Abstract

Robot technology is one of the pillars of modern society. Advances in information, electronic, and mechanical fields enable us to build and program machines to perform tasks in very different contexts, such as industry, surgery, and space missions.
Specifically, in manufacturing, robots are mainly used to perform repetitive and unhealthy works like assembly, welding and material handling, thanks to their mechanical robustness and ability to repeatedly perform the same movements with high accuracy and precision.

While in the early day, robot systems were constrained in isolated and known environments. Over the past few decades, robots have been asked to solve tasks in dynamic and unknown/partially-known environments, where they must **coexist** and **cooperate** with humans, while solving different **dynamic** tasks [1] (e.g. pick a requested object, whose position is not known a priori).

In this scenario, the desired characteristics of such robotic systems are: **(a) Adaptability to new conditions**, i.e., the system must be able to easily adapt to dynamic changes in system and environmental conditions, performing *"intelligent" behaviors* to handle these new scenarios and solve the desired task; **(b) Adaptability to new tasks**, i.e., the system must be able to easily adapt to both new variations of a known task and completely new tasks by exploiting experience to infer actions and solve them;

These requirements, can be challenging to achieve with traditional robot programming techniques based on hand-written policies and control methods. These conventional techniques often require a meticulous analysis of process dynamics, the construction of an analytical model, and the derivation of a control law that meets specific design criteria. This design process is tedious and time-consuming, particularly when high-level perception systems (e.g., cameras, microphones, motion sensors) are used to infer the state of the environment (such as the unknown position of a desired object relative to the end-effector) and the intentions of the human operator.

In contrast, significant advancements have been made by leveraging *learning techniques*, where the control policy is learned from data. This data can be generated either by **agent experience** [2] or by **expert demonstration** [3]. In the case of agent experience, there is a trial-and-error procedure where the control policy generates actions executed by an agent, which interacts with the environment. The parameters are then tuned according to the effectiveness of the actions, based on their impact on the environment relative to the task to be solved. In the case of expert demonstration, the control policy parameters are directly tuned using a dataset containing examples of task execution. The goal is to replicate the tasks observed in the dataset.

Specifically this thesis is framed in the context of *Learning from Demonstration* (LfD), a learning approach based on expert demonstration. According to the requirements of adaptability the thesis focus on a specific aspect of LfD named *Multi-Task LfD*. In this case, the control policy is not trained to execute a single task (e.g., picking an object) with the goal of generalizing across different objects and initial conditions [4, 5]. Instead, it is trained to handle various variations of a specific task (e.g., picking an object from different possible locations) [6] or even entirely different tasks (e.g., a single control policy that solves both picking and placing tasks as well as assembly tasks) [7, 8]. The goal is to generalize not only with respect to the objects being manipulated and the initial conditions but also with respect to the tasks themselves. This means that by leveraging the knowledge-sharing hypothesis, we can achieve a system capable of solving new variations.

In this scenario, the learning procedure is much more challenging because we need to include and define the **conditioning signal** (i.e., the signal that informs the policy about the task to execute, the object to manipulate, and the target placing location). Additionally, the environment can contain **multiple distractor objects** (e.g., objects that can potentially be manipulated but are not of interest for a given task variation).

Regarding the conditioning signal, there are at least two intuitive approaches. The first is through a natural language description of the task to be executed [9, 10, 7], and the second is through a video demonstration [6, 8]. In the former, the task is described using phrases that specify the task details, such as “Pick the red box and place it into the first bin”. Given in input the phrase, the system must be able to infer the intent of the task (i.e., the pick-place operation) and the object of interest (e.g., red-box for picking and first bin for placing), and correlate this information with the environment and robot state to effectively control the robot. In the latter, another agent (either a robot or a human operator) performs the task in a different environment configuration, records this execution, and provides the video as input to the control policy. The control policy must then infer the intent from the video (i.e., the task to be performed, the object to be manipulated, and the final state) and control the robot to complete the task according to the agent’s state, the environment’s state, and the commanded task. Inspired by how humans can learn to replicate tasks by simply observing their execution, the main goal of this thesis is to develop a system capable of replicating tasks shown in a video demonstration. This involves addressing challenges related to extracting task-relevant information from the video, such as identifying the manipulated object and its final position.

Regarding the issue of distractor objects, these are generally items that are not considered in manipulation operations, which simplifies the problem significantly. However, in the context proposed in this thesis, the problem is further complicated by the fact that the semantic meaning of an object of interest or a distractor is defined at run-time by the command itself. This means that if the initial configuration consists of four objects (e.g., four boxes of different colors), a specific object may or may not be of interest based on the command given to the robot.

The main contribution of this thesis is addressing the issue of distractor objects. Specifically, it was observed that a significant problem with the methods proposed in the literature is that while the learned control policy can generate valid trajectories, allowing the robot to reach, pick, and place objects, it often manipulates the wrong object. To address this problem, two main considerations have been made:

(1) The architectures proposed in current literature predominantly consist of end-to-end architectures, which translate high-level inputs such as images into corresponding actions. Consequently, the model must acquire an implicit representation capable of encoding both the task objective and the current state of the environment, including the location of the target object; **(2)** The learning procedure optimizes a metric that is not directly linked to task success but rather aims to replicate actions similar to those of the expert, on average.

These two aspects can lead to a control policy that is not able to effectively guide the robot toward the target object. Indeed, it was observed that one of the critical points during trajectory execution is the first steps. Small errors in these initial steps can result in reaching and consequently picking the wrong object.

Based on these considerations, this thesis evaluates the possibility of developing a system that explicitly reasons about the objects of interest (e.g., target object and placing location). The control module is

then directly informed with low-level information, such as the position of the target object.

To perform this explicit reasoning, a *Conditioned Object Detector* (COD) has been developed. This module, given the video demonstration and the current agent observation as input, predicts the class-agnostic bounding box related to the target object and the final placing location. This low-level positional information is then provided to the control module, which predicts the actions to perform.

The learning procedure is then divided into two steps. The first step involves training the *Conditioned Object Detector* (COD) module, which focuses on explicitly solving cognitive tasks, such as detecting regions of interest represented by the object to be manipulated and its final location. The second step involves training the *Object Conditioned Control Policy* (OCCP), which focuses on solving the control problem using low-level positional information that can be easily mapped into the corresponding actions.

The final system has been tested in both **multi-variation single-task** scenarios and **multi-variation multi-task** scenarios. Specifically, the system was evaluated on four different tasks: Pick-Place, Nut-Assembly, Stack-Block, and Button-Press. Each task had different variations based on the manipulated object and the final state. While the tasks share common properties, they also have specific characteristics. For example, the Nut-Assembly task involves contact-rich, precise manipulation, whereas Pick-Place can be solved in a much rougher manner.

Overall, the proposed methods demonstrated very promising behaviors and a general improvement over baseline methods that do not include object-related reasoning. This shows that solving manipulation tasks with an object-oriented approach can be an effective paradigm for LfD problems. Additionally, this approach provides interpretable information to the end user, as the predicted bounding boxes can be interpreted as the locations where the robot will move.

In conclusion, this thesis addresses the problem of leveraging object priors in the context of Multi-Task Imitation Learning. The proposed methods have been tested in both simulated and real-world scenarios, demonstrating their effectiveness.

Contents

ToDo.

- ToDo -

2

Chapter 1

Introduction

1.1 Application context

1.2 Motivation and thesis overview

1.3 State of the art

The chapter reviews the state of the art (SoTA) on the LfD problem. Specifically, Section 1.3.1 will focus on the formalization of the LfD problem. Then, Section 1.3.2 will present various approaches and methodologies for solving the LfD problem. This will include a detailed taxonomy of the different approaches, highlighting their pros and cons, and concluding with considerations relevant to the goals of this thesis.

1.3.1 Problem formulation

1.3.2 Learning from demonstration

1.3.2.1 Behavioral Cloning

This section is dedicated to the presentation of *Behavioral Cloning* (BC) methods.

Dynamical Movement Primitives
Dynamical Movement Primitives (DMPs), offer a robust framework for encoding and reproducing complex movements through differential equations and attractor dynamics.

Single-Task Imitation Learning
The *Single-Task Imitation Learning* refers to deep architecture designed to learn and replicate specific tasks from given demonstrations.

Interactive Imitation Learning
In *Interactive Imitation Learning* the learning process is augmented by interaction with a teacher, allowing for real-time feedback and adjustments to improve performance.

Multi-Task Imitation Learning
Multi-Task Imitation Learning enables the learning and execution of multiple tasks from a set of demonstrations, highlighting the scalability and versatility of these methods.

Object-Oriented Imitation Learning
Object-Oriented Imitation Learning focuses on learning behaviors in relation to specific objects and their interactions, providing a more structured and contextual approach to imitation learning.

1.3.2.2 Inverse Reinforcement Learning

This section will be dedicated to the introduction of the *Inverse Reinforcement Learning* (IRL) approach.

1.3.2.3 Generative Adversarial Imitation Learning

This section will be dedicated to the introduction of the *Generative Adversarial Imitation Learning* (GAIL) approach.

1.3.2.4 Learning from Observation

This section will be dedicated to the introduction of the *Learning from Observation* (LfO) approach.

1.3.2.4.1 Model-Free

1.3.2.4.2 Model-Based

1.3.3 Graph Neural Network for planning systems

Chapter 2

Conditioned object detector

In this chapter the COD is going to be described. Specifically, Section 2.1 will outline the detection problem being addressed. Section 2.2 will detail the proposed architecture designed to solve the described problem. Section 2.3 will discuss the experimental setup and present the results obtained from testing the proposed architecture.

2.1 Problem formulation

2.2 Architecture

2.3 Experiments

In this section, the performed experiments are going to be described. Specifically, in Section 2.3.1 the dataset used for training procedure will be described. Section 2.3.2 will report the obtained results.

2.3.1 Dataset

2.3.2 Results

This section presents the obtained results, divided into two main blocks. The first block (Section 2.3.2.1) discusses the results of the method trained to detect only the target object. The second block (Section 2.3.2.1) covers the results of the method trained to detect both the target object and the final (placing) location. For each method, results are reported for two different scenarios: first, where the method is trained in a single-task multi-variation scenario; and second, where the method is trained in a multi-task multi-variation scenario.

2.3.2.1 Target object detector

Single-task multi-variation scenario

Multi-task multi-variation scenario

2.3.2.2 Target object and final position detector

Single-task multi-variation scenario

Multi-task multi-variation scenario

Chapter 3

Object conditioned control policy

In this chapter the OCCP is going to be described. Specifically, Section 3.1 will outline the problem being addressed. Section 3.2 will detail the proposed architecture designed to solve the described problem. Section 3.3 will discuss the experimental setup and present the results obtained from testing the proposed architecture.

3.1 Problem formulation

3.2 Architecture

This section describes the entire policy architecture, specifically focusing on how the COD (Chapter 2) is integrated. There are two control architectures, differing in how the final control module predicts actions. Section 3.2.1 details the architecture that uses a single control module to predict actions for both the reaching and placing phases. In contrast, Section 3.2.2 describes the architecture that divides the control module into two distinct parts: one for computing actions during the reaching phase, and another for the placing phase.

3.2.1 Single control module

3.2.2 Double control modules

3.3 Experimental results

In this section, the performed experiments are going to be described. Specifically, in Section 3.3.1 the dataset used for training procedure will be described. Section 3.3.2 will report the obtained results.

3.3.1 Dataset

3.3.2 Results

This section presents the obtained results, divided into two main blocks. The first block (Section 3.3.2.1) discusses the results of the method described in Section3.2.1. The second block (Section 3.3.2.2) covers the results of the method described in 3.2.2. For each method, results are reported for two different scenarios: first, where the method is trained in a single-task multi-variation scenario; and second, where the method is trained in a multi-task multi-variation scenario.

3.3.2.1 Single control module

Single-task multi-variation scenario

Multi-task multi-variation scenario

3.3.2.2 Double control modules

Single-task multi-variation scenario

Multi-task multi-variation scenario

Chapter 4

Graph Neural Network for heuristic estimation

Chapter 5

Real world application

This chapter details the validation of the proposed methods in a real world scenario. Specifically, Section 5.1 describes the experimental setup. Section 5.2 discusses the dataset used to train the system. Finally, Section 5.3 presents the results obtained.

5.1 Experimental Setting

5.2 Dataset

5.3 Results

Chapter 6

Conclusions

Bibliography

[1] S. Bini, G. Percannella, A. Saggese, and M. Vento, "A multi-task network for speaker and command recognition in industrial environments," *Pattern Recognition Letters*, vol. 176, pp. 62–68, 2023.

[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction.* MIT press, 2018.

[3] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[4] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, 2018, pp. 5628–5635.

[5] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning.* PMLR, 2022, pp. 1678–1690.

[6] S. Dasari and A. Gupta, "Transformers for one-shot visual imitation," in *Conference on Robot Learning.* PMLR, 2021, pp. 2071–2084.

[7] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. S. Ryoo, G. Salazar, P. R. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. T. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "RT-1: robotics transformer for real-world control at scale," in *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, K. E. Bekris, K. Hauser, S. L. Herbert, and J. Yu, Eds., 2023. [Online]. Available: https://doi.org/10.15607/RSS.2023.XIX.025

[8] Z. Mandi, F. Liu, K. Lee, and P. Abbeel, "Towards more generalizable one-shot visual imitation learning," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2434–2444.

[9] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. Ben Amor, "Language-conditioned imitation learning for robot manipulation tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 139–13 150, 2020.

[10] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 3, pp. 7327–7334, 2022.

List of Figures

List of Tables