A Motion Capture and Imitation Learning-based Approach to Robot Control

A Preprint

Peteris Racinskis[*†] peteris.racinskis@edi.lv

Janis Arents[*†] janis.arents@edi.lv

Modris Greitans[†] modris_greitans@edi.lv

June 20, 2022

Abstract

Imitation Learning is a discipline of Machine Learning primarily concerned with replicating observed behavior of agents known to perform well on a given task, collected in demonstration data sets. In an industrial robotics context this presents the opportunity to replace explicit programming of behavior with demonstrations of the task to be performed. Motion capture is one of the methods for recording such data. It enables lesser model complexity compared to more indirect observation modalities such as visual data, yet requires additional data pre-processing if signals beyond a time series of effector positions and orientations are relevant to the task at hand. In this paper, an approach for motion capture-based imitation learning and implicit control signal estimation is introduced and evaluated on an object throwing task.

Keywords Imitation Learning · Motion capture · Robotics · Artificial neural networks · RNN

1 Introduction

Manipulator arms and other types of robots have become ubiquitous in modern industry and their use has been proliferating for decades, yet even now the primary method for programming these devices remains procedural code, hand-crafted by specially trained technicians and engineers. This significantly increases the cost and complexity of commissioning process nodes that utilize industrial robots. To this end, various alternative approaches grounded in machine learning have been proposed. Perhaps the most commonly studied are methods that fall under the umbrella of reinforcement learning, which optimize an agent acting on its environment through the use of an explicitly specified reward function [Sutton and Barto, 2018]. While offering strong theoretical guarantees of convergence, they require interaction with the environment for learning to occur. Furthermore, in practical settings the reward function for many tasks is very sparse, leading to large search spaces and inefficient exploration [Hester et al., 2018]. Finally, the reward function for a task may be unknown or difficult to specify analytically [Abbeel and Ng, 2004].

Imitation learning is a broad field of study in its own right that promises to overcome some or all of the aforementioned challenges, provided that it is possible to obtain a corpus of demonstrations showing how the task is to be accomplished. When dealing with industrial robotics, this is often the case as the tasks we wish to execute are ones that human operators already routinely perform. Therefore, we have developed an approach for recording actions taken by humans in the physical environment, programmatically producing a training data set structured in the form of explicit demonstrations augmented with additional, extrapolated state inputs and control signals, and training artificial neural network models to reproduce the trajectories therein as motion plans.

Given the inherent trade-off between task specificity and required model complexity when it comes to observation data modality and model output, motion capture has been selected as a convenient middle ground. It does not require a simulated environment as approaches that involve virtual reality do [Zhang et al., 2018, Dyrstad et al., 2018], obviates any issues that may come with having to learn motions in robot configuration space by operating in Cartesian coordinates,

[*]Robotics and Machine Perception Laboratory

[†]Institute of Electronics and Computer Science, Riga, Latvia

and does not necessitate the use of complex image processing layers as found in models with visual input modalities [Liu et al., 2018, Zhang et al., 2018]. The main drawbacks of this approach are that it requires motion capture equipment, which is more expensive than conventional video cameras, and Cartesian motion plans need an additional conversion step into joint space trajectories before it is possible to execute them on a real robot. Given the intended application of this system – programming robots to perform tasks in an industrial environment – the advantages of compact models, rapid training times and results that are possible to evaluate ahead of time were deemed to outweigh the drawbacks.

To help guide the development process and serve as a means of evaluation, a sample use case was selected – object throwing. While on the surface the task is almost entirely defined by elementary ballistics that can be programmed explicitly, it serves as a convenient benchmark for robot programming by way of demonstrations. The task was deemed sufficiently non-trivial when considered from the perspective of a Markov decision process or time-series model only given limited information about system state at any given time step, while also being suitable for intuitive evaluation by human observers due to its intuitively straightforward nature. Furthermore, the relationship between release position, velocity and target coordinates provides an obvious way to quantitatively evaluate model performance against training and validation data – extrapolated throw accuracy.

2 Preliminaries

In this article, an *agent* can be taken to mean the part of the system that acts based on state or observation information s in an *environment*, according to a *policy* π which is specified by a parametric *model* – a function $\pi_\theta(s)$ with parameters θ tuned in optimization and produces an output *action* a. In practice this means that references to the agent, model and policy are almost interchangeable in most contexts. A formalism common in both reinforcement and imitation learning contexts is the *Markov decision process* (MDP), formally given by [Attia and Dayan, 2018]

$$MDP = (S, A, T, R, I) \tag{1}$$

where S is the set of states s, A is the (formally discrete) set of actions a, $T : S \times A \to S$ is a state transition function that encapsulates the environment, $R : S \to \mathbb{R}$ is a reward function associated with each state and $I = p\left(s_0 \in S\right)$ represents the initial state distribution. In many imitation learning-related cases a reward function need not specified or considered. Moreover, if one is willing to break with the strict formal definition and give up the use of mathematical tools defined only on finite probability distributions, continuous action spaces can also be considered.

One important characteristic of the MDP is that it is history-agnostic – the future state distribution of the system is uniquely defined by its current state. While technically true for physical environments, it is often the case that instead of a complete state representation vector $\mathbf{s}$ we are instead operating on a more limited *observation* $\mathbf{o}$ (often interchangeably referred to as $\mathbf{s}$ for brevity), where each element o_i is given by some function $f_i(\mathbf{s})$. It is therefore possible that historical observations contain information about hidden system state variables even assuming the MDP formalism holds for the underlying environment. An example of this can be the case when individual observations contain only the current position of an object but not its derivatives [Zhang et al., 2018]. In such cases it may prove beneficial to break with the formalism further, by redefining the policy to operate on sequences of $k + 1$ previous states/observations

$$t, k, n \in \mathbb{N}, \pi_\theta : \{(s_n)_{t-k}^{t}\} \to A \tag{2}$$

which, by allowing for input sequences of variable length, may become a function defined on the entire known state history:

$$\pi_\theta : \{(s_n)_{1}^{t}\} \to A \tag{3}$$

These adjustments allow for the employment of sequence-to-sequence predictor architectures also studied in other areas of machine learning such as recurrent neural networks [Rumelhart et al., 1985] and transformers [Vaswani et al., 2017]. Finally, if continuous action spaces are permitted it is no great stretch to also consider cases where the action corresponds to a predicted future state that may be used for static motion planning or in a dynamic error-based controller

$$\pi_\theta : \{(s_n)_{1}^{t}\} \to S \tag{4}$$

which, when running the model on its own output, is equivalent to sequence building tasks encountered in domains such as text generation.

3 Related Work

In its simplest and most straightforward form – sometimes referred to as *behavioral cloning* – imitation learning can be reduced to a classification or regression task. Given a set of states and actions or state transitions produced by an unknown expert function, a model (*policy*) is trained to approximate this function. Even with very small parameter counts by modern standards, when combined with artificial neural networks this method has demonstrated some success as far back as the 1980s, provided a task with simple dynamics [Pomerleau, 1989].

However, it has since become apparent that pure behavioral cloning suffers from distribution shift – a phenomenon whereby the distribution of states visited by the policy diverges from that of the original training data set by way of incremental error, eventually leading to poor predictions by the model and irrecoverable deviation from the intended task. To improve the ability of policies to recover from this failure mode, various more complex approaches to the task have been proposed. One major direction of research has been the use of inherently robust, composable functions known as *dynamic motion primitives*, employing systems of differential equations and parametric models such as Gaussian basis functions to obviate the problem of distribution shift entirely – ensuring convergence towards the goal through the explicit dynamics of the system [Pastor et al., 2009]. Though showing promising results, it must be noted that the model templates used are quite domain-specific. Others have attempted to use interactive sampling with more general-purpose

4 Proposed approach

The overall approach proposed in this paper can be broken down into three main sections – the collection of observations in the physical environment (5.1), a pipeline for turning raw observation data into structured demonstrations (5.2) and neural network model implementations (5.3). Two general approaches to system evaluation and design feedback were employed. First, qualitative observations of the generated trajectories were made using spatial visualization in a virtual environment, followed by execution on simulated and real robots (5.4). Additionally, a series of quantitative metrics have been computed comparing system outputs with training and validation datasets (5.5).

5 Implementation

5.1 Data collection

5.2 Pre-processing, extraction of implicit control signals

bla bla bla

5.3 Models

bla bla bla

5.4 Visualization and execution

bla bla bla

5.5 Evaluation metrics

bla bla bla

6 Results

bla bla bla

7 Discussion

bla bla bla

References

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, pages 60–77. MIT press, 2018.

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 *Thirty-second AAAI conference on artificial intelligence*, 2018.

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5628–5635. IEEE, 2018.

Jonatan S Dyrstad, Elling Ruud Øye, Annette Stahl, and John Reidar Mathiassen. Teaching a robot to grasp real fish by imitation learning from a human supervisor in virtual reality. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7185–7192. IEEE, 2018.

YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1118–1125. IEEE, 2018.

Alexandre Attia and Sharone Dayan. Global overview of imitation learning. *arXiv preprint arXiv:1801.06503*, 2018.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA ARTIFICIAL INTELLIGENCE AND PSYCHOLOGY . . . , 1989.

Peter Pastor, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal. Learning and generalization of motor skills by learning from demonstration. In *2009 IEEE International Conference on Robotics and Automation*, pages 763–768. IEEE, 2009.