THE UNIVERSITY
of EDINBURGH

Architectures for online simulation-based inference applied to robot motion planning

Martin Andreev Asenov

Doctor of Philosophy

Institute of Perception, Action and Behaviour

School of Informatics

University of Edinburgh

2022

Abstract

Robotic systems have enjoyed significant adoption in industrial and field applications in structured environments, where clear specifications of the task and observations are available. Deploying robots in unstructured and dynamic environments remains a challenge, being addressed through emerging advances in machine learning. The key open issues in this area include the difficulty of achieving coverage of all factors of variation in the domain of interest, satisfying safety constraints, etc. One tool that has played a crucial role in addressing these issues is simulation - which is used to generate data, and sometimes as a world representation within the decision-making loop.

When physical simulation modules are used in this way, a number of computational problems arise. Firstly, a suitable simulation representation and fidelity is required for the specific task of interest. Secondly, we need to perform parameter inference of physical variables being used in the simulation models. Thirdly, there is the need for data assimilation, which must be achieved in real-time if the resulting model is to be used within the online decision-making loop. These are the motivating problems for this thesis.

In the first section of the thesis, we tackle the inference problem with respect to a fluid simulation model, where a sensorised UAV performs path planning with the objective of acquiring data including gas concentration/identity and IMU-based wind estimation readings. The task for the UAV is to localise the source of a gas leak, while accommodating the subsequent dispersion of the gas in windy conditions. We present a formulation of this problem that allows us to perform online and real-time active inference efficiently through problem-specific simplifications.

In the second section of the thesis, we explore the problem of robot motion planning when the true state is not fully observable, and actions influence how much of the state is subsequently observed. This is motivated by the practical problem of a robot performing suction in the surgical automation setting. The objective is the efficient removal of liquid while respecting a safety constraint - to not touch the underlying tissue if possible. If the problem were represented in full generality, as one of planning under uncertainty and hidden state, it could be hard to find computationally efficient solutions. Once again, we make problem-specific simplifications. Crucially, instead of reasoning in general about fluid flows and arbitrary surfaces, we exploit the observations that the decision can be informed by the contour tree skeleton of the volume, and the configurations in which the fluid would come to rest if unperturbed. This allows us

to address the problem as one of iterative shortest path computation, whose costs are informed by a model estimating the shape of the underlying surface.

In the third and final section of the thesis, we propose a model for real-time parameter estimation directly from raw pixel observations. Through the use of a Variational Recurrent Neural Network model, where the latent space is further structured by penalising for fit to data from a physical simulation, we devise an efficient online inference scheme. This is first shown in the context of a representative dynamic manipulation task for a robot. This task involves reasoning about a bouncing ball that it must catch – using as input the raw video from an environment-mounted camera and accommodating noise and variations in the object and environmental conditions. We then show that the same architecture lends itself to solving inference problems involving more complex dynamics, by applying this to measurement inversion of ultrafast X-Ray scattering data to infer molecular geometry.

Acknowledgements

I truly had a great time in the last few years, and have many people to be thankful to:

First of all, to my supervisor Subramanian Ramamoorthy, who guided me through interesting research avenues and ideas, but also for stepping back at times and letting me explore my own ideas, for establishing a lab environment with access not only every to every robot, sensor or computation hardware anyone might need, but also where you can feel at home, but most of all for always being understanding, even when he disagrees.

To my secondary supervisor Kartic Subr, for his critical comments, determination, and help in shaping my ideas. From the numerous meetings we have had throughout the years to key contributions in times of quickly approaching deadlines, the help was well above and beyond any of my expectations.

To Michael Burke, whose help was vital in the second part of my PhD - for his encouragement and great discussions as well as practical help with experiments and writing.

To all the members of the RAD group, for all the pleasant lunch discussions! Special thanks to Michael, Yordan, Todor, George, Arturas and Daniel for their comments on the draft of the thesis.

To my family, for all their support throughout the years. Not everyone has the great luxury of spending so many years in academia. For all the opportunities and support they have given me prior to, and during my PhD, I am very grateful.

To Ivayla Ivanova, my partner and best friend in life. I thank her for all the understanding, countless hours of help, and patience.

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC), as part of the CDT in RAS at Heriot-Watt University and The University of Edinburgh under Grant reference EP/L016834/1.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Martin Andreev Asenov*)

Publications

The following publications have been composed during the course of this doctorate:

M. Asenov, M. Rutkauskas, D.T. Reid, K. Subr, S. Ramamoorthy, Active localization of gas leaks using fluid simulation, *IEEE Robotics and Automation Letters*, Vol 4(2): 1776-1783, 2019. [12]

M. Rutkauskas, **M. Asenov**, S. Ramamoorthy, D.T. Reid, Autonomous multi-species environmental gas sensing using drone-based Fourier-transform infrared spectroscopy, *Optics Express*, Vol. 27, Issue 7, pp. 9578-9587, 2019. [189]

M. Asenov, M. Burke, D. Angelov, T. Davchev, K. Subr, S. Ramamoorthy, Vid2Param: Modelling of dynamics parameters from video, *IEEE Robotics and Automation Letters*, Vol 5(2): 414-421, 2020. [11]

M. Asenov, N. Zotev, S. Ramamoorthy, A. Kirrander, Inversion of ultrafast X-ray scattering with dynamics constraints, In Proc. *NeurIPS Workshop on Machine Learning and the Physical Sciences*, 2020. [13]

To my family.

Table of Contents

List of Figures

Acronyms

BO Bayesian Optimization. 20, 25, 26, 36, 43, 47

GP Gaussian Processes. 11, 24, 28, 52, 54, 59, 62, 64

NN Neural Networks. 5, 11, 13, 17, 20, 23, 25, 26, 28, 29, 90, 92, 93

SBI Simulation-Based Inference. 4, 5, 6, 22, 23, 91, 92, 93, 94

TSP Travelling Salesman Problem. 52, 54

UAV Unmanned Aerial Vehicle. 7, 9, 11

UQ Uncertainty Quantification. 5, 24, 25

VAE Variational Autoencoder. 5, 72, 74

Chapter 1

Introduction

1.1 Preface

Robotic systems have seen rapid development in the last few decades, driven by parallel progress in algorithms and modelling, hardware design, computational improvements, software developments tools, and practices. Those developments have led to broad deployment in different industrial environments, where precise specifications of the required tasks are available, such as assembly lines and car factories. However, few autonomous commercial systems are deployed in cases where reasoning about uncertainty in the surrounding environment is required. Many challenges are present such as acquiring necessary training data that captures the full complexity of the observed phenomena, deployment in a safe manner, and coming up with approaches for fast and accurate inference of parameters of interest from diverse sensory signals - fig.1.1.

There is an ever-growing interest in deploying robotic systems in environments where the underlying dynamics of the environment may not be fully known. Imposing physically grounded bias is vital for learning effectively in order to achieve generalisation with respect to task variability. It is critical autonomous robots can identify their surroundings, plan and act in them. Applications range from everyday interactions, where a robot should have a notion of different objects and their properties, to hazardous situations such as gas leakages, where reasoning about gases and wind is required, to medical domains, where among other things, one must manipulate liquids. Those problems share the common property that it might be hard to acquire diverse real-world data to learn robust policies. On the other hand, recent advances in computer graphics have made it possible to simulate many of the above-mentioned tasks [141]. It would be beneficial to have robotic systems able to learn in simulation to be then able to adapt

to the real world.

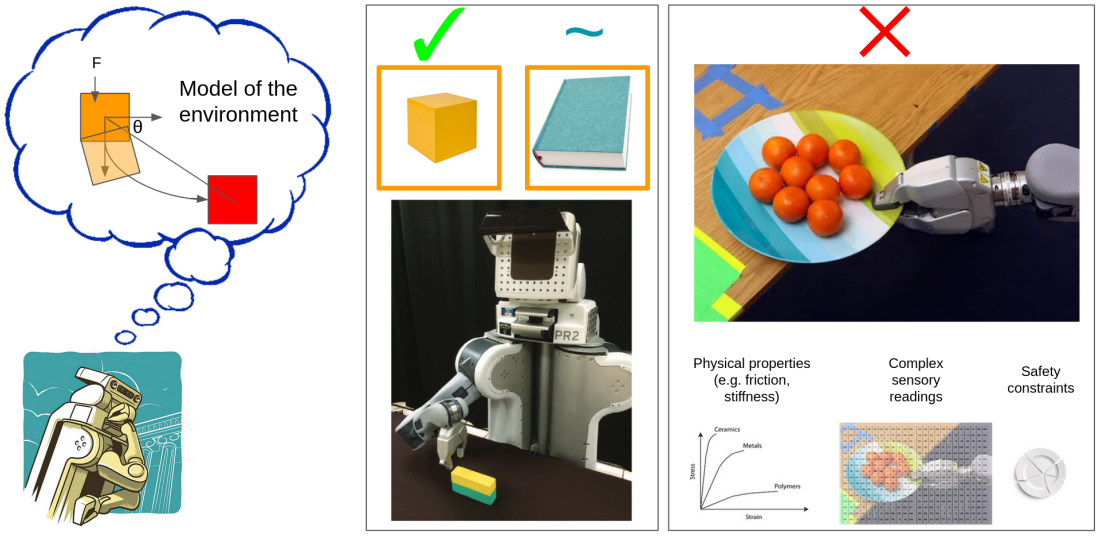


Figure 1.1: **Manipulation of objects.** Robots deployed in the real world need robust policies to deal with various complexities - visual observations, safety constraints, and objects with different physical properties. (Figure adapted with images from [211, 230, 2, 1])

Even the seemingly simplest of tasks often require reasoning about the dynamics of the environment and associated physical properties - see fig.1.2. Let us take as an example pick-n-place - a problem explored for at least a few decades [216]. While one can imagine a solution for trivial cases of simple rigid objects such as cubes, the problem can quickly become infeasible when considering all possible object variations. What if you need to grab and move a glass full of water, a Jenga block, a banana? Even a 'simple' pick-n-place task requires reasoning over grasping regions, arm movement speed, and strength of the grasp. While it is possible to come up with self-resetting environments that can make it possible to acquire real-world data [232], it is often challenging to do so due to safety reasons, a vast diversity of real-world objects, etc. At the same time, in different areas such as NLP, it has been observed that models can learn and do substantially improve with more and more data, with no clear ceiling of how much data can be absorbed [34]. The complexity of acquiring diverse sensory readings from a robotic system, coupled with the data hunger of modern machine learning methods, has led to the natural alternative of using simulations [171].

Coming up with suitable representation models of real-world phenomena is a big challenge of its own. While different physical aspects of our world, such as rigid objects, light, fluids, soft materials, etc., can be simulated, most of those approaches tend to be

approximate. As such, there is often a discrepancy between the actual observed data and the simulation model used. Even if such a discrepancy is limited or addressed [32], one must still infer the state of the simulation from some indirect measurements from the real world. This inference is not only hard but sometimes even ill-posed as multiple different observations can correspond to the same underlying setup (e.g., trying to infer volume from a single image).

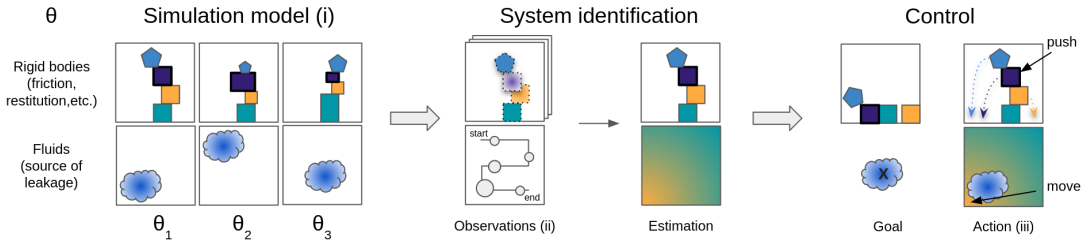


Figure 1.2: **Illustration of the main questions researched within this thesis.** This thesis studies the use of computational models for system identification and control in uncertain environments, including **(i)** the use of suitable complexity of simulation models, and specific sampling strategies, **(ii)** addressing system identification from complex sensory observations and **(iii)** planning under uncertainty and online inference.

1.1.1 Simulations as a Data Source

Simulation is the imitation of the operation of a real-world process or system over time [18]. A plethora of simulation models have been proposed and used by different communities, modelling different aspects of our world - from the dynamics of unobserved to the human eye phenomena like molecular dynamics to different aspects of any real-world scene such as in animated movies and games. To create seemingly complete representations, often multi-stage pipelines are used. For example, in a typical animation movie those are modeling and setup; animation; VFX, lighting and rendering; compositing.

Within the robotics community, tradionally some of the phenomena of interest include: multi-joint dynamics with contacts [219, 48], motion planning and inverse kinematics [209]. For a robotics system to be able to fully deal with the uncertainty of our surrounding world, it should be able to also reason about the *dynamics of the environment* - fluids [204], soft objects [141], different light conditions [173], and others. Simulation pipelines often requires a lot of computational resources (animation

movies) or produces approximate results when used in online settings (games). Within the robotics community, there is often the need for simulations to be fast *and* accurate - as such they are often constrained within specific dynamics.

When deciding on suitable simulation representations, it is essential to consider the task's requirements and *source of uncertainty* - whether variations in the task of interest come from appearance or dynamics. It is also often not trivial to decide on the simulation fidelity needed. Rendering a high fidelity image with multiple sources of light and multiple objects with different textures can take many hours, which might be infeasible when many images or even videos are needed.

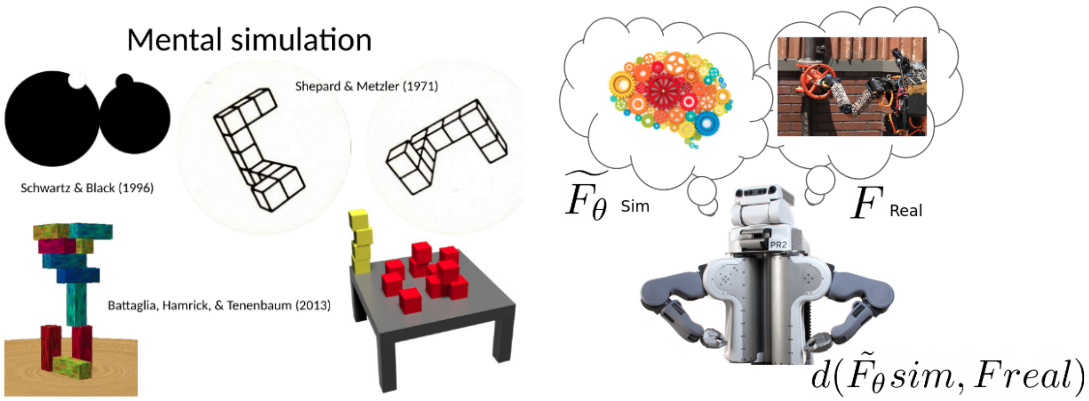
Computational requirements are essential, even for generating a static dataset, depending on the ratio of data needed and processing time. They often become critical if the simulation is to be used online as part of the optimization process. There are strict time requirements for how fast the simulation must be, as multiple simulation calls are often required as part of online control.

1.1.2 Simulation-Based Inference and Planning

Having a defined simulation model, we need to align it to the observed data, or 'invert' the data to estimate necessary parameters of interest. It is often hypothesized that people perform a similar alignment process by using approximate mental models of their surrounding world - as demonstrated in different experiments in the past few decades [200, 201, 23], e.g., are the two rotated objects the same; are the stacked tower of blocks going to fall as in fig.1.3.

Similar methodology has been explored and applied to many practical problems under different names - most notably system identification and Simulation-Based Inference (SBI). System identification is the process of identifying the governing equations and/or their parameters through sensory readings. It is a mature field, with the main two approaches focusing on linear time-invariant systems (LTI) and prediction-error approaches [134].

With the recent success of large-scale methods, SBI has seen increased interest in various fields such as biology, chemistry, and physics [49]. Two broad sets of methods are used - inference via sampling and comparing with observed data or amortizing the process by learning a model with simulated data traces. The most well-known standard approach is Approximate Bayesian Computation (ABC). In its simplest form, it is sampling simulation parameters θ (uniformly) and comparing the simulated data



(a) Mental simulation models (b) Aligning a simulation to observations

Figure 1.3: **Intuitive physics models in robotics.** (a) It has been repeatedly hypothesized that we use a mental simulation model to perform inference and future prediction [200, 201, 23] (b) This work explores methods for simulation alignment between the real observed world F and an internal simulation model $\widetilde{F}_\theta$

trace with the actual observed data - if the two traces are sufficiently close, the sample is accepted. The main drawback of ABC is slow convergence whenever the parameters of interest have high dimensions and/or the simulator is computationally expensive. Alternative methods have been proposed around the ideas of sampling around previous high likely points (Markov Chain Monte Carlo) or exploiting existing knowledge of the unknown probability density by sampling from a known distribution suspected to have a similar shape (Importance Sampling).

Regardless of the sampling techniques used, often for computationally expensive simulations and requirements for online inference, directly using the simulation is infeasible. Amortization of the inference process is a more well-suited approach, as computationally expensive steps do not need to be repeated at test time. Different approaches again exist, focusing on amortizing different parts of the inference process, such as the likelihood or the posterior. The development of different generative Neural Networks (NN), such as Variational Autoencoder (VAE), has only amplified the interest in the topic due to the general ability of NN models to handle high dimensional data and resolve time dependencies.

SBI is closely related to the topic of uncertainty propagation and optimization [169]. SBI can be rephrased as inverse Uncertainty Quantification (UQ) of the parameters of interest [49]. From a planning perspective, an important question is uncertainty

propagation, as robot actions are optimised for a given cost function. Those can often be solved again by sampling or by estimating a bound directly[73].

Robotics has some unique challenges, governed by the requirements for acting in a real environment. The inference process must proceed in real-time while accounting for the potential variability in the observations. There are almost always strict safety requirements for the executed actions, which must be incorporated into the synthesized control strategies. If we take the example of a pick-n-place - a glass full of liquid, a Jenga block, a banana - depending on the type of object, we might want to take different approaches and address complexities in physics, planning, and observations used.

To move a glass full of liquid, we must maintain suitable grasping regions without dropping the glass. Moreover, as we start moving the glass, we need to be careful of the robot arm acceleration to avoid sloshing - for example, gradually accelerating and then decelerating as we approach the target location. This process requires reasoning over the liquid's viscosity and other physical properties, which then determines the dynamics throughout the execution of the trajectory. As such, we can complete the task as quickly as possible while not spilling any liquid.

If we want to remove a Jenga block, we need to reason over the probability of the tower falling as we remove individual blocks. Often robot movements are imprecise. As the block of choice is removed, re-planning might be needed to correct the movement of the arm. To be playing optimally, another aspect to consider is other agents' behavior, accuracy, and internal model of selecting a block. A robot must deal with all this associated uncertainty and re-plan based on the current estimation of the environment.

Visual cues are crucial in real-world interactions to infer different physical properties such as density and elasticity. Grasping different objects like a banana or a stone requires knowledge of the force that needs to be applied - weak as not to destroy it, but strong enough not to drop it [72]. While this can be determined iteratively as the manipulation proceeds, it should be inferred directly from sensory observations for safety and efficiency reasons.

Those examples illustrate just some of the challenges associated with robotics systems acting in the real world. While SBI is a promising approach, different complexities need to be addressed, such as calibration with a computationally expensive simulator, planning from limited observations with safety constraints and, dealing with complex sensory inputs such as images. Notably, the inference and planning must often proceed online, as the robotic system interacts in the environment.

1.2 Problem Statement

This research is concerned with the problem of *online learning* in robotic systems based on an internal *model of the underlying physics*. The problem is to design techniques that exploit the available knowledge about the observed process by performing an online identification of the dynamics of the environment while completing the specified task. More specifically, we focus on tasks with *complex dynamics*, *uncertain environments*, or *rich sensory observations*, where it is crucial that the task is completed within a *limited time*. The core problem is that of *aligning* an existing model of physical interactions to the acquired observations - see fig.1.4.

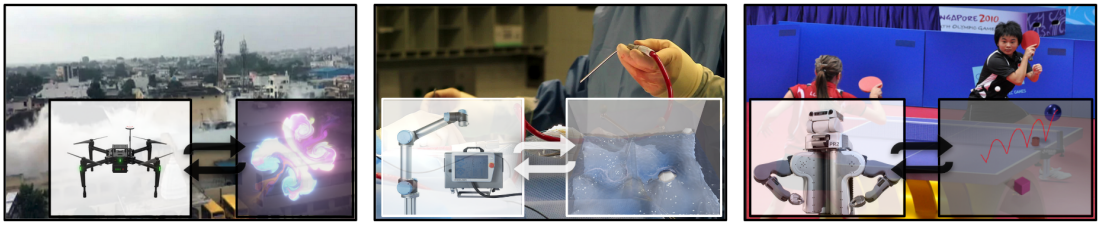


Figure 1.4: **Projects overview.** We propose different experimental setups with increasing complexities - (left) gas localization with a UAV using fluid simulation (center) autonomous suction of fluids for medical applications (right) modelling of dynamics parameters from video.

We are concerned with the question of incorporating known physics knowledge as a bias for various robotics tasks. In a typical robotics platform, we have access to different sensory inputs such as images, depth sensors, etc., which can be unified as a function F, which can sample the real world. Within this work, we assume to have an approximate model of the environment and respectively function $\widetilde{F}_\theta$, which can sample this approximate model (e.g., implemented as a computer graphics engine). We are interested in minimizing the mismatch between the observations from F and $\widetilde{F}_\theta$, and subsequently, use the optimal found parameters θ^* for planning.

The contribution of this work is the design and evaluation of *simulation alignment* methods of different robotics problems with varying *physical, planning, and observational complexity*, with the explicit goal of system identification and corresponding optimal planning. We study three different main areas, motivated by their varying complexities - see fig.1.5. First, we look into the problem of online gas localization by using fluid simulation as a model (*high physical complexity*). Secondly, we look into the problem of autonomous suction of liquids from partial observations of an unknown sur-

face (*high planning complexity*). Finally, we look into estimating physically meaningful parameters, such as restitution, from videos (*high observational complexity*).

1.3 Thesis Overview

This thesis explores the question of simulation alignment to tackle uncertain and dynamic environments by optimizing a distance metric between observed data and the model of interest. More specifically, we look into addressing the beforementioned complexities by adapting simulation based inference approaches. In the first part of the thesis, we look into problems with high degree of complexity in the underlying simulation model or planning procedure. To address those challenges, we study the use of reduced simulation representation and efficient sampling strategies for online inference. In the second part of the thesis, we focus on problems with rich sensory observations such as videos or other dense time series of sensory readings. We study methods for amortizing the inference process by learning an inverse model from data traces.

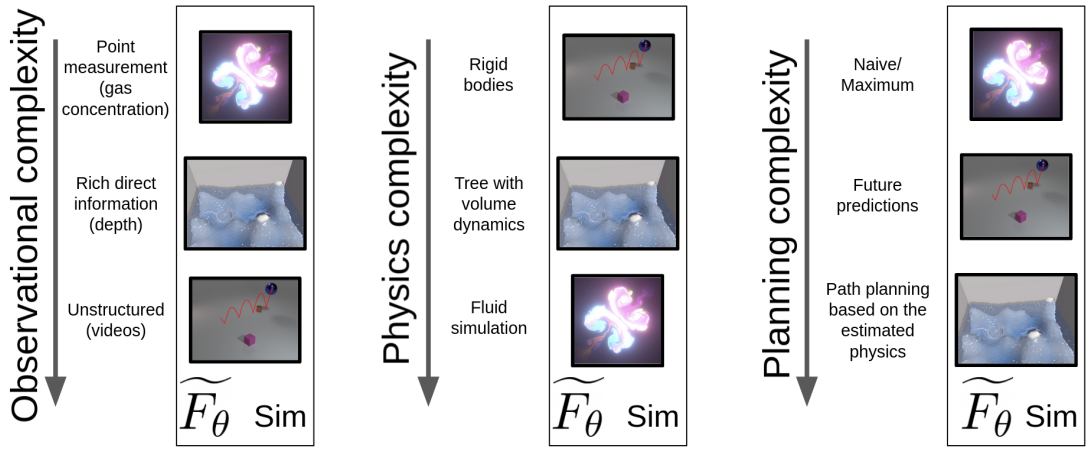


Figure 1.5: **Simulation alignment challenges.** Challenges associated with using a simulation as a model: (Left) Observational complexity - the type of sensory input used for alignment, varying from direct point measurements to unstructured input such as videos; (Middle) Physics complexity - the expressiveness of the simulation from rigid bodies to fluids; (Right) Planning complexity - estimating an optimal action based on the inference of the physical parameters.

1.3.1 Active Localization of Gas Leaks using Fluid Simulation

Simulation Alignment with Computationally Expensive Simulator

In this project, we are interested in performing system identification from direct measurements such as gas readings with a computationally expensive simulator. We study this in the context of localizing an open field gas leakage with a Unmanned Aerial Vehicle (UAV) by employing a fluid simulation as a model. Within the above-defined complexities, this problem is characterized by high physical complexity (fluid simulation with variable wind, gas dispersion, etc.), low observational complexity (direct gas concentration readings from a sensor and wind estimations from the UAV) and low planning complexity (go to the predicted source of the leakage). The UAV can sample the world for gas concentration g and wind speed and direction estimates w for a given location s and timestep t:

$$g, w = F(s,t) \tag{1.1}$$

We can also sample for gas measurements using a simulator for a defined location of the leakage l, where $\widetilde{W} = \{w\}$ is the spatio-temporal wind field and θ is any remaining parameters of the simulation (e.g., gas dispersion, simulation timestep, etc.):

$$\widetilde{g} = \tilde{F}(s,t,l|\theta,\tilde{W}) \tag{1.2}$$

We can compare the actual readings g with the simulated readings $\widetilde{g}$ to find the optimal placement of the gas source l^*, where d is a distance metric (e.g. MSE).

$$l^* = \sum d(g,\tilde{g}) \tag{1.3}$$

Within this work, we exploit the fact that the optimization can be performed by comparing the shifted simulated readings of a bigger simulation with the acquired readings, rather than running multiple simulations with different locations of the source:

$$\tilde{F}(l+\Delta l, s, t|\theta,\tilde{W}) = \tilde{F}(l, s+\Delta l, t|\theta,\tilde{W}) \tag{1.4}$$

The two sampling procedures are equivalent for spatially constant wind (but varying temporally); however, the second is much computationally cheaper. The inference speed is a critical requirement that allows performing this optimization online onboard of the UAV.

1.3.2 RoboSuction: Safe and Efficient Suction under Partial Observations

Safe Planning under Partial Observations with Reduced Simulation Representation

In this project, we are interested in performing system identification with challenging planning due to the ever-changing belief over the state of the phenomena. In practical terms, we study the problem of medical suction, where we need to clean all present liquid as quickly as possible while maintaining a distance from the surface underneath. The complexity comes from precisely this safety constraint while re-estimating the underlying surface and dynamics as suction is performed.

We have a robotic system able to go to a certain position s at a timestep t, which then after suction reveals partial observations of the uncovered surface l.

$$l = F(s,t) \tag{1.5}$$

Within this work, we propose an estimation of the surface $\tilde{h}$, additionally describing some of the volume information.

$$\tilde{h} = G(l) \tag{1.6}$$

The main contribution then becomes of coming up with a method $\hat{F}$, that given the surface estimation, and a safety threshold ε, estimates an optimal action a^* that would result is cleaning the remaining liquid in the shortest amount possible.

$$a^* = \hat{F}(\tilde{h}, \varepsilon) \tag{1.7}$$

1.3.3 Vid2Param: Modelling of Dynamics Parameters from Video

Simulation Alignment from Unstructured Sensory Input

In this project, we are interested in performing system identification of different physical parameters from unstructured observations such as video. Similar to the previous projects, we have a function able to acquire real-world readings at timestep t, albeit this time we assume no excitation. However, the resulting observations I have complex structure (rich, dense measurements such as images).

$$I = F(t) \tag{1.8}$$

Similarly, a simulation parameterized by θ can be used to generate simulated readings:

$$\tilde{I} = \tilde{F}(t, \theta) \tag{1.9}$$

The core problem we address is that of estimating θ, based on the difference between the simulated $\tilde{I}$ and observed I images.

$$\theta^* = \sum d(I, \tilde{I}) \tag{1.10}$$

The main contribution of this work is a model that can estimate encode information h from a sequence of simulated readings $\tilde{I}$, and based on that infer parameters of interest θ.

$$\theta, h_t = \hat{F}(\tilde{I}_t, h_{t-1}) \tag{1.11}$$

1.4 Major Contributions

The first part of this thesis looks into sampling based approaches and reduced simulation representations for solving inverse problems in robotics. The second part of the thesis looks into amortizing the inference process by learning an inverse mapping directly.

- An online algorithm for efficient sampling of a fluid simulation model for gas localization with a UAV. Under the assumption that wind is spatially constant (but temporally varying), running multiple simulations with varying locations of the gas leakage is equivalent to running a four times bigger simulation with the source in the center and taking shifted readings. The computational speed up from the latter allows the algorithm to be run online onboard of a UAV.

- An algorithm for sampling contour trees and GPs for a safe path for liquid clearning under partial observations. Under the assumption of stationarity of liquids and smooth underlying surface, the simulation can be reducted to a countour tree. The reduced fluids representation, combined with a surface and safety bound estimation, allows for planning for efficient cleaning of the remaining liquid, while avoiding hitting the surface.

- A NN model for learning inverse mappings of rich sensory inputs to parameters of interests. Amortizing the inference allows for online identification of physical parameters, while achieving better accuracy than traditional methods. By explicitly factoring in information from previous timesteps into the learning process,

the methods allows for accurate inversion in challanging domains such as robot control and molecular dynamics.

Chapter 2

Background

This thesis explores the question of using simulations as models to solve robotics tasks in dynamic and uncertain environments. This can proceed by using the simulation online as part of the control loop or as data for a learned model.

The renewed interest in learning-based methods is due to a combination of scaling up datasets [54], and adapting NN models to alternative forms of computation such as GPUs, while refining the structure and the training the process to address the challenges of large and complex datasets [120]. Those trends have impacted robotics research as a whole, although due to unique challenges in the field, it is less clear how and if learning-based methods should be used compared to traditional approaches. A significant challenge is acquiring real-world data to address the diversity of potential observations while the system is deployed safely. Using simulation has emerged as a natural alternative, albeit such methodology brings its own set of challenges [235].

A very general simulation model can be defined of the form:

$$x_t = f(x_{t-1}, u_{t-1}|\theta) \tag{2.1}$$

, where the simulation is defined as a function f, with state x_t, actions u_t and simulation parameters θ. This chapter starts by providing a broad overview of simulation models f (Sec.2.1) used for simulating robot and environment dynamics. Traditional simulators are covered, where the equations of motion are analytically defined. The section finishes with a discussion on learnable simulation models, where dynamics are learned from collected data traces of the phenomena. The second section of this chapter (Sec.2.2), looks into alignment of the specific parameters of the simulation θ with respect to real-world observations x_t. The section starts with some of the challenges of this alignment process, and then proceeds into discussion about potential solutions such as simulation

based inference and system identification. In the final section of this chapter (Sec.2.3), different methods for planning and control u_t are covered. This section starts with discussion over traditional approaches, and finishes with topics of particular interest to this thesis such as active acquisition of data and model-based apporaches.

This chapter aims to give a broad overview, while later chapters include an in-depth summary of specific advances within the targeted application areas.

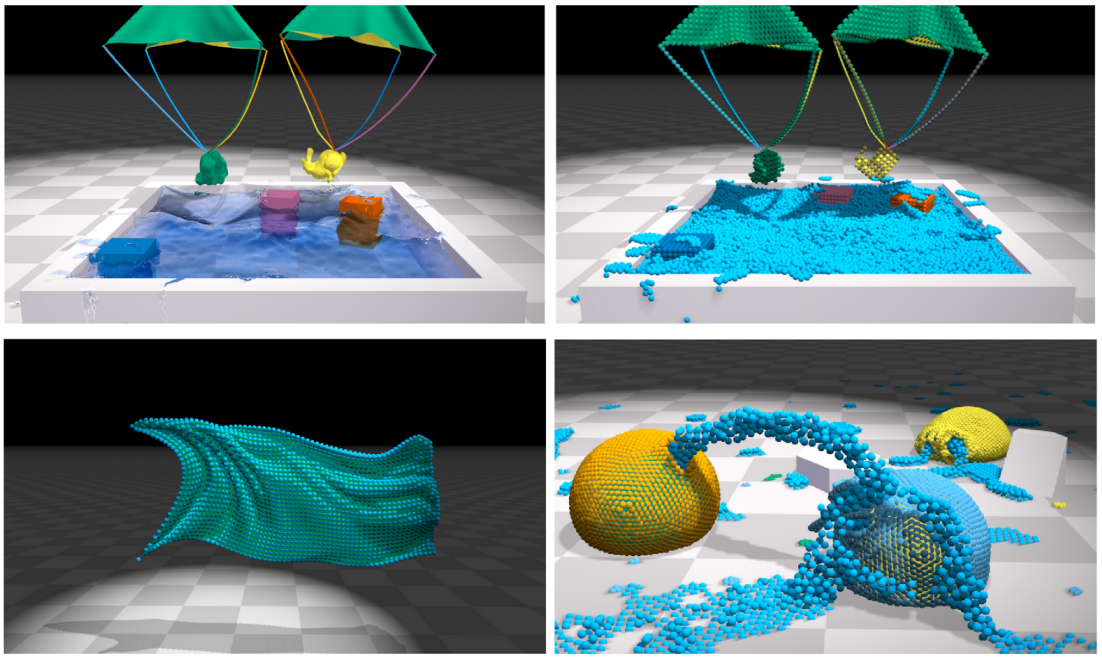
2.1 Simulation Models

This section starts by briefly covering simulation models used for describing robotics dynamics. In particular interest to this thesis, the section continues with overview of models used for object and fluid dynamics. This includes both traditional methods developed by the computer graphics community, but also novel ways of learning simulation models from raw data. Importantly, the section aims to analyse tradeoffs between different approaches in the context of coming up with suitable representations that can subsequently be used for simulation alignment and control.

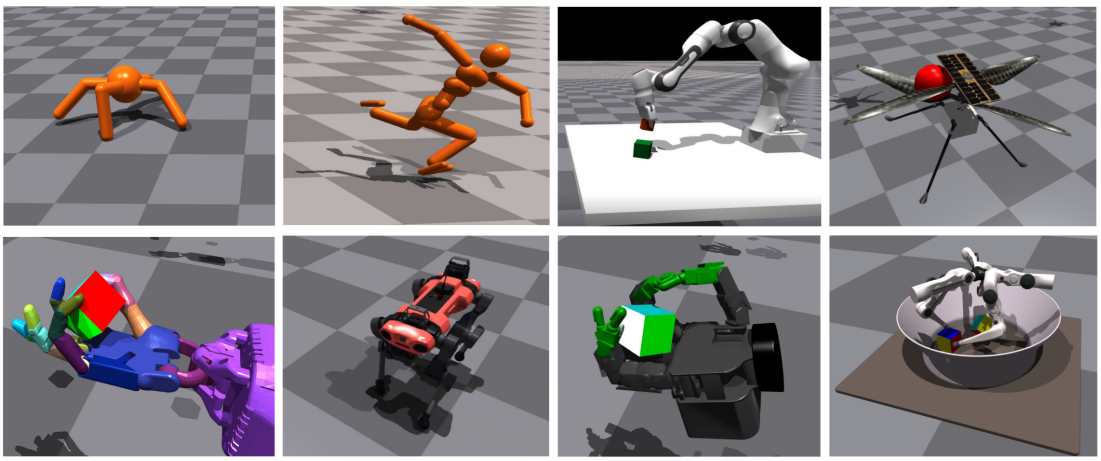
2.1.1 Robot Dynamics

Simulators have long been used within the robotics field to safely test the effect of potential actions before running them on a real robot. Before even the simplest robot movement, a reference trajectory must be computed to be passed down on the respective controller for execution [63]. This necessity for testing in simulation and the wide range of different robotic systems have led to the development of standardized communication protocols between different hardware components [181], motion planning frameworks [209], robot simulators [117], including handling contacts [219].

Beyond testing and verification of the validity of specific actions, simulators can also enable new capabilities such as real-time position based controllers, by implementing a fast enough inverse kinematics solvers [57]. Finally, they can additionally allow for learning in a safe, scalable way, while accounting for mismatch with the real system with techniques such as domain randomization [171]. While initial simulators were primarily focused on the kinematics of the respective robotics platform, as new emerging deployment scenarios are becoming of interest, new simulators are being proposed integrating not only object dynamics, but also realistic renderings [47].



(a) Particle-based simulation



(b) Simulating different robotic systems and interactions

Figure 2.1: **Simulating robot and object dynamics.** Simulators can provide an efficient way of testing actions, gathering data, and learning policies. [141] [143]

2.1.2 Object and Fluid Dynamics

Simulating the robot's dynamics or only part of the problem - if the has to interact with the world, we have to model that. Many formulations have been proposed to address different aspects of the dynamics of rigid objects, soft bodies, and fluids of our surrounding world, as seen in fig.2.1. Rigid bodies simulation deals with the particular case where objects are assumed to experience zero deformation under applied forces. The movement of a particular object, when forces are applied, needs to be estimated,

together with the more computationally expensive process of estimating contacts to avoid inter-penetration [19]. Traditional force-based methods enforce constraints where constraints and external forces are integrated between objects to avoid penetration [20]. However, this becomes challenging in situations with ever-changing constraints between different objects, such as highly dynamic environments with many interactions. Later, a more explicit model of impulse-based dynamics was proposed, where no constraints are imposed, but all internal and external forces are integrated to model contacts, and penetration [149]. While this can be less physically correct, it overcomes the challenges of working on acceleration level when estimating multi-body interactions. To further address some of the stability issues of those approaches, position-based dynamics were proposed, where updates are performed in the position space while estimating different constraints and updating the positions based on them [155].

Fluids are usually represented by partial differential equations in the form of the Navier-Stokes equations. The different components of the equations can be solved with Eulerian methods, where the simulation of fluids is described with several spatially fixed points, or Lagrangian, where the movement of its particles describes the fluids. By modifying the Eulerian framework with additional parameters, it is possible to simulate different phenomena [161] such as gases [204], melting objects [39], objects made out of sand [240], etc. Simplified models can be used in some instances when the exact dynamics of liquid are not critical, but it matters how the liquid moves in between connected components. In analyzing flood risk over terrains, a flow graph describing the connected terrains and their heights, together with a rain distribution function, can be sufficient [137].

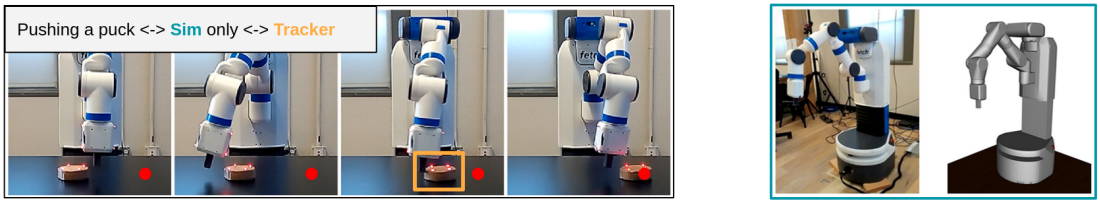
Real-world scenes are a combination of objects with different physical properties and behaviors. Position-based dynamics have allowed approximate but computationally stable simulation of rigid and soft objects, together with fluids under a common framework . This has led to impressive unified simulation capabilities of already mentioned phenomena, of melting and sand objects, gases, rigid objects, clothes, tearing, and many others. Modern implementations are computationally efficient to run in real-time [141], even on modest hardware. However, their approximate nature needs to be addressed if they are to be used as models for control.

2.1.3 Learnable Physics Engines

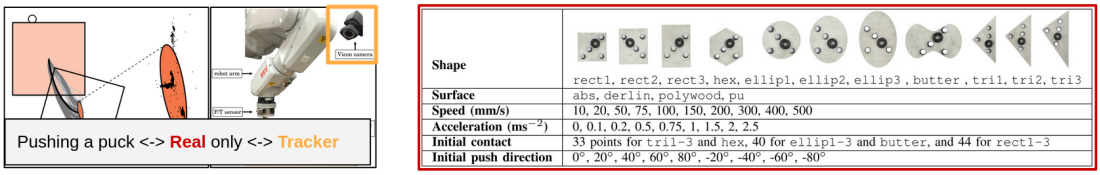
Traditional simulators have predefined procedures of how the state space evolves, defined by the specific implementation. This is often limiting, as seldom can any simulation capture the full complexity of the surrounding world. There is often a tradeoff between simulation fidelity and speed of execution, with accurate simulation taking a long time to compute. Finally, as part of modern end-to-end learning architectures, it is desirable that physics engines are fully differentiable. A recent line of work looks at whether a simulator can be learned from raw sensory data. Examples of this include learning general-purpose modular object-centric [22, 40] and particle-centric [154] physics engines. Those methods have the benefits of being general learnable models, in principle able to learn any interaction. The main challenge in developing learnable simulators is getting enough training data in terms of quantity (enough training examples) and the quality and granularity (segmenting different objects from a scene and recording their trajectories). Usually, this line of work is benchmarked by applying it to the existing physics simulator as we can gain access to ground truth and evaluate the performance.

Using similar architectures to learn a physics simulator from scratch is possible, but this is often impractical due to the amount of data needed, while knowledge of physics laws and dynamics is ignored. As such, it might be helpful to learn the *residual* part of the dynamics on top of an analytical solution. In the presence of high accuracy tracker systems, the residual learning can proceed directly from trajectory segments such as [6] by integrating [22]. Self-resetting robotics environments are often one of the requirements, even when relatively limited data is needed. By carefully integrating physics engine, robotics data gathering, and NN perception modules can lead to impressive performance in challenging tasks such as precise throwing of randomly shaped objects, while accounting for aerodynamics [232] as seen in fig.2.2. It is also essential to verify proposed approaches on a variety of error metrics, such as by integrating 'Violation of Expectations' and different concepts such as object persistence [175].

In addition to learning general-purpose simulators, a more constrained learning problem is one of bucketing of real-world pictures into a few physics scenarios [153], learning the dynamics of billiards [71] and parameterized fluid simulation [114, 197]. However, the problems mentioned above still hold even within those settings. A new line of work aims to explore models capable of learning dynamics [237] as well as



(a) Policy learnt in simulation via domain randomization using object tracker



(b) Using extensive real world data collection with self-resetting environments

(c) Combining simulations and real data

Figure 2.2: **Learning policies from real and simulated data.** Policies can be learnt from simulated, real data or both; by using precise trackers or directly from vision data.[171] [230] [7] [232]

to infer the structure of the world [67] from vision data in an end-to-end fashion. A tangentially related question is learning from micro-data, where one approach can be learning in simulation and fine-tuning on a few real observations. Meta-learning is one approach for achieving that [69, 68, 90], by optimizing over a family of tasks for faster convergence of the learning algorithm as new tasks are introduced. Using vision for data gathering for the methods as mentioned earlier is challenging, partly solved by introducing skip connections to help with object persistence as the robotics arm is moving around the scene [62] and using autoencoders for generating more real-world like training data [158].

2.2 Simulation Alignment

This section covers related work to the central question of this thesis, aligning the parameters of a simulation model to the acquired sensory readings. The section begins with a list of sources of error during simulation alignment and potential approaches

for addressing those errors. Major part of this section is dedicated to simulation-based inference approaches, which are subsequently adapted within the work of this thesis. Finally, the closely related fields of system indentification and uncertainty quantification are discussed.

2.2.1 Challenges

Once we have a physics simulator (traditional or learnable one), a complimentary question is how we can use it to perform inference and action selection. There are often many challenges as one has to account as seen in fig.2.3 for all the issues associated with the alignment between the real world and the simulated environment [113]:

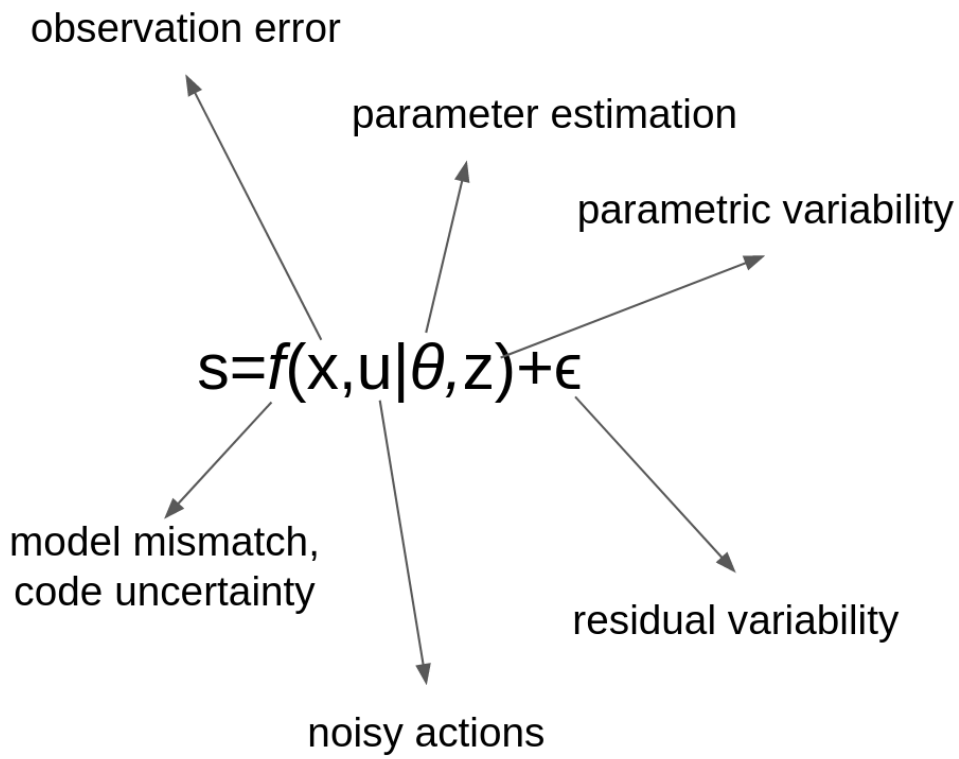


Figure 2.3: **Challenges in simulation alignment.** This thesis focuses on parameter estimation θ, and studies methods for efficient alignment when dealing with computationally expensive f or rich sensory observations x.

- Parameter estimation and uncertainty - select the parameters for the simulator model that generalizes and are accurate with respect to the scene in real world

- Model mismatch - account for the mismatch between the simulator and the real world, once the simulator has been calibrated
- Residual variability - uncertainty arising from stochasticity of the simulation or unobserved hidden parameters
- Parametric variability - uncertainty from unspecified/uncontrolled known parameters
- Observation error - inaccuracies in the observed sensory data
- Code uncertainty - even though the implementation of the simulation will generally, be known, there are still uncertainty or inaccuracies associated with it (similar to 'black box' argument about NN, where we know all the weights, but cannot provide reasoning about its output)

Some of the traditional approaches people have used before to address some of those issues include [113]:

- Interpolation - inferring the values of unobserved parameters combinations, based on already observed ones
- Uncertainty analysis - study the distribution of the output of the simulation, given the distributions of the individual parameters
- Sensitivity analysis - study the induced error of the output of the simulation, as a certain parameter change
- Calibration - estimation of parameters, via ad hoc search or Bayesian Optimization (BO)

Traditional approaches can give some intuition about the performance of the simulator, but it is clear that this is not enough to address all the challenges associated with using a simulator as/to learn a model.

2.2.2 People Reason using Noisy Simulator

People can learn and reason about the complex dynamics associated with the everyday world. We can predict movements (e.g., while playing tennis) and estimate physical properties (e.g., applying the necessary force while lifting an object). People learn

this while growing up by exploring and interacting with the world. However, it is unclear how this learning process happens and how we perform inference given our observations. One observation is the high correlation between our predictions and a noisy Newtonian probabilistic simulation [23] in different tasks - predicting a movement of billiards balls [79], estimating whether a tower of cubes is going to fall [23] and reasoning about liquids [21]. Interestingly, our predictions are rarely physically precise, suggesting that we use approximate models to make predictions of our surrounding dynamics, continuously updated as we observe.

2.2.3 Simulation-based Inference

Using a simulator as a model and optimizing its hyperparameters is one of the common approaches to incorporate physics knowledge about the world, especially when experimental data is not or hardly available. Simulation-based inference is an established field in natural sciences, where complex simulations have been developed to address different phenomena [49] as in fig.2.4. The inference process is one of refining a prior distribution of some parameters of interest as some observations are acquired. If we have observations x and parameters of interest θ, then we are interested in computing the posterior $p(\theta \mid x)$.

$$p(\theta \mid x) = \frac{p(x \mid \theta)p(\theta)}{p(x)} = \frac{p(x \mid \theta)p(\theta)}{\int p(x \mid \theta)p(\theta)\mathrm{d}\theta} \tag{2.2}$$

The challenge is that the data evidence $p(x)$ is often intractable - not available in closed form, or taking exponential time to compute [29].

Traditional methods are centered around sampling procedures, where the posterior is refined as more samples from the simulator are drawn. In the simplest form, this can be implemented as a rejection Approximate Bayesian Computation (rejection-ABC). Parameter θ is sampled (e.g., uniformly), forming simulated observation x_{sim}. If a simulated data x_{sim} is within the error margin ε to the observed data x_{obs} according to a distance metric ρ, the parameter θ is retained for the posterior distribution $p(\theta|x)$.

$$\rho(x_{sim}, x_{obs}) \leq \varepsilon \tag{2.3}$$

This method usually suffers from a very low acceptance ratio, especially in high dimensional data or small values of ρ. At the same time, if a sample is accepted, this does not influence subsequent drawn samples - even though the accepted sample signals a potential high-density probability area of the accepted parameter. Precisely this issue

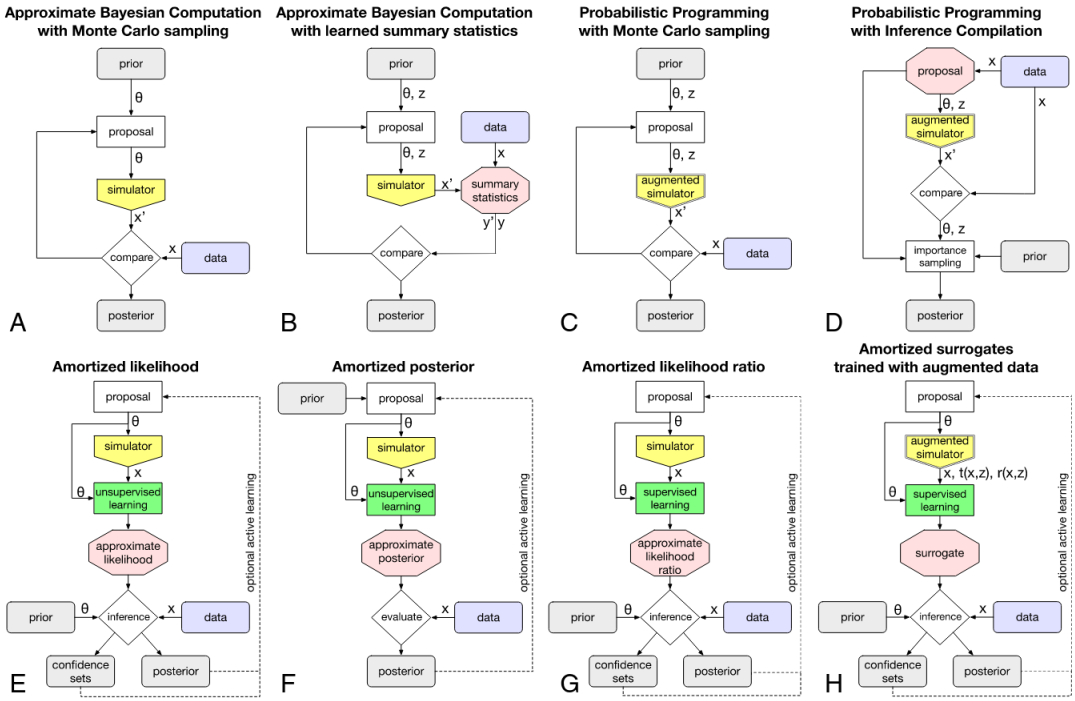


Figure 2.4: **Simulation-based inference methods.** SBI is a widely used technique in many fields, such as molecular dynamics, protein folding, etc. Two broad methods exist - techniques for sampling the simulator directly (top row) or amortizing the inference process (bottom row). This thesis explores both of those methodologies - sampling reduced simulation representations and learning direct inverse mappings from observations to parameters of interest - and their applicability in different robotics settings. Figure taken from [49].

is addressed by different Markov chain Monte Carlo (MCMC) methods by constructing a Markov Chain that has the same equilibrium distribution (after discarding the initial burn-in samples) as the desired one [145]. MCMC is well suited for cases of static, computationally expensive datasets to acquire, where the computational cost of MCMC itself may not be the primary concern.

Alternatively, a family of approximate densities $\mathcal{Q}$, e.g., Gaussian, can be introduced, where the parameters of the approximate distribution are *optimized* to match the posterior [29], using methods known as variational inference.

$$q^*(\theta) = \arg\min_{q(\theta)\in\mathcal{Q}} \mathrm{KL}(q(\theta)\|p(\theta \mid x)) \tag{2.4}$$

Replacing the *sampling* process of one of *optimization* can be beneficial, especially for larger datasets, where respective optimization methods can be leveraged. In their

standard form, both MCMC are VI are not sequential algorithms, meaning that as new data comes on the inference process must be recomputed.

Alternatively, a surrogate model can be learned, which is then alternatively used for inference, *amortizing* the process.

$$q^*(\theta \mid x) = \underset{q(\theta|x)\in\mathcal{Q}}{\arg\min} \mathrm{KL}(q(\theta \mid x) \| p(\theta \mid x)) \tag{2.5}$$

Different choices of models of q are possible, but most notably neural networks have been recently used [115], as a way to perform inference from high dimensional and complex data.

SBI is widely used across different application areas, such as analyzing TNT explosions [87], where a detailed simulator is used to calibrate to data from real explosions, Jet-in-crossflow simulations [184] or efficiently monitoring of environmental conditions in data centers [105]. From a robotics perspective, this can be applied to different manipulation tasks and inferring shapes and properties of objects [238] or jointly optimizing camera intrinsics and physical parameters from videos [100]. Those developments have shown that approximate simulators can enable the synthesis of complex behaviors such as pouring, that would have been hard to achieve through conventional means [135]. This draws on earlier observations from human psychophysics, e.g., [21], that people seem to be able to reason about the flow of liquids in situations where the available data is necessarily sparse. Of course, aligning a simulator is also a central method in model-based control, including learning a robot controller [118]. The authors show that it is possible to jointly optimize over system identification (finding a mathematical model of a dynamic system) and parameters estimation (optimizing the parameters of the found model).

The parallel development of deep NN and their successful vision application have led to the natural unification of the two approaches [229, 228]. Most of the practical NN based real-world applications have so far been based on supervised learning by using a collected dataset. Apart from the computation power and model architectures, a big challenge has been collecting labeled data. In [229], the authors argue that in addition to matching a real-world scene to a simulator (sliding object over an inclined surface), this could lead to estimating different physical properties of the objects, such as friction, mass, etc. This in turn could be used for training data for learning vision models being able to predict physics properties of objects, just based on how they look - arguably an intuition we develop as people as we grow up. Following this line of work, it has also

been shown that simulators can be used as part of a learnable vision model (in limited tabletop scenarios) [228]. A perception module extracts information about the objects from images, fed into physics and then graphics engine. Finally, the likelihood between sim and real is calculated in the image space rather than using extracted trajectories. Alternatively, it is also possible to estimate physical parameters of interest from videos unsupervised by introducing the physical equations themselves as a bottleneck in a fully differentiable autoencoder model [101].

2.2.4 System Identification

System identification is another popular field closely related to the question of simulation alignment. The field has seen two significant avenues of exploration [134], building linear state-space models from impulse signals [88] and later on focusing on prediction error methods [16], which are closely related to maximum likelihood methods and statistical inference [14]. While an established field, both avenues have seen continued interest in identifying the governing equations of a nonlinear system [35, 30] and parameter inference from a given model [49]. The core problems and avenues outlined in [15] remain relevant today - the class of systems, the input signals, and the criterion - and are a research topic within this thesis.

2.2.5 Uncertainty Quantification

UQ is the process of quantitative characterization of the degree of precision with which a quantity is measured [223]. The goal of UQ is to produce a measure of confidence for model predictions - a capability needed in critical scenarios such as autonomous vehicles and the medical domain, where actions can have serious consequences. Even beyond, whenever a robotic system is in the loop of people, robust UQ is crucial. Uncertainty is often classified into aleatoric, or uncertainty associated with the data itself; and epistemic, uncertainty associated with the model [111]. Aleatoric uncertainty is irreducible, as it is associated with the variations of multiple runs of the same experiment, where epistemic uncertainty can be improved with better modelling and/or data. Models such as Gaussian Processes (GP) are known for their principled treatment of uncertainty, as they can incorporate well-defined bias via their kernel functions, provide uncertainty estimate as part of their prediction, and can model both aleatoric and epistemic uncertainty (e.g., by horizontal/vertical lengthscale parameters for epistemic, and noise parameter for aleatoric in the squared exponential kernel). Alternatively, Bayesian

NN can also provide uncertainty estimates with many implementations proposed [3] - with dropout and model ensembles being the predominant techniques for obtaining UQ from NN [61].

2.3 Planning and Control

This section covers methods for synthesis of control strategies of robotic systems. Motivated by the scope of the thesis, model-based approaches and active acquisition of data are discussed. As an alternative to using simulation as a model, the section finishes with a short overview of model-free approaches and automatic data collections with a robotic systems.

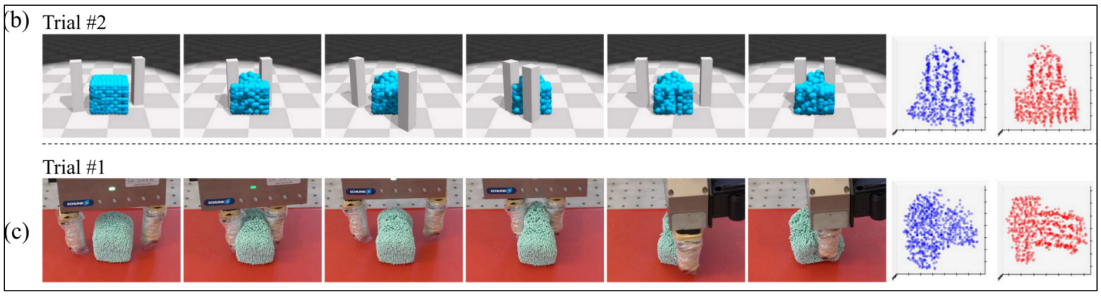
2.3.1 Manipulation Techniques

Within robotics, the taxonomy of manipulation techniques consists of Newtonian robots, Aristotelian robots, and Empirical robots [146]. Newtonian robots model actions using Newton's laws, Aristotelian by assuming things move only when pushed, and Empirical using collected observations. Based on the complications in classical mechanics (kinematics $\rightarrow$ statics $\rightarrow$ dynamics), we can define a hierarchy of manipulation techniques. Kinetic manipulation deals with action/s derived from kinematics, e.g., moving end-effector to a specific place. Static/Quasistatic manipulation also considers the placement of objects and any physics properties (excluding inertial or any other dynamics), e.g., putting an object on a table or straightening a deck of cards by squeezing it. Finally, dynamic manipulation considers the whole physics scene, including any dynamics associated with it, e.g., sliding an object over an inclined surface or pouring liquids.

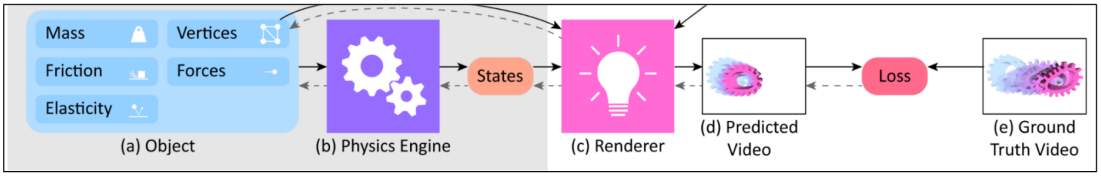
2.3.2 Motion Planning

Even if we assume movements and sensory data with no noise, planning motion for a defined task is still challenging. This has primarily been defined using two methods, often combining them [41]. One way is to define prior knowledge over the policy structure (e.g., movement primitives), policy parameters (e.g., learned from demonstrations), and dynamics (e.g., learned from a simulator). This allows for reducing the complexity of the search problem. An alternative option is to create a surrogate model of the reward (e.g., BO) or dynamics (e.g., model-based policy search), and query the model instead as in fig.2.6. End-to-end differentiable simulators (e.g. learned as a

NN module or constructed explicitly) can alternatively be used to synthesize control strategies efficiently as in fig.2.5 .



(a) Control from particle representations



(b) Control from images using fully differentiable engine and renderer

Figure 2.5: **Control using learnable physics engines.** Fully differentiable physics engines allow for efficient control strategies. [130] [157]

Aligning a simulator allows for a more efficient policy search by optimizing using the simulation rather than the real world. The main question becomes how one can guarantee that the simulator's policy will be successful once transferred to the real robot. One approach is to align a simulator and learn a policy by repeating a three steps process until the policy is learned [239]. First, the robot executes the current policy, and the data of the execution is recorded. Second, this data is used to run a simulation and update its hyperparameters. Finally, the updated simulator is used to learn a new policy. The process is repeated until convergence. Alternatively, the whole policy can be learned in a simulator and transferred to the real robot. One approach is to gather extensive simulated data. e.g., by using domain randomization [218, 176]. Alternatively, the complete model of the environment and the robot can be specified, e.g., learning to push a puck [170]. The development of this line of work has enabled making progress in challenging robotics tasks such as manipulating fluids [135], cloths [147], deformable objects [147].

An line of work aims so solve the problem of using RL with simulations and BO for more efficient training [144]. [220] defines a framework based on logic and geometric constraints able to constrain the world to allow for more efficient planning of multistep tasks. Extending this line of work, in [221] they demonstrated in simulation planning

of complex robotics reasoning tasks (e.g., use a stick, to get a long stick, to pull a ball that you want to grab).

A simulation models is almost never a perfect model of the phenomena of interest - as such there is a Sim2Real gap between the expressiveness of the model and the real environment. As such even if the optimization process of aligning the simulation converges, the mismatch may still be too big for a downstream task. Apart from carefully considering the model selection process, residual learning has been a proposed solution to this problem, by learning an additive model of the remaining dynamics, uncaptured by the analytical simulation model [232].

2.3.3 Classical Control

Two broad methods exist for control - open-loop controllers and closed-loop controllers. Open-loop control acts regardless of the output of the controlled system, while in closed-loop control, the control actions are adapted to a given objective function, conditioned on an observation. Probably the most widely used controllers are the Proportional-Integral-Derivative (PID) controllers, minimizing the three respective error signal terms between the observed and reference signal. Although this type of control does not guarantee optimality and works on a single input-output, it is widely applicable as it does not require knowledge of the underlying controlled system. If the observed system is known to be linear with Gaussian noise, then an alternative formulation as linear-quadratic-Gaussian (LQG) problem is often used. Under the separation principle, this problem has an optimal solution of system identification problem using a Kalman filter, from the control problem, by using Linear-Quadratic-Regulator (LQR). In the more general case of nonlinear system, the generalized method of Model predictive control (MPC), where the solution is computed over some fixed time window. The first action from the compute action sequence is executed, after which the process is repeated.

2.3.4 Active and Adaptive Acquisition of Data

Active acquisition of data is often defined as a trade-off between exploration and exploitation by formulating the problem as one of regression from sparse measurements. Representative examples of this approach include [203], which uses Gaussian process mixtures, and [132]. Kalman filter based estimation algorithms also work similarly [28]. To incorporate physics knowledge within this framework, the kernel of the Gaussian

Processes (or similar method) is refined to account for different effects indirectly, e.g., incorporating wind information [185, 162].

Another aspect of active sensing is the method for collecting samples to maximise a notion of information gain. While the underlying exploration-exploitation trade-offs can be posed formally in decision-theoretic terms, most practical techniques tend to be myopic in their operation. GP [124, 167] and the Kernel DM+V/W algorithm [162] address this question. One could also formulate this as the optimal design of sequential experiments [93]. However, this requires access to analytically defined dynamics models, which may be hard to construct for the specific scenario at hand.

GP are praised as a model able to define uncertainty (by taking multiple samples of the constructed GP), bias (by specifying the kernel function) and learning (by optimizing for the hyperparameters of the kernel). However, a common criticism is the computational complexity of the model as a function of the data. This is being addressed by optimising the inference procedure [86]. Alternatively, work is focused on combining NN with GP [77, 78, 127] and particle filters [103] to allow for more expressive and scalable models. It would be interesting to see whether those recent algorithmic developments could be helpful in designing a better exploration framework for manipulation tasks, e.g., performing actions to better gain information about the shape of an unknown object.

2.3.5 Model-free, End-to-end Approaches

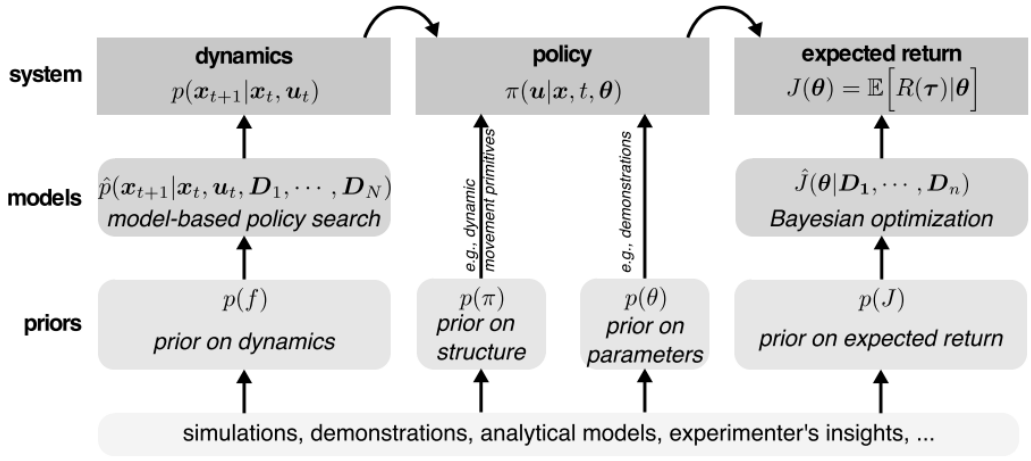


Figure 2.6: **Learning from micro-data.** Addressing learning from a handful of trials by modelling of the dynamics, policy, and cost function. Figure taken from [41]

A different approach to using a simulation in the optimization/mapping is to have a model-free approach and learn from data [5]. In this work, the forward and inverse kinematics are parameterized by a NN. The model is learned directly from images for the manipulation task of displacing objects. This was achieved by performing a total of 400 hours of training on a real system. The advantage of such methods is that the model is trained end-to-end by just using raw data. However, it is not clear how such work can scale and generalize to unseen objects and configurations. Work in this area benefits from careful experimental setup, as even challenging tasks such as playing billiards can be solved by tracking and simple model matching [76].

Deep learning nevertheless could provide benefits, both in terms of the complexity of the learned manipulation models, but also the arguably simpler model, requiring just a lot of data. As such, an interesting question is how can data collection using real robots be better automated. An example of such an approach is demonstrated [230], where a dataset of rigid bodies with different shapes pushed in different ways on top of different surfaces is presented. The pushing objects have marked holes easily identifiable by a vision system, which allows the robot arm to move the objects in their initial position, allowing for an unsupervised way of resetting the scene and efficient data collection. In a follow-up paper, it is demonstrated that such a dataset can be used to learn a probabilistic model for pushing objects [24].

2.4 Online Simulation-based Inference for Motion Planning

Robots need models of their environments, so they can plan and interact with them. The identification and subsequent planning must almost always proceed online as to be able to deal with the different dynamic situations. The online nature of the problem is precisely what makes it challenging.

Within this chapter, different simulations used for modeling different dynamics of the environment (Sec.2.1) were discussed. Subsequently, this thesis explores varying complexity of the simulation models - used from rigid bodies (Chapter 5) to fluids (Chapter 3) and molecular dynamics (Chapter 5). The core problem motivating the work in this thesis is simulation alignment, discussed in details in Sec.2.2. Simulation-based inference is the main technique used within this work, adapted to the specific needs and challenges of the robotics domain - sampling reduced simulation representations

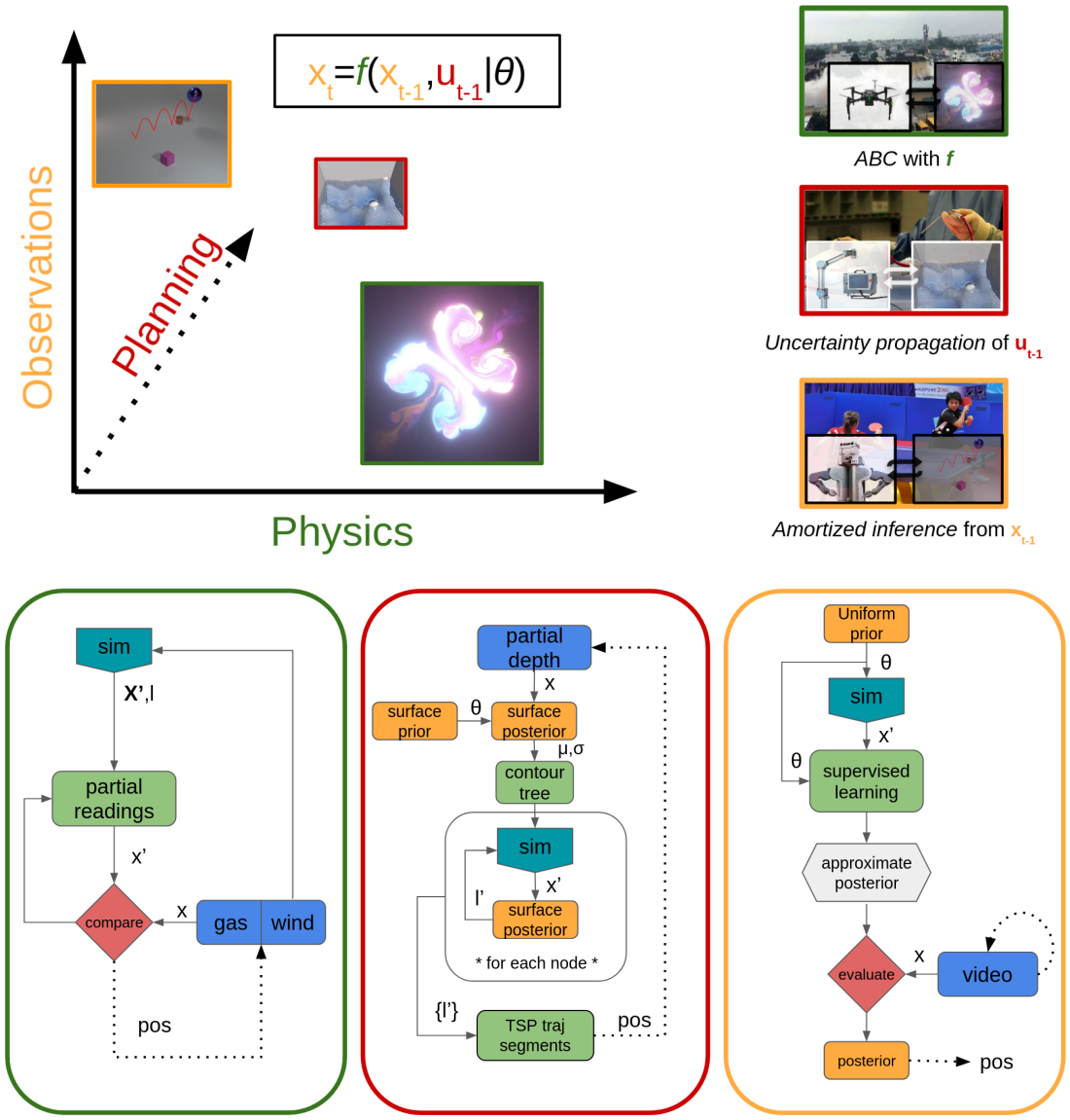


Figure 2.7: **Simulations based on their complexities.** Based on the complexities defined in fig.1.5, we define three different projects within the thesis, addressing different complexities. Note that there is often some correlation between the three axis of complexities - they are neither fully ortogonal or parallel. Within this thesis we focus on three different projects, focusing in turn of challenges associated with each complexity.

(Chapter 3 and 4) and learning inverse relations (Chapter 5). The reason for performing simulation alignment, is for the subsequent task of control for a given goal. Control strategies and the use of simulators within the control loop were covered in Sec. 2.3. This thesis adapts both myopic strategies (Chapter 3), but also long-term planning (Chapter 5), while also incorporating uncertainty (Chapter 4).

The contribution of the thesis can be summarized as methods for online simulation-

based inference - sampling reduced simulation representations or directly learning an inverse mapping between observations to parameters of interest. This thesis explores three projects, motivated by different challenges in the observations, physics and planning as seen in fig.2.7. Each of those complexities is studied in the following order: Chapter 3 studies simulation alignment with a computationally expensive fluid simulation for the task of gas localization (green in fig.2.7); Chapter 4 studies safe planning under uncertainty for the task of medical suction (red in fig.2.7); Chapter 5 studies simulation alignment from complex observations such as videos or x-ray scattering signal (orange in fig.2.7).

Chapter 3

Active Localization of Gas Leaks using Fluid Simulation

A key constraint in performing simulation-based inference is the computational cost of the simulation model itself. Often the inference proceeds by repeatedly sampling the model to identify the best parameter from a given prior. That cost can be reduced be lowering the number of simulation calls and/or the cost of the simulation run itself - for example by exploiting domain specific assumptions. This chapter looks at one approach for simulation-based inference, where the process can be reduced to just one simulation call.

In addition to simulated data, real observations are required for the inference process. Various sensors can be mounted on robotics systems to acquire measurements in spatio-temporal fields. Locating features within these fields and reconstruction (mapping) of the dense fields can be challenging in resource-constrained situations. In such cases, a model of the underlying complex dynamics can be exploited to discover informative paths within the field.

Within this chapter we look into the example problem of localization of the source of a gas leak from a small number of measurements. We use a fluid simulator as a model, to guide inference for the location of a gas leak. We perform localization via minimization of the discrepancy between observed measurements and gas concentrations predicted by the simulator. Our method is able to account for dynamically varying parameters of wind flow (e.g., direction and strength), and its effects on the observed distribution of gas. We develop algorithms for off-line inference as well as for on-line path discovery via active sensing. We demonstrate the efficiency, accuracy and versatility of our algorithm using experiments with a physical robot conducted in outdoor environments.

We deploy an unmanned air vehicle (UAV) mounted with a CO_2 sensor to automatically seek out a gas cylinder emitting CO_2 via a nozzle. We evaluate the accuracy of our algorithm by measuring the error in the inferred location of the nozzle, based on which we show that our proposed approach is competitive with respect to state of the art baselines.

Contributions In this chapter, we:

1. formulate gas leak localization as model-based inference using a fluid simulator as the model,
2. develop a practical optimization algorithm based on a single simulation per iteration,
3. develop an online algorithm that locates gas leaks using active sensing,
4. demonstrate that our algorithm results in acceptable localization error in real experiments.

3.1 Introduction

We address the problem of using a robot-mounted sensor to actively search for features of a spatially extended field, e.g., source of a leaking gas, or to map such a field from point measurements. This is useful in numerous applications, such as disaster response [156], scientific data collection in difficult and inaccessible environments [174] and urban emissions mapping [82].

From a robotics perspective, the core problem is that of synthesizing paths with respect to an objective such as quality of reconstruction of the underlying spatial field, or the accuracy of localization of a spatio-temporal event (e.g., source of a gas leak). Traditional approaches for solving such problems include 3D surface reconstruction algorithms [56] and regression models such as Gaussian process [104].

What makes the practical problem challenging, and many of the traditional methods harder to apply, are resource constraints and lack of control over the experimental domain. When the sensor is mounted on an Unmanned Aerial Vehicle (UAV) and flown around from a more distant launch location, the number of samples that may be collected is limited (typically due to power constraints, but also due to other issues such as contamination risks). Moreover, the true underlying phenomena are typically varying, such as when the field is created by dispersion under wind flows and one must reason

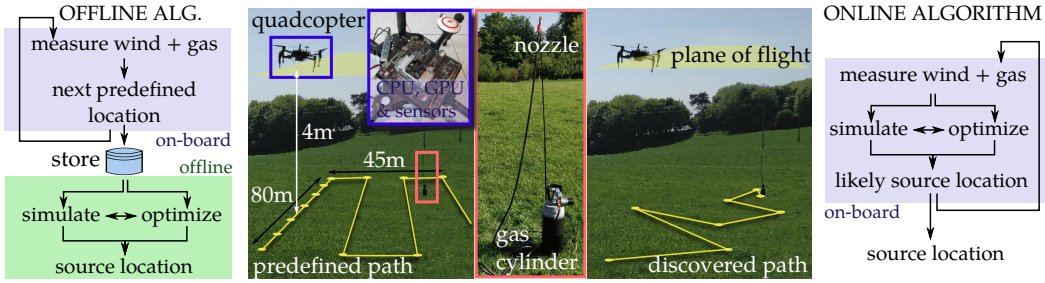


Figure 3.1: **Gas-leak localization** We place a gas cylinder releasing CO_2, via a nozzle, and measure gas concentrations, wind speeds and wind directions using a UAV. Our goal is to estimate the location of the nozzle from these point measurements. The UAV, which is equipped with sensors and on-board processors (CPU and GPU), flies on a fixed 2D plane 4m above the ground. We develop two algorithms for localization. Our *offline algorithm* (left) compares measurements, taken along a predefined path, with a fluid simulation (that considers measured wind). Our *online algorithm* (right) discovers a path to the source of the leak by repeatedly performing simulation and comparison steps on-board the UAV, and flying to the most likely location of the source given the measurements.

about this (e.g., to locate the true source location despite wind). Lastly, the overall task is made difficult by the inability to obtain detailed ground truth to evaluate models - if the objective were to be defined as reconstruction of a spatial or temporal field, then we may not have a detailed description of the true underlying field. Some recent works that are noteworthy in this area, and shown to be successful in experimental settings include, e.g., [203] which presents a method based on Gaussian process regression, and [162] which proposes another kernel based approximation model. The latter also accounts for the directionality of wind flows, by interpolating from past data with a modified kernel. In most such work, the treatment of wind as a cause of variation, and associated search strategies for actively locating environmental features (e.g., pinpointing the source location of a leak before wind flow disperses it) is either heuristic or considers wind to be constant over time (in which case reasoning about the flow may not be necessary).

In this chapter, we propose the use of a fluid simulation as the model of the underlying phenomenon which allows us to directly address these causes of variation. We pose the problem of estimating the location of an environmental feature (e.g., source) as that of optimizing the fit between a fluid simulation model and the point-wise measurements of the resulting scalar field. This is a process of calibrating the simulation model, which we show can be done online as measurements arrive (indeed, in a way that can be

implemented on resource constrained hardware within a standard commercial grade UAV) and used to drive search processes based on information-gain criteria.

The optimization problem can be solved in a number of ways. For the problem of estimating a spatially extended field given a sequence of measurements, one approach is to use BO, treating the true objective function implied by the environment as a random function with a Gaussian process prior. We develop an alternative approach, which exploits the shift-invariance of the phenomenon, that performs a larger simulation once at the beginning and poses location estimation as the problem of determining the optimal translation of a smaller field of interest. The use of the simulation model also allows us to address the active search version of this problem, where the sampling locations are determined online (and on-board), by iteratively computing a likelihood for the source location and flying to the point which maximizes this quantity. This proposed approach, which we call One-shot Grid Search (OGS), is efficient and requires relatively low computational resources for similar localization accuracy.

3.2 Related work and contributions

Robots used as sensing platforms Early work around autonomous sensing of physical phenomena involved ground-based mobile robots [185, 132, 203]. More recently, with the emergence of reasonably robust Unmanned Aerial Vehicle (UAV) platforms, often referred to as drones, they are being used as sensing platforms with benefits in terms of speed, manoeuvrability and ability to deal with hostile terrains, unobstructed by objects on the ground [**?** 98]. UAVs bring their own challenges, such as reduced on-board power and the difficulty of finding sensors that fit within the form factor. Sensing technology has also continued to develop, e.g., making it possible to use spectrometers on UAVs [52]. We conduct experiments with a commercial off-the-shelf CO_2 sensor, but note that the computational methods presented here are sensor agnostic, assuming only that the sensor obtains point measurements from a scalar field.

Using simulations as models Models provide numerous advantages in machine learning [27], enabling inferences from limited data, and in planning [81], enabling counter-factual reasoning [31] and guided search. However, defining the structure of models in a way that leads to efficient inference while maintaining fidelity to complex arrangements of physical causes tends to be non-trivial.

The phenomena we consider in this chapter involve gas flows. There is a long tradition of modelling such flows, including efficient computational methods aimed

at graphics and animation applications [206]. The development of efficient solvers is also driven in the engineering community by the need to simulate phenomena such as fluid-structure interaction, yielding fast and approximate solvers through position-based dynamics methods [140, 141]. Simulation frameworks have also been developed aimed specifically at easing the development and testing of GDM and GSL algorithms [150].

Development of advanced simulation tools has led to new milestones in learning challenging robotics tasks. In [135], the authors show that approximate simulators can enable the synthesis of complex behaviours such as pouring, that would have been hard to achieve through conventional means. This draws on earlier observations from human psychophysics, e.g., [21], that people seem to be able to reason about the flow of liquids in situations where the available data is necessarily sparse. These papers are situated within the broader topic of 'intuitive physics', which refers to the ways in which cognitive models of real-world physical phenomena seem to only require relatively simple representations of the true underlying phenomenon [23, 40]. In restricted settings, such representations have also been used for efficient neural network based model learning [5, 71] and calibration [229]. Such ideas have been explored within the problem of odour localization, by devising naive fluid models and search algorithms [119]; pre-computing dispersion maps using computational fluid dynamics and probabilistically weighting them at test time [191]; updating Gaussian analytical model using evolutionary strategies [131]; using matrix of static sensors [150] but so far haven't been scaled to realistic outdoor environments, where usually limited samples are available.

In this chapter, we utilise a reasonably accurate simulation of the phenomenon [205] but exploit simplifications inherent to the problem, such as that the dispersion process can be modelled on the 2-d plane [1] along which the point measurements are also being taken. Moreover, the process of dispersion is shift invariant [50], so that a single large simulation can be performed online, from which the flow patterns for different locations can be easily computed.

Active and adaptive measurements Specific problems such as the localization of gas sources have been approached using a variety of different algorithmic means. Bio-inspired approaches have been proposed devising heuristics to follow the wind gradient towards the source [214, 98]. In practice, such heuristics depend on the presence of specific environmental conditions, including constant wind speeds across the field of

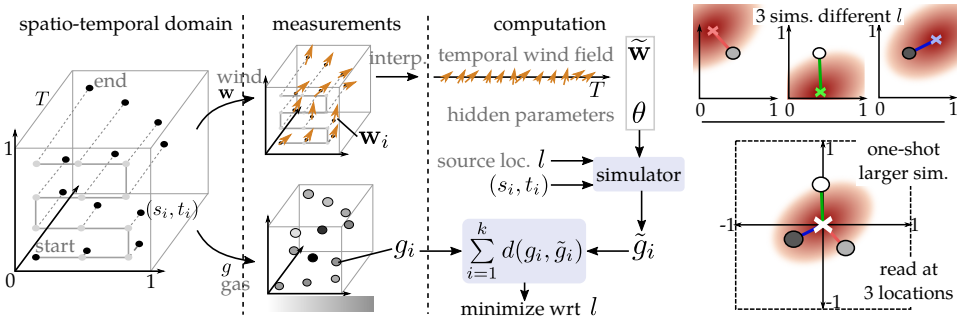
[1] We observe that our approach is invariant to some degree of (small) noise, i.e., the situation of plain fields and gently rolling hills. Many realistic applications are indeed sited in such terrain, e.g., a petroleum refinery in the periphery of which one might wish to perform emissions monitoring.

interest and the UAV being placed within the path of the gas. The source localization estimation has also been addressed using Bayesian methods, using particle filters in outdoor environment [164, 128]; Infotaxis which aim to maximize information gain by reducing the entropy [225]. Gaussian Markov random fields have also been used to address the problem of obstacles in indoor scenarios [151]. Another approach is to formulate the problem as one of regression from sparse measurements. Representative examples of this approach include [203], who use Gaussian process mixtures, and [132]. Kalman filter based estimation algorithms also work similarly [28]. A weakness of these methods has been that they do not explicitly consider the structure of the phenomenon in terms of a source and dispersion through wind flows, although refinements of the above procedures do indirectly account for these effects, e.g., [185, 162, 10].

Another aspect of active sensing is the method for collecting samples so as to maximise a notion of information gain. While the underlying exploration-exploitation tradeoffs can be posed formally in decision theoretic terms, most practical techniques tend to be myopic in their operation. Gaussian processes [124, 167] and the Kernel DM+V/W algorithm [162] address this question. One could also formulate this as optimal design of sequential experiments [93]. However, this requires access to analytically defined dynamics models which may be hard to construct for the specific scenario at hand. Notably, source term estimation was recently addressed with Bayesian estimation implemented using sequential Monte Carlo [96]. By using parameterized Gaussian plume dispersion model recursive Bayesian updates can be performed accounting for uncertainty in wind, dispersion, etc. This was additionally implemented on a UAV [97], performing outdoor localization of gas leaks using predefined flying pattern and ground station for performing computations. We formulate active sensing with a fluid simulation in the loop and devise an efficient algorithm for simulation alignment.

3.3 One-shot fluid simulation for localization of gas leaks

We localize the source of a leaking gas based on a discrete set of measurements of gas concentrations, wind speed and wind direction. We make the following assumptions: 1) there is a single source of gas within the domain of interest; 2) the ground plane is relatively flat; 3) gas and wind measurements are made on a plane parallel to the ground, and above the source of leakage; 4) wind flow is time-varying but spatially constant



(a) Summary of notation (b) Key idea

Figure 3.2: **Overview** We locate the source of the leak by comparing the output of our physically-based model, $\widetilde{g}_i$, with measurements at corresponding locations g_i. Rather than running multiple simulations with different values of l, we run one larger ($4\times$ domain) simulation and read multiple values from appropriate relative locations. We assume that $\widetilde{\mathbf{w}}$ is only a function of time (spatially invariant), and so simulation output is spatially shift-invariant. Although the simulation output is a 3D gas concentration field, we visualize the spatial distribution (blurry red splats) at a slice in time.

within the domain (i.e., obstacle-free domain, not large enough for local variation to be significant).

Problem formulation We define a spatio-temporal domain $S \times T$ over which fields of interest, such as gas concentration and wind flow, are defined. Without loss of generalization, we define $S \equiv [0,1] \times [0,1]$ and $T \equiv [0,1]$. We use a UAV to measure gas concentrations $g : S \times T \to \mathbb{R}$ and wind (speed and 2D direction) $\mathbf{w} : S \times T \to \mathbb{R}^2$ over this field. The sequence of measurement points are $s_i \in S$ and $t_i \in T$. We abbreviate measurements $g(s_i,t_i)$ and $\mathbf{w}(s_i,t_i)$ as g_i and $\mathbf{w}_i$ respectively (abbreviated in a shorter form in eq.1.1). These measurements are dependent on the location of the source of the gas $l \in S$ and hidden parameters θ, such as properties of the gas, which we assume remain fixed throughout an experiment.

We infer the most likely location of the source using a 2D Eulerian fluid simulator as our model. The simulator can be seen as a mapping from l to an approximate gas concentration field given the hidden parameters θ and a dense wind field. We rewrite this as a mapping from leak locations and spatio-temporal sites to gas concentrations $\widetilde{g} : S \times S \times T \to \mathbb{R}$, given θ and $\widetilde{\mathbf{w}}$. Here $\widetilde{\mathbf{w}} : T \to \mathbb{R}^2$ is the spatially constant but temporally dense wind field reconstructed from sparse spartio-temporal measurements $\mathbf{w}$ taken by the UAV. The gas concentration predicted by the model at s_i and t_i (where

measurements were taken) are abbreviated as $\widetilde{g}_i \equiv \widetilde{g}(l, s_i, t_i|\theta, \widetilde{\mathbf{w}})$ (note the slight change in notation with regards to eq.1.2, where $\widetilde{g}_i$ is marked as $\tilde{F}$). Our goal is to find a location l^* where measurements agree best with model predictions, i.e.,

$$l^* \equiv \underset{l \in S}{\operatorname{argmin}} \sum_{i=0}^{k} d(g_i, \widetilde{g}_i) = \underset{l \in S}{\operatorname{argmin}} \sum_{i=0}^{k} d\left(g(s_i, t_i),\ \widetilde{g}(l, s_i, t_i|\theta, \widetilde{\mathbf{w}})\right) \quad (3.1)$$

where $d(.,.)$ is an appropriate distance metric and k is the number of measurements taken by the UAV. fig. 3.2a summarises our notation.

One-shot Grid Search (OGS) The optimization in Eq. 3.1 can be performed naïvely by computing the objective function for many values of l, obtained by densely sampling S using a regular grid of $m \times n$ samples, and selecting the location l^* that yields the minimum value. This approach would be inefficient since it requires mn simulations to be performed. Our key observation is that *the model is shift invariant* if the parameters (θ and $\widetilde{\mathbf{w}}$) are spatially stationary. Concentrations $\widetilde{g}$ produced by mn simulations with different source locations are identical up to a translation:

$$\widetilde{g}(l + \Delta l, s_i, t_i|\theta, \widetilde{\mathbf{w}}) = \widetilde{g}(l, s_i + \Delta l, t_i|\theta, \widetilde{\mathbf{w}}) \quad \forall \quad (l + \Delta l) \in S,\ (s_i + \Delta l) \in S. \quad (3.2)$$

fig. 3.2b illustrates the central idea. In addition, if we use an Eulerian simulator, a single run of the simulator $\widetilde{g}$ with a source located at l_0^* can be evaluated at several $(s_i + \Delta l, t_i)$ cheaply. Rather than repeating the simulation with different values of l_k^*, we run the simulator once with $l_0^* \equiv (0,0)$ and read several values of the gas concentration field $M_{ij} \equiv \widetilde{g}(l_0^*, s_i + \Delta l_j, t_i, |\theta, \widetilde{\mathbf{w}}),\ j = 1, \cdots, mn$. We construct the entire $k \times mn$ matrix $\mathbf{M}$ using only one simulation. Each column of $\mathbf{M}$ contains $\widetilde{g}_i$ corresponding to a source location. We solve the optimization problem in Eq. 3.1 by identifying p as follows:

$$l^* = \Delta l_p \text{ where } \quad p = \underset{j \in \{1, \cdots, mn\}}{\operatorname{argmin}} \sum_{i=0}^{k} d\left(g_i, M_{ij}\right). \quad (3.3)$$

Since this OGS approach requires the shifted source (and samples) to be within the domain, we run the simulation on a larger domain $S \equiv [-1,1] \times [-1,1]$, always place the source at the origin and adjust the relative locations read appropriately (see fig. 3.2b). Thus, rather than running mn simulations, we run one simulation with four times as many Eulerian grid cells. We define q as the vector of distances of each column of $\mathbf{M}$ to g_i and use this to derive the likelihood for different source locations on a grid.

Wind estimation The above formulated approach relies on having access to wind estimates **w**. In order to acquire such measurements on a UAV, we use the pitch, yaw and rotation provided by the IMU which through a series of transformations can estimate the current wind [162, 163]. The rationale is that the wind speed and direction are directly related (through the transformation implied by a flight dynamics model) to the control signal that must be applied within the UAV, when hovering in place. Depending on the direction of the wind, the UAV will lean in a different way; the stronger the wind, the more it will lean. Using this approach the direction of the wind can be directly estimated, however the inclination angle of the UAV with respect to the ground has to be calibrated with respect to the strength of the wind speed. For calibration we use an off-the-shelf wind simulator provided with the commercial UAV we use. More precise ways of generating the reference wind fields, e.g., in a wind tunnel, could also be used [162]. Importantly, we show that our proposed method is robust to imprecise wind speed measurements.

Fluid simulator Another requirement for our method is having access to a simulator oracle $\widetilde{g}$. There are many ways to express a physical model of fluids, but we opt to use a stable Navier-Stokes solver due to its efficiency and ease of implementation [205]. The solver is realized by dividing the space into voxels and iteratively updating the velocity and density. The differential equation for solving for the density is linear with respect to the density term and thus easier to solve. For solving for the velocity a semi-Lagrangian technique is used, producing stable result like the density solver[2]. The simulator depends on different parameters - we assume that we have prior knowledge of the diffusion properties of the gas, *accurate* wind direction and *approximate* wind speed can be estimated as described above, and we normalize the gas readings to be invariant of the quantity of gas released. The rest of the parameters, number of cells (fidelity of the simulation), number of wind locations, simulation timestep, solver iterations, tend to be a trade-off between the accuracy and speed of the simulation.

3.4 Experiments

In this section, we aim to verify our main hypothesis - that using fluid simulation as a model during online gas localization task with a UAV leads to lower error compared to traditional approaches. For the purpose we devise a set of experiments, with the

[2]An online demo of the simulator and its behaviour can be found at https://gas-drone-simulation.neocities.org

main matric of localization error of the prediction versus the actual localization of the leakage. First, we perform a series of offline experiments, by collecting data with a UAV [3] flying at predefined waypoints. We compare our method to existing gas localization and mapping baselines found in literature. We then benchmark our optimization method against standard approaches for solving the proposed simulation alignment problem. Finally, we conduct sensitivity analysis of the different hyperparameters of the fluid simulation used.

Secondly, we carry out a set of online experiments, dynamically selecting new waypoints as part of the optimization procedure. We use readings from a noisy simulator as data to evaluate the performance of our method against other approaches. Finally, we perform active sensing experiments on a UAV using our algorithm.

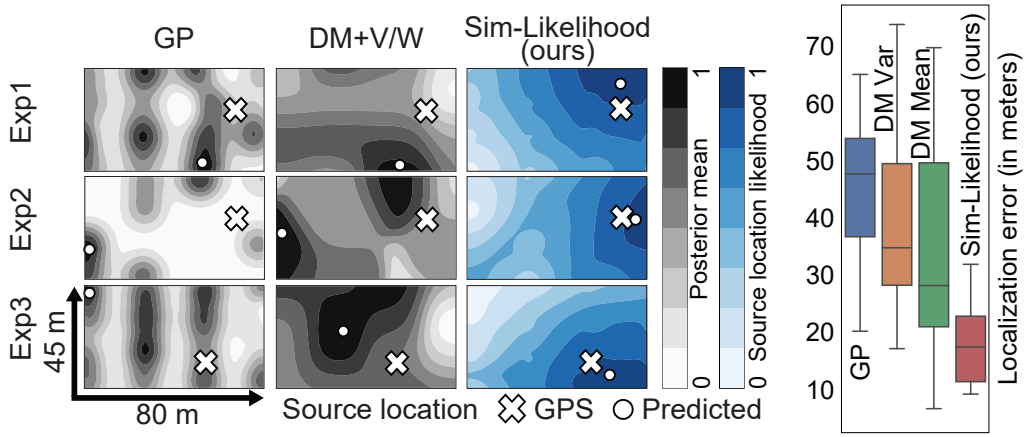
3.4.1 Offline algorithm

In order to evaluate the proposed approach we conducted a total of 13 flights. Each flight, taking approximately 10 minutes, visits 16 waypoints. We use a DJI M100 UAV, integrated with a TX2 for data logging and processing and CozIR-A CO_2 sensor. As a gas source we use a compressed CO_2 cylinder. The UAV flies at a constant height of 4 m, covering an area of 80 m $\times$ 45 m, with the bottle being placed at an unknown location somewhere within that area (see fig. 3.1). As we do not have access to the ground truth gas distribution, we evaluate our algorithm using *localization error* – the distance between the location of the maximum gas concentration and the true location of the cylinder (determined through GPS measurement).

Comparison with related work In the first set of experiments, we compare our algorithm against standard approaches from the literature, such as one using Gaussian Process regression [203] with a Radial Basis Function kernel, variance 15 and length-scale 7, as well as the TD Kernel DM+V/W algorithm [185], with cell size 0.2, kernel size 10, evaluation radius 10, time scale 1 and wind scale 0.001 (we perform grid search to find optimal parameters for the baselines). The collected air samples are used to fit a 2-d concentration map, with DM+V/W additionally using wind samples for reshaping the kernel function and scaling the readings based on the timestep taken. As previous approaches do not explicitly model the source location, we use the peak of the posterior as a proxy for this quantity [162]. We compare this against the computed l^* from OGS (Sim-Likelihood). For the simulation parameters, we use gas release 25, simulation

[3] We do not model the turbulent effect of the propellers - using a smoke flare, we visually inspected the effect of the propellers and found that it has little impact on the larger scale gas dynamics.

fidelity $1/1$ and diffusion $1e-4$ (see fig. 3.5). We show examples of the posterior mean (GP and DM+VW) and likelihood of the source of the gas (ours), together with the overall error between the methods in fig. 3.3.

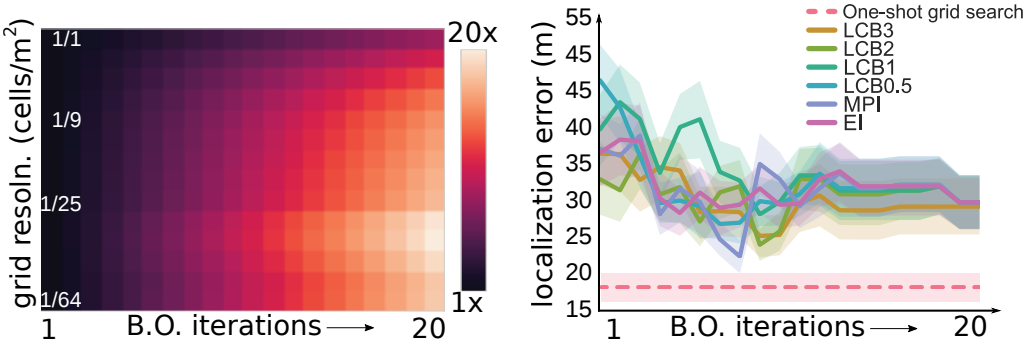


(a) Predictions (b) Error

Figure 3.3: **Predictions** (a) Posterior mean/variance of gas concentrations obtained using state of the art methods (column 1 [203] and column 2 [185]) against the likelihood computed using our method (column 3). Rows represent results of different experiments. The source location was the same for the first two experiments, but environmental conditions (wind) were different. (b) Overall error across all experiments.

BO vs one-shot grid search A standard approach to solving the optimization problem in Eq. 3.1 would be to treat the objective function as a random function with a Gaussian process (GP) prior over it. Then, based on each measurement, the prior could be updated to form a posterior distribution over the objective function. Based on this posterior, an *acquisition function* can be constructed to determine the next sample location. We perform experiments using this approach, in order to then compare the results against our proposed optimization procedure. We start with a GP prior using the Matern52 kernel, sampling from it twice to obtain a first (random) estimate l_0^*. Then, we estimate $d(g_i, \widetilde{g}(l_0^*, s_i, t_i|\theta, \widetilde{\mathbf{w}}))$ and use it to update the posterior (which only contains two samples). Based on this posterior, and using an acquisition function (Lower Confidence Bound, with alpha parameters 0.5, 1, 2 and 3, Expected Maximization or Maximum Probability of Improvement), we determine the next sampled source location l_1^*. We then continue sampling for new leak locations, as prescribed by the corresponding acquisition function. For each of these locations, we run simulations and update the GP posterior. We evaluate these results from BO against the corresponding values for OGS,

measuring running times and precision for the two optimization methods over a limited number of location samples l as shown in fig. 3.4.



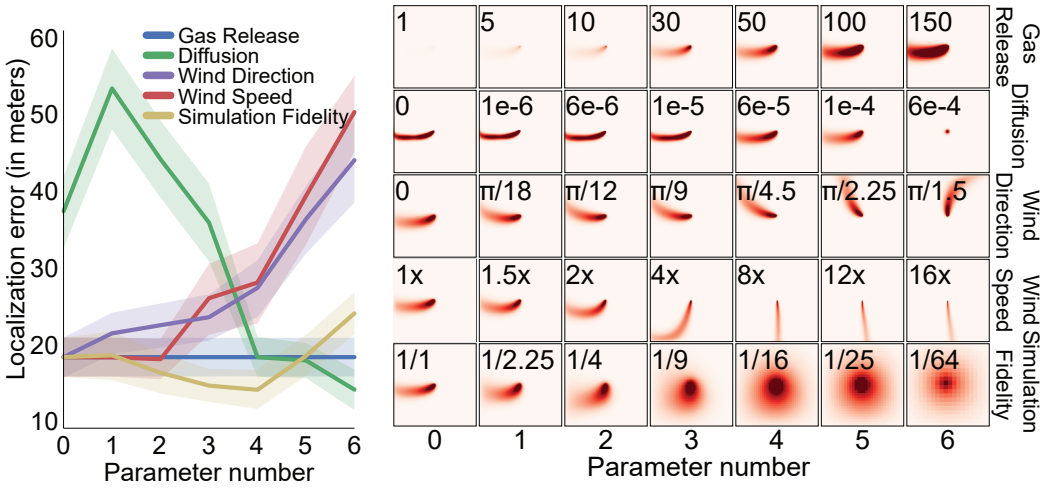
(a) speedup OGS: BO (b) Localization error

Figure 3.4: **One-shot grid search (OGS) vs Bayesian Optimization (BO)** (a) **Compute time** Speed up in the computational time for different resolution of the simulations. (b) **Localization error** We use all the experimental data collected from Sec. 3.4.1 to evaluate the localization error of different acquisition functions of BO with OGS.

Sensitivity to inaccuracy in parameters In order to determine the robustness of our approach to errors in the setup of the simulations, we perform a sensitivity analysis. We analyze the effect of different parameters on localization error by artificially perturbing the underlying values. We study the effects of inaccuracies in the quantity of gas released, diffusion coefficient, wind speed, wind direction and simulation grid resolution. We start with default values for the parameters (see caption of fig. 3.5), and then assess localization error when each parameter is individually modified to one of six different values. The resulting errors and the perturbed values of each parameter are shown in fig. 3.5.

3.4.2 Online algorithm

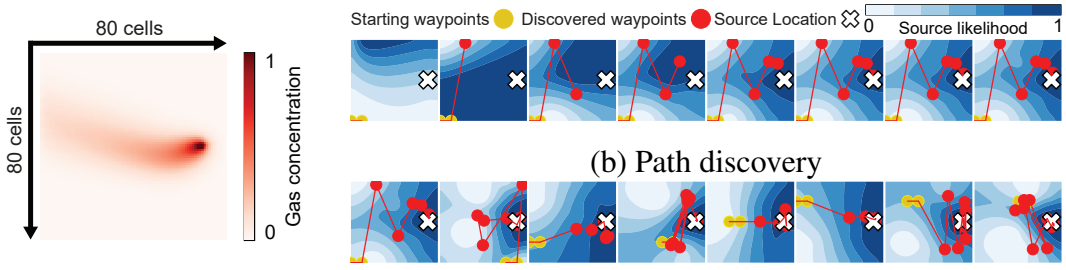
Noisy simulation We use sparse noisy readings taken every 20 seconds from a simulator (80×80 cells) with an arbitrary location for the gas source. We assume that the hyperparameters of the simulation are known and that wind flow is spatially constant, and we add up to 10% multiplicative noise to the simulated readings. We experiment with multiple start locations, both on and off the path of the gas as shown in fig. 3.6. We use the following parameters - gas release:20, diffusion:$6e-4$, wind speed:25 (simulator metric), wind direction:$\pi \pm \pi/2$ (primarily coming from the right, but uniformly changing its direction every 30 seconds).



(a) Sensitivity to hyperparameters

(b) Effect of the hyperparameters

Figure 3.5: **Sensitivity and effect of the hyperparameters** (a) **Sensitivity** Mean and variance localization error for all offline experiments for different parameters' variations. We start with the default parameters (gas release:50, diffusion:0.0001, wind speed:1x, wind direction:0, simulation fidelity:1 cell/1m^2) and individually vary each of them. (b) **Effect** Visualization of different values of the hyperparameters in a 160$\times$160 cell simulation. We use the wind flow from one of the experiments from Sec. 3.4.1 to drive the simulation, hence some of the observed dynamics.



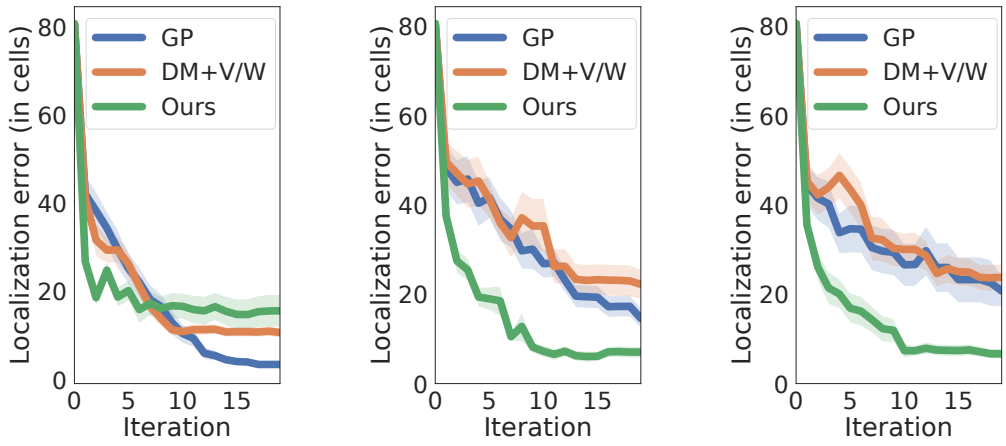
(a) Noisy simulator

(b) Path discovery

(c) Different initialization

Figure 3.6: **Active sensing using synthetic data** (a) **Noisy simulator** We use noisy readings from our model to evaluate the proposed approach – the source is placed in the middle-right, with the wind blowing East-West. (b) **Path discovery** Successive locations suggested during the discovery of one path are shown with red circles along with initialization locations (yellow circles). (c) **Different initialization** The convergence of the algorithm is illustrated for eight different starting locations.

Comparison with related work Using the noisy simulator, we also evaluate the convergence rate of different algorithms as shown in fig.3.7. We perform three sets of experiments - no wind in the simulation, constant wind and variable wind. An acquisition function is used for each of the algorithms to select each consecutive measurement point. For GP and DM+V/W we use Lower Confidence Bound with alpha parameter 3 and for our approach we use the suggested likely region. In addition to the baseline regression approaches, we also generate example trajectories for different wind conditions using Infotaxis with parametric plume model and parameters as in [225] as shown in fig. 3.8.



(a) No wind (b) Constant wind (c) Variable wind

Figure 3.7: **Convergence of active sensing** Convergence rate of different algorithms for different wind conditions - no wind (a), constant wind (b) and variable wind (c). In the experiments a noisy simulator was used.

Real experiments We perform three real-world active sensing experiments with the setup described in Sec. 3.4.1 (Supplementary material). The domain is $40m \times 40m$, and we use 1 grid cell per 4 m^2 in simulation. We perform optimization using OGS, for which the simulation and comparison calculations are performed within an on-board processor. After each taken reading (gas and wind), the UAV updates its beliefs about the likelihood of the source, saves the current state of the optimization and flies to the suggested by the optimization most likely source location l^*- repeating until the optimization procedure converges to the same location. In our experiments, we had strong winds from the Southwesterly direction in the first two experiments and weak varying winds in the third experiment. The estimated likelihood and waypoints

discovered, are shown in fig. 3.9.

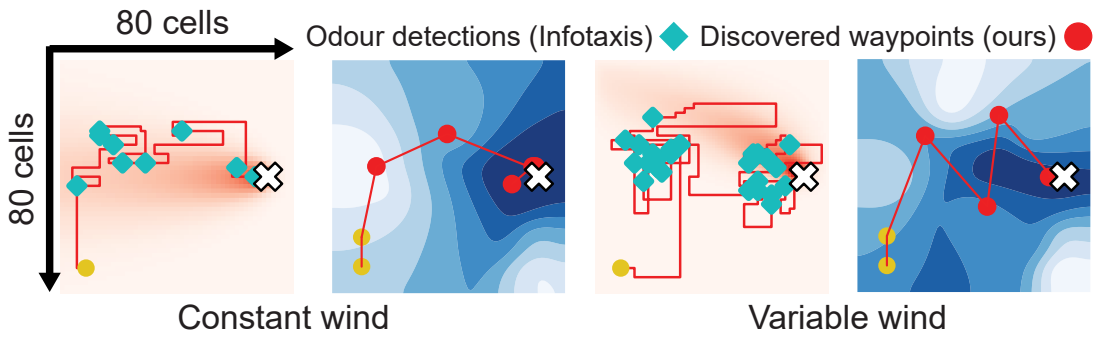


Figure 3.8: **Trajectories** Generated trajectories for the proposed approach and 'Infotaxis' with parametric plume model[225]. Odour detections with gas flow (for Infotaxis) and discovered waypoints with source likelihood map (for ours) are shown.

3.5 Interpretation of results and discussion

More accurate than state-of-the-art. In practice, alternate methods based on Gaussian process regression and the DM+V/W algorithm provide similar results. DM+V/W performs somewhat better, as it includes reshaping of the smoothing kernel based on wind information. As we show in fig. 3.3, these methods often predict the highest mean to be away from the true location, caused by the wind moving the released gas. Although DM+V/W indirectly accounts for those effects via wind kernel reshaping and time scaling, convergence is slower. By using wind measurements in simulation, we are able to capture these dynamics, allowing our proposed approach to achieve lower localization error ($< 20m$) than GP ($45m$) and DM+V/W ($< 40m$). Likewise, our approach achieved better performance than with standard BO approaches to compute the minimum ($35m$). Using fluid simulation yields a higher computational cost compared to traditional approaches. However, we show that our algorithm is fast (up to $20\times$ faster than BO) and can be implemented within a commercial on-board computer.

Fidelity and sensitivity. Our algorithm is robust to inaccurate parameters as seen in fig. 3.5. The extent of released gas did not have any impact on localization, as both simulated and real-world readings are normalized. However, it is sensitive to the diffusion coefficient (green curve) used in simulation, as well as to errors in wind estimation. We found that we can tolerate $2\times$ inaccuracy in wind speed and an offset of $\pi/12$ in direction from the calibrated mappings. We use a standard approach [162, 163]

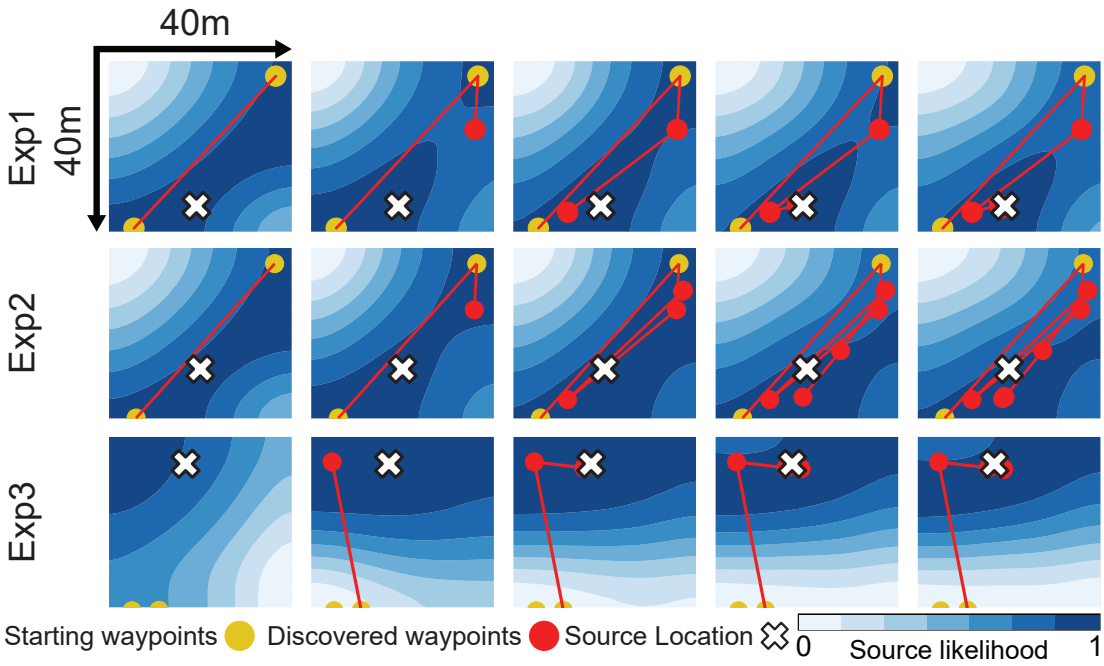


Figure 3.9: **Active sensing by UAV** The waypoints visited by the UAV. The fluid simulator and OGS are executed at each step, as described in the text. We initialize the algorithm with two predefined waypoints (yellow circles), after which the UAV automatically seeks the source using our online algorithm for active sensing. Our algorithm converges in 5, 7 and 4 iterations, in those examples.

to estimate wind, which lies within the required tolerances. We observed, curiously, that a grid resolution of 1 cell per $16m^2$ yields best results.

Active sensing. Our exploitation strategy consistently locates the source using very few samples, to similar accuracy as the offline algorithm. At each step, we update our uncertainty regarding possible locations. As with approaches that utilize gradient-search, we are able to follow the path of wind and gas readings. In contrast to those methods, we can also locate the source if we are not on the path of the gas. Importantly, we find that variable wind tends to be informative leading to faster convergence, unlike in the case of competing methods where this is a limiting factor. As we normalize the readings, we indirectly account for different rates of gas release. Thus even when on the path of the gas flow, the algorithm explores the neighbourhood of the path. Similarly, when not on the path of the gas, suggested locations by the algorithm sometimes oscillate before convergence. Having a stronger prior over the rate of gas release could further speed up this optimization process.

Source of error, limitations and future work. Although our method improves on state of the art in gas localization, the error is still about 15m. We identify two potential sources for this residual error. Firstly, our model is based on 2D space S while gas diffusion in the real world happens in 3 dimensions. Thus, even though our algorithm finds the highest density of the gas on the plane of flight, it may not be exactly above the nozzle releasing the gas. A second source of error is that we model the wind as spatially invariant, which is a coarse approximation for increasingly larger domains. It would be beneficial to extend this work by using simulations supporting obstacles and multiple sources [150]. This will require development of extensions of our method that are able to encapsulate the variability of the more complex simulations from limited samples, while preserving fast inference at run-time. An interesting avenue of research would be the inclusion of multiple UAVs within the localization process, e.g. by decoupling the here presented approach and the optimization problem of optimal trajectories for the individual drones.

3.6 Conclusion

This chapter looks at sampling approach for simulation-based inference - exploting domain specific knowledge to reduce the simulation calls needed to just one. This is explored within the example task of gas localization of a gas leakage with a UAV.

We formulate gas-leak localization as an optimization problem, minimizing the

discrepancy between simulated gas flow and point-wise measurements of leaking gas. We propose a practical optimization algorithm that can be used offline as well as online for active sensing. We evaluate our algorithm by implementing it on a UAV equipped with sensors to detect CO_2. Our algorithm is able to cope with dynamically varying wind, and efficiently localizes the source of leaks even if the UAV is not initialized on the path of the gas. We show through experiments that our proposed approach outperforms baselines from the literature in its ability to minimize localization error.

This chapter focuses on one way of improving the speed of a sampling procedure - decreasing the number of samples. This can be done by domain specific assumptions, but also systematic analysis of the importance of different factors of variation, simulation fidelity requirements, alterantive sampling procedures, etc. The next chapter further develops the idea of efficient sampling for simulation-based inference by additionally also incorporating reduced simulation representation.

Chapter 4

RoboSuction: Safe and Efficient Suction under Partial Observations

In the previous chapter, we demonstrated how we can use fluid simulation as a model for the practical task of gas localization with a UAV. We exploit experimentally observed properties of outdoor wind dynamics - much stronger spatial over temporal speed and direction correlation over small fields and time windows. Focusing only on temporal wind variation can significantly simplify the parameter inference process to a single simulation call for all possible source locations. We use a simple myopic strategy of selecting the position with the highest likelihood of location of the source.

This chapter takes a fundamentally similar approach to exploiting problem-specific experimental observations to use fluid simulation as a model. The problem of interest is suction of liquid over an unknown surface with a robotic arm. Assuming that there is no active source of leakage, the simulation model can be constructed as one of contour tree with piecewise approximation functions for the height-volume levels. We use this approximation together with a model estimating the surface to optimize over the *entirety* of the trajectory.

4.1 Introduction

Manipulation of fluids by robotics platforms is desirable in many practical applications but challenging due to different challenges associated with acquiring data and developing appropriate representations. This is further complicated in uncontrolled environments, due to the associated uncertainty - both because of different safety and dynamics contraints that can be present, but also for planning optimal actions. While

certain scenarios can be addressed, where the state and phenomena can be efficiently represented, it is often unclear how to perform planning of non-linear phenomena and limited observations.

Safe autonomous suction is desirable for various medical applications, such as cleaning up excess blood during open surgery. Different safety constraints are present and must be respected, such as not touching the underlying surface. Moreover, planning can be challenging due to liquid dynamics and occlusions, making it difficult to estimate the unknown surface. Some prior over the type can often be assumed, but inference of the exact surface is needed for safe and efficient cleaning of the remaining liquid.

Videos of liquid manipulation often contain rich information about the observed phenomena. However, it is challenging to acquire a representation of interest due to the challenges of obtaining ground truth data on real robotics platforms - accidental spillage, resetting the environment when acquiring data, etc. Simulating fluids is challenging in its regard, especially when coupled with rigid and soft objects. Challenges include the computational cost of the simulations and the approximate nature of the computed solution, both for appearance and dynamics.

This chapter proposes modelling the fluid phenomena as a contour tree with piece-wise height-volume approximation functions. To address the partial observations, we model the underlying surface as a GP. This reduced representation and model of the surface allows for formulating the problem as one of shortest path computation. Time to clean individual nodes of the tree can be estimated by uncertainty propagation from the GP surface model. Those individual costs can then be optimised as a set of standard Travelling Salesman Problem (TSP).

4.2 Related work

4.2.1 Robot manipulating liquids

Manipulation of fluids by robotic systems has been of particular interest, with recent developments in modelling fluids with particle-based simulations [193], incorporating differentiable fluid dynamics within neural networks [198], and visual detection of liquids [192, 183, 194]. The majority of research has been dedicated to solving the pouring tasks by calibrating with a particle-based simulation by pouring [136] or stirring [84]; combining visual features within RL [58] and MPC frameworks [196, 195]; or purely control approaches [112, 36]. In addition to pouring, the question of minimizing

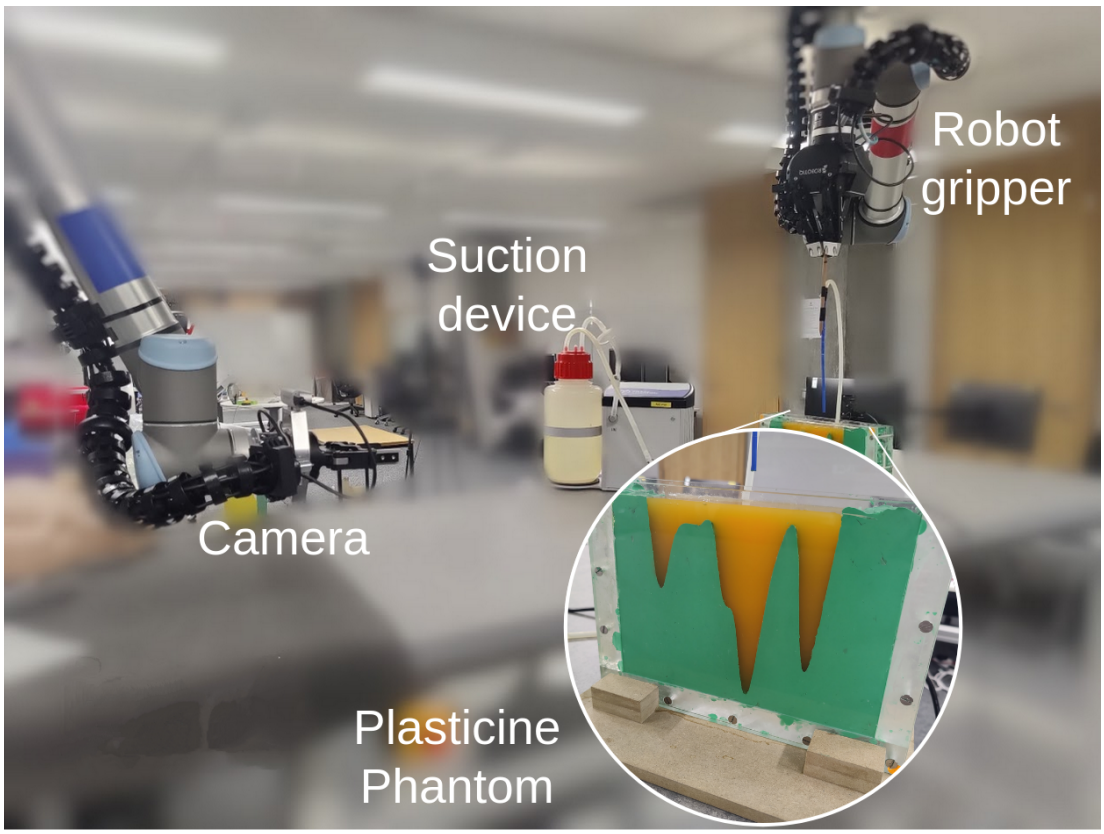


Figure 4.1: **Experimental setup.** We are interested in cleaning up liquid over unknown surfaces while avoiding contact in the shortest time possible. The robot has access to measurements only from the top, the transparent case is constructed to visualize the predictions as experiments are unrolled.

sloshing while handling liquids has also been explored [152]. While particle-based simulations have seen increased recent interest, more compact representations of fluid dynamics are possible, such as based on reeb graphs [138], that allow for more efficient computation where this is needed [43, 212, 137]. In this work, we explore the question of safe robot suction of fluids under limited observations, which has been of interest in the domain of robot surgery.

4.2.2 Planning under uncertainty

Planning methods include heuristics, creating objects shadows, computing complete policies, Monte Carlo methods, different approximations techniques, and provably safe methods [17, 73, 122]. A common way to reason about uncertainty is to use Gaussian Processes (GPs), where arbitrarily strong prior over the observed phenomena can be enforced by using an appropriate kernel and its parameters. As uncertainty estimates are

readily available within a GP, they have widely been used within control and planning frameworks [178, 177]. Moreover, safety has also been viewed in problems where a GP is used as a likelihood function with safety constraints that must be satisfied [210, 37] and its applicability to problems like quadrotors control [26]. While using GP for nonstationary processes can be challenging [233], different approximations have been proposed to reduce the computational time [106] [107]. By abstracting details, planning can often be solved as a travelling salesman problem (TSP) [165] [217]. In this work we use GP to model the underlying uncertainty, which with the addition of contour trees with volume information as a simulation model, can pose the problem of one of a TSP.

4.2.3 Robot surgery and suction

Robotic-assisted systems have been developed to enable and enhance surgeons' capabilities to perform complex surgeries [80, 213, 55]. With the wider deployment of robotics platforms in the operating theaters, recent work has explored whether different procedures can be automated, including cutting while distinguishing between normal and benign tissue by solving a proxy task of peeling a grapefruit [208]. The first and critical requirement for such systems in safety for the patient [95]. Even for more straightforward tasks such as removing overflowing blood during open surgery, it is essential not to damage exposed organs and tissue. The two main complexities that need to be addressed are the dynamics associated with the fluids and sources of leakage; the uncertainty coming from limited observations. Medical suction has recently been explored, with most of the work focusing on detecting where the fluid is and partially accounting for the dynamics by tracking the flow [186, 94]. In this work, we explore the complementary question of safety, particularly how we can execute safe actions while planning for optimal trajectories based on the current observations.

4.3 Problem Formulation

We consider the problem of cleaning liquid over unknown height function $f : S \to \mathbb{R}$, where without loss of generality $S \equiv [0,1]$. We start by describing eq.1.5, where some details are omitted, in more details. We have observations over the liquid level densely sampled over the domain S at each time point t, which we mark as $l_t : S \to \mathbb{R}$, where at t_0 that liquid level is constant over the entire domain S. As suction is performed, part of the underlying surface is unveiled, such that $l_t \approx f$ for some part of the domain S, and

access to a limited number of noisy measurements are available $y_i = f(x_i) + \varepsilon$, where $x_i \in S$. We are interested in cleaning all the liquid, such that $l_T \approx f$ over the entirety of the domain S, in the shortest time T possible, while maintaining a safe distance d from the surface f.

We make the following assumptions:

1) The function has the same smoothness properties throughout its domain

2) An initial safe action is performed, exposing some noisy measurements

3) There is no leakage of liquid within the surface

4.4 Methodology

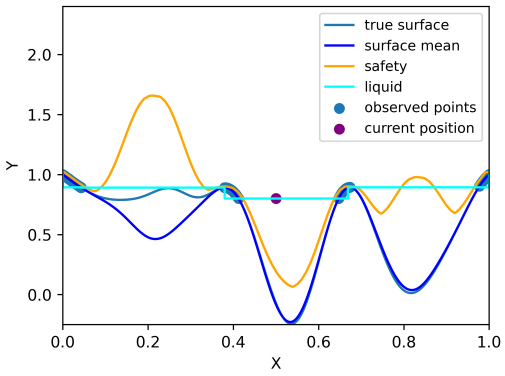
A naive solution is to use a no-model reactive approach, such as cleaning each cluster of liquid one by one by maintaining the center position of the nearest cluster. However, this has no safety guarantees for avoiding contact with the underlying surface. Different myopic strategies can be used, such as moving to nearby places to the current position of a defined safe region. Efficiently cleaning the liquid requires reasoning over the entirety of the trajectory is needed. Under the above presented assumptions, mainly the stationarity of the fluid and smooth underlying surface, this can be posed as a TSP between the surface's local minima. However, when safety constraints are present one needs to reason over distribution of potential surfaces and the safety bound evolution as actions are taken. In the following subsections, we describe how we can estimate the surface and define a safe region, and formulate the search problem a set of TSP while taking safety into account. We first describe how eq.1.6 can be implemented, with Sec.4.4.1 describing surface estimation and the safety bound and Sec.4.4.2 describing the reduced simulation representation and incorporating volume information. Finally, we describe eq.1.7 - how based on this representation an optimal action can be computed - in Sec.4.4.3. An overview of the proposed approach can be seen in fig.4.2.

4.4.1 Surface estimation and safety bound

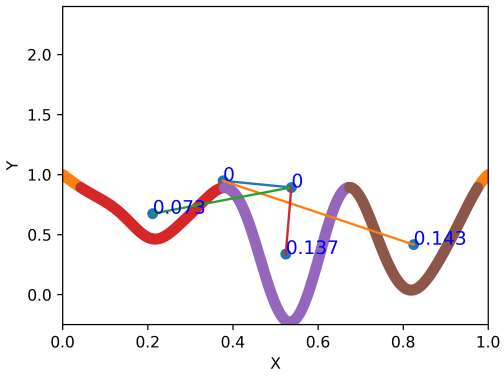
We model the surface as a Gaussian Processes $\tilde{f}(\cdot) \sim GP(\mu(\cdot), k_\theta(\cdot,\cdot))$,with a mean function $\mu : S \to \mathbb{R}$ and kernel function $k_\theta : S \times S \to \mathbb{R}$ **.** Different choices of kernel k_θ are possible, such as the Squared Exponential (SE), which parameters θ can be optimised with standard methods such as Bayesian Optimization. Around each point $x \in S$ standard confidence bound can be computed

$$h_{safe}(\cdot) := \mu(\cdot) + \alpha\sigma(\cdot) + \beta \tag{4.1}$$

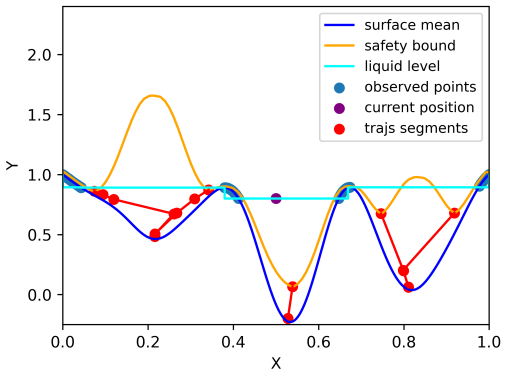
, where α is set based on the required safety threshold with an optional noise buffer β. This defines the safety region where suction can be performed safely. A schematic overview of the surface estimation and safety bound can be seen in fig.4.2, a).



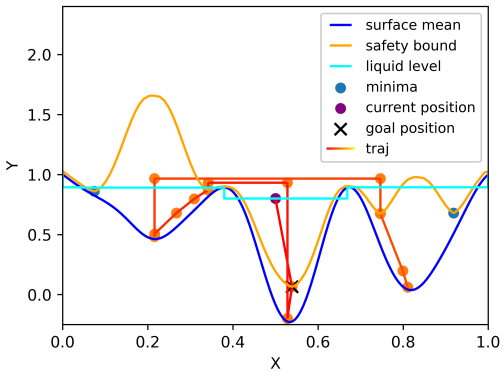
(a) Surface estimation and safety bound from observed points



(b) Contour tree with volume information from the surface estimation



(c) Trajectories for the individual components of the graph



(d) Best trajectory from the individual components

Figure 4.2: **Overview of the method.** We are interested in cleaning all safely reachable liquid in the shortest time possible. (a) Given the current uncovered surface points and the current liquid level, we fit a GP, compute a safety bound, and (b) construct a contour tree with volume information for the different segments. (c) Next, we individually compute the time required to clean the different segments by progressively selecting the point with the highest suction rate and updating the safety bound. (d) Finally, we solve a TSP problem with the individual segments and the distance between them. The goal position is executed, and then the process is repeated.

4.4.2 Approximate suction model

We construct a contour tree $T = \langle N, E \rangle$ based on the local extrema of the estimate of the surface (e.g., $\mu(\cdot)$), as an efficient representation of the current estimate of the underlying function, as seen in fig.4.2, b). Each node has the form $n = \{x_{min}, x_{max}, y_{min}, y_{max}, l, v, r, P, f_{forw}, f_{inv}\}$, where l is the current liquid height, v is the current volume, r is the rate of suction, $P = \{p_x, p_y\}$,$x_{min}, x_{max} \in S$, and for each p_x, p_y: $x_{min} < p_x < x_{max}$;$y_{min} < p_y < y_{max}$;$p_y = \mu(p_x)$. Additionally, we construct a piecewise function:

$$f_{forw}(p_y) = \int -\mu(p_x) dp_x, \mu(p_x) < p_y \tag{4.2}$$

approximating the relation liquid height to volume underneath, and the piecewise function $f_{inv} = f_{forw}^{-1}$ for the vise versa. Given the above defined approximation, we define an update function:

$$v_{t+1}, l_{t+1} = f(p_x, p_y, v_t, l_t) \tag{4.3}$$

, defined as following:

$$f(p_x, p_y, v, l) = \left\{ \begin{array}{cc} v, l & p_y > l \\ v - r, f_{inv}(v - r) & v - f_{forw}(p_y) > r \\ v - f_{forw}(p_y), f_{inv}(v - f_{forw}(p_y)) & v - f_{forw}(p_y) \leq r \end{array} \right\} \tag{4.4}$$

This allows for an efficient simulation of the suction phenomena, by using the tree for selecting the appropriate note for suction and the piecewise functions for computing the new liquid levels after a given action. Additionally, we construct a function for the current observations:

$$g(P, l) = \left\{ \begin{array}{cc} \{p_{x,} p_y\} & p_x, p_y \in P, p_y \leq l \end{array} \right\} \tag{4.5}$$

Within the next section, we unify and simplify the notation to $P_{t+1}, l_{t+1,} v_{t+1} = F(p_x, p_y, P_t, l_t)$, that encapsulates the contour tree creation, with the node construction, how nodes are update and how the current point observations are constructed - all defined above. We note l_t as the complete liquid level over all nodes, P_t as points densely sampled from $\mu(\cdot)$ and P_{t+1}as the observed points from the updated liquid level l_{t+1}. Within the next subsection we adopt this notation.

4.4.3 Planning under uncertainty

Planning over the entirety of the trajectory is challenging, as one needs to reason about the probability of each possible surface given the current observations. This is further complicated when planning with safety constraints, as the probability need to be updated at each timestep, as to update the safe region and potential actions. The safety bound h_{safe} defines the safe actions space, where suction can be performmed with low probability of approaching the underlying surface. We discretize this space, by only considering the actions at the local minimas of this safety bound. Alternatively, the complete trajectory for cleaning all liquid can be considered, by propagating uncertainty from the contour tree and GP as following:

1) Select action p_x, p_y from h_{safe}

2) Progress the model $P_{t+1}, l_{t+1,} v_{t+1} = F(p_x, p_y, P_t, l_t)$

3) Recompute h_{safe}

Even if we take a single surface estimation at each planning step (e.g., the mean of the GP), and we have just 2 possible actions at each steps (e.g., local minima of h_{safe}), there are still 2^n possible trajectories, where n is the number of steps. Alternatively, the optimization problem can be split as following:

1) Estimate the time needed to clean each individual node - e.g., always taking the next quickest action.

2) Estimate the time needed to traverse each node - once time is estimated for each individual node, we solve a set of standard TSPs (as the start point for cleaning a node may vary) for traversing the nodes in order.

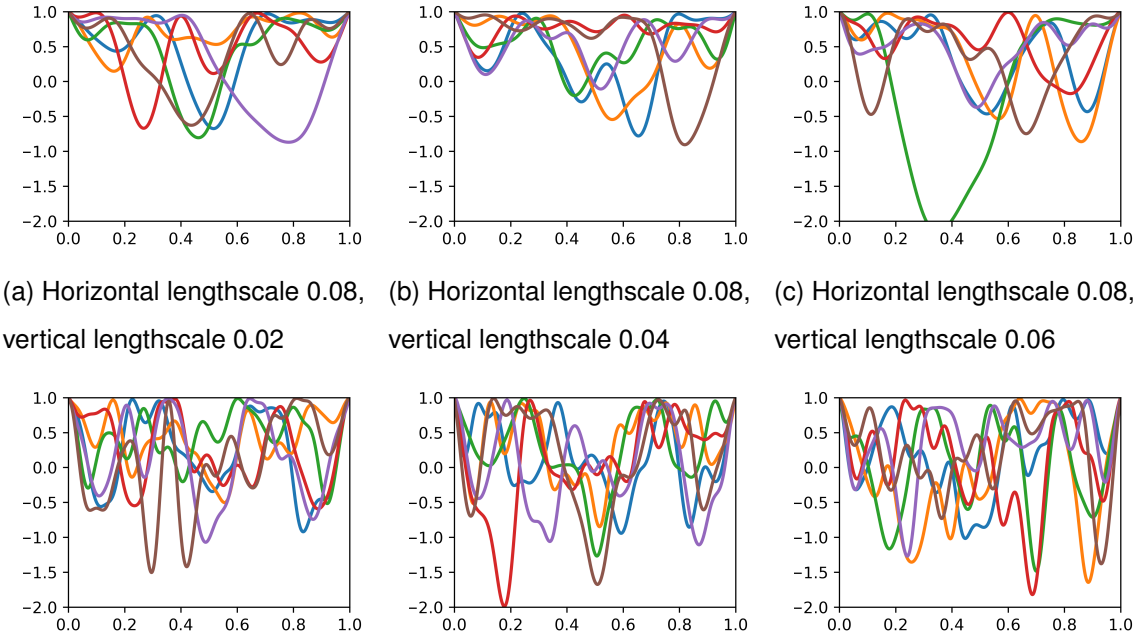
The described procedure can be seen in fig.4.2, c) and d).

Finally, we physically take the first action of the thus estimated trajectory with the robot arm, and replan based on the new observations.

4.5 Experiments

We perform a set of simulated experiments with procedurally generated surfaces. As a simulation model we use the described procedure in subsec.4.4.2. Within all experiments we assume the surface to be with the same smoothness properties throughout its domain, e.g., generated by a Gaussian Processes with a stationary squared exponential kernel, but don't assume to know its properties, e.g., horizontal and vertical lengthscale. Finally, to verify the applicability of the proposed method we perform an experiment on a

plasticine phantom. To evaluate different approaches we use two different metrics, distance to surface (safety) and time to clean all liquid (performance).



(a) Horizontal lengthscale 0.08, vertical lengthscale 0.02

(b) Horizontal lengthscale 0.08, vertical lengthscale 0.04

(c) Horizontal lengthscale 0.08, vertical lengthscale 0.06

(d) Horizontal lengthscale 0.04, vertical lengthscale 0.02

(e) Horizontal lengthscale 0.04, vertical lengthscale 0.04

(f) Horizontal lengthscale 0.04, vertical lengthscale 0.06

Figure 4.3: **GP surfaces.** The surfaces used for the experiments were generated from a Gaussian Process with a squared exponential kernel and varying hyperparameters.

4.5.1 Datasets

Two simulated datasets are generated. In the first, different surfaces are sampled from a GP with different hyperparameters. In the second, the GPs are additionally conditioned such that the surface have specific properties, e.g. certain number of initially visible peaks.

4.5.1.1 Vanilla GP surfaces

We generate a set of surfaces by sampling a Gaussian Process with a squared exponential kernel and different hyperparameters. We constrain the generated samples with fixed start and end points of the same height, with all other surface points below that height. The fixed points were implemented by fitting the GP model through them, the height constraint was implemented by rejecting any samples that don't meet this requirement.

We use the following hyperparameters: horizontal lenghtscale: $[0.08, 0.04]$ and vertical lengthscale: $[0.6, 0.3, 0.2]$. We generate a total of 50 surfaces per each type of parameter combination, so a total of $2x3x50 = 300$ surfaces. The thus generated GP surface samples can be seen in fig. 4.3.

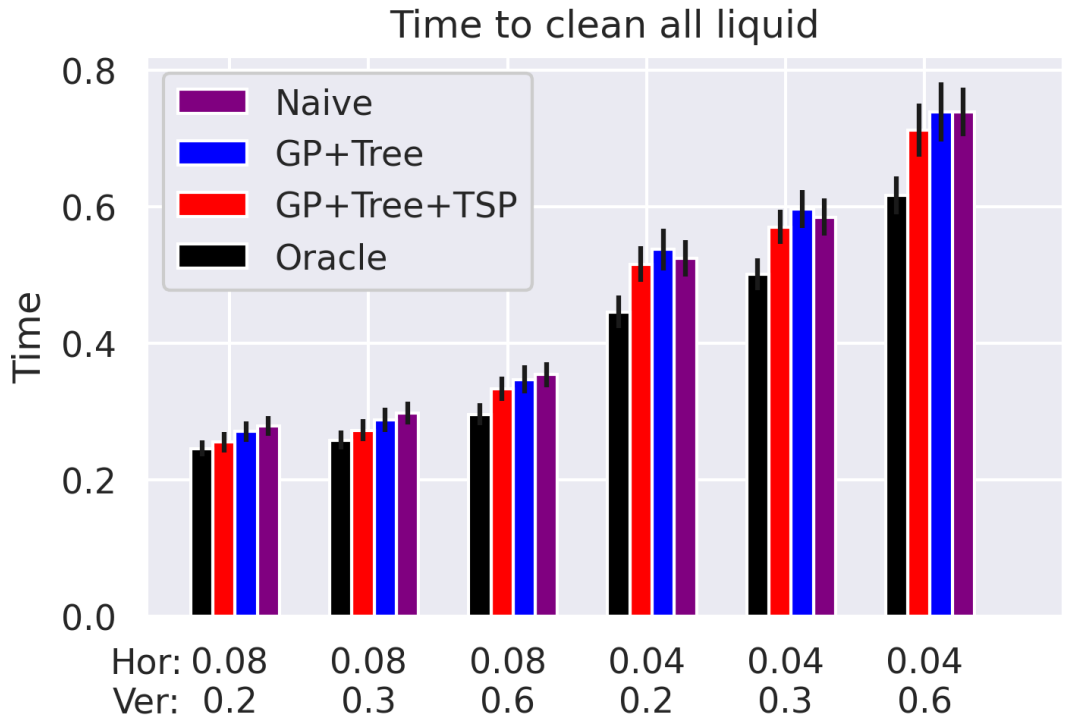
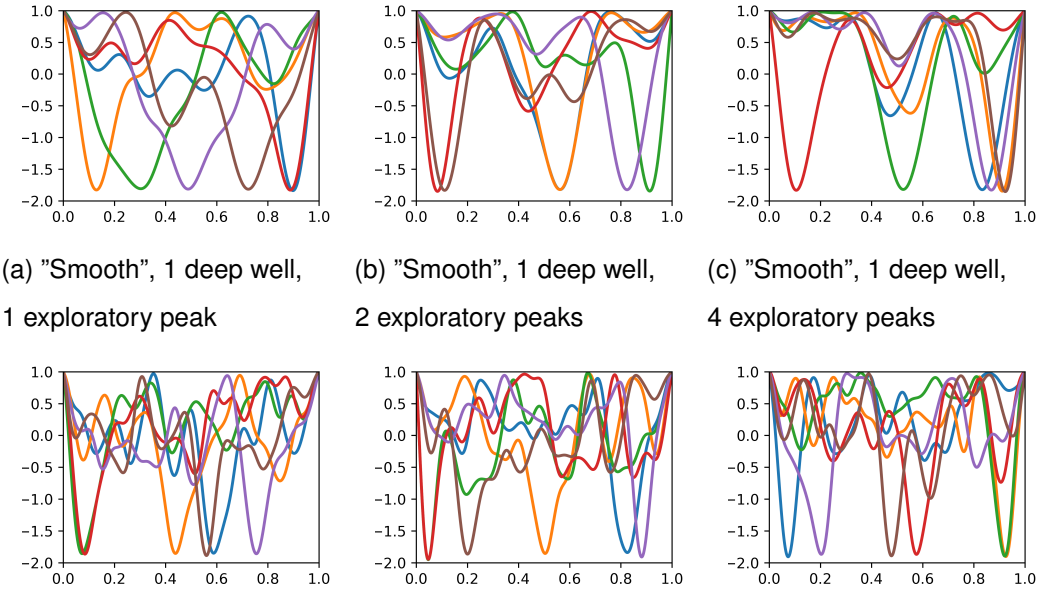


Figure 4.4: **Time to clean all liquid - GP surfaces - surfaces which are just samples from a GP with different hyperparameters**

4.5.1.2 Custom surfaces

To further quantify the performance of different methods, we generate a second dataset with more structured features. We vary the number of exploratory 'peaks' that appear after the initial predefined action. We additionally constrain all the surfaces to have a single considerably deeper well, compared to all the others. Both of those constraints were similarly implemented, by fitting the GP samples through a set of defined points and rejecting samples not fullfuling the given requirements. The final set of parameters are vertical lengthscale: 0.2 throughout all experiments, horizontal lenghtscale: $[0.08, 0.04]$, number of exploratory peaks: $[0, 1, 2, 3]$ and number of deep wells: 1 for all experiments. Again we generate 50 surfaces per all combination of conditions for a total of $1x2x4x1x50 = 400$. A set of the generated surfaces can be seen in fig.4.5.



(a) ”Smooth”, 1 deep well, 1 exploratory peak

(b) ”Smooth”, 1 deep well, 2 exploratory peaks

(c) ”Smooth”, 1 deep well, 4 exploratory peaks

(d) ”Noisy”, 1 deep well, 1 exploratory peak

(e) ”Noisy”, 1 deep well, 2 exploratory peaks

(f) ”Noisy”, 1 deep well, 4 exploratory peaks

Figure 4.5: **Custom surfaces.**

4.5.2 Baselines

We employ three set of baselines. As a lower bound of time needed to clean all liquid we compute a shortest path between all the local minima of each surface (Oracle). We implement no model baseline that select the next action by tracking the center of the closest liquid ’pond’ - as each ’pond’ is cleared, the closest nearby is further selected (Naive). Finally, we implement a myopic baseline combining subsec.4.4.2 and subsec.4.4.2, but without the full planning described in subsec.4.4.3 (GP+Tree). In the full proposed solution, we include the proposed planning procedure (GP+Tree+TSP).

4.5.3 Avoiding contact

By design the Oracle baseline maintains a distance from the surface as the surface is known. We observe that that the Naive baseline in general has minimum impact with the surface, but as seen in fig.4.7 and fig.4.8 this cannot be guaranteed at all times, e.g. a peak appearing right in the middle of the currently cleaned pond of liquid. Both the myopic (GP+Tree) and the full version (GP+Tree+TSP) of the here proposed method are able to maintain a desired distance from the surface, as the surface is explictily modeled and estimated based on the observed points and smoothness assumption.

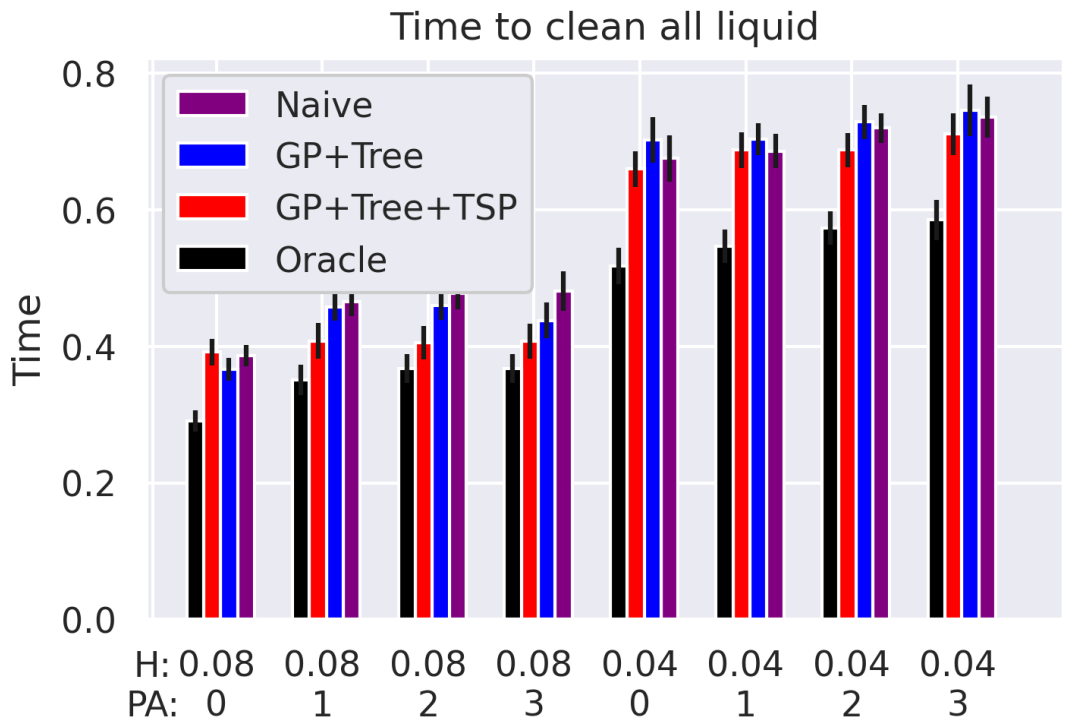


Figure 4.6: **Time to clean single well surfaces**. Time to clean surfaces with single 'deep' well, with variable smoothness (the higher the H parameter, the smoother the surface) and different number of initial exploratory points (PA) variable smoothness; variable initial exploratory peaks

(fig.4.7,fig.4.8).

4.5.4 Time to clean all liquid

We note that the Oracle baseline is the fastest, lower bound, method across all experiments. In the first set of experiments, with surfaces generated as GP samples, we the proposed full solution (GP+Tree+TSP) outperforms both the myopic formulation and Naive baseline as seen in Fig.4.4 - the proposed solution is on average 4.3 % faster than both myopic and naive baseline. In the second set of experiments, where we generate surfaces with single deep well and varying initial observed points and smoothness of the surface, we observe more varying results as seen in Fig.4.6. First of all, in one set of experiments our method performes the worst - in the case of no initial peak observations (PA: 0) and very smooth surface (H: 0.08). This goes to illustrate that if we have a relatively short experiment (the high smoothness of the surface), coupled with very

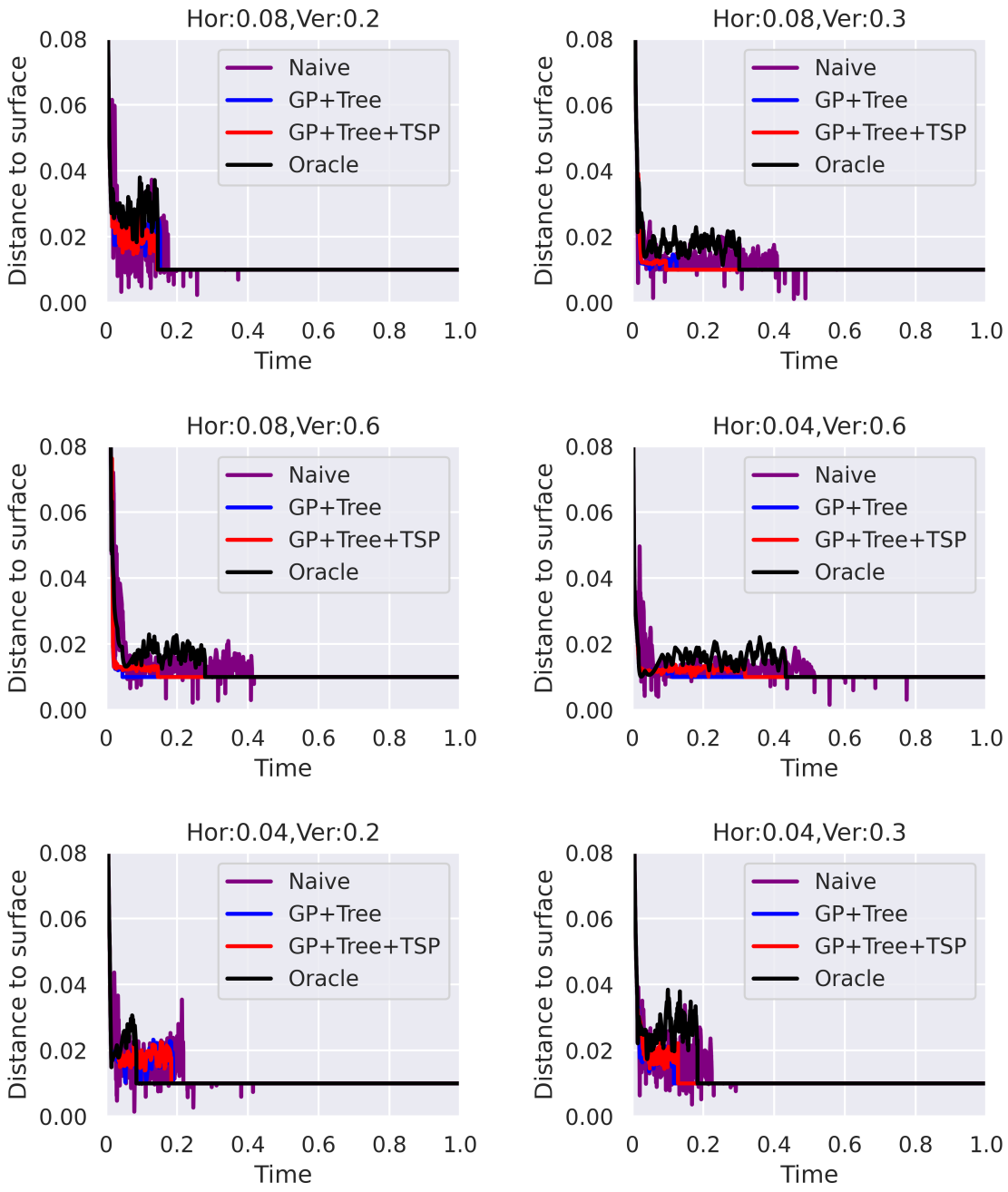


Figure 4.7: **GP surface samples distances.**

limited initial observations (no initial observed peaks), often simple myopic approaches might be better suited. As the initial observations are increased (H: 0.08, PA:1-3), we note that the proposed method approaches the performance of the Oracle baseline and the gap from other baselines increases - -1.4%, 12%, 15% and 15% (vs myopic); -7%, 11%, 12% and 7% (vs naive). For more complex surface (reduced smoothness

H:0.04), the main benefit of the proposed method is the overall smooth trajectory, that outperform the safe myopic approach and are on par or outperform the naive baseline, with the added benefit of safety. Expectedly for those more complicated surfaces, the gap with the Oracle baseline also increases. Overall for the complete set of experiments with dataset the proposed approach is 5.8 % (vs naive) and 5.3 % (vs myopic) faster.

4.5.5 Robot experiments

Finally, we perform robotics experiments, to compare the myopic and full version of our model. We construct a transparent enclosure using glass and plastic. Inside it we shape a smooth surface with plasticine as seen in Fig.4.1. We fill the phantom with juice and use a medical suction device attached to a gripper to clean all the liquid. Before starting the experiment the end position of the suction device and initial observed points are calibrated and normalized, so the predicted goal positions of the model can be directly used. Throughout the experiments, we assume that we see only the uncovered points above the liquid level for predictions (e.g., color segmented, top down view), while the transparent enclosure serves as a visualization of the surface underneath. For computational and safety reasons we precompute the entire execution trajectory for the experiment in advance by iteratively uncovering the parts of the real image that would have been cleaned by the robot. Finally, we execute the generated trajectories on the robot, with results which can be seen in Fig.4.9.

4.6 Results and Conclusions

In this chapter, we presented a safe long-term planning approach for cleaning liquid from partial observations. For the purpose we propose a reduced simulation representation based on contour trees with the inclusion of volume information, in addition to a modelling the surface as a GP to address the safety constraint of not hitting the surface.

Safety in medical suction. We demonstrate that we can maintain a safe distance from the surface by computing a standard confidence bound of the GP. We perform a set of experiments over different types of simulated surfaces, and demonstrate that naive approaches do indeed violate this constraint, while both of our formulations are able to avoid contact. Within our experiments we use uniformly smooth surfaces - an interesting future line of research would be whether similar methodology can be applied to more heterogenous and realistically looking surfaces by using non-stationary kernels

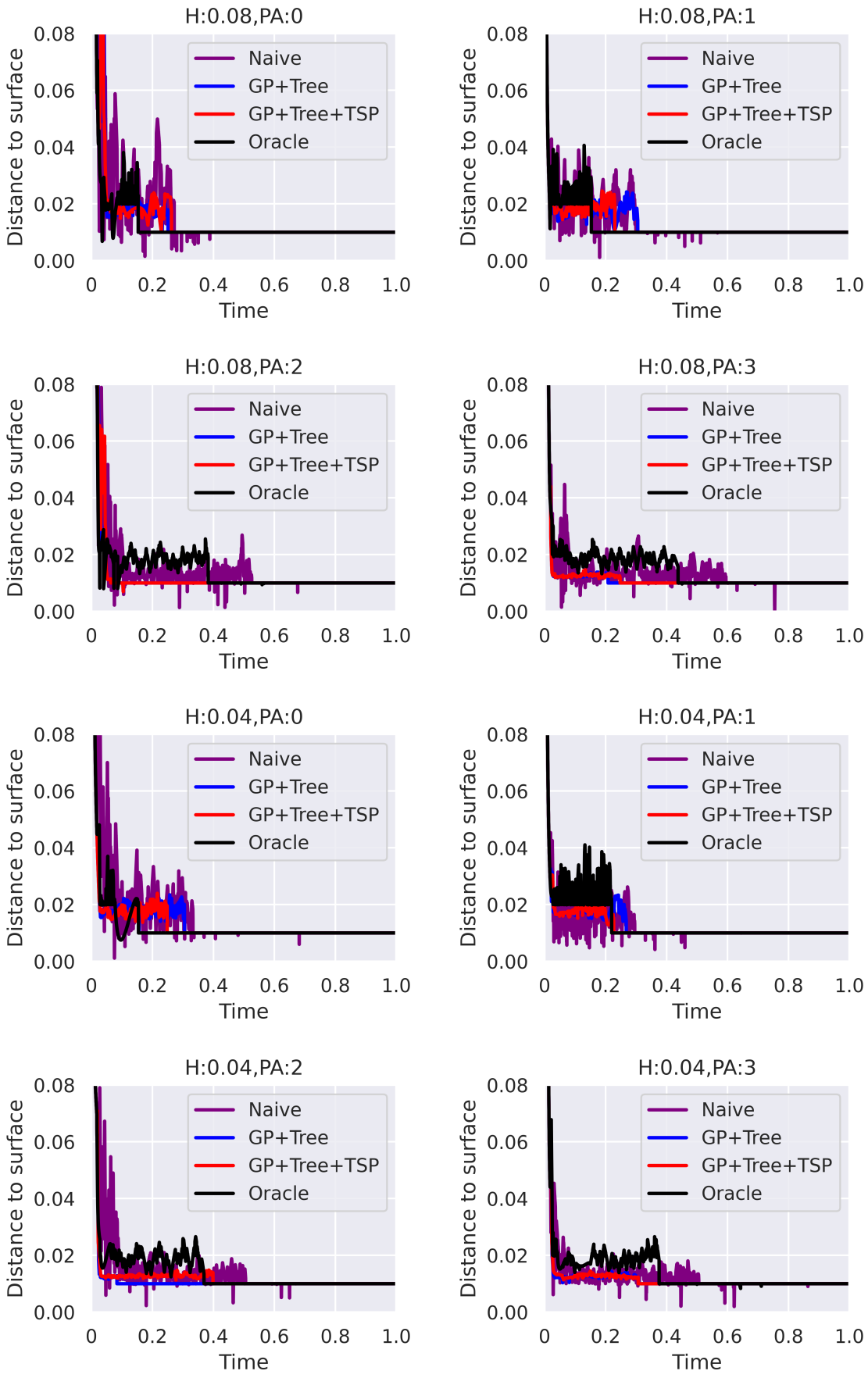


Figure 4.8: **Custom surface samples distances.**

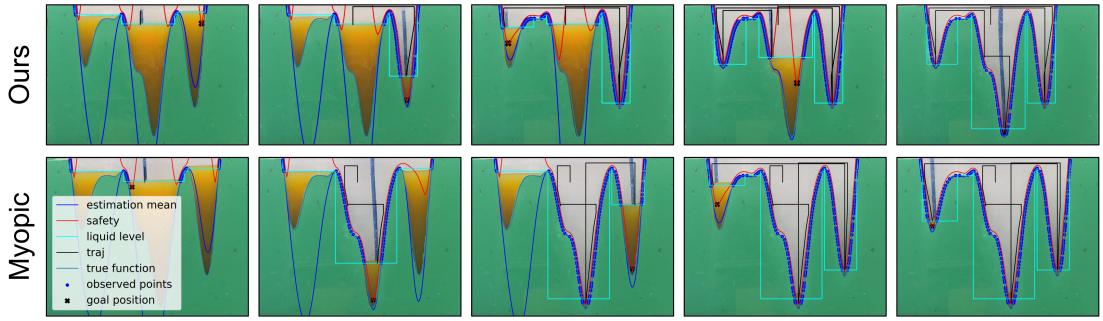


Figure 4.9: **Robotics experiments.** Comparison between the proposed approach and myopic strategy. The proposed method optimizes over the entirety of the trajectory, resulting in behavious such as leaving the deepest well to be cleaned last.

[168].

Long-term planning with changing state. A central question we address is how we can generate a trajectory that considers the entirety of the remaining liquid. We demonstrate that the proposed model outpeforms naive and myopic approaches solving the task. To further study the limitations of our model, we evaluate on a set of simulated surfaces with specific features such as number of observed peaks, a single deeper well, variable smoothness, etc. We show that given at least one exploratory peak, our method outpeforms other approaches, by generating overall smoother trajectories and optimizing for leaving the deepest well to be cleaned latest.

Constraints and future work. We perform a robotics experiment where we model a phantom with a plasticine to show the applicability of our method in real settings. It would be of interest to extend the proposed methodology to more realistically looking medical phantoms and the associated challenges of complex surfaces and visual observations. Furthermore, within our work we assume no active source of liquid, which again would be of interest as a future extension.

Chapter 5

Vid2Param: Modelling of Dynamics Parameters from Video

In the previous two chapters, we developed methods for efficient sampling of hand-crafted reduced simulation representations with the task of parameter inference and long term planning. In this chapter, we develop a method for learning the inverse mapping between sensory readings to parameters of interest directly. We apply our method for parameters inference from video and it's applicability for robot control, as well as inversion of ultrafast x-ray scattering data [1].

In this chapter, we focus on tasks with high observational complexity such as videos as other high dimensional, temporal data. Videos provide a rich source of information, but it is generally hard to extract dynamical parameters of interest. Inferring those parameters from a video stream would be beneficial for physical reasoning. Robots performing tasks in dynamic environments would benefit greatly from understanding the underlying environment motion, in order to make future predictions and to synthesize effective control policies that use this inductive bias. *Online* physical reasoning is therefore a fundamental requirement for robust autonomous agents. When the dynamics involves multiple modes (due to contacts or interactions between objects) and sensing must proceed directly from a rich sensory stream such as video, then traditional methods for system identification may not be well suited. We propose an approach wherein fast parameter estimation can be achieved directly from video. We integrate a physically based dynamics model with a recurrent variational autoencoder, by introducing an additional loss to enforce desired constraints. The model, which we call Vid2Param, can

[1] Sec. 5.4.2 is joint work with N. Zotev and A. Kirrander, who had major contributions in providing the data, describing the experimental setup, and establishing the used metrics. The contribution of this thesis is applying the proposed methodology to the inversion problem.

be trained entirely in simulation, in an end-to-end manner with domain randomization, to perform online system identification, and make probabilistic forward predictions of parameters of interest. This enables the resulting model to encode parameters such as position, velocity, restitution, air drag and other physical properties of the system. We illustrate the utility of this in physical experiments wherein a PR2 robot with a velocity constrained arm must intercept an unknown bouncing ball with partly occluded vision, by estimating the physical parameters of this ball directly from the video trace after the ball is released (subsec.5.4.1).

The proposed model can work with different types of temporal data, beyond videos. To further emphasise the ability of the proposed method to scale to complex dynamics and observations, we perform experiments on inversion of ultrafast X-ray scattering detector images. We compare against traditional baseline methods used in literature. We show that incorporating previous dynamics information leads to better inversion, partly mitigating the incomplete information in a single detector image (subsec.5.4.2).

5.1 Introduction

There is an ever growing need to perform robotic tasks in partially known environments. Reasoning about observed dynamics using ubiquitous sensors such as video is therefore highly desirable for practical robotics. Traditionally, the complexity of this reasoning has been avoided by investing in fast actuators [74] [159] and using very accurate sensing [109]. In emerging field applications of robotics, the reliance on such infrastructure may need to be decreased [85], while the complexity of tasks and environment uncertainty has increased [65]. As such, there is a need for better physical scene understanding from low-cost sensors and the ability to make forward predictions of the scene, so as to enable planning and control.

Recent advances have enabled video prediction conditioned on observations [126] and reasoning about complex physical phenomena [215]. Video streams provide a rich source of information, but it is often challenging to acquire the compressed structured representations of interests. Techniques for system identification, originally developed for process control domains, are aimed at this problem [133]. There are a number of different approaches to estimating parameters [102], and sometimes even model structure [129], from observed data.

Acquiring reduced representations of the environment and performing system identification have historically been disjointly solved, despite the rich contextual information

images often contain. This may lead to slower inference or even failure of the optimization should the state space model reduction be inaccurate. Yet, solving the problem jointly brings a set of practical challenges. First, it is difficult to acquire the necessary training data to cover the possible variations of the task of interest and generalize to unseen data. Secondly, a model needs to learn to perform probabilistic inference of parameters of interest and generation from a sequence of images, and capture long-term dependencies. From a practical robotics perspective, the model needs to be sufficiently fast to be able to perform system identification on-the-fly, while performing inference directly from images.

In this chapter, we focus on the case where a robot must perform robust system identification *online*, directly from a rich sensory stream such as video (including the implicit tasks of detecting and tracking objects). We pose the problem as learning an end-to-end model, where we regress from videos directly to parameters of interest. Furthermore, we structure the learned model so as to be able to make probabilistic future predictions in the latent space, to enable appropriate action selection. This allows for interacting in relatively unknown environments when system identification must be performed on-the-fly for the successful completion of a task.

Figure 5.1: **Overview and experimental setup.** We are interested in reasoning about the dynamics in an environment, by using a single video stream as a sensory input. To demonstrate the utility of our approach, we use a slowly actuated robotics arm to perform a 'stopping a bouncing ball' experiment, where the physical properties of the ball or height of the table are not known a priori and occlusions are present. Our model is able to perform online inference of the parameters of interests and generate plausible future trajectories.

We present a model that enforces physical conformity between videos and dynamical

parameters of interests. We integrate an analytic simulator with a recurrent latent model [46] by introducing an additional loss term for encoder-decoder mapping from a given sensory input (vision) to physical parameters and states (position, velocity, restitution factor, gravity, etc. in a parametric description of a physics-based model). We show that such a model can be trained with suitable domain randomization [218] in simulation and deployed in a real physical system. This model, which we call Vid2Param, allows for forward predictions envisioning possible future trajectories based on uncertainty in the estimate of the physical parameters. We demonstrate that such a model can indeed perform accurate system identification directly from videos by demonstrating that we achieve similar levels of performance as traditional system identification methods, which have access to ground truth starting trajectories. We perform experiments on simulated and real recorded videos of a bouncing ball with different physical properties. To illustrate the utility of this capability, we demonstrate this model on the task of intercepting a bouncing ball with a relatively slow moving robot arm and standard visual sensing.

5.2 Related Work

5.2.1 System Identification

System identification (SysID) is concerned with the problem of determining the structure and parameters of a dynamical system, for subsequent use in controller design. The best developed versions of system identification methods focus on the case of linear time-invariant (LTI) systems, although almost all of these methods have also been extended to the case of nonlinear and hybrid dynamical systems. With these more complex model structures, the computational complexity of identification can be relatively high even for moderately sized data sets. Examples of system identification procedures that could be applied to our problem domain, including the additional step of reducing model order, include the Eigen system realization algorithm [108] and Balanced POD (BPOD) [188] (which theoretically obtain the same reduced models [139]), and the use of feedforward neural networks [44]. BPOD can be viewed as an approximation to another popular method, Balanced truncation (BT) [190], which scales to larger systems.

Another way to approach the problem of identification is frequency domain decomposition [33]. Recent approaches in this vein include DMD [121] and Sindy [35], which allow for data driven, model-free system identification and can scale to high-

dimensional data. When performing SysID directly from a rich sensory stream like video, it is not always clear what the optimal reduced representation should be [4]. We exploit the fact that a physics based model of objects can provide useful regularisation to an otherwise ill-posed identification problem.

5.2.2 Simulation alignment

When a parametric system model is available, simulation alignment can be performed to identify the parameters of the system. A standard approach is to perform least squares minimization or maximum likelihood estimation, for instance computing best fit parameters to align simulator traces to observed data [229]. When simulation calls are expensive, a prior over the parameter space can be enforced, e.g., Gaussian Processes, and Bayesian Optimization can be used [187] [182] [136] [12]. Our approach is closely related to [234] as we use supervision during the training phase of our model, and then use this learned approximation at test time. We also employ domain randomization while training our model [171] and our work follows a similar line of reasoning to that of [42], which aims to align a simulator to real world observations as the model is being trained. We focus on the problem of aligning a model to online observations at test time, for predictive purposes.

5.2.3 Learnable Physics Engines

There has been increasing interest in learnable physics engines - for example learning complex factorization (at the object or particle level) from data [154] [22], using particle based networks for fluid dynamics [197] and in robotics [51]. By representing the problem in terms of a set of learnable components (graph representing objects/particle and relations, Navier Stokes equations, linear complementary problem for the above mentioned tasks) a physics engine can be learned from raw data. Similar approaches have been shown to scale to video data [227]. We explore the complementary problem of system identification (with an analytical or learned simulator), and propose a direct optimization approach by learning an inverse probabilistic physics engine. This builds upon ideas presented in [229], where an analytical simulator is used with traditional system identification approaches. Closely related work is presented in [180], where surface properties are learned using Physics and Visual Inference Modules.

A related question to learning interactions between objects is that of learning a state space system to represent these. This has been explored for individual objects [110]

[70], by using Kalman and Bayes filters for learning. State models and predictions have recently been explored in the context of videos involving multiple objects [92] through the use of Spatial Transformer Networks [99] and decompositional structures for the dynamics, as well as integrating the differential equations directly into a network [101].

Neural network based approaches have been explored wide range of complex physical phenomena such as simulating molecular dynamics [222, 199] as well as inverse problems in challenging physics domains such as reservoir simulations [160], light scattering by nanoparticles [172], deblending galaxy images [123] and fluid flow prediction [64] and many more [38].

5.2.4 Variational Autoencoder

VAE [115] have been extensively applied to image and video generation [66] [92]. Recently, VAE have been used in reinforcement learning to improve generalization by learning a master policy from a set of similar MDPs [9]. Closely related work is that of [7] where Variational RNNs are used to learn 'residual physics' [231] [202]. The addition of loss terms to the reconstruction and KL error terms have also been proposed, allowing for enforcement of multiple desired constraints [91] [8]. We extend this line of work, by demonstrating that such constraints can be applied in a recurrent model to satisfy physics properties.

5.3 Vid2Param for online system identification from videos

Problem formulation Given a set of sensory observations $x_{1:t-1}$, we are interested in predicting future observations x_t using a low-dimensional dynamics representation $p(z_t|z_{1:t-1})$.

$$\begin{aligned} p(x_t|x_{1:t-1}) &= \int p(x_t|z_t)\, p(z_t|z_{1:t-1})\, p(z_{1:t-1}|x_{1:t-1})\, dz \\ &= \int p\left(x_t|z'_t,\theta_t\right) p\left(z'_t,\theta_t|z'_{1:t-1},\theta_{1:t-1}\right) \\ &\qquad p\left(z'_{1:t-1},\theta_{1:t-1}|x_{1:t-1}\right) dz \end{aligned} \tag{5.1}$$

Here, we decompose the latent space $z = [z', \theta]$ into the physical dynamics parameters of interest, θ, and a remaining z' term, used for image reconstruction and to capture potentially un-modelled effects. We illustrate how this model can be learned in an end-to-end fashion from videos, and how we can use the predictions in the latent space for model predictive control.

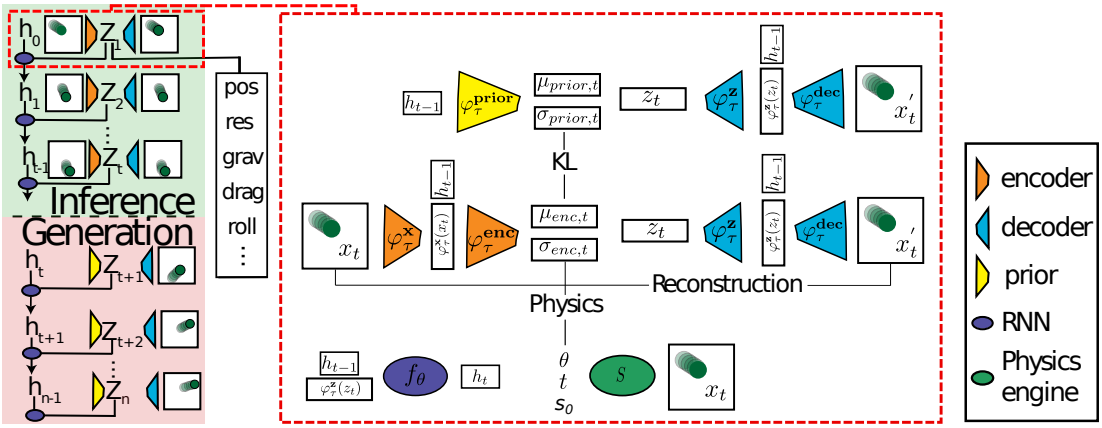


Figure 5.2: **Technical details and notation.** We propose an end-to-end model for performing system identification directly from rich sensory input, such as video. We based our model on a Variational Recurrent Neural Network (VRNN) [46]. To encode the physical properties of interests we propose an additional Gaussian negative log likelihood loss between the parameters of interest and part of the latent space. The inference and generation overview of the model (left) and the training procedure at each frame (right) can be seen in the figure. The presented figure describes the training pipeline. During online inference only the encoder network and RNN are used for estimating different parameters; or prior network and RNN for future trajectory generation.

Variational Recurrent Neural Networks We implement the inference process above using a modified recurrent VAE (VRNN) [46]. A VRNN consists of an RNN encoding the dynamics of the sequence, and a VAE conditioned on those dynamics, by including the hidden state of the RNN at each step. As shown in fig.5.2, the variational auto encoder φ_τ^{enc} (τ denotes the trainable parameters of the neural network) takes in a latent representation of the input $\varphi_\tau^x(x_t)$, in addition to the hidden state of an RNN, h_{t-1}. The encoder network then produces the mean and variance components, $\mu_{enc,t}$ and $\sigma_{enc,t}$ respectively, of a multivariate Gaussian distribution, q, that are also conditioned on h_{t-1}, thereby capturing information about the dynamics of the sequence until t:

$$\begin{aligned} q\left(z_t|x_{\leq t}, z_{<t}\right) &= \mathcal{N}\left(\mu_{enc,t}, diag\left(\sigma_{enc,t}^2\right)\right), \\ \text{where } \mu_{enc,t}, \sigma_{enc,t} &= \varphi_\tau^{enc}\left(\varphi_\tau^x\left(x_t\right), h_{t-1}\right) \end{aligned} \tag{5.2}$$

Similarly, the generative component of the VAE is also expanded by including the hidden state h_{t-1}:

$$\begin{aligned} p\left(x_t|z_{\leq t}, x_{<t}\right) &= \mathcal{N}\left(\mu_{dec,t}, diag\left(\sigma_{dec,t}^2\right)\right), \\ \text{where } \mu_{dec,t}, \sigma_{dec,t} &= \varphi_\tau^{dec}\left(\varphi_\tau^z\left(z_t\right), h_{t-1}\right) \end{aligned} \tag{5.3}$$

This means that the prior is no longer a standard normal distribution $\mathcal{N}(0,1)$ as in the case of a vanilla VAE, but is specified by a network conditioned on the hidden state h_{t-1}:

$$\begin{aligned} p\left(z_t|x_{<t}, z_{<t}\right) &= \mathcal{N}\left(\mu_{prior,t}, diag\left(\sigma_{prior,t}^2\right)\right), \\ \text{where } \mu_{prior,t}, \sigma_{prior,t} &= \varphi_\tau^{prior}\left(h_{t-1}\right) \end{aligned} \tag{5.4}$$

Finally, the RNN updates its hidden state h_t with a transition function f_ψ (with parameters ψ) by taking in a latent representation of the input $\varphi_\tau^x(x_t)$ a sample from $\varphi_\tau^z(z_t)$, and the previous hidden state h_{t-1}:

$$h_t = f_\psi\left(\varphi_\tau^x\left(x_t\right), \varphi_\tau^z\left(z_t\right), h_{t-1}\right) \tag{5.5}$$

VAEs are trained by maximizing an evidence lower bound [115]. Given the VRNN modifications, the overall loss, including a Kullback-Leibler (KL) divergence and reconstruction loss term, becomes:

$$\begin{aligned} \mathbb{E}_{q(z\leq T|x\leq T)}\Bigg[\sum_{t=1}^{T} \log p\left(x_t|z_{\leq t}, x_{<t}\right) - \\ \mathrm{KL}\left(q\left(z_t|x_{\leq t}, z_{<t}\right) \| p\left(z_t|x_{<t}, z_{<t}\right)\right)\Bigg]. \end{aligned} \tag{5.6}$$

In summary, the VRNN is a modified VAE that makes use of a learned low-dimensional dynamics model to make sequential predictions. However, the VRNN provides no

guarantees that the latent representation is meaningful, and as a result is unsuited for use in control or for system identification.

Vid2Param We propose combining the encoder-decoder factorization with dynamics modelling in the latent space z, conditioned on parameters of interest, θ. We introduce an additional loss to the standard VRNN to encourage encoding of physically meaningful parameters, by including a Gaussian negative log likelihood [166] loss between part of the latent space z and the physical parameters θ we are interested in (e.g., gravity, restitution, position, etc., in the case of a bouncing ball). The loss terms are scaled with non-negative numbers α, β and γ and we let N_θ represent the size of the parameter vector.

$$\begin{aligned}\mathbb{E}_{q(z\leq T|x\leq T)}\Big[\sum_{t=1}^{T}\Big(&\alpha\log p\left(x_t|z_{\leq t},x_{<t}\right)-\\&\beta\mathrm{KL}\left(q\left(z_t|x_{\leq t},z_{<t}\right)\|p\left(z_t|x_{<t},z_{<t}\right)\right)+\\&\gamma\sum_{i=1}^{N_\theta}\Big(\frac{1}{2}\ln(2\pi)+\frac{1}{2}\ln\left((\sigma^i_{enc,t})^2\right)+\frac{(\theta^i_t-\mu^i_{enc,t})^2}{2(\sigma^i_{enc,t})^2}\Big)\Big)\Big]\end{aligned} \tag{5.7}$$

Given trained networks, we can now perform probabilistic inference of physical parameters from sensory data such as a sequence of images using the factored portion of the latent space directly.

$$P(\theta^i_t|x_{\leq t})=\mathcal{N}(\mu^i_{enc,t},\sigma^i_{enc,t}) \tag{5.8}$$

Additionally, we can sample plausible future extrinsic properties (eg. positions) by recursively updating the model predictions to generate future states.

$$P(\theta^i_t|x_{<t},\theta_{<t})=\mathcal{N}(\mu^i_{prior,t},\sigma^i_{prior,t}) \tag{5.9}$$

As a final modification, we exclude x_t from the recurrent step, since all necessary information is already present in z_t. This speeds up the prediction in the latent space, as x does not need to be reconstructed and fed back into the network at every step. As such we can make recursive future predictions entirely in the latent space,

$$h_t=f_\psi\left(\varphi^z_\tau\left(z_t\right),h_{t-1}\right). \tag{5.10}$$

To summarise, the contributions of this chapter include:

1. Extension of the VRNN model with a loss term to encode dynamical properties.

2. Enabling faster future predictions in the latent space, along with uncertainty quantification through the identified parameters.

3. Evaluation of speed and accuracy of identification, against alternate approaches to system identification

4. Demonstration on a physical robotic system, in a task requiring interception of a bouncing ball whose specific physical parameters are unknown a priori, requiring online identification from the video stream.

5.4 Experiments

To evaluate the performance of the proposed method, we perform two set of experiments. In the first set of experiments we evaluate how well we can infer physical parameters of interest from videos (subsec.5.4.1). To demonstrate the utility of the approach we use the trained model for the task of stopping an unknown bouncing ball. In the second set of experiments, we evaluate how well the model scales on more complex observations and physics, such as inversion of ultrafast X-ray scattering detector images to molecular motion (subsec.5.4.2).

5.4.1 Videos and robot control

First, we perform a series of experiments on simulated videos, when ground truth is explicitly available. We compare our method against existing system identification methods, evaluating speed of estimation and accuracy of the identified parameters. Next, we evaluate our method on a set of real videos. Finally, we perform a physical experiment involving online system identification from a camera feed.

5.4.1.1 Setup

We use a bouncing ball as an example hybrid dynamical system. This is a particularly useful example, as the dynamics of the ball vary depending on the ball state, making system identification particularly challenging from high dimensional sensor data using classical techniques. The governing dynamics of the bouncing ball can be described

using the following set of ordinary differential equations:

$$S = \begin{cases} s_t = s_0 + \dot{s}_0 t + \frac{1}{2}\ddot{s}t^2, \ddot{s} = g - d & \text{Free fall} \\ \dot{s}_t^y = -e\dot{s}_{t-1}^y, \text{when } \dot{s}_t^y < 0; s_t^y = 0 & \text{Bounce} \\ \dot{s}_t^x = r\dot{s}_{t-1}^x, \text{when } \ddot{s}_t^y = 0 & \text{Rolling} \end{cases} \tag{5.11}$$

We use $s, \dot{s}, \ddot{s}$ for the current position, velocity and acceleration respectively, e is the coefficient of restitution and r the rolling resistance. Additional dynamic effects are often observed such as air drag $d = -c\dot{s}\sqrt{(\dot{s}^x)^2 + (\dot{s}^y)^2}/m$, where c is the drag constant and m is mass. Thus the acceleration becomes $\ddot{s} = g + d$, where g is the gravitational force ($g^x = 0$). As such, the system is completely determined by the initial state of the system $s, \dot{s}, \ddot{s}$ and its physical properties $e, r, m, c, g \in \theta$. It should be noted that the real world behaviour of any specific ball could deviate from this model depending on its shape, initial spin (Magnus effect), the presence of wind, and so on.

We make the following assumptions: 1) there is a single moving object, the bouncing ball 2) the ball bounces off a flat surface. We do not assume to know the physical properties of the ball, the height of the surface or the exact velocities of the ball, and we use a single low quality camera for sensing.

We use a parallel adaptive ODE solver to simulate data described by eq.5.11. We use these simulated trajectories to generate a sequence of images. We generate 10000 training and 100 test videos, with 200 timesteps/10 seconds, 28×28, with $e \in [0.6, 1.0]$, $g \in [-6.81, -12.81], d \in [0.05, 0.0005], r \in [0.0, 0.7]$. We split the parameters into 10 sub-ranges and alternately use them for training and testing, so no parameters used in the training data are available in the test data.

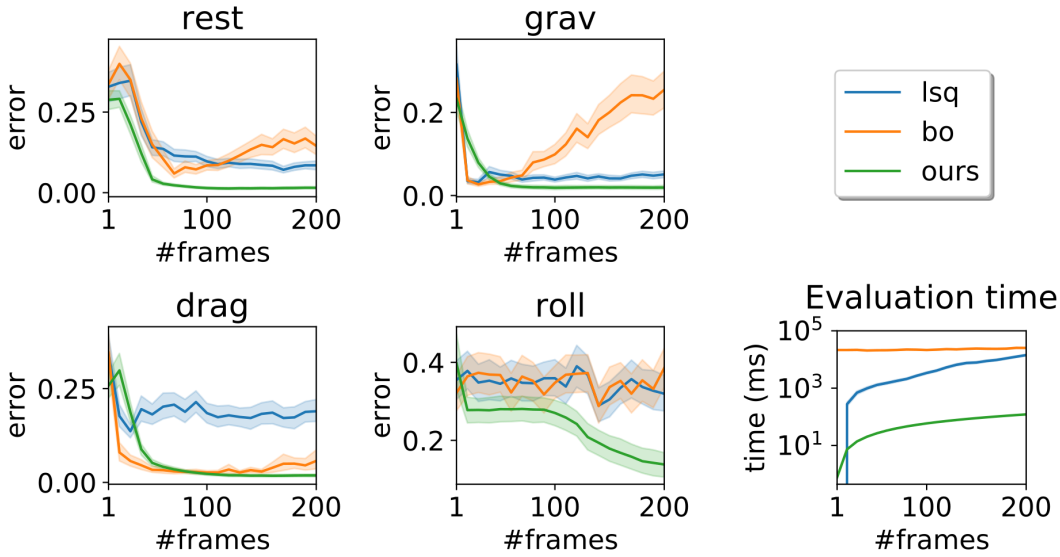
For the real videos and robot experiment, we trained a separate model with 5000 videos, 100×50 with 75 timesteps/10 seconds and the same physical parameters. Additionally, we add motion blur based on the velocity and black-out part of the frames to account for some of the missing/noisy data typically exhibited when using low-cost cameras. Additionally, we randomize the height of the plane on which the ball bounces. Our encoder-decoder network follows a similar architecture to [45] and for the RNN we use a standard LSTM network. We set $\alpha = 1$, $\beta = 1$ and $\gamma = 10$ throughout our experiments (eq.5.7). We use an NVidia 1080 Ti for training and laptop NVidia Quadro M2000M GPUs for testing the model. We use MSE as an accuracy metric throughout the chapter and normalize positions and parameters.

5.4.1.2 System identification

In this experiment, we evaluate the speed and accuracy of the proposed method against different simulation alignment approaches. We evaluate the likelihood of the observed trajectory with respect to simulated trajectories by performing MLE estimation (similarly to [229]). We sample 2000 trajectories by placing a uniform prior over the physical parameters and compare the simulated trajectories against those observed. We select the parameters which generated the least error between the trajectories, and compare these against parameters that generated the observed trajectory.

Secondly, we extend this by imposing a Gaussian Process prior over the parameters of interest and performing Bayesian Optimization [83] [182]. We use Expected Improvement as an acquisition function and use 20 optimisation steps. The baselines have access to the initial velocity, n number of positions (as such we don't need a tracker as in [229]) and an optimized ODE solver for sampling.

In contrast, our trained model receives *only* the video as an input, and no other parameters. Speed and accuracy benchmarks are shown in fig.5.3. It can be seen that our method has similar or better performance, despite not having access to the initial ground truth trajectory of the ball.



(a) Accuracy (b) Speed

Figure 5.3: **Performance of different system identification methods with variable number of observed frames.** (a) Overall error of the predicted normalized parameters (b) Speed of computation. We denote [229] as 'lsq' and [83] as 'bo'.

5.4.1.3 Forward predictions

Here, we evaluate the future prediction accuracy as frames are observed. In addition to previous baselines in Sec.5.4.1.2, we add an additional non-parametric model for system identification that approximates responses using a set of modes with different frequencies [121]. Three sets of predictions are evaluated - after 20, 50 and 100 frames respectively - until the end of the video at 200 frames, as shown in fig.5.4. We visualize example model predictions and their associated uncertainty in fig.5.5. Importantly, the proposed approach becomes more certain as additional frames are observed, highlighting the probabilistic nature of Vid2Param.

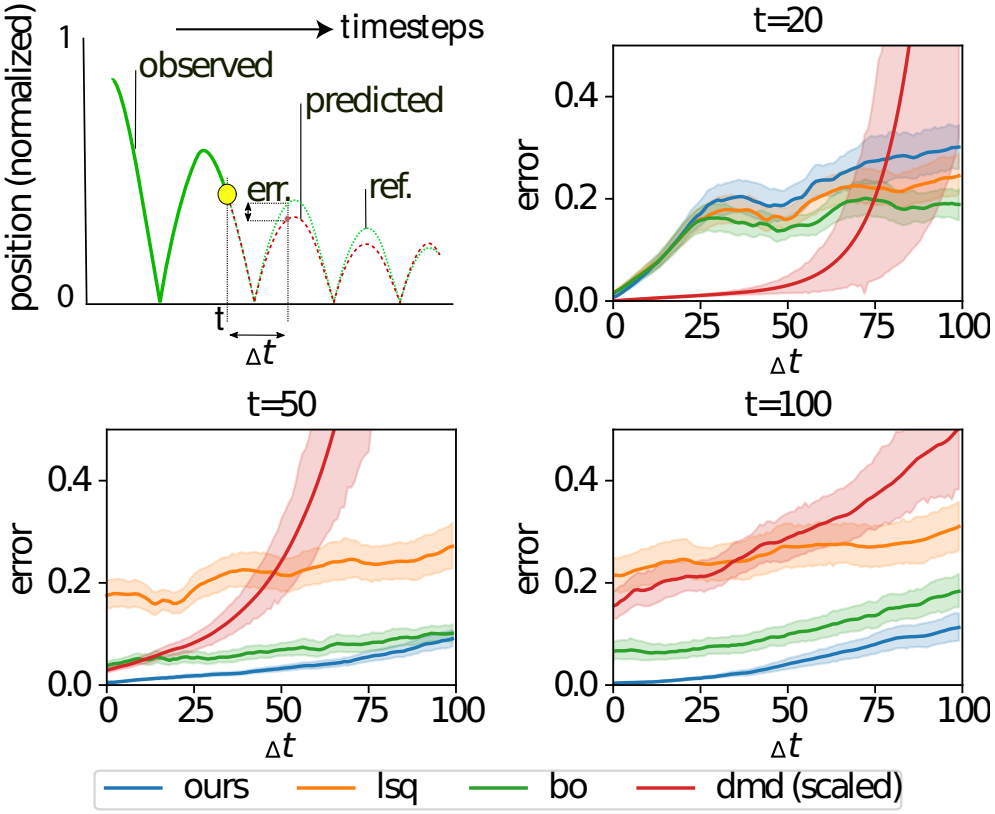


Figure 5.4: **Accuracy of forward prediction.** The accuracy is evaluated after 20, 50 and 100 frames are observed, predicting for the next 100 frames. The DMD error is scaled down 1k, 50 and 5 times respectively for predictions after 20, 50 and 100 observations. We note [229] as 'lsq', [83] as 'bo' and [53] as 'dmd'.

5.4.1.4 Varying physical properties and sensitivity analysis

In this experiment, we evaluate how well Vid2Param can estimate physical parameters when they are changing as the video is unrolled (using a model where parameters are

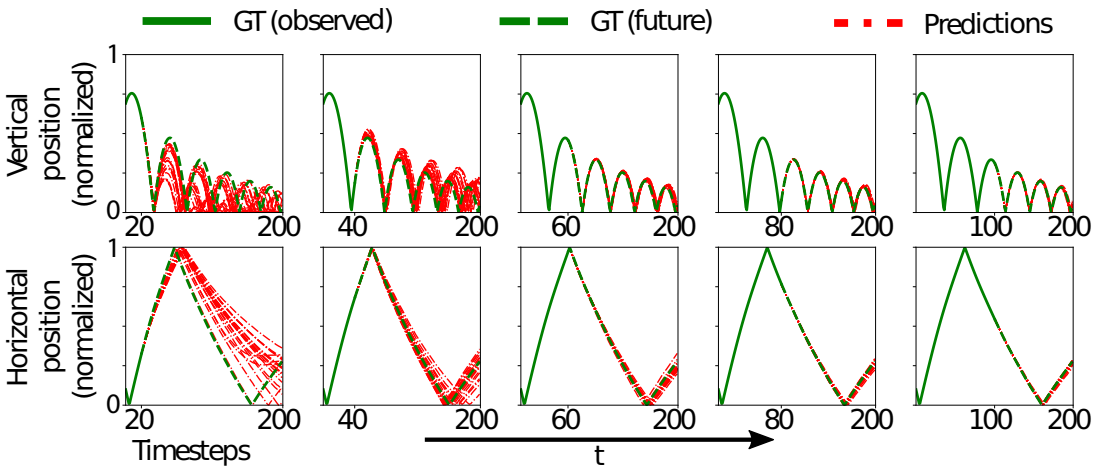


Figure 5.5: **Forward prediction uncertainty** after different numbers of observed frames (20, 40, 60, 80 and 100). Observed trajectory (green), future ground truth (dashed-green), predictions (red).

assumed to stay constant throughout the video). Therefore, this is a test of robustness or sensitivity of the model. We generate a new dataset, wherein the parameters change every 50 frames (2.5 seconds) in a given video. The results are shown in fig.5.6, and show that the proposed model can infer changing parameters, provided there is enough system excitation to facilitate this. For example, gravitation coefficients can only be inferred if the ball is bouncing, while the rolling coefficient can be inferred if the ball is rolling.

In this set of experiments we evaluate how well over method works outside of the domain randomization ranges (see Subsec.5.4.1.1), when noise in the ground truth labels are present or a reduced model is used. First, we test for extrapolation over unseen parameters by training on the first half of the range for different parameters - eg. $e_{train} \in [0.6, 0.8]$ (for restitution). We evaluate performance for increasing parameter deviations, $\Delta params$, in the test data - $e_{test} \in [0.8, 1.0]$. Secondly, we evaluate the performance when training with increasing additive parameter noise. Finally, we train with simpler physics model (fixed drag and rolling coefficient) and test how well we can estimate gravity and restitution when testing with the full model as the video is unrolled. Results can be seen in fig.5.7, with extrapolation performance slowly decaying outside of training regions and consistent performance even with large percentage of noise added to the training parameters.

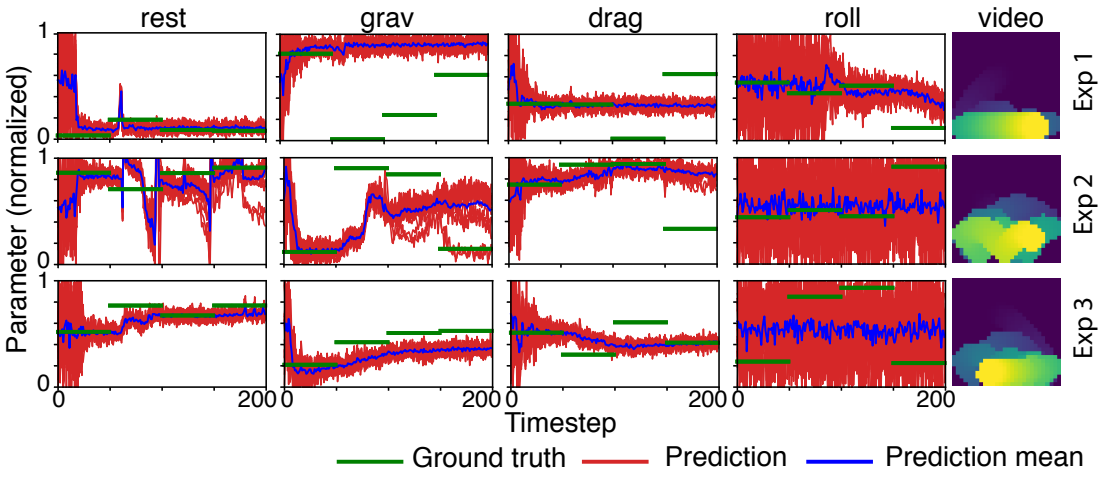


Figure 5.6: **System Identification from video with varying physical parameters.** The physical parameters of the bouncing ball change every 50 frames (4 times per video). We plot ground truth (green), predicted samples (red), and the predicted - mean (blue). Given enough excitation, our model can detect the change in the parameters.

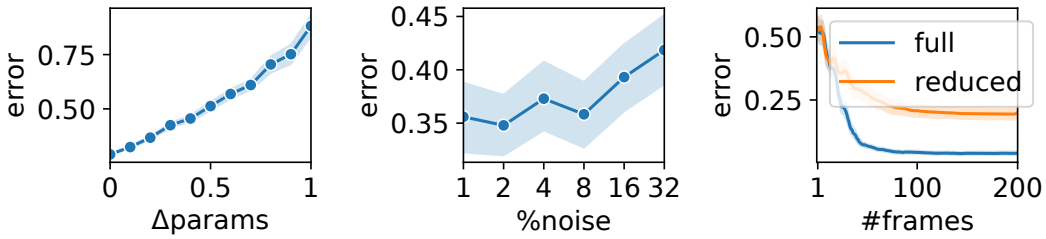
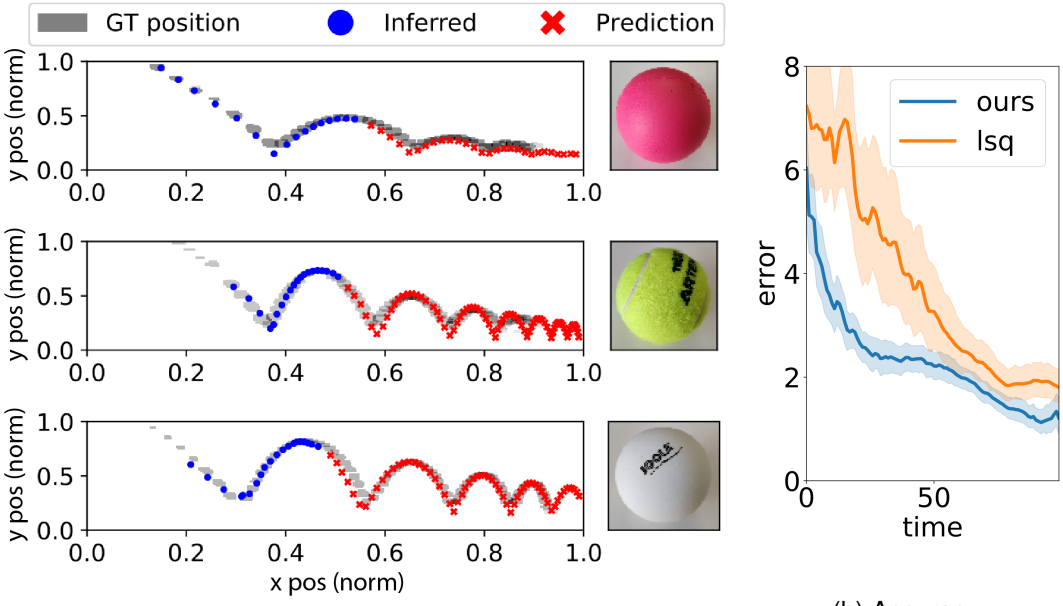


Figure 5.7: **Sensitivity analysis.** (Left) Extrapolation of parameters (Middle) Training with noisy data (Right) Training with less accurate model - fixed drag and rolling coefficient.

5.4.1.5 Real videos

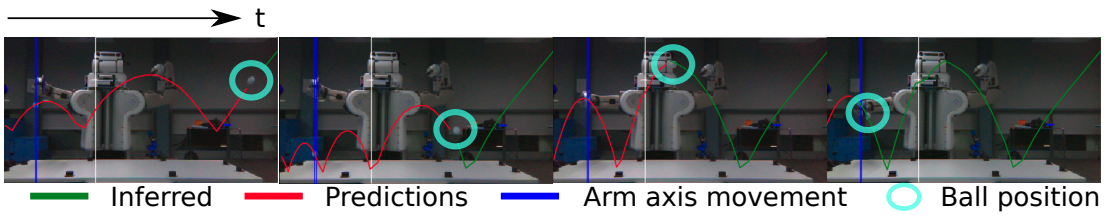
Here, we evaluate how well our method can scale to real videos. We record a set of videos, lasting between 1-5 seconds, of different types of bouncing balls - rubber, tennis and ping-pong balls. Since the exact physical properties of the balls are not available, we instead use the accuracy of the forward predictions as an evaluation metric. We compare the last ten positions of the ground truth position of the ball, with the forward predictions of a model as the video is unrolled. Here, we compare our method against [229], where we generate 5000 uniform samples of all physical parameters, horizontal and vertical velocities, horizontal and vertical position in a small region around the starting location of the ball - in order to account for some of the noise in the real videos. In fig.5.8 we show the convergence of our method for different types of balls, as well as the accuracy of the forward predictions as the video is observed.



(a) Convergence

(b) Accuracy

Figure 5.8: **Experiments with real videos.** We evaluate the performance of our method on real videos with bouncing balls with different physical properties (rubber, tennis, ping-pong). As the video is unrolled we compare the future predictions for the very last 10 frames of each video (the long term prediction accuracy) as explicit ground truth over the physical parameters is not available (a) Example model predictions for the three different types of balls, overlaid on the extracted positions from the images (b) Accuracy of the last 10 forward predictions as the video is observed. We denote [229] as 'lsq'.



Random Policy	8/35	(23%)
Random Policy (2x)	10/35	(29%)
Vid2Param	**27/35**	**(77%)**

Figure 5.9: **Experiments with PR2.** We perform a set of experiments, where a slowly actuated arm of the PR2 robot must intercept a bouncing ball, using only video as sensory input. We evaluate our method against a random policy feeding actions at 1Hz/2Hz. We demonstrate that our method can perform fast inference over the physical parameters of interest (e.g., restitution factor, height of the table) and correct future trajectories. Each experiment lasts about 2 seconds. At each throw of the ball, the model state is reset - so system identification is performed at every throw of the ball.

5.4.1.6 PR2 Robot experiments

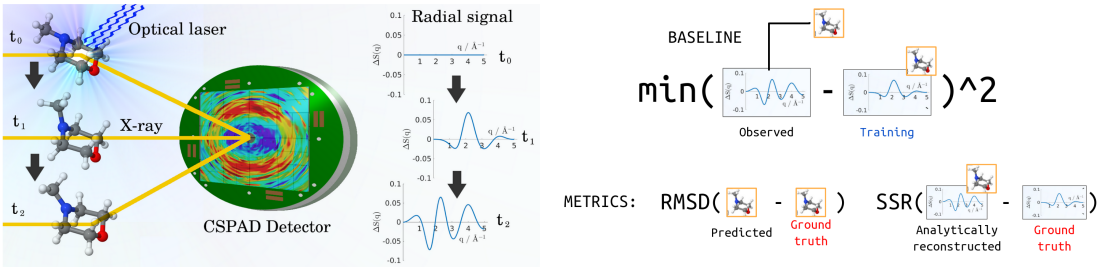
Finally, we evaluate the accuracy and speed of our method in an experiment where the PR2 robot uses its arm to intercept a bouncing ball from a visual feed, using a standard low-cost camera as sensory input. Firstly, the camera is calibrated with respect to the arm movements, so that predictions of the ball in the image, correspond to the same position of the gripper. No calibration with respect to the bouncing surface, position/velocity mappings, size of the ball, etc. are needed since these should be robustly dealt with by the model trained on randomized physics in simulation. The difference between two consecutive frames are fed directly into our model and the latent predictions are unrolled until the future predicted horizontal position is approximately the same as the horizontal position of the gripper of the arm. Then the generated vertical position of the ball is sent as a positional set point to the arm. An experimental run usually lasts for 2-3 seconds, during which the PR2 robot must infer the physics of the ball, predict its future trajectory and execute an action to intercept it. Our model runs at 20Hz on a standard laptop GPU, using IKFast for inverse kinematics of the arm. We use different types of ping-pong balls, in order to test how well our model can reach to balls with different physical properties. After each experiment (each throw of a ball), the model state is reset - so in each experiment we evaluate how well we can perform online system identification. Results can be seen in fig.5.9.

5.4.2 Molecular dynamics

In this set of experiments, we evaluate how well the proposed model can deal with inversion of complex molecular dynamics. Below, we describe the experimental setup, traditional baselines used within this. The subsection finishes with results and discussion over their significance.

5.4.2.1 Inversion with dynamics constraints

We are interested in recovering all atomic distances as a function of time, from a sequence of detector images. Our model allows the detector image at each time step, together with the dynamics carried by the recurrent part of the model, to be encoded into a latent space. Part of that latent space is simultaneously trained to represent the different distances in the given trajectory. As such, the model can account for time dependencies of the detector images and atomic distances, unlike traditional frame-by-frame methods, while also interpolating between the training examples.



(a) Experimental overview (b) Baseline and metrics

Figure 5.10: (5.10a) Overview of the experimental setup. We are interested in inferring molecular motion from a sequence of detector signals. (5.10b) Baseline and metrics. As a baseline we use the best frame fit from all training pairs. We use two metrics to evaluate the accuracy of the predicted atom-to-atom distances and the quality of the reconstructed signal.

An overview of the experimental data can be seen in fig.5.10a. We utilise simulated data to train, validate and test our model, which is done separately for each molecule. We use previously published full-dimensional quantum molecular dynamics (MD) simulations for three molecules – N-Methylmorpholine (NMM) [207], ethene (C_2H_4) [116], and carbon disulfide (CS_2) [25], which are are summarised in Table 5.11. When taken collectively, the full set of trajectories for a given molecule gives a statistical representation of the molecular wave packet. Here, we use them independently as if

they represent different modes of motion that the molecule may exhibit. Thus, our approach ignores phenomena such as wave packet width, dispersion and bifurcation. Nonetheless, the trajectories obey the physical principles of propagation (energy conservation, continuity, *etc.*) and, thus, encode the relevant physics of molecular motion. This is crucial because, as detailed in the introduction, the inversion problem is underdetermined – the simulated trajectories provide meaningful constraints on the type of motion. It should also be pointed out that the complexity of nuclear motion necessitates a case-by-case approach [148], which means that the network needs to be trained on the specific molecule under consideration, and the inversion of real-life data would only be as good as the simulated trajectories used to train the model.

We also use synthetic detector images calculated from these trajectories as follows.[2] To a first approximation, the electrons in the molecule can be considered to be bound to the atomic nuclei, the so called Independent Atom Model (IAM), in which case the elastic X-ray scattering recorded on the detector is proportional to the Debye scattering formula:

$$I(q,t) = \sum_A |f_A(q)|^2 + \sum_A \sum_{B \neq A} f_A(q) f_B(q) \frac{\sin(qR_{AB}(t))}{qR_{AB}(t)}, \qquad \Delta S(q,t) = \frac{I(q,t) - I(t,t_0)}{I(q,t_0)}. \tag{5.12}$$

In Eq. (5.12), the scattering intensity, $I(q,t)$, is function of the delay time between optical laser and the X-rays, t, and the magnitude of the scattering vector, q. $f_A(q)$ and $f_B(q)$ are tabulated quantities. $R_{AB}(t)$ denotes the interatomic distance between atoms A and B. In addition to the terms in Eq. (5.12), the intensity on the detector, $I(q,t)$, has a contribution from inelastic X-ray scattering, which to a first approximation is constant that does not evolve with time. To ensure that the approach is valid beyond the approximations implied in the IAM, our CS_2 data set uses a more advanced method to calculate the scattering signal directly from the *ab initio* electron density of the molecule [241].

As two of the molecules exhibit dissociation of the molecule structure, in two of the simulations we explicitly separate out this hard-to-tackle phenomenon. Therefore, we perform a set of five experiments – all NMM, all C_2H_4, all CS_2, and separately with only the dissociation-free C_2H_4 and CS_2 trajectories. We use a frame-by-frame χ^2 structural fitting as a baseline. More specifically, at each time step, we compute the sum of squared residuals (SSR) between the current detector image and all the reference

[2] Datasets, expanded results and source code can be seen at https://sites.google.com/view/mlscattering/

Molecule	NMM	C_2H_4	CS_2
Method	SHARC	AI-MCE [226]	SHARC [142]
Electronic states	X/3s/3p	S_0–S_2	S_0–S_3, T_1–T_4
Simulated time	1 ps	150 fs	1 ps
Time steps	201	151	201
# Train	85	800	753
# Val	11	100 (82)	94 (26)
# Test	11	100 (87)	95 (32)

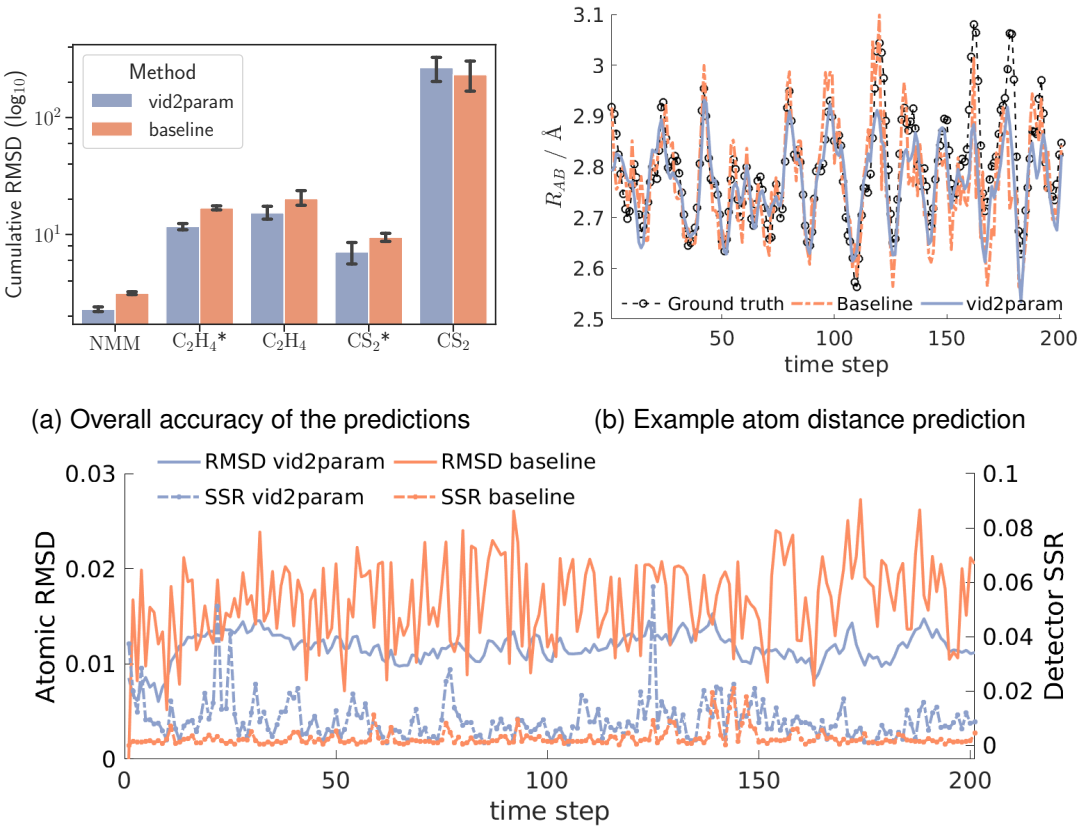
Figure 5.11: Quantum molecular dynamics simulations used to train, validate and test our model. The table reports only the number of intermediate time steps used in our model. The numbers in brackets are the trajectories that do not show dissociation. All three simulations use CASSCF *ab initio* level of theory.

detector images from the training data. We then assign the atom-atom distances from that reference detector image that has the lowest SSR.

5.4.2.2 Nuclear geometry recovery

We evaluate how well we can recover the true underlying molecular geometry from a sequence of detector images by using a mass-weighted root-mean-square deviation (RMSD) from the ground truth as seen in fig.5.10b. We observe that in all but one experiment the proposed method outperforms the baseline method (Figure 5.12a). Our model is able to *capture time dependencies and constraints associated with movements of atoms*. As exemplified in Figure 5.12c, the vid2param RMSD values for most individual time points are indeed below the baseline. Importantly, the vid2param exhibits much smoother behaviour compared to the large variations with time seen in the baseline. This is a manifestation of the fact that different time points are correlated. The smooth changes in the RMSD imply that the evolution of the molecular structure is also smooth, as expected physically.

Beyond the RMSD metric, the quality of the inversion of individual atom-atom distances depends on two factors. Physically, X-ray scattering is dominated by atoms with many electrons, hence, the detector signal has vanishing contributions from light atoms such as Hydrogen. Accurate estimates of the distances between such atoms are not possible even with our model. However, we noticed that vid2param tends to infer the correct oscillation periods, presumably, because of the correlation with the



(a) Overall accuracy of the predictions

(b) Example atom distance prediction

(c) Atom distance prediction vs signal reconstruction

Figure 5.12: (5.12a) Sum of atomic RMSD values for all time steps and all trajectories in the test part of the five data sets. The asterisk (*) denotes the data sets where trajectories with dissociation has been filtered out. (5.12b) O−N distance in a single NMM trajectory. (5.12c) RMSD and sum of squared residuals (SSR) as a function of time step for the baseline and vid2param methods.

motion of heavier atoms. The detector signal also has almost no contribution from atoms that are far away (see Eq. 5.12). While sometimes correlated motion can ensure that these distances are still adequately inferred, the process of dissociation possesses a challenge because coherence between the fragments is lost quickly. We believe this is the reason why our experiment with CS_2 where dissociation occurs fails against the baseline method. In the case of relatively heavy atoms and no dissociation, both vid2param and the baseline method perform quite well as illustrated on Figure 5.12b.

5.4.2.3 Analytic reconstruction of detector images

In this set of experiments, we evaluate how well the detector image can be *analytically* reconstructed from the estimated set of atomic distances using Eq. (5.12). This gives us a second metric which is not based directly on the molecular geometry but on its scattering fingerprint. A typical SSR between the ground truth and the reconstructed detector is illustrated also on Figure 5.12c. It is important to note that we try to reconstruct only the *visible* part of the detector – the part of the signal that is not detected, i.e. at large values of q, still contains vital information.

We observe that the baseline outperforms the proposed model as seen in Figure 5.12c, which at first may seem counter-intuitive. However, given the limited information in the detector image, even an exact match of the visible part of the signal at any given time does not guarantee a perfect estimate for the molecular structure via the atomic separations. Unlike the baseline method, vid2param learns atomic distances and *motion* constraints from previous time points, hence, producing a better estimate for the molecular structure without producing a perfect fit for the visible part of the detector image. This again suggests that because of the limited information in a single detector image, it is useful to incorporate time dependencies in the process of inferring the atomic distances, to carry out constraints and information from previous time steps.

5.5 Results and Analysis

Unified model for physical reasoning. In this work we present a model for inference of physical parameters and generation of plausible future states from videos. We observe that such a model can be trained in an end-to-end fashion and perform accurate system identification in simulated and real settings, and subsequently used for control. We constrain our experiments to a single object and show that using just a video stream we

can perform *online* system identification. Importantly, the proposed approach is able to generalise well to images captured from a real camera, despite only being trained on simulated data. This highlights the value of sim2real techniques for interpreting physical parameters in various applications, and its potential to enable reasoning about physical properties, from relatively low fidelity sensors bootstrapped by learning from simulation.

System identification. We observe that the proposed method can accurately infer different physical parameters, outperforming baselines from the literature. The magnitude of gravity and air drag can usually be inferred from observing just a few frames. Air drag can usually be inferred after observing a few more frames, as it is a function of both horizontal and vertical velocity, rather than just the vertical velocity as in the case of gravity. Restitution factors can be inferred a few frames after the ball has bounced for the first time. The rolling coefficient has a higher error, which starts to decrease towards the end of the videos. While traditional system identification methods can infer physical properties, which have a clear effect on the trajectory (such as restitution), it is challenging to infer properties that jointly contribute to a certain effect. By using domain randomization and a sim2real approach, our method can learn the difference in parameters of trajectories with similar appearance even when trained with noisy data. As such we can accurately estimate parameters with similar effect on the dynamics (such as gravity and air drag), as well as parameters whose effects are not observed until the end of the trajectory (such as rolling coefficients). Moreover, we also demonstrate that to an extent detect change in the parameters, as a video is unrolled (although this requires system excitation), and extrapolate to unseen parameters.

Forward predictions. We have shown that our model can perform forward predictions in the latent space, over parameters of interest such as physical state variables. The forward predictions bring out key aspects of the evolution of uncertainty, such as high variability before a bounce and lower variability soon after, high variability over the stopping point before rolling is observed, etc. The proposed approach outperforms both parametric and non-parametric baselines in its ability to accurately perform forward predictions.

Molecular dynamics. We address the issue of how to unravel the evolution of molecular structure given a time-series of detector images generated by a novel experimental technique called Ultrafast X-ray Scattering (UXS). While still in its early days, UXS has shown enormous capacity to elucidate fundamental questions in chemistry such as how molecules move, how bonds between atoms are broken or formed, and

even how electrons may change their position inside molecules. Nevertheless, there are still big gaps in our ability to analyse and interpret such experiments. We believe that the method presented here addresses a major limitation in the current way of extracting information from UXS data, thus improving our ability to unravel chemical dynamics.

Limitations and future work. We observe in robotics experiments that our model performs well in real settings. Nevertheless, we experienced some limitations arising from making the predictions based on a single image, e.g., the ball passing behind or in front of the gripper. Thus in the future it will be beneficial to extend this line of work by inferring future predictions from multiple sources of video stream from different locations, or incorporating depth sensing. We note that our proposed model is independent of the choice of encoder, decoder and training data. It would be of interest to explore how such a model would perform with richer training data (e.g., multiple objects) by using advanced domain randomization [236] or different encoder-decoder structure [22].

5.6 Conclusions

This chapter looks into a method for learning inverse mappings between rich sensory observations and parameters of interest. For the purpose an additional loss to a variational recurrent neural network is proposed, forcing consistency between simulator of interest and the latent space of the NN model. The proposed method outperforms existing baselines from literature, both in terms of speed and accuracy of the identification, and forward predictions as a consequence of that. To showcase the utility of the proposed approach, we evaluate on two challenging tasks. First, we address the task of stopping an unknown bouncing ball with a robot arm, performing online identification from a camera feed and using the proposed model for inference of the physics parameters. Further, we show its ability to generate future predictions of the ball position, laying the groundwork for much more sophisticated predictive motion planning schemes. Secondly, we evalulate the proposed method on a challenging problem in molecular dynamics. We demonstrate that the proposed method outperform standard frame-by-frame baselines used in literature, and is able to capture relevant dynamics required for an accurate inversion, showing the utility of the proposed beyond robotics.

Chapter 6

Future Work & Conclusion

6.1 Key Ideas

The dissertation aims to connect and extend ideas from SBI to the specific needs of *online* robot motion planning. A key requirement that this thesis addresses is that of strict computational limitations imposed by the need to act in dynamic and uncertain environments. As such the inference method must be fast enough for the given task, while the underlying simulation model expressive enough to capture the observed variations in the dynamics. This work looks into different problems with varying complexity of the simulation model or observations used - gas localization with a UAV using a fluid simulation (Chapter 3), safe medical suction (Chapter 4) and control from rich sensory observation such as videos (Chapter 5). This dissertation looks into two central approaches for solving inverse problems - sampling reduced simulation representations (Chapter 3 & 4) and learning inverse mapping directly from raw observations (Chapter 5).

Sampling reduced representations

The first two chapters of this dissertation look into reduced simulation representations for SBI. Those reduced representations are motivated by specific observations about the dynamics of the phenomena and *factors of variation*. In Chapter 3 the motivating problem of gas localization is examined. This work focuses on improving the efficiency of the sampling process, by exploiting domain specific knowledge about the observed dynamics - e.g. short-distance spatial invariance of wind speed and direction. To further strengten the analysis of dominant factors of variations, techniques such as sensitivity analysis can also be employed (Fig. 3.5). By focusing on the leading causes of difference in the observed data, we can devise an expressive model for the

task, while employing efficient sampling techniques. This allows for online inference in highly dynamic tasks and long term planning under uncertainty. In Chapter 4, this thesis builds up on those ideas in the context of medical suction. In addition to reducing the complexity of the simulation to fascilitate more efficient SBI, this work also studies how this can be incorporated into a planning procedure, taking uncertainty into account. This results in a safe and efficient completition of the task compared to naive and myopic solutions.

Learning inverse relations

Coming up with the right simplified simulation model to enable online inference might be challenging, and is domain specific. In the last chapter of the thesis we develop a method for direct regression to parameters of interest from complex observations. The inference process is *amortized* by structing the latent space of variational recurrent NN in accordance to a given simulator (Fig. 5.2). This allows for online inference from long sequence of high dimensional observations. The improved speed and accuracy of the proposed approach is verified in two challenging domains - robot control from video observations and inversion of molecular dynamics. This testifies for the broad applicability of the proposed method in robotics applications and beyond.

6.2 Future Work & Open Questions

This section discusses open questions and future research directions. This thesis explores various robotics domains, their respective complexities and challenges for applying SBI techniques. While this work showcases the promise of following this methodology, challenges remain. First of all, many of presented practical robotic examples, and the respective simulations used, remain limited. Making simulators fully differentiable is one area of research, that has the promise of better scalability and integration with modern machine learning methods. Learning them from raw data is another avenue of research, that can similarly address some of those challanges. From a robotics context, there are many more constrained task that would benefit from the here presented methodology. Those directions of research are discussed in order below.

Fully differentiable open worlds

Imposing the right structure and biases within neural networks have facilitated efficient learning of ever more complicated tasks. This has ranged from imposing cascades of filters applied across patches of images for classification [125] and efficiently propagating long-term information [89], to scaling attention mechanism for machine

translation [224]. While a lot of NN modules assume almost no prior knowledge of the training data, often there is a lot of *known* dynamics and physics that we understand and model. By making this existing knowledge in the form of differentiable programs, they can readily be included within the wide existing end-to-end differentiable models. Many examples of modelling different dynamics aspects have already been proposed including fluids and contacts [197] [60] [157], allowing for solving challenging inference tasks in an unsupervised way [101]. There is still, however, a gap between the complexity of tasks that can be constructed using those differentiable models, that has so far not matched the performance of supervised methods paired with vasts amount of data. Given the usual complexity of acquiring that amount of data within robot platforms, constructing expressive fully differentiable open world environments can be an important ingridient for solving hard control tasks.

Learning physics engines from raw data

SBI methods are of course relying on the existance of a simulation. While models of different aspects of dynamics such as contact, fluids, rigid bodies, etc. have been proposed, almost all of them tend to be *approximate* in their nature, unable to capture the full complexity of the real world. While often challenging, most of the robotics systems have the luxury to be able to observe and interact with their environment. As such it is possible in principle to build internal models of those observations, that can subsequently be used for control. A common way to approach this problem is to use as a model what is *known* and learn the *residual* unknown part of the observed process [232]. New developments in sensing [75] [179] can make this process significantly easier, allowing for readily available state representations and safe learning [234] - a concept explored within cognitive science [59].

Constrained, but dynamically and visually rich tasks

Modern robot arms surpass human level performance of precision and load-carrying capacity. Understanding the dynamics and visual variations of the environment on the other hand is still a major limiting factor for wider deployment of robot systems. The above-mentioned developments of simulations and data gathering can unlock many new applications, where there is a strong business case, and the opportunity to collect real-world labelled data. Learning *robust* policies, will mostly likely require *some knowledge* of the environment, but not it's exact state. While there is no consencus on the best way to approach open-ended tasks (e.g., driving in a city), current approaches can potentially handle constrained versions of the same problems (e.g., driving on a highway with certified marking). Many fields can benefit of automation of constrained

environments, but with variations in the dynamics and visuals - handling diverse objects within factories, non-invasive medical tasks, automating and scaling bench experiments.

6.3 Concluding remarks

In conclusion, we addressed the problem planning in dynamic environments, where the exact state or parameters of the modeled phenomena might not be known. In this thesis we propose multiple strategies of employing a simulation as a model to perform online SBI and solve the given task.

Bibliography

[1] Pr2 robot / willow garage. https://www.joshuaellingson.com/willow/. Accessed: 2021-10-22.

[2] Watch pr2 robot 'learn' how to smartly position an unstable object. https://www.popsci.com/technology/article/2013-02/watch-robot-figure-out-how-move-object/. Accessed: 2021-10-22.

[3] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 2021.

[4] A. Achille and S. Soatto. A separation principle for control in the age of deep learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:287–307, 2018.

[5] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in Neural Information Processing Systems*, pages 5074–5082, 2016.

[6] A. Ajay, M. Bauza, J. Wu, N. Fazeli, J. B. Tenenbaum, A. Rodriguez, and L. P. Kaelbling. Combining physical simulators and object-based networks for control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3217–3223. IEEE, 2019.

[7] A. Ajay, J. Wu, N. Fazeli, M. Bauza, L. P. Kaelbling, J. B. Tenenbaum, and A. Rodriguez. Augmenting physical simulators with stochastic neural networks: Case study of planar pushing and bouncing. In *IROS*, pages 3066–3073. IEEE, 2018.

[8] D. Angelov, Y. Hristov, and S. Ramamoorthy. Using causal analysis to learn specifications from task demonstrations. AAMAS '19, pages 1341–1349, 2019.

[9] I. Arnekvist, D. Kragic, and J. A. Stork. Vpe: Variational policy embedding for transfer reinforcement learning. *ICRA*, 2019.

[10] S. Asadi, S. Pashami, A. Loutfi, and A. J. Lilienthal. Td kernel dm+ v: Time-dependent statistical gas distribution modelling on simulated measurements. In *AIP Conference Proceedings*, volume 1362, pages 281–282. AIP, 2011.

[11] M. Asenov, M. Burke, D. Angelov, T. Davchev, K. Subr, and S. Ramamoorthy. Vid2param: Modeling of dynamics parameters from video. *IEEE Robotics and Automation Letters*, 5(2):414–421, 2019.

[12] M. Asenov, M. Rutkauskas, D. Reid, K. Subr, and S. Ramamoorthy. Active localization of gas leaks using fluid simulation. *IEEE Robotics and Automation Letters*, 4(2):1776–1783, 2019.

[13] M. Asenov, N. Zotev, S. Ramamoorthy, and A. Kirrander. Inversion of ultrafast x-ray scattering with dynamics constraints. In *Machine Learning and the Physical Sciences: Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[14] K. Astrom. Maximum likelihood and prediction error methods. *IFAC Proceedings Volumes*, 12(8):551–574, 1979.

[15] K. J. Åström and P. Eykhoff. System identification—a survey. *Automatica*, 7(2):123–162, 1971.

[16] K.-J. Åström and B. Torsten. Numerical identification of linear dynamic systems from normal operating records. *IFAC Proceedings Volumes*, 2(2):96–111, 1965.

[17] B. Axelrod, L. P. Kaelbling, and T. Lozano-Pérez. Provably safe robot navigation with obstacle uncertainty. *The International Journal of Robotics Research*, 37(13-14):1760–1774, 2018.

[18] J. Banks. *Discrete event system simulation*. Pearson Education India, 2005.

[19] D. Baraff. Physically based modeling: Rigid body simulation. *SIGGRAPH Course Notes, ACM SIGGRAPH*, 2(1):2–1, 2001.

[20] R. Barzel and A. H. Barr. A modeling system based on dynamic constraints. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 179–188, 1988.

[21] C. Bates, P. Battaglia, I. Yildirim, and J. B. Tenenbaum. Humans predict liquid dynamics using probabilistic simulation. In *CogSci*, 2015.

[22] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pages 4502–4510, 2016.

[23] P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013.

[24] M. Bauza and A. Rodriguez. A probabilistic data-driven model for planar pushing. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3008–3015. IEEE, 2017.

[25] D. Bellshaw, D. A. Horke, A. D. Smith, H. M. Watts, E. Jager, E. Springate, O. Alexander, C. Cacho, R. T. Chapman, A. Kirrander, and R. S. Minns. Ab-initio surface hopping and multiphoton ionisation study of the photodissociation dynamics of CS2. *Chemical Physics Letters*, 683:383–388, 2017.

[26] F. Berkenkamp, A. P. Schoellig, and A. Krause. Safe controller optimization for quadrotors with gaussian processes. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 491–496. IEEE, 2016.

[27] C. M. Bishop. Model-based machine learning. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 371(1984), 2013.

[28] J. L. Blanco, J. G. Monroy, A. Lilienthal, and J. Gonzalez-Jimenez. A kalman filter based approach to probabilistic gas distribution mapping. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 217–222. ACM, 2013.

[29] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[30] J. Bongard and H. Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.

[31] A. Bordallo, F. Previtali, N. Nardelli, and S. Ramamoorthy. Counterfactual reasoning about intent for interactive navigation in dynamic environments. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 2943–2950. IEEE, 2015.

[32] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4243–4250. IEEE, 2018.

[33] R. Brincker, L. Zhang, and P. Andersen. Modal identification of output-only systems using frequency domain decomposition. *Smart materials and structures*, 10(3):441, 2001.

[34] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[35] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[36] G. Camporredondo, R. Barber, M. Legrand, and L. Muñoz. A kinematic controller for liquid pouring between vessels modelled with smoothed particle hydrodynamics. *Applied Sciences*, 9(23):5007, 2019.

[37] L. Cardelli, M. Kwiatkowska, L. Laurenti, and A. Patane. Robustness guarantees for bayesian inference with gaussian processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7759–7768, 2019.

[38] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová. Machine learning and the physical sciences. *Reviews of Modern Physics*, 91(4):045002, 2019.

[39] M. Carlson, P. J. Mucha, R. B. Van Horn III, and G. Turk. Melting and flowing. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 167–174, 2002.

[40] M. B. Chang, T. Ullman, A. Torralba, and J. B. Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.

[41] K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, and J.-B. Mouret. A survey on policy search algorithms for learning robot controllers in a handful of trials. *ArXiv e-prints*, page arXiv:1807.02303, July 2018.

[42] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. *ICRA*, 2019.

[43] F. Chen, H. Obermaier, H. Hagen, B. Hamann, J. Tierny, and V. Pascucci. Topology analysis of time-dependent multi-fluid data using the reeb graph. *Computer Aided Geometric Design*, 30(6):557–566, 2013.

[44] S. Chen, S. Billings, and P. Grant. Non-linear system identification using neural networks. *International journal of control*, 51(6):1191–1214, 1990.

[45] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NeurIPS*, pages 2172–2180, 2016.

[46] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. In *NeurIPS*, pages 2980–2988, 2015.

[47] J. Collins, S. Chand, A. Vanderkop, and D. Howard. A review of physics simulators for robotic applications. *IEEE Access*, 9:51416–51431, 2021.

[48] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. 2016.

[49] K. Cranmer, J. Brehmer, and G. Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020.

[50] I. G. Damousis, M. C. Alexiadis, J. B. Theocharis, and P. S. Dokopoulos. A fuzzy model for wind speed prediction and power generation in wind parks using spatial correlation. *IEEE Transactions on Energy Conversion*, 19(2):352–361, 2004.

[51] J. Degrave, M. Hermans, J. Dambre, et al. A differentiable physics engine for deep learning in robotics. *Frontiers in neurorobotics*, 13, 2019.

[52] J. R. Demers, F. Garet, and J.-L. Coutaz. Atmospheric water vapor absorption recorded ten meters above the ground with a drone mounted frequency domain thz spectrometer. *IEEE sensors letters*, 1(3):1–3, 2017.

[53] N. Demo, M. Tezzele, and G. Rozza. PyDMD: Python Dynamic Mode Decomposition. *The Journal of Open Source Software*, 3(22):530, 2018.

[54] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[55] T. Deng, H. Wang, W. Chen, X. Wang, and R. Pfeifer. Development of a new cable-driven soft robot for cardiac ablation. In *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 728–733. IEEE, 2013.

[56] T. K. Dey. *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis (Cambridge Monographs on Applied and Computational Mathematics)*. Cambridge University Press, NY, USA, 2006.

[57] R. Diankov and J. Kuffner. Openrave: A planning architecture for autonomous robotics. *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, 79, 2008.

[58] C. Do, C. Gordillo, and W. Burgard. Learning to pour using deep deterministic policy gradients. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3074–3079. IEEE, 2018.

[59] N. Doidge. *The brain that changes itself: Stories of personal triumph from the frontiers of brain science*. Penguin, 2007.

[60] T. Du, K. Wu, P. Ma, S. Wah, A. Spielberg, D. Rus, and W. Matusik. Diffpd: Differentiable projective dynamics with contact. *arXiv preprint arXiv:2101.05917*, 2021.

[61] N. Durasov, T. Bagautdinov, P. Baque, and P. Fua. Masksembles for uncertainty estimation. *arXiv preprint arXiv:2012.08334*, 2020.

[62] F. Ebert, C. Finn, A. X. Lee, and S. Levine. Self-supervised visual planning with temporal skip connections. *arXiv preprint arXiv:1710.05268*, 2017.

[63] T. Erez, Y. Tassa, and E. Todorov. Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 4397–4404. IEEE, 2015.

[64] N. B. Erichson, M. Muehlebach, and M. W. Mahoney. Physics-informed autoencoders for lyapunov-stable fluid flow prediction. *Machine Learning for Physical Sciences Workshop, Advances in Neural Information Processing Systems*, 2019.

[65] Z. Erickson, H. M. Clever, G. Turk, C. K. Liu, and C. C. Kemp. Deep haptic model predictive control for robot-assisted dressing. In *ICRA*, pages 1–8. IEEE, 2018.

[66] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton, et al. Attend, infer, repeat: Fast scene understanding with generative models. In *NeurIPS*, pages 3225–3233, 2016.

[67] S. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[68] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[69] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. *arXiv preprint arXiv:1709.04905*, 2017.

[70] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *NeurIPS*, pages 3601–3610, 2017.

[71] K. Fragkiadaki, P. Agrawal, S. Levine, and J. Malik. Learning visual predictive models of physics for playing billiards. *arXiv preprint arXiv:1511.07404*, 2015.

[72] G. Franzese, A. Mészáros, L. Peternel, and J. Kober. Ilosa: Interactive learning of stiffness and attractors. *arXiv preprint arXiv:2103.03099*, 2021.

[73] K. M. Frey, T. J. Steiner, and J. P. How. Collision probabilities for continuous-time systems without sampling. *Proceedings of Robotics: Science and Systems. Corvalis, Oregon, USA (July 2020)*, 2020.

[74] N. Furukawa, A. Namiki, S. Taku, and M. Ishikawa. Dynamic regrasping using a high-speed multifingered hand and a high-speed vision system. In *ICRA*, pages 181–187. IEEE, 2006.

[75] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, et al. Event-based vision: A survey. *arXiv preprint arXiv:1904.08405*, 2019.

[76] W. Garage. The PR2 Plays Pool. `http://www.willowgarage.com/blog/2010/06/15/pr2-plays-pool`, 2010. [Online; accessed 9-July-2018].

[77] M. Garnelo, D. Rosenbaum, C. J. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. Whye Teh, D. J. Rezende, and S. M. A. Eslami. Conditional Neural Processes. *ArXiv e-prints*, page arXiv:1807.01613, July 2018.

[78] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. M. A. Eslami, and Y. Whye Teh. Neural Processes. *ArXiv e-prints*, page arXiv:1807.01622, July 2018.

[79] T. Gerstenberg, N. Goodman, D. Lagnado, and J. Tenenbaum. Noisy newtons: Unifying process and dependency accounts of causal attribution. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 34, 2012.

[80] M. T. Gettman, M. L. Blute, G. K. Chow, R. Neururer, G. Bartsch, and R. Peschel. Robotic-assisted laparoscopic partial nephrectomy: technique and initial clinical experience with davinci robotic system. *Urology*, 64(5):914–918, 2004.

[81] M. Ghallab, D. Nau, and P. Traverso. *Automated Planning and Acting*. Cambridge University Press, 2016.

[82] E. L. Glaeser and M. E. Kahn. The greenness of cities: carbon dioxide emissions and urban development. *Journal of urban economics*, 67(3):404–418, 2010.

[83] J. González. Gpyopt: A bayesian optimization framework in python, 2016.

[84] T. L. Guevara, R. Pucci, N. K. Taylor, M. Gutmann, S. Ramamoorthy, and K. Subr. To stir or not to stir: Online estimation of liquid properties for pouring

actions. In *Robotics: Science and Systems Workshop on Learning and Inference in Robotics: Integrating Structure, Priors and Models*, 2018.

[85] H. Hastie, K. Lohan, M. Chantler, D. A. Robb, S. Ramamoorthy, R. Petrick, S. Vijayakumar, and D. Lane. The orca hub: Explainable offshore robotics through intelligent interfaces. *arXiv preprint arXiv:1803.02100*, 2018.

[86] J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*, 2013.

[87] D. Higdon, J. Gattiker, B. Williams, and M. Rightley. Computer model calibration using high-dimensional output. *Journal of the American Statistical Association*, 103(482):570–583, 2008.

[88] B. Ho and R. E. Kálmán. Effective construction of linear state-variable models from input/output functions. *at-Automatisierungstechnik*, 14(1-12):545–548, 1966.

[89] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[90] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[91] Y. Hristov, A. Lascarides, and S. Ramamoorthy. Interpretable latent spaces for learning from demonstration. In *CoRL*, volume 87, pages 957–968. PMLR, 29–31 Oct 2018.

[92] J.-T. Hsieh, B. Liu, D.-A. Huang, L. F. Fei-Fei, and J. C. Niebles. Learning to decompose and disentangle representations for video prediction. In *NeurIPS*, pages 515–524, 2018.

[93] X. Huan and Y. M. Marzouk. Sequential bayesian optimal experimental design via approximate dynamic programming. *arXiv preprint arXiv:1604.08320*, 2016.

[94] J. Huang, F. Liu, F. Richter, and M. C. Yip. Model-predictive control of blood suction for surgical hemostasis using differentiable fluid simulations. *arXiv preprint arXiv:2102.01436*, 2021.

[95] M. N. Huda, H. Yu, and S. Cang. Robots for minimally invasive diagnosis and intervention. *Robotics and Computer-Integrated Manufacturing*, 41:127–144, 2016.

[96] M. Hutchinson, C. Liu, and W.-H. Chen. Information-based search for an atmospheric release using a mobile robot: Algorithm and experiments. *IEEE Transactions on Control Systems Technology*, 2018.

[97] M. Hutchinson, C. Liu, and W.-H. Chen. Source term estimation of a hazardous airborne release using an unmanned aerial vehicle. *Journal of Field Robotics*, 2018.

[98] M. Hutchinson, H. Oh, and W.-H. Chen. A review of source term estimation methods for atmospheric dispersion events using static or mobile sensors. *Information Fusion*, 36:130–148, 2017.

[99] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NeurIPS*, pages 2017–2025, 2015.

[100] M. Jaques, M. Asenov, M. Burke, and T. Hospedales. Vision-based system identification and 3d keypoint discovery using dynamics constraints. *arXiv preprint arXiv:2109.05928*, 2021.

[101] M. Jaques, M. Burke, and T. Hospedales. Physics-as-inverse-graphics: Joint unsupervised learning of objects and physics from video. *arXiv preprint arXiv:1905.11169*, 2019.

[102] T. A. Johansen and B. A. Foss. Identification of non-linear system structure and parameters using regime decomposition. *Automatica*, 31(2):321–326, 1995.

[103] R. Jonschkowski, D. Rastogi, and O. Brock. Differentiable particle filters: End-to-end learning with algorithmic priors. *arXiv preprint arXiv:1805.11122*, 2018.

[104] P. B. Jónsson, J. Wang, and J. Kim. Scalar field reconstruction based on the gaussian process and adaptive sampling. In *Ubiquitous Robots and Ambient Intelligence (URAI), 2017 14th International Conference on*, pages 442–445. IEEE, 2017.

[105] Y. Joshi and P. Kumar. *Energy efficient thermal management of data centers*. Springer Science & Business Media, 2012.

[106] V. Joukov and D. Kulic. Gaussian process based model predictive controller for imitation learning. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 850–855. IEEE, 2017.

[107] V. Joukov and D. Kulić. Fast approximate multi-output gaussian processes. *arXiv preprint arXiv:2008.09848*, 2020.

[108] J.-N. Juang and R. S. Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *J GUID CONTROL DYNAM*, 8(5):620–627, 1985.

[109] A. Kapur, A. Kapur, N. Virji-Babul, G. Tzanetakis, and P. F. Driessen. Gesture-based affective computing on motion capture data. In *International conference on affective computing and intelligent interaction*, pages 1–7. Springer, 2005.

[110] M. Karl, M. Soelch, J. Bayer, and P. van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432*, 2016.

[111] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? *arXiv preprint arXiv:1703.04977*, 2017.

[112] M. Kennedy, K. Schmeckpeper, D. Thakur, C. Jiang, V. Kumar, and K. Daniilidis. Autonomous precision pouring from unknown containers. *IEEE Robotics and Automation Letters*, 4(3):2317–2324, 2019.

[113] M. C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

[114] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, and B. Solenthaler. Deep fluids: A generative network for parameterized fluid simulations. *arXiv preprint arXiv:1806.02071*, 2018.

[115] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[116] A. Kirrander, K. Saita, and D. V. Shalashilin. Ultrafast X-ray Scattering from Molecules. *Journal of Chemical Theory and Computation*, 12(3):957–967, 2016.

[117] N. P. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IROS*, volume 4, pages 2149–2154. Citeseer, 2004.

[118] S. Kolev and E. Todorov. Physically consistent state estimation and system identification for contacts. In *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*, pages 1036–1043. IEEE, 2015.

[119] G. Kowadlo and R. A. Russell. Naïve physics for effective odour localisation. In *Proceedings of the Australian Conference on Robotics and Automation*, 2003.

[120] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[121] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[122] F. Laine, C.-Y. Chiu, and C. Tomlin. Eyes-closed safety kernels: Safety for autonomous systems under loss of observability. *arXiv preprint arXiv:2005.07144*, 2020.

[123] F. Lanusse, P. Melchior, and F. Moolekamp. Hybrid physical-deep learning model for astronomical inverse problems. *Machine Learning for Physical Sciences Workshop, Advances in Neural Information Processing Systems*, 2019.

[124] J. Le Ny and G. J. Pappas. On trajectory optimization for active sensing in gaussian process models. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*. IEEE, 2009.

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

[126] A. X. Lee, R. Zhang, F. Ebert, P. Abbeel, C. Finn, and S. Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

[127] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

[128] J.-G. Li, Q.-H. Meng, Y. Wang, and M. Zeng. Odor source localization using a mobile robot in outdoor airflow environments with a particle filter algorithm. *Autonomous Robots*, 30(3):281–292, 2011.

[129] Y. Li, H.-L. Wei, S. A. Billings, and P. G. Sarrigiannis. Identification of nonlinear time-varying systems using an online sliding-window and common model structure selection (cmss) approach with applications to eeg. *International Journal of Systems Science*, 47(11):2671–2681, 2016.

[130] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018.

[131] A. Lilienthal, F. Streichert, and A. Zell. Model-based shape analysis of gas concentration gridmaps for improved gas source localisation. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 3564–3569. IEEE, 2005.

[132] A. J. Lilienthal, M. Reggente, M. Trincavelli, J. L. Blanco, and J. Gonzalez. A statistical approach to gas distribution modelling with mobile robots-the kernel dm+ v algorithm. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009.

[133] L. Ljung. System identification. *Wiley Encyclopedia of Electrical and Electronics Engineering*, 2001.

[134] L. Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010.

[135] T. Lopez-Guevara, N. K. Taylor, M. U. Gutmann, S. Ramamoorthy, and K. Subr. Adaptable pouring: Teaching robots not to spill using fast but approximate fluid simulation. In *Conference on Robot Learning*, 2017.

[136] T. Lopez-Guevara, N. K. Taylor, M. U. Gutmann, S. Ramamoorthy, and K. Subr. Adaptable pouring: Teaching robots not to spill using fast but approximate fluid simulation. In *CoRL*, pages 77–86, 2017.

[137] A. Lowe, P. K. Agarwal, and M. Rav. Flood-risk analysis on terrains. *Communications of the ACM*, 63(9):94–102, 2020.

[138] T. Ma, Z. Wu, P. Luo, and L. Feng. Reeb graph computation through spectral clustering. *Optical Engineering*, 51(1):017209, 2012.

[139] Z. Ma, S. Ahuja, and C. W. Rowley. Reduced-order models for control of fluids using the eigensystem realization algorithm. *THEOR COMP FLUID DYN*, 25(1-4):233–247, 2011.

[140] M. Macklin and M. Müller. Position based fluids. *ACM Transactions on Graphics (TOG)*, 32(4):104, 2013.

[141] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33(4):153, 2014.

[142] S. Mai, P. Marquetand, and L. González. Nonadiabatic dynamics: The SHARC approach. *WIREs Computational Molecular Science*, 8(6):e1370, 2018.

[143] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

[144] A. Marco, F. Berkenkamp, P. Hennig, A. P. Schoellig, A. Krause, S. Schaal, and S. Trimpe. Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with bayesian optimization. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1557–1563. IEEE, 2017.

[145] P. Marjoram, J. Molitor, V. Plagnol, and S. Tavaré. Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328, 2003.

[146] M. T. Mason. *Mechanics of robotic manipulation*. MIT press, 2001.

[147] J. Matas, S. James, and A. J. Davison. Sim-to-real reinforcement learning for deformable object manipulation. *arXiv preprint arXiv:1806.07851*, 2018.

[148] R. J. D. Miller. Ultrafast imaging of photochemical dynamics: roadmap to a new conceptual basis for chemistry. *Faraday Discussions*, 194:777–828, 2016.

[149] B. Mirtich and J. Canny. Impulse-based simulation of rigid bodies. In *Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 181–ff, 1995.

[150] J. Monroy, V. Hernandez-Bennetts, H. Fan, A. Lilienthal, and J. Gonzalez-Jimenez. Gaden: A 3d gas dispersion simulator for mobile robot olfaction in realistic environments. *Sensors*, 17(7):1479, 2017.

[151] J. G. Monroy, J.-L. Blanco, and J. Gonzalez-Jimenez. Time-variant gas distribution mapping with obstacle information. *Autonomous Robots*, 40(1):1–16, 2016.

[152] L. Moriello, L. Biagiotti, C. Melchiorri, and A. Paoli. Control of liquid handling robotic systems: A feed-forward approach to suppress sloshing. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4286–4291. IEEE, 2017.

[153] R. Mottaghi, H. Bagherinezhad, M. Rastegari, and A. Farhadi. Newtonian scene understanding: Unfolding the dynamics of objects in static images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3521–3529, 2016.

[154] D. Mrowca, C. Zhuang, E. Wang, N. Haber, L. F. Fei-Fei, J. Tenenbaum, and D. L. Yamins. Flexible neural representation for physics prediction. In *NeurIPS*, pages 8813–8824, 2018.

[155] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109–118, 2007.

[156] R. R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. M. Erkmen. Search and rescue robotics. In *Springer Handbook of Robotics*, pages 1151–1173. Springer, 2008.

[157] J. K. Murthy, M. Macklin, F. Golemo, V. Voleti, L. Petrini, M. Weiss, B. Considine, J. Parent-Lévesque, K. Xie, K. Erleben, et al. gradsim: Differentiable simulation for system identification and visuomotor control. In *International Conference on Learning Representations*, 2020.

[158] A. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine. Visual reinforcement learning with imagined goals. *arXiv preprint arXiv:1807.04742*, 2018.

[159] A. Namiki, Y. Imai, M. Ishikawa, and M. Kaneko. Development of a high-speed multifingered hand system and its application to catching. In *IROS*, volume 3, pages 2666–2671. IEEE, 2003.

[160] V. S. Narayanaswamy, J. J. Thiagarajan, R. Anirudh, F. Forouzanfar, P.-T. Bremer, and X.-H. Wu. Designing deep inverse models for history matching in reservoir simulations. In *Machine Learning for Physical Sciences Workshop, Advances in Neural Information Processing Systems*, 2019.

[161] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. In *Computer graphics forum*, volume 25, pages 809–836. Wiley Online Library, 2006.

[162] P. P. Neumann, S. Asadi, A. J. Lilienthal, M. Bartholmai, and J. H. Schiller. Autonomous gas-sensitive microdrone: Wind vector estimation and gas distribution mapping. *IEEE robotics & automation magazine*, 19(1):50–61, 2012.

[163] P. P. Neumann and M. Bartholmai. Real-time wind estimation on a micro unmanned aerial vehicle using its inertial measurement unit. *Sensors and Actuators A: Physical*, 235:300–310, 2015.

[164] P. P. Neumann, V. Hernandez Bennetts, A. J. Lilienthal, M. Bartholmai, and J. H. Schiller. Gas source localization with a micro-drone using bio-inspired and particle filter-based algorithms. *Advanced Robotics*, 27(9):725–738, 2013.

[165] M. Nieuwenhuisen, D. Droeschel, M. Beul, and S. Behnke. Obstacle detection and navigation planning for autonomous micro aerial vehicles. In *2014 international conference on unmanned aircraft systems (ICUAS)*, pages 1040–1047. IEEE, 2014.

[166] D. A. Nix and A. S. Weigend. Estimating the mean and variance of the target probability distribution. In *ICNN'94*, volume 1, pages 55–60. IEEE, 1994.

[167] R. Ouyang, K. H. Low, J. Chen, and P. Jaillet. Multi-robot active sensing of non-stationary gaussian process-based environmental phenomena. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 573–580. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[168] C. J. Paciorek and M. J. Schervish. Nonstationary covariance functions for gaussian process regression. In *Advances in neural information processing systems*, pages 273–280, 2004.

[169] B. Peherstorfer, K. Willcox, and M. Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *Siam Review*, 60(3):550–591, 2018.

[170] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *arXiv preprint arXiv:1710.06537*, 2017.

[171] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *ICRA*, pages 1–8. IEEE, 2018.

[172] J. Peurifoy, Y. Shen, L. Jing, Y. Yang, F. Cano-Renteria, B. G. DeLacy, J. D. Joannopoulos, M. Tegmark, and M. Soljačić. Nanophotonic particle simulation and inverse design using artificial neural networks. *Science advances*, 4(6):eaar4206, 2018.

[173] M. Pharr, W. Jakob, and G. Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[174] D. Pieri and J. A. Diaz. In situ sampling of volcanic emissions with a uav sensorweb: Progress and plans. In *Dynamic Data-Driven Environmental Systems Science*, pages 16–27. Springer, 2015.

[175] L. Piloto, A. Weinstein, A. Ahuja, M. Mirza, G. Wayne, D. Amos, C.-c. Hung, and M. Botvinick. Probing physics knowledge using tools from developmental psychology. *arXiv preprint arXiv:1804.01128*, 2018.

[176] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542*, 2017.

[177] K. Polymenakos, A. Abate, and S. Roberts. Safe policy search with gaussian process models. *arXiv preprint arXiv:1712.05556*, 2017.

[178] K. Polymenakos, L. Laurenti, A. Patane, J.-P. Calliess, L. Cardelli, M. Kwiatkowska, A. Abate, and S. Roberts. Safety guarantees for planning based on iterative gaussian processes. *arXiv preprint arXiv:1912.00071*, 2019.

[179] W. Pouw, J. P. Trujillo, and J. A. Dixon. The quantification of gesture–speech synchrony: A tutorial and validation of multimodal data acquisition using device-based and video-based motion tracking. *Behavior research methods*, 52(2):723–740, 2020.

[180] S. Purushwalkam, A. Gupta, D. M. Kaufman, and B. Russell. Bounce and learn: Modeling scene dynamics with real-world bounces. *ICLR*, 2019.

[181] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[182] F. Ramos, R. C. Possas, and D. Fox. Bayessim: adaptive domain randomization via probabilistic inference for robotics simulators. *arXiv preprint arXiv:1906.01728*, 2019.

[183] A. L. Rankin, L. H. Matthies, and P. Bellutta. Daytime water detection based on sky reflections. In *2011 IEEE International Conference on Robotics and Automation*, pages 5329–5336. IEEE, 2011.

[184] J. Ray, S. Lefantzi, S. Arunajatesan, and L. Dechant. Bayesian calibration of a k-ε turbulence model for predictive jet-in-crossflow simulations. *AIAA Paper*, 2085:2014, 2014.

[185] M. Reggente and A. J. Lilienthal. Using local wind information for gas distribution mapping in outdoor environments with a mobile robot. In *Sensors, 2009 IEEE*, pages 1715–1720. IEEE, 2009.

[186] F. Richter, S. Shen, F. Liu, J. Huang, E. K. Funk, R. K. Orosco, and M. C. Yip. Autonomous robotic suction to clear the surgical field for hemostasis using image-based blood flow detection. *arXiv preprint arXiv:2010.08441*, 2020.

[187] D. Romeres, G. Prando, G. Pillonetto, and A. Chiuso. On-line bayesian system identification. In *2016 European Control Conference (ECC)*, pages 1359–1364. IEEE, 2016.

[188] C. W. Rowley. Model reduction for fluids, using balanced proper orthogonal decomposition. *International Journal of Bifurcation and Chaos*, 15(03):997–1013, 2005.

[189] M. Rutkauskas, M. Asenov, S. Ramamoorthy, and D. T. Reid. Autonomous multi-species environmental gas sensing using drone-based fourier-transform infrared spectroscopy. *Optics express*, 27(7):9578–9587, 2019.

[190] M. G. Safonov and R. Chiang. A schur method for balanced-truncation model reduction. *IEEE Transactions on Automatic Control*, 34(7):729–733, 1989.

[191] C. Sanchez-Garrido, J. Monroy, A. J. Gonzalez-Jimenez, et al. Probabilistic localization of gas emission areas with a mobile robot in indoor environments. 2018.

[192] P. Santana, R. Mendonça, and J. Barata. Water detection with segmentation guided dynamic texture recognition. In *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1836–1841. IEEE, 2012.

[193] C. Schenck. *Liquids & Robots: An Investigation of Techniques for Robotic Interaction with Liquids*. PhD thesis, 2018.

[194] C. Schenck and D. Fox. Detection and tracking of liquids with fully convolutional networks. *arXiv preprint arXiv:1606.06266*, 2016.

[195] C. Schenck and D. Fox. Towards learning to perceive and reason about liquids. In *International Symposium on Experimental Robotics*, pages 488–501. Springer, 2016.

[196] C. Schenck and D. Fox. Visual closed-loop control for pouring liquids. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2629–2636. IEEE, 2017.

[197] C. Schenck and D. Fox. Spnets: Differentiable fluid dynamics for deep neural networks. In *CoRL*, pages 317–335, 2018.

[198] C. Schenck and D. Fox. Spnets: Differentiable fluid dynamics for deep neural networks. In A. Billard, A. Dragan, J. Peters, and J. Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 317–335. PMLR, 29–31 Oct 2018.

[199] S. S. Schoenholz and E. D. Cubuk. End-to-end differentiable, hardware accelerated, molecular dynamics in pure python. *Machine Learning for Physical Sciences Workshop, Advances in Neural Information Processing Systems*, 2019.

[200] D. L. Schwartz and J. B. Black. Analog imagery in mental model reasoning: Depictive models. *Cognitive Psychology*, 30(2):154–219, 1996.

[201] R. N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, 1971.

[202] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung. Neural lander: Stable drone landing control using learned dynamics. *ICRA*, 2019.

[203] C. Stachniss, C. Plagemann, A. J. Lilienthal, and W. Burgard. Gas distribution modeling using sparse gaussian process mixture models. In *Robotics: Science and Systems*, volume 3, 2008.

[204] J. Stam. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 121–128, 1999.

[205] J. Stam. Real-time fluid dynamics for games. In *Proceedings of the game developer conference*, volume 18, page 25, 2003.

[206] J. Stam. *The Art of Fluid Animation*. CRC Press, 2015.

[207] B. Stankus, H. Yong, N. Zotev, J. M. Ruddock, D. Bellshaw, T. J. Lane, M. Liang, S. Boutet, S. Carbajo, J. S. Robinson, W. Du, N. Goff, Y. Chang, J. E. Koglin, M. P. Minitti, A. Kirrander, and P. M. Weber. Ultrafast X-ray scattering reveals vibrational coherence following Rydberg excitation. *Nature Chemistry*, 11(8):716–721, 2019.

[208] A. Straižys, M. Burke, and S. Ramamoorthy. Surfing on an uncertain edge: Precision cutting of soft tissue using torque-based medium classification. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4623–4629. IEEE, 2020.

[209] I. A. Sucan, M. Moll, and L. E. Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012.

[210] Y. Sui, A. Gotovos, J. Burdick, and A. Krause. Safe exploration for optimization with gaussian processes. In *International Conference on Machine Learning*, pages 997–1005. PMLR, 2015.

[211] Z. Sui, L. Xiang, O. C. Jenkins, and K. Desingh. Goal-directed robot manipulation through axiomatic scene estimation. *The International Journal of Robotics Research*, 36(1):86–104, 2017.

[212] A. Sumleeon and P. Kanongchaiyos. Keyframe control of fluid simulation using skeletal particles. In *The International Conference of Computational Methods*, pages 4–6, 2007.

[213] G. T. Sung and I. S. Gill. Robotic laparoscopic surgery: a comparison of the da vinci and zeus systems. *Urology*, 58(6):893–898, 2001.

[214] R. Sykes, D. Henn, S. Parker, and R. Gabruk. Scipuff-a generalized hazard dispersion model. Technical report, American Meteorological Society, Boston, MA (United States), 1996.

[215] T. Takahashi and M. C. Lin. Video-guided real-to-virtual parameter transfer for viscous fluids. In *Transactions on Graphics (SIGGRAPH ASIA 2019)*, 2019.

[216] G. P. Taylor. An automatic block-setting crane. *Meccano Magazine*, 23(3):172, 1938.

[217] A. Tika, N. Gafur, V. Yfantis, and N. Bajcinca. Optimal scheduling and model predictive control for trajectory planning of cooperative robot manipulators. *IFAC-PapersOnLine*, 53(2):9080–9086, 2020.

[218] 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 *IROS*, pages 23–30. IEEE, 2017.

[219] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[220] M. Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *IJCAI*, pages 1930–1936, 2015.

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

[222] S.-T. Tsai, E.-J. Kuo, and P. Tiwary. Learning molecular dynamics with simple language model built upon long short-term memory neural network. *arXiv preprint arXiv:2004.12360*, 2020.

[223] G. Van Belle. *Statistical rules of thumb*, volume 699. John Wiley & Sons, 2011.

[224] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[225] M. Vergassola, E. Villermaux, and B. I. Shraiman. 'infotaxis' as a strategy for searching without gradients. *Nature*, 445(7126):406, 2007.

[226] D. V. Shalashilin. Multiconfigurational Ehrenfest approach to quantum coherent dynamics in large molecular systems. *Faraday Discussions*, 153(0):105–116, 2011.

[227] N. Watters, D. Zoran, T. Weber, P. Battaglia, R. Pascanu, and A. Tacchetti. Visual interaction networks: Learning a physics simulator from video. In *NeurIPS*, pages 4539–4547, 2017.

[228] J. Wu, E. Lu, P. Kohli, B. Freeman, and J. Tenenbaum. Learning to see physics via visual de-animation. 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 153–164. Curran Associates, Inc., 2017.

[229] J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in neural information processing systems*, pages 127–135, 2015.

[230] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez. More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 30–37. IEEE, 2016.

[231] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *arXiv preprint arXiv:1903.11239*, 2019.

[232] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36(4):1307–1319, 2020.

[233] M. M. Zhang, B. Dumitrascu, S. A. Williamson, and B. E. Engelhardt. Sequential gaussian processes for online learning of nonstationary functions. *arXiv preprint arXiv:1905.10003*, 2019.

[234] T. Zhang, G. Kahn, S. Levine, and P. Abbeel. Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search. In *ICRA*, pages 528–535. IEEE, 2016.

[235] W. Zhao, J. P. Queralta, and T. Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744. IEEE, 2020.

[236] C. Zheng, T.-J. Cham, and J. Cai. T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In *ECCV*, pages 767–783, 2018.

[237] G. Zhu and C. Zhang. Object-oriented dynamics predictor. *arXiv preprint arXiv:1806.07371*, 2018.

[238] S. Zhu and A. Boularias. A physically-grounded and data-efficient approach to motion prediction using black-box optimization.

[239] S. Zhu, A. Kimmel, K. E. Bekris, and A. Boularias. Model identification via physics engines for improved policy search. *arXiv preprint arXiv:1710.08893*, 2017.

[240] Y. Zhu and R. Bridson. Animating sand as a fluid. *ACM Transactions on Graphics (TOG)*, 24(3):965–972, 2005.

[241] N. Zotev, A. Moreno Carrascosa, M. Simmermacher, and A. Kirrander. Excited Electronic States in Total Isotropic Scattering from Molecules. *Journal of Chemical Theory and Computation*, 16(4):2594–2605, 2020.