Structured modeling for robot decision-making

Johanna Hansen

School of Computer Science

McGill University, Montreal

August 2024

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of

Doctor of Philosophy

Abstract

In this thesis, we demonstrate the use of physics-informed inductive biases for a wide variety of learned models that control robot behaviour. In addition to reducing training time, models trained with a priori knowledge demonstrate better generalization and contextualization based on observation compared to pure physics-only or learning-only methods. These performance qualities are especially important for robots operating in uncertain and dynamic environments that require data that is difficult or costly to model in training.

This document presents a series of practical improvements in robotic perception and behaviour learning centred around building high-quality models of the robot's environment, its sensors, and the robot itself. We show the empirical effectiveness of incorporating structural bias for various robot tasks, including learning conditional computation for an extraterrestrial machine vision task, deploying a robot mapping team that relies on physics for propulsion, and robot arms that learn controllers for manipulation.

Abrégé

Dans cette thèse, nous démontrons l'utilisation de la physique comme biais inductif pour les modèles d'apprentissage contrôlant le comportement des robots. Nous montrons que les robots ayant accès à des priors informés par la physique pendant l'apprentissage nécessitent moins d'interactions avec l'environnement pendant la formation et présentent une meilleure généralisation par rapport aux méthodes purement physiques ou purement basées sur l'apprentissage. Ces qualités de performance sont particulièrement importantes pour les robots opérant dans des environnements incertains et dynamiques, difficiles ou coûteux à modéliser en formation.

Ce document présente une série d'améliorations pratiques dans la perception et l'apprentissage comportemental des robots, centrées sur la construction de modèles de haute qualité de l'environnement du robot, de ses capteurs et du robot lui-même. Nous démontrons l'efficacité empirique de l'intégration de biais structurels pour une variété de tâches robotiques, notamment l'apprentissage du calcul conditionnel pour une tâche de vision machine extraterrestre, le déploiement d'une équipe de cartographie robotique s'appuyant sur la physique pour la propulsion, et des bras robotiques qui apprennent des contrôleurs pour la manipulation.

Acknowledgements

I want to express my sincere gratitude to my supervisor, Prof. Gregory Dudek for welcoming me into the Mobile Robotics Lab (MRL) and providing me with his faith, encouragement, and the incredible opportunity of academic freedom. His tireless creativity and enthusiasm in both research and life are the template by which I hope to live. In addition to changing my own career trajectory through his positive force of nature, he has also welcomed a horde of kind, compassionate, and intelligent folks into the "MRL Family", many of which I am grateful to now call friends. I want to thank the "old guard" Ph.D.s (Anqi, Florian, Juan, Jimmy, Sandeep, Malika, and Travis) who were always a source of leadership and technical knowledge as I navigated field trials as a new graduate student. I especially thank Sandeep for his friendship and for making our boat field deployments such a positive experience. I would also like to thank my cohort (Nikhil, Andrew, Wei-Di, Lucas, Monica, Peter, Karim, Ed, Scott, Arnold, Abhisek, Bobak, Sahand, Chelsea, Gaspard, Jean-Gabriel, Jean-Francois, Khalil, Xiru and Yi Tian) for their friendship, collaboration, and technical discussions.

The wonderful technical staff at MRL enabled most of the experiments I conducted during my graduate work. These experiments would not have been possible without the support of Nik, Jeremy, or Ian and their dedicated work toward our research. Isabelle, Laurena, and Nick were always there to help me navigate the university bureaucracy.

I would also like to thank the lab alumni. Ioannis never fails to lend his hardware support or never-ending enthusiasm for marine robotics. Junead, thank you for your incredible role model as a mentor and your professional support. Yogesh for his research

in machine learning and marine robotics and for connecting me from WHOI to Greg and MRL. Dave has been such an enthusiastic and talented collaborator. I am so very thankful that he has boomeranged back from alumni to become an integral MRL professor.

I thank Joelle for agreeing to be my co-advisor halfway through my graduate program and graciously providing exceptional mentorship on demand, especially through my (slow) thesis-writing process. Her incredible talent for giving great feedback means that I always left our meetings with a delightful list of actionable changes.

While working at SwRI, WHOI, JPL, Samsung Research, and Boston Dynamics, I interacted with incredible researchers. Great colleagues make this type of technical research fun and continue to motivate me.

I am incredibly lucky to have gone through graduate school at the same time as my best friend and life partner. Thanks, Kyle for keeping me up late discussing machine learning problems for the past decade, alas, I think I have finally convinced you that robots are more complex than handwriting.

Finally, I am grateful to the funding agencies who have supported me throughout my Ph.D., including Samsung Research and the Jet Propulsion Lab, and especially to the Government of Canada for providing me with funding through NCFRN / NCRN. Finally, I would like to sincerely thank my dissertation committee members, Kaleem and Geoff, for their thoughtful guidance and review throughout my thesis review process.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ADCP ..... Acoustic Doppler Current Profiler

ASV ..... Autonomous Surface Vehicle

AUV ..... Autonomous Underwater Vehicle

DDPG ..... Deep Deterministic Policy Gradients

DH ..... Denavit-Hartenburg

DoF ..... Degrees of Freedom

DR ..... Domain Randomization

GP ..... Gaussian Process

GPS ..... Global Positioning System

NMS ..... Non-Maximum Suppression

OSC ..... Operational Space Controller

RL ..... Reinforcement Learning

ROMS ..... Regional Ocean Modeling Systems

Part I

Overview

Chapter 1

Introduction

1.1 Motivation

Perhaps the most well-known scientific robot is the Mars Curiosity Rover. Curiosity is a car-sized wheeled robot exploring the Gale Crater on Mars since 2012, achieving its scientific mission of mapping the region with acutely bandwidth-limited human supervision [Webster, 2013]. Given its lonely locale, Curiosity operates semi-autonomously. Scientists provide high-level directions that trade-off their research goals with what they anticipate the robot can accomplish.

The robot's planner orchestrates short-horizon obstacle avoidance, which is eventually interpreted down to low-level control of wheel propulsion and steering. The 225 million kilometres distance between Earth and Mars results in a 20-minute one-way communication delay between each command, meaning engineers back on Earth are rarely informed on the full state of the actuators, cameras, and scientific instruments needed to make real-time decisions about navigation and sampling, highlighting the importance of autonomy in robots. Despite its ability to operate without hands-on oversight, Curiosity still needs a fair amount of human input to balance a particular scientific mission with its own safety. This expert control by humans comes at a cost. Supervision is often tedious and expensive, and most robots deployed in natural environments still lack the sophis-

tication to perform complex tasks without instruction from human experts. Other scientific sampling robots, such as those depicted in Figure 1.1, have been deployed to harsh environments to gather data for scientific discovery and monitoring. These robots are equipped with specialized sensors that collect data as the robot travels to various parts of a survey region, collecting *in-situ* (local) observations of a phenomenon that is changing over a spatial and/or temporal scale. In Fig 1.1a, we see an example of Curiosity's semi-autonomous exploration. Scientists back on earth choose specified waypoints of interest (marked in green in the figure), and Curiosity chooses a safe path through the waypoints [Webster, 2013]. In Fig 1.1b, the Icefin robot [Meister et al., 2018] was used to capture novel data from under the Thwaites Glacier in Antarctica that have radically improved scientific understanding of underwater glacial melt rate, having important implications for climate modeling [Schmidt et al., 2023]. Robots working in under-ice environments require advanced localization capabilities to accurately map their sensor observations onto a world map that can be used for science. In Figure 1.1c, an unmanned aircraft is adaptively teleoperated by human experts to collect data during Hurricane Michael [JPL, 2016]. The image depicts the temperature of the atmosphere as collected by the drone overlaid on ground-based radar observation of the storm.

Robots are physical agents that interact with the world by observing their environment via sensors and carrying out actions based on these observations that enact physical change, such as movement. In addition to sensing used to inform navigation, scientific robots often require sensors that are instead utilized for gathering data for some downstream scientific task. When mobile robots are tasked to gather samples to provide insight into some physical system (such as the underside of a glacier), attaining more distinct and quality samples will generally improve understanding of the underlying data distribution. Although the distinct data observations gathered by robots are typically much less than the cost of mapping the same region with humans, robots gathering new data points still incur a cost. In the case of environmental sampling, the cost associated with collecting in-situ samples is often dominated by the battery energy, time, or risk required

to travel to disparate locations in a survey region. Inspired by this observation, this thesis specializes in developing practical systems that optimize cost-to-performance trade-offs in robot behaviour using informed priors.

Despite their accomplishments and sophistication, the "autonomous" robots in Figure 1.1 each require an entire team of professional engineers and programmers to help plan and prioritize their activity [Silver, 2010]. Though *truly* autonomous robots have the opportunity to fill important holes in scientific exploration and data acquisition, at this time, most autonomous scientific mobile vehicles which need to interact with their environment only perform tasks under the watchful eye of experts. Exploration robots that require such intense supervision tend only to fill roles that are much too costly, tedious, and/or dangerous for humans.

Outside of scientific robots, in recent years, we have seen a large increase in robots deployed commercially, such as those that handle boxes in warehouses or control row-driving tractors. In these settings, the robots interact with domain experts rather than specially trained engineers, with the key difference being that these machines are usually

(a) Mars mapping **(b)** Imaging under a glacier **(c)** Data inside a hurricane

Figure 1.1: This panel depicts the track of several scientific sampling robots. Starting from the left, we see the track of Curiosity [Webster, 2013] on Mars as guided by a semi-autonomous navigation planner, an image of the underside of the Thwaites glacier captured with the hybrid-remote vehicle, Icefin [Meister et al., 2018], and a collation of hurricane data captured from a remotely operated drone [JPL, 2016]. The robots are roughly ordered left-to-right by the degree to which they rely on human operators for decision-making. Images are from the respective cited publications with permission from the copyright holder.

confined to relatively controlled (structured) environments and specific tasks. Machines that interact with non-engineers in human spaces have historically taken on tasks with low sensitivity to failure and do not require complex reasoning, such as vacuum cleaning or lawn mowing.

1.2 Problem Statement

Structured environments are those in which the space or environment is clearly defined, usually consisting of static geometric shapes and constrained variables that may affect sensor observations (such as consistent lighting and flat flooring with known friction). However, most of the real world is unstructured, with infinite unknown and dynamic variables. A relatively well-structured environment can still seem unstructured to a robot's sensors as features like lighting, friction, or the sensors themselves vary.

Despite the lack of structure in the open world, humans tend to do a good job of operating in these environments. Part of this is because we can adapt and improve our *model* of the world over time. Following architectural cues, humans can walk into a new restaurant and quickly determine where the kitchen is. If that fails, we might utilize an exploratory crutch and follow one of the waitstaff away from a table they have just serviced, expecting them to return to the kitchen for a new tray. Traditional general-purpose planning algorithms like those often employed in mobile robotic systems would solve the same task by laboriously mapping the entire restaurant until they stumble upon an oven or a chef. Although the robot that exhaustively searches for the kitchen is likely to handle edge cases better, one can see how a robot with some built-in intuition can often solve the problem more efficiently. Our intention in this work is to combine the strengths of classical robotic algorithms while enabling robots to adapt to experience and improve their *world model* over time.

Indeed, in the context of scientific sampling robots, we know that *humans* make informed decisions about where to collect data based on their prior knowledge and exper-

tise, preferring to heavily sample unpredictable areas and take fewer measurements from well-known areas. For instance, we can reliably guess that the drone operator in Fig .1.1c chose the vehicle's path based on some prior understanding of hurricanes, the current atmospheric conditions, and the data collection mission. He probably wanted to maintain the vehicle's safety while exploiting the strengths of the sensor (HAMSR) observing the system. Since there were ground radar and satellite observations before beginning the flight, we expect that this data may have raised questions that could only be answered by capturing in-situ data.

1.3 Contribution to Original Knowledge

This thesis contributes to developing robot systems that can operate in unstructured environments by empowering them with structured bias that helps explain themselves and/or the world around them. We argue that with better modelling, the robot's environment appears more structured, thanks to a greater context.

This thesis begins with General (Part I) and Technical (Part II) Introductions and then proceeds into original contributions. In part III of this thesis, we look at two novel systems developed for scientific sampling robots. In Part IV, we introduce physics priors into the structure of neural controllers learned for performing dexterous manipulation. Throughout this thesis, we propose several techniques for improving autonomy, including the following:

1. Chapter 5 focuses on developing a model-based robotic system capable of collecting observations from marine environments at low cost. This system consists of sensors whose sample trajectory is determined primarily by exploiting flow fields for propulsion. This line of work requires accurate modelling of these sensors, called drifters, and led to a series of projects, including:

    - A large-scale collated dataset of paired drifter trajectories and weather phenomena [Hansen et al., 2022b].

- A method for utilizing randomly deployed drifters for improving adaptive sample collection with an autonomous surface vehicle [Manjanna et al., 2017b].
- A method for improving coverage with model-based search over drifter deployment points [Hansen and Dudek, 2018].
- A method for learning to adaptively sample a region with a heterogenous robot team composed of an autonomous surface vehicle and a team of drifters [Hansen et al., 2018].
- Improvements to boat hardware enabled real-world experiments [Huang et al., 2021].

2. In Chapter 7, we design and evaluate a new method for learning to adaptively utilize disparate sensing modalities in the context of visuotactile manipulation [Hansen et al., 2022a].

3. Finally, in Chapter 6, we discuss the method we developed for improving and stabilizing vision-based learning algorithms in multi-link manipulators by leveraging privileged differentiable kinematics in the model structure.

1.4 Contribution of Authors

- Chapters 1, 2, 3, and 4 provide the introduction and background material referenced throughout the thesis. The content and inspiration for these chapters from a variety of sources collected throughout my career, including *Computation Principals of Mobile Robotics* [Dudek and Jenkin, 2010c] and theses of David Silver [Silver, 2010], Florian Shkurti [Shkurti, 2019], Sandeep Manjanna [Manjanna, 2021], Ryan Lowe [Lowe, 2020], and Tegan Maharaj [Maharaj, 2022].

- In Chapter 5, we employ an autonomous surface vehicle (ASV) design known as *Gannet* that was developed and improved as part of a team effort. Sandeep Manjanna, a collaborator and fellow graduate student, led much of the design effort,

purchasing, and assembly. Nikolaos Pateromichelakis, an engineer in the Mobile Robotics Lab, helped design and build the mechanical aspects of the boat. I assisted with wireless and wired communications, power management, and various sensor integration. Jeremy Mallette, a summer engineer in the Mobile Robotics Lab, helped with the field tests and hardware updates. Travis Manderson consulted on the power and battery design. Ian Karp assisted with purchasing and transport. Much of the work that employed physical *drifters* utilizes hardware developed by the lab of Ioannis Rekleitis. The drifter software was developed in collaboration with Alberto Quattrini Li and Sandeep Manjanna.

- The improvements to the Gannet ASV used in Chapter 5 that enable drifter recovery were developed as part of two Mechanical Engineering Capstone Projects at McGill. Profs. Gregory Dudek, David Meger, and I advised a team of students, including Yuying Huang, Yiming Yao, Chris Jing, and Khaled Al Masaid. I developed the problem statement and design concept and led the assembly and field experiments. Yuying handled most of the mechanical modelling, sensor communications, and project organization. Yiming designed a new controller for the sensor and led much of the electrical engineering effort. Chris and Khaled helped study the feasibility of the mechanical design. Sandeep Manjanna and Jeremey Mallette guided ASV deployment. This work is associated with [Huang et al., 2021] in which Yuying, Yiming, and I collaborated on the writing with guidance from David and Sandeep. This project would not be possible without the aforementioned Gannet ASV.

- Chapter 5 also includes dataset and computational work carried out in the final quarter of the project. Khalil Virji contributed by developing a general interface for querying environmental data related to drifter trajectories. Travis Manderson helped develop an ensemble model for evaluating drifter trajectory estimates, and I developed and tuned the models.

- Chapter 7 is derived from the paper "Visuotactile-RL: Learning Multimodal Manipulation Policies with Deep Reinforcement Learning" presented at ICRA 2022. This work was performed while I was an intern at the Samsung AI Research Center in Montreal. I performed much of the code development to perform visuotactile simulation and adapted the RL agent to work with multimodal observations. Dmitriy Rivkin and Francois Hogan were also heavily involved in this effort, especially in speeding up the simulation and running early experiments. Francois also provided expertise in guiding the project from a manipulation and tactile perspective. David Meger, Michael Jenkin, and Greg Dudek provided leadership on architecture design and writing.

- Chapter 6 is based on Kinematic Critic, which was work done in collaboration with Kyle Kastner and Yuying Huang. This work developed from a discussion about forward kinematics while determining the calibration of the Jaco robot manipulator. Kyle connected that the transformation between rigid body links is fully differentiable, similar to a recurrent neural network with fixed parameters. We brainstormed different methods for using this insight and introduced the Kinematic-Critic architecture based on my experience with visual reinforcement learning algorithms such as DrQ. Kyle helped with the architecture design; I wrote the Kinematic Critic and performed the experiments. Yuying Huang found the DH parameters for various robots in the DM-Control suite. Aaron Courville, David Meger, and Gregory Dudek provided supervisory support and editing suggestions. Sahand Rezaei-Shoshtari worked on a version of this work where we attempted to learn the Denavit-Hartenberg parameters online, though this effort has not reached publication maturity.

1.5 Outline

This thesis begins with Part II devoted to introducing robots and models for learning robot behaviours. In Chapter 2, we'll look at how a robot makes observations about the world through sensors and then look at how to collate sensed observations into behaviour in Chapter 3. This background portion of the thesis will conclude by covering machine learning, modelling, and simulation topics in Chpater 4. In subsequent chapters, we investigate co-authored original research with relevant references. In Part III, we look at applications in scientific sampling and design a system to automatically collect marine observations with drifting sensor nodes. Finally, in Part IV, we examine how we can change model architecture via physics-induced priors to learn robot behaviours.

Part II

Technical Background

Chapter 2

Robot Observations

In this chapter, we'll discuss how robots observe and estimate the world around them. The first section focuses on robot *perception*, introducing various physical sensors robots use. In the remaining sections, we discuss *state estimation* methods for interpreting observation and how sensor understanding relates to robot shape and control.

2.1 Robotic Perception

Robots perceive the world around them using various sensors. Sensors are devices that measure an analog physical quantity and convert it into a signal that computers can use. Like mammals who utilize multiple senses in daily life, most robots require a collection of sensors to avoid obstacles, determine where they are in the world, and recognize objects for interaction. Sensor readings include range and visual sensors, which may provide geometric or qualitative information about the environment. They also include absolute and inertial position readings, which tell the robot about its pose. Real-world sensors are often limited in observational accuracy due to noise or bias.

An essential part of building a robotic system is selecting the best sensors for a particular task within budget, power, computational, and mechanical constraints. As batteries often power mobile robots, sensors' energy efficiency and relative payload requirements

are critical considerations. The following section will discuss different types of sensors and their tradeoffs.

Although performance can vary depending on the quality of a sensor, the data collected will be an imperfect observation of the world. Sensors are subject to internal and external noise and, therefore, cannot make perfect measurements. Sensors are limited in range and resolution, and their performance will depend on environmental conditions like temperature or fog.

2.1.1 Absolute Position Sensors

Absolute Position Sensors allow a robot to compute its position within some known frame using an external reference. Perhaps the best-known and most widely used system for absolute positioning is the Global Positioning System (GPS). Since 1995, GPS has provided latitude, longitude, altitude, and time to a capable receiver anywhere on Earth with a clear view of the sky. This system consists of 31 satellites in orbit around the earth, each emitting regular messages detailing the satellite's position and the precise time the message was transmitted. A GPS receiver that receives messages from at least four of the satellites can compare the relative distances (determined by the time of flight of the messages) between the satellites to determine its position to an accuracy, usually within tens of meters. Though GPS is handy for many terrestrial applications, it often fails when line-of-sight to the satellite is lost, such as when the receiver is indoors, underwater, or underground. In these cases, we must determine vehicle position within a local absolute positioning system or defer to inertial sensors for relative position, as discussed in Sec. 2.1.2.

Since GPS signals are unable to penetrate water effectively, robots working underwater turn to local absolute positioning systems such as **l**ong **b**ase**l**ine (LBL) or **u**ltra **s**hort **b**ase**l**ine (USBL). Long baseline measures the relative distance to acoustic transponders placed at known points underwater (usually spaced 100s of meters apart on the ocean floor). These systems, like those mentioned before, use the time of flight through the water to measure the range between transducers on the vehicle and the reference transpon-

ders. High-frequency LBL (> 300kHz) systems can achieve sub-centimetre position accuracy, while standard full ocean depth systems with ranges of 10 km can achieve a range-dependent precision of 0.1 to 10 m [Kinsey et al., 2006]. Deploying an LBL positioning system in the deep ocean can be costly. Before the system can provide positioning information, each base transponder must be precisely localized.

USBL systems require a single, small transducer array unit that can be mounted on the hull of a surface vehicle (usually a ship) with minimal calibration between the transducer and the surface vehicle's GPS and an orientation estimate of the boat. This array measures the phase difference of a signal received from an underwater transducer to determine the relative range and bearing. The noise errors associated with LBL and USBL positioning estimates are different. LBL is often subject to bias due to initial sensor calibration but suffers relatively low ongoing noise due to sizeable spatial diversity and the fixed position of the ranging devices. USBL systems tend to have high variance due to GPS / orientation errors registered at each positioning update, especially if sea conditions induce high orientation variability. Both systems suffer if the speed of sound in water is inaccurately estimated because sound velocity varies dramatically due to water temperature and density changes in the water column. In practice, sound velocity is only approximated by sparse measurements in the survey area, which means that the localization of vehicles working underwater is usually imprecise. On-vehicle sensors, which will be discussed in the following sections, combined with state estimation algorithms, can help improve state estimates based on local measurements.

2.1.2 Internal-State Sensors

Internal-state sensors measure information about the robot, such as battery level, propeller thrust, or internal temperature. Although internal-state sensors can play an essential role in a robot's behaviour, for instance, signalling to a robot that it should return to its base when its battery has run low, they are beyond the scope of this thesis.

Inertial Sensors

Inertial sensors can estimate a robot's relative position without active references from the outside world. Robotics-relevant inertial sensors are typically combined into a device called an *inertial measurement unit* (IMU). These devices consist of accelerometers and gyroscopes that measure linear and angular acceleration and typically update rapidly.

Accelerometers behave like a spring-mounted mass where displacement under acceleration is measured using Newton's law, and the spring-mass relationship to produce the Eq. 2.1 where k is the spring constant a, m is the mass and x is the distance the spring is moved from its equilibrium [Dudek and Jenkin, 2010a]. Accelerometers tend to perform poorly in the presence of time-varying acceleration such as heave [Kinsey et al., 2006].

$$a = \frac{kx^2}{m} \tag{2.1}$$

Gyroscopes measure angular acceleration about an axis by exploiting the conservation of angular momentum. Changes in the orientation of the gyroscope result in a force that can be measured. A gyroscope's accuracy (and cost) varies widely with the method in which angular momentum is measured.

Traditional, mechanical gyroscopes are made of a rapidly spinning mass suspended in a gimbal [Dudek and Jenkin, 2010a]. Mechanical systems are prone to drift over time due to imperfections and friction in the gimbal system. MEMS (micro-electro-mechanical system) gyroscopes are small and inexpensive systems often used in modern electronic devices. Because of their low price, they are used extensively in consumer robots. These devices estimate angular acceleration by measuring the change in position (and thus change in voltage) of a small vibrating mass [Maenaka, 2008]. The mass is often a piezoelectric material induced to vibrate at a constant rate by an applied voltage. Rotation causes increased lateral motion due to the Coriolis force. Optical gyroscopes are on the other end of the performance and cost spectrum from MEMS. Optical gyroscopes calculate rotation using the Sagnac effect. They work by measuring an apparent change in the length of

an optical cavity (usually a long, coiled optical fibre) in the presence of rotation. This is expressed by Eq. 2.2, where r is the radius of the cavity, Ω is the angular velocity, and c is the speed of light in the medium of the cavity. The apparent change in length of the light path under rotation produces a change in the observed frequency of the emitted light as described by Eq. 2.3, where λ is the wavelength.

$$dl = 4\pi \frac{r^2 \Omega}{c} \tag{2.2}$$

$$df = \frac{2r\Omega}{\lambda} \tag{2.3}$$

Pressure sensors can provide altitude above or below sea level when combined with standard air or water density equations. When vehicles use only inertial navigation systems (INS) for navigation, they perform what is called *dead reckoning*, meaning that the system does not use external references for localization. Robots that rely on dead reckoning are appealing because they do not need additional sensing infrastructure; they tend to suffer from estimation drift over time due to the highlighted limitations of internal sensors. It is common in practice for mobile robots to use a combination of relatively high-update rate inertial measurement sensors and relatively low-update rate absolute position measurements for positioning.

2.1.3 External-State Sensors

External-state sensors tell us something about the world around the vehicle and are divided further into *active* and *passive* sensors. **Active sensors** emit energy and then measure how that energy interacts with the environment while **passive sensors**, like cameras, measure the environment without emission and thus tend to be more energy efficient when compared to active sensors.

Range Sensors

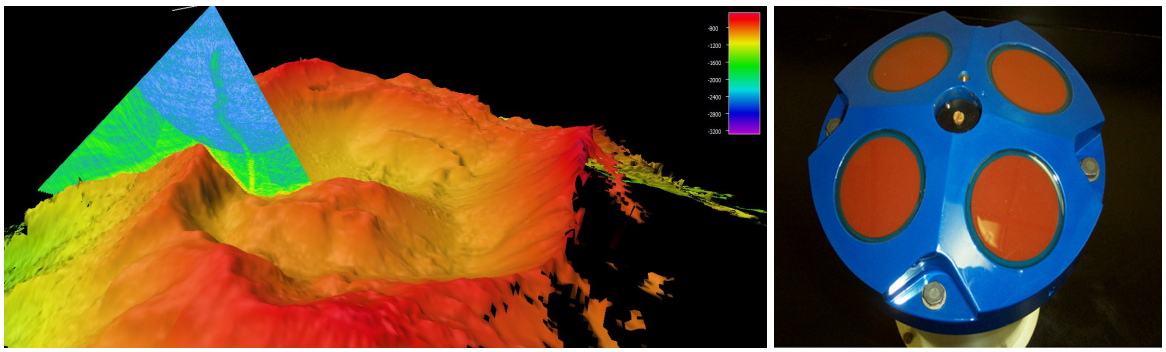
Range sensors are active sensors that measure distance by emitting a signal and then measuring its reflection on the environment. Range sensors work over a broad range of the frequency spectrum. **SoNAR** (**so**und **n**avigation **a**nd **r**anging) ranging systems utilize acoustic signals on the low end of the spectrum (3Hz-300kHz). In contrast, **RaDAR** (**ra**dio **d**etection **a**nd **r**anging) systems use radio signals (3-30 MHz). Higher frequencies tend to attenuate faster than lower frequency signals but can provide higher spatial resolution, though this typically comes at the cost of more complex receivers. Underwater ranging and communication systems use acoustic systems almost exclusively due to aggressive attenuation in water.

$$d = \frac{1}{2}ct \tag{2.4}$$

If the ranging system's transmitter and receiver are the same, the distance between the transmitter and the object from which the signal was reflected can be determined using Eq. 2.4. In this equation, c is the speed of the signal in the medium through which it travelled, and t is the time between the transmission and the return of the signal. Other attributes about the received signal, such as attenuation or phase shift, may provide additional information about objects in the environment.

Although we presented a simple equation for the range from the sensor to an object, this is often not enough detail to truly understand where the reflective object is located in environments with many objects or when the robot is moving. This is because the simple range calculation makes the naive assumption that the reflected signal is from a static object directly in front of the sensor, but this is often not the case.

Typically, transmitters utilized for range-finding will have a directional beam pattern in which the majority of energy is contained in a main *lobe* of a specified width. However, *side lobes* will also emit energy that can reflect from objects in an environment and complicate the range measurement. The effect of these multiple energy wavefronts can



(a) Multibeam **(b)** ADCP

Figure 2.1: Fig. 2.1a demonstrates a full bathymetry map of the Vailulu'u Seamount in American Samoa made with data from a multibeam sonar. In addition to the seafloor, the sensor has responded to a midwater plume of bubbles in the water column (shown in light green and neon yellow). Image courtesy of [National Oceanic and Atmospheric Administration Office of Ocean Exploration and Research, 2017]. Fig. 2.1b shows a typical Acoustic Doppler Current Profiler (ADCP) consisting of 4 transducers that alternate transmitting and receiving sound to measure particle speed in the water column. Image courtesy of [National Oceanic and Atmospheric Administration Office of Ocean Exploration and Research, 2023].

result in a circular arc of returns known as regions of constant distance (RCDs). One way to overcome the impact of RCDs is to use a more advanced transmitting or receiving system, such as *beamforming*. A system that uses beamforming receivers employs an array of physically separate receivers instead of a single receiver. This spatial disparity means the delay in a received signal between receivers provides information about the signal's direction.

Modern ground robots often employ LiDAR (Light Detection and Ranging) sensors, which emit light waves (typically 600-1000 nm wavelength) for range sensing. These sensors may use designs with multiple transmitters and receivers, moving or steered parts, or both to reduce component cost and/or increase coverage.

In underwater applications, particulates in the medium quickly degrade transmission at wavelengths in the visible spectrum. Thus, most robots in this domain utilize range finders in the acoustic spectrum. Sidescan Sonars emit a conical pulse at 100-500 khz

towards the region of interest (typically the seafloor), recording an observation directly under the sensor as the vehicle moves throughout a survey region. The result of these responses can be stitched together to form a single-channel image with the magnitude of the pixels in the resulting image indicating the strength of the response. This magnitude of reflection is influenced by several factors which can be somewhat challenging to disentangle, including distance (if an object is closer, its reflectance will be of greater magnitude), hardness (granite rock will reflect stronger than silt), and angle of incidence.

Multibeam sonars also operate in the acoustic range but typically consist of an array of transducers and produce 3-dimensional point clouds of data. This allows more precise maps of the seafloor and in the water column than side-scan sonar. See Fig. 2.1a for an example of an environment map made from a multibeam sonar.

The Acoustic Doppler Current Profiler (ADCP) is another external sensor that uses sound waves to observe the external world. The ADCP (see Fig. 2.1b) measures the velocity of currents in the water column by measuring the displacement of sound from tiny particles in the water. This sensor exploits the Doppler effect, which tells us that particles moving away from the transducer will have a higher frequency response than particles moving toward the transducer. Current measurement is an essential observation for scientific purposes, but can also be used by robots to estimate relative speed and drift.

Visual Sensors

We know that vision is a powerful sense for humans, providing a dense representation of the world that allows us to avoid obstacles, recognize predators, and understand written and non-verbal communication signals. Since cameras are passive, they tend to use less power, have a higher capture rate, and can be obtained at a lower cost than active range-finding sensors such as sonar. Although camera perception is limited to the visual spectrum and is occluded by particulates such as fog, dust, or snow, this sensor is essential to most modern robots. Even if the robot does not interpret the data directly, the recorded footage is often helpful to human operators.

Camera data is typically observed as sequential observations in rasterized colour (red-green-blue or RGB) images. A single observation of an object does not provide explicit information regarding the distance to the object or the object's size, but multiple views of the same object from known locations can provide explicit geometrical information. Systems that leverage cameras for navigation achieve distinct viewpoints by capturing a scene with either narrow-baseline or wide-baseline camera systems, where **baseline** refers to the distance between two cameras with overlapping scenes. **Narrow-baseline**, or simply **stereo** vision, involves mounting two cameras at a known distance from each other on the robot and capturing images simultaneously from both sensors. Corresponding features captured in both images can be compared to the known offset between the cameras to determine the range from the robot to the object. **Wide-baseline**, or **monocular systems** estimate geometry by using a moving camera which captures multiple views of an object from estimated locations based on robot movement. A calibrated narrow-baseline produces more accurate results due to the known physical constraint between the cameras compared to a wide-baseline, which relies on an algorithmic estimate of movement between frames. Narrow baseline systems are often limited to capturing small scenes by the necessarily narrow (due to the physical size of the robot) view difference between the cameras.

Touch Sensors

High-contact touch and manipulation, in general, remain challenging problems in robotics. One of the main limitations of developing robots capable of natural contact interactions and dexterity is the relative immaturity of touch-sensing technology. Current sensors cannot provide sensitive and responsive touch at a high resolution without considerable expense or size.

One approach on the high-cost scale is BioTac [Wettels et al., 2008], a fingertip-inspired touch sensor. This sensor has a small form factor consisting of a multi-electrode array and a hydro-acoustic sensor embedded in a soft silicon "skin."

On the low-cost end of the spectrum, visuotactile sensors (shown in Fig. 7.1) enable touch sensing with off-the-shelf cameras. Initially introduced by GelSight [Yuan et al., 2017], this class of sensors converts physical contact into an image using a camera positioned under a pliable gel. This sensor design can provide high-resolution touch sensing that is helpful when manipulating and localizing small objects in the hand. Visuotactile sensors may also have markers embedded on the gel membrane to improve membrane displacement estimates [Dong et al., 2017], or the gel may be made to be transparent, allowing vision and tactile through the sensor [Hogan et al., 2021].

Force-Torque Sensors

Force-torque sensors can measure linear and rotational forces. They are typically designed using a combination of 6-axis strain gauges, which measure resistance as pressure is applied. These sensors are often found in crucial joints in robot manipulators and help measure force when interacting with external objects.

2.1.4 Precision and Noise

Most sensors measure continuous signals in both time and amplitude that must be discretized by sampling the signal in time and quantizing the voltage. Discretization is a many-to-few mapping process that causes an inherent loss of information from the analog signal. When converting a sensor's perception of the world into a digital representation, we seek to optimize the number of bits per second or encoding rate required to represent the signal in digital format. Too few bits will result in an unacceptable loss in the resolution of the original signal, but too many bits may be expensive (in time, computation, and cost).

In addition to resolution lost due to digital quantization, intrinsic or extrinsic noise often degrades the sensor signal. Noise is any unwanted electrical signal that interferes with or distorts the desired signal. Sensor characteristics, such as electromagnetic interference or manufacturing imperfections, cause intrinsic noise. Extrinsic noise is caused by

external factors such as the temperature of interference from other sensors or the environment. In range-finding sensors, external noise in the form of reflections is a notable cause of error. Noise can be countered by the fusion of sensors with complementary noise characteristics (such as lidar and cameras) and the calibration of sensors to particular extrinsic noise in a known environment.

2.2 Robot State Estimation

Robots interact with the world around them by observing the environment through sensors and taking control actions that change the environment and/or themselves. Often, these sensors provide only noisy measurements of limited precision. In addition, the robot's actions, often expressed as mechanical force, interact with the physical world in a way that is imperfectly modelled. Formally, the robot's *state*, x_t, is defined as the collection of all parameters about the robot and its environment that may influence its future, including its pose.

We utilize the following notation as defined in Thrun's Probabilistic Robotics [Thrun et al., 2005] to discuss robot state:

- **Observation**, denoted by o, specifies a collection of sensor readings that inform robot state. Sensor measurements generally improve confidence in a current state estimate.

- **Action**, denoted by u, describe the robot's estimated change in position. In our context, a control action is usually an applied wheel torque or propeller thrust, which we expect will move the robot based on the vehicle's dynamics. Control actions tend to decrease confidence in a state estimate due to imperfectly modelled forces in the environment.

- **State**, denoted by x, describes the robot's pose and any parameters from the robot or the environment that may impact the robot's future.

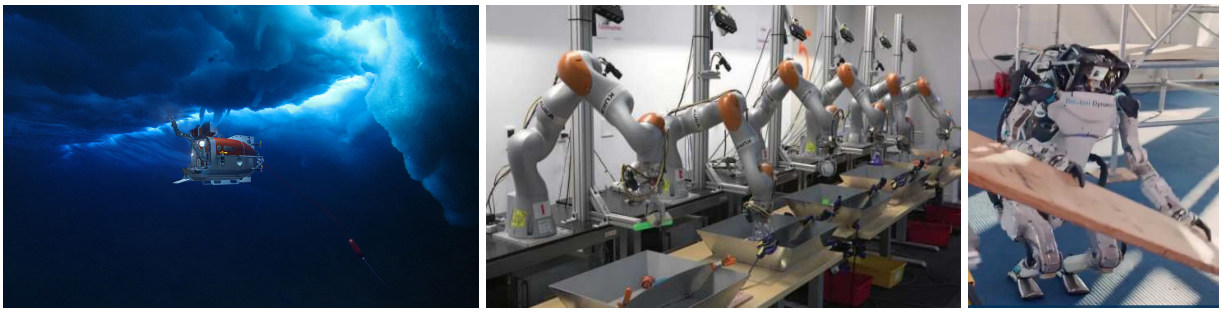
It is often convenient to make the simplifying assumption that state is Markovovian such that each state, x_t, is a complete summary of past states $x_{0:t-1}$. This assumption means that the current state is conditionally independent of prior states given the current state as expressed in Eq. 2.5.

$$p(x_t|x_{0:t-1}, o_{1:t-1}, u_{1:t}) = p(x_t|x_{t-1}, u_t) \tag{2.5}$$

2.2.1 Morphology and State

Robotic platforms come in a wide variety of shapes and sizes. This morphology will exacerbate or reduce uncertainty in state estimation. This thesis will cover two classes of robots: mobile robots and articulated robot manipulators or robot arms. As the name suggests, mobile robots have the freedom to move about the world and require sensing to avoid obstacles. They usually need a means of propulsion and control. There are significant challenges in safely and efficiently moving a robot around the real world while avoiding static and dynamic obstacles. Estimating the actual state is crucial for mobile robots aiming to generate a precise map, which may be needed for downstream uses such as scientific discovery. Mobile robots operating in mostly free space, such as aerial or underwater robots, need to estimate pose in 6 degrees of freedom (6-DOF) described by cartesian position (X, Y, Z) and orientation (*yaw*, *pitch*, *roll*). Rover-style vehicles on flat ground often only need to estimate pose in (X, Y) Cartesian coordinates and *yaw*, typically reducing the difficulty of accurately estimating pose.

Manipulators refer to the class of robots broadly meant to function similarly to a human arm, with usually 2-10 rotational or hinge joints connected by rigid links and ending with an end-effector. Manipulators can be fixed to a mount or operate on a moving base or body. Given accurate sensing of joint encoding, the pose of the arm and end-effector can be calculated with a process known as *forward kinematics*. Many of the challenges faced in this class of robots involve the difficulty of perceiving and interacting with objects external to the robot's end-effector.



(a) Mobile **(b)** Arm **(c)** Legged

Figure 2.2: Robot morphology dictates much of the challenges faced in state estimation. This diagram shows the scientific mapping mobile robot, NUI, working under ice in Fig. 2.2a [Jakuba, 2014], mounted Kuka IIWA arms picking objects from bins in Fig. 2.2b [Levine et al., 2018], and the Atlas legged robot moving a heavy board in Fig. 2.2c [Deits and Koolen, 2023]. Images are copied from the referenced publication with permission from the copyright holder.

Fig. 2.2 illustrates the state estimation challenges induced by various robot morphology. The scientific mobile robot Nereid Under Ice (NUI), estimates 6-DOF state with assistance from sensors that reflect off of (often dynamic) sea ice, the seafloor, and (always dynamic) water currents. Constantly moving currents and ice make estimating state relative to these obstacles challenging. Robot arms on fixed platforms can use forward kinematics to find a reasonable pose estimate but often need external sensors such as cameras to provide insight into the surrounding environment. Legged robots face the state estimate challenges of both classes above robots and have the added difficulty of needing to induce stability through active balance to improve sensor measurements.

2.2.2 Estimating State

Robot state estimation is the process by which a robot utilizes sensor information to determine its state or configuration. State cannot be known explicitly since it results from a series of noisy measurements. One approach to state estimation is to assign conditional

probability distributions to each of the possible hypotheses as to the actual state in a posterior distribution referred to as a *belief* (see Eq. 2.6).

$$bel = p(x_t|o_{1:t}u_{1:t}) \tag{2.6}$$

The Bayes filter [Fox et al., 2003] is a recursive algorithm that can be used for calculating

Algorithm 1 Bayes filter

1: **procedure** $(bel(x_{t-1})$, μ_t, $o_t)$
2: **for all** x_t **do**
3: $\bar{bel}(x_t) = \int p(x_t|\mu_t, x_{t-1})bel(x_{t-1})dx_{t-1}$
4: $bel(x_t) = \eta\, p(o_t|x_t)bel(x_t)$
5: **end for**
6: **end procedure**

belief from time-varying phenomena, including sensor and control data. Alg. 6 describes the two steps of updating belief, action and measurement update steps. The action step updates belief based on the previous belief distribution, the previous state, and the current control input. The measurement update step updates belief based on the sensor input with a normalization constant η.

The Bayes filter is usually intractable for real-world problems and must be approximated. The Kalman filter is a tractable implementation of the Bayes filter that approximates the state and measurement probability density functions by their first two moments, mean (μ) and covariance (Σ). Published by Rudolf Kalman in 1960 [Kalman, 1960], the Kalman filter is a core technique for estimation in Gaussian systems because of its low computational complexity.

For a Kalman filter to be optimal, the initial belief must be Gaussian, and the state transition and measurement probabilities must be linear functions with Gaussian noise. The state transition probability, $p(x_t|u_t, x_{t-1})$, is expressed as a linear function with added Gaussian noise. The random Gaussian noise vector, ϵ, describes uncertainty in the state transition with a mean of zero and covariance. The designer must explicitly define a state transition control model (often painstakingly). The Kalman filter works similarly to the

Bayes filter in that we use the process model to recursively predict the next state based on control actions and noisy observations. The current state is a linear combination of the expected state based on the control input and the difference between the predicted and actual measurements.

Despite its elegance and efficiency, the Kalman filter's assumption of linearity means it has limited real-world applications where the theory holds. Successors which can handle nonlinear estimation include the extended Kalman filter (EKF) [Anderson and Moore, 1979] and the Unscented Kalman filter (UKF) [Julier and Uhlmann, 1997].

The EKF is lightweight and efficient, making it a popular tool for state estimation in modern robotic systems. The EKF relies on propagating a Gaussian Random Variable analytically through a first-order linearization of the nonlinear system, which can result in significant errors and even divergence when the modelled system is not linear. The Unscented Kalman Filter relaxes the necessity of linearity near the mean by using an unscented transform with a similar computational cost.

The Particle Filter [Doucet et al., 2001] is another state estimation tool, but unlike the Kalman filter and its variants, this algorithm doesn't require the posterior to be of a Gaussian form. Particle filters can represent complex, multimodal beliefs even in the face of considerable uncertainty but often have higher computational requirements in practice. This Monte Carlo method approximates the posterior with random points (called particles), runs through a weighted system model, and then re-samples according to actual observations, producing an ensemble of estimates.

In the past decade, research has attempted to overcome the limitations of classic belief state estimation algorithms through machine learning. These include the Deep Kalman Filters [Krishnan et al., 2015], which learn a generative model to fit a sequence of observations and actions to transition dynamics and emission distributions. Other modern solutions, such as the robot arm farm shown in Fig. 2.2b, forego explicit state estimates and learn a mapping directly from visual observations to actions with neural networks.

2.2.3 Localization and Mapping

A robot usually needs to represent spatial features internally to plan motion toward a goal and recognize locations in its environment. This representation of space is generally called a *map*. A map details information about objects in an environment, including spatial configuration and other attributes such as maximum traversal speed or relative safety. In some problems, we will assume that a robot is given a map of the environment in advance and must determine its location within the map. Other issues will require the robot to develop its own map of an area.

Pose estimation or localization problems occur when a robot has a map describing its environment but isn't explicitly given its location. The robot must determine its location and orientation within the map by finding correspondences between its sensor observations and features in the map. Many aspects impact the difficulty of localization problems, including the quality of sensor information and how similar the environment is over space or dynamic objects within the environment.

Mapping involves building a map of the environment containing all objects and their locations. In pure mapping tasks, the robot is given its position but must relate sensed information about the environment to its current location. Mapping problems can be complicated when sensors are corrupted by noise or scenes are disturbed by moving objects.

If a robot lacks a map and knowledge about its position, it must solve a problem that is generally more difficult than localization or mapping alone. Simultaneous Localization and Mapping (SLAM) describes the computational approach to building and updating a map of an unknown environment while also solving the problem of the robot's location within this map.

EKF algorithms have achieved wide use in SLAM (Simultaneous Localization and Mapping) systems where the robot attempts to both make a map of its environment and discover its pose in said environment [Mouragnon et al., 2006, Klein and Murray, 2007, Davison et al., 2007].

In this chapter, we examined the sensor input used by robots to make decisions. In the next two chapter, we'll cover how robots select actions and make sense of observations through modeling.

Chapter 3

Actions

Robots are deployed with a wide variety of autonomy, from machines that select all their actions autonomously to those that humans fully teleoperate. This thesis will focus on robot decision-making systems where humans play minimal roles in the online control process.

Robot control can be framed as a Markov Decision Process (MDP) [Puterman, 2014]. In this mathematical formulation, outcomes are influenced by both randomness and agent control. According to the MDP assumption, at each timestep, t, an agent observes a state s_t and chooses an action a_t. Most real-world systems are actually Partially Observed MDP (POMDP) [Monahan, 1982] since the true full state of the world and the robot can be difficult to ascertain through sensing.

A *map* is a representation of the real environment, often including obstacles and robot location. In traditional robot planning literature, robots typically operate in an environment with a known map of obstacles. They are tasked to develop plans from initial to goal poses while avoiding known obstacles within their map. In this formulation, the robot only needs to localize within the map and act according to its plan. A *configuration space* represents all of the possible kinematic states of a robot within the map with a dimension for every degree of freedom in the robot. Any space in the configuration space that the robot can occupy without encountering an obstacle is known as *free space*, and a *path* is a

set of trajectories through free space between the robot's initial pose and a goal location. Depending on the task, the robot may wish to minimize the time or distance of the path or to maximize some other condition, such as the robot's safety.

3.1 Deliberative Control

Planning algorithms can work over maps by converting a map into a graph that can be searched to find a traversal path. Points in free space are often represented as nodes connected to another node if there is a path through free space between them. Graph search methods such as Dijkstra's algorithm [Dijksta, 1959] or A* [Hart et al., 1968] can be utilized to find paths from start to goal states.

Important limitations of these classic planning frameworks are that they 1) must have a map of the environment in advance, 2) the map must be faithfully converted to a graph representation, and 3) the start and goal configurations must be specified. In real systems, a reliable environmental map is often unavailable. Even if the map is available, small errors in position estimates for the robot or obstacles may cause the planned route to be invalid. To address this limitation, we can utilize online algorithms, which allow robots to update their plan as they observe information along their path [Dudek and Jenkin, 2010b]. One popular algorithm for optimal dynamic planning is the **D* algorithm** [Stentz, 1997], introduced in 1997 by Stentz. The D* algorithm develops an initial plan using A* and then adapts as new information is available.

Planning the entire path in complex environments, as described in the previous section, is often not computationally practical or complete. Probabilistically complete, sampling-based planning methods were introduced in the mid-1990s, which can be used in configuration spaces higher than 2D (such as many-link robot manipulators). Sampling-based planning methods like the **probabilistic roadmap (PRM)** [Kavraki et al., 1996], which was published in 1996 by Kavraki and Latombe, allow planning in large configuration spaces. PRMs sample the configuration space probabilistically in two phases. In the learning

phase, the algorithm generates random nodes in the configuration space and then constructs a roadmap based on the nodes that occur in free space. In the query phase, a probabilistic search is conducted over the graph. **Rapidly-exploring random trees (RRT)** [Lavalle, 1998a] were introduced in 1998. The RRT is a probabilistic algorithm that grows a space-filling tree within free space beginning at the initial pose.

Monte-Carlo Tree Search (MCTS) [Kocsis and Szepesvári, 2006, Coulom, 2007] is an any-time, best-first tree-search algorithm that can solve sequential decision-making tasks in large state spaces through planning [Browne and Powley, 2012]. Given an accurate representation of the future and unlimited time to compute, MCTS achieves high performance by *rolling out* many possible future scenarios to acquire an approximate (Monte Carlo) estimate of the value of taking a specific action from a particular state. Recently, MCTS has been combined with learned approximators to estimate the value of particular states, such as in the landmark work of AlphaGoZero [Silver et al., 2017b] and its successor [Silver et al., 2017a], which operated in Go, Shogi, and Chess.

Planning agents, like those that use the aforementioned algorithms, find actions at each decision point by considering future scenarios from their current state against a model of the world [Lavalle, 1998b, Kocsis and Szepesvári, 2006, Stentz, 1995, van den Berg et al., 2006].

Though typically slower at decision time than model-free agents, agents which use planning can be configured and tuned with explicit constraints. Planning-based methods can also reduce the compounding of errors for sequential decisions by directly testing long-term consequences from action choices, balancing exploitation and exploration, and generally limiting issues with long-term credit assignments. In addition, planning-based methods can directly adapt to differences in new environments and evaluation because standard planning approaches operate on each new environment from scratch.

The class of *model-based* control methods reference a model for forward rollouts of the future. The model is typically either simple and known (such as the aforementioned maps

or chess games), calculated by physics (which is probably at least a little bit wrong due to poor estimates of friction), or learned through observation.

Reinforcement learning (RL) is an online learning paradigm where an agent learns a *policy* through trial-and-error with its environment. In model-based reinforcement learning systems, agents optimize for future states given the current observed state and a sequence of imagined actions. The longer the set of predicted actions, the harder it is to predict the state correctly, as error compounds as we get farther from the true observation and experience state aliasing.

3.2 Reactive Control

Model-free reinforcement algorithms rely on controllers to make action selections without a lookahead and without an explicit model of the world. Reactive controllers are typically employed in situations where the model is too complex to specify when the model is expected to change over time, or underneath more complex controllers.

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \tag{3.1}$$

The Proportional–Integral–Derivative (PID) controller (Eq. 3.1) is a classic, simple, and robust algorithm that is often employed as a low-level joint or wheel controller. The proportional gain, K_p, integral gain, K_i, and derivative gain, K_d are tuned so the control signal, $u(t)$ reduces error $e(t)$ from the desired value.

On-Policy reinforcement learning agents directly use the output or samples from the policy to train the agent. REINFORCE [Williams, 1992b] is a high-variance, low-bias algorithm in which RL agents sample a trajectory from the current policy and then perform an update based on Eq. 3.2. To improve stability, most REINFORCE algorithms are implemented with a baseline value β where β is often set to be the average discounted returns or the estimate of the state's value. Proximal Policy Optimization (PPO) [Schulman et al., 2017] is a popular policy-gradient-based off-policy method for training reinforce-

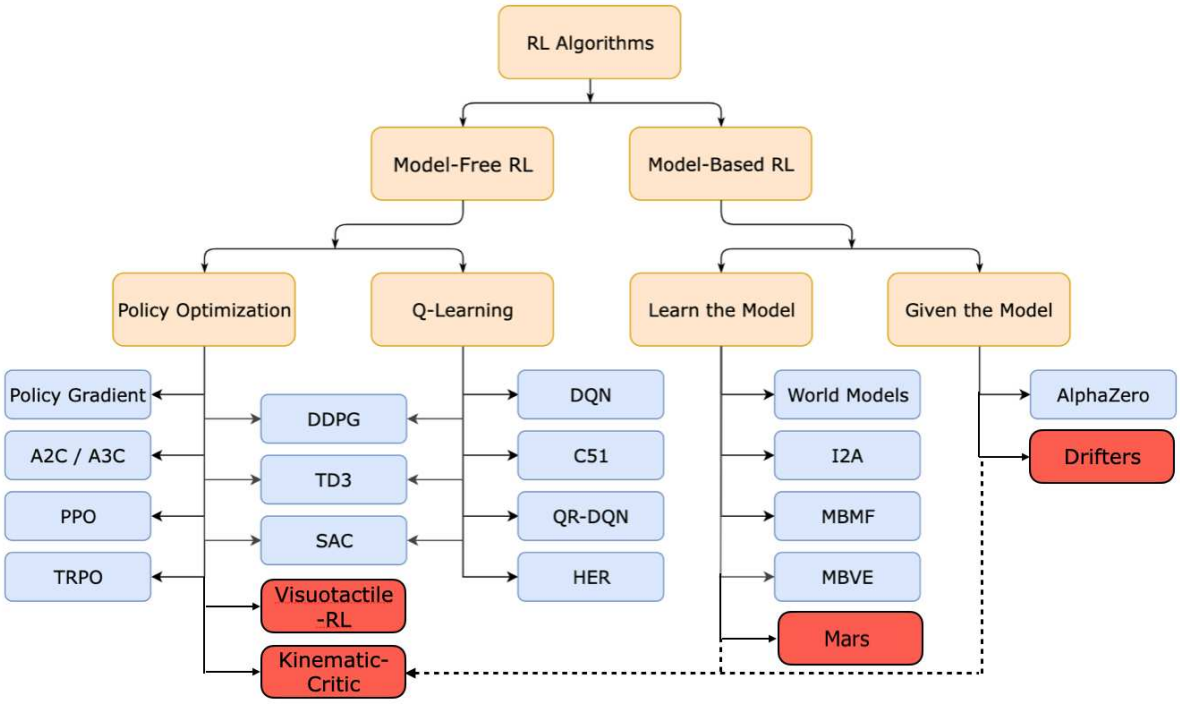


Figure 3.1: Taxonomy of RL algorithms adapted from the Spinning Up example [OpenAI, 2023]. In addition to standard algorithms in blue, our contributions are added in red.

ment learning agents that uses the notion of a Trust Region to prevent the agent from diverging during training. PPO estimates the *advantage* or difference between the value of a state and the value of the state after taking an action as shown in Eq. 3.3.

$$\theta \leftarrow \theta + \alpha \cdot \nabla_\theta \sum_{t=0}^{T} \log \pi_\theta(a_t|s_t) \cdot r_t \tag{3.2}$$

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s) \tag{3.3}$$

When the agent has access to state transitions from a different policy (typically derived from a previous iteration of the current policy) rather than the output of the current policy, the agent is considered to be *off-policy*. The ability to leverage out-of-distribution examples enables imitation learning and typically reduces the number of interactions with the environment needed for training.

$$Q^*(s,a) = \mathbb{E}[R_{t+1} + \gamma \cdot \max_{a'} Q^*(s',a') | S_t = s, A_t = a] \tag{3.4}$$

Q-Learning [Watkins and Dayan, 1992] is a popular method for off-policy methods for optimizing RL agents. Q-learning employs an iterative update where the Q-function is updated for a single state-action pair at each iteration according to the Bellman Equation. Richard E. Bellman came up with the Bellman equation in the 1950s to describe the expected return of taking a particular action in a particular state, given the expected returns of taking other actions in other states. In these RL equations, the value function, $V(s)$, represents the expected future reward starting from state s, $A(s)$ is the set of available actions in state s, and a is the action taken by the agent. γ is a tunable discount factor indicating the importance of future rewards, r. In $Q - learning$, the Q expresses the quality of a particular state and action pair.

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot \Big(r + \gamma \cdot \max_{a'} Q(s',a') - Q(s,a)\Big) \tag{3.5}$$

3.3 Imitation Learning

There are various methods of using expert demonstrations to reduce the exploration space for reinforcement learning agents. *Imitation Learning* is a method for learning to copy expert (often human) behaviour given examples. Often, imitation is challenged by the lack of matching domains between the expert and the robot. For instance, we can train a robot to pour a glass of water (as in [Langsfeld et al., 2014]) by observing human demonstrations without access to the human's underlying state. This approach helps mitigate the challenge of *exploration* in reinforcement learning but can also fail to generalize to out-of-distribution (often failure) states. *Behavior Cloning* is a fully supervised variant of IL where the agent learns to mimic an expert's policy without interacting with the environment or querying an expert. *Inverse Reinforcement Learning* is a variant of *Imita-*

tion Learning where a reward function is learned based on expert demonstrations, then a policy is learned from that reward function.

3.4 Multi-Agent Systems

There are many applications where it is advantageous for teams of robots (of the same morphology or not) to work together to accomplish a common goal. Collaborative mobile robot teams are especially attractive for environmental mapping tasks where a *divide-and-conquer* strategy can significantly reduce the time required for a survey. While mapping is generally faster with multiple robots, division of labour, communication, and data propagation become more complex. In addition to avoiding environmental obstacles, multi-robot teams must avoid colliding with each other and divide the survey space appropriately.

Multi-robot path planning is usually split into approaches using a single central planner and decoupled planners for each robot [Dudek and Jenkin, 2010b]. Centralized planning requires reliable communication between the robots and the central supervising planner. This communication is often difficult to achieve in field robotics, where communication may be limited by distance and interference from the environment, so decoupled planners are often employed.

The existence of multiple sensing vehicles can also be exploited to improve decision-making throughout the mission [Mitchell et al., 2023]. Collaborative localization has been shown to improve individual robot localization performance in a robot team with limited communication [Luft et al., 2016, Anderson and Hollinger, 2021]. In this localization scheme, the robots share information only when they've collected a relative measurement between each other. This is appealing for operating in environments where long-range, constant communication is expensive or impossible (like in underwater environments).

In the next chapter, we look at how models of the robot and its environment can be used to improve action selection.

Chapter 4

Models for Robot Decision Making

4.1 Learning from Data

Machine Learning (ML) refers to a broad class of methods that arrive at an algorithm based on experience (from data) rather than by pre-determined rules. This is useful for a wide range of problems in which formulating and writing rules for a problem is complex or unknown.

Although electronic sensors have existed for decades, recent developments in lightweight computers, sensors, and batteries coupled with reliable wireless communication and localization schemes allow us to gather information from environments that were previously seen as too costly or risky to the instrument. All of this newly generated data means that we have more information that describes human [Lane et al., 2010], built [Zanella et al., 2014], and natural [Hart and Martinez, 2006, Villarini et al., 2008] dynamics and distributions. The data from these sensors can be used to train machine learning models to model similar systems.

In addition to data collected from natural systems, artificial worlds developed from actual observations of human environments [Xia et al., 2018], simulation environments based on the natural world [Manderson and Dudek, 2018], and computer games [Belle-

mare et al., 2012] allow artificial agents to gain nearly unlimited experience in new physics and visual states without risk to the hardware [Choi et al., 2021].

Many ML methods make assumptions about the data used as experience. Perhaps the most significant assumption affecting robot learning is the assumption that samples (or experience) are **i.i.d**, meaning that the data was collected independently and that it is identically distributed (coming from the same underlying distribution). Most ML algorithms also have *hyperparameters* settings chosen outside of the learning process.

We consider three main formulations of machine learning problems:

- In the standard **Supervised Learning** task, we task a model to learn to predict a label with its paired input data. Object recognition in images is a common and significant problem in mobile robotics that is often solved via classification, where supervision is given in the form of object labels for a given image.
- In **Unsupervised Learning**, no label is given, but the model is tasked with uncovering the dataset's properties.
- **Reinforcement Learning** relies on an agent (model) collecting experience and using this experience to choose better actions, given state observations. Instead of labels, *agents* learn to map states perceived from their environment to actions that result in numerical rewards. This learned mapping is called a **policy**.

Supervised Learning can use various supervision forms to map input to outputs. In **regression tasks**, the model is asked to predict a numerical value given some input. Many tasks require **structured output**, requiring outputting a vector in which relationships exist between members of the vector. A prime example of structured output includes predicting the relevant objective of pixel-wise image segmentation of images. Models are often asked to learn the characteristics of a dataset and then generate or **synthesize** new examples similar to those in the training data.

In **transfer learning**, a model is trained on one (usually large or widely available) dataset and then asked to adapt to or *transfer* to another (generally smaller) dataset. Using

this technique, we can leverage the large dataset for generalization and the specialized dataset for *fine-tuning*.

Many modern models are **multi-task** and solve multiple problems. A popular example of such a model is Mask R-CNN [He et al., 2017], which learns image-wise, instance segmentation of objects in images (as its name implies) as well as object-level bounding boxes and human key points all in the same model. In **multimodal learning**, the model predicts output from multiple data sources. For instance, we may have multiple sensor streams, such as IMU and image data observed by our robot, that we may need to use to solve for localization.

4.2 Models for Decision Making

Reinforcement learning agents often operate in environments where generalization between similar states is useful. A model for performing the non-linear function approximation between states is with a neural network. Neural networks can also be employed to map states to actions directly. A **forward model** maps control input to the change of system state while an **inverse model** estimates the reverse and maps system change to control input.

State observations can have condensed, vector-formatted observations (often consisting of joint position, velocity, and state of task-relevant objects) or high-dimensional inputs such as camera observations, or even multimodal inputs consisting of combinations of different sensors that must be interpreted by the agent into actions.

Learning accurate models of the environment has long been a goal in model-based reinforcement learning and unsupervised Learning. Model-based agents have some advantages over agents which are model-free. Perhaps the most pertinent is that a good model will mean that the agent does not need to collect real experience to map from states to actions and instead can query its model to play out potential scenarios.

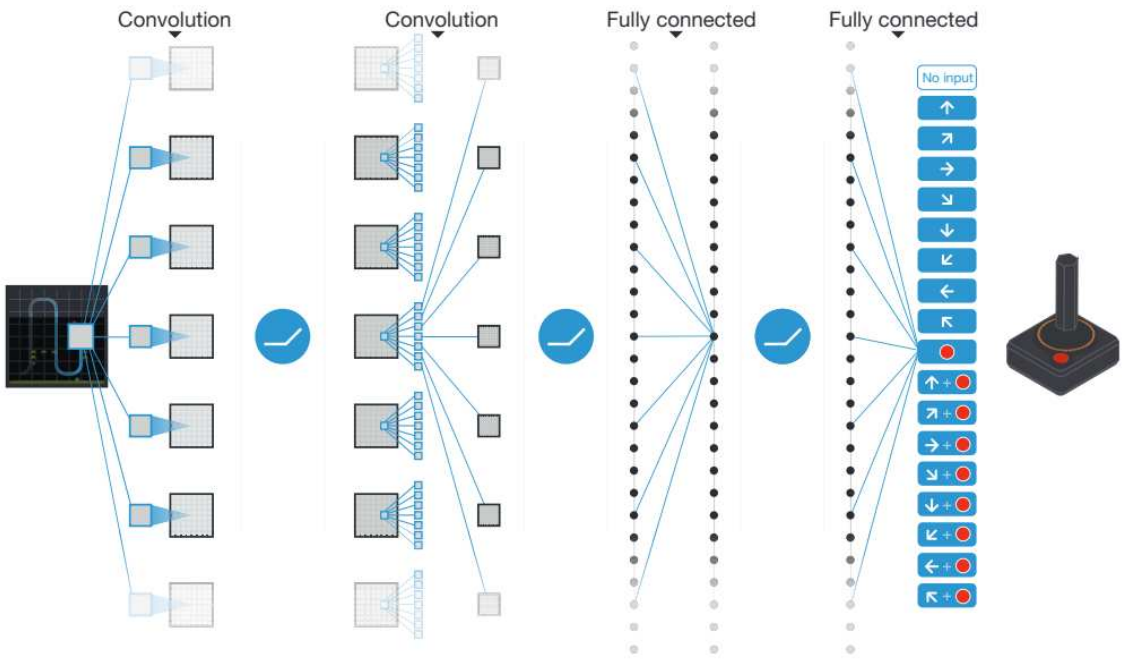


Figure 4.1: The model architecture from the seminal work of [Mnih et al., 2015]. The RL agent perceives a stacked history of the past four images as *state* and outputs discrete actions to control Atari agents.

A common reason to use model-based controllers in robotics applications is their increased interpretability to humans. When a model-based robot fails to behave as expected, humans can replay the model state, often identifying the reason the model failed and working to improve it. One example of this is in System Identification (SysID). Traditionally (and still command today), roboticists employed the statistical process of SysID to build mathematical models of real robot attributes. These physical models can provide stable insight for agents who must reason about prior actions or plan future actions based on underlying physical phenomena.

Modern approaches often incorporate physics-motivated inductive biases into learning architectures. In Deep Lagrangian Networks [Lutter et al., 2019], the physics prior is encoded as a differential in the network topology. This inductive bias enables improved extrapolation outside the training set since the underlying physics is true in regimes not specifically seen in the dataset. In AugWM [Ball et al., 2021], agents are trained in simula-

tion with simple dynamics transformation that seeks to capture expected changes in the physical properties of the robot. Agents are conditioned on the sampled augmentation as input context, which allows rapid estimation of the context of the real environment during evaluation.

Work has shown the power of learning action-conditional models for training decision-making agents which operate in perceptual states [Ha and Schmidhuber, 2018, Schmidhuber, 2015, Buesing et al., 2018, Oh et al., 2015, Graves, 2013] and combining planning and with environment models [Silver et al., 2016, Zhang et al., 2018, Pascanu et al., 2017, Guez et al., 2018, Anthony et al., 2017, Guez et al., 2018]. In World Models [Ha and Schmidhuber, 2018], a recurrent neural network (RNN) learns a conditional sequence model over the latent space of a Variational Autoencoder (VAE) trained on perceptual input [Kingma and Welling, 2013].

Recent work has seen a surge in imitation-based agents which rely on high-quality expert examples such as Aloha [Fu et al., 2024] and Decision Transformer [Chen et al., 2021] that use transformer model architectures [Vaswani et al., 2017].

4.3 Simulated Experience for Learning

Reinforcement learning agents employed in robotics often require many unique interactions with the environment during training. Practitioners often leverage simulated environments because this training can be too costly on real hardware. Designing sufficiently realistic and appropriately challenging tasks in simulation is a delicate and meticulous job for practitioners, as the environment and robot should accurately prepare the algorithm for deployment in the real world.

An early success in sim-to-real transfer with neural networks was the ALVINN [Pomerleau, 1988] project, which demonstrated autonomous road following in 1988. This robot was trained in a simple simulator and learned to map input from images and a laser range finder into steering direction using a 3-layer feedforward neural network.

Simulation experience, while helpful in generating data, often does not provide perfect transfer to real robots due to the sim-to-real gap [Jakobi et al., 1995] that often results from an imperfect match between the simulated training environment and the states and observations encountered on the real system. A well-acknowledged challenge of leveraging simulation environments for training or refining robot policies is the potential for visual and dynamics mismatch between the simulation environment and the deployed environment. To bridge the sim-to-real gap, one can employ dynamics randomization (DR) to the simulation environment. In DR, various world and robot physics attributes are randomized during training. This randomization includes varying the mass and inertia of the robot components, the objects they interact with (if applicable), and varying friction coefficients and properties. The agent learns to become robust to the dynamics changes seen in simulation, thereby making it robust to the inevitable sim-to-real gap.

In one such example of this approach, [Yu et al., 2017] learn a control policy that is trained under domain randomization and then utilize Online System Identification with state and action to predict dynamics model parameters online, producing a robot capable of robust control over a wide range of conditions. In [OpenAI et al., 2019], *automatic domain randomization* increased randomization during training, successfully transferring a simulation learned policy to a hand control policy that could solve a Rubik's Cube.

In the next chapter, we examine various methods introduced in this thesis. Our systems encompass tools covered in Chapters 2, 3, and 4 where we leverage structural knowledge about state estimation and available actions to develop new models for robots scientific sampling robots in Part III and table-top manipulators in Part IV.

Part III

Informed Scientific Sampling with Algorithmic Priors

Chapter 5

Physics-Informed Sampling in Flows with Passive Sensors

5.1 Introduction

In this chapter, we tackle problems at the confluence of robot morphology and structured prediction, examining how we can combine known fundamentals about the world, robot, and sensor limitations to improve access to data that describes aquatic environments. We frame these problems around goal-oriented, mobile robots working in large-scale spaces where *large-scale spaces* are defined as worlds in which the robot must change its physical position to acquire full scene understanding from multiple vantage points. This chapter encompasses work presented in several conference publications including:

- We introduce a robot team that leverages unactuated robots for all or part of a surveying mission.
- Develops a model-based algorithm for optimizing drifter sensor path planning with a tuned ocean-physics model.
- Builds an algorithm for complimentary surveying with a team of drifters and autonomous boats.

- Introduces hardware innovation for drifter shepherding.

.

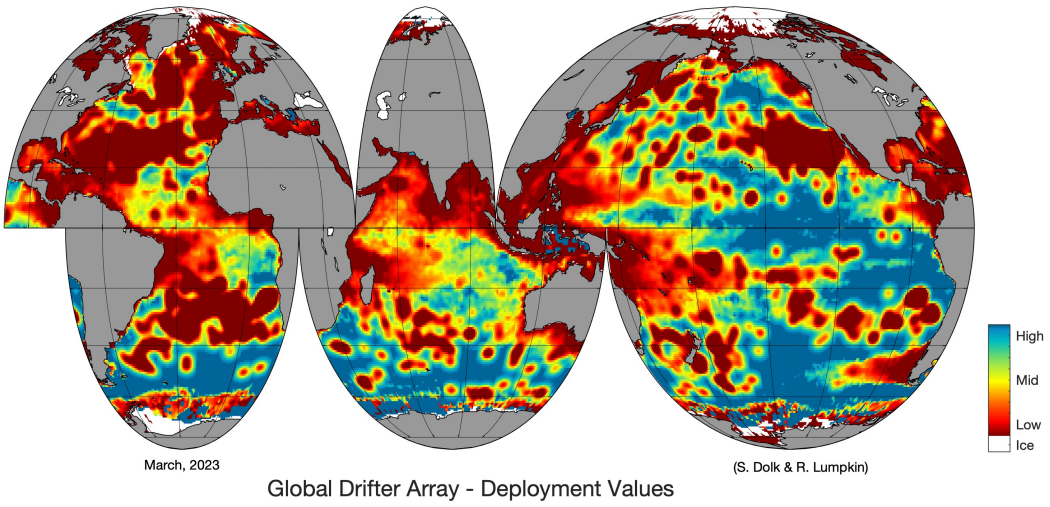


Figure 5.1: This map of global drifter deployment location values in March of 2023 [NOAA] is calculated using the method from Dolk and Lumpkin [Lumpkin et al., 2012]. Locations that they found to be associated with high-value deployments are shaded blue and less desirable deployment locations are found in red.

Scientific understanding and modelling of spatiotemporal phenomena related to the water, weather, and climate at the marine-air boundary layer are key to preserving the health of marine ecosystems and improving climate predictions. Predictive models of these interactions, which help us forecast a number of important phenomena, including climate change, weather, and marine health, require extensive *in-situ* data collection to improve accuracy. Spatiotemporally extended local measurements across the world's lakes, oceans, and rivers entail significant investments in physical infrastructure in an interface that is often harsh, requiring sensors that are robust to waves, storms, and extreme temperatures and remote, necessitating long operating times without human intervention or battery changes.

One such tool for gathering these scientific samples is floating sensors called *drifters*. Drifters are propelled strictly or predominately by the ambient wind and/or water cur-

rents and have been used extensively in oceanography [Wilson et al., 1996, Soreide et al., 2001, Lumpkin et al., 2007]. These devices can be equipped with any number of sensors (such as temperature, pH, or cameras) that enable them to sample the local environment and report geotagged data back to a central server. Because they inherently require no or very little intelligent control, propulsion, or human intervention, these devices are attractive for various scientific applications that are sensitive to cost and do not have precise spatial requirements on samples.

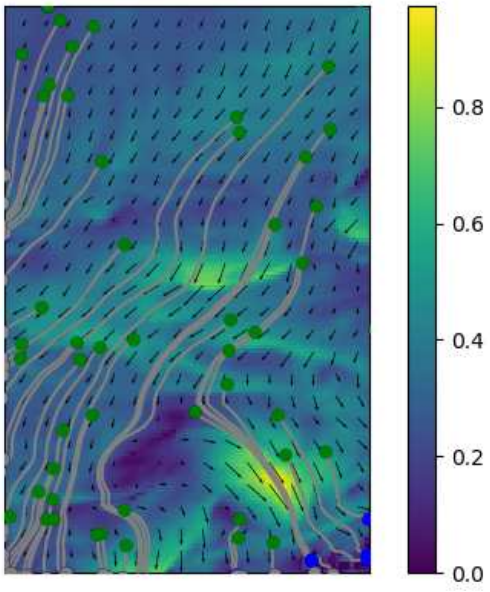


Figure 5.2: In this figure, drifters were deployed randomly over a simulated flow field to illustrate example trajectories from a deployment point (represented by a green dot). The flow field direction is described in each grid cell by an arrow. Speed is represented by the colour map in m/s according to the attached colour bar. This particular flow field is one of 40 used for evaluation and serves as the ground truth flow field, $\vec{V}$, for Figures 5.13 and 5.10.

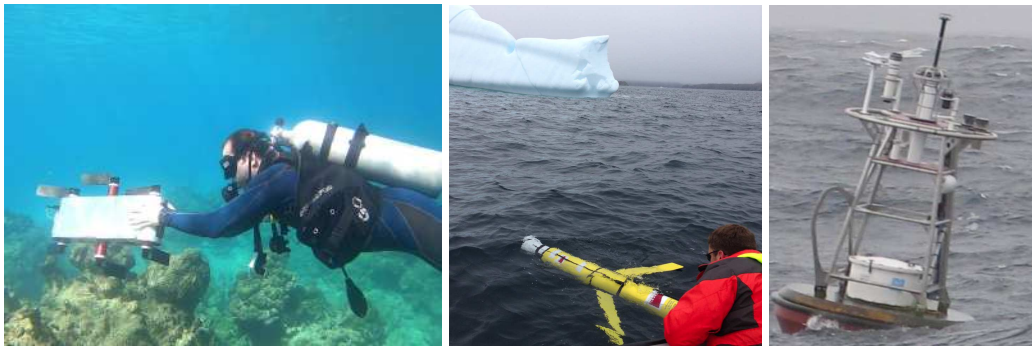
5.2 Problem Statement

Although passive sensors cannot control their own movement, their fate can be estimated given the known physical characteristics of the drifter and an estimate of extrinsic factors.

These external factors, namely wind, waves, and currents, can be difficult to measure and (more so) predict, especially in boundary conditions near coastlines or in areas with poor forecasts, such as the remote ocean. In addition, the complex interactions occurring between the drifter at the air and water boundary will not be fully accounted for; therefore, accurate drifter prediction, even a few days into the future, remains a significant unsolved challenge [DARPA-FFT, 2021]. In this work, we examine the problems related to drifter forecasting. This work aims to develop a low-cost, non-static sensor system to collect measurements in a flow field with minimal investment by exploiting natural currents to sample trajectories over a survey region. We introduce a collated dataset (described in Section 5.3.4 for use in machine learning that enables ocean-scale drifter trajectory forecasting (the position of the drifter over time as described by latitude and longitude). We also discuss our work in developing robot-in-the-loop mapping with drifters, optimizing for both coverage and data reconstruction accuracy in Section 5.5.

5.3 Background

The concept of the drifting data collector, perhaps beginning with a message in a bottle [Sverdrup et al., 1942], has long been employed to gather information about bodies of water [Soreide et al., 2001]. The largest network of drifters is organized by the Global Drifter Program and has been conducting global-scale data collection in the world's oceans since 1979 [Lumpkin et al., 2007]. This program's 3000+ operational drifters gather a plethora of scientific readings that have been integral in building and validating the ocean models [Blockley et al., 2012], improving the accuracy of remote sensing estimates [Perez et al., 2023] and informing transport calculations of other passive particles such as marine debris [Maximenko et al., 2012] and radioactive contaminants [Nagatani et al., 2013].



(a) AUV **(b)** Glider **(c)** Mooring

Figure 5.3: An overview of sampling of robot morphologies used in aquatic surveying. Fig. 5.3a shows the Aqua AUV [Dudek et al., 2007] performing informed visual mapping of a coral reef in [Koreitem et al., 2020]. Fig. 5.3b shows a partially controlled Slocum glider [Schofield et al., 2007] surveying the underside of icebergs by taking informed dives in [Zhou et al., 2019]. Fig. 5.3c shows the sensor payload of a typical surface mooring from [Kamphaus et al., 2008].

5.3.1 Vehicles for *In-Situ* Marine Sampling

Robots are widely used for scientific discovery in marine environments, reducing the need for laborious manual data acquisition campaigns [Madrid and Zayas, 2007]. Marine robots face unique challenges when compared to warehouses or even other field robots. The most obvious obstacle is the harsh operating conditions - especially saltwater, which is corrosive and can induce failure in electronic systems.

The mobile systems we work with in this chapter all must carry their own source of power, and power failure or depletion can often mean the loss of a system in the ocean. There is often little or no infrastructure for communications, meaning robots must work without access to the internet and with limited connectivity. Operating locations are often remote, which makes ease of deployment with a small engineering team an important factor.

Most surveying systems involve a trade-off between cost, mobility, and resolution in time and space. Traditional marine surveying schemes typically lack spatial or temporal resolution (and sometimes both) because of the expense of continuously sampling

harsh marine environments. Sophisticated robots, such as autonomous surface vehicles (ASVs) or autonomous underwater vehicles (AUVs), can achieve high-sampling resolution by using thrusters to control their position. Controllable robots can perform scientific mapping with an exhaustive [Williams et al., 2010, Pizarro et al., 2014] or selective [Girdhar and Dudek, 2016] sampling strategy according to research needs. Autonomous surface vehicles (ASVs) have increased prominence in near-shore surface observation operations [Ferri et al., 2015, Madeo et al., 2020, Pradalier et al., 2019]. These often unsophisticated and intuitive systems fill various niches such as persistent monitoring [Koay et al., 2017], collecting in situ collections of both physical samples [Manjanna et al., 2018, Flaspohler et al., 2018, Das et al., 2015a] and observational samples [C. L. Gentemann et al., 2020, Manjanna et al., 2020].

There have been consistent efforts to reduce the cost (both human and monetary) of acquiring *in-situ* scientific marine data (see Table 5.1). Gliders are a low-cost platform that uses various, mostly passive designs to move through the water. This morphology excels at travelling far distances over long periods with minimal servicing, but it is at the mercy of the local environment for much of its movement. Slocum gliders [Schofield et al., 2007] are shaped like a torpedo and use pumps to gently change buoyancy over time, allowing the vehicle to glide up and down through the water slowly. These machines are partially controllable as they can change the position in the water column to catch ambient currents.

Fully passive sensors such as fixed moorings [Chavez et al., 1997] can be deployed at a low cost but suffer in spatial resolution as they have no means to move about a survey region. The low cost of individual moorings makes deploying arrays of static sensors (such as in [Brainard et al., 2009]) an attractive approach for long-term monitoring. This deployment method can take measurements at high frequency (depending on power constraints) but is usually too costly to deploy at a high spatial resolution over a large region.

Platform	Vehicle	Deployment	Controllable	Water Column	Mobile
AUV	$$$$	$$$$	Yes	Yes	Yes
ASV	$$	$$	Yes	No	Yes
ASV+Winch	$$	$$	Yes	Mostly	Yes
Glider	$$$	$$$	Partial	Mostly	Yes
Drifters	$	$	No	Possible	Yes
ASV+Drifters	$$	$$	Partial	Possible	Yes
Mooring	$	$	No	Possible	No

Table 5.1: A comparison of autonomous data collection platforms. Although the values are estimates and can vary depending on sensor fidelity, cost generally increases with controllability, depth, and endurance. Drifters can be built for less than $100 and surveying AUVs/ROVS are often hundreds of thousands of dollars.

Drifting sensors, though inexpensive and offering flexible deployment, cannot be explicitly utilized for actively gathering information from scientifically important positions as they are not spatially controllable after release. Drifting data collectors come in many shapes and forms, including the most simplistic floating drifter, which captures a 2-dimensional profile of the water surface. There are also subsurface drifters which can gather data in three dimensions (such as Argo profiling floats [Jayne et al., 2017]) and moored sensors [Doherty et al., 1999] that are anchored in place (either at the surface, at depth, or can move throughout the water column). One appeal and differentiation of ocean drifters from other morphologies discussed in this section is that once they are deployed, they need not necessarily be recovered (as opposed to more expensive platforms), which saves considerable engineering time and resources. Most drifters are deployed without the intention of recovery re-deployment, making them attractive from an engineering standpoint, while raising some questions regarding their eventual destiny in the environment.

5.3.2 Estimating Drifter Trajectories

Passive objects on the sea surface experience basic disturbance forces as described in Eq. 5.1 where V is the velocity of the object, m is the mass of the object, and m' is the acceleration of water particles along the hull (underside) of the object [Hodgins and Hod-

gins, 1998]. Drifting objects also experience Stokes drift, a *downwave* force caused by water particles in a wave field. Although Stokes drift is a strong factor in advection (the movement of matter by the fluid flow) for suspended objects (such as larvae or sediment), it is difficult to measure in field campaigns and has been excluded from most empirical models.

$$(m + m')\frac{dV}{dt} = \sum F \tag{5.1}$$

Estimating the trajectories of drifters in flow fields requires solving for a time-evolving forecast that is dependent on the forcing fields, primarily: wind, wave, and current vectors. Physics-based simulators are widely used in fluid dynamics research such as aircraft, atmospheric modeling, and oceanography. The accuracy of these physics simulations is highly dependent on the fidelity of the numerical methods and the quality of the input data and boundary conditions.

All experiments in this dissertation utilize OpenDrift [Dagestad et al., 2018], an open-source, parameterized particle advection engine for drifter trajectory predictions. OpenDrift is a physics-based simulator developed primarily by the Norwegian Meteorological Institute that works by numerically solving Lagrangian equations of fluid dynamics using drifting particles as a reference.

The Leeway Model [Breivik and Allen, 2008] is a popular empirical method contained within OpenDrift for predicting the *leeway* or relation between wind and the motion of a drifting object. The Leeway Model was developed by analyzing the real trajectories of 63 categories of search and rescue objects that were compiled by the US Coast Guard [Allen and Plourde, 1999]. The trajectory model of the drifter for Leeway models (Eq. 5.2) utilizes the second-order Runge-Kutta method to find the arc traced by the leeway vector on the surface current. Trajectories are calculated as an ensemble, with external forcing functions (wind and water current) embedded in V along with random perturbations to represent the heteroscedastic nature of the observed data (Eq. 5.3).

$$x(t) - x_0 = \int_0^t V(t')dt' = \int_0^t [L(t') + u_w(t')]dt' \tag{5.2}$$

$$a_n = a + \epsilon_n/20 b_n = b + \epsilon/2.0 \tag{5.3}$$

5.3.3 Drifter Deployment Values

The oceanographic community has extensively examined the effect of a deployment location on a particular drifter's trajectory over its lifetime at the ocean scale. It is well-recognized that deploying drifters in high-value locations will maximize drifter data coverage, reduce deployment redundancy, and increase drifter lifetimes. [Lumpkin et al., 2012] examined factors affecting drifter lifetimes over a global fleet of drifters, focusing their efforts on identifying regions of the globe where drifters tended to die very quickly. They identified a key metric of the drifter half-life which is the rate in days at which half of the drifters released from a particular location stop communicating due to electronics failure or drowning (posed to be due to harsh weather conditions). Other metrics are based on the relative value of new observations considering the configuration of the currently deployed drifter array. A map of the current (depicted in Fig. 5.1) drifter deployment value formulation is updated monthly on NOAA's website.

In [Molcard et al., 2006], the authors show that directing the initial drifter positions along the out-flowing branch of Lagrangian boundaries optimized the relative dispersion of drifters. This work also demonstrated that the performance of drifter data assimilation into ocean models strongly depends on the independence of the observed drifter trajectories. In [Salman et al., 2008], the authors compare long-term multi-drifter deployment strategies and show how different release formations affect data assimilation performance. They demonstrate that total error reduction requires the dispersion of drifters, but local errors are reduced by targeting specific flow features.

Environmental Data Details				
Data Source	**Variable**	**Forecast**	**Spatial**	**Temporal**
DARPA FFT	Drifter	NA	NA	24 hrs
Global Forecast System (GFS)	Wind	10 days	0.25°	1-3 hrs
Real Time Ocean Forecast System (RTOFS)	Current	8 days	0.08°	1-3 hrs

Table 5.2: Drift-NCRN [Hansen et al., 2022b] describes our collation of the highest resolution / longest environmental forecasts available to the public at the time of the Darpa-FFT Challenge.

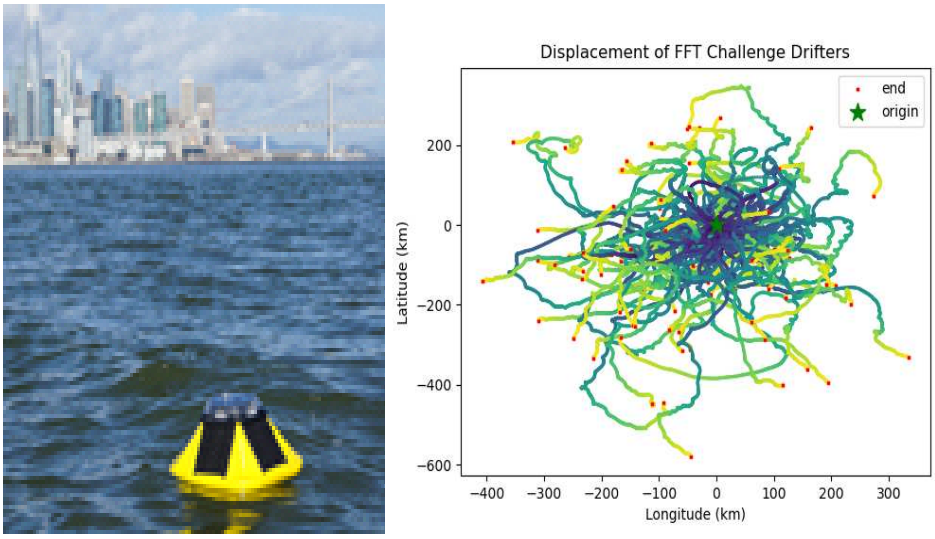
Though a drifter's low cost is what makes it an attractive approach for marine sampling, its simplicity also makes it a difficult platform from which to sample efficiently. A traditional drifter's initial deployment point is its only controllable aspect. The remaining sample trajectory, and thus the amount of the survey area covered by the sensor, is dictated by the ambient flow field. When a drifter in a drifter network is poorly deployed or confronted with unstable currents, it can become trapped in an eddy or quickly rushed out of a survey area, collecting redundant or few samples for the survey. Thus, the need for a strategic deployment is necessary to achieve any form of efficient coverage with drifters.

This was evident in our earlier work, [Manjanna et al., 2017a], in which we introduced an adaptive sampling technique that relied on deployed drifters for explorative scouting for an ASV performing adaptive sampling. In these experiments, the drifters tended to clump together or rapidly exit the survey area, reducing their utility.

5.3.4 Flow Field Datasets

In the previous sections, we discussed the limitations to gathering *in-situ* data in real marine environments. Real-world data from dense flowfields is not (widely) available due to the complexities of acquiring high-resolution samples. Much of the work in this chapter utilizes flow fields generated from the prominent Regional Ocean Modeling Systems (ROMS) model [Shchepetkin and McWilliams, 2005], which utilizes efficient physical and numerical algorithms to estimate flow. By using a numerical model, we can produce a

regular output grid of wind and water flow field vectors to use as a source of ground truth for algorithms related to robot sampling.



(a) Spot Drifter **(b)** Spaghetti of Trajectories

Figure 5.4: Fig. 5.4a shows the Spot drifter used in the DARPA-FFT competition [DARPA-FFT, 2021]. Fig. 5.3.4 depicts the dataset of ninety drifter trajectories over the twenty days prior to the start of the DARPA-FFT competition, translated to the same origin. The color indicates the day of the sample with the red point indicating termination and a star indicating the start date.

We will also utilize time and spatially tagged real drifter trajectories and combine these trajectories with measured and estimated values of environmental conditions throughout the trajectory. One source of this real data was generated during the DARPA Forecasting Floats in Turbulence (FFT) Challenge [DARPA-FFT, 2021]. This challenge was run in real-time in the Fall of 2021 with the goal of improving the state of algorithms used for drifter forecasting. The challenge released ninety GPS-enabled drifters (Fig. 5.4a) throughout the Atlantic Ocean and asked competitors to predict the drifter fates ten days into the future, given twenty days of prior tracking data. The competition highlighted the difficulty of drifter trajectory forecasting as fewer than half (14 of the 31 teams) of the organizations participating in the challenge predicted *any* of the ninety drifter positions on day 10 to within 32 km of the true position.

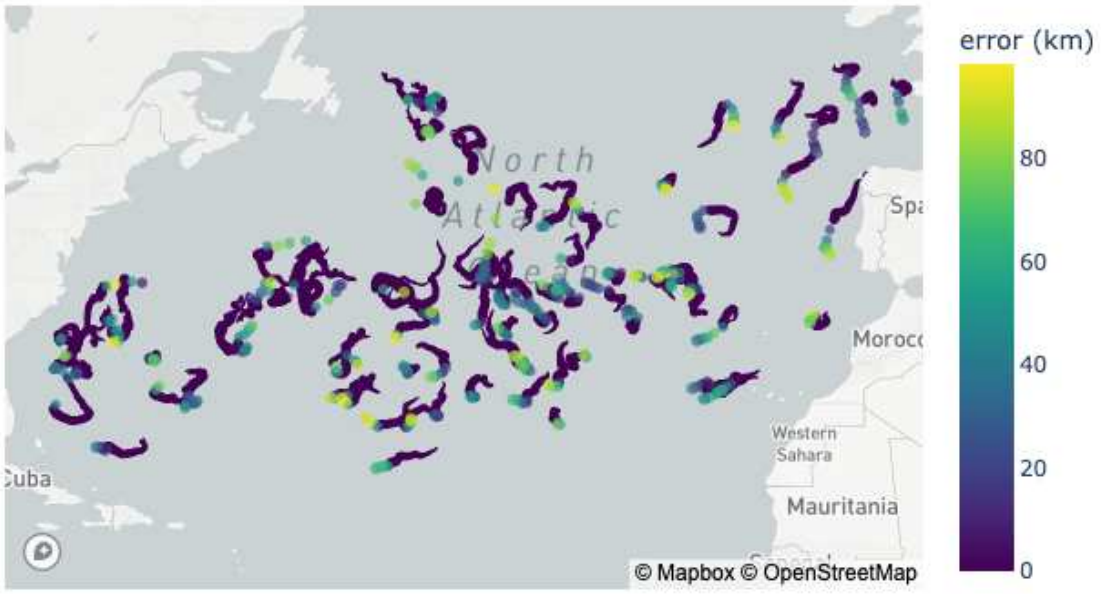


Figure 5.5: The full DARPA-FFT dataset of drifter trajectories in the Atlantic Ocean. Training data (days 0-20) is shown in purple traces. Our trajectory prediction error using the Leeway Model is shown on the test set (days 20-30 of the competition).

Drifter trajectories are a function of external factors including wind, waves, and local weather (see Fig. for an illustration of trajectory independence). At the time drifter predictions are made, one necessarily needs to rely on forecasts for these parameters as they have not yet occurred. At the time of this project, NOAA ocean current forecasts were only available for the first eight to ten days of the trajectory prediction, depending on resolution and time-frequency. Significant forecast error, especially as the time horizon of the forecast increases, makes the data unreliable and it is important to capture this lack of accuracy in the dataset used for tuning prediction models. Furthermore, these forecasts are typically ephemeral data products (at least to the public) due to data storage limitations. In our dataset paper [Hansen et al., 2022b], we collect the data products needed to complete the FFT competition (described in Tab. 5.2) and release them to the public in a format common to machine learning research.

5.4 Related Work

In typical self-propelled robotic systems, exogenous (external) forces are usually considered a mildly antagonistic influence to be counteracted to reach a desired pose. In deployed under-actuated or un-actuated systems, however, external forces are the propulsive mechanism used to achieve robot movement. Much of the previous work on coverage and deployment of sensor systems assumes either an environment where agents can move deliberately [Hollinger and Sukhatme, 2013] or systems in which only initial placement in a static environment is considered [Wu et al., 2012, Rahimi et al., 2005a]. [Pontbriand et al., 2015, Hollinger et al., 2012] demonstrates the use of autonomous underwater vehicles to extract data from arrays of underwater static sensing platforms, exploiting a complimentary team composed of a single high-cost but high-mobility robot and the high spatiotemporal resolution of low-cost of a team of static sensors.

The following paragraph will discuss related work in trajectory planning in marine flow fields, ordered from full-actuation to passive systems. In [Smith et al., 2010, Kularatne and Hsieh, 2015], the authors present methods for physically tracking features of interest by utilizing the output of predictive ocean models to inform planning for marine vehicles. [Kularatne et al., 2018] find energy-efficient motion plans in flow fields for fully actuated vehicles using graph search methods. Several works have also considered the use of drifters for mapping [Manjanna et al., 2017a, Shkurti et al., 2012, Das et al., 2012] and have studied the interception of drifting sensors [Meghjani et al., 2016].

Perhaps more relevant to our work are systems which enable under-actuated vehicles to strategically move through flow fields according to environment models. These situations impose limitations on planning, which makes initial deployment points important. [Pereira et al., 2013] present techniques for risk-aware path planning and demonstrate AUVs navigating in strong currents, utilizing regional ocean models to avoid trajectories where collision with external obstacles is more likely. In [Leonard et al., 2007], the authors describe a performance metric for determining optimal paths with self-directed

underwater gliders, which minimizes the error in the model estimate of the sampled field. In [Inanc et al., 2005], the authors demonstrate an approach to optimal trajectory planning for an underwater glider using Lagrangian Coherent Structures with global flow geometry based on approximate ocean current models. [Hollinger et al., 2016] explores the use of modelling the uncertainty in ocean model predictions for improving glider navigation in busy shipping channels. They leverage Gaussian processes (GPs) augmented with interpolation variance to provide confidence measures on predictions and improve planning in simulation and field deployment.

Active drifters, which have some limited ability to influence their trajectory, are an active area of research for use in strategic sampling tasks. [Kwok and Martínez, 2010] optimized for coverage in a riverine environment with drifters equipped with limited velocity control. In [Kwok and Martínez, 2012], a team of underactuated agents exploit flowfields in their environment to maximize the total coverage area. In [Ouimet and Cortés, 2014] active underwater drifters alter their depth to use ocean flow fields that are carefully modeled in advance for coordinated rendezvous to enable efficient recovery. The active drifter system proposed in [Molchanov et al., 2015] performs spreading and aggregation in an ocean simulation model by lowering or raising individual drifter drogues to take advantage of ocean currents measured *in-situ* by individual drifters to reach a goal point.

5.5 Robot-Drifter Teams

So far in this chapter, we've discussed the complexities and costs associated with capturing aquatic data, focusing on problems induced by platform morphology. We've also highlighted the limitations of publicly available models for predicting environmental conditions (namely wind and current) that influence the performance of low-cost platforms such as passive drifters, which are wholly dependent on the local flow field for collecting spatially diverse samples. In this section, we will discuss two related efforts to marry

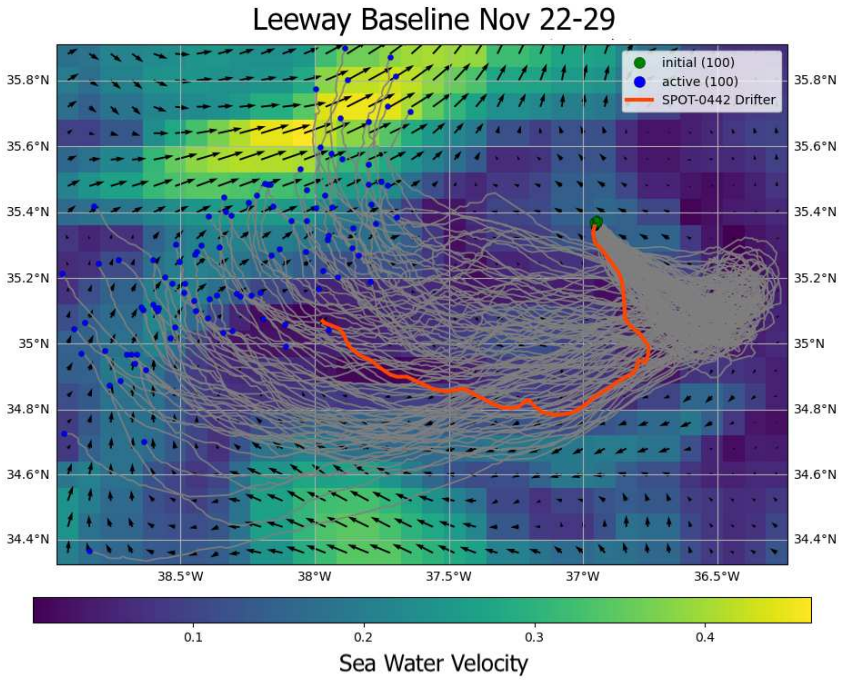


Figure 5.6: Predictions (gray) for the fate of drifter, *Spot 0442* of the DARPA-FFT dataset compared to predictions from the Leeway model of OpenDrift using a 10m seeding radius and environmental dataset collated in Drift-NCRN.

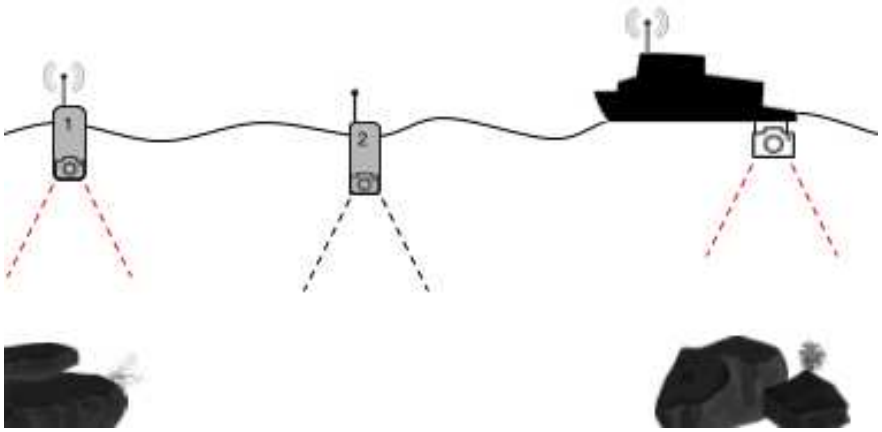


Figure 5.7: A system overview depicting two deployed drifters that relay information about their local environment back to an autonomous boat.

the advantages of intelligent surface vehicles with low-cost floating sensors (Fig.5.7) to leverage their inherent advantages for sampling. In Section 5.6, we develop a physics-informed model for estimating drifter placement value with the goal of optimizing cov-

erage over a survey region and then extend this work in Section 5.7 to enable adaptive sampling of an ASV and drifter team to optimize reconstruction of an unknown survey region. First, we'll formulate a common vocabulary and mathematical description.

5.5.1 Problem Formulation

We introduce a mapping problem in which a network of passive floating sensors is used to collect samples in a body of water. This system employs an iterative measurement and modeling scheme to incrementally deploy drifting sensors so as to collect new observations, despite only controlling the initial deployment point of the sensor. Once deployed, sensors are moved about a survey area by ambient surface currents which can only be observed *in-situ*.

Drifter trajectories can be modelled effectively with an accurate flow field representation; however, at the basin-sized scale ($< 5km^2$ survey region with sample resolution of $< 20m$), we consider when mapping with the ASV drifter team, the flow field for most bodies of water is largely unknown before the survey begins and is only observable *in-situ*. Even if the dynamic model of the flow field is known, the question of optimal drifter placement remains. An exhaustive placement search could be performed through the known flow field with a particle trajectory simulator, however, this search is generally too costly to perform in real-time deployments. In the sections that follow, we describe our approach to overcoming these two limitations and present a method for non-actuated sensor coverage optimization in partially observed flow fields that is able to run in a real-time system.

We propose and implement a method for coverage of survey region, R, with drifters which handles these bottlenecks by iteratively assimilating flow field observations and assimilating them into a full flow field model. Our two objectives (sample coverage and accurately modelling the flow field) are somewhat dichotomous because sensor placements that maximize our understanding of the sample region are also inherently risky

since the drifter will be placed in a part of the flow field that is currently poorly modelled.

We define the flow field as V which is a distribution of water and wind velocity over R, denoted in Euclidean space as $\vec{V} = \vec{V}(x, y, z, t)$. Velocity at every point in $\vec{V}$ is defined by components in each coordinate direction as described by $\vec{V} = u\vec{i} + v\vec{j} + w\vec{k}$. Each velocity component (u,v,w) is a function of (x,y,z,t) as defined in Equation 5.4. We assume that z is a constant (at the water's surface) and that the vertical current is negligible since the drifters have substantial positive buoyancy. In addition, we assume that the flow field is a steady flow in that it does not change over our observation window, leaving $\vec{V} = \vec{V}(x, y)$ and $\vec{V} = u\vec{i} + v\vec{j}$.

Each deployed drifter is denoted by the time-independent point, d_n, which also describes the drifter's initial deployment point in $d_n = (\bar{x}_{init}, \bar{y}_{init})$. A deployed drifter has a trajectory $\bar{D}_n$, which is the collection of the points it has visited in R. The collection of observed samples along $\bar{D}_n$, $(\bar{x}, \bar{y}, \bar{u}, \bar{v})$ is denoted by $\bar{O}_n$. We refer to the collection of trajectories from all deployed drifters as $\bar{D}$ and the collection of observed samples from all deployed drifters as $\bar{O}$. The total number of drifters for a survey is denoted by n_d.

$$\vec{V} = u(x, y, z, t)\vec{i} + v(x, y, z, t)\vec{j} + w(x, y, z, t)\vec{k} \tag{5.4}$$

Given a perfectly known flow field, $\vec{V}$, and an initial location of a point particle, a trajectory of the particle through $\vec{V}$ can be calculated by integrating using the advection equation as a function of time. With unlimited computation time, we could use the perfect flow field model to estimate future trajectories of drifters which have been deployed throughout R and choose the most appealing point for deployment of the real sensor. This simple approach to drifter deployment is, unfortunately, intractable in real systems due to both the vast number of possible deployment points and the high computational cost of solving for each possible trajectory. Moreover, we lack a perfect model of the flow field model, since we only have isolated, in-situ measurements at $\bar{O}$. In addition, since our drifters are not point particles, but floating sensors, we realize that we will only imper-

fectly estimate advection with the help of physical parameters describing drifter shape. These factors make evaluating potential deployment points difficult, but our approach in the following sections seeks to make the problem tractable. First, we discretize the survey space, R, into $m \times n$ square cells of size r, where r is related to the sample validity of the sensors used in the drifters.

Our iterative trajectory planning approach picks new deployment points based on estimated futures developed from observations made by sensors. These observations allow us to update our estimate of the flow field, $\hat{V}$ through data assimilation as described in Section 5.5.2. Instead of calculating trajectories for all points in the flow field, we have developed a process of point proposals (Section 5.6.2) to reduce the number of trajectories that need to be calculated. We assess the trajectories from proposed points using a scoring mechanism that combines the expected improvement of $\hat{V}$ with a parameter that encourages drifter trajectories that observe new points in R. Because we initially had no knowledge of the flow field, we always chose the very center of the survey area as the deployment point for the first drifter.

(a) Visual Drifter **(b)** ASV and Drifter Deployment

Figure 5.8: Hardware description for drifter field experiments.

5.5.2 Flow Field Assimilation from Sensor Observations

Flow information from deployed sensors is collected, assimilated, and used to estimate the value of sampling a location in the region of interest. The assimilation of drifter velocity measurements into a current model has been widely studied in the oceanography

community [Allen and Billing, 1988, Salman et al., 2006, Kamachi and O'Brien, 1995, Tinka et al., 2009, Sun et al., 2017], with most practitioners employing a Bayesian filtering approach to integrating data.

We will also take a Bayesian approach, formulating the estimation of unobserved points as a regression problem and utilizing a Gaussian process (GP) to approximate the flow field for the entire survey area. Gaussian processes properties make them a common tool for modelling spatiotemporal processes as seen in [Singh et al., 2010, Zhao et al., 2016, Kim et al., 2011, Reece et al., 2011]. GPs are non-parametric functions for estimation that do not make strong assumptions about the underlying distribution of the data, making them ideal for the online estimation of previously unseen flow fields. In addition, they give a probability distribution over output that will prove useful when driving exploration.

For this sampling problem, assume we have n sample points $x_i, i = 1, 2, ...n$ and corresponding vector labels of the flow field $y = (y_1, y_2, ..., y_n)$. For a new point x_*, we want to predict $y_* = f(x_*)$. The regression function, f, is a Gaussian process, thus the distribution of the values of f at any finite number of points is a Gaussian distribution.

$$\begin{pmatrix} y \\ y_* \end{pmatrix} \sim N\left(0, \begin{pmatrix} K & K'_* \\ K_* & K_{**} \end{pmatrix}\right), \tag{5.5}$$

where K is the covariance matrix for the labeled points, K_* is the covariance vector between the new point and the labeled points, and K_{**} is the inherent measurement noise. Then,

$$p(y_*|y) \sim N(K_* K^{-1} y, K_{**} - K_* K^{-1} K'_*). \tag{5.6}$$

In this GP, we utilize an exponential kernel (Eq. 5.7) with $\sigma = 1$ and the length parameter $l = 0.2$. These hyperparameters were found experimentally on the training dataset.

$$K(x, x') = \sigma^2 \exp(\frac{(x - x')^2}{2l^2}) \tag{5.7}$$

The *length* parameter regulates how far the GP will extrapolate from an observed data point. The σ parameter is a scaling factor that controls the average distance from the mean.

The current GP model is referred to as $\hat{V}$, the true flow field as $\vec{V}$, and all observed flow field measurements as $\bar{O}$. The covariance matrix of the GP provides us with a measure of uncertainty, U, of the flow field estimate that describes the similarity between every pair of the input points $K(x, x')$. We will use U to encourage exploration into points in the survey region that are not currently well modelled.

5.6 Active Mapping for Coverage

In this section, we consider the problem of collecting measurements in as many cells of an evenly gridded survey area as possible in a limited amount of time using only passive drifters for data observations deployed in the survey region. We incrementally measure and estimate these flow fields in real-time from limited local observations and then harness those estimates to effectively cover a sample space by selecting valuable deployment locations.

5.6.1 Drifter Trajectory Estimates

Trajectory estimates are computed twice each time we search for a new deployment point. The first estimation is performed to find n_f potential future trajectories from the last known point of each deployed drifter. These future path estimates of the deployed drifters are collectively referred to as $\hat{D}$ and will be used to evaluate the proposed deployment point's paths against the futures of drifters which are already deployed. In our experiments, we set $n_f = 4$ and randomly seed a 2-meter location around the last known location of each deployed drifter. This noise in the seed location helps account for errors that will undoubtedly be imposed in the often imprecise automatic drifter deployment process due to mechanical and environmental factors. The second trajectory estimate is

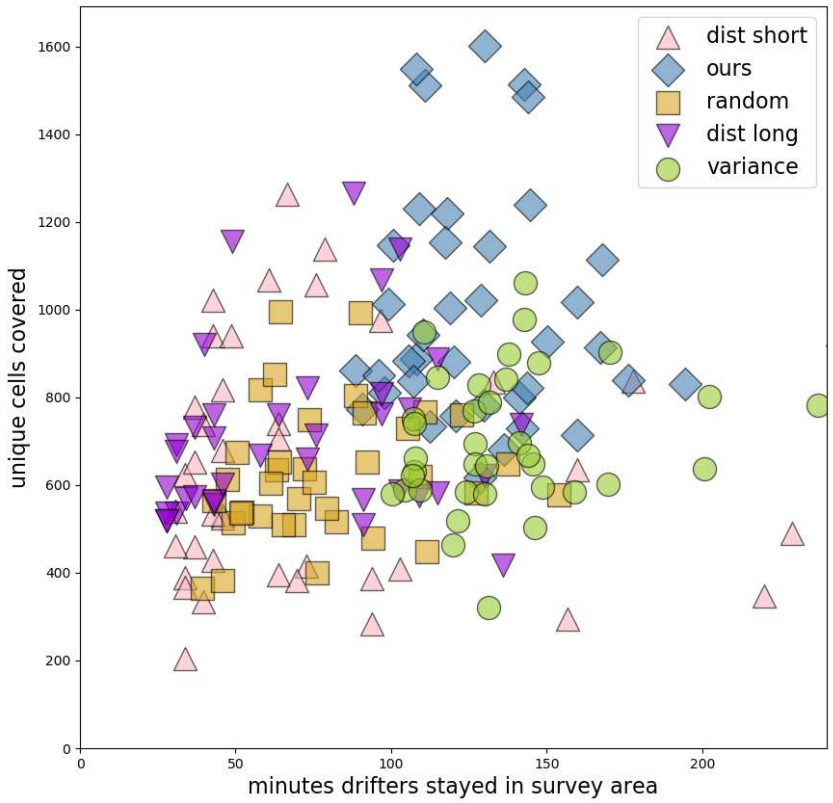
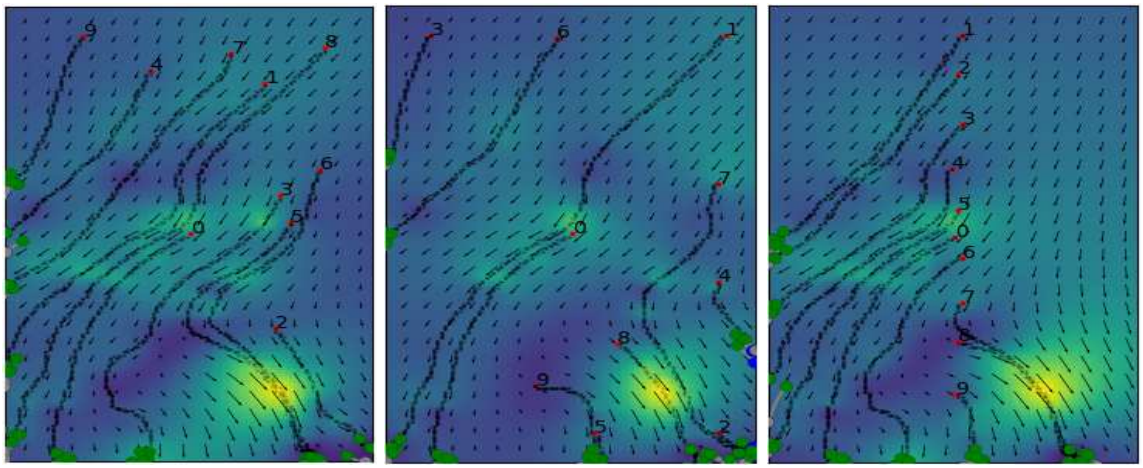


Figure 5.9: Performance comparison over 40 flow field maps as described in Section 5.6.4. Good surveys contain many unique points and have drifters which stay in R for as long as possible (i.e. the top right corner is best). Our deployment scheme performed best in 37 of the 40 schemes we examined.

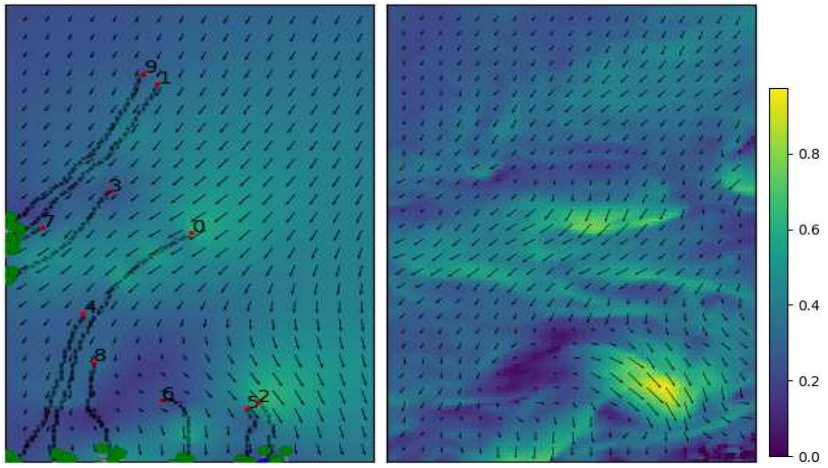
computed over all of the proposed deployment points. These proposal point trajectories, $\hat{P}$ will be scored and ranked as described in Section 5.6.3 to choose the optimal deployment point.

5.6.2 Deployment Point Proposals

We expect flow field estimates of $\hat{V}$ which are near $\vec{V}$ to enable us to more accurately model trajectories. When searching for a new deployment point, we must balance the desire to gain new observations which improve our estimate of $\vec{V}$ while still exploiting what we already know about the flow field to achieve long trajectories which cover as many new points of R as possible. However, estimated trajectories which pass through



(a) ours **(b)** variance **(c)** distributed long



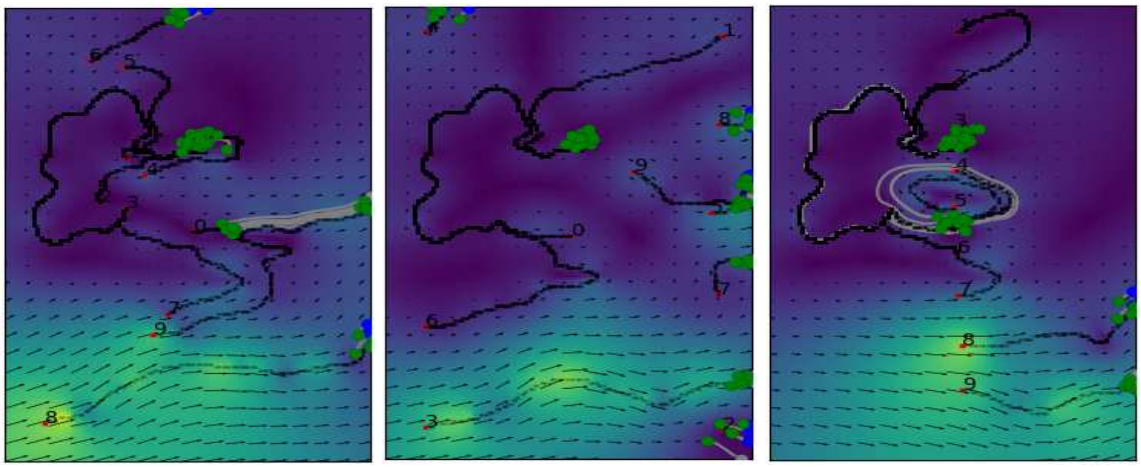
(d) random **(e)** ground truth **(f)**

Figure 5.10: This panel shows the deployments (numbered red points) chosen for ten drifters under four different deployment schemes as discussed in Section 5.6.4. The true flow field for these experiments is depicted in Fig. 5.2 and also shown in Fig. 5.10e. The background and arrows in the experimental figures depict each deployment scheme's estimate of $\vec{V}$ at the terminal state of the survey. The comparative performance of these strategies in this flow field is shown in Fig. 5.12a.

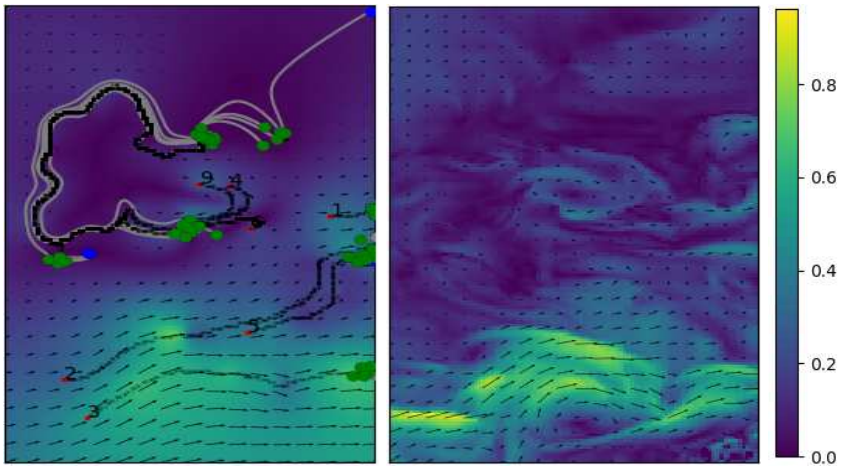
areas where $\hat{V}$ has low confidence (U is high) may be inaccurate, resulting in unpredicted paths which perform poorly. In the following sections, we discuss how we choose d_n to balance our desire to better model $\vec{V}$ with our quest to cover R.

These incentives are realized in a 2D matrix dubbed the *preference field* of size $m \times n$. The preference field, Φ, is initialized to equal a normalized U. A receding safety buffer sets points along the edge of R to zero. This receding safety buffer prevents proposals along the edges of R which may be appealing to our sampler because of high values in U, but could result in short trajectories when the drifter is actually released. The receding safety buffer allows proposals to slowly become riskier as we gather observations of $\vec{V}$. In our experiments, we relate the number of edge cells in the safety buffer to the number of drifters already deployed, n. The number of edge cells zeroed in Φ is calculated by $((n * (\min(m, n) - b_n)/n_d) + b_n)$ for each drifter deployment, with the minimum edge buffer, b_n, chosen to be ten cells. Next, each point in Φ that a deployed drifter has already observed is set to 0. Then $\frac{1}{n_f}$ is subtracted from Φ for each point in $\hat{D}$. Lastly, dilation is performed over Φ which increases the boundaries of the drifter tracks, discouraging proposal points near deployed trajectories.

Rejection sampling is performed on all of the coordinates in Φ until we have acquired 1000 unique points in R. We further prune this set of points using a fast spatial filtering process known as non-maximum suppression (NMS) until we have p_n promising points ($p_n = 100$ in our experiments). NMS works by greedily selecting high-value proposals while deleting nearby proposals which cover the same area [Rosenfeld and Thurston, 1971]. Our proposal approach was inspired by modern object recognition systems in image processing pipelines such as [Ren et al., 2015] in which an object detector proposes many windows around each object in an image, and then the windows are thinned out based on their overlap by NMS to produce the most likely window directly over an object. The proposal points resulting from this process are pictured as yellow points in Fig. 5.13b over Φ.



(a) ours **(b)** variance **(c)** distributed long



(d) random **(e)** ground truth **(f)**

Figure 5.11: This panel depicts the deployments (numbered red points) chosen for ten drifters under four different deployment schemes. All of the methods performed reasonably poorly (see Fig. 5.12b) in this chaotic flow field (Fig. 5.11e) because it contains many eddies and regions with near zero current. The backgrounds (which reference Fig. 5.11f) and arrows in the experimental figures depict each deployment scheme's estimate of $\vec{V}$ at the terminal state of the survey.

5.6.3 Deployment Point Selection

At this step in the deployment pipeline, we have filtered the set of potential deployment points down to a manageable number of points, p. To select the best p, we estimate trajectories for each p under $\hat{V}$ (an example of this step is shown in Fig. 5.13c). Each proposal point's trajectory, $\hat{P}_n$, is scored by summing the unique points in R that are visited with respect to a *value field*, Γ. Γ is the same as Φ, except that the receding safety border is replaced with a narrow, fixed zero-value border, b_n to remove value from risky edge deployments. In our experiments, we found $b_n = 10$ to be reasonable, but this parameter can be adjusted. Proposal rewards are depicted in Fig. 5.13d with yellow markers which reflect the relative score of a point. The proposal reward with the highest value score is selected, and the ASV is instructed to drive to this deployment point to offload the next drifter. In our experiments, the ASV waits for 5 minutes before considering the next deployment.

5.6.4 Drifter Coverage Experiments

To obtain large-scale statistically valid quantitative results with access to ground truth, we report an analysis using archival geophysical flow fields from the Norwegian ROMS dataset [Shchepetkin and McWilliams, 2005] with simulated sensor and vehicle placements.

ours	long	var	short	rand
32	6	2	0	0

Table 5.3: Number of surveys out of the 40 flow fields tested in which a particular deployment scheme sampled the most unique cells from all the baselines.

The data set consists of an array of 100×155 measurements of in-situ ocean current. In the original dataset, each data point was separated by $800m$, but we rescaled the spacing between sample points to correspond to a measurement interval of $5m$ so as to make the survey area small enough to be feasibly traversed by a typical ASV and so that our

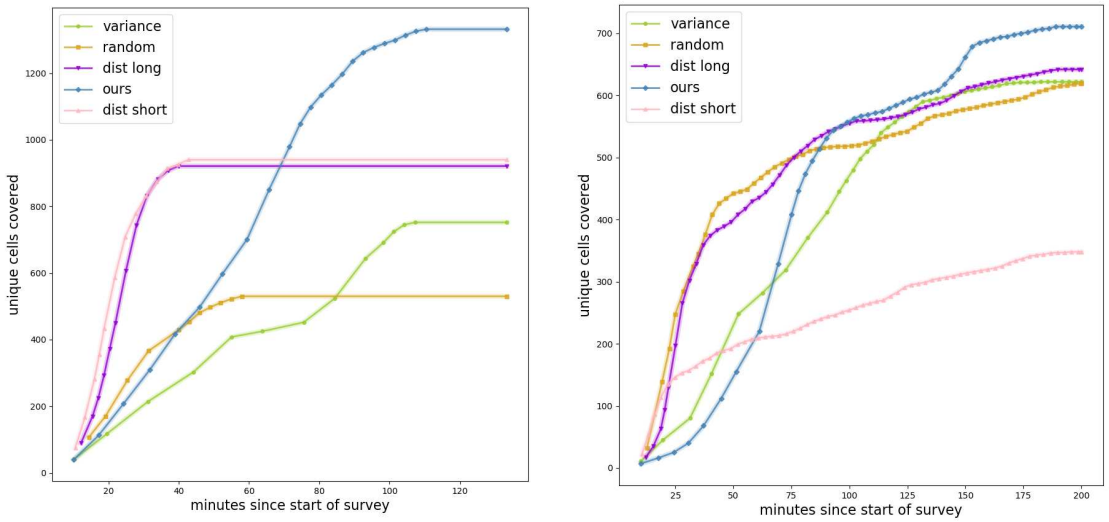
assumptions of a steady flow field were reasonable. This change made our testbed, R, equal to a size of $500m \times 775m$.

For each simulation, we initialize a simulated ASV with an average speed of $3m/s$ to an initial position of $(0, 0)$. The ASV begins each survey with n_d=10 drifters and must decide where to place each of them. When the ASV chooses a deployment point, our simulator generates ten plausible paths based on the ground truth flow field that are 8 hours in length from the requested deployment point and randomly assigns one of these paths to the newly "deployed" drifter. The boat pauses for 30 seconds to release the drifter before considering the deployment point for the next drifter. Each released drifter will sample the flow field at twice the Nyquist rate for the flow field until it exits R. In these experiments, we add a 1e-6 normal noise factor to each current observation. We assume that neither the flow field nor the drifter load affects the speed of the ASV. An experiment terminates when all drifters have left R or after 8 hours.

Results from the experiments described in this section can be seen in Fig. 5.9. Our approach covered the most unique cells in 92.5% (37 out of the 40) flow fields. Variance, distributed long, and distributed short each were the highest-performing deployment schemes on one of the 40 flow fields. The flow fields in which our approach struggled contained eddies and regions with low currents that were difficult to model correctly. See Fig. 5.11 for an example of a difficult survey area. Although the variance-based deployment occasionally produced a better estimate of $\vec{V}$ in a shorter amount of time, it often squandered drifters by deploying them at points that guaranteed they had short trajectories.

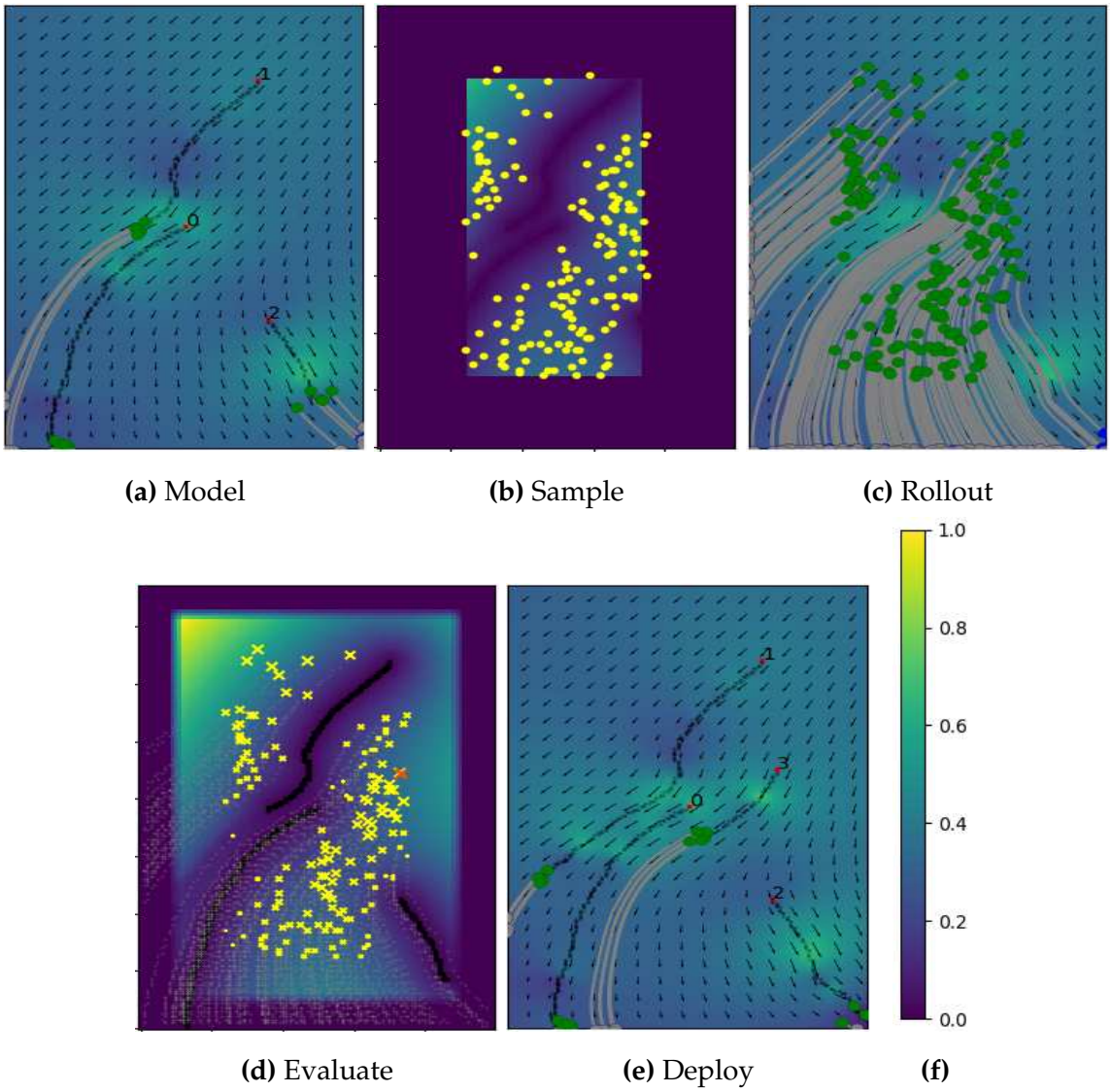
Fig. 5.13 shows a visual representation of the drifter deployment pipeline. The true flow field is shown in Fig. 5.2; however, at the decision point depicted in Fig. 5.13a, we have only observed a small portion of the survey area. Fig. 5.13a shows $\hat{V}$ for the survey after three drifters have been deployed. This can be compared to the true flow field $\vec{V}$ which is pictured in Fig. 5.2. The flow field estimate is closer to the true flow field near where the drifters have sampled and regressed to the mean far away from data points.

These flow field observations, $\bar{O}$, are shown with black dots. Initial deployment points are shown by numbered red dots. These current observations were assimilated into an estimate of the flow field, $\hat{V}$, which is shown in the background of Fig. 5.13a. We use $\hat{V}$ to estimate possible future trajectories (shown in grey points in Fig. 5.13a) from the last observed position for each deployed drifter (green dots). These future trajectories, $\hat{D}$, are combined with previously observed points, $\bar{D}$, uncertainty, U, and an edge mask to build a heuristic field over R that describes a preference for deployment points. From this *preference field*, Φ, (seen in the colormap of Fig. 5.13b), we sample n_p proposal points, p, (yellow points) for evaluation. Next, we estimate possible trajectories for these points with $\hat{V}$ as shown in Fig. 5.13c. In Fig. 5.13d, we evaluate the trajectories (light grey) of p against the *value field*, Γ, which is depicted in the colormap. Γ is constructed from a combination of U, $\bar{D}$, $\hat{D}$, and a safety buffer, rewarding long trajectories which traverse unseen points. Also in Fig. 5.13d, we show the relative score of the proposal points by the size of their yellow markers. The highest scoring point is marked in red and becomes the deployment point. In the final map in this panel, we show the new $\hat{V}$ after deploying the fourth drifter. The colormaps in Figures 5.13a, 5.13c, and 5.13e colormaps refer to Fig. 5.2's colorbar for flow speed in m/s. The colormaps for Figures 5.13b and 5.13d refer to the colorbar in Fig. 5.13f.



(a) Typical Flowfield **(b)** Chaotic Flowfield

Figure 5.12: This figure depicts comparisons of deployment schemes on a typical and challenging flowfield over time. A higher number of unique cells covered is desirable and we see that our method out-performs baselines, sampling more unique cells during the runtime of the batteries. Fig 5.12a shows performance on the flow field visualized in Fig. 5.10 over time. Fig. 5.12b shows results on a chaotic flowfield visualized in Fig. 5.11.



(a) Model **(b)** Sample **(c)** Rollout

(d) Evaluate **(e)** Deploy **(f)**

Figure 5.13: This sequence of figures details the steps performed to determine a deployment point for a fourth drifter into a partially observed flow field.

5.7 Collaborative Sampling with Drifters

When spatial precision is necessary, environmental surveys of coastal areas traditionally require sophisticated (costly) robotic vehicles or significant human effort to gather the samples to effectively model a region. Although the drifter-driven sampling system presented in Section 5.6 introduced a method for increasing coverage with low-cost drifters, passive sensors controlled only at the deployment point are not always practical for actively collecting samples from particular phenomena in particular flow fields. We extend this work to enable observations from the drifters and the ASV, enabling both morphologies to exploit their strengths. The ASV adaptively samples expected information-rich regions, while the drifters are strategically deployed to maximize long-term exploration. We investigate the utility of this mixed-modality surveying scheme and provide empirical results from ocean flow simulations.

5.7.1 Adaptive Sampling

Although complete sampling of regions at a higher than Nyquist frequency is almost always ideal, practical constraints often limit the time or resources that can be used to collect data in real systems. However, in some instances, such as those in which high-variable data clusters in spatial regions, an adaptive sampling approach can yield similar modelling errors with a significant reduction in the time/energy resources needed [Das et al., 2015b, Low et al., 2008, Rahimi et al., 2005b, Singh et al., 2007, Fiorelli et al., 2006, Chadwick et al., 2016].

We demonstrate the utility of a non-uniform sampling technique for exploiting expected high-information regions of a survey area using a low-resolution survey by the ASV as a prior [Manjanna and Dudek, 2017] and samples gathered from randomly deployed drifters as a prior in [Manjanna et al., 2017a]. In [Hansen and Dudek, 2018], we found drifter deployment points that optimized for survey coverage. In this work, we combine our previous work into a comprehensive technique for optimizing modelling er-

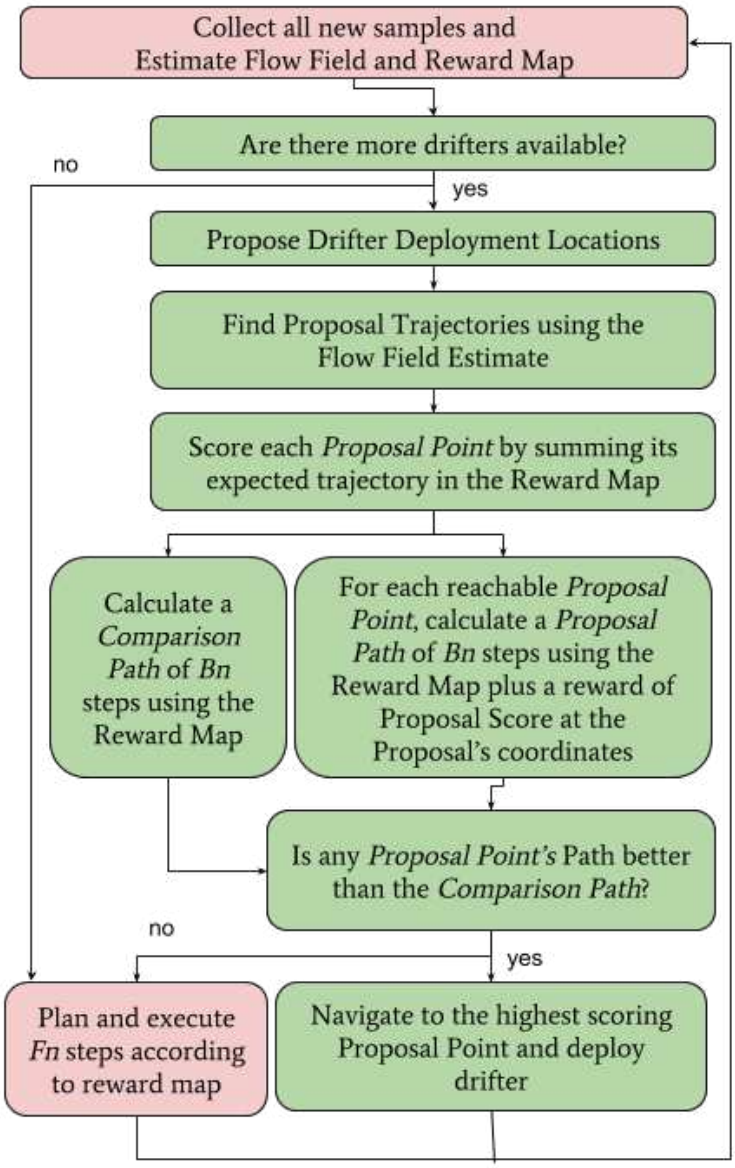


Figure 5.14: Adaptive sampling with drifters: planning process

ror with an ASV which adaptively samples and deploys drifters so as to split the burden of spatial sampling. Others have shown that drifters can be used to exploit the unique nature of flow fields to transport drifters for sampling, search, and exploration [Meghjani et al., 2016, Shkurti et al., 2012, Alam et al., 2018, Aoyagi et al., 2004]. In [Das et al., 2012], the authors present an approach to multi-day sampling of time and spatially varying oceanographic phenomena in which drifters inform autonomous underwater vehicles of the movement of features of interest. Like our approach, others have utilized flow fields

for tracking features of interest [Kularatne and Hsieh, 2015] and for generating informed paths [Kularatne et al., 2018, Inanc et al., 2005, Kwok and Martínez, 2010]. To survey a large-scale ocean front with a team of gliders and surface vehicles, [McCammon et al., 2021] employs a Monte Carlo Tree Search planner that leverages knowledge about the deployed environment by performing state estimates with a Gaussian Process that has access to an informed nearest neighbour prior and a drifting reference frame.

We assume that all N_d drifters used in the experiment are carried on the ASV and can be autonomously deployed in D_p seconds (where $D_p = 10$ in our experiments). The deployment of a drifter takes time and energy to travel to an appealing launch location, but thereafter, the data collected from the drifter comes at no cost to the ASV.

The ASV and all drifters can collect compatibly, transformable samples from $1)$ the specified *phenomena of interest* and $2)$ from the flow field at the same fixed depth. In this project, we select the *phenomena of interest* to be the flow field itself for the purpose of improving visualization, however, we can easily optimize for different phenomena such as oxygen, visual observation, or temperature.

Formally, we are considering the problem of physically collecting point measurements over a defined marine region, S, with one ASV and a number, N_d of marine drifters so as to reconstruct the spatially distributed *phenomena of interest*, I, as accurately as possible with minimal time invested. At each update, we assimilate samples that have been collected by the ASV and deployed drifters.

5.7.2 Adaptive Path Planning

To drive exploration and mapping into areas of expected high information gain, we use the uncertainty estimate from the assimilation step, R, as a reward function for finding the value of sampling each cell in S. We utilize Value Iteration, an approach for finding optimal policies as defined by the Bellman equation in a Markov Decision Process [Bellman, 1957]. The value of sampling each cell from a given state is described by V^* (Equation 5.8). Once V^* is found, the optimal policy π^* from a given state (s) can be found by tak-

ing the action a with the largest value. In our experiments, we set the discount factor, $\gamma = 0.95$.

$$V^*(s) = \max_a \left(R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^*(s') \right), \tag{5.8}$$

$$\pi^*(s) = arg\max_a \left(R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^*(s') \right) \tag{5.9}$$

Ideally, for full Markovian guarantees, after each sample is collected, data assimilation would be recomputed and a new reward map calculated; however, this is computationally intractable. Empirically, we have found that performing data assimilation update every $F_n = 15$ step works well in practice. We negate the reward map at the coordinates of each step as it is planned with Value Iteration as an approximation to the full recomputation as described in [Manjanna and Dudek, 2017].

5.7.3 Strategic Drifter Deployment

We model trajectories both for proposed drifter deployment points and for already deployed drifters. The estimated paths of deployed drifters populate the dynamic Reward Map, R which ultimately helps the model decide on the next deployment point.

For each deployed drifter, we seed n_f points ($n_f = 5$ in these experiments) in a $2\,\mathrm{m}$ radius around the last known location and use OpenDrift to find possible future trajectories so we can update R to reflect the best estimate of where the drifters will travel. For each deployed drifter, $\frac{1}{n_f}$ is subtracted from R at each point in the estimated future trajectories and is used in path planning and finding proposal points. The expected future trajectory of a newly deployed drifter is shown in gray in Fig. 5.15b and is reflected in the R which corresponds to the background of Figures 5.15d and 5.15e.

It is computationally expensive to predict drifter trajectories, so we use a method of reducing the number of points to evaluate as originally described in Section 5.6.

Trajectories are found for the top proposal points by simulating paths from each of the points for the expected battery life of the drifters (4 hours in these experiments) with our best estimate of the flow field, $\hat{V}$ (shown in Fig. 5.15a). The resulting trajectories are scored by summing the expected sampling points in R. The highest scoring proposal reward (the top 10 of the 20 evaluated in the experiments presented here) is then passed to the ASV decision process for consideration (their respective size shows proposal reward in the pink xs in Fig. 5.15b).

5.7.4 Decision State: Drifter or Drive?

While the ASV still has undeployed drifters, it must decide at each update step whether to deploy a drifter at a proposal point or adaptively sample the area. Our system provides a variable from which to control this decision point, B_n. At each decision point, the ASV plans a *comparison path* using Value Iteration of B_n steps and finds the total score that would be achieved with the R through those points. Alternatively, for each proposal point, the ASV calculates a path of maximum length B_n, which travels through the point and finds the score of this path plus the expected value of the proposal point, which was calculated in the previous step. This can be thought of as planning a non-optimal path to the proposal point, but with a bonus of earning the full trajectory of the drifter for free. If any of the proposed paths score higher than the comparison path, then that path is executed up to the proposal point, and the drifter is deployed.

If the comparison path scores better than any of the proposal paths or if there are no drifters left to deploy, then F_n steps are executed. At this point, data assimilation is restarted and the process repeats itself, as seen in Fig. 5.14.

5.7.5 Collaborative Drifter Experiments

To obtain quantitative results under ground truth, we evaluate our approach using 25 archival flow fields from a Regional Ocean Modeling Systems (ROMS) dataset with sim-

ulated sensor and vehicle placements. This data set consists of an array of 100×155 measurements of in-situ ocean current. We rescaled the original dataset from $800 \times 800\,\mathrm{m}$ grid cell size to an interval of $5 \times 5\,\mathrm{m}$ in order to make the region feasibly traversable for a typical battery-powered ASV. As a result, we compared our results over 40 different ocean flows over a region of size $500 \times 775\,\mathrm{m}$.

Our simulated ASV kept an average speed of $1\,\mathrm{m/s}$ and is capable of choosing 1 of 8 actions corresponding to movement into adjacent cells at each time step. We assume that neither the flow field nor the drifter load affects the speed of the ASV. Our simulation allows the ASV to operate for 2 simulated hours after the experiment begins.

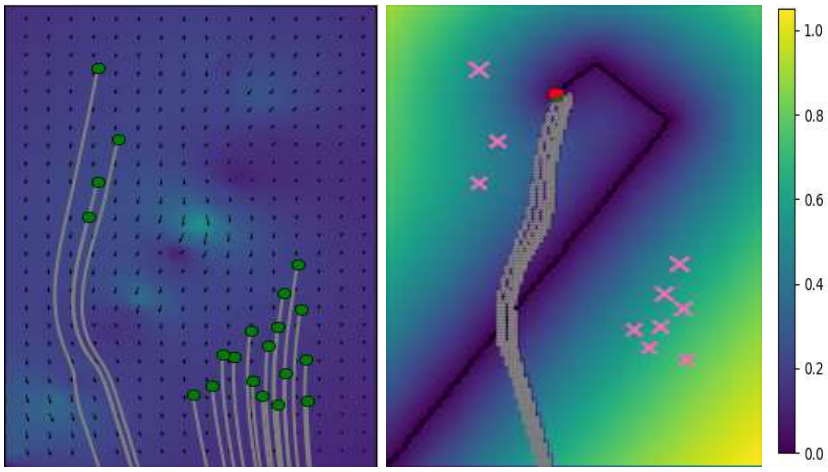
Each released drifter will take a sample every 5 seconds until it exits S or its time limit expires. In the simulations depicted here, the drifters collect data for 4 hours after their initial release. The entire experiment terminates when all deployed drifters have left S or have expired, though the boat stops sampling after it reaches its time limit.

Each adaptive experiment starts the same way, with the ASV at coordinate $(0, 0)$ and driving 75% of the way to the diagonal corner of the survey area. After reaching this point, the first update is run, complete with data assimilation, deployment proposals, path comparisons, and finally the next path.

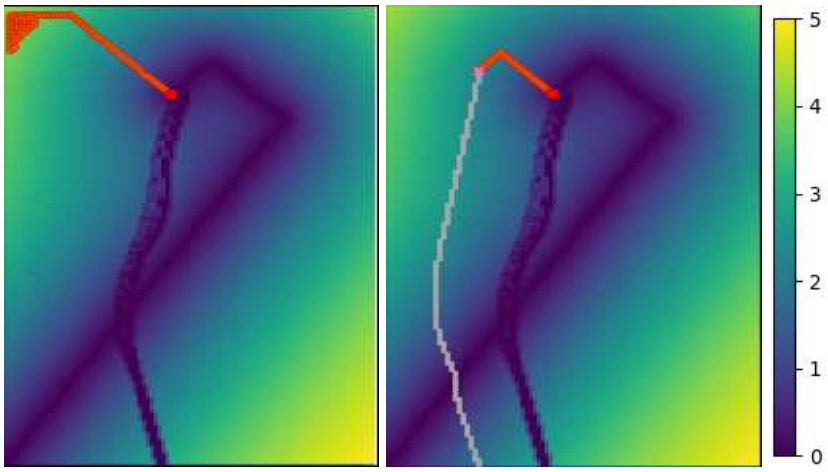
In Fig. 5.17a we show results from all 25 simulated flow fields. Our system has an initial advantage over the ASV-only deployments, as it is able to achieve a more comprehensive model faster. As expected, however, after appropriate survey time, the ASV-only experiments sufficiently covered the region in a more complete manner. The selection of B_n for a survey will determine how selective the ASV is when deploying drifters. We see in Fig. 5.17a that surveys in flow fields with more variability perform better without drifters of large B_n. In the future, we hope to learn and adapt this parameter during surveys.

We have also conducted field experiments (see Fig. 5.8) which demonstrate the proof of concept of our systems. We were able to successfully communicate with distributed

sensors over WiFi and assimilate flow data collected from GPS measurements. During the field experiments, we lacked an automated deployment mechanism.

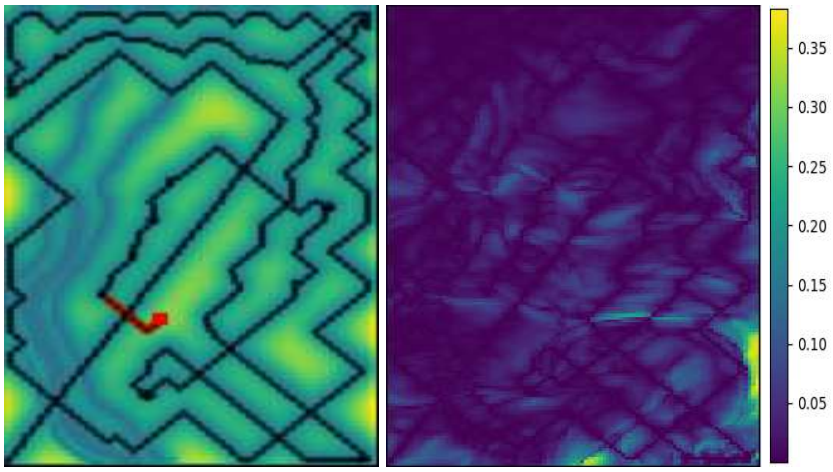


(a) Proposals **(b)** Decision State **(c)**

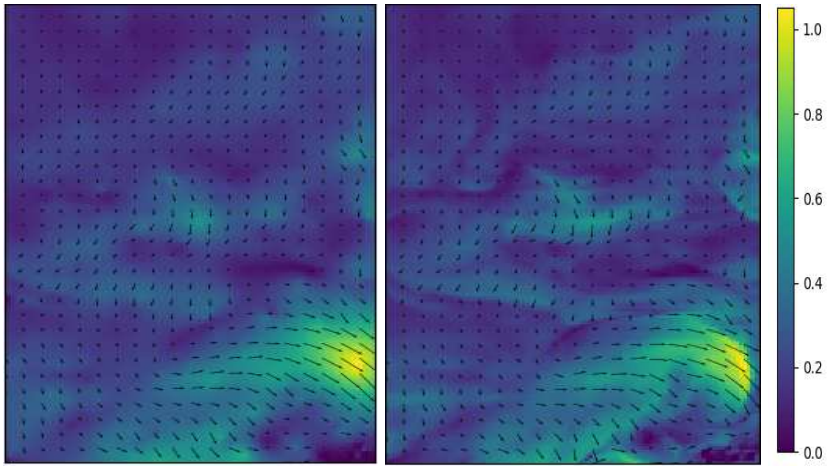


(d) Adaptive Comparison **(e)** Deploy **(f)**

Figure 5.15: This figure demonstrates a decision point in which the ASV considers the deployment of a second drifter into the flow field 16 minutes after starting the experiment. This experiment used 5 drifters with $F_n = 5$, and $B_n = 90$. The current estimate of the flow field is depicted in the background of Fig. 5.15a with the colorbar in Fig. 5.15c representing estimated velocity. Drifter proposal points near the current position of the ASV are depicted in Fig. 5.15a with their expected trajectories, given the current estimate of V. The experiment state is depicted in Fig. 5.15b with the ASV's current position shown with a red square marker. The ASV has just finished deploying a drifter, which has an expected trajectory plotted in gray. The Proposal Points are plotted as pink x markers with their size correlating to their expected value along their trajectories. The reward map is plotted in the right three plots with their colorbar shown in Fig. 5.15f. Figures 5.15d and 5.15e with expected values of 304.64 and 802.76 show the two paths considered in red. In this case, the second drifter was deployed.

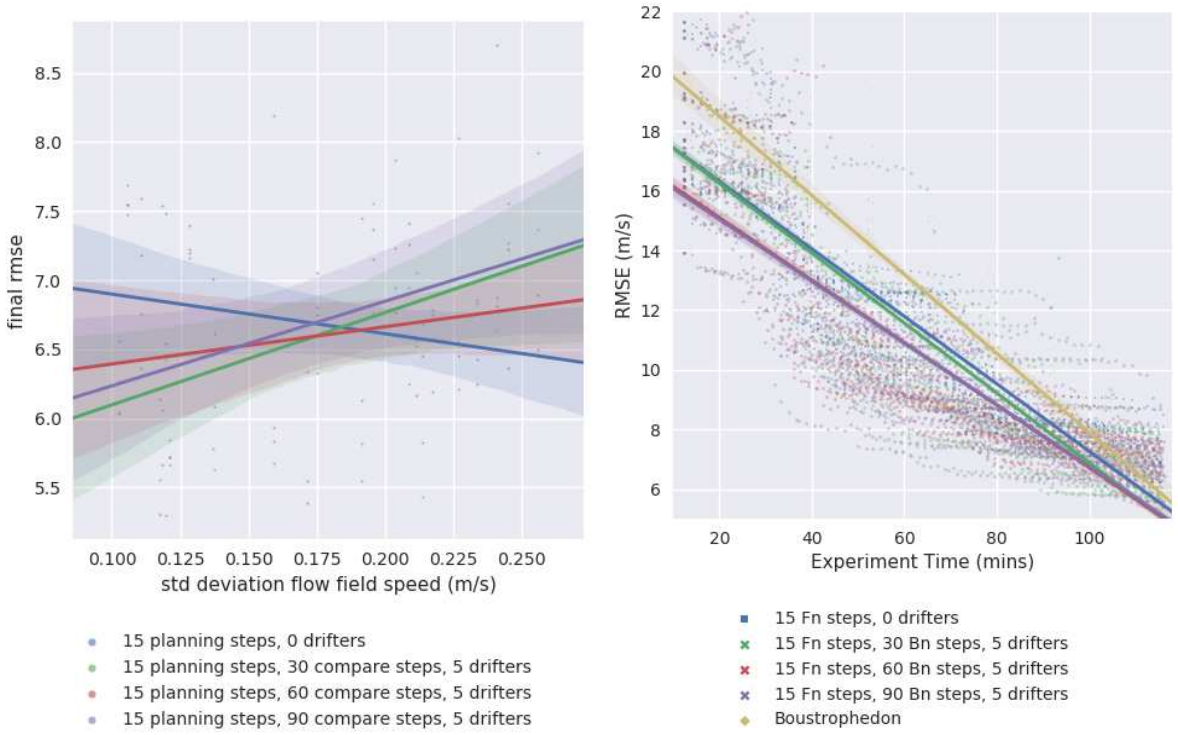


(a) Team Finished Path **(b)** Team Final RMSE **(c)**



(d) Final Estimate **(e)** True Flowfield **(f)**

Figure 5.16: This series depicts an ASV and drifter team sampling a basin. Fig. 5.16a shows the full boat path (black points) at the end of an experiment (last position shown in red) and all sampled positions (violet). The background is the current *Reward Map*, R, which references the colour bar shown in Fig. 5.15f. This experiment used 5 drifters with $F_n = 5$, and $B_n = 90$. Fig. 5.16e is the true flow field, and Fig. 5.16d is what the experiment estimated the flow field to be at the end of the experiment. The backgrounds of Figures 5.16e and 5.16d refer to speed in $\mathrm{m/s}$ in the colorbar seen in Fig. 5.16f. Fig. 5.16b shows the Root Mean Square Error (RMSE) in $\mathrm{m/s}$ of Fig. 5.16d with respect to the ground truth. The colorbar in Fig. 5.16c provides the metric for Fig. 5.16b.

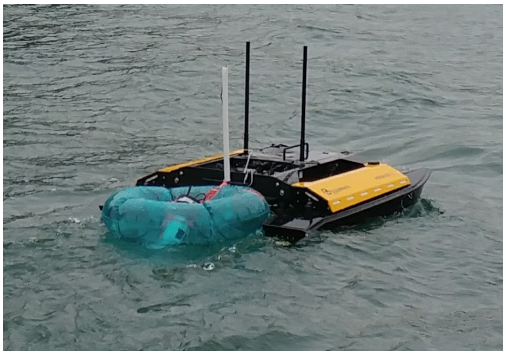


(a) Final RMSE

(b) Online RMSE

Figure 5.17: Fig. 5.17a shows the resulting final RMSE after the heterogenous ASV team sampled for 2 hours. Drifter data was integrated until the drifter exited the survey area or expired. Our approach outperforms the adaptive planning approach when the flow field is more predictable. However, when the flow field is less predictable, drifters are sometimes placed in sub-optimal deployment points due to a poor estimate of the true data. Fig. 5.17b shows the RMSE at each point in which data was assimilated for all 25 test maps. The lines indicate the trend found with a first-order, bootstrapped regressor ($n = 1000$) with shading showing the 95% confidence interval. This plot demonstrates the advantage of using a tuned adaptive drifter system to effectively model a region quickly. Given enough survey time, the different schemes converge.

5.8 Improving Drifter Hardware



(a) Heron ASV

(b) Gannet ASV

Figure 5.18: Fig. 5.18a shows the Clearpath Heron ASV maneuvering a drifter to a new position. Fig. 5.18b shows our Gannet ASV collecting bathymetry data of a coral reef.

Two small boats were used in our marine experiments: the Heron and Gannet. The Clearpath Heron Autonomous Surface Vehicle (ASV), shown in Fig. 5.18a, is a battery-propelled catamaran equipped with observational sensors for capturing features of a body of water from the surface. The Gannet ASV (Fig. 5.18b) is a floating vehicle with a differential drive thruster configuration. The Gannet, built by our team at McGill, enables us to collect controlled spatial measurements along the surface of a body of water using completely off-the-shelf components. It also allowed for a larger payload and endurance than the Heron ASV. Onboard electronics are stored in a waterproof bin situated between two kayak catamarans and can be customized with a variety of sensors to log or stream geo-located sensor observations including video (underwater or above-water), echo soundings, and various environmental measurements.

We created a system that enables autonomous surface vehicles to deploy and use non-actuated drifters for marine surveys. Our design, known as Buoy-Acquisition Boat Equipment (BABE), allows for long-term surveys with diverse spatial measurements. BABE allows for surveys to be launched with little operational overhead due to the complementary functions of the ASV and drifters. Our system includes scalable storage tracks, a

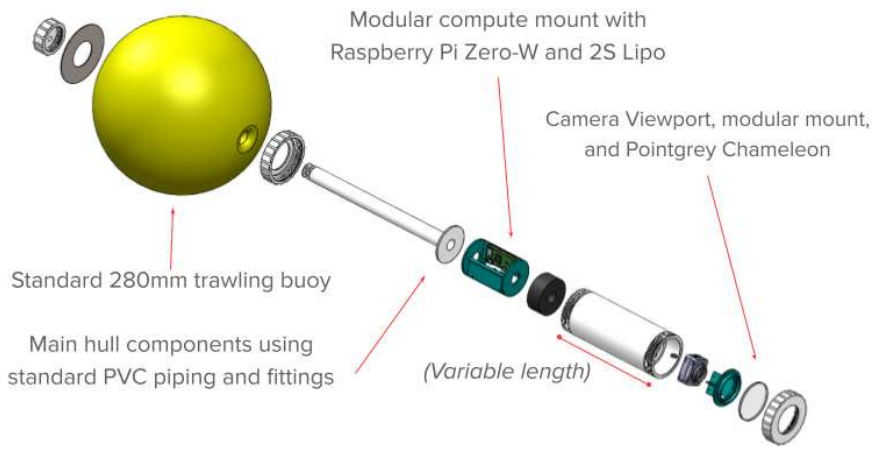


Figure 5.19: BABE drifter with hull design and internals.

capture mechanism that sorts incoming drifters, and a deployment mechanism to release drifters into the survey area.

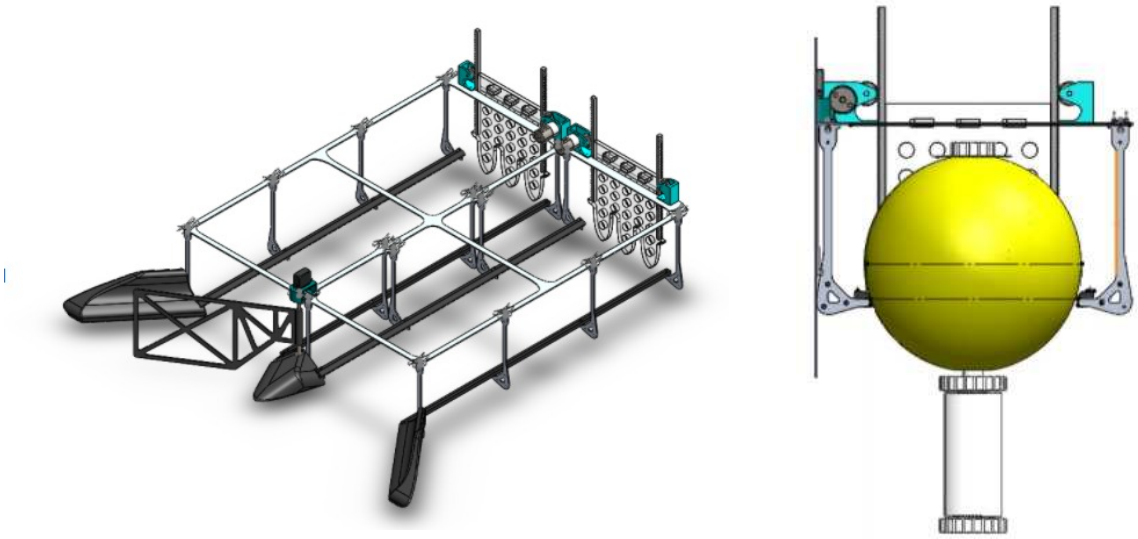
The BABE drifters are designed to fit into the rails track of the ASV with a buoy providing both flotation and visibility. The buoy is sourced from polyvinyl chloride (PVC), a standard material in the marine sector that is also easily obtainable in the consumer market. A central bore allows for communication and power transfer throughout the system while allowing radio-based wireless communication above the water.

Several key objectives drive the design of the BABE system, including ergonomic deployability in the field by a single operator, robust capture and deployment mechanisms for high-confidence autonomy, and a fail-closed design to further bolster this confidence. Figure 5.20a displays the core mechanism and Figure 5.19 shows the drifter platform.

Our design is driven by the following principles:

1. **Deployable**: This system is meant to be mounted on a human portable ASV. At 3kg, the BABE ASV can be fielded by a single operator and can be completely disassembled to enable transport by air.

2. **Reliable**: The entry and exit points of the tracks are fitted with one-way gates inspired by the paddles of a pinball machine. The capture mechanism is simple, and



(a) Isometric view

(b) Front view

Figure 5.20: BABE empty isometric and front view of a single track with a stowed drifter. The system is configured with two tracks to support a total of six drifters (Fig. 5.19).

the remainder of the drifter storage system is passive, using the inertia and drag of the drifters against the movement of the ASV to achieve movement onto the tracks.

3. **Fail-closed design**: Reduce the risk of drifter loss if power loss occurs on the ASV.

4. **Extendable**: Additional tracks can be added for more sorting options upon capture and deployment.

The gates are actuated and able to sort the buoys upon capture, which allows advanced manipulation of the buoys. For instance, one track can be used for stowing drifters with depleted batteries, while another track is used for capturing and re-deployment. Fig. 5.20a shows a canal-style gate that completely blocks the exit of each track.

BABE storage racks are slightly submerged below the waterline of the paired drifters (Fig. 5.19) and made of a slippery plastic to allow the drifters to be propelled along the chute by drag as the ASV moves through the water (see Fig. 5.20).

5.9 Discussion and Retrospective

We investigate relationships between survey time, energy, and modelling error and present a tunable algorithm for selectively choosing when to add additional drifters to the survey. We also show when drifters are not useful such as when there is an unstable or low-velocity flow field or when survey time is not limited. Our work exploits the use of structural bias and physics to develop a robot team composed of unactuated robots that is capable of productively surveying in flow fields.

Practical environmental surveys require trade-offs between cost, mobility, and spatial or temporal resolution. We can effectively observe environmental phenomena by exploiting the efficiencies of both active and passive sensor platforms. We show that the proposed heterogeneous system selectively samples these environments to achieve faster modelling results in many scenarios.

It is also essential to consider the added complexity of adding drifters to a survey team. Each additional sensor adds more hardware that must be maintained and repaired. In addition, if drifters are to be reused, they must be physically retrieved at the end of the experiment. Although they will typically have long battery life, drifters almost always cover a region more slowly than actuated vehicles.

In the next few chapters, we transition to a discussion of structured modelling in the context of tabletop manipulation.

Part IV

Learning with Structure-Induced Priors

Chapter 6

Leveraging Kinematics for Learning Manipulation Policies

Forward kinematics, as described by the Denavit-Hartenberg (DH) [Denavit, 1955] parameterization for rigid-body robots, is fully differentiable with respect to input joint angles, given fixed link-relative geometric information, and including this differentiable module improves the training of reinforcement learning agents on an array of benchmark manipulation tasks. This work exploits this fact and introduces a method for learning manipulation agents incorporating robot forward kinematics as a differentiable module for reinforcement learning systems. Forward kinematics, as described by Denavit-

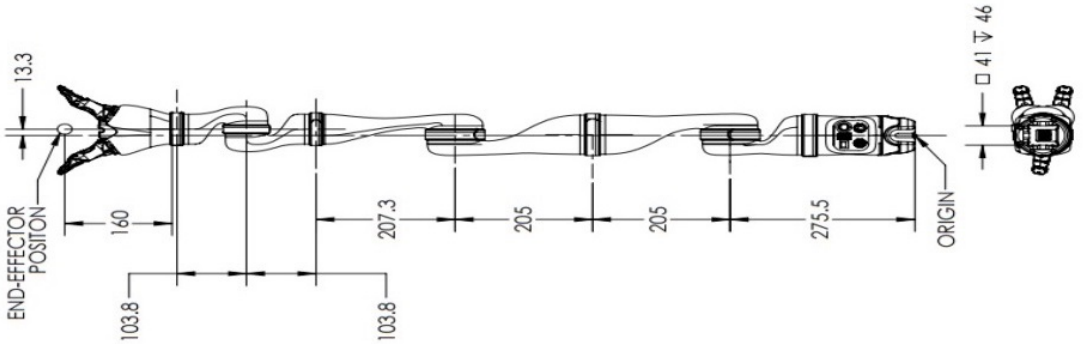


Figure 6.1: DH parameters used for forward kinematics for the Jaco 7DOF Robot as given by the manufacturer, Kinova.

Hartenberg (DH) parameterization for rigid-body robots, is fully differentiable with respect to input joint angles, given fixed link-relative geometric information. Including this differentiable module into the structure of a reinforcement learning agent improves training speed, stability, and overall performance. By incorporating this information only in the critic, the final learned policy used to predict joint actions from image input *does not directly depend* on receiving input joint information, instead learning the necessary behaviour implicitly via the kinematics-informed critic. We illustrate this approach by modifying the critic in a modern pixel-based actor-critic baseline to be a Kinematic Critic and ablating across variations that provide similar pose information but without the kinematic bias in the network architecture. Results are given across several manipulation tasks and two robot arms in Robosuite [Zhu et al., 2020]. We additionally demonstrate a simulation-learned policy running on a real Jaco 7DOF robot.

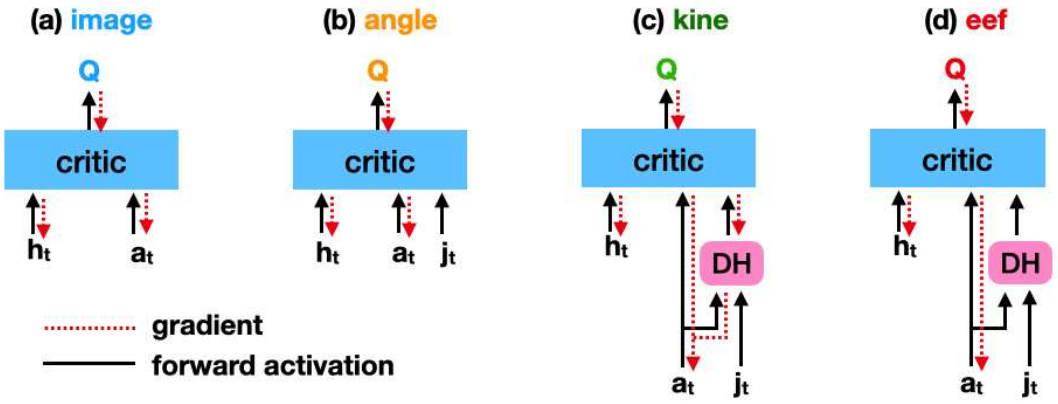


Figure 6.2: Diagrams illustrating the 4 critic architectures tested in this chapter. The (a) ***image*** variation represents the standard DrQv2 critic setup, the (b) variation shows the critic network with joint ***angle*** directly, (c) is ***kine*** method, which exploits the Denavit-Hartenberg parameterization of forward kinematics to provide a lightweight, differentiable model of the end effector pose to the critic, and finally, in ***eef***, we test the importance of differentiability through the kinematic function by removing the gradients while still giving the critic access to the end effector pose. Critic setups (b), (c), and (d) are a core contribution to this work, and we study the performance of each variant in Section 6.3

This chapter incorporates the classic Denavit-Hartenburg (DH) [Denavit, 1955] method of parameterizing robot kinematics as a differentiable function for improving reinforcement learning agents on several robotic manipulation tasks. The DH method of joint parameterization was developed in 1955 by Jacques Denavit and Richard Hartenberg [Denavit, 1955] and has been a staple tool for defining kinematic functions for rigid-body robots.

Robots learning to solve manipulation tasks must inherently reason about controlling their own body. Models and controllers based on explicit analytic physical parameterization traditionally need detailed information such as manipulator redundancies, kinematic limits, friction, acceleration, and/or inertia, which can be difficult to define exhaustively and measure accurately. When solving control tasks without explicit information about the robot's kinematics, such as model-free reinforcement learning (RL), dynamics and structure prediction are inherently coupled with task-solving in the agent. In this chapter, we argue for a middle ground between fully human-defined controllers, which require defining parameters such as friction and inertia, which are often difficult to accurately estimate, measure, or simulate, and fully learned agents who do not know the robot they are tasked with controlling. Our approach incorporates the easily defined and (typically) constant factors of robot geometry into the learning pipeline, resulting in better overall learned control policies.

Recently introduced automatic differentiation tools such as Pytorch [Paszke et al., 2019] and Jax [Bradbury et al., 2018] enable easy incorporation of the DH-defined kinematics function (and its Jacobian) into deep learning models with full gradient propagation. Recent innovations have illustrated the appeal of incorporating differentiable functions [Riba et al., 2020] into learning systems or building entire environments [Hu et al., 2020, Jatavallabhula et al., 2021, Gradu et al., 2021] as a method for improving the learning of complex functions which have differentiable physical simulators. Our approach maintains much of the simplicity of model-free RL while harnessing well-defined robot

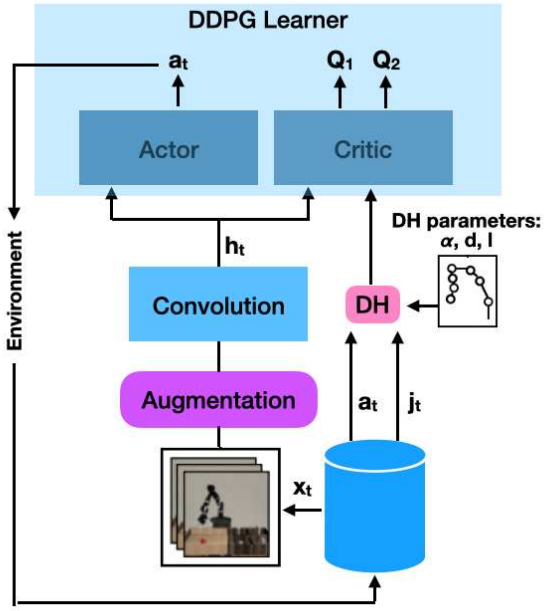


Figure 6.3: In the Kinematic Critic architecture, we differentiate through a function expressed using Denavit-Hartenburg kinematics during training. The DH function is given the constant α, d, and l parameters as well as the joint angle state, j_t, and relative angle action, a_t, which are added together to form θ in Eq. 6.2. Note that the sampling of action a_t admits differentiation with respect to the action distribution predicted by the actor, using the reparameterization trick [Kingma and Welling, 2014, Williams, 1992a].

kinematics as a differentiable structural bias without the overhead of fully differentiable environments.

The main contributions of this chapter are as follows:

- We describe a method to incorporate forward kinematics with the Denavit-Hartenberg (DH) function in an automatic differentiation framework for use in a deep reinforcement learning algorithm.
- We demonstrate notable improvement on a suite of robot manipulation benchmarks in simulation on Jaco and Panda robots over a standard pixel-based actor-critic algorithm.

- We show the importance of differentiating through the kinematic function over architectures that provide the same joint information without the structural bias of kinematics.

6.1 Background

Kinematics

Forward kinematics allows computing a robot's kinematic chain for a particular configuration to find the pose of the end-effector from *joint parameters* and joint angles, where *joint parameters* are known constants that define the geometric relationship between the serial links of a rigid body chain. There are several conventions for assigning joint parameters [Rocha et al., 2011], but in this work, we only explore the Denavit-Hartenburg [Denavit, 1955] parameterization method. DH is advantageous because it is easily calculable, is often provided by default by robot manufacturers, and is differentiable with respect to input joint angles, given geometric information such as lengths and relative link rotations (see Fig 6.1).

For each joint, i, in a rigid body, the DH parameters are described by d_i (distance from joint i to the actuator axis $i-1$), θ_i (angle rotation about axis $i-1$), a_i (the distance of the joint i along actuator axis $i-1$), and α_i (the angle between actuators of axis i and axis $i-1$).

In this convention, coordinate frames are serially assigned to two links between a joint such that the first transformation is associated with the joint (Z). The second is assigned to the link (X) such that the full kinematic equation assigned to a robot with n links can be described in minimal joint parameters by the transformation T in Eq. 6.1.

$$T = Z_1 * X_1 * Z_2 ... Xn - 1[Z_n][X_n] \tag{6.1}$$

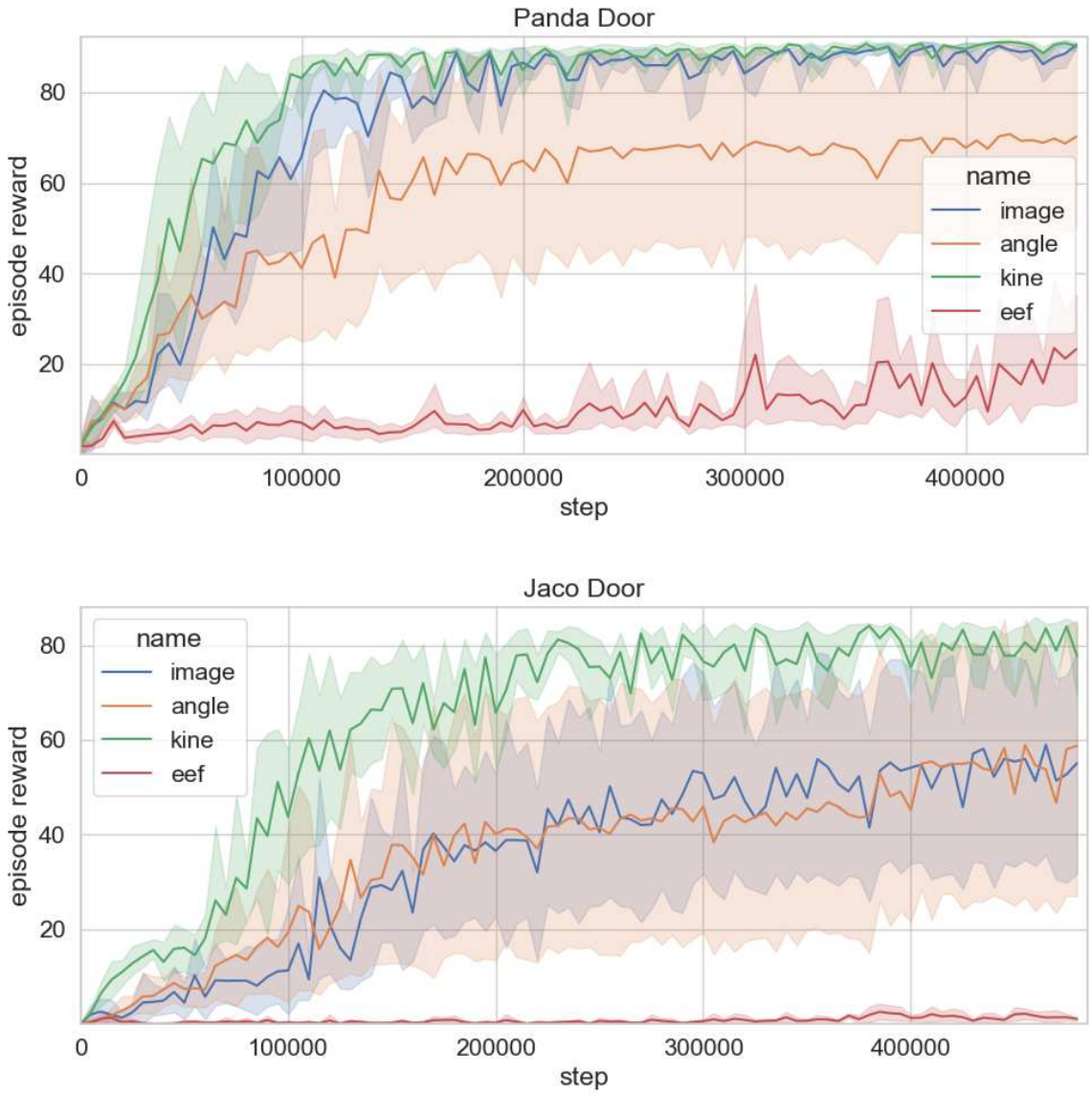


Figure 6.4: Evaluation curves depicting the mean (solid line) and the shaded 95 percent confidence interval around the mean, with performance measured over 5 randomly chosen seeds. The Kinematic Critic architecture (green) outperforms other ablations on average, including agents with critics which directly calculate the expected end-effector pose without back-propagating gradient information (red) and agents which are given robot joint angles directly (orange). The stark discrepancy between the green and red reward curves emphasizes the power of allowing gradient propagation.

$$[T] = \prod_{i=1}^{n} {}^{i-1}T_i(\theta_i) \tag{6.2}$$

In this convention, transformations are serially performed between n links with joint parameters θ_i as specified in Eq. 6.2. The value of a joint angle, ${}^{i-1}T_i(\theta_i)$, shown in Eq. 6.3 describes the transformation from link i to the previous link.

$${}^{i-1}T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.3}$$

In this work, the DH parameters (α, d, and l) are assumed constant throughout training and not directly optimized via backpropagation or other means. Joint angles, specified by θ, are optimized by the neural control policy of the reinforcement learning agent.

Given a fixed set of DH parameters, it is possible to describe the Jacobian from the end-effector position through the DH transformation into joint angle space and thus any preceding functions (such as neural network layers of a policy network), thus enabling the use of this function as part of a deep neural network trained by backpropagation. The importance of this differentiability is further detailed in Sec. 6.3.

Learned Controllers

The choice of controller to use for robot learning can have a large impact on task difficulty - for instance, a ping-pong robot will need precise control of both end-effector pose and velocity, while the task of box stacking may be simpler to learn with a Cartesian controller [Martín-Martín et al., 2019] where low-level control of joints is handled by a controller specified from expert robot knowledge [Khatib, 1995]. In Cartesian (also known as Task) Space, actions are the end-effector's target position and/or orientation in Euclidean space (x, y, z). These high-level control methods can simplify many tasks but often make assumptions about the operating environment and agent structure to perform the complex task of Cartesian space to joint space mapping, which may make accounting

for nuance, such as correcting for a grasped object's weight or avoiding obstacles more difficult.

At the other end of the control spectrum are agents which learn to control robots directly from the torque applied to motors [Yarats et al., 2021a, Srinivas et al., 2020, Hafner et al., 2019a, Deisenroth and Rasmussen, 2011]. This can make solving tasks requiring longer-term planning challenging, as agents typically need to operate at higher control rates while also learning physical dynamics.

In this chapter, we bridge the utility of high and lower-level controllers, utilizing joint-angle control to enable precise control of joints while biasing this control with a forward kinematics function that provides structural bias of end-effector pose. Similarly, JAiLeR [Kumar et al., 2021] details a paradigm for training a joint angle controller which maps from Cartesian space to joint space using model-free RL on proprioceptive state observations. JAiLeR relies on curriculum learning to produce an inverse kinematics controller with similar performance to OSC on goal-conditioned reach tasks, demonstrating the capability to incorporate obstacle avoidance directly into the observation state space of the agent. Unlike JAiLeR, an agent trained with a Kinematic Critic operates on images rather than joint states during evaluation, requires no special curriculum during training, and is shown to work on other complex tasks in addition to inverse kinematics. We bridge these two approaches using joint-level control by employing joint-position controllers but incorporate a kinematic structure into the model to facilitate learning of the end-effector pose in Cartesian space.

Model-Based Control

The Kinematic Critic may be considered a Model-Based Controller as the module computes the forward kinematics for a given agent action though the accuracy of this model is not corrected by any special loss term. Most traditional controllers also use the robot's physical attributes and/or its environment for operation. In the widely used Operational Space Controller [Nakanishi et al., 2008, Khatib, 1995], the accuracy of the physical param-

eterization of the robot, particularly the mass matrix, can vary based on load and configuration, and performance can deteriorate quickly if this estimate is inaccurate [Nakanishi et al., 2008].

There is a large field of research on developing physics models for use by robot controllers. One can impose varying degrees of prior knowledge about the system into the model, including kinematic equations and factors such as mass, friction, or inertia [Lutter et al., 2021a,b, East et al., 2020]. Despite its appeal, prescribed knowledge can be difficult to define for particular robots but can potentially provide both generalization and interpretability.

On the other hand, purely data-driven models [Deisenroth and Rasmussen, 2011, Higuera et al., 2018] use function approximators to fit the complex dynamics that govern robot control. Data-driven approaches tend to be limited by the coverage of their dataset and are slower to train, but are often simpler to implement and can excel if the true system characteristics differ from those that might have been prescribed by (often practically limited) model factorizations.

Most deployed systems blend prescribed physics with learned models that attempt to overcome the inevitable dynamics errors of our physics assumptions in complex robots. Williams *et al.* [Williams et al., 2017] demonstrates the power of learning a controller with factorized dynamics models. They incorporate kinematic equations on several robotic systems, including an aggressive driving task. Model-based Action-Gradient-Estimator Policy Optimization (MAGE) [D'Oro and Jaśkowski, 2020] is a continuous-control DDPG actor-critic algorithm that explicitly trains the critic to provide action-gradients by back-propagating through a learned dynamics model. OSCAR [Wong et al., 2022], introduces a data-driven variant of OSC that learns to adapt the physics model online for task-specific and task-agnostic manipulation. For a more exhaustive review of design choices in incorporating learned dynamics models with physics, refer to Lutter *et al.* [Lutter et al., 2021a].

Parameter	Value
Controller Type	Joint Position
Control Rate	10 Hz
Impedance Mode	Fixed
Max Action Step	0.15 radians
KP	$(30, 60, 50, 70, 60, 70, 90)$
Damping Ratio	$(0.1, 0.17, 0.2, 0.3, 0.1, 0.1, 0.1)$
Actor Input	3 consecutive RGB frames
Camera	Frontview
Reward	Dense
Expl. Stddev. Schedule	linear$(1.0, 0.1, 500000)$
Environment Uniform Init	Reach: Target 0.1m from EEF Door: $\pm.02m$, ±0.25rad from origin Lift: Cube $\pm0.03m$ from origin Can: Can $\pm0.145m \times \pm0.195$ region
Robot Joint Init	Gaussian, stddev=0.2

Table 6.1: Kinematic-Critic Experiment Hyperparameters

Differentiable Physics Engines

Incorporating differentiable physics into robot learning agents has become increasingly effective and efficient thanks to the continually improving fidelity and speed of simulators [Jatavallabhula et al., 2021]. For instance, Deluca [Gradu et al., 2021], a Jax-based library introduces several fully differentiable classic control tasks and ChainQueen [Hu et al., 2019] presents a real-time differentiable simulator for deformable objects. Millard *et al.* [Millard et al., 2020] demonstrate a differentiable simulator for rigid body dynamics with realistic integrators constrained to the laws of physics. The strength of their simulator accuracy is demonstrated over long-horizon adaptive model-predictive control (MPC).

Learning Pose with Kinematic Constraints

We study the problem of learning to solve manipulation tasks from images, where the robot must complete a task with state information provided from camera observations. During training, our agent also has access to its joint angles. When learning robot policies from images, an agent must implicitly learn to map the image and its actions back to

its pose. Learning this action-observation mapping of multi-link articulated objects is inherently difficult because the pose is high-dimensional and has structural constraints inherent in the rigid body chain that makes up the robot, which is compounding and not always visible in the single-camera view provided. Past work on mixture density estimation using neural networks [Bishop, 1994] utilizes a neural network that, given input joint angles and assuming fixed link lengths, predicts the parameters of a mixture distribution over end-effector positions for robot kinematics of a simplified two-link arm. Though this work was not utilized directly for control, this example introduces the more general kinematics concepts at play in our work.

The computer vision community has extensively tackled the problem of finding kinematics from images (or a sequence of images), especially for finding human poses. Deep Kinematic Pose Regression [Zhou et al., 2016] embeds a differentiable kinematic object model into a neural network for predicting pose from images. Their network predicts the joint motion parameters of an object while learning directly about the joint location loss described by a kinematics chain or kinematics tree. This model is demonstrated on a toy 2D robot and 3D human pose, achieving state-of-the-art results on the Human3.6M dataset [Ionescu et al., 2013]. The formulation of the kinematics loss based on pixel input bears similarity to the overall approach featured in our work. However, Deep Kinematic Pose Regression was used in a supervised learning setting with a direct loss of pose rather than for robotic control.

Learning Kinematic and Robot Tasks from Images

There has been an enormous spectrum of work published on learning kinematic [Pavllo et al., 2018] and robotic tasks from images [Levine et al., 2016] and/or expert traces [Peng et al., 2020]. Like our approach, Asymmetric Actor-Critic [Pinto et al., 2017] employs an actor-critic training algorithm in which the critic is trained on rich state observations, while the actor only receives image observations. They emphasize the benefits of fully

(a) Jaco Door Opening Trained with Images and a Kinematic Critic (*kine*)

(b) Jaco Door Opening Trained with Images Only (*image*)

(c) Representative frames from Kinematic Critic (top row) and Image Only (bottom row) agents solving the Door Opening task. We find that Kinematic Critic agents show higher coordination among joints (especially evident in the Jaco arm), producing policies in which the end effector moves smoothly and efficiently through space.

utilizing the simulator to speed up training with auxiliary tasks and add robustness in sim2real.

The key concepts which unify many of these works are:

- Prior knowledge of the *skeleton* of the agent, including important information such as the number of links and link length
- Focus on task-specific training schemes (sometimes with the inclusion of expert traces, behaviour cloning, and/or imitation learning)
- Frequent use of curriculum learning, replay buffers, and other training techniques common to reinforcement and continual learning.

Many of these methods also utilize a combination of auxiliary objectives in addition to the main task loss or task reward.

6.2 Method

We employ a Deep Deterministic Policy Gradients (DDPG) [Lillicrap et al., 2019] reinforcement learning agent that utilizes an actor-critic [Konda and Tsitsiklis, 1999] network structure. As in Asymmetric Actor-Critic [Pinto et al., 2017], our approach uses the actor and critic as two separate networks to give extra information to the critic during training. Our reinforcement learning agent is built on the successful DrQv2 [Yarats et al., 2021a] network architecture. DrQv2, which iterated on its predecessor DrQ [Yarats et al., 2021c], utilizes image augmentation to achieve sample-efficient high performance on continuous robot control tasks. DrQv2 employs a DDPG learner with uses n-step returns to estimate TD error. As in DrQ [Yarats et al., 2021c], DrQv2 [Yarats et al., 2021a], and TD3 [Fujimoto et al., 2018], the practical implementation of these Q functions uses clipped Double Q-learning, duplicating estimate calculations through independent networks with identical architectures f_1 and f_2, resulting in $Q1_*$ and $Q2_*$. For more details on this formulation and overall problem setting, see DrQv2 [Yarats et al., 2021a]. All of our environment hyperparameters match the official implementation of DrQv2, aside from the replay buffer size, which we adjust from 1M to 500k stored image states. To adapt DrQv2 from Deepmind Control Suite [Tassa et al., 2020] Torque Control to Robosuite [Zhu et al., 2020] Joint Position Control, we do not repeat actions. Instead, the agent runs its controller, requesting relative joint angles at a constant control rate. Please see Tab. 6.1 for additional details.

We test the *image* critic architecture from DrQv2, with *angle*, *kine* (Kinematic Critic), and *eef* ablations as described in Fig. 6.2. These ablations were chosen to test the influence of the choice of representation of the additional information provided in Kinematic Critic. In the *angle* ablation, current robot joint angles are concatenated with the state, h_t (as encoded by a convolutional network), and actions, though it does not benefit from explicit knowledge of the kinematic structure of the robot. For both *eef* and *kine*, we estimate the expected pose of the end-effector for a given relative action by computing forward kinematics from the current joint angle using the DH method. In *kine*, the Kine-

matic Critic enables differentiation that the agent can use as a gradient path for learning and backpropagation. However, in the *eef* experiments, we *prevent* gradient propagation through the DH function to study the importance of *gradient flow*, while providing access to roughly the same information as the *kine* variant.

$$Q_{im} = f(h_t, a_t) \tag{6.4}$$

$$Q_{angle} = f(h_t, a_t, j_t) \tag{6.5}$$

$$Q_{kine} = f(h_t, DH(a_t + j_t)) \tag{6.6}$$

$$Q_{eef} = f(h_t, DH(sg(a_t) + j_t))) \tag{6.7}$$

Equation group 6.7 outlines the core critic calculations used by our tested architectures, as outlined in Figure 6.2. We denote the parameterized critic architecture as f, sg for stop gradient, and DH for the Denavit-Hartenburg calculation as described in subsection 6.1. h_t is the intermediate hidden activation resulting from a convolutional network over the input image at time t, x_t, $h_t = conv(x_t)$, a_t is the relative action sampled from the actor sub-network, and j_t denotes the absolute joint position, as given by angle encoders on robot joints.

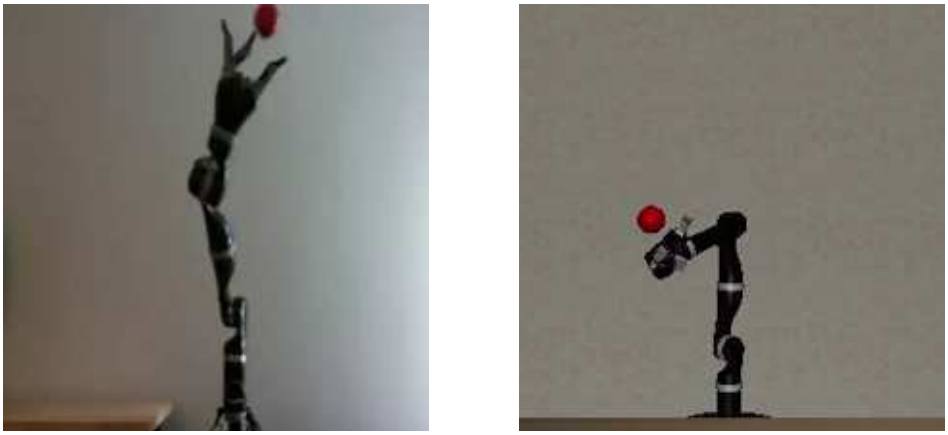
The primary contributions of our method, as compared to the aforementioned background, are as follows

- Our work builds directly on a strong actor-critic visual RL baseline without expert traces, imitation learning, or behaviour cloning.
- We utilize environmental rewards and do not formulate pose-specific losses or employ multitask training. We provide end-effector pose information (via a differentiable DH function) through the internal workings of the critic sub-network, thus allowing the overall actor-critic model to decide how best to use this information to maximize overall reward. This means useful pose-related dynamics (such as

smoothness or stability) are learned implicitly, without explicit pose-specific losses or rewards.

- Rather than using the entire feature-based state information in the critic like Asymmetric Actor-Critic [Pinto et al., 2017], we use only joint angles of the robot along with the DH parameterization to improve training. This is advantageous as joint angles are likely well-modelled in sim2real transfer for various robot platforms.

- Dynamics learning is relegated to the underlying RL agent, and we do not explicitly factorize dynamics learning using physics knowledge - only kinematic structure, on a per-timestep basis, is used. While kinematic structure in this work is closely related to the adjoint calculations used in the Articulated Body Algorithm (ABA) [Featherstone, 2014, Lutter et al., 2021b], as well as the kinematic calculations in Deep Kinematic Pose Regression [Zhou et al., 2016], QuaterNet [Pavllo et al., 2018], or many other works utilizing kinematic chain or tree calculations, it is not coupled with estimations of velocity, acceleration, mass, inertia, and other



(a) Jaco Reach Real **(b)** Jaco Reach Sim

Figure 6.6: Input images from the real (left) and sim (right) ReachBall task for the Jaco arms. The small sim2real gap and data augmentation allowed us to run the sim-trained policy on the real task.

physics related quantities. Instead, we utilize a combination of RL and standard joint-position controllers for overall problem dynamics.

- We operate directly from pixels, with no goal conditioning or state information [Kumar et al., 2021] beyond the knowledge of the robot geometry (assumed constant throughout training on a per-task basis, and given to the critic sub-network) and robot joint angles (which are also used by the controller). The actor only consumes images as inputs, which is common in continuous control from pixels in reinforcement learning [Yarats et al., 2021a, Srinivas et al., 2020].

6.3 Experiments and Discussion

We investigate the impact of adding differentiable kinematics structure into the critic of an image-based reinforcement learning architecture for a set of tasks trained in Robosuite [Zhu et al., 2020], a robotics simulation framework powered by the MuJoCo physics engine [Todorov et al., 2012].

We utilize a Joint Position controller with fixed impedance parameters for all experiments. The joint position controller has a max action step of 0.15 radians for all joints. We tune the controller for Jaco to set KP and damping ratios for the seven joints to $(30, 60, 50, 70, 60, 70, 90)$ and $(0.1, 0.17, 0.2, 0.3, 0.1, 0.1, 0.1)$, respectively. Agents view the scene with a single RGB camera and receive a dense reward for all tasks.

In the PickPlaceCan task, we greatly improved sample efficiency on all agents by adapting the dense reward in Robosuite to include a dense reward that encourages each fingerpad of the manipulator to make contact with the object of interest (in this case, the can). This touch-based reward was especially important in the compliant 3-finger Jaco gripper, where collecting the benchmark *grasp* reward was difficult as it required caging with all fingers.

We demonstrate performance on four tasks: Door, Reach, Lift, and Can (listed in the order of increasing difficulty) for the Jaco 7DOF manipulator and the Panda 7DOF arm.

Object position and robot initial joints are randomized on reset, with a range of variability set per task as default in Robosuite. All performance curves depict the mean (solid line) and the 95 percent shaded confidence interval around the mean over 5 randomly chosen seeds of episodic reward over agent training steps in evaluation.

Our experiments show the differentiability of the DH function is critical to driving effective learning of the network. Simply providing the relevant proprioceptive information to the critic, such as in the *angle* variant (orange), did not seem to help much over the pure *image* variant (blue). This is particularly evident in the reward traces for the Jaco Door opening task shown in Fig 6.4. By propagating information from the DH function inside the critic, through the action space, into the actor and finally through the convolutional encoder networks, we see that the Kinematic Critic agent (green) greatly outperforms the *eef* variant (red) despite both agents receiving the end effector pose as input into the critic. Enabling gradient flow through a kinematics function improves learning speed, stability, and overall performance in every case we tested. This performance increase is particularly notable on tasks requiring a wide distribution of coordinated poses, especially on the Jaco, which was less stable in our simulation.

We also demonstrate the Reach policy on a real Jaco arm (see Fig. 6.9f) by iteratively syncing the simulator to the real setting, predicting actions based on simulator frames, and then applying the predicted action back to the real robot. More sophisticated real2sim or sim2real [Mozifian et al., 2020] approaches could be integrated symbiotically with the Kinematic Critic to improve performance in this setting but fall outside the scope of the current chapter.

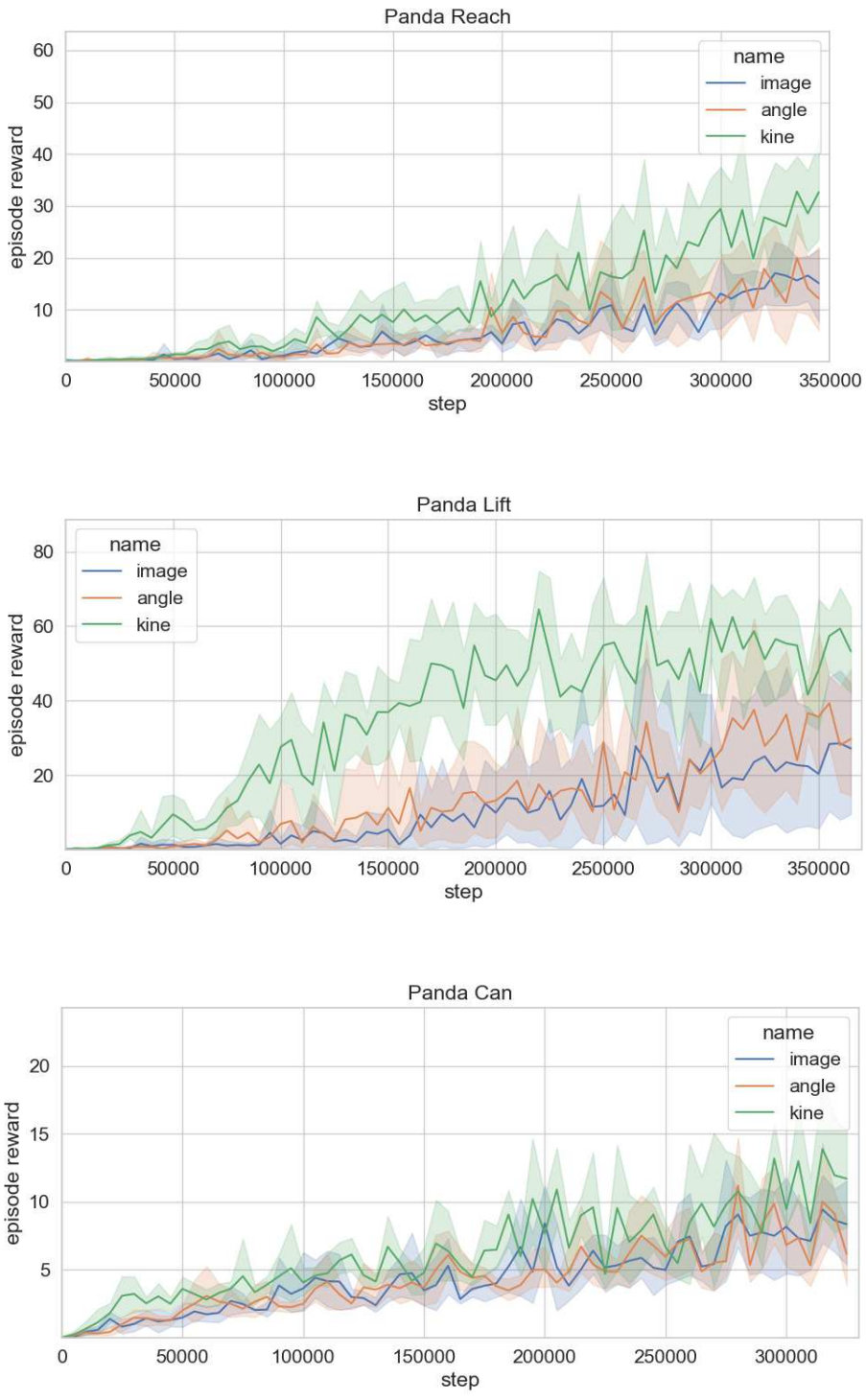


Figure 6.7: Reach, Lift, and Can tasks on the Panda show that our Kinematic-Critic (green) improves learning speed and final performance across the board compared to agents that do not leverage robot kinematics in critic training, and instead observe only images (blue) or images and joint angles (orange).

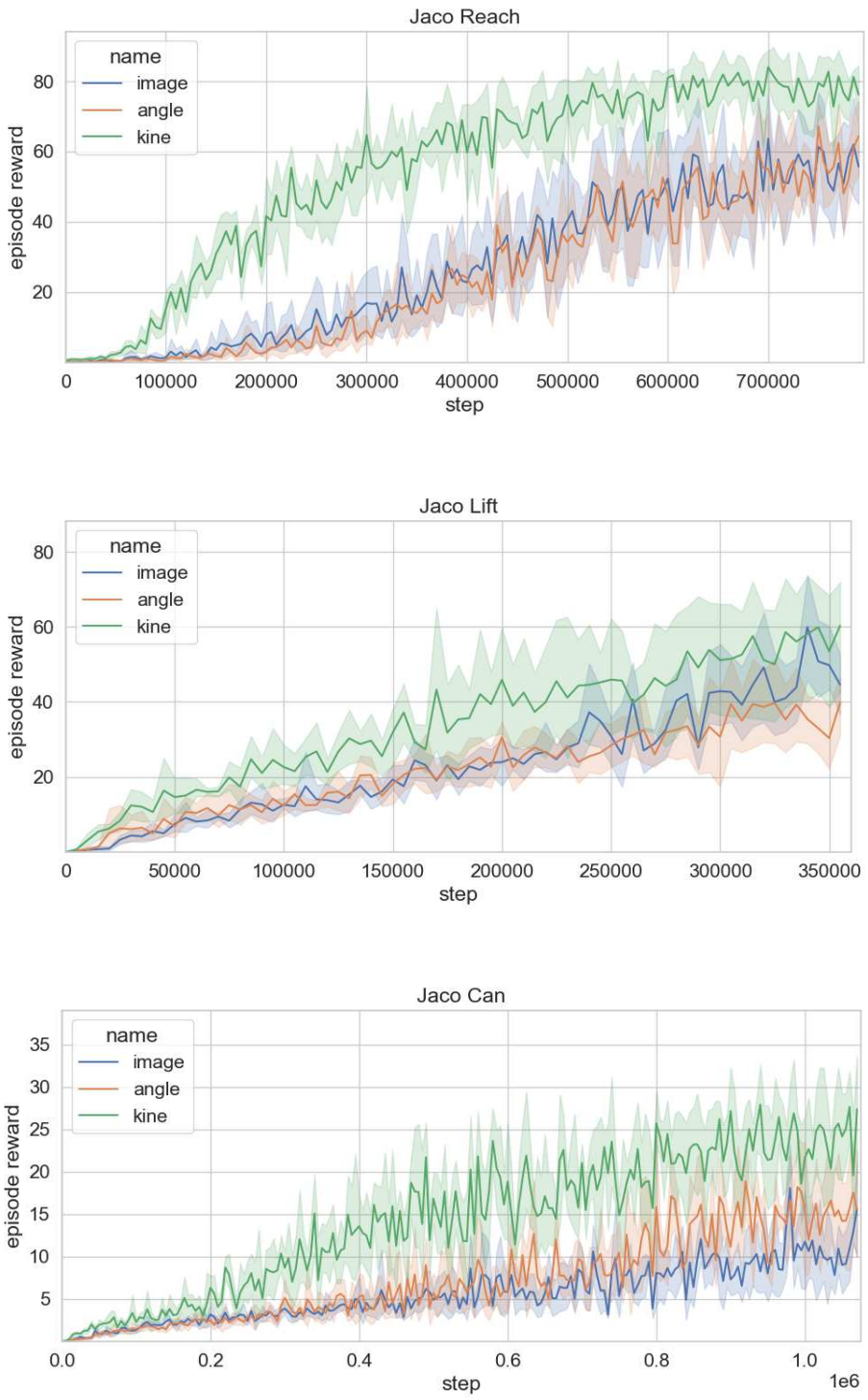
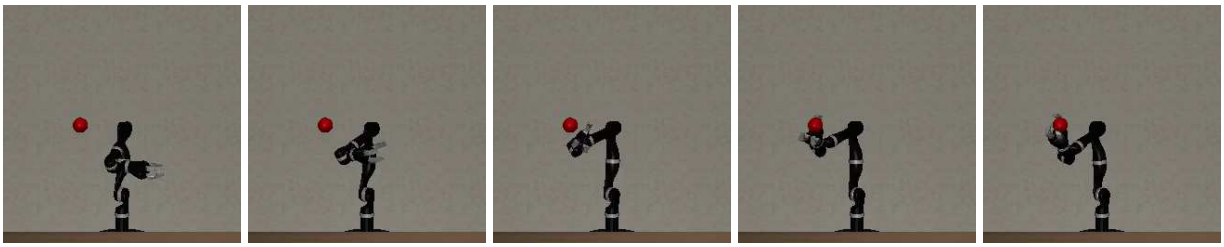


Figure 6.8: Reach, Lift, and Can tasks on the Jaco show that our Kinematic-Critic (green) significantly improves learning speed and final performance across the board compared to agents that do not leverage robot kinematics in critic training and instead observe only images (blue) or images and joint angles (orange).

(a) Jaco Pick Can

(b) Panda Lift Block

(c) Panda Open Door



(d) Jaco Reach Ball

(e) Jaco Reach Real

(f) Training visual policies with a Denavit-Hartenburg view of robot joint positions enables agents to learn complex visual policies that can operate in the real world. This figure depicts successive frames of our Kinematic Critic performing robotic manipulation tasks. Full videos can be found at `https://johannah.github.io/kinematic-critic`

6.4 Discussion and Retrospective

This work introduced a method of incorporating differentiable Denavit-Hartenburg (DH) transformations into an actor-critic reinforcement learning algorithm. By training a critic with access to DH while training its actor only on images, we learn vision-based policies for complex manipulation tasks with better performance than variants with access to the same joint state information. Our evaluation shows that the differentiability of the DH transformation in the critic is crucial for effective training. Overall, this method improves upon strong actor-critic baselines across several benchmark tasks and solves all tasks.

Physics-informed learning methods are a growing and popular area of research. PyPose [Wang et al., 2023] is a new robotics-oriented library that combines deep perceptual models with physics-based optimization. Along with a suite of other robotics tasks, they integrate dynamics and control tasks into an end-to-end learning framework and demonstrate the framework with learning-based MPC by parameterizing the cost functions and dynamics and learning these values with automatic differentiation.

In the next chapter, we look at using the known structure of sensor performance to influence the model architecture in a learned controller for robust grasping.

Chapter 7

Learning Multimodal Manipulation Policies

Manipulating objects with dexterity requires timely feedback that simultaneously leverages the senses of vision and touch. This chapter focuses on the problem setting where visual and tactile sensors provide pixel-level feedback for Visuotactile reinforcement learning agents. Following the theme of this thesis, we look at how we can change the structure of the model to encourage the agent to exploit both forms of high-dimensional input.

We investigate the challenges associated with multimodal learning and propose improvements to existing RL methods, including tactile gating, tactile data augmentation, and visual degradation. When compared with visual-only and tactile-only baselines, our Visuotactile-RL agents showcase (1) significant improvements in contact-rich tasks, (2) improved robustness to visual changes (lighting/camera view) in the workspace, and (3) resilience to physical changes in the task environment (weight/friction of objects).

7.1 Introduction

The synergy between the senses of vision and touch is fundamental to how animals interact with the world around them. While vision is informative of an object's pose and

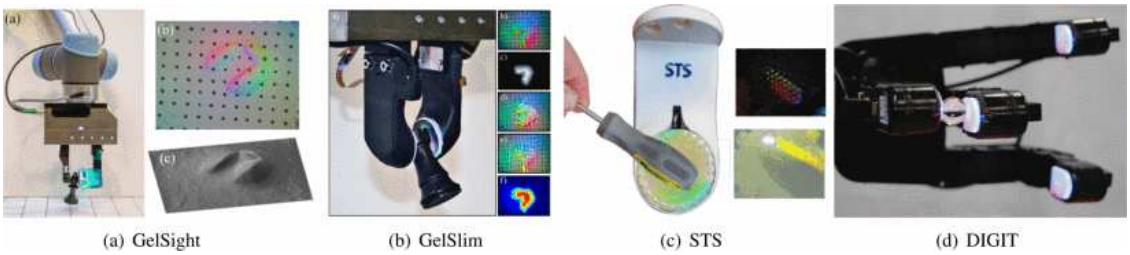


Figure 7.1: Optical tactile sensors. This class of high-resolution tactile sensors renders pixel-level images that emphasize the contact geometry of objects as they interact with the robot finger. From left to right, these sensors show a Gelsight [Li and Adelson, 2013], GelSlim [Donlon et al., 2018], See-Through-your-Skin sensor (STS) [Hogan et al., 2021], and DIGIT [Lambeta et al., 2020] sensor.

general shape, the sense of touch provides accurate feedback on the location of the contact, interaction forces, and the object's material properties. Despite the well-understood importance of visual and tactile feedback in human manipulation [Bozzacchi et al., 2014], robotic systems struggle to integrate both modalities with the ease humans display.

We develop model-free deep reinforcement learning agents that learn to utilize high-resolution visual and tactile information for manipulation tasks. Reinforcement learning approaches have shown an impressive ability to learn expressible controllers for robotic manipulation [Yarats et al., 2021b]. Most techniques are developed to use visual data from a third-person view optical camera [Yarats et al., 2021b] or combine camera observations with low-resolution tactile sensing [Lee et al., 2019]. With the recent development of modern pixel-based tactile sensors such as Gelsight [Yuan et al., 2017], Omnitact [Padmanabha et al., 2020], GelSlim [Donlon et al., 2018], and STS [Hogan et al., 2021], there is an opportunity to provide robots with high-resolution touch feedback. Here, we consider a challenging triad: dexterous manipulation tasks, high-resolution visual and tactile sensors, and reinforcement learning.

In over 200 experiments, we illustrate the benefits of the Visuotactile-RL paradigm and investigate the challenges of this high-resolution, multimodal observation space. In addition to evaluating state-of-the-art reinforcement learning algorithms, we propose train-

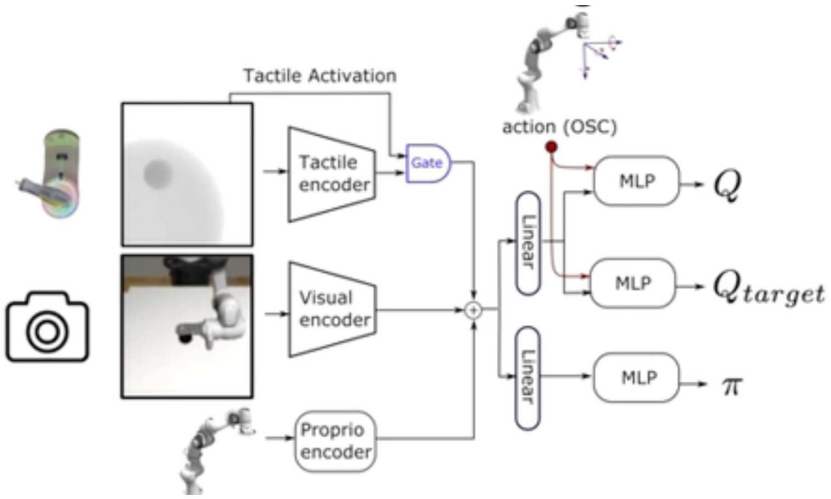


Figure 7.2: Visuotactile-RL network architecture diagram of MP-DrQv2 with tactile gating. The network takes in raw visual, tactile, and proprioceptive observations and encodes them with modality-specific modules. All training is done online with a TD error objective. We find that including a tactile gate to control the flow of tactile gradients through the network as a function of the contact state improves learning on tactile-rich tasks.

ing techniques and network architectures to overcome the difficulties faced in this multi-faceted sensing regime.

While visual sensing provides continuous feedback to the agent during interactions, tactile sensing only provides feedback when the sensor interacts with the environment physically. During contact, tactile sensing provides valuable information related to the interaction: location, shape, and interaction forces at contact. For example, when opening a door, visual feedback drives the reaching phase, while our attention quickly shifts to tactile cues once contact is made with the handle to obtain more precise information about the moment of contact, the handle location, and the grasp stability.

The fundamentally discontinuous nature of physical interactions poses important challenges for policy learning methods. While this is acknowledged and well-studied within the model-based planning and control community[Kroemer et al., 2010, Posa et al., 2014, Hogan and Rodriguez, 2020, Toussaint et al., 2018], it is often overlooked in reinforcement learning. Even as data-driven methods do not explicitly require reasoning over sensory

discontinuities so long as the full state is observable, they must cope with unbalanced datasets where only a small fraction of examples include tactile information, as illustrated in Fig. 7.4. This leads to several important questions regarding the applicability and effectiveness of reinforcement learning methods for policy learning. How do we prevent the agent from over-biasing its attention on visual feedback, which is more prevalent in the dataset? Do the discontinuities associated with contact interactions negatively impact learning stability?

While tactile-only feedback controllers have been show-cased for tactile-centric tasks such as peg insertion [Dong et al., 2021], here we investigate a more general scenario where the object does not begin in contact with the desired location in the environment. In this chapter, we investigate the limitations of the existing RL algorithm for multimodal policy learning and propose novel perceptual architectures and training procedures to overcome them. The main contributions of this work are:

- **Analysis of Visuotactile-RL** We present an in-depth analysis of the performance of state-of-the-art reinforcement learning algorithms and various data augmentation and training strategies. We test a variety of perceptual frameworks and architectures to fuse the visuotactile sensing modalities best and present a numerical experiment study on three simulated manipulation robotic tasks in the Robosuite [Zhu et al., 2020] simulation framework.

- **Multimodal Perception Architecture** We introduce tactile gating, a learning mechanism that addresses the intermittent nature of tactile feedback. A tactile gate in the tactile perceptual module prevents the flow of tactile feedback through the network without detected contact. We show that tactile gating can improve learning performance and help the agent better exploit tactile sensing when used in tandem with visual feedback on contact-rich tasks.

The long-term objective of this research is to design control strategies for contact-rich robotic manipulation tasks that exploit multimodal sensing. The rest of this chapter

is structured as follows. Section 7.3 reviews relevant work related to tactile manipulation. We also introduce and discuss Data Regularized Q Learning (DrQ and its follow-up DrQv2), model-free reinforcement learning algorithms used as the base learning algorithms for learning continuous control from high-dimensional input. We introduce tactile gating, a learning mechanism that controls the flow of tactile feedback through the agent's network, which is effective for the multimodal contact-rich task. We present our experimental setup and experimental results on three robotic manipulation simulation tasks.

7.2 Related Work

7.2.1 Tactile Sensing for Robotics

The sense of touch is a rich and critical source of feedback during robot manipulation. Traditional tactile sensing has included the measurement of shape, texture, and forces in various directions, among other key attributes [Tiwana et al., 2012], gathered using a wide variety of tactile measuring technologies [Chi et al., 2018]. A new generation of optical tactile sensing [Yuan et al., 2017, Donlon et al., 2018] employs cameras embedded in a compliant gel capable of imaging the contact surface at high resolution. These sensors capture the deformations of a reflective soft surface as it makes contact with the world. This enables high-resolution reasoning about contact geometry and slip and contact forces.

Recently, several works have developed tactile policies that can exploit the rich information provided by optical tactile sensors for robotic manipulation. Tian et al. [Tian et al., 2019] develop a model-based tactile controller using pixel-level feedback to manipulate small objects. Hogan et al. [Hogan et al., 2020] develop closed-loop tactile controllers for a dual palm manipulation system able to manipulate objects with dexterity on a tabletop. Wang et al. [Wang et al., 2020] showcase a robotic system capable of swinging up and stabilizing objects by using the rich feedback provided by optical tactile sensors to estimate the object's physical parameters. In Dong et al. [Dong et al., 2021], model-free RL is used

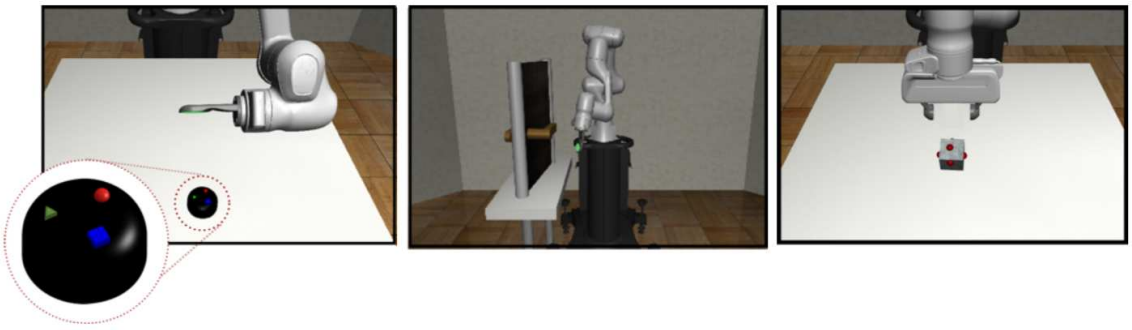
to develop a tactile policy to align an object and environment with a tactile-based feedback insertion policy. These methods all approach the tactile manipulation problem by looking at feedback from the touch signal without considering the multimodality problem of observing manipulation from visual sensors.

7.2.2 Visuotactile Manipulation

There are several recent works in the literature that explore how best to fuse visual and tactile feedback in the context of reinforcement learning. Van Hoof et al. [Van Hoof et al., 2016] showed that a Variational Autoencoder (VAE) [Kingma and Welling, 2014] perceptual architecture is effective in extracting meaningful state representations from visual and tactile inputs on a simple manipulation task. This architecture is extended to multimodal control in [Lee et al., 2019] on a peg insertion task with visual and force-torque sensing. Church et al. [Church et al., 2022] use Proximal Policy Optimization (PPO) [Schulman et al., 2017] with pixel-level visuotactile inputs to teach a simulated robot arm to perform several tactile-rich tasks in a simulation environment. Unlike the tasks that are the focus of this work, the tasks in Church et al. feature sustained contact interactions between the surface of the tactile sensor and the manipulated object. Our work contrasts with these previous approaches by focusing on the learning performance in the presence of intermittent tactile feedback and evaluating the robustness of the learned policies to perturbations to the physical and visual properties of the environment.

7.2.3 RL for high-dimensional inputs

There have been some recent advances in developing RL approaches that learn policies directly from pixel-level feedback. A popular approach has been to extract information from image observations using learned models as in SLAC [Lee et al., 2020], SAC-AE [Yarats et al., 2021d], PlaNet [Hafner et al., 2019b] and Dreamer agents [Hafner et al., 2019b, 2020, 2023]. However, by employing data augmentation, DrQ [Yarats et al., 2021c]



(a) TactileReach (b) Door (c) TactileLift

Figure 7.3: We consider three robotic tasks implemented in the Robosuite simulation framework. In TactileReach (A), the task is precisely making contact with one of the three textures (square, triangle, sphere). In Door (B), the task is to open a hinged door with a robotic palm. In TactileLift (C), the task is to grasp and raise an object with a robotic gripper to a minimum height.

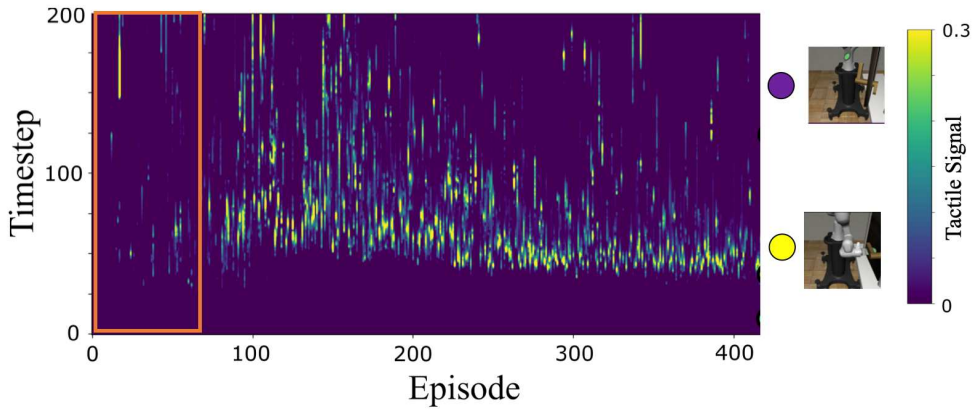


Figure 7.4: Tactile interactions vs. learning experience. Tactile sensing provides intermittent feedback as the sensor interacts with the environment. As the agent learns to interact with the environment during a door-opening task, it explores the door handle in three stages. Under 100 episodes, it makes very few tactile interactions, resulting in sparse tactile observations. From 100 to 300 episodes, it obtains rich tactile observations as it learns to turn the handle. Once the behaviour is learned (around 300 episodes), the agent only receives tactile feedback during the handle-turning phase around eval-step 50.

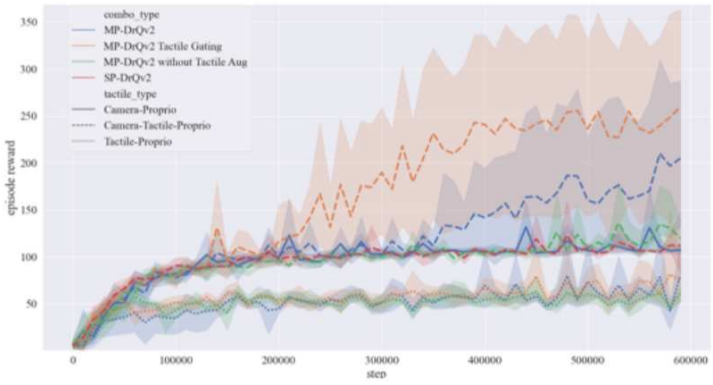


Figure 7.5: Tactile Gating improves learning speed on tactile-critical tasks. This plot demonstrates the evaluation reward by training step for TactileReach. Tactile gating (orange) significantly improves the speed at which the agent learns to solve this visuotactile task compared to the multimodal baseline, MP-DrQv2 (blue).

showed that state-of-the-art performance could be achieved in a model-free setting with a Soft Actor-Critic (SAC) agent. The Data Regularized Q (DrQ) learning approach demonstrates that image-based RL agents tend to overfit observed data. This work was followed by DrQv2, which exchanges SAC for a DDPG learner and multi-step Q updates. The idea that diverse data improves learning and robustness, a well-known concept in the field of computer vision, is also employed in robotic sim2real tasks. Domain randomization (DR), where a simulated environment is randomized during agent training [Tobin et al., 2017] is a common tool for improving robustness in sim2real.

7.3 Background

We consider a Markov Decision Process (MDP) defined by the tuple (S, A, p, r), where S is the set of continuous states, A is the set of continuous actions, $p : S \times S \times A \to \mathbb{R}^+$ represents the probability density of the next state $s_{t+1} \in S$ given the current state $s_t in S$ and the current action $a_t in A$. A stochastic policy is a mapping $\pi : S \times A \to \mathbb{R}^+$. The environment returns a reward $r : S \times A \in [r_{min}, r_{max}]$ at every state transition. The objective of

reinforcement learning is to find an optimal policy π^* from the set of admissible policies Π that maximizes the total returns given a reward function.

$$\pi^* = argmax_{\pi \in \Pi} \sum_t \mathbb{E}_{(s_t, a_t)\ p(p^i)}[r(s_t, a_t)] \tag{7.1}$$

DrQv2 [Yarats et al., 2021a] is an off-policy actor-critic RL algorithm that efficiently learns a policy directly from pixels without a model. Due to its recent success in learning robotic control from images, we employ this approach as the basis for our Visuotactile-RL agents. DrQv2 yields state-of-the-art performance by using image perturbations to regularize the value function.

Adapting the DDPG algorithm, DrQv2 incorporates n-step returns, employs a decaying schedule for exploration noise, and computes the Q-function over image observations, which undergo a randomly sampled image shift during training.

As DrQv2 owes much of its high performance to the augmentation of images during training, it is natural to ask whether this technique will translate from the stationary visual setting to the tactile paradigm. For instance, DrQv2 performs a random shift of the input image, a common data augmentation computer vision pipeline, but may not apply to visuotactile observations. We know that this operation will not preserve the underlying state of the tactile observation as it introduces an effective relative position shift between the observed scene and the tactile sensor. We further explore this concept and test the idea of tactile augmentation in Section 7.6.

7.4 Methodology

This effort aims to investigate the challenges of visuotactile-RL to develop agents capable of robustly fusing the senses of vision and touch for robotic manipulation. We focus on the problem setting where visual information is provided as an RGB image and where tactile feedback is provided in pixel-level measurements, as is typical for novel optical tactile sensors.

One of the biggest challenges in achieving robust multimodal policies is that the tactile signal can be difficult to exploit when combined with visual feedback. We hypothesize that this occurs due to an unbalanced dataset where only a fraction of examples includes tactile information. The distribution of tactile examples is also non-uniform over training. We illustrate this data imbalance problem in Fig. 7.4, which shows the intermittent feedback tactile measurements provide as the sensor interacts with the environment. Early in the training process, the agent makes infrequent tactile encounters. Still, as learning progresses and the tactile sensor is activated more often due to successful manipulation, the model must now deal with an increasingly useful tactile modality.

Perhaps due to the intermittent tactile signal, baseline models become overly reliant on visual information and ignore the tactile input in tasks that do not explicitly require tactile (Door and TactileLift). To evaluate the capability of multimodal agents to utilize either modality, we propose using domain randomization to test the ability of agents to exploit either vision or touch. Our evaluation will involve testing agents in a setting in which one of the modalities has been altered from the training environment and compared to a baseline agent specifically trained under domain randomization.

Tactile Gating

Inspired by Long Short Term Memory (LSTM) networks [Hochreiter and Schmidhuber, 1997] and Highway Networks [Srivastava et al., 2015], we introduce a gating mechanism that dynamically controls the flow of the information to the agent state at each time step based on the usefulness of the tactile signal. This technique, which we call tactile gating, utilizes a hard gate activated during contact, where contact is detected by monitoring the depth image from the tactile observation. The gate remains closed when no tactile activation prevents gradient propagation to the tactile encoder.

Visual Degradation

We also investigate whether degrading the visual signal can improve multimodal performance. Motivated by the incentive to better exploit tactile feedback, during training, we reduce the quality of the visual measurements to encourage the system to focus its attention on tactile cues. In the image dropout training paradigm, the signal from the camera is randomly removed from the observation for a fraction of the interactions.

7.5 Task Descriptions

We investigate Visuotactile-RL on a suite of tasks where the agent must learn to exploit visual information to find and establish contact with an object and then use its tactile sensor to interact with it.

Experiments are performed in the Robosuite [Zhu et al., 2020] simulation framework, which uses MuJoCo [Todorov et al., 2012] as a physics engine. In all scenarios, the RL agent controls the agent using an Operation Space Controller (OSC) [Khatib, 1987] operating at 20 Hz on end-effector pose. Our results are reported on the Panda robot arm, which uses proprioceptive feedback and at least one other pixel-based sensing modality to complete the task.

We simulate the output of the tactile sensor by rendering the contact geometry relative to the perspective of the robot manipulator. While there are several available simulators for optical-based tactile sensors such as TACTO [Wang et al., 2022], Tactile-Gym: RL [Church et al., 2022], and Geometric Contact Rendering [Villalonga et al., 2021]. We use Geometric Contact Rendering as detailed in [Villalonga et al., 2021] to simulate the tactile imprint. This technique consists of clipping the depth image obtained from the perspective of the robot's fingertip to a threshold value corresponding to the half-width of the silicone membrane. The main motivation to use this technique is that it results in faster simulation speeds that translate to lower agent training times. Note that a well-established procedure based on the photometric stereo allows for reconstructing

the depth image from the raw, tactile imprint from optical tactile sensors, as depicted in Fig. 7.1.

We focus on three simulated robot tasks: TactileReach, Door, and TactileLift (see Figure 3). Each task is evaluated on three sensor combinations: 1) camera-proprio, 2) camera-tactile-proprio, and 3) tactile-proprio, where camera denotes a third-person view on the environment, tactile refers to an image-based imprint of the contact, and proprio is the position and velocity of the robot joints.

TactileReach: In TactileReach, shown in the leftmost image of Fig. 7.3, the agent is tasked with touching a tactile feature on the surface of the cylinder in the presence of two distracting tactile features with a palm end-effector. We design this task to test the performance of visuotactile controllers across both modalities. The robot must use vision to reach out from a starting position to the cylinder, randomly initialized on the workspace. Since the tactile features are not observable by the camera, tactile feedback is necessary to intentionally align the palm with the target tactile feature and receive a full reward. The reward schedule is the same as the Reach component of the Robosuite Lift task, except that success is defined as the visuotactile sensor making precise contact with the target texture.

Door: We consider a standard Robosuite door opening task but outfit the robot arm with a palm end-effector. The task requires the agent to turn an articulated handle to open a door with a randomly generated door frame position and rotation offset.

TactileLift: The TactileLift task is adapted from the standard Robosuite Lift task to test tactile sensing. This task requires the agent to grasp and raise a randomly positioned box to a minimum height. We equip the robot with a parallel jaw gripper with two tactile sensors. Images from the two tactile sensors are stacked on the channel axis and treated as one observation. At the start of each episode, we randomly generate small spherical protrusions on the box's surface. The spheres are made tactile obstacles by reducing the simulated friction on their surface to make grasping the box more difficult.

Best Agent: % of Visual Agent				
	visual+tactile	tactile gating	visual dropout	tactile-only
reach	228.18	219.09	-9.09	-32.73
door	0.54	-0.27	-1.08	-0.27
lift	-3.96	-7.01	-58.54	-84.45
Dynamics Randomized: % of Visual Agent				
door	25.45	10.75	4.66	-2.15
lift	2.98	4.68	-58.30	-68.51
Camera Randomized: % of Visual Agent				
reach	28.89	102.22	126.67	93.33
door	-6.82	25.00	312.50	315.91
lift	-7.84	-21.57	68.63	11.76

Figure 7.6: Results on tasks reported as the mean improvement over the base vision-only agent in evaluation over 10 rollouts for the most performant agent trained under each paradigm. The multimodal paradigm was essential for the tactile reach task and under dynamics randomization. Visual degradation allowed the agent to perform well under visual randomization.

7.6 Experiments

This section demonstrates that Visuotactile-RL is powerful in scenarios involving 1) rich contact interactions, 2) visual randomizations, and 3) perturbed dynamic parameters.

All RL experiments are performed using the default DrQv2 hyperparameters [Yarats et al., 2021a]. We changed the size of the replay buffer to $600,000$ to accommodate the longer training time needed to learn manipulation. In addition to our analysis using DrQv2, we develop and evaluate a pixel-based variant of TD3 [Fujimoto et al., 2018], dubbed DrTD3, which utilizes the same encoder architecture and data augmentation strategy as MP-DrQv2, but without the n-step TD error estimates and scheduled exploration noise.

Perception Architecture: We investigate two image encoder architectures: **MultiPath** (MP) and **SinglePath** (SP). In the MultiPath paradigm, a unique encoding network is used

for vision and tactile, while in SP, both modalities share the same encoder. Our agent utilizes the past 3 frames as an input state, resulting in an observation size of $(9 \times 84 \times 84)$ for the RGB image and an observation size of $(3 \times 84 \times 84)$ for the tactile depth image when using the palm tactile sensor and $(6 \times 84 \times 84)$ when utilizing the parallel jaw gripper sensors. In the SinglePath setting, images from both sensors are combined on the channels axis, producing an input of $(12 \times 84 \times 84)$ to the convolutional encoder with the palm sensor. We employ the same convolutional encoder architecture described in DrQv2 [Yarats et al., 2021a] in all pixel-based encoders in this work. We consider MP vs SP in the table in Fig. 7.6 and find that overwhelmingly, the MP architecture offers higher performance, justifying the increase in the number of model parameters and wall clock training time necessary for separate encoders.

Tactile Gating: The inclusion of tactile gating in the model architecture improves the learning speed in contact-rich tasks. We present the learning curve for TactileReach in Fig. 7.5 where using a tactile gate provides a performance improvement of 25 percent. We also note that this agent needs fewer environment interactions to learn to solve the task and can exploit the tactile sensor more effectively than baseline methods. For the Door task, including tactile gating did not significantly alter the learning performance but improved robustness to visual perturbations. We hypothesize that this is because tactile reasoning is less critical for successfully executing these tasks in the simulated environment.

Visual Degradation: We test several methods of visual degradation where we reduce the quality of the visual observations to encourage the agent to utilize tactile information. We tested visual degradation by training agents with visual dropout and DR Visual and found that visual dropout produces more performant agents in both the standard environment and DR Visual evaluation. Note that visual degradation techniques have a negative impact on overall system performance but do seem to improve agent reliance on tactile information. Given the promising performance of image dropout to improve multimodal sensing, this technique should be explored further, for example by exploiting

the simulation environment to implement conditional camera degradation that depends on the presence of a tactile signal.

Tactile Augmentation: We find that Tactile Augmentation improves performance on most tasks as shown in columns **MP-DrQv2** and **DrQv2 without Tactile Augmentation** in Fig. 7.6. This suggests that the successful augmentation approaches in pixel-based RL may translate well to the tactile signals.

Robustness to Domain Shift: To quantify agent robustness to physical changes in the environment, we employ domain randomization on physical dynamics (friction, weight, etc) for objects in the scene. This environment setup is referred to as DR Dynamics, as shown in Fig. 7.6, and is employed in some training experiments and for evaluation in relevant tasks. High performance under DR Dynamics suggests the agent may utilize the tactile sensor for feedback when environmental changes are not evident in the visual sensor. Our evaluations are done on all objects in the scene besides the robot. TactileLift policies evaluated in out-of-training distribution with dynamics randomization. Multimodal policies improve robustness to these dynamics changes such as weight, friction of the box. We find that tactile is critical to solving this evaluation, with multimodal agents performing better than visual-only or tactile-only policies when faced with randomized dynamics. The agent trained on domain randomization performed best overall, but Tactile Gating produced the single highest-performing agent on this evaluation.

We also test the agents under DR Visual, which perturbs the visual image's lighting conditions and camera location. This randomization significantly alters the camera's viewpoint, often causing the object of interest to be absent from the scene. We employ the default Robosuite DR wrapper for randomization and sample a new environment configuration for each episode. We sho that agents that effectively learn to exploit tactile information perform better under strong visual changes than visual-only agents. The visual-only agent fails under strong visual perturbations, while the tactile-only agent remains robust as it does not observe the randomization. Comparing the multimodal models, we find that Tactile Gating performs similarly to the baseline (MP-DrQv2) without

requiring intentional degradation of the visual sensor during training (DR Visual and Camera Dropout).

7.7 Discussion and Retrospective

This chapter explores the ability of deep reinforcement learning methods to fuse and exploit visual and tactile feedback to learn manipulation policies. We focus on the problem setting where tactile feedback is provided by optical-based sensors that render high-resolution pixel-level information. We find that the fusion of both modalities results in optimal performance on a set of manipulation tasks and improved robustness to system perturbations in the dynamics and lighting conditions.

Key to these results is the inclusion of a tactile gate that controls the flow of tactile feedback through the agent's network. We show that tactile gating results in an agent that can exploit tactile sensing earlier and achieve higher performance than benchmarks that employ common encoder-based perceptual modules. Additionally, we show that data augmentation techniques adapted from DrQv2 on both tactile and visual streams benefit robust learning. While this is well-known for visual feedback, we show that this technique also significantly improves tactile sensing.

This chapter intersects with several high-growth research fields, including multimodal learning, reinforcement learning for robotics, and visuotactile sensing.

Multimodal learning has seen significant improvements, notably when paired with self-supervised learning and moving toward foundation models trained on large datasets. CLIP [Radford et al., 2021] trains an image encoder and a text encoder to learn to match pairings between images and text, resulting in a model that shows impressive zero-shot performance in classification. A similar method could be employed to learn a mapping between the visual and tactile input sources in this chapter, resulting in a learned embedding that could be applied to various downstream tasks.

In tactile sensing, efforts have been made to simultaneously (1) improve simulation fidelity and (2) reduce the sim2real gap. [Narang et al., 2021] introduce a finite element method (FEM) model of the BioTac sensor that allows for fast and accurate simulation. This paper also learns a cross-modal self-supervised latent representation (shown in Fig.7.7) to reduce the sim2real gap from a small supervised dataset of contact patches represented in sim and real. This method of learning a direct mapping from the physics model to the real observation highlights the strengths of combining physics and data-driven methods.

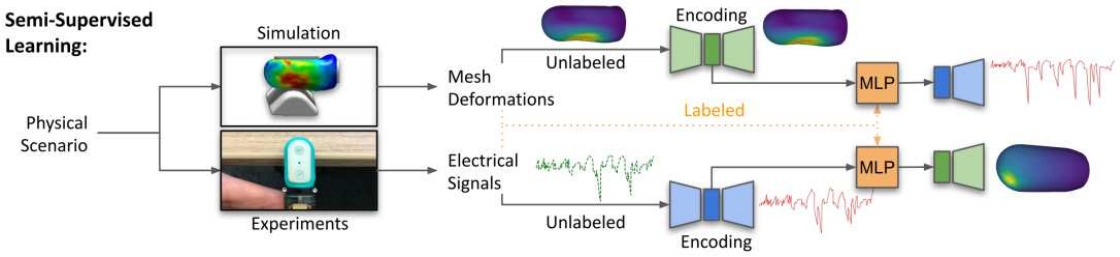


Figure 7.7: Learning method for reducing the sim2real gap through self-supervision on a real small dataset introduced in [Narang et al., 2021]. Figure from [Narang et al., 2021].

In reinforcement learning for robotics, like this work, there has been increased interest [Yu et al., 2022, Yarats et al., 2022] in analyzing the data on which agents are trained. Better simulation [Gao et al., 2022] of robots, objects, and robot-object interactions improves the accuracy of agents. [Qin et al., 2023] reinforces some of the themes of our work, showing that the choice of representation (such as point clouds) and task-specific supervision (in the form of contact-based rewards) can help generalize to new objects in the real world.

In the next chapter, we summarize the contributions of this thesis and contemplate the future directions of this line of research.

Chapter 8

General Conclusion

The work presented in this thesis is an investigation into applications of robot control policies that work by incorporating structural bias from prior knowledge into controllers that learn about their observed environment. This dissertation reflects rapid changes in how roboticists approach algorithms for interpreting observations and defining controllers in robots. The time period covered in this work, beginning in 2016, was marked by a notable shift in the zeitgeist, as robotics tended away from years of expert-defined systems and towards learning-based methods. This dissertation leverages state-of-the-art algorithms and machine learning models while leveraging years of expertise in expert systems. We show how physics-based models can boost performance in learned systems, especially in domains with difficult-to-acquire data.

8.1 Summary of Contributions

In the following section, we revisit the objectives presented in Section 1.3 and summarize the contributions presented in this thesis. Specifically, we examine how we can exploit the following structural bias to improve robot behaviour:

1. **Developing robots that can deploy contextually variable control schemes**: In the introduction to this document, we discussed that a major limitation of modern

robots is their lack of ability to take environmental context into account and significantly modify behaviour. Over the course of this thesis, we have introduced several improvements for robustifying robot deployment.

(a) In Chapter 5, we tackle the costly problem of ocean basin mapping and show that by building an intelligent surveying system that leverages the heterogeneity of various robot morphologies, we can efficiently survey a region. Given a mapping objective, our system works by contextually predicting the time vs value of an ASV releasing a drifter at a particular position. We demonstrate that this system is robust, improving the surveying capacity over a mapping ASV working alone and showing superior map quality over a team of drifters deployed without predictive deployment.

(b) In Chapter 7, we show a method for adding inductive bias into a learned controller so that a robot manipulator learns to utilize two complementary sensor modalities, vision and touch contextually. This problem is difficult because the *tabular rasa* learning agent rarely encounters useful touch observations early in training, so it simply learns to ignore touch input in favour of vision, but ultimately, touch is needed to complete the dexterous tasks fully. We add synthetic noise and artificial gating to vision observations to force the agent to leverage both modalities and show that agents trained with this prior are able to complete dexterous tasks even when one of the input modalities is corrupted, contextually switching to the modality which serves as the best information source at the current state.

2. **Leverage known physics in learning paradigms** This thesis argues that learning-based methods need not re-learn all physics but can exploit known physics and learn residuals on top of known equations.

(a) In Chapter 5, we rely on physical models of ocean currents to predict drifter flow and use these predictions to inform active drifter distribution.

(b) Less directly, in Visuotactile-RL (Chapter 7), we leverage our knowledge about the relative performance of vision and tactile sensors at various parts of manipulation (approach vs object interaction) to improve learning performance.

(c) Chapter 6, we directly incorporate physics equations describing forward kinematics into the architecture of a vision-based learned controller for a robot manipulator. We show that without optimizing the end-effector pose directly, the resulting learned controller learns faster and more stable policies than comparison baselines.

3. **Exploit information heterogeneity** Truly robust robots know how to adapt to varying information at deployment time. They use the right sensor for a particular job, make informed guesses about control strategies, and effectively backtrack when they encounter problems.

   (a) In our effort to build a controller that could operate on Martian terrain for sample tuble re-localization where we may not be able to account for environmental or domain shift, we developed a meta-model that could predict performance between a classical method that had access to pose labels collected years prior and a learned method that could operate reactively based on observations. We showed how the meta-model successfully learned to choose the classical method when the sample tube was artificially masked in the given observation and the direct method when feature mapping was challenging due to the lack of landmark features such as rocks.

   (b) In Chapter 5, we use the ability of low-cost sensors to achieve spatial diversity and utilize the strengths of different robot morphologies to achieve better sampling at a lower cost. Passive drifters are deployed at positions where they are expected to have long trajectories over parts of the survey space that were previously unseen. At the same time, the ASV uses its ability to control its pose to sample regions of the survey area with complex or unknown water currents.

(c) Chapter 7 shows a robot that learns to use touch and/or vision sensors to manipulate objects robustly. In this case, our training paradigm encourages the robot to effectively attend to touch when dexterity is required and vision when objects are not in manipulation range. When vision is corrupted, we show that the agent is able to adapt to a more tactile-dependent policy.

(d) In Chapter 6, we give extra information during training to the critic, including proprioceptive joint-angle information propagated through forward kinematics to improve and stabilize policy learning. At evaluation time, the robot only needed images to make actions but learned significantly faster than baselines whose critic only had access to images during training.

8.2 Limitations

In this section, we discuss some of the limitations of this dissertation, mostly related to challenges that arise from the need to deploy robots with real-world sensing into their intended environment.

- **Hardware Experiments** This dissertation primarily demonstrates performance on datasets or in simulation rather than on real hardware. Our drifter solutions mostly present algorithmic performance in simulation but pair it with real-world deployments that are demonstrations only due to the inability to observe the true ground truth state of the complex real-world system. Although we designed a fully autonomous drifter deployment and recovery system, this system was never built due to Covid restrictions and associated delays.

- **Online Learning** The physics-based planning method presented in Chapter 5 is unique among the presented solutions in that the robots refine policies based on online observations. Other Chapters present solutions that pre-train control policies and then deploy them to the target environment with little online refinement. This

can make the proposed solutions susceptible to domain shifts that are likely to be encountered over the life of the robot.

- **Sim-to-Real** Real robot deployments face sim-to-real challenges as sensor observations and robot states are not perfectly captured in simulation. Robustness in the shift from simulation to on-robot deployment was mostly ignored in this dissertation. In the manipulation work specifically, we expect to face challenges due to the visual transfer of simplistic synthetic worlds to the complexity of the real world.

As is common in the necessarily cross-disciplinary field of robotics, this thesis blends progress from multiple domains, including computer science, electrical engineering, and mechanical engineering, and works to improve data accessibility for scientific sampling domains for oceanographers, environmental scientists, and geologists.

8.3 Future Work and Perspective

Many robotic programs are simple at a high level (move and grab), but reliable behaviours require knowledge of context that can be hard to program explicitly (a robot should walk on a pathway but avoid puddles). In our effort to produce robots that can operate robustly in open-world environments, we were often limited by access to data and external context. These limitations are rapidly fading thanks to two major changes in the machine learning landscape, (1) large foundational models trained on internet-scale data and (2) increasing access to task-specific data.

8.3.1 Foundation Models in Robotics

At the time of this writing, large **foundation models** [Bommasani et al., 2021] trained on diverse, task-agnostic, internet-scale data are beginning to be leveraged to provide broad generalization that enables robots to operate in diverse environments [Brohan et al., 2022] and interact with humans via conversational language [Brohan et al., 2023]. In addition

to providing real-world grounding, large foundation models also demonstrate in-context learning [Brown et al., 2020], allowing for effective adaptation to new tasks with online prompting for controlling behaviour through language [Yu et al., 2023]. We expect large-scale models to continue to help provide useful and easily specifiable priors for robots, especially in human-built environments.

8.3.2 Low Domain-Gap Expert Demonstrations

Access to expert behaviour data has been a limitation in robotics, where real-world interaction requires instrumentation to achieve demonstrations of successful task performance without a large domain or morphology mismatch. This really sets the field apart from tasks such as computer vision, language processing, or speech generation, where large, labelled, and curated datasets are readily available. We may be entering an era where this limitation ceases due to increased investments in (1) low-cost demonstration hardware, (2) improvements in off-morphology algorithms, (3) improved simulations, and (4) more fielded robots bootstrapping data collection.

We have seen significant advances in low-cost solutions for collecting demonstrations without domain shift from platforms such as Mobile Aloha [Fu et al., 2024]. Improvements in learning controllers capable of translating morphology from observations may allow us to effectively leverage large datasets of video of humans interacting with the world [Yuan and Makoviychuk, 2023, Hassan et al., 2023]. It is also fairly safe to anticipate that simulation and world models, in general, will continue to improve as new methods emerge for generating physically realistic video. Improved world bounds in simulation will naturally reduce the necessity of acquiring costly real-world data on robotic hardware. Finally, as the number of fielded robots increases, access to large datasets of robot behaviour in the wild will become available to train increasingly large and more general models for planning and control.

8.4 Concluding Remarks

This dissertation set out to provide robots with informed inductive biases that could improve their behaviour in the real world. Through four diverse projects, we showed an ability to build systems that were able to contextualize based on their environment, learn from limited data due to useful inductive biases, and exploit information heterogeneity through sensing. We hope that the contributions made in this thesis inspire further work in developing fielded robots that make use of physics-informed and learned information.

Bibliography

T. Alam, G. M. Reis, L. Bobadilla, and R. N. Smith. A data-driven deployment approach for persistent monitoring in aquatic environments. In *IEEE International Conference on Robotic Computing (IRC)*, pages 147–154, Jan 2018. doi: 10.1109/IRC.2018.00030.

A. A. Allen and C. B. Billing. Spatial objective analysis of small numbers of lagrangian drifters. In *OCEANS '88. A Partnership of Marine Interests. Proceedings*, pages 860–864 vol.3, Oct 1988. doi: 10.1109/OCEANS.1988.23624.

A. A. Allen and J. V. Plourde. Review of leeway: field experiments and implementation. Technical report, COAST GUARD RESEARCH AND DEVELOPMENT CENTER GROTON CT, 1999.

B. Anderson and J. Moore. *Optimal Filtering*. Prentice-Hall, 1979.

J. Anderson and G. A. Hollinger. Communication planning for cooperative terrain-based underwater localization. *Sensors*, 21(5):1675, 2021.

T. Anthony, Z. Tian, and D. Barber. Thinking fast and slow with deep learning and tree search. *CoRR*, abs/1705.08439, 2017.

H. Aoyagi, Y. Michida, M. Inada, H. Otobe, and R. Takimoto. Experiment of particle dispersion on the sea surface with gps tracked drifters. In *OCEANS '04. MTTS/IEEE TECHNO-OCEAN '04*, volume 1, pages 139–145 Vol.1, Nov 2004. doi: 10.1109/OCEANS.2004.1402908.

P. J. Ball, C. Lu, J. Parker-Holder, and S. Roberts. Augmented world models facilitate zero-shot dynamics generalization from a single offline environment. In *International Conference on Machine Learning*, pages 619–629. PMLR, 2021.

M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *CoRR*, abs/1207.4708, 2012.

R. Bellman. Dynamic programming princeton university press. *Princeton, NJ*, 1957.

C. M. Bishop. Mixture density networks. Technical report, 1994.

E. W. Blockley, M. J. Martin, and P. Hyder. Validation of foam near-surface ocean current forecasts using lagrangian drifting buoys. *Ocean Science*, 8(4):551–565, 2012. doi: 10.5194/os-8-551-2012. URL `https://www.ocean-sci.net/8/551/2012/`.

R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

C. Bozzacchi, R. Volcic, and F. Domini. Effect of visual and haptic feedback on grasping movements. *Journal of neurophysiology*, 112(12):3189–3196, 2014.

J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

R. Brainard, R. Moffitt, M. Timmers, G. Paulay, L. Plaisance, N. Knowlton, J. Caley, F. Fohrer, A. Charette, C. Meyer, et al. Autonomous reef monitoring structures (arms): A tool for monitoring indices of biodiversity in the pacific islands. In *11th Pacific Science Inter-Congress, Papeete, Tahiti*, 2009.

Ø. Breivik and A. A. Allen. An operational search and rescue model for the norwegian sea and the north sea. *Journal of Marine Systems*, 69(1-2):99–113, jan 2008. doi: 10.1016/j.jmarsys.2007.02.010.

A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on Robot Learning*, pages 287–318. PMLR, 2023.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

C. Browne and E. Powley. A survey of monte carlo tree search methods. *Intelligence and AI*, 4(1):1–49, 2012. ISSN 1943-068X. doi: 10.1109/TCIAIG.2012.2186810.

L. Buesing, T. Weber, S. Racanière, S. M. A. Eslami, D. J. Rezende, D. P. Reichert, F. Viola, F. Besse, K. Gregor, D. Hassabis, and D. Wierstra. Learning and querying fast generative models for reinforcement learning. *CoRR*, abs/1802.03006, 2018.

J. P. S. C. L. Gentemann, P. L. Mazzini, C. Pianca, S. Akella, P. J. Minnett, P. Cornillon, B. Fox-Kemper, I. Cetinić, T. M. Chin, J. Gomez-Valdes, and J. Vazquez-Cuervo. Saildrone: adaptively sampling the marine environment. In *Bulletin of the American Meteorological Society*, volume 101, page E744–E762, 2020. doi: https://doi.org/10.1175/BAMS-D-19-0015.1.

B. Chadwick, C. Katz, J. Ayers, J. Oiler, M. Grover, A. Sybrandy, J. Radford, T. Wilson, and P. Salamon. Gps drifter technologies for tracking and sampling stormwater plumes. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–10, Sept 2016. doi: 10.1109/OCEANS.2016.7761010.

F. Chavez, J. Pennington, R. Herlien, H. Jannasch, G. Thurmond, and G. Friederich. Moorings and drifters for real-time interdisciplinary oceanography. *Journal of Atmospheric and Oceanic Technology - J ATMOS OCEAN TECHNOL*, 14, 10 1997. doi: 10.1175/1520-0426(1997)014⟨1199:MADFRT⟩2.0.CO;2.

L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

C. Chi, X. Sun, N. Xue, T. Li, and C. Liu. Recent progress in technologies for tactile sensors. *Sensors*, 18(4):948, 2018.

H. Choi, C. Crump, C. Duriez, A. Elmquist, G. Hager, D. Han, F. Hearl, J. Hodgins, A. Jain, F. Leve, C. Li, F. Meier, D. Negrut, L. Righetti, A. Rodriguez, J. Tan, and J. Trinkle. On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward. *Proceedings of the National Academy of Sciences*, 118(1):e1907856118, 2021. doi: 10.1073/pnas.1907856118. URL `https://www.pnas.org/doi/abs/10.1073/pnas.1907856118`.

A. Church, J. Lloyd, N. F. Lepora, et al. Tactile sim-to-real policy transfer via real-to-sim image translation. In *Conference on Robot Learning*, pages 1645–1654. PMLR, 2022.

R. Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *Computers and Games: 5th International Conference, CG 2006, Turin, Italy, May 29-31, 2006. Revised Papers 5*, pages 72–83. Springer, 2007.

K.-F. Dagestad, J. Röhrs, Ø. Breivik, and B. Ådlandsvik. Opendrift v1. 0: a generic framework for trajectory modelling. *Geoscientific Model Development*, 11(4):1405–1420, 2018.

DARPA-FFT. Darpa forecasting floats in turbulence challenge. *https://oceanofthings.darpa.mil*, 2021.

J. Das, F. Py, T. Maughan, T. O'Reilly, M. Messié, J. Ryan, G. S. Sukhatme, and K. Rajan. Coordinated sampling of dynamic oceanographic features with underwater vehicles and drifters. *The International Journal of Robotics Research*, 31(5):626–646, 2012. doi: 10.1177/0278364912440736. URL https://doi.org/10.1177/0278364912440736.

J. Das, F. Py, J. B. Harvey, J. P. Ryan, A. Gellene, R. Graham, D. A. Caron, K. Rajan, and G. S. Sukhatme. Data-driven robotic sampling for marine ecosystem monitoring. *The International Journal of Robotics Research*, 34(12):1435–1452, 2015a. doi: 10.1177/0278364915587723. URL https://doi.org/10.1177/0278364915587723.

J. Das, F. Py, J. B. Harvey, J. P. Ryan, A. Gellene, R. Graham, D. A. Caron, K. Rajan, and G. S. Sukhatme. Data-driven robotic sampling for marine ecosystem monitoring. *The International Journal of Robotics Research*, 34(12):1435–1452, 2015b. doi: 10.1177/0278364915587723. URL https://doi.org/10.1177/0278364915587723.

A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):1052–1067, June 2007. ISSN 0162-8828.

M. Deisenroth and C. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pages 465–472. Omnipress, 2011.

R. Deits and T. Koolen. Picking Up Momentum, Jan. 2023. URL https://www.bostondynamics.com/resources/blog/picking-momentum.

R. S. Denavit, Jacques; Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *Trans ASME J. Appl. Mech*, 23:215–221, 1955.

E. W. Dijksta. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

K. Doherty, D. Frye, S. Liberatore, and J. Toole. A moored profiling instrument. *Journal of Atmospheric and Oceanic Technology*, 16(11):1816–1829, 1999.

S. Dong, W. Yuan, and E. H. Adelson. Improved gelsight tactile sensor for measuring geometry and slip. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 137–144. IEEE, 2017.

S. Dong, D. K. Jha, D. Romeres, S. Kim, D. Nikovski, and A. Rodriguez. Tactile-rl for insertion: Generalization to objects of unknown geometry. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6437–6443. IEEE, 2021.

E. Donlon, S. Dong, M. Liu, J. Li, E. Adelson, and A. Rodriguez. Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1927–1934. IEEE, 2018.

P. D'Oro and W. Jaśkowski. How to learn a useful critic? model-based action-gradient-estimator policy optimization. *Advances in Neural Information Processing Systems*, 33: 313–324, 2020.

A. Doucet, N. De Freitas, N. J. Gordon, et al. *Sequential Monte Carlo methods in practice*, volume 1. Springer, 2001.

G. Dudek and M. Jenkin. *Computational Principles of Mobile Robotics*, chapter 4: Non-Visual Sensors and Algorithms, pages 82–123. Cambridge University Press, New York, NY, USA, 2nd edition, 2010a. ISBN 0521692121, 9780521692120.

G. Dudek and M. Jenkin. *Computational Principles of Mobile Robotics*, chapter 6: Representing and Reasoning about Space, pages 167–210. Cambridge University Press, New York, NY, USA, 2nd edition, 2010b. ISBN 0521692121, 9780521692120.

G. Dudek and M. Jenkin. *Computational Principles of Mobile Robotics*. Cambridge University Press, New York, NY, USA, 2nd edition, 2010c. ISBN 0521692121, 9780521692120.

G. Dudek, P. Giguere, C. Prahacs, S. Saunderson, J. Sattar, L.-A. Torres-Mendez, M. Jenkin, A. German, A. Hogue, A. Ripsman, et al. Aqua: An amphibious autonomous robot. *Computer*, 40(1):46–53, 2007.

S. East, M. Gallieri, J. Masci, J. Koutník, and M. Cannon. Infinite-horizon differentiable model predictive control. *arXiv preprint arXiv:2001.02244*, 2020.

R. Featherstone. *Rigid body dynamics algorithms*. Springer, 2014.

G. Ferri, A. Manzi, F. Fornai, F. Ciuchi, and C. Laschi. The hydronet asv, a small-sized autonomous catamaran for real-time monitoring of water quality: From design to missions at sea. *IEEE Journal of Oceanic Engineering*, 40(3):710–726, 2015. doi: 10.1109/JOE.2014.2359361.

E. Fiorelli, N. E. Leonard, P. Bhatta, D. A. Paley, R. Bachmayer, and D. M. Fratantoni. Multi-AUV control and adaptive sampling in monterey bay. *IEEE Journal of Oceanic Engineering*, 31(4):935–948, 2006.

G. Flaspohler, N. Roy, and Y. Girdhar. Near-optimal irrevocable sample selection for periodic data streams with applications to marine robotics. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5691–5698, 2018. doi: 10.1109/ICRA.2018.8460709.

V. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello. Bayesian filtering for location estimation. *IEEE pervasive computing*, 2(3):24–33, 2003.

Z. Fu, T. Z. Zhao, and C. Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117*, 2024.

S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

R. Gao, Z. Si, Y.-Y. Chang, S. Clarke, J. Bohg, L. Fei-Fei, W. Yuan, and J. Wu. Objectfolder 2.0: A multisensory object dataset for sim2real transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10598–10608, 2022.

Y. Girdhar and G. Dudek. Modeling curiosity in a mobile robot for long-term autonomous exploration and monitoring. *Autonomous Robots*, 40(7):1267–1278, 10 2016.

P. Gradu, J. Hallman, D. Suo, A. Yu, N. Agarwal, U. Ghai, K. Singh, C. Zhang, A. Majumdar, and E. Hazan. Deluca - A differentiable control library: Environments, methods, and benchmarking. *CoRR*, abs/2102.09968, 2021. URL `https://arxiv.org/abs/2102.09968`.

A. Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

A. Guez, T. Weber, I. Antonoglou, K. Simonyan, O. Vinyals, D. Wierstra, R. Munos, and D. Silver. Learning to search with mctsnets. *CoRR*, abs/1802.04697, 2018.

D. Ha and J. Schmidhuber. World models. *CoRR*, abs/1803.10122, 2018.

D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019b.

D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

J. Hansen and G. Dudek. Coverage optimization with non-actuated, floating mobile sensors using iterative trajectory planning in marine flow fields. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2018. URL http://johannah.github.io/publications/iros2018driftercoverage.pdf.

J. Hansen, S. Manjanna, A. Q. Li, I. Rekleitis, and G. Dudek. Autonomous marine sampling enhanced by strategically deployed drifters in marine flow fields. In *OCEANS'18 MTS/IEEE Charleston*, pages 1–7, October 2018. URL http://johannah.github.io/publications/iros2018driftercoverage.pdf.

J. Hansen, F. Hogan, D. Rivkin, D. Meger, M. Jenkin, and G. Dudek. Visuotactile-rl: Learning multimodal manipulation policies with deep reinforcement learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8298–8304. IEEE, 2022a.

J. Hansen, K. Virji, T. Manderson, D. Meger, and G. Dudek. Drift-ncrn: A benchmark dataset for drifter trajectory prediction. https://github.com/johannah/drift-predict, 2022b.

J. K. Hart and K. Martinez. Environmental sensor networks: A revolution in the earth system science? *Earth-Science Reviews*, 78(3):177 – 191, 2006. ISSN 0012-8252.

P. Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. doi: 10.1109/tssc.1968.300136. URL https://doi.org/10.1109/tssc.1968.300136.

M. Hassan, Y. Guo, T. Wang, M. Black, S. Fidler, and X. B. Peng. Synthesizing physical character-scene interactions. *arXiv preprint arXiv:2302.00883*, 2023.

K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. URL http://arxiv.org/abs/1703.06870.

J. C. G. Higuera, D. Meger, and G. Dudek. Synthesizing neural network controllers with probabilistic model-based reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2538–2544. IEEE, 2018.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `http://dx.doi.org/10.1162/neco.1997.9.8.1735`.

D. O. Hodgins and S. L. Hodgins. *Phase II leeway dynamics program: development and verification of a mathematical drift model for liferafts and small boats*. Seaconsult Marine Research Limited, 1998.

F. R. Hogan and A. Rodriguez. Feedback control of the pusher-slider system: A story of hybrid and underactuated contact dynamics. In *Algorithmic Foundations of Robotics XII: Proceedings of the Twelfth Workshop on the Algorithmic Foundations of Robotics*, pages 800–815. Springer, 2020.

F. R. Hogan, J. Ballester, S. Dong, and A. Rodriguez. Tactile dexterity: Manipulation primitives with tactile feedback. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 8863–8869. IEEE, 2020.

F. R. Hogan, M. Jenkin, S. Rezaei-Shoshtari, Y. Girdhar, D. Meger, and G. Dudek. Seeing through your skin: Recognizing objects with a novel visuotactile sensor. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1218–1227, January 2021.

G. A. Hollinger and G. S. Sukhatme. Sampling-based motion planning for robotic information gathering. In *Robotics: Science and Systems*, Jun 2013. URL `http://robotics.usc.edu/publications/841/`.

G. A. Hollinger, S. Choudhary, P. Qarabaqi, C. Murphy, U. Mitra, G. S. Sukhatme, M. Stojanovic, H. Singh, and F. Hover. Underwater data collection using robotic sensor networks. *IEEE Journal on Selected Areas in Communications*, 30(5):899–911, 2012.

G. A. Hollinger, A. A. Pereira, J. Binney, T. Somers, and G. S. Sukhatme. Learning uncertainty in ocean current predictions for safe and reliable navigation of underwater vehicles. *Journal of Field Robotics*, 33(1):47–66, 2016.

Y. Hu, J. Liu, A. Spielberg, J. B. Tenenbaum, W. T. Freeman, J. Wu, D. Rus, and W. Matusik. Chainqueen: A real-time differentiable physical simulator for soft robotics. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6265–6271, 2019. doi: 10.1109/ICRA.2019.8794333.

Y. Hu, L. Anderson, T.-M. Li, Q. Sun, N. Carr, J. Ragan-Kelley, and F. Durand. Difftaichi: Differentiable programming for physical simulation. *ICLR*, 2020.

Y. Huang, Y. Yao, J. Hansen, J. Mallette, S. Manjanna, G. Dudek, and D. Meger. An autonomous probing system for collecting measurements at depth from small surface vehicles. In *OCEANS 2021: San Diego–Porto*, pages 1–6. IEEE, 2021.

T. Inanc, S. C. Shadden, and J. E. Marsden. Optimal trajectory generation in ocean flows. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 674–679, June 2005. doi: 10.1109/ACC.2005.1470035.

C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

N. Jakobi, P. Husbands, and I. Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *Advances in Artificial Life: Third European Conference on Artificial Life Granada, Spain, June 4–6, 1995 Proceedings 3*, pages 704–720. Springer, 1995.

M. Jakuba. Nui technical overview, 2014. Accessed: 2018-09-30.

K. M. Jatavallabhula, M. Macklin, F. Golemo, V. Voleti, L. Petrini, M. Weiss, B. Considine, J. Parent-Levesque, K. Xie, K. Erleben, L. Paull, F. Shkurti, D. Nowrouzezahrai, and S. Fidler. gradsim: Differentiable simulation for system identification and visuomotor

control. *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=c_E8kFWfhp0.

S. R. Jayne, D. Roemmich, N. Zilberman, S. C. Riser, K. S. Johnson, G. C. Johnson, and S. R. Piotrowicz. The argo program: Present and future. *Oceanography*, 30(2):18–28, 2017. ISSN 10428275, 2377617X. URL http://www.jstor.org/stable/26201840.

JPL. Nasa jpl satellites dissect powerful hurricane matthew. https://www.jpl.nasa.gov/news/news.php?feature=6643, 2016. Accessed: 2010-09-30.

S. J. Julier and J. K. Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–193. Spie, 1997.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

M. Kamachi and J. O'Brien. Continuous data assimilation of drifting buoy trajectory into an equatorial pacific ocean model. *Journal of Marine Systems*, 6(1):159 – 178, 1995. ISSN 0924-7963. Data Assimilation in Manine Science.

R. Kamphaus, M. Cronin, C. Sabine, S. Emerson, C. Meinig, and M. Robert. New surface mooring at station papa monitors climate. *PICES Press*, 16(2):26–27, 2008.

L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, Aug 1996. ISSN 1042-296X. doi: 10.1109/70.508439.

O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

O. Khatib. Inertial properties in robotic manipulation: An object-level framework. *The International Journal of Robotics Research*, 14(1):19–36, 1995. doi: 10.1177/027836499501400103.

K. Kim, D. Lee, and I. Essa. Gaussian process regression flow for analysis of motion trajectories. In *2011 International Conference on Computer Vision*, pages 1164–1171, Nov 2011. doi: 10.1109/ICCV.2011.6126365.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

J. C. Kinsey, R. M. Eustice, and L. L. Whitcomb. A survey of underwater vehicle navigation: Recent advances and new challenges. In *IFAC Conference of Manoeuvering and Control of Marine Craft*, volume 88, 2006.

G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007.

T. B. Koay, A. Raste, Y. H. Tay, Y. Wu, A. Mahadevan, S. P. Tan, J. Lim, M. Chitre, and C. N. Ong. Interactive monitoring in reservoirs using NUSwan – preliminary field results. *Water Practice and Technology*, 12(4):806–817, 12 2017. ISSN 1751-231X. doi: 10.2166/wpt.2017.089. URL https://doi.org/10.2166/wpt.2017.089.

L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006: 17th European Conference on Machine Learning Berlin, Germany, September 18-22, 2006 Proceedings 17*, pages 282–293. Springer, 2006.

L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *Proceedings of the 17th European Conference on Machine Learning*, ECML'06, pages 282–293, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-45375-X, 978-3-540-45375-8. doi: 10.1007/11871842_29.

V. Konda and J. Tsitsiklis. Actor-critic algorithms. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

K. Koreitem, F. Shkurti, T. Manderson, W.-D. Chang, J. C. G. Higuera, and G. Dudek. One-shot informed robotic visual search in the wild. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5800–5807. IEEE, 2020.

R. G. Krishnan, U. Shalit, and D. Sontag. Deep kalman filters, 2015. URL `https://arxiv.org/abs/1511.05121`.

O. B. Kroemer, R. Detry, J. Piater, and J. Peters. Combining active learning and reactive control for robot grasping. *Robotics and Autonomous systems*, 58(9):1105–1116, 2010.

D. Kularatne and A. Hsieh. Tracking attracting Lagrangian coherent structures in flows. In *Robotics: Science and Systems (RSS)*, 2015. doi: 10.15607/RSS.2015.XI.021.

D. Kularatne, S. Bhattacharya, and M. A. Hsieh. Optimal path planning in time-varying flows using adaptive discretization. *IEEE Robotics and Automation Letters*, 3(1):458–465, Jan 2018. doi: 10.1109/LRA.2017.2761939.

V. Kumar, D. Hoeller, B. Sundaralingam, J. Tremblay, and S. Birchfield. Joint space control via deep reinforcement learning. *International Conference on Intelligent Robots and Systems (IROS)*, 2021.

A. Kwok and S. Martínez. Coverage maximization with autonomous agents in fast flow environments. *Journal of Optimization Theory and Applications*, 155(3):986–1007, Dec 2012. ISSN 1573-2878. doi: 10.1007/s10957-012-0113-7. URL `https://doi.org/10.1007/s10957-012-0113-7`.

A. Kwok and S. Martínez. A coverage algorithm for drifters in a river environment. In *Proceedings of the 2010 American Control Conference*, pages 6436–6441, June 2010. doi: 10.1109/ACC.2010.5531467.

M. Lambeta, P.-W. Chou, S. Tian, B. Yang, B. Maloon, V. R. Most, D. Stroud, R. Santos, A. Byagowi, G. Kammerer, et al. Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. *IEEE Robotics and Automation Letters*, 5(3):3838–3845, 2020.

N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell. A survey of mobile phone sensing. *IEEE Communications Magazine*, 48(9):140–150, Sept 2010. ISSN 0163-6804. doi: 10.1109/MCOM.2010.5560598.

J. D. Langsfeld, K. N. Kaipa, R. J. Gentili, J. A. Reggia, and S. K. Gupta. Incorporating failure-to-success transitions in imitation learning for a dynamic pouring task. 2014.

S. M. Lavalle. Rapidly-exploring random trees: A new tool for path planning. Technical report, 1998a.

S. M. Lavalle. Rapidly-exploring random trees: A new tool for path planning. Technical report, 1998b.

A. X. Lee, A. Nagabandi, P. Abbeel, and S. Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *Advances in Neural Information Processing Systems*, 33:741–752, 2020.

M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg. Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8943–8950. IEEE, 2019.

N. E. Leonard, D. A. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. E. Davis. Collective motion, sensor networks, and ocean sampling. *Proceedings of the IEEE*, 95(1): 48–74, Jan 2007. ISSN 0018-9219. doi: 10.1109/JPROC.2006.887295.

S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International journal of robotics research*, 37(4-5):421–436, 2018.

R. Li and E. H. Adelson. Sensing and recognizing surface textures using a gelsight sensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning, 2019.

K. H. Low, J. M. Dolan, and P. Khosla. Adaptive multi-robot wide-area exploration and mapping. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, pages 23–30. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

R. Lowe. *Learning and evaluating neural network models for human-machine communication*. McGill University (Canada), 2020.

L. Luft, T. Schubert, S. I. Roumeliotis, and W. Burgard. Recursive decentralized collaborative localization for sparsely communicating robots. In *Proceedings of Robotics: Science and Systems*, AnnArbor, Michigan, June 2016. doi: 10.15607/RSS.2016.XII.016.

R. Lumpkin, M. Pazos, N. Oceanographic, and A. Administration. Measuring surface currents with surface velocity program drifters: the instrument, its data, and some re-

cent results||. chapter two of lagrangian analysis. In *and Prediction of Coastal and Ocean Dynamics*. University Press, 2007.

R. Lumpkin, N. Maximenko, and M. Pazos. Evaluating where and why drifters die. *Journal of Atmospheric and Oceanic Technology*, 29(2):300–308, 2012.

M. Lutter, C. Ritter, and J. Peters. Deep lagrangian networks: Using physics as model prior for deep learning. *arXiv preprint arXiv:1907.04490*, 2019.

M. Lutter, L. Hasenclever, A. Byravan, G. Dulac-Arnold, P. Trochim, N. Heess, J. Merel, and Y. Tassa. Learning dynamics models for model predictive agents. *arXiv preprint arXiv:2109.14311*, 2021a.

M. Lutter, J. Silberbauer, J. Watson, and J. Peters. A differentiable newton-euler algorithm for real-world robotics, 2021b.

D. Madeo, A. Pozzebon, C. Mocenni, and D. Bertoni. A low-cost unmanned surface vehicle for pervasive water quality monitoring. *IEEE Transactions on Instrumentation and Measurement*, 69(4):1433–1444, 2020. doi: 10.1109/TIM.2019.2963515.

Y. Madrid and Z. P. Zayas. Water sampling: Traditional methods and new approaches in water sampling strategy. *TrAC Trends in Analytical Chemistry*, 26(4):293–299, 2007. ISSN 0165-9936. doi: https://doi.org/10.1016/j.trac.2007.01.002. URL `https://www.sciencedirect.com/science/article/pii/S0165993607000039`. Emerging tools as a new approach for water monitoring.

K. Maenaka. Mems inertial sensors and their applications. In *2008 5th International Conference on Networked Sensing Systems*, pages 71–73, June 2008. doi: 10.1109/INSS.2008.4610859.

T. Maharaj. Generalizing in the real world with representation learning. *arXiv preprint arXiv:2210.09925*, 2022.

T. Manderson and G. Dudek. Gpu-assisted learning on an autonomous marine robot for vision based navigation and image understanding. In *OCEANS'18 MTS/IEEE Charleston*, pages 1–7, October 2018.

S. Manjanna. Multi-robot planning strategies for non-myopic spatial sampling. 2021.

S. Manjanna and G. Dudek. Data-driven selective sampling for marine vehicles using multi-scale paths. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6111–6117, 2017.

S. Manjanna, J. Hansen, A. Q. Li, I. Rekleitis, and G. Dudek. Collaborative sampling using heterogeneous marine robots driven by visual cues. In *2017 14th Conference on Computer and Robot Vision (CRV)*, June 2017a.

S. Manjanna, J. Hansen, A. Q. Li, I. Rekleitis, and G. Dudek. Collaborative sampling using heterogeneous marine robots driven by visual cues. In *2017 14th Conference on Computer and Robot Vision (CRV)*, pages 87–94. IEEE, 2017b.

S. Manjanna, A. Q. Li, R. N. Smith, I. Rekleitis, and G. Dudek. Heterogeneous multi-robot system for exploration and strategic water sampling. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4873–4880, 2018. doi: 10.1109/ICRA.2018.8460759.

S. Manjanna, H. Van Hoof, and G. Dudek. Policy search on aggregated state space for active sampling. In J. Xiao, T. Kröger, and O. Khatib, editors, *Proceedings of the 2018 International Symposium on Experimental Robotics*, pages 211–221, Cham, 2020. Springer International Publishing. ISBN 978-3-030-33950-0.

R. Martín-Martín, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg. Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1010–1017. IEEE, 2019.

N. Maximenko, J. Hafner, and P. Niiler. Pathways of marine debris derived from trajectories of lagrangian drifters. *Marine Pollution Bulletin*, 65(1):51 – 62, 2012. ISSN 0025-326X. doi: http://dx.doi.org/10.1016/j.marpolbul.2011.04.016. URL `http://www.sciencedirect.com/science/article/pii/S0025326X11002189`. At-sea Detection of Derelict Fishing Gear.

S. McCammon, G. Marcon dos Santos, M. Frantz, T. P. Welch, G. Best, R. K. Shearman, J. D. Nash, J. A. Barth, J. A. Adams, and G. A. Hollinger. Ocean front detection and tracking using a team of heterogeneous marine vehicles. *Journal of Field Robotics*, 38(6): 854–881, 2021.

M. Meghjani, S. Manjanna, and G. Dudek. Multi-target rendezvous search. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016*, pages 2596–2603. IEEE, 2016.

M. Meister, D. Dichek, A. Spears, B. Hurwitz, C. Ramey, J. Lawrence, K. Philleo, J. Lutz, J. Lawrence, and B. E. Schmidt. Icefin: redesign and 2017 antarctic field deployment. In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–5. IEEE, 2018.

D. Millard, E. Heiden, S. Agrawal, and G. S. Sukhatme. Automatic differentiation and continuous sensitivity analysis of rigid body dynamics. *arXiv preprint arXiv:2001.08539*, 2020.

C. Mitchell, G. Best, and G. Hollinger. Sequential stochastic multi-task assignment for multi-robot deployment planning. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3454–3460. IEEE, 2023.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 02 2015.

A. Molcard, A. Poje, and T. Özgökmen. Directed drifter launch strategies for lagrangian data assimilation using hyperbolic trajectories. *Ocean Modelling*, 12(3-4):268–289, 2006. ISSN 1463-5003. doi: 10.1016/j.ocemod.2005.06.004.

A. Molchanov, A. Breitenmoser, and G. S. Sukhatme. Active drifters: Towards a practical multi-robot system for ocean monitoring. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 545–552, May 2015. doi: 10.1109/ICRA.2015.7139232.

G. E. Monahan. State of the art—a survey of partially observable markov decision processes: theory, models, and algorithms. *Management science*, 28(1):1–16, 1982.

E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 363–370. IEEE, 2006.

M. Mozifian, A. Zhang, J. Pineau, and D. Meger. Intervention design for effective sim2real transfer. 2020.

K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima, and S. Kawatsuma. Emergency response to the nuclear accident at the fukushima daiichi nuclear power plants using mobile rescue robots. *Journal of Field Robotics*, 30(1):44–63, 2013. ISSN 1556-4967.

J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27 (6):737–757, 2008.

Y. Narang, B. Sundaralingam, M. Macklin, A. Mousavian, and D. Fox. Sim-to-real for robotic tactile sensing via physics-based simulation and learned latent projections. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6444–6451. IEEE, 2021.

National Oceanic and Atmospheric Administration Office of Ocean Exploration and Research. Multibeam sonar, 2017. URL https://oceanexplorer.noaa.gov/technology/sonar/multibeam.html.

National Oceanic and Atmospheric Administration Office of Ocean Exploration and Research. Acoustic doppler current profiler, 2023. URL https://oceanexplorer.noaa.gov/technology/acoust-doppler/acoust-doppler.html.

NOAA. AOML PhOD - Global Drifter Program. URL https://www.aoml.noaa.gov/phod/gdp/value_maps.php.

J. Oh, X. Guo, H. Lee, R. Lewis, and S. Singh. Action-conditional video prediction using deep networks in atari games. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 2863–2871, Cambridge, MA, USA, 2015. MIT Press.

OpenAI. Spinning up in deep rl, 2023. URL https://spinningup.openai.com/.

OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang. Solving rubik's cube with a robot hand, 2019.

M. Ouimet and J. Cortés. Coordinated rendezvous of underwater drifters in ocean internal waves. In *53rd IEEE Conference on Decision and Control*, pages 6099–6104, Dec 2014. doi: 10.1109/CDC.2014.7040344.

A. Padmanabha, F. Ebert, S. Tian, R. Calandra, C. Finn, and S. Levine. Omnitact: A multi-directional high-resolution touch sensor. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 618–624. IEEE, 2020.

R. Pascanu, Y. Li, O. Vinyals, N. Heess, L. Buesing, S. Racanière, D. P. Reichert, T. Weber, D. Wierstra, and P. Battaglia. Learning model-based planning from scratch. *CoRR*, abs/1707.06170, 2017.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

D. Pavllo, D. Grangier, and M. Auli. Quaternet: A quaternion-based recurrent model for human motion. In *British Machine Vision Conference (BMVC)*, 2018.

X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020.

A. A. Pereira, J. Binney, G. A. Hollinger, and G. S. Sukhatme. Risk-aware path planning for autonomous underwater vehicles using predictive ocean models. *Journal of Field Robotics*, 30(5):741–762, 2013. ISSN 1556-4967. doi: 10.1002/rob.21472. URL `http://dx.doi.org/10.1002/rob.21472`.

R. C. Perez, G. R. Foltz, R. Lumpkin, J. Wei, K. J. Voss, M. Ondrusek, M. Wang, and M. A. Bourassa. Oceanographic buoys: Providing ocean data to assess the accuracy of variables derived from satellite measurements. In *Field Measurements for Passive Environmental Remote Sensing*, pages 79–100. Elsevier, 2023.

L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel. Asymmetric actor critic for image-based robot learning, 2017.

O. Pizarro, S. B. Williams, M. Johnson-Roberson, D. Steinberg, and M. Bryson. Reef mapping and monitoring assisted by machines. In *Ocean Sciences 2014 Meeting*, 2014.

D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

C. Pontbriand, N. Farr, J. Hansen, J. C. Kinsey, L.-P. Pelletier, J. Ware, and D. Fourie. Wireless data harvesting using the auv sentry and whoi optical modem. In *OCEANS 2015-MTS/IEEE Washington*, pages 1–6. IEEE, 2015.

M. Posa, C. Cantu, and R. Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014.

C. Pradalier, S. Aravecchia, and F. Pomerleau. Multi-session lake-shore monitoring in visually challenging conditions. In *Proceedings of the Conference on Field and Service Robotics (FSR). Springer Tracts in Advanced Robotics*, 2019.

M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley Sons, 2014.

Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang. Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *Conference on Robot Learning*, pages 594–605. PMLR, 2023.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

M. Rahimi, M. Hansen, W. J. Kaiser, G. S. Sukhatme, and D. Estrin. Adaptive sampling for environmental field estimation using robotic sensors. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3692–3698, Aug 2005a. doi: 10.1109/IROS.2005.1545070.

M. Rahimi, M. Hansen, W. J. Kaiser, G. S. Sukhatme, and D. Estrin. Adaptive sampling for environmental field estimation using robotic sensors. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3692–3698, 2005b.

S. Reece, R. Mann, I. Rezek, and S. Roberts. Gaussian Process Segmentation of Co-Moving Animals. In A. Mohammad-Djafari, J.-F. Bercher, and P. Bessiére, editors, *American Institute of Physics Conference Series*, volume 1305 of *American Institute of Physics Conference Series*, pages 430–437, Mar. 2011. doi: 10.1063/1.3573650.

S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015. URL `http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal pdf`.

E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Winter Conference on Applications of Computer Vision*, 2020. URL `https://arxiv.org/pdf/1910.02190.pdf`.

C. Rocha, C. Tonetto, and A. Dias. A comparison between the denavit hartenberg and the screw-based methods used in kinematic modeling of robot manipulators. *Robotics and Computer-Integrated Manufacturing*, 27(4):723–728, 2011. ISSN 0736-5845. doi: https://doi.org/10.1016/j.rcim.2010.12.009. Conference papers of Flexible Automation and Intelligent Manufacturing.

A. Rosenfeld and M. Thurston. Edge and curve detection for visual scene analysis. *IEEE Transactions on Computers*, C-20(5):562–569, May 1971. ISSN 0018-9340. doi: 10.1109/T-C.1971.223290.

H. Salman, L. Kuznetsov, C. K. R. T. Jones, and K. Ide. A method for assimilating lagrangian data into a shallow-water-equation ocean model. *Monthly Weather Review*, 134

(4):1081–1101, 2006. doi: 10.1175/MWR3104.1. URL https://doi.org/10.1175/MWR3104.1.

H. Salman, K. IDE, and C. K. R. T. JONES. Using flow geometry for drifter deployment in lagrangian data assimilation. *Tellus A*, 60(2):321–335, 2008. ISSN 1600-0870. doi: 10.1111/j.1600-0870.2007.00292.x.

J. Schmidhuber. On learning to think: Algorithmic information theory for novel combinations of reinforcement learning controllers and recurrent neural world models. *CoRR*, abs/1511.09249, 2015.

B. Schmidt, P. Washam, P. Davis, K. Nicholls, D. Holland, J. Lawrence, K. Riverman, J. Smith, A. Spears, D. Dichek, et al. Heterogeneous melting near the thwaites glacier grounding line. *Nature*, 614(7948):471–478, 2023.

O. Schofield, J. Kohut, D. Aragon, L. Creed, J. Graver, C. Haldeman, J. Kerfoot, H. Roarty, C. Jones, D. Webb, et al. Slocum gliders: Robust and ready. *Journal of Field Robotics*, 24 (6):473–485, 2007.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

A. F. Shchepetkin and J. C. McWilliams. The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model. *Ocean Modelling*, 9:347–404, 2005.

F. Shkurti. *Algorithms and systems for robot videography from human specifications*. McGill University (Canada), 2019.

F. Shkurti, A. Xu, M. Meghjani, J. C. G. Higuera, Y. Girdhar, P. Giguere, B. B. Dey, J. Li, A. Kalmbach, C. Prahacs, K. Turgeon, I. Rekleitis, and G. Dudek. Multi-domain monitoring of marine environments using a heterogeneous robot team. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1447–1753, Portugal, Oct. 2012.

D. Silver. *Learning Preference Models for Autonomous Mobile Robots in Complex Domains*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, December 2010.

D. Silver, H. van Hasselt, M. Hessel, T. Schaul, A. Guez, T. Harley, G. Dulac-Arnold, D. P. Reichert, N. C. Rabinowitz, A. Barreto, and T. Degris. The predictron: End-to-end learning and planning. *CoRR*, abs/1612.08810, 2016.

D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. P. Lillicrap, K. Simonyan, and D. Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR*, abs/1712.01815, 2017a.

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–, Oct. 2017b.

A. Singh, A. Krause, C. Guestrin, W. J. Kaiser, and M. A. Batalin. Efficient planning of informative paths for multiple robots. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, volume 7, pages 2204–2211, 2007.

A. Singh, F. Ramos, H. D. Whyte, and W. J. Kaiser. Modeling and decision making in spatio-temporal processes for environmental surveillance. In *2010 IEEE International Conference on Robotics and Automation*, pages 5490–5497, May 2010. doi: 10.1109/ROBOT.2010.5509934.

R. N. Smith, Y. Chao, P. P. Li, D. A. Caron, B. H. Jones, and G. S. Sukhatme. Planning and implementing trajectories for autonomous underwater vehicles to track evolving ocean processes based on predictions from a regional ocean model. *The International Journal of Robotics Research*, page 0278364910377243, 2010.

N. N. Soreide, C. E. Woody, and S. M. Holt. Overview of ocean based buoys and drifters: present applications and future needs. In *MTS/IEEE Oceans 2001. An Ocean Odyssey.*

Conference Proceedings (IEEE Cat. No.01CH37295), volume 4, pages 2470–2472 vol.4, 2001. doi: 10.1109/OCEANS.2001.968388.

A. Srinivas, M. Laskin, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020.

R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

A. Stentz. The focussed d* algorithm for real-time replanning. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pages 1652–1659, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-363-8.

A. Stentz. *Optimal and Efficient Path Planning for Partially Known Environments*, pages 203–220. Springer US, Boston, MA, 1997. ISBN 978-1-4615-6325-9. doi: 10.1007/978-1-4615-6325-9_11.

W. Sun, N. Sood, D. Dey, G. Ranade, S. Prakash, and A. Kapoor. No-regret replanning under uncertainty. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6420–6427, May 2017. doi: 10.1109/ICRA.2017.7989758.

H. U. Sverdrup, M. W. Johnson, and R. H. Fleming. *The oceans, their physics, chemistry, and general biology, by H. U. Sverdrup, Martin W. Johnson and Richard H. Fleming*. Prentice-Hall, inc New York, 1942.

Y. Tassa, S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, and N. Heess. dmcontrol: Software and tasks for continuous control, 2020.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*, chapter 2: Recursive State Estimation, pages 13–39. The MIT Press, 2005. ISBN 0262201623.

S. Tian, F. Ebert, D. Jayaraman, M. Mudigonda, C. Finn, R. Calandra, and S. Levine. Manipulation by feel: Touch-based control with deep predictive models. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 818–824. IEEE, 2019.

A. Tinka, I. Strub, Q. Wu, and A. M. Bayen. Quadratic programming based data assimilation with passive drifting sensors for shallow water flows. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 7614–7620, Dec 2009. doi: 10.1109/CDC.2009.5399663.

M. I. Tiwana, S. J. Redmond, and N. H. Lovell. A review of tactile sensing technologies with applications in biomedical engineering. *Sensors and Actuators A: physical*, 179:17–31, 2012.

J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *IROS*, pages 5026–5033. IEEE, 2012. ISBN 978-1-4673-1737-5.

M. A. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. 2018.

J. van den Berg, D. Ferguson, and J. Kuffner. Anytime path planning and replanning in dynamic environments. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2366–2371, May 2006. doi: 10.1109/ROBOT.2006.1642056.

H. Van Hoof, N. Chen, M. Karl, P. van der Smagt, and J. Peters. Stable reinforcement learning with autoencoders for tactile and visual data. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 3928–3934. IEEE, 2016.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.

M. B. Villalonga, A. Rodriguez, B. Lim, E. Valls, and T. Sechopoulos. Tactile object pose estimation from the first touch with geometric contact rendering. In *Conference on Robot Learning*, pages 1015–1029. PMLR, 2021.

G. Villarini, P. V. Mandapaka, W. F. Krajewski, and R. J. Moore. Rainfall and sampling uncertainties: A rain gauge perspective. *Journal of Geophysical Research: Atmospheres*, 113(D11), 2008.

C. Wang, S. Wang, B. Romero, F. Veiga, and E. Adelson. Swingbot: Learning physical features from in-hand tactile exploration for dynamic swing-up manipulation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5633–5640. IEEE, 2020.

C. Wang, D. Gao, K. Xu, J. Geng, Y. Hu, Y. Qiu, B. Li, F. Yang, B. Moon, A. Pandey, Aryan, J. Xu, T. Wu, H. He, D. Huang, Z. Ren, S. Zhao, T. Fu, P. Reddy, X. Lin, W. Wang, J. Shi, R. Talak, K. Cao, Y. Du, H. Wang, H. Yu, S. Wang, S. Chen, A. Kashyap, R. Bandaru, K. Dantu, J. Wu, L. Xie, L. Carlone, M. Hutter, and S. Scherer. PyPose: A library for robot learning with physics-based optimization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

S. Wang, M. Lambeta, P.-W. Chou, and R. Calandra. Tacto: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors. *IEEE Robotics and Automation Letters*, 7(2):3930–3937, 2022.

C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

G. Webster. Nasa's mars curiosity debuts autonomous navigation. https://mars.nasa.gov/news/nasas-mars-curiosity-debuts-autonomous-navigation/, 2013. Accessed: 2018-09-30.

N. Wettels, V. J. Santos, R. S. Johansson, and G. E. Loeb. Biomimetic tactile sensor array. *Advanced Robotics*, 22(8):829–849, 2008.

G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou. Information theoretic mpc for model-based reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1714–1721. IEEE, 2017.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992a. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement learning*, pages 5–32, 1992b.

S. B. Williams, O. Pizarro, J. M. Webser, R. J. Beaman, I. Mahon, M. Johnson-Roberson, and T. Bridge. AUV-assisted Surveying of Drowned Reefs on the Shelf Edge of the Great Barrier Reef, Australia. *Journal of Field Robotics*, 27(5):675–697, 2010.

T. C. Wilson, J. A. Barth, S. D. Pierce, P. M. Kosro, and B. W. Waldorf. A lagrangian drifter with inexpensive wide area differential gps positioning. In *OCEANS 96 MTS/IEEE Conference Proceedings. The Coastal Ocean - Prospects for the 21st Century*, volume 2, pages 851–856 vol.2, Sep 1996. doi: 10.1109/OCEANS.1996.568340.

J. Wong, V. Makoviychuk, A. Anandkumar, and Y. Zhu. Oscar: Data-driven operational space control for adaptive and robust robot manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

X. Wu, M. Liu, and Y. Wu. In-situ soil moisture sensing: Optimal sensor placement and field estimation. *ACM Trans. Sen. Netw.*, 8(4):33:1–33:30, Sept. 2012. ISSN 1550-4859.

F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese. Gibson Env: real-world perception for embodied agents. In *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018.

D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021a.

D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021b.

D. Yarats, I. Kostrikov, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021c. URL `https://openreview.net/forum?id=GY6-6sTvGaf`.

D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10674–10681, 2021d.

D. Yarats, D. Brandfonbrener, H. Liu, M. Laskin, P. Abbeel, A. Lazaric, and L. Pinto. Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning. *arXiv preprint arXiv:2201.13425*, 2022.

T. Yu, A. Kumar, Y. Chebotar, K. Hausman, C. Finn, and S. Levine. How to leverage unlabeled data in offline reinforcement learning. In *International Conference on Machine Learning*, pages 25611–25635. PMLR, 2022.

W. Yu, J. Tan, C. K. Liu, and G. Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017.

W. Yu, N. Gileadi, C. Fu, S. Kirmani, K.-H. Lee, M. G. Arenas, H.-T. L. Chiang, T. Erez, L. Hasenclever, J. Humplik, et al. Language to rewards for robotic skill synthesis. *arXiv preprint arXiv:2306.08647*, 2023.

W. Yuan, S. Dong, and E. H. Adelson. Gelsight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017.

Y. Yuan and V. Makoviychuk. Learning physically simulated tennis skills from broadcast videos. 2023.

A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi. Internet of things for smart cities. *IEEE Internet of Things Journal*, 1(1):22–32, Feb 2014. ISSN 2327-4662. doi: 10.1109/JIOT.2014.2306328.

A. Zhang, A. Lerer, S. Sukhbaatar, R. Fergus, and A. Szlam. Composable planning with attributes. *CoRR*, abs/1803.00512, 2018.

Y. Zhao, F. Yin, F. Gunnarsson, F. Hultkratz, and J. Fagerlind. Gaussian processes for flow modeling and prediction of positioned trajectories evaluated with sports data. In *2016 19th International Conference on Information Fusion (FUSION)*, pages 1461–1468, July 2016.

M. Zhou, R. Bachmayer, and B. deYoung. Mapping the underside of an iceberg with a modified underwater glider. *Journal of Field Robotics*, 36(6):1102–1117, 2019.

X. Zhou, X. Sun, W. Zhang, S. Liang, and Y. Wei. Deep kinematic pose regression. *CoRR*, abs/1609.05317, 2016. URL `http://arxiv.org/abs/1609.05317`.

Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. *CoRR*, abs/2009.12293, 2020. URL `https://arxiv.org/abs/2009.12293`.