Copycat: Unsupervised Skill Acquisition for Imitation

Mrinal Verghese, Vignesh Rajmohan

***Abstract*— Skill acquisition is an important component of robot manipulation. In this work, we present a method for discovering and learning robot skills from unlabeled video demonstrations of human behaviour. By clustering demonstrations of behaviour in a latent space and identifying predicates in demonstrations, we are able to autonomously learn skills. We evaluate our method by tasking a robot to imitate human demonstrations of skills. Our method is able to successfully identify and learn 4 out of 6 skills present in demonstrations and correctly imitates the human demonstrator in 3 out of 6 demonstrations.**

I. Introduction

Manipulation skills are an important part of robot manipulation research. They enable a level of abstraction and allow the robot to reason and plan in a high level action space. In robot learning problems they have been shown to greatly reduce the dimensionality of the problem and significantly improve data efficiency [1]. However, generating skills can require significant engineering effort and maintaining a library of skills would require adding further skills as the robot encounters new tasks. As such, we seek to explore methods to discover and learn new skills for robots in an unsupervised manner. There exists large labeled datasets of human activity and object interaction [2] [3] and even more unlabeled videos of human activity on the internet. We would like to leverage these data sources to learn manipulation skills from the behaviour they contain.

A crucial component that enables this work is the work by Rothfuss and Ferreira *et. al* [4]. They explored learning an embedding network that produced a latent space where videos of tasks show strong intra-class similarity. They also briefly explored using this latent space to index past experiences and generate robot behaviour. We seek to continue their efforts by using their network to facility skill discovery and learning.

A. Contributions

In this work, we use the latent space from Rothfuss and Ferreira *et. al* to explore clustering demonstrations of skills and learning skills from those clusters. We collect our own dataset of skill demonstrations, embed them in the latent space, cluster them, and extract the primitives to execute the skill on a real robot. To test our discovered skills, we embed a new skill demonstration in the latent space, find its closest skill cluster, execute that skill on a real robot, and verify the robot executed the same behaviour as the demonstrator. We show that our system is capable of recognized four out of the six skills present in the demonstrations, and can successfully reconstruct them to copy the demonstrator.

II. Related Work

There is a broad range of prior work on learning skills or primitives from demonstrations. Dynamic Movement Primtives are a common technique for learning trajectories that can the be parameterized by goal loaction, trajectory duration, and other factors [5]. They have show broad success in robot control. Lioutikov *et. al* explored learning primitves from demonstrations using Probabilistic Movement Primitives, an extension of Dynamic Movement Primitives [6]. They segment behaviour in unlabeled demonstration trajectories and then use Probabilistic Movement Primitives to find commonalities in the segments. They show their results on a drawing task and a furniture assembly task. Konidaris *et. al* also explored symbolic skill discovery for long-horizon manipulation tasks [7]. Their work is capable of learning symbolic environment representations, as well as preconditions and skill transition models. We seek to further expand on these works by learning skills, their preconditions, and their outcomes from unlabeled video demonstrations.

III. Methods

A. Hardware Setup

For our experimental setup, we create two sets of objects, a pair of red and blue cubes for the human agent to interact with, and a pair of steel nuts, painted red and blue for the robot to manipulate. To assist in this manipulation, the DeltaZ robot has an electromagnet attached to its end-effector. The electromagnet is connected to a power supply via a MOSFET, that can turn it on and off based on a signal from the same Arduino microcontroller that controls the DeltaZ robot. Because of issues with residual current in the electromagnet, layers of duct tape are added to the top of the nuts and also painted, to increase their distance from the electromagnet and ensure they detatch when the magnet is turned off. Figure 1 shows the robot setup.

We employ an Intel Realsense D435 RGB-D camera for tracking objects. We use color segmentation based on the HSV color space to identify our objects in the scene and the KCF tracker [8] from the openCV vision library to track objects under partial occlusion. In order to accurately manipulate objects , we find the extrinsic matrix between the camera frame and the world frame (we assume the robot is at the world frame origin). We place a colored dot on the robot end-effector to track it, and then move the end effector to 30 randomly sampled points in the robot workspace. We record the position of the robot end-effector in the world frame, and the observed position of robot end-effector in the camera frame. Finally we use an optimzer to find the SE(3)

Fig. 1. The robot setup with the DeltaZ robot and attached electromagnet, Realsense camera, and colored blocks to be manipulated.

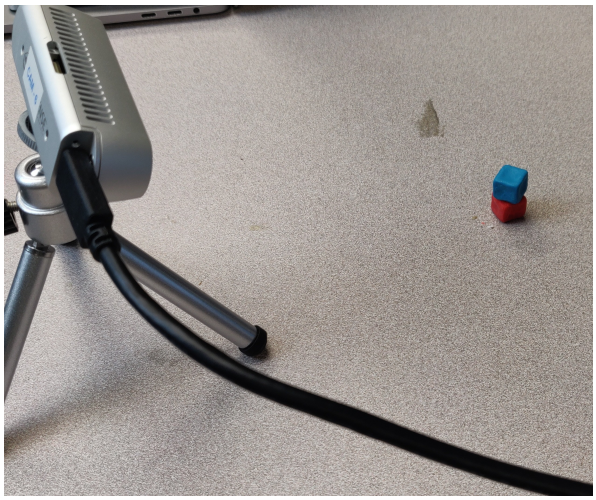

Fig. 2. View of the demonstration capturing setup. A human agent manipulations the red and blue blocks to show examples of skills.

transformation that minimized the difference between the robot-frame points and the transformed camera-frame points.

To manipulate the robot objects, we define two primitives a grasp primtive and a place primitive. The grasp primitive is paramterized by block to be grasped. The robot actuates to the block's coordinates, found via the camera and its extrinsics, and then activates the electromagnet to grasp the block. The place primitive is parameterized by a location in the robots workspace. The robot actuates to this location, and deactivates its electromagnet to drop the block. The combination of these two primitives allows us to move blocks around the robot workspace.

B. Data collection

We define three different skills each with two variants for our problem. We consider stacking skills, where an agent must stack one block on another, moving skills, where an agent must move both blocks to one side of the environment, and unstacking skills, where an agent must remove the top block from a stack of blocks. For the stacking and unstacking skills, the two skill variations depend on which block is on top of the stack, red or blue. For the moving skills, the skill variations depend on which side of the environment the blocks end up on, left or right. For each skill variation, we collection 20 demonstration videos of a human agent executing the skill using an RGB camera, for a total of 120 skill demonstrations. A view of our demonstration collection setup can be seen in figure 2.

C. Parsing predicates

In order to ultimately identify which primitives comprise each demonstration, we first parse the demonstrations by identifying which predicates are true at the beginning, end, and during the demonstration. For the beginning and end of the demonstration, we identify four predicates, *Clear(Red)*, *Clear(Blue)*, *On(Red,Blue)*, *On(Blue,Red)*, where *Clear(A)* defines if object A is graspable, and *On(A,B)* defines if object A is on top of object B. During the demonstration we also identify four predicates, *Moved(Red)*, *Moved(Blue)*, *Right(Red)*, *Right(Blue)*, where *Moved(A)* indicates object A moved during the demonstration, and *Right(A)* indicates if object A moved right (implying if object A moved, but did not move right, it must have moved left).

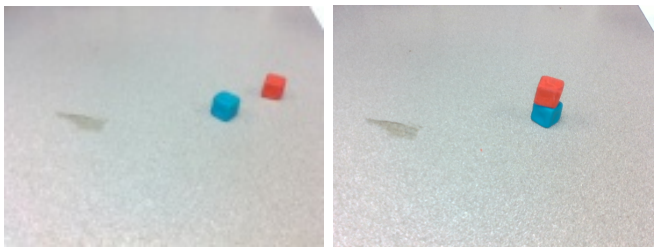

Fig. 3. The first and last frame of a stacking demonstration.

To identify if the *On(A,B)* predicate was true, we looked to see if object A was within a certain distance of object B and object A's y-coordinate was greater than object B's y-coordinate using pixel coordinates in the camera frame with the origin in the bottom left of the image. The *On(A,B)* predicate also informed the *Clear(B)* predicate, if the former was true, the latter must have been false. To identify if the *Moved(A)* predicate was true, we examined if the change in position between object A in the first frame of the demonstration, and the last frame of the demonstration was greater than a threshold. To identify the *Right(A)* we examined if object A moved, and the movement was in the positive x direction. Figure 3 shows the first and last frame from a stacking demonstration, and table I shows the predicates for this demonstration.

D. Network embedding

To embed our demonstration in a latent space, we draw on the work of Rothfuss and Ferreira *et. al* [4]. They train a network to embed video demonstrations of tasks in a latent space. Their network follows an encoder decoder

TABLE I

PREDICATE VALUES BEFORE AND AFTER THE DEMONSTRATION IN FIGURE 3

	Clear(R)	*Clear(B)*	*On(R,B)*	*On(B,R)*
Beg.	True	True	False	False
End	True	False	True	False

	Moved(R)	*Moved(B)*	*Right(R)*	On(B)
Dur.	True	False	False	False

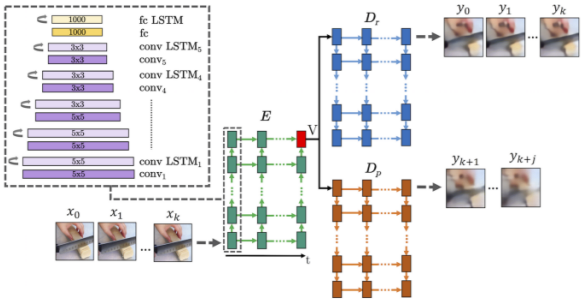

Fig. 4. The network architecture presented in [4]

structure. Given a task video of n frames, they define a visual experience as $X = X_r||X_p = x_1, \ldots, x_k||x_{k+1}, \ldots, x_n$, where X_r is the portion of video frames until frame k, and X_p is the remaining $n-k$ frames. Their network follows an encoder-decoder structure where the encoder E maps the input maps the input X_r to a latent vector V.

$$V = E(X_R) \tag{1}$$

This latent representation is forwarded to two decoders, the first, D_r tries to reconstruct the input frames X_r and the second D_p tries to predict the subsequent frames X_p. Formally this looks like

$$\begin{aligned} D_r(V) &= Y_r = y_1, \ldots, y_k \\ D_p(V) &= Y_p = y_{k+1}, \ldots, y_n \end{aligned} \tag{2}$$

where y_i are frames output by the decoders. The outputs Y_r, Y_p are compared to the ground truth X_r, X_p and the loss is computed as a weighted average of the image reconstruction loss, and the gradient difference loss between the input and output.

$$L = (1-\eta)L_{mse} + \eta L_{gdl} \tag{3}$$

.

The encoder is comprised of alternating ConvLSTM layers are regular convolution layers in order to capture both spatial and temporal relationships in the input videos. The Decoder is similarly structured but with deconvolution layers instead. Figure 4 shows their network structure

Their network is trained on the "20BN-something-something" dataset, a dataset approximately 100 thousand clips of human object interactions [3]. To embed our demonstrations in the network, we sample six frames from the demonstration, the first and last frame, and then 4 evenly distributed frames in between. We downsample these frames from 640×480 pixels to 133×100 pixels, feed them into the encoder, and save the 2000-dimensional latent space. We then use Principle Component Analysis across our datatset of 120 demonstrations to project the latent vectors to a 50-dimensional space. This 50-dimensional latent space representation of our skill demonstration dataset is the representation we use for clustering.

E. Demonstration clustering

The next step in our process is to cluster the unlabeled latent representations of our skill demonstrations into new skills. To accomplish this we use K-Means clustering with $K = 6$ as we expect to have six skill clusters. For each skill cluster, we take the predicate tables for all demonstrations in that cluster, and average their values, treating True as 1 and False as 0. We threshold the resulting predicate table at 0.5, considering cells with values above that to be True, and cells with values below that to be false.

F. Primitive selection

The last step in our process is to take a given predicate table, and use it to inform which combination of primitives will most closely match the predicates. There are three cases to consider based on the *Moved(A)* predicate, the case where no blocks move, which is trivial, the case where one block moves, and the case where both blocks move. If one block moves, say the red block, we first execute the grasping primitive to pickup that block. We then check the *Right(A)* predicate and the *On(A,B)* predicate. If the red block ends up on the blue block (*On(Red,Blue)* = True), we execute the place primitive parameterized by the blue blocks location, otherwise we parameterize the place primitive by a location on the left or right side of the workspace, depending on the value of the *Right(Red)* predicate. If two blocks move, we first check both the end *On(A,B)* predicates . If either is true, we move block B to a side of the workspace depending on the *Right(B)* predicate via a combination of grasp and place primitives. We then grasp the other block, block A, and place it on top of block B by excuting the place primitive parameterized by block B's location. If neither of the *On(A,B)* predicates are true, we grasp and place each block based on its *Right(A)* predicate.

IV. RESULTS

A. Network Embedding Results

We first examine the inter and intra-class similarity of skills demonstrations embedded in the latent space. We examin two metrics for similarity, cosine similarity and L2-distance. Figure 5 shows the similarity matrices for both of these methods.

We also explore classification of new demonstrations using a nearest neighbors approach in the latent space. The results of this experiment are show in the confusion matrix in figure 6. L2-distance showed slightly higher performance with 63% classification accuracy and that is the similarity metric we will be using going forward.

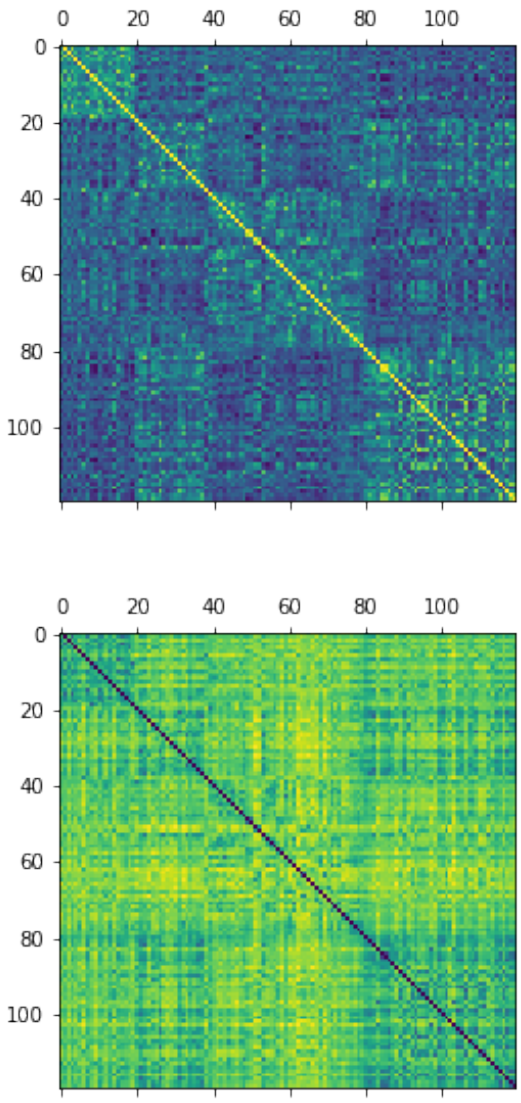

Fig. 5. Matrices showing similarities between demonstrations in the latent space using Cosine similarity and L2-distance respectively. Each 20 demonstrations is a different skill variation, in order they are: stack red-blue, stack blue-red, move left, move right, unstack red-blue, unstack blue-red.

Both these results show strong intra-class similarity for the skill variants: stack red-blue, stack blue-red, move left, and move right. However the unstack skills show little distinction between their variants. In fact without the unstack skills, the classification accuracy jumps to 85%.

B. Skill Reconstruction

We proceed to examine the correlation between the reconstructed skills and the original skills in the demonstrations by looking at the reconstructed predicate tables for the skill clusters. Table II shows both the original predicates for each of the skills, and the predicates for each of the six reconstructed skills as the top and bottom tables respectively. The predicates are arranged as the four beginning predicates, the four end predicates, and the four during predicates. Some reconstructed skills have clear correlation to original skills.

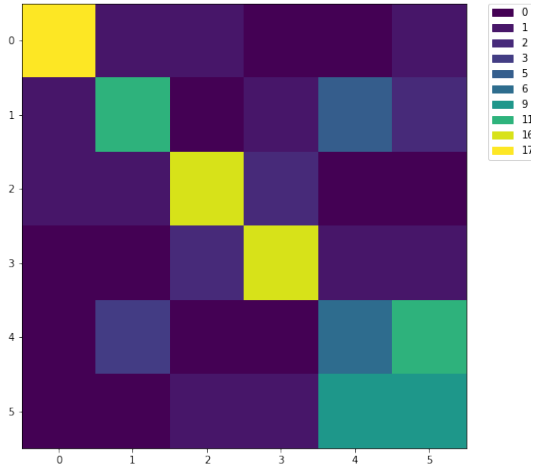

Fig. 6. Confusion matrix for a nearest neighbors classification in the latent space. The six skills in order are: stack red-blue, stack blue-red, move left, move right, unstack red-blue, unstack blue-red.

Skill 6 is stack red-blue, and skill 4 is stack blue-red. Move right is skill 2, but move left is represented in both skills 1 and 5. The lack of strong intra-similarity shown in the similarity matrix results would indicate the unstack skill demonstrations form poor clusters and that idea is reinforced here. None of the reconstructed skills correlate to the unstack skills. Skill 3, the closest reconstructed skill to unstacking, has the incorrect preconditions, but recognizes that only the blue block should be moved, and has the correct outcome.

C. Demonstration Imitation

Lastly, we test our method by evaluating its capacity to imitate a human demonstrator using its learned skills. We embed a new human demonstration in the latent space and compare it to our skill clusters to find the closest skill cluster. We then execute the primitives associated with the predicate table for that skill, and compare the resultant robot behaviour to the human demonstration.

The move demonstrations were able to successfully recover their skills. The new move left demonstration executed skill 5, one of the valid move left skills, and the new move right demonstration executed skill 2, the valid move right skill. The stack demonstrations were less successful. The stack red-blue demonstration correctly execute skill 6 the stack red-blue skill. However, the stack blue-red demonstration executed skill 3 which had no direct correlation to any original skill. This resulted in the robot correctly picking up the blue block, but placing it next to the red block, instead of on top. The results of the stack demonstration skill execution can be seen in figure 7. The unstack skills didn't have correlated learned skills, so they were unable to properly imitate the demonstration. The unstack demonstrations happened to be closest to a stack skill, and a move skill, but because their preconditions were not met, their execution did not lead to

TABLE II
SKILL PREDICATE TABLE

	Clear(R)	*Clear(B)*	*On(R,B)*	*On(B,R)*	*Clear(R)*	*Clear(B)*	*On(R,B)*	*On(B,R)*	*Moved(R)*	*Moved(B)*	*Right(R)*	Right(B)
Stk R-B	True	True	False	False	True	False	True	False	True	False	N/A	False
Stk B-R	True	True	False	False	False	True	False	True	False	True	False	N/A
Mv L	True	True	False	False	True	True	False	False	True	True	False	False
Mv R	True	True	False	False	True	True	False	False	True	True	True	True
Ustk R-B	True	False	True	False	True	True	False	False	True	False	N/A	False
Ustk B-R	False	True	False	True	True	True	False	False	False	True	False	N/A

	Clear(R)	*Clear(B)*	*On(R,B)*	*On(B,R)*	*Clear(R)*	*Clear(B)*	*On(R,B)*	*On(B,R)*	*Moved(R)*	*Moved(B)*	*Right(R)*	Right(B)
Skill 1	True	True	False	False	True	True	False	False	True	True	False	False
Skill 2	True	True	False	False	True	True	False	False	True	True	True	True
Skill 3	True	True	False	False	True	True	False	False	False	True	False	False
Skill 4	True	True	False	False	False	True	False	True	False	True	False	True
Skill 5	True	True	False	False	True	True	False	False	True	True	False	False
Skill 6	True	True	False	False	True	False	True	False	True	False	False	False

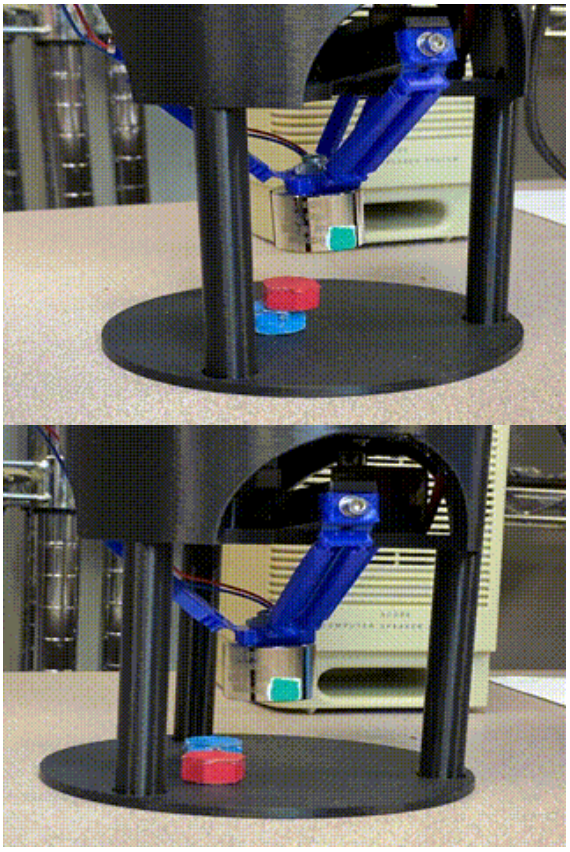

Fig. 7. The result of executing the closest learned skill to a stack red-blue demonstration, and a stack blue-red demonstration respectively. The former executed the correct learned skill, but the latter misidentifies the skill resulting in incorrect imitation.

meaningful manipulation. Refer to the companion video for more detailed imitation results.

V. CONCLUSION

In this work we explored learning skills, and their preconditions and outcomes from unlabeled video demonstrations. We evaluated our method by tasking a robot to copy a human demonstrator using skills it learned in an unsupervised manner. We were able to reconstruct 4 out of the 6 skills present in the demonstrations, and when testing imitation the robot correctly imitated the demonstrator in 3 out 6 demonstrations. While this work does have limitations, we believe these are exciting preliminary results for this line of investigation.

A. Limitations and Future work

Certain elements of this work were tailored to the skills we tested and lack generality. The method for parsing predicates from demonstrations, and learning preconditions and outcomes was based heavily on prior knowledge of the range of skills. Future work should explore more general ways of identifying relevant predicates at various points in demonstrations, possibly leveraging methods like those found in [7] or [9]. Future work could also consider more complex models for indexing behaviour in the latent space, and other methods of clustering embedded demonstrations. Lastly, due to computational constraints, we were only able to experiment with a small dataset. It would be interesting to see this work extended to the full ActivityNet or 20BN dataset to identify skills present in those demonstrations. We are excited to see how this exploration could lead to new methods for robot skill aquistion, and ultimately smarter and safer robots.

ACKNOWLEDGMENT

This work would not have been possible without the code provided by Jonathan Rothfuss and Fabio Ferreira. We also appreciate the assistance of Oliver Kroemer and his insights into this work.

REFERENCES

[1] Dalal, Murtaza Pathak, Deepak Salakhutdinov, Ruslan. (2021). Accelerating Robotic Reinforcement Learning via Parameterized Action Primitives. Thirty-Fifth Conference on Neural Information Processing Systems.

[2] F. C. Heilbron, V. Escorcia, B. Ghanem and J. C. Niebles, "ActivityNet: A large-scale video benchmark for human activity understanding," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 961-970, doi: 10.1109/CVPR.2015.7298698.

[3] Goyal, Raghav Kahou, Samira Ebrahimi Michalski, Vincent Materzynska, Joanna Westphal, Susanne Kim, Heuna Haenel, Valentin Fruend, Ingo Yianilos, Peter Mueller-Freitag, Moritz Hoppe, Florian Thurau, Christian Bax, Ingo Memisevic, Roland. (2017). The "Something Something" Video Database for Learning and Evaluating Visual Common Sense. 5843-5851. 10.1109/ICCV.2017.622.

[4] Rothfuss, J., Ferreira, F., Aksoy, E.E., Zhou, Y., Asfour, T. (2018). Deep Episodic Memory: Encoding, Recalling, and Predicting Episodic Experiences for Robot Action Execution. IEEE Robotics and Automation Letters, 3, 4007-4014.

[5] Schaal, S. (2006). Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In Adaptive motion of animals and machines (pp. 261-280). Springer, Tokyo.

[6] Lioutikov, R., Neumann, G., Maeda, G., Peters, J. (2017). Learning movement primitive libraries through probabilistic segmentation. The International Journal of Robotics Research, 36(8), 879–894.

[7] Konidaris, G.D., Kaelbling, L.P., Lozano-Perez, T. (2018). From Skills to Symbols: Learning Symbolic Representations for Abstract High-Level Planning. J. Artif. Intell. Res., 61, 215-289.

[8] Y. Yang and G. Bilodeau, "Multiple Object Tracking with Kernelized Correlation Filters in Urban Mixed Traffic," 2017 14th Conference on Computer and Robot Vision (CRV), 2017, pp. 209-216, doi: 10.1109/CRV.2017.18.

[9] Huang, De-An Xu, Danfei Zhu, Yuke Garg, Animesh Savarese, Silvio Fei-Fei, Li Niebles, Juan Carlos. (2019). Continuous Relaxation of Symbolic Planner for One-Shot Imitation Learning.