoal guidance: i.e. where our kitchen is, where the coffee machine is and how to use our coffee machine, this knowledge won't transfer to robots in other houses. Even for the individual robot the solution may be brittle, as simply moving the location of the coffee cups may cause the task to fail.

The biggest learning challenge for complex tasks is the complexity itself. Any complex task would almost always require a large number of steps to complete an episode. This is especially true for those with terminal-only sparse rewards, which result in a large state space and low sample efficiency. This is particularly true for tasks with terminal-only sparse rewards. The longer the average trajectory is, the broader we can expect an unbounded state-space to become, and the lower our sample efficiency will be. In an Imitation Learning setting, the use of expert trajectories helps alleviate the "vanishing reward" problem by providing feedback at each step of the trajectory. However, the exploration and data efficiency problems remain. The second challenge we seek to address is generalization. In an Imitation Learning setting, the data efficiency challenge mentioned above will often manifest as a relatively restricted set of expert trajectories. As such learning to perform a complex task often involves repetitively training on a small set of sample tasks. This can easily lead to over-fitting on the training task set or the specific training examples of the tasks. A common approach to mitigate this, is to design the model hierarchically as shown in figure 1. In this case each stage of the model is intended to specialize in solving a certain class of problems. This can simplify generalization within a subtask, but also exacerbates problems with data sparsity, as each submodel will only be exposed to a small portion of the training data.

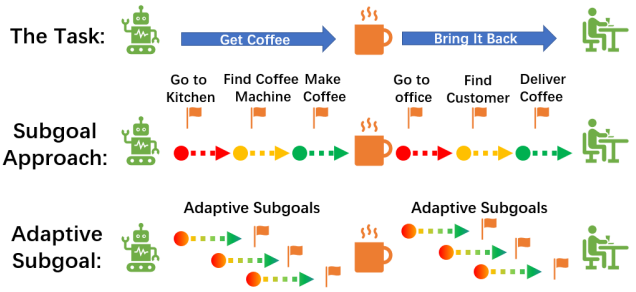

Fig. 1. In the "make coffee and bring it back" task, the traditional subgoal approach segments the complex task into smaller, manageable subtasks with clear endpoints and boundaries. In contrast, the proposed CASE approach generates novel compositional subgoals such as move forward and turn, which are generated at every step.

Our approach, Compositional Adaptive Subgoal Estimation (CASE), fully exploits the dataset to build a compositional task representation space and generate novel subgoals dynamically. We treat a single complex task episode as a sequence of smaller implicit tasks or subtasks with each still requiring multiple steps to complete. However, the need for long-term planning (and the brittleness of divergences) is alleviated. Importantly, unlike previous approaches, we do not explicitly define a finite set of subtasks with hard boundaries (i.e. "navigate to kitchen", "make coffee" etc.). Instead, the subtasks can be any small sub-trajectory towards the overall goal (e.g. "Move 3 meters forward", "turn 90 degree left" etc.) at a lower instruction level and are generated on-the-fly through compositionality rather than predefined by human intervention. These sub-goals do not need to correspond to any pre-defined task, nor do they need to have been previously observed during training.

Finally, an Imitation Learning policy is trained, using the learned compositional representation as it's state space, and with targets set via the adaptive subgoal estimation. This

approach enables an Imitation Learning policy to be trained using a learned compositional representation as its state space. Adaptive subgoals are generated on-the-fly via compositionality, providing additional flexibility, and allowing the agent to adapt to errors and avoid deadlock in unachievable subtasks. In addition, the learning task is simplified as long-term planning happens via the compositional space and the agent can focus on short-term execution which allows better performance in one-shot generalization over unseen tasks. With this approach, we are able to outperform imitation learning policies using the same compositional state space without the adaptive subgoal by over 30% in unseen task generalization.

In summary, the contributions of this paper are:

1) A novel approach estimate subgoal waypoints via a compositional task embedding space referred to as CASE
2) An Imitation Learning approach for complex compound tasks, based on online-subgoal estimation
3) An evaluation of one-shot task generalization for the policy, based on subgoal generalization

II. LITERATURE REVIEW

A. Imitation Learning

Imitation Learning (IL) approaches [8], [17], [21] utilizes a dataset of expert demonstrations to guide the learning process. Naturally, imitation learning relied heavily on the quality of expert demonstration. However, these approaches perform poorly with increasing episode length and have issues with transferring and generalization. [10], [23]

Gupta et al. [7] built upon HIRO [15] by generating subgoals with unstructured demonstrations to learn semantically meaningful behaviours. Similarly, IRIS [12] reproduce short demonstration sequences and generate sub-goals with a selection mechanism for low-level imitation. TACO [18] rely on task sketches and aligns the sub-task demonstrations into sequences to generalize to longer tasks.

Our approach ensures that all sub-goals are "in the right direction", which improves the model's ability to generalize to previously unseen tasks.

B. Compositional Model & Latent Plan Space

A compositional model is a model that encodes structural relationships. [2], [14] The work of CPV [6], creates compositional models that contains task plans and can be arithmetically operated upon. The work of Corey et al. [11], includes a repertoire of reusable behaviours learned and organized in an embedding space. Our work further enhanced the organizational capability and flexibility of the compositional model in the learned latent plan space.

III. METHODOLOGY

We will first clarify some terminology: A **task** is defined as a singular goal the agent must complete through a series of interactions with the environment. A **task sequence** is a collection of multiple tasks with no set order and may or may not depend on each other. Regardless of task dependencies, we allow the individual tasks within a sequence to be completed in any order. We further specify a **complex task** as a specified goal that involves the completion of a sequence of sub-tasks. In our framework, the **subgoal waypoint** is a state in the expert reference trajectory located in the "near future" of the current agent's state. Note that the current trajectory and reference trajectory are both solving the same task sequence, but are operating in different environments. Thus the subgoal waypoint cannot be used directly to guide the agent's trajectory.

We create a compositional latent space to represent both individual tasks and task sequences, where each unique task corresponds to a distinct point in the latent space. A task sequence also corresponds to a unique point in the latent space, which is the summation of the embeddings for each subtask within the sequence. This helps to draw a connection between "complex tasks" and "task sequences" as defined above. Both the singular complex task, and any (achievable) sequences of all its dependent subtasks, should map to the same point within the latent space. This compositional approach makes manifest the lack of ordering specified above. The summation of subtask embeddings is an commutative operation, therefore changing the order of the summation does not change the final embedding. In order to learn this compositional task embedding, constraints are codified as a number of regularization losses in addition to the concatenation of learned latent. We then train agents to use the learned task embedding as their state representation when selecting actions. This provides a compositional definition of both the current environment and the tasks to be completed. In an imitation learning framework, for each training trajectory $s_0 ... s_N$, an expert reference trajectory which completes the same task sequence in a different environment is provided.

A. Compositional representation

A compositional representation is an embedding which encodes structural relationships between the items in the space [14]. This compositional representation allows the agent to operate on an embedding of the tasks remaining to be done, without ever explicitly defining the target end-state. As such, the entire task embedding is a summation of subtask embeddings between various time steps. Further more, any task embedding between two states $(a, b, c...)$ is the summation of embeddings for all combinations of time steps in between. This leads us to define the constraint,

$$\vec{v}_{a:b} = \vec{v}_{a:c} + \vec{v}_{c:b} \qquad \forall c \quad \text{where} \quad a < c < b \qquad (1)$$

To prevent accidentally enforcing a specific ordering during the completion of these subtasks, the representation is built with commutativity, i.e. $\vec{v}_{a,b} + \vec{v}_{c,d} = \vec{v}_{c,d} + \vec{v}_{a,b}$ This is a very powerful representation for computing encodings of implicit groups of subtasks. However, in a complex task sequence, the $\vec{v}$ often embeds a long trajectory which consist of many tasks. This makes the learning process difficult, as information about far future tasks is a distraction from completing the current task.

B. Plan Arithmetic and Subgoal Waypoints

In one-shot imitation learning, the agent must perform a task (or sequence of tasks) conditioned on one reference example of the same task. In our work we further generalize this by allowing the current and reference task to be performed under different environments. The agent is trained with many sequences of other tasks in other environments and then provided with an expert trajectory as the reference to guide the new task, with no additional learning. Humans are adept at this: generalizing previous experiences to newly defined problems. However, for machine learning this is extremely challenging, and represents an important stepping stone towards general AI.

During training, the agent is given two trajectories, the training trajectory O and expert trajectory O^{ref} with matching task lists. It then learns a policy to perform online prediction of the actions in one trajectory, conditioned on the other trajectory as the reference. In the running example 'getting coffee', the agent will be provided with trajectories of retrieving coffee from a different office with a different floor layout. Learning how to make coffee without relying on specific meta-knowledge about a particular environment is vital for improving generalization. In imitation learning, the agent is provided with an expert trajectory, which performs the same sequence of tasks at an optimal level.

To be more specific, a visual approach to task specification is taken. During both training and testing, the agent is given an image of the desired goal state of the current episode (O_T), as well as the goal state of the reference episode ($O^{ref}_{0:T}$). It is also given an image of the current state (O_t), and an image of a future subgoal state from the reference trajectory (O^{ref}_I). It is important to emphasise that the agent is not provided with any future knowledge about the current trajectory, beyond the target goal state which is used to specify the task to be completed. Subgoals are drawn from the future of the reference trajectory, not the current trajectory ($O^{ref}_I \in O^{ref}_{0:T}$).

The model will first encode both the compositional representation of the current state to the goal state ($O_{t:T}$), and the compositional representation of the reference sub-goal to the goal state of the reference episode($O^{ref}_{I:T}$). It will then use the difference between the two ($O^{ref}_{I:T}$ - $O_{t:T}$) to predict the next action. Let $\vec{u}_{0,T} = g_\phi(O_{a:b})$ embed the observation pair at state a and b into the compositional representation with encoder g and parameters ϕ. We can estimate a subgoal state O^{ref}_I within the reference trajectory $\{O^{ref}_0, O^{ref}_T\}$, and create a compositional representation from this waypoint state to the goal state of the expert trajectory $\vec{v} = g_\phi(\{O^{ref}_I, O^{ref}_T\})$. Let $\vec{u} = g_\phi(\{O_t, O_T\})$ be the representation from the current state to the goal state, then we can calculate a waypoint representation $\vec{W}$ with the following subtraction in the latent domain:

$$\vec{W} = \vec{u} - \vec{v} = g_\phi(\{O_t, O_T\}) - g_\phi(\{O^{ref}_I, O^{ref}_T\}) \quad (2)$$

At timestep t, equation 2 estimates an approximation $\vec{W} = g_\phi(\{O_t, O^{ref}_I\})$ of the trajectory from the current state of the agent to the subgoal waypoint without having to explicitly know the waypoint along the current trajectory. This representation is then used as input for policy network $\pi\left(a_t|O_t, \vec{W}\right)$ to determine the actions of the agent.

To choose the subgoal waypoint, we assume the agent is always on the optimal path, therefore it's progress in the task is proportional to that the expert trajectory. As such when we choose the waypoint, we take the state p^{ref} in the reference trajectory, which has the same percentage of completion as in the training episode with episode length T: $\frac{p^{ref}}{T^{ref}} = \frac{t}{T}$, then add a fixed number k steps to ensure the waypoint is in the "near future" ($I = p^{ref} + k$). One potential issue with this approach is that the length of each subtask is unknown. If the current subtask in training episode is significantly longer or shorter than the expert trajectory, then the waypoint may fall into a different subtask. This will result in a misleading demonstration and potentially confuse the agent in the current task. However, this issue can be avoided with the length k of the subtask. As k increases in an episode, the chance of the subgoal state R_I landing in a different task in the reference sequence increases. The new task in the reference sequence is likely not an ideal demonstration for the current task in the training sequence. The optimal value of k varies depending on the tasks and the working environment as well as the subgoal system applied for learning. However, we expect the agent to be able to adapt to this situation, as any state from the following subtask will already reflect the completion of the current subtask.

Based on our new definition of the subgoal policy, the action loss becomes:

$$\begin{aligned} &L_a(O_t, O^{ref}_I) = \\ &\quad -log\Big(\pi\Big(\hat{a}_t|O_t, g_\phi(\{O_t, O_T\}) - g_\phi(\{O^{ref}_I, O^{ref}_T\}\Big)\Big) \end{aligned} \quad (3)$$

C. Policy and encoder learning

Additionally, there are two regularization losses using the triplet margin loss. The L_H enforces the compositionality of the latent space by ensuring that the sum of the embeddings for partial completion ($u_{0:t}$) and the embedded to-do vector ($u_{t:T}$) are equal to the embedding for the entire task ($u_{0:T}$).

$$L_H(O_0, O_t, O_T) = l_m(g_\phi(O_{0:t}) + g_\phi(O^{ref}_{t:T}) - g_\phi(O^{ref}_{0:T})) \quad (4)$$

where l_m is a truncated L1 loss with a margin equal to 1. The second regularization loss L_P tries to ensure that similarity in the latent space corresponds to semantically similar tasks. To this end, we ensure that the embedding of our agent's trajectory is similar to that of the embedding of the expert's reference trajectory

$$L_P(O_0, O_t, O_T) = l_m(g_\phi(O_{0:T}) - g_\phi(O^{ref}_{0:T})) \quad (5)$$

Thus the loss function for the framework is expressed as the weighted sum of the three losses: $L = L_a + \lambda_H L_H + \lambda_P L_P$.

IV. EXPERIMENTS

We evaluate performance on previously unseen combinations of tasks and randomly generated environments, especially on long sequences of tasks. A shared 4-layer CNN state encoder g_ϕ encodes the current state to goal state subtrajectory and sub-goal state to reference goal state subtrajectory. The resulting latent will be processed according to Eq.1, and fed into the policy network to estimate the action. In each experiment we contrast several variants of our own approach, including the effect of the current image branch and the additional compositionality losses. We also compare against the current state-of-the-art in compositional IL [6]. Additionally, we include an ablation study on the "near future" subgoal lookahead parameter k. In all other experiments we set $k = 4$. We also set the loss weightings $\lambda_H = \lambda_P = 1$.

A. Environment

We trained our agent on the Craft World environment [5], a 2D world with a top-down grid view and discrete actions. The agent can move in one of 4 directions at each step. The environment contains objects such as trees, rocks, axes, wheat, and bread, and the agent can interact with them via pick up and drop off actions. The object moves with the agent when it has been picked up, and can cause transformations to other objects in the environment. For example, if the agent carries an axe to a tree, the tree will be transformed into a log, which then can be transformed into a house once the agent picks up a hammer and brings it to the log. It is apparent that this environment, makes it possible to define complex long-horizon tasks such as "make bread" or "build house" which include many implicit subgoals. Furthermore, these tasks can be combined into sequences such as ["make bread", "eat bread", "build house"]. This eliminates the need for skill list labels [16] or language based skill description [19] which limits the generalisation to unseen tasks and sequences.

For training and testing, a random map is used to generate a number of tasks in sequence with no specific order, and an expert trajectory is generated through greedy search to ensure optimal solutions. To test one-shot generalization, the set of training tasks is different from the set of testing tasks, requiring generalization from the reference trajectory.

We also used this environment to emulate real world navigation problem, and demonstrate our agent on a live turtlebot3 [1] for indoor navigation. We use turtlebot to collect a small set of real world map, then process these maps into a format recognizable by the agent. Both the start point and goal are randomly generated on the map, the dataset size is 5000 episodes which is much smaller than the multi-task training dataset.

B. One-shot task generalization and ablation study

During the generalization test, we train the agent for 6000 epochs, and test the agent's ability to perform in a completely unseen environment with unseen tasks. We use the work of CPV [6] as our benchmark since it is the backbone framework used in our experiments. Table I shows the results of the generalization test. The CASE agent outperforms the original SOTA [6] benchmark by about 30% in unseen task success rate. In the target navigation tasks, shown in table II, the CASE agent still out-performs the enhanced CPV, as well as the SOTA [22], which uses a similar network backbone and is more capable in combinational generalization. The CASE approach is still able to outperform the modified SOTA consistently on these navigation tasks. In the live demo, the agent is given the robot's current location along with the target location marked on a pre-processed map. The agent will control the robot to navigate towards the target.

We also demonstrate the increased data efficiency and learning capability by evaluating the performance disparity between our model and backbone SOTA under varying size of both training and testing data, as well as episode lengths.(Episode length is measured by varies number of tasks per sequence contained in the episode.) As shown in Figure 2(right), When given less training data, a reduction in dataset size accentuates the performance divergence in unseen task testing scenarios. The SOTA model manifests an ascending learning curve during training, yet exhibits an erratic saturation at an early stage during testing. Conversely, the CASE model exhibits a consistent performance improvement in both training and testing throughout the experimental trials.

Figure 2(left) illustrates the performance variation with differing episode lengths during training. The inherent limitations of the backbone network prevent it from assimilating the given dataset adequately, whereas the CASE model effectively learns extended chains of task sequences. This pronounced distinction arises from CASE's adeptness in handling protracted episode learning. Subsequent experiments corroborate that, under the conditions of abbreviated episodes (2-4 task sequences per episode, considerably shorter than those employed in testing CASE), the backbone network necessitates a dataset size that is 3-5 times larger to approach CASE's performance.

Model	Best Performance	Average Performance	Standard Deviation
CPV-FULL [6]	0.432	0.392	0.0166
CASE	0.689	0.641	0.0133
CASE+CI	0.701	0.676	0.0139
CASE+CI+L	**0.712**	**0.687**	0.0167

TABLE I

An ablation study on the different components of the network: Current state image (CI) and assistive losses (L).

Finally, we tested several settings for the "near future" lookahead parameter k. When $k = 4$ the agent's performance is maximized, but the graph also indicates some sensitivity to the parameter k, with unstable performance at lower values. This may be due to the inconsistency in the length of the randomized subtasks between the training episodes and expert trajectories. This mismatch in step distance between the current state and subgoals may cause the generated subgoal to point towards steps before the completion of the current subtask in the expert trajectory for small k values

Model	Best Performance	Average Performance	Standard Deviation
CPV-FULL [6]	0.770	0.695	0.052
SKILL-IL [22]	0.790	0.714	0.045
CASE	**0.810**	**0.715**	0.048

TABLE II

Performance of target navigation tasks in seen environment but random starting and target point.

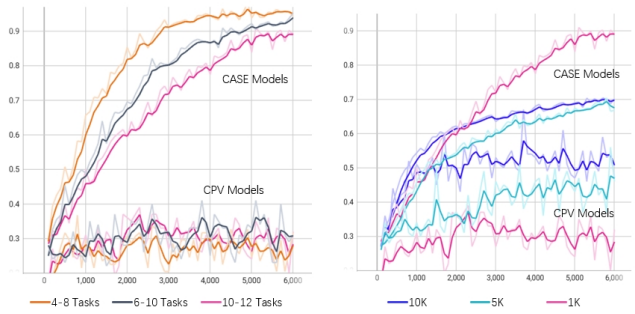

Fig. 2. The performance gap resulted under different length of episodes(left) and dataset size(right). The CASE technique enabled learning capability in small dataset and long episode where the backbone model would fail.

and the reverse for larger k values. In most cases the agent is able to deal with this: a subgoal for the following task is still easier to learn from than the entire remaining trajectory. Nevertheless, it may be interesting for future work to explore the automatic computation of the optimal k parameter during compositional subgoal estimation.

V. CONCLUSIONS

In this work, we proposed CASE, an approach to learn a compositional task representation which enabled novel subgoal estimation from reference trajectories in IL. This makes it significantly easier to learn long and complex sequences of tasks, including those with implicit or poorly defined subtasks. With this technique, we developed an IL agent which can generalize to previously unseen tasks with a success rate of around 70%. This represents an improvement of around 30% over the previous SOTA.

However, this approach can be developed further in future work. As discussed in section IV-B, using a fixed value for the k-step lookahead parameter may be suboptimal. Experiments indicate that performance and stability may be improved by developing an adaptive lookahead window, based on recent developments in the broader field of subgoal search. [4]

ACKNOWLEDGMENT

This work was partially supported by the UK Engineering and Physical Sciences Research Council (EPSRC) grant agreement EP/S035761/1 "Reflexive Robotics".

REFERENCES

[1] Robin Amsters and Peter Slaets. Turtlebot 3 as a robotics education platform. In *International Conference on Robotics in Education (RiE)*, pages 170–181. Springer, 2019.

[2] Xihan Bian. *Multi-Task Autonomy For Robotics*. PhD thesis, University of Surrey, 2023.

[3] Sonia Chernova and Manuela Veloso. Interactive policy learning through confidence-based autonomy. *Journal of Artificial Intelligence Research*, 34:1–25, 2009.

[4] Konrad Czechowski, Tomasz Odrzygóźdź, Marek Zbysiński, Michał Zawalski, Krzysztof Olejnik, Yuhuai Wu, Łukasz Kuciński, and Piotr Miłoś. Subgoal search for complex reasoning tasks. *Advances in Neural Information Processing Systems*, 34:624–638, 2021.

[5] Coline Devin. craftingworld. original-date: 2019-07-11T16:56:42Z.

[6] Coline M Devin, Daniel Geng, Pieter Abbeel, Trevor Darrell, and Sergey Levine. Compositional plan vectors. 2019.

[7] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.

[8] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[9] M Imtiaz, Yuansong Qiao, and Brian Lee. Comparison of two reinforcement learning algorithms for robotic pick and place with non-visual sensing. *Int. J. Mech. Eng. Robot. Res*, 10(10):526–535, 2021.

[10] Sergey Levine and Vladlen Koltun. Guided policy search. In *International conference on machine learning*, pages 1–9. PMLR, 2013.

[11] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. PMLR, 2020.

[12] Ajay Mandlekar, Fabio Ramos, Byron Boots, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Dieter Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4414–4420. IEEE, 2020.

[13] Tien Ngo Manh, Cuong Nguyen Manh, Dung Pham Tien, Manh Tran Van, Duyen Ha Thi Kim, and Duy Nguyen Duc. Autonomous navigation for omnidirectional robot based on deep reinforcement learning. *International Journal of Mechanical Engineering and Robotics Research*, 9(8):1134–1139, 2020.

[14] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[15] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

[16] Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. Zero-shot task generalization with multi-task deep reinforcement learning. In *International Conference on Machine Learning*, pages 2661–2670. PMLR, 2017.

[17] Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, 3(6):233–242, 1999.

[18] Kyriacos Shiarlis, Markus Wulfmeier, Sasha Salter, Shimon Whiteson, and Ingmar Posner. Taco: Learning task decomposition via temporal alignment for control. In *International Conference on Machine Learning*, pages 4654–4663. PMLR, 2018.

[19] Tianmin Shu, Caiming Xiong, and Richard Socher. Hierarchical and interpretable skill acquisition in multi-task reinforcement learning. *arXiv preprint arXiv:1712.07294*, 2017.

[20] David Silver, James Bagnell, and Anthony Stentz. High performance outdoor navigation from overhead data using imitation learning. *Robotics: Science and Systems IV, Zurich, Switzerland*, 1, 2008.

[21] Bian Xihan, Oscar Mendez, and Simon Hadfield. Robot in a china shop: Using reinforcement learning for location-specific navigation behaviour. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5959–5965. IEEE, 2021.

[22] Bian Xihan, Oscar Mendez, and Simon Hadfield. Skill-il: Disentangling skill and knowledge in multitask imitation learning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7060–7065. IEEE, 2022.

[23] Manizheh Zand, Krishna Kodur, and Maria Kyrarini. Automatic generation of robot actions for collaborative tasks from speech. In *2023 9th International Conference on Automation, Robotics and Applications (ICARA)*, pages 155–159. IEEE, 2023.