UNIVERSITY OF TECHNOLOGY SYDNEY

Increased Task Perception for Adaptable Human-Robot Collaboration

by

James Poon

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

at the
Centre for Autonomous Systems
Faculty of Engineering and Information Technology
University of Technology Sydney

August 2018

Declaration of Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Portions of the research presented in this thesis were undertaken within a series of research exchanges, as part of a collaboration between my primary supervisor and Assoc. Prof. Takamitsu Matsubara at the Intelligent Systems Control Laboratory, Nara Institute of Science and Technology, Japan. As a result, the components of this thesis which were undertaken collaboratively by myself and others within the scope of these exchanges are as follows:

- The work with Gaussian Processes in Section 2.3 was undertaken in collaboration with Daisuke Tanaka during a visit to UTS in October 2014, supported by the Global Initiative Program provided by the Japanese Ministry of Education, Culture, Sports, Science & Technology.

- The Dynamic Policy Programming path planner in Section 3.4 was built together with Yunduan Cui during a visit to UTS from October 2015 to January 2016, supported by the 2015-2017 Japanese Society for the Promotion of Science Bilateral Joint Research Projects (Open Partnership).

- Both Convolutional Neural Networks in Chapter 4 were built together with Yunduan Cui during a visit to UTS in March 2017, again supported by the 2015-2017 Japanese Society for the Promotion of Science Bilateral Joint Research Projects (Open Partnership).

- The work in Chapter 5 was undertaken in collaboration with Yunduan Cui during my visit to NAIST from January to May 2016, and in the immediate months following. My visit was supported by both the New Energy and Industrial Technology Development Organization, Japan, and the UTS Faculty of Engineering & IT Higher Degree by Research Students Research Collaboration Experience.

Signed: Production Note:
Signature removed prior to publication.

Increased Task Perception for Adaptable Human-Robot Collaboration

by

James Poon

Submitted to the Faculty of Engineering and Information Technology
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Abstract

This thesis presents investigation into enhancing the robustness and adaptability of robot action generation in human-interactive scenarios, by means of a heightened level of task or scene perception which in turn leads to a lessened reliance upon the observed behaviours of the robot's human counterpart.

In human-robot interaction under the learning from demonstration paradigm, the demonstration is most often carried out by able experts who are capable of performing the task with a very high degree of proficiency while also considering the robot's physical limitations (movement speed limits, joint singularities, etc.). As a result of this, the actions of the robot's partner in the obtained training samples can be considered to be near-optimal. A disparity then naturally arises when working with end-users whose performance may be hindered by a range of factors such as disability, inexperience, or fatigue. The lack of task-specific goodness in these observed partner behaviours can then lead to unpredictable or unsafe robot actions in demonstration learning frameworks where an arguably excessive emphasis is placed upon the partner performing their share of the task at a skill level comparable to that of the demonstrators.

As gathering a sufficiently large quantity of training data samples to encompass such a broad scope of human aptitude is generally infeasible, it becomes arguable that a greater emphasis for robot action modelling should instead be placed upon the task or the work scene that both agents are operating within. An example of this is in collaborative object handling between two humans; one would naturally generate suitable actions for the task by considering the movements of the leader alongside the object and the space they are moving through. The information derived from the latter two observations increases the chance that imperfections in leader behaviour can be adequately compensated for. This allows for an improved adaptability to novel task conditions, and also increased robustness when the observations of partner behaviour are insufficiently informative for safe action planning. These benefits can be primarily attributed to the trained models being more resilient against a lack of informativeness or task goodness in observed partner behaviour, by instead supplementing such missing fine details with information directly drawn from the immediate environment in which the interactive activity is taking place.

This concept of increased task and environmental perception is assessed across two significantly different human-robot interaction paradigms: intelligent wheelchair navigation, and physical humanoid collaboration. For wheelchair navigation, a framework for the generation of expert-stylized short-term paths that can be concatenated for traversal 'anywhere' is realized as a flexible adaptation upon the conventional approach of static long-term destinations within known occupancy maps. The reliance upon immediately available on-board sensor data, as opposed to the more conventionally restrictive features such as platform position within the map, allows proactively assisted traversal through settings novel to demonstration data without the need for retraining goal inference models. For physical humanoid collaboration, robust robot action generation is achieved when faced with novel task conditions and ambiguous partner observation, serving as an intuitive extension to action generation postulated solely upon briefly observed partner movements. This is evaluated in a collaborative object covering exercise by a human-humanoid team, where object parameters automatically drawn from visual scene data compensates for uninformative human partner observation.

Thesis Supervisors: Jaime Valls Miro and Gamini Dissanayake

Acknowledgements

First and foremost I would like to express my sincerest thanks to my primary supervisor Prof Jaime Valls Miro for providing me with the abundance of guidance and freedom in exploring a range of interesting research topics over the last several years. The significance of the many opportunities provided to me cannot be understated in what has been a frankly life-changing learning experience, starting from since I was an under-graduate intern and culminating in this body of work. Also from the UTS Centre for Autonomous Systems I would like to thank Dr Gavin Paul, Prof Sarath Kodagoda, and Dr Alen Alempijevic for the casual contracting and teaching opportunities over the course of my graduate study. I have genuinely enjoyed my involvement in your respective courses, and learned many things in doing so. This research was supported by the Australian Government Research Training Program.

My sincere gratitude goes to Prof Takamitsu Matsubara at the Nara Institute of Science and Technology's Intelligent System Control lab, for our on-going collaboration since late 2014. Particular thanks must be expressed for ISC's Dr Yunduan Cui, who has become one of my closest friends over the last few years. Thank you for the support and good (and occasionally, not so good) times in Japan; working alongside you at both ISC and CAS, and over the Internet in between, has simply been nothing short of fantastic. I would also like to express gratitude to Mrs Mioko Fukuda at the International Institute for Advanced Studies for providing accommodation during my 2016 exchange visit.

I would like to thank all my colleagues at CAS for your support and advice during my time here; particularly, but in no specific order: Jean Kyle Alvarez, Richardo Khonasty, Karthick Thiyagarajan, Christian Reeks, Antony Tran, Julien Collart, Cedric le Gentil, Lei Shi, James Unicomb, David Hunt, Michael Behrens, Mohammad Norouzi, and Freek de Bruijn. Thanks also goes to the entirety of ISC, particularly Prof Kenji Sugimoto, Dr Daisuke Tanaka, Murase Masaki, Haifeng Han, and Juan Rodriguez; thank you all for making my visits to your lab so memorable.

Finally I wish to thank my mother for her support through everything, and my grandfather who never really understood what exactly I was doing but believed in me regardless. And old Kiki, who keeps the clouds away.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

ANN	Artificial Neural Network
CAS	Centre for Autonomous Systems
CNN	Convolutional Neural Network
DMP	Dynamic Movement Primitive
DOF	degree of freedom
DPP	Dynamic Policy Programming
DTW	Dynamic Time Warping
DWA	Dynamic Window Approach
EaIP	Environment-adaptive Interaction Primitive
GP	Gaussian Process
IP	Interaction Primitive
ISC	Intelligent System Control Lab
LfD	learning from demonstration
LIDAR	laser scanner
MDP	Markov Decision Process

NAIST	Nara Institute of Science and Technology
PMD	powered mobility device
RBFN	Radial Basis Function Network
UTS	University of Technology Sydney
YOLO	You Only Look Once v2

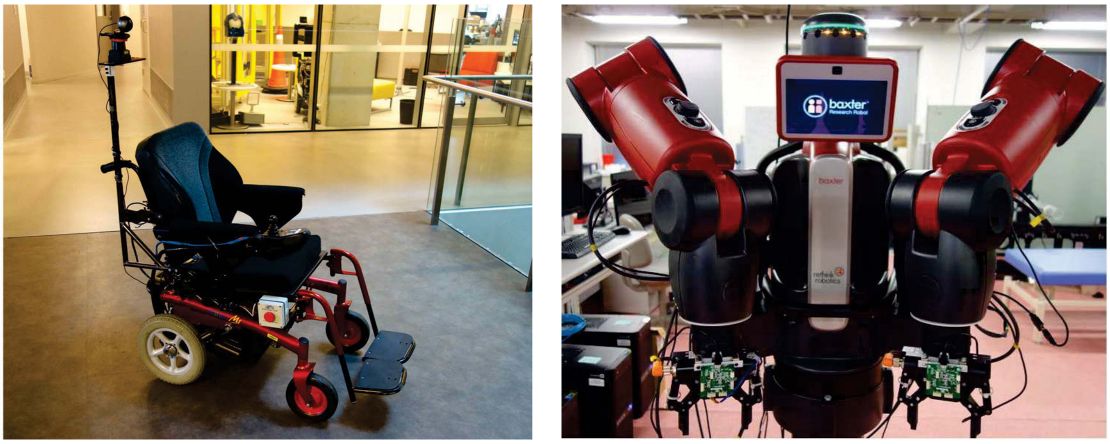
Chapter 1

Introduction

1.1 Background and Motivation

There is a worldwide trend of an ageing population where by 2050 the global population of people aged over 60 is projected to reach 2.1 billion, double that in comparison to 2015 (United Nations 2015). In Australia, where the research towards this thesis was undertaken, the ratio of working-aged people to retirees is also expected to halve between 2015 and 2050 (McCrindle Research 2014). For the years to come this indicates a significant increase in the predicted burden on care staff and the relatives of those suffering from physical and/or cognitive deficiencies onset by age or other causes. Assistive robots working closely with the elderly or the disabled to improve or maintain their quality of life are expected to have a significant impact (Tegart 2010) in alleviating the burden imposed upon caregivers from such activities, allowing their time to be allocated towards other aspects of care.

The last decade has also seen a significant increase in interest towards 'social' human-interactive robots (Vircikova et al. 2012) across both domestic and industrial environments. With estimates that approximately 1.3 million new industrial robots have been installed into factories worldwide between 2015 and 2018 (International Federation of Robotics 2015), there also follows a strong research incentive to accelerate the rate at which robots can be incorporated alongside human workers.



(A) UTS CAS autonomous wheelchair.

(B) NAIST ISC Baxter humanoid.

FIGURE 1.1: Examples of collaborative robots: intelligent mobility aids and humanoid platforms designed to work safely in close proximity with people.

In order to facilitate integration of human-interactive robotic systems in both domestic and industrial settings, a main objective of the human-robot interaction research area is to investigate methodologies towards the synthesis of robot actions for systems such as those shown Figure 1.1 that work directly or in close physical collaboration with a human counterpart. It follows that these robots should behave in a manner that end-users or people nearby would not deem dangerous, unpredictable, or unnatural; i.e. their behaviors should mimic what a human would do under similar conditions. This is particularly important in assistive systems (e.g. Fig. 1.1a) to minimize the risk of rejection by the user. There is a broad range of recommended guidelines concerning the issues of user safety, comfort and psychological acceptance when manually defining behaviours for interactive robots; for a survey we direct the reader to Lasota et al. (2014). As an alternative, Learning from Demonstration (LfD) (Argall et al. 2009) aims to model suitable robot behaviors from "expert" demonstration against task/environmental stimuli. Although beyond the scope of this thesis, it bears noting that these heuristic design guidelines can be potentially incorporated as additional operational protocols, e.g. a hard-coded safety buffer for a robotic arm.

There are several key advantages of LfD that make it appealing in the context of human-robot interaction. The first is that the difficulty in manually crafting seemingly natural actions is largely bypassed as there is a reduced need for hand tuning. Secondly, the models built in LfD can often encompass a broad range of operating conditions which would otherwise require a significant

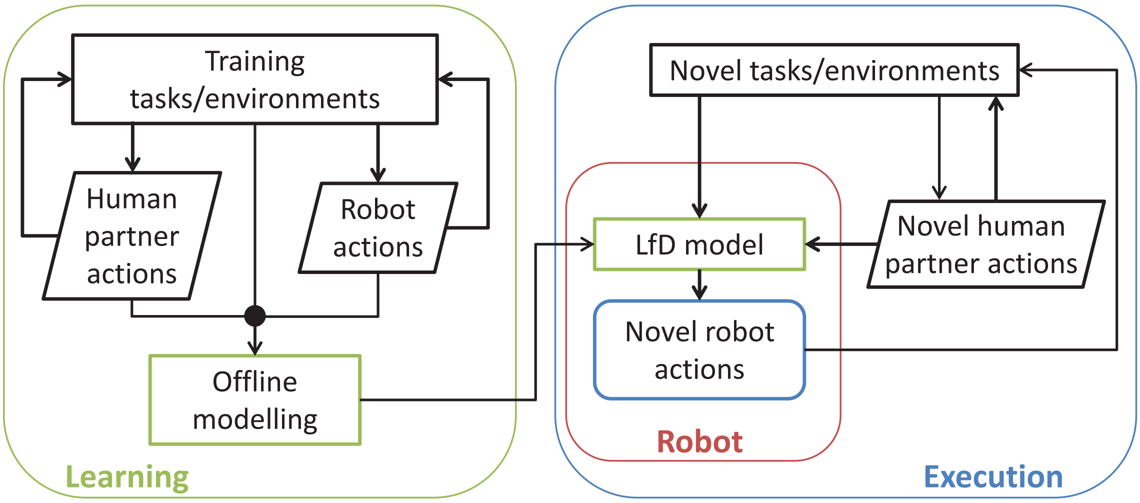


FIGURE 1.2: Broad schematic of a typical human-interactive robot learning framework.

quantity of parameterized cases. Finally, human skills can be better leveraged in robot programming; for example the similitude between human bodies and complex humanoids designed to operate in human-oriented environments allows for an intuitively direct connection between teacher and robot agent.

1.2 Overview of Research Problem

A broad schematic overview of a typical LfD framework operating within a human-interactive context is shown in Figure 1.2. For data collection during the offline learning phase, robot actions can come from a second person controlling the robot by means such as remote tele-operation or working the joints by hand (kinesthetic programming), or alternatively from the actions of the human partner directly e.g. in the case of PMDs. For action planning under novel operating conditions, the LfD model then considers both observations of partner behaviour and task/environmental knowledge from an external oracle or from robot perception to generate demonstration-stylized behaviours.

In this sense the LfD model essentially plays the part of a mapping function, transforming human partner observations and extracted task/environmental attributes into robot activity plans. The exact design of the model can be vastly different on a case-by-case basis, with formulation dependent upon desired effects during the inference phase as equally as more concrete criteria such as data availability or the representation of robot action plans. For example in Chapter 2, an Artificial Neural Network first serves as the LfD model when user intention inference is performed in a greedy manner, then is later superseded by a Gaussian Process regression approach due to the latter's ability to yield uncertainties along with its inferences. As the availability of an up-to-date *a-priori* occupancy map is assumed for that Chapter (a reasonable assumption for a relatively small and un-changing environment such as an average home), these models are well suited to inferring the intention likelihoods of a set of static key points meaningfully distributed throughout the maps. However as later Chapters do away with the *a-priori* map in order to to facilitate generalization towards novel environments, such an approach can no longer be applied as directly. Instead this gives rise to the heuristic model detailed in Chapter 3 built around the assumption of a limited quantity of training data by incorporating a significant amount of domain knowledge, and the black-box deep learning based approach in Chapter 4 which is rendered viable through the availability of a larger training data pool.

As the training data gathered for model building is typically carried out by experts with a near-complete level of task and domain knowledge, there exists a risk of discrepancy when using the trained LfD model in the "real world" based on human factors alone. The performance of end-users may be inhibited by a range of causes including fatigue, inexperience, physical/cognitive disability, haste, external distraction, or others. As the behaviours of the human partner may end up lacking in sufficient application-specific goodness for the planning of suitable and safe robot responses, it is then arguable that a greater emphasis should be placed upon careful consideration of the task with regards to the environment the human-robot team is operating in. The key benefit of down-playing the significance of nuances in human behaviour is that the model's inferences become more attuned to phenomena directly concerning the work scene itself, giving rise to the potential for increased flexibility in novel settings or increased robustness when faced with uninformative partner activity. Another consequence is that training data also does not have to necessarily cover such an extensive range of human aptitude.

The aim of this thesis is thus investigating a higher degree of task/environmental consideration in LfD model building, and the benefits of doing so in improving both the inference of partner intent and/or subsequent robot action planning performed by the framework that the learned model is integrated into. In this thesis, the heightened integration of task/environmental information is explored within the context of two human-robot interaction paradigms: assistive PMD navigation and physical human-humanoid collaboration.

For assistive PMD navigation, greater utilization of on-board sensor data serves as the foundation for intelligent pro-active navigation assistance working within a small moving window around the PMD. While existing approaches map the user input and PMD pose to likely destinations within a larger map, replacing pose data with sensor measurements and long-term goals with short-term waypoints allow for continued functionality in places novel to training. This is especially advantageous as the benefits of active assistance can be leveraged without constant map upkeep and retraining.

Applying this concept to human-humanoid collaboration is fundamentally similar to the PMD problem, albeit with a different sensor configuration to reflect the change in operating context and the objective of generating end-effector trajectories rather than mobile robot paths. Additional task parameters derived directly from scene observation serve to augment parameter inference for robot actions during collaborative exercises, implemented as an extension to the Interaction Primitives framework (Ben Amor et al. 2014) which only consider a brief window of observed partner behaviour at the onset of the interactive activity for the purpose of robot action planning. The framework allows for end-effector trajectory planning while also being sufficiently amenable to the incorporation of additional observations, so as a result we take it as a foundation upon which the broader concept of greater perceptual consideration is explored. The benefits of a reduced reliance upon this partner observation include increased robustness when dealing with ambiguous partner behaviour during the observation phase, as well as accelerating robot response time as the observation window can be shortened.

1.2.1 Assumptions

The PMD platform is assumed to be equipped with a 2D laser scanner, and operating on sufficiently level ground such that its measurements for the purposes of localization or path planning are not subject to significant disturbance. The absence of dynamic obstacles is also assumed. Computational resources are assumed sufficient to avoid undesirable delays or instances such as the PMD starting down an incorrect branch of a fork despite the inferred intention indicating otherwise. Chapter 2 assumes that the PMD is always reasonably localized within the *a-priori* map, while Chapters 3-4 assume goodness of localization for the duration of travel through each moving window. The uncertainty of the laser scanner is largely assumed to be negligible for the purpose of mapping across all three Chapters, due to the relatively high precision of the unit utilized in this thesis in comparison to the $5cm$ resolution of both *a-priori* and local occupancy maps. The users' joystick input is also assumed to have at least some task goodness, particularly in Chapters 3-4. Joystick input is taken as the sole means of user observation as they still remain the most broadly adopted approach to PMD control, and the hardware noise within the unit itself is assumed to be negligible.

Chapter 5 assumes that the kinematics of the humanoid robot will allow it to perform the generated Cartesian end-effector trajectories, and that subsequent computation time for inverse kinematics is assumed to be sufficiently fast to avoid significant delays between action planning and robot movement. The environmental parameter vector $\boldsymbol{e}$ is assumed to remain constant throughout each task, as its contents are ideally task relevant but not dependent on human/robot activity. The human partner observations are also assumed to have minimal associated uncertainty, and again must possess some appreciable degree of task relevance during both demonstration and execution.

The quantities of LfD training data used are assumed to meet or exceed a minimum quantity required for a reasonable degree of adaptability towards novel conditions in proof-of-concept evaluations. As a result exploring the size or possible contents of an optimally effective training set is taken to be out of scope for this body of work.

1.3 Contributions

- A sequential long-term navigational intention estimator for PMD users, for scenarios in which *a-priori* occupancy maps are feasible (Ch. 2). The problem is approached with a probabilistic Gaussian Process filter model, sparsified for real-time inference.
- A heuristic estimator for short-term intention inference based on the immediately observable environment around the PMD and user input (Ch. 3). A deterministic model is built from demonstration data, capable of generalizing to novel settings from limited training samples.
- A continuous approach to short-term intention inference which naturally combines intention estimation and demonstration-stylized pathing (Ch. 4). A Convolutional Neural Network and a Gaussian Process based approach each parse demonstration data in a decoupled fashion to generate probabilistic feasible traversal maps.
- Real-time local path planning for mobile robots which considers both the surrounding environment and user intentions (Chs. 3, 4). Reinforcement learning based optimization schemes are used to fuse local occupancy maps with inferred intention data for the rapid generation of viable robot paths.
- Augmenting physically collaborative human-robot activities beyond purely human-driven action planning with the incorporation of additional task/environmental cues (Ch. 5) drawn from the interaction scene. This extension to an existing human-humanoid interactive framework increases robustness in robot action planning, over basing actions solely upon observed partner behaviors.

The first contribution aims to serve as a juxtaposition to the next three, which function as elements within a real-time assistive navigation framework relying upon environmental perception instead of map pose data to disambiguate human user inputs. The primary motivation behind such a framework is to demonstrate that consideration of the immediate space around both human/robot agents can adequately overcome insufficiently informative user observation (inputs) for the purpose of intelligent collaboration.

1.4 Outline of Thesis

The following Chapters of this thesis are arranged as follows. Please note that there is no Chapter solely dedicated to the summarization of relevant literature; instead each Chapter contains its own section for related works towards the beginning.

- Chapter 2 documents work towards the learning of long-term destination inference for PMDs, with inferences taking the form of static locations distributed within a *a-priori* global occupancy map. This problem is presented as a classification exercise with the LfD models considered in this Chapter built upon the pose of the PMD within the map and the user's input signal, in order to set the stage for the next two Chapters in which such a map is no longer available. Two learning models are considered here: an Artificial Neural Network operating in a deterministic fashion as an initial proof-of-concept, followed by a sequentially filtered Gaussian Process which is sparsified for the purpose of real-time inference. Experiments conducted on the CAS wheelchair on UTS campus demonstrate the efficacy of both approaches in terms of intention convergence and real-time capability.
- In Chapter 3, the reliance upon static destinations in Chapter 2 is addressed with the first iteration of a "locally" assistive framework, by replacing map pose data with on-board sensor measurements and static long-term navigational objectives with short-term waypoints in the previous Chapter's overall learning scheme. Focusing upon a small moving window around the PMD allows for pro-active navigational assistance in novel environments, which was not previously feasible without retraining. This framework incorporates both deterministic intention estimates from a heuristic model designed to leverage a limited quantity of training data, and a reinforcement learning based short-term path planning scheme incorporating demonstration-stylized path primitives and local occupancy maps. Experiments conducted in both a simulated PMD with a range of able users, and on the real wheelchair with a disabled volunteer validate the approach in accurately inferring user intent and quickly planning safe paths to local intentions.
- A newer iteration of the assistive framework is presented in Chapter 4. A less restrictive training data pool allows for more conventional learning models to be applied, namely Convolutional Neural Networks and Gaussian Processes. To reduce complexity of the learning

space, sensor data and user input signals are initially decoupled. Then a continuous traversal likelihood heatmap is generated from each observation, and combined prior to path planning. As each heatmap naturally incorporates both intention estimation and demonstration stylized path planning, this approach no longer requires separate models for path primitives and does not draw upon such a large amount of heuristic task knowledge. Experiments comparing the two frameworks reveal similar inference accuracies, however this approach features a notably faster computation time which is beneficial given the on-line nature of the assistive settings this thesis is ultimately aimed towards.

- In Chapter 5, we apply the increased leverage of environmental cues towards physically collaborative activities between a static humanoid robot and a human, with the goal of enhancing the robustness of their interaction. In contrast to previous Chapters, on-board sensors suited to mobile platforms such as laser scanners are replaced with vision of the work scene around both agents, and the objective transitions towards the generation of an end-effector trajectory rather than a path for a vehicle; however the broader concept remains unchanged. By allowing the robot to also perceive the task scene rather than having to base its actions solely upon observations of its partner, there is a reduced chance of undesirable actions stemming from partner behavior that fails to adequately capture nuances of the task at hand. Implementation takes the form of an extension to the Interaction Primitives framework, by augmenting the joint distribution of human and robot action trajectory parameters with additional information drawn from scene observation. The proposed extension is first validated in a toy simulation problem, then in real experiments between the NAIST ISC Baxter humanoid robot and a human partner in a collaborative object manipulation exercise.

- Chapter 6 summarizes the work presented throughout this thesis, and concludes with an overview of the outcomes, limitations, and potential avenues for future investigation.

- Appendix A details the UTS CAS instrumented wheelchair (Fig. 1.1a), including an evaluative comparison to its simulated counterpart. Appendix B details the NAIST ISC Baxter robot system (Fig. 1.1b). Appendices C-D respectively introduce the various supervised and reinforcement learning methodologies applied throughout. Appendix E overviews other assorted algorithms.

1.5 Publications

The initial work on long-term goal inference for PMDs using an Artificial Neural Networks was first presented in the 2015 International Conference on Rehabilitation Robotics (Poon et al. 2015). A second iteration utilizing GPs was presented at the 2015 International Symposium on Robot and Human Interactive Communication (Matsubara et al. 2015).

Our first approach towards short-term goal inference was presented at the 2017 International Conference on Robotics and Automation (Poon et al. 2017), following a presentation of preliminary findings at the 2016 annual conference of the Robotics Society of Japan (Cui et al. 2016b). A later manuscript containing more extensive experimentation was submitted to Advanced Robotics (Poon et al. 2018a). The most recent iteration, utilizing a Convolutional Neural Network and Gaussian Processes, is under review for the 2018 International Conference on Intelligent Robots and Systems (Poon et al. 2018b).

The initial work on incorporating environmental cues for improved humanoid-human interaction was presented at the 2016 International Conference on Humanoid Robots (Cui et al. 2016a). A manuscript documenting later efforts towards a more complete framework has been accepted for publication in Autonomous Robots (Cui et al. 2018).

The complete list of publications resultant from the research undertaken towards this thesis is as follows:

- **J. Poon**, Y. Cui, J. V. Miro, and T. Matsubara. *Learning Mobility Aid Assistance via Decoupled Observations*. In Proc. IEEE International Conference on Intelligent Robots and Systems (IROS), 2018. Under review.
- **J. Poon**, Y. Cui, J. V. Miro, and T. Matsubara. *Learning from Demonstration for Locally Assistive Mobility Aids*. Advanced Robotics. Under review.
- Y. Cui, **J. Poon**, T. Matsubara, J. V. Miro, K. Sugimoto, and K. Yamazaki. *Environment-adaptive Interaction Primitives through Visual Context for Human-Robot Motor Skill Learning*. Autonomous Robots. Accepted.

- **J. Poon**, Y. Cui, J. V. Miro, T. Matsubara, and K. Sugimoto. *Local Driving Assistance from Demonstration for Mobility Aids*. In Proc. IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 5935-5941.

- Y. Cui, **J. Poon**, T. Matsubara, J. V. Miro, K. Sugimoto, and K. Yamazaki. *Environment-Adaptive Interaction Primitives for Human-Robot Motor Skill Learning*. In Proc. IEEE-RAS International Conference on Humanoid Robots, 2016, pp. 711-717.

- Y. Cui, **J. Poon**, J. V. Miro, T. Matsubara, and K. Sugimoto. *Optimal Control Approach for Active Local Driving Assistance in Mobility Aids*. In 34th Annual Conference of the Robotics Society of Japan, 2016.

- **J. Poon** and J. V. Miro. *Learning by Demonstration for Co-Operative Navigation with Assistive Mobility Devices*. In Australasian Conference on Robotics and Automation (ACRA), 2015.

- **J. Poon**, J. V. Miro, and R. Black. *A Passive Estimator of Functional Degradation in Power Mobility Device Users*. In Proc. IEEE International Conference on Rehabilitation Robotics (ICORR), 2015, pp. 997-1002.

- T. Matsubara, J. V. Miro, D. Tanaka, **J. Poon**, and K. Sugimoto. *Sequential Intention Estimation of a Mobility Aid User for Intelligent Navigational Assistance*. In Proc. 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), 2015, pp. 444-449.

- **J. Poon** and J. V. Miro. *A Multi-modal Utility to Assist Powered Mobility Device Navigation Tasks*. In Proc. 6th International Conference on Social Robotics, 2014, pp. 300-309.

- B. Andonovski, J. V. Miro, **J. Poon**, and R. Black. *An Automated Mechanism to Characterize Wheelchair User Performance*. In Proc. 5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics (BIOROB), 2014, pp. 444-449.

Chapter 2

Global Intention Estimation for Mobility Aids

In this Chapter, work towards intention estimation within an *a-priori* map for PMDs is detailed, documenting the modelling of demonstration data and experimental results with the CAS PMD. The contents of this Chapter encompass the following publications: Poon et al. (2015); Matsubara et al. (2015). This Chapter serves as a broad introduction to research initiatives within the area of learning intelligent PMD assistance from demonstration, setting the stage for an increased leverage of environmental observation as follows in Chapters 3-4. We note that the building in which this Chapter's experiments were conducted was closed for demolition prior to 2016, hence later PMD experiments in the subsequent Chapters do not revisit the area seen here.

In the Background (Sec. 2.1) we begin with a brief summary of instrumented PMDs for navigation assistance which incorporate collision avoidance methodologies designed for use with autonomous mobile robots, which we term as "passive" assistance. This is followed with a review of "active" PMD assistance, introduced to overcome the need for the manual definition of additional navigational behaviours which an end-user would perform in everyday life such as parking alongside furniture items. "Global" destination points are set within a known map, and during runtime their likelihoods as user intentions are inferred. The likeliest destination is then typically taken as a navigational objective for mobile robot path planning strategies, or the platform can simply follow the paths taken by an able demonstrator.

Here we approach the problem of global intention estimation through the use of two different learning models: Artificial Neural Networks (ANN) and Gaussian Process Regression (GPR), introduced respectively in Appendices C.2 and C.3. These models were selected for this problem as they are highly data-driven, and are also utilized for other purposes in later Chapters. Each possess their own advantages and disadvantages within the context of this problem. Inferences via ANN are fast in terms of processing time with a computational complexity determined by network topology, whereas GPs are noticeably slower at $\mathcal{O}(N^3)$ for N training samples. GPs also naturally provide uncertainty which lends to their suitability to Bayesian filtering over time, while ANNs are more akin to a "black box" from which a measure of inference uncertainty cannot be readily acquired. As such, for the work in Section 2.2 the ANN estimates are treated in a greedy manner, while the methodology in Section 2.3 addresses intention estimation as a sequential Bayesian estimation problem with a GP built upon sparsified training data to facilitate real-time inferences.

2.1 Background

2.1.1 Passive PMD Assistance

Early iterations of intelligent PMDs often provided assistance in the form of collision avoidance via algorithms such as the Vector Field Histogram (Borenstein and Koren 1991) designed for autonomous vehicles, or the execution of pre-defined behaviors such as docking with furniture. Prominent examples of such systems include the Bremen autonomous wheelchair (Lankenau et al. 1998), and the NavChair (Simpson et al. 1998; Simpson and Levine 1999) shown in Figure 2.1.

Later works introduced the concept of shared control for the purposes of rectifying potentially hazardous user inputs, where the user collaboratively drove the PMD alongside a control framework. The safe commands from such controllers were often merged with the user's commands as a weighted sum, based on metrics drawn from their driving performance (Li et al. 2011; Urdiales et al. 2011; Goil et al. 2013) (Fig. 2.2a).

Active impeding or filtering of user inputs provided an alternate means for shared control, instrumented on devices such as joysticks with a variable physical resistance (Urbano et al. 2004), and

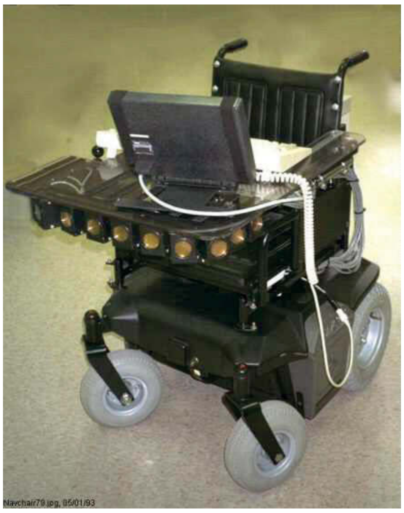
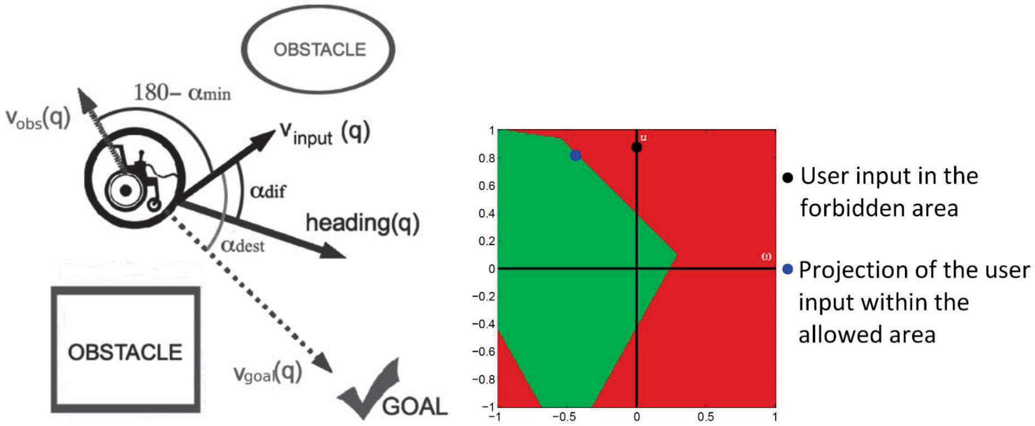


FIGURE 2.1: The Navchair intelligent wheelchair with on-board PC and ultrasonic sensor array, capable of obstacle avoidance and pre-programmed behaviors such as wall following and doorway navigation.



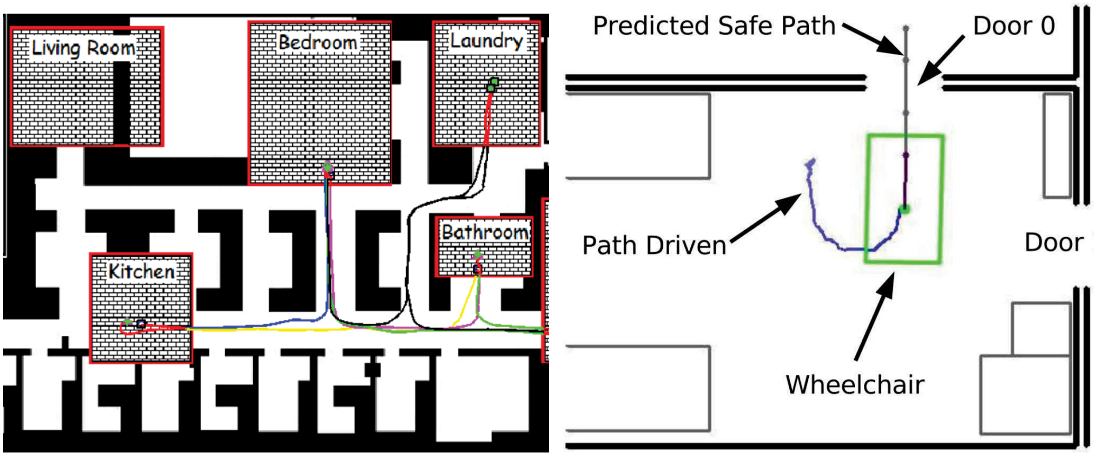
(A) Multi objective-based user input weighing.

(B) Virtual limiting of user input (joystick plane).

FIGURE 2.2: A: Weighing user command signal v_{input} against obstacle avoidance signal v_{obs} and goal tracking signal v_{goal}, based on heuristics α. From Urdiales et al. (2013b). B: Virtual limiting of the user input on the 2D joystick plane to fall within a safe region constantly reshaped by perceived obstacles. From Devigne et al. (2016).

the virtual limiting of the joystick's range of motion to achieve a similar outcome via software alone (Devigne et al. 2016) (Fig. 2.2b).

It bears noting that several of these works feature reactive layers within a hybrid framework taking into account a longer-term goal provided at the onset of the navigational exercise (e.g. Figure 2.2a). However vehicular command signals are still derived in a reactive manner, rather than from a path to the goal that is being followed. The key shortcoming of such systems was a lack of focus on the



(A) Rooms as intentions throughout a domestic environment (Patel et al. 2014).

(B) Doorways as intentions to facilitate passage (Carlson and Demiris 2008).

FIGURE 2.3: Examples of potential navigational intentions in a home setting.

path a user may wish to take; instead their focus was on what the user had attempted to input at a given instant. The problem was formulated as inferring a likely joystick signal that a healthy user might give when faced with the same conditions, rather than inferring a physical destination for navigation algorithms to subsequently guide the PMD towards. As such, we term such approaches "passive"; although capable of mitigating collisions and including situational heuristics, by large these frameworks are incapable of leveraging the robustness benefits of longer-horizon motion planning and control which would naturally also cover the events for which situational behaviours need to be crafted.

2.1.2 Global Intention Estimation for Active PMD Assistance

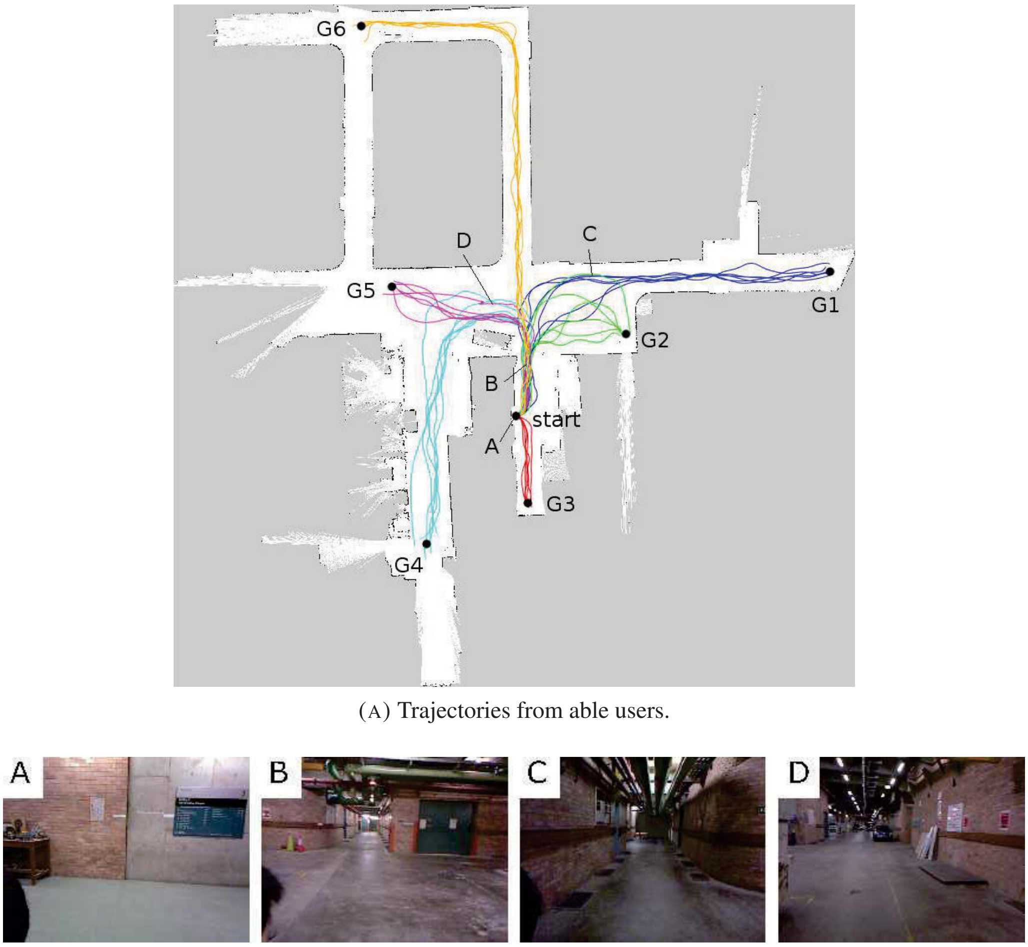
To address the shortcomings of passive assistance, researchers in the assistive PMD space moved towards the inference of user intention as destinations in their environment in order to provide "active" assistance through autonomous path planning and control strategies. Under this paradigm the user is largely removed from the control loop; the main purpose of their inputs is to instead impart cues broadly indicating where they wish to go.

To date much of the work in this area has focused on modeling the intention likelihoods of hand-picked points within a setting which is mapped *a-priori*, with the likeliest point taken as a navigation objective. An example of this is shown in Figure 2.3, with potential intentions being places such as the Laundry or the Kitchen (Fig. 2.3a), or doorways to specific rooms (Fig. 2.3b). These models are usually built upon training data gathered from able-bodied demonstrators or disabled end-users. Likelihoods can also be inferred through other means than demonstration (Carlson and Demiris 2008, 2012), however such approaches require significant tuning and heuristic configuration. Demonstration data typically consists of the PMD's pose within the map as 2D Cartesian co-ordinates and a yaw, and an input from the user. From this data, predictions of the likeliest destination can then be modeled; past work in applying models with consideration of temporal information include the Partially Observable Markov Decision Process (Taha et al. 2011) and Hierarchical Hidden Markov Model (Patel et al. 2014). Other works have approached destination inference by measuring the likelihood as a correlation between fine motion paths planned to the destinations, and a personalized user model (Demeester et al. 2006).

It bears noting that due to the regularity at which inferences can be made, uncertainty between destinations that are still sufficiently far away can be acceptable (Huntemann et al. 2007) as long as resultant PMD movement happens to align with the user's immediate expectations until an updated inference becomes available.

2.2 Global Intention Estimation via Artificial Neural Networks

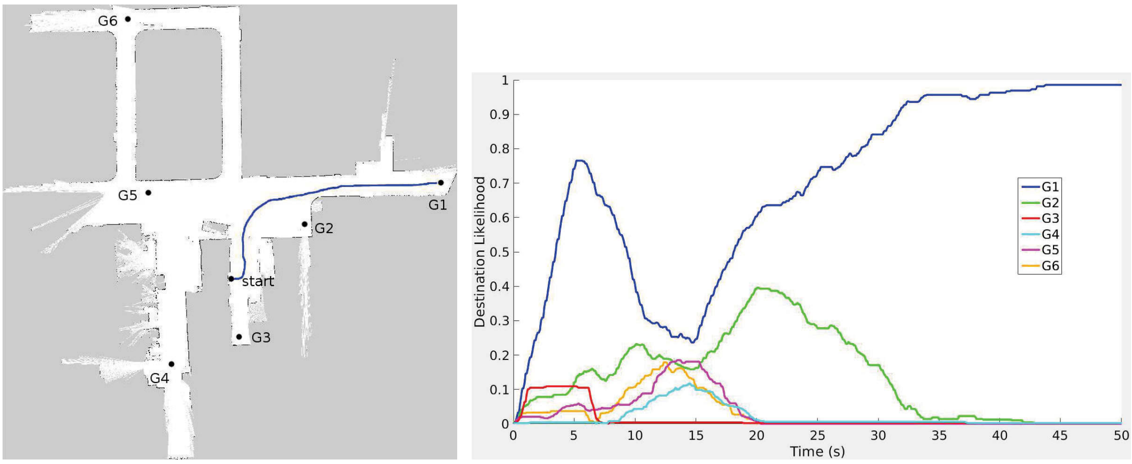
ANN training data consisted of six runs aboard the CAS PMD to each one of six possible destinations of interest {G1,...,G6} arbitrarily chosen in an indoor environment within an underground portion of UTS campus, resulting in a total of 36 training examples from able users (Fig. 2.4). Each user was free to determine their own preferred platform handling characteristics. Joystick input, laser scanner measurements, and footage from an on-board camera were logged. The trajectories were later recovered from a map built in an earlier exercise; both map-building and *a-posteriori* PMD localization were performed via Gmapping (ROS Wiki 2015). In Figure 2.4a, white regions represent free space, black regions represent perceived obstacles and gray regions are unknown due to occlusions.



(A) Trajectories from able users.

(B) On-board camera stills from several locations within the map, for illustrative purposes. Image data was not utilized in modelling.

FIGURE 2.4: Demonstrated trajectories to goals {G1,...,G6} on UTS campus from which training data was drawn. The area shown in the upper map is approx. $72 \times 72m$.



(A) Path to G1 (blue).

(B) Destination likelihood convergence.

FIGURE 2.5: Convergence of ANN destination likelihood estimates during a run to G1.

The input layer (5 neurons total) of the network consisted of the user joystick signals (horizontal and vertical axes positions) and the PMD's pose from Gmapping (2D Cartesian position and orientation). The output layer took the form of six likelihoods (0,1), giving the network the ability to provide an indication of certainty about the destinations. Between the input and output was a hidden layer with 15 neurons, empirically determined as an upper bound beyond which fitting did not improve. Data from the laser scanner was used for localization purposes, however it was not directly fed into the ANN. This due to the undesirably high dimensionality of a laser scan, and risk of ambiguity compared to pose data in cases such as travelling down a featureless corridor. Streaming image data was also captured, albeit primarily for visualization purposes. Figure 2.5 shows an example of destination likelihood convergence over the duration of a single bout to one of the six destinations. Even without time-series filtering, convergence can be seen towards G1 as PMD movement progresses.

For evaluation of the ANN's robustness, 5 able users then had their joystick signals subject to increasing levels of 0-mean Gaussian noise interference during runs to several destinations. Table 2.1 shows the times required to obtain 90% confidence. It can be seen that even under heavy distortion of user input signals, reliance upon the position of the PMD still allows for convergence to the user's true intention within the first half ($45 \pm 12\%$) of overall travel time.

These experiments were run on an Intel i3 laptop with 4 GB RAM; even with such a relatively low amount of computing power, ANN inference time was on the order of milliseconds. Although

σ	Trial				
	1	2	3	4	5
0	21.0	19.2	18.5	22.3	23.7
5	20.2	20.3	19.6	21.8	17.5
10	23.0	22.3	24.4	21.5	23.0
15	24.3	23.8	25.1	22.4	26.0
20	24.7	24.3	25.3	23.9	26.2
25	26.2	25.6	26.8	25.1	27.7

(A) Destination G1 (52s)

σ	Trial				
	1	2	3	4	5
0	12.3	12.0	11.2	14.1	13.0
5	14.2	12.9	14.3	15.1	13.6
10	14.9	16.6	13.2	14.7	15.2
15	16.1	16.3	18.5	16.0	14.8
20	16.5	19.4	16.3	16.9	13.4
25	16.8	15.0	17.0	18.1	15.3

(B) Destination G2 (33s)

σ	Trial				
	1	2	3	4	5
0	3.3	2.9	4.3	3.8	2.3
5	3.2	4.7	2.8	2.6	2.1
10	3.8	4.1	5.3	2.4	2.7
15	3.6	2.4	4.8	4.8	4.1
20	3.7	6.3	1.7	4.8	2.9
25	4.6	4.3	4.4	5.8	3.2

(C) Destination G3 (16s)

σ	Trial				
	1	2	3	4	5
0	20.1	17.6	19.6	22.1	20.2
5	21.6	21.0	23.2	20.4	21.5
10	21.8	20.0	25.0	22.4	22.1
15	25.9	23.6	26.5	23.6	28.5
20	25.7	27.7	26.8	23.4	26.1
25	28.0	26.3	29.8	27.3	27.7

(D) Destination G4 (52s)

TABLE 2.1: 90% likelihood times (s) under varying joystick input noise σ. Times in parantheses represent the mean travel duration from unhindered training samples.

these intention estimates are not inherently sequential, a time series of estimates can be filtered to yield a best estimate at a given moment.

2.3 Global Intention Estimation via Gaussian Process Regression

Unlike ANNs, the predictive GP distribution allows for sequential estimation (Saal et al. 2010) of global user intention. In this Section the square exponential kernel (Eq. C.5) was chosen for two reasons: it has been proven successful in capturing data correlations in a wide range of applications, and its simplicity makes it suitable for the analytical computation within the Bayesian estimation framework.

For an on-line and real-time algorithm of user intention estimation given the current *belief* (probability density function) of the user intention, the following sequential Bayesian estimation problem is considered:

$$p_t(\boldsymbol{g}) \leftarrow \frac{p(\boldsymbol{y}_t|\boldsymbol{g},\boldsymbol{x}_t)p_{t-1}(\boldsymbol{g})}{p(\boldsymbol{y}_t|\boldsymbol{x}_t)}, \tag{2.1}$$

where $\boldsymbol{y}$ is the user action, $\boldsymbol{x}$ is the PMD's pose in the map, and $\boldsymbol{g}$ is the user intention. Unless $p(\boldsymbol{y}|\boldsymbol{g},\boldsymbol{x})$ is a linear Gaussian model, the above problem is analytically intractable. Therefore, approximations need to be applied.

Regarding the GP action model when the prior/posterior $p(\boldsymbol{g})$ is restricted by a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu},\Sigma)$: by exploiting the analytical tractability of the mean and covariance of the predictive distribution, a moment-based filtering technique (Deisenroth et al. 2009) can be applied to derive the following update schemes:

$$\begin{aligned} \boldsymbol{\mu}_t &= \boldsymbol{\mu}_{t-1} + \boldsymbol{C}_t\boldsymbol{G}_t^{-1}(\boldsymbol{y}_t - \boldsymbol{m}_t) && (2.2)\\ \Sigma_t &= \Sigma_{t-1} - \boldsymbol{C}_t\boldsymbol{G}_t^{-1}\boldsymbol{C}_t^{\mathrm{T}} && (2.3) \end{aligned}$$

where $\boldsymbol{\mu}_t$ is the mean of the Gaussian over $\boldsymbol{g}$ at time step t and Σ_t is the covariance. The a-th element of $\boldsymbol{m}_t$, the (a,b)-th element of $\boldsymbol{g}_t$, and $\boldsymbol{C}_t$ can be analytically evaluated respectively for $a,b = 1,2,\ldots,d_y$ as follows:

$$\begin{aligned} m_a &= \boldsymbol{\beta}_a^{\mathrm{T}} q_a,\\ G_{ab} &= \boldsymbol{\beta}_a^{\mathrm{T}} Q_{ab}\boldsymbol{\beta}_b - m_a m_b + \delta_{ab}\left(\alpha_a^2 - \mathrm{Tr}((K_a + \sigma_a^2 I)^{-1} Q_{aa})\right),\\ \boldsymbol{C} &= \Xi - \boldsymbol{\mu} m^{\mathrm{T}}. \end{aligned}$$

Here we define $\boldsymbol{\beta}_a = K_a^{-1}\boldsymbol{y}_a$ where $\boldsymbol{y}_a$ is the training data of y_a. Note that, for simple notation, index t is dropped. The i-th element of q_a, the (i,j)-th element of Q_{ab}, and the a-th column vector of Ξ are defined respectively for $i,j = 1,2,\ldots,N$ where $N = \sum_{\ell=1}^{L} N_\ell$, and $a,b = 1,2,\ldots,d_y$ as follows:

$$
\begin{aligned}
q_{ai} &= \alpha_a^2 \det\Big(\Sigma\big(H_a^{\boldsymbol{g}}\big)^{-1} + I\Big)^{-\frac{1}{2}} \\
&\quad \times \exp\Big(-\tfrac{1}{2}\big(\boldsymbol{\mu} - \boldsymbol{g}^{(i)}\big)^{\mathrm{T}}\big(\Sigma + H_a^{\boldsymbol{g}}\big)^{-1}\big(\boldsymbol{\mu} - \boldsymbol{g}^{(i)}\big)\Big) \\
&\quad \times \exp\Big(-\tfrac{1}{2}\big(\boldsymbol{x} - \boldsymbol{x}^{(i)}\big)^{\mathrm{T}}\big(H_a^{\boldsymbol{x}}\big)^{-1}\big(\boldsymbol{x} - \boldsymbol{x}^{(i)}\big)\Big), \\
Q_{ab,ij} &= \alpha_a^2\alpha_b^2 \det\Big(\Big(\big(H_a^{\boldsymbol{g}}\big)^{-1} + \big(H_b^{\boldsymbol{g}}\big)^{-1}\Big)\Sigma + I\Big)^{-\frac{1}{2}} \\
&\quad \times \exp\Big(-\tfrac{1}{2}\big(\boldsymbol{g}^{(i)} - \boldsymbol{g}^{(j)}\big)^{\mathrm{T}}\big(H_a^{\boldsymbol{g}} + H_b^{\boldsymbol{g}}\big)^{-1}\big(\boldsymbol{g}^{(i)} - \boldsymbol{g}^{(j)}\big)\Big) \\
&\quad \times \exp\Big(-\tfrac{1}{2}\big(\boldsymbol{g}^{ij} - \boldsymbol{\mu}\big)^{\mathrm{T}} R^{-1}\big(\boldsymbol{g}^{ij} - \boldsymbol{\mu}\big)\Big) \\
&\quad \times \exp\Big(-\tfrac{1}{2}\big(\boldsymbol{x} - \boldsymbol{x}^{(i)}\big)^{\mathrm{T}}\big(H_a^{\boldsymbol{x}}\big)^{-1}\big(\boldsymbol{x} - \boldsymbol{x}^{(i)}\big)\Big) \\
&\quad \times \exp\Big(-\tfrac{1}{2}\big(\boldsymbol{x} - \boldsymbol{x}^{(j)}\big)^{\mathrm{T}}\big(H_b^{\boldsymbol{x}}\big)^{-1}\big(\boldsymbol{x} - \boldsymbol{x}^{(j)}\big)\Big), \\
\boldsymbol{\xi}_a &= \sum_{i=1}^{N} \beta_{ai} q_{ai} \Big(\big(H_a^{\boldsymbol{g}}\big)^{-1} + \Sigma^{-1}\Big)^{-1} \times \Big(\big(H_a^{\boldsymbol{g}}\big)^{-1}\boldsymbol{g}^{(i)} + \Sigma^{-1}\boldsymbol{\mu}\Big),
\end{aligned}
$$

where β_{ai} stands for the i-th element of $\boldsymbol{\beta}_a$. In addition, $\boldsymbol{g}^{(i)}$ and $\boldsymbol{x}^{(i)}$ are the i-th training data of $\boldsymbol{g}$ and $\boldsymbol{x}$ respectively, and $\boldsymbol{g}^{ij}$ and R are defined as follows:

$$
\begin{aligned}
\boldsymbol{g}^{ij} &= H_b^{\boldsymbol{g}}\big(H_a^{\boldsymbol{g}} + H_b^{\boldsymbol{g}}\big)^{-1}\boldsymbol{g}^{(i)} + H_a^{\boldsymbol{g}}\big(H_a^{\boldsymbol{g}} + H_b^{\boldsymbol{g}}\big)^{-1}\boldsymbol{g}^{(j)} \\
R &= \Big(\big(H_a^{\boldsymbol{g}}\big)^{-1} + \big(H_b^{\boldsymbol{g}}\big)^{-1}\Big)^{-1} + \Sigma.
\end{aligned}
$$

The computational complexity of the above update scheme is $\mathcal{O}(d_y^3) + \mathcal{O}((d_{\boldsymbol{g}} + d_{\boldsymbol{x}})d_y^2 N^2)$ where N is the number of training data for GP and $d_{\boldsymbol{x}}$ denotes the dimensionality of $\boldsymbol{x}$. In the event that additional trajectories need to be folded into the existing model, re-training can be performed immediately following their demonstration. Although there does not exist a strongly defined bound on the volume of training data required to achieve a certain degree of fitting, it bears noting that performance of can always be monitored and the model updated as necessary with additional data as it becomes available. For real-time estimation, reasonable computation complexity needs to

```
input  : Training data sets $\mathcal{D}_a$ and hyper-parameter sets $\gamma_a$ for $a = \{1, 2, \ldots, d_y\}$, Maximum size of
         subset $\eta$, Size of randomized subset $\kappa$
output: Set of indices $\mathcal{S}$

Initialize index set $\mathcal{I} = \{1, 2, \ldots, N\}$; $\mathcal{S} = \phi$
Set $\boldsymbol{P} := []$.
repeat
    Choose $\kappa$ elements randomly from $\mathcal{I}$: $\mathcal{M} \subseteq \mathcal{I}$.
    Find $\arg\min_{i \in \mathcal{M}} \sum_a Q_a(\tilde{\mathbf{P}} \chi_a(\tilde{\mathbf{P}}))$ where $\tilde{\mathbf{P}} = [\mathbf{P}, \mathbf{e}_i]$.
    Move $i$ from $\mathcal{I}$ to $\mathcal{S}$.
    Set $\mathbf{P} := [\mathbf{P}, \mathbf{e}_i]$.
until $size(\mathcal{S}) = \eta$
```

Algorithm 1: A sparse greedy algorithm for multi-dimensional GP regression.

be enforced. Since the dominant factor in complexity is the number of training samples N, it is reduced by extracting a subset as follows in Section 2.3.1.

2.3.1 Subset Selection to Accelerate Intention Estimation

This section presents a sparse greedy algorithm inspired by Sparse Greedy Quadratic Minimization (Smola and Bartlett 2001) to reduce the number of GP training data samples. Its fundamental concept is to sequentially select a data point from the training set in a greedy manner to minimize the approximation error of the maximum a posteriori probability (MAP) estimate of output y. We can thus reduce the number of training data N used within the GP regression machinery to facilitate faster execution of the intention estimator in real-time while preserving prediction accuracy. The algorithm for multi-dimensional problems is shown in Algorithm 1. In the Algorithm, $\mathbf{e}_i$ refers to the i-th column vector of a $N \times N$ identity matrix, and the functions Q_a, χ_a are defined in Smola and Bartlett (2001) as follows:

$$\begin{aligned} Q_a(\boldsymbol{\alpha}) &= -\boldsymbol{y}_a^{\mathrm{T}} K_a \boldsymbol{\alpha} + \tfrac{1}{2} \boldsymbol{\alpha}^{\mathrm{T}} (\sigma_a^2 K_a + K_a^{\mathrm{T}} K_a) \boldsymbol{\alpha}, \\ \chi_a(\mathbf{P}) &= \left(\mathbf{P}^{\mathrm{T}} (\sigma_a^2 K_a + K_a^{\mathrm{T}} K_a) \mathbf{P} \right)^{-1} \mathbf{P}^{\mathrm{T}} K_a^{\mathrm{T}} \boldsymbol{y}_a. \end{aligned}$$

Q_a is the negative log posterior of training targets $\boldsymbol{y}_a$ given the training input covariance matrix K_a and $\boldsymbol{y}_a$'s associated measurement noise σ_a^2, while χ_a is a solution that minimizes Q_a.

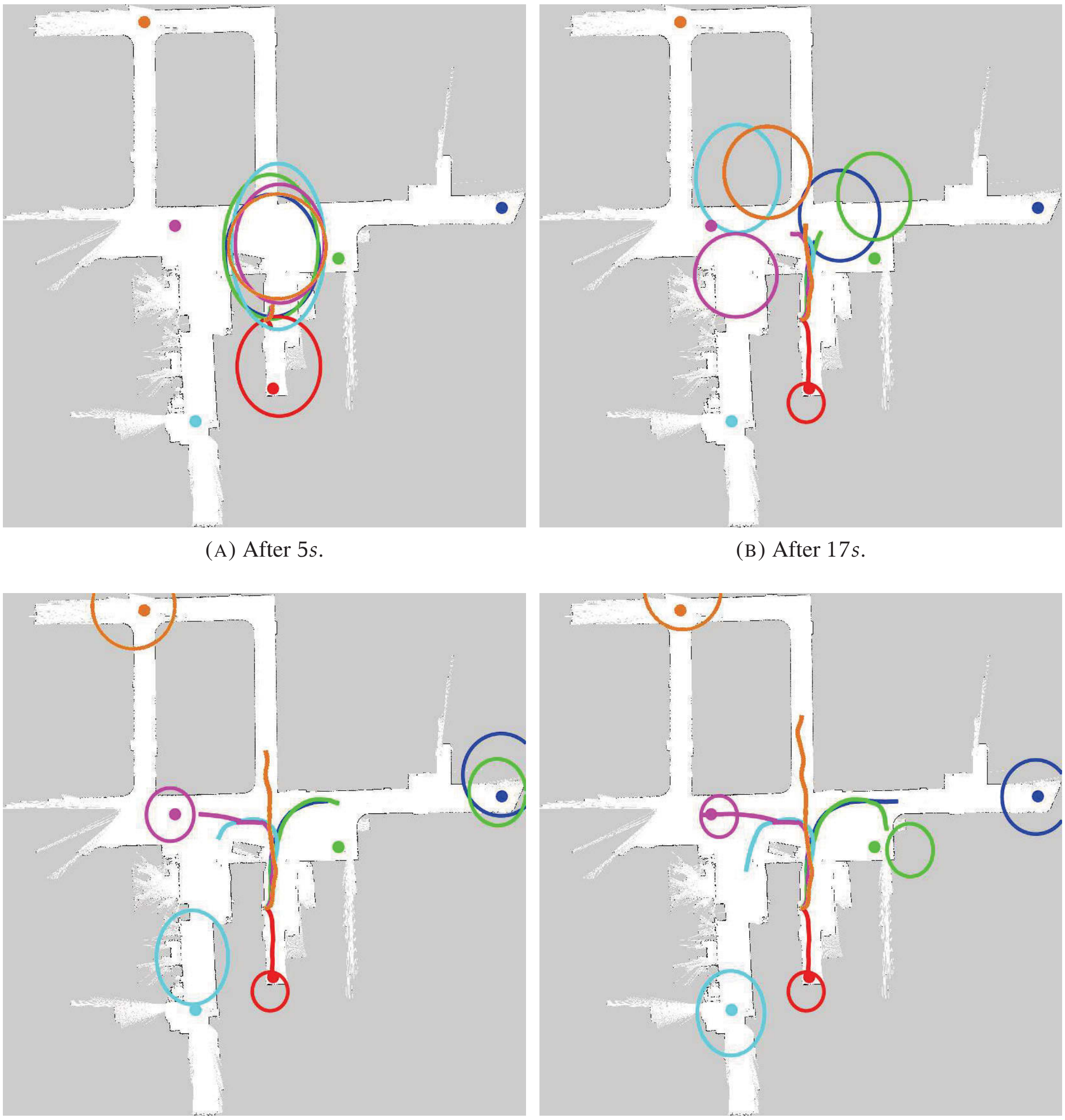
2.3.2 Experimentation

The intention estimation problem is defined as follows:

- User action $\boldsymbol{y} = [y_v, y_\omega]^T$, where y_v and y_ω are the forward and horizontal joystick positions respectively.
- The state of the wheelchair in the environment is defined as $\boldsymbol{x} = [x_r, y_r, \theta_r, v_r, \omega_r]$, again from Gmapping. x_r and y_r represent the wheelchair's 2D Cartesian position within the map, and θ_r is its orientation. v_r and ω_r are the wheelchair's linear and and angular velocities.
- User intention is defined by $\boldsymbol{g} = [g_x, g_y]$; the map co-ordinates of the user's intended destination.

Five out of six runs to each destination in Section 2.2 for a total $N = 9513$ was used for training both a full GP model and a sparse GP with $N = 800$. The remaining runs to each destination were used for evaluation. GP hyper-parameters (Eq. C.3) were learned via Differential Evolution (Storn and Price 1997). Figure 2.6 shows GP performance for the full model at several instances along each run, while Figure 2.7 shows nearest-neighbour classification results over time. Initially the estimated ellipses do not yield much information. Then as paths begin to diverge past the $15s$ mark, the ellipses begin to move towards their true positions and eventually they all correctly converge with reduced covariances.

At $17s$, the estimate for the cyan path eventually arriving at G4 has the nearest neighbour of G5 (Fig. 2.6b). By this point the cyan and magenta paths traversed so far are still quite similar. However over the next ten seconds the user indicates their desire to turn counter-clockwise with the PMD responding accordingly (Fig. 2.6c), and as a result the GP's estimate shifts down towards G4. A similar case of ambiguity being resolved over time can be observed for the green ellipse in Figures 2.6c-2.6d. This supports the notion mentioned at the end of Section 2.1.2 that uncertainty between destinations still sufficiently far away from the PMD can be considered to be acceptable in cases where one destination lies close to demonstrated paths leading to some other destination further away, or when performing estimates at positions in the map where there is significant overlap between demonstrated paths. In both cases, resultant PMD behaviours would still align with the user's expectations at least until an updated intention estimate becomes available.



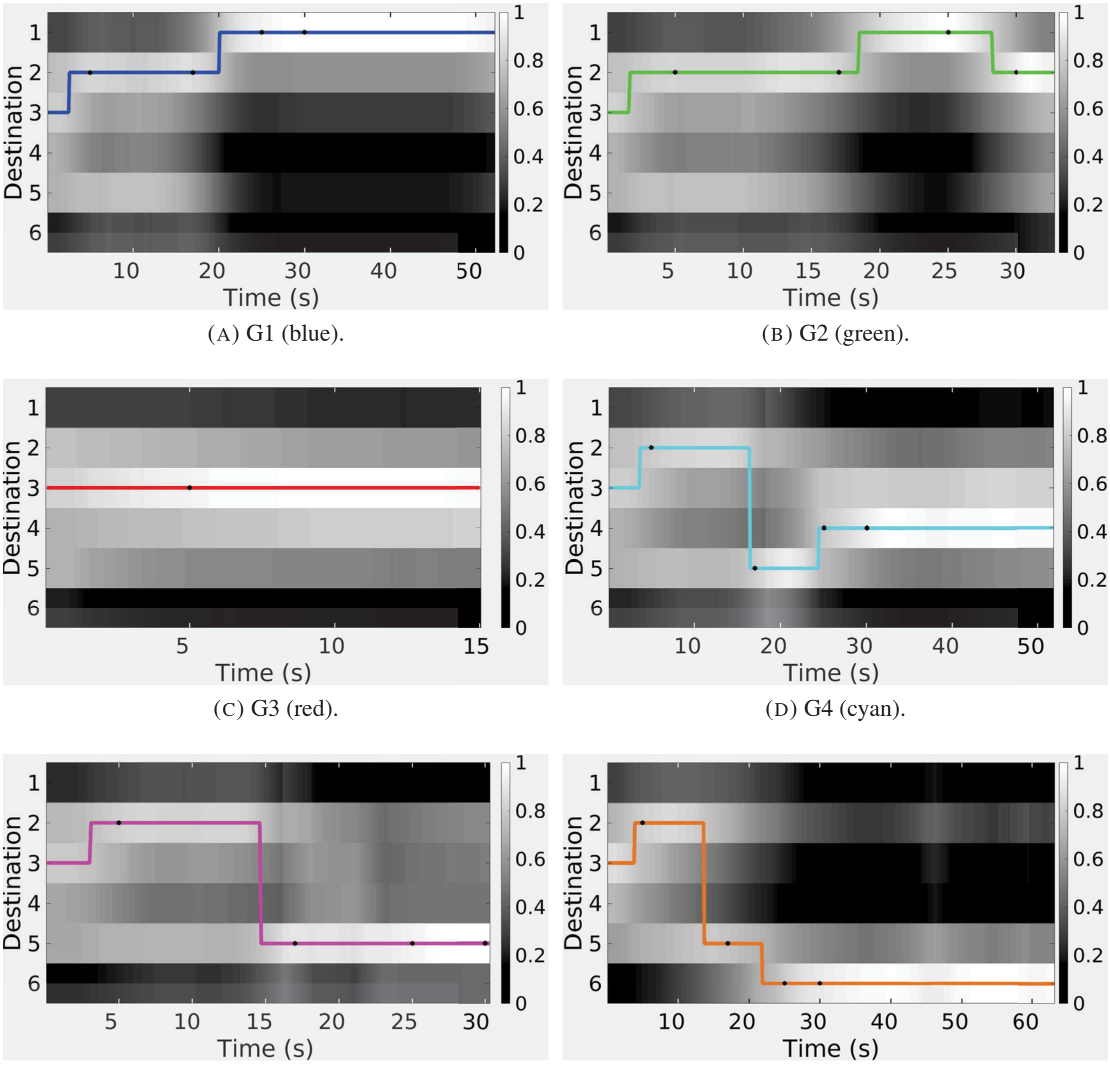
(A) After 5*s*.

(B) After 17*s*.

(C) After 25*s*.

(D) After 30*s*.

FIGURE 2.6: Illustrative examples of user intention estimation with the full GP user action model. The six goals are indicated by points of different colours, with their corresponding trajectories and GP ellipses in matching colour. By 30*s*, all the ellipses have almost reached their goals with reduced covariances.



(A) G1 (blue).

(B) G2 (green).

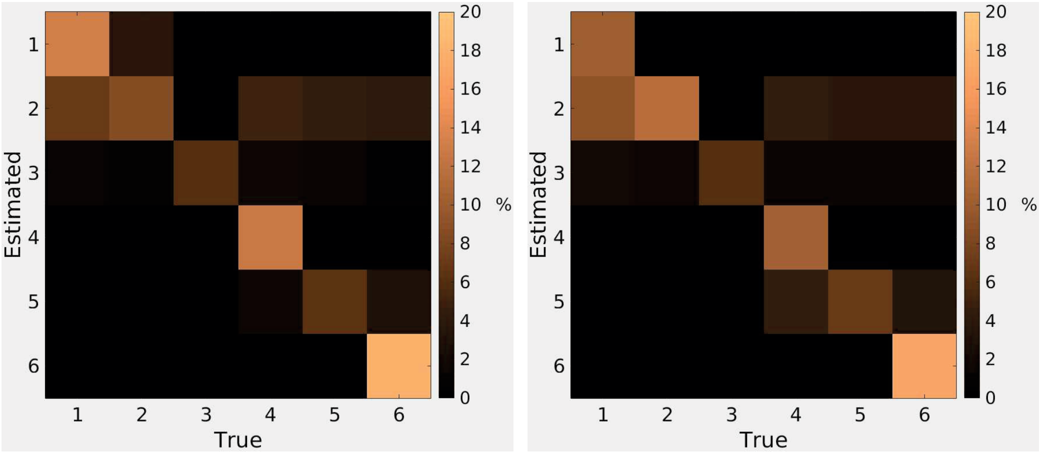
(C) G3 (red).

(D) G4 (cyan).

(E) G5 (magenta).

(F) G6 (tan).

FIGURE 2.7: Convergence of GP estimates for each validation run in Figure 2.6. Likelihoods shown in gray are derived from Mahalanobis distances of the GP estimate to each potential destination. Coloured lines plot the likeliest goal (nearest neighbour), with black marks along the lines indicating $5, 17, 25, 30s$.



(A) Full GP model (65% accuracy).

(B) Sparse GP model (62% accuracy).

FIGURE 2.8: Confusion matrices over all 1795 testing samples.

	Full GP (N=9153)	Sparse GP (N=800)
CPU	10.575 ± 0.34	0.041 ± 0.006
GPU	0.866 ± 0.062	0.028 ± 0.001

TABLE 2.2: Average computation time (s) for a belief update.

2.3.2.1 Comparison with Sparse Model

Figure 2.8 shows confusion matrices for both full and sparse GPs over all 1795 testing samples. The full GP achieved an overall 65% accuracy. The sparse GP achieved an overall accuracy 62% and an 83% agreement (Cohen's kappa $\kappa = 0.78$) with the full GP, indicating comparable behaviour. It bears noting that these accuracies include estimates over the initial 0-15s which feature a high level of ambiguity between most of the destinations.

An Ubuntu laptop with 16 GB RAM, Intel i7-6700HQ CPU, and Nvidia 980M GPU was used for this evaluation. The computation time for belief updates with CPU and GPU are shown in Table 2.2. From these experiments, the sparse GP user action model on either architecture is sufficiently fast to carry out the necessary computation for a relatively large environment such as this in an on-line manner.

2.4 Chapter Summary

This Chapter presents a sequential estimator for longer-term intention estimation for PMDs, taking the form of a sparsified Gaussian Process regression model. The GP is trained upon the pose of the PMD within a known occupancy map along with the user input signal, to yield a likely destination within the map. This modelling is also initially evaluated in the form of an Artificial Neural Network classifier. However such global approaches are only suitable to places which are not subject to regular restructure and preferences are unlikely to change so that *a-priori* map upkeep remains viable, for example within an individualized home setting as opposed to larger-scale public environments such as a shopping complex. Due to this reliance upon static maps and destinations, generalization to new environments is impossible without re-training. Scalability is an additional issue regardless of the underlying inference mechanism, given that destinations are static points. Covering all potentially meaningful destinations for a given area within a range of realistic operating scenarios can quickly become intractable; for example the numerous points required in an ordinary kitchen to enable easy access to individual utilities already creates the need for a substantial quantity of training data, let alone taking into account other amenities typically distributed throughout a domestic setting.

To address these limitations, in the next two Chapters we instead turn to modelling short-term intention estimations by instead considering the immediately available environmental observations as perceived by on-board sensors. In terms of modelling the PMD pose is essentially replaced with sensor data, and static destinations are replaced with traversal likelihood information within the PMD's immediate space. Chapter 3 details a heuristic approach for deterministic local intention estimation followed with expert-stylized path planning, while Chapter 4 merges intention and pathing with models yielding continuous probability maps of estimated traversal likelihood.

Chapter 3

Local Intention Estimation for Mobility Aids

This Chapter details work on local intention estimation for PMDs and the modelling of path planning behaviours from expert demonstration, and documents experimentation both in simulation and on the CAS PMD. The contents of this Chapter encompass the following publications: Cui et al. (2016b); Poon et al. (2017, 2018a).

Algorithms to identify situations with which PMD users experience difficulty and would serve as potential candidates for global intention allocation, such as relatively narrow doorways, have also been developed for assistance to be provided as they are encountered (Derry and Argall 2013). However the classification of what is sufficiently interesting to serve as a destination is still a heuristic determined situationally by the end-user or a carer, and the detection algorithm itself may perform unreliably when taken out of the areas it was attuned towards due to factors such as unforeseen changes in physical layout or illumination. A question that naturally arises from such works concerning the design of heuristic triggers for spurts of active navigation is then, why not have active assistance all the time?

We aim to adapt the global intention estimation from the previous Chapter so that *a-priori* mapping and long-term localization is no longer required, by instead considering user signals and immediately available environmental observations from sensors on-board the PMD. The motivation of this change is to enable functionality beyond the environment(s) from which training data was curated,

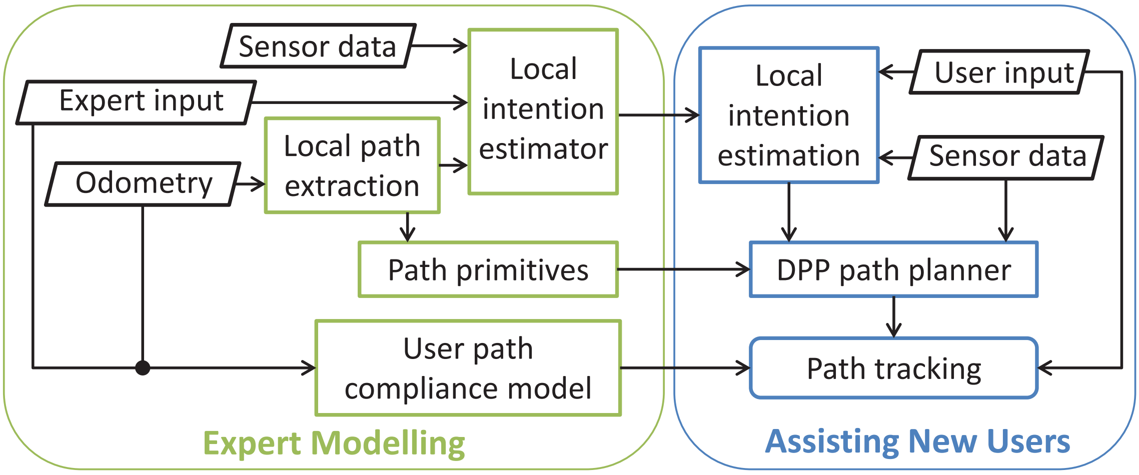


FIGURE 3.1: Overview schematic of the locally assistive framework.

and the scalability issue with static goals. Essentially, the input to the overall learning model is comprised of a user signal and measurements from on-board sensory apparatus suited for robotic navigation such as a planar laser scanning unit (LIDAR). The estimation target of the learning model is now an arbitrary point within the PMD's immediate local space to serve as a short-term navigational objective, rather than one from a limited set within a large-scale map.

The overall objective of the locally assistive framework (Fig. 3.1) detailed in this Chapter is the generation of expert-styled short-term paths, when only given immediate environmental observation and user input. Following a summary of related works in Section 3.1 concerning short-term predictive navigation, Section 3.2 covers the extraction of training data by first decomposing expert demonstration into short local paths. Section 3.3 details the design of a deterministic local intention estimator, followed by path primitive modelling and path planning with consideration to the local environment via Dynamic Policy Programming in Section 3.4. Dynamic Policy Programming is suited to this task due to its optimization with consideration of an external "baseline" policy, which in this work takes the form of a rough path estimated from the built models. Experimentation in both simulation and on the real CAS PMD with a range of test users is then documented in Section 3.5. Quantitative evaluation of both the framework's predictive accuracy and assistive abilities follow in Section 3.6.

3.1 Background

3.1.1 Local Intention Estimation

There has been scant investigation towards the inference of "local" intentions which solely exist in the PMD's frame of reference. The selection of a short-term destination point from joystick orientation was also utilized as a seed for socially-aware global intention-based navigation in Narayanan et al. (2016), albeit in an extremely narrow angular window of approximately 20 degrees about the joystick's orientation (Fig. 3.2).

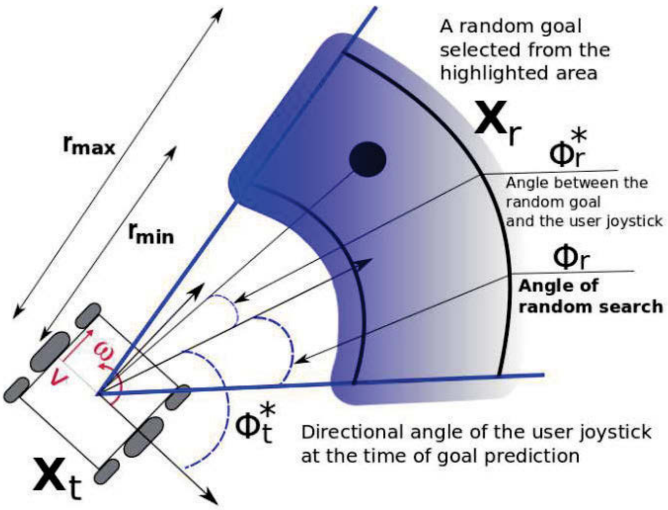
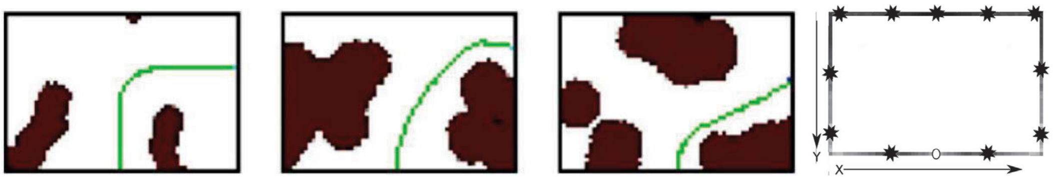

A random goal selected from the highlighted area
X_r
Φ_r^*
Angle between the random goal and the user joystick
Φ_r
Angle of random search
r_{max}
r_{min}
ω
v
Φ_t^*
Directional angle of the user joystick at the time of goal prediction
X_t

FIGURE 3.2: Short-term intention estimation within an angular window for seeding global intention inference. From Narayanan et al. (2016).

Intentions in Ballesteros et al. (2017) are defined as points along the edge of a local planning window (Fig. 3.3). However, we aim to utilize the entirety of the planning window to draw intentions from, while only considering points which are immediately visible to the platform and not concealed by environmental obstructions. This approach is taken in order to avoid reliance upon future sensor measurements for intention inference; important for operating in novel environments where the availability or structure of future measurements is unpredictable.

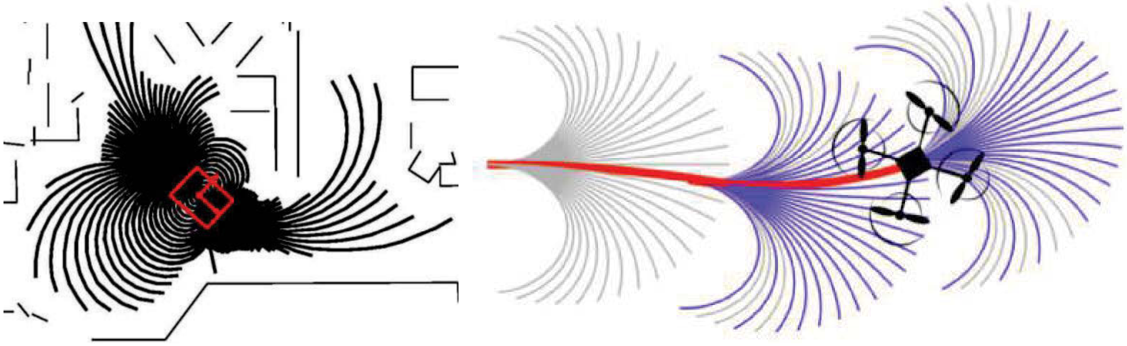
A different approach is proposed by Huntemann et al. (2013) where the immediate intention is taken as the likeliest path from a set of arcing paths (Fig. 3.4a), with likelihoods of paths determined from the robot's position within the map and user input. The concept of learning likelihood distributions over a set of these arcing "velocity primitives" was also more recently utilized in



(A) Paths ending in similar positions.

(B) End-point clustering.

FIGURE 3.3: Intentions along the edge of the local map (B) modelled via short demonstration paths (green lines in A) terminating close by. From Ballesteros et al. (2017).



(A) Projected within a mapped space. (Huntemann et al. 2013)

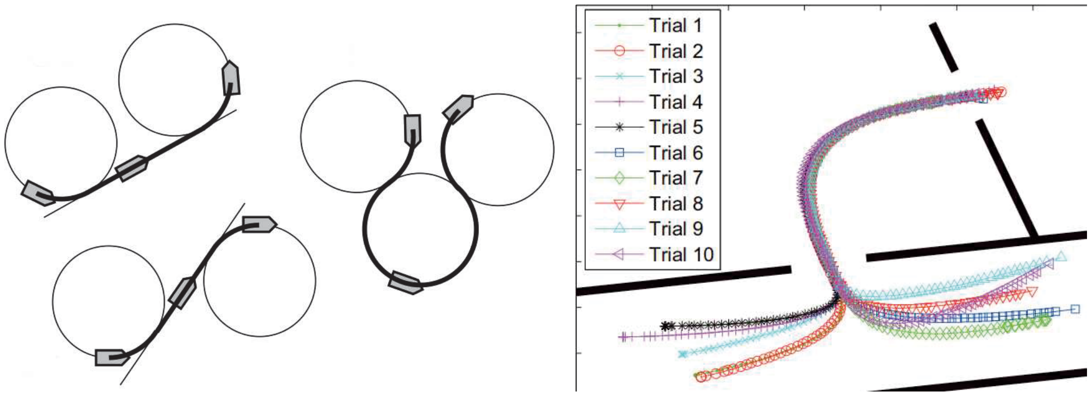
(B) Concatenated for navigation. (Yang et al. 2017)

FIGURE 3.4: Examples of velocity primitives for PMDs (left) and micro-UAVs (right), obtained by forward-projecting linear and angular velocity pairs.

Yang et al. (2017), albeit for micro-UAVs (Fig. 3.4b). Although these arc paths remove the need for path planning as a next step, they do not consider the more complex maneuvres that PMD users need to undertake for challenging navigation tasks within restrictive spaces as the actual user intention in space is not taken into account. As a result, the use of velocity primitives is arguably closer to that of the reactive methodologies outlined in the previous Chapter.

3.1.2 Path Planning from Demonstration

For PMD travel to be comfortable to the potentially frail user on board, the paths followed should be smooth in order to minimize jerk in platform movement. Naturally styled paths can be achieved from algorithms originally applicable to autonomous mobile robots such as Dubin's Paths (Furtuna and Balkcom 2010) or Bezier splines (Chen et al. 2012; Wang et al. 2012) (Fig. 3.5). However when operating in real environments with complex obstacles, these methods designed for cases of no obstacles or "clean" environments only consisting of walls would rapidly become undesirably elaborate due to requiring significant amounts of additional engineering to meet adequate robustness under such conditions.



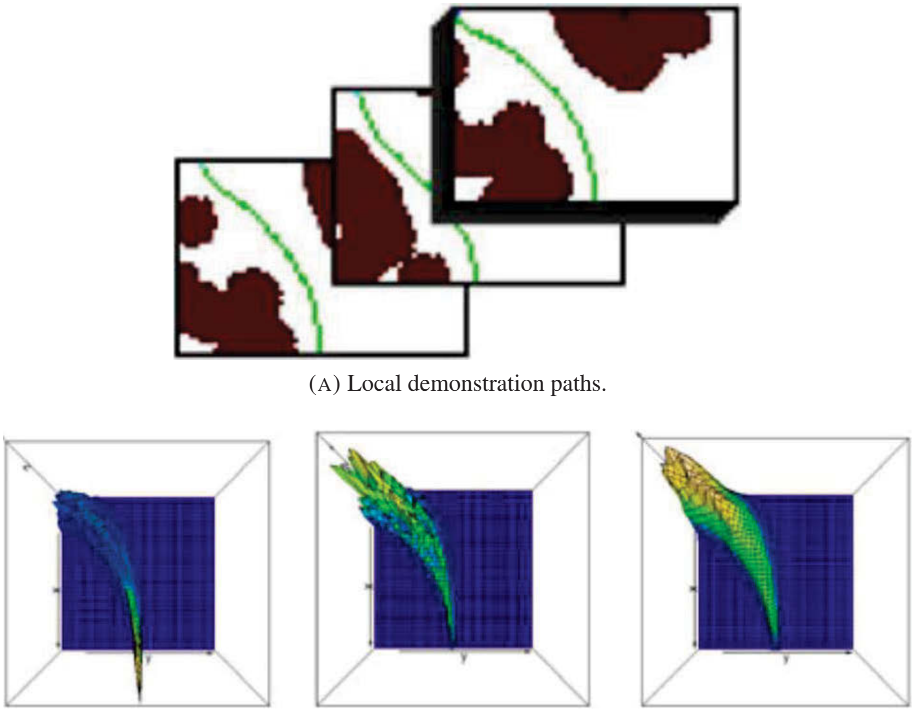
(A) Paths for a Dubin's car model. (Furtuna and Balkcom 2010).

(B) Fitted Bezier splines as paths (Wang et al. 2012).

FIGURE 3.5: Examples of parameterized smooth path planners.

On the other hand, demonstration learning for path planning is a much less explored area by comparison, with arguably the most common approach being the use of path primitives as seen in Park et al. (2008); Yin and Chen (2014); Jiang et al. (2016); Ballesteros et al. (2017). An example is shown in Figure 3.6 where paths taken through situations sharing similar physical obstacles and exit points are weighted and smoothed, resulting in user-stylized traversal likelihood matrices. Unlike the naturally styled approaches, spatially adapted primitives work within a much shorter planning horizon due to their relatively low geometric complexity, often consisting of a library of basic curves stylized by the demonstrated behaviors observed from training data.

Such primitive paths from demonstration are also utilized within this Chapter, however we additionally take into account the environment around the PMD in the form of a local occupancy

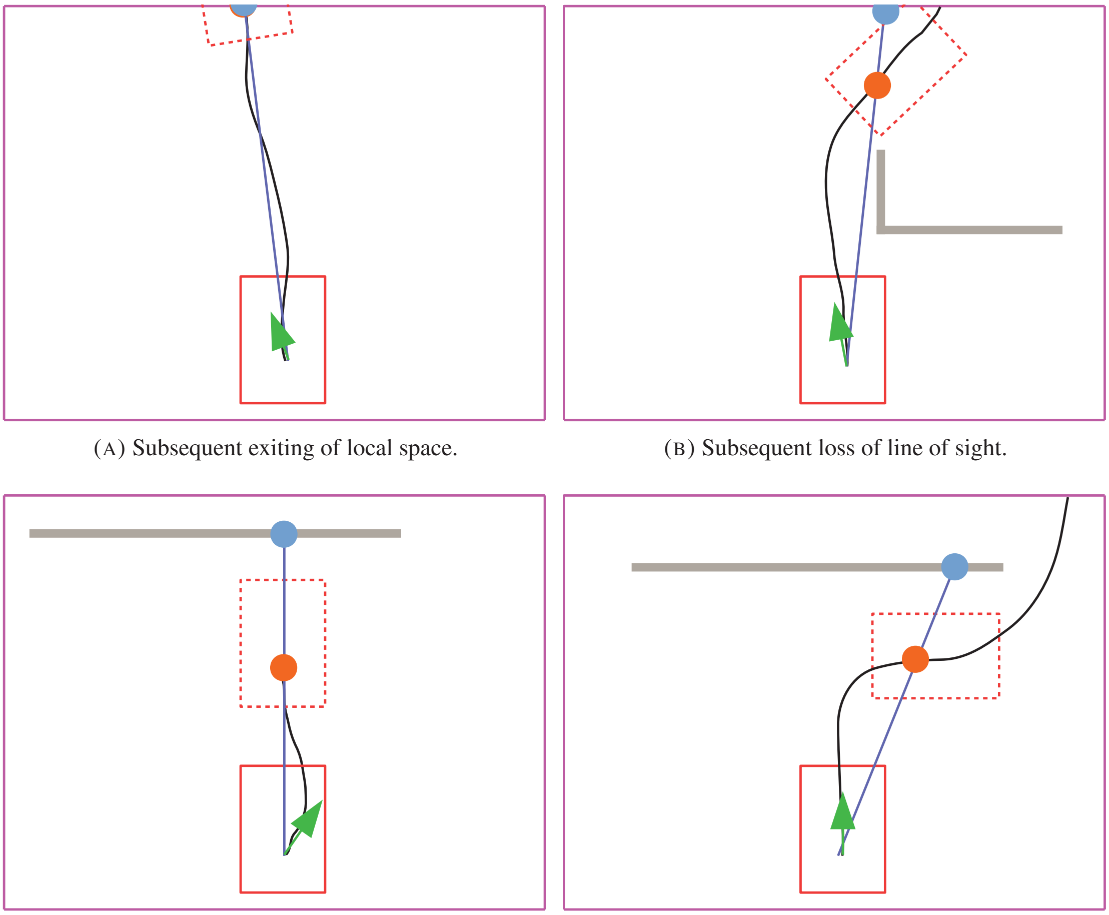


(A) Local demonstration paths.

(B) Merging of similar paths. From left to right: raw traversal likelihood matrix, then after weighting cells based on distance to origin, then after Gaussian smoothing.

FIGURE 3.6: Short demonstration paths (green) are first drawn from training data (top) and combined, resulting in user-stylized path matrices (bottom). From Ballesteros et al. (2017).

grid to enhance safety. We approach the problem of path planning as a grid-world Markov Decision Process (Pashenkova et al. 1996), and use Dynamic Policy Programming (DPP) to plan paths that smoothly combine local occupancy grids with the demonstration path serving as DPP's baseline policy. Regardless of how a path is obtained, path following strategies such as Pure Pursuit (App. E.1) or the Dynamic Window Approach (App. E.2) can then commence.



(A) Subsequent exiting of local space.

(B) Subsequent loss of line of sight.

(C) Inflection in vertical joystick axis.

(D) Orientation change of ≥ 90°.

FIGURE 3.7: Conditions for extracting local paths with origin at the solid red PMD footprint, from a longer expert path (black). Training data at the local termination point (orange point) is recorded, consisting of the expert joystick input (green arrow) at the origin, and the range of the beam (blue line capped with blue circle) from an on-board LIDAR that covers the termination point. Gray contours represent edges of obstacles perceived by the LIDAR, and magenta lines bound the PMD's local path-planning space.

3.2 Extracting Local Paths from Demonstration Data

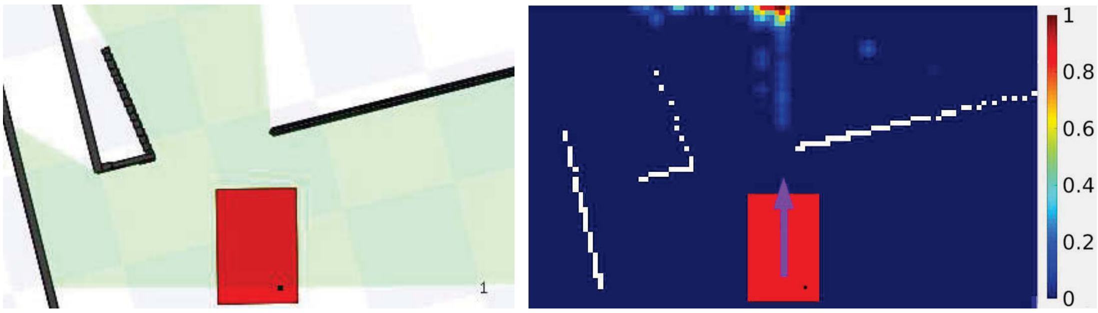
In the case of global intention inference, training data commonly consists of the user driving to each potential destination several times and contains time series information of data such as the user inputs, platform odometry and possibly measurements from on-board sensors for additional consideration. For estimating a local goal $\boldsymbol{g}^*$, there is also the need to break down training data into a workable series of paths. To extract such paths we propose several path termination criteria illustrated in Figure 3.7. For all N instances of training data, a local path is obtained when any of the criteria are met. The training data in this work contains:

- $\boldsymbol{X}_{1,2,\ldots,N}$: PMD pose (x, y, θ), consisting of Cartesian position (x, y) and orientation θ.
- $\boldsymbol{Y}_{1,2,\ldots,N}$: expert actions (2D Cartesian joystick position).
- $\boldsymbol{Z}_{1,2,\ldots,N}$: polar co-ordinates $(r, \theta)_n^{1,2,\ldots|\boldsymbol{z}|}$ of obstacle points across the LIDAR's 180° horizontal field of view.

In determining a suitable local window size for both demonstration path extraction and subsequent path planning, we first consider the physical properties of PMDs. The unit shown in Fig. 1.1a measures approximately 0.8m to the front edge and 0.7m across, thus requiring a significant amount of space to comfortably complete maneuvres such as doorway alignment and approach. In Australia where this research was undertaken, an open circulation space of approximately 2m square is a requirement for building planners (Standards Australia 2009) for spaces such as landing areas near doors for disabled access bathrooms, so we take this as a lower bound. On the basis that we are concerned with only an immediate path to track, it follows that the usefulness of planned path points diminish as path length increases; when we are able to constantly replan paths within a reasonable time period, there is no need to plan too far ahead beyond challenges in the immediate vicinity. Hence the requisite quantity of local space can be thought of as a softly defined region, bounded between the requisite minimum of planning space as implicated by the PMD's physical characteristics and an upper limit beyond which a path planning cycle becomes unacceptably slow. In this work we consider a front portion of a 5m square moving window (magenta rectangles in Fig. 3.7) centred at the PMD, allowing for an acceptable path planning time while also exceeding the lower bound necessary for maneuvrability. It bears noting that the size of this window should be adjusted to meet specific user requirements.

3.3 Intention Estimation

This section covers the estimation of user intention as a goal point $\boldsymbol{g}$ within the PMD's local space, to which expert-styled paths can be planned. We first model an intention within the local window around the PMD as the discretized cell with the highest likelihood of being a local path termination point. A cell size of 0.05m square is taken to allow for an acceptable path planning time. Rather than attempting to match new LIDAR data $\boldsymbol{z}^*$ to $\boldsymbol{Z}$ in their entirety as this would be



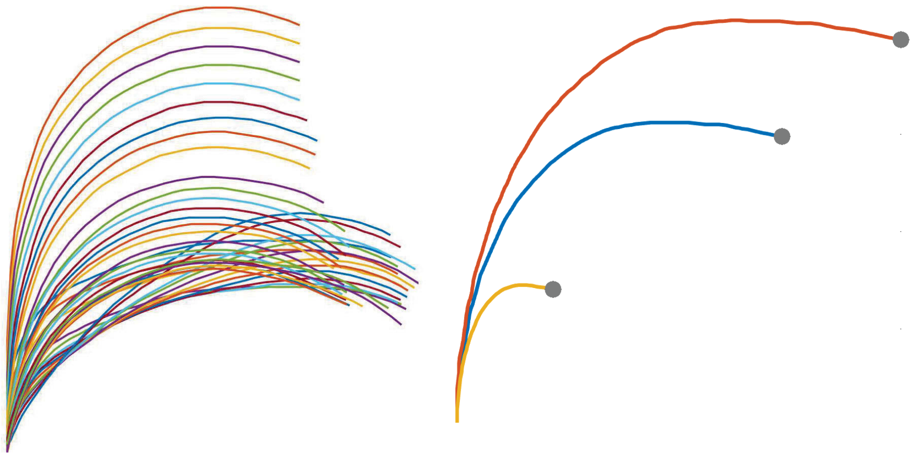
(A) PMD simulated via Player/Stage, viewed from above.

(B) Resultant intention heatmap from forward joystick input.

FIGURE 3.8: Example of local intention estimation. A shows a simulated PMD (red) with a laser scan (green) observing nearby obstacles (black). B shows the resultant intention distribution given a forward joystick input (purple arrow) on the PMD (orange triangle), with perceived obstacles in white. Path planning continues in Fig. 3.10.

easily overfitted, we instead consider individual laser scan beams in conjunction with user input $\boldsymbol{y}$. This is done with the aim of both mitigating effects from the curse of dimensionality, and to allow for improved tolerance to large variation in environmental structure while retaining some information on the latent relationship between $\boldsymbol{z}$ and the sense of space the expert afforded the PMD. For each path indexed $n = 1, 2, ..., N$ a training data tuple for the cell corresponding to goal $\boldsymbol{G}_n = [x^g, y^g]$, the Cartesian point at which the path terminated, is recorded. A tuple consists of $\{\boldsymbol{Y}_n, r_n^i\}$ where i is the index of the beam terminating at each blue circle in Figure 3.7, as determined by $\arg\min_{i=1,2,...,|\boldsymbol{z}|} ||\theta_n^i - \text{arctan2}(y^g, x^g)||$. Beams reaching beyond the local planning space are capped at the boundary of the local window (magenta lines in Fig. 3.7).

Only the edges of obstacles within the local window perceived by $\boldsymbol{z}$ are considered, to remove reliance upon obstacle points from future measurements. Each grid cell then has its own one-class classifier built in the form of a Radial Basis Function Network (App. C.1) utilising a strong 0 prior, as the data can only contain positive examples. Taking all classifiers into consideration and normalizing across the grid cells thus yields an intention likelihood estimate $P(\boldsymbol{g}^*|\boldsymbol{y}^*, \boldsymbol{z}^*)$ given new joystick input $\boldsymbol{y}^*$ and laser range information $\boldsymbol{z}^*$. From the intention distribution (Fig. 3.8), the cell with the highest likelihood is taken as the intended goal point $\boldsymbol{g}^*$. To reduce computational cost in our experiments, only classifiers for cells which recieved training data were queried.



(A) Demonstrated paths sharing similar endpoint orientations.

(B) Paths to gray circles from scaling the resultant primitive.

FIGURE 3.9: A: Paths from training data sharing a similar endpoint orientation are gathered. B: Their resultant primitive can then be spatially scaled to arbitrary positions (gray circles).

3.4 Short-term Path Planning

3.4.1 Path Primitives from Demonstration

From the demonstration data, local paths with similar endpoint orientations are gathered and a primitive is obtained as a least-squares solution after spatially scaling the paths to their average end point. In this work we use 17 discretized endpoint orientions evenly spaced between $\pm\pi$ for creating expert-styled path primitives. 17 was determined to be the upper bound beyond which an orientation would have no data associated with it. For $\boldsymbol{g}^*$ a path primitive is then selected based on the nearest average end point, and is then spatially scaled to reach it. Fig. 3.9 shows an example of a scaled primitive extracted from multiple expert paths.

As opposed to the arc paths in Yang et al. (2017), our primitives are in real space instead of the velocity domain in a manner similar to Ballesteros et al. (2017) albeit in a deterministic manner. In the next section the scaled primitive is taken as a basis for path planning, in conjunction with environmental information.

3.4.2 Path Planning via Dynamic Policy Programming

DPP (App. D.2) is used as a path planning mechanism in this work due to its ability to optimize a Markov Decision Process with consideration to the divergence from a baseline policy. As a result, the baseline and the initial conditions of the MDP become smoothly merged together over the course of optimization.

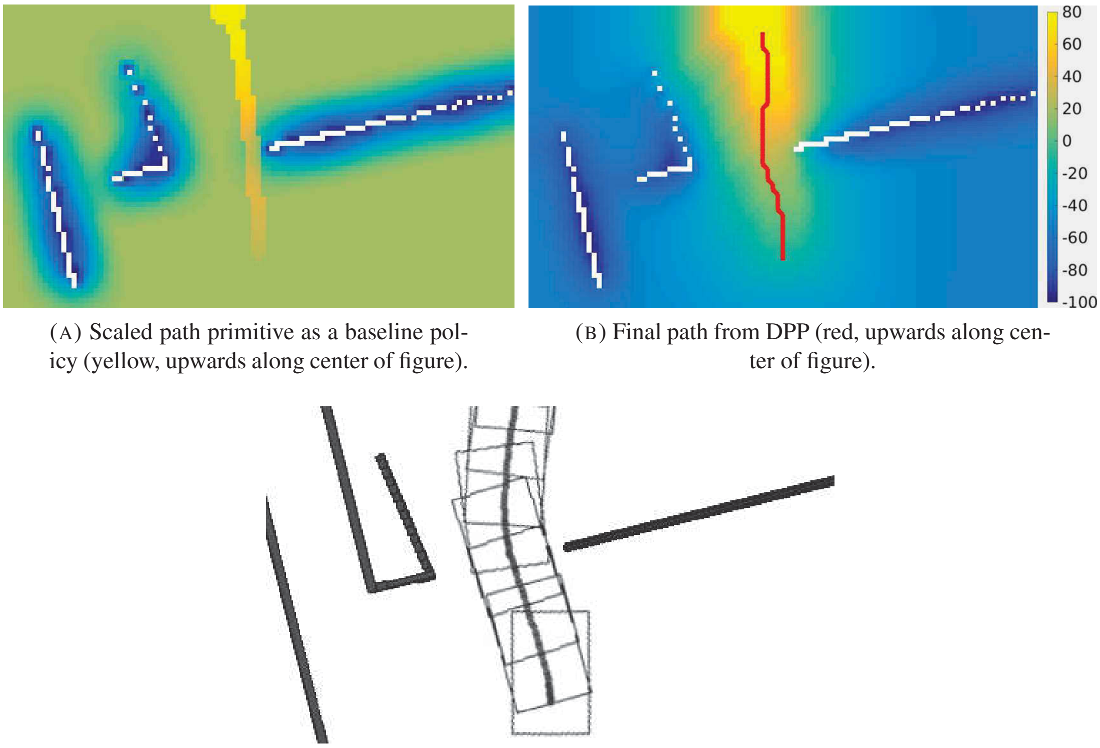
In order to use the inferred local path with DPP, it is taken to serve as the baseline (Eq. D.6). Cells along the path receive increasingly positive reward, whereas cells near obstacles perceived by the laser scanner receive an increasingly negative reward. After optimization, the resultant path drawn from DPP's final policy is then followed. An example of path planning can be seen in Fig. 3.10.

Given initial action preferences $\Psi_0(\cdot,\cdot)$, DPP parameters γ, η and the number of iterations K, the process of the path generator is summarised in Algorithm 2. Notation is detailed in Appendix D.2.

```
input : $\Psi_0(\cdot,\cdot)$, $\gamma$, $\eta$, $K$
/* DPP loop                                                    */
for $k = 1, 2, ..., K-1$ do
    for $(s, a) \in \mathcal{S} \times \mathcal{A}$ do
        calculate $s^{'}$, the next state of $s$ under action $a$
        $\mathcal{T}_k \Psi_k(s,a) = r^{a}_{ss'} + \gamma \mathcal{M}_\eta \Psi_k(s^{'})$
        $\Psi_{k+1}(s,a) = \Psi_k(s,a) + \mathcal{T}_k \Psi_k(s,a) - \mathcal{M}_\eta \Psi_k(s)$
    end
end
/* generate path                                               */
$i = 0$; $s_0 \leftarrow$ position of occupancy grid origin
while $s_i$ is within occupancy grid do
    $a_i = \arg\max_{a \in \mathcal{A}} \Psi_{k+1}(s,a)$
    calculate $s_{i+1}$, the next state of $s_i$ under action $a_i$
    $i = i + 1$
end
output: path $\{s_1, s_2, ..., s_{i-1}\}$
```

Algorithm 2: DPP path generator.

Path tracking via the Pure Pursuit tracking algorithm (Sec. E.1) commences immediately once a path is acquired from DPP and transformed into global co-ordinates. This algorithm was chosen primary due to its simplicity, however more complex trackers such as DWA Fox et al. (1997) are also applicable. Pure Pursuit utilises a control point on the path a fixed lookahead distance (0.8m in



(A) Scaled path primitive as a baseline policy (yellow, upwards along center of figure).

(B) Final path from DPP (red, upwards along center of figure).

(C) Resultant PMD movement from path tracking.

FIGURE 3.10: Example of path planning with DPP, from the intention resultant from Fig. 3.8. A scaled path primitive is set as the baseline policy shown in A with a high reward (light yellow) while perceived obstacles (white) from the PMD's point of view are inflated and given a low reward (dark blue cells). Gradient ascent across the reward landscape after DPP optimization yields the final DPP path (red) in B, with resultant PMD movement shown in C.

this work) ahead of the vehicle, used for deriving linear and angular velocities. The consideration of this control point is the motivation for Pure Pursuit's usage in this work for the user compliance model introduced next. The magnitude of the joystick's polar displacement scales the tracker's output linear and angular velocities, allowing the user to always control the rate of PMD motion.

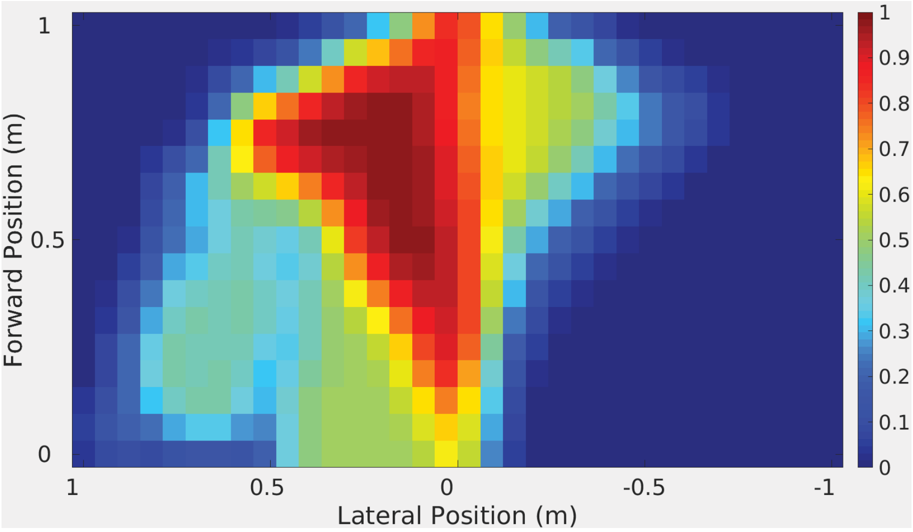


FIGURE 3.11: User compliance map for potential Pure Pursuit control points, from a forward-left joystick signal.

3.4.3 User Compliance for Path Tracking

As a user should be able to terminate a path if PMD behaviour is leading towards a point which does not correlate to the user's true intentions, we define user compliance as the compatability of their input to the path that is currrently being tracked. To achieve this, we consider the control point from Pure Pursuit; given that this point is always within the PMD's reference frame it serves as a suitable objective to assess the compliance of the user's input against, in order to gauge the relevance of the remainder of the current path.

In a similar manner to the construction of the intention estimation model, a compliance model is built for potential control points within the lookahead distance while only considering joystick input, resulting in a compliance estimate in range (0,1) given a control point and y^*. A visualization is provided in Figure 3.11, showing the compliance distribution for potential control points resultant from a forward-left input signal. Either reaching the end of the path or encountering insufficient compliance for a control point (below a threshold of 0.9 in this work) results in a new path request.

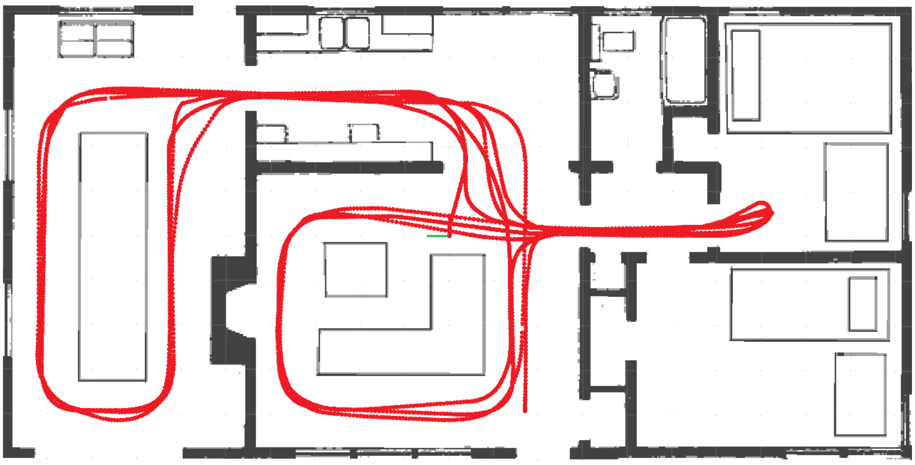


FIGURE 3.12: Training data (red) from an able demonstrator in a 20 × 10m simulated home environment.

3.5 Experimentation

Training data was provided by an able expert in a Player/Stage (Gerkey et al. 2003) simulation of the CAS wheelchair driving inside the home environment depicted in Fig. 3.12. The driver was tasked with navigating throughout the entirety of the environment's accessible space while maintaining safe distances to obstacles and walls. PMD odometry, user input and simulated laser scanner data were logged at 10 Hz. A simulated environment is used here to allow for flexibility in creating suitable environments for the capture of expert behaviours. As demonstrated in Section A.2, the simulation behaves acceptably closely with the real PMD for the purposes of data collection and user interaction.

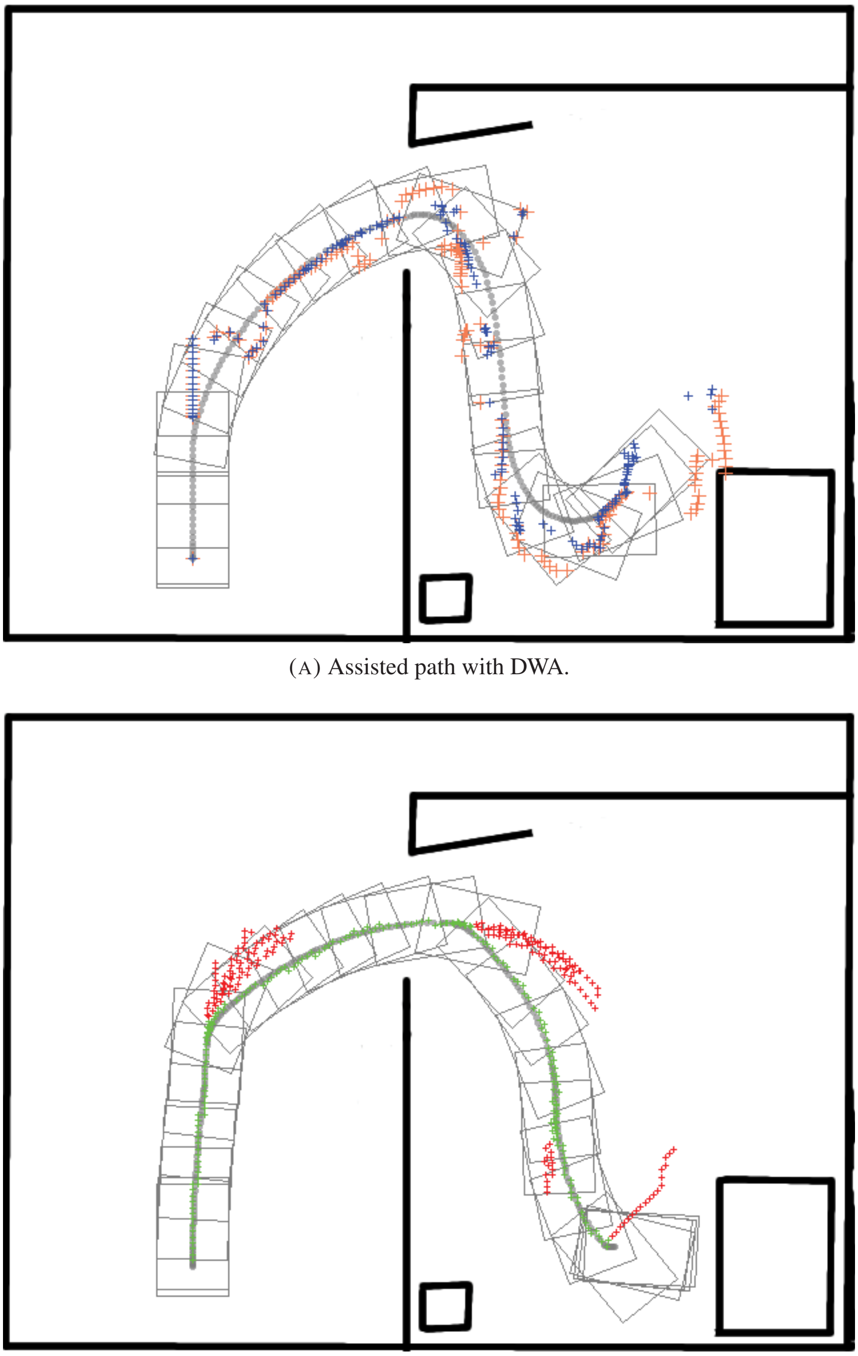
Several Figures in this Section contain assisted trajectories with either the Dynamic Window Approach (DWA) or the framework introduced in this Chapter. In these Figures, gray rectangles represent PMD footprints. In the Figures visualizing the performance of our framework, green marks indicate compliant path points which were followed by the PMD, while the first 50% of non-compliant remaining path points are shown in red. In the Figures visualizing the performance of DWA the projected goals are drawn in orange, and the resultant safe control points from DWA for the derivation of command velocities are drawn in blue. DWA input goal points were obtained by forward projecting the user's input by one second. DWA remains a popular collision avoidance algorithm still widely in use at present, for example in ROS' navigation package (Marder-Eppstein et al. 2010). For an introduction to the DWA algorithm, we direct the reader to Appendix E.2.

Experimentation was conducted on the same machine used in Section 2.3.2.1. An average intention inference and path generation cycle took <0.5s, excluding minor latencies arising from MATLAB/ROS (Quigley et al. 2009) communications. For path planning, a reward of -100 was allocated to obstacle cells and a 1 cell wide border around the planning space. Cells near obstacles received a negative reward from Gaussian blurring of the obstacle reward map with a σ of 2 cells with all other cells set to 0. Cells along the path were then allocated a linearly increasing reward from 10 to 100 from beginning to end. DPP learning parameters γ, η and K were set to 0.99, 0.001 and 100.

3.5.1 Experiments in a Simulation Environment

For comparison of our framework with reactive assistance without additional behavioral heuristics, a disabled volunteer (for details about the volunteer, refer to Sec. 3.5.2) first attempted a simulation task in a basic interior environment, consisting of passing through a doorway and docking at a table. She was first aided by DWA in Figure 3.13a. The user was able to traverse the doorway without difficulty, however the propensity of collision avoidance algorithms such as DWA to be drawn to open space made docking by the table prohibitive despite user commands in the table's direction. In Figure 3.13b she re-attempted the task assisted with our framework. Our framework allowed for both traversal through the doorway and docking at the table.

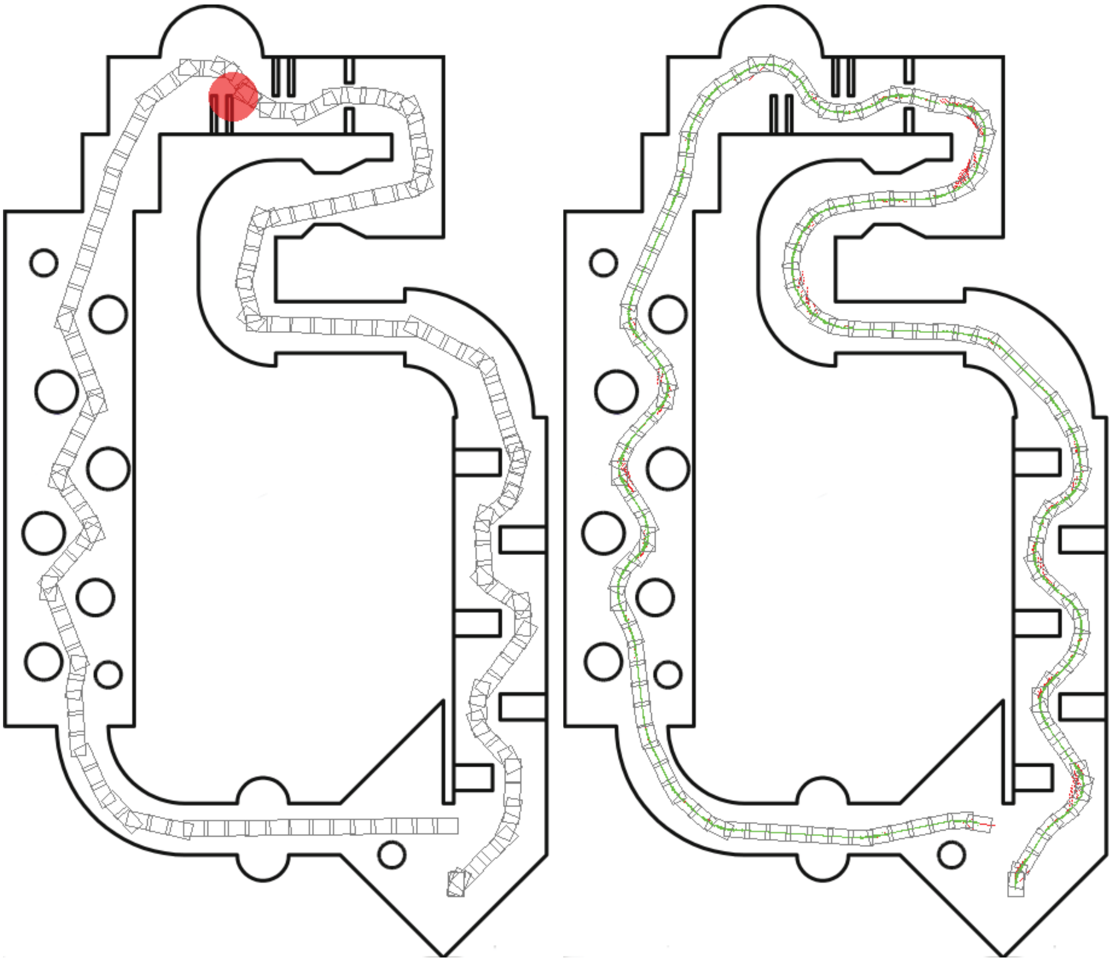
Further evaluation in simulation was undertaken in a larger environment (Fig. 3.14) comprising of a standard course which has also been used for experimental evaluation in Vanhooydonck et al. (2010). PMD users suffering from limb paresis or other motor skill losses, and users of alternative PMD input apparatus such as chin-operated joysticks, are commonly only able to provide rough indications (Vanhooydonck et al. 2010) of desired direction. To imitate such coarseness with able users, their joystick input signals were discretized amongst 5 evenly spaced joystick orientations. 10 test users drove around the test course twice with hampered input signals; once without assistance, and once with the assistive framework in operation.



(A) Assisted path with DWA.

(B) Assisted path with our framework.

FIGURE 3.13: Visual comparison between paths assisted by Dynamic Window Approach and our framework, taken by the disabled volunteer user. A: Projected goals are drawn in orange, and the resultant safe control points from DWA for the derivation of command velocities are drawn in blue. B: path points followed by the PMD are drawn in green, while the first 50% of discarded path points are drawn in red. More details for both are available in Section 3.5.



(A) Unassisted path with collision point(s) highlighted in red.

(B) Assisted path, more details in Section 3.5.

FIGURE 3.14: Visual comparison between unassisted and assisted paths (gray) of an able volunteer with a simulated input disability, driving counter-clockwise in a $21 \times 37m$ simulated environment starting from the bottom-right corner.

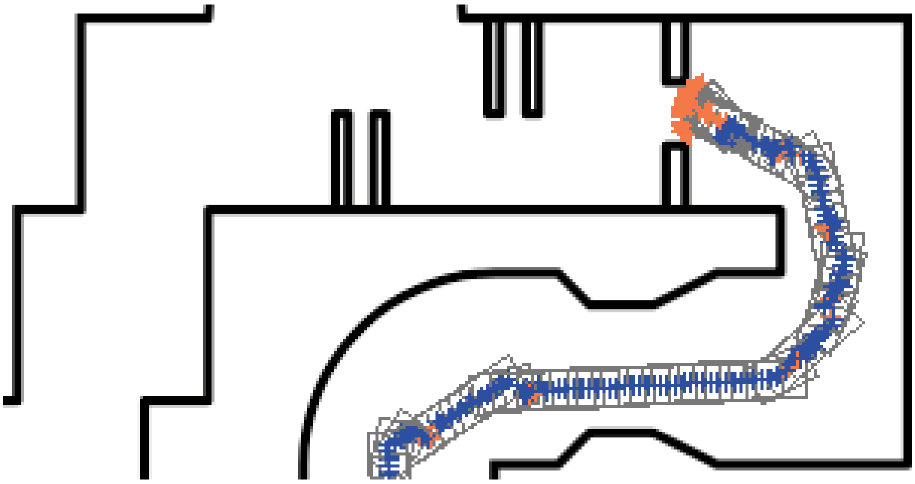


FIGURE 3.15: Assisted path with DWA, more details available in Section 3.5. Here the user was unable to pass through the narrow gap due to a poor initial alignment, causing the PMD to become stuck.

Figure 3.15 shows an attempt on the simulated course assisted by DWA. Although the forward projection of the user input indicates desired traversal through the narrow gap, the reactive linear and angular velocities from DWA were unable to guide the PMD through and as a result the PMD became stuck in an undesirable position which took a reversal maneuvre to resolve.

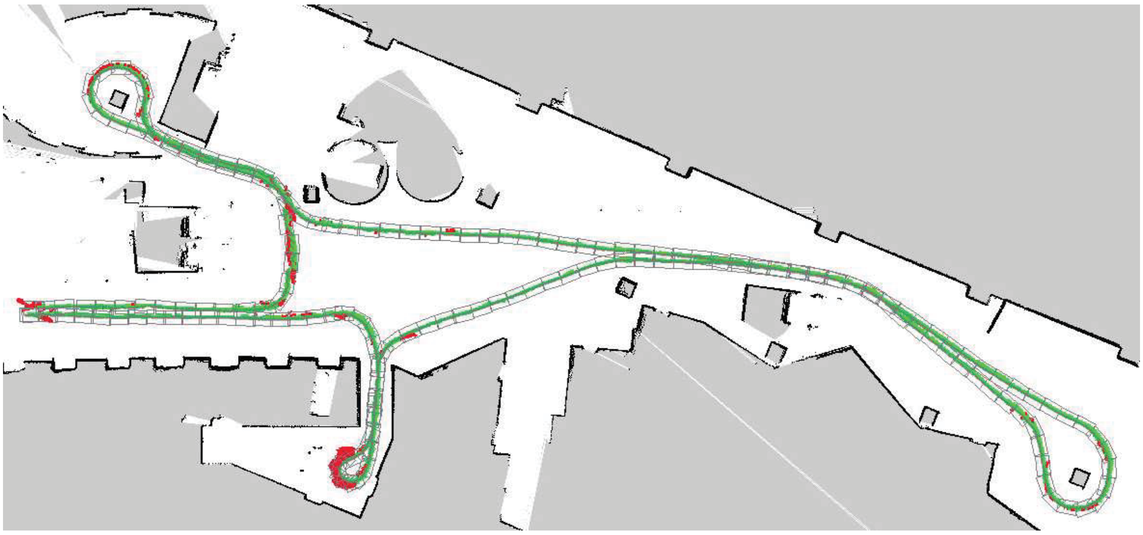


FIGURE 3.16: Visualization of the intention estimator and path generator, run *a-posteriori* for each of the logged points along an able user's free path. More details available in Section 3.5. Conducted on UTS campus in a space measuring approx. $60 \times 27m$, on the CAS wheelchair.

3.5.2 Experiments in Real Settings

PMD localization and map building for the real maps presented in this Section were obtained via Hector SLAM (Kohlbrecher et al. 2011); as in Chapter 2, white regions represent free space, black regions represent perceived obstacles and gray regions are unknown due to occlusions.

Initial evaluation of the intention estimator on the CAS semi-autonomous wheelchair is shown in Fig. 3.16. The intention estimator and path planner was run *a-posteriori* along a route driven by an able user during a map-building exercise on part of UTS campus. Given a path point p generated at time-step t, PMD poses $\boldsymbol{X}^*_{1:T}$ and user inputs $\boldsymbol{Y}^*_{1:T}$, we define incompliance as the compliance estimator falling below the 0.9 threshold used in experimentation for all $(p', \boldsymbol{Y}^*_{t:T})$, where p' is p w.r.t. $\boldsymbol{X}^*_{t:T}$. A root mean square error (RMSE) of $0.1m$ was obtained between PMD positions and generated path points, showing the system's ability to generalize to large open areas despite being trained on data obtained from a relatively restrictive house-like indoor space.

A 63 year old female (Fig. 3.17) volunteered to test the combined assistive framework on the UTS CAS wheelchair. Due to complications from spine injuries caused by a vehicular accident several years before, she is unable to walk without heavy reliance upon a cane and has been considering PMD prescription as she is unfit to drive a car.

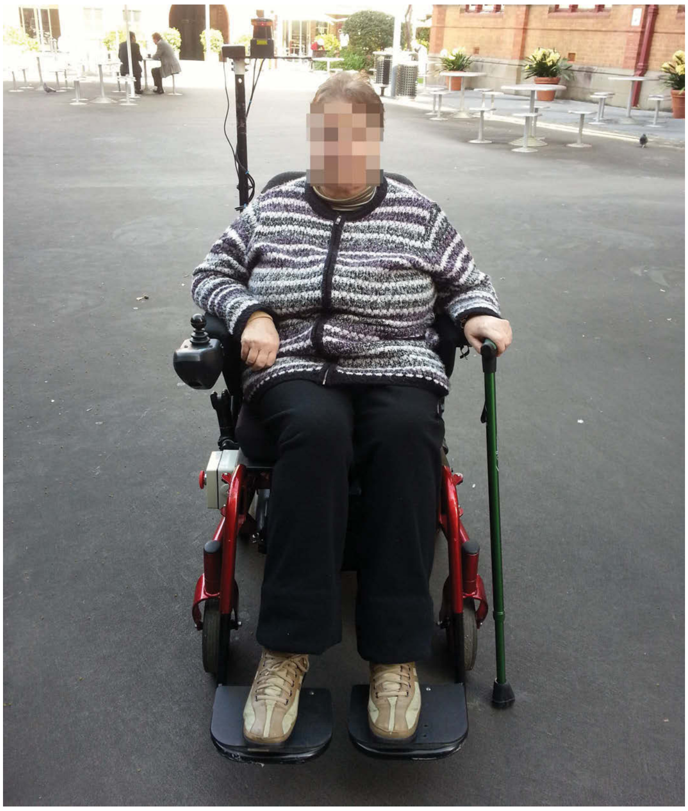
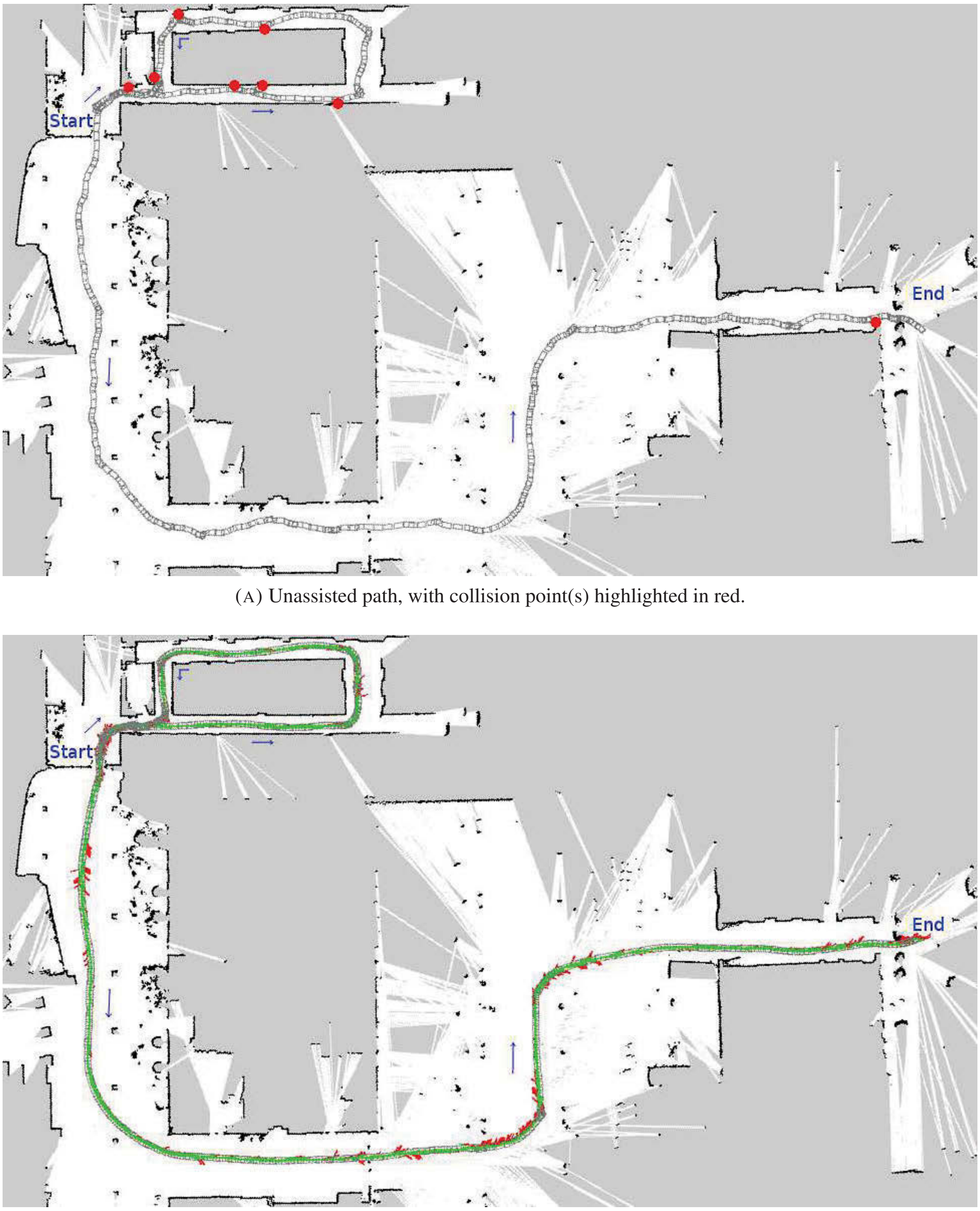


FIGURE 3.17: Disabled volunteer on the CAS wheelchair.

Experimentation [1] on the real platform took place within the publicly accessible campus of a technical college neighboring UTS, following approximately 10 minutes of familiarization after which the volunteer deemed herself sufficiently confident. As shown in Fig. 3.18, the route featured several doorways and narrow corridors, as well as open areas. Foot traffic at the time was sufficiently sparse that no moving entities had an effect on the user or PMD behavior; this is reasonable given both the general tendency of pedestrians to give PMDs a wide berth, and our significantly truncated operating envelope.

When asked about her experience, the volunteer stated that she felt more confident during the attempts with the assistive framework active, as having "something" acting as a buffer between her inputs and the system's behavior gave her the confidence (McGarry et al. 2012) to drive in a less overly-cautious manner. This is noticeable in instances such as docking at the table in Fig. 3.13, where with assistance she felt more comfortable in attempting to dock the PMD closer to the wall formed by the room's bottommost edge. She also received the impression that the system

[1] Video available at `http://youtu.be/EPjV2dCe46E`.



(A) Unassisted path, with collision point(s) highlighted in red.

(B) Assisted path with our framework. For more details, refer to Section 3.5.

FIGURE 3.18: Visual comparison between unassisted and assisted paths (gray) taken by the disabled volunteer on the CAS wheelchair, with rectangular frames representing PMD footprint. The area shown in these figures measures approx. $97 \times 58m$.

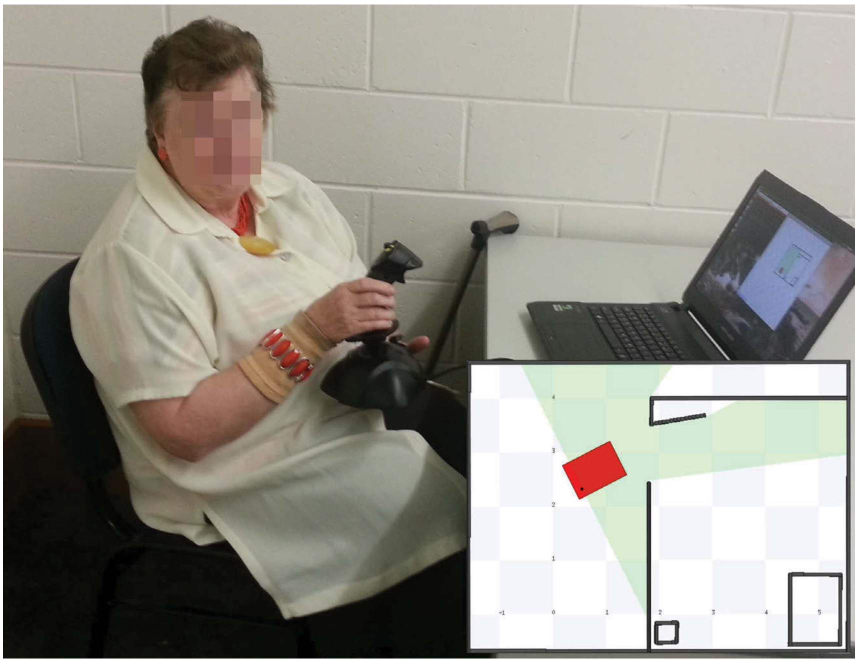


FIGURE 3.19: Disabled volunteer undertaking simulated PMD experiments. A still from Stage is overlaid (details in caption of Fig. 3.8).

was “pulling” the PMD towards empty spaces, most noticeably when passing through particularly restrictive areas.

We were also informed that the volunteer found using the default joystick on the CAS wheelchair to be slightly awkward to use due to its position, and she found herself supporting her elbow and forearm on the armrest to aid her in manipulating the joystick due to her back problems. She felt as if this posturing negatively impacted her driving during the unassisted attempt (Fig. 3.18a). As noted in the study by Esquenazi (2016) the reliance upon forearm support to assist in armrest-mounted joystick manipulation is not uncommon; during the simulated experiments she had preferred to position the joystick closer to her torso when given the freedom to choose its placement (Fig. 3.19).

3.6 Evaluation of Experimental Results

In this section we evaluate the experimental results from the simulation experiment in the standard course (Fig. 3.14) and the real experiments undertaken by the disabled volunteer. All simulated unassisted and assisted runs from the 10 test users were truncated at the point where the PMD first re-entered the starting area. From Urdiales et al. (2013a) we evaluate several task metrics (Table 3.1) that can be calculated without considering the users' deviations to some optimal route, as we uphold that users should not be punished for adhering to their particular driving styles eg. keeping closer to one side in a corridor.

Metric	Unassisted	Assisted
Avg. angular jerk (rad/s^3)	0.07 ± 0.03	0.03 ± 0.02
Time taken (s)	166 ± 25	146 ± 16
Distance travelled (m)	96 ± 1	97 ± 2
Total collisions	5	0
Steering entropy	0.57 ± 0.02	0.55 ± 0.03

TABLE 3.1: Driving metrics from simulation experiment.

Metric	Unassisted	Assisted
Avg. angular jerk (rad/s^3)	0.03 ± 0.03	0.025 ± 0.03
Time taken (s)	566	252
Distance travelled (m)	238	220
Total collisions	8	0
Steering entropy	0.57	0.6

TABLE 3.2: Driving metrics from real experiment.

The same metrics extracted from the real experiment follow in Table 3.2. From the simulation experiments the angular jerk experienced by the PMD when the assistive framework was in operation decreased slightly in comparison to driving unassisted; the paths taken by the PMD also appear visually smoother, particularly noticeable in the slalom portions on the left sides of Figures 3.14a-3.14b. Due to the real test course largely consisting of straight corridors, the jerk experienced by the disabled user was not as significantly affected in the real experiment.

The distances travelled in both experiments are similar between unassisted and assisted, however the course completion times taken by the users in simulation were noticeably shorter with a reduction of over 10%. The difference in completion times is far larger in the real experiment due to several factors. Firstly the user scraped along the walls on several occasions which resulted in significant losses of speed and the need to stop completely to recover. By contrast such side-on contact is not properly handled in the physics of the simulation environment, and so there was no hindrance to movement in the simulated experiments when such events occurred. Secondly when unassisted the user drove at an average linear velocity of 0.6 m/s when moving freely compared to 0.85 m/s while assisted, whereas from the simulation experiments less disparate average linear velocities of 0.59 ± 0.07 and 0.67 ± 0.07 m/s were respectively observed. There were no collisions in both simulation and real experiments when users drove with assistance, whereas a total of 5 and 8 collision events occurred in simulation and on the real PMD respectively.

From these experiments the steering entropy (App. E.3) observed between the able users with simulated disabilities and the disabled volunteer are very similar at approximately 0.57, indicating that the simulated disability provided a level of task impedance comparable to the difficulties due to volunteer's various health issues. However it bears noting that this closeness in steering entropy does not necessarily imply that the nuances of the disabled volunteer's driving behaviours were accurately reflected in the disability that was simulated; rather that the able users experienced a similar difficulty to the disabled volunteer in imparting task effective input signals.

3.6.1 Assistive Framework Metrics

Table 3.3 documents several metrics directly concerning the framework's performance from both simulated and real experiments. The user obedience metric from Li et al. (2011) is defined as $\eta = \exp(-0.003 \times |\alpha|)$ where α is the angular command error (degrees) between PMD and goal. Here we take α to be the angular error to the inferred intention from the last compliant path point.

Metric	Simulated	Real
User obedience	0.87 ± 0.11	0.88 ± 0.09
Path compliance (%)	64 ± 33	56 ± 40
PMD path RMSE (m)	0.04 ± 0.03	0.1 ± 0.07

TABLE 3.3: Framework metrics from both experiments.

The closeness of the user obedience metric to the experimental compliance threshold of 0.9 indicates a strong correlation between intention estimates and PMD behavior for the duration that the framework considered the user's inputs to be acceptably signifying path correctness. Although the observed percentage-wise path compliance and utilisation of each path is arguably equal to or greater than in the work by Huntemann et al. (2007), it is worth noting that this does not necessarily translate into superior overall performance given that both works can always provide new paths that are more aligned to the user's intention at a given instant. The RMSE between compliant path points and PMD positions also indicates reliable performance of Pure Pursuit in serving as the PMD's path tracking controller.

3.7 Chapter Summary

This Chapter presents a locally assistive framework for PMDs with a deterministic short-term intention estimation mechanism (Sec. 3.3) built from demonstration data which has been designed with a substantial quantity of task knowledge. A real-time stochastic optimal short-term path planning strategy (Sec. 3.4) complements the intention estimation, taking into account both the PMD's immediate surroundings and path planning based on demonstration data. The intention estimator is built in a heuristic manner in order to leverage a relatively restrictive amount of training data against a much more complex learning space than the one seen in the previous Chapter. This is due to the high dimensionality of environmental observations from on-board sensory apparatus, and that local intentions can be arguably anywhere within a small window around the PMD instead of a discrete point set. Experiments across a diverse test user base show both good predictive accuracy of the LfD models and the promise of the overall framework in providing adequate navigational assistance. However with a larger demonstration data pool, conventional learning methodologies with a much reduced reliance on task knowledge become viable even with the complex learning space of this problem.

In the next Chapter, we decompose such a large learning space into two individual domains of sensor data and user input. Two separate LfD models yield continuous traversal likelihood maps across the immediate space around the PMD, as opposed to a singular path to a deterministic goal point as in this Chapter. The continuous approach results in an overall more streamlined framework, and removes the need for separately modelled path primitives (Sec. 3.4.1) and user compliance (Sec. 3.4.3).

Chapter 4

Local Intention Estimation for Mobility Aids via Decoupled Observations

This Chapter presents a revised version of the framework described in Chapter 3, aimed at reducing the amount of expert task knowledge leveraged in building the learning models used for locally intelligent mobility assistance. To approach this, we decompose the massive $sensor\ data \times user\ input$ space and instead individually address the much smaller $sensor\ data + user\ input$ spaces with LfD models that handle each data source in an independent fashion. As per the previous Chapter training data is again obtained from expert demonstrators controlling a simulation, with the methodology proposed here also being equally suited to modelling upon real data. The contents of this Chapter encompass Poon et al. (2018b).

The new framework is outlined in Figure 4.1, utilizing a Convolutional Neural Network (CNN) and a Gaussian Process (GP) approach to handle the sensor data and user input respectively. Each model yields a continuous traversal likelihood heatmap around the PMD combining intention estimation and expert pathing, with a local path then being drawn from the fusion of both heatmaps. This framework no longer requires the generation of stylized primitives or a separate compliance model, as path structure is captured by the CNN and a quantitative compliance estimate comes naturally from the GP's heatmaps. Both heatmaps are created as images that are readily compatible with each other despite a stark difference in modality between user input and sensor data sources.

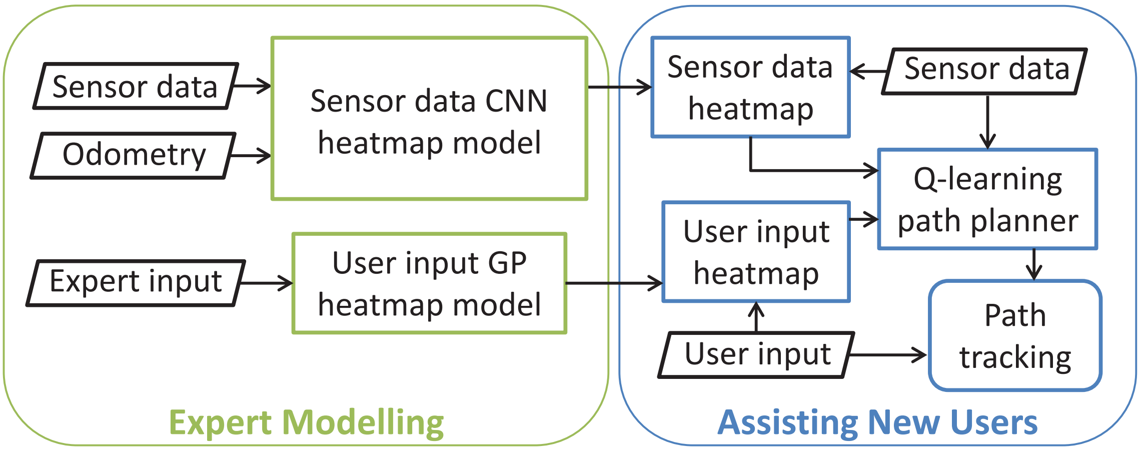


FIGURE 4.1: Overview schematic of the revised locally assistive framework.

An additional benefit comes from their amenability towards image transformation and processing techniques.

Following a brief overview of relevant literature (Sec. 4.1), the framework depicted in Figure 4.1 is detailed. The CNN model for generating heatmaps from sensor data is described in Section 4.3, the heatmap generation via GP trained on user input in Section 4.4, and finally the Q-learning path planner in Section 4.5. Experiments in an on-line assistive setting, and a *a-posteriori* comparison with the methodology described in Chapter 3 and a naive CNN trained on un-decoupled data, follow in Section 4.6.

We note that the planning space in this Chapter is a full $5 \times 5m$ square centred at the PMD's origin, as both heatmap generators are amenable to future sensor configurations providing 360 degree perception around the platform. However given the configuration of the CAS PMD (App. A) with a single forward-facing LIDAR, the empty portions of Figures 4.6, 4.11, and 4.12 corresponding to the unseen space to the rear of the PMD have been cropped.

4.1 Background

The methodology in this Chapter roughly follows a similar strategy to that of Chapter 3; a path planning landscape is inferred from sensor data and user input to be optimized as a MDP which yields a path to be tracked.

For processing sensor data, we aim to transform short-range occupancy maps into traversal likelihood heatmap images. This is done with the modelling of an image-to-image mapping function; common applications of such transformations include the pixel-wise semantic labelling of complex image data for tasks such as scene recognition (Fig. 4.2) and object segmentation (Fig. 4.3). Convolutional Neural Networks (App. C.2.1) are widely used in this area, as seen in Krizhevsky et al. (2012); Simonyan and Zisserman (2014); Szegedy et al. (2015) among others.

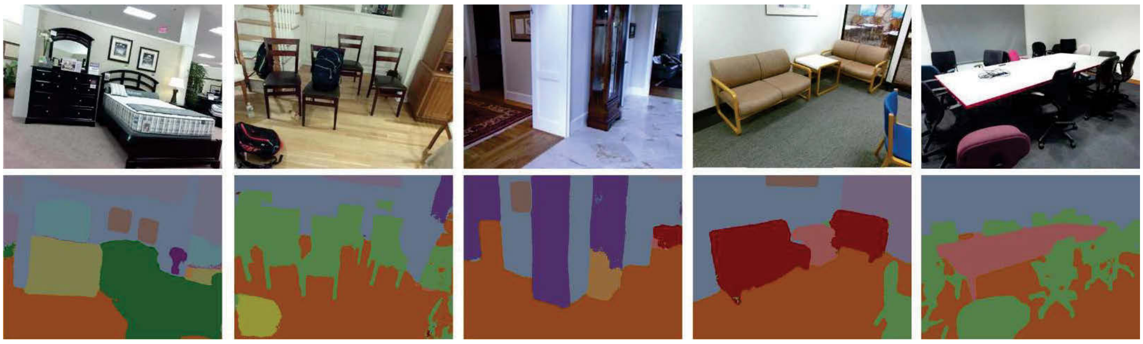


FIGURE 4.2: Example segmentation of image scenes via CNN semantic mapping. From Badrinarayanan et al. (2015).

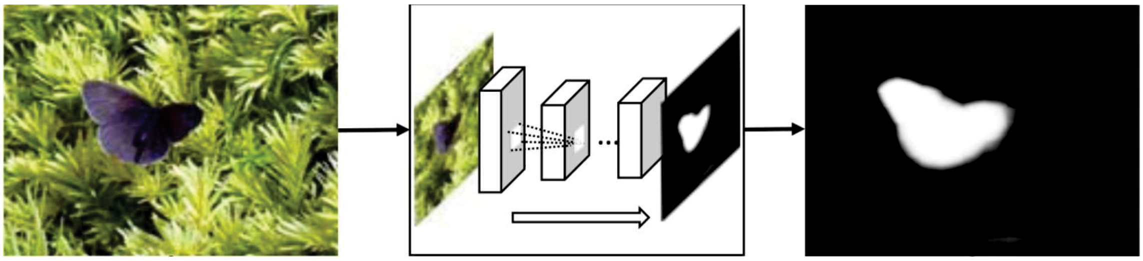
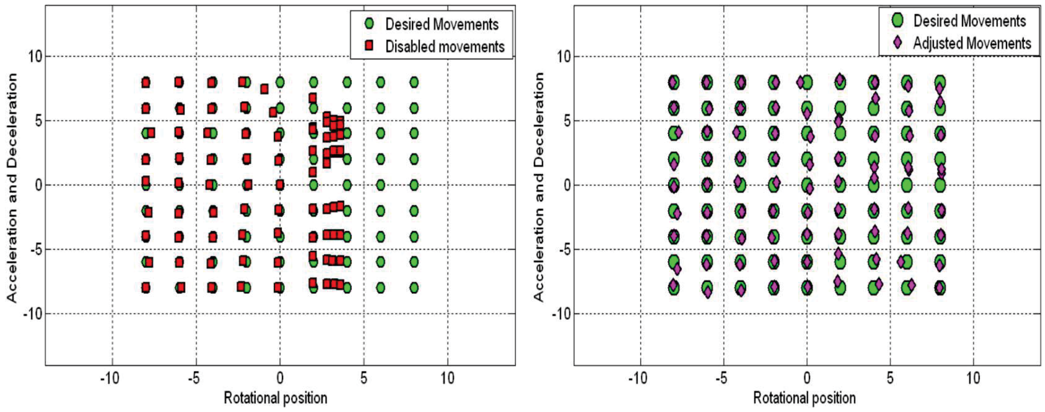


FIGURE 4.3: Estimated saliency map from CNN object detection. From Li et al. (2016).

Other methodologies have also been explored including Gaussian Processes (Freytag et al. 2012) and Support Vector Machines (Varma et al. 2016). However the reliance of SIFT features in the former work make it less suited to this particular task due to the simplistic binary nature of the input images (an example is in Figure 4.7a), and the pixel-wise classification in the latter makes



(A) Training data modelling disabled user inputs to their desired positions.

(B) Resultant ANN remapping.

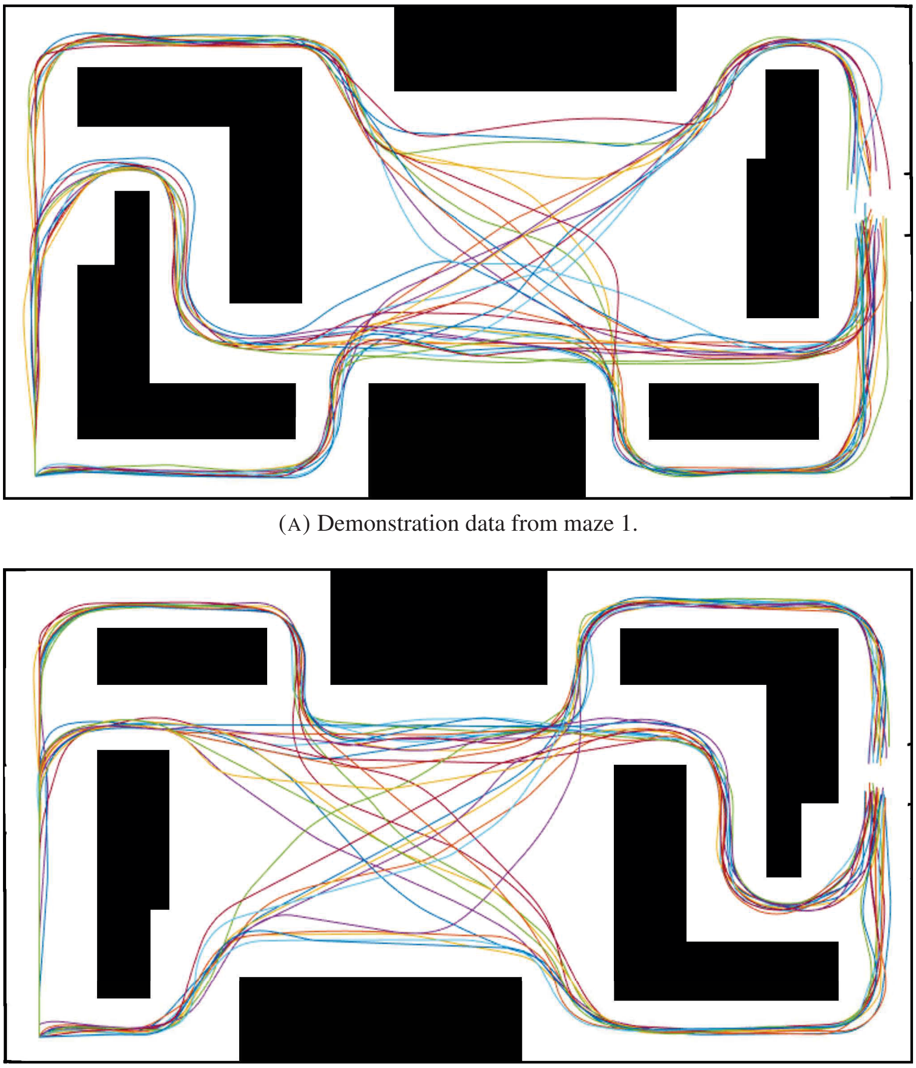
FIGURE 4.4: Deterministic joystick remapping using an Artificial Neural Network for each axis. From Rabhi et al. (2013).

it undesirably slow for real-time applications especially on portable computing systems. As speed is of a high priority in facilitating real-time path planning an optimized CNN library is utilized in this work for the purpose of heatmap generation from occupancy map matrices, where both input and output each possess a high dimensionality.

For the parsing of user input, firstly we model a "desired" signal of the user against their potentially unintentional input via GPs as the uncertainty associated with the resultant signal is required for probabilistic heatmap generation. Such remappings of user input to their latent desired signals have been previously explored in works such as Rabhi et al. (2013) by means of Artificial Neural Networks (Fig. 4.4), where the output was taken directly to serve as a PMD control signal. A GP model was built in Huntemann et al. (2013) to model raw user inputs against their desired linear and angular velocity commands. The estimated velocities were then used for shaping the likelihood distribution over a discrete curvature-based path primitive set (Sec. 3.1). These remapped user inputs can then be taken directly for platform control purposes, or for utilization in subsequent navigational frameworks. A similar approach is taken in this Chapter; here the safe GP signals are utilized for generating probabilistic traversal heatmaps, with the primary difference being that these heatmaps are continuous across the 2D space around the PMD.

4.2 Demonstration Data

For building the models in this Chapter, 8 able users drove around the environments depicted in Figure 4.5 assisted by DWA (App. E.2) for a total of 72 runs. Platform odometry, LIDAR measurements, and both raw user inputs and collision-free "safe" inputs derived from DWA's output PMD velocities were logged.



(A) Demonstration data from maze 1.

(B) Demonstration data from maze 2.

FIGURE 4.5: Training data (starting from bottom-left corner) from 8 able users in a pair of $20 \times 10m$ simulated mazes.

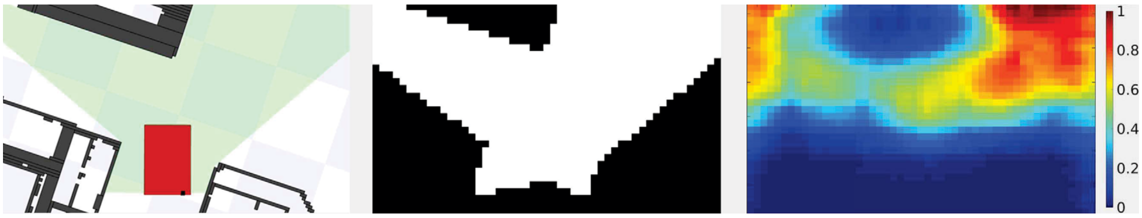
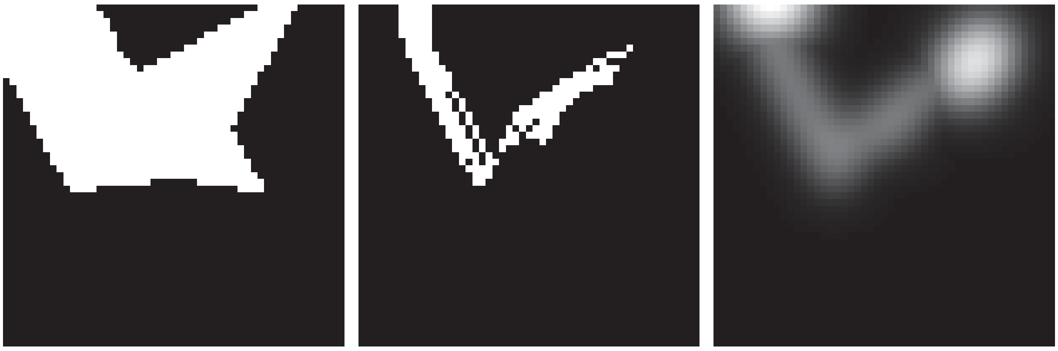


FIGURE 4.6: CNN generation of a traversal heatmap (right) from the occupancy map (centre) corresponding to the LIDAR readings (green) in the left image.

4.3 Sensor Data Heatmaps from CNN

The objective of the CNN heatmap generator is to learn both user intention and pathing from environmental observations alone; an example is shown in Figure 4.6. The inclusion of demonstrated paths removes the need for the primitives from Chapter 3.

A local $5 \times 5m$ free-space map (Fig. 4.7a) is created for each instance of recorded LIDAR data, which serves as input training data for the CNN. A corresponding image consisting of all demonstration paths taken through the occupancy map's visible free space (Fig. 4.7b) then serves as the training target (Fig. 4.7c). Then the entire image is subject to Gaussian smoothing in order to fuse nearby paths together. For the experiments in this Chapter, paths were drawn at 0.5 and end-points dilated at 1.0, against a background of 0.0.



(A) CNN training input.

(B) Visible portions of demonstrated paths.

(C) CNN training target.

FIGURE 4.7: Example of CNN training data. A: Training input consisting of a local free-space map (white) from a single LIDAR sweep, with the PMD's origin at the centre. B: parts of demonstrated paths within the visible local free-space. C: Training target by fusing the visible path portions together and dilating their end-points.

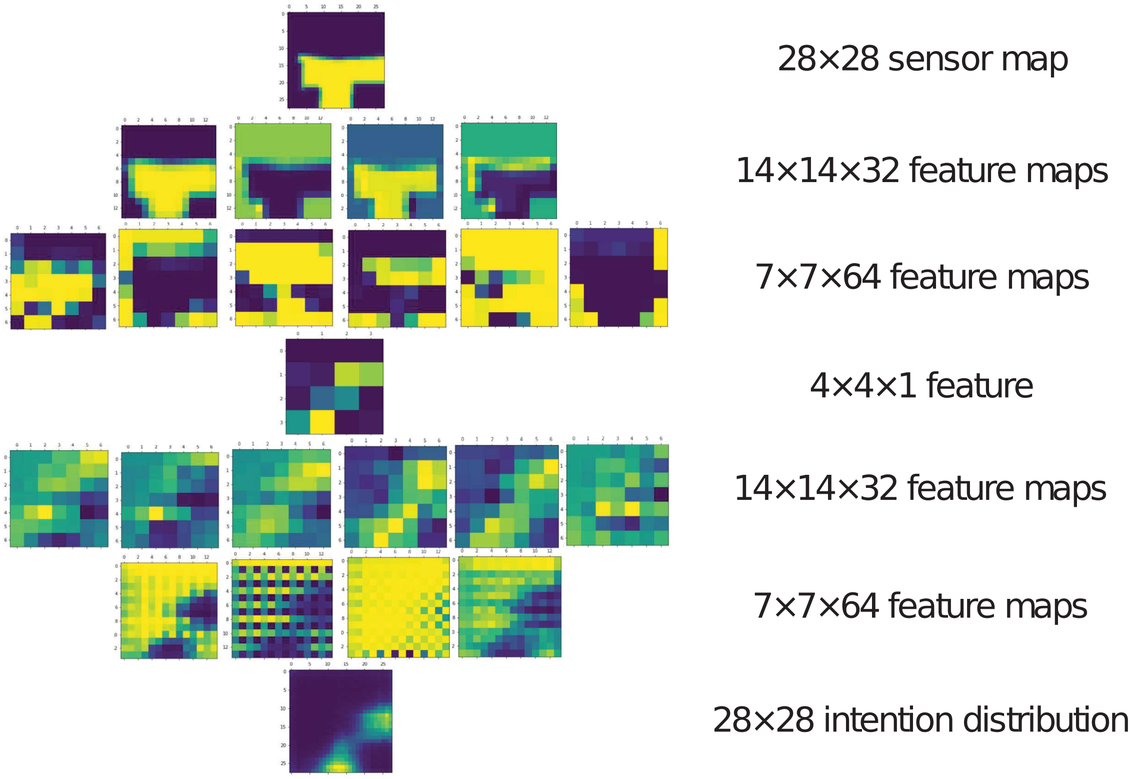


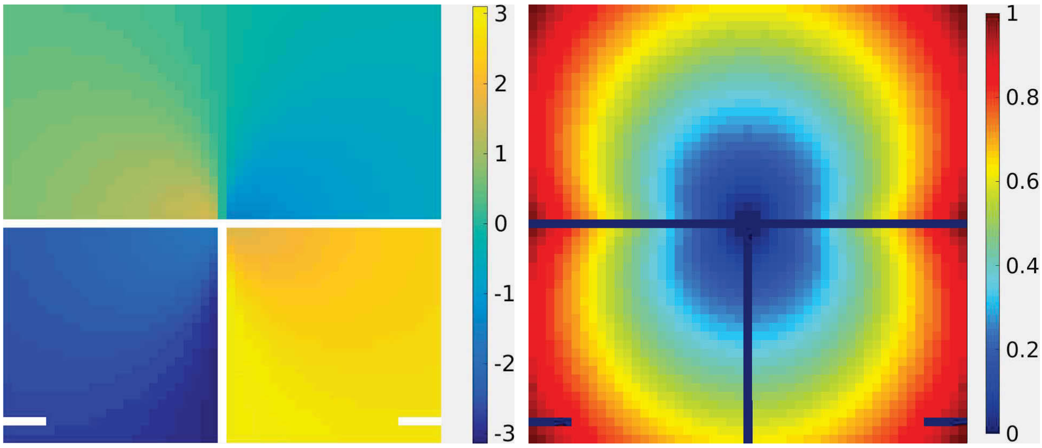
FIGURE 4.8: Topology for the local heatmap CNN (top to bottom).

The CNN is then created via TensorFlow (Abadi M et al. 2015), with the topology shown in Figure 4.8. The resultant network thus allows for the fast generation of local traversal heatmaps from local occupancy maps.

4.4 User Input Heatmaps from GP

User signals were omitted during the building of the CNN in order to avoid having to, for example, gather enough training data to adequately encompass all suitable joystick signals for the branching paths shown in Figure 4.7. Hence the overall objective in this Section is to derive similar traversal heatmaps as a continuous version of the arcing path primitives in Figure 3.4, using estimated desired joystick orientations. Section 4.4.1 overviews the generation of an offline movement template, which can be transformed with a user input model for fast user heatmap generation as follows in Section 4.4.2.

4.4.1 PMD Movement Template



(A) Joystick orientation template.

(B) Scaled distance filter.

FIGURE 4.9: A: Movement template containing joystick orientations (radians, along the colourbar) corresponding to projected PMD behaviour. White cells indicate a lack of coverage by the projected paths, or positions where paths were capped. B: Scaled distance filter from mean path lengths to each cell.

The movement template is a matrix of the same size and resolution as the local planning window. A set of arcing paths from the PMD's origin are first generated in a dense manner to facilitate near-total coverage of the template's area. These paths are terminated at both the edge of the map and where platform rotation exceeds ±180°. The linear and angular velocity command for each path is then mapped to a 2D joystick input (i, j) and a joystick orientation $h = \mathtt{arctan2}(i, j)$. Here we define S as the set of paths that passed through a given template cell, truncated at the cell.

The movement template (Fig. 4.9a) finally stores a value h_t in each cell, defined as the mean h associated with the paths in S.

To increase the effectiveness of active path planning, additional measures are taken to discourage the inference of goals which are too close to the platform's origin so that the planning space is more fully utilized. In the case of the sensor data CNN this was done by setting the path end-points in the training target images to a higher value; here a distance filter is utilized to further adjust the raw heatmap cell values. A second matrix stores the mean path length of S. These lengths are scaled to $(0, 1)$ across the template, resulting in the distance filter shown in Figure 4.9b.

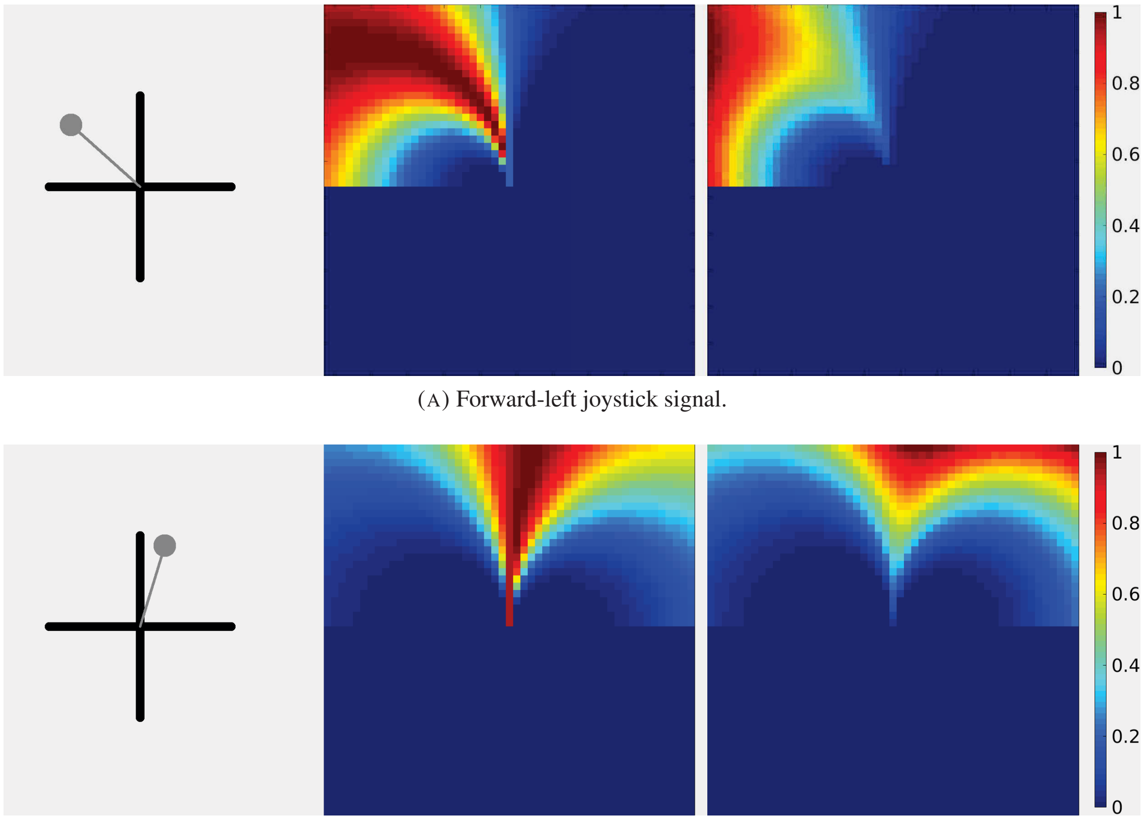
4.4.2 Generating Heatmaps

For modelling a raw user input to a desired input, the collision-free "safe" joystick signals obtained from Section 4.2 are taken as training targets to their corresponding raw signals. Here these safe signals are taken as the user's desired inputs, as the avoidance of obstacles is a desirable trait in good demonstration. This mapping is performed using a Gaussian Process (App. C.3) for each joystick axis. GPs are used here due to their consideration of uncertainty which is taken into account when transforming the movement template. Denoting i and j as horizontal and vertical joystick positions, the GPs thus yield the respective distributions $(\mu_i, \sigma_i^2), (\mu_j, \sigma_j^2)$ of estimated desired joystick axis values. To reduce the quantity of training data, the training inputs to the GP are discretized into a 10×10 grid with their respective training outputs possessing both a mean and variance. The square exponential kernel (Eq. C.5) is used again in this model, with the hyper-parameters of both GPs also optimized via Differential Evolution. We note that with this approach, the GPs can be tailored to individual end-users with driving data gathered in a similar fashion to Section 4.2.

In order to utilize the inferred joystick signals for creating heatmaps in physical space, the mean and variance of the user's intended joystick orientation h are calculated via propagation of uncertainty (Matte 2013; Ku 1966):

$$\mu_h = \arctan2(\mu_i, \mu_j) \tag{4.1}$$

$$\sigma_h^2 = \sigma_i^2 \left(\frac{\mu_j}{\mu_i^2 + \mu_j^2} \right)^2 + \sigma_j^2 \left(\frac{-\mu_i}{\mu_i^2 + \mu_j^2} \right)^2 \tag{4.2}$$



(A) Forward-left joystick signal.

(B) Forward-right joystick signal.

FIGURE 4.10: Left column: Input joystick signals. Centre column: raw heatmap from weighing the movement template in Figure 4.9a. Right column: element-wise product of the raw heatmap with the distance filter in Figure 4.9b, to be taken as the final user input heatmap.

A raw heatmap can then be rapidly computed by assigning a weight w to each cell of the movement template. In this work we take w as the probability density at the cell's h_t:

$$w = \frac{1}{\sqrt{2\pi\sigma_h^2}} \times \exp\left(\frac{-(\mu_h - h_t)^2}{2\sigma_h^2}\right) \tag{4.3}$$

This heatmap is then multiplied in an element-wise fashion with the distance filter for a final user input heatmap after scaling to $(0,1)$; examples of user GP heatmap generation can be seen in Figure 4.10.

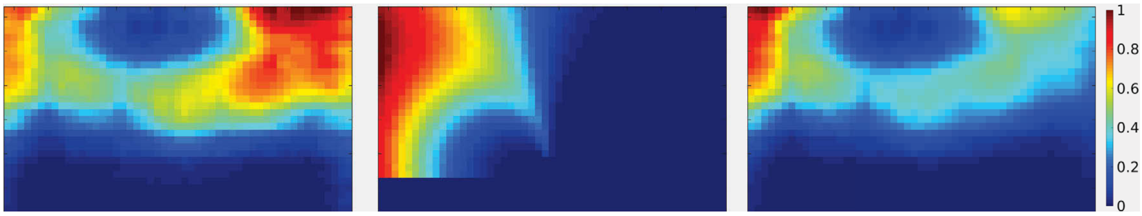


FIGURE 4.11: Combining I_{sensor} (left) from Fig. 4.6 and I_{user} (middle) from Fig. 4.10a yields the final intention heatmap I_{intent} (right).

4.5 Path Planning via Q-learning

From a sensor data heatmap I_{sensor} and a user input heatmap I_{user}, an overall heatmap I_{intent} can then be obtained as their element-wise product (Fig. 4.11).

For an introduction to Q-learning, we direct the reader to Appendix D.1. In this problem $\mathcal{S}$ is defined as a local grid-world around the PMD sharing size and resolution with I_{intent}, and $\mathcal{A}$ as a set of 9 moves from a grid cell to one of its immediate neighbors or to itself. I_{intent} is taken as $\mathcal{R}$, with an added border of negative reward cells. As an additional safety measure, cells near perceived obstacles are penalized by the superimposition of an inflated negative reward obstacle map. After optimization, coarse path planning then commences by means of gradient ascent across $\arg\max_{a \in \mathcal{A}} \mathcal{Q}(\mathcal{S}_{k+1}, a)$ starting from the cell corresponding to the PMD's origin. This is summarized in Algorithm 3, with an example visualized in Figure 4.12.

In our experiments, $\alpha = 0.7$, $\gamma = 0.9$, and iteration cap $K = 100$. The low reward for obstacle cells and the border was set at -100. K was set to 100 in order to impose an upper bound on computation time, while α and γ were empirically determined to avoid divergence from the goal during learning while keeping a desirable rate of convergence for real-time application. For an overview of a more comprehensive setting of these hyper-parameters, we direct the reader to Claesen and Moor (2015).

The resultant coarse path is then smoothed via interpolation, and tracked by the PMD. As per the previous Chapter, it remains necessary to be able to determine a signficant deviation from the user's intent for the purpose of punctual replanning. Rather than building a separate compliance model similar to that of Section 3.4.3, the value of the cell within I_{user} corresponding to the path tracking algorithm's control point can instead readily serve as a quantitative estimate of user

```
input : $\mathcal{S}$,$\mathcal{A}$,$\mathcal{R}$, $\alpha$,$\gamma$, $K$
/* Q-learning loop                                                    */
$Q \leftarrow -100_{\mathcal{S}\times\mathcal{A}}$
for $k = 1, 2, ..., K-1$ do
    $\mathcal{Q} = \mathcal{Q} + \alpha\Big(\mathcal{R}_{k+1} + \gamma \times \arg\max_{a\in\mathcal{A}} \mathcal{Q}(\mathcal{S}_{k+1}, a) - \mathcal{Q})\Big)$
end
/* gradient ascent path planning                                      */
$i = 0$; $s_i \leftarrow$ position of occupancy grid origin
while $s_i$ is within occupancy grid do
    $a_i = \arg\max_{a\in\mathcal{A}} Q(s_{k+1}, a)$
    $s_{i+1} \leftarrow$ state from executing $a_i$ at $s_i$
    $i = i + 1$
end
output: coarse path $\{s_1, s_2, ..., s_{i-1}\}$
```

Algorithm 3: Path planning via Q-learning.

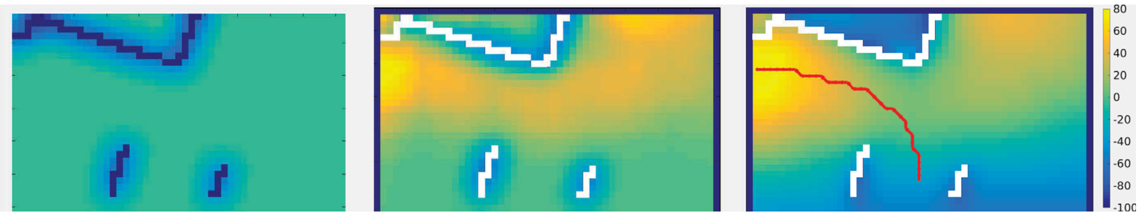


FIGURE 4.12: I_{intent} from Fig. 4.11 is first penalized with an inflated obstacle map (left), resulting in $\mathcal{R}$ (middle) after the addition of a negative reward border. The right image shows the post-optimization landscape, with a coarse path (red) planned to the highest reward area. In the latter two images, obstacles are represented in white for distinctiveness.

compliance. When this compliance value falls below a certain threshold (0.9 again in the following experiments), a new path is requested.

Metric	Unassisted	Assisted
Avg. angular jerk (rad/s^3)	0.04 ± 0.01	0.02 ± 0.01
Time taken (s)	162 ± 22	150 ± 20
Distance travelled (m)	97 ± 2	97 ± 1
Total collisions	4	0
Steering entropy	0.58 ± 0.03	0.59 ± 0.02

TABLE 4.1: Driving metrics from on-line simulation experiment.

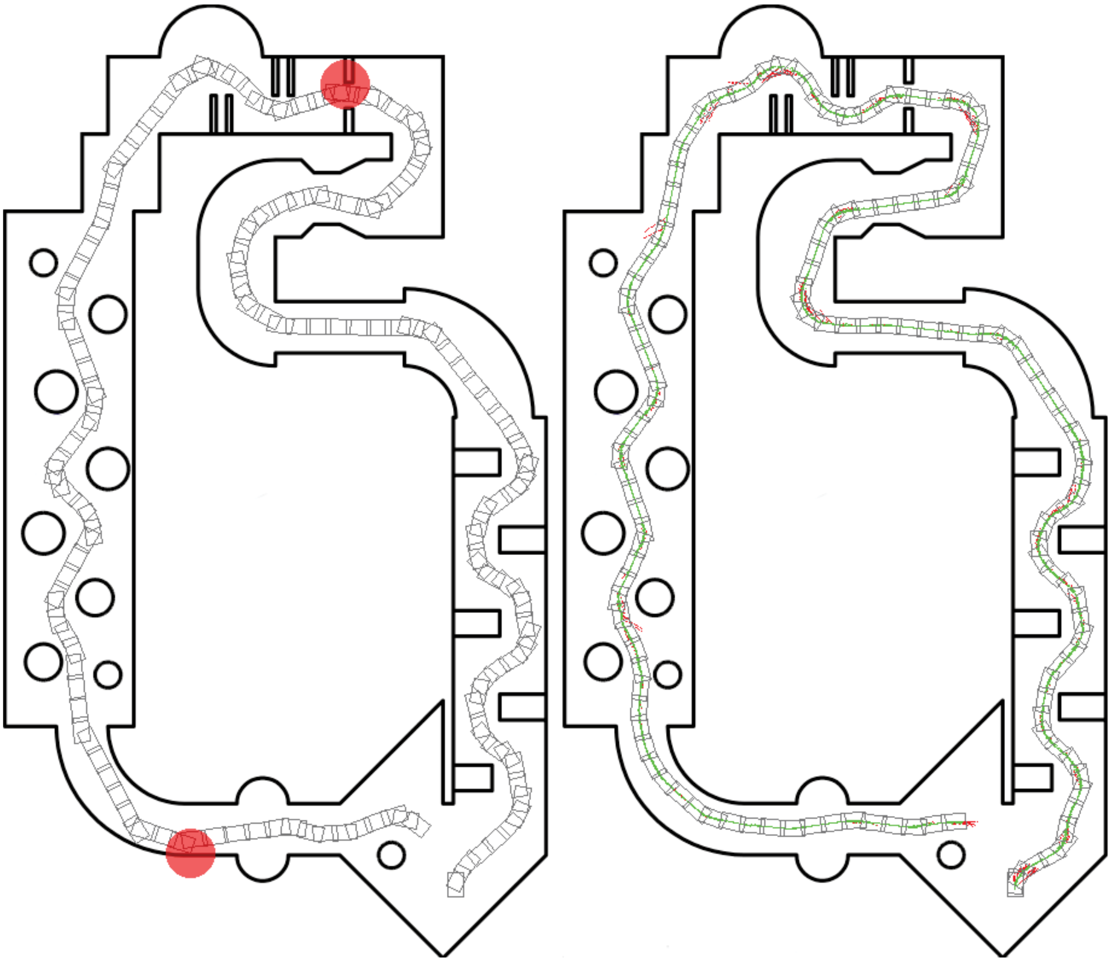
4.6 Experimentation

In this section we first assess the assistive capability of the framework in an on-line navigation exercise, then perform a *a-posteriori* evaluation of predictive accuracy against the methodology detailed in Chapter 3 and a naive CNN incorporating user signals into its training input data. Experimentation was conducted on the same machine used in Section 2.3.2.1.

4.6.1 On-line Assistance

5 able users were tasked with navigating around the standard simulation course previously seen in Section 3.5.1, with their input signals subject to the same imposed disability. Each test user drove around the course twice; once without assistance, and once with the assistive framework in operation. (Fig. 4.13).

Assessment metrics from Urdiales et al. (2013a) and Nakayama et al. (1999) are shown in Table 4.1. In this experiment, the angular jerk experienced by the user halved with the assistive framework was in operation. Several collisions also occurred without assistance due to the difficulties imposed by the simulated disability symptoms, whereas the assisted trials were collision-free. The distances travelled in both experiments are similar between unassisted and assisted, however the course completion times taken were noticeably shorter with assistance. Finally the steering entropy was similar across both trials, indicating a consistent task effectiveness on the users' part.



(A) Unassisted path with collision point(s) highlighted in red.

(B) Assisted path, more details in Section 3.5.

FIGURE 4.13: Visual comparison between unassisted and assisted paths (gray) of an able volunteer with a simulated input disability, driving counter-clockwise in a $21 \times 37m$ environment starting from the bottom-right corner.

4.6.2 *A-posteriori* Comparisons

For an off-line comparison against the heuristic mechanism from the previous Chapter and a naive CNN, 30 unassisted runs were recorded from 10 able users (Fig. 4.14) driving safely to arbitrary destinations in a $20 \times 19m$ simulated environment novel to those used in recording training data. We then assess the *a-posteriori* post-optimization heatmaps from which PMD paths are derived, by concatenating individual heatmaps over the course of each run (Alg. 4).

```
input : odometry $\mathbf{o}_{1,2,...,N}$, occupancy maps $\mathbf{M}_{1,2,...,N}$, heatmaps $\mathbf{H}_{1,2,...,N}$
$H_{concat} \leftarrow 0_{map\,size}$; $C \leftarrow 0_{map\,size}$
for $n = 1, 2, ..., N$ do
    $v \leftarrow$ positions of visible cells within $\mathbf{M}_n$
    $\mathbf{H}_n[v^c] = 0$; $\mathbf{H}_n \leftarrow \mathbf{H}_n$ squashed $\in (0, 1)$
    $v' \leftarrow v$ transformed by $\mathbf{o}_n$
    $H_{concat}[v'] = H_{concat}[v'] + \mathbf{H}_n[v]$
    $C[v'] = C[v'] + 1$
end
$w \leftarrow$ positions of $C > 0$
$H_{concat}[w] = H_{concat}[w]/C[w]$
output: concatenated heatmap $H_{concat}$
```

Algorithm 4: Local heatmap concatenation.

In building the naive CNN, the user input is incorporated into the CNN input image (Fig. 4.15). Both joystick axes are first squashed between (0,1), then each scaled axis value fills a lower quarter of the input image which is largely empty due to the lack of sensor coverage. Its respective training target image also only features the PMD's immediate future path for that given instance, as opposed to the multiple paths shown in Figure 4.7. The topology from Figure 4.8 was re-used.

Examples of concatenated heatmaps are shown in Figure 4.16, and several metrics for comparison are presented in Table 4.2. The naive CNN produced heatmaps which were broadly distributed across open spaces and tended to excessively accumulate reward in restrictive areas, indicating heatmaps poorly suited to path planning. The approach in Chapter 3 performs better with less extreme changes in reward allocation, and finally the CNN/GP approach produces the most consistent reward allocation throughout the movement. This is reflected in the reward allocated to the PMD's actual route taken through the concatenated heatmaps, with the naive CNN's reward at the PMD positions being noticeably lower than the other two methods. As a result, Chapter 3 and the CNN/GP method each achieved a good path inference accuracy with an RMSE (root mean square

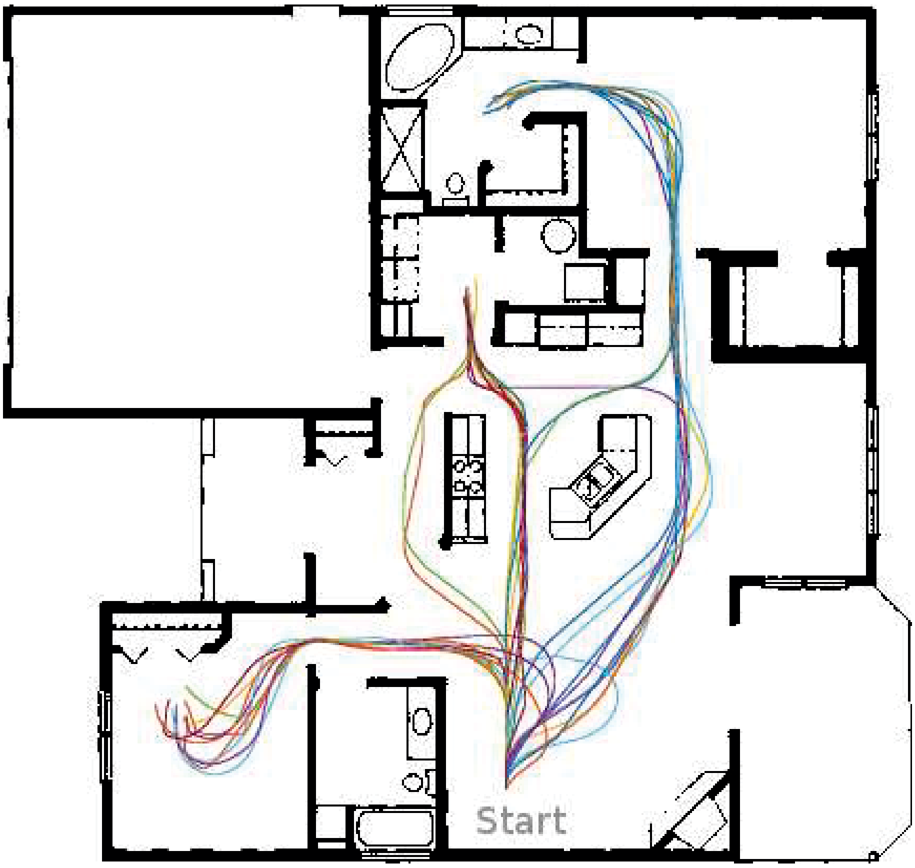
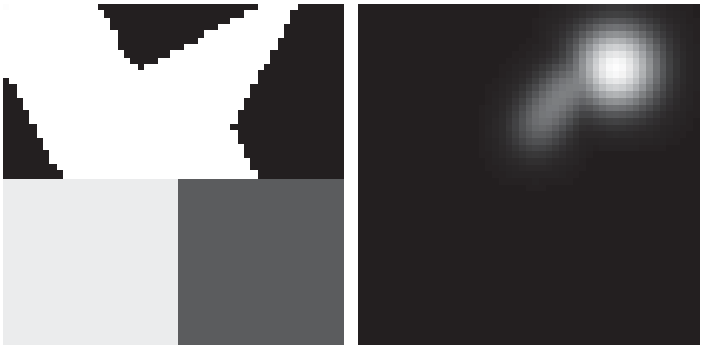


FIGURE 4.14: 30 runs from 10 able users in a typical domestic environment approx. $20 \times 20m$.



(A) CNN training input.

(B) CNN training target.

FIGURE 4.15: Example of naive CNN training data. A: the top half contains a map of local free-space (white) from a LIDAR sweep. The bottom two quarters are filled with vertical and horizontal joystick axis values squashed to (0,1). B: pathing map only considering the PMD's immediate future route from this particular training data instance.

Metric	Approach in Chapter 3	Naive CNN	CNN/GP
Avg. path computation time (s)	0.5	0.2	0.2
Avg. reward at PMD positions	0.62 ± 0.10	0.56 ± 0.17	0.67 ± 0.051
Planned/actual path RMSE (m)	0.05 ± 0.12	0.14 ± 0.18	0.047 ± 0.09

TABLE 4.2: *A-posteriori* framework evaluation metrics, comparing the CNN/GP approach against Chapter 3 and a naive CNN. Path computation time represents the entire processing duration, from data reception to the end of the coarse path generation step.

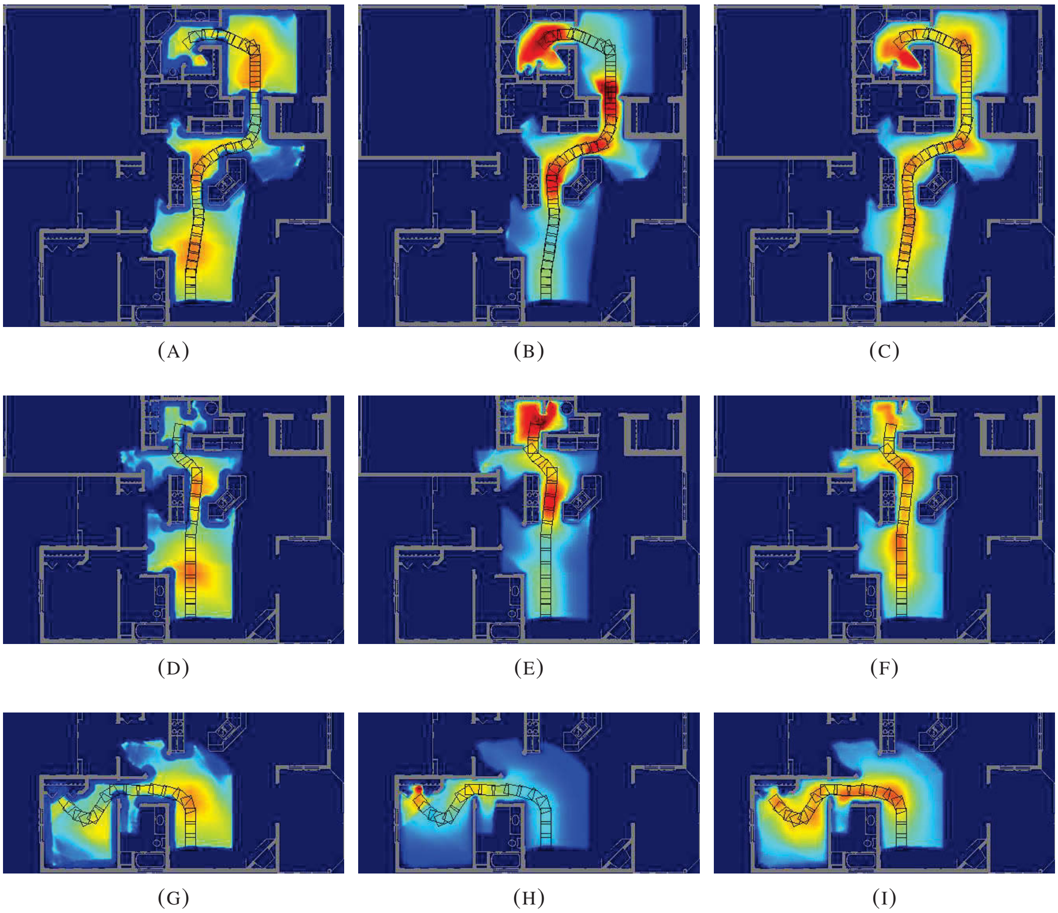

(A) (B) (C) (D) (E) (F) (G) (H) (I)

FIGURE 4.16: Concatenated *a-posteriori* heatmaps for three runs (rows), with PMD footprints drawn in black. Left column: heuristic approach in Chapter 3. Center column: naive CNN. Right column: CNN/GP introduced in this Chapter.

error) of approximately $0.05m$, whereas the naive CNN performed much worse at $0.14m$. By also sparsifying the GP data, the CNN/GP featured a much faster computation time than the previous Chapter at less than a quarter second; almost indistinguishable to that of the naive CNN.

From the three approaches compared here, overall the CNN/GP performs the best in terms of predictive accuracy and computation time.

4.7 Chapter Summary

In this Chapter a simplified framework in comparison to the one detailed in Chapter 3 is presented. For the purpose of reducing learning complexity, sensor data and user input are decoupled into individual sub-spaces. Their individual models, built with a Convolutional Neural Network and a Gaussian Process based approach respectively in Sections 4.3-4.4, both yield continuous short-term heatmaps of traversal likelihood which naturally combine intention estimation and expert pathing. These heatmaps then readily facilitate path planning (Sec. 4.5) in a similar fashion as before, with Q-learning replacing DPP as an optimizer since path primitives are no longer required. Such decoupling allows avoidance of the very large learning space which required significant expert knowledge to address in the previous Chapter, and relies on fewer heuristic measures. Comparison with the framework presented in Chapter 3 reveals similar performance in terms of *a-posteriori* path prediction accuracy at a $0.05m$ RMSE, but with a noticeably lower computation time and a simplified overall framework design. The relatively poor performance of a naive CNN trained upon combined sensor/user data also shows the benefits of decoupling the learning space.

From these two Chapters, the greater consideration of environmental observation from on-board perception has facilitated active PMD assistance in navigation within novel environments, which has been infeasible for previous LfD methodologies focused upon user behaviours and static environments. In the next Chapter, we further explore this concept of heightened scene observation within the context of interaction with upper-body static humanoid robots, with the goal of improving robot action generation in collaborative human-robot exercises in cases where partner behaviour can easily be insufficiently informative in communicating relevant properties of the interactive task to be undertaken without additional perceptual information. Despite this paradigm shift, the concept of greater consideration of perceptual information remains the same; albeit with a different sensing modality (vision) and the output of end-effector trajectories required for the shared activity instead of mobile robot paths.

Chapter 5

Environmental Adaptation for Human-Humanoid Collaboration

In this Chapter, we explore a greater degree of external influence within the learning of physically collaborative behaviours between humans and static humanoid robotic agents. The main differences of this work to that of the previous two Chapters are that on-board sensor data is now replaced with vision of the scene around both agents for a more informative sensing modality required when undertaking more complex human-robot tasks. Moreover, end-effector trajectories are planned for the robot to participate in these activities as opposed to vehicular paths. However the overall concept remains largely unchanged, despite the shift in setting.

It bears noting that the work in this Chapter is not necessarily targeted towards disabled individuals, whereas the focus upon PMDs in the previous three Chapters may make such an assumption implicit. Here a lack of sufficient task goodness arises more from ambiguity within a temporal window of human partner observation rather than any physical or cognitive impairments on their part, although the methodologies that follow are nevertheless tunable for a given individual's specific requirements by gathering training samples from them specifically. Such tailoring is conceptually comparable to the user signal remapping outlined in Section 4.1.

The Interaction Primitives framework (IPs) (Ben Amor et al. 2014) was designed for the generation of end-effector trajectories to allow the robot to undertake a broader range of possible collaborative activities than in the case of the PMD, given observations of movements of the robot's human partner. Furthermore, the framework itself is also amenable to expansion in order to allow for the consideration of additional information sources. IPs thus readily serve as an experimental basis in this Chapter for exploring a greater degree of scene perception within a human-humanoid interactive context; an extension (Fig. 5.1) to the IP framework is presented here, allowing for a comparison against a learning methodology in which only human observations are taken into account for the generation of robot actions.

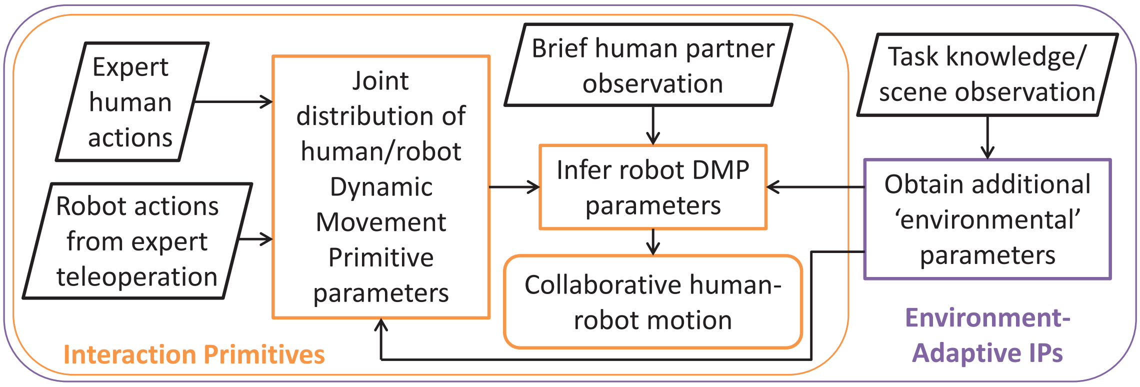

Expert human actions
Robot actions from expert teleoperation
Joint distribution of human/robot Dynamic Movement Primitive parameters
Brief human partner observation
Infer robot DMP parameters
Collaborative human-robot motion
Task knowledge/ scene observation
Obtain additional 'environmental' parameters
Interaction Primitives
Environment-Adaptive IPs

FIGURE 5.1: Schematic overview of the complete EaIP framework: an extension upon IPs to consider additional environmental parameters from task/environmental consideration.

Interaction Primitives learn a joint distribution between partner observations and Dynamic Movement Primitive (DMP) parameters for robot actions. Here an additional task observation vector is folded into the distribution, with the objective of enhancing robustness in parameter inference during collaborative human-robot exercises. These task parameters can either come from expert task knowledge, or from scene observation by robot perception; hence we refer to these augmented IPs as Environment-adaptive Interaction Primitives (EaIPs). The contents of this Chapter encompass the following publications: Cui et al. (2016a, 2018).

Following an overview of background literature in Section 5.1, DMPs, IPs, and EaIPs are detailed in Sections 5.2, 5.3, and 5.4 respectively. For our evaluation problem of a human and a robot covering furniture together, a toy simulation is first explored in Section 5.5. This is followed by further evaluation and comparison to conventional IPs on the NAIST Baxter system in Section 5.6, with the additional parameters coming from expert knowledge and then from a deep learning engine applied to images of the work scene.

5.1 Background

Due to the strong physical correlation between humanoid robots and our own bodies, LfD provides an intuitive means of programming behaviours into such systems which are acceptably natural to humans. It is common in the learning of more complex robot tasks to base actions from inferences drawn from training data, rather than to tune operating parameters manually. Depending on the scenario at hand this data is usually obtained by teleoperation of the robot such as by kinesthetic teaching (Fig. 5.2) e.g. Kormushev et al. (2010); Rozo et al. (2013); Kronander and Billard (2014) or if possible, direct control of the robot which will be executing autonomous behaviors e.g. Goil et al. (2013); Soh and Demiris (2015) although the latter is often restricted to lower DOF systems such as mobile ground vehicles.

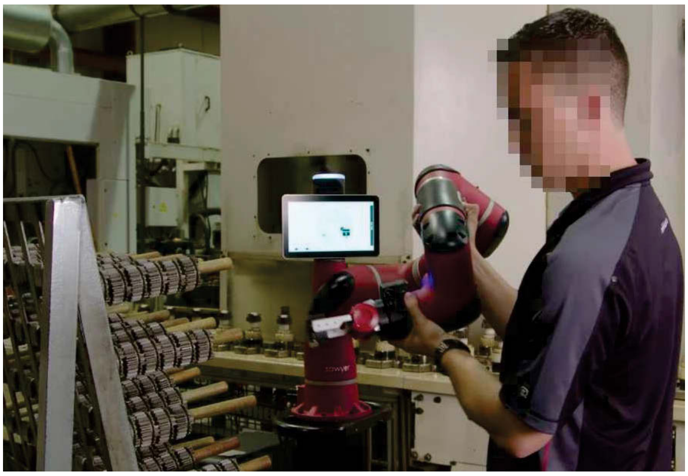


FIGURE 5.2: Kinesthetic programming of a Sawyer robot. From Rethink Robotics (2016).

By gathering data consisting of desirable robot behavior and observations of a partner in a human-robot interactive exercise, an aim of learning from this data is the synthesis of suitable robot actions after anticipating outcomes of human actions to achieve fluent interaction, as is the case with both IPs and EaIPs. The ability of a robot control framework to correctly act when faced with variation in both overall task conditions and behavior of the robot's human partner is vital in maintaining both safety and task compliance. Examples of adaptation in accordance to partner anticipation include the work by Huang and Mutlu (2016), using Support Vector Machines to predict eye gaze and treat it as the human partner's focus of attention, which formed the basis for subsequent robot activity. A reinforcement learning-based (Sutton and Barto 1998) approach to human intention

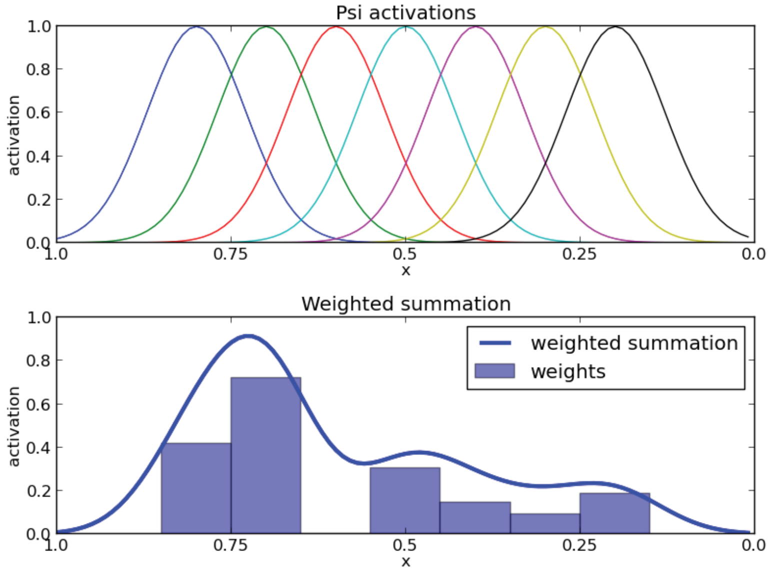


FIGURE 5.3: Dynamic Movement Primitives, encoding a second-order 1D trajectory as a weighted sum of several Gaussian basis functions. From Dewolf (2013).

estimation is investigated by Awais and Henrich (2013), where it is followed by a particle filter for probabilistic action selection.

The issue of generalization also necessitates a mechanism allowing for corresponding variation in robot actions, while maintaining some essence of the movements recorded during the demonstration phase. The Dynamic Movement Primitive has become widely used to this end, with examples of recent work including Ude et al. (2010); Matsubara et al. (2011); Tan et al. (2012); Liu et al. (2014); Prada et al. (2014); Fitzgerald et al. (2015); Mandery et al. (2016). Usually modelled from demonstration data, a DMP encodes the structure of a 1 DOF trajectory (Fig. 5.3). Their ability to be subject to spatial and temporal scaling makes them well suited to action generalization, broadly defined as the mapping of task observations to suitable parameters for DMPs corresponding to each DOF of a robot. Libraries of DMPs have also been built to allow for consideration of a range of actions with greatly varying underlying structure, in order to undertake activities such as handwritten character generation (Meier et al. 2011) and table tennis swings (Mülling et al. 2013). Interaction Primitives (Fig. 5.4) extend the DMP framework; given demonstration data of a task, a joint Gaussian distribution is built consisting of human actions and DMP parameters. From a brief human observation period, suitable DMP parameters corresponding to appropriate robot behaviors can then be drawn.

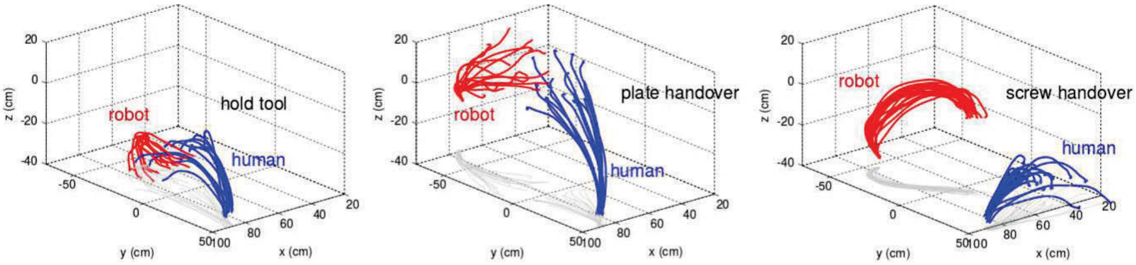


FIGURE 5.4: Robot action generation via IPs solely based upon observations of human movement, without consideration of the context of the task driving his/her behavior. From Maeda et al. (2017).

In this work IPs are augmented with environmental observations of the task being undertaken by the human-robot team, resulting in EaIPs whose inferences are not solely postulated upon a brief snapshot of partner behaviour. In our scenario, environmental parameters take the form of features describing the object that serves as the focus of the joint activity. Utilizing these parameters allows us to increase robustness under changing task conditions, as opposed to the work by Maeda et al. (2017) which relied upon the reaching behavior of the human partner in determining a suitable reciprocative action rather than considering the nature of the actual object the partner was reaching for. An additional benefit from the enhanced robustness of EaIP is that the partner observation duration can be heavily reduced, allowing for a faster robot response time.

The collaborative manipulation of objects remains a challenging objective, as the nature of the object and any of its interactive counterparts being manipulated must be considered in conjunction with the behavior of the human partner. Sheng et al. (2015) uses reinforcement learning and observations of human beha vior to keep a table held between the human and robot in a horizontal alignment, while Lawitzky et al. (2010) devised a framework that aimed to share the load between the two agents. As an alternative for robots with multimodal sensory capabilities, Kruse et al. (2015) considered both the forces applied to a sheet of fabric held taut between the human and the robot as well as perceived deformations. The manipulation of deformable materials is still a challenging area of investigation due to a prohibitively large state space, although work is being done towards the modelling of smaller deformable items such as pieces of clothing e.g. Doumanoglou et al. (2014).

The direct application of deep learning from visual cues to the problem of robot control is quite challenging primarily due to the difficulties in accumulating requisite amounts of training data and forms a relatively recent initiative, e.g. Finn et al. (2016); Pinto and Gupta (2015). Rather than directly generating robot actions, here a Convolutional Neural Network serves as a mechanism to reduce uncertainty by examining the task scene for additional information, which in turn augments observations of the human partner for the EaIP. A brief introduction to CNNs can be found in Appendix C.2.1, however here the CNN is not fully convolutional (has no decoder component) as it instead serves as a deep classification engine.

5.2 Dynamic Movement Primitives (Ijspeert et al. 2013)

Within the area of trajectory control and planning, Dynamic Movement Primitives are widely used to stably represent complex motor actions from demonstration without significant parameter tuning, that can be flexibly adjusted upon runtime. In this work, DMPs encode 1D trajectories of both human and robot movements. In application to real systems a DMP is built for each degree of freedom (DOF). For each DOF, a time-series trajectory is defined as the following:

$$\ddot{y}(t) = \left(\alpha_y\left(\beta_y\left(g - y(t)\right) - \left(\frac{\dot{y}(t)}{\tau}\right)\right) + f(x_t)\right)\tau^2 \tag{5.1}$$

α_y and β_y are conventional PD (proportional-differential) control coefficients. y and g are respectively the state variable and goal value which serves as a spatial scaling factor. τ is a temporal scaling factor, and t is the time step. $f(x_t)$ is the forcing function built by M Gaussian basis functions $\boldsymbol{\psi}$ and a corresponding $M \times 1$ weights vector $\boldsymbol{w}$:

$$f(x_t) = \frac{\sum_{i=1}^{M} \boldsymbol{\psi}_i(x_t) w_i x_t}{\sum_{j=1}^{M} \boldsymbol{\psi}_j(x_t)} = \phi(x_t)^T \boldsymbol{w}. \tag{5.2}$$

x serves as a canonical system defined as $\dot{x} = -\alpha_x x \tau$, with the initial $x_0 = 1$.

To encode a T step trajectory $\boldsymbol{y} = [y(t), \dot{y}(t), \ddot{y}(t)]_{t=1:T}^T$ using $\boldsymbol{w}$, the forcing function value at each t required to reproduce the input trajectory at t is first calculated:

$$f(x_t) = \frac{1}{\tau^2}\ddot{y}(t) - \alpha_y\left(\beta_y\left(g - y(t)\right) - \frac{\dot{y}(t)}{\tau}\right). \tag{5.3}$$

Expressing the DMP as $\boldsymbol{f} = \boldsymbol{\phi}\boldsymbol{w}$, with basis functions $\boldsymbol{\phi} = [\phi(x_1), ..., \phi(x_T)]^T$ and
$\boldsymbol{f} = [f(x_1), ..., f(x_T)]^T$, the weights vector $\boldsymbol{w}$ can then be obtained via least squares regression:

$$\boldsymbol{w} = (\boldsymbol{\phi}^T \boldsymbol{\phi})^{-1} \boldsymbol{\phi}^T \boldsymbol{f}. \tag{5.4}$$

The key benefit of using DMPs in robot control applications besides their ease of spatial/temporal scaling via g and τ is in their robustness; arbitrarily complex $f(x_t)$ are subject to the decay of x over time, and certainty of goal convergence comes from the incorporated PD mechanism.

5.3 Interaction Primitives (Ben Amor et al. 2014)

Motivated by the desire of engaging in cooperative activities between human partners and robots using DMPs, Interaction Primitives were proposed to extend DMPs to facilitate human-robot interaction tasks. After modelling from demonstration data, IPs achieve human-robot co-operation through the following steps:

1. Observe a partial trajectory from the human partner, then infer DMP phase x^*
2. Obtain a distribution of DMP parameters for all robot DOF, and execute

5.3.1 Phase Estimation

Dynamic Time Warping (DTW) (App. E.4) is employed to estimate the phase of observed human movement. Given a partial human movement $[\boldsymbol{y}_1^*, ..., \boldsymbol{y}_{T'}^*]^T$, and a full reference movement from prior demonstration $[\boldsymbol{y}_1, ..., \boldsymbol{y}_T]^T$, DTW measures the similarity between the two temporal sequences and yields the index t^*. This identifies the frame in the reference movement which produces a minimal distance with respect to the observed query movement, i.e. $[\boldsymbol{y}_1^*, ..., \boldsymbol{y}_{T'}^*]^T$ is close to $[\boldsymbol{y}_1, ..., \boldsymbol{y}_{t^*}]^T$. The estimated phase of a partially observed human movement is therefore defined as:

$$x^* = \exp\left(-\alpha_x \left(\frac{t^*}{T} \right) \tau \right). \tag{5.5}$$

5.3.2 DMP Parameter Inference

As training samples, S sets of N DOF trajectories are time-scaled to the same duration T:

$$\boldsymbol{Y} = [\boldsymbol{Y}_{human}, \boldsymbol{Y}_{robot}] = \begin{bmatrix} \boldsymbol{y}_1^1 & \cdots & \boldsymbol{y}_N^1 \\ \vdots & \ddots & \vdots \\ \boldsymbol{y}_1^S & \cdots & \boldsymbol{y}_N^S \end{bmatrix} \tag{5.6}$$

where N is the combined robot control and human observation DOF. Here we define $\boldsymbol{y}_i^j$, $\boldsymbol{w}_i^j$ and g_i^j as the trajectory, weights vector and target position of the i-th DOF in the j-th demonstration respectively. Thus $\boldsymbol{\theta}^{[j]} = [\boldsymbol{w}_1^{j\,T}, g_1^j, ..., \boldsymbol{w}_N^{j\,T}, g_N^j]^T, j = 1, ..., S$ is the DMP parameter vector as modelled from $[\boldsymbol{y}_1^j, ..., \boldsymbol{y}_N^j]$. $p(\boldsymbol{\theta})$, the distribution among the parameter vector samples $\boldsymbol{\theta}^{[j]}, j = 1, ..., S$, then follows:

$$p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_{\boldsymbol{\theta}}, \Sigma_{\boldsymbol{\theta}}), \tag{5.7}$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}} = \frac{\sum_{j=1}^{S} \boldsymbol{\theta}^{[j]}}{S}, \tag{5.8}$$

$$\Sigma_{\boldsymbol{\theta}} = \frac{\sum_{j=1}^{S} (\boldsymbol{\theta}^{[j]} - \boldsymbol{\mu}_{\boldsymbol{\theta}})(\boldsymbol{\theta}^{[j]} - \boldsymbol{\mu}_{\boldsymbol{\theta}})^T}{S} \tag{5.9}$$

where $\boldsymbol{\theta} = [\boldsymbol{\theta}_{human}, \boldsymbol{\theta}_{robot}]^T$ contains the parameter vectors of both human and robot.

After obtaining phase x^*, $\boldsymbol{Y}^*_{human} = [\boldsymbol{y}_1^*, ..., \boldsymbol{y}_n^*]^T$ is resampled from the observed movement where $n < N$ is the dimensionality of human observation. The unavailable trajectories of the robot $\boldsymbol{Y}^*_{robot}$ are set to $\mathbf{0}$. Defining $\boldsymbol{Y}^* = [\boldsymbol{Y}^*_{human}, \boldsymbol{Y}^*_{robot}]$, the prediction of both human and robot's parameter vector is represented by:

$$p(\boldsymbol{\theta}|\boldsymbol{Y}^*) \propto p(\boldsymbol{Y}^*|\boldsymbol{\theta})p(\boldsymbol{\theta}). \tag{5.10}$$

The likelihood $p(\boldsymbol{Y}^*|\boldsymbol{\theta})$ is modelled by a Gaussian distribution over the forcing function:

$$p(\boldsymbol{Y}^*|\boldsymbol{\theta}) \sim \mathcal{N}(\boldsymbol{F}^*|\boldsymbol{\Omega\theta}, \sigma^2\boldsymbol{I}) \tag{5.11}$$

where $\boldsymbol{F}^*$ has two parts: $\boldsymbol{F}^*_{human} = [\boldsymbol{f}^*_1, ..., \boldsymbol{f}^*_n]^T$ is the observed forcing function of $\boldsymbol{Y}^*_{human}$. Its elements are given by:

$$f_i^*(x_t) = \frac{1}{\tau^2}\ddot{y}_i^*(t) - \alpha_y\Big(-\beta_y y_i^*(t) - \frac{\dot{y}_i^*(t)}{\tau}\Big). \tag{5.12}$$

$\boldsymbol{F}^*_{robot}$ is the unavailable forcing function of robot and set as $\boldsymbol{0}$. The matrix $\boldsymbol{\Omega\theta}$ contains the forcing function in terms of $\tilde{\boldsymbol{\phi}}_t = [\phi(x_t)^T, \alpha_y\beta_y]$ over learning samples for $1 \leq t \leq t^*$:

$$\boldsymbol{\Omega\theta} = \begin{bmatrix} \tilde{\boldsymbol{\phi}} & 0 & ... & ... \\ 0 & \tilde{\boldsymbol{\phi}} & 0 & ... \\ \vdots & \vdots & \vdots & \vdots \\ 0 & ... & ... & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_1 \\ g_1 \\ \vdots \\ \boldsymbol{w}_N \\ g_N \end{bmatrix} \tag{5.13}$$

with the robot's $\tilde{\boldsymbol{\phi}}$ in $\boldsymbol{\Omega}$ set to 0. σ^2 is the observation noise.

Given likelihood $p(\boldsymbol{Y}^*|\boldsymbol{\theta})$, $p(\boldsymbol{Y}^*, \boldsymbol{\theta}) = p(\boldsymbol{Y}^*|\boldsymbol{\theta})p(\boldsymbol{\theta})$ is another joint Gaussian distribution:

$$p(\boldsymbol{Y}^*, \boldsymbol{\theta}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{F}^* \\ \boldsymbol{\theta} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\Omega\theta} \\ \mu_\theta \end{bmatrix}, \begin{bmatrix} \boldsymbol{A} & \boldsymbol{\Omega}\Sigma_\theta \\ \Sigma_\theta\boldsymbol{\Omega}^T & \Sigma_\theta \end{bmatrix}\right) \tag{5.14}$$

with $\boldsymbol{A} = \sigma^2\boldsymbol{I} + \boldsymbol{\Omega}\Sigma_\theta\boldsymbol{\Omega}^{\mathrm{T}}$. The mean and variance of the conditional distribution $p(\boldsymbol{\theta}|\boldsymbol{Y}^*)$ are then derived as follows:

$$\begin{aligned} \mu_{\theta|\boldsymbol{y}^*} &= \mu_\theta + \Sigma_\theta\boldsymbol{\Omega}^T\boldsymbol{A}^{-1}(\boldsymbol{F}^* - \boldsymbol{\Omega}\mu_\theta), \\ \Sigma_{\theta|\boldsymbol{y}^*} &= \Sigma_\theta - \Sigma_\theta\boldsymbol{\Omega}^T\boldsymbol{A}^{-1}\boldsymbol{\Omega}\Sigma_\theta. \end{aligned} \tag{5.15}$$

After obtaining $\boldsymbol{\theta}$, the robot is actuated by running DMPs with parameter vector $\boldsymbol{\theta}_{robot}$ from phase x^* onwards.

5.4 Environment-adaptive Interaction Primitives

An issue with IPs is that in situations where the initial human partner observation $[\boldsymbol{y}_1^*, ..., \boldsymbol{y}_{T'}^*]^T$ can often be too ambiguous for reliable DMP parameter inferences. The system can either wait for more distinctive partner activity which may result in an unnaturally long pause before action on the robot's part can commence, or risk generating DMPs prematurely which may result in unsafe or unpredictable behaviors.

To reduce uncertainty, EaIPs introduce environment parameters $\boldsymbol{e}$ representing features embodying task-critical environmental properties into IPs along with human observation trajectories $p(\boldsymbol{y}^*)$, i.e. computing a joint distribution $p(\boldsymbol{Y}^*, \boldsymbol{\theta}, \boldsymbol{e})$. Depending on context, the content of $\boldsymbol{e}$ can vary widely from physical sizes of objects to perceived obstructions; these parameters can also be provided from models built on the same training data for the underlying IP or from heuristics, however in this thesis we propose object bounding boxes to encourage generalizational ability.

For recording trajectories with these environment parameters, a new training sample is defined as $\boldsymbol{Y}_e$ with an additional DOF for $\boldsymbol{E} = [\boldsymbol{e}^1, ..., \boldsymbol{e}^S]^T$:

$$\boldsymbol{Y}_e = [\boldsymbol{Y}_{human}, \boldsymbol{Y}_{robot}, \boldsymbol{E}] = \begin{bmatrix} \boldsymbol{y}_1^1 & \dots & \boldsymbol{y}_N^1 & \boldsymbol{e}^1 \\ \vdots & \ddots & \vdots & \vdots \\ \boldsymbol{y}_1^S & \dots & \boldsymbol{y}_N^S & \boldsymbol{e}^S \end{bmatrix}. \tag{5.16}$$

The environmental parameters' weight vector $\boldsymbol{w}_e$ is then obtained as a least-squares solution from demonstration data: $\boldsymbol{w}_e = \boldsymbol{e}\boldsymbol{\theta}^T(\boldsymbol{\theta}\boldsymbol{\theta}^T)^{-1}$.

Given t^* steps observing trajectories $\boldsymbol{Y}^*$ and environmental parameter $\boldsymbol{e}^*$, we combine the observing forcing function and environmental parameter to $\boldsymbol{F}_e^* = [\boldsymbol{f}_1^*, ..., \boldsymbol{f}_N^*, \boldsymbol{e}^*]^T$ and get a distribution similar to Eq. 5.14:

$$\mathcal{N}\left(\begin{bmatrix} \boldsymbol{F}_e^* \\ \boldsymbol{\theta} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\Omega}_e \boldsymbol{\theta} \\ \mu_\theta \end{bmatrix}, \begin{bmatrix} \boldsymbol{A}_e & \boldsymbol{\Omega}_e \Sigma_\theta \\ \Sigma_\theta \boldsymbol{\Omega}_e^T & \Sigma_\theta \end{bmatrix}\right), \tag{5.17}$$

$$\boldsymbol{\Omega}_e\boldsymbol{\theta} = \begin{bmatrix} \tilde{\phi} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \\ & w_e & \end{bmatrix} \begin{bmatrix} w_1 \\ g_1 \\ \vdots \\ w_N \\ g_N \end{bmatrix}, \tag{5.18}$$

$$\boldsymbol{A}_e = \begin{bmatrix} \sigma^2 & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & \sigma^2 & 0 \\ 0 & \dots & 0 & \sigma_e^2 \end{bmatrix} + \boldsymbol{\Omega}_e \Sigma_\theta \boldsymbol{\Omega}_e^T. \tag{5.19}$$

The mean and variance of the distribution $p(\boldsymbol{\theta}|\boldsymbol{Y}_e^*)$ are then derived by plugging Eqs. 5.18-5.19 into Eq. 5.15.

5.5 Simulation Experiments

The aim of the simulation challenge is to neatly pass over a rectangular object of varying size. Partner observation ($\boldsymbol{Y}_{human}$ in Eq. 5.16) took the form of mouse cursor movement across a GUI. As there is no sensor data to be parsed, here the environmental parameters are the dimensions of the object (px) to be passed over. From the training samples in Figure 5.5 EaIPs were able to generate suitable trajectories as shown in Figure 5.6 even with no observation data present, by capturing the underlying “style” (Matsubara et al. 2011) of movements for improving the task success rate.

Figure 5.7 compares the DTW distance to training samples between IPs and EaIPs for this task over varying lengths of initial observation, where it can be seen that EaIPs yielded a much improved DTW distance to training samples when given a short observation period, with IPs only performing comparably when given lengthy (≥ 100 steps) observations along both horizontal and vertical image axes (Fig. 5.7a). When only provided with partial observations along the horizontal image

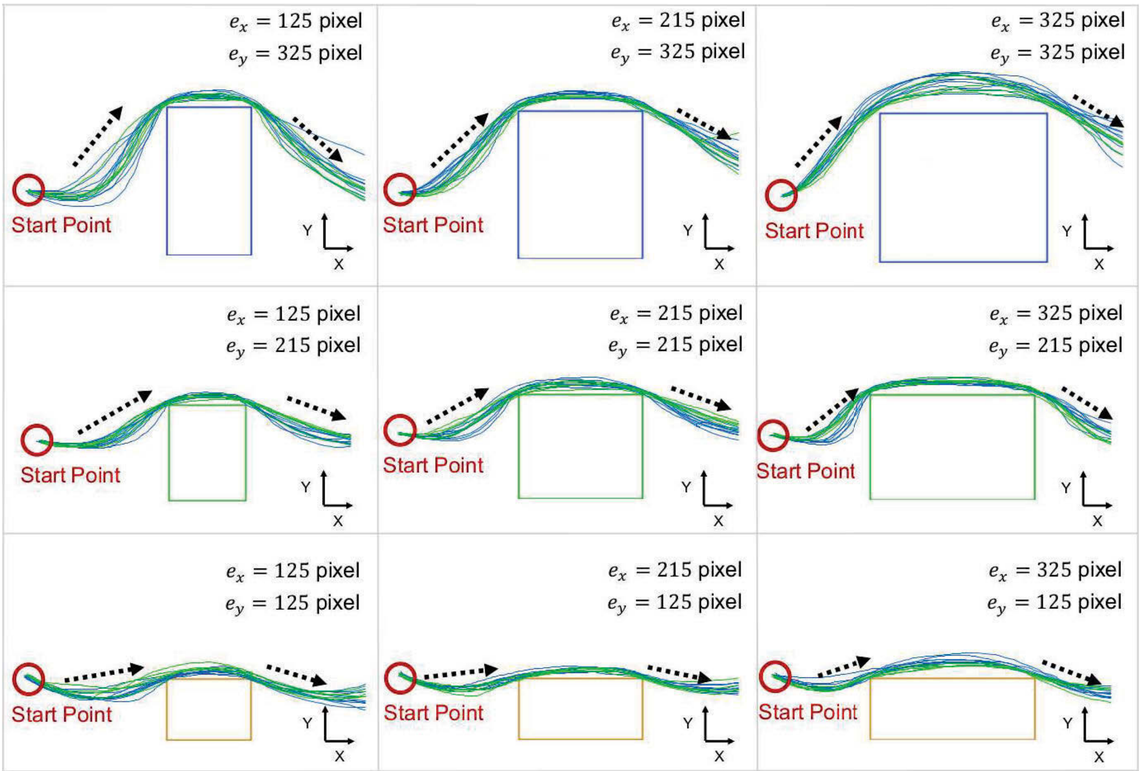


FIGURE 5.5: Training data for simulation experiment of passing over rectangular objects. Training paths are shown in blue, while examples of generated paths are shown in green. $e_{x,y}$ represent the object dimensions, and are provided as environmental parameters.

axis (Fig. 5.7b) EaIPs were able to maintain similar performance to the full observation case due to the consideration of additional parameters, whereas the prediction accuracy of IPs was diminished.

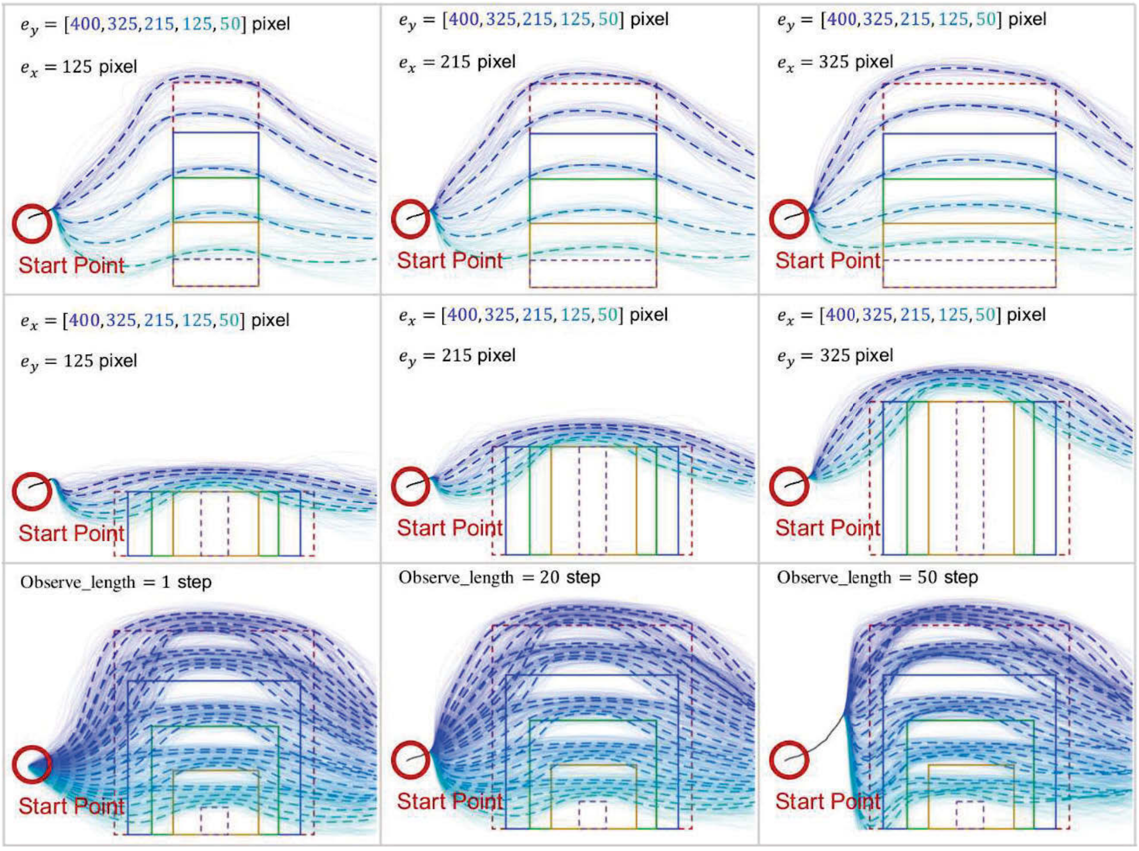
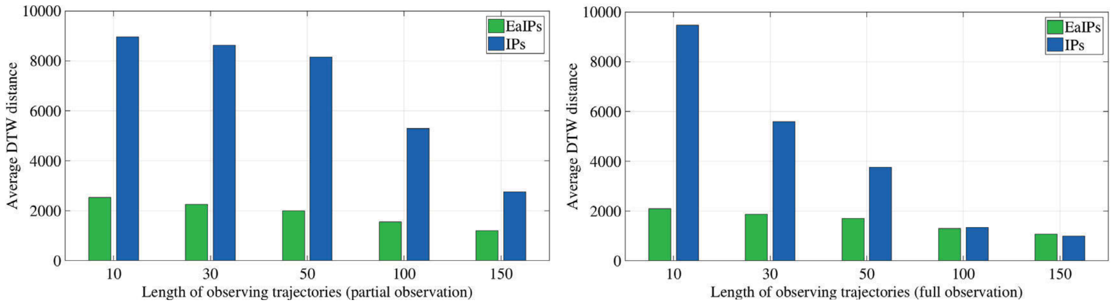


FIGURE 5.6: Testing of EaIPs in passing over rectangular objects. Dashed trajectories indicate the mean of numerous solid trajectories generated for each object, with novel objects indicated by dashed borders. "Observe_length" refers to the length of the observation sequence made available to EaIP.



(A) DTW distance for single observation (horizontal image axis only).

(B) DTW distance for full observation (both image axes).

FIGURE 5.7: Dynamic Time Warping distances (unitless) to training samples for IPs and EaIPs in the simulation task. A smaller DTW distance indicates a better correlation between two sets of time series data.

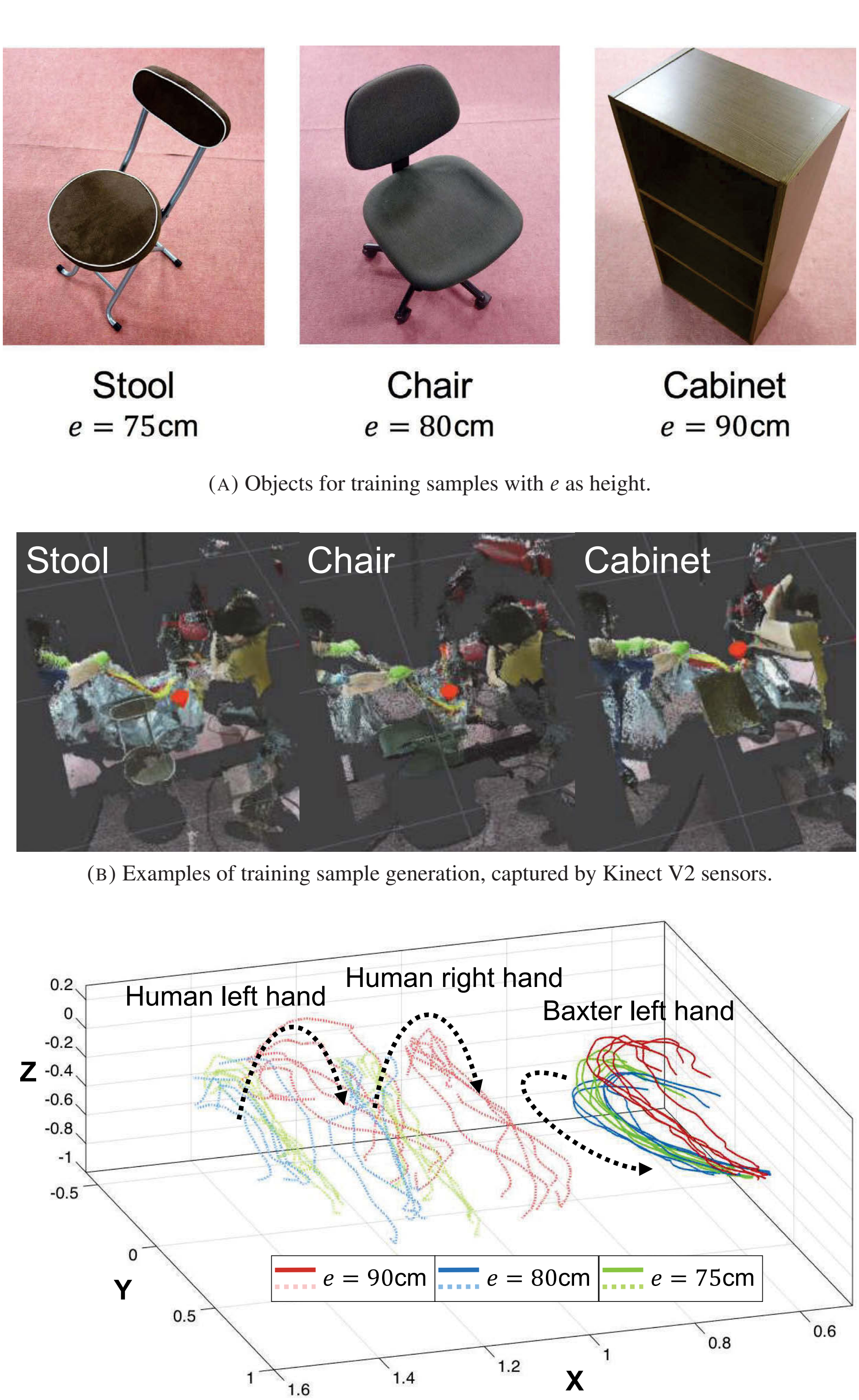
5.6 Real Experiments

Validation of the EaIP framework in a real setting was conducted on the NAIST ISC Baxter research robot (Appendix B). Partner observation ($\boldsymbol{Y}_{human}$ in Eq. 5.16) took the form of 3D centroids from blob tracking of brightly colored cleaning gloves worn on both hands. Sample trajectories then consisted of the poses of the Baxter's grippers and the partner's hand positions. For ease of differentiability, the imaginary axes of gripper orientation Quaternions were learned instead of their Euler equivalents. To perform its share of the collaborative activity (covering an object with a large plastic bag), the robot held a point along the opening of a large bag stationary in its right gripper, while its left arm swept over the object to be covered while holding another part of the bag's lip. Training data was generated by kinesthetic operation of the robot's left arm in tandem with the glove-wearing interaction partner, with each training trajectory fit to 500 steps.

Experiments in this Chapter were run on an Ubuntu machine with 62 GB RAM, Intel i7-5820K CPU, and Nvidia GTX 960 GPU. Trajectory generation took less than one second, although inverse kinematics for the Baxter's arm consumed more time after that.

5.6.1 Results using Environmental Parameters *e* from Ground Truth

Five training trajectories were gathered for each of three objects (Fig. 5.8): a 75 cm high stool, an 80 cm high office chair and a 90 cm high cabinet. Figure 5.9 shows the Baxter's left gripper trajectory in the execution of EalP for covering each of the three known objects with a new interaction partner, as well as a novel fourth object: the 90 cm high cabinet elevated by an additional 10 cm. It can be seen that each trajectory is suitably changed to satisfy each environmental parameter, and is robust to both inconsistent starting positions of the partner's hands and significant noise from glove tracking throughout their motion. Only 50 partner observations (1.5 seconds) were required to converge on the parameter sets that resulted in the trajectories shown. Figure 5.10 shows object coverage by the interaction partner and the EalP-generated trajectories from Figure 5.9, showing appropriate movements within the Baxter arm's range of motion.



(A) Objects for training samples with e as height.

(B) Examples of training sample generation, captured by Kinect V2 sensors.

(C) 15 training sets (includes human left/right hand, and Baxter left gripper).

FIGURE 5.8: Training trajectories for three objects: a stool, chair, and cabinet.

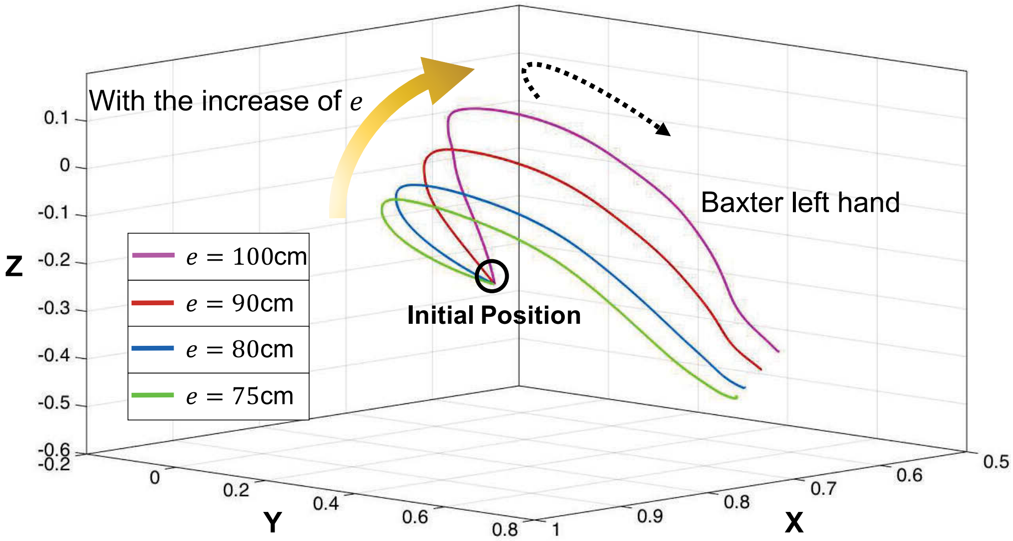


FIGURE 5.9: Baxter left gripper trajectory from EalP across various objects.

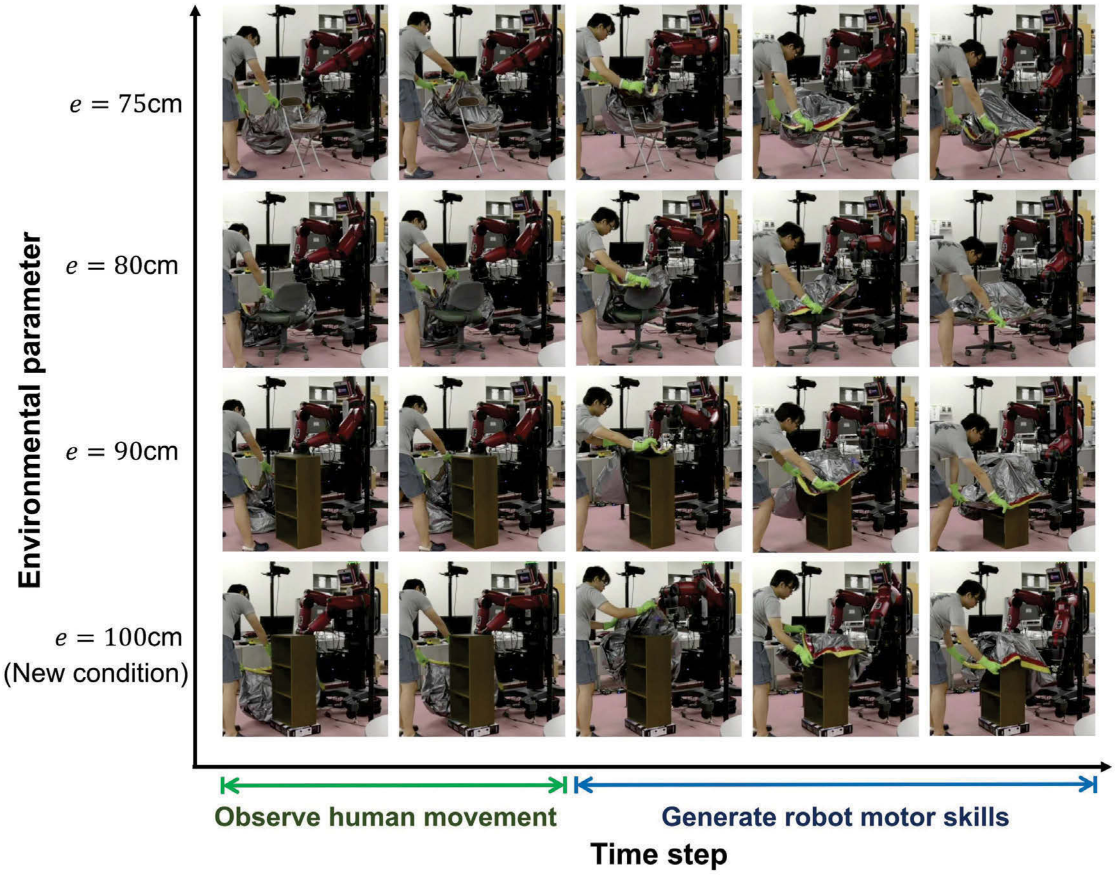


FIGURE 5.10: Results of EalP trajectory generation in cooperative covering task with ground truth e.

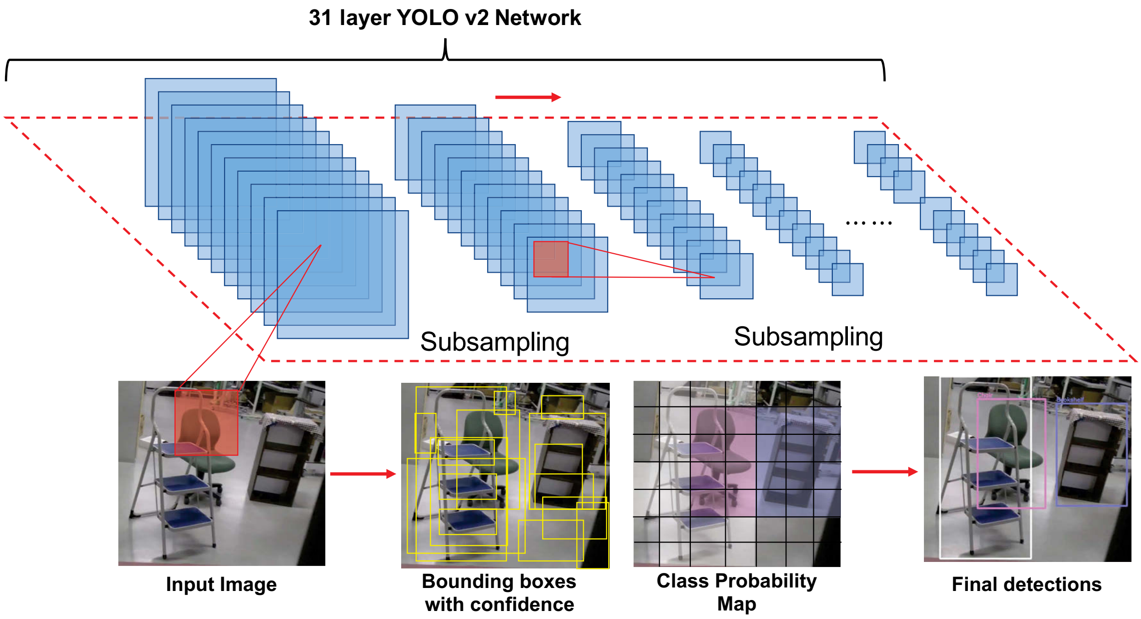


FIGURE 5.11: Deep learning topology for object detection. Convolutions over 31 subsampling layers in YOLO's CNN yield object label and bounding box estimates.

5.6.2 Results using Environmental Parameters *e* from Deep Learning

In this experiment, visual contextualization takes the form of the bounding box properties of the object serving as the focus of the interactive exercise, and are taken as ***e*** in Eq. 5.16. This information is obtained from the YOLO (You Only Look Once v2) (Redmon and Farhadi 2016) CNN object detector (Fig. 5.11). YOLO is particularly well suited to process scene information for EaIP due to its capability to detect objects in real-time from a video stream, as its runtime complexity is far less than that of competing architectures such as VGG-16 (Simonyan and Zisserman 2014) while its bounding boxes provide a degree of correlation to the physical space that EaIPs operate in that can also be readily exploited by the EaIP framework. YOLO processes images as grids, with each grid cell providing estimates of class labels and bounding boxes of objects that may reside within/around it. These estimates are then combined into an overall probability map based on each cell's likeliest label, from which final object detections are drawn.

Although suitable environmental parameters can indeed be also drawn from other information sources such as depth cameras or additional sensory apparatus, we maintain it is less cumbersome to use a vision-based object classifier. As training data is often available in a limited quantity for

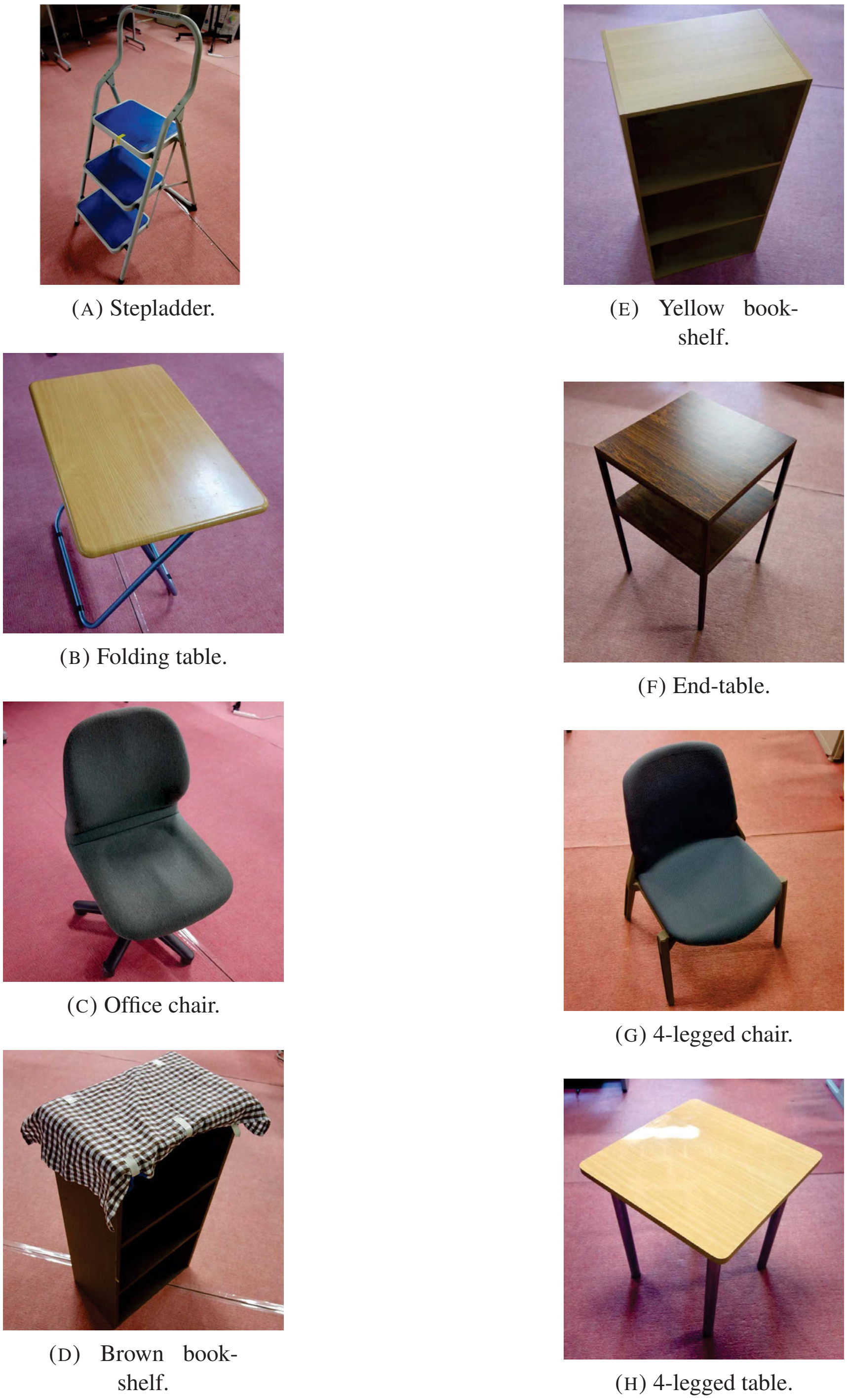
EaIP modelling, models built for parameter regression upon such data would easily be overfit, and it is also undesirable to have to craft heuristics for each task that EaIP is to be applied to. Since classifiers such as YOLO can be exposed to a far broader range of objects than in the EaIP training set, its features immediately allow the EaIP to generalize to new objects and can be used directly as environmental parameters without the need for task-specific modulation.

Figure 5.12 shows both training and testing object sets in this experiment, with ten sample trajectories recorded for each training object. The EaIP framework with YOLO was evaluated on both sets. Figure 5.14 shows the 3D left end-effector paths from both IP and EaIP for all eight objects given a 1s observation period (approximately 10% of average task duration), from a frontal view of the robot's YZ plane. It can be seen that EaIP better preserves the path structure from training samples (Fig. 5.13), and IP's paths possess very little variation between objects. The actions from IP (Fig. 5.14a) resulted in collisions with the three larger objects as shown in Figures 5.16-5.17, whereas EaIPs were able to cover all eight objects successfully [1].

As shown in Figure 5.15b, paths sufficiently close to the 100% observation path can be obtained from EaIP given as little as 10% of partner observation, whereas IPs would require nearly 70% of partner observation to reach a comparable RMSE. It also bears mentioning that even with a 100% of partner observation, in this case IPs still failed to generate a safe trajectory (Fig. 5.15a).

The ability of the EaIP framework to leverage increasingly robust object detection methodologies from the vision community allows for generalization towards novel objects beyond the scope of EaIP training data; at present this is particularly important in demonstration-based human-robot systems as suitable training data is difficult to obtain in large volumes. Our results show that even with a limited quantity of EaIP training data samples, the use of a classfier's bounding box estimate of the object between the human and robot was sufficiently informative in allowing for the robust generation of viable robot actions due to their correlation to physical space.

[1] Video of IP/EaIP for the stepladder and folding table available at `http://youtu.be/59zr_7r3Lfo`.



(A) Stepladder.

(B) Folding table.

(C) Office chair.

(D) Brown bookshelf.

(E) Yellow bookshelf.

(F) End-table.

(G) 4-legged chair.

(H) 4-legged table.

FIGURE 5.12: Training objects (left column) and testing objects (right column).

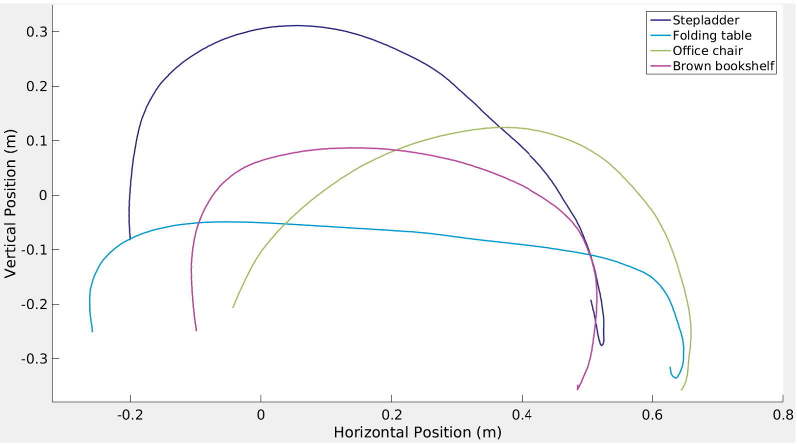
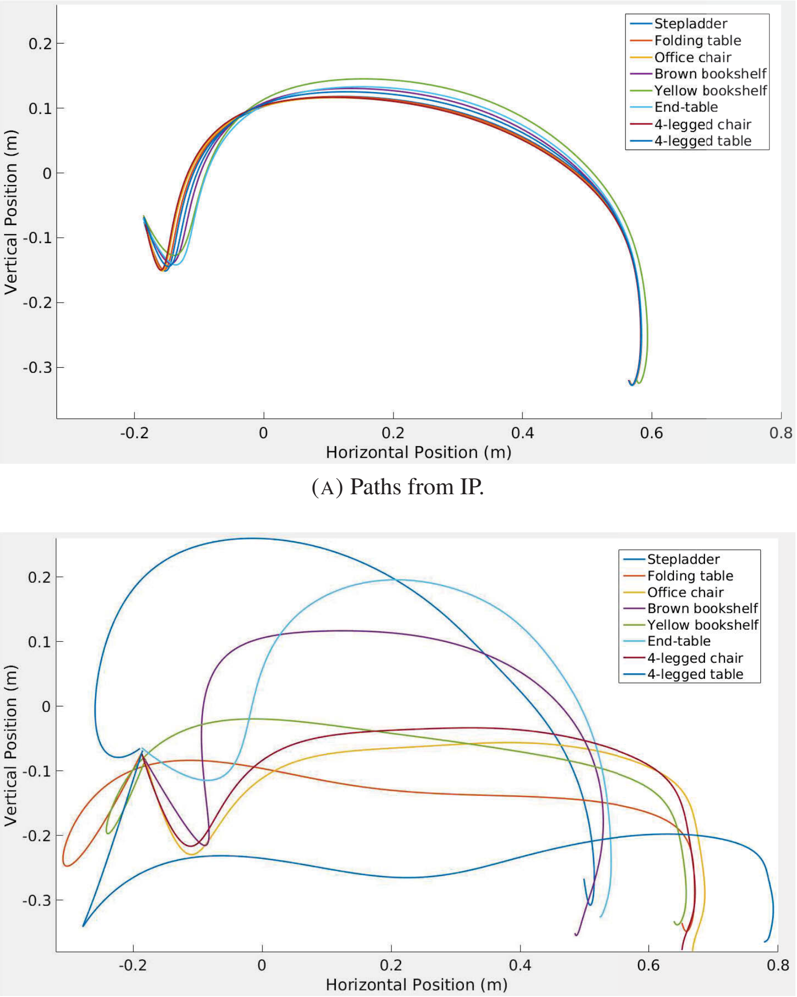


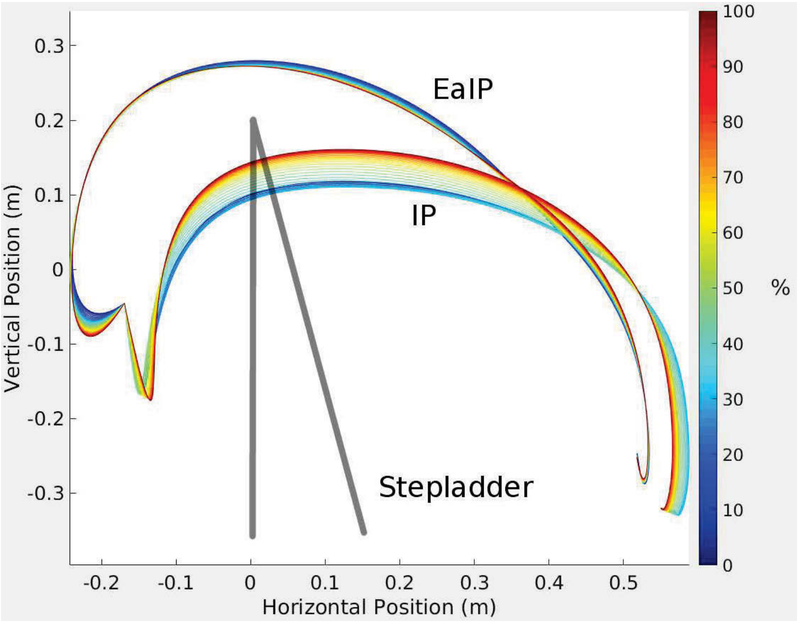
FIGURE 5.13: Mean training data of the Baxter's left end-effector, from 10 samples for each training object.



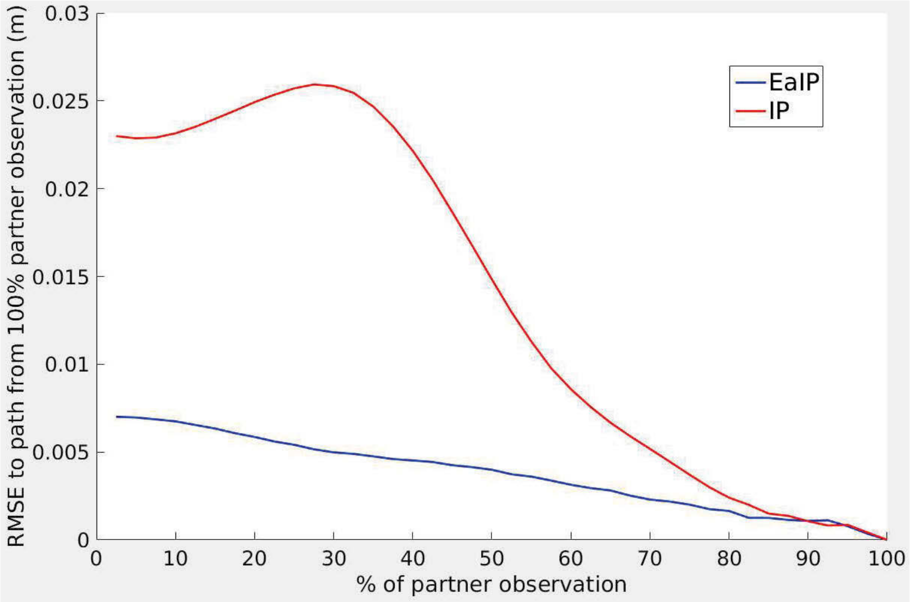
(A) Paths from IP.

(B) Paths from EaIP.

FIGURE 5.14: Baxter left end-effector paths from IP and EaIP.

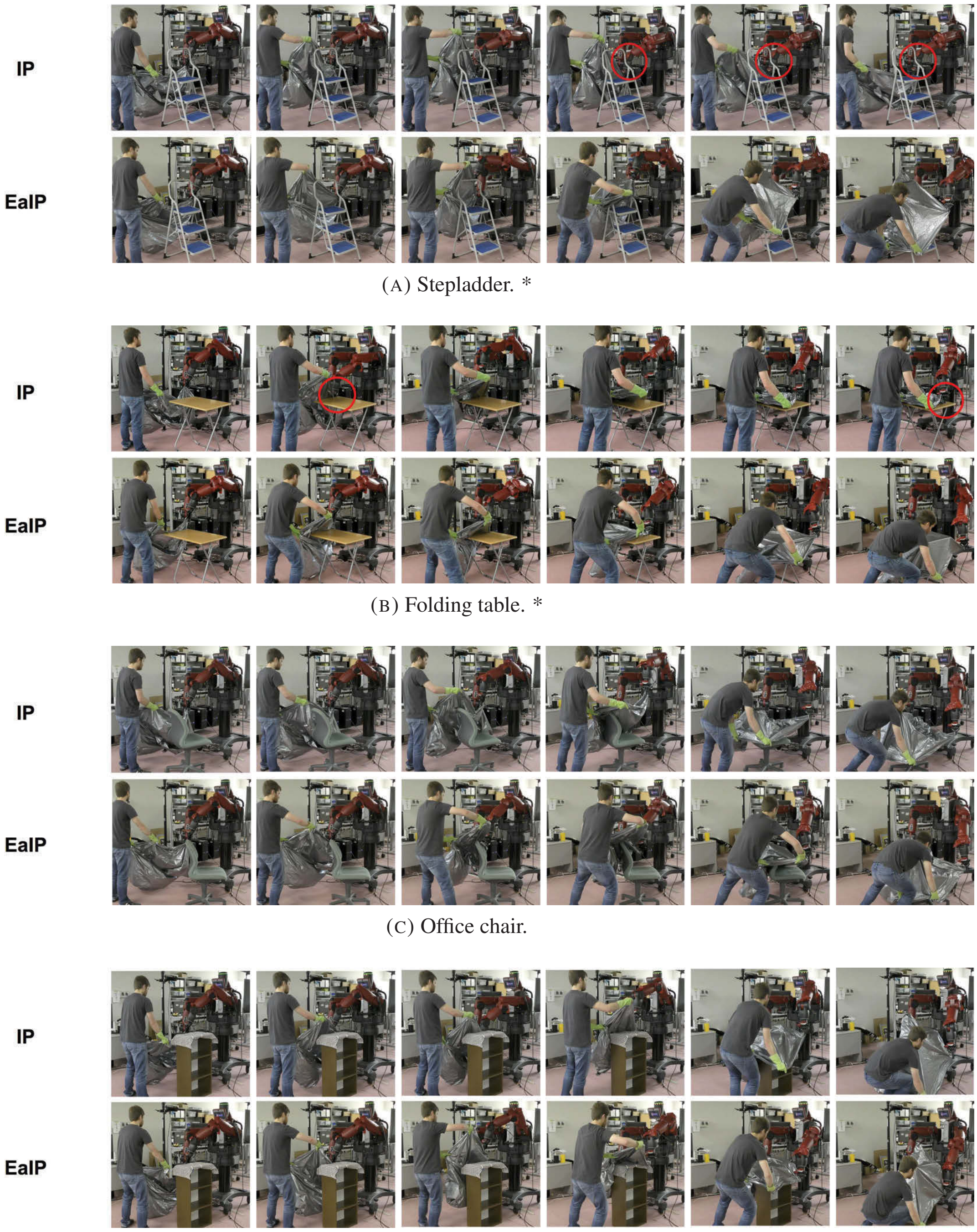


(A) Paths from IP and EaIP under increasing % of partner observation, with the stepladder visualized in opaque black.



(B) RMSE of the above to IP/EaIP's respective final paths.

FIGURE 5.15: A: convergence of *a-posteriori* paths given partner observations of varying duration for IP and EaIP in the stepladder covering exercise (Fig. 5.16a). B: RMSE to their respective convergent paths generated under 100% partner observation. A minimum observation of 10 steps (<3%) is taken to avoid severely premature inferences.



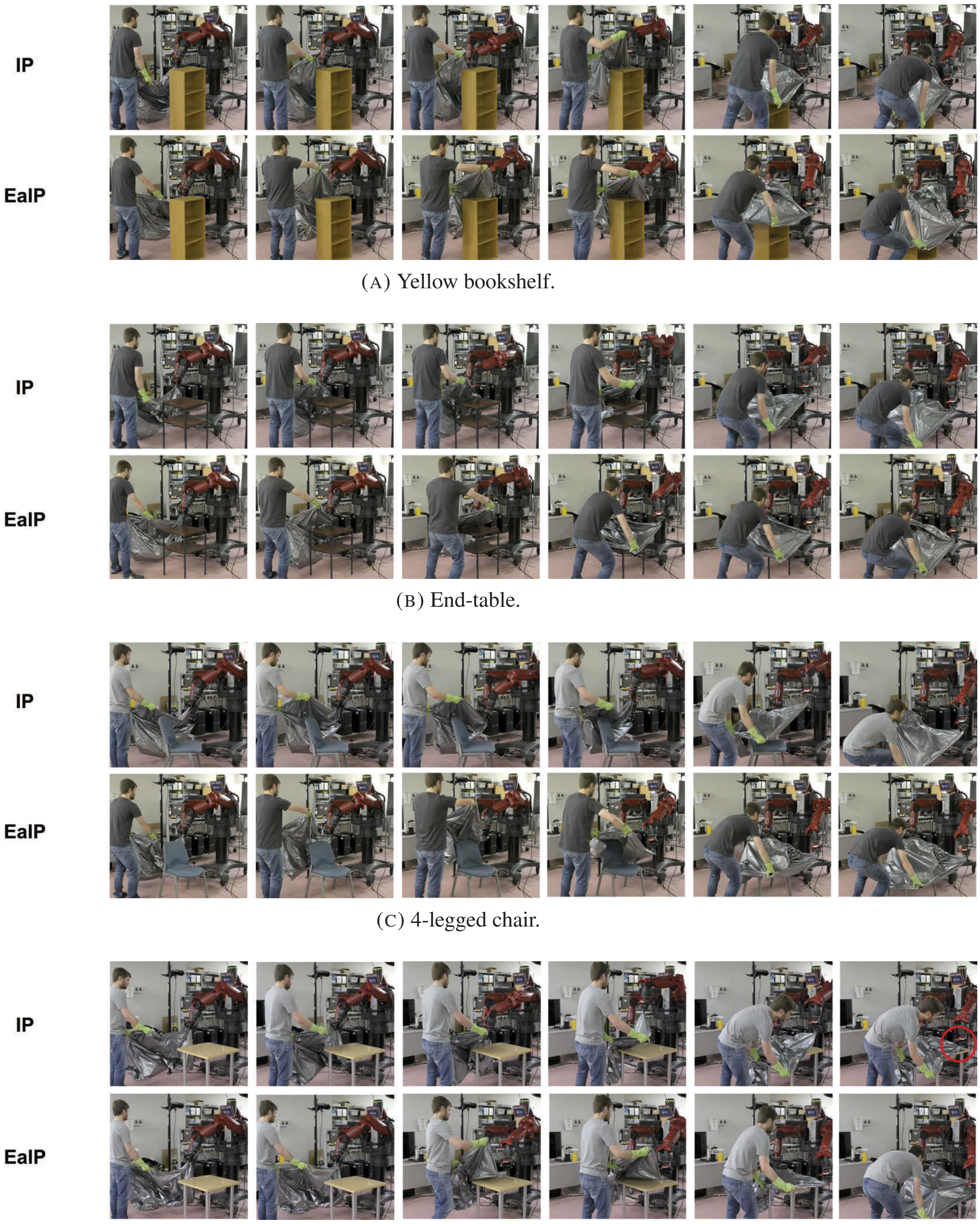
(A) Stepladder. *

(B) Folding table. *

(C) Office chair.

(D) Brown bookshelf.

FIGURE 5.16: Behavior of IP and EaIP on the training set. * indicates the object resulted in collision (red ellipses).



(A) Yellow bookshelf.

(B) End-table.

(C) 4-legged chair.

(D) 4-legged table. *

FIGURE 5.17: Behavior of IP and EaIP on the novel object set. * indicates the object resulted in collision (red ellipses).

5.7 Chapter Summary

This Chapter presents further evaluation of the concept of greater scene consideration within the context of physical human-humanoid collaboration. This is undertaken as an extension to the Interaction Primitives framework by folding additional parameters obtained from visual scene observation into the joint human/robot distribution, as the IP framework lends itself well to additional observation sources for robot action generation in the human-interactive domain. As such a distribution bases resultant robot actions solely upon a brief partner observation, the key benefit of the presented augmentation is a reduction in the reliance upon the task goodness of the human partner during this critical observation window.

We evaluate our Environment-adaptive Interaction Primitives against standard IPs in a collaborative task of a human and a robot covering furniture together with a large plastic bag. In this evaluation, we first provide expert-derived task parameters as a proof of concept, then train the system to derive such parameters automatically via deep learning. Our results show that the consideration of additional parameters greatly enhance robustness and safety in robot action generation, when the robot is faced with partner behaviour which is insufficiently informative during the brief partner observation window. This in turn allows the robot to safely generalize to novel operating conditions; in the context of our experiments, this took the form of changing the object to be covered to ones of taller or wider size than those used in training data. In comparison, as IPs are solely based upon partner movement, the robot was unable to generate safe behaviours in cases where the conditions faced by its partner did not warrant sufficiently noticeable deviations in observed movement.

Chapter 6

Conclusions

6.1 Summary of Thesis

In this thesis, an increased level of integration of task/environmental parameters into the design of learning from demonstration models serving as robot action planners in human-robot interactive frameworks is investigated. The motivation behind this is to reduce the dependence of such frameworks on parameters derived from observation of human inputs or behaviours, which may often be either sub-optimal or insufficiently informative to ensure safe robot action planning.

This concept was evaluated within the scope of two significantly different human-robot interaction paradigms: intelligent assistive navigation for PMDs, and physically collaborative undertaking of an object handling task by a team comprised of a human and a humanoid robot. Results demonstrate that shifting the emphasis of the learning model towards the task/environment, and the subsequent reduction of reliance upon task goodness in observed human behaviours, has appreciable benefits across both interaction domains.

Following an overview of the developments documented in this thesis and their respective outcomes, Section 6.2 finishes with potential directions for further investigation drawn from perceived limitations concerning the body of work presented here.

6.1.1 Intelligent PMD Navigation

A sequential approach to global PMD intention estimation is presented in Chapter 2 for places in which upkeep of a *a-priori* occupancy map is feasible, such as in a home setting. Due to their inherent consideration of uncertainty which allows for inferences to be analytically filtered over time, a Gaussian Process regression model was selected. The aim of the learning model in this context is to consider both the user input signal and PMD pose within the known map as inputs, and to then yield an estimate of the likeliest goal from a set of destinations distributed throughout the known environment. This is first roughly undertaken by a greedy Artificial Neural Network as a proof-of-concept. By sparsifying the Gaussian Process model, it is able to provide sequentially filtered estimates and uncertainties in real-time on data obtained from driving the CAS PMD around a portion of UTS campus.

The next two Chapters present work towards local intention estimation, designed to preserve the benefits of active navigational assistance in novel environments or places where occupancy map upkeep is not a viable option. On a conceptual level the intention estimation problem has not significantly changed: the pose of the PMD as part of the input to the learning model is now replaced with information derived from immediately available sensor data, and the output estimate is now a waypoint within the immediate space around the PMD rather than a static destination an arbitrary distance away. Under this change the leveraging of the robot's perception allows for generalization to places beyond those seen in training, while also compensating for deficiencies in partner behaviour in a similar manner to the PMD pose data in Chapter 2.

Given a limited quantity of training data samples, a heuristic model drawing upon a significant amount of domain knowledge is first built in Chapter 3, yielding deterministic local intention estimates. After inferring a short-term user intention, demonstration-stylized path primitives are considered alongside local occupancy maps in a stochastic optimal planner to yield a path for the PMD to follow. Experiments evaluating the closeness of the planned paths to user intent performed both *a-posteriori* and on-line with a variety of users demonstrate the predictive and planning capabilities of this assistive framework.

For a larger training data pool, a revised framework with a reduced reliance upon task knowledge is presented in Chapter 4. Here, a deep learning Convolutional Neural Network is trained in conjunction with a Gaussian Process approach for the generation of two local heatmaps by initially considering sensor data and user input in a decoupled fashion. Each continuous heatmap represents an estimate of traversal likelihood in the PMD's immediate space, by folding expert pathing into intention estimation. This conceptually simpler framework is able to perform comparably in terms of predictive capability to the methodology in Chapter 3 while also being computationally faster, as demonstrated by an offline evaluation.

6.1.2 Human-Humanoid Collaboration

In contrast to PMDs, in this context on-board sensor data is replaced with an externalized vision system and the final output of the learning framework is an end-effector trajectory facilitating the collaborative execution of a broad range of co-operative human-robot activities instead of a path for a mobile ground vehicle, but the overall concept of enhanced scene consideration nevertheless follows through. As the Interaction Primitives (Ben Amor et al. 2014) robot learning framework is built for the parameterized generation of end-effector trajectories based on partner observation and the framework itself is amenable to the incorporation of additional observation sources, it readily serves as a foundation for the evaluation of a greater degree of scene perception within the human-humanoid interactive domain. Chapter 5 thus presents work towards augmenting Interaction Primitives with additional "task" parameters drawn from the scene around both agents. These parameters can come from task knowledge heuristics, or be derived through perception of the environment around the human-robot team; allowing the robot to perceive the environment around it is intuitively closer to natural human-human behavior in a similar scenario.

IPs first model a joint Gaussian distribution between parameterized human and robot trajectories. During execution, a short observation of the trajectory of its human partner then allows for robot trajectory parameters to be drawn. A noticeable limitation arises from a general lack of informativeness at the onset of the joint action during which the partner observation is taken, given that it must be brief to avoid an undesirable delay in robot movement. It is this shortcoming that the inclusion of additional parameters in our proposed Environmental-adaptive Interaction Primitives aims to address, with the objective of increasing the robustness of resultant robot actions.

In our validation experiments of a human and a robot collaboratively sweeping a large plastic bag over furniture items, we evaluate EaIPs with additional parameters firstly taken as the ground truth heights of the objects to be covered, then in the form of bounding box properties from deep learning object detection. Comparison with IPs show that EaIPs can still generate safe trajectories when the team is covering novel items which are abnormally high/wide when faced with a brief observation window, whereas IPs can not ensure safe robot trajectories when the behaviour of its partner does not sufficiently indicate changing conditions.

6.1.3 Contributions and Outcomes

In summary, the main contributions of this thesis encompass the following:

- Sequential global PMD navigational intention estimation within a known map, using a sparsified Gaussian Process regression model.
- A deterministic short-term local intention estimator for PMDs, solely utilizing on-board sensor data and user input to enable functionality in novel settings without *a-priori* maps.
- Continuous local PMD intention estimation by processing sensor data and user input in a decoupled manner, which naturally merges short-term destination inference with desirable path planning behaviours during learning.
- Real-time stochastic short-term path planning, via reinforcement learning schemes fusing intention estimates with local occupancy maps for feasible robot paths.
- The incorporation of additional task parameters to alleviate the issue of robustness in purely human activity-driven robot action generation, within the context of collaborative human-humanoid interaction.

6.2 Future Research Directions

6.2.1 Intelligent PMD Navigation

The on-line adaptive modelling of the inputs of new disabled users against their estimated desired signals would be beneficial for larger-scale PMD deployment, as the effects of disabilities upon driving behaviours would almost certainly evolve over time in accordance with ongoing rehabilitation or degradation. Doing so would remove the need for gathering data from disabled users solely for their profiling which may prove difficult with end-users with complex cognitive issues, especially in the case of data collection within simulation as per this thesis. Alternative means of user input such as wearable electro-encephalography devices (Carlson and del R. Millan 2013) may also be better suited to individuals suffering from significant physical disabilities than conventional joysticks; depending on data structure the models built to interpret user signals would have to be adapted to suit.

Depth-sensing cameras such as the Microsoft Kinect may eventually prove better suited for real-world navigation where obstacles at differing heights such as artefacts along the ground or protruding items such as tables would also need to be taken into account for pathing or control purposes. Dynamic obstacles such as pedestrians and vehicular traffic, which have been largely ignored in this thesis, are also important factors for consideration before a larger-scale public evaluation can become feasibly safe. A sensor arrangement allowing for 360 degree visibility around the PMD would also further enhance user safety and fully utilize the square planning envelope in Chapter 4.

Of more immediate interest would be a larger-scale clinical trial with a substantial pool of disabled individuals for evaluation of the proposed frameworks. As training data is gathered in a simulation setting, it is also feasible to explore the optimal minimum training set required for an acceptable level of generalizational ability.

6.2.2 Human-Humanoid Collaboration

A higher degree of partner observation may be beneficial for both IP and EaIP. In our experiments, only the person's hands were tracked, whereas skeleton tracking would provide more informative partner observations than hands alone albeit at the cost of requiring more training data to allow for this increased detail to be utilized efficiently. In terms of robot control, with a greater amount of training data samples it may be possible to directly generate joint trajectories for the robot, rather than generating Cartesian end-effector trajectories followed by a considerably computationally demanding conversion step into joint space.

The classification label of objects that the interaction is focusing upon, or objects recognized as those that the human is reaching for, may also serve as part of the task parameter set for EaIP. This would have the effect of reducing dependence upon locations which the partner is reaching for (Maeda et al. 2017), and shifting focus to the type of action they wish to perform next. A trained regression engine for task parameters, rather than classification, may also improve generalization capability.

For sequential activities such as furniture assembly which consist of multiple tasks executed in an order loosely dictated by the human partner, incorporating IPs or EaIPs as action generation modules within a larger transitional framework such as a Hierarchical Hidden Markov Model would allow for additional robustness as the learning system can identify likely next actions based on tasks which remain to be undertaken.

Appendix A

UTS CAS Wheelchair

A.1 Hardware Overview

The UTS Centre for Autonomous Systems wheelchair used for the experiments in Chapters 2-3 is a modified Invacare Roller M1 fitted with a forward-facing Hokuyo UTM-30LX LIDAR, shaft-mounted wheel encoders, and an Arduino-based hardware interface to both read user input signals and spoof joystick commands from the PC back to the controller module. Its overall system diagram is shown in Figure A.1.

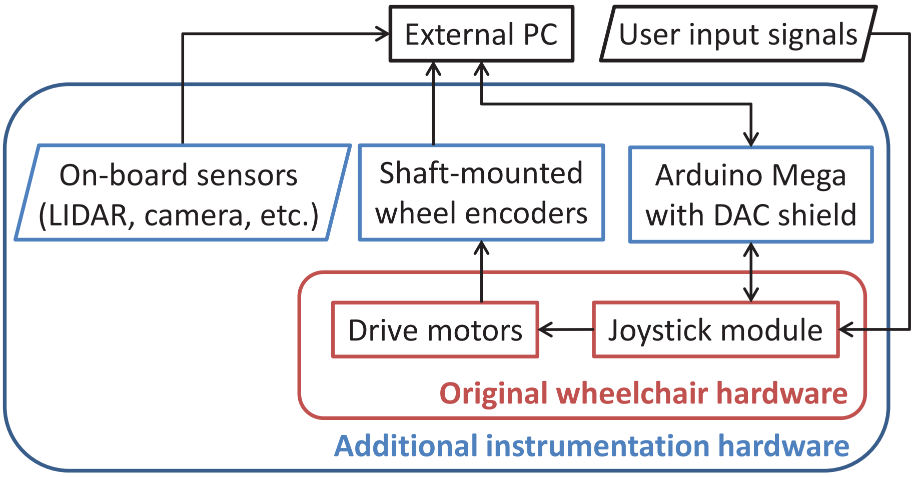
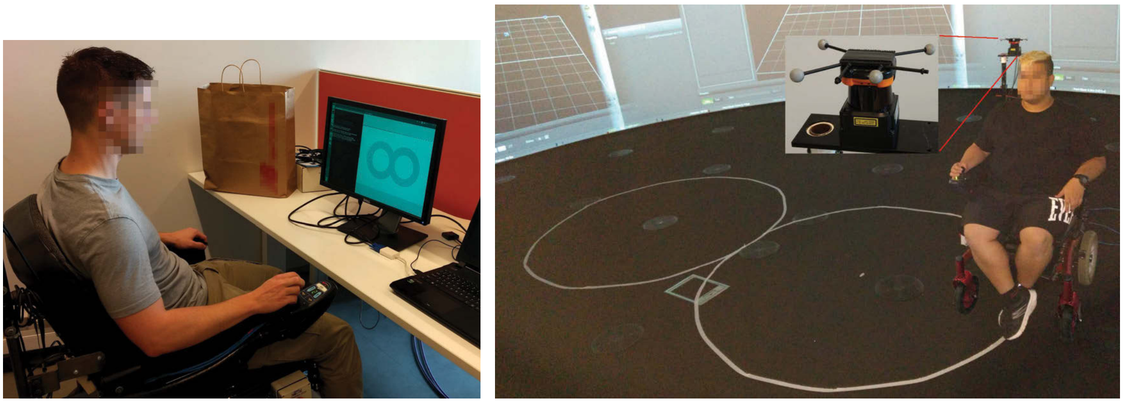

External PC
User input signals
On-board sensors (LIDAR, camera, etc.)
Shaft-mounted wheel encoders
Arduino Mega with DAC shield
Drive motors
Joystick module
Original wheelchair hardware
Additional instrumentation hardware

FIGURE A.1: High-level schematic of the UTS CAS instrumented wheelchair.



(A) Simulation experiment using the integrated wheelchair joystick.

(B) Real experiment in the Data Arena, overlaid with a close-up of a 6-DOF tracking marker on top of the wheelchair's LIDAR unit.

FIGURE A.2: Figure-8 driving experiments using the wheelchair's hardware, in both simulation and the UTS Data Arena.

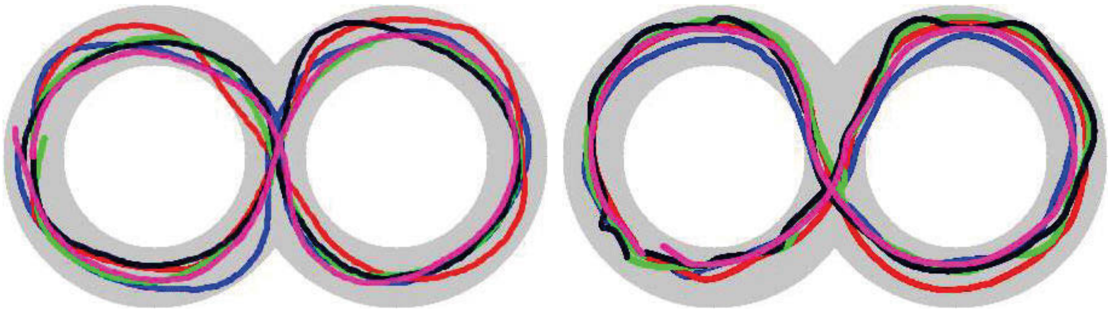
A.2 Comparison with Simulated PMD

This section documents a comparison of platform behavior and behavior of users when controlling the wheelchair in reality and in Player/Stage (Gerkey et al. 2003), to validate the in-silico PMD utilized for some of the experimental work in Chapters 3 and 4.

Five able-bodied users drove around a $4{\times}2m$ figure-8 in both simulation (Fig. A.2a) and in the UTS Data Arena (Fig. A.2b), an augmented reality/motion capture facility capable of real-time marker tracking with millimeter accuracy which serves as a localization ground truth. The resultant paths are shown in Figure A.3.

Driving metrics from these experiments are presented in Table A.1; no collisions occurred throughout. The total distances travelled and the low path deviation indicate strong task relevance of the data collected, and validity for this analysis. A similar level of steering entropy and angular jerk was also observed, representing a similar level of participant task effectiveness between both experiments. Completion time was several seconds longer in the Arena than in simulation; we attribute this to the slightly higher forward joystick signal in simulation, and also to friction from the Arena floor's acoustic padding.

Given the similarities in the results from both experiments we argue that the simulation is a viable evaluation setting for the wheelchair in terms of human factors; it can be seen that users perform



(A) Paths taken by users in simulation.

(B) Paths taken by users in the Data Arena.

FIGURE A.3: Resultant figure-8 paths from simulation and the UTS Data Arena. The gray circles have a thickness of 0.5m.

similarly in terms of task completion and efficiency.

Metric	Simulated	Real
Distance travelled (m)	13 ± 0.4	12.7 ± 0.8
Avg. path deviation (m)	0.05 ± 0.01	0.07 ± 0.006
Steering entropy	0.56 ± 0.03	0.61 ± 0.04
Avg. angular jerk (rad/s^3)	0.01 ± 0.01	0.02 ± 0.01
Avg. forward joystick signal	0.46 ± 0.1	0.41 ± 0.1
Time taken (s)	42 ± 8	46 ± 6

TABLE A.1: Driving metrics from the figure-8 experiments.

Appendix B

NAIST ISC Baxter Robot

The Baxter system at the Nara Institute of Science and Technology's Intelligent System Control Lab (Fig. B.1) consists of a Baxter upper-body humanoid from Rethink Robotics, whose default end-effectors have been replaced with Robotiq 2-finger grippers fitted with tactile sensors.

Environmental perception of the robot is enhanced (Fig. B.2) by 3 overhead Kinect v2 units, extrinsically calibrated via a 2D augmented reality tracking marker and the Baxter's own eye-in-hand cameras. The calibration is performed as a multi-stage process; firstly the Baxter's arms are moved as to not obstruct the Kinects' view of the marker, and then marker → Kinects transforms are estimated. Next, one of the Baxter's eye-in-hand cameras are positioned above the marker and a Baxter → marker transform is estimated. Finally the Baxter → Kinects transforms are computed by combining the estimated transforms together, allowing for incoming Kinect data to be readily transformed into the Baxter's co-ordinate frame.

For the experiments in Chapter 5, tracking of the gloves worn on the human's hands was performed by a HSV blob tracking/filtering module distributed across 3 PCs with each PC processing the point cloud data from a single Kinect. Additional ROS modules were written to handle IP/EaIP trajectory data from MATLAB matrices and to guide the Baxter through them.

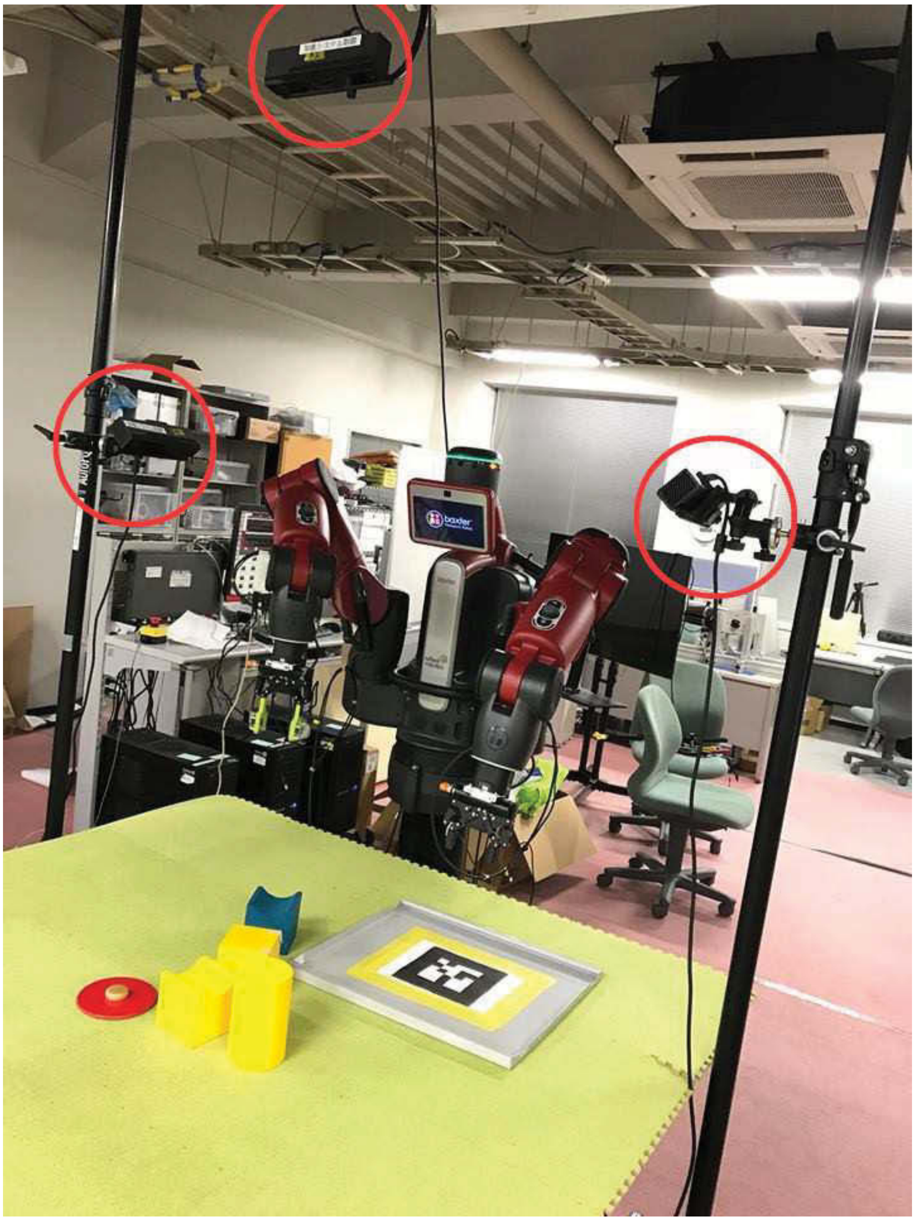


FIGURE B.1: Complete Baxter setup at NAIST during extrinsic Kinect calibration. The black-and-white pattern on the table is the AR marker. Kinect units are circled in red.

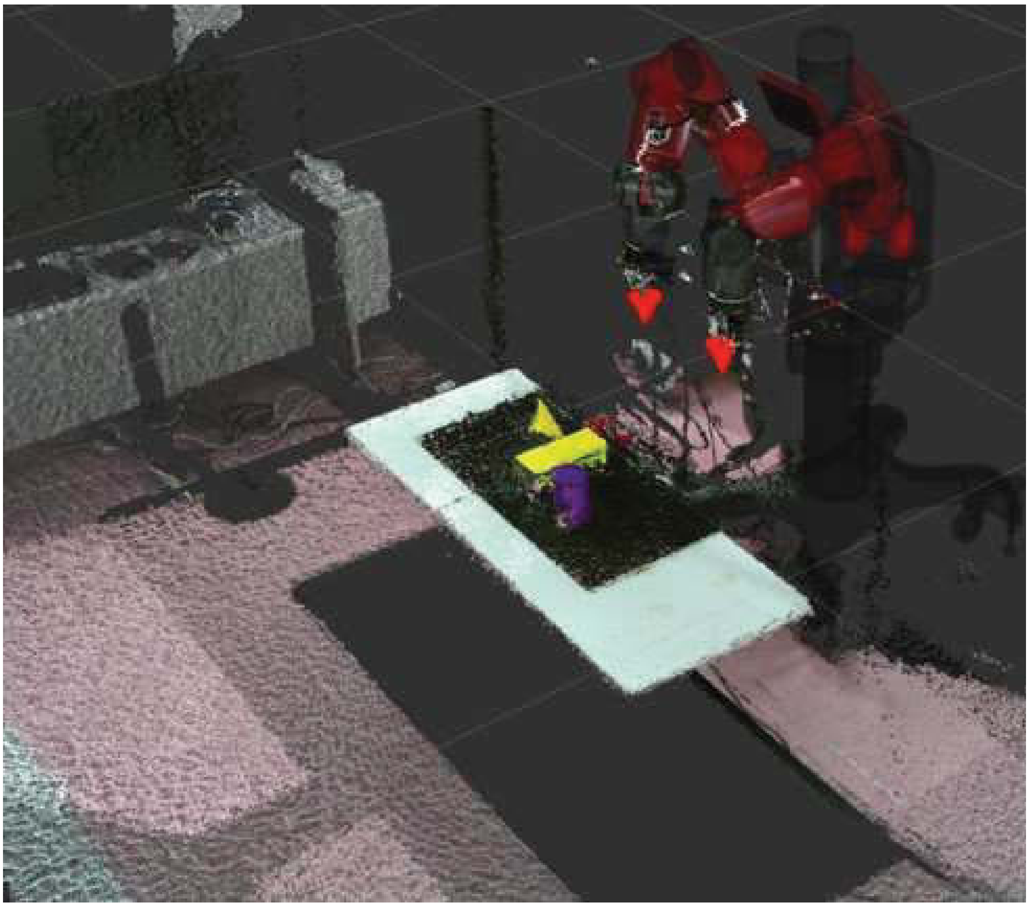


FIGURE B.2: A concatenated point cloud in the Baxter's co-ordinate frame.

Appendix C

Supervised Learning Models

C.1 Radial Basis Function Networks (Orr 1996)

A RBFN consists of n Radial Basis Functions (RBFs); this thesis uses Gaussian RBFs each with their own mean $\boldsymbol{\mu}$ but sharing a common variance σ^2, and assumes a 0 prior. For N query points $\boldsymbol{X}^*$, its corresponding estimate vector Y^* is the product of a $N \times n$ "activation" matrix A and a $n \times 1$ weight vector $\boldsymbol{w}$. For Gaussian RBFs, A_{ij} is the probability density at $\boldsymbol{X}_i^*$ from $\mathcal{N}(\boldsymbol{M}_j, \sigma^2)$.

Given training inputs $\boldsymbol{X}$ and targets $\boldsymbol{Y}$, a RBFN can be quickly built for each dimension of $\boldsymbol{y}$ as per Algorithm 5 (MathWorks 2017). Y represents a single dimension (column vector) of $\boldsymbol{Y}$. Additional criteria can also be applied, such as imposing an upper limit on the number of RBFs or ensuring that each RBF occupies a unique $\boldsymbol{\mu}$.

```
input  : training inputs $\boldsymbol{X}$, training targets $Y$, basis function hyper-parameters $\theta$, error threshold $s$
$\boldsymbol{M} \leftarrow []$; $\boldsymbol{w} \leftarrow []$; $e \leftarrow \sup|Y|$
while $e \geq s$ and additional termination criteria not met do
    $\boldsymbol{M} \leftarrow [\boldsymbol{M}; \boldsymbol{X}_i]$, where $i$ is the index where $e$ occurred
    $A \leftarrow$ activation values given $\boldsymbol{X}, \boldsymbol{M}, \theta$
    $\boldsymbol{w} \leftarrow$ least squares fit of $Y/A$
    $Y^* = A\boldsymbol{w}$; $e \leftarrow \sup(Y^* - Y)^2$
end
output: RBF means $\boldsymbol{M}$, weights $\boldsymbol{w}$
```

Algorithm 5: Building a Radial Basis Function Network.

C.2 Artificial Neural Networks (Funahashi 1989)

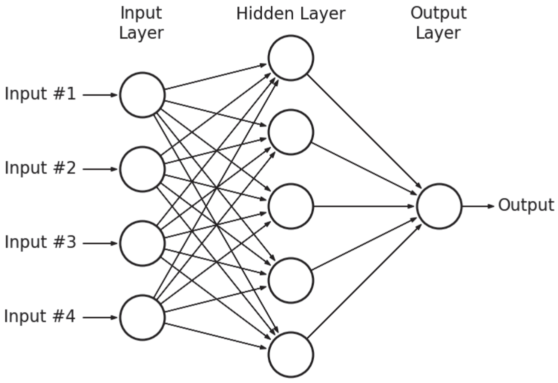


FIGURE C.1: A 3-layer Artificial Neural Network. From VanderPlas (2013).

Artificial Neural Networks map an input layer to an output layer through one or more hidden layers. The neurons in each layer (depicted by circles in Fig. C.1) following the initial input layer each possess an individual weight vector $\boldsymbol{w}$, and the output value a of each neuron is a weighted sum z of the preceding layer passed through a non-linear "activation" function $f(z)$. Among the most commonly utilized activation functions are the sigmoid function (Eq. C.1) and the tanh function (Eq. C.2).

$$a = f(z) = \Big(1 + \exp(-z)\Big)^{-1} \tag{C.1}$$

$$a = f(z) = \frac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)} \tag{C.2}$$

ANNs are typically trained by initializing the weights to random values, and iteratively optimized against training targets in a stochastic manner. Widely used methods of weight optimization include conventional back-propagation of target error (Nielson 2017) or heuristics such as the Rprop family (Igel and Husken 2000). Algorithm 6 outlines the back-propagation method, with $\circ$ denoting the element-wise product of two vectors. Typically, neuron weight updates $\delta \boldsymbol{W}$ are averaged over multiple training samples with a small positive learning rate γ. There also exist further heuristics to improve network performance, such as Dropout (Srivastava et al. 2014) to increase robustness when input elements may be invalid or absent in practice.

As the topology of the network or the designation of its weights do not directly reflect upon the latent function that the ANN is approximating, the ANN is considered to be a black box model.

```
input : training input $X$, training target $Y$, activation function $f(z)$ and first-order derivative
        $f'(z)$, pseudo-random weights $\boldsymbol{W}$, learning rate $\gamma$
$\boldsymbol{Z} \leftarrow$ neuron weighted sums; $\boldsymbol{A} = f(\boldsymbol{Z})$
$\boldsymbol{D}_n = Y - \boldsymbol{A}_n$; $\delta\boldsymbol{W}_n = \boldsymbol{A}_{n-1} \circ \boldsymbol{D}_n$
for $l = n-1; l > 1; l = l-1$ do
    $\boldsymbol{D}_l = (\boldsymbol{W}_{l+1} \circ \boldsymbol{D}_{l+1}) \circ f'(\boldsymbol{Z}_l)$; $\delta\boldsymbol{W}_l = \boldsymbol{A}_{l-1} \circ \boldsymbol{D}_l$
end
$\boldsymbol{W} = \boldsymbol{W} + \gamma \times \delta\boldsymbol{W}$
output: trained weights $\boldsymbol{W}$
```

Algorithm 6: Back-propagation for a n-layer Artificial Neural Network.

C.2.1 Convolutional Neural Networks (Lecun et al. 1998)

CNNs are trained and run in a similar fashion to ANNs. The key differences are that weights are now shared between neurons in each layer, and each neuron only considers a sliding window of the previous layer instead of the previous layer in its entirety. This is due to CNNs being designed around the handling of structured input data such as images, whereas ANNs do not inherently consider the potential for patterns within regions of adjacent input elements.

Image-to-image mapping is commonly done by means of an encoder/decoder system; a low-dimensionality latent feature matrix is first extracted from the input image (the "encoding" step), followed by the latent features being transformed back into a target image (the "decoding" step). An example CNN structured for this purpose is shown in Figure C.2.

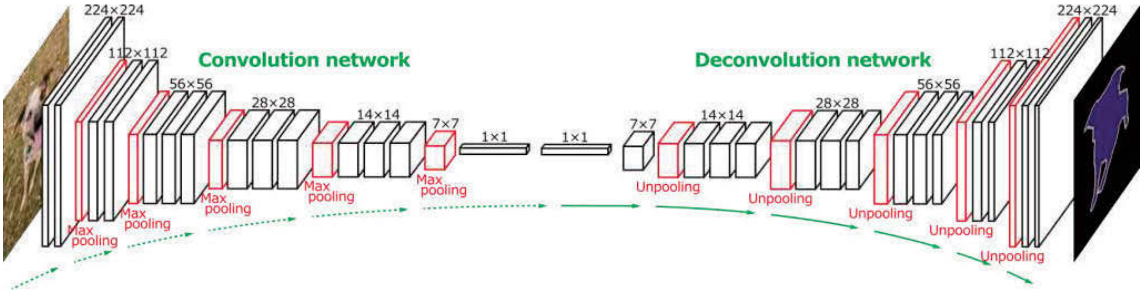


FIGURE C.2: Example schematic of a fully convolutional Neural Network, consisting of an encoder followed by a decoder. From Ozhiganov (2016).

C.3 Gaussian Process Regression (Rasmussen and Williams 2005)

One of the most promising probabilistic techniques for building generative models are non-parametric regression models, i.e., Gaussian Process Regression, given their simplicity and flexibility. While computational complexity across both time and memory consumption remains a bottleneck, several sophisticated approximation schemes are available nowadays to reduce their burden for more practical applications (Candela and Rasmussen 2005; Snelson and Ghahramani 2005).

The non-linear function f is first assumed:

$$\boldsymbol{Y} = \boldsymbol{f}(\boldsymbol{X}) + \boldsymbol{\epsilon},$$

where $\boldsymbol{Y}$ are training targets, $\boldsymbol{X}$ are training inputs, and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \Sigma_{\boldsymbol{\epsilon}})$ is observation noise distributed normally with $\Sigma_{\boldsymbol{\epsilon}} = \text{diag}\left\{\sigma_1^2, \sigma_2^2, \ldots, \sigma_{d_y}^2\right\}$.

To learn such a relationship from data, a GP is fitted with the prior for each dimension of $\boldsymbol{y}$ as follows:

$$\boldsymbol{y}_a \sim \mathcal{GP}(0, K_a),$$

for $a = 1, 2, \ldots, d_y$. The observation noise is $\epsilon_a \sim \mathcal{N}(0, \sigma_a^2)$. For the remainder of this Section the index a is dropped, and we denote Y and y as the a^{th} column vectors/elements of $\boldsymbol{Y}$ and $\boldsymbol{y}$ respectively.

Each covariance matrix K is of size $N \times N$, where N is the number of training samples. It is populated by means of a covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$, with element $K_{ij} = k(\boldsymbol{X}_i, \boldsymbol{X}_j)$. A covariance function (also known as a kernel) can be applied in the case of GPs if it is symmetrical; i.e. $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}', \boldsymbol{x})$, and positive semidefinite; i.e. $c^{\mathrm{T}} K c \geq 0$ for all $c \in \mathbb{R}^N$.

As the maximum *a-posteriori* estimate of k's hyper-parameter set $\boldsymbol{\theta}$ occurs where $p(Y|\boldsymbol{X}, \boldsymbol{\theta})$ is greatest, $\boldsymbol{\theta}$ can be optimized by maximizing the log of the marginal likelihood:

$$\log p(Y|\boldsymbol{X}, \boldsymbol{\theta}) = -\frac{1}{2} Y^{\mathrm{T}} K^{-1} Y - \frac{1}{2} \log |K| - \frac{N}{2} \log 2\pi. \tag{C.3}$$

For inferring y^* at query point $\boldsymbol{x}^*$, $[Y; y^*]$ are treated as samples from some multivariate Gaussian:

$$\begin{bmatrix} Y \\ y^* \end{bmatrix} = \mathcal{N}\left(0, \begin{bmatrix} K & K_*^{\mathrm{T}} \\ K_* & K_{**} \end{bmatrix}\right)$$

where $K_* = [k(\boldsymbol{x}^*, \boldsymbol{X}_1), k(\boldsymbol{x}^*, \boldsymbol{X}_2), ..., k(\boldsymbol{x}^*, \boldsymbol{X}_N)]$ and $K_{**} = k(\boldsymbol{x}^*, \boldsymbol{x}^*)$ respectively.

Then the predictive conditional distribution for y^* can be derived as:

$$p(y|Y) = \mathcal{N}(K_{a*}K^{-1}Y, K_{**} - K_*K^{-1}K_*^{\mathrm{T}}). \tag{C.4}$$

C.3.1 Common Kernels

The square exponential function is one of the most widely used kernels:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \alpha^2 \exp\left(-\frac{1}{2}||\boldsymbol{x} - \boldsymbol{x}'||^{\mathrm{T}} H^{-1} ||\boldsymbol{x} - \boldsymbol{x}'||\right), \tag{C.5}$$

where α^2 is the variance of f. $H^{\boldsymbol{x}} \in \mathbb{R}^{d_x}$ is a diagonal matrix with positive elements, adjusting the scale of each dimension of $\boldsymbol{x}$ respectively.

Another is the Matern family. For a distance d between two points:

$$k(d) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\frac{d}{\rho}\right)^{\nu} K_{\nu}\left(\sqrt{2\nu}\frac{d}{\rho}\right), \tag{C.6}$$

where Γ is the Gamma function, K_ν is the second kind of modified Bessel function, and ρ and ν are non-negative covariance parameters. As $\nu \to \infty$, k resembles the square exponential function. Common values of ν for machine learning purposes are 1.5 and 2.5.

Appendix D

Reinforcement Learning Methodologies

D.1 Q-learning (Sutton and Barto 1998)

Here we define the Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \alpha, \gamma)$. $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions, and $\mathcal{R}(s, s', a)$ is the reward from state s to state s' under the action a. $\alpha \in (0, 1)$ is a learning rate, and $\gamma \in (0, 1)$ is a discount factor. Q-learning learns this MDP through recursive updates of the policy $\mathcal{Q}$:

$$\mathcal{Q}(\mathcal{S}_k, \mathcal{A}_k) = \mathcal{Q}(\mathcal{S}_k, \mathcal{A}_k) + \alpha\Big(\mathcal{R}_{k+1} + \gamma \times \arg\max_{a \in \mathcal{A}} \mathcal{Q}(\mathcal{S}_{k+1}, a) - \mathcal{Q}(\mathcal{S}_k, \mathcal{A}_k)\Big) \tag{D.1}$$

D.2 Dynamic Policy Programming (Azar et al. 2012)

Stochastic optimal control learns a MDP defined by a 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$. $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions, $\mathcal{T}^a_{ss'}$ is the transition probability from state s to state s' under the action a, $r^a_{ss'} = \mathcal{R}(s, s', a)$ is the reward from state s to state s' under the action a. We define $\mathcal{S}$ as the local window around the PMD discretized into a grid-world for path planning, and $\mathcal{A}$ as a list of 9 actions for moving from a grid cell to any of its immediate neighbors or remaining in place. $\gamma \in (0, 1)$ is the discount factor. The policy $\pi(a|s)$ denotes the probability of taking the action a under the state s. The value function $V_\pi(s) = \lim_{k \to \infty} \mathbb{E}_\pi\Big[\sum_{k=1}^{K} \gamma^{k-1} r_{s_{t+k}} | s_t = s\Big]$ is the expected return when the process starts in s and the decision maker follows the policy π. The solution of MDP is an optimal policy π^* that attains the maximum expected reward:

$$\pi^* = \arg\max_{\pi} \sum_{\substack{a\in\mathcal{A}\\ s'\in\mathcal{S}}} \pi(a|s)\mathcal{T}^a_{ss'}\Big(r^a_{ss'} + \gamma V^*(s')\Big), \forall s \in \mathcal{S}. \tag{D.2}$$

DPP builds a new value function by adding the Kullback-Leibler divergence to the reward function $\mathcal{R}$ as a penalty. The Kullback-Leibler divergence between the policy π and the baseline policy $\bar{\pi}$, and the new value function V are defined as:

$$g^{\pi}_{\bar{\pi}}(s) = KL(\pi\|\bar{\pi}) = \sum_{a\in\mathcal{A}} \pi(a|s)\log\Big(\frac{\pi(a|s)}{\bar{\pi}(a|s)}\Big), \tag{D.3}$$

$$V_{\bar{\pi}}(s) \triangleq \lim_{k\to\infty} \mathbb{E}_{\pi}\Bigg[\sum_{k=1}^{K}\gamma^{k-1}\Big(r_{s_{t+k}} - \frac{1}{\eta}g^{\pi}_{\bar{\pi}}(s_{t+k-1})\Big)\Bigg|s_t = s\Bigg] \tag{D.4}$$

where $\mathbb{E}_{\pi}$ denotes the expectation over transition model $\mathcal{T}$ and the current policy π. $\eta \in (0,1)$ is a constant that controls the Kullback-Leibler divergence term.

The action preferences function (Sutton and Barto 1998) for all state action pairs $(s,a) \in \mathcal{S}\times\mathcal{A}$ in the k-th iteration are defined as $\Psi_k(s,a) = \frac{1}{\eta}\log\bar{\pi}^{k-1}(a|s) + \sum_{s'\in\mathcal{S}}\mathcal{T}^a_{ss'}\Big(r^{\,a}_{ss'} + \gamma V^{k-1}_{\bar{\pi}}(s')\Big)$. It represents the closed form of the optimal policy π^* following:

$$V^{k+1}_{\bar{\pi}}(s) = \frac{1}{\eta}\log\sum_{a\in\mathcal{A}}\exp(\eta\Psi_k(s,a)), \tag{D.5}$$

$$\bar{\pi}^{k+1}(a) = \frac{\exp\Big(\eta\Psi_k(s,a)\Big)}{\sum_{a'\in\mathcal{A}}\exp\Big(\eta\Psi_k(s,a')\Big)}. \tag{D.6}$$

The optimal action preferences function determines DPP's optimal policy according to Eq. D.6. The update recursion of Ψ follows:

$$\Psi_{k+1}(s,a)=\Psi_k(s,a)-\mathcal{M}_\eta\Psi_k(s)+\sum_{s'\in\mathcal{S}}\mathcal{T}^a_{ss}\Big(r^a_{ss'}+\gamma\mathcal{M}_\eta\Psi_k(s')\Big) \tag{D.7}$$

where $\mathcal{M}_\eta \Psi_k(s)$ is the the Boltzmann soft-max operator:

$$\mathcal{M}_\eta \Psi(s) = \sum_{a \in \mathcal{A}} \frac{\exp\Big(\eta \Psi(s,a)\Big)\Psi(s,a)}{\sum_{a' \in \mathcal{A}} \exp\Big(\eta \Psi(s,a')\Big)}. \tag{D.8}$$

Following Eq. D.7, DPP updates the action preferences function Ψ to iteratively improve its value function to the optimal one while considering the smoothness in the policy update (controlled by η).

Appendix E

Miscellaneous Algorithms

E.1 Pure Pursuit (Coulter 1990)

Pure Pursuit is a fast and simple control algorithm, yielding both a local "control point" and the radius of curvature (ratio of linear to angular velocity) required to reach it.

```
input : path points $P$, robot pose $o$, lookahead distance $l$, linear velocity $v$
$p \leftarrow$ closest point in $P$ that is $l$ ahead of $o$
$g \leftarrow p$ w.r.t. $o$
$r \leftarrow$ required arc radius to reach $g$
$\omega = v/r$
output: local control point $g$, angular velocity $\omega$
```

Algorithm 7: Pure Pursuit path tracking.

E.2 Dynamic Window Approach (Fox et al. 1997)

DWA considers a range of "rollout" linear and angular velocity pairs (Fig. E.1) and determines the most suitable pair from a reward function considering both the immediate environment and a local point objective in the robot's reference frame. Here $V = (v, \omega)$ denotes the robot's "twist", comprised of its linear and angular velocities.

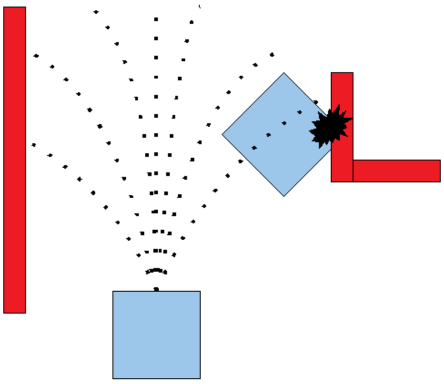


FIGURE E.1: Trajectory rollout of linear and angular velocity pairs. From ROS Wiki (2017).

```
input  : platform twist $V$ and limits $V_{max}$, platform acceleration increments $V'$ and limits $V'_{max}$,
         rollout duration $t$, local occupancy grid $z$, local objective $g$, cost function $\mathcal{R}$
$\mathbf{V} \leftarrow$ candidate twists based on $V_{robot}$, $V_{max}$, $V'$, and $V'_{max}$
$V_{best} = V$, $r_{best} = \mathcal{R}(V_{best}, t, z, g)$
for $V^* = \mathbf{V}_1, \mathbf{V}_2, ..., \mathbf{V}_{|\mathbf{V}|}$ do
    $r = \mathcal{R}(V^*, t, g, z)$
    if $r < r_{best}$ then
        $V_{best} = V^*$, $r_{best} = r$
    end
end
output: optimal twist $V_{best}$
```

Algorithm 8: Dynamic Window Approach local motion controller.

E.3 Steering Entropy (Nakayama et al. 1999)

Originally devised to assess user workload in driving exercises, steering entropy has also been utilized to measure other attributes such as impairment (Kersloot et al. 2003) or task effectiveness (Yang et al. 2017). For each input steering angle u_t in a time series U, an input prediction error e_t is taken as the difference between u_t and a second order Taylor approximation $\hat{u}_t$:

$$\hat{u}_t = u_{t-1} + (u_{t-1} - u_{t-2}) + \frac{1}{2}\Big((u_{t-1} - u_{t-2}) - (u_{t-2} - u_{t-3})\Big). \tag{E.1}$$

A frequency distribution P of all e is then discretized into 9 bins (Fig. E.2), of which the Shannon entropy (Shannon 2001) is taken as the steering entropy of the user over the entirety of U:

$$H = \sum_i -P_i \times log_9 P_i. \tag{E.2}$$

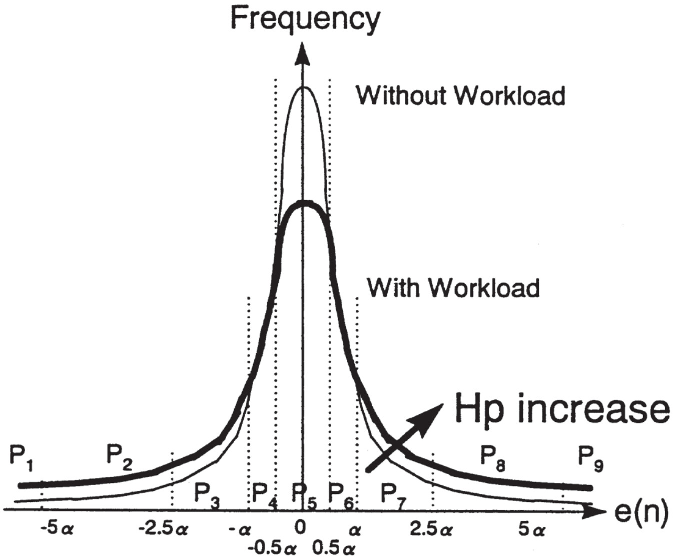


FIGURE E.2: Steering entropy binning, where α is the 90 percentile value. From Nakayama et al. (1999).

E.4 Dynamic Time Warping (Sakoe and Chiba 1978)

For two time series S and T respectively containing m and n elements, DTW returns a $m \times n$ cost matrix allowing for the alignment of corresponding points. The basic variant of DTW is shown in Algorithm 9; a common distance measure is $d(x,y) = |x - y|$.

```
input  : $S = s_1,...,s_m$, $T = t_1,...,t_n$
$D = 0_{m,n}$; set first row and column of $D$ to $\infty$
for $i = 1; i <= m; i = i + 1$ do
    for $j = 1; j <= n; j = j + 1$ do
        $D_{ij} = d(s_i, t_j) + \inf(D_{i-1,j}, D_{i,j-1}, D_{i-1,j-1})$
    end
end
output: cost matrix $D$
```

Algorithm 9: Basic Dynamic Time Warping algorithm.

Bibliography

Abadi M et al (2015) TensorFlow: Large-scale machine learning on heterogeneous systems. `https://www.tensorflow.org/`. Accessed: 14/03/2017. [p. 61]

Argall BD, Chernova S, Veloso M and Browning B (2009) A survey of robot learning from demonstration. *Robot. Auton. Syst.* 57(5): 469–483. [p. 2]

Awais M and Henrich D (2013) Human-robot interaction in an unknown human intention scenario. In: *Frontiers of Information Technology (FIT), 2013 11th International Conference on*. pp. 89–94. [p. 76]

Azar MG, Gómez V and Kappen HJ (2012) Dynamic policy programming. *The Journal of Machine Learning Research* 13(1): 3207–3245. [pp. xi and 119]

Badrinarayanan V, Kendall A and Cipolla R (2015) Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR* . [p. 57]

Ballesteros J, Urdiales C, Velasco ABM and Ramos-Jimenez G (2017) A biomimetical dynamic window approach to navigation for collaborative control. *IEEE Transactions on Human-Machine Systems* PP(99): 1–11. [pp. 31, 32, 33, 34, and 38]

Ben Amor H, Neumann G, Kamthe S, Kroemer O and Peters J (2014) Interaction primitives for human-robot cooperation tasks. In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, pp. 2831–2837. [pp. x, 5, 73, 80, and 101]

Borenstein J and Koren Y (1991) The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation* 7(3): 278–288. [p. 14]

Candela JQ and Rasmussen CE (2005) A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research* 6: 1939–1959. [p. 116]

Carlson T and del R Millan J (2013) Brain-controlled wheelchairs: A robotic architecture. *IEEE Robotics Automation Magazine* 20(1): 65–73. [p. 103]

Carlson T and Demiris Y (2008) Human-wheelchair collaboration through prediction of intention and adaptive assistance. In: *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. pp. 3926–3931. [pp. 16 and 17]

Carlson T and Demiris Y (2012) Collaborative control for a robotic wheelchair: Evaluation of performance, attention, and workload. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 42(3): 876–888. [p. 17]

Chen L, Wang S, Hu H and McDonald-Maier K (2012) Bezier curve based trajectory planning for an intelligent wheelchair to pass a doorway. In: *Proceedings of 2012 UKACC International Conference on Control*. pp. 339–344. [p. 33]

Claesen M and Moor BD (2015) Hyperparameter search in machine learning. *CoRR* abs/1502.02127. [p. 65]

Coulter R (1990) Implementation of the pure pursuit path tracking algorithm. Technical report, Carnegie Mellon University, Pittsburgh. [pp. xi and 123]

Cui Y, Poon J, Matsubara T, Miro JV, Sugimoto K and Yamazaki K (2016a) Environment-adaptive interaction primitives for human-robot motor skill learning. In: *Humanoid Robots, 2016 IEEE-RAS International Conference on*. pp. 711–717. [pp. 10 and 74]

Cui Y, Poon J, Matsubara T, Miro JV, Sugimoto K and Yamazaki K (2018) Environment-adaptive interaction primitives through visual context for human-robot motor skill learning. *Autonomous Robots* Accepted. [pp. 10 and 74]

Cui Y, Poon J, Miro JV, Matsubara T and Sugimoto K (2016b) Optimal control approach for active local driving assistance in mobility aids. In: *The 34th annual conference of the Robotics Society of Japan*. [pp. 10 and 29]

Deisenroth MP, Huber MF and Hanebeck UD (2009) Analytic moment-based gaussian process filtering. In: *Proceedings of the 26th Annual International Conference on Machine Learning (ICML), Montreal, Quebec, Canada*. pp. 225–232. [p. 21]

Demeester E, Huntemann A, Vanhooydonck D, Vanacker G, Degeest A, Van Brussel H and Nuttin M (2006) Bayesian estimation of wheelchair driver intents: Modeling intents as geometric paths tracked by the driver. In: *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. pp. 5775–5780. [p. 17]

Derry M and Argall B (2013) Automated doorway detection for assistive shared-control wheelchairs. In: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. pp. 1254–1259. [p. 29]

Devigne L, Narayanan VK, Pasteau F and Babel M (2016) Low complex sensor-based shared control for power wheelchair navigation. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 5434–5439. [p. 15]

Dewolf T (2013) Dynamic movement primitives. `https://studywolf.wordpress.com/2013/11/16/dynamic-movement-primitives-part-1-the-basics/`. Accessed: 20/08/2017. [p. 76]

Doumanoglou A, Kargakos A, Kim TK and Malassiotis S (2014) Autonomous active recognition and unfolding of clothes using random decision forests and probabilistic planning. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 987–993. [p. 77]

Esquenazi A (2016) Why joystick location matters for power wheelchair users. `http://www.homecaremag.com/news/why-joystick-location-matters-power-wheelchair-users`. Accessed: 22/09/2016. [p. 50]

Finn C, Tan XY, Duan Y, Darrell T, Levine S and Abbeel P (2016) Deep spatial autoencoders for visuomotor learning. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 512–519. [p. 78]

Fitzgerald T, Goel A and Thomaz A (2015) A similarity-based approach to skill transfer. In: *Women in Robotics Workshop at the Robotics: Science and Systems Conference. Rome, Italy*. [p. 76]

Fox D, Burgard W and Thrun S (1997) The dynamic window approach to collision avoidance. *Robotics Automation Magazine, IEEE* 4(1): 23–33. [pp. xi, 39, and 123]

Freytag A, Frohlich B, Rodner E and Denzler J (2012) Efficient semantic segmentation with gaussian processes and histogram intersection kernels. In: *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. pp. 3313–3316. [p. 57]

Funahashi KI (1989) On the approximate realization of continuous mappings by neural networks. *Neural Networks* 2(3): 183 – 192. [pp. xi and 114]

Furtuna AA and Balkcom DJ (2010) Generalizing dubins curves: Minimum-time sequences of body-fixed rotations and translations in the plane. *The International Journal of Robotics Research* 29(6): 703–726. [p. 33]

Gerkey BP, Vaughan RT and Howard A (2003) The player/stage project: Tools for multi-robot and distributed sensor systems. In: *In Proceedings of the 11th International Conference on Advanced Robotics*. pp. 317–323. [pp. 42 and 106]

Goil A, Derry M and Argall BD (2013) Using machine learning to blend human and robot controls for assisted wheelchair navigation. In: *Rehabilitation Robotics (ICORR), 2013 IEEE International Conference on*. pp. 1–6. [pp. 14 and 75]

Huang CM and Mutlu B (2016) Anticipatory robot control for efficient human-robot collaboration. In: *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. pp. 83–90. [p. 75]

Huntemann A, Demeester E, Poorten E, Van Brussel H and De Schutter J (2013) Probabilistic approach to recognize local navigation plans by fusing past driving information with a personalized user model. In: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. pp. 4376–4383. [pp. 31, 32, and 58]

Huntemann A, Demeester E, Vanacker G, Vanhooydonck D, Philips J, Brussel HV and Nuttin M (2007) Bayesian plan recognition and shared control under uncertainty: assisting wheelchair drivers by tracking fine motion paths. In: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 3360–3366. [pp. 17 and 53]

Igel C and Husken M (2000) Improving the rprop learning algorithm. In: *Proc. of the Second Intl. Symposium on Neural Computation*. [p. 114]

Ijspeert AJ, Nakanishi J, Hoffmann H, Pastor P and Schaal S (2013) Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation* 25(2): 328–373. [pp. x and 79]

International Federation of Robotics (2015) World robotics 2015 industrial robots. Technical report, International Federation of Robotics, Germany. [p. 1]

Jiang M, Chen Y, Zheng W, Wu H and Cheng L (2016) Mobile robot path planning based on dynamic movement primitives. In: *2016 IEEE International Conference on Information and Automation (ICIA)*. pp. 980–985. [p. 33]

Kersloot T, Flint A and Parkes A (2003) Steering entropy as a measure of impairment. Technical report, TRL Limited, United Kingdom. [p. 124]

Kohlbrecher S, Meyer J, Stryk OV and Klingauf U (2011) A flexible and scalable slam system with full 3d motion estimation. In: *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE. [p. 47]

Kormushev P, Calinon S and Caldwell DG (2010) Robot motor skill coordination with em-based reinforcement learning. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 3232–3237. [p. 75]

Krizhevsky A, Sutskever I and Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L and Weinberger KQ (eds.) *Advances in Neural Information Processing Systems 25*. pp. 1097–1105. [p. 57]

Kronander K and Billard A (2014) Learning compliant manipulation through kinesthetic and tactile human-robot interaction. *IEEE Transactions on Haptics* 7(3): 367–380. [p. 75]

Kruse D, Radke RJ and Wen JT (2015) Collaborative human-robot manipulation of highly deformable materials. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 3782–3787. [p. 77]

Ku HH (1966) Notes on the use of propagation of error formulas. *Journal of Research of the National Bureau of Standards* 70C(4): 263–273. [p. 63]

Lankenau A, Meyer O and Krieg-Bruckner B (1998) Safety in robotics: the bremen autonomous wheelchair. In: *Advanced Motion Control, 1998. AMC '98-Coimbra., 1998 5th International Workshop on*. pp. 524–529. doi: 10.1109/AMC.1998.743591. [p. 14]

Lasota PA, Fond T and Shah JA (2014) A survey of methods for safe human-robot interaction. *Foundations and Trends in Robotics* 5(4): 261–349. [p. 2]

Lawitzky M, Mortl A and Hirche S (2010) Load sharing in human-robot cooperative manipulation. In: *19th International Symposium in Robot and Human Interactive Communication*. pp. 185–191. [p. 77]

Lecun Y, Bottou L, Bengio Y and Haffner P (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11): 2278–2324. [pp. xi and 115]

Li Q, Chen W and Wang J (2011) Dynamic shared control for human-wheelchair cooperation. In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. pp. 4278–4283. [pp. 14 and 52]

Li X, Zhao L, Wei L, Yang M, Wu F, Zhuang Y, Ling H and Wang J (2016) Deepsaliency: Multi-task deep neural network model for salient object detection. *IEEE Transactions on Image Processing* 25(8): 3919–3930. [p. 57]

Liu Z, Hu F, Luo D and Wu X (2014) Visual gesture recognition for human robot interaction using dynamic movement primitives. In: *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. pp. 2094–2100. [p. 76]

Maeda G, Ewerton M, Neumann G, Lioutikov R and Peters J (2017) Phase estimation for fast action recognition and trajectory generation in human-robot collaboration. *The International Journal of Robotics Research* . [pp. 77 and 104]

Mandery C, Borras J, Jochner M and Asfour T (2016) Using language models to generate whole-body multi-contact motions. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 5411–5418. [p. 76]

Marder-Eppstein E, Berger E, Foote T, Gerkey B and Konolige K (2010) The office marathon: Robust navigation in an indoor office environment. In: *International Conference on Robotics and Automation*. [p. 42]

MathWorks (2017) Radial basis neural networks. `https://mathworks.com/help/nnet/ug/radial-basis-neural-networks.html`. Accessed: 14/10/2017. [p. 113]

Matsubara T, Hyon SH and Morimoto J (2011) Learning parametric dynamic movement primitives from multiple demonstrations. *Neural Networks* 24(5): 493–500. [pp. 76 and 84]

Matsubara T, Miro JV, Tanaka D, Poon J and Sugimoto K (2015) Sequential intention estimation of a mobility aid user for intelligent navigational assistance. In: *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. pp. 444–449. [pp. 10 and 13]

Matte (2013) arctan with two arguments. `http://planetmath.org/sites/default/files/texpdf/37105.pdf`. Accessed: 16/12/2017. [p. 63]

McCrindle Research (2014) Demand vs supply: the aged care puzzle. Technical report, McCrindle Research, Australia. [p. 1]

McGarry S, Moir L and Girdler S (2012) The smart wheelchair: is it an appropriate mobility training tool for children with physical disabilities? *Disability and Rehabilitation: Assistive Technology* 7(5): 372–380. [p. 48]

Meier F, Theodorou E, Stulp F and Schaal S (2011) Movement segmentation using a primitive library. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 3407–3412. [p. 76]

Mülling K, Kober J, Kroemer O and Peters J (2013) Learning to select and generalize striking movements in robot table tennis. *The International Journal of Robotics Research* 32(3): 263–279. [p. 76]

Nakayama O, Futami T, Nakamura T and Boer ER (1999) Development of a steering entropy method for evaluating driver workload. In: *SAE Technical Paper*. SAE International. [pp. xi, 67, 124, and 125]

Narayanan VK, Spalanzani A and Babel M (2016) A semi-autonomous framework for human-aware and user intention driven wheelchair mobility assistance. In: *Intelligent Robots and Systems, 2016 IEEE/RSJ International Conference on*. pp. 5775–5780. [p. 31]

Nielson M (2017) How the backpropagation algorithm works. `http://neuralnetworksanddeeplearning.com/chap2.html`. Accessed: 11/09/2017. [p. 114]

Orr M (1996) Introduction to radial basis function networks. Technical report, University of Edinburgh, Scotland. [pp. xi and 113]

Ozhiganov I (2016) Object detection using fully convolutional neural networks. `http://rnd.azoft.com/object-detection-fully-convolutional-neural-networks/`. Accessed: 22/09/2017. [p. 115]

Park DH, Hoffmann H, Pastor P and Schaal S (2008) Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. In: *Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots*. pp. 91–98. [p. 33]

Pashenkova E, Rish I and Dechter R (1996) Value iteration and policy iteration algorithms for markov decision problem. Technical report, University of California at Irvine. [p. 34]

Patel M, Miro JV and Dissanayake G (2014) A probabilistic approach to learn activities of daily living of a mobility aid device user. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA Hong Kong, China*. pp. 969–974. [pp. 16 and 17]

Pinto L and Gupta A (2015) Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. *CoRR* abs/1509.06825. URL `http://arxiv.org/abs/1509.06825`. [p. 78]

Poon J, Cui Y, Miro JV and Matsubara T (2018a) Learning from demonstration for locally assistive mobility aids. *Advanced Robotics* Under review. [pp. 10 and 29]

Poon J, Cui Y, Miro JV and Matsubara T (2018b) Learning mobility aid assistance via decoupled observations. In: *2018 IEEE International Conference on Intelligent Robots and Systems (IROS)*. Under review. [pp. 10 and 55]

Poon J, Cui Y, Miro JV, Matsubara T and Sugimoto K (2017) Local driving assistance from demonstration for mobility aids. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 5935–5941. [pp. 10 and 29]

Poon J, Miro JV and Black R (2015) A passive estimator of functional degradation in power mobility device users. In: *2015 IEEE International Conference on Rehabilitation Robotics (ICORR)*. pp. 997–1002. [pp. 10 and 13]

Prada M, Remazeilles A, Koene A and Endo S (2014) Implementation and experimental validation of dynamic movement primitives for object handover. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 2146–2153. [p. 76]

Quigley M, Gerkey B, Conley K, Faust J, Foote T, Leibs J, Berger E, Wheeler R and Ng A (2009) Ros: an open-source robot operating system. In: *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*. [p. 43]

Rabhi Y, Mrabet M, Fnaiech F and Gorce P (2013) A feedforward neural network wheelchair driving joystick. In: *2013 International Conference on Electrical Engineering and Software Applications*. pp. 1–6. [p. 58]

Rasmussen CE and Williams CKI (2005) *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press. [pp. xi and 116]

Redmon J and Farhadi A (2016) Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242* . [p. 90]

Rethink Robotics (2016) Sixty seconds with sawyer: Let's rethink robot programming. `https://www.youtube.com/watch?v=NwZvmD2McVI`. Accessed: 21/08/2017. [p. 75]

ROS Wiki (2015) gmapping. `http://wiki.ros.org/gmapping`. Accessed: 24/10/2017. [p. 18]

ROS Wiki (2017) dwa_local_planner. `http://wiki.ros.org/dwa_local_planner`. Accessed: 23/09/2017. [p. 124]

Rozo L, Calinon S, Caldwell D, Jimenez P and Torras C (2013) Learning collaborative impedance-based robot behaviors. In: *AAAI Conference on Artificial Intelligence*. [p. 75]

Saal HP, Ting J and Vijayakumar S (2010) Active estimation of object dynamics parameters with tactile sensors. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and System (IROS), Taipei, Taiwan*. pp. 916–921. [p. 21]

Sakoe H and Chiba S (1978) Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing* 26(1): 43–49. [pp. xi and 125]

Shannon CE (2001) A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.* 5(1): 3–55. [p. 124]

Sheng W, Thobbi A and Gu Y (2015) An integrated framework for human-robot collaborative manipulation. *IEEE Transactions on Cybernetics* 45(10): 2030–2041. [p. 77]

Simonyan K and Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556. [pp. 57 and 90]

Simpson RC and Levine SP (1999) Automatic adaptation in the navchair assistive wheelchair navigation system. *IEEE Transactions on Rehabilitation Engineering* 7(4): 452–463. [p. 14]

Simpson RC, Levine SP, Bell DA, Jaros LA, Koren Y and Borenstein J (1998) *NavChair: An assistive wheelchair navigation system with automatic adaptation*. pp. 235–255. [p. 14]

Smola AJ and Bartlett PL (2001) Sparse greedy Gaussian process regression. In: *Advances in Neural Information Processing Systems 13*. MIT Press, pp. 619–625. [p. 23]

Snelson E and Ghahramani Z (2005) Sparse Gaussian processes using pseudo-inputs. In: *Advances in Neural Information Processing Systems 18*. pp. 1257–1264. [p. 116]

Soh H and Demiris Y (2015) Learning assistance by demonstration: Smart mobility with shared control and paired haptic controllers. *Journal of Human-Robot Interaction* 4(3): 76–100. [p. 75]

Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R (2014) Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15(1): 1929–1958. [p. 114]

Standards Australia (2009) As 1428.1-2009-design for access and mobility-general requirements for access-new building work. Technical report. [p. 36]

Storn R and Price K (1997) Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization* 11(4): 341–359. [p. 24]

Sutton RS and Barto AG (1998) *Reinforcement learning: An introduction*. MIT press Cambridge. [pp. xi, 75, 119, and 120]

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V and Rabinovich A (2015) Going deeper with convolutions. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 1–9. [p. 57]

Taha T, Miro JV and Dissanayake G (2011) A POMDP framework for modelling human interaction with assistive robots. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China*. pp. 544–549. [p. 17]

Tan H, Du Q and Wu N (2012) A framework for cognitive robots to learn behaviors through imitation and interaction with humans. In: *2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*. pp. 235–238. [p. 76]

Tegart (2010) Smart technology for healthy ageing. Technical report, ATSE. [p. 1]

Ude A, Gams A, Asfour T and Morimoto J (2010) Task-specific generalization of discrete and periodic dynamic movement primitives. *IEEE Transactions on Robotics* 26(5): 800–815. [p. 76]

United Nations (2015) World population ageing 2015. Technical report, United Nations Department of Economic and Social Affairs (Population Division), New York. [p. 1]

Urbano J, Terashima K, Miyoshi T and Kitagawa H (2004) Impedance control for safety and comfortable navigation of an omni-directional mobile wheelchair. In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 2. pp. 1902–1907 vol.2. [p. 14]

Urdiales C, Fdez-Carmona M, Peinado G and Sandoval F (2013a) Metrics and benchmarking for assisted wheelchair navigation. In: *Proc. of IEEE International Conference on Rehabilitation Robotics (ICORR)*. [pp. 51 and 67]

Urdiales C, Perez EJ, Peinado G, Fdez-Carmona M, Peula JM, Annicchiarico R, Sandoval F and Caltagirone C (2013b) On the construction of a skill-based wheelchair navigation profile. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 21(6): 917–927. [p. 15]

Urdiales C, Peula JM, Fdez-Carmona M, Barrué C, Pérez EJ, Sánchez-Tato I, del Toro JC, Galluppi F, Cortés U, Annichiaricco R, Caltagirone C and Sandoval F (2011) A new multi-criteria optimization strategy for shared control inÂăwheelchair assisted navigation. *Autonomous Robots* 30(2): 179–197. [p. 14]

VanderPlas J (2013) Neural network diagram. `http://www.astroml.org/book_figures/appendix/fig_neural_network.html`. Accessed: 22/06/2017. [p. 114]

Vanhooydonck D, Demeester E, Hüntemann A, Philips J, Vanacker G, Van Brussel H and Nuttin M (2010) Adaptable navigational assistance for intelligent wheelchairs by means of an implicit personalized user model. *Robot. Auton. Syst.* 58(8). [p. 43]

Varma MKS, Rao NKK, Raju KK and Varma GPS (2016) Pixel-based classification using support vector machine classifier. In: *2016 IEEE 6th International Conference on Advanced Computing (IACC)*. pp. 51–55. [p. 57]

Vircikova M, Smolar P and Sincak P (2012) Current trends in human-robot interaction:towards collaborative & friendly machines. In: *12th Scientific Conference of Young Researchers*. [p. 1]

Wang S, Chen L, Hu H and McDonald-Maier K (2012) Doorway passing of an intelligent wheelchair by dynamically generating bezier curve trajectory. In: *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. pp. 1206–1211. [p. 33]

Yang X, Sreenath K and Michael N (2017) A framework for efficient teleoperation via online adaptation. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 5948–5953. [pp. 32, 38, and 124]

Yin X and Chen Q (2014) Learning nonlinear dynamical system for movement primitives. In: *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. pp. 3761–3766. [p. 33]