Title

Robot Learning from Interactions with Physics-realistic Environment: Constructing Big Task Platform for Training AI Agents

Author

Xie, Xu

Publication Date

2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Robot Learning from Interactions with Physics-realistic Environment: Constructing Big Task Platform for Training AI Agents

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Statistics

by

Xu Xie

2021

ABSTRACT OF THE DISSERTATION

Robot Learning from Interactions with Physics-realistic Environment:
Constructing Big Task Platform for Training AI Agents

by

Xu Xie
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2021
Professor Song-Chun Zhu, Chair

Robot learning from interactions is a crucial topic in the joint field of computer vision, robotics, and machine learning. Interactions are ubiquitous in daily life, concrete instances comprise object-object, robot-object, and robot-robot interactions. Learning from interactions to an intelligent robot system is important because it helps the robot to generate a sense of physics, meanwhile planning and acting reasonably. To achieve this purpose, one primary challenge that remains in the community is the absence of dataset that can be leveraged to study the diverse categories of interactions. To create those datasets, the interaction data should be realistic such that it reflects the underlying physical process. Further, we argue that learning interactions through simulations is a promising approach to synthesize and scale up diverse forms of interactions. This dissertation focuses on robot learning from interactions in Mixed Reality (MR) as well as leveraging the state-of-the-art physical simulation to construct virtual environments to afford *Big Tasks*. There are four major contributions along this pathway:

1. Robot learning object manipulation skills from human demonstrations. Instead of

directly learning from a robot-object manipulation dataset that is hard to generalize, we alternatively seek an approach to create a human-object manipulation dataset and let the robot learn from the demonstration. We claim that the key attribute of building such dataset embodies the realistic hand-object interaction that involves a setup that can faithfully capture the fine-grained raw motion signals. This leads us to develop a tactile glove system and collect informative spatial-temporal sensory data during hand manipulations. An event parsing pipeline is proposed upon the hand interactions that are transferable to the robot's end and learn the manipulation skill.

2. A virtual testbed to construct rich interactive tasks. The major limitation of collecting real-world interaction data can be summarized as three folds: i) a specific setup is needed to trace one form of interaction, ii) amount of efforts need to spend on data cleaning and labeling, and iii) a single dataset is not capable to capture different modalities of interactions at the same time. To overcome those issues, we propose and develop a virtual testbed, *VRGym* platform, for realistic human-robot interactive tasks (Big Tasks). In VRGym, the pipelines we developed are able to synthesize diverse photo-realistic 3D scenes that incorporate various forms of interactions through physics-based simulation. Given available rich interactions, we expect to grow a general-purpose agent from the interactive tasks and advance the research areas of robotics, machine learning as well as cognitive science.

3. Robot learning from imperfect demonstrations — *small data*. In the area of learning from demonstration, interacting with objects, one essential element is the creation of expert demonstrations. However, non-trivial efforts are needed when collecting those demonstrations and a large portion of them contains failure cases. We develop the demonstration setup for learning objects grasping skills upon VRGym platform with VR human interfaces. Human performers interact with the virtual scene by teleoperating the virtual robot arm. At the same time, the demonstration is evaluated through physics simulation such that even a perfect task plan may fail during the execution. Given the sparsity of demonstrations, we think the failed ones are valuable in addition to the perfect demonstration. This en-

lightens us to exploit the implicit characteristics of small data in the presence of imperfect demonstrations.

4. A game platform for large-scale social interactions. Social interactions are another important branch that goes beyond physical only interactions. To develop a general-purpose agent, it has to properly infer other agents motion or intentions and applies socially acceptable behaviors when interacting in the scene. Inspired by those facts, we leverage a popular computer game platform, Grand Theft Auto (GTA), to automatically construct fruitful realistic social interactions in the simulated urban scenarios. The city transportation system, including vehicles and pedestrians, can be fully controlled by the developed modding scripts. The GTA platform is a supplement to VRGym that extends robot learning from interactions to a larger scale. We utilize it to synthesize multi-vehicle driving scenarios and study the problem of trajectories prediction as to the basis of intentions inference. We highlight the safety aspect by predicting collision-free trajectories that accord with the social norm for vehicle driving.

The dissertation of Xu Xie is approved.

Hongjing Lu

Demetri Terzopoulos

Ying Nian Wu

Song-Chun Zhu, Committee Chair

University of California, Los Angeles

2021

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am fortunate enough to meet and collaborate with a collection of remarkable people:

My advisor Dr. Song-Chun Zhu, for his dedicated guidance and mentorship, opening a profound perspective of understanding computer vision, robotics and general AI, for teaching me to do frontier research, for providing me extensive opportunities in my career.

Dr. Demetri Terzopoulos, for his support in continuing projects and papers we collaborated with Upenn team together, special thanks to him spending time offer help on research works.

Dr. Hongjing Lu, for her careful guidance on cognitive science, especially the process and conduct code doing cognitive experiments.

Dr. Ying Nian Wu, for the knowledge and mind enjoyable moments, always shed lights on exploring the beautiful parts of doing research.

Dr. Yixin Zhu, my peer advisor, opening the door for me to learn every building blocks of doing research, as well as patiently guide me to cultivate the taste of science.

Feng Gao, Mark Edmonds, Hangxin Liu, Chi Zhang, Zhenliang Zhang, Yuxing Qiu, Baoxiong Jia, Ziyuan Jiao, Zeyu Zhang, Sirui Xie, Xiaojian Ma, Shuwen Qiu, my peer roommates in the office, going through endless papers and deadlines together.

Siyuan Huang, for offering the opportunity demonstrating my work in CVPR workshop.

Dr. Siyuan Qi and Dr. Yuanlu Xu, for your collaborations on the research as well as helpful guidance on my career.

Dr. Brandon Rothrock, a senior project leader in strengthening my engineering skills on problem solving.

Dr. Cheewei Wong, for accepting me to attempt intercourse research during my early stage of graduate study.

Dr. Wengang Zhou and Dr. Houqiang Li, my very first teachers introduce me to the field of computer vision.

My appreciation further extends to the entire VCLA labmates.

Finally, special thanks to my closest friends Ning Wang, Tong Xia, Yue Huang, Yidan Sun,

Sixiao Chen, Yaqi Zhang, Zhicheng Pan, Junhui Hu, Junzhe Zhang, Sijia Huang, Han Du as well as my parents, for accompanying with my Ph.D. journey.

Portions of this work were supported by DARPA XAI N66001-17-2-4029, ONR MURI N00014-16-1-2007, DARPA SIMPLEX N66001-15-C-4035.

VITA

2017–2021	Graduate Research Assistant, UCLA VCLA, Dr. Song-Chun Zhu
2020	Research Intern, Facebook Reality Labs, Dr. Weiguang Si
2018–2020	Research Engineer Intern, CARA Inc., Dr. Song-Chun Zhu
2019	Student Best Paper Award, SIGAI 2019
2018	Ph.D. Candidate in Statistics, UCLA
2015–2017	M.S. in Electrical and Computer Engineering, UCLA
2015	Young Fellow Scholarship Award, MS-IEEE, Microsoft Research Asia
2015	Honorable Title of Excellent Undergraduate Students, USTC
2014	Undergraduate Research Intern, Oxford University, UK
2013	The 1^{st} Place of Robot Game Competition, USTC

PUBLICATIONS

Congestion-aware Multi-agent Trajectory Prediction for Collision Avoidance. **X. Xie**, C. Zhang, Y. Zhu, Y. Wu, S.C. Zhu, *ICRA*, 2021

Reconstructing Functionally Equivalent and Interactive 3D Scenes via Panoptic Mapping

and Physical Common Sense. M. Han, Z. Zhang, Z. Jiao, **X. Xie**, Y. Zhu, H. Liu, S.C. Zhu, *ICRA*, 2021

Trajectory Prediction with Latent Space Energy-Based Model. B. Pang, T. Zhao, **X. Xie**, Y. Wu, *CVPR*, 2021

A Tale of Two Explanations: Enhancing Human Trust by Explaining Robot Behavior. M. Edmonds*, F. Gao*, H. Liu*, **X. Xie***, S. Qi, B. Rothrock, Y. Zhu, Y. Wu, H. Lu, S.C. Zhu, *Science Robotics*, 2019

Learning Virtual Grasp with Failed Demonstrations via Bayesian Inverse Reinforcement Learning. **X. Xie***, C. Li*, C. Zhang, Y. Zhu, S.C. Zhu, *IROS*, 2019

VRGym: A Virtual Testbed for Physical and Interactive AI. **X. Xie***, H. Liu*, Z. Zhang, Y. Qiu, F. Gao, S. Qi, Y. Zhu, S.C. Zhu, *ACM TURC*, 2019

VRKitchen: an Interactive 3D Environment for Learning Real Life Cooking Tasks. X. Gao, R. Gong, T. Shu, **X. Xie**, S. Wang, S.C. Zhu, *ICMLW*, 2019

HighFidelity Grasping in Virtual Reality using a Glovebased System. H. Liu*, Z. Zhang*, **X. Xie**, Y. Zhu, Y. Liu, Y. Wang, S.C. Zhu, *ICRA*, 2019

Intentionbased Behavioral Anomaly Detection. F. Hung*, **X. Xie***, A. Fuchs*, M. Walton, S. Qi, Y. Zhu, D.Lange, S.C. Zhu, *AAAIW*, 2019

Unsupervised Learning using Hierarchical Models for HandObject Interactions. **X. Xie***, H. Liu*, M. Edmonds, F. Gao, S. Qi, Y. Zhu, B. Rothrock, S.C. Zhu, *ICRA*, 2018

Interactive Robot Knowledge Patching using Augmented Reality. H. Liu*, Y. Zhang*, W. Si, **X. Xie**, Y. Zhu, S.C. Zhu, *ICRA*, 2018

Feeling the Force: Integrating Force and Pose for Fluent Discovery through Imitation Learning to Open Medicine Bottles. M. Edmonds*, F. Gao*, **X. Xie**, H. Liu, S. Qi, Y. Zhu, B. Rothrock, S.C. Zhu, *IROS*, 2017

A Glovebased System for Studying Hand-Object Manipulation via Joint Pose and Force Sensing. H. Liu*, **X. Xie***, M. Millar, M. Edmonds, F. Gao, Y. Zhu, V. Santos, B. Rothrock, S.C. Zhu, *IROS*, 2017

CHAPTER 1

Introduction

In the history of computer vision, Marr [Mar82] presents a computational perspective of addressing "what is where" when looking at the scene. This direction is further developed by the line of works [Zhu03, TZ02, BZ03, ZM07, WSF07, JQZ18, WGH19] pursuing a statistical framework for image modeling, parsing and generation. The milestones achieved by the community of computer vision continuously enlighten many close areas. Its concept, methodology and toolkit benefit various AI related research such as natural language processing and robotics learning. In the field of robotics learning, one key building block of a robot system is the perception module that receives and processes visual information. Computer vision algorithms and methods equipped on robots enables development of diverse applications in scene understanding and activity reasoning.

Another aspect can not be neglected for a robot system is the ability of sensing and processing multimodal information. In real world scenarios, different modalities other than visual information are equivalently valuable to be leveraged. For example, when robot grasps a object using its gripper, the tactile information may help decide the status of firm grasp. Besides, the driving trajectory of an vehicle may reflect its driving behaviors as well as intentions. On one hand, the robot develop the sense of entity fluents through interactive motions. It increases the degree of information that visual sensing may not bring. On the other hand, it is able to affect entity fluents hence induces the change of environment. This chain reaction indicates *interactions* is a crucial factor implicitly owned by a robotics system. Presenting with those evidences, we highlight learning from interactions is the crucial stage

to grow the true intelligence of a robot system. Learning through interactions, a robot gradually broadens its "intelligibility" towards *big tasks* achievement. In this dissertation, we are focusing on the topic of robot learning from realistic interactions to serve big tasks by answering the following questions:

1. **What would be the plausible data to realistically describe the robot-object interaction?** Manipulation is the most common category among robot-object interactions. A typical manipulation model composes multiple stages describing different object fluents and corresponding motion primitives. Learning a manipulation skill, simply acquiring visual information is not sufficient. Visual information alone lacks tactile motions to understand the interactions. For example, the robot gripper motion may be occluded sometimes when grabbing one object. Moreover, visual data may not reveal adequate knowledge to capture subtle motions. To resolve this, we propose to capture two crucial forms of data, *pose* and *force* that potentially convey rich information to describe object manipulations.

In literature, there are extensive study on tracing body pose [KB13], estimating force from vision [ZZC15, PKQ15] and soft-body simulation [WMZ13, ZZM13]. Instead of applying those existing methods, we develop a glove-based system that is capable of sensing raw signal during manipulation. For those devices, they have merits of carrying convenient and integrated solutions that can be natural for collection of ground truth hand data. Tactile gloves have been presented for various of robotics applications [DSD08] and are still active in research. Descriptions of our tactile glove design are detailed in the later chapter. The data collected from our tactile glove provides the robot with a demonstration resource to learn manipulation skills.

2. **What would be the suitable approach generalizing the manipulation events?** The purpose of modeling the manipulation events during objects interaction is to present the robot an interpretable spatial and temporal understanding for its own acting plan and execution. To the category of the most state-of-the-art action recognition algorithms, segmenting or parsing the events of manipulation sequences remains challenge. We investigate this

problem by studying one typical hand-object manipulation – *Open Bottles* that a number of movement primitives were involved: *grasp*, *push*, *twist* and *pull*. Given this setup, we propose an fully unsupervised learning method for manipulation event segmentation, recognition and parsing. To capture the temporal properties during manipulation, we present a temporal hierarchical structure using a grammar model — temporal And-Or graph (T-AOG).

During the object manipulations, we argue that force information is an indispensable element to better understanding manipulation motions. Aiming at force acquisition, we utilize our developed tactile glove to obtain force readings during manipulations. Data collected using the tactile glove reliably retrieves the contact forces that could be the limitation of visual system alone. Given the force exerted on the palm, we learn a "push-down" type of action as well as a set of motion primitives that form an action sequence. Leveraging the visually latent force information, our setup is able to *see* the forces during hand-object interactions. Another issue we have to tackle with is to transfer the raw signal (poses and forces) collected from the tactile glove to the robot that has different embodiments. To satisfy this purpose, we reconstruct the semantic meanings from manipulation motions to action events allowing the generalization of abstract knowledge. The temporal knowledge of the action sequence is represented by the learned T-AOG. It captures the hierarchical structure of temporal sequences. Our whole pipeline is developed fully unsupervisedly. And the action segmentation results compared to other baseline methods further prove the effectiveness of our model.

3. **What could be the promising direction to scale up diverse forms of physical realistic interactions pursuing big tasks?** For the routine methodology of robot learning from interactions, researchers are heavily focusing on collecting one source of interactive demonstrations, and let the robot learns an interactive model from the demonstrated dataset. However, this typical approach could be limited in the sense that retrieving multimodal information of interactions remain challenging, even a sophisticated hardware system may not cover every aspect of sensory information to study the related problems. In addition, there are always amount of efforts need to be paid on labeling the raw data. It becomes

especially tough while labeling multiple entity fluents that describing the interactive events. Facing those issues, we propose a insightful solution by developing a virtual reality testbed – *VRGym* which targets on providing physical realistic human and robot interactions by constructing simulated environments.

Different from the rapidly growing supervised learning using deep learning methods [HS06], such as object recognition [HZR15], robot grasping [LLS15, LHP16], learning from demonstrations [ACV09] and board games [SHM16, BS18, MSB17] often requires an intensive labeling process, task specific setup and lacks of explicit task representations, we in contrast highlight the task generalization when learning from interactions that an intelligent agent has to rapidly adapt to new tasks and grow the knowledge to achieve goals in a wide range of environments ("Big Tasks"). Utilizing the advance of game industry, assets of game contents, scenes and objects, are available for creating the virtual environments. In this period, mature physics based simulation and graphics rendering enable more realistic simulations than before. With the support of those characteristics, various forms of interactions are able to be performed using synthetic data during simulations. Availability of rich interactions enable the tasks development for the purpose of agent learning. Existing platforms such as [KMG17, SDL18, XZH18] are pushing towards this direction by making both research and engineering efforts. However, in terms of the development of interactions, the prior platforms lack involvement of human, another source of performing interactions for high level tasks. We argue that with human in the loop, social interactions including intention predictions and task collaborations could be performed. In this sense, developing the simulation environment for big tasks where virtual robots realistically interact with a human would assist the incremental learning towards robotics deployment in the home environment.

4. **What would be a practical approach to study large scale multi-agent interactions in simulated environments?** Large scale multi-agent interactions could be commonly noticed in the daily activities. One exemplar scenario is the urban traffic system where vehicles and pedestrians create fruitful interactions through commuting. As for

simulating those interactions in a physical realistic perspective, it requires the simulation environment maintains a complete traffic system such as schedule of traffic signal, path planning of commuting vehicles. Instead of scripting all detailed movements to support the simulated tasks, we leverage a game platform – "Grand Theft Auto (GTA)" that has existing game scenarios to study related topics. Majority of the game scenes are modeled on Los Angeles and its vicinity. The urban area has been carefully designed by gathering footage from field trips. For layout of road networks, it has been crafted with the help of Google Map projection.

In the multi-vehicle road driving, an autonomous vehicle needs to have a comprehensive understanding of the scenes its surrounded by. For example, when driving in an intersection, the maneuver a driver decides to conduct not only depends on traffic rules but is dynamically influenced by other vehicles in the same scene. The maneuver decision becomes particularly important in safety-critical cases. Among different choices of maneuver, each of which has its own consequence. Some of them may result in serious safety issues, such as crashing. In this sense, predicting multiple vehicles' future trajectories in socially interactive context is becoming remarkably crucial. In GTA platform, where authentic agents activities are supported by its physically realistic game engine, we develop a modding script hooked up with game process to generate vehicle interaction scenes. The script is capable of spawning the vehicles with different attributes meanwhile capturing and manipulating their real-time states and motions. To tackle with the issue of trajectory prediction under safety-critical scenarios, we propose an interactive learning approach that decouples the framework of "Sense-Learn-Reason-Predict". Using GTA platform, the evaluation of different methods can be directly simulated in the generated safety-critical scenarios that demonstrates its potential in conducting automobile research.

CHAPTER 2

Object Interactions: A Physical Realistic Approach for Manipulation Skill Learning

In this chapter, we discuss our approach for robot object interaction – learning manipulations skills. The way we study this problem is under the framework of Learning from Demonstration (LfD). Concretely, we let human perform the bottle opening task and the robot learns the corresponding skill from the demonstrated data. The data stream is captured by guaranteeing the physical realistic properties. The whole pipeline consists of the following major components:

- We develop a tactile glove based hardware system for raw data sensing during hand-object manipulation. The invisible force is incorporated with hand poses for event segmentation and parsing during fine-grained manipulation tasks.
- We propose an unsupervised learning framework that learns a temporal grammar model (T-AOG) for hand-object interactions. It composes of automatic clustering, event segmentation, labeling and grammar induction. This knowledge representation on actions from heterogeneous sensory data can be generalized to the robot.
- We transfer the learned temporal grammar model for manipulation actions onto a Baxter robot by solving a correspondence problem. The developed embodiment mapping function enable the robot to reason about its haptic measurements. Together with the temporal grammar model, they jointly form the manipulation model for robot to learn object manipulations.

2.1 Tactile Glove

(a) (b)

Figure 2.1: Prototype consisting of (a) 15 IMUs on the dorsum of the hand and (b) 6 integrated Velostat force sensor with 26 taxels on the palmar aspects of the hand.

In literature, efforts are made to design tactile glove systems to address limitations such as cost and portability. Our tactile glove, in particular, jointly senses pose and force information. Our thinking reflects the need for capturing the dynamics involved in fine manipulative actions rather than only considering the kinematics. To achieve this purpose, we create and demonstrate our developed tactile glove system that integrates both pose and force sensing over large hand areas. The sensors we adopt provides reliable spatial resolution as well as non-confined natural hand motions. The profile of our tactile glove is depicted in Fig. 2.1.

2.1.1 System Design

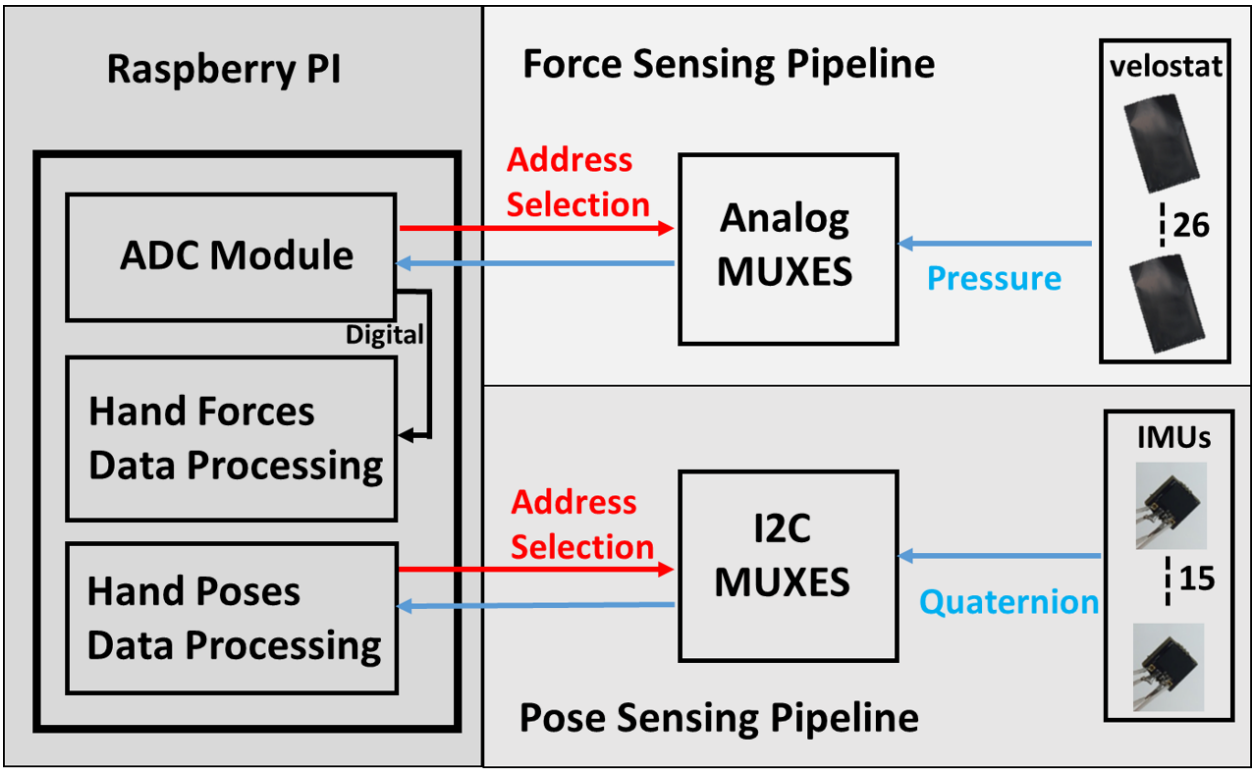


Figure 2.2: Overall system schematic.

Development of our tactile glove consists of both hardware and software implementation. The system integrates a glove and a processing unit (Raspberry Pi) for raw data acquisition. Two sensing networks are integrated to form the schematic which is presented in Fig. 2.2. The network of 15 IMUs measures the orientations of the hand palm and phalanx for elaborated hand pose reconstruction. The network of 6 Velostat force sensors are attached to hand palm and each finger to measure the contact forces.

For pose sensing pipeline, the pose estimation module is composed of 15 Bosch BNO055 9DoF IMUs. Among them, 1 IMU is mounted to the palm center of the glove, 12s are mounted to three phalanxes on four fingers and 2 IMUs are mounted to the distal and intermediate phalanges on the thumb. For each IMU, sensor fusion is performed yielding a global-frame quaternion measuring the orientation for each hand phalanx. All IMU sensors

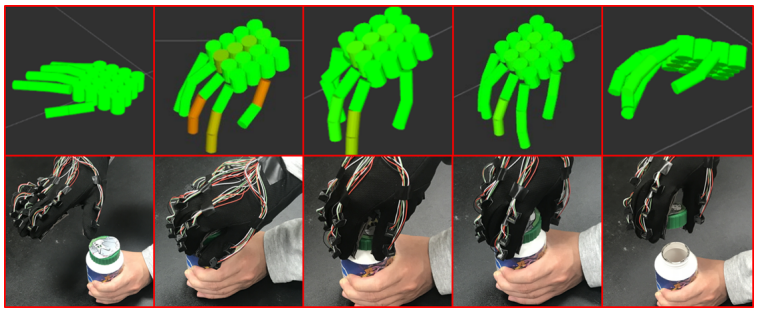
are networked over a pair of I^2C buses in star configuration, the I^2C multiplexer is mounted on the star center to connect I^2C bus interfaces available on a single Raspberry Pi board. The multiplexer acts as the controller to collect hand poses for the entire glove. In addition, the IMUs are fixed into small 3D-printed housings which are sewn into the glove fabric. Those wiring and setup increases the flexibility when performing hand motions.

For force sensing pipeline, the Velostat material is deployed to cover the surface of the glove. We layer small strips of Velostat between two outer conductive fabric with conductive thread stitched into it. The braided wire is then soldered to itself and form the circuit loop that hold the materials in place. Analog multiplexers are mounted and connected to Raspberry Pi's GPIO. The voltage signal are converted to force value by conducting calibrations.

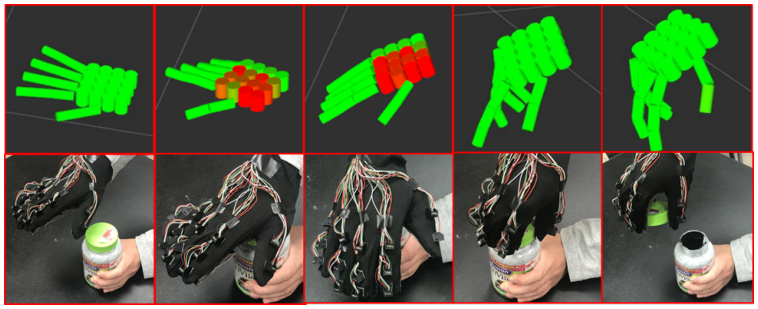
Prototype of our tactile glove is designed in Fig. 2.1. Functionality of the force sensing is realized by deploying five 2x1 customized Velostat force sensors on each finger / thumb that each of them detects the contact pressure. And a single 4x4 sensor covers the glove's palm area. Sensors arrangement are described in Fig. 2.1. The multiple Velostat sensors are connected in parallel via a multiplexer that access a each sensor periodically. The on-board Analog-to-Digital converter (ADC) is extended to integrate with a voltage divider to capture the voltage readings. This setup enables the force measurement distributed on the hand. On the other hand, pose sensing is provided by 15 IMUs on each phalanx and the palm. Those IMUs are controlled by multiplexers mounted on palm and connected to the single-board computer, Raspberry Pi, for pose acquisition. By setting up the software processing in ROS, the processed raw data can be accessed remotely and visualized in the real-time on desktop workstation.

2.1.2 Visualizing the Manipulative Actions

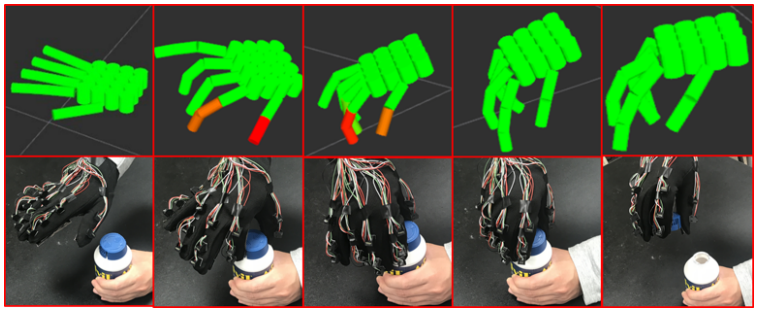
Utilizing the developed tactile glove system, we conduct a series of manipulative actions when opening three types of medicine bottles. Because of the lock mechanism among the bottles



(a) *Bottle 1*, regular twist to open



(b) *Bottle 2*. pressing the lid to open



(c) *Bottle 3*, pinching lock to open

Figure 2.3: Action sequences and visualizations of opening three types of bottles

are different, it requires particular actions to open the bottle lid. Fig. 2.3 demonstrates action sequences of opening three types of bottles when wearing the tactile glove. *Bottle 1* is the normal medicine bottle that can be opened by simply twisting the lid. *Bottle 2* has the safety lock and needs simultaneously pressing down and twisting to open the lid. *Bottle 3* requires the pinching to open the safety lock. Note that in the presence of the safety lock, some manipulative actions for *Bottle 2* and *Bottle 3* are hard to recognize from the visual sequence. Recovering the force exerted by the hand is the key to capture the

actions. By leveraging the ROS Unified Robot Description Format (URDF), we reconstruct the hand motions by creating a hand model with pre-defined structure and connected joints. Parameters such as phalanx lengths and palm dimensions are measured beforehand. The first row of Fig. 2.3a, 2.3b and 2.3c are sampled frames when visualizing the manipulative action sequences using the tactile glove for three types of bottles. The second row provides the actual sequences captured by RGB camera correspondingly.

Visualization results in the first row of Fig. 2.3 illustrate the additional force information representing different fine-grained manipulative actions when opening the bottles. For hand poses, Fig. 2.3b has palm pressing down while Fig. 2.3c is akin to a gripping gesture. According to the responses of force markers, different patterns indicate varying contact points on human hand. Comparing Fig. 2.3b and 2.3c, the former contains high responses around palm area, the later has only two responses concentrated on distal thumb and index finger. Given no existence of force information, motions of opening *Bottle 1* and *Bottle 3* are not able to be identified (Fig. 2.3a and 2.3c). This ability of tracing the visually unobservable forces proves the merit of our proposed design in studying object manipulations.

2.2 Unsupervised Hierarchical Modeling of Hand-Object Interactions

In literature, abundant approaches are proposed to model the actions. One important line of works propose recognizing the actions in different scenarios. In the community of computer vision, people demonstrate the capability of 3D poses estimation and action recognition by learning from RGB-D videos [WZZ13, WLW14, WNX14, WZZ17, QHW17] . However, those works concentrate on full-body action recognition instead of hand scale manipulations. Besides visual only information, [BF08, BSF09, ZZM13] estimate contact forces using the mass-spring system. These works rely on prior knowledge of properties such as physical and appearance of the manipulated objects. Our approach, instead, uses a tactile glove to

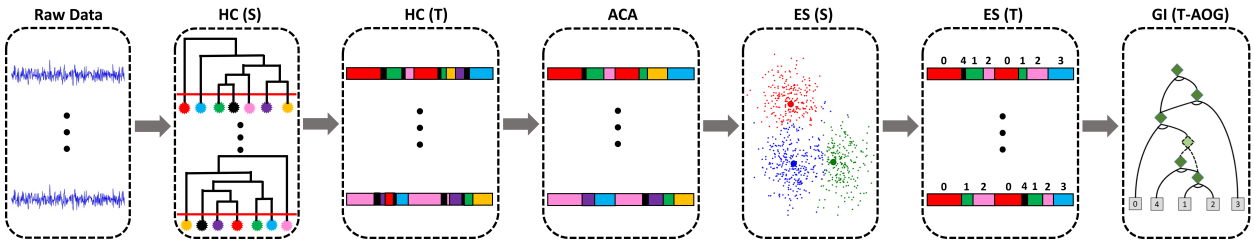


Figure 2.4: Unsupervised learning pipeline of hand-object motion recognition. After collecting the raw data using a tactile glove, a spatial (HC (S)) and temporal (HC (T)) hierarchical clustering is performed on both force and pose data. An aligned cluster analysis (ACA) is adopted to further reduce the noise. Event segmentation (ES (S) and ES (T)) is achieved by merging motion primitives based on the distance measured by DTAK. Finally, a grammar is induced (GI) based on the segmented events, forming a T-AOG.

measure the forces does not depend on such assumptions.

To achieve the modeling of hand-object interactions, we incorporate invisible force in addition to the hand pose based methods for motion segmentation and parsing. The framework we propose models noisy and heterogeneous sensory data that can be generalized to other hand-object interaction tasks. Our unsupervised learning pipeline, Fig. 2.4, incorporates spatial and temporal clustering, event segmentation and grammar induction. A temporal grammar model (T-AOG) is learned to significantly enhance the motion recognition results compared to naive clustering strategies.

2.2.1 Temporal Representation

In our proposed method, a structural grammar model *Temporal And-Or Graph (T-AOG)* [ZM07] is introduced to represent the temporal task structures. An AOG is a directed graph that describes the stochastic context free grammar (SCFG). This representation is hierarchical and compositional. The formal definition of the AOG is a five-tuple $G = (S, V, R, P, \Sigma)$, where S is a start symbol; V denotes a set of nodes in terms of non-terminal nodes V^{NT} and

terminal nodes V^T: $V = V^{NT} \cup V^T$; $R = \{r : \alpha \to \beta\}$ is a set of production rules represent the top-down sampling from a parent node α to its child nodes β; $P : p(r) = p(\beta|\alpha)$ is the probability for each production rule; Σ is the language synthesized by the grammar which includes all valid sentences.

For all *non-terminal* nodes in an AOG, they can be categorized into two types $V^{NT} = V^{AND} \cup V^{OR}$. An *And-node* is used to represent the compositional relations. It can be decomposed into multiple child nodes. An *Or-node* is used to represent alternative relations. It has multiple mutually exclusive child nodes. A *terminal* node represents an entity that is not further decomposed or has different configurations. A *parse graph (pg)* is an instance of the AOG that contains the decomposed And-nodes and one child node of the Or-nodes. Particularly, a temporal AOG (T-AOG) represents a set of all possible parse graph to execute a certain task. Semantically, the start node S represents an event category, ex. opening a medicine bottle. Terminal nodes V^T represents the set of motion primitives that a human or robot can execute. An And-node is decomposed into sub-tasks or child nodes' motion primitives. An Or-node represents alternatives to perform a sub-task. A *pg* for a task is a sub-graph of T-AOG that embodies the temporal structure of the task scenario.

Illustrated in Fig. 2.5, raw sensory data is segmented for semantic parsing. Pose and force sequences Γ are extracted on input sequence I during $[1, T]$. Motion primitive a_t is labeled on each frame and the whole motion sequence is denoted as $A = \{a_t\}$. Segmentation of the motion events is defined as $\mathcal{T} = \{\gamma_k\}, k = 1, \cdots, K$, where $\gamma_k = [t_k^1, t_k^2]$ represents the k-th interval where has the same motion label. Formally, we denote a_{γ_k} as the motion label for segment I_{γ_k}.

2.2.2 Learning Motion Primitives

Our unsupervised learning approach extract motion primitives through spatial and temporal clustering for hand-object interactions. For spatial clustering, we apply the agglomerative hierarchical clustering that merges the spatial wise similar features in a bottom-up manner.

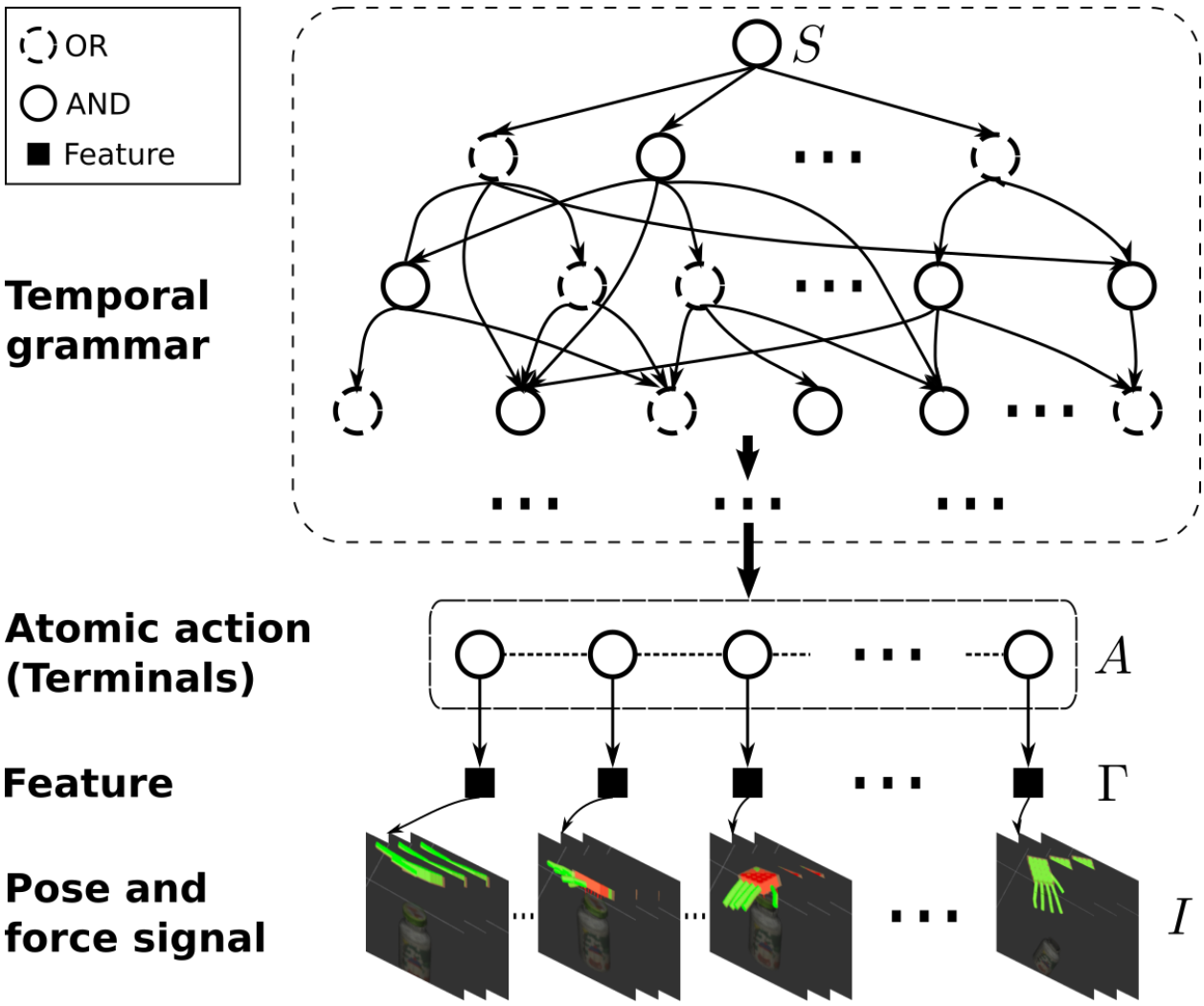


Figure 2.5: Illustration of the T-AOG. The T-AOG is a temporal grammar in which the terminal nodes are motion primitives of hand-object interactions.

This approach does not require the pre-defined cluster number for the clustering. Specifically, we adopt Wards agglomerative method and the merging rule is determined by the following formula in each iteration:

$$\begin{aligned}\triangle(A,B) = &\sum_{i\in A\cup B} ||\vec{x}_i - \vec{m}_{A\cup B}||^2 - \sum_{i\in A} ||\vec{x}_i - \vec{m}_A||^2 \\ &- \sum_{i\in B} ||\vec{x}_i - \vec{m}_B||^2 \\ =&\frac{n_A n_B}{n_A + n_B} ||\vec{m}_A - \vec{m}_B||^2,\end{aligned} \tag{2.1}$$

where A, B denote two separate clusters, m_A, m_B are the cluster centers, and $\triangle(A,B)$ is the cost to merge clusters A and B.

Spatially, the Ward agglomerative method hierarchically groups data by measuring the feature distance which lacks temporal consistency. Given natural temporal constraints on hand manipulation, we have to alleviate the issue by making the spatial clusters consistent in temporal domain. To achieve this, we adopt Aligned Cluster Analysis (ACA) [ZTH08] to reduce the temporal noise by measuring Dynamic Time Alignment Kernel (DTAK) [ZTH13]. ACA is extended upon kernel k-means clustering. It is formulated as an optimization problem on versatile energy which can be solved using coordinate descent:

$$s^* = \arg\min_s \mathbf{J}(\mathbf{G}, s) = \sum_{c=1}^{k}\sum_{i=1}^{m} g_{ci} D_c(\mathbf{X}_{[s_i, s_{i+1})}), \tag{2.2}$$

where $\mathbf{G}_{k\times n}^T \mathbf{1}_k = \mathbf{1}_n$ is the indicator matrix, $g_{ci} = 1$ when sample $\mathbf{X}_i$ is assigned to cluster c, and D_c measures the kernel distance between data point and cluster center. In practice, 2.2 could be solved through dynamic programming. It is equivalent with solving the following Bellman's equation [ZTH13]:

$$\mathbf{J}(v) = \min_{v - n_{\max} < i \le v} (\mathbf{J}(i-1) + \min_g \sum_{c=1}^{k} g_c D_\psi^2(\mathbf{X}_{[i,v]}, \dot{\mathbf{z}_c})), \tag{2.3}$$

where $D_\psi^2(\mathbf{X}_{[i,v]}, \dot{\mathbf{z}_c})$ is the squared kernel distance between segment $\mathbf{X}_{i,v}$ and cluster center c. $n_{\max}$ defines the maximum length of the segment.

2.2.3 Event Segmentation

To learn a descriptive temporal grammar model, semantic label for each segment is required. This issue can be easily solved when having a single motion sequence where each segment corresponds to one semantic meaning. Given multiple sequences performing the same task, we need to extract the semantic labels across those sequences.

Having two segmented sequences $\mathbf{X}_{[S_1,S_2...S_n]}$ and $\mathbf{Y}_{[S_1,S_2...S_m]}$, the semantic labels are assigned by merging different segments to clusters where each cluster has segments "close" in distance. The distance measure on two segments is following DTAK [ZTH13] criterion for similarity estimation:

$$\mathcal{D}(\mathbf{X}_{S_i}, \mathbf{Y}_{S_j}) = \tau_{\left[\mathbf{X}_{S_i}, \mathbf{Y}_{S_j}\right]}, \tag{2.4}$$

where $\mathbf{X}_{S_i}, \mathbf{Y}_{S_j}$ are candidate segments, $\tau_{[\mathbf{X}_{S_i}, \mathbf{Y}_{S_j}]}$ is the metric on similarity recursively calculated using DTAK. After grouping the segments, we apply k-means algorithm on groups of segments such that each cluster corresponds to one semantic label. The motion event labels are then be represented as the cluster IDs.

2.2.4 Grammar Induction and Inference

For each motion sequence, semantic labels are provided as the motion primitives to learn a T-AOG grammar model. The grammar is built up from a set of sequence instances, *pg*s, by maximizing a posterior probability [TPZ13]. Initially, a grammar is starting from the root node which is a Or-node, then it branches to the And-nodes. The And-node contains decomposition to represent instances. At the beginning, the grammar has small prior probability but the maximal likelihood of the training samples. Then intermediate non-terminal nodes are generated in a bottom-up way to optimize the posterior probability. A greedy search strategy is applied to identify good grammar fragments that is added into the grammar structure.

Given the learned grammar $\mathcal{G}$, our goal is to seek for the optimal motion label sequence A^* that best explains the observation. Denote the manipulating pose and force sequence as Γ, the objective is to maximize the posterior probability by:

$$A^* = \arg\max_A p(A|\Gamma, \mathcal{G}) = \arg\max_A p(\Gamma|A)p(A|\mathcal{G}), \tag{2.5}$$

where $p(\Gamma|A)$ is the likelihood measure given the observed motion label sequence, $p(A|\mathcal{G})$ is the parsing probability of the parse graph given grammar $\mathcal{G}$. The likelihood is further expanded as:

$$p(\Gamma|A) = \prod_{k=1}^{K} p(\Gamma_{\gamma_k}|a_{\gamma_k}) = \prod_{k=1}^{K} \prod_{t=t_k^1}^{t_k^2} p(\Gamma_t|a_{\gamma_k}), \tag{2.6}$$

where k is the segment index, γ_k is the kth segment. We fit a Gaussian distribution to the learned clusters on training samples. The prior $p(A|\mathcal{G})$ in 2.5 is calculated as the Viterbi parsing likelihood indicating the best parse of the terminals.

Since it is computational intractable to attain the optimal label sequence, we propose a Gibbs sampling scheme that can be done in terms of two steps: i) acquire the initial labeling on motion sequences through our unsupervised learning pipeline, and ii) the labels are refined pursuing 2.5 through Gibbs sampling with simulated annealing. The semantic label is assigned to a motion sequence according to 2.5 as:

$$a'_{\gamma_k} \sim p(\Gamma_{\gamma_k}|a_{\gamma_k})p(A'|\mathcal{G}), \tag{2.7}$$

where a'_{γ_k} is the refined label of segment Γ_{γ_k}, A' is the refined label sequence by updating the label a_{γ_k} to a'_{γ_k}. The simulated annealing is applied by tuning the log probability with a temperature parameter. The temperature is decreasing through the sampling process until the labeling achieves convergence.

2.2.5 Performance Evaluation

Our metric to this motion sequence labeling problem is evaluated by measuring the frame-wise accuracy. The ground truth segmentation is manually labeled by visualizing the raw data sequence in ROS RVIZ. Three types features are considered for performance comparison: i) the hand *pose* feature in terms of Euler angles, ii) the hand *force* feature in terms of force magnitude, and iii) the *force vector* which combines both pose and force feature. Hyper parameters fixed for comparison are: i) cluster number $k = 5$ and maximum segment length $n_{max} = 200$.

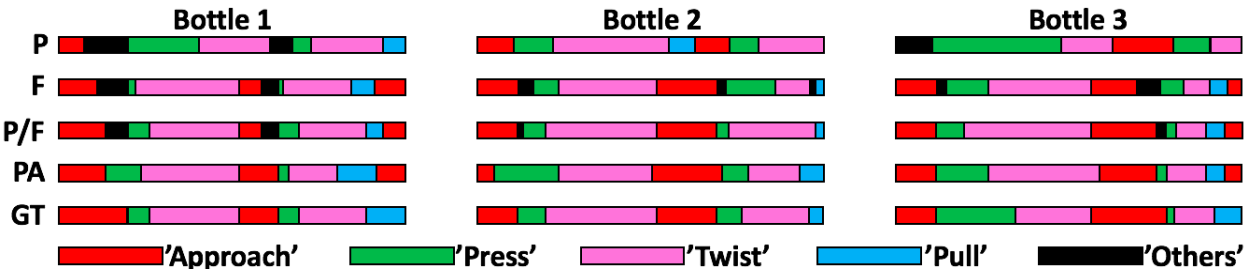


Figure 2.6: Qualitative evaluation. Event segmentation and recognition of opening Bottle 1, 2, and 3, from left to right, respectively. P denotes *pose only feature*, F *force only feature*, P/F *force vector feature*, PA *with parsing*, and GT *ground truth*. Each segment represents one type of motion primitive which color is determined by the ground truth sequence.

The segmenting results on motion primitives of opening *Bottle 1, 2, 3* are demonstrated in Fig. 2.6. The corresponding quantitative results are presented in Table 2.1. According to the rates of accuracy, results using hand pose only has the worst performance than the other two. It proves our argument that force information is an indispensable element in modeling hand-object interactions. Besides, feature incorporates both hand pose and force outperforms the single source of feature.

In addition, we perform semantic parsing using the learned T-AOG and compare the results with the clustering only ones. Qualitatively, as depicted in Fig. 2.6, the T-AOG parsing method recovers the noisy and mislabeled segments and obtains more coherent parsing re-

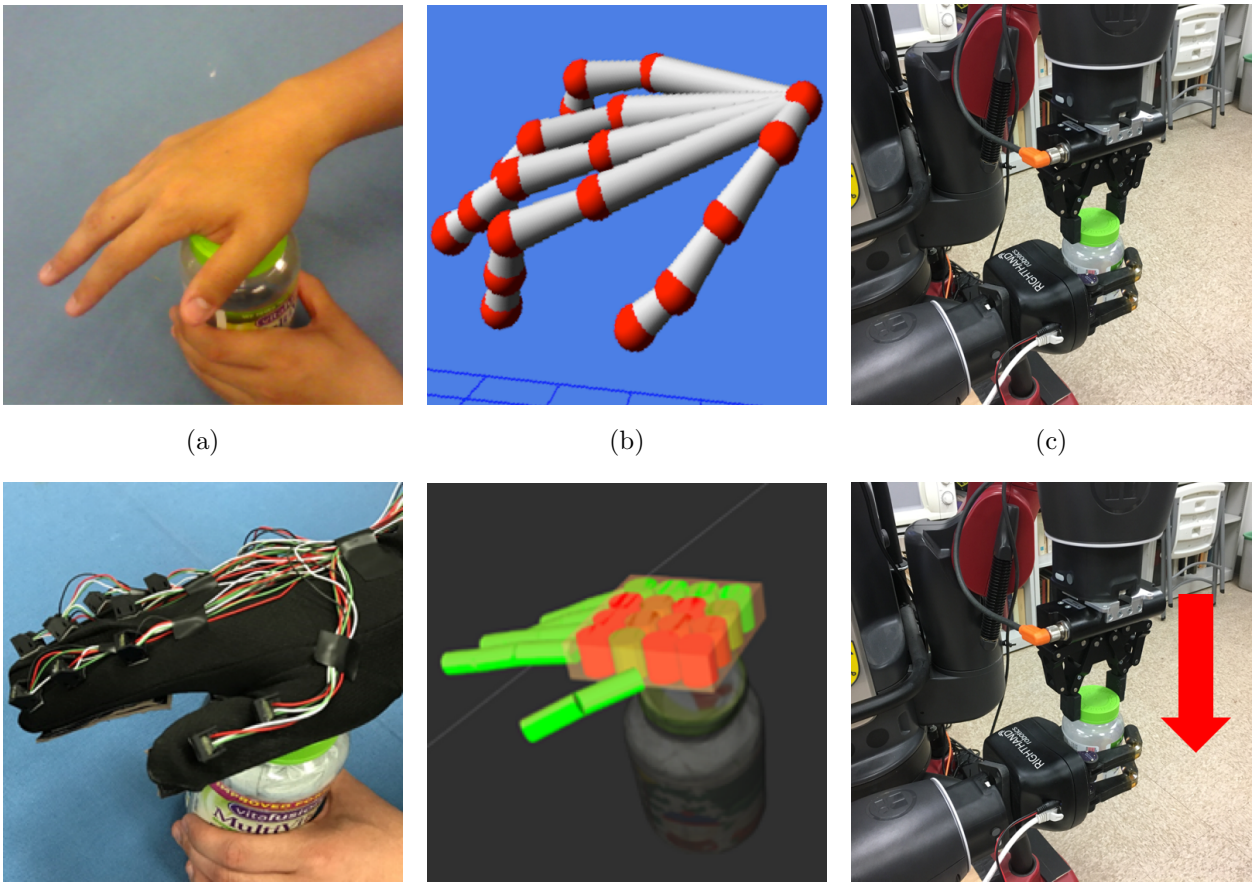
Table 2.1: Quantitative Evaluation. With clustering only, we use the hand pose, in the forms of Euler angles of each phalanx; hand force, as scalars; and the combination of pose and force as force vectors as feature inputs. Including force factor yields higher correspondence with ground truth sequence. Parsing the events with T-AOG on top of the clustering, the performance improves significantly.

	Clustering only			With T-AOG
	Pose only	Force only	Pose and Force	Pose and Force
Bottle 1	55.3%	67.5%	70.3%	**78.6%**
Bottle 2	62.0%	70.9%	76.2%	**82.5%**
Bottle 3	54.1%	71.1%	72.9%	**78.5%**

sults. Quantitatively, shown in Table 2.1, the performance enhancement of applying T-AOG has been remarkably improved. It further verifies the effectiveness of learning a temporal grammar model for motion events segmenting or parsing than clustering only methods.

2.3 Robot Learning Manipulation Skill from Demonstrations

In the field of robot Learning from demonstration (LfD) [ACV09], one primary purpose is to exploit the meaningful knowledge in the demonstrations for skill learning. In particular, Imitation Learning (IL) becomes one popular stream to achieve this purpose. In general, it can be realized by two frameworks: i) *Behavior cloning*. The learner directly mimics the performer's demonstrated behaviors by pursuing a supervised manner [HD94, MGH09, PJK16, XSX16, SGR17], and ii) *Inverse reinforcement learning* [AN04, RA07, ZMB08]. Our work is categorized to behavior cloning that can handle the non-Markovian cases. [HLD16]



(a) (b) (c)

(d) (e) (f)

Figure 2.7: Given a RGB-D-based image sequence (a), although we can infer the skeleton of hand using vision-based methods (b), such knowledge cannot be easily transferred to a robot to open a medicine bottle (c), due to the lack of force sensing during human demonstrations. We utilize the tactile glove (d) and reconstruct both forces and poses from human demonstrations (e), enabling robot to directly observe forces used in demonstrations so that the robot can successfully open a medicine bottle (f).

adopts imitation learning approach by using a data glove to open standard bottles. However, this work is too simplified to cope with the manipulation task – opening medicine bottles. Opening a medicine bottle requires complex manipulations rather than rotating only. [SSS17] uncovers haptic components from teleoperation. Compared to the existing works, we collect

the data using tactile glove which induce more natural and diverse demonstrations. Moreover, instead of applying recurrent neural networks (RNNs) to model the task dynamics, we adopt an explicit grammar model for motion generation which is capable of planning upon long term temporal dependencies.

For our problem definition, a real robot, Baxter, learns from human opening the medicine bottles. Since the robot has different embodiment with human (Fig. 2.7), transfer the learned knowledge from the human end would be the key to solve this issue. Following the unsupervised approach of learning motion events segmentation, we leverage the temporal grammar model T-AOG and treat it as the source of *top-down* process to guide the robot manipulations of an unseen medicine bottle. In addition, a *bottom-up* process is learned by modeling the raw signal from the robot end-effector. It executes the task by encoding the transition between task pre- and post-conditions. We incorporate both processes to learn a robot manipulation model for opening the medicine bottles.

2.3.1 System Setup and Architecture

Our robot system is a dual-armed 7-DoF Baxter robot invented by Rethink Robotics. To enable the manipulation ability, we equip a ReFlex TackkTile gripper on the right wrist (grasping bottles), and a Robotiq S85 parallel gripper on the left (operating bottle lids). Localization and tracking of object (bottles) are realized by utilizing Simtrack [PK15] with a Kinect sensor. The backend system runs on ROS and the robot arm motion planning is realized by using *MoveIt!* library. To achieve object grasping, we generate the grasping poses by implementing a geometry based grasping planner.

The system architecture for robot learning the object manipulation skills consists three major components which are depicted in Fig. 2.8: i) Learning: the learning phase contains *top-down* and *bottom-up* processes. The *top-down* process builds a temporal representation T-AOG to incorporate the valid motion sequences. The *bottom-up* process learns three neural networks from the raw sensory data. ii) Inference: it is achieved by two stream of parsing.

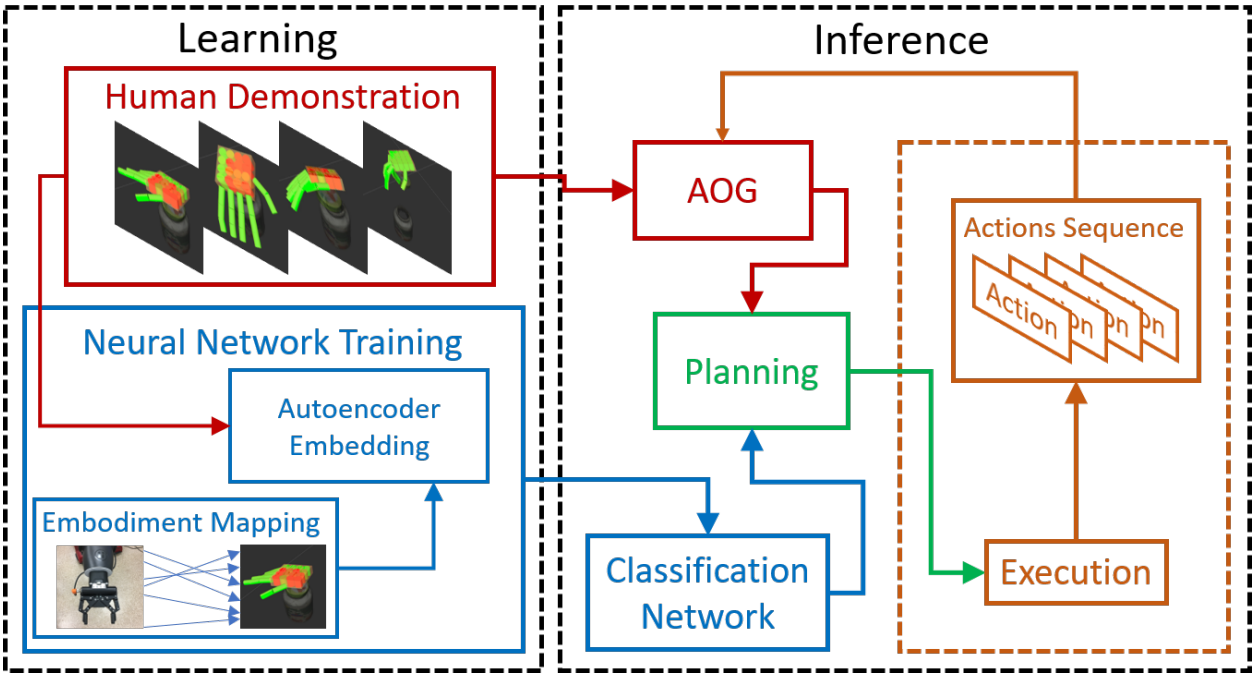


Figure 2.8: System architecture. Blue: action planning using fluents as a bottom-up process. Red: action planning using AOG as a top-down process. Green: action planning. Brown: robot execution.

The top-down term is calculated by Earley parser [Ear70], while the bottom-up term is computed through embodiment mapping and motion classification network. iii) Execution: The robot plans the next atomic action by combining top-down and bottom-up results and affords execution.

2.3.2 Top-down Planning Using AOG

We denote the *top-down* term as $p(pg_{k+1}|pg_k)$ that plans the next atomic action given the history of action sequence. It models the long temporal dependencies between all previous atomic actions and the next one. The planning is fulfilled by adopting Earley parser and computing the parsing likelihood.

Given the learned T-AOG $\mathcal{G}$ on human motion sequences through grammar induction.

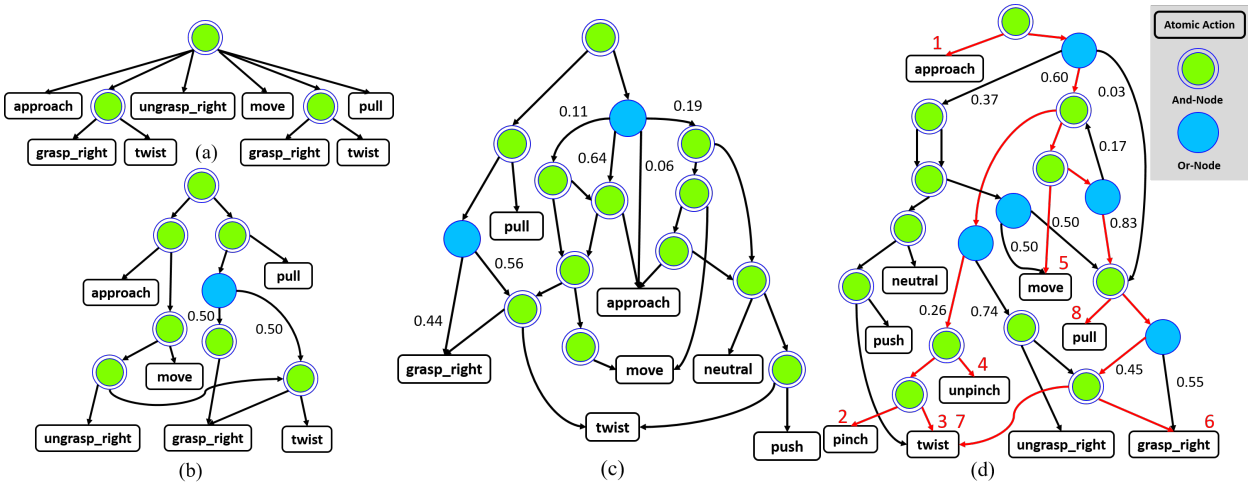


Figure 2.9: T-AOG induced from human demonstrations using 1 example (a), 5 examples (b), 36 examples (c), and 65 examples (d). (d) also shows an valid parse graph in an AOG, highlighted in red. Numbers indicate temporal ordering of atomic actions.

Examples for opening the medicine bottles are illustrated in Fig. 2.9. A grammatically complete parse graph $s = (a_0, \ldots, a_K)$, the parsing likelihood is equivalent with Viterbi likelihood which can be noted as $p(s)$. As for an incomplete parse graph, $pg_k = (a_0, \ldots, a_k)$, the parsing likelihood can be computed by summing over all possible grammatically complete parse graph starting from pg_k:

$$p(pg_k) = \sum_{s \in \mathcal{G}, s_k = pg_k} p(s). \tag{2.8}$$

The top-down term $p(pg_{k+1}|pg_k)$ can be calculated through Bayes' rule using the Earley parsing likelihood. For the purpose of atomic action planning, the top-down term encodes long term temporal context based on T-AOG.

2.3.3 Bottom-up Planning Using Fluents

Encoding distributions of raw motion signals and corresponding action labels to a fluent space, the changes in fluent space correspond to fluent changes. Given a fluent function, it maps the scene configuration, s_k, to a scalar, $f(s_k) \mapsto \mathbb{R}$. A fluent change is then represented by a transition between two scene configurations, $\nabla f(s_i, s_j) = f(s_j) - f(s_i)$. For the action space, given atomic action performed at step k as a_k, scene configurations of the pre-condition s_k and the pos-condition s_{k+1}, the atomic action can be formulated by the fluent change as $\nabla f^{a_k} = \{\nabla f_i(s_k, s_{k+1}), \quad i = 1...n\}$. Towards the atomic action planning, we denote the *bottom-up* term as $p(a_{k+1}|a_k, f_k)$ which plans the next atomic action on the condition of the current action and observed fluent. Given the finite set of atomic actions, we convert this planning task to a classification problem and realize it by learning a neural network to select next action with highest probability.

To construct the fluent space, we use an auto-encoder to convert the scene configuration into a low dimensional fluent as shown in Fig. 2.10(a). The fluent space is represented as a 8 dimensional embedding and is then reconstructed to get the full feature representation. The objective is to minimizes the L2 loss between the original and the constructed features:

$$l(\theta; \mathbf{x}^h) = \frac{1}{N} \sum_{i=1}^{N} (x_i^h - \psi(x_i^h; \theta))^2, \tag{2.9}$$

where x_i^h denotes one demonstration sample made by human and $\psi(x_i; \theta)$ denotes the auto-encoder construction.

Planning of the bottom-up term is a multi-class classifier that outputs 1 atomic action selected from 13 action labels. In Fig. 2.10(b), the classifier takes the embedding encoded in the fluent space as input and one-hot encoding of the current atomic action. To approximate the classifying probability, a softmax layer is applied to convert the prediction in terms of the categorical distribution. The model is implemented in terms of fully connected layers with sigmoid activation. It is trained by minimizing the cross-entropy loss that is effective

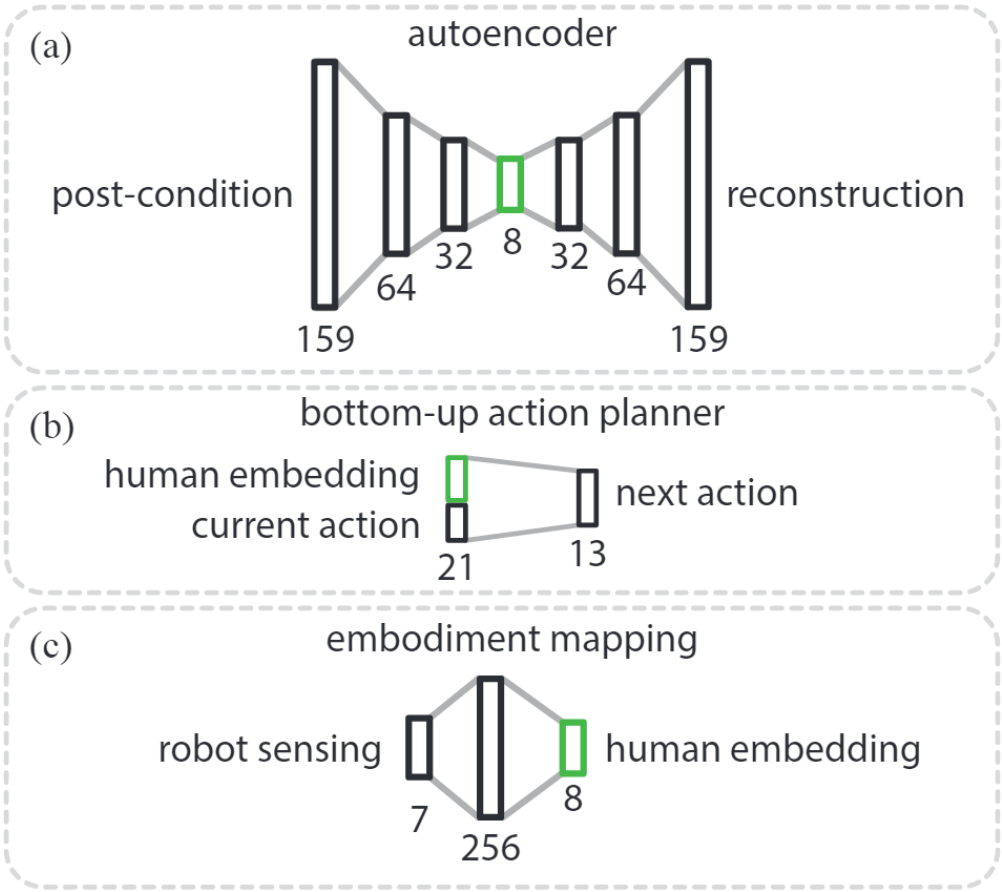


Figure 2.10: (a) Autoencoder to project human demonstration into low-dimensional subspace. (b) Classifier used to plan the next action using a low-dimensional embedding of human tactile feedback. (c) Embodiment mapping used to map robot states to equivalent human demonstration states. Each rectangle represents a vector, and each corresponding number is the length of the vector. The green rectangle represents the low-dimensional human embedding vector.

in coping with classification problems. Combining the top-down temporal constrains from grammar parsing, the bottom-up planning incorporates raw motion signals to guide the robot's actual manipulations.

In addition, we bridge state connections during manipulations between human and the robot by embodiment mapping. The goal is to approximate a function $s_h = \hat{\phi}(s_r)$ that s_h

represents the human state in the demonstration while s_r represents the robot state during execution. Depicted in Fig. 2.10(c), we create a neural network to map robot haptic sensory signal from the leaned embedding from the human demonstration. we use only a small fraction of training data points that sampled from the learned T-AOG to guide the robot executions. It ensures only successful robot states are considered to map to the successful demonstrated states. Loss function for the embodiment mapping is specified as:

$$l(\theta; \mathbf{x}^h, \mathbf{x}^r) = \frac{1}{N} \sum_{i=1}^{N} (\phi(x_i^h) - \hat{\phi}(x_i^r; \theta))^2, \tag{2.10}$$

where $\mathbf{x}^h$ and $\mathbf{x}^r$ are human and robot states respectively. ϕ represents the embedding in fluent space, and $\hat{\phi}$ denotes the mapping function. During execution, the robot maps its state to the equivalent counterpart of human's. Then it uses the human state to plan the next atomic action by using the bottom-up planner.

2.3.4 Robot Performance on Execution

To evaluate the task performance of opening medicine bottles, an manipulative sequence is treated as successful if the robot opens the bottle lid at the final step, otherwise is failure. More than 300 trials are conducted by opening all types of bottles. In some cases, there are multiple ways to open a medicine bottle. Bottles have no safety lock, for example, can be opened either by performing action sequence "push and twist" or "pinch and twist". To handle those cases, we consider them equivalently and propose two levels evaluation criteria: i) End results only. See if an action sequence can open a bottle successfully, ii) Efficiency. See if the successful action sequence can be performed efficiently. The execution results can be categorized as follow:

leftmirgin=* Success, where the robot successfully executed an action sequence that exactly matches one of the human demonstrations;

leftmiirgiin=* Success, while using at least one extra or wrong action;

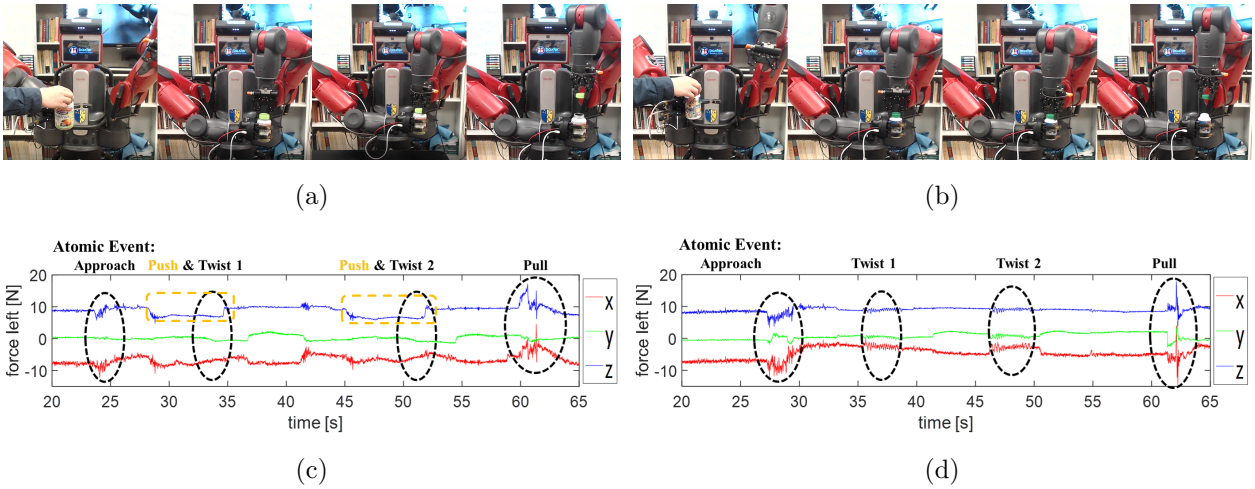


Figure 2.11: (a) Robot opening bottle 3, showing actions *approach*, *push*, *twist*, and *pull* from left to right. (b) Robot opening bottle 5, showing actions *approach*, *grasp*, *twist*, and *pull*. Force-torque sensor readings while opening bottle 3 (c) and bottle 5 (d), showing clear, distinguishable differences from raw sensor data.

leftmiiirgiiin=* Failure due to using the wrong action sequence;

leftmivrgivn=* Failure due to execution failure (e.g. low motor execution accuracy or grasping failure).

The qualitative results can be found in Fig. 2.11 where Fig. 2.11(a) shows the action sequence having "push" action, and Fig. 2.11(b) shows the action sequence without "push" action. Three sets of quantitative experiment results are summarized. Table 2.2 illustrates the execution results only considering *top-down* planning where the atomic action sequence is sampled from T-AOG. It captures the underlying ordering of the execution but ignoring the haptics information during the manipulation. Table 2.3 illustrates the execution results only using *bottom-up* planning where the atomic action sequence is sequentially predicted by neural networks. It incorporates the haptic signal by robot torque sensors while missing the long temporal dependencies when conducting the planning. Table 2.4 presents the results integrating both *top-down* and *bottom-up* planning. This yields a large performance

gain compared to either alternatives (Table 2.2 and Table 2.3). The rate of success cases are drastically improved while the failure rate with wrongly planned action sequence drops significantly. The results solidly proves our proposed methodology for object manipulations.

Overall, we systematically study the object interactions through the framework of learning from demonstrations. For human data collection, we develop a glove system to collect fine-grained tactile information to overcome the existing issue on studying object interactions. Modeling of atomic actions are conducted in an fully unsupervised manner and a temporal grammar structure T-AOG is learned for manipulation event parsing and inference. For robot execution, we develop two streams of planning by integrating both *top-down* and *bottom-up* information. Our task performance on opening medicine bottles further verify the effectiveness of our proposed way of learning object manipulation skills.

Table 2.2: Baseline 1, top-down only planning

Evaluation	**bot. 1**	**bot. 2**	**bot. 3**	**bot. 4**	**bot. 5**
Success	8.7%	5.6%	4.4%	8.7%	26.1%
Success (extra/wrong)	21.7%	5.6%	34.8%	47.8%	39.1%
Failure (action)	69.6%	77.7%	60.8%	34.8%	30.4%
Failure (execution)	0%	11.1%	0%	8.7%	4.4%

Table 2.3: Baseline 2, bottom-up only planning

Evaluation	**bot. 1**	**bot. 2**	**bot. 3**	**bot. 4**	**bot. 5**
Success	4.4%	0%	4.4%	0%	4.4%
Success (extra/wrong)	13%	11.8%	30.4%	42.9%	17.4%
Failure (action)	82.6%	76.4%	65.2%	57.1%	78.2%
Failure (execution)	0%	11.8%	0%	0%	0%

Table 2.4: Proposed, top-down and bottom-up planning

Evaluation	**bot. 1**	**bot. 2**	**bot. 3**	**bot. 4**	**bot. 5**
Success	8.7%	17.6%	17.4%	20%	60.9%
Success (extra/wrong)	52.2%	17.6%	65.2%	73.3%	17.4%
Failure (action)	39.1%	64.8%	13%	6.7%	21.7%
Failure (execution)	0%	0%	4.4%	0%	0%

CHAPTER 3

VRGym: Virtual Testbed for Big Tasks Construction

Generally speaking, robot learning from interactions requires a creation of dataset that dedicates at capturing the signals describe the interactive process. In practice, consummately building a single dataset to fulfill the purpose is far from promising. In real world interactions, there could be multimodal data sources that are functioning together on one interactive task. A sophisticated hardware setup, e.g. tactile glove, needs to be developed to faithfully captures all aspects of sensory information. Additional efforts and cost have to be paid to support the ground truth data labeling. Moreover, for real world tasks, diverse form of interactions are commonly co-existing. For example, when human is collaborating with a robot for furniture assembling, the robot not only has to interact with furniture parts for correct object manipulation, but has to properly collaborate with the human on the task planning to generate reasonable interacting behaviors. Hence, the traditional methodology is becoming reluctant to study robot interactive tasks.

On the other hand, we propose to construct a *testbed* that affords *big tasks* development by leveraging virtual environments. Inside virtual environments, entities fluents are fully controlled by the physics based simulations. In addition, abundant forms of interactions are able to be defined and synthesized in a automatic manner. As for the solution, we deliver a physical and photo realistic virtual testbed — *VRGym* for big tasks construction. VRGym provides functionality to simulate various kinds of interactive tasks for robot learning. There are three direct advantages VRGym owns over the traditional methodology: i) labeling free with automatic ground truth generation, ii) generalizable tasks get rid of task specific con-

figurations, iii) maintaining explicit representation and structure to handle task variations. Compared to other existing simulation platforms for robot learning, we develop VR interfaces for human to interact with the virtual scenes directly. In this sense, VRGym platform is not simply treated as a simulation platform, but a human-robot co-existing environment that has more potential in synthesizing interactive tasks.

3.1 System Characteristics

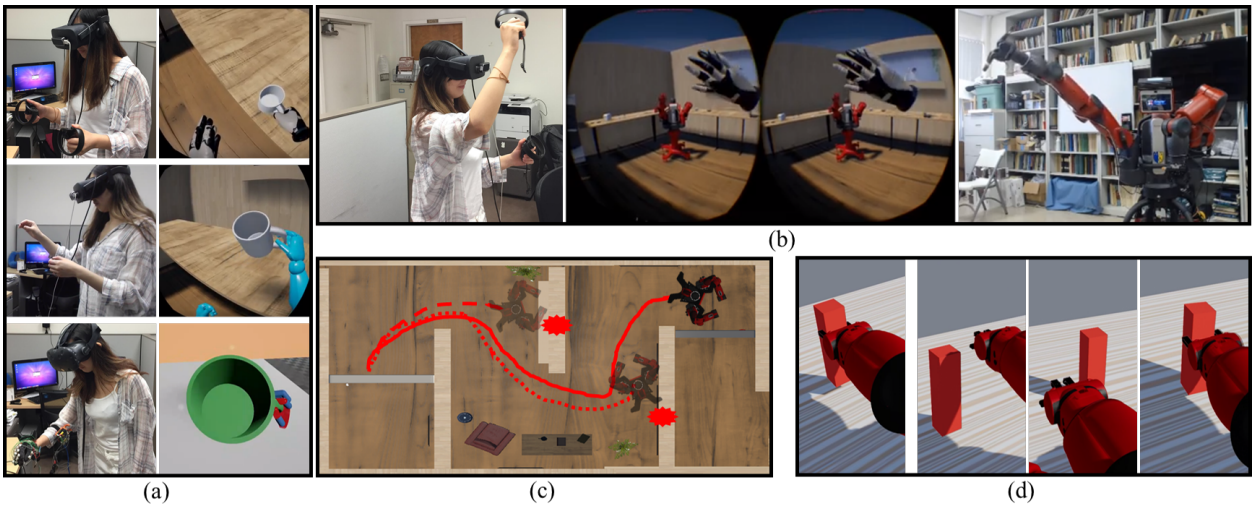


Figure 3.1: (a) VRGym integrates three types of input devices, providing human manipulation in an increasing resolution using Oculus Touch, LeapMotion, and a data glove, from top to bottom. (b) The VRGym-ROS bridge allows physical human/robot agent meet virtual agents inside a virtual world, providing the capability of social interactions. (c) The training of the robot navigation using RL inside VRGym. The robot successfully navigates to the goal without collisions after about 10,000 episodes. (d) The learning of object manipulation using human demonstrations (leftmost) and IRL (right three) inside VRGym.

Especially, VRGym is proposed to support the need of robot learning with fruitful interactions combining the advancement of VR technique. We target to address three critical issues: i) What is the optimal approach to reflect human embodiment in VR? ii) What

aspects can well-developed algorithms or models be adopted in the simulation? iii) What level of interactions can VR based simulation afford? To study those questions, VRGym is pushing the frontier of the state-of-the-art simulators in terms of the following perspectives:

Fine-grained human embodiment Human embodiment in the loop of simulation is a non-trivial task. The prevalent simulation platforms support limited human control. the embodiment is usually achieved by scripting or defining a finite motion set for remote control. In VRGym, whereas, we integrate hardware setups to enable multimodal human inputs. Those setups are supplementary to traditional VR input devices. Our setup embodies the detailed avatar motions by providing a whole body sensing. During interactions, the simulation reflects its physical effect on human including both body and hand poses. Fig. 3.1(a) demonstrates different resolution levels of manipulations realized in VRGym.

High compatibility with existing robotic systems VRGym is developed to be compatible with the most popular robotics framework, Robot Operating System (ROS). The data sharing routine between ROS and the virtual environment is achieved by developing an efficient bi-directional communication interface. Our *VRGym-ROS* bridge enables robotics applications that can be performed in the virtual environment. Fig. 3.1(b) plots an instance of person and robot are interacting in the VRGym. Practically, all ROS compatible packages or resources can be leveraged by VRGym will little effort for diverse research purposes, such as model learning, evaluations and benchmarking.

Multiple grandularities of interactions Given the human embodiment and ROS integration developed in VRGym, fruitful interactions are able to be synthesized for learning purposes with different grandularities of resolutions. It supports the forms of interactions as only providing perception information to sophisticated ones as learning complex robot grasping skills. Fig. 3.1(c) depicts how an virtual robot learns a navigation policy through visual based Reinforcement Learning (RL), and Fig. 3.1(d) presents a setup of robot Learning from Demonstration (LfD). Grasping policy is obtained utilizing Inverse Reinforcement Learning (IRL).

3.2 System Architecture

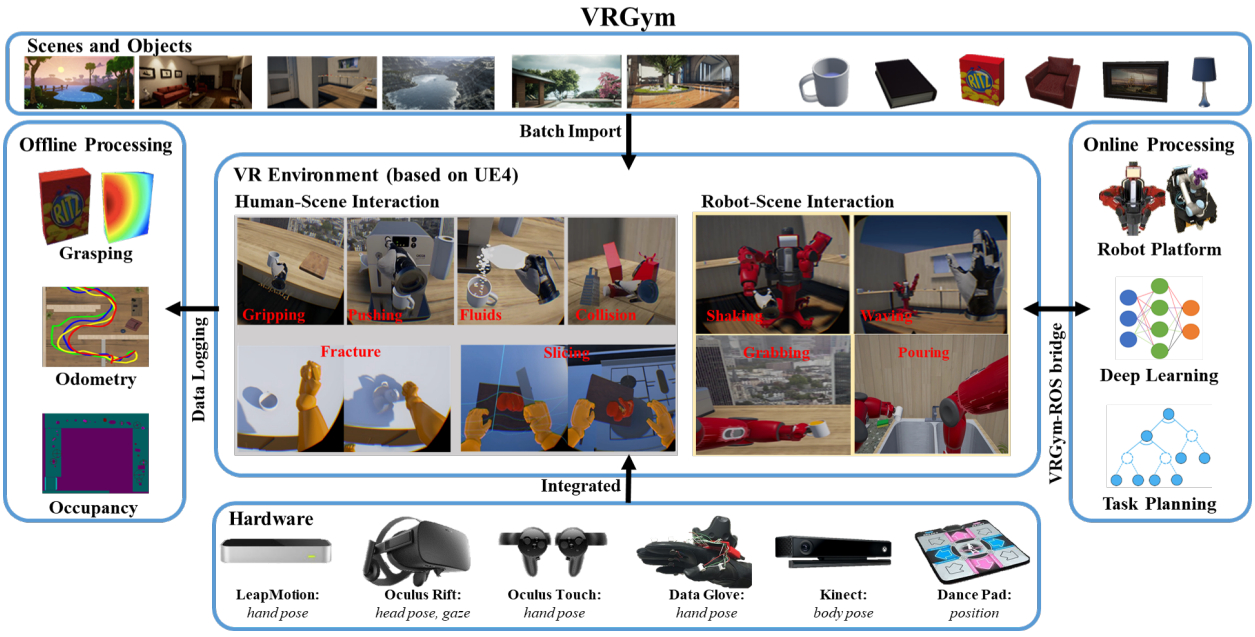


Figure 3.2: System architecture of VRGym, consist of three major components: (i) Hardware modules for human data input. (ii) Scene modules batch import various category of scenes as well as diverse objects, derived from different resources such as 3D modeling tools, scanned models, and automatically generated synthetic data. (iii) VR environment serves as an ideal testbed, where both a human and a robot can perform diverse tasks. The inherent physics-simulation engine enables realistic human-scene interactions and robot-scene interactions.

Sketch of VRGym's system architecture is illustrated in Fig. 3.2. For realization of interactive tasks, VRGym provides diverse realistic scenes and simulations for both human and robots. The data collection is automatically conducted by logging the ground truth data during the task performance. To fulfill those purposes we develop and integrate three modules in VRGym: i) *scene module* that renders realistic 3D scenes and objects given configurations, ii) *environment module* is developed upon UE4 game engine with physical simulations for rich interactive tasks and iii) *VR hardware module* imports human inputs to VRGym for human embodiment.

3.2.1 Scene Module

Creation of simulation environments requires the building blocks of scenes and objects. They are the key elements to enhance the diversity for VRGym. To satisfy the goal, we design and develop several pipelines, both online and offline, to import or generate scenes into VRGym. The pipeline is customized in the sense that users can specify their needs through a configuration file. Scene module provided in VRGym largely increases the variety of static environments. Meanwhile, ground truth information such as RGB, depth and segmentation frames are automatically captured in real-time. The automatic synthesized data generation and labeling enables model learning and robotics applications.

In details, VRGym incorporates large scale open source datasets collected from the web parser [CDF17, SYZ17], or generated automatically from the existing object assets [JQZ18, QZH18, YYT11] as depicted on the top of Fig. 3.2. VRGym also provides interfaces for users to manually construct scenes for more specific task requirements. Compared to existing datasets or platforms, none of those has the capacity to satisfy all the constraints.

For individual objects, the scene module can directly import the standard mesh files into VRGym. Open source CAD datasets [CWS15, CFG15] provides mesh files that are convenient to be leveraged by VRGym. In addition, customized complex objects such as 3D scanned object models using RGB-D sensors can be imported in VRGym for specific task design. Users can further adjust the static object models by assigning different properties after the import.

3.2.2 Environment Module

In VRGym, the simulation is conducted by UE4 which is an advanced real-time physics-based simulation engine. Compared to other simulation environments that only provides rigid body or symbolic planning typed simulations, the environment module in VRGym affords a diverse set of physics simulation for interactive tasks. The realistic physical effects

Figure 3.3: Examples of various physics-based simulation for diverse tasks in VRGym beyond merely rigid-body simulation in other 3D virtual environments. (Top) Pouring water. (Bottom) Folding clothes.

including rigid body, soft body, collision, fluid, cloth, slicing and fracture. Typical examples are demonstrated at the center of Fig. 3.2 and more concrete ones are presented in Fig. 3.3. Given the articulated physics simulation in environment module, subtle objects state or fluent changes are diversify and realistic existing during virtual robot and scene interactions. The authentic physical effects provide crucial visual experience that minimize the gap between the simulation and real world tasks.

3.2.3 VR Hardware Module

VRGym has its own unique aspect by realizing human embodiment in 3D virtual environments compared to existing platforms. Human embodiment in the virtual environment is achieved by representing the physical human as an avatar in real-time. To achieve this, one aspect of developing the VR hardware module we highlight is the real-time tracking on physical human such that the human movements and manipulations would be accurately reflected in the virtual environment. This results in developing a humanoid mesh which is able to deform to afford different motions based on the underlying tracked body and hand skeleton poses.

Setup for VR hardware module in VRGym contains: i) Kinect RGB-D sensor. It tracks and maps the human skeleton to the virtual avatar by developing a Kinect plugin for UE4, ii) Oculus headset. It tracks real-time human head pose and gazing, iii) Dancing pad. It provides the control panel to navigate the avatar inside the virtual scene, and iv) three types input interfaces to provide different resolutions of human manipulation. The realization of human embodiment in virtual environment emphasizes VRGym's ability of conducting tasks with rich interactions. The three types of hardware interfaces are specified as follow:

- *Oculus Touch Controller*. We incorporate Touch controller to enable a direct attachment of virtual objects on avatar hand or gripper if the grasp event is triggered. The overall grasping effect is akin to the firm grasp with the least realistic level of human-object interactions. This manipulation style is useful in conducting event level tasks, e.g. the fine-grained motions is not required.

- *LeapMotion*. We integrate the off-the-shelf commercial hand pose sensing products to satisfy real-time visual based hand gesture recognition. The LeapMotion sensor is mounted on the headset. When integrated into the virtual environment, the tracked hand poses will be reflected by avatar's hand model, though sometimes visual occlusions or sensory failure may result in noisy hand poses.

- *VR Glove System*. We extend our tactile glove by integrating it into VRGym aiming at providing the finest-grained manipulation. To track the real-time glove pose, a Vive Tracker is placed above hand palm to obtain the hand's global positioning. Rest of the phalanx orientations are calculated by the IMU network using forward kinematics. The VR glove system provides reliable hand pose sensing and targets on complex interactive tasks where subtle hand motions are critical in learning the manipulation skills.

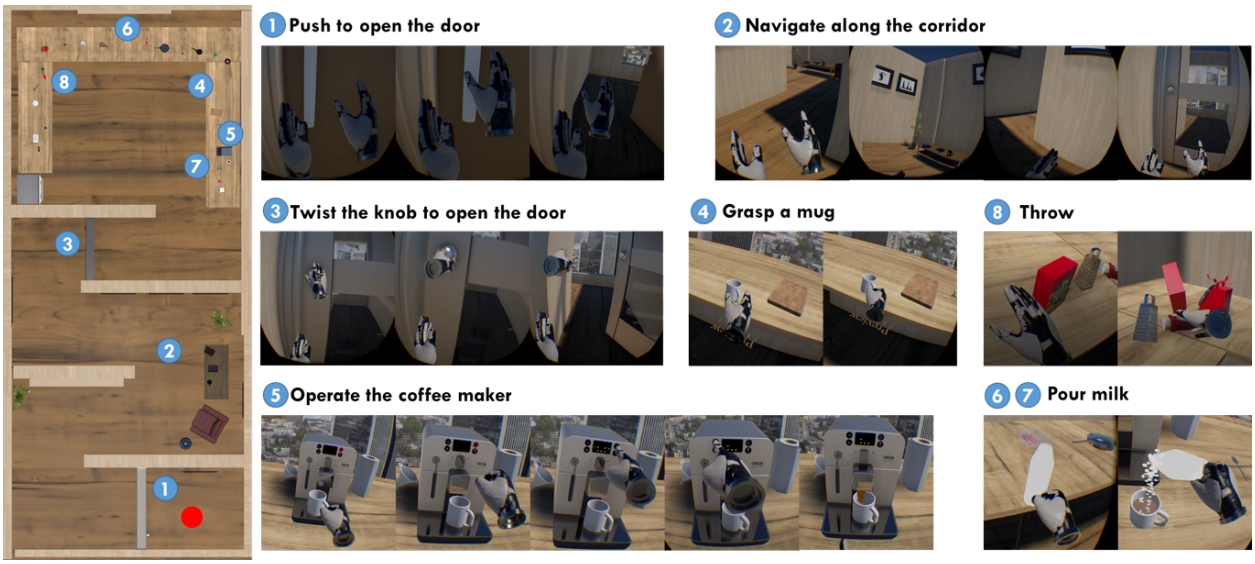


Figure 3.4: A human agent performs a series of actions in a virtual scene using Oculus Touch controllers. (Left) Action sequence from a top view of a virtual indoor environment. (Right) Sequences of the performed actions. Specifically, the human agent starts at the red dot as shown in the left, (1) pushes a door, (2) navigates along the hall, (3) twists a door to enter the kitchen, and (4)-(7) makes a cup of coffee. This process involves (i) large movements using the human embodiment provided in VRGym (navigating along the hallway), (ii) complex operations (operating the coffee maker), (iii) fine-grained manipulations (twisting the doorknob), and (iv) physics-rich controls (pouring milk).

3.3 Software Development

In VRGym, we develop two software interfaces that provides the testbed to train and benchmark various interactive tasks. The first one is the data collection system that is implemented upon the hardware interfaces. The purpose is to keep tracing of the multimodal information during different kinds of interactions between the agents and environment. The second interface, VRGym-ROS bridge, is developed to extend the virtual environment compatible with robotics algorithms and methods.

Fig. 3.4 illustrates an virtual environment built in VRGym. The environment itself affords

semantically diverse interactive tasks for multi-purpose agent. In comparison, conducting similar tasks in the real world along with data collection would become extremely intractable. In this environment, an agent or avatar, is initialized at the staring point which is represented as the red dot in Fig. 3.4. The virtual agent is navigating to the final goal inside the kitchen meanwhile accomplishing several sub-tasks. Steps to achieve this task including *navigations*, for example, go along the corridor and open the kitchen door. And *manipulations*, e.g. make a cup of coffee. The operation involves grasping and moving a mug, pushing several buttons sequentially on coffee machine. The process requires a elaborated *task planning* that is being executed by physics simulation.

3.3.1 Agent Data Collection

In VRGym, ground truth labeling is automatically generated when human avatar or robot is interacting with the scene. There are two typical scenarios we demonstrate VRGym's data collection and associated applications.

Grasping Different levels of manipulation are made feasible in VRGym. Among them, the VR glove device [LZX19] makes finer-grained manipulations feasible. Visualization of the object manipulations by using the VR glove are illustrated in Fig. 3.5(a). After obtaining the grasping data on various objects, we merge the data point and form grasping heat maps on object surfaces to indicate the likelihood of grasping points distribution on object models. In addition, Fig. 3.5(b) shows the averaging heat maps collected from 10 human subjects. In an heat map, the hotter area reflects the denser grasping points are, and implies more likely a human avatar would grasp close to that area.

Odometry VRGym offers functionality to collect agents' odometry data when performing the task. Fig. 3.5(c) demonstrated the collected odometry performed by 5 human subjects with different experience playing with VR. The navigation is achieved by using Oculus

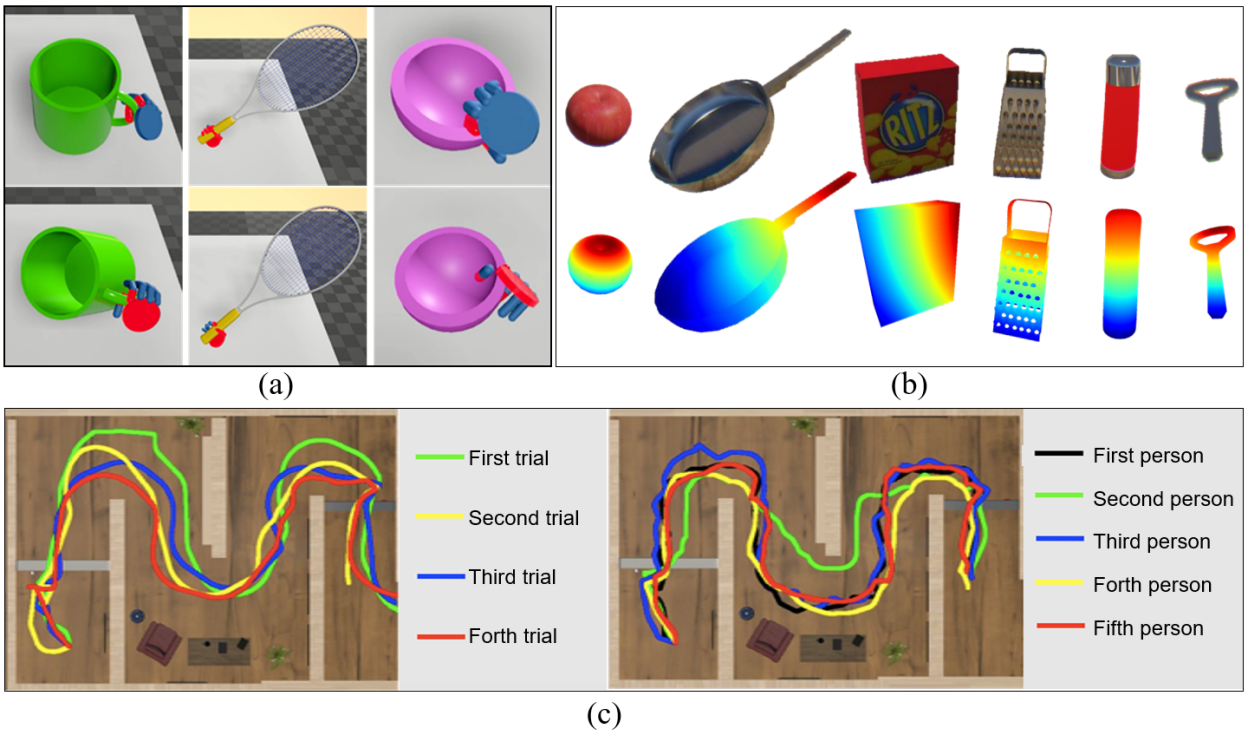
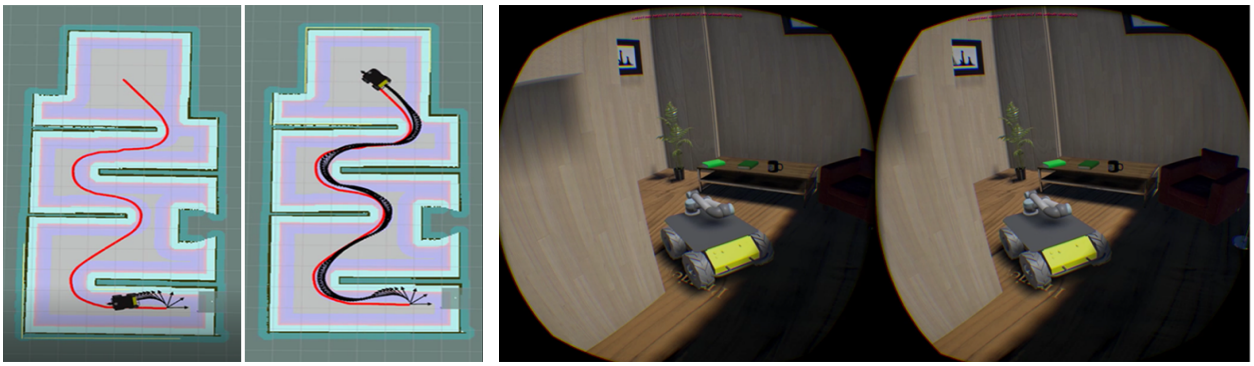


Figure 3.5: (a) Grasp a mug, a tennis racket, and a bowl. The red area indicates the contact force between the virtual hands and the object. (b) Visualization of the collected human grasp data. Top: a set of 3D objects. Bottom: the average grasp heat map generated by multiple subjects. (c) Visualization of footprint from different subjects.

Touch controller where the buttons are mapped to atomic motion signal for virtual agent movements.

3.3.2 ROS Interface

The VRGym is introduced to be compatible with the popular open source robotics framework – ROS. A customized communication bridge is developed for data transferring. Off-the-shelf robot models and algorithms are able to play inside the virtual environments and vice versa. For example, the automatically generated virtual scene in VRGym can be exported to Gazebo simulator and performing a robotics task with different ROS packages.



(a) (b)

Figure 3.6: VRGym-ROS bridge. (a) The robot navigation in the scene imported into the Gazebo, exported from the VRGym. The red curve indicates the path planned by the robot's global planner. The black curve is the actual trajectory executed by the robot. (b) A Husky-UR5 robot is imported into VRGym from ROS to guide the way and open the door for a human agent.

Implementation The ROS interface, VRGym-ROS bridge, is developed under the standard TCP/IP protocol for the purpose of communicating with the popular robotics algorithms. Given this interface, robot models can be smoothly imported to VR environments. Control signals can be as well applied to move the virtual robots by the backend ROS packages. Data stream transferring between the ends are connecting either physical or virtual robots. Format of the data type we define is by leveraging the JSON format. JSON parsers for both VRGym and ROS are deployed to further improve the consistency. In the scenarios where the environment has multiple agents, each will open a port in the protocol to support a specified data stream. Utilizing the VRGym-ROS bridge, training and evaluating human robot interactions (HRI) can be conducted seamlessly when playing with VRGym. Compared to the existing robotics simulators, Gazebo or V-Rep, we enable the human in the loop to study the related problems in the pure virtual environment.

Evaluation Performance of our VRGym-ROS bridge is evaluated on a virtual Clearpath Husky robot, see Fig. 3.6. The robot in a VRGym scene is performing a navigation task. The 3D robot model is imported from ROS and a number of SLAM algorithms and path planning methods are feasible to be evaluated. In Fig. 3.6(a), the mapping results are acquired by using the GMapping package provided in ROS. The planned path is plotted in the red curve, while the ground truth odometry is depicted as the black curve. Fig. 3.6(b) presents the user's view while the robot is navigating. As demonstrated in this task, algorithms for virtual robot performing in the diverse scenes is realized smoothly by leveraging the VRGym-ROS bridge.

3.4 VRGym Experiments

We conduct set of experiments in VRGym to further demonstrate its capability and usability to support learning from interactive tasks. Towards human robot interactions, human intention prediction and social interaction tasks are performed. VRGym is a multi-purpose testbed in the sense that prevalent machine learning algorithms are able to be benchmarked in the VRGym.

3.4.1 Experiment 1: Intention Prediction

Problem such as intention prediction is usually reluctant to be conducted on a physical robot because of tiny error tolerance. The incorrect intention prediction may result unwanted behavior for both human and robot. In VRGym, studying intention prediction particularly becomes suitable due to multimodal data can be utilized for the inference process. Typical instances include: human trajectories, human body poses, object states, visual information, etc.. Since tracing and collect those information are available in VRGym, predicting intention is made possible to be investigated by unique setup.

In this experiment, we conduct different algorithms of human intention prediction and

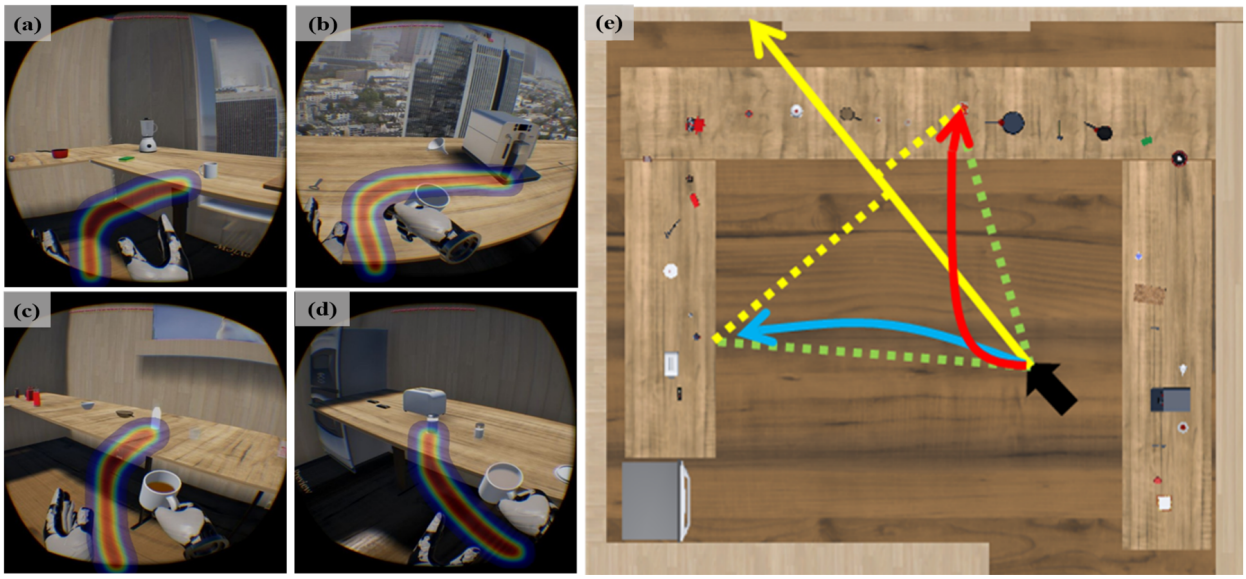


Figure 3.7: Intention predictions in a coffee-making task. (a) Grab a cup. (b) Use the coffee machine. (c) Pour milk. (d) Add sugar. (Right) Visualization of three intention prediction algorithms. Blue and Red: sampled paths from the grammar model [QHW17]. Green: straight-line distance. Yellow: prediction by shortest perpendicular distance (dashed lines) from objects to the ray direction (solid arrow) based on avatar's location.

analyze the potential of VRGym as a testbed for related tasks. For human data collection, 20 subjects are recruited to play with an interactive task, Fig. 3.7, in a virtual kitchen scene. The virtual kitchen contains 20 objects that are placed on top of three long tables. In Fig. 3.7, the agent is required to start from the entrance (red dot) and performs four steps to finish a coffee making task: grasp the coffee mug, operate the coffee maker, grasp milk bottle and add milk, grasp sugar can and add sugar. One thing worth to notice is that this task can be performed in different orders. One example of subject's task trajectory is plotted in Fig. 3.7. The task requires all subjects to finish the coffee making task by using the available objects in the virtual kitchen.

The illustrative results compared among three methods are presented in Fig. 3.7(e). Qualitative results are shown in Fig. 3.7(a-d). They indicate the human agent's intention

in terms of heat maps when interacting with the scene objects. The hotter area on the images reflect higher probability of the intention. The semantic level prediction is achieved by learning from full history of the human's task performance which incorporates navigation trajectories and grasping data collected using VRGym.

3.4.2 Experiment 2: Social Interaction

Figure 3.8: Human robot interactions in VRGym. A Baxter robot (a) waves hands and (b) shake hands with a virtual human agent.

Social interactions in addition to physical interactions is another critical form that naturally happens in human-robot co-existing environments. The problem we would study is let the virtual robot understand the human's social behaviors, such as hand shaking and waving, then respond to them reasonably. VRGym provides interface for human embodiment in the virtual scenes that closes the gap of studying human-robot interactions in the simulation.

Participants We conduct the experiment with 10 subjects participating in. The algorithm we implement is following [SGR17] which learns the robot social affordance. More technical details can be referred to the original work.

Results We demonstrate qualitative results in Fig. 3.8. Specifically, the robot waves its arm as the response to the human avatar's hand waving as presented in Fig. 3.8(a). And the robot stretches out its end-effector and shakes its hand with human avatar. The response action signals are sent through the VRGym-ROS bridge. The corresponding motion planning for the robot arm is computed at the ROS end, the planned results are transferring back to the virtual Baxter robot to reflect its appropriate response to the human avatar's social behaviors. By this case study, our VRGym is proposed to push the direction of learning the human-robot interaction in simulation environments. Safety and cost issue can be avoid accordingly compared to the real world cases.

3.4.3 Experiment 3: RL Algorithms Benchmark

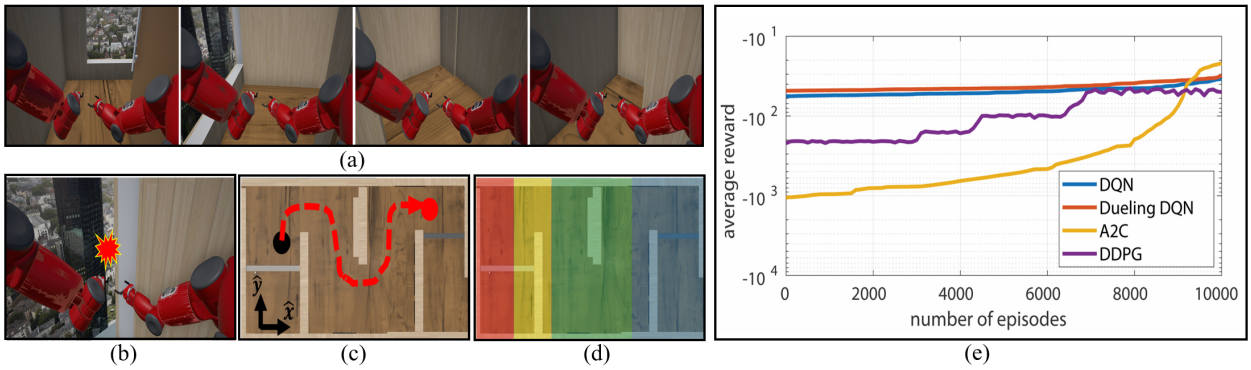


Figure 3.9: Settings for the RL training inside VRGym environment for an indoor maze navigation task. (a) First-person view of a virtual robot. (b) The robot collides with a wall, triggering negative rewards. (c) An eagle view of the indoor navigation task. (d) Rewards assigned in different color zones (red, yellow, green and blue) from low to high. (e) The performances of the tested RL algorithms.

VRGym is introduced as a multi-purpose testbed which offers the playground and learn robot models by leveraging off-the-shelf algorithms. In Fig. 3.9, a virtual Baxter robot is aiming at learning a motion policy that enables it to navigate in a 3D maze-like indoor corridor. During the learning phase, the robot is learning the motion cost by itself when interacting with the scene. This trial and error training strategy follows the standard Reinforcement Learning (RL) framework. The RL algorithms are running backend at ROS, motion signals are transferred to the robot in the virtual scene.

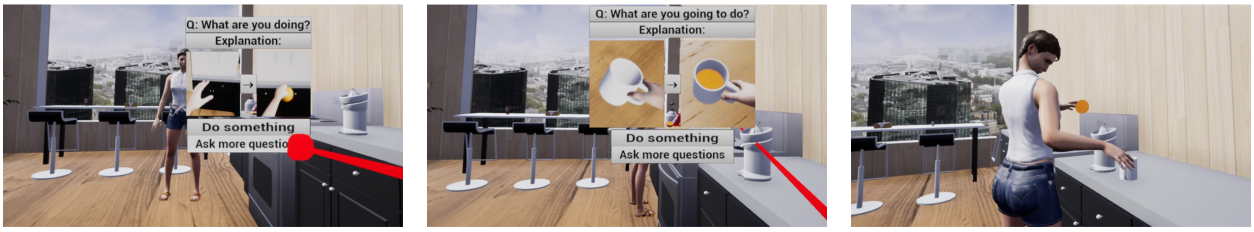
Compared to other existing platforms, such as OpenAI Gym, the proposed VRGym has two primary merits conducting benchmarking:

- *Sophisticated Interactions.* With the integration of articulated physical simulation, VRGym provides realistic physical interactions to reflect more effective policy learning.
- *Physical RL Agent.* Enabled by model importing, VRGym is capable of realizing a virtual but physical robot model interacting inside the virtual scene. This configuration makes it flexible to transfer the robot model to different environment setups to validate the learned policy.

The state-of-the-art deep RL algorithms we adopt for benchmarking of the navigation task are **DDPG** [DCH16], **DQN** [MKS15], **Actor-Critic** [MBM16] and **Dueling-DQN** [WSH16]. We extract real-time robot first-person camera view in pixels as the state input. The quantitative results of the mentioned four algorithms are demonstrated in Fig. 3.9(e).

Further, VRGym is as well capable of benchmarking algorithms in the category of Learning from Demonstrations. Trajectories of expert (human) demonstrations are automatically collected and labeled for robot to imitate. Prevalent Inverse Reinforcement Learning algorithms are performed as well which details can be referred to the next chapter.

3.4.4 Experiment 4: Embodied Agent Task Learning



(a) (b) (c)

Figure 3.10: A humanoid robot in the virtual kitchen scene is performing juice making task. (a)(b) Human-Robot collaborations by giving explanations. (c) Robot task execution.

To achieve the purpose of training a general purpose intelligent agent, learning from a embodied task that implicitly incorporate multiple steps and elaborated task planning is crucial to be studied. VRGym offers this ability by setting up the virtual environments for embodied task learning. We expect to "grow a crow" (agent learning as smart as a crow) by leveraging the outstanding aspects of VRGym.

Fig. 3.10 demonstrate how the humanoid robot performs the embodied task – making orange juice while providing explanations to the human collaborator. Fig. 3.10(a) showcases the robot provides its explanation of "picking up an orange to make juice". And Fig. 3.10(b) presents the robot gives the explanation of "getting the orange juice by tacking a cup". Those are key steps reasoning that to fulfill the sub-task goals. Fig. 3.10(c) illustrates the frame of execution at the final step where the robot puts the orange inside the squeezer and grabs a cup to fetch the juice. The embodied task can be achieved once all sub-tasks pre and post conditions are all satisfied. It requires the sophisticated physics engine to faithfully evaluate the execution. Further, explanation is also an important supplementary to the embodied task such as the robot is required to explain its own behavior to verify the task planning and execution. VRGym is developed to boost those related facets of research which we think would be indispensable to grow an general-purpose agent.

3.5 Appendix

3.5.1 Simulation Platforms

We summarize simulation platforms with detailed comparisons w.r.t various criterion and prove the uniqueness of VRGym in Table 3.1.

Robotics simulation platforms The Robotics Operating System (ROS) plays an important role on open source robotics development. Simulators such as Gazebo and V-Rep provides advanced software packages for robot modeling and motion planning. As a robotics framework, however, they lack human embodiment such as virtual reality (VR) to involve human interactions.

Virtual training platforms This includes OpenAI Gym [BCP16] and MuJoCo [ETT15]. They are developed to evaluate and benchmark various robotics algorithms. Tough those platforms are easy to setup and integrates compact playground like environments, they are still insufficient of studying problems such as robot learning from interactions which are crucial for physical agents.

Physics based simulation In game industry, people are developing game contents by incorporating sophisticated physics simulation. For example, CARLA [DRC17] is an open source project that provides simulation for autonomous driving. AirSim [SDL18] offers photorealistic rendering of outdoor scenes designed for drone navigation. However, those two platforms focus on specific tasks, e.g. learning of vehicles or drone navigation. Further the more general-purpose platforms are made available to AI community, such as AI2THOR [KMG17] and Gibson [XZH18]. AI2THOR offers articulated 3D indoor environments where agents can be controlled to interact with the scenes and objects and perform tasks. Whereas the symbolic level interactions are not physical realistic and no human em-

bodiment makes study of robot social interactions not available. Gibson is mainly targeting on agent indoor navigation problems. Although the agent can interactive with the scenes, the level of manipulations an agent can afford is not sufficient.

Table 3.1: Comparison with existing 3D virtual environments. Scale: Contains a large number of scenes. Physics: Supports physics-based simulation on agents and objects. Real: Provides a life-like rendering. Action: Object states can be changed by actions. Fine-grained: Enables fine-grained actions and simulates plausible object state changes. Human: Humanoid agents. Multi: Supports a multi-agent setting.

Environment	Scale	Physics	Real	Action	Fine-grained	Human	Multi
SUNCG [SYZ17]	✓						
Matterport3D [CDF17]	✓						
Malmo [JHH16]	✓			✓			✓
DeepMind Lab [BLT16]							
VizDoom [KWR16]							✓
MINOS [SCD17]	✓		✓				
HoME [BPA17]	✓	✓	✓				
Gibson [XZH18]	✓	✓	✓			✓	
House3D [WWG18]	✓	✓	✓				
AI2-THOR [KMG17]		✓	✓	✓			
VirtualHome [PRB18]		✓	✓	✓		✓	
VRGym	✓	✓	✓	✓	✓	✓	✓

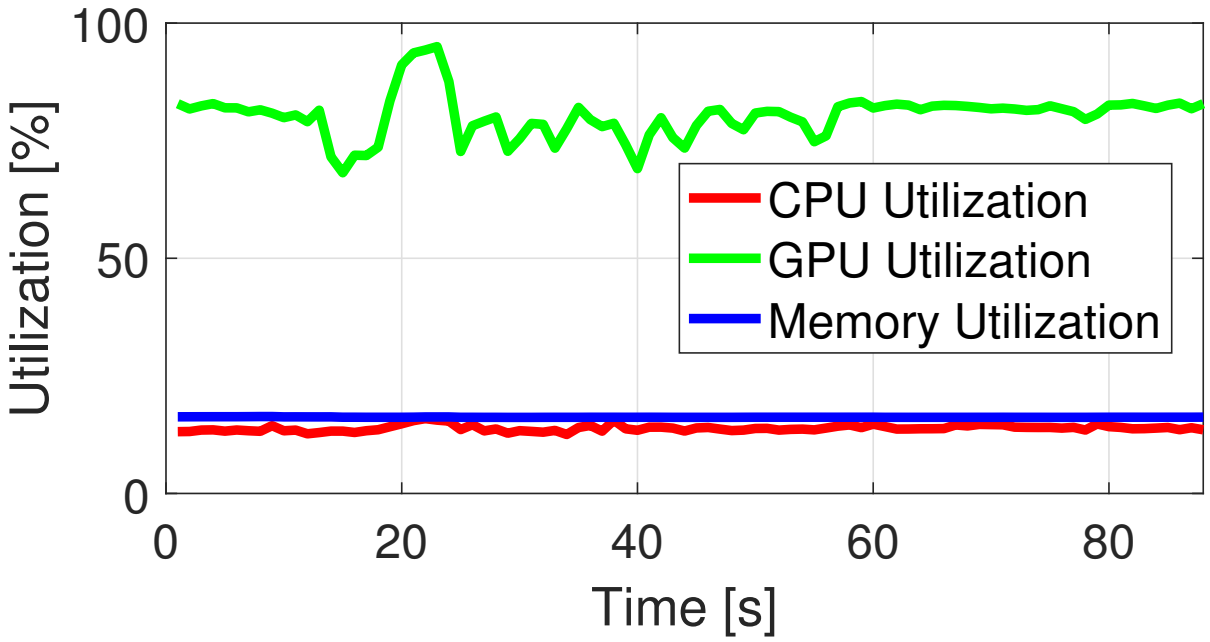


Figure 3.11: System performance: GPU (green), memory (blue), and CPU (red) utilization.

3.5.2 System Performance

VRGym is executing the task simulation in real-time (30fps) on a modern desktop equipped with 8700K CPU, a DDR4 memory set 64GB in total and an EVGA GTA 1080 Ti GPU. Fig. 3.11 depicts the system performance when running physics simulation of manipulation tasks with human input devices that basically requires essential software and hardware interfaces. The results reflects VRGym is running efficiently on CPU and memory utilization. Since physics simulation heavily relies on parallel computing, more GPU power is required to satisfy the purposes. The performance results verify that VRGym can be deployed on modern computers or servers without additional dependencies.

3.5.3 Evaluation of Data Communication

Performance of VRGym's data communication is summarized in Fig. 3.12 where 20 ROS packages are sent individually for concurrent connection. According to the results, the

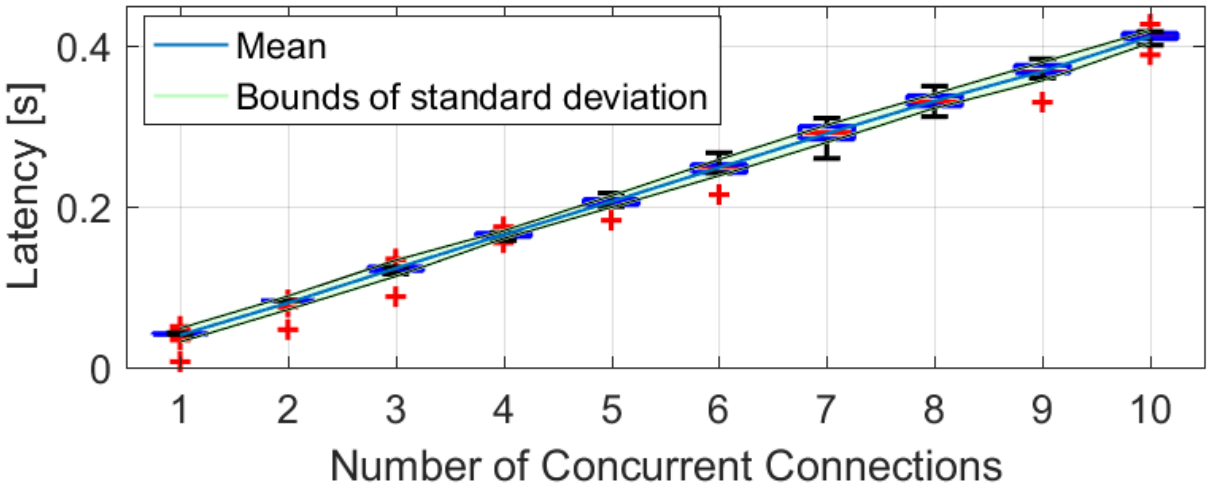


Figure 3.12: Evaluation of the latency in VRGym-ROS communication bridge. Each connection contains 20 packages, in which 512Kb data was sent. Linear regression is fitted to the mean of the latency $t(9) = 2.9025, p = 0.01, r^2 = 0.9998$, indicating a strong linear trend with respect to the increase of the concurrent connections.

communicating latency for VRGym-ROS bridge increases from 0.04s to 0.41s. We apply the linear regression to fit the parameters of the latency. The results indicate a strong linear relation between the latency and the concurrent connections.

3.6 Conclusion

As proposed, VRGym is developed as a promising physical and photo realistic simulation platform that embodied as a virtual testbed for robot learning interactive big tasks. The testbed can be leveraged for general-purpose agent training and evaluations. The most innovative aspect of VRGym is the human embodiment which is realized by a virtual avatar using a wide range of hardware setups for body, hand and manipulative sensing to a fine-grained level. Popular robotics framework ROS can be integrated to VRGym through the customized VRGym-ROS bridge. Various evaluations have been conducted in VRGym, and

the performance indicates the data communication with ROS is robust to support diverse robotics applications. Demonstrating with four experiments, it further verifies VRGym is effective in performing different levels of interactive tasks. Along with multimodal sensory information, VRGym automatically collects ground truth data with generated labels that shows the merit of studying multiple forms of interactions in simulation environments. In addition, VRGym is a multi-purpose testbed for physical robot interactive tasks such as RL and IRL. This capability offers training robot with advanced machine learning methods. We are confident that VRGym would be potential to boost the development of robotics applications. It benefits the direction of learning from interactions to ultimately achieve *big tasks*.

CHAPTER 4

Learning from Interactions: Exploiting Small Data

Interactions for robot learning are critical in the sense that a fruitful source of data is generated during the interaction. Those data includes multimodal sensory information that is further modeled statistically for a robot to generate the manipulation policy, representation of knowledge and growth of commonsense. Nowadays, one popular trend of research is to learn from large scale statistical data. People are making efforts to interpret the robot interactions by building up machine learning models that heavily rely on the quantity of data. However, one notable downside of this strategy is "applying huge data while only achieving a small task". In contrast, we are exploring the feasibility of achieving the task learning with reasonable performance by only providing a small portion of data. The angle we are discussing is to follow the community of Learning from Demonstration (LfD). Specifically, we are investigating the problem of robot learning grasping skills through IRL framework with a limited demonstration data.

4.1 Problem Overview

Inverse Reinforcement Learning (IRL) [Rus98] aims to learn a reward function under the framework of Markov Decision Process (MDP) [Put14]. Reward function will in turn function as a supervision signal that best explains the observed expert demonstrations. One crucial assumption made for every IRL problem is: the experts knows the underlying ground truth reward function when performing the demonstrations. And policy or actions are being executed on every state in demonstrations are assumed to be optimal. However, this assump-

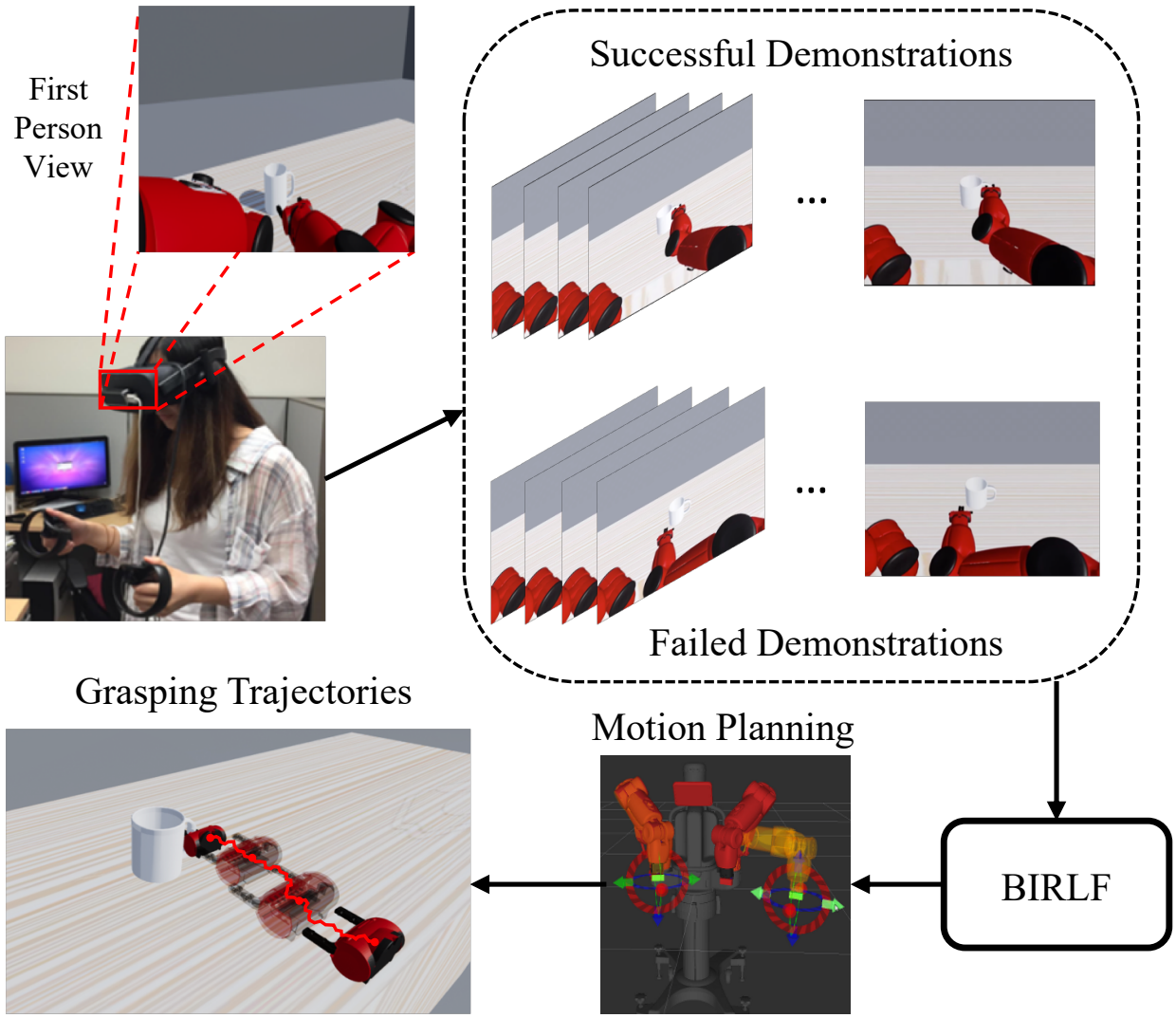


Figure 4.1: Human demonstrators perform a complex grasping task with both successful and failed demonstrations in the virtual environment. With such demonstrations, the proposed BIRLF infers the reward function of the observed human demonstrator in a given task. The robot then learns the policy to achieve the demonstrated task, reproducing the grasping trajectories with a motion planner.

tion is too strong that results in a hard constraint when human performers are executing the task on the data collection stage: all failure demonstrations are discarded while only keeps the successful demonstrations.

Instead, we think this constraint is unnecessary. We have two folds reasoning: i) failure demonstrations are more readily available than the successful ones , ii) intelligent agent, like human, is able to exploit on a finite set of demonstrations even with failure cases. In addition, comparing the demonstration trajectories between successful and failure ones, failed demonstrations are highly resemble successful ones. There are only differences existing at a few critical trajectory points. In learning from interactions, it becomes manifest when robot is learning the grasping skills where the failed demonstrations are distinctive from the successful ones in very few lasts steps such that the failed trajectories result in contact points that are slightly off stable grasps. According to our observations on collected demonstration trajectories, the failed ones are always not made to be failed intentionally by performers. Both successful and failed demonstrations are pursuing the same trajectory distributions for the task.

In literature, some works propose data efficient approaches [ET16, CRK17] to handle learning with failed demonstrations. Failed demonstrations are usually treated as noise modeled by [CAN08, SBS08, VGL13, RSB09, MLF10, ZLN14]. However, we argue that failed demonstrations should be utilized as well for the seek of learning from "small data": i) it is both time-consuming and demanding for performers to collect only successful demonstrations for complex interactive tasks, and ii) failed demonstrations, as described, contain useful information as well may help gaining of task performance.

To effectively leverage failed demonstrations, people carefully propose algorithms to incorporate them to improve IRL learning. [SMW16] introduces an early stage IRLF where a optimization problem is formulated by adding new constraints in the framework of maximum causal entropy [Zie10]. It encourages the learned policy to exhibit a different distribution from failed demonstrations, meanwhile acquiring better convergence property with successful demonstrations only. [LCO16, CLO16] apply Gaussian process, to generate a proficiency term for both successful and failed demonstrations. The experiment potentially shows the effectiveness of adopting failed demonstrations on task of *GridWorld*. On the other hand, RL

community also proposes approaches in coping with non-optimal demonstrations. [GLY18] learns the policy from both the optimal and sub-optimal demonstrations. However, those IRLF methods relies on low dimensional state or action space, some are defining explicit feature representations. They are reluctantly to be extended to complex interactive tasks such as robot grasping object will be discussed in the later content.

In our approach, we naturally extend low dimensional representations with complex environment setup where continuous state and action spaces are adopted for robot manipulation. In addition, our method learns the feature representations in an automatic manner and composes the learning of reward function by sampling through Markov Chain Monte Carlo (MCMC) process. Overall, our method alternates between a weight sampling using a MCMC sampler and feature function learning by policy iteration. This learning strategy assists the model to leverage function approximators to better estimate the true reward functions. Further, we formulate our method using Bayesian learning, interpreted as BIRLF, that is achieved by adding geometric convex constraints which is the first attempt to cast IRLF under Bayesian framework.

The effectiveness of our proposed approach is evaluated on our developed VRGym platform. We setup an environment where an virtual Baxter robot is tasked to grasp various virtual objects in a distance. For collecting demonstrations, we leverage the human VR interface to teleoperate robot arm which motion planning is achieved by running Gazebo on ROS backend. See Fig. 4.1 for task setup. In each trial, the performer is using VR devices to control the virtual robot to achieve the object grasping. In details, human subjects are trying to move the robot's end-effector and configure the hand pose as the planned state which would mostly likely results in a stable grasp. The evaluation of the grasping is performed by VRGym physics engine. Given those setups, both successful and failed demonstrations are automatically logged and labeled when performing the task. Efforts in data collection process are largely reduced.

4.2 Background and Notations

4.2.1 Markov Decision Process

In Markov decision process (MDP), the learning environment can be defined as a MDP tuple $\mathcal{M} = \{S, A, T, \gamma, r\}$ [Put14], where S denotes the sate space, A the action space, $T(s'|s, a)$ the environment transition probability from state s to s' by taking action a, γ the discount factor, and r the reward function. In IRL problem for LfD tasks, the MDP tuple becomes $\mathcal{M} \setminus r$, while a set of trajectories $\mathcal{D} = \{d_1, \ldots, d_N\}$ are provided from expert demonstrations. Each trajectory d_i is represented as a sequence of state-action pairs.

4.2.2 Policy and Value Function

A policy for an learning agent is defined as a mapping function from state space to action space: $\pi : S \mapsto A$. Given a policy π, the state value function $V^\pi(s)$ is defined as the expected accumulative reward gained for an agent start from state s and executes the policy π thereafter. The value function is mathematically defined as:

$$V^\pi(s) = \mathbb{E}_{a\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)|s_0 = s\right]. \tag{4.1}$$

A state-action value function Q could be similarly defined for each state action pair (s, a):

$$Q^\pi(s, a) = r(s) + \gamma \mathbb{E}_{s'\sim T(\cdot|s,a)}[V^\pi(s')]. \tag{4.2}$$

The policy $\pi(s)$ is the optimal policy under the reward function $r(s)$ if the following inequalities hold for all states and actions [NR00]:

$$\begin{aligned} \mathbb{E}_{s' \sim T(\cdot|s,\pi(s))}[V^{\pi}(s')] &\geq \mathbb{E}_{s' \sim T(\cdot|s,a)}[V^{\pi}(s')] \\ Q^{\pi}(s,\pi(s)) &\geq Q^{\pi}(s,a) \\ \forall a \in A, s \in S. & \end{aligned} \tag{4.3}$$

4.2.3 Bayesian Inverse Reinforcement Learning and PolicyWalk

BIRL [RA07] presents a statistical view by modeling the uncertainty of reward function in probability distribution. The performer $\mathcal{A}$ executes the task and collects a trajectory $d = \{(s_0, a_0), (s_1, a_1), ..., (s_T, a_T)\}$. Assuming $\mathcal{A}$ is maximizing the accumulative reward while accomplishing the task using a stationary policy, the posterior probability of reward function r follows Bayes theorem can be formulated as:

$$P(r|d) = \frac{P(d|r)P(r)}{P(d)} = \frac{1}{Z}\exp(\alpha \Sigma_t Q(s_t, a_t; r))P(r), \tag{4.4}$$

where Z is the normalizing constant, α the degree of confidence, and $Q(s_t, a_t; r)$ the state-action value function given reward function r. Estimating the posterior in practice is difficult. PolicyWalk [RA07] is proposed to resolve the issue by conducting an efficient sampling procedure. Given a fixed reward function r, the MCMC sampler optimizes on-policy π to update the estimation of Q function. In the stage of MCMC, it samples $\tilde{r}$ from the neighborhood of the current r. After jumping to the chain of $\tilde{r}$, the new optimal policy only requires several update of policy iterations from the previous learned one. [RA07] also proves the property of rapid mixing of the Markov chain with a uniform reward prior, which offering a theoretical treatment on convergence.

4.3 Bayesian Inverse Reinforcement Learning with Failure

In this section, we detail our proposed method learning with failure demonstrations. We formulate our method in Bayesian framework by incorporating the value function, a set of

inequalities Eq. 4.3. The inequalities of policy optimal condition implicitly define a valid convex set in terms of halfspaces. We denote the trajectory set of successful and failed demonstrations as D and F respectively in the following.

4.3.1 Problem Formulation

Based on Eq. 4.4, the posterior probability of reward function r under the full demonstrations $\Psi = D \cup F$ can be expressed as:

$$P(r|\Psi) \propto P(\Psi|r)P(r). \tag{4.5}$$

We apply an MCMC chain to sample the reward function from the posterior. The reward function r not only matches the statistics of trajectories in D but captures the implicit difference between D and F. To realize this thinking, we decompose the likelihood term in Eq. 4.5 into:

$$P(\Psi|r) \propto \exp(\alpha \Sigma_{d \in \Psi, t} Q(s_{d,t}, a_{d,t}; r) + \beta \Delta U(D, F; r)), \tag{4.6}$$

where the first term is identical with Eq. 4.4, $\Delta U(D, F; r)$ in the second term measures the *potential* difference between D and F with a coefficient β.

4.3.2 Halfspace Induced Potential

Further, we characterize $\Delta U(D, F; r)$ on different trajectory sets by utilizing Eq. 4.3. And we propose the scheme to quantitatively compute the potential term among different solution spaces. Assume we are parameterizing the reward function r as $r(s) = \omega^T \phi(s)$, where ω is the feature weights and $\phi(s)$ compactly encodes the state features. Given this representation, the Q function can be reformulated by:

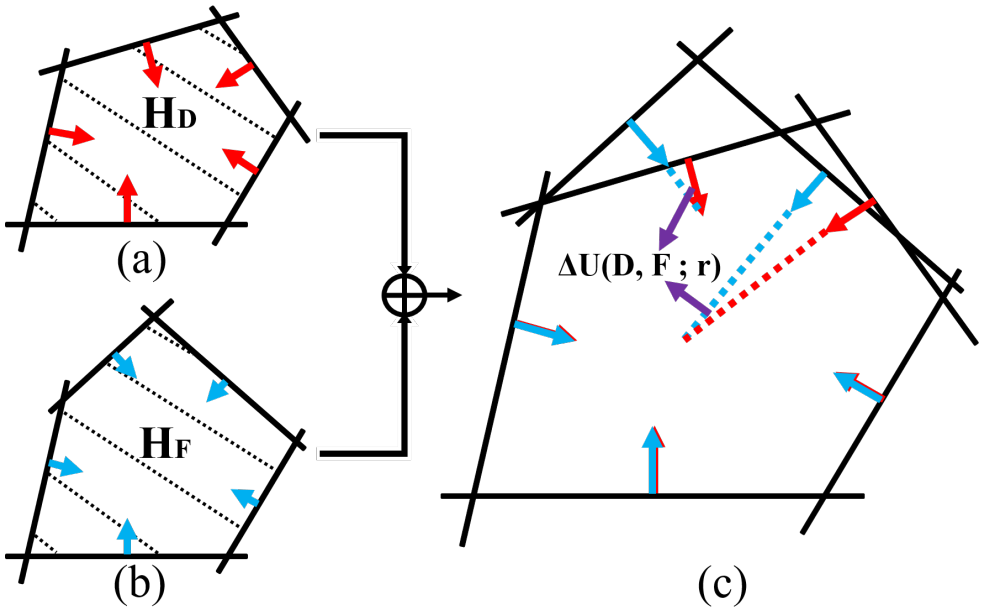


Figure 4.2: Intersections of halfspaces (dashed area) with normal vectors (red and blue arrows) at boundaries for (a) successful demonstrations $\mathbb{H}_D$ and (b) failed demonstrations $\mathbb{H}_F$. (c) Measuring the potential difference among two demonstration sets.

$$Q(s,a;r) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \omega^T \phi(s_t) | s_0 = s, a_0 = a\right] = \omega^T \mu(s,a), \tag{4.7}$$

where $\mu(s,a) = \mathbb{E}[\Sigma_{t=0}^{\infty} \gamma^t \phi(s_t) | s_0 = s, a_0 = a]$ is the function of feature expectation. Plugging this expression into Eq. 4.3, we derive the following inequity:

$$\omega^T(\mu(s,\pi(s)) - \mu(s,a)) \geq 0, \forall s \in S, a \in A. \tag{4.8}$$

The solution of ω in Eq. 4.8 forms the intersection of halfspaces. Following Eq. 4.8, we constitute the set of halfspaces for the demonstration set. For successful demonstrations, we denote their halfspaces $\mathbb{H}_D$ as:

$$\mathbb{H}_D : \{\omega | \omega^T(\mu(s_t,a_t) - \mu(s_t,a')) \geq 0, \forall s_t, a_t \in D, a' \in A\}, \tag{4.9}$$

and $\mathbb{H}_{D^\pi}$ for current policy as:

$$\mathbb{H}_{D^\pi} : \{\omega | \omega^T(\mu(s_t, \pi(s_t)) - \mu(s_t, a')) \geq 0, \forall s_t \in D, a' \in A\}. \tag{4.10}$$

The corresponding definition on failure demonstration set $\mathbb{H}_F$ and $\mathbb{H}_{F^\pi}$ can be applied in the same way.

Further, the definitions above have a corresponding set of normal vectors, denoted as N, w.r.t its boundary faces of halfspaces. For example, the normal vector set of $\mathbb{H}_{D^\pi}$ is computed by:

$$N_{\mathbb{H}_{D^\pi}} = \left\{ \frac{\mu(s_t, \pi(s_t)) - \mu(s_t, a')}{||\mu(s_t, \pi(s_t)) - \mu(s_t, a')||_2}, \forall s_t, a_t \in D, a' \in A \right\}. \tag{4.11}$$

Similarly, normal vector sets $N_{\mathbb{H}_D}$, $N_{\mathbb{H}_F}$, and $N_{\mathbb{H}_{F^\pi}}$ can be obtained as well. The graphical illustration on those calculations can be referred to Fig. 4.2.

Given the normal vectors, we define the potential term $\Delta U(D, F; r)$ as:

$$\Delta U(D, F; r) = \text{sim}(N_{\mathbb{H}_D}, N_{\mathbb{H}_{D^\pi}}) - \text{sim}(N_{\mathbb{H}_F}, N_{\mathbb{H}_{F^\pi}}), \tag{4.12}$$

The $\text{sim}(\cdot, \cdot)$ function over two normal vector sets can be generalized from one over two normal vectors. For instance, the vector similarity function is defined by:

$$\text{sim}(n_1, n_2) = (1 + \cos(n_1, n_2))/2, \tag{4.13}$$

then similarity over two normal vector sets is computed by fulfilling the following steps: i) Get the most similar vector pair selected from two sets w.r.t Eq. 4.13. ii) Compute the similarity score and remove them from their own sets. iii) Continue i) and ii) until one of the two sets is empty. iv) The average of all similarity scores obtained is considered as the similarity between two normal vector sets.

4.3.3 Algorithm

Algorithm 1 BIRLF algorithm

Collect demonstrations (e.g., trajectory sets for grasping) D and F
Initialize network parameters θ
Randomly initialize ω as an unit vector $||\omega||_2 = 1$
$\phi(s)$, $\mu(s,a)$, $\pi(a|s)$:= PolicyIteration(θ, ω)
while not done **do**
    Randomly sample $\tilde{\omega}$ from the neighborhood of ω
    Compute $\tilde{Q}_{D^\pi}(s_t, \pi(s_t)), \forall s_t, a_t \in D$ by Eq. 4.7
    Compute $\tilde{Q}_D(s_t, a_t), \forall s_t, a_t \in D$ by Eq. 4.7
    if $\exists s_t, a_t \in D$, $\tilde{Q}_{D^\pi}(s_t, a_t) < \tilde{Q}_D(s_t, a_t)$ **then**
        $\tilde{\phi}(s)$, $\tilde{\mu}(s,a)$, $\pi(\tilde{a}|s)$:= PolicyIteration(θ, $\tilde{\omega}$)
        Compute potential $\Delta U(D, F; r)$
        With probability $\min\{1, \frac{P(\tilde{r}|\Psi)}{P(r|\Psi)}\}$, $\omega := \tilde{\omega}$, $\pi := \tilde{\pi}$
    else
        With probability $\min\{1, \frac{P(\tilde{r}|\Psi)}{P(r|\Psi)}\}$, $\omega := \tilde{\omega}$
    end if
end while
Output θ and ω

Given the proposed Bayesian framework, we apply Eq. 4.4 for MCMC sampling on reward function. The PolicyWalk algorithm is adopted on sampling feature weights ω given a learned $\phi(s)$. The feature functions $\phi(s)$ and feature expectation function $\mu(s,a)$, we use function approximators, deep networks, to estimate the true values. The policy learning is achieved by using Deep Deterministic Policy Gradient (DDPG) [DCH16]. The network architecture is depicted in Fig. 4.3, which composes four modules. $\phi(\cdot)$ takes input state s and outputs the state feature. Actor module $A(\cdot)$ outputs the policy distribution from $\phi(s)$. In addition

to the original DDPG, the network also outputs feature expectation $\mu(s,a)$ computed by the critic module $C(\cdot)$ which estimates the value aggregating the action from $A(\cdot)$ and contextual feature output from the mapping module $M(\cdot)$. The value function $Q(s,a)$ hence can be obtained by the inner product of ω and $\mu(s,a)$. Given fixed feature weights ω, the network is updated by policy gradient. The process of learning can be regarded as policy iteration since ω is sampled from the Markov chain. Our proposed BIRLF algorithm is summarized in Alg. 1.

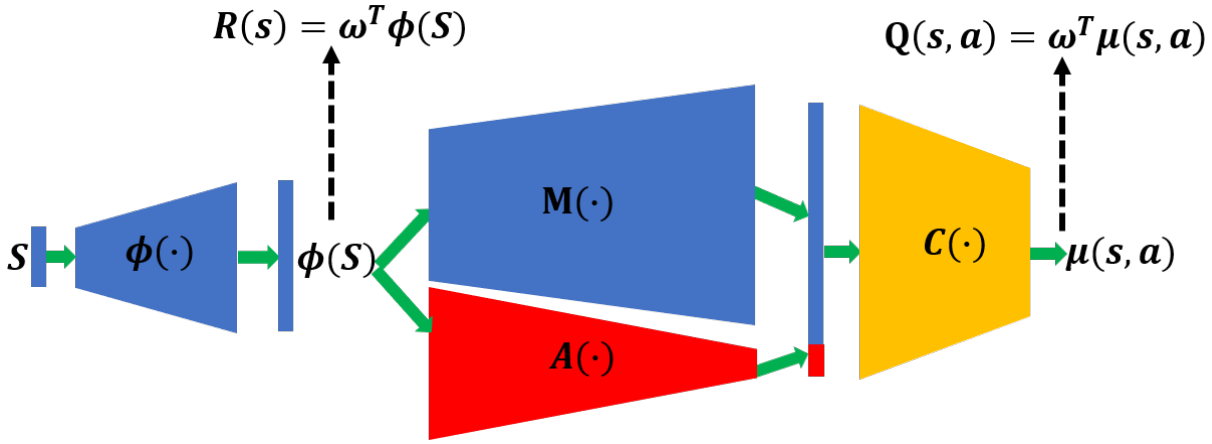


Figure 4.3: The network architecture for the proposed BIRLF algorithm. $\phi(\cdot)$ is the state feature module, $M(\cdot)$ the mapping module, $A(\cdot)$ the actor module, and $C(\cdot)$ the critic module.

4.3.4 Alternative

Eq. 4.12, in practice, is unstable at the early stage of learning. The difference term $\Delta U(D,F;r)$ would fluctuate around zero due to limited amount of data, resulting in a biased estimation of the reward function. We resolve this issue by incorporating an additional term to stabilize the posterior distribution by ensuring the summation of the potential is above zero, that is:

$$P(\Psi|r) \propto \exp\left(\Sigma_{d\in\Psi,t} Q(s_{d,t},a_{d,t};r) + \beta_1 \Delta U(D,F;r) + \beta_2 |\Delta U(D,F;r)|\right), \tag{4.14}$$

where an absolute term $|\Delta U(D, F; r)|$ is appended to Eq. 4.12. In our evaluation, we initialize $\beta_2 \geq \beta_1$ and apply simulated annealing to let β_2 gradually approach β_1. In the worst scenario where the MCMC chain incorrectly treats F as the successful demonstration set ($\Delta U(D, F; r) < 0$), the last two terms in Eq. 4.14 will cancel each other ($\beta_1 = \beta_2$) and result in the original posterior identical with BIRLF in Eq. 4.4. Utilizing this alternative would guarantee the task performance of our method by incorporating the standard BIRL scheme as the baseline.

4.4 Task Evaluations

4.4.1 VRGym Experimental Setup

We setup the virtual environment in VRGym for demonstrations collection and task evaluation. As shown in Fig. 4.1, the behavior control of the virtual Baxter robot is realized by running MoveIt motion planning package on ROS backend. The human control of the robot arm is enabled by teleoperation. The Oculus Rift VR controllers are used for robot grippers motion. The virtual scene is setup for grasping task where contains a virtual Baxter robot and a randomly spawned object to be grasped on top of a table. The Baxter robot is initialized with the untucked pose. When human performer is teleoperating the robot grippers, the movement of robot arm is confined within a prescribed range of space as long as the robot's joint limits are satisfied.

4.4.2 Interactive Grasping Task

The goal of this virtual grasping task is to let the robot generate a stable grasping trajectory of provided objects. Human performers are put in the loop of virtual robot's embodiment. The motion flexibility is constrained by robot's physical joint limits. The direct benefit of such design bridges the gap between different embodiments meanwhile results in more chance

of making failed demonstrations.

4.4.3 Demonstration Data Collection

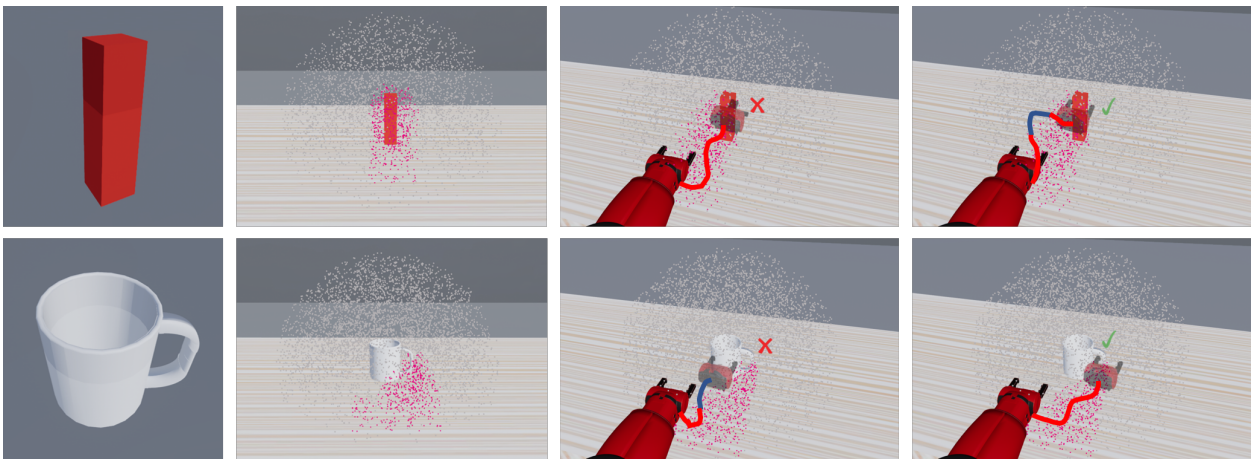


Figure 4.4: 1st column: target objects: a cuboid and a mug. 2nd column: visualization of the set S_G (in red dots) and the set $S_{D\setminus G}$ (in gray dots); see text for details. 3rd column: failure cases. 4th column: success cases. Blue trajectories gain less rewards than the red ones.

Human subjects are tasked in a training phase to get familiar with the teleoperation. They are equipped with a pair of Oculus Rift controllers and asked to control the virtual Baxter robot's gripper from robot's first person view. VGym physics engine plays the role of evaluating a performed grasping trajectory. The grasping attempt is successful only if the object is stably grasped. All grasping trajectories, including both successful and failed ones, are collected on-the-fly. In summary, 120 demonstrations are collected for grasping of two types of objects: cuboids (87 successful and 33 failed) and mugs (76 successful and 44 failed). Graphical examples are illustrated in Fig. 4.4.

4.4.4 Training Routine

After collection human demonstrations, the virtual Baxter robot is spawned in the same location to perform the grasping while updating its own grasping policy. To satisfy the purpose, an IRL module is developed on the ROS backend along with the motion planner, see Fig. 4.1, to autonomously perform the robot grasping.

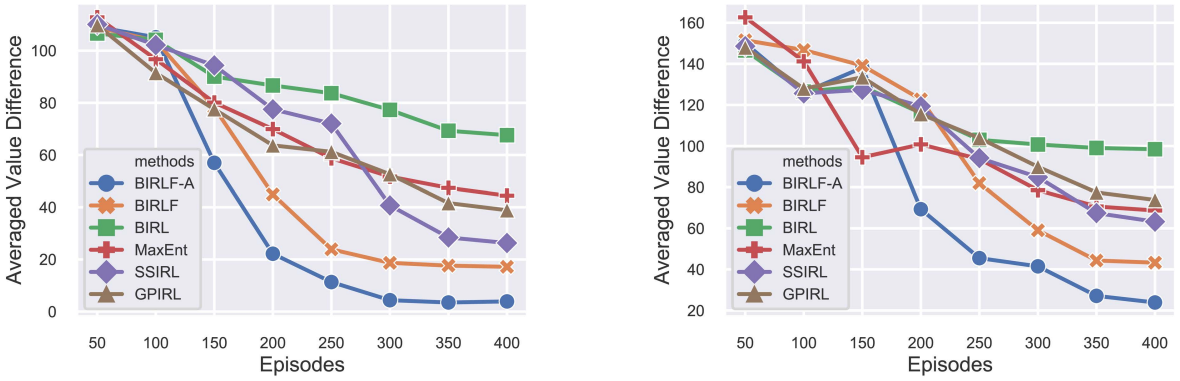
The state space of the virtual environment is defined as $\{s_w, s_{t_1}, s_{t_2}\} = \{(x_w, y_w, z_w), (x_{t_1}, y_{t_1}, z_{t_1}), (x_{t_2}, y_{t_2}, z_{t_2})\}$, a 9-D vector where w, t_1 and t_2, where w, t_1 and t_2 are denoting griper *wrist*, *left tip end*, and *right tip end* respectively. The action space is parameterized using a 4-D vector $(a_x, a_y, a_z, a_t) \in [-1.0, 1.0]^4$: the first three fields denote the normalized motion along 3D Cartesian axis in the global frame, the last field a_t is the 1-D motion of two gripper tips.

In one of training episodes, the robot sends its current state s through VRGym-ROS bridge received by the IRL module. By estimating the reward value as well as updating the policy, The IRL module generates an action a from its policy and sends it back to the virtual robot. Once received the action message, the robot end-effector will execute the action with assist of motion planning and obtains the next state s'. This training process will continue until the IRL reaches the convergent status.

For one rollout, it will be terminated if one of the following four conditions are satisfied: i) a successful grasping, ii) the object target falls down, iii) robot arm touches things other than the object, or iv) the action sequence is too long, such as more than 100 steps in our setting. The grasping validation process will be invoked to determine the grasping result.

4.4.5 Experiment Results

The experiment results are detailed by using both cuboid and mug objects. he ground truth reward function $r(s)$ in the virtual environment is defined for different IRL methods comparison as:



(a) Cuboids

(b) Mugs

Figure 4.5: Averaged value difference (AVD) for different IRL methods.

$$r(s) = \begin{cases} 0.02, & s \in \Omega_G \\ 0.01, & s \in \Omega_{D \setminus G} \\ -0.01, & \text{otherwise}, \end{cases} \tag{4.15}$$

where Ω_G is the convex hull formed by the success frames in the demonstrations, that is the set S_G of $\{s_{i,T_i} : \forall (s_{i,T_i}, a_{i,T_i}) \in D\}$ depicted as red dots in Fig. 4.4. And $\Omega_{D \setminus G}$ is the convex hull of the set $S_{D \setminus G}$ of $\{s_{i,j} : \forall (s_{i,j}, a_{i,j}) \in D, \forall j \neq T_i\}$ depicted as gray dots in Fig. 4.4. The design of the ground truth reward function results in higher accumulated reward for the red trajectories in Fig. 4.4 while lower ones for blue segments.

Baselines for comparison are state-of-the-art IRL methods:

- **BIRL [RA07].** Bayesian IRL presents reward estimation under uncertainties using a probability distribution.
- **MaxEntIRL [ZMB08].** Maximum Entropy IRL is another probabilistic approach to model the reward function. It provides a well-defined, and globally normalized distribution by matching the feature expectation. The state visiting frequencies are empirically estimated from sampled trajectory rollouts.

- **SSIRL [VGL13].** Semi-supervised IRL relaxes the assumption that not requiring all demonstrations to be made from the expert (allowing failed demonstrations). This approach is inspired by semi-supervised SVM [BD99] which treats feature expectations as labels and optimizes the objective following the minimax scheme. The feature expectations are empirically estimated by leveraging sampled trajectory rollouts.

- **GPIRL [LPK11].** Gaussian Process IRL treats the reward as a nonlinear function. The reward is approximated through Gaussian process. The reward structure is determined by the kernel function. The state and action visiting frequencies are empirically estimated from the sampled trajectory rollouts.

The notations for our proposed IRLF method using Eq. 4.12 as BIRLF and the one using Eq. 4.14 as BIRLF-A. The hyper parameters for evaluation are set as: $\alpha = 1.0$, $\beta = 10^2$ for BIRLF, and $\alpha = 1.0$, $\beta_1 = 10^2$, and $\beta_2 = 10^3$ for BIRLF-A. The annealing is applied every 50 episodes to linearly decrease β_2 from 10^3 to 10^2.

As for evaluation metrics on reward function, we adopt the following two measurements for all IRL methods:

- **Average Value Difference (AVD).** This metric measures the difference of the averaged returns between successful expert demonstrations and the ones generated by the IRL methods.

- **Mean Squared Error (MSE).** This metric calculates relative reward difference. Each time step of trajectories from the successful expert demonstrations and the IRL generated ones are calculated.

Fig. 4.5 illustrates the results of the proposed method BIRLF along with other baseline methods. For methods that consider the failed demonstrations, such as BIRLF and SSIRL, we create a shared demonstration dataset by setting $|D| = 2|F|$. For methods that only adopt the expert demonstrations, such as BIRL, MaxEntIRL and GPIRL, the demonstration

dataset consists only successful demonstrations. To make fair comparison across different methods, we use the same size of demonstration dataset for those methods. After finishing 400 training episodes, as indicated in Fig. 4.5, our proposed methods, BIRLF and BIRLF-A converge and attain the minimum AVD.

In addition, we evaluate the MSE results of different IRL methods on grasping two objects. The results are summarized in Table 4.1. The MSEs are calculated after 600 episodes of training to guarantee all the methods could reach the convergence. The performance are presented by keeping the size of full demonstration set ψ fixed while setting different size ratio between the successful and failed demonstrations $|D| : |F|$. For methods only take expert demonstrations, such as BIRL and MaxEntIRL, only D is provided for training. As indicated in Table 4.1, our proposed approaches BIRLF and BIRLF-A in general achieve the lowest MSE compared to other baseline methods. Further, we notice BIRLF-A basically outperforms BIRLF which verifies the effectiveness of our proposed alternative scheme.

4.5 Conclusion

Given the purpose of learning an interactive skill from human demonstrations. As demonstrations are usually hard to be obtained, learning an interactive policy from experts demonstrations becoming the issue of learning from "small data". Our work incorporates the failed demonstrations that we argue are valuable for agent to make use of. Our solution is realized by proposing the Bayesian IRLF for grasping task in continuous state/action spaces. The halfspaces extracted from policy optimality are leveraged to model the difference between the successful and failed demonstration set. The proposed BIRLF method is evaluated in the virtual robot grasping setup. Compared to other existing IRL and IRLF baselines, our method achieves the state-of-the-art performance and verifies the effectiveness of incorporating failed demonstrations to exploit "small data".

Table 4.1: Quantitative evaluation (MSE) of different IRL methods on grasping cuboids and mugs. The performance is measured across different ratio configurations $|D| : |F|$.

Methods	$\|D\| : \|F\|$					
	1 : 1		2 : 1		4 : 1	
	Cuboid	Mug	Cuboid	Mug	Cuboid	Mug
BIRLF-A	**0.90**%	**3.95**%	**0.72**%	**2.44**%	0.84%	**2.53**%
BIRLF	1.30%	5.88%	0.78%	3.12%	**0.82**%	3.93%
BIRL	8.00%	9.67%	7.15%	8.42%	7.20%	9.09%
SSIRL	2.17%	5.98%	1.15%	5.19%	1.18%	5.26%
MaxEntIRL	4.33%	6.10%	3.99%	5.77%	4.13%	6.04%
GPIRL	4.81%	7.20%	3.65%	6.29%	3.79%	6.84%

CHAPTER 5

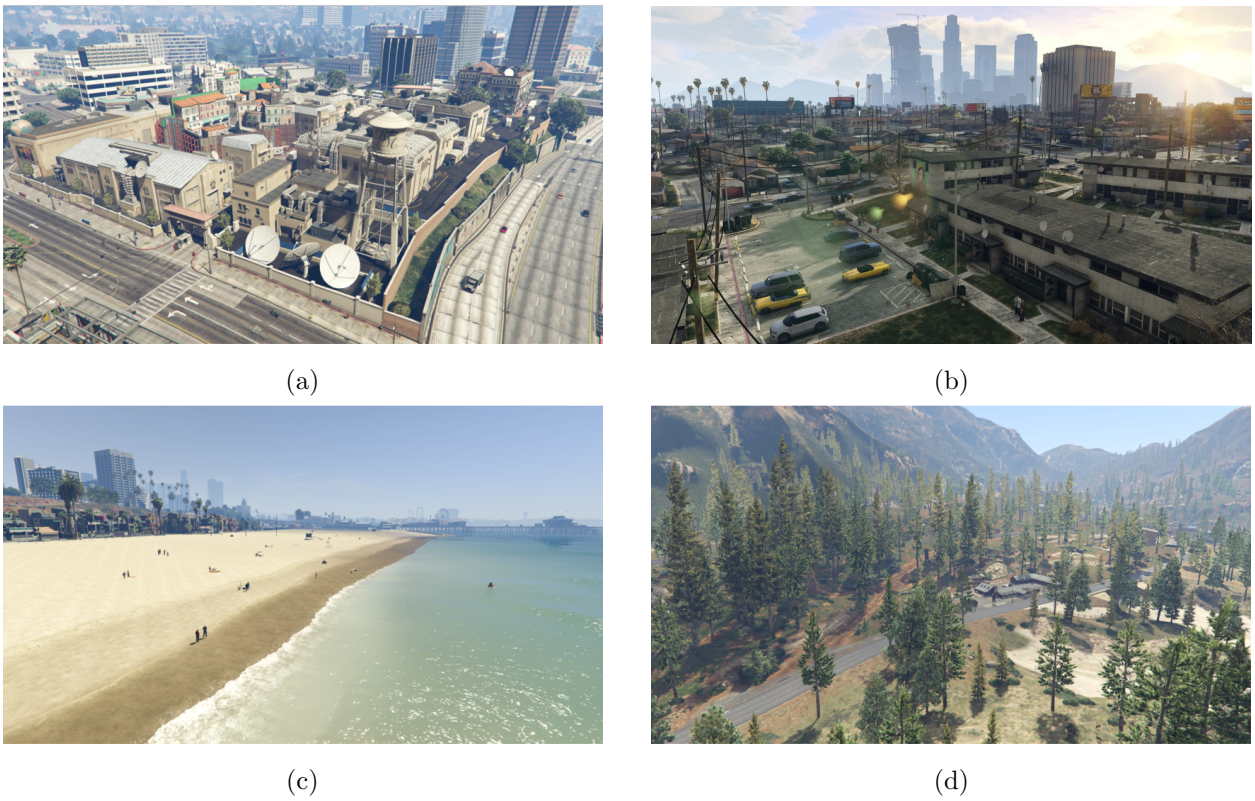
Learning from Interactions: Large Scale Behavioral Predictions

In the previous chapters, we cover the aspects of robot learning physical interactions in both real world and virtual environments. In addition to physical interactions, learning social interactions are another crucial category that are indispensable to grow a general purpose agent. Research of social interactions would potentially solve many existing challenges. For example, an autonomous vehicle driving on road has to correctly infer other agents (vehicles and pedestrians) motions or behaviors such that: i) safety requirements could be guaranteed, ii) more complete reasoning of theory-of-mind (TOM) would be generated, and iii) advanced sensing of social norms would be derived for cooperative navigation. Hence, we treat robot learning from the multi-agent social interactions as a corner stone to boost the development of autonomous agents.

On the other hand, As physical and photo realistic simulation platforms, such as VRGym, provides the seamless interactive tasks compared to the real world, it would be promising in studying related issues of social interactions in the simulation environments. Specifically, the problem we are interested is the behavioral predictions problem in multi-vehicle driving scenarios. Considering when traffic rules are not explicitly given, correctly predicting vehicles future trajectory is critical for collision avoidance. To flexibly control and scale the variants including vehicles behaviors as well as road conditions, we leverage a prevalent computer game platform — *Grand Theft Auto (GTA)* to automatically synthesize multiple vehicles driving scenarios in a urban scale environment.

5.1 GTA Playground

5.1.1 Urban Environments



(a) (b)

(c) (d)

Figure 5.1: Exemplar GTA game scenes captured in the game process. (a) an Urban highway traffic system. (b) an Urban local street. (c) a beach ocean scene. (d) a countryside scene. All the scenes are generated through in-game photo realistic rendering.

The GTA game provides an open world that was modeled on area of Los Angeles. Early developments of this game are constituted by in-game render and design. To construct photo realistic 3D urban scenes, field trips are made to collect city photo and video footage. Fig. 5.1 illustrates the typical game scenes which are photo realistic and fully interactive in the large scale.

5.1.2 Visual Ground Truth Generation

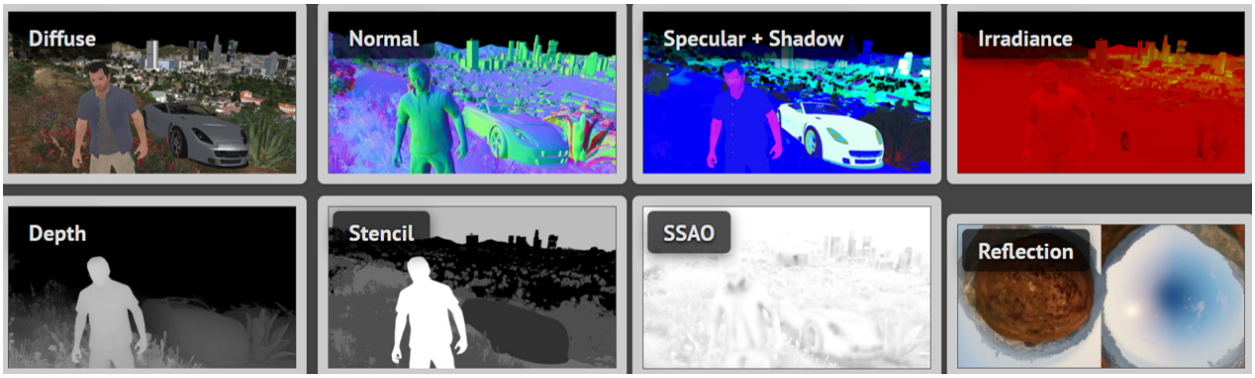


Figure 5.2: G-Buffer shading results on GTA game. The pixel shader fetches data from different combination of buffers and compute the final shading value of the pixel in HDR. Visual ground truth such as depth and instance segmentation are obtained by hooking with the rendering process.

Given the photo realistic scene, we design and develop a pipeline to collect visual ground truth information along with the game's rendering process. The GTA maintains a set of buffers called *G-Buffer* which separately draws shading related information to calculate the meshes shading. Those buffers are eventually combined to compute the final shading of each pixel. We develop the modding script and hook with the rendering process to obtain various buffer shading ground truth as follow:

- **Diffuse map.** It stores intrinsic color of the mesh that depends on the material property of the mesh rather than influenced by the lighting.
- **Normal map.** It stores normal vector information for each pixel.
- **Specular map.** It stores information containing specular and reflections.
- **Irradiance map.** It stores the irradiance of each pixel receiving from the sun light.

- **Depth map.** It stores the distance of each pixel from the camera view. Opposite to the common used depth map, depth map in GTA game applies the logarithmic Z-buffer and results in "close white and far dark" rendering.

- **Stencil map.** It stores information regarding each mesh entity drawn. Different IDs are assigned to all pixels of one category of meshes.

Fig. 5.2 demonstrates the G-buffer shading results on one game frame. For extraction of visual information, our developed modding scripts capture the rendered depth and stencil information from the G-buffer. Along with the RGB stream, those are collected as raw visual data serves multiple interactive tasks.

5.1.3 Game Control for Interactive Tasks

Figure 5.3: Realistic agent interactive scenes are synthesized in GTA game. By developing modding scripts, behaviors of game agents are accessible for control hence increases the diversity of interactions.

We develop a set of script pluggings utilized to get fully control of them. In GTA game, two types of agents involved are *vehicles* and *pedestrians*. The modding scripts, developed in C++, are hooked with the main game process. Those scripts are designed in multiple purposes such as data acquisition, agent motion control and environmental variable control. Given this flexibility, we are able to extend our study on learning from interactions to a large scaled simulator, especially highlighting the social aspects.

(a) (b) (c)

Figure 5.4: Sketch of autonomous driving in GTA game. (a) Weather control for road conditions. (b) 3D bounding boxes information extracted for on-scene agents. (c) Heatmap predictions for driving maneuvers.

Fig. 5.3 illustrates several examples where all the interactive scenes are synthesized by executing the modding scripts. Specifically, we spawn and assign the targeted agents with expected motions to afford certain types of social behaviors. According to Fig. 5.3, the scripted agents could behave naturally with other scene objects or agents. This verifies our thinking of leveraging a game platform to investigate realistic social interactions. In addition, we showcase a autonomous driving scene according to Fig. 5.4. We script a vehicle agent with fully autonomous ability navigating on roads. The driving policies are derived according to the ground truth internal game states. While driving on road, there are several game properties can be modified or monitored. Fig. 5.4(a) illustrates the flexible weather control to add uncertainty on the driving quality. Fig. 5.4(b) demonstrates the ground truth context information of extracting agents 3D bounding boxes (in green cuboids). Fig. 5.4(c) presents an auxiliary information on driving decisions by plotting the driving intentions

in terms of trajectory heat maps in real-time. Given the modding development on GTA game platform, we are synthesizing and inspecting multi-vehicle driving scenarios to afford reasonable navigating interactions.

5.2 Vehicle Driving Interactions — Safety Critical View

5.2.1 Three Streams Modeling Approaches

In vehicle driving, one of the most critical requirements would be safety. Correctly predicting on-scene vehicles intention would largely solve the safety issues since knowing other vehicles intentions provides more contextual information for decision making on driving maneuvers. In our work, we represent the intentions as the form of vehicles future trajectories. And our primary objective is to accurately predict multi-vehicle future trajectories as well as guarantee the safety. However, this problem is challenge due to intricate interactions exhibited in multi-agent systems, especially when it comes to collision avoidance. To resolve this issue, we propose to unsupervisedly learn an intermediate contextual information denoted as "congestion pattern" meanwhile devising a novel "Sense-Learn-Reason-Predict" framework for trajectory predictions. It integrates three modeling approaches [RPH19]

Physics Based Trajectory Prediction The physics based approaches adopt physics modeling [Zhu91] are defined in "Sense-Predict" framework. The modeling involves direct forward simulating pre-defined and explicit dynamic models based on Newton's laws of motion. However, when face the noisy real-world sensory data, physics based approaches are too brittle to guarantee descent performance. It becomes more challenge coping with multi-agent scenarios where different agents may possess different dynamic models [LJ05].

Pattern Based Trajectory Prediction The pattern based approaches [TIT93] learn function approximators directly from the statistical characteristics of the data, following a

"Sense-Learn-Predict" framework. The principle of this stream of works provide available large scale datasets. Whereas those family of methods suffer from poor interpretability and easy to overfit on a large space of parameters.

Planning Based Trajectory Prediction The planning based approaches [BG04] model this problem by assuming the rational agents stand and minimizing different forms of costs. In this family, forward planning and inverse planning are two categories following the "Sense-Reason-Predict" scheme. However, this normative perspective constrains what agents ought to do which may not truly reflect read-world scenarios since human decision making often deviates from extreme rationality [Ste96, KT73].

Those streams of thoughts have been mostly developed individually in literature. On the other hand, we argue that those works are not conflicting. We are thinking of an appropriate approach to effectively integrate them and form a new "Sense-Learn-Reason-Predict" framework. Regarding this, we propose to unsupervisedly learn an intermediate representation that injects a better inductive bias for *pattern based* approaches. This representation learned as a contextual information that incorporates both rational agent assumption in *planning based* approaches and physical constraints in *physics based* approaches. In addition, we would pursue a multi-stage learning process that affords better interpretability and obtain collision free trajectory predictions.

5.2.2 Approach Overview

In tackling the problem of *collision free multi-agent* trajectory predictions, the pattern based approaches aims at regressing the future trajectory directly from the training data by fitting various contextual cues such as discrete cell [KW98, THK09, KSM13, BCA16, MCK18], continuous position [JDH11, FLG15, KMS17], and graph based representations [LFH03, VFL09, CLH16]. Those first order modeling approaches often lack semantic based representations. The higher order pattern based approaches incorporate some context in terms of relations

modeling [KKK17, AL17, PKK18, DCS19, DT18b, DT18a, DLL19, LMZ19, TS19, ZXM19, SAS19], but they present limited ability in emerging collision free trajectory predictions or related verification. On the contrary, our approach models the collision free trajectories by incorporating context cues of *congestion* as well as the simulating verification.

Our proposed approach possess three aspects of merits over prior methods. First, the way our contextual cues (congestion) is learned via fully unsupervised graph based generative learning. The graph nodes are vehicles, and graph edges measure the spatial relation between two vehicles. This scheme subsumes the Social Force (SF) adopted in physics based approaches. The context is learned through graph modeling toward a holistic view. Further, the relative distance between two vehicles is implicitly encoded as cost function which characterize the planning based modeling of the environment. The contextual cue we learned is represented as a Gaussian Mixture Model (GMM). It unsupervisedly captures the various modes in congestion patterns and reduce parameter space to speed up the training process.

Second, the "Sense-Learn-Reason-Predict" framework is decoupled in terms of two subsequent processes: i) A teacher model *senses* and *learns* the contextual patterns, a form of knowledge, pursuing a pure pattern based approach. ii) A student model, instead of directly learns from observations, *reasons* the knowledge by minimizing a cost compared to what the teacher model learns, pursuing a planning based approach. The student model predicts the future trajectories simultaneously along with the reasoning process. There are fundamental differences between our approach and GAN based models [GJF18, SKS19, KSM19]. The GAN models require both the discriminator and generator focus on predicted trajectory without explicit context modeling. While in our approach, the student model generates the future trajectories under the constraints by explicitly matching the contextual cues provided by the teacher model. The major benefit of this design results in an inductive bias to speed up the learning on generating the trajectories.

Third, the contextual matching process is achieved by formulating an optimization problem that bridges the learning of congestion patterns and collision free trajectory predictions.

The objective is to constrain the generated trajectories by matching a learned congestion pattern distribution. Utilizing the variational parametrization, an upper bound is derived to make the optimization problem computational tractable.

Our proposed model is realized as an "encoder-pooling-decoder" architecture which is compatible with many existing works, e.g. [AGR16, DT18a]. According to our evaluations, the proposed method is more superior than others on collision trajectory predictions, especially the synthetic dataset collected on GTA game platform. Further, our method demonstrates state-of-the-art prediction performance on one public benchmark NGSIM US-101 highway dataset [CH07].

5.3 Collision Free Trajectory Prediction

In this section, we detail our proposed method pursuing congestion aware multi-vehicle trajectory predictions. Fig. 5.5 presents the overview of the method. We first introduce the problem definition of multi-vehicle trajectory prediction by following [AGR16]. Second, we bring up the learning of congestion patterns for multi-vehicle driving. It details how the teacher model learns the contextual cues. Then we formulate an optimization problem of congestion pattern matching. It helps the student model to reason about teacher model's contextual information. This optimization problem is jointly solved with trajectory generation by training a generic encoder-pooling-decoder model for collision free trajectory predictions.

5.3.1 Problem Definition

In multi-vehicle trajectory predictions, the goal is to predict the future trajectories of all vehicles given their observed trajectory histories. $\{\zeta^m, m = 1, ..., n\}$ is the set of trajectories for all n vehicles. The position of the mth vehicle at time step t is denoted as a local 2D coordinates $\zeta_t^m = (x_t^m, y_t^m)$. In a time span of $t = 1 : T_h$, the observed history trajectories are $\zeta_h = \{\zeta_{t=1:T_h}^m\}$, and the future trajectories to time step T_p is $\zeta_p = \{\zeta_{t=T_h+1:T_p}^m\}$. The

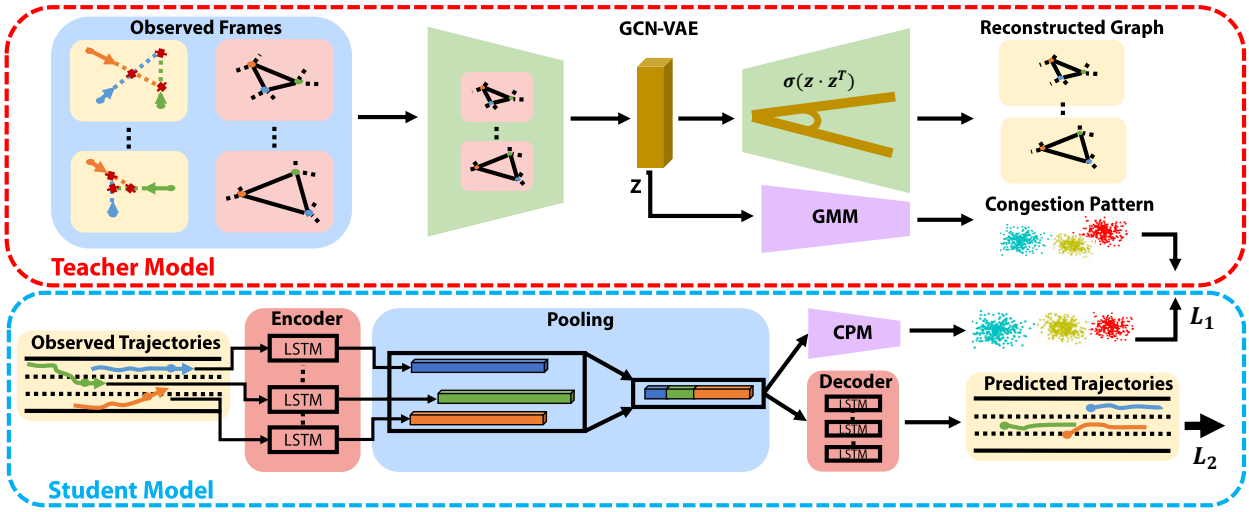


Figure 5.5: The proposed architecture for congestion-aware multi-agent trajectory prediction. The teacher model (top) is composed of the frame-wise graph construction module, and the GCN-VAE graph encoder and decoder. The learned latents are passed to a GMM and used to unsupervisedly learn the multi-modal congestion patterns. The student model (bottom) makes prediction based on the observed trajectories. It follows the encoder-pooling-decoder design and uses the CPM module to match the teacher's congestion patterns. The loss terms L_1 and L_2 are defined in 5.6 and 5.9, respectively.

multi-vehicle trajectory prediction then be formulated as estimating the posterior probability $P(\zeta_p|\zeta_h)$.

5.3.2 Congestion Patterns

Multiple vehicles in the navigation scenarios convey meaningful contextual information to model the social interactions. We propose an unsupervised way of learning the congestion patterns that not only implicitly comprise vehicle positions and intentions, but provide contextual cues about safety critical information such as collisions. Congestion patterns [XYL13, XVZ18] are mostly been described as the dynamic equations of the vehicle spatial relations or clustering that groups the observations into different categories. We

argue that those conventional definitions are sensitive to agent numbers and physical properties. It could be non-trivial to exactly identify the congestion patterns by applying a set of pre-defined rules. On the other hand, we propose to learn the congestion patterns through graph based generative learning given the trajectory history $o = \zeta_h$. The congestion patterns are treated as hidden information that implicitly incorporates various modes by learning a Gaussian Mixture Model.

Graph Representation We model the congestion patterns by embedding the physical constraints in the following manner. Given two vehicles $u, v \in \mathcal{V}_t$, the graph $A_t = (\mathcal{V}_t, \mathcal{E}_t)$ at time frame $t \in \{1, .., T_h\}$ is created by their 2D coordinates (x_t, y_t) and velocities $(\dot{x}_t, \dot{y}_t)$. The graph adjacency matrix $\mathcal{E}_t = \{\mathcal{E}_t^{uv}\}, u, v = 1, .., n$ is defined as follow:

$$\mathcal{E}_t^{uv} = \mathcal{E}_t^{vu} = \begin{cases} 1/t_c^{uv}, t_c^{uv} > 0 \\ 0, t_c^{uv} = 0 \end{cases}, \tag{5.1}$$

where $t_c^{uv} = \max(-\frac{\Delta^{uv}x \times \Delta^{uv}\dot{x} + \Delta^{uv}y \times \Delta^{uv}\dot{y}}{\Delta^{uv}\dot{x}^2 + \Delta^{uv}\dot{y}^2}, 0)$ is the estimated collision time. By intuition, a larger weight reflects a higher cost of collision which are all encoded in the graph representation.

Generative Learning We perform the unsupervised learning of the congestion pattern by leveraging Variational Auto Encoders (VAEs) [KW13] which learns the latent congestion pattern z through parametrization. In our case, we adopt the graph VAE method embodied as Graph Convolution Networks (GCN) [KW16] by reconstructing the graph representation A_t.

Gaussian Mixture Model Given the multi-modal nature of congestion patterns, we utilize a Gaussian Mixture Model (GMM) to account for various modes. To fulfill this, we treat the congestion pattern z as another random variable [DMG16, JZT16, YCL19] and

build the following GMM:

$$\mathcal{Q}(z) = \sum_{i}^{M_{\mathcal{Q}}} \lambda_i q_i(z), \tag{5.2}$$

where the mixture component $q_i(z)$ is a Gaussian distribution, λ_i is the mixture weight, and $M_{\mathcal{Q}}$ is the hyperparameter denoting the mixture number. The mixture model is learned by applying the stochastic EM [Cel85] algorithm. Since congestion pattern z is learned from the observation o, the mixture model is denoted as $\mathcal{Q}(o)$ in the following content.

5.3.3 Matching Congestion Patterns

After learning the congestion patterns by the teacher model, the student model aims to generate the trajectory simultaneously match the congestion patterns learned by the teacher model. It is the objective to pursue the collision free trajectory predictions. Noting the student congestion pattern as $\mathcal{P}(o)$, the congestion pattern matching can be formulated as measuring the difference between two probability distributions:

$$\mathbb{D}_{KL}(\mathcal{P}(o) \| \mathcal{Q}(o)). \tag{5.3}$$

Given multiple modalities of $\mathcal{Q}(o)$, $\mathcal{P}(o)$ also follows a Gaussian mixture, $\mathcal{P}(o) = \sum_{j}^{M_{\mathcal{P}}} \omega_j p_j(o)$, where the mixture number $M_{\mathcal{P}}$ does not have to be same with $M_{\mathcal{Q}}$.

For Eq. 5.3, there is no analytical solution to calculate the KL divergence between two mixture distributions. We propose to solve this by optimizing a variational upper bound. Inspired by [HO07, XGN19], our variational parametrization is performed in the following. We decompose the mixture weights $\omega_j = \sum_{i}^{M_{\mathcal{Q}}} \alpha_{ij}$ and $\lambda_i = \sum_{j}^{M_{\mathcal{P}}} \beta_{ij}$ and rewrite the objective in Eq. 5.3 as:

$$\begin{aligned}\mathbb{D}_{KL}(\mathcal{P}(o)\|\mathcal{Q}(o)) &= -\int \mathcal{P}(o)\log\frac{1}{\mathcal{P}(o)}\left(\sum_{i,j}\beta_{ij}q_i(o)\right)\\ &= -\int \mathcal{P}(o)\log\frac{1}{\mathcal{P}(o)}\left(\sum_{i,j}\frac{\beta_{ij}q_i(o)\alpha_{ij}p_j(o)}{\alpha_{ij}p_j(o)}\right).\end{aligned} \tag{5.4}$$

Applying Jensen's inequality, Eq. 5.4 can be transformed to:

$$\begin{aligned}\mathbb{D}_{KL}(\mathcal{P}(o)\|\mathcal{Q}(o)) &\leq -\int \mathcal{P}(o)\sum_{i,j}\frac{\alpha_{ij}p_j(o)}{\mathcal{P}(o)}\log\frac{\beta_{ij}q_i(o)}{\alpha_{ij}p_j(o)}\\ &= \sum_{i,j}\alpha_{ij}\mathbb{D}_{KL}(p_j(o)\|q_i(o)) + D_{KL}(\alpha\|\beta).\end{aligned} \tag{5.5}$$

Hence the objective of optimizing Eq. 5.3 is converted to minimize its upper bound:

$$\min_{\{p_j\},\alpha,\beta} L_1 = \sum_{i,j}\alpha_{ij}\mathbb{D}_{KL}(p_j(o)\|q_i(o)) + \mathbb{D}_{KL}(\alpha\|\beta). \tag{5.6}$$

The convergence of this optimization problem could be guaranteed in [HO07].

To solve Eq. 5.6, the parameters $\{p_j\}$, α, and β are iteratively optimized. Given fixed α and β, $\{p_j\}$ can be updated by minimizing:

$$\begin{aligned}&\min_{\{p_j\}}\sum_{i,j}\alpha_{ij}\mathbb{D}_{KL}(p_j(o)\|q_i(o))\\ &= \sum_{i,j}\alpha_{ij}\left(\mathbb{E}_{p_j(o)}[-\log q_i(o)] - \mathbb{H}[p_j(o)]\right).\end{aligned} \tag{5.7}$$

With $\{p_j\}$ learned, α and β are updated by the closed form solutions:

$$\alpha_{ij} = \frac{\omega_j\beta_{ij}\exp^{-\mathbb{D}_{KL}(p_j(o)\|q_i(o))}}{\sum_{i'}\beta_{i'j}\exp^{-\mathbb{D}_{KL}(p_j(o)\|q_{i'}(o))}}, \quad \beta_{ij} = \frac{\lambda_i\alpha_{ij}}{\sum_{j'}\alpha_{ij'}}. \tag{5.8}$$

The overall algorithm for the congestion pattern matching (CPM) is summarized in Alg. 2.

Algorithm 2 Congestion Pattern Matching (CPM)

1: Input: the learned congestion patterns $\mathcal{Q}(o)$
2: Initialize α_{ij} and β_{ij}
3: **while** not convergent **do**
4: Fix α_{ij} and β_{ij} and optimize $\{p_j\}$ using 5.7
5: Fix $\{p_j\}$ and update α_{ij} and β_{ij} using 5.8
6: **end while**

5.3.4 Trajectory Predictions

The student model jointly predict future trajectories and match the teacher model's congestion patterns. In Fig. 5.5, the student model comprises an encoder that encodes the observed trajectories, a pooling module models the vehicle spatial interactions and a decoder that recursively generate the future trajectories. Social features after the pooling module is taken to performing the congestion pattern matching, in Eq. 5.6, with the teacher model.

The proposed student model can be learned end-to-end by iteratively minimizing the congestion pattern matching loss in Eq. 5.6 and trajectory prediction loss:

$$L_2 = -\frac{1}{m}\sum_{m}\sum_{t=T_h+1:T_p} \log P(\zeta_{p_t}^m|\zeta_h^m), \tag{5.9}$$

where ζ_h^m and ζ_p^m are the observed and predicted trajectories.

Implementation Details The whole model composes of one teacher model and one student model. Both of them are implemented using neural networks. For the teacher model, it adopts the GCN-VAE architecture with latent dimension of 64. Then it is followed by fully connected layers to learn the congestion patterns in terms of a Gaussian mixture distribution. The training is performed unsupervisedly with a learning rate of 1×10^{-4}. For the student model, it is compatible with many existing encoder-pooling-decoder architectures [AGR16, DT18a]. The proposed one is implemented following [DT18a]. Both encoder

and decoder are embodied using LSTMs with fixed hidden dimension of 128. The *CPM* module is implemented as another deep Gaussian mixture model in terms of fully connected layers. We set the mixture number for both teacher and student model as a tunable hyper parameter. The learning rate for student model is 3×10^{-3}. Both models are developed in PyTorch libraries.

5.4 Method Evaluation

5.4.1 Datasets Specification

(a) Scenario 1

(b) Scenario 2

(c) Scenario 3

(d) Scenario 4

Figure 5.6: Sample top-views of the four scenarios in our GTA dataset.

GTA Dataset We leverage the GTA game platform to evaluate the collision avoidance which relies on physical realistic vehicle driving simulations. In this dataset, we collect vehicle driving trajectories under safety critical scenarios demonstrating rich vehicle interactions. The scenarios created by developing modding scripts can be categorized in four types (Fig. 5.6): i) highway vehicle driving where vehicles are following the flow, ii) local vehicle driving where vehicles overtaking can frequently happen, iii) intersection vehicle driving where traffic rules are not given on crowded driving scenes, and iv) crazy vehicle driving where some vehicles are scripted with aggressive behaviors that almost lead to collisions. For our experiments, we split the entire dataset including four categories into 3 folds for training and 1 fold for testing. The time span used for observations are set as 3s for all trajectories. Statistics of GTA dataset is summarized in Table 5.1.

Table 5.1: GTA dataset statistics.

Total Clips	Vehicle Trajectories	Highway Trajectories	Local Trajectories
3300	27813	18229	9584
Following Events	Overtaking Events	Collision Events	
7055	2300	890	

NGSIM Dataset We utilize the public available highway driving dataset NGSIM US-101 [CH07] to evaluate the accuracy of trajectory predictions. This dataset is presented as the standard benchmark to showcase the competitiveness of the proposed method. The NGSIM contains real highway traffic data stream that is often captured more than a time span of 45 minutes. We set the evaluation protocol similar with GTA dataset to inspect the performance.

5.4.2 Baslines and Metrics

We compare our proposed method with a set of state-of-the-art trajectory prediction baseline methods [DT18a, TS19, ZXM19, GJF18, AGR16, SLV18, DRT18, HE16, BPW18, BPL19] and inspect the following metric results:

Collision Rate It directly evaluates the performance of safety driving regarding collision avoidance. On GTA dataset, we calculate the collision rate by counting the collision events among the predicted trajectories and normalized by the total number to trajectories for all trials. The ground truth collision events would be automatically acquired by running the game simulation.

Root Mean Squared Error (RMSE) It estimates the accuracy of the predicted trajectories. For each predicting time horizon, from 1s-5s, we calculate the RMSE across predicted trajectories. For baseline methods that generate trajectories using GANs [ZXM19, GJF18], we sample k predicted results and select the "best" one w.r.t L_2 norm for evaluation.

5.4.3 Quantitative Results

Results on GTA Dataset Table 5.2 and 5.3 present the quantitative results for trajectory predictions. Our proposed method, denoted as *CF-LSTM*, achieves the best performance regarding both collision rate and RMSE on GTA dataset. The results demonstrate our method's strength in multi-vehicle trajectory predictions. In detail, our CF-LSTM shows the lowest error rates across all given game scenarios in Table 5.2. According to Scenario 4 where the vehicles may not be assumed of rational driving, every method results in the highest chance of collision in such challenging case. Further, in the cases of intersection driving as Scenario 3, we notice the collision chance becomes higher in crowded space (vehicles meet with each other) when no traffic rules are provided. As for RMSE, our CF-LSTM still obtains the best result across all the baseline methods. According to Table 5.3, the accuracy metric

Table 5.2: Collision rate (%) on GTA dataset

Methods	V-LSTM	CS-LSTM	S-GAN	CF-LSTM (Ours)
Scenario 1	4.219	3.086	3.372	**2.909**
Scenario 2	5.830	4.345	**4.015**	4.170
Scenario 3	8.331	6.997	5.805	**5.397**
Scenario 4	11.676	9.500	8.923	**8.766**
Avg	7.514	5.982	5.529	**5.310**

Table 5.3: RMSE on GTA dataset.

Methods	V-LSTM	CS-LSTM	S-GAN	CF-LSTM (Ours)
Scenario 1	1.88	1.25	1.40	**1.11**
Scenario 2	1.91	1.84	**1.74**	1.76
Scenario 3	2.98	2.55	2.67	**2.42**
Scenario 4	3.02	2.89	2.96	**2.76**
Avg	2.45	2.13	2.19	**2.01**

RMSE alone does not tell apart the difference between Scenario 3 and 4 for all the methods. It implies the accuracy measure of trajectory prediction would not perfectly reflect the driving safety. Measure the collision rate, for instance, could effectively support the evaluation in a more complete perspective.

Table 5.4: RMSE on NGSIM dataset.

Times(s)	CV	C-VGMM+VIM	V-LSTM	S-LSTM	CS-LSTM	MFP	MATF GAN	VAE	GAIL-GRU	PS-GAIL	CF-LSTM (Ours)
1s	0.73	0.66	0.66	0.65	0.61	**0.54**	0.66	0.68	0.69	0.60	0.55
2s	1.78	1.56	1.64	1.31	1.27	1.16	1.34	1.72	1.51	1.83	**1.10**
3s	3.13	2.75	2.94	2.16	2.09	1.89	2.08	2.77	2.55	3.14	**1.78**
4s	4.78	4.24	4.59	3.25	3.10	2.75	2.97	3.94	3.65	4.56	**2.73**
5s	6.68	5.99	6.60	4.55	4.37	**3.78**	4.13	5.21	4.71	6.48	3.82

Results on NGSIM Dataset Quantitative results on the NGSIM dataset is presented in Table 5.4. Our proposed *CF-LSTM* significantly outperforms the deterministic physics based methods such as C-VGMM+VIM [DRT18], pattern based methods V-LSTM, S-LSTM [AGR16] and CS-LSTM [DT18a] as well as surpasses the planning based methods GAIL-GRU [HE16] and PS-GAIL [BPW18, BPL19]. CF-LSTM largely improves the previous state-of-the-art baselines that comparable with MFP [TS19]. Whereas MFP requires additional scene semantics for single time step predictions while CF-LSTM does not. Lastly, CF-LSTM outperforms MATF GAN [ZXM19], one recent generative prediction method. Those results verify the competitiveness of CF-LSTM on accurate trajectory predictions.

5.4.4 Qualitative Results

Congestion Patterns On GTA dataset, we visualize the learned congestion patterns in terms of mode weights in Fig. 5.7. The mode weights are illustrated as the categorical distribution as λ_i ($M_{\mathcal{Q}} = 4$) of the learned Gaussian Mixture Model $\mathcal{Q}(o)$ on two driving scenarios. Top two rows of Fig. 5.7 shows a driving behavior involves lane changing and overtaking. When the event of overtaking occurs, the second component becomes more evident than others compared to the relatively uniform distributions at the start and end of the driving series. The distributional shift could also be inspected at the bottom two rows in Fig. 5.7 as well. In this intersection driving scenario, the vehicles must yield to

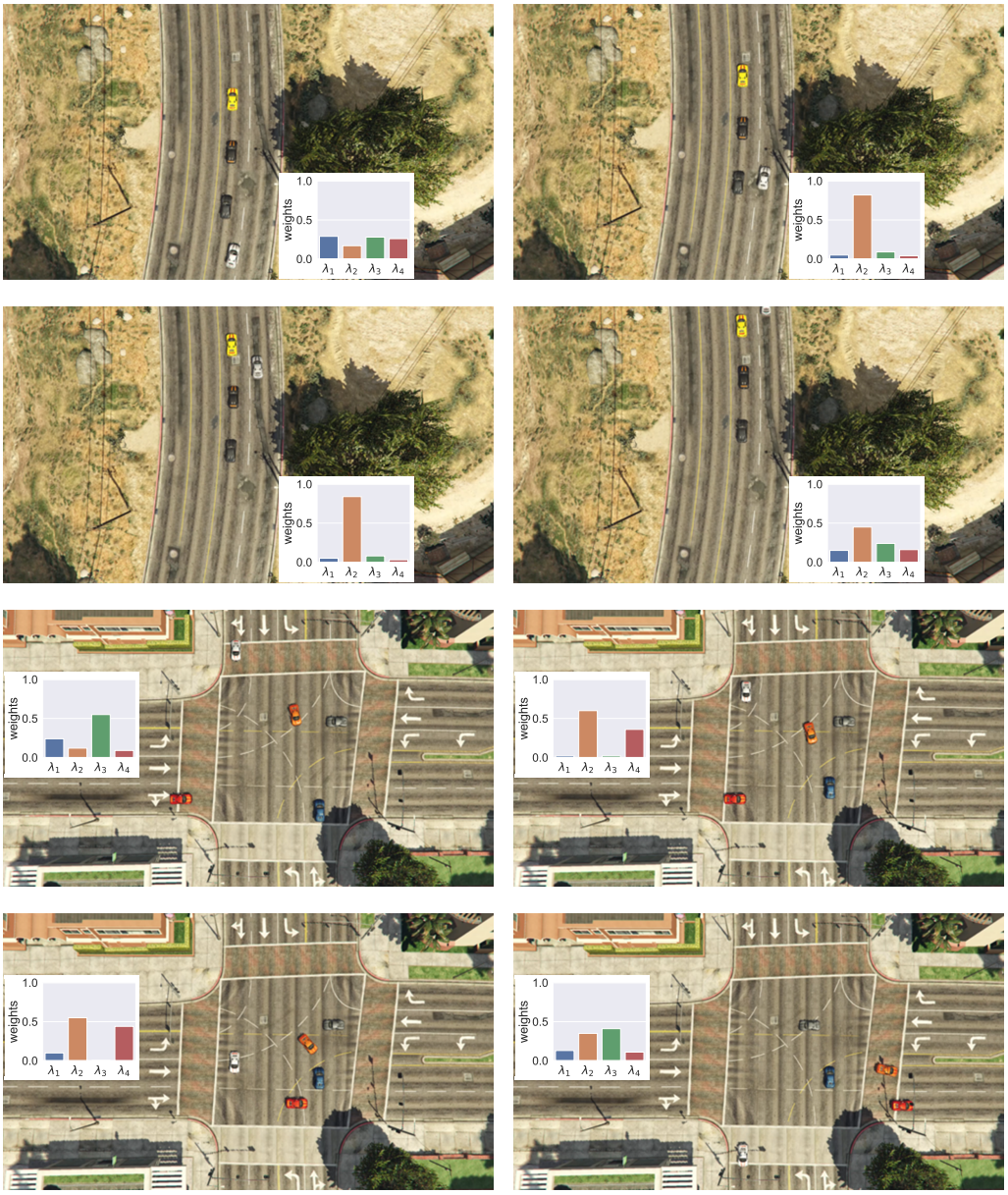


Figure 5.7: Mixture weights of congestion patterns learned in the driving scenarios. Top: the mixture weight distribution during overtaking. Bottom: the mixture weight distribution in a crowded intersection.

each other for collision avoidance. Some mixture weights are evidently firing as vehicles are coming closer compared to the start and end frames. The observational results further verify

our assumptions that learning congestion patterns indeed reflect the contextual semantics of safety critical scenes.

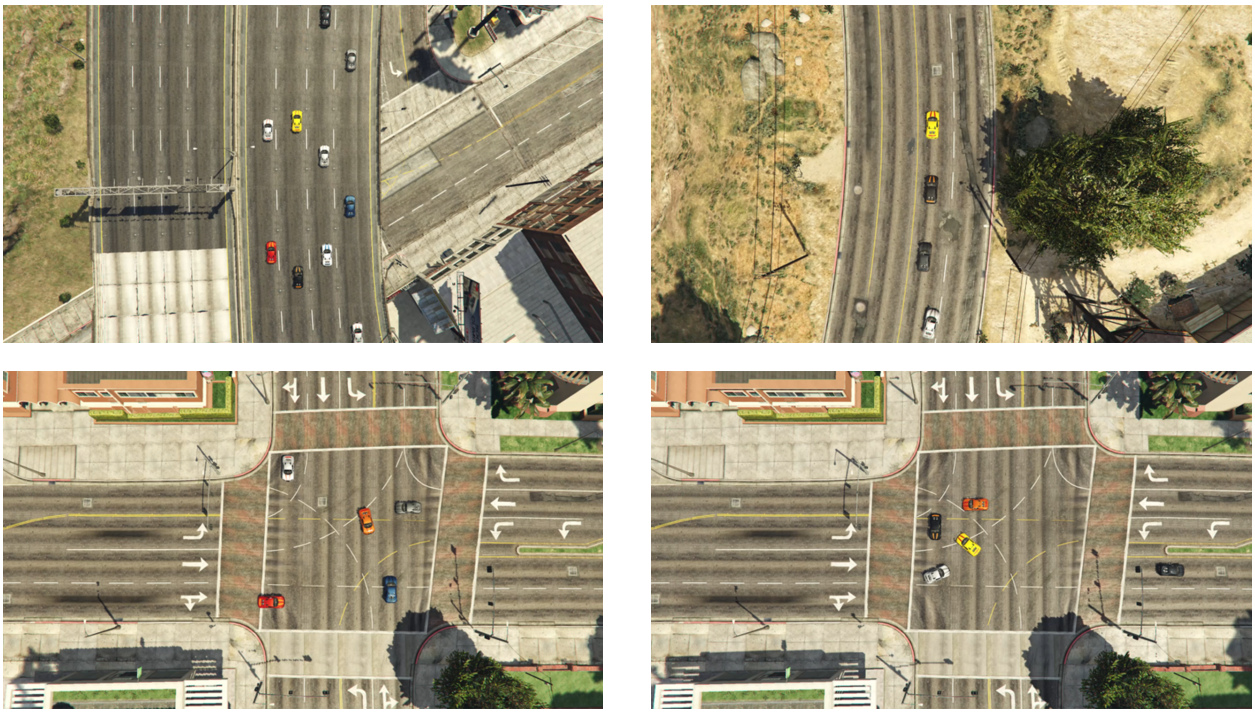


Figure 5.8: Four types of scenarios are used to qualitatively demonstrate the trajectory predictions.

Trajectory Predictions The Qualitative results on predicted vehicle trajectories are depicted in Fig. 5.8 and 5.9. The baseline methods we draw compared to our proposed CF-LSTM are S-GAN and CS-LSTM on four types driving scenarios. For highway driving scenario, as shown in the first row of Fig. 5.9, our CF-LSTM generates more accurate trajectories compared to other baselines on every time interval. For local driving scenario depicted in the second row of Fig. 5.9, our CF-LSTM successfully captures the overtaking tendency of the lane changing vehicle. The predicted trajectory also keeps a relative safe distance from other vehicles. Other baselines either diverge from the ground truth or fail to keep a safe driving distance. For intersection driving scenario presented in the third row of Fig. 5.9,

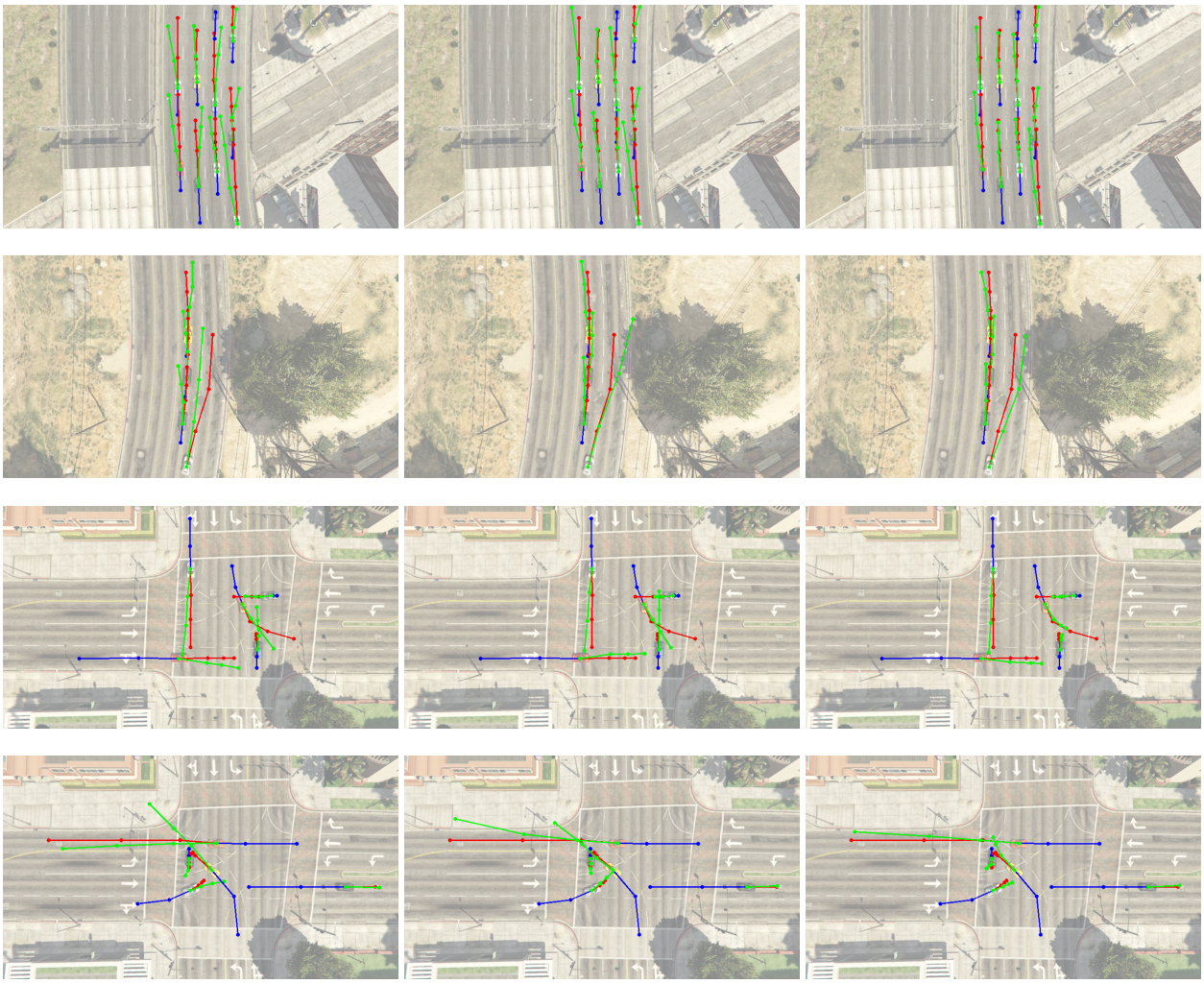


Figure 5.9: Qualitative results of trajectory prediction in four types of scenarios in Fig. 5.8 from the proposed CF-LSTM and two baselines. Blue: observed trajectories. Red: ground-truth future trajectories. Green: predicted future trajectories.

our CF-LSTM shows the tendency of vehicles yielding in the crowded driving situations. It would be the correct sign to avoid collision when vehicles are getting closer. Compared to other methods, it exhibits more reasonable vehicle interacting behaviors by learning from the contextual cues – congestion patterns. The last row in Fig. 5.9 shows an extreme case

of abnormal driving behaviors where one vehicle is scripted aggressively driving through the intersection and eventually crashes into another vehicle. Baselines S-GAN or CS-LSTM fails to be aware of this dangerous situation from the predicted trajectories. Our CF-LSTM instead, presents the tendency of deceleration and yielding based on the trajectories. Those qualitative results and analysis verify the efficacy of our proposed method targeting on safety critical driving scenarios.

5.5 Conclusion

In this chapter, we study the problem of multi-vehicle trajectory prediction which involves the fruitful contextual information of social interactions. The primary challenge of solving this issue is to generate future trajectories that afford collision avoidance. Our approach proposes to explicitly learn congestion patterns as contextual cues in a fully self supervised manner and decouple the "Sense-Learn-Reason-Predict" framework into a modeling process of teacher-student learning. The optimization problem then is formulated to alternatively learn congestion patterns matching and joint trajectory predictions. Our method is evaluated on both public NGSIM US-101 dataset and GTA game scripted multi-vehicle driving dataset. The GTA game provides physical realistic simulations to evaluate the collision events. According to the results, our method achieves the best performance compared to the baselines in the sense of collision free trajectory predictions.

CHAPTER 6

Conclusion

In this dissertation, we investigate one of the most crucial issues in developing the general-purpose robot agent through learning from interactions. Learning from interactions introduces an active manner of robot learning that assemble what human is growing. To satisfy this objective, physics based interactive environments — *big tasks* are constructed to boost the study along this direction. In our pathway, we explore and answer four aspects of existing questions to systematically pursue robot learning from interactions.

1. **Physical realistic invisible data describes the robot-object interactions.** We propose to trace two crucial forms of raw data, pose and force that both are not visible for perception processing. To leverage the tactile information, we develop the tactile glove equipped with wearable sensors and perform the data collection. The hardware setup keeps tracking of interactive events during the object manipulations, such as opening medicine bottles, which reflects underlying physical process.

2. **Temporal grammar model generalizes the manipulation skills.** We inspect the event recognition of a raw motion sequence during manipulation by proposing an unsupervised learning framework. To generalize the temporal knowledge from human manipulation to the robot's end, a grammar model T-AOG is learned to hierarchically represent the manipulations for event parsing. The robot obtains the learned temporal knowledge to afford both *top-down* and *bottom-up* planning processes and executes the planned atomic motion by embodiment mapping. The overall learning and inference pipeline showcase the validity of our proposed approach of acquiring manipulation skills.

3. **Constructing physical realistic and interactive virtual environments for bit tasks.** Facing the challenges of real world setups, we develop a virtual testbed, *VRGym*, to continue the line of robot learning from interactions towards big tasks achievement. Utilizing the advance of game industry, mature physics based simulation and graphics rendering could be leveraged to synthesize more realistic and diverse interactions. Human is also embodied in virtual environments by integrating different VR interfaces. Diverse exemplar tasks such as intention prediction, human robot collaborations and long horizon embodied planning are performed to justify the potential of VRGym growing a general-purpose agent realizing the true intelligence.

4. **Simulation environment for large scale multi-agent social interactions.** We utilize the game platform, *GTA*, to study urban level social interactions. By developing toolkit of modding scripts, control of game agents, e.g. vehicles, and extraction of ground truth information could be seamlessly performed in real-time. To afford socially acceptable driving behaviors, multi-vehicle trajectory predictions is studied for the purpose of collision avoidance. A decoupled learning framework is proposed to learn social contextual cues in a self supervised manner. The GTA platform proves its capability in studying safety critical interactive scenarios.

For future work, we would further expand the topic of robot learning from interactions into two down stream directions: i) *Explainable Intelligence.* Along with the increase of model complexity learning complicated tasks, the agent should derive a sense of explaining its behavior and decision during task execution. An explainable robot system not only satisfies the intelligibility, but builds up *trust* between human agents to attain the stage of becoming a socially mature agent. ii) *Representations for Generalization.* To effectively cope with multiple embodied tasks, ability of generalization is important for an intelligent agent to quickly adapt to the new task environments. In addition, representation learning would be a promising strategy to learn meaningful task abstractions. One feasible approach of learning the representation is self supervised learning based on multimodal information.

Supported by physical realistic and task rich virtual environments, foundations are erected to push forward the representation learning for embodied big tasks.

REFERENCES

[ACV09] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. "A survey of robot learning from demonstration." *Robotics and autonomous systems*, 2009.

[AGR16] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. "Social lstm: Human trajectory prediction in crowded spaces." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[AL17] Florent Altché and Arnaud de La Fortelle. "An LSTM network for highway trajectory prediction." In *IEEE International Conference on Intelligent Transportation Systems*, 2017.

[AN04] Pieter Abbeel and Andrew Y Ng. "Apprenticeship learning via inverse reinforcement learning." In *International Conference on Machine Learning (ICML)*, p. 1. ACM, 2004.

[BCA16] Lamberto Ballan, Francesco Castaldo, Alexandre Alahi, Francesco Palmieri, and Silvio Savarese. "Knowledge transfer for scene-specific motion prediction." In *European Conference on Computer Vision (ECCV)*, 2016.

[BCP16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. "Openai gym.", 2016.

[BD99] Kristin P Bennett and Ayhan Demiriz. "Semi-supervised support vector machines." In *Advances in Neural Information Processing Systems (NIPS)*, 1999.

[BF08] Marcus A Brubaker and David J Fleet. "The kneed walker for human pose tracking." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[BG04] Allison Bruce and Geoffrey Gordon. "Better motion prediction for people-tracking." In *International Conference on Robotics and Automation (ICRA)*, 2004.

[BLT16] Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. "Deepmind lab." *arXiv preprint arXiv:1612.03801*, 2016.

[BPA17] Simon Brodeur, Ethan Perez, Ankesh Anand, Florian Golemo, Luca Celotti, Florian Strub, Jean Rouat, Hugo Larochelle, and Aaron Courville. "HoME: A household multimodal environment." *arXiv preprint arXiv:1711.11017*, 2017.

[BPL19] Raunak P Bhattacharyya, Derek J Phillips, Changliu Liu, Jayesh K Gupta, Katherine Driggs-Campbell, and Mykel J Kochenderfer. "Simulating emergent properties of human driving behavior using multi-agent reward augmented imitation learning." In *International Conference on Robotics and Automation (ICRA)*, 2019.

[BPW18] Raunak P Bhattacharyya, Derek J Phillips, Blake Wulfe, Jeremy Morton, Alex Kuefler, and Mykel J Kochenderfer. "Multi-agent imitation learning for driving simulation." In *International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[BS18] Noam Brown and Tuomas Sandholm. "Superhuman AI for heads-up no-limit poker: Libratus beats top professionals." *Science*, 2018.

[BSF09] Marcus A Brubaker, Leonid Sigal, and David J Fleet. "Estimating contact dynamics." In *International Conference on Computer Vision (ICCV)*, 2009.

[BZ03] Adrian Barbu and Song-Chun Zhu. "Graph partition by Swendsen-Wang cuts." In *International Conference on Computer Vision (ICCV)*, p. 320. IEEE, 2003.

[CAN08] Adam Coates, Pieter Abbeel, and Andrew Y Ng. "Learning for control from multiple demonstrations." In *International Conference on Machine Learning (ICML)*, 2008.

[CDF17] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. "Matterport3D: Learning from RGB-D Data in Indoor Environments." In *International Conference on 3D Vision (3DV)*, 2017.

[Cel85] Gilles Celeux. "The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem." *Computational Statistics Quarterly*, 1985.

[CFG15] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. "Shapenet: An information-rich 3d model repository.", 2015.

[CH07] James Colyar and John Halkias. "US highway 101 dataset." *Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-07-030*, 2007.

[CLH16] Yu Fan Chen, Miao Liu, and Jonathan P How. "Augmented dictionary learning for motion prediction." In *International Conference on Robotics and Automation (ICRA)*, 2016.

[CLO16] Sungjoon Choi, Kyungjae Lee, and Songhwai Oh. "Robust learning from demonstration using leveraged Gaussian processes and sparse-constrained optimization." In *International Conference on Robotics and Automation (ICRA)*, 2016.

[CRK17] Konstantinos Chatzilygeroudis, Roberto Rama, Rituraj Kaushik, Dorian Goepp, Vassilis Vassiliades, and Jean-Baptiste Mouret. "Black-box data-efficient policy search for robotics." In *International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[CWS15] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. "Benchmarking in Manipulation Research." *IEEE Robotics & Automation Magazine*, 2015.

[DCH16] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. "Benchmarking deep reinforcement learning for continuous control." In *International Conference on Machine Learning (ICML)*, 2016.

[DCS19] Wenchao Ding, Jing Chen, and Shaojie Shen. "Predicting vehicle behaviors over an extended horizon using behavior interaction network." In *International Conference on Robotics and Automation (ICRA)*, 2019.

[DLL19] Shengzhe Dai, Li Li, and Zhiheng Li. "Modeling vehicle interactions via modified LSTM models for trajectory prediction." *IEEE Access*, 2019.

[DMG16] Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. "Deep unsupervised clustering with gaussian mixture variational autoencoders." *arXiv preprint arXiv:1611.02648*, 2016.

[DRC17] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. "CARLA: An Open Urban Driving Simulator." In *Annual Conference on Robot Learning (CoRL)*, 2017.

[DRT18] Nachiket Deo, Akshay Rangesh, and Mohan M Trivedi. "How would surround vehicles move? a unified framework for maneuver classification and motion prediction." *IEEE Transactions on Intelligent Vehicles*, 2018.

[DSD08] Laura Dipietro, Angelo M Sabatini, and Paolo Dario. "A survey of glove-based systems and their applications." *Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 2008.

[DT18a] Nachiket Deo and Mohan M Trivedi. "Convolutional social pooling for vehicle trajectory prediction." In *Workshops of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[DT18b] Nachiket Deo and Mohan M Trivedi. "Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms." In *IEEE Intelligent Vehicles Symposium*, 2018.

[Ear70] Jay Earley. "An efficient context-free parsing algorithm." *Communications of the ACM*, 1970.

[ET16] Peter Englert and Marc Toussaint. "Combined Optimization and Reinforcement Learning for Manipulation Skills." In *Robotics: Science and Systems (RSS)*, 2016.

[ETT15] Tom Erez, Yuval Tassa, and Emanuel Todorov. "Simulation tools for model-based robotics: Comparison of Bullet, Havok, MuJoCo, ODE and PhysX." In *International Conference on Robotics and Automation (ICRA)*, 2015.

[FLG15] Sarah Ferguson, Brandon Luders, Robert C Grande, and Jonathan P How. "Real-time predictive modeling and robust avoidance of pedestrians with uncertain, changing intentions." In *Algorithmic Foundations of Robotics XI*, 2015.

[GJF18] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. "Social gan: Socially acceptable trajectories with generative adversarial networks." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[GLY18] Yang Gao, Ji Lin, Fisher Yu, Sergey Levine, Trevor Darrell, et al. "Reinforcement learning from imperfect demonstrations." *arXiv preprint arXiv:1802.05313*, 2018.

[HD94] Gillian M Hayes and John Demiris. *A robot controller using learning by imitation*. University of Edinburgh, Department of Artificial Intelligence, 1994.

[HE16] Jonathan Ho and Stefano Ermon. "Generative adversarial imitation learning." In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[HLD16] Bidan Huang, Miao Li, Ravin Luis De Souza, Joanna J Bryson, and Aude Billard. "A modular approach to learning manipulation strategies from human demonstration." *Autonomous Robots*, 2016.

[HO07] John R Hershey and Peder A Olsen. "Approximating the Kullback Leibler divergence between Gaussian mixture models." In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2007.

[HS06] Geoffrey E Hinton and Ruslan R Salakhutdinov. "Reducing the dimensionality of data with neural networks." *science*, 2006.

[HZR15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[JDH11] Joshua Joseph, Finale Doshi-Velez, Albert S Huang, and Nicholas Roy. "A Bayesian nonparametric approach to modeling motion patterns." *Autonomous Robots*, 2011.

[JHH16] Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. "The Malmo Platform for Artificial Intelligence Experimentation." In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

[JQZ18] Chenfanfu Jiang, Siyuan Qi, Yixin Zhu, Siyuan Huang, Jenny Lin, Lap-Fai Yu, Demetri Terzopoulos, and Song-Chun Zhu. "Configurable 3d scene synthesis and 2d image rendering with per-pixel ground truth using stochastic grammars." *International Journal of Computer Vision*, 2018.

[JZT16] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. "Variational deep embedding: An unsupervised and generative approach to clustering." *arXiv preprint arXiv:1611.05148*, 2016.

[KB13] Andrea Kleinsmith and Nadia Bianchi-Berthouze. "Affective body expression perception and recognition: A survey." *Transactions on Affective Computing*, 2013.

[KKK17] ByeoungDo Kim, Chang Mook Kang, Jaekyum Kim, Seung Hi Lee, Chung Choo Chung, and Jun Won Choi. "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network." In *IEEE International Conference on Intelligent Transportation Systems*, 2017.

[KMG17] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. "AI2-THOR: An interactive 3d environment for visual AI.", 2017.

[KMS17] Tomasz Piotr Kucner, Martin Magnusson, Erik Schaffernicht, Victor Hernandez Bennetts, and Achim J Lilienthal. "Enabling flow awareness for mobile robots in partially observable environments." *Robotics and Automation Letters (RA-L)*, 2017.

[KSM13] Tomasz Kucner, Jari Saarinen, Martin Magnusson, and Achim J Lilienthal. "Conditional transition maps: Learning motion patterns in dynamic environments." In *International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[KSM19] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. "Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks." In *Advances in Neural Information Processing Systems (NIPS)*, 2019.

[KT73] Daniel Kahneman and Amos Tversky. "On the psychology of prediction." *Psychological review*, 1973.

[KW98] Eckhard Kruse and Friedrich M Wahl. "Camera-based observation of obstacle motions to derive statistical data for mobile robot motion planning." In *International Conference on Robotics and Automation (ICRA)*, 1998.

[KW13] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114*, 2013.

[KW16] Thomas N Kipf and Max Welling. "Semi-supervised classification with graph convolutional networks." *arXiv preprint arXiv:1609.02907*, 2016.

[KWR16] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. "Vizdoom: A doom-based ai research platform for visual reinforcement learning." In *Computational Intelligence and Games*, 2016.

[LCO16] Kyungjae Lee, Sungjoon Choi, and Songhwai Oh. "Inverse reinforcement learning with leveraged Gaussian processes." In *International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[LFH03] Lin Liao, Dieter Fox, Jeffrey Hightower, Henry Kautz, and Dirk Schulz. "Voronoi tracking: Location estimation using sparse and noisy sensor data." In *International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[LHP16] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. "Continuous control with deep reinforcement learning." In *International Conference on Learning Representations (ICLR)*, 2016.

[LJ05] X Rong Li and Vesselin P Jilkov. "Survey of maneuvering target tracking. Part V. Multiple-model methods." *IEEE Transactions on Aerospace and Electronic Systems*, 2005.

[LLS15] Ian Lenz, Honglak Lee, and Ashutosh Saxena. "Deep learning for detecting robotic grasps." *International Journalof Robotics Research (IJRR)*, 2015.

[LMZ19] Jiachen Li, Hengbo Ma, Wei Zhan, and Masayoshi Tomizuka. "Coordination and trajectory prediction for vehicle interactions via Bayesian generative modeling." In *IEEE Intelligent Vehicles Symposium*, 2019.

[LPK11] Sergey Levine, Zoran Popovic, and Vladlen Koltun. "Nonlinear inverse reinforcement learning with gaussian processes." In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[LZX19] Hangxin Liu, Zhenliang Zhang, Xie Xu, Yixin Zhu, Yue Liu, Yongtian Wang, and Song-Chun Zhu. "High-Fidelity Grasping in Virtual Reality using a Glove-based System." In *International Conference on Robotics and Automation (ICRA)*, 2019.

[Mar82] David Marr. "Vision: A computational approach." Freeman.[aAC],1982.

[MBM16] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. "Asynchronous methods for deep reinforcement learning." In *International Conference on Machine Learning (ICML)*, 2016.

[MCK18] Sergi Molina, Grzegorz Cielniak, Tomáš Krajník, and Tom Duckett. "Modelling and predicting rhythmic flow patterns in dynamic environments." In *Annual Conference Towards Autonomous Robotic Systems*, 2018.

[MGH09] Manuel Muhlig, Michael Gienger, Sven Hellbach, Jochen J Steil, and Christian Goerick. "Task-level imitation learning using variance-based movement optimization." In *International Conference on Robotics and Automation (ICRA)*. IEEE, 2009.

[MKS15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. "Human-level control through deep reinforcement learning." *Nature*, 2015.

[MLF10] Francisco S Melo, Manuel Lopes, and Ricardo Ferreira. "Analysis of Inverse Reinforcement Learning with Perturbed Demonstrations." In *European Conference on Artificial Intelligence (ECAI)*, 2010.

[MSB17] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisỳ, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker." *Science*, 2017.

[NR00] Andrew Y Ng, Stuart J Russell, et al. "Algorithms for inverse reinforcement learning." In *International Conference on Machine Learning (ICML)*, 2000.

[PJK16] Chris Paxton, Felix Jonathan, Marin Kobilarov, and Gregory D Hager. "Do what I want, not what I did: Imitation of skills by planning sequences of actions." In *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016.

[PK15] Karl Pauwels and Danica Kragic. "Simtrack: A simulation-based framework for scalable real-time object pose detection and tracking." In *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015.

[PKK18] Seong Hyeon Park, ByeongDo Kim, Chang Mook Kang, Chung Choo Chung, and Jun Won Choi. "Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture." In *IEEE Intelligent Vehicles Symposium*, 2018.

[PKQ15] Tu-Hoa Pham, Abderrahmane Kheddar, Ammar Qammaz, and Antonis A Argyros. "Towards force sensing from vision: Observing hand-object interactions to infer manipulation forces." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[PRB18] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. "VirtualHome: Simulating Household Activities via Programs." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Put14] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[QHW17] Siyuan Qi, Siyuan Huang, Ping Wei, and Song-Chun Zhu. "Predicting Human Activities Using Stochastic Grammar." In *International Conference on Computer Vision (ICCV)*, 2017.

[QZH18] Siyuan Qi, Yixin Zhu, Siyuan Huang, Chenfanfu Jiang, and Song-Chun Zhu. "Human-centric Indoor Scene Synthesis Using Stochastic Grammar." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[RA07] Deepak Ramachandran and Eyal Amir. "Bayesian inverse reinforcement learning." In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[RPH19] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrila, and Kai O Arras. "Human motion trajectory prediction: A survey." *arXiv preprint arXiv:1905.06113*, 2019.

[RSB09] Nathan D Ratliff, David Silver, and J Andrew Bagnell. "Learning to search: Functional gradient techniques for imitation learning." *Autonomous Robots*, 2009.

[Rus98] Stuart Russell. "Learning agents for uncertain environments." In *International Conference on Machine Learning (ICML)*, 1998.

[SAS19] Shashank Srikanth, Junaid Ahmed Ansari, Sarthak Sharma, et al. "INFER: INtermediate representations for FuturE pRediction." *arXiv preprint arXiv:1903.10641*, 2019.

[SBS08] David Silver, James Bagnell, and Anthony Stentz. "High performance outdoor navigation from overhead data using imitation learning." *Robotics: Science and Systems (RSS)*, 2008.

[SCD17] Manolis Savva, Angel X Chang, Alexey Dosovitskiy, Thomas Funkhouser, and Vladlen Koltun. "MINOS: Multimodal Indoor Simulator for Navigation in Complex Environments." *arXiv preprint arXiv:1712.03931*, 2017.

[SDL18] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. "Airsim: High-fidelity visual and physical simulation for autonomous vehicles." In *Field and service robotics*. Springer, 2018.

[SGR17] Tianmin Shu, Xiaofeng Gao, MS Ryoo, and Song-Chun Zhu. "Learning Social Affordance Grammar from Videos: Transferring Human Interactions to Human-Robot Interactions." In *International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

[SHM16] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. "Mastering the game of Go with deep neural networks and tree search." *nature*, 2016.

[SKS19] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. "Sophie: An attentive gan for predicting paths compliant to social and physical constraints." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[SLV18] Edward Schmerling, Karen Leung, Wolf Vollprecht, and Marco Pavone. "Multimodal probabilistic model-based planning for human-robot interaction." In *International Conference on Robotics and Automation (ICRA)*, 2018.

[SMW16] Kyriacos Shiarlis, Joao Messias, and Shimon Whiteson. "Inverse reinforcement learning from failure." In *Autonomous Agents and Multiagent Systems (AAMAS)*, 2016.

[SSS17] Jaeyong Sung, J Kenneth Salisbury, and Ashutosh Saxena. "Learning to Represent Haptic Feedback for Partially-Observable Tasks." *International Conference on Robotics and Automation (ICRA)*, 2017.

[Ste96] Edward Stein. *Without good reason: The rationality debate in philosophy and cognitive science*. Clarendon Press, 1996.

[SYZ17] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. "Semantic Scene Completion From a Single Depth Image." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[THK09] Simon Thompson, Takehiro Horiuchi, and Satoshi Kagami. "A probabilistic model of human motion and navigation intent for mobile robot path planning." In *International Conference on Autonomous Robots and Agents*, 2009.

[TIT93] Satoshi Tadokoro, Yutaka Ishikawa, Toiiioaki Takebe, and Toshi Takamori. "Stochastic prediction of human motion and control of robots in the service of human." In *IEEE Systems Man and Cybernetics Conference*, 1993.

[TPZ13] Kewei Tu, Maria Pavlovskaia, and Song-Chun Zhu. "Unsupervised structure learning of stochastic and-or grammars." In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[TS19] Charlie Tang and Russ R Salakhutdinov. "Multiple futures prediction." In *Advances in Neural Information Processing Systems (NIPS)*, 2019.

[TZ02] Zhuowen Tu and Song-Chun Zhu. "Image segmentation by data-driven Markov chain Monte Carlo." *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2002.

[VFL09] Dizan Vasquez, Thierry Fraichard, and Christian Laugier. "Incremental learning of statistical motion patterns with growing hidden markov models." *IEEE Transactions on Intelligent Transportation Systems*, 2009.

[VGL13] Michal Valko, Mohammad Ghavamzadeh, and Alessandro Lazaric. "Semi-supervised apprenticeship learning." In *European Workshop on Reinforcement Learning*, 2013.

[WGH19] Ying Nian Wu, Ruiqi Gao, Tian Han, and Song-Chun Zhu. "A tale of three probabilistic families: Discriminative, descriptive, and generative models." *Quarterly of Applied Mathematics*, 2019.

[WLW14] Jiang Wang, Zicheng Liu, and Ying Wu. "Learning actionlet ensemble for 3D human action recognition." In *Human Action Recognition with Depth Cameras*. Springer, 2014.

[WMZ13] Yangang Wang, Jianyuan Min, Jianjie Zhang, Yebin Liu, Feng Xu, Qionghai Dai, and Jinxiang Chai. "Video-based hand manipulation capture through composite motion control." *ACM Transactions on Graphics (TOG)*, 2013.

[WNX14] Jiang Wang, Xiaohan Nie, Yin Xia, Ying Wu, and Song-Chun Zhu. "Cross-view action modeling, learning and recognition." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[WSF07] Ying Nian Wu, Zhangzhang Si, Chuck Fleming, and Song-Chun Zhu. "Deformable template as active basis." In *International Conference on Computer Vision (ICCV)*. IEEE, 2007.

[WSH16] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. "Dueling Network Architectures for Deep Reinforcement Learning." In *International Conference on Machine Learning (ICML)*, 2016.

[WWG18] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. "Building generalizable agents with a realistic and rich 3D environment." *arXiv preprint arXiv:1801.02209*, 2018.

[WZZ13] Ping Wei, Nanning Zheng, Yibiao Zhao, and Song-Chun Zhu. "Concurrent action detection with structural prediction." In *International Conference on Computer Vision (ICCV)*, 2013.

[WZZ17] Ping Wei, Yibiao Zhao, Nanning Zheng, and Song-Chun Zhu. "Modeling 4D human-object interactions for joint event segmentation, recognition, and object localization." *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017.

[XGN19] Jianwen Xie, Ruiqi Gao, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. "Representation Learning: A Statistical Perspective." *Annual Review of Statistics and Its Application*, 2019.

[XSX16] Caiming Xiong, Nishant Shukla, Wenlong Xiong, and Song-Chun Zhu. "Robot learning with a spatial, temporal, and causal and-or graph." In *International Conference on Robotics and Automation (ICRA)*. IEEE, 2016.

[XVZ18] Haoyi Xiong, Amin Vahedian, Xun Zhou, Yanhua Li, and Jun Luo. "Predicting traffic congestion propagation patterns: a propagation graph approach." In *Proceedings of the ACM SIGSPATIAL International Workshop on Computational Transportation Science*, 2018.

[XYL13] Lin Xu, Yang Yue, and Qingquan Li. "Identifying urban traffic congestion pattern from historical floating car data." *Procedia-Social and Behavioral Sciences*, 2013.

[XZH18] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. "Gibson Env: Real-World Perception for Embodied Agents." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[YCL19] Linxiao Yang, Ngai-Man Cheung, Jiaying Li, and Jun Fang. "Deep Clustering by Gaussian Mixture Variational Autoencoders with Graph Embedding." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[YYT11] Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F Chan, and Stanley J Osher. "Make it home: automatic optimization of furniture arrangement." *ACM Transactions on Graphics (TOG)*, 2011.

[Zhu91] Qiuming Zhu. "Hidden Markov model for dynamic obstacle avoidance of mobile robot navigation." *IEEE Transactions on Robotics and Automation*, 1991.

[Zhu03] Song-Chun Zhu. "Statistical modeling and conceptualization of visual patterns." *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2003.

[Zie10] Brian D Ziebart. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. PhD thesis, CMU, 2010.

[ZLN14] Jiangchuan Zheng, Siyuan Liu, and Lionel M Ni. "Robust Bayesian Inverse Reinforcement Learning with Sparse Behavior Noise." In *AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

[ZM07] Song-Chun Zhu and David Mumford. *A stochastic grammar of images*. Now Publishers Inc, 2007.

[ZMB08] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. "Maximum Entropy Inverse Reinforcement Learning." In *AAAI Conference on Artificial Intelligence (AAAI)*. Chicago, IL, USA, 2008.

[ZTH08] Feng Zhou, Fernando De la Torre, and Jessica K Hodgins. "Aligned cluster analysis for temporal segmentation of human motion." In *Automatic Face & Gesture Recognition, 2008. FG'08. 8th IEEE International Conference on*, 2008.

[ZTH13] Feng Zhou, Fernando De la Torre, and Jessica K Hodgins. "Hierarchical aligned cluster analysis for temporal clustering of human motion." *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013.

[ZXM19] Tianyang Zhao, Yifei Xu, Mathew Monfort, Wongun Choi, Chris Baker, Yibiao Zhao, Yizhou Wang, and Ying Nian Wu. "Multi-agent tensor fusion for contextual trajectory prediction." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[ZZC15] Yixin Zhu, Yibiao Zhao, and Song Chun Zhu. "Understanding tools: Task-oriented object modeling, learning and recognition." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[ZZM13] Wenping Zhao, Jianjie Zhang, Jianyuan Min, and Jinxiang Chai. "Robust real-time physics-based motion control for human grasping." *ACM Transactions on Graphics (TOG)*, 2013.