2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)
17-20 October, 2021. Melbourne, Australia

# Modeling Object's Affordances via Reward Functions

Renan Lima Baima[1] and Esther Luna Colombini[2]

2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC) | 978-1-6654-4207-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/SMC52423.2021.9658915

***Abstract*— Object affordance learning is the ability to process information about objects and how to use them. Embedding this knowledge in robots is an essential step for the development of intelligent and truly autonomous agents. This work proposes the development of a framework for learning affordances for robotic manipulation. The proposed approach was implemented as a reinforcement function in a network with a Soft Actor-Critic (SAC) algorithm and trained in simulation with a humanoid robot. Among different affordance complexities (touching and grabbing the object), the results show a rate of up to 95% correctness in the best scenario, with the agent properly performing all desired actions. Such results suggest that it is possible to define reward functions representing the object's affordances for different objects.**

## I. INTRODUCTION

For a robot to achieve its goals autonomously, it has to detect and manipulate objects accordingly. Although some metadata related to the properties of objects or artifacts are already available in databases, their magnitude, meaning, or how a robot can use or infer such information is not. The complexity in this process relies on how to embed in intelligent agents the meaning intrinsic to the perception of properties and actions that agents can execute to each object, such as: "Pick up," "Touch," "Reach," "Hold," so on.

Such knowledge, called object's affordance, has an abstracted meaning to humans, and each individual performs actions in a personalized way, using the available senses and actuators appropriately. According to Gibson [1], an object's affordances are the "properties of an object that determine what kind of actions an actor can perform on it." Affordance in robotics restricts the action space's scope once the robot will favor specific actions to be performed with specific objects [2]. Therefore, the problem revolves around how a robot can recognize an item in an environment, process its contextual information, and infer what type of affordances it can perform.

One approach that could help address learning in such a scenario is Reinforcement Learning (RL) [3]. Reinforcement Learning is a machine learning paradigm that allows an agent to improve its knowledge by interacting with the environment. It has been widely employed in robotics and, due to recently proposed methods such as Soft Actor-Critic (SAC) [4], it has been able to achieve great success in more complex tasks.

*This work was supported by CNPq and CAPES

[1]Renan Lima Baima is with Institute of Computing, State University of Campinas, Av. Albert Einstein, 1251, Brazil ra188960@ic.unicamp.br

[2]Esther Luna Colombini is with Institute of Computing, State University of Campinas, Av. Albert Einstein, 1251, Brazil esther@ic.unicamp.br

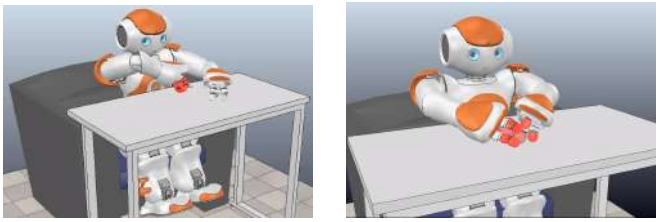


Fig. 1: Touch Affordance Example

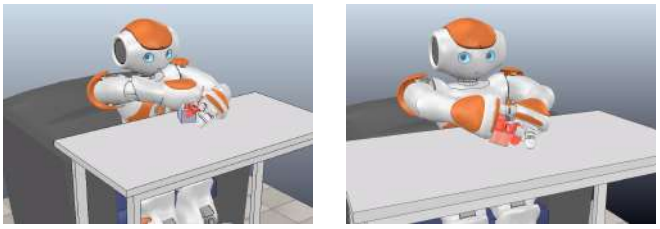


Fig. 2: Grasp Affordance Example

The strategy chosen here systematically links affordances and reward functions. We show that it is possible to design the reward function to model and learn an object's affordance by interacting with it via RL. We also demonstrate that we can add terms to this reward function to learn more complex affordances. This work shows how the policies learned in such a manner lead to the desired outcome of specific affordances and how we can reuse more complex affordances to solve simpler tasks,

## II. RELATED WORK

Gibson initially described affordance [5] as a list of actions that one agent can employ in an object, disregarding the context and the specifics of the agent [5], [6] as a list of actions that one agent can employ in an object, disregarding the context and the specifics of the agent [5], [6]. Affordances do not represent a property or quality of agents nor objects, but rather their in-between relation's characteristics [1]. Each object has limited affordances associated with it. The number of possible actions is also limited, reducing the computational complexity of choosing which to perform. As affordance relates to objects and agents, there are many different objects associated with affordances for various agents or tasks, e.g., a ball for a soccer player has an affordance to kick. Still, to a basketball player, the affordance is to bounce.

Usually, robots need to learn how to interact with objects by their limited way of perceiving their surroundings (e.g., image feature extraction, proximity sensors, and actuator

modulation), enabling them to synthesize affordance learning to build their cognitive affordance knowledge. According to Gaver [7], there are three categories of affordance: perceptible, hidden, and false; in other words, the object's affordance exists whether it is perceived or not by the agent, which changes the state of perception.

Reinforcement learning (RL) is one of the multiple strategies in AI that roboticists have been using to propose solutions to the autonomous manipulation problem. It is due mainly to new algorithms and architectures released over the past decade with astounding outcomes, beating state-of-the-art results and solving every day's new tasks. For example, Heess et al. [8] present simulated humanoid and animal-inspired legged models learning to perform actions (such as walk, jump, and avoid obstacles). Hence, learning how to perform affordances by interacting with objects might be carried out via RL.

One of the most successful model-free algorithms nowadays in Deep Reinforcement Learning (DRL) is Soft Actor-Critic (SAC) [9], the approach chosen for our implementation. SAC works by learning a policy $\pi_\theta$ and two Q-functions $Q_{\phi_1}$, $Q_{\phi_2}$, training a stochastic policy with entropy regularization and exploring in an on-policy way. The most significant advantage of SAC is that it proposes to solve the convergence brittleness by maximizing the trade-off between expected reward and aleatory in the policy; that is to say, it proposes to solve the exploration-exploitation duality, leading to a more efficient algorithm.

The use of affordances has been proven beneficial in robotics compared to other alternatives [10], [11], [12]. One of the many implementations shows that considering affordances may improve object recognition [13], [14], and human activity recognition [15], [16].

Robotic Reinforcement Learning approaches usually relate target objects, actions, and effects, with frameworks focusing on heuristical models that allow this learning. Ugur et al. [17] propose a staged developmental framework based on behavior. Initially, the robot knows basic individual motions (i.e., reach and catch); in a later phase, it uses those motions to discover its own set of fundamental actions. Finally, the robot keeps discovering new actions on different objects [18] until it learns the effects relative to actions.

Kaiser et al. [19] present a strategy that employs geometric data to extract self-body affordances (i.e., support lean, grasp). Montesano et al. propose an affordance formalization [20] applied to a full probabilistic Bayesian network model, which then learns the relation of an object, action, and outcome through a framework that identifies grasping points of different objects [21].

The precursor work of affordance in robotics focused on learning basic affordances of objects. Some of the most common object affordances that are learned in these contexts are: liability [22], rollability [23], pushability [24], traversability [25], and graspability [26]. When approaching the challenge of learning grasp affordances, one strategy is to estimate many different object positions that permit a strong grasp. The learned affordances have been used for action selection in an imitation game [27], [20] and planning a sequence of actions to achieve given tasks/goals [28].

The classification methodologies applied in [29] and [30] suggest that most works learn affordances by using trial and error techniques along with imitation from human demonstrations, employing visual feedback. This framework differs from the one we propose once no human demonstration is required in our work. Table I extends the analysis made by [29], showing how our work relates to those evaluated.

TABLE I: Similar affordance works classification (Adapted [29])

| Papers | Robotics Platform | Robotics Task | | | Perception | | | Actions | | Means of Data | | | Relations | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Manipulation | Navigation | Action Prediction | Visual | Proprioception | Kinesthetic | Tactile | Primitive | Complex | Trial and Error | Heuristics | Deterministic | Probabilistic | Planning |
| Present Work | NAO | X | | | | X | | X | X | X | X | | | X | X |
| [31] | Kurt 3D | | X | | X | | | | X | X | | X | X | | X |
| [28] | Gifu hand | X | | | X | | | | | X | | X | X | | X |
| [17] | Gifu hand | X | | | X | | | | X | | | X | X | | |
| [18] | Kuka arm | X | | | X | | | | | X | | X | X | | X |
| [20] | Baltazar | X | | | X | X | | | X | | | X | | X | |
| [21] | Baltazar | X | | | X | | | | X | | | X | | X | |
| [27] | Baltazar | X | | | X | | | X | X | | | X | | X | |
| [32] | CRS A251 | X | | | X | | | | X | | | X | X | | |
| [33] | PR2 | X | | | X | | | | X | | | X | X | | |
| [34] | Baby-BotCog | X | | | X | | | | X | | | X | X | | |
| [35] | PR2 | | | X | X | | | | X | X | | X | | X | X |
| [36] | Armar III | X | | | X | X | | | | X | | X | X | | |
| [19] | Armar III | X | | | X | X | | | | X | | X | X | | X |

Additionally, the work [2] presented by Daniel et al. also shows that none of the previews works treat affordance itself systematically and rigorously from the RL perspective, and only a few link affordances and general value function systematically, limiting the possibilities of predictions as embodiments of affordances in the real world usage. Although the previously mentioned works learned complex or even conjunctions of actions, they fail to learn the affordance of the object policy itself. Affordances like a sphere's rollability, an object's reachability, and, above all, how to reach and interact in a simple, optimized way are essential. In this way, as their agents did not learn simple affordances, they could prevent incrementally acquiring new affordances.

## III. MATERIAL AND METHODS

In this section, we describe the material and methods employed in our work.

### A. Material

All experiments were conducted in a simulated environment implemented in CoppeliaSim [37], using the NAO robot model [38]. In addition, we employed the PyRep toolkit to faster our executions [39], using the Rlkit Reinforcement Learning framework [40] to build our environments. SAC code's implementation and structure follow the author's instruction [4], [9] adapted to this project's requirements. In all experiments, we only control the joints of the arms and hands of the robot without employing any dynamic model of the robot.

### B. Method

In this work, we aim at having a robot capable of autonomously learning object affordances by interacting and identifying them in a simulated controlled environment, using fully online Deep Reinforcement Learning techniques. The experiments will evaluate if the same reward function can express an affordance to distinct objects. Hence, our robot will learn distinct affordances like touch and grasp for different objects such as a cube, a cylinder, and a sphere. We also assess if the robot can generalize its ability to act on different previously trained objects.

We designed two reward functions for two distinct tasks (touch and grasp). We used the same reward function to learn affordances on 3 different objects (sphere, cube, cylinder). We implemented the SAC algorithm with the MDP formulated in the next section to learn the relationship between the agent and the object. This implementation allows the agent to learn motor actions by interacting with the object guided by the reward functions. In this step, the robot learns the coordination of its motors and its perceptions. According to the reward obtained by the robot interaction, it autonomously adjusts the learned strategy. Unlike other works, the RL agent's reward function models affordance rather than a formal knowledge structured database. Effectively, the reward function represents the ability. In this case, each affordance requires a distinct reward function. For example, to define whether an object is **catchable**, the reward function will have to reward pressure sensor maintenance within a specific range, identifying whether there is something in hand.

Once we learned an affordance, we evaluate the possibility of extending it to other objects with distinct characteristics. In this scenario, we evaluate how much of the learned affordance is extensible for other objects.

To validate and evaluate the obtained policies, we randomly initialize $400$ scenes for the individual affordances against the trained object, unknown objects, and not trained object's initial placement.

## IV. Affordance Learning as an MDP

Mathematically, an RL problem can be formulated as a Markov Decision Process (MDP). An MDP is defined by the tuple $(S, A, P, R, \gamma)$, where $S$ is the set of states, $A$ is a finite set of actions, $P$ is the state-transition probability matrix, $R$ is the reward function, and $\gamma$ is the discount factor. In our case, $P$ is unknown to the agent, and $R$ will be defined by the agent's affordance definition. In this section, we will describe all these elements along with the episode termination condition.

### A. Observation space

To choose an observation space, we should consider the agent's information to learn object affordances through manipulation. To improve the learning process's efficiency, we should avoid redundancy; even though it does not cause any problems, it increases complexity. The reward function and the observation strategies vary depending on the desired affordance and its strategy.

*1) Sensorial observation inputs:* To manipulate an object, the agent needs information from the environment. Without this sensory information, no manipulation can be perceived nor learned. Therefore, we add sensory information in the observation vector.

For all trained policies, the following observations are considered: i) as the our robot has 13 symmetrical revolute joints on each arm's side, the current angular position of its **26 joints** $\{\theta J_t^1, ..., \theta J_t^{26}\}$ are added at every time step $t$ in the observation vector. The remaining joints' information is not added to the observation space once we focus on the manipulation problem, and thereby the legs, head, nor hip are relevant and controlled.

We also add the robot's absolute $x$ and $y$ coordinates $\{R_t^x, R_t^y\}$ to the observation vector. **The joints' position in the last episode** $\{\theta J_{t-1}^1, ..., \theta J_{t-1}^{26}\}$ are also in the observation vector as the last state's information is relevant to the affordance-cause-effect.

**The collision status of each phalanx and the hand's palms** with the object (there are 8 phalanxes in each hand + the palm collision) are also added in the observation vector $\{Col_t^1, ..., Col_t^{18}\}$. However, unlike the observation input mentioned earlier, this information is represented by a Boolean value (True, False).

**Object's size factor**: Although we do not know precisely the size of objects, humans can infer the object's size factor, i.e., whether it is "big," "small," "regular." As the object's size varies along training, we add to the observation vector a size factor $\Delta O_t$. It works as an approximation of the object size, assuming values between 0.7-1.3.

**Object's status**: The orientation $\{\theta O_t^\alpha, \theta O_t^\beta, \theta O_t^\gamma\}$ and the position of the object $\{\phi O_t^x, \phi O_t^y, \phi O_t^z\}$ relative to the agent's body is also something that humans can estimate and, therefore, is also added to the observation vector.

*2) Affordance-relative Observation Inputs:* In addition to the observations mentioned above, we have also defined other variables specific to each affordance.

**Touch affordance.** When learning this affordance, we also add to the observation vector the palm hands' distance to the object $\{\Lambda O_t^{leftpalm}, \Lambda O_t^{rightpalm}\}$. This information is especially appraised in the reward function and it stimulates the agent to keep interacting with the object.

**Grasping Affordance.** When learning this affordance, the following information is also used as part of the observation vector: i) the object's catchable status. If three or more parts of the robot's hand phalanges or palm are in contact with an object, $O_t^{catch}$ is set to true. Otherwise, false; ii) if the object was caught $O_t^{caught}$. We consider that if $O_t^{catch}$ is true for a consecutive specified number of seconds, the object was caught.

**The Observation Vector.** The complete observation vector is composed of:

$$
\begin{aligned}
O_t = \{A_{t-1}\} \cup \{\theta J_t^1, ..., \theta J_t^{26}\} \cup \{R_t^x, R_t^y\} \cup \\
\{\theta J_{t-1}^1, ..., \theta J_{t-1}^{26}\} \cup \{Col_t^1, ..., Col_t^{18}\} \cup \Delta O_t \cup \\
\{\theta O_t^\alpha, \theta O_t^\beta, \theta O_t^\gamma\} \cup \{\phi O_t^x, \phi O_t^y, \phi O_t^z\} \cup \\
\{\Lambda O_t^{leftpalm}, \Lambda O_t^{rightpalm}\} \cup O_t^{catch} \cup O_t^{caught},
\end{aligned} \tag{1}
$$

where $\{A_{t-1}\}$ is the action set from the last step, if $t \geq 1$. Otherwise, it is a set of zeros with the same size as the action space.

### B. Reward Function

Our reward functions are modeled in terms of affordances. We defined them in terms of the expected outcome when the corresponding affordance is accomplished. We have rewards and penalties associated with each affordance, as described next.

*1) Affordance-relative rewards:* **Touch affordance**: This component of the reward was designed to stimulate the agent's interaction with the object. It is relative to the **object's touch status**. We consider that, if the number of collisions between the agent's hands and the object is greater than three, a high reward is given. If at least one collision is present, a smaller reward is given. Assuming that a $true$ value equals 1 in $Col_t^j$, we summarize this component by:

$$
r_t^{touch} = \begin{cases} 80 & \text{if } \sum_{j=1}^{18} Col_t^j \geq 3 \\ 10 & \text{if } 0 < \sum_{j=1}^{18} Col_t^j < 3 \\ \text{-5} & \text{otherwise.} \end{cases} \tag{2}
$$

**Grasping Affordance**: This component of the reward is relative to the **object's catchable status** and **object was caught**. A positive reward of 6400 for a true **object was caught** and 800 for a true **object's catchable status**. **object's touch status** are also employed in this reward to stimulate the agent approaching the object. Assuming that a $true$ value equals 1, we summarize this component by:

$$
r_t^{grasp} = \begin{cases} 6400 & \text{if } O_t^{caught} = 1 \text{ and } (\Lambda O_t^{palmLeft} = 1 \\ & \text{or } \Lambda O_t^{palmRight} = 1) \\ 800 & \text{if } O_t^{catch} = 1 \\ \text{-400} & \text{otherwise} \end{cases} \tag{3}
$$

*2) Penalties:* The following terms are added to the reward function in the form of punishments.

**No interaction Punishment** One of the first observed flaws in the learned manipulation is that the agent initially tried to move its actuators regardless of where to randomly. To prevent this from happening, we added a penalty term proportional to no **object's touch status**, **as mentioned on equation 2**. However, we observed that the policy quickly increased interaction with the object, leading to movement exploitation.

**Palm hands' distance to the object**: The object-hand distance punishment is a negative reward proportional to the distance from the robot hand to the object. If the robot is not interacting with the object, it receives a negative value equal to the distance of both hands to the object times 50 plus the **no interaction reward**; otherwise, no punishment is given, according to:

$$
\Delta O_t^{palm} = -50 * \Lambda O_t^{palmLeft} - 50 * \Lambda O_t^{palmRight}. \tag{4}
$$

**Object let go penalty**: We have introduced this penalty to prevent the robot from letting the object go, **as mentioned in equation 3**. It prevents it from touching and releasing the object and making random moves to maximize single movements instead of learning the affordance.

**The reward composition**: the final reward function is:

$$
R_t += r_t^{touch} + r_t^{grasp} + \Delta O_t^{palm} \tag{5}
$$

### C. Action Space

Like the observation space, we must define the action space of our MDP. In our case, we will control all 26 revolute joints related to the robot's arms, hands, and fingers. We control the robot by the joints' angular position, with a fixed torque, velocity, and movement limited by factory motor limitations.

As mentioned earlier, all 26 arm-related joints in NAO are revolute, and all have limitations in their movement range. Therefore, we keep all as limited by the factory setup. Hence, the action space is defined by an vector of angular joint control that will be output to the robot $\mathcal{A}_t = \{\theta J_t^1, ..., \theta J_t^{26}\}$.

We also perform random movements in each motor at each reset, so the agent starts in a new state. The object is also randomly placed in a specific table location at each reset. This process adds randomness and variability to each episode, aiding SAC to generalize the policy learned.

### D. Episode termination

In theory, the most logical method is to terminate the episode after reaching the predefined affordance permanently. However, this strategy limits the agent/object interaction and affordance maximization, being interpreted as penalizing the agent. Thus, we opted not to stop the episode unless it reached a limited number of incremental steps or the object felt off the table. Although this may seem like a harmless modification, this late termination freed our agent from interacting freely with the object, exploring failures, different situations, points of contact, and maximizing affordance until it did not reach the maximum steps.

## V. Results

### A. Learned Policies

Our experiments aim to verify if the agent can learn how to manipulate different objects while the reward function defines which affordance is under learning.

First, we decided to evaluate whether it could achieve such affordances without the proposed rewards structure. This means we just gave a very positive reward when achieving the affordance in a regular reinforcement strategy. However, the robot could not learn any affordances with this approach. In this failed strategy, the robot only receives rewards related

to the affordance of grasping the object only if the agent does it, with no additional intermediate rewards.

Considering the MDPs described earlier, we trained three policies with the same reward function for three different objects (ball, cube, and cylinder). Although the same affordances are harder to learn with particular objects' geometry, the robot learned them all. Figure 3 shows the average reward per episode for the touch and grasp affordances for each object. These experiments show that the same abstract definition of affordance employed (the concept of touching by sensing the object's contact with the hand) allowed learning the affordances of different objects. We also see that the object's geometry influences the learning curve—the same happening with the affordance complexity. As touch is more effortless than grasp, the first is quicker to learn.

When comparing the grasp reward of the cube with the others in Figure 3, we can see that the agent faces an initial difficulty. However, it is still learned in the specified number of episodes. Nevertheless, our algorithm develops well over the ball/sphere scenario even though its rollability affordance sometimes causes the robot to lose the ball. For this case, the policy reaches a peak of 3000, contrasting with the cylinder and cube around 1500. This could be caused because the ball fits better the particular structure of the robot's hand.

### *B. Learned Affordances*

Figure 1 shows the results of experiments with our learned policies for the **touch affordance**. In this case, we show that the robot learned the touch affordance with one hand and with both. It also learned how to touch it from above and from beside. Figure 2 shows an example of the policy of **grasp affordance** trained with the sphere but tested with unknown objects during training. As it grasps cylinders and cubes with both hands or single-handed, the robot also masters the object's size differences and different geometric forms. We will show next how much these learned policies are transferable.

### *C. Learned Affordance Quality and Generalization*

TABLE II: Touch Affordance Executions Success Rate

| Policy Tested Heat-Map Image | Environment: Object Trained | Environment: Object Tested | Success criteria: Successful Trials | Success criteria: Success Rate | Policy Coverage: Object Known? |
|---|---|---|---|---|---|
| Figure 4a.I | Sphere | Cube | 284 | 71.00% | No |
| Figure 4a.II | | Cylinder | 343 | 85.75% | No |
| Figure 4a.III | | Sphere | 377 | 94.25% | Yes |
| Figure 4c.I | Cube | Cube | 348 | 87.00% | No |
| Figure 4c.II | | Cylinder | 311 | 77.75% | No |
| Figure 4c.II | | Sphere | 296 | 74.00% | Yes |
| Figure 4e.I | Cylinder | Cube | 385 | 96.25% | No |
| Figure 4e.II | | Cylinder | 341 | 85.25% | No |
| Figure 4e.III | | Sphere | 282 | 70.50% | Yes |

To assess each learned policy's effectiveness regarding the desired affordance, we run the six learned policies over 400 random initializations with a maximum of 250 steps. Whenever the agent drops the object or achieves the affordance, the episode is stopped, we reset the environment, and the policy restarts. Finally, we count how many steps were necessary to execute the task.

TABLE III: Grasp Affordance Executions Success Rate

| Policy Tested Heat-Map Image | Environment: Object Trained | Environment: Object Tested | Success criteria: Successful Trials | Success criteria: Success Rate | Policy Coverage: Object Known? |
|---|---|---|---|---|---|
| Figure 4b.I | Sphere | Cube | 189 | 47.25% | No |
| Figure 4b.II | | Cylinder | 224 | 56.00% | No |
| Figure 4b.III | | Sphere | 295 | 73.75% | Yes |
| Figure 4d.I | Cube | Cube | 364 | 91.00% | Yes |
| Figure 4d.II | | Cylinder | 309 | 77.25% | No |
| Figure 4d.III | | Sphere | 286 | 71.50% | No |
| Figure 4f.I | Cylinder | Cube | 346 | 86.50% | No |
| Figure 4f.II | | Cylinder | 281 | 70.25% | Yes |
| Figure 4f.III | | Sphere | 320 | 80.00% | No |

We test each policy with the object used to train it and those it did not know during training. For each test, the object is randomly positioned in a predetermined area. The object's size also differs, which increases the scene's complexity.

Tables II and III present the success rates for each affordance evaluated and the object used in the test phase. Figure 4 shows the object's position when the affordance is completed. When the object felt, or the policy reached the maximum steps, we did not take the position and considered it a failure. The circle's size is proportional to how many steps it took till the affordance was executed. The larger it is, the more steps it takes to execute the affordance, and vice versa. The circle's heat map color codes how many times it takes the agent this specific number of steps to complete the affordance; the more it happened, the redder it is.

Table II shows that the Touch affordance in the best scenario achieves 96.25% of episodical success. This best-case relates to a policy trained on the object's original geometry. Thus, among 400 random resets, it succeeds in 385 trials. Figure 4a shows that whether the object has a cubic or cylinder (not trained) or spherical (trained) form, it performs the affordance mostly in 4 steps in various regions of the table. Even when the policy does not know the object (cube or cylinder), it succeeds in 71.00% of the worst-case trials for the cubic form.

Likewise, Table III shows that the Grasping affordance in the best scenario achieves 91.00% of episodical success. Its best-case (Figure 4d) relates to a policy trained on the object's original geometry. Thus, among 400 random resets, it succeeds in 364 trials. It also shows that whether the object has a Cubic (trained) or spherical or cylindrical (not trained) form, it performs the affordance mostly in 2 steps in various regions of the table, and even when the policy does not know the object (sphere), it succeeds in 71.50% of trials. Moreover, when we tested that policy in the touch affordance environment, it succeeds in 82.25% of executions.

Many methods envision the ability of robots to perceive and imitate the way humans learn during infant development. However, mainly from the perspective of robotic tasks, there are still many aspects that remain unclear. Some of these

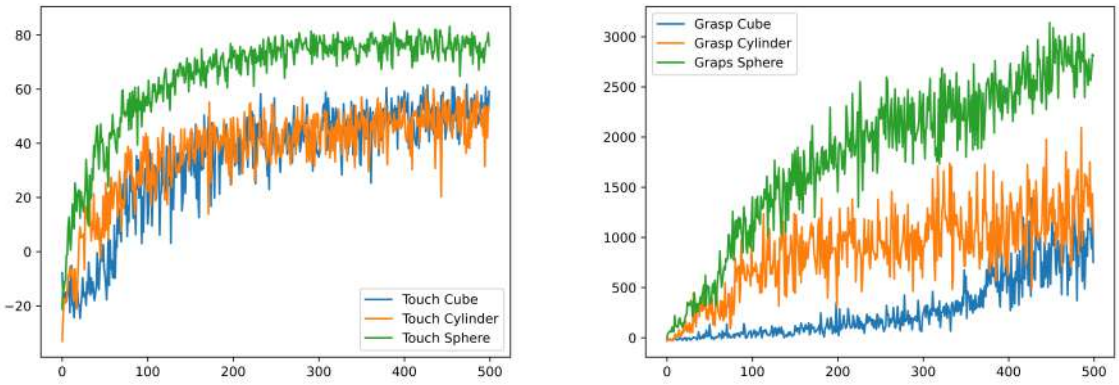



Fig. 3: Average reward per episode curves for the touch (left image) and grasp (right image) affordances. We trained three policies with the same reward function for 3 different objects. Although the same affordances are harder to learn with particular objects' geometry, the robot learned them all.

aspects include ambiguity about how affordability affects tasks, the lack of data sets that represent the components of affordable tasks, and the lack of standardized metrics for evaluating robotic tasks that require the concept of affordability, all of which are well addressed on [29].

Our formulation verified that, although it is possible to link the affordance definition to the reward function, it must contain intermediate elements. This could also be true for incremental affordances, e.g., the grab affordance has the touching affordance as a previous step. Penalties were also important to achieve a more consistent learning process. That is, the agent would focus on fewer actions for better results.

## VI. CONCLUSIONS

In this work, we proposed learning affordance through object manipulation. To that end, we designed an affordance learning framework with objects comprehensive enough to guarantee a higher learning abstraction level in a manipulation scenario. These distinct abstraction levels were designed through a relationship between affordance and reinforcement learning, where the reward function represents the affordance itself. Results showed that we could learn an object's affordance while manipulating an object. Moreover, we showed that the same reward function could learn similar affordances with distinct objects. Finally, the results show that the learned policies could also complete their goals tested on different objects. For future work, we aim at addressing how incremental changes in the reward function complexity could relate to incremental affordances.

## ACKNOWLEDGMENT

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Code 001 and Conselho Nacional de Desenvolvimento Cientifico e Tecnologico (CNPq) (133343/2019-7)

## REFERENCES

[1] J. J. Gibson, *The Ecological Approach to Visual Perception*. Houghton Mifflin, 1979.

[2] D. Graves, J. Günther, and J. Luo, "Affordance as general value function: a computational model," *Adaptive Behavior*, p. 1059712321999421, 2021.

[3] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," 2011.

[4] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft actor-critic algorithms and applications," 2019.

[5] J. J. Gibson, *The perception of the visual world.* Houghton Mifflin, 1950.

[6] ——, "The theory of affordances," *GIESEKING, Jen Jack; MANGOLD, William; KATZ, Cindi; LOW, Setha*, pp. 56–60, 2014.

[7] W. W. Gaver, "Technology affordances," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '91. New York, NY, USA: ACM, 1991, pp. 79–84. [Online]. Available: http://doi.acm.org/10.1145/108844.108856

[8] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami, *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv preprint arXiv:1707.02286*, 2017.

[9] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1861–1870.

[10] H. Min, C. Yi, R. Luo, J. Zhu, and S. Bi, "Affordance research in developmental robotics: A survey," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 8, no. 4, pp. 237–255, 2016.

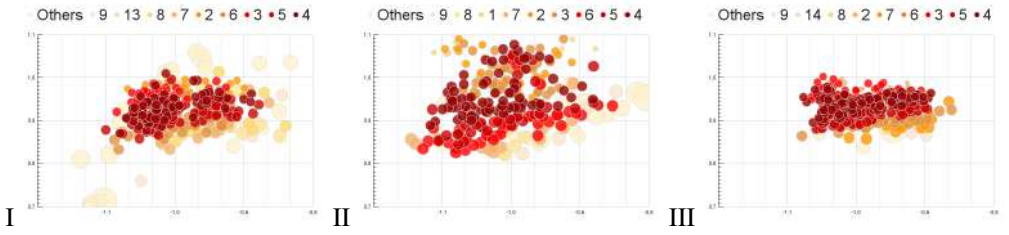
[11] P. Zech, S. Haller, S. R. Lakani, B. Ridge, E. Ugur, and J. Piater, "Computational models of affordance in robotics: a taxonomy and systematic classification," *Adaptive Behavior*, vol. 25, no. 5, pp. 235–271, 2017.

[12] C. Wang, "Use of affordances for efficient robot learning," Ph.D. dissertation, Delft University of Technology, 2017.

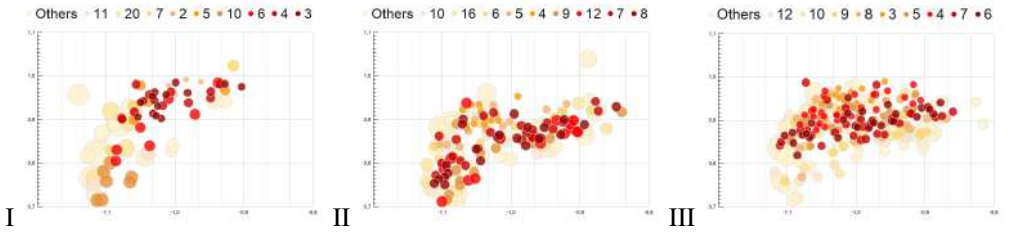
[13] C. Castellini, T. Tommasi, N. Noceti, F. Odone, and B. Caputo, "Using object affordances to improve object recognition," *IEEE Transactions on Autonomous Mental Development*, vol. 3, no. 3, pp. 207–215, 2011.

[14] T. Hermans, J. M. Rehg, and A. Bobick, "Affordance prediction via learned object attributes," in *IEEE International Conference on Robotics and Automation (ICRA): Workshop on Semantic Perception, Mapping, and Exploration*, 2011, pp. 181–184.

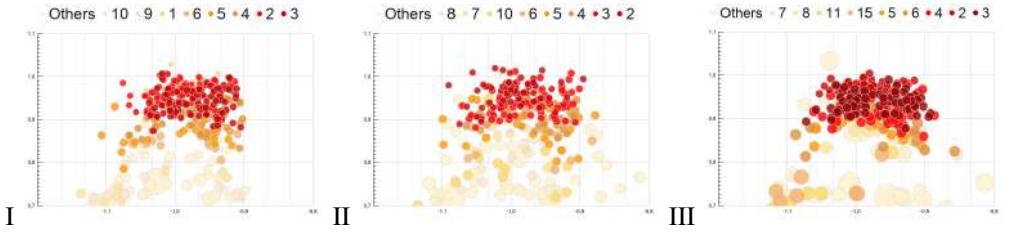
[15] H. S. Koppula, R. Gupta, and A. Saxena, "Learning human activities and object affordances from rgb-d videos," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 951–970, 2013.



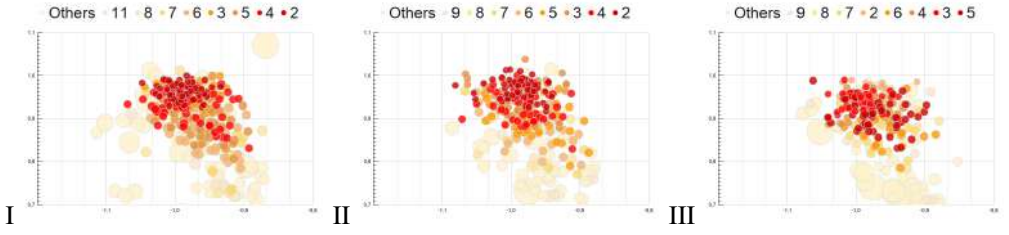

(a) **Sphere Touch** Affordance Execution; Policy Tested with I-Cube, II-Cylinder and III-Sphere respectively



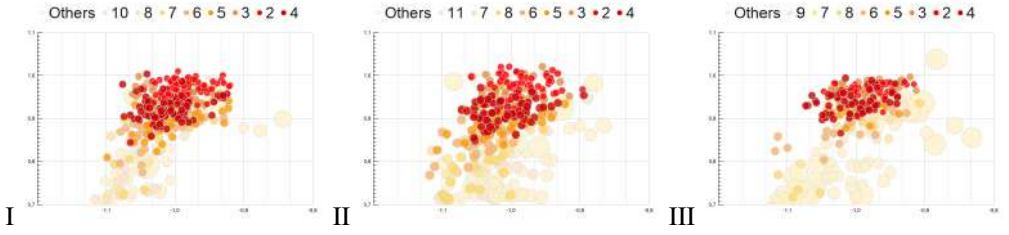

(b) **Sphere Grasp** Affordance Execution; Policy Tested with I-Cube, II-Cylinder and III-Sphere respectively



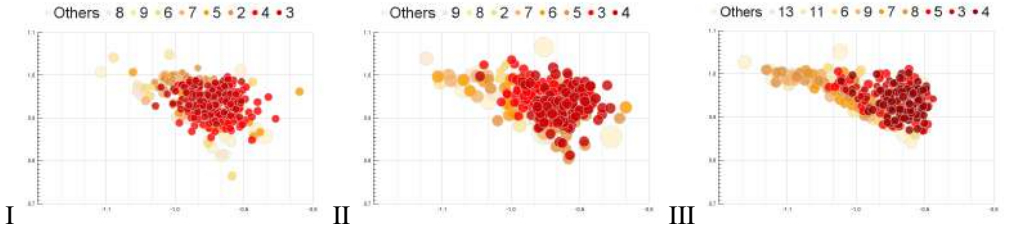

(c) **Cube Touch** Affordance Execution; Policy Tested with I-Cube, II-Cylinder and III-Sphere respectively




(d) **Cube Grasp** Affordance Execution; Policy Tested with I-Cube, II-Cylinder and III-Sphere respectively




(e) **Cylinder Touch** Affordance Execution; Policy Tested with I-Cube, II-Cylinder and III-Sphere respectively




(f) **Cylinder Grasp** Affordance Execution; Policy Tested with I-Cube, II-Cylinder and III-Sphere respectively

Fig. 4: Position-Frequency Heat-Map of Successful Affordance Execution; The size of the dot represents how many steps it took for the robot to execute the desired affordance, the color represents how frequent it is to the robot execute over the total chosen test cases; The bigger, the more steps it takes; The hotter (darker red) the most frequent it is

[16] Y. Sun, S. Ren, and Y. Lin, "Object–object interaction affordance learning," *Robotics and Autonomous Systems*, vol. 62, no. 4, pp. 487–496, 2014.

[17] E. Ugur, Y. Nagai, E. Sahin, and E. Oztop, "Staged development of robot skills: Behavior formation, affordance learning and imitation with motionese," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 2, pp. 119–139, 2015.

[18] E. Ugur and J. Piater, "Bottom-up learning of object categories, action effects and logical rules: From continuous manipulative exploration to symbolic planning," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2627–2633.

[19] P. Kaiser, E. E. Aksoy, M. Grotz, and T. Asfour, "Towards a hierarchy of loco-manipulation affordances," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2839–2846.

[20] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, "Affordances, development and imitation," in *2007 IEEE 6th International Conference on Development and Learning*. IEEE, 2007, pp. 270–275.

[21] ——, "Modeling affordances using bayesian networks," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2007, pp. 4102–4107.

[22] L. Paletta, G. Fritz, F. Kintzler, J. Irran, and G. Dorffner, "Perception and developmental learning of affordances in autonomous robots," *KI 2007: Advances in Artificial Intelligence*, pp. 235–250, 2007.

[23] B. Ridge, D. Skočaj, and A. Leonardis, "Self-supervised cross-modal online learning of basic object affordances for developmental robotic systems," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5047–5054.

[24] M. Sánchez-Fibla, A. Duff, and P. F. Verschure, "The acquisition of intentionally indexed and object centered affordance gradients: a biomimetic controller and mobile robotics benchmark," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 1115–1121.

[25] T. Yamashiro and K. Ito, "Comparative study of affordance-based navigation and model-based navigation: Experimental analysis of learning ability of mobile robot that taps objects with a stick for navigation," in *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*. IEEE, 2011, pp. 372–377.

[26] T. Geng, J. Wilson, M. Sheldon, M. Lee, and M. Hülse, "Synergy-based affordance learning for robotic grasping," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1626–1640, 2013.

[27] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, "Learning object affordances: from sensory–motor coordination to imitation," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 15–26, 2008.

[28] E. Ugur, E. Oztop, and E. Sahin, "Goal emulation and planning in perceptual space using learned affordances," *Robotics and Autonomous Systems*, vol. 59, no. 7, pp. 580–595, 2011.

[29] P. Ardón, È. Pairet, K. S. Lohan, S. Ramamoorthy, and R. Petrick, "Affordances in robotic tasks–a survey," *arXiv preprint arXiv:2004.07400*, 2020.

[30] N. Yamanobe, W. Wan, I. G. Ramirez-Alpizar, D. Petit, T. Tsuji, S. Akizuki, M. Hashimoto, K. Nagata, and K. Harada, "A brief review of affordance in robotic manipulation research," *Advanced Robotics*, vol. 31, no. 19-20, pp. 1086–1101, 2017.

[31] E. Ugur, E. Şahin, and E. Oztop, "Unsupervised learning of object affordances for planning in a mobile manipulation platform," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 4312–4317.

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

[33] T. Hermans, J. M. Rehg, and A. F. Bobick, "Decoupling behavior, perception, and control for autonomous learning of affordances," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 4989–4996.

[34] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini, "Learning about objects through action-initial steps towards artificial cognition," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 3. IEEE, 2003, pp. 3140–3145.

[35] H. S. Koppula and A. Saxena, "Anticipating human activities using object affordances for reactive robotic response," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 14–29, 2016.

[36] P. Kaiser, M. Grotz, E. E. Aksoy, M. Do, N. Vahrenkamp, and T. Asfour, "Validation of whole-body loco-manipulation affordances for pushability and liftability," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 920–927.

[37] E. Rohmer, S. P. N. Singh, and M. Freese, "V-rep: a versatile and scalable robot simulation framework," in *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[38] M. Cognetti, P. Mohammadi, and G. Oriolo, "Whole-body motion planning for humanoids based on com movement primitives," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 1090–1095.

[39] S. James, M. Freese, and A. J. Davison, "Pyrep: Bringing v-rep to deep robot learning," 2019.

[40] V. Pong, M. Dalal, S. Lin, and A. Nair, "Rlkit: Reinforcement learning framework and algorithms implemented in pytorch, 2019," *URL https://github. com/vitchyr/rlkit*, 2019.