# Whole-Body Motion Planning in Dynamic Environments




Mark Nicholas Finean

St Cross College

University of Oxford

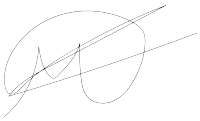
A thesis submitted for the degree of

*Doctor of Philosophy*

Michaelmas 2021

# Declaration

This thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. I declare that this thesis is entirely my own work, and except where otherwise stated, describes my own research.



Mark Nicholas Finean
Oxford, UK
January 2022

# Acknowledgements

I would first like to thank my supervisor, Dr Ioannis Havoutis. I am incredibly grateful for the opportunity to work with you and the rest of the Dynamic Robot Systems group at the Oxford Robotics Institute. You gave me the freedom to explore areas that interested me and always fully supported my endeavours throughout this DPhil journey.

I would also like to express my immense gratitude to Dr Wolfgang Merkt. You have had a tremendous impact on my research by providing guidance and expertise whenever I needed it. You constantly prove to be a fount of knowledge, and it has been a pleasure working with you.

To my fantastic collaborator, Luka Petrović, I am thrilled that we had the opportunity to work together. As researchers, we can easily find ourselves in niche areas where few people are interested. It was great to share ideas and ultimately work on an exciting project together.

I want to extend my thanks and acknowledge the generous support of UK Research and Innovation (UKRI) and the UK Engineering and Physical Sciences Research Council (EPSRC) for making my studies possible through the Autonomous Intelligent Machines and Systems Centre for Doctoral Training (AIMS CDT). I would also like to thank Wendy Poole – your ability to get things done is unrivalled, and you have done so much to make the AIMS CDT such a great experience.

Finally, I would like to thank my parents, Mike and Margaret. I could not have completed my DPhil without you and your unwavering support. Throughout the highs and lows of this DPhil and life in general, you are always there for me.

# Abstract

**The real world is dynamic.**

Over the previous decades, we have seen phenomenal advances across the fields of robotics. From the algorithms used to calculate how a robot moves to the hardware on which it is executed, we find ourselves at a point in time where robots appear as though they could realistically be deployed in our homes to carry out complex tasks. And still, we are not there yet.

Despite the tremendous progress that has been made towards enabling such a technological feat, there remains a disconnect between our accomplishments within the robotics community and the applications in which we hope to see them operate. Few works consider the full integration of pipelines and features required to make a robot operate autonomously in real-world environments. As such, fundamental research problems have remained unidentified and thus unaddressed.

In this thesis, we embark on the ambitious, practical goal of enabling a human support robot to perform whole-body manipulation tasks in real-world indoor environments and in the presence of moving obstacles, such as humans. As we strive to achieve this goal, we uncover new challenges and propose novel solutions.

By focusing on trajectory optimisation-based motion planning to achieve reactive motion re-planning, we introduce and develop the concept of *predicted composite distance fields* for embedding predicted trajectories of moving obstacles into a distance field representation of the environment. We show that this technique can leverage parallel processing to enable a significant performance boost over state-of-the-art GPU-optimised methods of generating distance fields from scratch.

In keeping with the application-driven theme of this thesis, we develop and present a fully integrated motion planning and perception framework. After comparing the different approaches presented in the literature, we propose a fusion of the GPU-based perception framework, GPU-Voxels, and our adaptation of the state-of-the-art optimisation-based motion planner, GPMP2. We demonstrate that our framework can accomplish challenging tasks in dynamic, real-world environments on physical robots.

In the course of this research, we identify a previously unexplored and key challenge for mobile robots, *Trajectory-Constrained Active Gaze Control*, and present a state-of-the-art greedy-optimisation based solution. We believe that identifying this problem is a significant milestone in the pursuit of autonomous robots that operate in dynamic environments. By optimising the capabilities of sensors at our

disposal, we can provide both more effective collision avoidance and reduce the need for a multitude of robot-mounted sensors.

To provide further robustness to our framework in the presence of dynamic obstacles, and humans in particular, we incorporate a predictive element. To this end, we propose the Receding Horizon And Predictive Gaussian Process Motion Planner 2 (RHAP-GPMP2), which uses our method of *predicted composite distance fields* to provide pre-emptive motion planning. We further integrate this with a proposed method for human trajectory prediction that combines intention recognition with trajectory optimisation-based motion planning. Our resultant framework poses a new state-of-the-art benchmark for reactive whole-body motion planning in dynamic environments.

To aid the analysis of our trajectory prediction method, we develop and release the Oxford Indoor Human Motion (Oxford-IHM) dataset. We additionally open-source the code developed in this DPhil research in the hope that this may pave the way for further research in not just pushing our theoretical achievements but also making real-world systems.

# Contents

**List of Figures** **xi**

**List of Abbreviations** **xiii**

**1 Introduction** **1**
- 1.1 Motivation . . . 2
- 1.2 Contributions . . . 5
- 1.3 Publications . . . 6
- 1.4 Thesis Outline . . . 7
- 1.5 Impact . . . 9

**2 Literature Review** **11**
- 2.1 Motion Planning . . . 12
  - 2.1.1 Problem Description . . . 12
  - 2.1.2 Motion Planning Constraints and Criteria . . . 13
  - 2.1.3 Search-based . . . 14
  - 2.1.4 Sampling-based . . . 16
  - 2.1.5 Trajectory Optimisation . . . 19
  - 2.1.6 Learning-based . . . 23
  - 2.1.7 GPMP2 . . . 24
- 2.2 Environment Representations . . . 30
  - 2.2.1 Occupancy Grids . . . 31
  - 2.2.2 Octrees . . . 32
  - 2.2.3 Voxel Hashing . . . 33
  - 2.2.4 Euclidean Distance Transforms . . . 34
- 2.3 Integrated Frameworks and State-of-the-Art . . . 37
- 2.4 Dynamic Obstacle Prediction . . . 39
  - 2.4.1 Motion Planning with Prediction . . . 41
- 2.5 Active Perception for Collision Avoidance . . . 42
- 2.6 Toyota Human Support Robot . . . 43

**3 Predicted Composite Distance Fields** **45**

**4 Real-time Mapping and Motion Planning** **59**

**5 Active Gaze Control** **73**

**6 Integrating Mobile Manipulation with Human Trajectory Prediction** **87**

**7 Summary and Future Work** **111**
7.1 Predicted Composite Distance Fields . . . . . 112
7.2 Real-time Mapping and Motion Planning . . . . . 112
7.3 Active Gaze Control . . . . . 113
7.4 Integrating Mobile Manipulation with Human Trajectory Prediction 113
7.5 Limitations and Future Work . . . . . 114
7.5.1 Active Gaze Control . . . . . 114
7.5.2 Integrated Motion Planning, Perception and Prediction Framework . . . . . 115
7.6 Conclusion . . . . . 117

**References** **119**

# List of Figures

1.1 A Toyota Human Support Robot (HSR). . . . . 4

2.1 An example 2D motion planning task with two large obstacles (grey). The task is to find a collision-free trajectory between the start (blue star) and goal (yellow star) states. . . . . 13
2.2 An illustration of a search-based motion planning approach. As the search algorithm progresses, states are deemed valid (blue) or invalid (red). The states of the solved and typically shortest trajectory are shown in green. . . . . 15
2.3 An illustration of a sampling-based motion planning approach. The graph comprises invalid nodes (red), valid nodes (blue), and edge connections. The solved trajectory is highlighted (green). . . . . 17
2.4 An illustration of trajectory optimisation-based motion planning with the optimised trajectory indicated by the curved line (green). Starting with an initialisation (dashed line), the trajectory is optimised subject to an objective function and any additional constraints. This process typically encourages collision avoidance by penalising states close to obstacles. . . . . 19
2.5 An example factor graph. . . . . 28
2.6 An example voxel grid – a regular grid structure of discrete 3D volumes called *voxels*. . . . . 31
2.7 An example octree data structure in which a voxel can be recursively sub-divided. . . . . 32
2.8 A comparison of Euclidean Distance Transforms (EDTs) and Signed Distance Fields (SDFs) as derived from an example bitmap. . . . . 34
2.9 An example Truncated Signed Distance Field (TSDF). . . . . 35

4.1 An example 2D motion planning task used to compare signed and unsigned distance fields. The sequence of rectangles illustrates the support states of the straight-line initialisation between the start and goal states used for the optimisation. The blue line is provided for clarity and joins consecutive support states. . . . . 69
4.2 A comparison between the trajectories generated using signed and unsigned distance fields throughout the optimisation process for an example task. . . . . 71

# List of Abbreviations

| | |
|---|---|
| **1-D, 2-D, 3-D** | One-, two- or three-dimensional |
| **AI** | Artificial Intelligence |
| **AICO** | Approximate Inference Control |
| **CHOMP** | Covariant Hamiltonian Optimisation for Motion Planning |
| **CPU** | Central Processing Unit |
| **DoF** | Degrees of Freedom |
| **FK** | Forward Kinematics |
| **GPU** | Graphics Processing Unit |
| **GP** | Gaussian Process |
| **GPMP** | Gaussian Process Motion Planner |
| **GPMP2** | Gaussian Process Motion Planner 2 |
| **GTSAM** | Georgia Tech Smoothing and Mapping |
| **HSR** | Human Support Robot |
| **IK** | Inverse Kinematics |
| **IMU** | Inertial Measurement Unit |
| **ITOMP** | Incremental Trajectory Optimisation for Motion Planning |
| **LfD** | Learning from Demonstration |
| **LTV-SDE** | Linear Time-Varying Stochastic Differential Equation |
| **MAP** | Maximum A Posteriori |
| **MAV** | Micro Aerial Vehicle |
| **MILP** | Mixed-Integer Linear Programming |
| **MPC** | Model Predictive Control |
| **NLP** | Nonlinear Programming |
| **OMPL** | Open Motion Planning Library |
| **PCDF** | Predicted Composite Distance Field |
| **PRM** | Probabilistic Road Maps |

**RAM** . . . . . Random-Access Memory

**RL** . . . . . Reinforcement Learning

**ROS** . . . . . Robot Operating System

**RRT** . . . . . Rapidly-exploring Random Tree

**SBPL** . . . . . Search-Based Planning Library

**SDF** . . . . . Signed Distance Field

**SIPP** . . . . . Safe Interval Path Planning

**SLAM** . . . . . Simultaneous Localisation And Mapping

**SQP** . . . . . Sequential Quadratic Programming

**STOMP** . . . . Stochastic Trajectory Optimisation for Motion Planning

**TAMP** . . . . . Task and Motion Planning

**TrajOpt** . . . . Trajectory Optimisation for Motion Planning

**UAV** . . . . . Unmanned Aerial Vehicle

# 1 Introduction

## Contents


**1.1 Motivation** . . . . . **2**
**1.2 Contributions** . . . . . **5**
**1.3 Publications** . . . . . **6**
**1.4 Thesis Outline** . . . . . **7**
**1.5 Impact** . . . . . **9**

## 1.1 Motivation

In recent years, tremendous advances across the fields of robotics have enabled the current state-of-the-art robots to perform phenomenal tasks. In the DARPA Subterranean Challenge 2021 [1], we saw quadruped robots map and navigate complex underground environments. We have seen impressive control demonstrations, such as parkour and backflips on humanoid robots, from Boston Dynamics's ATLAS robot [2]. In a world with such impressive robot hardware constructions and well-developed research areas, such as perception and motion planning, we must naturally ask "why aren't robots more integrated into our daily lives?" and "why don't I have a robot cleaning my house yet?"

Early on in this DPhil research, we became eagerly involved in conducting and applying research to participate in the RoboCup@Home competition. RoboCup@Home is one of the different leagues of the well-known international robotics competition, RoboCup. The competition aims to encourage innovation and development in service robotics for domestic environments. The competition tasks are typically similar to those performed by humans in a home environment, such as emptying rubbish bins, finding items in a different room, and tidying up the floor. Prior to starting an 'examined' task, competing teams are often allowed time in the competition arena to perform mapping and often carefully script behaviours. This whole process ultimately inspired and motivated the research presented in this thesis as we became interested in posing additional research questions such as "how can we still perform the tasks but without any prior mapping time?" and "what if objects move or someone walks in the way?" Addressing the challenges posed by these types of questions is critical in our pursuit of enabling robots to robustly perform tasks in real-world environments and in the presence of humans.

The primary challenges that we face arise from two fundamental observations of the world that we live in:

1. The world is often **dynamic**.

2. **Time matters**.

*The world is often dynamic* — environments such as a household, or an office, are dynamic. In this context, we use the term *dynamic* to describe an environment in which objects may be non-static as a robot operates and performs tasks. In typical environments, such as a household, humans will move around, perform tasks, and even rearrange furniture and objects. For robots to operate in such dynamic environments, they must adapt and react to changes in the environment, whether it is suddenly avoiding a human or placing a cup down on a table that has recently been moved. While a plethora of research has been conducted in fields such as motion planning and perception, far less has focused on being applicable in real, dynamic environments. Dynamic environments place great importance on live sensing capabilities since one can no longer assume a static environment; instead, any changes that will influence future actions, such as motion planning, must be quickly observed.

*Time matters* — in a similar vein, there is considerable research that performs computations either offline or with long planning times online. In a real dynamic environment as we have described, updates and computations need to be performed quickly and online in order for a robot to react to changes in the local environment. Failure to observe changes in the environment and redetermine a robot's response fast enough can be dangerous and lead to collisions.

The process of autonomously performing complex tasks is multi-faceted, with numerous avenues for research, many of which will not be covered in this thesis. To manipulate objects, there is the field of "grasp synthesis" in which one determines the appropriate grasp position to pick up an object. In "Task and Motion Planning (TAMP)", one can determine the sequence of actions required to accomplish complex tasks; this might be required in cases such as picking up objects inside a fridge or cupboard that require other objects to be moved first. In this thesis, we focus on the core capabilities of being able to perform a single action robustly in the face of a changing environment, i.e. achieve a given goal state.

The "whole-body" aspect makes this problem particularly challenging. A simple 2D navigation task only requires us to consider representing the environment in 2D and similarly the motion planning only needs to be performed in a low-dimensional space, for example, $[x, y, \theta]$ where $x$ and $y$ are the robot's position in 2D space and

$\theta$ is the robot's orientation. The computational overhead associated with perception and motion planning becomes much greater in higher dimensions. To perform manipulation tasks, many robots have an articulated arm; the Toyota HSR has both a holonomic base and an articulated arm with five degrees-of-freedom. In contrast to 2D navigation, motion planning is thus performed in a higher-dimensional configuration space for the robot, and to accurately model collision-avoidance, should use a fully 3D representation of the environment. For the dynamic environments that we are interested, the added complexity and associated computational costs pose challenges that require the careful consideration of which motion planning methods to adopt, as discussed in Chapter 2.

We set ourselves the practical task of enabling a Toyota Human Support Robot (HSR) (Figure 1.1) to perform whole-body loco-manipulation tasks in the presence of moving humans. This challenge has guided our research, revealing new problems to be solved and ultimately shaping the contributions of this thesis, as described in the following section.




**Figure 1.1:** A Toyota Human Support Robot (HSR).

## 1.2 Contributions

The primary contributions of this thesis are:

- Introducing the concept of *predicted composite distance fields* and demonstrating their use in avoiding moving obstacles (Chapter 3).
- Exploring the practical details of integrating real-time distance field generation with fast motion planning (Chapter 4).
- Introducing the novel problem of *Trajectory-Constrained Active Gaze Control* and providing a greedy-optimisation based solution (Chapter 5).
- Providing an open-source framework for enabling fast motion re-planning in dynamic environments (Chapter 4 and Chapter 6).
- Using GPU-accelerated *predicted composite distance fields* with a novel motion planning-based human trajectory prediction method to demonstrate person avoidance across a range of challenging scenarios (Chapter 6).
- The development and release of the Oxford Indoor Human Motion (Oxford-IHM) dataset – a goal-based human trajectory prediction dataset that includes RGB-D data from a moving robot's perceptive and motion-capture ground truth positions (Chapter 6).

In Section 1.3, we list the publications that have resulted from this research. In Section 1.4, we provide an outline of the thesis before summarising the impact of our research in Section 1.5.

While this thesis provides an integrated framework encompassing motion planning, perception, and human motion prediction, we do not claim to provide a 'complete' system. In real-world robotics applications and future work, we anticipate our framework to be part of a more extensive system to exhibit fully autonomous behaviours. A household cleaning robot, for example, may extend our work by incorporating semantics to understand the objects around it; some will pose simply as obstacles, while others may influence the tasks themselves. Robots such as a Toyota HSR have additional capabilities, such as speech, that were not explored

in this research. In practice, a higher level of 'task planning' is required to bring a robot's capabilities together and chain actions. However, for loco-manipulation, a robot must ultimately perform lower-level 'motion planning' to determine its movements, motivating our contributions.

## 1.3 Publications

The main contributions of work in this thesis are presented in the following publications and manuscripts under review:

- Predicted Composite Signed-Distance Fields for Real-Time Motion Planning in Dynamic Environments
  *Mark Nicholas Finean, Wolfgang Merkt, and Ioannis Havoutis*
  The International Conference on Automated Planning and Scheduling, 2021 [3]

- Simultaneous Scene Reconstruction and Whole-Body Motion Planning for Safe Operation in Dynamic Environments
  *Mark Nicholas Finean, Wolfgang Merkt, and Ioannis Havoutis*
  IEEE International Conference on Intelligent Robots and Systems, 2021 [4]

- Where Should I Look? Optimised Gaze Control for Whole-Body Collision Avoidance in Dynamic Environments
  *Mark Nicholas Finean, Wolfgang Merkt, and Ioannis Havoutis*
  IEEE Robotics and Automation Letters (RA-L) [5]

- Motion Planning in Dynamic Environments Using Context-Aware Human Trajectory Prediction
  *Mark Nicholas Finean, Luka Petrović, Wolfgang Merkt, Ivan Marković, and Ioannis Havoutis*
  The International Journal of Robotics Research (IJRR) (under review) [6]

In addition to the aforementioned publications, and in line with our ambition to achieve practical results, we further provide open-source access to the integrated framework developed during this course of research[1] and, as we describe in Chapter 6,

[1] `https://github.com/ori-drs/integrated-dynamic-motion-planning-framework`

we additionally release a dataset for human trajectory prediction in which humans move between intended goals[2].

## 1.4 Thesis Outline

In this section, we outline the story of this thesis and the work that we have undertaken in enabling whole-body manipulators to perform tasks in real-world and dynamic environments.

In Chapter 2, we provide an extensive literature review of the core areas that are relevant to the contributions presented in this thesis. We use this opportunity to provide the reader with a broader understanding of the context of our research, while more specific literature reviews are provided as part of the paper contributions presented in subsequent chapters. Chapter 2 describes the *Motion Planning* problem and provides the reader with some of the various methods that can be used to approach the problem. With this background, we provide further detail on the core motion planning approach that we adopt throughout this thesis – an extension of a state-of-the-art trajectory-optimisation method. Complementary to motion planning algorithms is the availability of collision information about the environment; as such, we will discuss various *Environment Representations* that have been developed over the years, culminating with a dominant theme of this thesis – *distance fields*. We use the term *distance fields* to broadly refer to distance transforms, regardless of their metric and whether they are *signed* or *unsigned*. We will discuss state-of-the-art works that provide fully integrated motion planning and perception systems. We will then discuss the prior literature on human trajectory prediction within the context of motion planning and dynamic obstacle avoidance before concluding the literature review with a review on active perception for collision avoidance and a brief description of the primary robot used in this research, the *Toyota Human Support Robot (HSR)* [7].

In Chapter 3, we propose a method for how a state-of-the-art trajectory optimisation method can be adapted to perform motion planning while accounting for the predicted motion of moving obstacles – by considering a time series

[2] `https://ori-drs.github.io/oxford-indoor-human-motion-dataset`

of predicted environment representations, we perform time-configuration space planning with optimisation-based methods. To achieve this feat, we introduce the concept of *predicted composite distance fields* (PCDFs)– a method of generating distance fields that we show to be significantly faster than computing them from scratch. We combine our proposed methods into a novel framework that embeds the predicted trajectories of moving obstacles directly into the signed distance fields used for motion planning, enabling pre-emptive collision avoidance in dynamic environments. We verify our approach on a physical 7-DoF Franka Emika Panda robotic arm and demonstrate the avoidance of moving obstacles by accounting for their predicted trajectories.

In Chapter 4, we develop an integrated framework for motion planning and perception in dynamic environments. We address practical concerns such as whether 'signing' distance fields is beneficial in motion planning tasks and investigate three state-of-the-art approaches for generating distance fields in environment reconstruction. We use our findings to develop a hybrid system that links a receding-horizon implementation of GPMP2 [8] with a GPU-optimised perception pipeline. We validate our approach on a physical Toyota HSR with live sensor data and accomplish various whole-body motion planning tasks in dynamic environments.

In Chapter 5, we identify the novel problem of *Trajectory-Constrained Active Gaze Control* for collision avoidance. For robots with independently controllable sensors, such as the Toyota HSR's head camera, there is both an opportunity and a challenge to observe specific regions of the workspace and provide observations that are relevant to the robot's currently planned trajectory. In this work, we detail the problem and propose an optimisation-based approach that greedily determines the optimal camera pose for subsequent timesteps. We believe that this is the first 'smart' controller for gaze control that jointly optimises for safe obstacle avoidance and environment exploration. We implement our method in 3D by building upon the GPU-based perception framework discussed in Chapter 4 and show that our controller outperforms benchmark approaches. We verify our approach and its ability to perform in dynamic environments with demonstration on a physical HSR robot.

In Chapter 6, we demonstrate our thesis goal of performing whole-body tasks with a physical mobile manipulator and avoiding the predicted trajectories of a human in the scene. To achieve this, we provide multiple extensions to the approach presented in Chapter 4. We provide the first implementation of PCDFs (introduced in Chapter 3) within a GPU-optimised framework (described in Chapter 4) and provide a benchmarking analysis of our composite approach against a state-of-the-art, GPU-optimised algorithm for computing Euclidean Distance Transforms (EDTs) – we show that our method significantly reduces the time needed to compute subsequent distance fields. To achieve our ambition of being able to avoid the predicted trajectories of humans in a scene, we exploit recent advances in image segmentation estimate the positions of people in the robot's workspace and remove dynamic obstacles from the maintained voxelmap of the static scene. We propose a novel method for human trajectory prediction that combines *intention recognition* with trajectory optimisation-based motion planning. In the course of this research, we further develop and release the Oxford Indoor Human Motion (Oxford-IHM) dataset – a human trajectory prediction dataset in which the humans move between intended goals.

Finally, in Chapter 7 we provide a summary of the work presented in this thesis followed by an outline of the future directions for this research.

## 1.5 Impact

The overarching contribution of this thesis is the development of an integrated perception and whole-body motion planning framework that is demonstrated to enable a mobile manipulator to robustly perform tasks in a real-world environment and account for the predicted motion of humans its workspace.

We introduce and detail the concept of *predicted composite distance fields* – a fast method for generating predicted distance fields as a means of enabling reactive collision avoidance. This method is particularly useful for optimisation-based motion planning algorithms, which have previously resorted to geometric collision avoidance costs due to the prohibitively long computation times for distance fields [9].

In the course of this research, we identify a previously unexplored and key challenge for mobile robots, *Trajectory-Constrained Active Gaze Control*. This problem is faced by robots that have additional degrees of freedom to position and orient their sensors, such as a head camera. We present a greedy-optimisation based solution and show that by optimising the positions of such sensors, we can achieve more effective collision avoidance and map exploration than baseline methods, presenting a new state-of-the-art approach.

We hope that the detailing of our work, the release of the Oxford-IHM dataset, and the release of open-source code will motivate and encourage others to continue this work.

*Those who cannot remember the past are condemned to repeat it.*

— George Santayana, 1905 [10]

# 2 Literature Review

## Contents


**2.1 Motion Planning . . . . . 12**
2.1.1 Problem Description . . . . . 12
2.1.2 Motion Planning Constraints and Criteria . . . . . 13
2.1.3 Search-based . . . . . 14
2.1.4 Sampling-based . . . . . 16
2.1.5 Trajectory Optimisation . . . . . 19
2.1.6 Learning-based . . . . . 23
2.1.7 GPMP2 . . . . . 24
**2.2 Environment Representations . . . . . 30**
2.2.1 Occupancy Grids . . . . . 31
2.2.2 Octrees . . . . . 32
2.2.3 Voxel Hashing . . . . . 33
2.2.4 Euclidean Distance Transforms . . . . . 34
**2.3 Integrated Frameworks and State-of-the-Art . . . . . 37**
**2.4 Dynamic Obstacle Prediction . . . . . 39**
2.4.1 Motion Planning with Prediction . . . . . 41
**2.5 Active Perception for Collision Avoidance . . . . . 42**
**2.6 Toyota Human Support Robot . . . . . 43**

## 2.1 Motion Planning

A robot is ultimately a collection of hardware components that can perform actions autonomously. Sensors, such as lasers and cameras, are used to take measurements that enable calculations to be made about itself and the environment, while mechanisms, such as servos and motors, can exert forces and torques in order to change the state of the robot or that of the environment. By sequentially exerting specific forces and torques, such as in the joints of a robotic arm, a robot can accomplish a desired task such as reaching out in front or closing a gripper. For a given task, the algorithmic determination of the required time series of actions that the robot must make is the process of 'motion planning'.

Motion planning forms a core research area of robotics and a key part of this thesis. As such, in the following sections, we will explore the problem of motion planning and the various ways in which it can be solved. We will summarise the current state-of-the-art approaches and finally describe the approach implemented in our work.

### 2.1.1 Problem Description

Given the state space for a robot, $\mathcal{X}$, the general motion planning problem is to determine a sequence of states that results in the robot transitioning from a starting configuration, $\boldsymbol{\theta}_s \in \mathcal{X}$, to a goal state, $\boldsymbol{\theta}_g \in \mathcal{X}$, such that all states in the trajectory are *feasible*. In this thesis, we will generally refer to this resultant sequence of states as a *trajectory*, $\boldsymbol{\theta}(t) :\rightarrow \mathbb{R}^D$, where $D$ is the dimensionality of the robot's state. States can be robot configurations as well as include higher-order derivatives. The condition of *feasibility* requires that all states in the trajectory obey any additional constraints of the problem. Typical constraints that we wish to impose include 'collision avoidance' and that the robot's joint limits are respected. States that result in a collision or violate constraints are considered *infeasible*. A feasible trajectory is one for which all of the executed states are feasible and results in the goal state being achieved, regardless of how efficient it is. We can consider the state space as the union of a valid and invalid set, $\mathcal{X} = \mathcal{X}_{\text{valid}} \cup \mathcal{X}_{\text{invalid}}$. We thus require that all states in the resultant trajectory belong the subset of the state-space that obeys

constraints, i.e. $\boldsymbol{\theta}(t_i) \in \mathcal{X}_{\text{valid}} \quad \forall \quad t_s \leq t_i \leq t_g$, where $t_s$ and $t_g$ are the start and finish times of the trajectory. Over the years, a plethora of methods has been developed to solve the problem of determining a feasible trajectory, $\boldsymbol{\theta}(t)$, each with its advantages and disadvantages. In the following section, we will describe some of the constraints and criteria that motion planners may need to consider. With these in mind, we will then explore some of the different motion planning paradigms, describe the current state-of-the-art, and outline the specific approach that has been selected for use and development in this thesis. To aid our discussion, we will use Fig. 2.1 as an illustrative example of a simple 2D motion planning task for various motion planning approaches to solve for a collision-free trajectory.



**Figure 2.1:** An example 2D motion planning task with two large obstacles (grey). The task is to find a collision-free trajectory between the start (blue star) and goal (yellow star) states.

### 2.1.2 Motion Planning Constraints and Criteria

As previously described, a common constraint in motion planning is that of collision avoidance. As illustrated in Fig. 2.1, a robot's environment often contains regions of free space as well as regions in which objects in the environment act as obstacles. We thus wish to determine a sequence of states that avoid colliding with the environment. This is commonly known as the 'piano mover's problem' for a rigid body. Although necessary, this collision avoidance alone is generally not sufficient.

In the case of articulated robots, such as robotic arms, we must respect *joint limits* - an inequality constraint in which the respective physical range of a motion for each robots joint must be obeyed.

Whether due to hardware constraints or safety requirements of the environment in which a robot operates, its range of motion, speed, and ability to accelerate is often constrained. Consideration of bounds on velocity, acceleration, and force, in addition to collision-avoidance, is known as *kinodynamic planning*.

Another important criterion to account for in motion planning is that of *optimality*. In the real-world execution of trajectories, we often have a specific metric that we want to optimise dependent on the task. For battery-operated mobile robots, one may formulate the motion planning problem to solve for trajectories that minimise metrics such as path length or total energy consumption. For robotic arms, one may instead or additionally impose *smoothness* constraints. Smoothness is a term used to describe how smoothly a robot transitions between states in a given trajectory - erratic motions are undesirable in robotics. While aiming to minimise energy consumption may lead to smoother trajectories, one can consider other constraints, such as explicitly penalising the distance between consecutive states.

As we will see in the following sections, different motion planners account for constraints differently, resulting in certain planners being preferable depending on the application.

### 2.1.3 Search-based

*Search-based* methods approach motion planning by firstly discretising the problem and constructing a graph representation. The task of solving for a trajectory is then reduced to how to search the graph best. An illustration of this approach is shown in Fig. 2.2. While there are many different discretisation strategies, such as approximate cell decomposition [11] and multi-resolution lattices [12], the particular method of discretisation used is likely to depend on the context of the problem. Often a regular grid or lattice structure is used in motion planning to discretise the configuration space of a robot; with a finer discretisation of the space, one can

achieve a more accurate final goal state approximation to this input goal state, and a more efficient trajectory, at the expensive of a greater computation cost.

Early search-based solutions to the problem include Dijkstra's algorithm [13] which greedily iterates through graph nodes to find the optimal path. The extensive nature of an algorithm such as Dijkstra's can result in significant memory usage and slow computation times, particularly in higher dimensions, before a solution is found. However, such an algorithm has the benefit of two powerful properties: *completeness* and *optimality*. An algorithm that is considered *complete* is guaranteed to find a solution if it exists and terminates otherwise. An algorithm that is considered *optimal* is guaranteed to find the best solution if one exists. By incorporating

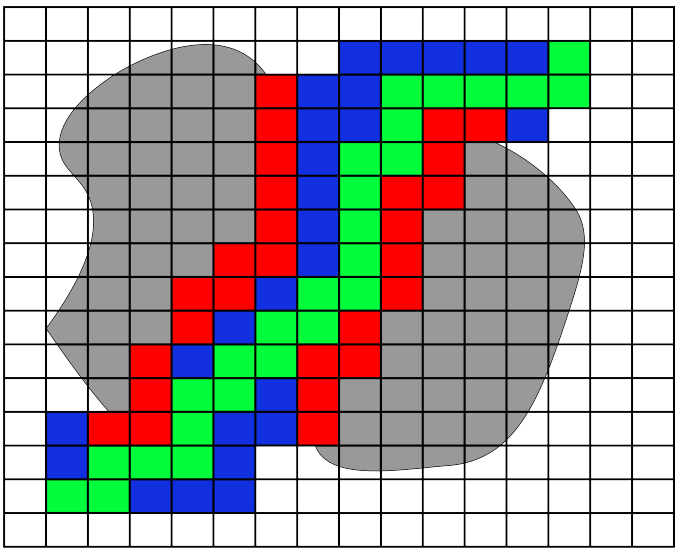


**Figure 2.2:** An illustration of a search-based motion planning approach. As the search algorithm progresses, states are deemed valid (blue) or invalid (red). The states of the solved and typically shortest trajectory are shown in green.

additional *heuristics*, such as the estimated distance from a given node to the goal, algorithms such as A* [14] are able to find solutions much faster and maintain the appealing properties of completeness and optimality. A relaxation of the problem gives rise to algorithms such as weighted A*, which can typically return a sub-optimal solution much faster.

When considering dynamic environments, constructing a graph poses additional challenges because each node needs to be designated as a valid state or not. This designation needs to be updated as new observations of the environment are made, and a re-planned trajectory returned if the currently planned path becomes infeasible.

There are a number of search-based algorithms that have been developed and demonstrated in scenarios that contain moving obstacles. Menon et al. [15, 16] use ARA* with informative heuristics to smoothly pick up moving objects from a conveyor belt. Focussed Dynamic A* [17] and D* Lite [18] have re-planning capabilities that enable new trajectories to be returned faster than planning from scratch when new information is obtained. Safe Interval Path Planning (SIPP) [19] uses the concept of 'safe intervals' to perform mobile navigation on a PR2 robot in dynamic environments. Likhachev et al. propose the Anytime Dynamic A* (AD*) [20] algorithm, which has the ability to re-plan, as in D*-Lite, while achieving the anytime performance of ARA*.

However, the key drawback of search-based algorithms remains their scalability to high-dimensional spaces. This problem is often dubbed the *curse of dimensionality* – for high-dimensional state spaces, explicitly constructing the valid state space, $\mathcal{X}_{\text{valid}}$, is generally considered intractable. Hierarchical Dynamic Roadmaps (HDRMs) [21] have been proposed as a way of addressing this by creating memory-efficient configuration-workspace embeddings. However, they assume the use of static environment maps for graph construction and have not been demonstrated for floating-base robots.

To accomplish our ultimate goal of fast whole-body motion re-planning on a mobile manipulator in dynamic environments, we require real-time capabilities for planning and re-planning smooth trajectories for high degree of freedom (DoF) robots, i.e. robots with a mobile base and articulated arms.

### 2.1.4 Sampling-based

While search-based motion planners explicitly use graph representations, sampling-based methods are able to solve using continuous-space representations – this said, many sampling-based approaches still use a discrete representation of the space. In contrast to search-based methods, sampling-based motion planners typically offer faster planning, particularly in high-dimensional spaces, at the expense of lower-quality paths and weaker guarantees on optimality. They achieve this by using sampling to approximate $\mathcal{X}_{\text{valid}}$.

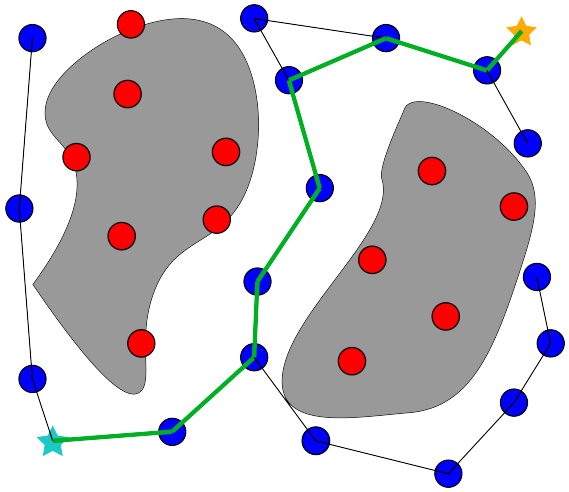


**Figure 2.3:** An illustration of a sampling-based motion planning approach. The graph comprises invalid nodes (red), valid nodes (blue), and edge connections. The solved trajectory is highlighted (green).

Rapidly-exploring Random Trees (RRTs) [22], Probabilistic Road Maps (PRMs) [23], and their derivatives [24–26], are commonly used sampling-based methods. In the case of RRTs, the start configuration is used as a root node for a tree to grow from. As samples are drawn from the state-space, connections to the tree will be attempted from states that are deemed to be feasible. The tree is grown until the goal configuration has been added and can then be efficiently traced to determine a feasible trajectory. In bi-directional variants of RRT [27], a tree is also grown from the goal configuration. Whereas RRT-based methods grow between the start and goal configurations, PRMs construct a roadmap of the free configuration space first by sampling the state space for feasible nodes and using local planning between them. With the graph constructed, the roadmap is ready for the *query phase*, in which start and goal configurations can be inserted as additional nodes that are to be connected to the roadmap via a local planner. An example of such a roadmap is shown in Fig. 2.3.

One of the problems of sampling-based planners is the quality of the solutions that are produced when they are found. Sampling-based methods can often result in redundant and jerky motions, necessitating a post-processing step that incurs further computational cost and time. To address this issue, Karaman et al. [26] introduced the asymptotically optimal planners, RRT* and PRM*, which aim to find the optimal path for a given metric, such as path length.

Sampling-based motion planners have been widely adopted in the robotics literature due to their effectiveness at solving high-dimensional problems, and due to the widely-used open-source implementations of them, such as the Open Motion Planning Library (OMPL) [28]. Compared to search-based planning, sampling-based approaches achieve superior computational efficiency at the expense of completeness guarantees – at best, sampling-based algorithms are deemed to be *probabilistically* or *asymptotically complete*. By this, we mean that if a solution exists, an asymptotically complete algorithm will find a solution with a probability that approaches unity as the state-space sampling tends to infinity.

For the research aims presented in this thesis, we need to strongly consider a motion planner's ability to satisfy constraints and solve for trajectories quickly, adapting them to an evolving environment. Regarding constraints, it can be difficult for sampling-based planners to satisfy equality constraints. For example, consider the task of a robot arm picking up and placing a full cup of water on a table. To avoid spilling the water, the robot arm must maintain a specific end-effector orientation and thus obey an equality constraint. Within the high-dimensional space of robot configurations, the states that obey this equality constraint will lie on a plane or constraint manifold. For a sampling-based planner to find a solution would require a dense and potentially intractable sampling of the high-dimensional space. While there are works that use techniques such as projection [29, 30] or tangent space sampling [31, 32] to satisfy these constraints, this comes at the cost of computation time. In contrast, considering multiple objectives is relatively simple in optimisation-based motion planning. Furthermore, while sampling-based planners such as RRT and PRM are suitable for static environments, naively re-constructing a state-space graph and finding a solution from scratch each time the environment changes is computationally expensive and time-consuming.

While sampling-based re-planning strategies and extensions have been explored [33–35], reported planning and re-planning times are still much larger than can be achieved with trajectory optimisation-based methods as discussed in the following section.

### 2.1.5 Trajectory Optimisation

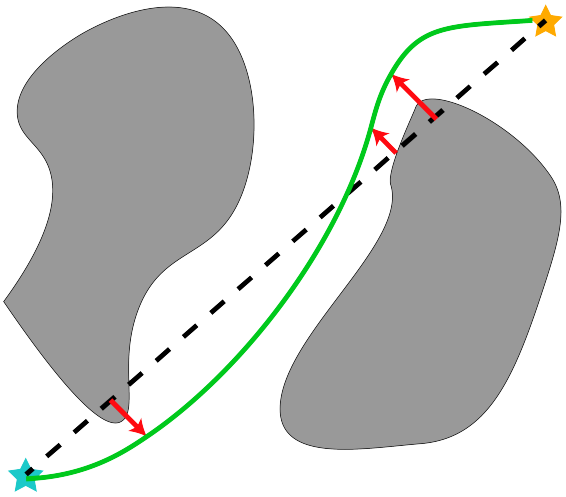


**Figure 2.4:** An illustration of trajectory optimisation-based motion planning with the optimised trajectory indicated by the curved line (green). Starting with an initialisation (dashed line), the trajectory is optimised subject to an objective function and any additional constraints. This process typically encourages collision avoidance by penalising states close to obstacles.

In trajectory optimisation-based methods, motion planning is framed as an optimisation problem. The optimisation of an objective function is generally responsible for obstacle avoidance, constraint costs, and the generation of smooth trajectories. An illustration of this approach is shown in Fig. 2.4. Leading on from our description of the motion planning problem in Section 2.1.1, the trajectory optimisation paradigm views the motion planning problem as:

$$\begin{aligned} \text{minimise} \quad & \mathcal{F}[\boldsymbol{\theta}(t)] \\ \text{subject to} \quad & \mathcal{G}[\boldsymbol{\theta}(t)] \leq 0, \quad i = 1, \ldots, n_{\text{ineq}} \\ & \mathcal{H}[\boldsymbol{\theta}(t)] = 0, \quad i = 1, \ldots, n_{\text{eq}} \end{aligned} \tag{2.1}$$

where $\mathcal{F}[\boldsymbol{\theta}(t)]$ is the cost functional to be optimised. Typically, the cost functional is used to encode collision avoidance and encourage smooth trajectories, for example, by penalising the distance between consecutive states or minimising internal energy assigned to the trajectory. This framing of collision avoidance differs from the search- and sampling-based methods discussed previously. Previously, states were discarded if they resulted in collision (e.g. as part of rejection sampling); we call this a *hard constraint*. While some optimisation methods [36, 37] do account for collision

avoidance as a hard constraint, it can also be implemented as a *soft constraint*, meaning that it is encouraged but not strictly enforced by the algorithm. $\mathcal{G}[\boldsymbol{\theta}(t)]$ and $\mathcal{H}[\boldsymbol{\theta}(t)]$ are inequality and equality constraint functionals, respectively. As discussed previously, a common inequality constraint that we wish to impose is the respect for joint limits, such as angle, velocity, and torque. Equality constraints are usually task-dependent and include fixing start and goal configurations for the optimisation. For example, in manipulation problems, an object may need to be grasped at a specific angle and thus requires the robot's end-effector to achieve a specific pose.

A prominent algorithm in trajectory optimisation-based motion planning is Covariant Hamiltonian Optimization for Motion Planning (CHOMP) [38, 39] and CHOMP-related algorithms [40, 41]. These methods iteratively improve an initial trajectory that is not necessarily feasible by using *covariant gradient descent* on a cost function across the states of the trajectory. The technique is covariant in respect of the trajectory updates being only a function of the trajectory itself rather than the particular representation or time-parametrisation used [39]. The initial trajectory that is input to the algorithm can be provided by another motion planner, with CHOMP effectively performing post-processing of a trajectory, or it can be provided purely as an initialisation or 'initial guess'; a common approach is to initialise the planner with a straight-line trajectory [8, 36].

CHOMP models the cost of a trajectory using a prior term, $f_{\text{prior}}$, which accounts for smoothness, and an obstacle avoidance term, $f_{\text{obs}}$, which assigns a cost based on the robot's proximity to obstacles. The objective function can thus be written as

$$\mathcal{F}[\boldsymbol{\theta}] = f_{\text{prior}}[\boldsymbol{\theta}] + \lambda f_{\text{obs}}[\boldsymbol{\theta}], \tag{2.2}$$

where $\lambda$ is tuning parameter that determines the trade-off between the two cost terms. While $f_{\text{prior}}[\boldsymbol{\theta}(t)]$ can be extended to higher-order derivatives, Zucker et al. [38] present their approach using a smoothness term that takes the form of an integral over squared first-order derivatives,

$$f_{\text{prior}}[\boldsymbol{\theta}] = \frac{1}{2}\int_0^1 \left\| \frac{d}{dt}\boldsymbol{\theta}(t) \right\|^2 dt. \tag{2.3}$$

The collision avoidance term, $f_{\text{obs}}$, is calculated as an arc-length parametrised line integral of the robot's collision representation through a cost field of the environment. For thoroughness, this term can be formulated as:

$$f_{\text{obs}}[\boldsymbol{\theta}] = \int_0^1 \int_u \left\| \frac{d}{dt} x(\boldsymbol{\theta}(t), u) \right\| du\, dt, \tag{2.4}$$

where we define $c$ as a workspace potential $c : \mathbb{R}^3 \to \mathbb{R}$ that is used to assign a cost to a given part of the robot's body, $u$, at point $\boldsymbol{x}$ in the workspace. We note that optimisation-based methods such as CHOMP require cost gradients to guide the optimisation to a local minimum. In the case of collision avoidance, signed distance fields are used to represent the environment so that gradients can be easily obtained as well as provide a workspace potential, $c$. In contrast, stochastic motion planners such as STOMP [42] can be used with more general cost functions that are non-smooth or non-differentiable, for which gradients are not available. A 'stochastic gradient' can be generated instead by considering many trajectories. CHOMP uses a simplified robot model to make the optimisation tractable. Each link of the robot can be approximated by a geometric shape primitive such as a sphere or capsule. Collision checking for a spherical robot model is thus reduced to distance field checking for each sphere and comparing the distance to the radius of the sphere.

CHOMP's update rule is derived by linearising the objective function in Eqn. 2.2 around the current trajectory. Using a first-order Taylor series approximation, Zucker et al. [38] derive a covariant update rule that can be iteratively applied to a trajectory, performing gradient descent and consequently optimising the trajectory. In practice, finite differencing is used to approximate time derivatives so that updates can be applied to a discretised trajectory.

While CHOMP uses a projected gradient descent method of optimisation, Schulman et al. [36, 37] presented an alternative method, Trajectory Optimisation for Motion Planning (TrajOpt), which uses Sequential Quadratic Programming (SQP) to perform the optimisation. While the two methods similarly perform trajectory optimisation n, iteratively improving an initial trajectory, TrajOpt has two key differences from CHOMP [36]. Firstly, while CHOMP uses the distance field checking method described previously, TrajOpt uses convex-convex continuous-time

collision checking; rather than using a simplified spherical model of the robot, TrajOpt uses the convex hull of the robot meshes used to represent each link. This has the effect of providing a more accurate representation of the robot model for collision checking. Furthermore, due to the continuous-time native of the collision checking, TrajOpt can often find feasible trajectories using comparatively fewer states than CHOMP, enabling a reduced computational cost. However, in practice, a densely parameterised trajectory is still required to provide a smooth output trajectory and to account for collision avoidance of small obstacles.

To overcome the large computational cost associated with using a large number of states to represent a trajectory, continuous-time trajectory representations such as B-splines [43] and kernel methods [44] have been explored. More recently, Mukadam et al. [45] proposed the Gaussian Process Motion Planner (GPMP) which samples trajectories from a Gaussian Process (GP) to provide continuous-time trajectories. A key benefit of their approach is that trajectories are parameterised by a small number of states, while the trajectory can be queried at any point in time using GP interpolation. In further work [8], Mukadam et al. translated their GPMP approach into a probabilistic view of motion planning, re-framing the motion planning problem as probabilistic inference on a factor graph. By exploiting sparsity in the problem formulation and using solvers with quadratic convergence rates, their resulting algorithm, GPMP2, provides a new state-of-the-art motion planner. Since the development of GPMP2, there have been numerous extensions and applications of it [46–50].

ITOMP [9] uses a stochastic motion planning approach for operation in dynamic environments. They incorporate the predicted motion of moving obstacles by considering a short time horizon and computing a conservative local bound on the trajectory of each obstacle.

While the primary limitation of trajectory optimisation-based motion planners is their susceptibility to local minima, in contrast to sampling and search-based methods, we believe that the benefits of their superior performance in planning times make them strong candidates for achieving the goals of this thesis.

GPMP2 is a state-of-the-art approach for motion planning by trajectory optimisation, and we believe it offers great potential for achieving fast re-planning for whole body manipulators in dynamic environments. As such, we dedicate Section 2.1.7 to reviewing this approach and motivating our adoption of it in this thesis.

### 2.1.6 Learning-based

Over the past decade, we have seen a tremendous rise in the application of machine-learning methods across numerous fields. So too have we seen the adoption of such techniques in motion planning.

Ye and Alterovitz propose DGMP [51], a motion planning approach that uses multi-component rapidly-exploring road maps to incrementally compute trajectories and optimise a cost metric that is learned from expert demonstrations. Qureshi and Yip propose DeepSMP [52], which uses neural networks to improve the efficiency of sampling for unbiased sampling-based motion planners and thus reduce their computation time. In further work, Qureshi et al. propose MPNet [53] and use deep neural networks to both generate samples for a sampling-based motion planner and to generate collision-free trajectories. By learning from expert trajectories, they use a trained model to iteratively determine the next robot state for each step in a trajectory.

Learning-based methods come with their own disadvantages. A significant limitation is their ability to generalise to different tasks and environments than what they were trained on. To this end, there is a substantial amount of literature dedicated to *transfer learning* in which learnt models are adapted or re-trained so that they can be deployed in different environments [54–56]. In the cases of DeepSMP and MPNet, as described above, reported planning times are still inferior to those that can be achieved by trajectory optimisation-based methods.

In the following section, we will resume our discussion of GPMP2 as a state-of-the-art trajectory optimisation method to provide appropriate background for the motion planning used in further chapters of this thesis.

### 2.1.7 GPMP2

GPMP2 is ultimately the motion planner of choice for extension and use within this thesis. As such, we dedicate this section to justifying our choice and explaining how the algorithm works. After motivating our decision to work with the GPMP2 motion planner, we will explore how Gaussian Processes can be used to generate continuous-time trajectories, how motion planning can be formulated as probabilistic inference, and how factor graphs can be used to perform probabilistic inference. With these points explored, we will have a solid understanding of how GPMP2 operates, making clear the challenges and contributions presented in this thesis. This section closely follows the outline derivations provided by [8, 57–59].

#### Motivation

As previously described, trajectory optimisation-based methods typically return trajectories faster than search-based and sampling-based methods, particularly when accounting for any post-processing needed to obtain smooth trajectories. Within the category of trajectory optimisation-based motion planners, GPMP2 offers state-of-the-art performance while also offering additional appealing attributes for our applications.

Firstly, in recent years there has been a greater interest in accounting for uncertainty across the fields of robotics and artificial intelligence (AI). This only seems natural since uncertainty is pervasive in the real world, from sensor measurements and predicted motions to the actuation of joints. Given our interest in dynamic environments with moving obstacles, being able to formally account for uncertainty is very powerful. The probabilistic framework on which GPMP2 is built enables us to explicitly model uncertainty through the use of covariance matrices.

In contrast to many motion planning approaches [36, 38, 39, 42], GPMP2 uses a continuous-time trajectory representation. This has the benefit of being able to query the trajectory at any time of interest and counter the computational cost associated with finely discretising a trajectory.

Lastly, due to the probabilistic inference formulation of the motion planning problem that Mukadam et al. adopt, they demonstrate that GPMP2 can be easily

extended for efficient re-planning (iGPMP2) and use efficient optimisation tools developed in the SLAM community [60, 61].

### Gaussian Process Models for Continuous-Time Trajectories

A Gaussian Process (GP) is a generalisation of the Gaussian probability distribution and is effectively a method of assigning a probability distribution over all possible functions that could be used to fit a selection of data points. This 'assignment' is achieved via a *prior* which embeds our 'prior beliefs' that we have about how the functions should behave. For example, we may wish to impose our belief that the function is 'smooth' or that it should vary on a specified timescale.

In motion planning, we are concerned with finding the states of our robot at given points in time so that they may be executed. The functions that we are interested in are thus functions that map time to robot states, i.e. $\boldsymbol{\theta}(t)$. We can consider trajectories as samples from a vector-valued GP where "a vector-valued GP is a collection of random variables, any finite number of which have a joint Gaussian distribution" [59].

We characterise a vector-valued GP by a vector-valued mean function, $\boldsymbol{\mu}(t)$, and a matrix-valued covariance function, $\boldsymbol{\mathcal{K}}(t, t')$. For a given selection of times $\boldsymbol{t} = [t_0, \ldots, t_N]$, we can form the mean vector and covariance kernel as:

$$\boldsymbol{\mu} = [\boldsymbol{\mu}_0 \ \ldots \ \boldsymbol{\mu}_N]^\top, \tag{2.5}$$

$$\boldsymbol{\mathcal{K}} = [\boldsymbol{\mathcal{K}}(t_i, t_j)]|_{ij, 0 \leq i,j \leq N}\,. \tag{2.6}$$

Given these functions, continuous-time trajectories can be sampled from the associated vector-valued GP:

$$\boldsymbol{\theta}(t) \sim GP(\boldsymbol{\mu}(t), \boldsymbol{\mathcal{K}}(t, t')), \tag{2.7}$$

where $\boldsymbol{\theta}$ has the joint Gaussian distribution $\boldsymbol{\theta} = [\boldsymbol{\theta_0} \ \ldots \ \boldsymbol{\theta_N}]^\top \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\mathcal{K}})$.

These $\boldsymbol{\theta_i} \in \mathbb{R}^D$ are the *support states* that parameterise the continuous-time trajectory $\boldsymbol{\theta}(t)$, where $D$ is the dimensionality of the robot's state space. In the context of our motion planning problem, these are the states, or timesteps, that we will evaluate and update in the optimisation process.

The GP defines a prior on the function space for our trajectory:

$$p(\boldsymbol{\theta}) \propto \exp\left\{ -\frac{1}{2} \left\| \boldsymbol{\theta} - \boldsymbol{\mu} \right\|_{\boldsymbol{\mathcal{K}}}^2 \right\}, \tag{2.8}$$

where $\left\| \boldsymbol{\theta} - \boldsymbol{\mu} \right\|_{\boldsymbol{\mathcal{K}}}^2 \doteq (\boldsymbol{\theta} - \boldsymbol{\mu})^\top \boldsymbol{\mathcal{K}}^{-1} (\boldsymbol{\theta} - \boldsymbol{\mu})$ is the Mahalanobis distance.

Intuitively, the covariance function embeds our constraints on the function space, such as how certain we are about points that the trajectory must pass through and how smoothly it should vary. The *prior* applies these beliefs and constraints to the space of functions from which we sample our trajectory. As we will see in the following subsection, this prior will contribute to our motion planning objective function as a cost functional and penalise trajectories that deviate from the mean.

### Motion Planning as Probabilistic Inference

As presented by Dong et al. [57], the motion planning problem can be viewed as an inference problem. To show this, we must consider our determination of the trajectory from a probabilistic perspective.

Qualitatively, we want to find the trajectory that is most likely to obey our constraints and beliefs about the trajectory. Having already discussed the GP prior, we now introduce the *likelihood*. The likelihood function specifies the conditional probability distribution of *events* being satisfied for a given variable. *Events* themselves are problem-specific – in the case of estimation problems, the likelihood function is used to encourage a posterior distribution of trajectories that satisfy measurements. In the motion planning context, we wish to encourage a posterior distribution that is collision-free. In other words, we want to solve for a trajectory, $\boldsymbol{\theta}$, that provides the most likely function given a collection of binary events, $\boldsymbol{e}$, that indicate whether the trajectory is collision-free at given timesteps.

We define the likelihood of a given trajectory satisfying the events, $\boldsymbol{e}$, or being collision-free, as $l(\boldsymbol{e}; \boldsymbol{\theta}) \propto p(\boldsymbol{e}|\boldsymbol{\theta})$. We define the likelihood as a distribution in the exponential family [57],

$$l(\boldsymbol{\theta}; \boldsymbol{e}) = \exp\left\{ -\frac{1}{2} \left\| h(\boldsymbol{\theta}; \boldsymbol{e}) \right\|_{\boldsymbol{\Sigma}}^2 \right\}, \tag{2.9}$$

where $h(\boldsymbol{\theta}; \boldsymbol{e})$ is a vector-valued cost function for the trajectory. In practice, this cost is determined in a similar manner to CHOMP whereby a robot is represented

by a set of spheres that approximate the body. For each timestep in the trajectory, $i$, a hinge loss cost is calculated for each of the spheres and collected in the vector cost $h(\boldsymbol{\theta_i})$. "$\boldsymbol{\Sigma}$ is a diagonal matrix and the hyperparameter of the distribution" [59].

Formally, we can compute a *posterior distribution*, $p(\boldsymbol{\theta}|\boldsymbol{e})$, from a likelihood and a prior, $p(\boldsymbol{\theta})$, using Bayes' rule:

$$p(\boldsymbol{\theta}|\boldsymbol{e}) = \frac{p(\boldsymbol{\theta})p(\boldsymbol{e}|\boldsymbol{\theta})}{p(\boldsymbol{e})} \propto p(\boldsymbol{\theta})p(\boldsymbol{e}|\boldsymbol{\theta}) \tag{2.10}$$

The optimal trajectory $\boldsymbol{\theta}^*$ is then given by the *maximum a posteriori* (MAP) estimator:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\boldsymbol{\theta}|\boldsymbol{e}) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\boldsymbol{\theta}|\boldsymbol{e}) l(\boldsymbol{e};\boldsymbol{\theta}) \tag{2.11}$$

By substituting our expressions for the prior (Eq. 2.8) and likelihood (Eq. 2.9) into Eq. 2.11, we can see the duality between inference and optimisation.

$$\begin{aligned} \boldsymbol{\theta}^* &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\boldsymbol{\theta}|\boldsymbol{e}) l(\boldsymbol{e};\boldsymbol{\theta}) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \Big\{ -\log\Big(p(\boldsymbol{\theta}|\boldsymbol{e}) l(\boldsymbol{e};\boldsymbol{\theta})\Big)\Big\} \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left\{ \frac{1}{2} \left\| \boldsymbol{\theta} - \boldsymbol{\mu} \right\|^2_{\boldsymbol{\mathcal{K}}} + \frac{1}{2} \left\| h(\boldsymbol{\theta};\boldsymbol{e}) \right\|^2_{\boldsymbol{\Sigma}} \right\} \end{aligned} \tag{2.12}$$

From Eq. 2.12, we can see that the inference problem is equivalently viewed as a non-linear least-squares optimisation problem. As a well-studied problem, there are many different methods to solve it; common iterative methods include Gauss-Newton and Levenberg-Marquardt, which solve with a quadratic convergence rate. While employing a fast-converging solver provides GPMP2 with a performance boost over its predecessor, GPMP, the real power of GPMP2 comes from carefully selecting the kernel used to derive the form of $\boldsymbol{\mu}(t)$ and $\boldsymbol{\mathcal{K}}(t, t')$.

Mukadam et al. use a "structured kernel generated by a linear time-varying stochastic differential equation" (LTV-SDE) [8]. The key result of this choice is that we obtain a Gauss-Markov model that "generates a GP with an exactly sparse tri-diagonal precision matrix (inverse kernel) that can be factored" [8]. Due to the resultant Markovian structure of the trajectory and sparsity of $\boldsymbol{\mathcal{K}}^{-1}$, we can factor the posterior distribution so that MAP inference can be equivalently performed by inference on a factor graph. Additionally, the Markovian structure results in a prior that encourages *smoothness*.

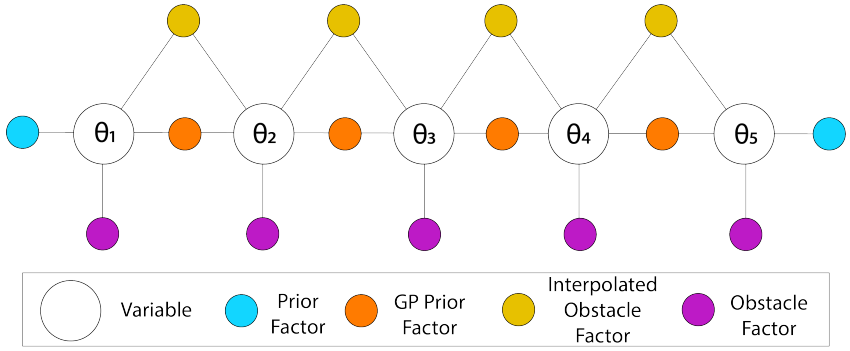



**Figure 2.5:** An example factor graph.

### Factor Graphs for Inference

Framing the motion planning problem as probabilistic inference has been explored in earlier works such as Approximate Inference Control (AICO) [62]. However, while AICO performs approximate inference using probabilistic tools such as Expectation Propagation [63], GPMP2 exploits the duality derived in Eq. 2.12 and uses efficient SLAM tools [64] to solve the resultant non-linear least-squares problems on *factor graphs*.

As described in [65], a *factor graph* is a bipartite graph that can be used to express how a global function, such as our desired trajectory, is the product of 'local' functions, such as local constraints between subsequent timesteps. A factor graph, $\mathcal{G} = \{\Theta, \mathcal{F}, \xi\}$, consists of *variables nodes*, $\theta_i \in \Theta$, *function nodes*, $f_i \in \mathcal{F}$, and a set of *connections/edges* between variable and function nodes, $\xi$. A variable node is used to represent each of the variables in the global function. Connections are made between variables nodes and function nodes; there is a function node for each *factor*, and intuitively, these nodes express how the connected variables behave together. An example factor graph is shown in Fig. 2.5.

**Factorising the Prior** As a result of the kernel choice made by Mukadam et al., i.e. a structured kernel generated by a LTV-SDE, the prior can be written as a product of factors,

$$p(\boldsymbol{\theta}) \propto f_0^p(\boldsymbol{\theta}_0) f_N^p(\boldsymbol{\theta}_N) \prod_{i=0}^{N-1} f_i^{gp}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{i+1}). \tag{2.13}$$

The factors $f_0^p(\boldsymbol{\theta}_0)$ and $f_N^p(\boldsymbol{\theta}_N)$ define prior distributions on the start and goal states of the trajectory respectively. $f_i^{gp}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{i+1})$ defines the GP prior factor between subsequent states.

**Factorising the Likelihood** The likelihood, $l(\boldsymbol{\theta}; \boldsymbol{e})$, which is responsible for collision avoidance, can similarly be factored as a product of obstacle factors for each state in the trajectory. Using the GP framework, it is efficient to perform interpolation (Gaussian Process Interpolation [57]) between states and evaluate collision costs at a finer resolution than chosen between support states; it is thus possible to assign collision costs to *interpolated states*. Accounting for interpolated states, the likelihood can be written as

$$l(\boldsymbol{\theta}; \boldsymbol{e}) = \prod_{i=0}^{N-1} \left\{ f_i^{obs}(\boldsymbol{\theta}_i) \prod_{j=1}^{n_{ip}} f_{\tau_j}^{intp}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{i+1}) \right\}. \tag{2.14}$$

We denote the number of interpolated states between subsequent support state pairs as $n_{ip}$. $\tau_j$ is the interpolated time for an interpolated state.

The regular obstacle factor assigns a unary cost to each variable node defined as

$$f_i^{obs}(\boldsymbol{\theta}_i) = \exp\left\{ -\frac{1}{2} \left\| \boldsymbol{h}(\boldsymbol{\theta}_i) \right\|^2_{\boldsymbol{\Sigma}_{obs}}, \right\} \tag{2.15}$$

where $\boldsymbol{h}(\boldsymbol{\theta}_i)$ is a $M$-dimensional vector-valued obstacle cost function, and $\boldsymbol{\Sigma}_{obs}$ is an $M \times M$ isotropic hyperparameter matrix given by $\boldsymbol{\Sigma_{obs}} = \boldsymbol{\sigma}_{obs}\boldsymbol{I}$

The interpolated obstacle factor is a binary obstacle cost that similarly describes the obstacle cost at an interpolated time, $\tau_j$, between consecutive support states $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i+1}$. The interpolated obstacle factor is defined as

$$f_{\tau_j}^{intp}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{i+1}) = \exp\left\{ -\frac{1}{2} \left\| \boldsymbol{h}_{\tau_j}^{intp}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{i+1}) \right\|^2_{\boldsymbol{\Sigma}_{obs}}, \right\} \tag{2.16}$$

The obstacle cost functions $\boldsymbol{h}(\boldsymbol{\theta}_i)$ and $\boldsymbol{h}_{\tau_j}^{intp}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_{i+1})$ allocate greater values when a state results in collision, or 'close' to collision with obstacles. $\boldsymbol{\sigma}_{obs}$ is a tuning parameter that determines how large the cost should be and thus the trade-off between collision avoidance and GP prior smoothness terms.

**Revisiting the Posterior Distribution** By comparing Equations 2.10, 2.13, and 2.14, we can see that in our motion planning problem, the posterior distribution can be factorised as

$$p(\boldsymbol{\theta}|\boldsymbol{e}) \propto \prod_{m=1}^{M} f_m(\Theta_m), \tag{2.17}$$

where M is the total number of function nodes and $\Theta_m$ is the variable node subset on which $f_m$ operates. All of the factors described above belong to the exponential family of distributions.

**Summary** We saw in Eq. 2.12 that there is a duality between viewing the motion planning problem as a non-linear least squares optimisation problem and as a probabilistic inference problem. Furthermore, we have seen that by carefully selecting the kernel used to generate our GP, the posterior distribution can be factorised (Eq. 2.17). This formulation enables GPMP2 to exploit the resultant sparsity in the problem with "sparse Cholesky decomposition followed by forward–backward passes" [8], to iteratively solve for the MAP trajectory, similar to AICO. In practice, GPMP2 is implemented using the Georgia Tech Smoothing and Mapping (GTSAM) C++ library [66], using SAM [64] to perform efficient inference on a factor graph.

## 2.2 Environment Representations

Following on from our discussion of GPMP2 and other motion planning algorithms, we will now explore the various choices of *environment representation* with which a motion planner can interact. An *environment representation* is a critical prerequisite for enabling collision avoidance. Whichever motion planning algorithm is chosen, in order to provide collision-free trajectories, it will require the ability to query a data structure that contains information about the environment and the locations of obstacles that it needs to avoid. Over the years, many such data structures have been developed to choose from. In this section, we describe some of the most common approaches and highlight their strengths and limitations. We will focus

only on discussing methods that are applicable in 3D environments, excluding representations such as elevation maps [67] and multi-level surface maps [68].

Finally, we will provide a short discussion of the environment representations used throughout the rest of this thesis – probabilistic voxelmaps and EDTs.

### 2.2.1 Occupancy Grids

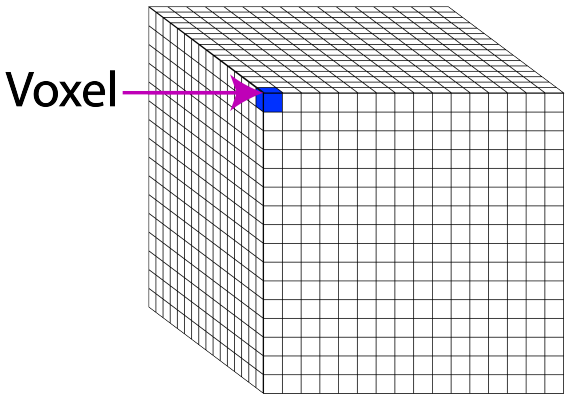



**Figure 2.6:** An example voxel grid – a regular grid structure of discrete 3D volumes called *voxels*.

The simplest approach one may think of when representing an environment is to discretise the workspace into a grid of cells in 2D, or cubes in 3D, to represent an area or volume of space; such a representation is shown in Fig. 2.6. We call these 3D cubes *voxels*. The data structure that we have described is thus commonly known as a *voxel grid*. While these voxel grids can comprise binary values to denote occupancy, a common approach is to accumulate a probability of each voxel being occupied, enabling incremental or Bayesian updates to incorporate new observations [69–71].

*Occupancy grids*, or *occupancy maps*, are widely used in 2D navigation and SLAM problems [72–76]. One of the main limitations of using fixed voxel grid structures becomes most notable when scaling up to higher dimensions. For 3D environments, a voxel grid requires a large allocation of memory; particularly for large outdoor environments, the representation is inefficient at exploiting sparsity in the environment and does not allow for finer resolutions in the areas of interest, i.e. around occupied regions of space.

Another limitation is that the size of the environment needs to be known and allocated in memory prior to mapping or one must incur a computational cost for copying voxel grids each time the grid is expanded [77].

### 2.2.2 Octrees

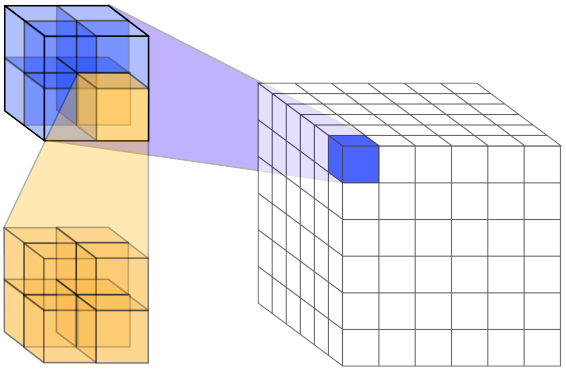


**Figure 2.7:** An example octree data structure in which a voxel can be recursively sub-divided.

To efficiently represent 3D objects up to a specified resolution, Meagher [78] showed that the memory required to represent an object was "on the order of the surface area of the object", and pioneered the technique of 'Octree Encoding' or *octrees*. Octrees are a hierarchical tree structure in which each node represents a voxel. "Each voxel is recursively subdivided into eight sub-volumes until" a specified voxel size is reached [77], as shown in Fig. 2.7. The hierarchical structure facilitates a smaller memory footprint because, for a given resolution, sparse volumes of the workspace can be represented with coarser resolutions.

Hornung et al. [77] built upon the concept of octrees and introduced a further method of compression to use probabilistic occupancy estimation while still maintaining compact 3D models. The resultant open-source library, *OctoMap*, has become a widely used tool within the robotics community, enabling large scale mapping reconstructions. Sodhi et al. [79] use a graph-based SLAM optimisation with an associated set of submaps to incrementally update a 3D global occupancy map based on OctoMap. Building on another octree-based framework, SuperEight [80], Vespa et al. [81] introduce a method of dynamically selecting the resolution at

which to best integrate new observations into a reconstruction and perform multi-scale dense SLAM.

A limitation of octrees, when compared to fixed-grid data structures, is the additional overhead in accessing the data; single, random query data access on an octree is of time complexity $\mathcal{O}(d) = \mathcal{O}(\log n)$ where $n$ is the number of nodes, and $d$ is the tree depth [77]. Furthermore, in GPU implementations the performance is inhibited because "tree traversal leads to thread divergence" [82].

### 2.2.3 Voxel Hashing

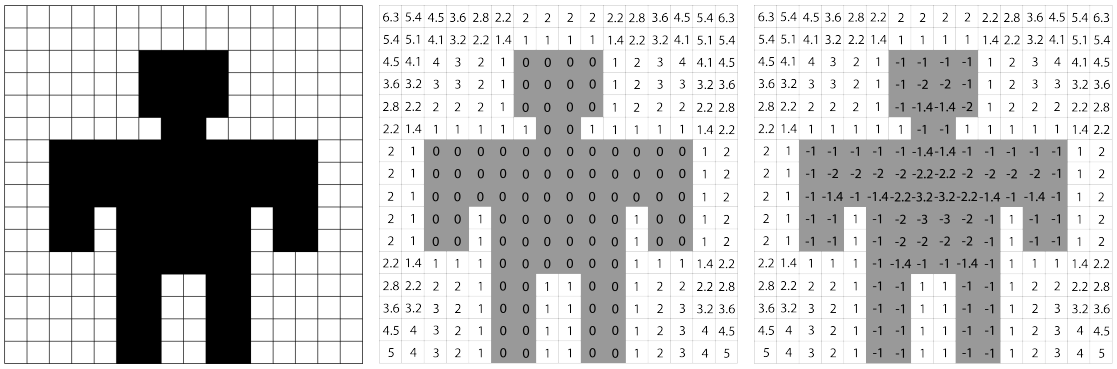
An alternative representation is that of *voxel hashing*. By only storing dense surface data in voxels where measurements are observed, spatial hashing provides a much more compact and memory-efficient representation than regular voxel grid structures, enabling finer-resolution and larger-scale environments to be reconstructed.

While previous works have explored spatial hashing techniques for rendering and collision detection [83, 84], Niesner et al. [82] describe the use of a spatial hashing scheme for online surface reconstruction. While retaining the ability to use efficient parallelised operations, their proposed approach removes the overhead associated with regular grids and hierarchical data structures. *Hash tables* also have the advantage of being an inherently dynamic data structure – the insertion or deletion of elements have an expected time complexity of $\mathcal{O}(1)$. However, as described by Alcantara et al. [84], it is often unnecessary to have a dynamic data structure on a GPU, and the table can be rebuilt from scratch in the event of any data items changing.

The hash table data structure proposed by Niesner et al. makes use of another environment representation – the Truncated Signed-Distance Field (TSDF). At this point, it is worth recalling that we require an environment representation not just to reconstruct the environment but as a means of enabling collision avoidance. Methods such as voxel grids, octrees, and hash tables can be used to provide occupancy data for queried points in space; this information is sufficient for sampling-based, search-based, and stochastic motion planners because it allows a motion planner to determine which states are valid and in the stochastic case

to derive stochastic gradients. However, in the case of optimisation-based motion planners that do not use stochastic gradients, such as GPMP2, the aforementioned data representations alone are not sufficient to guide the optimisation. This leads us to Euclidean Distance Transforms (EDTs).

### 2.2.4 Euclidean Distance Transforms



**(a)** Environment bitmap **(b)** EDT **(c)** SDF

**Figure 2.8:** A comparison of Euclidean Distance Transforms (EDTs) and Signed Distance Fields (SDFs) as derived from an example bitmap.

A 'distance transform' is an operation that can be applied to bitmaps to derive a map, whereby each pixel or voxel is mapped to the smallest distance to an obstacle. Although a number of different distance metrics can be applied, e.g. 'chessboard' and 'city block', the Euclidean distance is most relevant and commonly used in motion planning and navigation applications; we refer to these transforms as Euclidean Distance Transforms (EDTs).

Closely linked to the concept of an EDT is the Signed Distance Field (SDF) – an SDF is a resultant field that is obtained when the EDT of an inverted bitmap is subtracted from an EDT of the original bitmap. We illustrate this in Fig. 2.8c, and one can see that this process results in negative distance values being allocated within surface boundaries, rather than zero values, as is the case for an EDT.

Given an EDT, or SDF, calculated over a given workspace, gradients can be computed simply by a combination of tri-linear interpolation and finite differencing [85]. In the case of SDFs, the negative values enable valid gradients to also be obtained within surface boundaries.

The concepts of EDTs and SDFs have long been around in the computer graphics community [86–88]. For surface reconstruction, an isosurface can be extracted from the zero-point of the function using methods such as raycasting or a marching cubes algorithm [89].

While pioneering work [90] in artificial potential fields for obstacle avoidance relied on analytical equations for geometric shape primitives, the advent of CHOMP [38, 39] introduced Euclidean Signed Distance Fields (ESDFs) into the motion planning literature as a means of performing collision-checks and computing functional gradients of obstacle costs. Numerous motion planning applications have adopted this representation for obstacle avoidance [9, 40, 45, 91].

Besides the ease of calculating gradients, a benefit of using distance fields is that "collision-checks for robot links against the environment require constant time and are independent of the environment complexity" [36].

### Truncated Signed Distance Fields

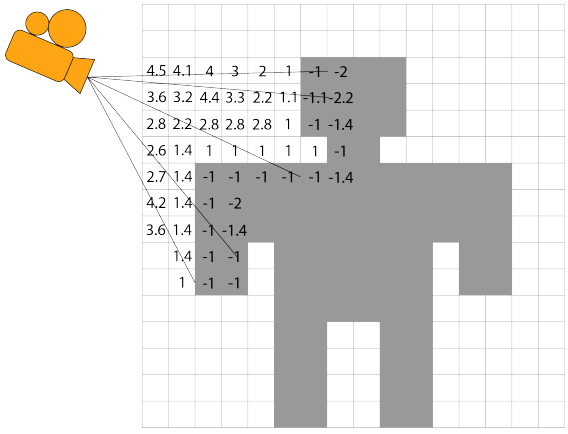



**Figure 2.9:** An example Truncated Signed Distance Field (TSDF).

In a seminal work by Curless and Levoy [92], they introduced the volumetric representation of 'cumulative weighted signed distance functions' in order to fuse range images and ultimately extract a surface manifold. Their method for incrementally integrating range images is now commonly called a Truncated Signed Distance Field (TSDF). In this variant of a signed distance field, voxel values are truncated to within a specified distance of the zero point, and the distances

themselves are from the viewpoint of the sensor rather than a global distance perspective. We illustrate this in Fig. 2.9.

The introduction of TSDFs was an essential precursor for paradigm-shifting dense SLAM reconstruction literature such as KinectFusion [93] and Kintinuous [94]. Using TSDFs as the primary form of environment representation, KinectFusion demonstrated the impressive reconstruction of 3D environments from RGB-D data in real-time. Similarly, Niesner et al. [82] perform 3D surface reconstruction by fusing TSDFs in a hash table structure.

For motion planning and collision avoidance, a TSDF representation itself does not supply the appropriate distance information. To leverage the advances made by KinectFusion, Wagner et al. [95] use the TSDF model generated by KinectFusion to build an ESDF that is used to compute obstacle avoidance costs for trajectory planning with CHOMP.

Rather than converting a full TSDF representation to an ESDF, Oleynikova et al. [96] proposed *Voxblox* to build ESDFs incrementally out of TSDFs. Another recent incremental ESDF mapping approach is FIESTA [91] which uses doubly-linked lists and a voxel-indexing framework to efficiently update an ESDF when new voxels become occupied. Both Voxblox and FIESTA are further explored in Chapter 4.

**Summary**

While there are numerous different data structures and representations that can be used to reconstruct our surroundings from sensor data, we must consider the requirements of our motion planning system. In Section 2.1.7, we described our motivation for building on GPMP2 as a motion planner. Within the space of trajectory optimisation-based motion planners, Euclidean distance fields provide a natural representation that can be used for both collision-detection querying and to obtain local gradients. As such, distance fields form a common theme of this thesis and will be further explored in later chapters.

## 2.3 Integrated Frameworks and State-of-the-Art

As seen in the preceding sections, the challenges of motion planning and efficiently representing the environment are well-studied in the literature. However, more often than not, these problems are considered in isolation. A dominant theme of this thesis is the importance of considering these two fields as interlinked in order to provide practical applications. While having a fast motion planner is essential for performing tasks in dynamic environments, it alone will not enable a robot to avoid fast approaching obstacles. As Kappler et al. [97] demonstrate and quantify, tight integration of real-time perception and reactive motion generation are critical. Particularly in dynamic manipulation scenarios, they show that having fast feedback is more beneficial than accurate representations of the environment. In this section, we review instances in the literature that demonstrate a tight integration of these two key areas.

Early work by Rusu et al. [98] integrates sampling-based motion re-planning and online-generated obstacle maps using a PR2 robot with 2D laser scans. In this work, the base and arm are considered separately, using a voxel-based collision map for 3D collision-checking. Kaelbling and Lozano-Pérez [99] similarly use a PR2 robot to perform complex mobile manipulation tasks. They use a hierarchical planning approach for task planning and perform motion planning with RRT, using an octree representation of the workspace to enable collision-checking and avoidance. In line with earlier works, Kaelbling and Lozano-Pérez focussed on accomplishing pick-and-place tasks without consideration for moving obstacles.

Over the years, hardware capabilities have increased since the PR2 robot was developed. In particular, low-cost depth cameras and GPUs have become more common-place.

Maier et al. [100] perform localisation and construct the scene representation in real-time to use for collision avoidance on a NAO humanoid robot. Their work integrated point cloud data from a Asus Xtion Pro Live depth camera into an octree used for path planning. However, this work does not extend to whole-body motion planning in 3D and does not consider the trajectories of moving obstacles.

Merkt et al. [101] demonstrate an integrated motion planning and perception framework for both fixed and floating-base platforms. This work uses an octree-based scene representation for motion planning while maintaining a dense TSDF-based voxelmap, as in Kintinuous [94], for accurate dense scene reconstruction. Using both sampling and optimisation-based planning algorithms [102], Merkt et al. generate collision-free trajectories for manipulation tasks. However, this framework does not demonstrate predictive or reactive behaviour in response to dynamic obstacles. Instead, execution is stopped if potential collisions are detected, and a new trajectory is planned.

Husain et al. [103] demonstrate real-time tracking and grasping of objects using dynamic motor primitives on a WAM robot arm.

As previously described in Section 2.1.5, Park et al. [9] consider fast motion re-planning in dynamic environments. However, they fall short of demonstrating ITOMP on real robots using live sensor data.

The incremental mapping frameworks of Voxblox [96] and FIESTA [91] were both integrated with motion planners and demonstrated as integrated frameworks; however, in both cases, they were used for 3D path planning of Mobile Aerial Robots (MAVs).

Of significant relevance to this thesis is the work by Hermann et al. and the development of GPU-Voxels [104–106]. Using GPU programming to massively parallelise the process of collision-detection, the authors present a unified framework for perception and motion planning – they demonstrate mobile manipulation on a 15-DoF robot by using the incremental search algorithm, D*-Lite, to re-plan trajectories online in response to changes in the observed environment. In later work, Juelg et al. [107] integrate an implementation of the Parallel Banding Algorithm (PBA) [108] to perform massively parallel computations of EDTs. They demonstrate the use of EDTs with a 3D wavefront planner for UAV navigation. While the authors note the potential of this work to be used in articulated robots, investigation of this was left for further work.

The GPU-Voxels framework, and its functionality for highly parallel EDT computations, offers a previously unexplored opportunity for trajectory optimisation-based motion planners to be brought to the forefront of enabling whole-body motion planning in dynamic environments in the presence of moving obstacles. As such, it forms a core building block for the work presented in this thesis.

## 2.4 Dynamic Obstacle Prediction

Recalling the goal that we set out in Chapter 1, this thesis aims to enable household robots to operate robustly even in the presence of humans. One of the critical difficulties that humans pose for robots is that they act as *dynamic obstacles*, i.e. objects that will not necessarily remain static over time and must be avoided to prevent collisions with the robot. To account for this, a recurrent theme within this thesis is a desire for robots to not only be reactive to changes in their surroundings but also to predict and anticipate the motions of dynamic obstacles.

The importance of motion prediction extends across multiple fields, from service robotics and self-driving vehicles to advanced surveillance systems [109]. For our particular interest in whole-body manipulators, being able to predict how dynamic obstacles move within a scene will enable us to achieve more efficient trajectories and improved collision avoidance; we will further discuss this in our work that we will present in Chapter 3. This section aims to provide the reader with an overview of the current state of 'dynamic obstacle prediction' and its integration with motion planning.

Over the past decade, in particular, there has been significant interest in applying motion prediction to autonomous road vehicle applications, concerning both predicting the behaviour of pedestrians [110–112] and the vehicles themselves [113–115]. Predicting the behaviour of other vehicles and pedestrians is vital for ensuring that autonomous vehicles operate safely; this remains a key challenge to be overcome to enable the wider adoption of autonomous vehicles on public roads.

While simple kinematic models, such as a constant velocity model, may readily be applied to inanimate moving objects [116], alternative methods are needed to predict trajectories that capture the complexity of human behaviour [6]. Human motion

prediction methods generally fall into three categories; physics-based, pattern-based and planning-based methods [109]. In early works, human trajectory forecasting was performed using hand-crafted methods based on domain knowledge. Helbing and Molnár [117] presented their seminal work on the Social Force model – a physics-based model that uses attractive and repulsive forces to account for both social contexts as well as physical interactions such as obstacles. Elfring et al. [118] built upon this concept and combined it with intention estimation using a hidden Markov model, enabling them to greatly improve the prediction accuracy for human motion. Numerous extensions to the social forces model have since been proposed [119–122]. While social forces are focused on human-human interactions, pioneering work by Kitani et al. [123] demonstrated the importance of understanding the semantics of a scene for predicting human behaviours. Ballan et al. [124] extract semantic context descriptors for outdoor environments and use these features with a Dynamic Bayesian Network to predict trajectories for multiple agent classes, such as pedestrians and cyclists. One key difficulty with physics-based methods [125, 126] is that the parameters are highly dependent on prior knowledge and cannot capture the complexity of all types of pedestrian behaviour [127].

By contrast, pattern-based methods use data to train models that accurately describe human behaviours [128–131]. Such models have become dominant in recent years due to their performance for long-term prediction in complex, semantically-rich environments. For example, recent work by Cao et al. [132] incorporates semantic information via a three-stage deep learning-based framework. They use trained neural networks to extract the scene context from RGD images and exploit it to predict a human's intended goal; this goal is then used to predict the human's 3D path. Other deep learning approaches have also been applied to trajectory prediction, particularly with the numerous large driving [133–135] and pedestrian [136–138] datasets that have been developed in recent years. Mozaffari et al. [115] provide a comprehensive review on this topic and discuss the most recent deep learning approaches for predicting the behaviour of autonomous vehicles. For example, Wulfmeier et al. [139] use a human-driving dataset and Inverse Reinforcement Learning to train neural networks that model an expert's driving behaviour and

extract learned cost maps. However, few existing datasets capture human motions in a typical indoor environment like an office or home environment. The THÖR dataset [140] uses a stationary 3D LiDAR scanner and a motion capture system to record human trajectories within a large laboratory. More recently, Kratzer et al. [141] proposed the MoGaze dataset, which includes full-body motion capture data, eye-gaze tracking, and 3D models of the workspace.

Planning-based approaches work similarly to motion planning itself. By assuming that a person intends to achieve a specific goal state and will move according to certain constraints, such as avoiding obstacles, planning-based approaches will employ a planner to determine a person's predicted movements. Numerous works follow this approach [142–146] and they tend to offer good performance for long-term prediction.

### 2.4.1 Motion Planning with Prediction

For the aims of this thesis, we are keenly interested in the intersection of human trajectory prediction and its integration with motion planning. Various works have investigated this in the context of autonomous vehicles. Albrecht et al. [147] use rational inverse planning [148] to determine the goals of other vehicles and subsequently determine a vehicle's planned manoeuvres using a Monte Carlo Tree Search (MCTS) algorithm. Eiras et al. [149] demonstrate an effective motion planning framework for simulated urban environments that uses a two-stage optimisation method; mixed-integer linear programming (MILP) is used to enforce hard constraints on the motion planning problem, which then initialises a second non-linear programming (NLP) stage.

Kruse et al. [150] and Möller et al. [151] provide comprehensive reviews on human-aware robot navigation, highlighting the vast literature base on the topic. However, few works have demonstrated frameworks that integrate high-dimensional motion planning, such as articulated robots, with human trajectory prediction. Mainprice et al. [152] use STOMP [42] in simulation experiments with predicted human motions from a learnt set. Park et al. [153] also leverage offline learning

of human motions in their proposed motion planner, I-Planner, to avoid human trajectories in the local workspace of a 7-DoF robot arm.

## 2.5 Active Perception for Collision Avoidance

This section provides the reader with a brief literature review on active perception approaches for collision avoidance. We define 'active perception' generally as the process of modelling and developing control strategies that further enhance the acquisition of sensor data about the environment.

The vast majority of literature that interlinks motion planning with active perception is focused on variants of the 'Next Best View' (NBV) problem [154]. In this problem, the challenge is to determine the next viewpoint for a camera sensor to best observe an object [155–157] or scene [158–160]. Being able to address this problem is important for generating fast and accurate reconstructions of objects and scenes. Particularly in recent years, there has been significant interest in using UAVs to generate extensive reconstructions, necessitating the optimisation of trajectories to minimise the path length while ensuring sufficient or optimal coverage of the reconstruction target.

Alternative utility functions and applications have also been considered for active perception. Nunez-Varela et al. [161] consider gaze allocation during visually-guided manipulation tasks. In this work, they consider where a humanoid robot's gaze should be directed in order to best estimate the positions of objects, given a set of landmarks, such as objects on a table. Fox et al. [162] propose an 'active localisation' approach based on Markov localisation which determines the best directions of a robot's sensors to localise the robot. Costante et al. [163] argue that for robots with 'controllable vision', motion planning should be performed in a perception-aware manner to favour the observance of texture-rich areas in the workspace. By observing texture-rich areas through the incorporation of photometric information about the scene, they demonstrate an improvement in localisation accuracy compared to standard path-planning approaches. Lidoris et al. [164] similarly aim to minimise localisation and map estimation errors by combining trajectory planning and with active gaze direction.

In this thesis, we are predominantly concerned with being able to perform collision-free trajectories and as such require effective collision avoidance. Our ability to avoid obstacles is inextricably linked to our ability to perceive the obstacles we need to avoid. As we will see in Chapter 5, we thus believe that there is a strong motivation for using active gaze to provide effective collision avoidance. Unfortunately, we believe this area is currently lacking in the literature, ultimately motivating our contributions presented in Chapter 5.

Work by Seara et al. [165] is one of the few that considers active gaze control for a vision-guided humanoid in a collision-avoidance context; however, they consider only objects that are already known and aim to maximise the predicted information gain for their positions to maintain view of them in the scene. Droeschel [166] propose a gaze control strategy that maintains a gaze direction in the direction of travel unless objects are identified within the camera's field of view. If these objects are within a distance threshold from the robot and have not been recently re-observed, the robot's gaze is directed towards the object's last known position. Such a strategy is likely to perform poorly when the robot performs complex motions such as following circular paths around a corner or obstacle since the gaze will be in a direction that is tangential to the trajectory followed.

## 2.6 Toyota Human Support Robot

The primary robot that motivates the research in this thesis, and is used for hardware experiments, is the Toyota Human Support Robot (HSR). The HSR was developed to address the increasing demand for solutions to long-term elderly care in countries with ageing populations [7]. As such, the robot incorporates a number of design features that enable it to perform loco-manipulation tasks within an indoor environment. In this section, we describe the robot's specifications and their relevance to the research presented in this thesis.

Working our way from the ground level, the mobile base of the HSR uses two drive wheels in conjunction with three casters to enable smooth holonomic motion via a differential drive system. The resultant omni-directional motion of

the base is an important feature for enabling dexterous behaviours, particularly due to the construction of the robotic arm.

The articulated arm has five degrees of freedom: a linear torso lift joint followed by a series of four revolute joints (arm flexion, arm roll, wrist flexion, and wrist roll). Attached to the end of the arm is a two-fingered gripper with an in-built RGB camera that acts as the typical end-effector. The gripper is force-compliant and can exert varying levels of force dependent on the commands given.

On the topic of sensing, a laser range sensor (UST-20LX) is mounted onto the front of the robot's base, enabling 2D environment mapping and the detection of obstacles. Mounted onto the body of the robot is a 2-DoF head (tilt and pan) on which there are three camera sensors: RGB-D Asus Xtion Pro Live, a stereo camera and a wide-angle camera. Built into the body is a 6-DoF inertial measurement unit (IMU).

In this research, we focus on the head-mounted RGB-D sensor to provide environment perception while performing loco-manipulation tasks with the 3-DoF base and 5-DoF arm.

Regarding compute capabilities, the robot has an on-board $4^{\text{th}}$ Gen Intel Core i7 CPU, 16 GB RAM, 256 GB SSD, and an NVIDIA Jetson TK1 embedded GPU.

The HSR weighs approximately 37 kg and, due to the linear torso joint, has a height range to the top of the head of 1005 mm to 1350 mm.

# 3

# Predicted Composite Distance Fields

In the first contribution of this thesis, we extend a state-of-the-art trajectory optimisation method, GPMP2, to perform motion planning and account for the predicted motion of moving obstacles. As discussed in Section 2.2.4, signed distance fields are a natural, and thus commonly used, method of environment representation for use with trajectory optimisation-based motion planners. However, to account for dynamic obstacles, prior work has resorted to alternative methods of allocating costs due to the computational cost associated with calculating signed distance fields [9].

In this chapter, we introduce the concept of *predicted composite distance fields* (PCDFs) as a method of generating signed distance fields faster than computing them from scratch when considering dynamic obstacles. We show that our technique provides a substantial speed-up compared to computations performed from scratch. We further present a novel framework for motion planning in dynamic environments that uses composite signed-distance fields to embed the predicted trajectories of moving obstacles into the motion planning problem. To verify our approach, we perform simulation and hardware experiments in which our method enables a 7-DoF Panda arm to avoid a moving obstacle.

The following manuscript was presented at The International Conference on Automated Planning and Scheduling, 2021 [3] along with a supplementary video[1].

[1] `https://www.youtube.com/watch?v=vJH8qBRMedw`

# Predicted Composite Signed-Distance Fields for Real-Time Motion Planning in Dynamic Environments

**Mark Nicholas Finean, Wolfgang Merkt, and Ioannis Havoutis**
Oxford Robotics Institute, University of Oxford
{ mfinean, wolfgang, ioannis }@robots.ox.ac.uk

**Abstract**

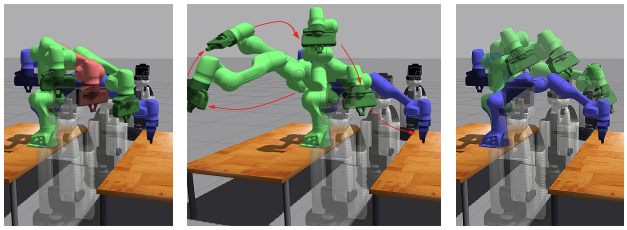
We present a novel framework for motion planning in dynamic environments that accounts for the predicted trajectories of moving objects. We explore the use of *composite signed-distance fields* in motion planning and detail how they can be used to generate signed-distance fields (SDFs) in real-time to incorporate predicted obstacle motions; to achieve this, we introduce the concept of *predicted signed-distance fields*. We benchmark our approach of using composite SDFs against performing exact SDF calculations on the workspace occupancy grid. Our proposed technique generates predictions substantially faster and typically exhibits an 81–97% reduction in time for subsequent predictions. We integrate our framework with GPMP2 to demonstrate a full implementation of our approach in real-time, enabling a 7-DoF Panda manipulator to smoothly avoid a moving obstacle in simulation and hardware experiments.

## 1 Introduction

To integrate autonomous systems into our daily lives, we need to ensure that they operate safely in the dynamic world around them. We must, therefore, develop robots that can adapt their movements as necessary to avoid moving obstacles. This holds whether we consider robots in a household environment avoiding humans and pets, autonomous vehicles on a road avoiding other cars and pedestrians, or robots operating in a dynamic industrial setting.

There has been significant research into motion planning; however, the current ability of robots to react in dynamic environments, with moving obstacles, is still lacking. Many approaches rely on fast re-optimisation to account for changes in the environment, yet do not account for the predicted motions of the obstacles. By neglecting predicted motions, robots encounter failure cases, as depicted in Figure 2, where depending on the speed of the obstacle, the robot repeatedly plans to move in front of the moving obstacle. This scenario occurs because the optimisation can become trapped in a local minimum from which it cannot escape since it does not have full information of the obstacle trajectory during the early iterations. The resultant trajectory is both sub-optimal and potentially dangerous. In this paper, we show that the solution to this is to incorporate predicted obstacle motions into the planning problem.

Copyright © 2021, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.



(a) (b) (c)

Figure 1: Real-time collision avoidance in a dynamic environment using our proposed framework. The 7-DoF Panda arm is performing a pick-up task, positioning the end-effector over a gap onto another table. During execution, a Toyota HSR travels along the gap, acting as a moving obstacle. Poses are shown in blue (start and goal configurations), green (collision-free), or red (collision). 1a) Without environment updates during execution. 1b) With environment updates and trajectory re-optimisation during execution. 1c) Our method - including obstacle trajectory prediction. We incorporate ESDF predictions which are generated at 400 Hz. The update loop runs at 3.7 Hz at the start and 10 Hz for the remainder of the trajectory.

Motion planners that would otherwise use signed-distance fields are forced to adopt alternative methods, such as binary collision costs (Park, Pan, and Manocha 2012), to incorporate predictions. This is because the computation time for calculating SDFs prohibits real-time usage—i.e., the ability to generate predictions at a much higher frequency than environment observations are made, typically 30 Hz. Our results motivate using predictions and show that we can generate real-time predictions for signed-distance fields by using *composite signed-distance fields*.

We integrate our method of composite SDF predictions into our novel framework which uses the GPMP2 motion planner (Mukadam et al. 2018) to solve for collision-free trajectories in environments with moving obstacles.

The key contributions of this paper are:

1. Introduction of *predicted signed-distance fields* for mo-

tion planning in the presence of moving obstacles.
2. Extension of GPMP2 to plan using predicted obstacle trajectories.
3. A novel framework for motion planning in dynamic environments using composite signed-distance fields.

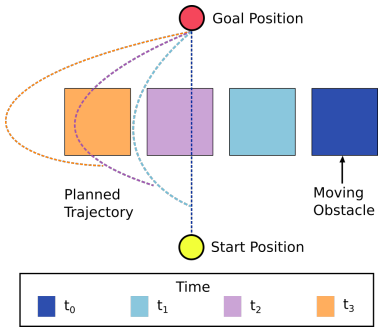



Figure 2: Toy example showing the inherent limitation of the execute-and-update method common in trajectory optimisation: The task is to plan a collision-free trajectory to reach the goal state in the presence of moving obstacles. During execution, we continue to observe the latest position of the moving obstacle and update the planned trajectory. As an obstacle approaches the intersection with the current planned trajectory, this repeated optimisation can result in the trajectory plan chasing in front of the obstacle.

## 2 Related Work

Motion planning in the presence of moving obstacles is an open problem, while solutions generally fall into two categories. The first is to assume full prior knowledge of the moving obstacle trajectories in the scene (Menon, Cohen, and Likhachev 2014; Hsu et al. 2002; Merkt, Ivan, and Vijayakumar 2019). The second approach is 'continuous replanning' in which the motion planner either re-optimises and adapts the current planned trajectory, or considers multiple trajectory modes at any one time, such as in ITOMP (Park, Pan, and Manocha 2012), and smoothly switches between them as new information is provided (Kolur et al. 2019; Zucker et al. 2013). However, there have been few works that incorporate predicted obstacle motions into the motion planning of articulated systems.

Nam *et al.* model obstacle motions as a random walk process and assign an artificial potential based on the probability density function of their predicted positions (Yun Seok Nam, Bum Hee Lee, and Moon Sang Kim 1996). Park *et al.* proposed IAMP which integrates ITOMP with human motion prediction based on supervised learning (Park, Park, and Manocha 2019). Li and Shah implement a probabilistic roadmap approach in which the roadmap is constructed using an obstacle motion prediction, essentially assuming a prior knowledge of the trajectory (Li and Shah 2019).

To further explore this problem, we first consider the environment representations used in motion planners.

Binary occupancy information enables search-based methods such as A* and D*-Lite, as well as sampling-based methods like Probabilistic Road Maps (PRMs) and Rapidly-exploring Random Trees (RRTs), to perform collision avoidance (Hart, Nilsson, and Raphael 1968; Koenig and Likhachev 2002; Kavraki et al. 1996; Kavraki, Kolountzakis, and Latombe 1998; LaValle 1998; Kuffner and LaValle 2000). In contrast, continuous optimisation-based approaches additionally require gradients so commonly use a Euclidean distance transform (EDT), or Euclidean signed-distance field (ESDF), to represent the environment.

When planning in 3D space, the environment is most often split into an array of discrete volume elements. We define a 'voxel' to represent a single discrete volume in space. An ESDF is a 3D voxelgrid in which each cell contains the signed-distance to the closest obstacle surface. The sign denotes whether the cell is within or outside the surface boundary. In addition to providing distance information which can be used to assign collisions costs, tri-linear interpolation enables us to obtain gradients.

Motion planners such as CHOMP, TrajOpt, and GPMP2 use ESDFs to represent the environment and demonstrate them to be very effective in enabling fast planning in static environments (Ratliff et al. 2009; Schulman et al. 2014; Mukadam et al. 2018). However, the limitations of SDFs become apparent when we look to apply them in dynamic environments.

ITOMP considered obstacle trajectory prediction for use in dynamic environments. However, in contrast to their general approach for static collision costs, which uses EDTs, they implement a simple occupancy cost for predicted motions based on "geometric collision detection between the robot and moving obstacles" (Park, Pan, and Manocha 2012). At each time-step in the optimisation a binary cost is allocated for every moving obstacle that is in collision. The reason for their approach is that EDTs are commonly considered to have CPU compute times that are not fast enough for real-time performance, thus are pre-computed and assumed to be static.

Various methods have been proposed to reduce the compute time for exact EDTs. Maurer *et al.* show that an exact distance transform can be calculated in linear time (Maurer, Qi, and Raghavan 2003). Approximate methods, such as the Chamfer Distance Transform (CDT) and Fast Marching Method (FMM), have also been proposed (Jones, Baerentzen, and Sramek 2006) to achieve fast approximations. In robotics, we often accumulate information over time; this is in contrast to areas such as medical image processing, where an ESDF will be constructed from all of the information available. This has lead to significant work being conducted into faster integration of new occupancy information into signed-distance fields. Oleynikova *et al.* found that Truncated Signed-Distance Fields (TSDFs) are faster to construct than Octomaps, a widely used package that maps the environment using octrees (Hornung et al. 2013), and proposed, VoxBlox, a method to build an ESDF from projective TSDFs incrementally (Oleynikova et al. 2017). In contrast to ESDFs, TSDFs use a 'projected distance' metric and consider distances only within a truncated radius of surface boundaries. TSDFs are commonly used in vision for surface reconstruction (Newcombe et al. 2011; Whelan, Kaess, and

Fallon 2012).

Despite the speed-ups achieved by state-of-the-art methods, they are not able to calculate SDFs fast enough for use in a real-time motion planner that uses what we term *'predicted signed-distance fields'* - predictions of what the ESDF of the workspace will be for future times based on estimates of obstacle geometry, velocity, and direction. We propose using *composite signed-distance fields* to generate predicted signed-distance fields in real-time.

Zucker *et al.* first mentioned the concept of composite SDFs, noting that distance fields are compositional under the `min` operation (Zucker et al. 2013). Thus the distance field computation can be reduced to a minimisation across a set of pre-computed distance field primitives. In the computer graphics community, composition of object-centric SDFs is used to unionise objects in 'Constructive Solid Geometry' as well as to combine the SDFs of individual objects into a single SDF of the scene for ray-marching (Reiner, Mückl, and Dachsbacher 2011).

Zucker *et al.* use composite SDFs in 'CHOMP-R' to quickly update their environment representation in response to a changing environment. However, to our knowledge, they have not been used in a predictive manner. We provide further analysis of this technique and leverage it in our novel framework which extends it to the domain of integrating predicted object trajectories into motion planning.

To demonstrate our framework, we integrate it with GPMP2 which formulates motion planning as probabilistic inference on a factor graph and uses Gaussian Processes (GPs) to represent continuous-time trajectories with a small number of states (Mukadam et al. 2018). Trajectories are found by using the GTSAM framework (Dellaert 2012) to exploit sparsity in the problem and perform fast probabilistic inference on factor graphs. Collision avoidance is applied via the use of 'obstacle factors', which evaluate the collision-free likelihood, using a hinge-loss obstacle cost for each time-step. The hinge-loss is calculated, in both CHOMP and GPMP2, by using an approximate robot model, comprised of spheres, to query the signed-distance field (Ratliff et al. 2009; Mukadam et al. 2018). However, Mukadam *et al.* use a single pre-computed SDF of the surrounding environment that is shared by all of the obstacle factors (Mukadam et al. 2018) and so does not incorporate the dynamics of moving obstacles. Kolur et al. consider dynamic environments but update all of the obstacle factors to use the same updated SDF and then re-optimise to solve for an updated trajectory plan (Kolur et al. 2019).

# 3 Methods

Part 3.1 describes our method of predicting trajectories for dynamic obstacles. Part 3.2 details our novel framework for predicting environment representations, using composite signed-distance fields, and then outlines how they can be integrated with a motion planner. We describe an experiment to examine the speed-up attained by using composite signed-distance fields in contrast to exact computations. In Part 3.3, we describe how we adapted the GPMP2 motion planner to react in dynamic environments as well as an experiment which demonstrates its applicability. In Parts 3.4 and 3.5, we demonstrate our fully integrated framework in simulation and on hardware, respectively. In both, a 7-DoF Panda manipulator successfully avoids a moving obstacle.

## 3.1 Obstacle Trajectory Prediction

To demonstrate that our approach enables the incorporation of obstacle trajectory predictions, we consider a simple prediction strategy for the occupancy of the workspace. Consider a 3D workspace, $\mathcal{W}$, which comprises of a set of $i = 1, \ldots, n$ stationary objects, $\mathcal{O}_i^s$, and $j = 1, \ldots, m$ moving (dynamic) objects, $\mathcal{O}_j^d$. At a given time, $t$, a moving object with a centroid position, $\mathbf{x}_j(t)$, will travel at velocity $\mathbf{v}_j(t)$. We write the collection of discretised positions and velocities as $\mathbf{X}(t) = [\mathbf{x}_1(t), \mathbf{x}_2(t), \ldots, \mathbf{x}_m(t)]^T$ and $\mathbf{V}(t) = [\mathbf{v}_1(t), \mathbf{v}_2(t), \ldots, \mathbf{v}_m(t)]^T$.

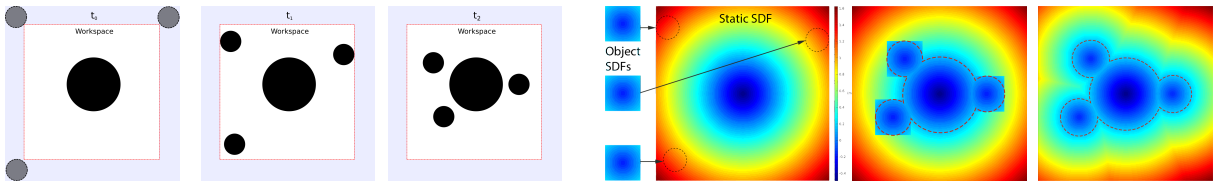
In our simulated experiments, we assume noiseless sensor data and subsequently an accurate occupancy grid of the workspace, $\mathcal{G}(t)$, at each time-step. At any given time, we can identify and isolate occupied regions of the workspace. From these regions, we calculate the centroid and list of voxels associated with each object. By comparing two successive frames, at times $t$ and $t + \delta t$, we identify which objects are moving and estimate their respective velocities from the motion of each centroid—this enables us to identify occupancy grids associated with the static scene, $\mathcal{G}_{static}$, and the moving obstacles, $\mathcal{G}_{moving}(t)$.

With centroid and velocity information, we propagate each object using a constant-velocity model to produce a predicted trajectory of the centroid. This method inherently gives accurate trajectory predictions when an object is moving in a straight line with constant-velocity. However, we emphasise that the prediction model is fully replaceable and that a more sophisticated object tracking and trajectory prediction can be implemented within our framework, e.g. a KLT tracker or Unscented Kalman Filter (Jianbo Shi and Tomasi 1994; Raj et al. 2016).

## 3.2 Composite Signed-Distance Field Generation

Building on the previous description of how to identify the static and moving parts of the scene, we use $\mathcal{G}_{static}$ to calculate the static SDF $\mathcal{S}_{static}$. Next, we consider an occupancy box, $\mathcal{B}_j$, around each of the $m$ moving objects, $\mathcal{O}_j^d$, that we have identified in $G_{moving}$. Each occupancy box is chosen to be of equal size to $\mathcal{O}_j^d$, plus an additional $\epsilon$ in all directions. For each $\mathcal{B}_j$, we calculate the corresponding object SDF, $\mathcal{S}_j$. The resultant SDF is the smallest cuboid which encloses all voxels in 3D space with signed-distance values equal to or less than a distance $\epsilon$ away from the surface of the object. We introduce $\epsilon$ as a safety margin desired to be penalised in collision avoidance. When this method is integrated with a motion planner, $\epsilon$ should equal the desired safety margin around objects to be penalised plus the size of the largest collision sphere associated with the approximate robot model.

Similar to $\mathbf{X}(t)$ and $\mathbf{V}(t)$, we denote the stacked vector of moving SDFs at a given time as $\boldsymbol{\mathcal{S}}(t)$. In practice, we may accumulate more information about the shape of each moving object, hence $\boldsymbol{\mathcal{S}}(t)$ is a function of time. However, for demonstration, our simulations assume noiseless sensor




(a) (b) (c) (d) (e) (f)

Figure 3: 3a, 3b and 3c depict a toy example of three moving spheres entering the workspace in which a static obstacle (large central sphere) is present. White regions in the workspace represent free space, while black regions are occupied. 3d shows an SDF of the static environment, with the tracked objects in 3b overlaid for illustration purposes. For each of the tracked objects in the scene, an SDF is calculated and associated with them. By tracking the positions of the moving objects, we infer their velocities in order to make predictions of their future positions. At these future positions, we can superpose the object SDFs onto the static SDF using a `min` operation—the result is a composite SDF. 3e shows a composite SDF for $t_2$. For comparison, 3f shows the corresponding exact SDF. Critically for motion planning, the two are identical for distances up to $\epsilon$ away from the obstacle surface boundaries, as indicated by the red dashed line.

**Algorithm 1** Real-time Update Loop

**Usage:** Continuously updates the motion planner

*Initialisation* :

$\boldsymbol{\mathcal{S}} = getObjectSDFs()$
$\mathcal{S}_{static} = getStaticSDF()$

*Execution :*

**while** not shutdown **do**
  $\boldsymbol{\mathcal{S}} = updateObjectSDFs()$
  $\mathbf{X} = updateObjectPositions()$
  $\mathbf{V} = updateObjectVelocities()$
  **for** $t$ in $times_to_predict$ **do**
    $\mathbf{X}_{predicted} = calcFutureObjPositions(t)$
    $\mathcal{S}_{predicted} = predictSDF(t, \mathbf{X}, \mathbf{V}, \boldsymbol{\mathcal{S}}, \mathcal{S}_{static})$
    $updateMotionPlanner(t, \mathcal{S}_{predicted})$
  **end for**
  $reoptimiseAndSendTrajectory()$
**end while**

**Algorithm 2** $predictSDF(t, \mathbf{X}, \mathbf{V}, \boldsymbol{\mathcal{S}}, \mathcal{S}_{static})$

**Output:** $\mathcal{S}_{predicted}$

$\mathcal{S}_{predicted} = \mathcal{S}_{static}$
**for** $j = 1 : m$ **do**
  $inds = getOccupiedIndices(t, \mathbf{x}_j, \mathbf{v}_j)$
  $\mathcal{S}_{predicted}(inds) = min(\mathcal{S}_{predicted}, \mathcal{S}_j)$
**end for**
**return** $S_{predicted}$

data and so do not acquire additional shape information in subsequent steps; we thus refer to $\boldsymbol{\mathcal{S}}(t)$ simply as $\boldsymbol{\mathcal{S}}$.

For any future time, $t_f$, we can predict the occupancy of the workspace by propagating the current position $\mathbf{x}_j(t_n)$ of each moving obstacle SDF $\mathcal{S}_j$ to its predicted position using velocity $\mathbf{v}_j(t_n)$. Similar to considering an occupancy grid as the superposition of all occupied regions, we consider the workspace SDF as the superposition of all object SDFs in the scene—this forms the basis of a composite SDF.

To calculate the resultant composite SDF for a given time, we take the minimum voxel value (i.e., signed-distance) of all overlapping scene and object SDFs. By construction, the composite SDF is guaranteed to be accurate up to the distance $\epsilon$. We illustrate this process in Figure 3.

Algorithms 1 and 2 describe how we use this method in an update loop for motion planning with SDF predictions generated in real-time. While obstacle prediction in motion planning is not novel, to our knowledge this has not been performed by predicting the SDF of the environment for different time-steps to solve for a trajectory. Our approach thus relates to time-configuration space planning, explicitly encoding the state of the environment at different points in time (Merkt, Ivan, and Vijayakumar 2019).

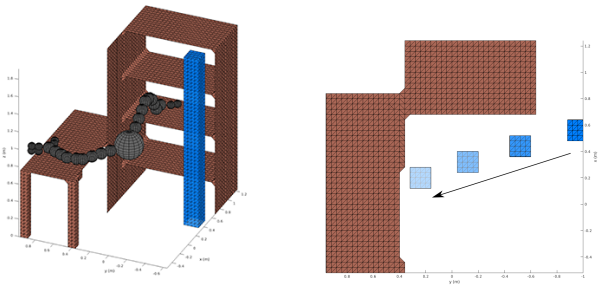
To evaluate the speed-up achieved using composite SDFs, we constructed a set of four dynamic datasets modelled on the 'table and cabinet' setup presented in (Mukadam, Yan, and Boots 2016). Each of the environments contains a table and cabinet with the addition of: a) one small moving box, b) two small moving boxes, c) one moving pillar, d) two moving pillars. Figure 4 shows aerial and three-quarter perspective views of the 'single moving pillar' setup. We also include a simpler environment consisting of an empty workspace with a large moving block.

For each dynamic environment, we simulate 31 different time-steps in a $96 \times 96 \times 96$ workspace with $4\,\mathrm{cm}$ resolution. At each time-step, we produce composite SDF predictions for the remaining time-steps. We perform the same simulation and benchmark against the exact SDF computations as calculated in GPMP2, using MATLAB's `bwdist` function, on the predicted occupancy grids.

### 3.3 Dynamic GPMP2

We adapt the original GPMP2 implementation to enable rapid updates of the SDF associated with each obstacle factor *independently*. The process of quickly updating obstacle factors facilitates a motion planner that can re-optimise and adapt planned trajectories to a changing environment.

To examine the effect of using independent SDFs for each obstacle factor, and to assess the importance of incorporat-



(a) (b)

Figure 4: 'Single moving pillar' environment - Aerial and three-quarter perspective views of an example environment in which a tall pillar traverses the floor. Example start and goal configurations are depicted using a sphere approximation of the WAM robot.

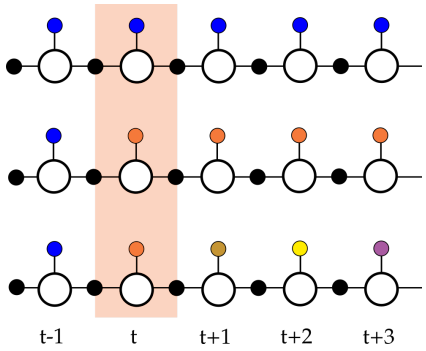



Figure 5: Factor graphs associated with three different feedback scenarios. Variable nodes are shown in white, prior factors in black, and coloured obstacle factors. *Top* - the obstacle factors share the same SDF and remain unchanged during execution. *Middle* - as the trajectory is executed, the current and all future obstacle factors are replaced with the latest SDF. *Bottom* - as the trajectory is executed, the current obstacle factor is updated to the latest observed SDF while all future obstacle factors receive updated SDF predictions.

ing obstacle dynamics, we considered and compared three different scenarios in dynamic environments. Each scenario considers a different degree of perception; they are as follows:

1. *Static:* Assumes a static workspace and all obstacle factors remain unchanged throughout the execution of the trajectory. The factor graph is thus optimised only once.
2. *Execute-and-update:* Obstacles factors are initialised with the initial SDF of the scene. During execution, at each time-step, all obstacle factors with a time-index greater than or equal to the current time are updated to use the current observation of the workspace. The trajectory is re-optimised and executed at each time-step in an iterative process.
3. *Full prior knowledge:* Each obstacle factor is provided with the known SDF of the environment at the associated time. The factor graph has maximum knowledge of the environment evolution and is thus only optimised once.

The factor graphs associated with each of the perception scenarios are illustrated in Figure 5.

To test each scenario, we used a 7-DoF WAM robot arm in simulation to plan trajectories in the 'single moving pillar' setup shown in Figure 4. Plans were conducted over seven different combinations of start and goal configuration, and a range of moving obstacle speeds, $0.1\,\mathrm{m/s}$ to $1.0\,\mathrm{m/s}$. We recorded the resultant smoothness (GP) cost associated with each trajectory and the number of collisions that the robot made with the environment.

To formalise the task, we constructed a factor graph comprising of 31 time-indexed variable nodes, separated by $0.1\,\mathrm{s}$ intervals, to form a $3\,\mathrm{s}$ time horizon. Prior factors were added to specify the start and goal configurations, and Gaussian priors used to connect variable nodes, as in (Mukadam et al. 2018). We assigned an obstacle factor to each variable node and interpolated obstacle factors between pairs of variable nodes with a discretisation of $\tau = 5\mathrm{ms}$ ($n_{int} = 19$) to provide dense collision checking along the trajectory. Further parameters used were $\epsilon = 0.2\,\mathrm{m}$ to encourage safe clearance of obstacles, as well as $\sigma_{cost} = 0.2$, as a result of tuning, to weight the relative costs of trajectory smoothness and collision avoidance. The environment was discretised at $4\,\mathrm{cm}$ resolution in a $96 \times 96 \times 96$ voxel workspace.

We performed another similar experiment on the experimental setup shown in Figure 1, in which a Franka Emika Panda 7-DoF robotic arm was tasked with reaching across the gap between two tables to achieve a 'pick-up' goal configuration. During execution, a Toyota HSR traverses the walkway between the tables, acting as a moving obstacle. For this setup, we used 42 different combinations of start and goal configuration. The moving obstacle speed was chosen to be in the range of $1.0\,\mathrm{m/s}$ to $2.0\,\mathrm{m/s}$, most closely resembling walking speeds of a human. We chose $\tau = 0.05\mathrm{s}$ ($n_{int} = 1$) to save compute time and because the time interval between variable nodes was already small. Other parameters used were $\sigma_{cost} = 0.05$ and $\epsilon = 0.3\,\mathrm{m}$ based on cost tuning. As before, the environment was discretised at $4\,\mathrm{cm}$ resolution in a $96 \times 96 \times 96$ voxel workspace.

### 3.4 Closed-loop Implementation

We combined our predictive framework using composite signed-distance fields with our dynamic approach, applied within GPMP2, to solve planning problems in dynamic environments. The control loop is shown in Algorithm 1. In each iteration of the control loop, we observe the environment and update the workspace occupancy grid. We populate the occupancy grid by tracking the position of model states in the Gazebo simulation and inserting occupied regions into the occupancy grid at corresponding locations.

Using the methods discussed in Section 3.1, we segment and track obstacles online. A constant-velocity model is applied to each of the segmented moving objects, enabling us to predict the future locations of each object and generate composite SDFs for all obstacle factors with time-index equal to or greater than the current time. To our knowledge, this is the first work to implement and update different SDFs

for each time-step as in time-configuration space planning. A prior factor is also added for the current time-index to set the current configuration of the robot. The factor graph is then re-optimised and the new trajectory is merged with the current trajectory being executed on the robot.

We demonstrate our closed-loop implementation of the proposed framework on a Panda 7-DoF robotic arm in Gazebo, using the same task as described previously in which the Panda reaches across a gap to place the end-effector over a table, as in a pick-and-place task. We set the HSR base movement speed at $1.4\,\mathrm{m/s}$ to test against a speed comparable with the average walking pace (Hermodsson et al. 1994; Nolan et al. 2018; Wang et al. 2019).

### 3.5 Hardware Experiments

We validate our dynamic GPMP2 approach in the presence of uncertainty in the execution of the trajectory, as well as the pose of the dynamic obstacle, in hardware experiments. We use a Franka Emika Panda arm in conjunction with a Realsense D435i RGB-D camera for object tracking and detection. We used *apriltag_ros* (Malyuta et al. 2019) to identify the pose of an AprilTag (Wang and Olson 2016) fiducial marker affixed to a dynamic obstacle (box). For safety purposes, we used a longer time-horizon of $5\,\mathrm{s}$ and an epsilon value of $0.4\,\mathrm{m}$ to encourage both lower joint speeds and comfortable collision avoidance. The robot was positioned on a single table and tasked with a variety of motions, including the same motion as in the previous experiment, and avoiding the box as moved by the operator.

## 4 Evaluation

We conducted experiments using an 8-core Intel Core i7-9700 CPU @ $4.50\,\mathrm{GHz}$ and $2133\,\mathrm{MHz}$ DDR4 RAM. Signed-distance field computations were performed in series, as in the original GPMP2 implementation, using the `bwdist` function in MATLAB. GTSAM was configured to use multi-threading enabled for linearising the factor graph.

### 4.1 Composite SDF Prediction

Computation times for composite signed-distance fields and full computation from $96 \times 96 \times 96$ occupancy grids are shown in Table 1.

The composite initialisation time, highlighted in grey, comprises of the time to calculate the SDFs associated with the static workspace, $\mathcal{S}_{static}$, and each of the moving obstacles in the scene, $\boldsymbol{\mathcal{S}}$. This initial time penalty must be paid at the startup of the experiment or any subsequent time you wish to update the object and static environment SDFs. However, our results show that beyond this initialisation penalty, it is significantly faster to generate predicted composite SDFs than to perform full SDF calculations; we see a $7.9\,\mathrm{x}$ to $28.0\,\mathrm{x}$ speed-up for subsequent predictions depending on the discretisation of the workspace. This has a significant impact on enabling SDF predictions to be generated in real-time—conducting our experiment for a $96 \times 96 \times 96$ discretisation of the workspace, as in our closed-loop implementation, gives a mean SDF generation rate of $540\,\mathrm{Hz}$, compared with $36\,\mathrm{Hz}$ for the benchmark.

### 4.2 Dynamic GPMP2

The performance of the three scenarios described in Section 3.3, on the 'single moving pillar' dataset, are presented in Figure 6. The figure shows an absence of data for the obstacle speed of $0.5\,\mathrm{m/s}$; at $0.5\,\mathrm{m/s}$, the obstacle intersects the goal configurations at the end of the $3\,\mathrm{s}$ time horizon, resulting in inevitable collisions. Therefore, we exclude results for $0.5\,\mathrm{m/s}$ from our analysis. Of 69 planning results, we exclude three for which none of the planning scenarios found a collision-free trajectory.

Our results strongly support the incorporation of future obstacle motions into motion planning. Figure 6 shows that in most cases, including the obstacle trajectory motion into the planning significantly increases the likelihood of both finding a collision-free trajectory and reducing the jerk associated with the trajectory. Furthermore, the results motivate the use of our proposed framework in which we generate predicted ESDFs in real-time.

We note here that our implementation of a dynamic obstacle factor in GPMP2 incurs a cost of $\approx 2\,\mathrm{ms}$ to be replaced by an obstacle factor with a different SDF.

### 4.3 Closed-Loop Implementation

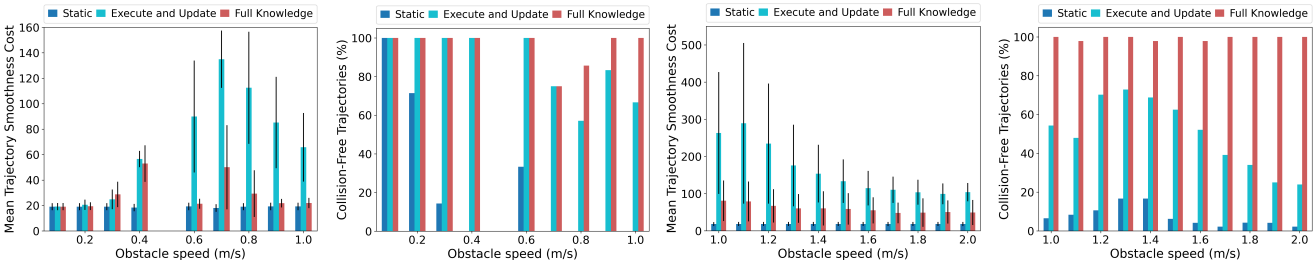
Our closed-loop implementation was successful in providing re-optimised trajectories fast enough for a Panda arm to avoid a moving HSR robot while reaching across a gap between two tables. Figure 1 illustrates the trajectory taken in each of the three cases. The 'static' case would result in a collision and the 'execute and update' would exhibit erratic movements. Figure 7 shows a comparison of the resultant joint position trajectories for the 'execute and update' against our composite prediction method. The results lend further support that by including *predicted SDFs* into motion planning, we can obtain smoother robot trajectories.

This experiment was conducted using a $96 \times 96 \times 96$ workspace with $4\,\mathrm{cm}$ resolution. The parameters for the motion planner were $n_{int} = 1$, $\sigma_{obs} = 0.05$, $\epsilon = 0.3\,\mathrm{m}$. A time horizon of $3\,\mathrm{s}$ was given, with variable factors time-indexed in steps of $\delta_t = 0.1\mathrm{s}$, resulting in 31 support states. The first iteration of the control loop results in the slowest update frequency because more future factors need to be updated and more predictions are generated. Averaged over 100 runs, our implementation runs at $3.4\,\mathrm{Hz}$ in the early iterations, with a standard deviation of $0.1\,\mathrm{Hz}$. In the later iterations, we achieve an update frequency of $\approx 10\,\mathrm{Hz}$. We demonstrate the resultant trajectories for each of the three cases in the accompanying video. [1]

### 4.4 Hardware Experiments

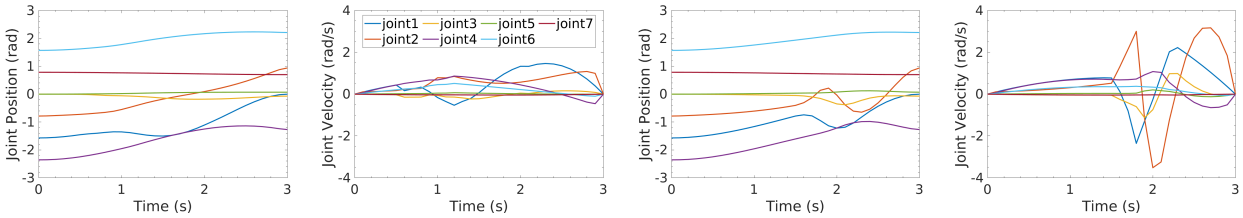
Our implementation performed well on the real robot arm, producing re-planned trajectories which successfully avoided the dynamic object in many cases. An example comparison against using composite SDFs but without prediction is shown in Figure 8. By accounting for the predicted motion of the obstacle, the manipulator is able to avoid the obstacle by following a much smoother trajectory.

[1] https://youtu.be/vJH8qBRMedw



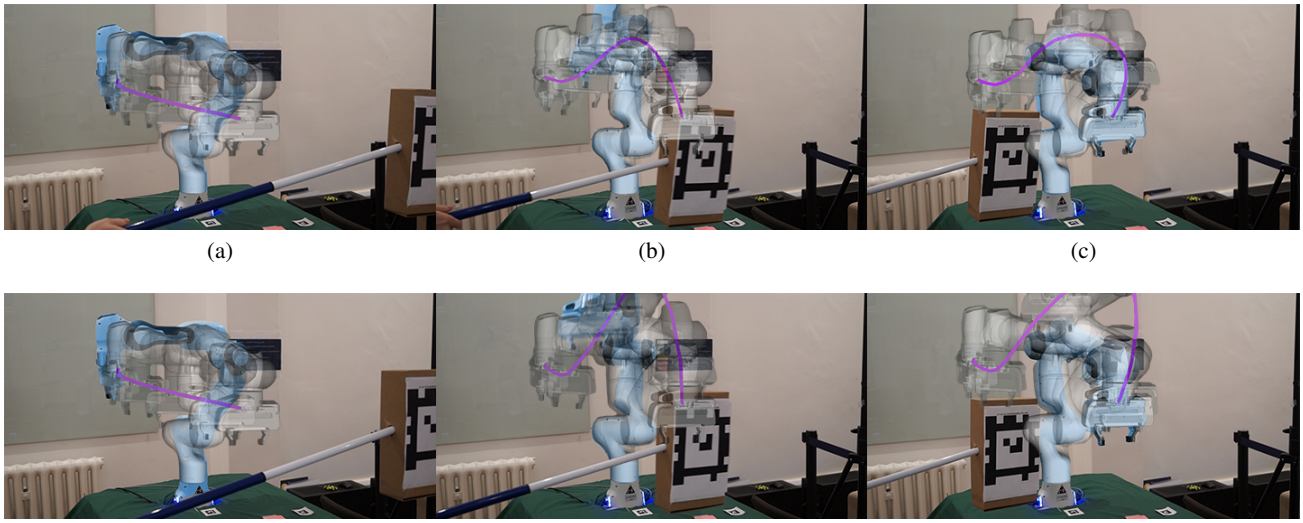

(a) (b) (c) (d)

Figure 6: Results for experiments conducted on the WAM arm in the 'single moving pillar' setup are shown in Figures 6a and 6b. Results for the Panda arm avoiding a moving HSR are shown in 6c and 6d. We see that in all cases, using full knowledge of obstacle trajectories provides at least an equal chance of finding a collision-free trajectory using GPMP2; for most obstacle speeds, the improvement is significant.




(a) Our method - including prediction (b) Execute and update

Figure 7: Example trajectories for joint position and velocity. Our results highlight that the inclusion of environment prediction in motion planning can lead to smoother trajectories which exhibit less erratic movements.



(a) (b) (c)

(d) (e) (f)

Figure 8: *Top row* - online motion planning using our approach of composite signed distance fields and prediction. *Bottom row* - using composite signed distance fields but without prediction. From left to right, the images show the planned trajectory for times $t = 0\,\mathrm{s}$, $t = 2.5\,\mathrm{s}$ and $t = 5.0\,\mathrm{s}$ (resultant trajectory). The blue highlight indicates the configuration of the robot at the current time-step while the purple line show the planned end-effector trajectory. The predictive component of our framework enables a smoother resultant trajectory, as shown in Figure 8c.

| | Workspace Size (Voxels Per Side) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 64 | 96 | 128 | 160 | 192 | 224 | 256 | 288 | 320 |
| Composite Init. (ms) | $13 \pm 1$ | $31 \pm 4$ | $65 \pm 3$ | $126 \pm 6$ | $211 \pm 3$ | $329 \pm 6$ | $475 \pm 7$ | $659 \pm 5$ | $936 \pm 23$ |
| Full Computation (ms) | $9.3 \pm 1.0$ | $28 \pm 2$ | $62 \pm 3$ | $121 \pm 5$ | $217 \pm 11$ | $338 \pm 9$ | $493 \pm 13$ | $687 \pm 18$ | $960 \pm 28$ |
| Composite Prediction (ms) | $0.6 \pm 0.6$ | $1.9 \pm 0.2$ | $4.0 \pm 0.9$ | $7.4 \pm 0.5$ | $24 \pm 6$ | $42 \pm 2$ | $62 \pm 3$ | $86 \pm 3$ | $121 \pm 3$ |
| Repeat Prediction Speed-Up | **15.5x** | **28.0x** | **15.5x** | **16.4x** | **9.0x** | **8.0x** | **8.0x** | **8.0x** | **7.9x** |

Table 1: Signed-Distance Field Timing Analysis

We found that while neglecting to predict the obstacle motion was also effective at avoiding obstacles due to the high re-plan frequency, the robot would exhibit erratic behaviours, in particular when the obstacle position coincided with the desired arm goal state.

## 5 Discussion

Our proposed framework gives compelling results in support of incorporating predicted obstacle trajectories into motion planning in time-configuration space. While we implemented a constant-velocity model to demonstrate our framework's novel ability to include obstacle trajectory predictions directly into SDF-based based motion planning, one can replace this with a more complex model that takes into account any prior information we have on the obstacle's trajectory, including its history. Exploring different prediction models for obstacle motion was beyond the scope of this work and as such we did not explore the avoidance of obstacles that follow complex motions, such as curved trajectories; we leave this for future work. Without using a more complex model, our current implementation is likely to fail in circumstances where the obstacle suddenly changes direction or stops.

A limitation of the proposed method is the potentially large memory requirements associated with assigning and storing different signed-distance fields for different time-indexed obstacle factors. For example, if a $300 \times 300 \times 300$ discretisation of the workspace is required for an application, our previous problem of a $3\,\mathrm{s}$ time horizon, discretised into 31 time-steps, will require $\approx 6.24\,\mathrm{GB}$ RAM.[2] We thus encounter a trade-off; minimising the workspace size results in quicker computation and lower memory requirements, however, to monitor a larger workspace requires the resolution to be decreased. We believe that this method could be adapted in future work to perform the superposition on a query basis, rather than computing the composite signed-distance field for the entire workspace.

Generating composite predictions for each time-step, as well as the updating of the obstacle factors, are operations which seem suitable for parallelisation. By updating factors in parallel, we believe that significantly higher update rates could be achieved allowing for more interactive and responsive behaviour.

We have demonstrated benefits of predicted composite SDFs that are independent of state-of-the-art SDF computing techniques. In our future work, we will evaluate different methods that can leverage our technique and framework, particularly parallel implementations. We also note that while pre-computing a static scene is acceptable for a fixed-base robot arm, it is less useful in the context of a mobile robot performing tasks in an unmapped region of space. To address this, we are working on integrating our approach with ESDFs that are updated from live point-cloud data.

[2] A `double` is stored using $8\,\mathrm{bytes}$. $300^3 \times 8 = 2.16 \times 10^8\ \mathrm{bytes} = 206\,\mathrm{MB}$. $31 \times 206 \approx 6.24\,\mathrm{GB}$

## 6 Conclusion

This paper explored the application of composite signed-distance fields to motion planning in dynamic environments. We first exploited the speed of the `min` operation for composition of SDFs, and secondly, that signed-distance field representations must only be accurate up to a specified $\epsilon$ for motion planning problems. We show that composite SDFs can be used to provide significant speed-up for generating SDFs when accounting for moving obstacles. We investigated motion planning with a GPMP2 implementation which uses dynamic obstacles factors, enabling the planner to account for moving obstacles. Over a range of tasks and two different robot platforms, our results show that by incorporating predicted obstacle trajectories, we can significantly reduce the smoothness cost of trajectories and the rate of collisions. We leveraged composite SDFs and a dynamic GPMP2 implementation to present a novel framework which exploits the sparsity of the workspace and the compositional nature of signed-distance fields to generate real-time predictions of the workspace SDF, *predicted signed-distance fields*. We verified our approach on a real 7-DoF Panda arm with obstacle tracking performed using fiducial sensing, demonstrating that it can plan trajectories in dynamic environments and successfully avoid moving obstacles.

## Acknowledgments

This research was supported by (1) the UK Engineering and Physical Sciences Research Council (EPSRC) through the University of Oxford Centre for Doctoral Training, Autonomous Intelligent Machines and Systems (AIMS, grant references EP/L015897/1 and EP/R512333/1), the UK RAI Hub for Offshore Robotics for Certification of Assets (ORCA, grant reference EP/R026173/1), and the EPSRC grant Robust Legged Locomotion (EP/S002383/1), as well as (2) the European Commission under the Horizon 2020 project Memory of Motion (MEMMO, project ID: 780684).

## References

Dellaert, F. 2012. Factor Graphs and GTSAM: A Hands-on Introduction. Technical report, Georgia Tech.

Hart, P.; Nilsson, N.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. on Systems Science and Cybernetics* 4(2): 100–107.

Hermodsson, Y.; Ekdahl, C.; Persson, B. M.; and Roxendal, G. 1994. Gait in male trans-tibial amputees: A comparative study with healthy subjects in relation to walking speed. *Prosthetics and Orthotics International* 18(2): 68–77.

Hornung, A.; Wurm, K. M.; Bennewitz, M.; Stachniss, C.; and Burgard, W. 2013. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Autonomous Robots* 34: 189–206.

Hsu, D.; Kindel, R.; Latombe, J. C.; and Rock, S. 2002. Randomized kinodynamic motion planning with moving obstacles. *The Int. J. of Rob. Research* 21(3): 233–255.

Jianbo Shi; and Tomasi. 1994. Good features to track. In *Proc. IEEE CVPR*, 593–600. IEEE Comput. Soc. Press.

Jones, M.; Baerentzen, J.; and Sramek, M. 2006. 3D distance fields: a survey of techniques and applications. *IEEE Trans. on Visualization and Computer Graphics* 12(4): 581–599.

Kavraki, L.; Kolountzakis, M.; and Latombe, J.-C. 1998. Analysis of probabilistic roadmaps for path planning. *IEEE Trans. on Rob. and Autom.* 14(1): 166–171.

Kavraki, L.; Svestka, P.; Latombe, J.-C.; and Overmars, M. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. on Rob. and Autom.* 12(4): 566–580.

Koenig, S.; and Likhachev, M. 2002. Improved fast replanning for robot navigation in unknown terrain. In *Proc. IEEE Int. Conf. on Robot. and Automat. (ICRA)*, 968–975.

Kolur, K.; Chintalapudi, S.; Boots, B.; and Mukadam, M. 2019. Online Motion Planning Over Multiple Homotopy Classes with Gaussian Process Inference. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2358–2364.

Kuffner, J.; and LaValle, S. 2000. RRT-connect: An efficient approach to single-query path planning. In *Proc. IEEE Int. Conf. on Robot. and Automat. (ICRA)*, 995–1001.

LaValle, S. M. 1998. Rapidly-exploring random trees : a new tool for path planning. Technical report, Iowa State University.

Li, S.; and Shah, J. A. 2019. Safe and efficient high dimensional motion planning in space-time with time parameterized prediction. In *Proc. IEEE Int. Conf. on Robot. and Automat. (ICRA)*, 5012–5018.

Malyuta, D.; Brommer, C.; Hentzen, D.; Stastny, T.; Siegwart, R.; and Brockers, R. 2019. Long-duration fully autonomous operation of rotorcraft unmanned aerial systems for remote-sensing data acquisition. *Journal of Field Robotics* .

Maurer, C. R.; Qi, R.; and Raghavan, V. 2003. A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 25(2): 265–270.

Menon, A.; Cohen, B.; and Likhachev, M. 2014. Motion planning for smooth pickup of moving objects. In *Proc. IEEE Int. Conf. on Robot. and Automat. (ICRA)*, 453–460.

Merkt, W.; Ivan, V.; and Vijayakumar, S. 2019. Continuous-Time Collision Avoidance for Trajectory Optimization in Dynamic Environments. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 7248–7255.

Mukadam, M.; Dong, J.; Yan, X.; Dellaert, F.; and Boots, B. 2018. Continuous-time Gaussian process motion planning via probabilistic inference. *The Int. J. of Rob. Research* 37(11): 1319–1340.

Mukadam, M.; Yan, X.; and Boots, B. 2016. Gaussian Process Motion planning. In *Proc. IEEE Int. Conf. on Robot. and Automat. (ICRA)*, 9–15. IEEE.

Newcombe, R. A.; Fitzgibbon, A.; Izadi, S.; Hilliges, O.; Molyneaux, D.; Kim, D.; Davison, A. J.; Kohi, P.; Shotton, J.; and Hodges, S. 2011. KinectFusion: Real-time dense surface mapping and tracking. In *Proc. IEEE ACM Int. Symp. Mixed Augment. Reality (ISMAR)*, 127–136.

Nolan, C. M.; Kon, S. S. C.; Patel, S.; Jones, S. E.; Barker, R. E.; Polkey, M. I.; Maddocks, M.; and Man, W. D.-C. 2018. Gait speed and pedestrian crossings in COPD. *Thorax* 73(2): 191–192.

Oleynikova, H.; Taylor, Z.; Fehr, M.; Siegwart, R.; and Nieto, J. 2017. Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 1366–1373.

Park, C.; Pan, J.; and Manocha, D. 2012. ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments. In *Proc. Int. Conf. Automat. Planning and Scheduling (ICAPS)*, 207–215.

Park, J. S.; Park, C.; and Manocha, D. 2019. I-Planner: Intention-aware motion planning using learning-based human motion prediction. *The Int. J. of Rob. Research* 38(1): 23–39.

Raj, A.; Sivaraman, A.; Bhowmick, C.; and Verma, N. K. 2016. Object tracking with movement prediction algorithms. In *Proc. IEEE Int. Conf. Ind. Inf. Syst. (ICIIS)*, 285–290.

Ratliff, N.; Zucker, M.; Bagnell, J. A.; and Srinivasa, S. 2009. CHOMP: Gradient optimization techniques for efficient motion planning. In *Proc. IEEE Int. Conf. on Robot. and Automat. (ICRA)*, 489–494.

Reiner, T.; Mückl, G.; and Dachsbacher, C. 2011. Interactive modeling of implicit surfaces using a direct visualization approach with signed distance functions. Technical Report 3, Karlsruhe Inst. of Tech.

Schulman, J.; Duan, Y.; Ho, J.; Lee, A.; Awwal, I.; Bradlow, H.; Pan, J.; Patil, S.; Goldberg, K.; and Abbeel, P. 2014. Motion planning with sequential convex optimization and convex collision checking. *The Int. J. of Rob. Research* 33(9): 1251–1270.

Wang, J.; and Olson, E. 2016. AprilTag 2: Efficient and robust fiducial detection. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 4193–4198.

Wang, Z.; Liu, R.; Wu, X.; and Liu, Z. 2019. *Estimation Models for the Safety Level of Indoor Space Pedestrian Flows*, 9–18. Springer.

Whelan, T.; Kaess, M.; and Fallon, M. 2012. Kintinuous: Spatially extended kinectfusion. *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras* 7.

Yun Seok Nam; Bum Hee Lee; and Moon Sang Kim. 1996. View-time based moving obstacle avoidance using stochastic prediction of obstacle motion. In *Proc. IEEE Int. Conf. on Robot. and Automat. (ICRA)*, volume 2, 1081–1086.

Zucker, M.; Ratliff, N.; Dragan, A. D.; Pivtoraiko, M.; Klingensmith, M.; Dellin, C. M.; Bagnell, J. A.; and Srinivasa, S. S. 2013. CHOMP: Covariant Hamiltonian optimization for motion planning. *The Int. J. of Rob. Research* 32(9-10): 1164–1193.

## Further Remarks

This work proposed the use of composite distance fields as a fast method of incorporating the predicted motions of obstacles directly into a distance field representation. While we successfully demonstrated its effectiveness in simulation and hardware experiments, there are many aspects of this technique to further explore in future work, some of which are discussed in the paper. One such aspect that we wish to discuss here further is the choice of trajectory prediction methodology.

In this work, we implemented a simple constant-velocity model for demonstration of the technique while emphasising that it can be substituted with more sophisticated prediction methods. Alternative prediction methods may offer the benefit of greater predictive accuracy and the ability to capture a greater complexity of motion. Humans in particular are often motivated by a goal that they intend to reach and are influenced by social factors; as a result, they tend to follow more complex trajectories than inanimate objects. We consider this in Chapter 6 and demonstrate the integration of composite distance fields with a complex human trajectory prediction method.

Another topic for further work is how to account for uncertainty in an obstacle's predicted trajectory. For a single predicted trajectory, as in the presented paper, one can attempt to address uncertainty in predicted obstacle positions by choosing a sufficiently large safety margin, $\epsilon$, for obstacle avoidance. By using a large safety margin, optimised trajectories will tend to maintain a sufficient distance from the predicted positions of obstacles; however, this may also lead to jerkier trajectories around obstacles due to increase radius of effect that obstacle avoidance terms will have relative to smoothness terms in the objective function. Single predicted trajectories are appropriate when considering the motions of inanimate dynamic objects such as a ball rolling across the floor. However, for animate objects such as robots and humans, it is inherently difficult to make long-term trajectory predictions because many different trajectories are often possible. As shown in work by Kolur et al. [49], agents may simultaneously consider multiple trajectories and switch between them throughout the evolution of a task as new observations are made. There are numerous ways in which we believe that multiple trajectories for

dynamic obstacles can be considered while using our method of composite distance fields. The most straightforward approach assumes that obstacles will follow the most likely trajectory. Under this assumption, we revert to the scenario in the presentation paper and compute distance fields for a single trajectory. A limitation of this method is that if the most likely trajectory assumption is incorrect at a later timestep, an alternate possibility will not have been considered in advance. The robot is thus required to compute a new sequence of composite distance fields and perform motion planning to react to the obstacle trajectory. An alternate approach would be to perform motion planning for multiple obstacle trajectory scenarios simultaneously. We can select and transition between optimised trajectories as we refine our belief about which trajectory a dynamic obstacle will follow. While the forward planning of this method is appealing, it is computationally expensive. In theory, distance field calculations and optimisation for multiple scenarios can be performed in parallel; however, in practice, this will result in very high memory usage due to the memory limitations discussed in the paper.

## Statement of Authorship for joint/multi-authored papers for PGR thesis

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

| | |
|---|---|
| Title of Paper | Predicted Composite Signed-Distance Fields for Real-Time Motion Planning in Dynamic Environments |
| Publication Status | Published |
| Publication Details | M. N. Finean, W. Merkt, and I. Havoutis. "Predicted Composite Signed-Distance Fields for Real-Time Motion Planning in Dynamic Environments." In *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 31, pp. 616-624. 2021. |

### Student Confirmation

| | | | |
|---|---|---|---|
| Student Name: | Mark Nicholas Finean | | |
| Contribution to the Paper | Developed the concepts and ideas<br>Implementation of the concepts (code)<br>Devised and conducted simulation and hardware experiments<br>Data collection<br>Produced and edited figures and the associated presentation video<br>Wrote the paper with co-authors | | |
| Signature |  | Date | 10/12/2021 |

### Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

| | | |
|---|---|---|
| Supervisor name and title: Dr. Ioannis Havoutis | | |
| Supervisor comments | | |
| Signature  | Date | 14/01/2022 |

# 4

# Real-time Mapping and Motion Planning

The previous chapter introduced a new framework for motion planning in dynamic environments. Through simulation and hardware experiments, we showed that by accounting for the predicted trajectories of moving objects within the robot's workspace, we could obtain better, safer trajectories. However, key limitations of this work were firstly the lack of live sensing and secondly, the need to pre-compute a distance field of the static scene. While this was still a practical approach for a fixed robot arm, this is not sufficient for a mobile manipulator exploring uncertain environments. These limitations were the primary motivations for the work presented in this chapter.

This chapter develops the fundamental framework that underlies the overarching framework presented in 6. As described in previous chapters, optimisation-based motion planners often pre-compute signed-distance fields and operate in static environments due to the prohibitively expensive computation times associated with distance fields. This chapter aims to remove this computational barrier and enables a state-of-the-art optimisation-based motion planner to perform in dynamic environments by integrating a fast perception pipeline that repeatedly calculates distance fields online. We ask the practical question of whether the 'signing' of distance fields is necessary or valuable in motion planning; to our knowledge, this is an important implementation detail that has not previously been addressed in the literature. In our endeavour to generate and update distance fields online,

we investigate and benchmark three state-of-the-art approaches for generating distance fields in environment reconstruction. We leverage our findings to develop a hybrid system that interlinks a real-time mapping module with the fast whole-body motion re-planning provided by our receding-horizon implementation of GPMP2. Our integrated system constrats with previous works and enables optimisation-based whole-body motion planning in dynamic environments. To highlight this contribution and the limitations from our previous work that we address, we validate our framework on a real-world Toyota Human Support Robot and accomplish a number of whole-body motion planning tasks in dynamic environments using live sensor data.

The following manuscript [4] (DOI:10.1109/IROS51168.2021.9636860) was presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021, along with a supplementary video[1].

[1] `https://ieeexplore.ieee.org/document/9636860/media`

# Simultaneous Scene Reconstruction and Whole-Body Motion Planning for Safe Operation in Dynamic Environments

Mark Nicholas Finean[1], Wolfgang Merkt[1], and Ioannis Havoutis[1]

***Abstract*— Recent work has demonstrated real-time mapping and reconstruction from dense perception, while motion planning based on distance fields has been shown to achieve fast, collision-free motion synthesis with good convergence properties. However, demonstration of a fully integrated system that can safely re-plan in unknown environments, in the presence of static and dynamic obstacles, has remained an open challenge. In this work, we first study the impact that signed and unsigned distance fields have on optimisation convergence, and the resultant error cost in trajectory optimisation problems in 2D path planning, arm manipulator motion planning, and whole-body loco-manipulation planning. We further analyse the performance of three state-of-the-art approaches to generating distance fields (Voxblox, Fiesta, and GPU-Voxels) for use in real-time environment reconstruction. Finally, we use our findings to construct a practical hybrid mapping and motion planning system which uses GPU-Voxels and GPMP2 to perform receding-horizon whole-body motion planning that can smoothly avoid moving obstacles in 3D space using live sensor data. Our results are validated in simulation and on a real-world Toyota Human Support Robot (HSR).**

## I. Introduction

In recent years, we have seen tremendous advances across many fields of robotics, from hardware to vision and planning. As the capabilities of robots has increased, the question is now "when will we see large scale integration into our daily lives?". A key concern that needs to be solved before robots become commonplace is that of safety; we require robots to be reliable and interact safely with their surroundings. Key hereto is the ability to recognise and reason about static and dynamic obstacles in real-time to prevent collision and injury to people, the environment, and the robots themselves.

There is significant research in motion planning that focuses on or assumes a static environment, however, this assumption breaks down in the real-world where our surroundings are often dynamic. For robots to become more widely used, such as in household environments, they must be able to perform motion planning and collision avoidance in the presence of moving obstacles. Considerable research in the mapping and scene reconstruction communities has achieved the ability to reconstruct environments in great detail in real-time based on voxel grids [1], [2], Truncated Signed Distance Fields (TSDFs) [3], [4], surfels [5], [6], and octrees representations [7] using CPU or GPU computation. For instance, this capability has been used in path planning for Micro Aerial Vehicles (MAVs) [8]–[11]. At the same time, work in the motion planning community has proposed solutions for fast planning with real-world sensed data, enabling the generation of trajectories that avoid collisions with static and dynamic obstacles in discrete-time [12]–[15] and continuous-time [16], [17]. However, there has been little work in combining the two to provide a usable integrated framework for real-time re-planning in dynamic environments, providing motivation for the research presented here.

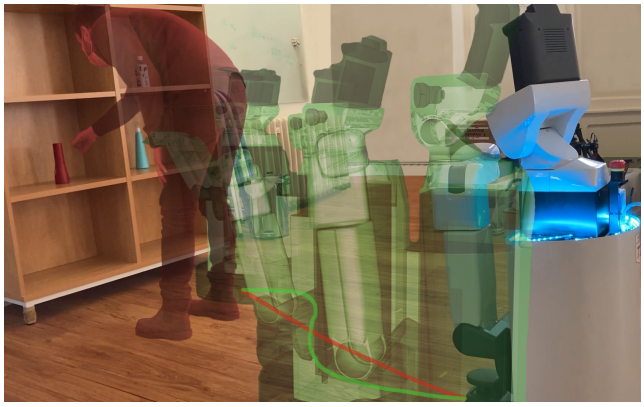


Fig. 1. Real-time re-planning and avoidance of moving obstacles using the HSR. The robot was tasked with picking up a bottle from a shelf. During execution, a human demonstrator walked into the planned trajectory of the robot. Using our integrated motion planning and real-time mapping system, the robot successfully integrated the dynamic obstacle without any prior in real-time and re-planned around the newly observed obstacle in a receding-horizon fashion.

In this work, we provide the first comparison of the impact that Signed Distance Fields (SDFs) and Unsigned Distance Fields (USDFs) have on trajectory optimisation convergence and the error cost for navigation planning, motion planning, and whole-body loco-manipulation. We explore a selection of state-of-the-art mapping and reconstruction packages to determine the best performing method for integration with a whole-body motion planner. After discussing our motivation for motion planner selection, we describe its integration with the reconstruction pipeline.

Overall, we present a full framework to concurrently map the environment in 3D and perform fast re-planning online in a dynamic environment with a whole-body mobile manipulator (HSR). The key contributions of this paper are:

1) Integration of trajectory optimisation-based whole-body

[1] Oxford Robotics Institute, University of Oxford

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC), award reference 1904254, for the University of Oxford Centre for Doctoral Training, Autonomous Intelligent Machines and Systems (AIMS). This work was partly supported by the UKRI/EPSRC grants [EP/S002383/1], [EP/R026084/1] and [EP/R026173/1] and the EU H2020 Project MEMMO(780684). This work was part of the Human-Machine Collaboration Programme, supported by a gift from Amazon Web Services. Email: mfinean@robots.ox.ac.uk

motion planning with fast GPU-based distance field reconstruction.
2) The first demonstration of real-time reactive collision avoidance using whole-body trajectory optimisation and live sensor data in unknown 3D environments for loco-manipulation on articulated systems (8 DoF).
3) Exploration of incrementally built Euclidean Distance Transforms (EDTs) for the motion planning of whole-body and articulated systems.
4) A comparison of the impact that USDFs and SDFs have on trajectory optimisation convergence and optimality.

## II. Related Work

We separate the relevant literature into three categories; we first explore the state-of-the-art in motion planning in dynamic 3D environments and make the case for using a trajectory optimisation approach (GPMP2). We then explore the options for online mapping and reconstruction of environment distance fields. Lastly, we discuss the prior work in exploring integrated systems to provide collision avoidance in real-world environments with moving obstacles.

### A. Motion Planning

Motion planning is a well-studied problem with many possible approaches. Sampling-based algorithms and graph search methods [18]–[20] are among the most common techniques to apply. They are mathematically appealing because we can select for desirable characteristics such as completeness, however they often require post-processing to address smoothness of the motions. While there are optimal planners which aim to address this, they suffer from the curse of dimensionality for robots with many degrees of freedom [21]. A key consideration in this work was a motion planner's ability to plan in dynamic environments, placing emphasis on fast planning times and ideally the ability to re-plan smoothly in response to changes in the environment. Trajectory optimisation-based methods are well suited to these criteria.

Trajectory optimisation algorithms, such as CHOMP, STOMP and TrajOpt, operate by minimising an objective function to solve for a feasible and optimal trajectory [12], [16], [22]. In the cases of CHOMP and STOMP, the trajectory is finely discretised to generate trajectories that avoid obstacles and maintain smoothness; however, fine discretisation is computationally expensive. TrajOpt and our prior work [17] represent the trajectory using fewer states by introducing continuous-time collision checking. Yet, to achieve smoothness, a fine discretisation may still be required in practice.

Another approach to reduce computational requirements is to use kernel embeddings to represent the trajectory. GPMP2 [13], for instance, leverages Gaussian Process Interpolation to achieve a continuous trajectory representation using a small number of discrete points. Similar to AICO [23], GPMP2 treats the motion planning problem as probabilistic inference on a factor graph. GPMP2 is built upon the GTSAM framework [24] which uses factor graphs and Bayes networks as the underlying objects to frame an optimisation problem for the most probable configuration or plan. Factor graphs are popular in the Simultaneous Localisation and Mapping (SLAM) community and working in this paradigm offers the flexibility to use efficient tools, such as incremental Smoothing and Mapping (iSAM) to perform fast incremental inference for re-planning [25], [26]. Mukadam et. al. show iSAM-based re-planning (iGPMP2) to be an order of magnitude faster than planning from scratch [13]. The superior planning speed and ability to quickly re-plan makes the GPMP2 factor graph formulation an appealing candidate for use in dynamic environments.

### B. Collision Avoidance

Some approaches in the literature view obstacles from a more theoretical viewpoint and assume prior knowledge of their shape, size and position. Ratliff et. al. model obstacle constraints as analytical inequality constraints with a margin in RieMo [15]. Merkt et. al. [27] use primitive shapes and smooth hinge losses to penalise collisions, whereas TrajOpt [16] applies approximate convex decomposition to leverage efficient convex shape distance computations. These methods are not suitable for real-world planning in unknown environments as prior knowledge about objects is not always available.

Sampling-based algorithms such as RRTs and PRMs require only binary occupancy information to represent the environment. This information can be stored in a simple 3D voxel grid whereby the environment is discretised into a regular grid, with each voxel storing the binary occupancy of that position in space. While this method provides fast memory access, it can require a large amount of memory to represent an environment. Octree methods provide a more memory-efficient representation with Octomap being a commonly used framework [28].

In contrast, trajectory optimisation-based motion planners require gradients. While stochastic algorithms, such as STOMP, obtain gradients using binary occupancy information, non-stochastic approaches require a continuously varying environment representation from which gradients can be obtained. CHOMP, TrajOpt, and GPMP2 use Euclidean Signed Distance Fields (ESDFs) to represent the environment [12], [13], [16]. ESDFs have been shown as an effective method for use in static environments however their CPU compute times generally prohibit real-time performance in a dynamic environment and are thus typically pre-computed and assumed to be static [14]. Further, they commonly require the integration of an occupancy grid prior to computing an ESDF. GPMP2 [13] assumes this to be given, while other approaches can compute an ESDF from alternative representations such as an Octomap.

To improve the computation time for ESDF updates, Lau et al. presented an efficient method for incrementally updating an ESDF from occupancy maps [29]. Oleynikova et al. extended this approach with Voxblox to build and update ESDFs incrementally out of TSDFs [9]. Usenko et. al. introduced 'ewok' which uses a fixed-size sliding window around the position of a MAV to incrementally build ESDFs

from occupancy [11]. Han et. al. use doubly-linked lists to present a time-efficient method of incrementally building USDFs [8]. The aforementioned incremental methods have been demonstrated in 3D path planning environments for MAVs, however to our knowledge, they have not been demonstrated for use in whole-body motion planning for articulated systems or mobile manipulation platforms.

As the computations required for computing occupancy information and ESDF are inherently SIMD-parallelisable, significant research has also been conducted into optimising signed distance field calculations on GPUs. Jülg et. al. present a comprehensive comparison of fast exact 3D EDT implementations [10]. In particular, they show the Parallel Banding Algorithm (PBA), developed by Cao et. al. [30], to be "well suited for fast online GPU-based distance field computation".

### C. Real-World Systems for Dynamic Obstacle Avoidance

The vast majority of fully integrated systems have focused on mobile aerial robots. Voxblox integrated a local trajectory optimisation planner in static environments, while FIESTA uses kinodynamic path searching.

Alwala and Mukadam built upon the GPMP2 framework to present Joint Sampling and Trajectory Optimisation (JIST) and demonstrate effective collision avoidance in 2D navigation tasks as well as on a 7-DoF Sawyer robot arm [31] in simulation. However, JIST still uses pre-computed signed distance field calculations and was not verified on a real-world system with live sensor data.

Kaldestad et. al. [32] provide the first demonstration of collision avoidance in real-time on a real robot using parallel GPU processing. They calculate virtual forces to send to the robot impedance controller on a 7-DoF KUKA Arm. However, they use a restricted 2.5D environment model that is reset every time new depth sensor data is processed.

Hermann et. al. demonstrate mobile manipulation planning and re-planning on a GPU to operate in unknown environments by using grid-based planning techniques [2], [33]. The planning times reported are an order of magnitude greater than those achieved in similar tasks using optimisation-based methods [13]. The authors also observed that their method resulted in re-planning times that vary depending on how far along the current trajectory a new obstacle is observed. Jülg et al. [10] built upon the GPU-Voxels framework [2] to introduce highly optimised GPU calculations for EDTs. Due to the high speed of performing EDTs, they demonstrate real-time potential-field based motion planning of mobile aerial robot platforms in a fully 3D environment. Exploring the possibility of planning manipulator motions was beyond the scope of their work, motivating the work presented here.

To our knowledge, real-time receding-horizon trajectory optimisation has not been performed online in an unknown environment for a mobile manipulator.

## III. Signed vs Unsigned Distance Fields

Despite distance transforms being prominent in the literature for motion planning and MAV path planning, we could not find any justification for whether distance fields should be signed or unsigned in motion planning. For implementations such as wavefront planners, one would expect a signed distance field to have no impact in a static environment as long as the initial starting state is not in collision. In contrast, initial trajectories for a trajectory-optimisation based motion planner may start in collision (particularly when using the common "straight-line" initialisation); these approaches require gradients to perform updates and it would seem intuitive to require a signed distance field and continuous gradients throughout an obstacle to provide gradients that 'push' the trajectory out of collision. To verify this, we performed a series of experiments to analyse and compare trajectories generated using signed and unsigned distance fields.

### A. Methods

Experiments were carried out in simulation for 2D path planning, 7-DoF arm manipulation (Franka Panda), and whole-body motion planning with a 5-DoF manipulator on a holonomic base (HSR, 8-DoF). For each of the three cases, we generated 18 different robot states to produce a set of 153 pairs of start and goal configurations. We generated an obstacle set comprising of 100 cuboids of randomly generated size in the range of $0.0\,\mathrm{m}$ to $1.0\,\mathrm{m}$ for each dimension; each obstacle was associated with a randomly generated position in the workspace. With both USDFs and SDFs, we used GPMP2 with consistent parameter settings as a motion planner for each of the $15\,300$ motion planning problems. We used the Levenberg-Marquardt method of optimisation with the initial damping parameter set as $0.01$. The optimisation stopped if there was a relative decrease in error smaller than $10^{-5}$ or if it reached $100$ iterations.

### B. Results

To calculate the 'failure rate', we exclude cases in which the planner failed to find a collision-free trajectory using either the SDF or USDF. Similarly, we only compare trajectory costs across collision-free trajectories since trajectories that result in a collision would not be executed.

A summary of our findings is presented in Table I. In the *2D Navigation* experiments, we find that our intuition of SDF gradients 'pushing' the trajectory out of collision is confirmed, with plans that use USDFs having a failure rate that is 18.3 times greater. For a 2D planning example, this can be explained by entire query states being inside obstacles with no distance or distance-gradient information and the only gradients to 'pull' out obstacles come from the Gaussian Process smoothness factors. However, in higher dimensions we found that SDFs provide no practical advantage over USDFs for motion planning tasks in absolute terms, with almost all planning problems being solved successfully in both cases. We further validated this finding in complex narrow passage examples on a 7-DoF Panda arm manipulator where SDFs and USDFs performed equally well from infeasible straight line initialisations through the obstacle. This can be explained in articulated systems, such as in the *Arm* and *Whole-Body* tasks, by gradients being available at multiple

other locations that are not in collision; these gradients will assist in 'pulling' states out of collision.

Considering that a SDF calculation typically takes around twice as long to compute as a USDF, we conclude that USDFs are preferable to use on articulated and whole-body systems, particularly when operating in dynamic environments since the faster re-planning speed enables better adaption to changing environments.

## IV. Real-time Scene Reconstruction

We used the state-of-the-art mapping packages Voxblox [9] (CPU), FIESTA [8] (CPU), and GPU-Voxels [2] (GPU) to generate distance fields from live sensor data and provide direct query access to the obstacle factors used in GPMP2. Voxblox generates SDFs whereas GPU-Voxels and FIESTA produce USDFs. To provide a more thorough comparison, we adapted GPU-Voxels and FIESTA to optionally produce SDFs using their native methodology. We achieve this by calculating distance fields for both the occupancy map and the inverse occupancy map; the values are then subtracted to produce a SDF. In the case of GPU-Voxels, this was done such that the inverse distance field could be calculated in parallel. In practice, however, GPUs are still limited by the number of threads available and so for large environments, the two distance transforms will still be performed sequentially. For FIESTA, we implemented the inverse distance transform in a similar manner to how the original distance transform is calculated but reversed the roles of unoccupied and occupied cells found in ray-casting.

To compare the performance of the mapping frameworks, we used the Cow and Lady dataset, as first presented by Oleynikova et. al. [9]. We chose this dataset because it uses a small, indoor scene with multiple objects and accessories in the room; similar to a typical environment in which a mobile service robot might be deployed. The dataset features a rosbag file in which real RGB pointcloud data was collected using a Kinect v1 depth camera, along with published ground-truth pose transforms of the camera frame using a vicon sensor.

We evaluated each package across a range of resolutions, while retaining full spatial coverage, on the $142\,\mathrm{s}$ dataset. For each evaluation, the log file was played in real-time to simulate live operation and the update rate was recorded using an 8-core Intel Core i7-9700 CPU @ $4.50\,\mathrm{GHz}$ and $2133\,\mathrm{MHz}$ DDR4 RAM. We found that FIESTA did not operate successfully using multiple threads and so ran this package with a single thread; Voxblox was run in multi-threaded mode with 8 threads. GPU-Voxels was run on a Nvidia RTX 2060 GPU (1920 CUDA cores).

### A. Results

Results of our mapping framework comparison are shown in Table II. For Voxblox, we present the results for a propagation distance of $0.8\,\mathrm{m}$; we require a propagation distance at least as large as the sum of the maximum distance penalised in trajectory optimisation and the maximum radius of the spheres used to represent a robot's collision model. The maximum sphere radius used in our collision model of the HSR is $0.3\,\mathrm{m}$ and the maximum penalty distance used in our later experiments is $0.5\,\mathrm{m}$. We note that at the smallest propagation distance for which we measured performance, $0.1\,\mathrm{m}$, FIESTA and GPU-Voxels still outperformed Voxblox.

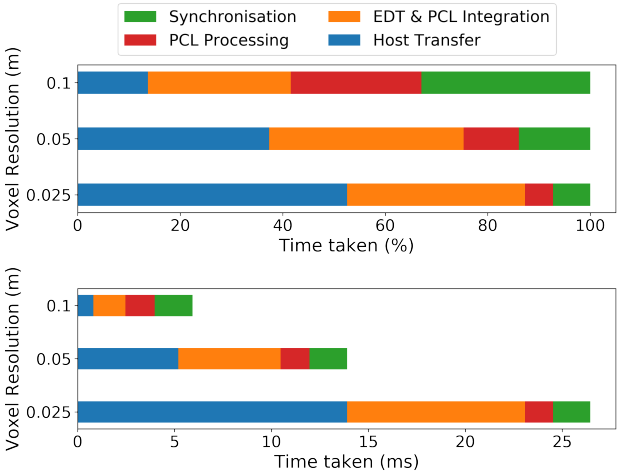



Fig. 2. A breakdown of the time spent in our GPU-Voxels-based update loop. *Synchronisation* – the time spent finding camera pose transforms within a time tolerance of pointcloud timestamps in the update queue. *PCL Processing* – time taken to resize and apply reference frame transformations to the incoming pointclouds prior to integration. *EDT and PCL Integration* – pointclouds are integrated into a probabilistic voxelmap via raycasting and a EDT is then calculated using the PBA algorithm. *Host Transfer* – time spent copying the full EDT from the GPU to memory on the host computer.

Our results show that despite the additional time required by GPU-Voxels to transfer the distance field from the GPU, it is the fastest mapping framework in all cases. Figure 2 illustrates a breakdown of the time spent in our GPU-Voxels update loop. While the time spent in *Synchronisation* and *PCL Processing* is essentially constant, we see that as we increase the voxel resolution and grid size, the time taken to transfer data from the GPU to the host becomes more significant.

## V. Integrating Mapping and Motion Planning

As previously discussed, we elected to use GPMP2 as our motion planner of choice. Using the GPMP2 framework, collision avoidance is implemented via the use of regularly spaced, time-indexed obstacle factors in a factor graph. Obstacle factors query the distance field to allocate a hinge-loss obstacle cost and an associated gradient for optimisation. In this work, all obstacle factors were linked to use the real-time updated distance field as maintained by the package used. Based on the results shown in the previous section, GPU-Voxels was used as the mapping framework for the rest of this work.

We run the mapping concurrently on a separate thread from the motion planning framework. Fast distance query access is provided to the obstacle factors by running GPU-Voxels within the same ROS node and providing them with pointers to the EDT memory address.

While we experimented with different methods of maintaining and updating a factor graph, we found the most effective way of planning to be that shown in Algorithm

TABLE I
COMPARISON BETWEEN USING SDFS AND USDFS FOR COLLISION AVOIDANCE

| | | USDF | SDF | Relative Difference (USDF/SDF) |
|---|---|---|---|---|
| Failure Rate (%) | Navigation | 2.9 | **0.16** | 18.3 |
| | Arm | **0.00028** | 0.00037 | 0.75 |
| | Whole-Body | 0.00092 | **0.00018** | 5.0 |
| Iterations ($\mu \pm \sigma$) | Navigation | $\mathbf{7.57 \pm 5.90}$ | $8.24 \pm 6.66$ | 0.9 |
| | Arm | $\mathbf{7.03 \pm 7.34}$ | $7.12 \pm 7.71$ | 1.0 |
| | Whole-Body | $5.84 \pm 4.96$ | $\mathbf{5.81 \pm 4.91}$ | 1.0 |
| Valid Trajectory Cost ($\mu \pm \sigma$) | Navigation | $68.6 \pm 504$ | $\mathbf{61.5 \pm 379}$ | 1.1 |
| | Arm | $\mathbf{7.82 \pm 60.7}$ | $7.83 \pm 60.73$ | 1.0 |
| | Whole-Body | $2.81 \pm 14.7$ | $\mathbf{2.74 \pm 11.5}$ | 1.0 |

TABLE II
COMPARISON OF TIME TAKEN (ms) TO COMPUTE DISTANCE FIELDS FROM POINT CLOUD DATA IN STATE-OF-THE-ART MAPPING FRAMEWORKS

| | | Resolution (m) | | |
|---|---|---|---|---|
| | Distance Field | 0.025 | 0.05 | 0.10 |
| Voxblox (CPU) | Signed | $1232.5 \pm 677.6$ | $210.1 \pm 124.1$ | $41.4 \pm 29.4$ |
| | Unsigned | - | - | - |
| Fiesta (CPU) | Signed | $1652.0 \pm 526.1$ | $176.8 \pm 26.9$ | $178.1 \pm 25.6$ |
| | Unsigned | $176.4 \pm 206.5$ | $31.4 \pm 17.7$ | $11.6 \pm 2.7$ |
| GPU-Voxels (GPU) | Signed | $\mathbf{36.2 \pm 8.3}$ | $\mathbf{17.5 \pm 0.4}$ | $\mathbf{6.9 \pm 1.6}$ |
| | Unsigned | $\mathbf{25.3 \pm 5.6}$ | $\mathbf{13.4 \pm 2.9}$ | $\mathbf{5.6 \pm 1.4}$ |

1. After providing the algorithm with a goal pose, consisting of a base pose $(x, y, \theta)$ and joint configuration for the arm, we estimate the time needed to achieve the goal state and construct a factor graph accordingly with a fixed $\delta_t$ between variable nodes. Prior factors are imposed on the current robot state and the goal state. We use a straight-line trajectory initialisation and optimise until convergence criteria are met (relative error decrease of $0.01$ or $50$ iterations). The trajectory is then interpolated to a finer discretisation and executed on the robot.

After the initial trajectory execution, we use timer callbacks to regularly check whether the current trajectory is collision-free and within an error tolerance of the cost when it was first calculated; this is achieved using the real-time updated EDT. If either of these criteria are not met, we obtain our current pose, re-estimate how long it will take to achieve the goal position, re-build the graph and re-optimise. To re-optimise, we use the previous trajectory and re-fit it to the new graph discretisation. This enables us to retain information from previous optimisations and we found it to provide $\approx 30\%$ speed-up in optimising when compared to using a straight-line initialisation each re-planning iteration.

While re-building the graph each iteration may seem unnecessary, it provides multiple benefits. Firstly, this method avoids book-keeping and pruning of factors in the past. Secondly, it affords us the freedom to easily re-parametrise and change the planning horizon as new information is acquired.

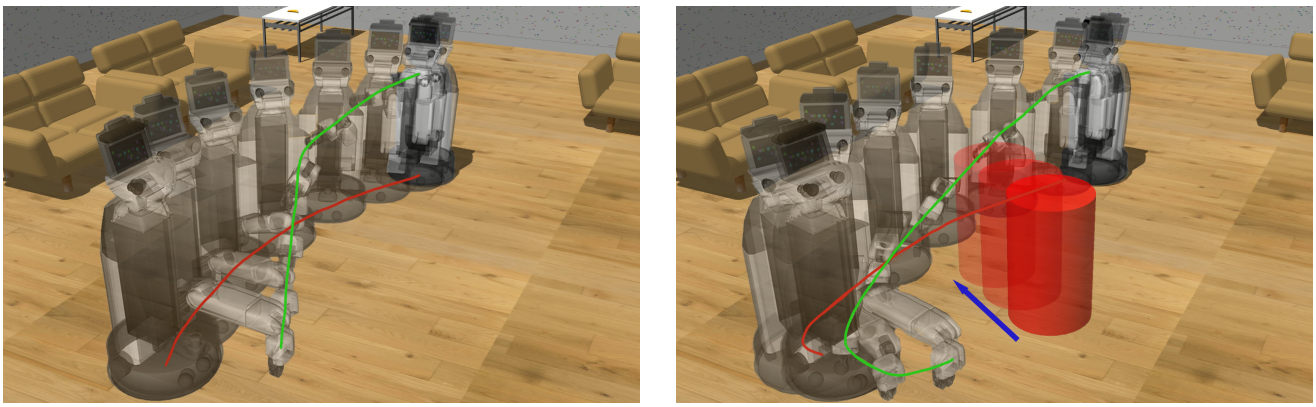
**Algorithm 1** Motion Planning Pipeline

**Input:** Goal state $\mathbf{x}_g$
**Usage:** Re-planning and execution
*Initial optimisation* :
$\mathbf{x}_c = \text{getCurrentPose}()$
$\boldsymbol{\mathcal{P}} = \text{estimateParameters}(\mathbf{x}_c, \mathbf{x}_g)$
$\boldsymbol{\mathcal{G}} = \text{buildGraph}(\mathbf{x}_c, \boldsymbol{\mathcal{P}})$
$\tau_{straight} = \text{initialiseTrajectory}(\mathbf{x}_c, \mathbf{x}_g, \boldsymbol{\mathcal{P}})$
$\tau = \text{optimise}(\tau_{straight}, \boldsymbol{\mathcal{G}})$
$\tau_{int} = \text{interpolateTrajectory}(\tau)$
executeTrajectory($\tau_{int}$)
*Re-planning* :
**while** !isGoalReached() **do**
  **if** stillValid($\tau$) **then**
    Continue
  **end if**
  $\mathbf{x}_c = \text{getCurrentPose}()$
  $\boldsymbol{\mathcal{P}} = \text{estimateParameters}(\mathbf{x}_c, \mathbf{x}_g)$
  $\boldsymbol{\mathcal{G}} = \text{buildGraph}(\mathbf{x}_c, \boldsymbol{\mathcal{P}})$
  $\tau_{init} = \text{refitTrajectory}(\mathbf{x}_c, \boldsymbol{\mathcal{P}})$
  $\tau = \text{optimise}(\tau_{init}, \boldsymbol{\mathcal{G}})$
  $\tau_{int} = \text{interpolateTrajectory}(\tau)$
  executeTrajectory($\tau_{int}$)
**end while**

### A. Simulation Experiments

We conducted simulation experiments using an 8-core Intel Core i7-9700 CPU @ 4.50 GHz and 2133 MHz RAM. GPU calculations were performed on a NVIDIA GeForce RTX 2060 graphics card (1920 CUDA cores).

The implementation described in the previous section was tested on a range of whole-body motion tasks in the presence of moving obstacles, from general navigation tasks within a room to reaching in shelves and picking objects up from the floor. Figure 3 illustrates a scenario in which the robot was tasked with achieving a goal state from which it can pick up an object from the floor. During execution, a large object (red cylinder) moves into the planned path of the robot, forcing



(a) (b)

Fig. 3. Simulated task in which the robot is required to achieved a 'pickup' goal state. During execution, a large (red) obstacle traverses across the planned robot trajectory, forcing it to re-plan online and adapt to a collision-free trajectory. Green lines illustrate the planned end-effector trajectory while red lines show the planned path of the base.

it to re-plan smoothly around the obstacle.

We found that the robot robustly avoids moving obstacles, however this is still dependent on whether the robot perceives the object—if the head camera cannot see an obstacle then inevitably it will not be registered in the distance field used for motion planning.

While we can evaluate whether our current trajectory is collision-free and within error tolerances at 250 Hz, when a new trajectory is requested, our re-plan loop runs at 10 Hz.

### B. Hardware Experiments

We implemented our system for execution on a HSR. The robot has an onboard 4th Gen Intel Core i7 (16 GB RAM) processor on which joint controllers and sensing operates. The RGB-D sensor is an Xtion PRO LIVE delivering pointclouds at VGA resolution similar to the sensor in the benchmark dataset. Motion planning and mapping were performed on an external laptop with an Intel Core i7-10875H CPU, 32 GB 2666 MHz RAM, and a NVIDIA GeForce RTX 2070 SUPER GPU (2560 CUDA cores). To increase throughput of point-cloud processing and retain 30 Hz pointcloud updates, we streamed depth images to the laptop via a wired connection and performed point-cloud conversion locally. To provide a good resolution for manipulation tasks in confined environments such as shelves, we used a voxel resolution of 2.5 cm and a grid size of $320 \times 320 \times 128$. We tested our implementation in a variety of environments with moving obstacles:

*1) Shelf Pickup:* The robot is required to reach deep into a shelf to pick up an object; during execution a human demonstrator walks into the planned robot trajectory. Later on in the same task, a wooden plank is placed across the end-effector path.

*2) Floor Pickup:* The robot is required to travel to a location and pick up a bottle from the floor. During execution, a human demonstrator walks into the planned robot trajectory.

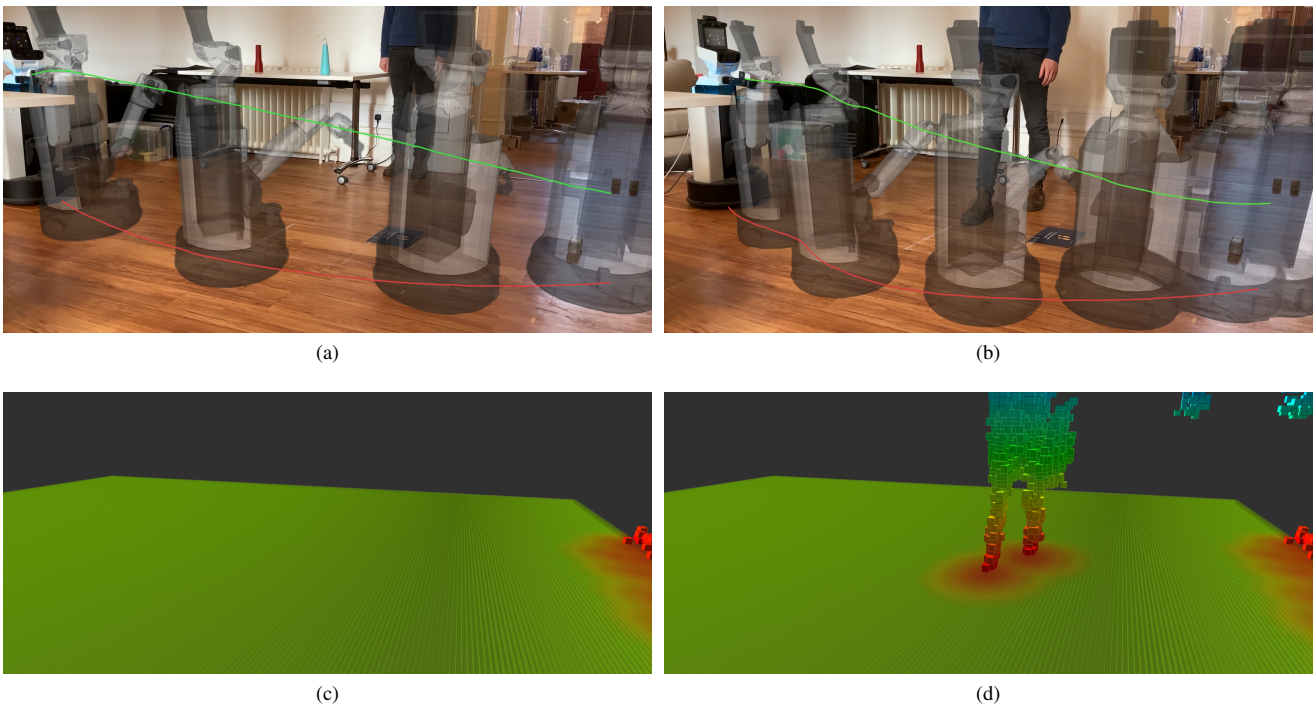
*3) Table Pickup:* The robot is tasked with picking up an object from a table. During execution, a human demonstrator moves an object across the planned trajectory.

In all three cases, the robot re-planned and successfully achieved the desired goal state without collisions. Figure 1 shows the *shelf pickup* and how the robot's resultant trajectory takes a different path to its initial plan in order to avoid an obstacle that moves into the scene during execution. Figure 4 shows the environment perception and re-planning during the *floor pickup* task. In particular, Figure 4d emphasises the clarity of the distance field reconstruction obtained in real-time. The real-time update of the SDF can also be seen in the accompanying video.

## VI. Discussion and Future Work

For mapping, in contrast to the incremental methods discussed, a limitation of GPU-Voxels is its fixed-size memory allocation on the GPU. In the current release of GPU-Voxels, the maximum number of blocks is limited to 65 535. With 1024 threads per block, this corresponds to a maximum environment grid of 67 107 840 voxels that can be computed in a single invocation. The maximum grid size is thus restricted to approximately $704 \times 704 \times 128$, or $17.6\,\mathrm{m} \times 17.6\,\mathrm{m} \times 3.2\,\mathrm{m}$ at a 2.5 cm resolution. While this may be prohibitive for large scale, multi-room operation at high resolution, for robots operating within a single workspace however, this is likely sufficient to provide high resolution mapping at a superior compute speed. In practice, one is not likely to need a high resolution for planning across large distances and a coarse representation could be used while maintaining a finer resolution in the vicinity of the robot. Current CUDA devices can support much larger number of blocks than the GPU-Voxels limit and so we believe that with further work, this package could support larger environment grid sizes.

As mentioned previously, iSAM-based re-planning can be used to provide re-planned trajectories much faster than planning from scratch — prior work [13] cites a possible one order of magnitude speed-up depending on task. The effectiveness of this method is particularly applicable when tail factors are changed, such as the goal state prior. This is relevant when tracking a moving object to grasp and is a



(a) (b)

(c) (d)

Fig. 4. Demonstration of re-planning in an unknown, dynamic environment on a physical HSR. The robot was tasked with picking up a bottle from the floor. During execution, a human demonstrator walked towards the planned robot trajectory, requiring a re-planned trajectory to be calculated and executed to avoid collision. Green lines illustrate the planned end-effector trajectory while red lines show the planned path of the base. The bottom row illustrates the distance field generated at the corresponding times; we show both a plane through the distance field and a pointcloud after thresholding at zero distance. The interactive re-planning can more readily be appreciated in the accompanying video.

promising feature which we hope to include in future work with the integration of object detection.

In building a practical system which performed both motion planning and real-time mapping of the environment, we make two notable observations which require further work. Firstly, the positioning of the head camera quickly became the primary limitation for the system. As the robot is executing and re-planning a trajectory, determining where to aim the head camera—i.e. Next Best View planning for receding-horizon trajectory optimisation—is an interesting problem in itself which is more commonly explored in the context of building 3D models of objects or structures [34], [35], rather than for use in motion planning. Naïve heuristics, such as always looking in the direction of motion or looking a specified distance along the planned trajectory, are prone to failure cases, in particular on curved trajectories. We mention this problem as an interesting observation for the community and as a project for further work. The second limitation is that inherent with performing trajectory optimisation which assumes a static environment. In previous work, we highlighted that using trajectory optimisation for re-planning in the presence of moving obstacles can result in trajectories that repeatedly plan to go into the path of the moving obstacle. In practice, this leads to sub-optimal trajectories. In future work, we will integrate the work presented here with methods such as predicted composite signed distance fields to account for the predicted future motion of moving obstacles as in time-configuration space planning [36].

Due to the common use of factor graphs in state estimation, another avenue for future work is to interleave planning using GPMP2 with state estimation. To this end, Mukadam et. al. demonstrated simultaneous trajectory estimation and planning (STEAP) of a PR2 robot operating within a known 3D workspace [37]. In their work, they repeatedly perform inference on a factor graph spanning from the start state to the goal state, while adding measurement factors, to incorporate new sensor measurements and observations, during execution. The resultant trajectory after each optimisation provides a solution to both the motion planning and state estimation problems. The primary limitation they cite is that it can only operate in "known, static environments" because the SDF computation provides a "major, computational bottleneck". We believe that by leveraging the contributions presented in our work, STEAP could be extended to provide simultaneous mapping, localisation and planning in dynamic environments.

A key implementation note is that point cloud observations are useful not only for the occupied space but also for clearing space via ray-casting. In our work, we did not wish to classify the floor as a collision object in order to provide stronger gradients around real obstacles. Points registered within $3\,\mathrm{cm}$ of the ground were used in the ray-casting for clearing, however the occupied point at the end of the ray was not inserted into the voxelmap.

Finally, transferring information between CPU and GPU (device-to-host) became a more dominating factor in com-

puting EDTs for finer resolutions. To avoid this requirement, motion planning directly on the GPU could be explored. Previous work based on GPU-Voxels performed grid-based motion planning and environment mapping directly on the GPU to eliminate the transfer costs [2], [10], [33]. Hence, a possible avenue for future work could look into implementing optimisation-based motion planning on the GPU.

## VII. Conclusion

This paper presents a fully integrated system to update the environment representation in real-time and perform re-planning in dynamic environments. We show experimentally that signed distance fields provide no significant benefit to motion planning on articulated systems when compared to their cheaper-to-compute unsigned counterparts. We analyse a selection of state-of-the-art mapping libraries and show GPU-Voxels to be a superior candidate for using in whole-body trajectory optimisation problems. We integrate GPMP2 with GPU-Voxels to produce a hybrid mapping and motion planning system which can provide real-time mapping and whole-body motion planning to smoothly avoid moving obstacles. Our findings are verified both in simulation and on a physical HSR across a range of tasks and successfully re-plan safely in response to dynamic obstacles.

## References


[1] M. Niesner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3D reconstruction at scale using voxel hashing," *ACM Transactions on Graphics*, vol. 32, no. 6, 2013.

[2] A. Hermann, F. Drews, J. Bauer, S. Klemm, A. Roennau, and R. Dillmann, "Unified GPU voxel collision detection for mobile manipulation planning," in *IEEE/RSJ IROS*, oct 2014, pp. 4154–4160.

[3] R. A. Newcombe, A. Fitzgibbon, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, and S. Hodges, "KinectFusion: Real-time dense surface mapping and tracking," in *IEEE ISMAR*, Oct 2011, pp. 127–136.

[4] T. Whelan, M. Kaess, and M. Fallon, "Kintinuous: Spatially extended kinectfusion," *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, p. 7, 2012.

[5] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison, "ElasticFusion: Dense SLAM without a pose graph," in *Robotics: Science and Systems*, vol. 11, 2015.

[6] R. Scona, M. Jaimez, Y. R. Petillot, M. Fallon, and D. Cremers, "StaticFusion: Background Reconstruction for Dense RGB-D SLAM in Dynamic Environments," in *IEEE ICRA*, 2018, pp. 3849–3856.

[7] F. Steinbrücker, J. Sturm, and D. Cremers, "Volumetric 3D mapping in real-time on a CPU," in *IEEE ICRA*, 2014, pp. 2021–2028.

[8] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots," in *IEEE/RSJ IROS*, 2019, pp. 4423–4430.

[9] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning," in *IEEE/RSJ IROS*, 2017, pp. 1366–1373.

[10] C. Jülg, A. Hermann, A. Roennau, and R. Dillmann, "Fast online collision avoidance for mobile service robots through potential fields on 3D environment data processed on GPUs," in *IEEE ROBIO*, 2018.

[11] V. Usenko, L. Von Stumberg, A. Pangercic, and D. Cremers, "Real-time trajectory replanning for MAVs using uniform B-splines and a 3D circular buffer," in *IEEE/RSJ IROS*, 2017, pp. 215–222.

[12] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *IEEE ICRA*, May 2009, pp. 489–494.

[13] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots, "Continuous-time Gaussian process motion planning via probabilistic inference," *The Int. J. of Rob. Res.*, vol. 37, no. 11, pp. 1319–1340, 2018.

[14] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments," in *ICAPS*, 2012, pp. 207–215.

[15] N. Ratliff, M. Toussaint, and S. Schaal, "Understanding the geometry of workspace obstacles in Motion Optimization," in *IEEE ICRA*, vol. 2015-June, no. June, 2015, pp. 4202–4209.

[16] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The Int. J. of Rob. Res.*, vol. 33, no. 9, pp. 1251–1270, 2014.

[17] W. Merkt, V. Ivan, and S. Vijayakumar, "Continuous-Time Collision Avoidance for Trajectory Optimization in Dynamic Environments," in *IEEE/RSJ IROS*, Nov 2019, pp. 7248–7255.

[18] S. M. LaValle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," Iowa State University, Tech. Rep., 1998.

[19] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Rob. and Aut.*, vol. 12, no. 4, pp. 566–580, 1996.

[20] S. M. LaValle, *Planning Algorithms*. Cambridge Univ. Press, 2006.

[21] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," in *Robotics: Science and Systems*, vol. 6, May 2011, pp. 267–274.

[22] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *IEEE ICRA*, May 2011, pp. 4569–4574.

[23] M. Toussaint, "Robot trajectory optimization using approximate inference," in *ICML*, 2009.

[24] F. Dellaert, "Factor Graphs and {GTSAM}," *Technical Report*, no. GT-RIM-CP&R-2012-002, pp. 1–27, 2012.

[25] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. Robot.*, vol. 24, no. 6, pp. 1365–1378, 2008.

[26] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "ISAM2: Incremental smoothing and mapping using the Bayes tree," *The Int. J. of Rob. Res.*, vol. 31, no. 2, pp. 216–235, 2012.

[27] W. Merkt, V. Ivan, and S. Vijayakumar, "Leveraging Precomputation with Problem Encoding for Warm-Starting Trajectory Optimization in Complex Environments," in *IEEE/RSJ IROS*, 2018, pp. 5877–5884.

[28] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees," *Autonomous Robots*, 2013.

[29] B. Lau, C. Sprunk, and W. Burgard, "Improved updating of Euclidean distance maps and Voronoi diagrams," in *IEEE/RSJ IROS*, 2010, pp. 281–286.

[30] T. T. Cao, K. Tang, A. Mohamed, and T. S. Tan, "Parallel Banding Algorithm to compute exact distance transform with the GPU," in *ACM SIGGRAPH I3D*, 2010, pp. 83–90.

[31] K. V. Alwala and M. Mukadam, "Joint sampling and trajectory optimization over graphs for online motion planning," 2020.

[32] K. B. Kaldestad, S. Haddadin, R. Belder, G. Hovland, and D. A. Anisi, "Collision avoidance with potential fields based on parallel processing of 3D-point cloud data on the GPU," in *IEEE ICRA*, 2014.

[33] A. Hermann, J. Bauer, S. Klemm, and R. Dillmann, "Mobile manipulation planning optimized for gpgpu voxel-collision detection in high resolution live 3d-maps," in *ISR/Robotik*, 2014, pp. 1–8.

[34] C. Collander, W. J. Beksi, and M. Huber, "Learning the Next Best View for 3D Point Clouds via Topological Features," in *IEEE ICRA*, 2021.

[35] R. Border, J. D. Gammell, and P. Newman, "Surface Edge Explorer (see): Planning next best views directly from 3D observations," in *IEEE ICRA*, 2018, pp. 6116–6123.

[36] M. Finean, W. Merkt, and I. Havoutis, "Predicted Composite Signed-Distance Fields for Real-Time Motion Planning in Dynamic Environments," in *ICAPS*, 2021.

[37] M. Mukadam, J. Dong, F. Dellaert, and B. Boots, "STEAP: simultaneous trajectory estimation and planning," *Autonomous Robots*, vol. 43, no. 2, pp. 415–434, 2019.

## Further Remarks

While the main focus of this chapter was the development of an integrated motion planning and perception framework, we wish to provide further analysis of our results addressing the question of whether signed or unsigned distance fields should be used in motion planning.

As discussed in Chapter 2, optimisation-based methods of motion planning are prone to local minima; optimised trajectories may thus result in collision even if the task itself may be solvable for a *complete* planner. To provide an appropriate comparison between signed distance fields and unsigned distance fields, we discarded results in which our motion planner failed to generate a collision-free trajectory for either type of distance field. The remaining failure cases are thus directly attributable to the kind of distance field used. The difference in failure rate between signed and unsigned distance fields arises from different gradients experienced during the optimisation process. To illustrate this, we will examine one of the 2D planning tasks used in our experiments. Figure 4.1 shows a 2D

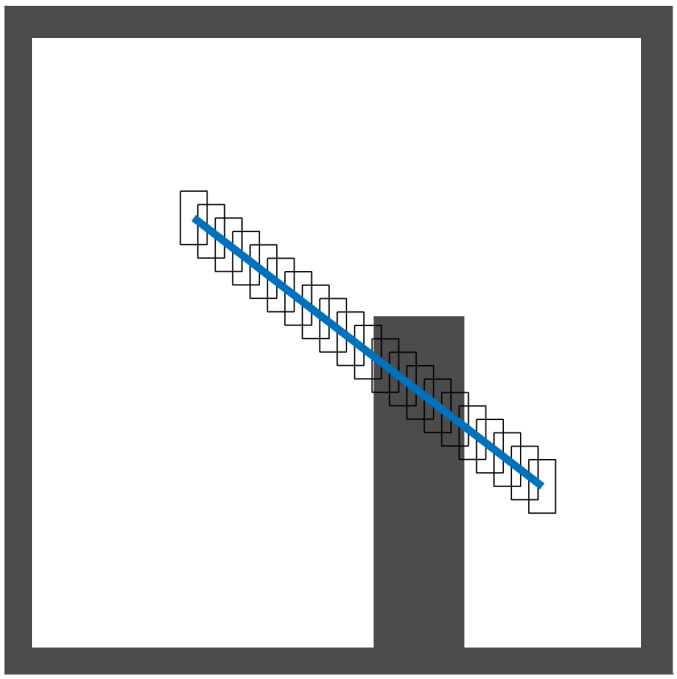
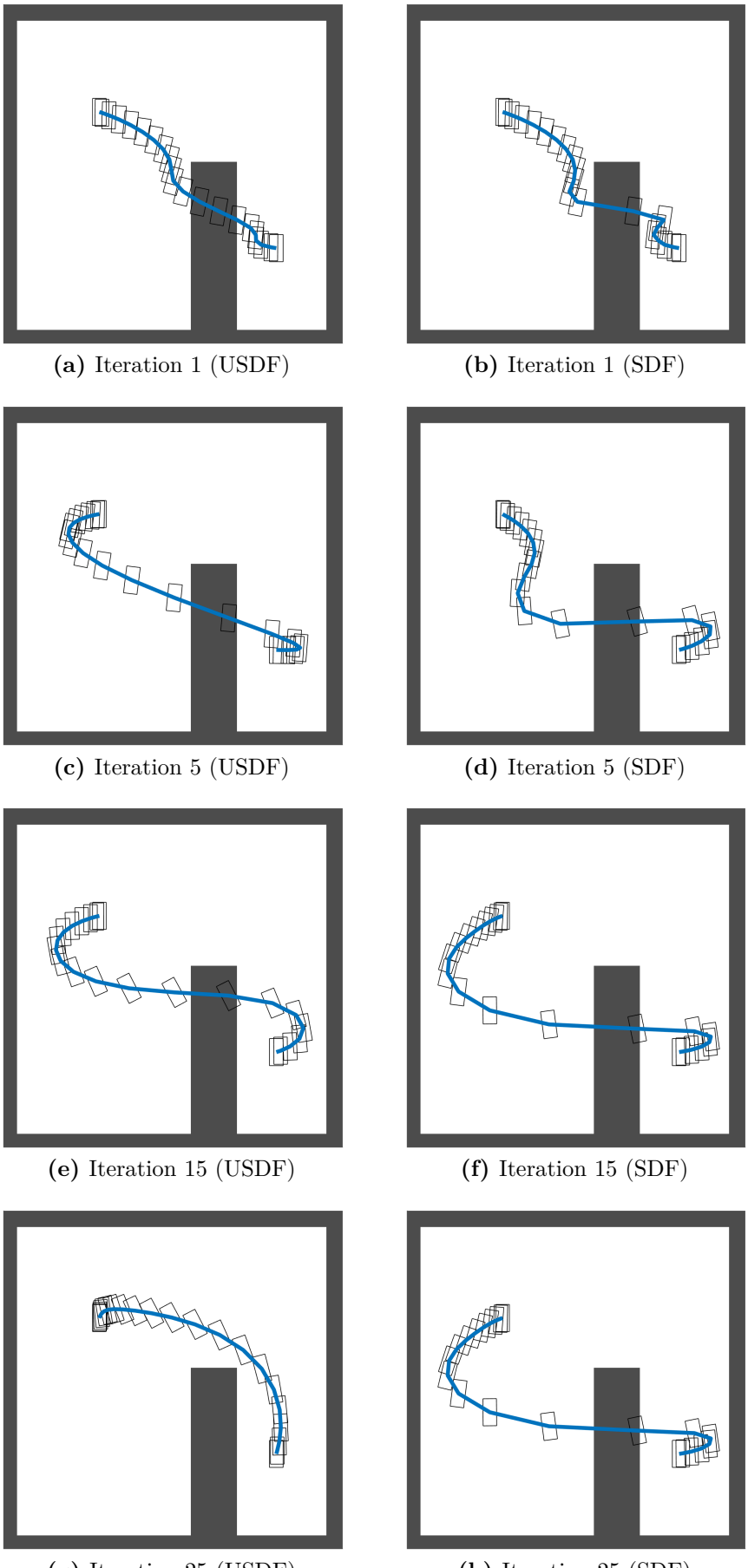


**Figure 4.1:** An example 2D motion planning task used to compare signed and unsigned distance fields. The sequence of rectangles illustrates the support states of the straight-line initialisation between the start and goal states used for the optimisation. The blue line is provided for clarity and joins consecutive support states.

environment used in our experiments, along with the trajectory initialisation for a given task superposed on top. For this scenario, we found that the motion planner returned a collision-free trajectory when using an unsigned distance field but not when using a signed distance field. The reason for this is explained by Figure 4.2 which shows a significant difference between the trajectories even after the first iteration. While the unsigned distance field does not provide gradients to push states colliding states out of collision, the signed distance can push most colliding states out of collision in the first iteration. The curvatures of the trajectories continue to diverge in further iterations; in the case of the signed distance field, the trajectory becomes stuck in local minima, while for the unsigned distance field, it is gradually optimised to become collision-free.



**(a)** Iteration 1 (USDF) **(b)** Iteration 1 (SDF)

**(c)** Iteration 5 (USDF) **(d)** Iteration 5 (SDF)

**(e)** Iteration 15 (USDF) **(f)** Iteration 15 (SDF)

**(g)** Iteration 25 (USDF) **(h)** Iteration 25 (SDF)

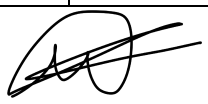
**Figure 4.2:** A comparison between the trajectories generated using signed and unsigned distance fields throughout the optimisation process for an example task.

**Statement of Authorship for joint/multi-authored papers for PGR thesis**

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

| | |
|---|---|
| Title of Paper | Simultaneous Scene Reconstruction and Whole-Body Motion Planning for Safe Operation in Dynamic Environments |
| Publication Status | Published |
| Publication Details | M. N. Finean, W. Merkt, and I. Havoutis. "Simultaneous Scene Reconstruction and Whole-Body Motion Planning for Safe Operation in Dynamic Environments". In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* 2021. |

Student Confirmation

| | | | |
|---|---|---|---|
| Student Name: | Mark Nicholas Finean | | |
| Contribution to the Paper | Developed the concepts and ideas<br>Implementation of the concepts (code)<br>Devised and conducted simulation and hardware experiments<br>Data collection<br>Produced and edited figures and the associated presentation video<br>Wrote the paper with co-authors | | |
| Signature |  | Date | 14/01/2022 |

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

| | | |
|---|---|---|
| Supervisor name and title: Dr. Ioannis Havoutis | | |
| Supervisor comments | | |
| Signature  | Date | 14/01/2022 |

# 5
# Active Gaze Control

In Chapter 4, we presented a framework for motion planning in dynamic environments that combined fast optimisation-based motion planning with a GPU-leveraged perception pipeline and distance field calculations. In this work, we achieved impressive results in simulation and hardware experiments; however, by creating an integrated framework and subjecting it to real-world problems, we identified further challenges to be addressed, motivating the work presented in this chapter.

In our previous work, we found that the primary limitation when conducting experiments was the robot's ability to 'look at the right place'. In many instances during our experimenting, the main reason for failure was that the HSR's head-mounted RGB-D camera did not maintain good positioning throughout the task and would not necessarily observe relevant changes in the environment. This problem was particularly prominent since the robot's holonomic base can move omni-directionally. A simple example that illustrates the problem is to consider sending the robot a goal command to move 1 m to the right in a previously unobserved room. A motion planner will likely plan a direct path that does not require the robot's yaw to change at all. However, in executing this trajectory, a forward-facing head camera will not observe anything in the direction of the robot's travel; this is problematic for collision avoidance since the region of space that the robot plans to traverse will not have been recently observed.

In this chapter, we pose the novel question of "given a planned robot trajectory, where should the camera be directed in order to provide both successful collision avoidance and map exploration in potentially dynamic scenes?". To solve this problem, we propose an optimisation-based approach that greedily determines the optimal camera pose for the next timestep. Building on our previous framework, we use a combination of voxelised rewards and camera motion primitives.

To the best of our knowledge, this work presents the first 'smart' controller for gaze control that considers poses based on their utility for obstacle avoidance and environment exploration. In comparison to benchmark methods, our method outperforms both in its ability to efficiently explore the surroundings, and in assisting the motion planner to generate collision-free trajectories.

The following manuscript is published in the IEEE Robotics and Automation Letters (DOI:10.1109/LRA.2021.3137545) [5], along with a supplementary video[1].

[1] `https://ieeexplore.ieee.org/document/9661405/media`

IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 7, NO. 2, APRIL 2022 1095

# Where Should I Look? Optimised Gaze Control for Whole-Body Collision Avoidance in Dynamic Environments

Mark Nicholas Finean, *Graduate Student Member, IEEE*, Wolfgang Merkt, *Member, IEEE*, and Ioannis Havoutis, *Member, IEEE*

***Abstract*—As robots operate in increasingly complex and dynamic environments, fast motion re-planning has become a widely explored area of research. In a real-world deployment, we often lack the ability to fully observe the environment at all times, giving rise to the challenge of determining how to best perceive the environment given a continuously updated motion plan. We provide the first investigation into a 'smart' controller for gaze control with the objective of providing effective perception of the environment for obstacle avoidance and motion planning in dynamic and unknown environments. We detail the novel problem of determining the best head camera behaviour for mobile robots when constrained by a trajectory. Furthermore, we propose a greedy optimization-based solution that uses a combination of voxelised rewards and motion primitives. We demonstrate that our method outperforms the benchmark methods in 2D and 3D environments, in respect of both the ability to explore the local surroundings, as well as in a superior success rate of finding collision-free trajectories – our method is shown to provide 7.4x better map exploration while consistently achieving a higher success rate for generating collision-free trajectories. We verify our findings on a physical Toyota Human Support Robot (HSR) using a GPU-accelerated perception framework.**

***Index Terms*—Mobile manipulation, RGB-D perception, collision avoidance.**

## I. Introduction

FAST re-planning, especially for robots operating in dynamic environments, is a challenging and active research area [1]–[5]. The vast majority of this prior work focuses on instances with a fixed camera, whether external or robot-mounted. While limited in the 'field-of-view,' or number of sensors, many mobile robots such as the Toyota Human Support Robot (HSR), or humanoid robots, have movable cameras or sensor heads; this presents the unique challenge of determining where the camera should be looking as the robot moves in real-world environments.

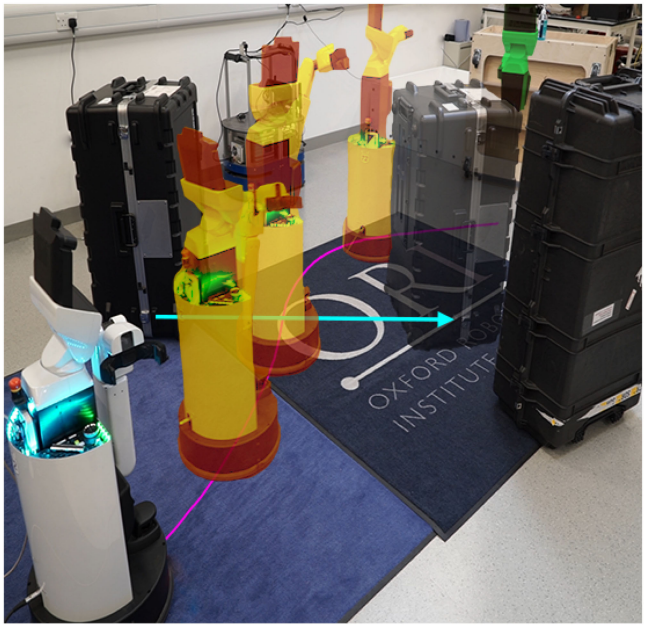


Fig. 1. Using our method of active gaze control, a Toyota Human Support Robot successfully avoided dynamic obstacles to achieve a goal state (green) from which to pick up a bottle. During execution, a human moved the leftmost large black case into the path of the planned robot trajectory – the direction of motion is shown with an arrow (cyan). Our optimised gaze control enabled the head-mounted camera to perceive the changes in the environment; the re-planning framework safely and continuously re-planned and executed the task.

In our previous research [6], we proposed and demonstrated an integrated perception and motion planning pipeline. We found that collisions and failure cases of the framework primarily occurred due to poor positioning of the head camera during trajectory execution and re-planning — in other words, the ability of any motion planner to provide collision-free trajectories is limited by the effectiveness of the perception system in continuously updating the environment representation and identifying obstacles. Robots such as the HSR have additional degrees of freedom, such as pan and tilt, to control the position of the head-mounted camera. We found that simple behavioural heuristics for the camera, such as a fixed camera pose or continuous panning, provided unsatisfactory results and merited further

Manuscript received September 9, 2021; accepted December 7, 2021. Date of publication December 23, 2021; date of current version January 4, 2022. This letter was recommended for publication by Associate Editor Kaiyu Hang and Editor Markus Vincze upon evaluation of the reviewers' comments. This work was supported by UKRI/EPSRC through the University of Oxford Centre for Doctoral Training, Autonomous Intelligent Machines, and Systems (AIMS) under Grants EP/L015897/1, EP/R512333/1, EP/S002383/1, EP/R026084/1, and EP/R026173/1, and in part by the Human-Machine Collaboration Programme, supported by a gift from Amazon Web Services. *(Corresponding author: Mark Nicholas Finean.)*

The authors are with Oxford Robotics Institute, University of Oxford, OX2 6NN Oxford, U.K. (e-mail: mfinean@robots.ox.ac.uk; wolfgang@robots.ox.ac.uk; ioannis@robots.ox.ac.uk).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3137545, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3137545

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

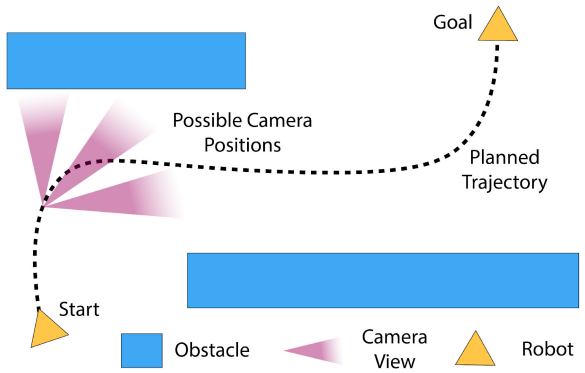



Fig. 2. At any snapshot in time during a motion planning task, the robot will have partially observed the environment and planned a trajectory accordingly. The problem that we address is in determining where a movable/repositionable, e.g. head-mounted, camera should be directed in order to provide the most relevant observations for effective collision avoidance and optimised map exploration.

investigation. To our knowledge, this work provides the first investigation into active gaze control for motion planning when constrained by a trajectory. We propose a greedy, voxelised cost-based optimization as an effective solution and show that our method outperforms heuristic approaches. Further verification of our approach is conducted on a physical HSR.

### A. Problem Statement

We address the problem of "given a planned robot trajectory, where should the camera be directed in order to provide both successful collision avoidance and map exploration in potentially dynamic scenes?" Fig. 2 illustrates the problem. The challenge largely arises because once we start working within the paradigm of dynamic and unknown environments, we must take a more cautious view of our surroundings – there is no guarantee that an area that was observed in the past will remain static. In particular, for robots operating in environments where humans co-occupy the workspace, there may be the possibility of a human crossing the planned path of the robot; the robot should thus perceive areas of space that it plans to occupy in plenty of time to react to dynamic obstacles.

## II. Related Work

Literature on robot camera positioning typically aims to address the 'Next Best View' (NBV) problem whereby the aim is to determine where the camera should be positioned to obtain maximum information gain for the reconstruction of an environment or object model [7]–[11].

Our problem is subtly different for two reasons. Firstly, in our problem the camera trajectory is constrained by the planned trajectory of the mobile manipulator on which it is mounted. Secondly, we do not wish to maximise the observance or information gain of an object, or the environment, as in the NBV problem; this in itself does not necessarily assist collision avoidance. Instead, our goal is to optimise our perception of the environment *such that* we achieve resultant trajectories that are collision-free for a given task. In other words, we want our observations of the environment to be relevant to the task and planned robot trajectory.

We believe that the concept of 'trajectory-constrained active gaze control' is a novel problem that has not previously been addressed. A simple approach is to use a fixed head position without any active gaze control. Maier *et al.* present and demonstrate an integrated approach for localisation, mapping, and planning in 3D environments using RGB-D cameras [12]. Using a NAO humanoid robot, they fix the position of the head-mounted depth camera so that the optical axis intersects the floor at a 30° angle. The authors found this to be "the best compromise between observing the near range for obstacle detection and looking ahead for localization and path planning". While the authors found this to be sufficient, the approach does not fully utilise the ability of the head camera to perceive its surroundings, particularly in a dynamic environment, and is likely to result in collision for holonomic locomotion.

Works that have considered active gaze control have applied it to different problems than what we are concerned with. Lidoris *et al.* [13] present an algorithm to combine trajectory planning and gaze direction control for usage in SLAM. Their objective for gaze control is to minimise estimation errors while exploring an unknown environment.

The work of Seara *et al.* is most similar to our problem in determining where to look for obstacle avoidance [14]. They consider active gaze control for a vision-guided humanoid and determine the pan and tilt of a mounted head camera, however, their objective is to maximise the predicted information gain for the position of *known* objects in the scene. By contrast, our approach focuses on being applicable in unknown environments with 3D collision avoidance.

## III. Greedy Voxelised Cost Optimization

Our approach is divided into two parts; the first dealing with reward assignment and the second with the optimization.

### A. Reward Assignment

Our primary concern at all times is that of safety and achieving a collision-free trajectory. Many sampling and search-based motion planners use swept trajectory occupancy to determine valid states for the robot. This assumes that we already know the occupancy of voxels within the swept trajectory. In an unknown, or dynamic environment, this swept trajectory needs to be recently observed by the perception system, e.g. a camera sensor, in order to ensure that it consists solely of free space. If obstacles occupy this space, the motion planner requires this information in order to re-plan for a collision-free trajectory. We propose using a reward for head camera poses that observe swept trajectory regions of the workspace. By observing these regions, the motion planning framework can react and re-plan to avoid collisions with obstacles that are found to be in the way.

Since trajectories can have large time horizons, we propose a tiered reward that depends on how soon the swept trajectory steps occur in time, thus prioritising the observance of states that occur sooner. This is illustrated in Fig. 3.

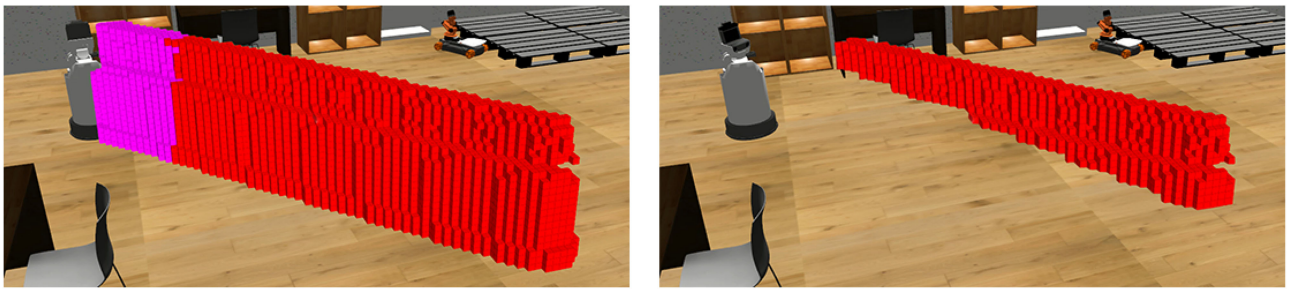


Fig. 3. A visualisation of the swept trajectory rewards as described in our approach to active gaze control. *Left:* A full swept trajectory of rewards that will contribute to the total reward of field-of-view cones that contain intersecting voxels. The earlier part of the trajectory (pink) is a higher priority region, with higher reward due to temporal proximity. *Right:* As voxels are observed, their 'time last observed' is reset to zero and thus below the threshold for allocating a reward.

Beyond safety, we wish to explore the environment and maintain a broad perception of the workspace. Being aware of peripheral surroundings is important for enabling the robot to find alternative paths but also for identifying dynamic obstacles early on and incorporating their trajectories into the motion planning. We therefore wish to encourage the robot to have an objective that rewards exploration in its perception strategy. We achieve this by penalising regions of space based on the time that they were last observed. We thus attribute the 'time last observed,' $t_i$, to each voxel in a maintained voxelmap of the environment. Each time a sensor measurement is received from the camera, the last observed time for all voxels is incremented by one. Voxels within the camera's current field-of-view cone are reset to zero since they have been observed. We thus maintain a voxelmap of observation times.

Formally, we consider a robot trajectory, $\mathbf{X}(t, T)$, that starts execution at time $t$ and has a planned duration of $T$. For such a trajectory, we construct a voxelised map of the swept occupancy, such that the occupancy of a voxel, $v_i$ is:

$$v_i = \begin{cases} 0 & \text{Not occupied} \\ \tau & \text{Occupied,} \end{cases} \tag{1}$$

where $\tau = [0, T]$ is the earliest time index of the trajectory which results in occupancy.

Using the intuition previously described, we wish to assign reward to regions of space that have not been recently observed. Among these, we want to firstly prioritise observation of the swept robot trajectory, with greater reward assigned to the earlier part of the trajectory and less reward assigned to the later section. Secondly, we wish to provide a reward for broader exploration of the environment. We reflect these three types of reward by assigning each voxel with a reward, $r_i$, determined by:

$$r_i = \begin{cases} c_1 & v_i > 0 \text{ and } v_i <= \tau_s \text{ and } t_i >= \tau_c \\ c_2 & v_i > 0 \text{ and } v_i > \tau_s \text{ and } t_i >= \tau_c \\ max(c_3 t_i, 1) & \text{otherwise} \end{cases} \tag{2}$$

where $c_1$, $c_2$, and $c_3$ are positive constants to be tuned – these parameters determine the balance between observing the near and far sections of the swept trajectory and exploring the environment. $\tau_s$ represents the farthest step in the currently planned trajectory that is deemed to be a high priority for observation. An example reward allocation for the swept trajectory volume is shown in Fig. 3. $\tau_c$ represents the user-defined threshold for a safe 'time last observed' – a smaller value will result in a more conservative behaviour that is more suited for dynamic environments with fast moving objects. In contrast, a large value is suitable for static environments. We emphasise that in (2), $c_1$ and $c_2$ rewards are only allocated if the voxel has not recently been observed, i.e. $t_i >= \tau_c$.

We use the camera's projected cone of vision (field-of-view cone) to reset voxel observation times, rather than raycasting, so as to avoid reward allocation for regions of space that we know cannot currently be observed. Raycasting in the perception pipeline is performed between the camera and an observed point-cloud against which the rays will terminate in collision. However, in the gaze control problem, we do not consider whether regions of space are denoted as free, occupied, or unknown—only whether they have been observed recently enough. For example, some camera positions may result in facing a wall; in these cases, a raycasting approach would terminate at the wall (leaving the space behind the surface as 'unknown'), and not reset 'time last observed' counters on the other side of the wall. The lack of resetting these counters may wrongly encourage the optimization to face the wall again so as to observe these unobservable voxels. Hence, we instead opt for a field-of-view cone which instead can result in voxels on the other side of surface boundaries being labelled as observed for the purposes of gaze control optimization. On first inspection, the reader may be inclined to think that in these instances, the method will now discourage the robot from observing the other side of the wall for future times. However, this is not a problem due to our method of incrementing the 'time last observed,' independent of their occupancy, and applying a threshold, $\tau_c$. In our wall analogy, over time, the robot will try to allocate reward to the previously unobservable areas of space and if the robot has moved to a location in which the robot's vision is no longer obstructed by a wall, then the voxels may be observed in the optimization.

### B. Optimization

With the reward of each voxel updated at every timestep, we consider a set of motion primitives local to the camera's current position. For each motion primitive, we consider the 'field-of-view' cone and calculate the weighted sum of rewards.

1098 IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 7, NO. 2, APRIL 2022

We implement a greedy optimization approach and select the motion primitive with the greatest reward for execution.

When considering a two-dimensional workspace, we found that spatially discretised costs are easily performed using a CPU. However, in three-dimensional workspaces, we leveraged our previous work [6] by using a GPU-accelerated perception framework based on GPU-Voxels [15], [16], enabling us to maintain a reactive update frequency for the gaze control. By integrating gaze control with our previous framework, we are able to re-use the same voxelmap information for both whole-body motion planning and gaze control optimization. In a large $256 \times 256 \times 64$ voxelised workspace, at a resolution of $5$ cm, we found that our gaze controller operates at $4.9$ Hz while optimising over 144 motion primitives – we note that within each iteration to determine the best camera position, not all 144 motion primitives will necessarily undergo full evaluation as those which violate joint constraints will be terminated early and allocated a maximally negative reward. Higher rates could be achieved by reducing the voxelmap size, reducing the maximum distance of the camera's field-of-view cone, or by considering fewer motion primitives. Hardware specifications are: NVIDIA RTX 2060 GPU, 8-core Intel Core i7-9700 CPU @ 4.50 GHz and 2133 MHz DDR4 RAM.

## IV. Heuristic Baseline Methods

Since, to the best of our knowledge, there does not appear to be a comparable 'smart' approach in the literature, we consider three baseline heuristic methods to compare against our proposed optimization method:

1) Constant: The camera remains in a fixed position with respect to the base of the robot.
2) Panning: The camera continuously pans at a constant rate throughout the task. The tilt of the camera is fixed such that the optical axis intersects the ground at 30 degrees.
3) Look Ahead: The camera is directed towards the future base position of the robot at a fixed number of time steps ahead in the planned trajectory.

To evaluate the methods, we perform benchmarking on a variety of tasks in both 2D and 3D. In all experiments, the robot is not provided with any knowledge of the environment prior to receiving a goal command, after which all mapping and perception is obtained via live sensor measurements.

## V. Evaluation

We evaluate our method in static 2D and 3D environments (Sections V-A and V-B respectively), followed by evaluation of the methods in a dynamic 3D simulation. Finally, we perform hardware experiments in dynamic scenes (Section V-C).

### A. 2D Benchmarking

In 2D, we consider a $1000 \times 1000$ workspace in which the outer perimeter is occupied by walls. Static obstacles are generated randomly within the workspace in which a robot performs motion planning tasks between start and goal state pairs $(x, y, \theta)$.

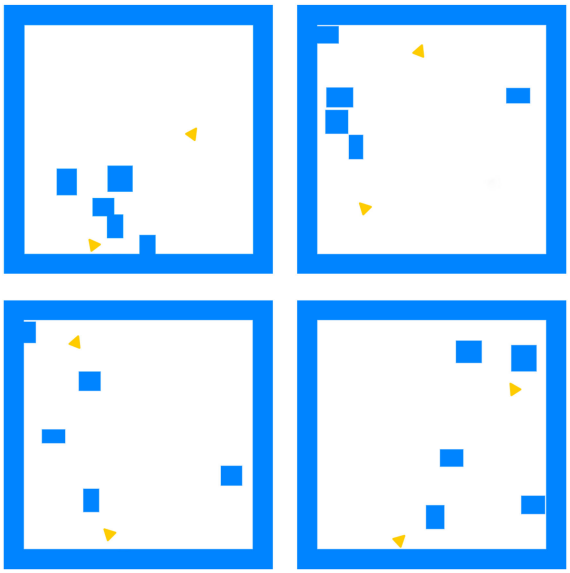


Fig. 4. A sample of 2D motion planning tasks comprising of a perimeter wall, obstacles (blue), and a start/goal state pair (yellow triangles).

We generate 150 different environments, comprising of five static rectangular obstacles of random size between 10 and 30 units (cells) in each dimension. We generate 20 different start/goal pairs for evaluation on these environments. A *task* comprises of an environment selection and a start/goal pair. For each task, we implemented each of the four head behaviours in tandem with a GPMP2 motion planner [4] executed in a receding-horizon manner as detailed in our previous work [6]. We consider a mounted camera with pan joint values in the range of $[-\frac{\pi}{2}, \frac{\pi}{2}]$, relative to the front of the robot. For the motion primitives in our optimization, we consider values at $\frac{\pi}{16}$ intervals within this range, resulting in 17 possible head camera positions to greedily optimise. To evaluate the success of each method, we consider both whether the resultant trajectory was collision-free, and the percentage of the environment explored during execution. A selection of 2D task environments are shown in Fig. 4.

By conducting a parameter sweep for a series of similar tasks, we found the optimum values of $c_1$, $c_2$, and $c_3$, in our setup to be $10^6$, $10^3$, and 1 respectively. We found that values of $\tau_s = 3$ and $\tau_c = 3$ performed well. For the field-of-view cones, we use an opening angle of $\frac{\pi}{2}$ and a distance cutoff of 200 pixels.

Of the 3000 tasks in total, we retained a randomly sampled subset of 2000 tasks in which at least one method succeeded in generating a collision-free trajectory – this provides us with a fair comparison across the methods and provides assurance that the motion planning problem was solvable.

Results from the 2D benchmarking experiments indicate that our *Optimised* method, along with the *Look Ahead* method, achieve the greatest success rates for producing collision-free trajectories (95%). This is likely due to maintaining a much better perception of the region of space directly in front of the robot. As shown in Fig. 7, these two methods result in the robot traversing cells which have typically been more recently observed. We note again that this feature is of particular interest

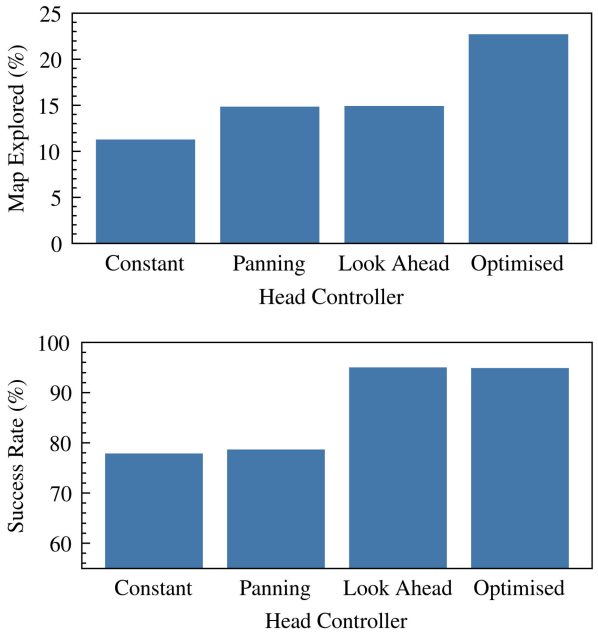



Fig. 5. Top: A comparison of the mean map exploration, as a percentage of the total workspace, achieved by each of the gaze behaviours on our 2D dataset. Bottom: A similar comparison of the success rate for each gaze behaviour in achieving collision-free resultant trajectories. We see that our *Optimised* method provides significantly greater map exploration while maintaining the joint highest success rate for achieving collision-free trajectories.

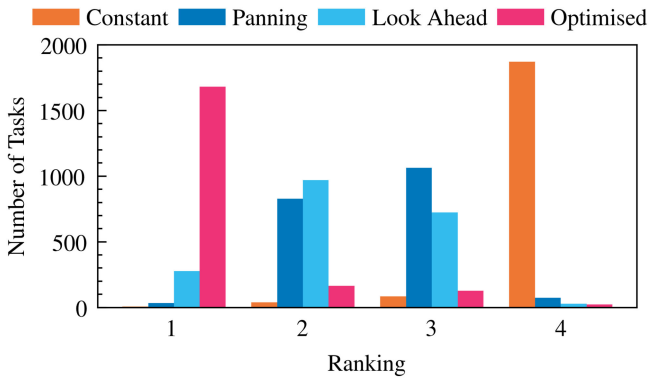



Fig. 6. For each task, we rank the gaze control behaviours in a tiered approach, firstly prioritising successful collision avoidance, followed by the map coverage achieved during execution. Across our 2D dataset of tasks, we find that our *Optimised* method outperforms all other methods in 94% of tasks.

when considering dynamic environments, since an old observation of a cell may no longer be valid. The additional benefit of our method is a significantly enhanced perception of the surrounding environment due to the exploration reward in the objective function. Fig. 5 shows the mean map coverage achieved by each of the four methods over the subset of tasks in which all methods succeeded in finding collision-free trajectories — our method provides a 52 % to 102% improvement over the benchmark approaches in exploring the environment during execution.

As previously discussed, the primary consideration of an effective gaze controller is in providing successful collision avoidance, followed by any additional ability to observe the surroundings. We therefore ranked the four methods hierarchically

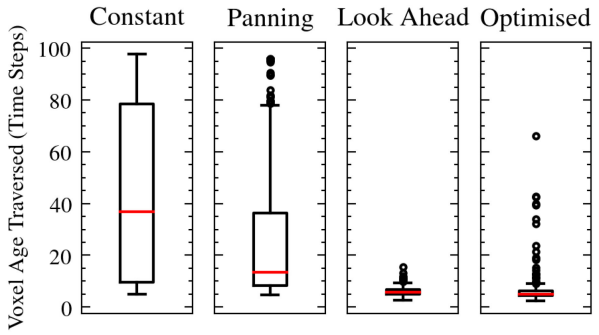



Fig. 7. Distribution of the last time that cells were observed at the time of robot occupancy. *Last observed times* were initialised and clipped at 100 time steps. Times for voxels that are observed by the camera are reset to zero and incremented for each subsequent time-step that they are not observed. To ensure safe locomotion, cells should be recently observed prior to the robot moving to occupy them. Our *Optimised* method allows us to parameterise our 'cautiousness' of the environment and trade-off the conservative behaviour of the *Look Ahead* method, with greater map exploration.

– firstly by whether trajectories were collision-free and secondly by the total map coverage. Results of this ranking are shown in Fig. 6 whereby our method ranks first 94% of the time.

### B. 3D Benchmarking

We conducted a series of 3D simulation experiments using a Toyota HSR robot in a variety of static Gazebo environments as shown in Fig. 8, as well as a dynamic environment shown in Fig. 9. For each environment, a base goal $(x, y, \theta)$ was provided and our entire pipeline (integrated motion planning, perception, and head controller) was executed so that the HSR would autonomously navigate towards the goal. In each task, after receiving the goal location, the head camera was instructed to first look towards the goal destination and calculate the initial trajectory plan; after which all head movements were controlled by the head controller under investigation.

In our optimization, we consider 16 joint values for the panning angle and nine values for the tilt, resulting in 144 possible head configurations to optimise over. We recorded whether trajectories were collision-free and the total portion of the map observed during execution. Each task was repeated 10 times for each head behaviour.

In our simulated 3D and hardware experiments, we use a 0.5 s interval between motion planning time-steps and found that a threshold value of $\tau_s = 15$ performed well, corresponding to a time threshold of 7.5 s. We similarly found $\tau_c = 150$ voxelmap updates to perform well. For reference, although we continuously re-estimate the trajectory time-horizon during execution, a typical starting horizon for tasks such as in the hardware experiment is $\sim 35$ s. We use a pyramidal field-of-view cone for an ASUS Xtion Pro Live camera, matching the parameters for the sensor characteristics [17], with horizontal and vertical angles of $58^\circ$ and $45^\circ$ respectively. In line with the sensor's recommended 'distance of use,' we apply a distance cutoff of 3.5 m .

Our findings are further validated in the 3D case with results shown in Fig. 10. In particular, we find that our method is

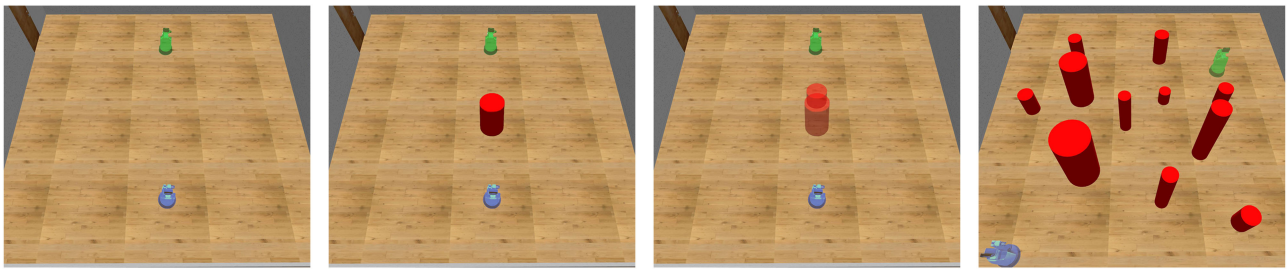
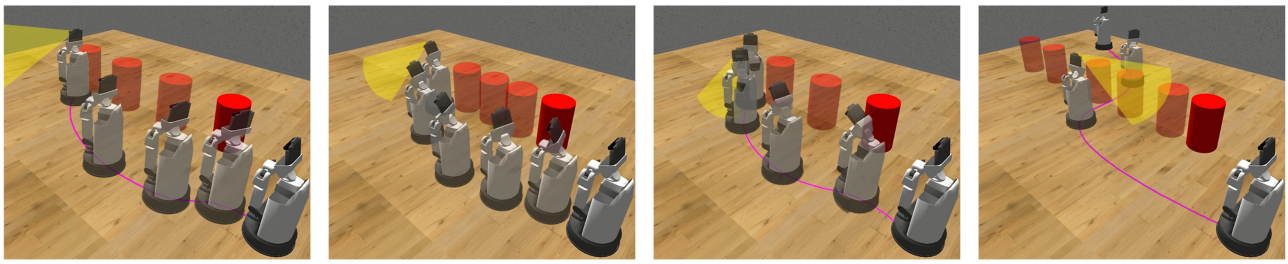


Fig. 8. Aerial view of the four static 3D simulation environments: 'Free,' 'One Obstacle,' 'Occluded' (a second smaller obstacle is hidden behind the first), and 'Clutter'. Start and goal robot states are shown as blue and green respectively.



(a) Constant (b) Panning (c) Look Ahead (d) Optimised (Ours)

Fig. 9. Our simulated 3D dynamic experiment highlights the benefit of using an active gaze control approach that aims to maintain a broader perception of the environment. In the examples shown, all three benchmark methods demonstrated collisions with the dynamic obstacle. In contrast, our *Optimised* method, which balances the objectives of map exploration and observing the swept volume of the robot's currently planned trajectory, is able to successfully perceive and avoid the dynamic obstacle.

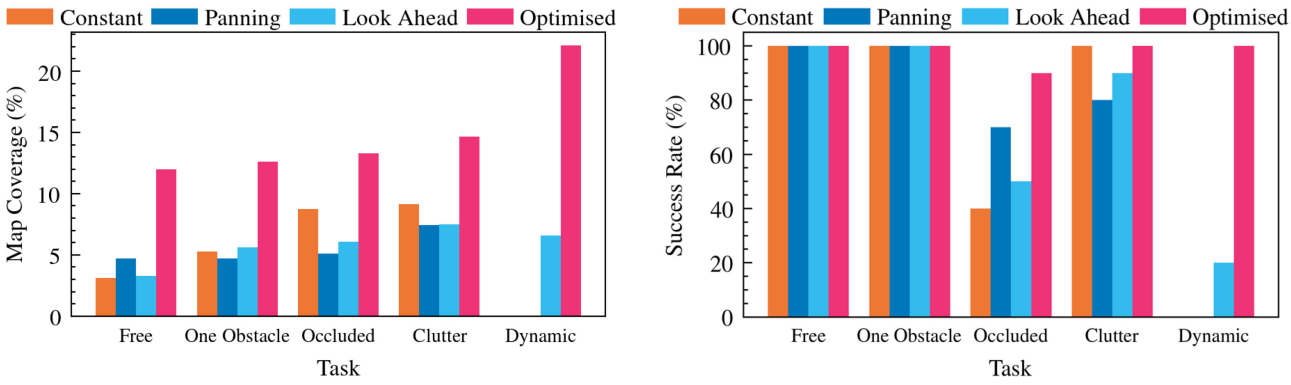



Fig. 10. Resulting success rate and mean map coverage achieved by each gaze control method across our 3D simulation tasks. Our *Optimised* method provides consistently high success rates in generating collision-free trajectories, while providing up to 7.4× improved map exploration in the static environments. Crucially, in the dynamic environment, the *Optimised* approach was the only method to reliably achieve collision-free trajectories.

more robust at providing collision-free trajectories and typically provides 2-3x greater map exploration during execution on the static environments, despite taking a similar amount of time. In particular, we note that our proposed optimization method provides a much higher success rate than the benchmarks on the 'Occluded' environment. In this environment, vision of a second smaller obstacle is occluded by the first. The benchmark methods would typically fail to observe the back of the first obstacle as a trajectory around it was executed, thus resulting in collision.

In the 3D dynamic task, the *Constant* and *Panning* methods failed to generate a collision-free trajectory in any of the 10 repeated trials. The *Look Ahead* behaviour found a collision-free trajectory in two of the trials, narrowly passing in front of the obstacle. In contrast, using the *Optimised* approach resulted in a 100% success rate and achieved significantly greater map coverage, as shown in Fig. 10.

For the dimensions given previously, the generation of the costmap takes approximately 0.5 ms. Cost evaluations for each of the primitives are then performed using this costmap. We found the cost evaluation for a single camera primitive to be $2.4 \pm 0.4$ ms . In the optimization, camera primitives that result in a position outside the allowed joint ranges bypass the optimization and return an infinite (negative) cost. Over an entire trajectory, in which the camera position was determined 160 times, our optimization approach achieved a mean computation

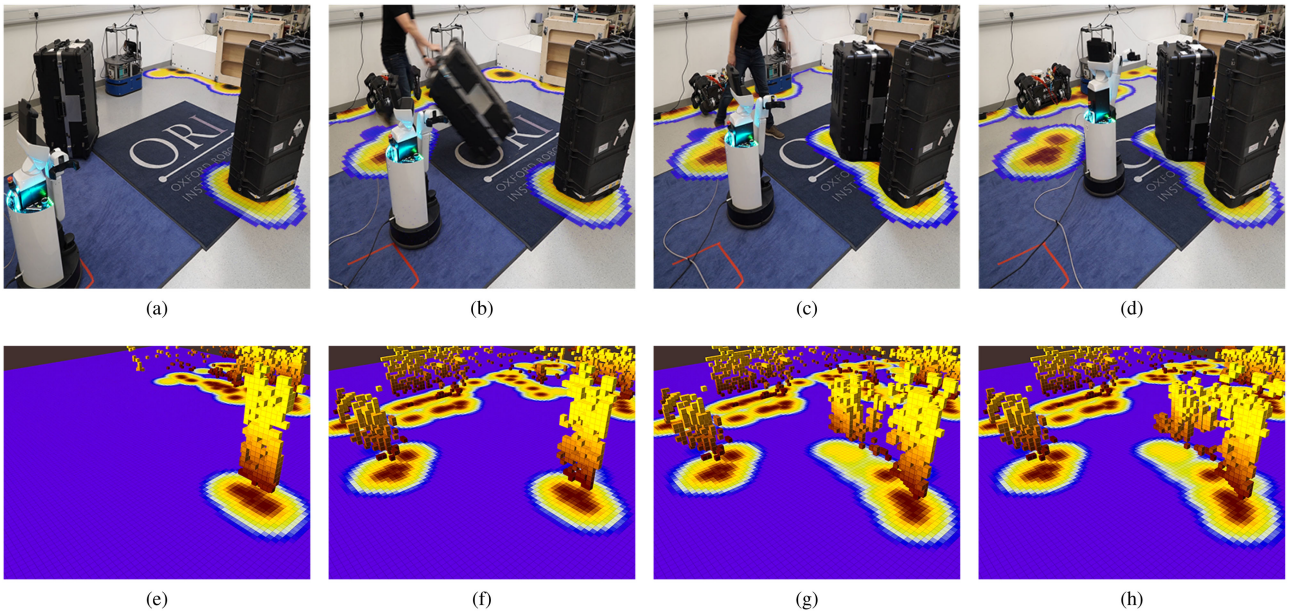


Fig. 11. We validate our proposed method for optimised gaze control on a 8-DoF Toyota Human Support Robot in a live dynamic environment. With time increasing in the images, from left to right, we overlay the calculated ground distance field generated in real-time by the perception system. Between Fig. 11(a) and Fig. 11(b), the camera perceives the large obstacle prior to moving. As the dynamic obstacle is moved, the gaze controller continues to balance map exploration with maintaining recent observations of the swept whole-body robot trajectory. In Fig. 11(c), we see that the robot observes the dynamic changes in the environment and is able to quickly re-plan accordingly to avoid collision. Fig. 11(d) shows the robot taking a collision-free path around the newly observed obstacle. Note that while we visualise the projected ground distance fields, our method operates using the full 3D voxelmap as shown in the row of images below.

time of $205 \pm 32$ ms to determine the best camera pose for the next step.

### C. Hardware Experiments in Dynamic Environments

To demonstrate our approach on hardware, we use a physical Toyota HSR – an 8-DoF mobile manipulator with a holonomic base and a head-mounted Asus Xtion Live RGB-D camera. The head motion is controlled by a further two degrees of freedom (pan/tilt). The robot is provided with a whole-body goal state to pick up a bottle on the other side of a room. During execution, a human walks into the scene and places a large obstacle in the planned path of the robot. The task is shown in Fig. I-A, while a series of snapshots from this experiment are shown in Fig. 11.

We found that by using our method, the robot successfully re-observed the space in front of it, perceiving the dynamic change in the environment and successfully re-planning its trajectory to avoid obstacles and pick up the bottle. Due to the height of the head (and the torso translation joint), this was particularly apparent when the obstacle crossed the path close to the robot; the heuristic methods would typically look *over* the obstacle and fail to re-observe the space immediately in front of the robot's base while the optimised method would alternately observe the space in front of the robot and the last planned robot trajectory. As a result, the observed behaviour of waiting until the path was clear appeared natural.

## VI. Discussion

Our greedy optimization approach proves to be a simple solution to the trajectory-constrained gaze control problem that we have described. By only considering the next camera pose, although not meeting real-time requirements of $\sim 30$ Hz, we are able to achieve a sufficiently high update rate of $\sim 5$ Hz. Over longer time-horizons, our method will likely provide sub-optimal solutions that are less smooth than gaze trajectory optimization performed over a time horizon. Optimising over longer horizons is an interesting topic for further work, however the main concern is potentially large compute times arising from generating and evaluating rewards for subsequent time-steps. This limitation is also shared by the state-of-the-art solutions for the typical next best view problem.

The benefits of our approach will inevitably vary depending on the specifications of the robot on which it is implemented. For example, robots with a slower, more restricted head motion are likely to see smaller gains in map coverage. Conversely, as hardware improves and systems become more agile, we expect that our approach will lead to further performance gains across a range of platforms.

In examining the trajectory-constrained gaze control challenge and developing our optimization-based approach, we have primarily focused our testing on static environments. Whilst we demonstrate our approach in dynamic environments, to ensure that our approach provides a robust solution, further work is required in exploring the link between the tuneable parameters and the dynamic obstacles that we expect to encounter. While it would be interesting to conduct further experiments in complex scenarios, and stress-test the approach in the presence of non-convex obstacles, these tests are more relevant for testing the motion planning method since the gaze control optimization, as presented here, is agnostic to obstacles; we will thus explore this in future work. Additionally, we will explore the incorporation of additional priors that may be available, such

as semantic information, into the optimization to better observe dynamic obstacles. Humans in particular are likely to move in smooth, structured motions with a goal in mind — a probability distribution over a known workspace could thus be used to represent the places that a human is likely to occupy – these regions can be prioritised in the optimization. Within the current framework, this could be achieved by assigning additional attributes to voxels for incorporation into the reward function. Further work in this area will extend our previous work [18] of using predicted composite signed distance fields to account for predicted dynamic obstacle trajectories; this framework requires the perception system to observe and track dynamic obstacles local to the robot. Accounting for the predicted movement of moving obstacles will help us avoid scenarios in which the motion planner repeatedly plans for the robot to go into the path of the moving obstacles.

In this work, our primary focus has been to make observations that are relevant for collision avoidance, with additional exploratory reward being aimed at observing regions of space that have not been recently observed. An interesting avenue to explore in future work would be to combine this with the typical NBV problem and consider information gain in the reward function to promote efficient reconstruction of the environment. The difficulty in this approach will likely be determining the trade-off between the rewards for reconstructing the environment and for collision avoidance, i.e. the swept trajectory term.

A limitation that we observed when using active head behaviours is an increased error in pointcloud synchronisation and the effects of a rolling vs global shutter. Fast camera movements can result in 'phantom' observations as well as inaccuracies in the position of observed objects. These errors can be mitigated by both maintaining a tight constraint on the time-synchronisation of robot joint states and received sensor measurements, as well as limiting the speed of head camera movements.

On a similar point, our method uses a weighted reward which determines the trade-off between exploring the environment and monitoring the planned trajectory ahead. These weights are likely to be determined by the environment in which the robot is operating and may evolve over time. In static environments, or scenes in which dynamic obstacles move more slowly, the time threshold can be increased.

The problem that we have addressed in this work could also merit further investigation with reinforcement learning methods, particularly for known environments where data can be accumulated over time. We believe that this may be applicable since the reward function is intuitive yet difficult to parameterise, i.e. we wish to reward better observance of the environment and behaviours that result in collision-free trajectories.

## VII. Conclusion

We provided the first description of active gaze control for trajectory-constrained mobile robots and proposed a novel solution that uses a greedy optimization of voxelised rewards across motion primitives. We compared our method to several benchmark approaches, in 2D and 3D, and show that it outperforms in both collision avoidance and general perception of the environment. In a 3D dynamic environment, we demonstrate our method to be the only approach that can reliably achieve a collision-free trajectory and avoid the moving obstacle. We further verified our findings on a physical Toyota Human Support Robot (HSR), using a GPU-accelerated perception framework, and demonstrated that our system can robustly operate in unknown and dynamic environments using live sensor measurements.

## References


[1] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments," in *Proc. Int. Conf. Automat. Plan. Scheduling*, 2012, pp. 207–215.

[2] S. Li and J. A. Shah, "Safe and efficient high dimensional motion planning in space-time with time parameterized prediction," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 5012–5018.

[3] P. S. Schmitt, F. Witnshofer, K. M. Wurm, G. V. Wichert, and W. Burgard, "Modeling and planning manipulation in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, 176–182.

[4] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots, "Continuous-time Gaussian process motion planning via probabilistic inference," *Int. J. Rob. Res.*, vol. 37, no. 11, pp. 1319–1340, 2018.

[5] D. Kappler *et al.*, "Real-time perception meets reactive motion generation," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1864–1871, Jul. 2018.

[6] M. N. Finean, W. Merkt, and I. Havoutis, "Simultaneous scene reconstruction and whole-body motion planning for safe operation in dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021.

[7] C. Potthast and G. S. Sukhatme, "A probabilistic framework for next best view estimation in a cluttered environment," *J. Vis. Commun. Image Representation*, vol. 25, no. 1, pp. 148–164, 2014.

[8] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon next-best-view planner for 3D exploration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1462–1468.

[9] R. Border, J. D. Gammell, and P. Newman, "Surface edge explorer (see): Planning next best views directly from 3D observations," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 6116–6123.

[10] C. Collander, W. J. Beksi, and M. Huber, "Learning the next best view for 3D point clouds via topological features," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 12207–12213.

[11] M. Krainin, B. Curless, and D. Fox, "Autonomous generation of complete 3D object models using next best view manipulation planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 5031–5037.

[12] D. Maier, A. Hornung, and M. Bennewitz, "Real-time navigation in 3D environments based on depth camera data," in *Proc. IEEE Humanoids*, 2012, pp. 692–697.

[13] G. Lidoris, K. Kühnlenz, D. Wollherr, and M. Buss, "Combined trajectory planning and gaze direction control for robotic exploration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2007, pp. 4044–4049.

[14] J. F. Seara, K. H. Strobl, and G. Schmidt, "Information management for gaze control in vision guided biped walking," *in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.,* 2002, vol. 1, pp. 31–36.

[15] A. Hermann, F. Drews, J. Bauer, S. Klemm, A. Roennau, and R. Dillmann, "Unified GPU voxel collision detection for mobile manipulation planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 4154–4160.

[16] C. Jülg, A. Hermann, A. Roennau, and R. Dillmann, "Fast online collision avoidance for mobile service robots through potential fields on 3D environment data processed on GPUs," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2018, pp. 921–928.

[17] D. M. Swoboda, "A comprehensive characterization of the asus xtion pro depth sensor," in *Proc. Eur. Conf. Education Robot.*, 2014, pp. 1–5.

[18] M. N. Finean, W. Merkt, and I. Havoutis, "Predicted composite signed-distance fields for real-time motion planning in dynamic environments," *Proc. Int. Conf. Automat. Plan. Scheduling*, vol. 31, no. 1, pp. 616–624, 2021.

## Further Remarks

Since publication, we have identified a few remarks that would supplement this paper and provide further clarity on specific points.

Firstly, in the presented paper, we use a hierarchical ranking method to evaluate each gaze control approach. As discussed in the paper, this arises from the want of providing an evaluation metric that accounts for collision avoidance being the primary consideration while also accounting for exploratory performance. In the two-dimensional scenarios, we felt this was a particularly valuable method of evaluating the approaches. The look-ahead approach and our proposed method performed similarly well regarding their ability to result in collision-free trajectories. In contrast, our method provided a 52% boost in map coverage as measured by the success rate across all tasks. However, the hierarchical ranking method emphasises how often we can expect to obtain this superior performance – we show that our method is the best-performing approach 94% of the time. Our results suggest that our method is not the optimal controller for a relatively small number of cases. These cases naturally arise from our method using an optimisation approach in which exploratory and collision avoidance terms are formulated as a trade-off. We could further tune parameters to exhibit more conservative behaviour and consistently observe points directly in front of the robot; this will result in better collision avoidance, but this will come at the cost of degraded performance with respect to map exploration across tasks. We believe that future work investigating adaptive parameter tuning, dependent on the task and perceived environment, could alleviate or solve this problem.

Following on from the previous point, in the paper's discussion section, we mention that another exciting area for future work would be incorporating semantic information. Such information could be incorporated at multiple levels within our proposed method, firstly, at the global or 'task' level of parameter tuning. While we provide the reader with our choice of parameters and intuition behind their choosing, parameter selection is likely dependent on the specific task and environment in question. Cluttered and highly dynamic environments pose a significant risk of collision to the robot and require conservative motion planning and perception

behaviours. Such an environment can be identified by semantics such as the presence of multiple persons and a high density of obstacles in the workspace, such as tables and walls. Conversely, sparser environments which lack the presence of dynamic obstacles may enable us to relax our need to re-observe regions of space and instead focus on more exploratory behaviour. Such an investigation would be well-suited to using deep learning methods, using data over numerous trials to learn the optimum mappings of parameters for different tasks. The second way we could incorporate semantic information is at the voxelised environment representation used for reward assignment. As semantic labels are identified at the image level, these can be assigned as attributes to voxels during raycasting. With semantic labels now assigned for regions of space, one can vary the optimisation parameters spatially; intuitively, regions of space in which a door or table are present may be re-observed less frequently than those in which a dynamic obstacle has been observed. This approach could similarly be performed by explicitly modifying the reward function and introducing additional terms that favour the observance of specific semantic labels.

The optimisation-based solution that we present greedily selects camera poses. One of the disadvantages of this proposed approach is that it can result in jerky movements when transitioning between consecutively selected sensor poses. Within this method, one can minimise this effect by reducing the set of considered sensor poses to those closer to the current pose; however, this will inevitably reduce the effectiveness of the optimisation. The natural solution to this is to optimise the camera poses over a long time-horizon and even consider the dynamics of the sensor pose transitions – the problem then becomes very similar to motion planning itself and producing a smooth trajectory for execution. Our primary concern with such an approach is the associated computational cost. As proposed, our method a runs at $\sim 5\,\text{Hz}$ which is sufficiently fast to enable pose updates at a comparable rate to the $\sim 10\,\text{Hz}$ update rate of our motion re-planning framework presented in Chapter 4. By considering and evaluating camera poses over multiple time steps, the computational time associated with the optimisation will increase while also introducing a performance trade-off by introducing a smoothness constraint into the

optimisation. Investigating the addition of a smoothness constraint and optimising over multiple camera poses is an exciting topic for further research.

**Statement of Authorship for joint/multi-authored papers for PGR thesis**

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

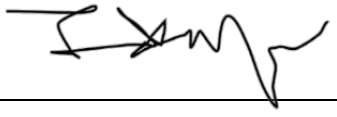
| | |
|---|---|
| Title of Paper | Where Should I Look? Optimised Gaze Control for Whole-Body Collision Avoidance in Dynamic Environments |
| Publication Status | Published |
| Publication Details | M. N. Finean, W. Merkt, and I. Havoutis. "Where Should I Look? Optimised Gaze Control for Whole-Body Collision Avoidance in Dynamic Environments". In *IEEE Robotics and Automation Letters.* 2021. |

Student Confirmation

| | | | |
|---|---|---|---|
| Student Name: | Mark Nicholas Finean | | |
| Contribution to the Paper | Developed the concepts and ideas<br>Implementation of the concepts (code)<br>Devised and conducted simulation and hardware experiments<br>Data collection<br>Produced and edited figures and the associated presentation video<br>Wrote the paper with co-authors | | |
| Signature |  | Date | 14/01/2022 |

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

| | | |
|---|---|---|
| Supervisor name and title: Dr. Ioannis Havoutis | | |
| Supervisor comments | | |
| Signature  | Date | 14/01/2022 |

# 6

# Integrating Mobile Manipulation with Human Trajectory Prediction

In Chapter 3, we presented the concept of *predicted composite distance fields* (PCDFs) and demonstrated their use in enabling collision avoidance of the predicted trajectories of moving obstacles. Limitations of this work included the lack of live sensing and the use of a simple trajectory prediction method, i.e. a Kalman filter. Additionally, we hypothesised that the power of PCDFs could be amplified by using GPUs to leverage the parallelism in the method.

In Chapter 4, we presented an integrated framework using a GPU-based perception pipeline with fast trajectory optimisation-based motion planning. This work provides a natural building block on which we could further explore and demonstrate the use of PCDFs in enabling trajectory optimisation-based motion planners to account for the predicted motion of moving obstacles.

In this final contribution, we build upon the GPU-optimised framework described in Chapter 4 to implement PCDFs and demonstrate that we can significantly reduce the computation time needed for subsequent distance field compositions when compared to a state-of-the-art GPU-optimised approach to calculating distance fields. To achieve our goal of enabling predictive human trajectory avoidance, we explore the trajectory prediction of humans and propose a novel method for human trajectory prediction that combines *intention recognition* with trajectory

optimisation-based motion planning. We integrate our trajectory prediction method within the re-planning framework and demonstrate our ultimate goal of enabling a mobile manipulator to perform whole-body tasks while avoiding the predicted trajectory of a human operator. To enable appropriate evaluation, we further develop and release the Oxford Indoor Human Motion (Oxford-IHM) dataset[1] – a new human trajectory prediction dataset in which the humans move between intended goals. The dataset includes approximately one hour of RGB-D data from a static external camera as well as data from a moving robot's head-mounted camera. We use Vicon motion-capture to record ground-truth trajectories.

The following manuscript has been submitted to The International Journal of Robotics Research and is under review. A supplementary video[2] accompanies this work and the code has been made publicly available[3].

[1] `https://ori-drs.github.io/oxford-indoor-human-motion-dataset`
[2] `https://youtu.be/gdC3mpZNjG4`
[3] `https://github.com/ori-drs/integrated-dynamic-motion-planning-framework`

# Motion Planning in Dynamic Environments Using Context-Aware Human Trajectory Prediction

The International Journal of Robotics Research
XX(X):1–21
©The Author(s) 2022
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

**Mark Nicholas Finean[*1], Luka Petrović[*2], Wolfgang Merkt[1], Ivan Marković[2], Ioannis Havoutis[1]**

**Abstract**

Over the years, the separate fields of motion planning, mapping, and human trajectory prediction have advanced considerably. However, the literature is still sparse in providing practical frameworks that enable mobile manipulators to perform whole-body movements and account for the predicted motion of moving obstacles. Previous optimisation-based motion planning approaches that use distance fields have suffered from the high computational cost required to update the environment representation. We demonstrate that GPU-accelerated *predicted composite distance fields* significantly reduce the computation time compared to calculating distance fields from scratch. We integrate this technique with a complete motion planning and perception framework that accounts for the predicted motion of humans in dynamic environments, enabling reactive and pre-emptive motion planning that incorporates predicted motions. To achieve this, we propose and implement a novel human trajectory prediction method that combines intention recognition with trajectory optimisation-based motion planning. We validate our resultant framework on a real-world Toyota Human Support Robot (HSR) using live RGB-D sensor data from the onboard camera. In addition to providing analysis on a publicly available dataset, we release the Oxford Indoor Human Motion (Oxford-IHM) dataset and demonstrate state-of-the-art performance in human trajectory prediction. The Oxford-IHM dataset is a human trajectory prediction dataset in which people walk between regions of interest in an indoor environment. Both static and robot-mounted RGB-D cameras observe the people while tracked with a motion-capture system.

**Keywords**

Motion Planning, Trajectory Optimisation, Trajectory Prediction, Dynamic Environments, RGB-D Perception

## 1 Introduction

In this work, we focus on the deployment of mobile manipulators in dynamic indoor workspaces, such as a household environment. When robots operate in real-world environments, particularly where humans may co-occupy the workspace, safety is paramount. There is an extensive literature base in the space of 'autonomous road vehicles' for understanding and predicting 'pedestrian' trajectories to assist motion planning and collision avoidance (Rehder et al. 2018; Batkovic et al. 2018; Flores et al. 2019; Luo et al. 2018). In contrast, less work has focused on accounting for the predicted trajectories of humans when planning whole-body robot motions in indoor environments, motivating the work presented here.

Compared to the static case, dynamic environments pose many additional challenges that need to be addressed for robots to operate safely and efficiently. To perform tasks in the presence of moving obstacles, motion planning calculations must be performed online and *quickly* for a robot to react to environmental changes that would otherwise result in collisions. For reactive behaviour to take place, the robot's perception pipeline must be continuously updated online so that changes can be perceived sufficiently fast for the motion planning pipeline to react in time. In our previous work (Finean et al. 2021), we presented an integrated framework to enable such reactive behaviour by using a receding-horizon implementation of GPMP2 (Mukadam et al. 2018) in conjunction with the fast GPU-based perception pipeline within GPU-Voxels (Hermann et al. 2014; Juelg et al. 2018). While this work enabled reactive whole-body motion planning in response to a dynamic environment, it lacked understanding of dynamic elements in the environment and any reasoning about how they may move in the future. We further introduced the concept of *predicted composite distance fields* (Finean et al. 2020) as a fast method of incorporating the predicted motion of moving obstacles directly into a *distance field* (signed or unsigned) representation of the environment for time-configuration space planning. Using this method, separate environment representations are maintained for each timestep in the planned robot trajectory and moving obstacles are propagated using a motion prior, most commonly a constant-velocity model (CVM). We hypothesised that further improvements could be achieved by firstly leveraging parallelism in the problem and utilising GPUs to perform the 'compositing', and secondly

[1]Oxford Robotics Institute, University of Oxford, UK
[2]Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia
[*]These authors contributed equally.

**Corresponding author:**
Mark Nicholas Finean, Oxford Robotics Institute, University of Oxford, UK.
Email: mfinean@robots.ox.ac.uk

by incorporating additional scene insights for more realistic obstacle trajectory predictions.

In this paper, we propose an integrated framework for predictive whole-body motion planning in dynamic environments. For motion planning, we propose the Receding Horizon And Predictive Gaussian Process Motion Planner 2 (RHAP-GPMP2) – a receding-horizon motion planner that uses composite distance fields to account for the predicted motion of humans in the workspace. We use a state-of-the-art image segmentation method to identify humans and remove dynamic objects from the maintained voxelmap of the static scene. We investigate the task of human motion prediction and propose a planning-based human trajectory prediction method that combines human intention recognition with trajectory optimisation. To explore the problem and aid our analysis of the methods, we further produce and release a dataset for human trajectory prediction that includes robot-perspective RGB-D sensor data. We validate our complete framework in hardware experiments and demonstrate effective collision avoidance across multiple scenarios using a trajectory optimisation-based approach to whole-body motion planning in the presence of moving obstacles; one such scenario is shown in Fig. 1.

The key contributions of this paper are:

- A receding-horizon motion planner that uses composite distance fields to perform time-configuration space motion planning – Receding Horizon And Predictive Gaussian Process Motion Planner 2 (RHAP-GPMP2).
- A novel goal-oriented planning-based human trajectory prediction method that combines human intention recognition with trajectory optimisation.
- A comparison of the performance boost provided by GPU-calculated *predicted composite distance fields* with a state-of-the-art algorithm (PBA).
- Experimental verification of our integrated framework on an 8-DoF mobile manipulator in 3D dynamic environments using live sensor data.
- Release of the Oxford Indoor Human Motion (Oxford-IHM) dataset which comprises human-motion trajectories in an indoor environment, including motion-capture ground truth trajectories, static RGB-D camera images, and RGB-D data captured from the perspective of a moving robot.
- An open-source release of our framework which combines human motion prediction with GPU-optimised predicted composite distance fields for trajectory optimisation-based motion planning*.

## 2 Related Work

In this research, we focus on the development of an integrated framework at the intersection of environment mapping, whole-body motion planning in dynamic environments, and human motion prediction. In the following sections, we review the relevant work across these areas.

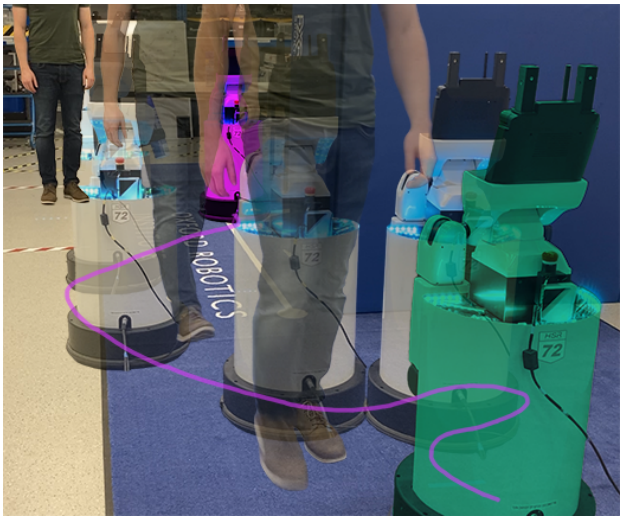


**Figure 1.** A Toyota Human Support Robot (HSR) is given a whole-body goal to place an object on a table at the other end of the room. A wall obstructs the robot's path and, during execution, a person walks towards a goal located behind the robot. Using our proposed approach and accounting for the predicted trajectory of the person, the robot re-plans to pre-emptively move out of the person's path before continuing towards the goal.

### 2.1 Perception and Motion Planning in Dynamic Environments

Although much of the mapping literature has focused on static environments (Newcombe et al. 2011; Whelan et al. 2012, 2015), there have been works that consider dynamic environments. Static-Fusion (Scona et al. 2018) uses geometric clustering to segment and filter out dynamic obstacles from RGB-D images and fuse observations into a dense static reconstruction of the environment. PoseFusion (Zhang and Nakamura 2020) combines OpenPose (Cao et al. 2021) with ElasticFusion (Whelan et al. 2015) to reconstruct the static scene while removing humans from the reconstruction. While OpenPose can be used to estimate the positions of body joints of humans within an RGB image, other dynamic scene reconstruction methods consider instance segmentation or optical flow techniques to separate dynamic parts of the scene from the static background. For example, Xie et al. (2021) perform feature-based RGB-D Simultaneous Localisation and Mapping (SLAM) in dynamic environments using Mask R-CNN (He et al. 2020) image segmentation and optical flow-based motion detection. While their approach demonstrates state-of-the-art localisation accuracy, they report an average processing time per frame of $0.42\,\mathrm{s}$ and "up to $1.10\,\mathrm{s}$ when mask inpainting" is required. As these numbers show, image segmentation is generally an expensive process and such long computation times may result in behaviours that lack reactivity when deployed on robots in real dynamic environments. Zhang et al. (2019) similarly use Mask R-CNN as a method of detecting potentially dynamic obstacles with a SLAM framework.

*Code available at: https://github.com/ori-drs/integrated-dynamic-motion-planning-framework

To achieve real-time run-rates for SLAM in dynamic environments, MaskFusion (Runz et al. 2019) supplements semantic instance segmentation (Mask R-CNN) with geometric segmentation. ReFusion uses a Truncated Signed Distance Field (TSDF) based mapping approach to build static maps of the environment and filter out dynamic objects by using the residuals "from the registration and the representation of free space" (Palazzolo et al. 2019). As with most of the SLAM literature, the aforementioned mapping systems consider the SLAM problem in isolation and do not consider integration with a motion planner. The reverse is also typically true, whereby motion planners in dynamic environments neglect the need for mapping to take place concurrently with live sensing.

Park et al. (2012) propose a parallel optimisation approach to motion re-planning in dynamic environments with ITOMP. To perform collision avoidance, they utilise pre-computed Euclidean Distance Transforms (EDTs) for static obstacle costs and use geometric collision detection to assign dynamic obstacle costs. However, their method was only tested in simulation and neglects consideration of the need to reconstruct the static environment using live sensor data in the presence of dynamic obstacles.

Voxblox (Oleynikova et al. 2017) and FIESTA (Han et al. 2019) propose incremental mapping frameworks and demonstrate them online. While FIESTA uses a kinodynamic path search method (Zhou et al. 2019), Voxblox is integrated with a trajectory optimisation-based motion planner similar to CHOMP (Oleynikova et al. 2016). In both cases, only 3D path planning for aerial vehicles is performed rather than whole-body motion planning as we propose.

GPU-Voxels (Hermann et al. 2014) is a GPU-optimised framework for multiple environment data structures that can be used for collision avoidance. Hermann et al. (2014) combine their perception pipeline with a D*-Lite motion planner to demonstrate a mobile robot re-planning in response to newly observed objects. However, they do not demonstrate reactive whole-body behaviour in dynamic environments; this is likely due to the 'curse of dimensionality' posed by search-based motion planners.

Juelg et al. (2018) built upon the GPU-Voxels framework to explore fast, exact 3D EDT implementations, such as the Parallel Banding Algorithm (PBA) (Cao et al. 2010). They use this work to perform fast motion planning for aerial robots with potential field and wavefront planners, integrated with a GPU-based perception framework that leverages the parallelism in EDT computations.

Of particular interest for our research is the additional aspect of accounting for *predicted trajectories*. Mainprice and Berenson (2013) used learnt human motions to predict the workspace occupancy for usage with STOMP (Kalakrishnan et al. 2011) in simulation experiments. Park et al. (2019) proposed I-Planner which similarly uses offline learning of human actions to generate predicted human motions for use in motion planning within the workspace of a 7-DoF robot arm.

To the best of our knowledge, there does not yet exist a fully integrated perception, motion planning, and prediction pipeline that can enable mobile manipulators to predict the trajectories of moving obstacles and subsequently avoid them in whole-body motion planning tasks. We address this in the work presented here.

### 2.2 Human Motion Prediction

Motion prediction plays an important role in ensuring the safety of robots and autonomous systems; anticipating how objects will move in a scene enables robots to act in a pro-active manner and pre-emptively respond to changes in a dynamic environment to avoid collisions. For inanimate dynamic objects, such as a rolling ball, we can commonly rely on a purely physics-based model, where simple kinematic models (e.g. constant velocity, constant acceleration) often suffice for enabling collision-free robot operation (Schöller et al. 2020). However, when robots operate in environments alongside humans, safety is of paramount importance and there is a need for more advanced motion prediction to capture the complexity of human behaviour. This complexity stems from both internal (goal intent, semantics) and external stimuli (environmental priors, actions of surrounding agents) that influence human motion. The multitude of human motion prediction methods can be categorised by their modelling approach as physics-based, pattern-based and planning-based methods (Rudenko et al. 2020b).

Physics-based methods predict human motion by propagating the current state through an explicit dynamical model (Helbing and Molnar 1995; Treuille et al. 2006; Pellegrini et al. 2009; Zanlungo et al. 2011; Karamouzas et al. 2009; Hermann et al. 2015). These methods are typically efficient, interpretable, and work very well for short-term prediction. However, in most cases they do not capture the complexity of the real world, ignoring environmental cues and the possible goals of a person. Notably, Pellegrini et al. (2009) propose a physics-based method that considers a future destination and the surrounding environment. However, their approach relies on a bird's-eye view and is thus not deployed on a real robot using live sensor data. Hermann et al. (2015) integrate human trajectory prediction with motion planning and perception by using swept-volume based extrapolation on live RGB-D data. However, they use the obtained motion prediction only for stopping a robot's movement when potential collisions are detected, rather than performing motion re-planning and adapting the robot's trajectory.

Pattern-based methods attempt to capture human motion behaviour by training function approximators, e.g. neural networks or Gaussian processes, on pre-recorded data (Alahi et al. 2016; Ghosh et al. 2017; Bartoli et al. 2018; Amirian et al. 2019; Yu et al. 2020; Cao et al. 2020; Kratzer et al. 2020). These models have become dominant in recent years due to their performance for long-term prediction in complex, semantically-rich environments. However, they require offline learning with large amounts of training data and offer limited transferability to novel environments due to poor generalisation capabilities. Cao et al. (2020) generate a diverse synthetic dataset to bypass the tedious data collection and propose a learning framework that exploits scene context to improve generalisation; however, their method is computationally demanding and is not deployed on a real robot. Kratzer et al. (2020) utilise a recurrent neural network for encoding short-term dynamics and account for

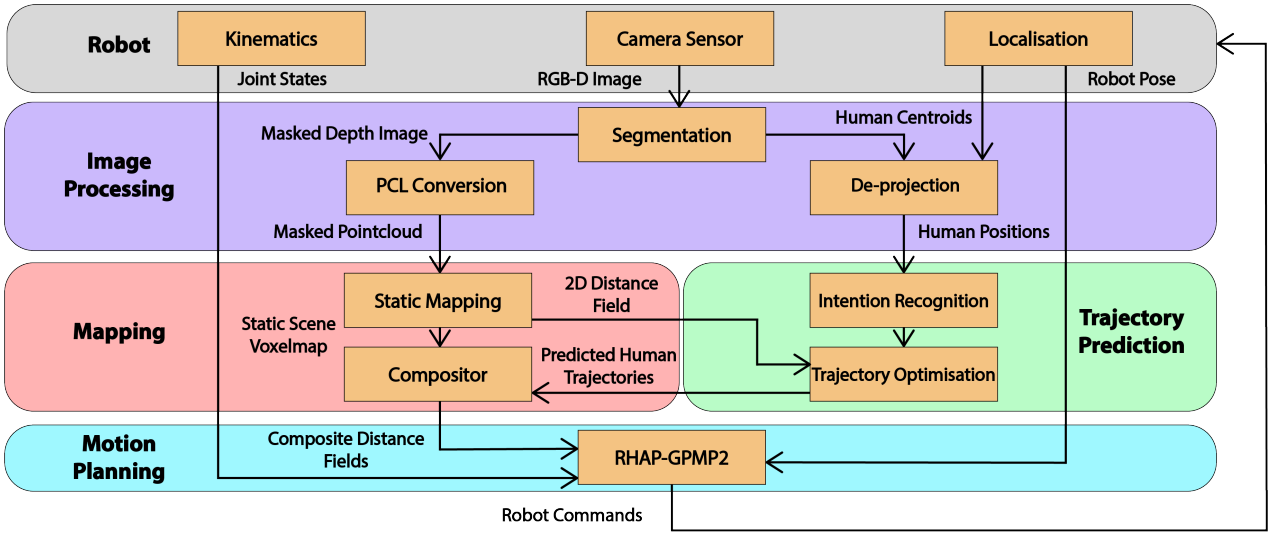



**Figure 2.** Flowchart of our Integrated Framework.

environmental constraints with trajectory optimisation; this method disregards the existence of multiple possible goals and is not demonstrated in real environments.

Planning-based approaches assume that a person is moving through an environment towards an existing goal while avoiding obstacles (Ziebart et al. 2009; Vasquez 2016; Rösmann et al. 2017; Kitani et al. 2012; Best and Fitch 2015; Rudenko et al. 2018; Zhi et al. 2021). These approaches offer a good balance between long-term prediction performance and capacity for generalisation, but in most cases they require an explicitly defined static map of the environment with the possible goal locations provided, making them difficult to apply on a real robot in an unknown environment. Best and Fitch (2015) present a Bayesian framework for intention estimation and use probabilistic roadmaps to obtain trajectory predictions. However, as a consequence of using a sampling-based planning method that does not account for smoothness, the predicted trajectories exhibit rapid changes of direction that are uncharacteristic of humans. Ziebart et al. (2009) predict goal-directed behaviours of pedestrians by solving a soft-maximum Markov Decision Process (MDP) with maximum entropy inverse reinforcement learning (Ziebart et al. 2008). They demonstrate real-time robot operation that accounts for human motion prediction but their method has limited generalisation capabilities since it relies on learning human reward functions from observed data.

To ensure safe robot operation in indoor environments, we utilise context-specific information and devise a novel trajectory optimisation-based method for human motion prediction that respects the underlying dynamical model and environmental cues. Our proposed method, as detailed in Secs. 7.1 and 7.2, offers a hybrid approach that can be deployed in unknown environments without any prior knowledge but can also incorporate information acquired both offline and online by learning possible human goals from observed data.

## 3 Proposed Framework Overview

In this section, we outline the modular nature of our proposed framework, as illustrated in Fig. 2, before examining each module in greater detail.

The first part of our proposed framework processes RGB - D images from the camera sensor to extract the information required by other modules. Using a neural network-based instance segmentation method (described in Sec. 5) on the RGB images, we generate masks of objects that are detected in the scene. While our method can be trivially adapted to other identified obstacles, in this research, we focus solely on utilising the masks of detected humans. If masks are produced above a user-specified threshold score, $\alpha_{mask}$, we apply the masks to their corresponding depth images; we do this to first extract the estimated position of the humans in the scene, and secondly as a method of removing dynamic obstacles from the scene prior to converting depth images to point clouds. The filtered pointclouds are used to provide live updates to the maintained voxelmap of the static scene, while the extracted positions of people in the scene are passed to the trajectory prediction module.

Using the estimated human positions provided by the image processing module, we perform human trajectory prediction. As described in Sec. 2.2, being able to account for the predicted motion of humans is important for safe robot task execution. While the future trajectory of an inanimate object can often be predicted using constant-velocity or constant-acceleration models, human behaviour is more complex and requires a different modelling approach. We propose a hybrid trajectory prediction module (detailed in Sec. 7) that uses a lightweight planning-based approach to perform prediction while retaining the ability to incorporate 'learnt' or *prior* information. The proposed trajectory prediction method can rely solely on information obtained from live sensor data, requiring no prior training or initialisation. As shown in Fig. 2, our trajectory prediction module can be divided into *Intention Recognition* (Sec. 7.1) and *Human Trajectory Optimisation* (Sec. 7.2).



(a) (b) (c)

**Figure 3.** Photos of the marker arrangements used for motion capture tracking of the robot, environment, and humans. Left: A custom 3D-printed frame for mounting markers on the head of an HSR robot. Middle: Multiple objects were tracked and placed in the environment to provide human 'goal' markers. Additional markers were placed at the two entrance/exit locations of the arena. Right: An example setup of the static RGB-D camera within the scene. A custom camera housing was made to attach and calibrate the Vicon markers.

After predicting trajectories for humans in the scene, we forward these predictions to the mapping module so that the predicted positions of people in the scene can be composited into distance fields that are maintained for each timestep in our proposed motion planning algorithm, Receding Horizon And Predictive Gaussian Process Motion Planner 2 (RHAP-GPMP2). With the composite distance fields being continuously updated from the latest observations and trajectory prediction information, our motion planning algorithm is able to re-plan and enable reactive, pre-emptive robot behaviours to avoid moving people in the workspace.

## 4 Human Motion Dataset



**Figure 4.** An example map configuration used in the Oxford-IHM dataset. As the Vicon-tracked person walks between goals in the scene, the HSR robot is manually controlled by a human operator to move around the scene and maintain vision of the person.

For this research, we are concerned with robots that operate in indoor environments co-occupied by humans. In order to evaluate any proposed method for human motion prediction, we require an appropriate dataset that, in line with our ambitions for robot autonomy, contains sensor data from the robot's perspective.

While there are a number of publicly available RGB-D datasets, the majority are aimed at applications of SLAM (Sturm et al. 2012; Handa et al. 2014), object detection (Janoch et al. 2011; Lai et al. 2011), or human activity recognition (Sung et al. 2012; Zhang and Parker 2011). Sturm et al. (2012) present an RGB-D dataset where sensor data is collected from the robot's perspective for the purpose of evaluating SLAM systems. However, the dataset lacks the presence of moving humans and their ground truth trajectories for which we can try to predict.

Munaro and Menegatti (2014) released the most relevant dataset for our purposes, the Kinect Tracking Precision (KTP) Dataset. The dataset comprises RGB-D images recorded from a robot's perspective in a scene where humans are moving around while the robot performs locomotion. Totalling only four minutes of video recording, we do not believe that the KTP dataset captures an accurate representation of human motion over a wide enough range of behaviours for prediction purposes. In their recordings, humans move either in a linear or random manner. While this may be how human motion appears sometimes, we believe that humans usually have *intent*, i.e. a destination and task in mind. For example, people will often travel to their desks, a bookcase, or exit through a door in an office environment. Before people enter a room, we could intuitively generate a prior map of where they will go and refine our belief as their trajectory progresses.

The concept of *intent* motivates a dataset in which humans are performing tasks relevant to the environment. The THÖR dataset (Rudenko et al. 2020a) provides human motion trajectories and broad goal locations within an indoor experiment; however, it uses 3D LiDAR scans from a stationary sensor rather than RGB-D data from a robot perspective. To address the need for a robot perspective RGB-D dataset with task-based human motion trajectories, we propose and release the Oxford Indoor Human Motion (Oxford-IHM) Dataset[†]. We summarise a selection of the most relevant publicly available datasets that include human trajectories, alongside our own, in Table 1.

[†]Dataset available at `https://ori-drs.github.io/oxford-indoor-human-motion-dataset`

| Dataset | Environment | Duration | External Sensors | Robot Perspective | Motion Capture | Goals | Map |
|---|---|---|---|---|---|---|---|
| KTP (Munaro and Menegatti 2014) | Empty Room | 4m 40s | ✘ | RGB-D | ✓ | ✘ | ✘ |
| ATC (Brscic et al. 2013) | Shopping Centre | 41 days | RGB-D | ✘ | ✘ | ✓ | ✘ |
| THÖR (Rudenko et al. 2020a) | Laboratory | 60 mins | 3D LiDAR, RGB, Eye Tracking | ✘ | ✓ | ✓ | ✓ |
| L-CAS (Yan et al. 2017) | Office | 49 mins | ✘ | 3D LiDAR | ✘ | ✘ | ✘ |
| MoGaze (Kratzer et al. 2021) | Laboratory | 180 mins | Eye Tracking | ✘ | ✓ | ✓ | ✓ |
| **Oxford-IHM** | Laboratory/Office | 60 mins | RGB-D | RGB-D | ✓ | ✓ | ✓ |

**Table 1.** A Comparison of Publicly Available Indoor Datasets with Ground Truth Human Trajectories

### 4.1 Data Acquisition

Our dataset was recorded in a large indoor laboratory within which we constructed an arena similar to an office environment. Under a Vicon motion capture setup, we created the arena of interest, measuring $7.1\,\mathrm{m} \times 4.2\,\mathrm{m}$, with perimeter walls and two entrance/exit locations. Within the arena, multiple large objects, such as a desk, were arranged in multiple configurations to act both as potential goals and static obstacles. As a tracked human walks between goals in the arena, we recorded RGB-D images from both a static Intel Realsense D435 camera (Fig. 3c) and a Toyota Human Support Robot's head-mounted ASUS Xtion Pro Live camera. Additionally, we recorded the robot's `tf` data which details the robot's 3D pose and joint transformations over time.

While the robot supports an Ethernet connection for data transfer, to avoid trailing cables and maintain recording bandwidth, we opted to control the robot wirelessly and record robot data locally. We used *chrony* time-synchronisation between the robot's onboard clock and an external laptop (Intel Core i7-10875H CPU, $32\,\mathrm{GB}$ $2666\,\mathrm{MHz}$ RAM, and an NVIDIA GeForce RTX 2070 SUPER GPU). The external laptop was used to record Vicon marker data and RGB-D image data from the static Realsense camera.

During recording, the robot was remotely controlled and navigated around the arena, generally in such a manner that it maintained vision of the tracked person. The overhead motion capture setup was used to record the ground truth locations of goals, entrance/exit locations, the robot, and the person in the scene. The motion capture arrangement consisted of 18 Vicon Vero 2.2 cameras ($2.2\,\mathrm{MP}$ with $850\,\mathrm{nm}$ IR emitters). We calibrated the cameras using a Vicon Active Wand v2 to achieve a sub-millimetre average residual tracking error. For accurate tracking, unique IR marker configurations were affixed to each trackable object. For person tracking, reflective markers were attached to helmets and calibrated to align each helmet's orientation with a person's gaze direction. In the cases of the robot and the external camera, custom mounts were 3D printed (Fig. 3) and used to calibrate each object's tracked pose with its respective internal camera frame.

Our dataset consists of $\approx 60$ minutes of rosbag data split approximately equally across four different map configurations, each with three runs. For each map configuration, we used the ROS *hector_mapping* package (Kohlbrecher et al. 2011) and the robot's onboard Hokuyo UST-20LX laser range sensor to produce a 2D map of the arena. To provide additional variation, our dataset uses two people with the map configurations split equally between them.

## 5 Image Processing

As discussed in Sec. 2.2, numerous approaches have been explored in previous efforts to perform dense environment mapping in dynamic environments (Scona et al. 2018; Zhang and Nakamura 2020; Cao et al. 2021; Palazzolo et al. 2019), with many employing image segmentation techniques (Xie et al. 2021; Runz et al. 2019; Zhang et al. 2019).

### 5.1 Image Segmentation

A commonly used state-of-the-art method of image segmentation is Mask R-CNN (He et al. 2020). Mask R-CNN is an extension of Faster R-CNN (Ren et al. 2017) that predicts the mask of an object in parallel with bounding box recognition. Image segmentation is typically an expensive operation to perform – He et al. (2020) reported a frame rate of $5\,\mathrm{Hz}$ on an NVIDIA Tesla M40 GPU. For our purposes of using online environment reconstructions for motion planning in dynamic environments, we require minimal latency between making sensor observations and them being reflected in a motion planner's collision-checking ability. As such, in this work we build on recent advances in image segmentation performance.

While numerous works have sought to improve Mask R-CNN, few works focus on improving the speed of the instance segmentation (Lee and Park 2020). Lee and Park (2020) introduced CenterMask and CenterMask-Lite, anchor-free one-stage instance segmentation methods that outperform the current state-of-the-art – the authors report that CenterMask-Lite with a VoVNetV2-39 backbone achieves a frame rate of $35\,\mathrm{Hz}$ on an NVIDIA Titan Xp GPU. We performed a local benchmarking of the latest release, CenterMask2-Lite, against Mask R-CNN on an RGB-D camera stream and found the lightweight CenterMask2-Lite to run $3.2\times$ faster than Mask R-CNN – $13.4\,\mathrm{Hz}$ compared with $4.2\,\mathrm{Hz}$. This test was performed using: NVIDIA RTX 2060 GPU, 8-core Intel Core i7-9700 CPU @ $4.50\,\mathrm{GHz}$ and $2133\,\mathrm{MHz}$ DDR4 RAM.

Due to the importance of fast perception and motion planning when operating in a dynamic environment, we

| | MOTP (m) | | MOTA (%) | | FP (%) | | FN (%) | |
|---|---|---|---|---|---|---|---|---|
| Task | Ours | KTP | Ours | KTP | Ours | KTP | Ours | KTP |
| Back and Forth | **0.176** | 0.196 | **91.7** | 88.97 | 5.9 | **2.4** | **1.1** | 8.5 |
| Random Walk | 0.171 | 0.171 | **76.0** | 70.93 | **2.3** | 9.8 | **8.6** | 18.9 |
| Side-by-Side | 0.151 | **0.146** | 85.9 | **87.22** | **0.7** | 1.2 | **6.7** | 11.6 |
| Running | **0.136** | 0.143 | 91.6 | **94.57** | **0.0** | 1.1 | **4.2** | 4.4 |
| Group | 0.198 | **0.181** | **59.4** | 47.91 | **1.7** | 9.1 | **15.8** | 42.53 |

**Table 2.** Image Processing and Human Position Estimation Benchmarking. For comparison, the KTP performance metrics are replicated from (Munaro and Menegatti 2014).

elected to exploit the enhanced performance provided by CenterMask2-Lite for image segmentation in our perception pipeline.

### 5.2 Human Position Estimation

As described previously, and illustrated in Fig. 2, we perform image segmentation on a stream of RGB images and use the results for both maintaining the static representation of the environment and for estimating the position of humans in the scene. For each RGB frame that contains masks labelled as a *person* with a score above the specified threshold, $\alpha_{mask}$, we apply the masks to the corresponding time-synchronised depth image. In this work, we found $\alpha_{mask} = 0.7$ to perform well in consistently masking people even when partially obstructed by obstacles. Using the masked depth image, we extract a depth to associate with the person. In this work, we use the median depth and pixel position of a person's mask to calculate the person's 3D position in the workspace – this position is passed onto the tracking and prediction module.

### 5.3 Object Masking and Pointcloud Conversion

To filter out the dynamic element of the scene, we apply valid *person* masks to their corresponding depth images. The filtered depth images are converted to pointclouds for integration into the maintained voxelmap of the static environment. We found that it is beneficial to apply a `dilation` to the *person* masks before converting to pointclouds. By enlarging the segmentation masks, we reduce leakage from the dynamic masks into the static voxelmap. We use OpenCV to perform four iterations of dilation with a $5 \times 5$ kernel.

### 5.4 Evaluation

To validate our image processing pipeline and demonstrate its effectiveness, we evaluate our method on the KTP Dataset (Munaro and Menegatti 2014), using the same metrics as used in their evaluation (Bernardin and Stiefelhagen 2008). Due to the ground truth positions in this dataset corresponding to a person's tracked head location, Munaro and Menegatti (2014) use the "centroid of the cluster points belonging to the head of the person" and add a $10\,\mathrm{cm}$ offset in the viewpoint direction. To provide a similar comparison, rather than using the entire *person* mask for position extraction, we use the top $10\,\%$ of the mask to correspond with the head. We similarly add an offset in the viewpoint direction and found $12.5\,\mathrm{cm}$ to give the best results.

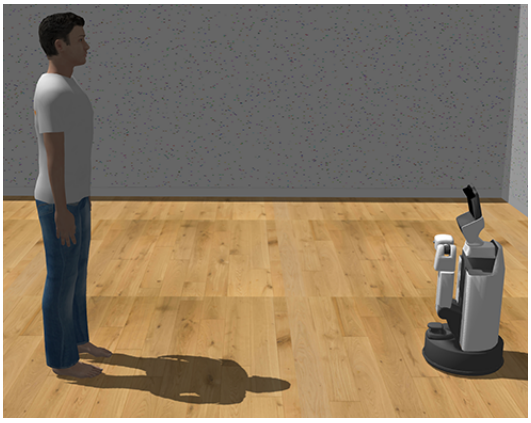
The Multiple Object Tracking Precision (MOTP) metric indicates the ability of a 'tracker' to estimate object positions accurately. The Multiple Object Tracking Accuracy (MOTA) metric indicates the reliability of a tracker to identify objects in an image frame. While we explore instance segmentation in this work, we do not perform 'tracking' between frames – as such, we assume no incorrect object associations in the calculation of the MOTA metric. Additional recorded metrics of interest are the False Positive (FP) and False Negative (FN) rates. Our benchmarking results are presented in Table 2.

Importantly for this research, we produce similar MOTP values, indicating that our method achieves a similarly competitive accuracy in estimating the position of people in a scene while additionally providing masks for use in the environment reconstruction. In the following section, we discuss how the extracted positions of people feed into our trajectory prediction pipeline.

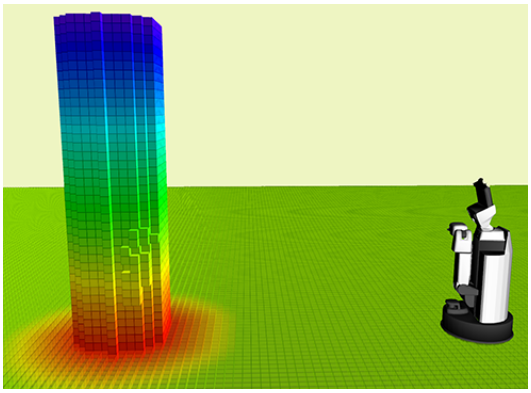
## 6 Mapping - Predicted Composite Distance Fields

In our preliminary work on *predicted composite distance fields* (Finean et al. 2020), we suggested that further computational gains could be achieved by performing *compositions* within a GPU-leveraged framework since the core operation of the method is the *min* operation and thus highly parallelisable. In this work, we explore the gains achievable with such an implementation.

Two components are required to generate composite distance fields. Firstly, one needs to maintain a distance field for the environment. Depending on the problem, this may be a static distance field that is computed once at the start of the experiment or continuously updated and maintained as in our framework. Secondly, we require a distance field associated with each (moving) object that is to be *composited* onto the environment distance field. These distance fields can similarly be continuously updated to represent a live model of the obstacles being tracked. In this work, we are interested in human collision avoidance and so the fine voxelised detail of a human is not necessary; instead, we represent humans with similarly sized cylinder shape primitives. The use of primitive shapes is beneficial since we do not need to be concerned with monitoring the shape of the humans and maintaining a live model; rather, we only need to compute the distance field of the primitive shape once and subsequently track the human positions. However, we note that using shape primitives is a choice in this work rather than a limitation – the distance field could equally be continuously updated to represent an accurate model of a dynamic obstacle as it is observed. There is a large literature base on the dense reconstruction of deformable objects Dou et al. (2015); Yu et al. (2015).



(a)



(b)

**Figure 5.** Top: An example scene in simulation (Gazebo) in which a person is detected by the robot's onboard RGB-D camera. Bottom: The resultant 3D occupancy grid after thresholding the composite distance field for the scene; the distance field of a cylinder is has been composited onto the detected position of the person.

Given the predicted positions for all dynamic obstacles in the scene for a given time, we can perform a composition of the aforementioned distance fields. This is achieved with a parallel `min` operation between the environment distance field and those of the humans at their predicted positions. An example composition is shown in Fig. 5.

In the case of only considering a single distance field of the environment for each observation update, i.e. no prediction, our method will not be of benefit since only one distance field computation is required. The benefit of our composition approach is apparent when multiple *subsequent distance fields* are required for each environment update loop, such as our motion planning approach as detailed in Sec. 8 which considers multiple predicted distance fields of the environment for each update loop. As such, we perform benchmarking with respect to the calculation time for *subsequent distance fields* against PBA. Hardware specifications used were: NVIDIA RTX 2060 GPU, 8-core Intel Core i7-9700 CPU @ 4.50 GHz and 2133 MHz DDR4 RAM.

Benchmarking results are shown in Fig. 6 where we provide comparisons both including and excluding the time to transfer distance fields from the GPU to the host device. Excluding the transfer times from GPU to host, Fig. 6b shows our composite method to reduce computation time by $89\,\%$ to $93\,\%$. Unfortunately, at the short timescales that we achieve, the transfer time becomes a dominant factor, accounting for over $90\,\%$ of the overall update time for the composite distance field. However, our composite method still provides a significant performance boost, even after accounting for the transfer time, when compared to a full PBA calculation, cutting the computation time for the resultant distance field by $40\,\%$ to $53\,\%$.

## 7 Human Trajectory Prediction

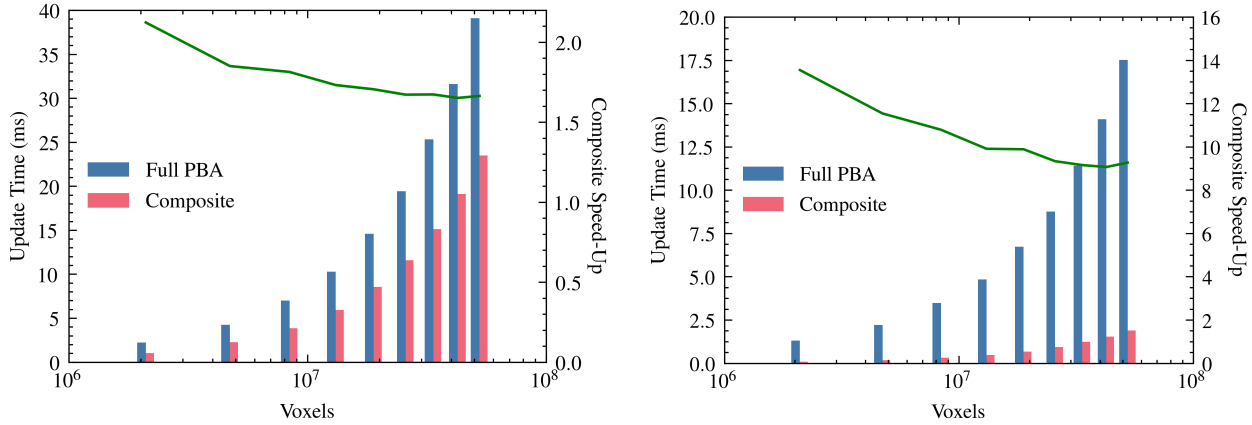
In indoor environments, a person typically moves towards an intended goal, such as a door to exit through or towards an object to pick up, rather than in a random manner. Therefore, the first component of our trajectory prediction module is *intention recognition* in which we try to determine a person's intended goal.

### 7.1 Intention Recognition

Suppose there exists a set, $\mathcal{G}$, of $K$ possible goals for a person in the environment, $\boldsymbol{g}_k \in \mathcal{G}$, where a goal is represented by a 2D position vector $\boldsymbol{g}_k = [x_{g_k}, y_{g_k}]^T$. The purpose of the intention recognition module is first to recognise the possible goal locations in the environment that might be of interest to the person, i.e. $\mathcal{G}$, and secondly to identify which of these goals is a person's current intended goal, $\boldsymbol{g}$.

In practice, we believe that $\mathcal{G}$ can be learnt over time as objects and areas of interest are observed and identified. Similarly, we believe that we can infer possible goal locations by observing human motion data over time. To explore this idea, we consider a simple *occupancy analysis* method. Given the recorded positions of a person over time, we discretise the positions across a 2D grid of arbitrary size and monitor the number of visitations for each cell, $n_i$, where the corresponding velocity is less than some threshold, $v_{\text{thres}}$. In the rest of this paper, we use $v_{\text{thres}} = 0.3\,\text{ms}^{-1}$; this value is much less than the average human walking velocity meaning that grid states which are frequently passed through will not be mislabeled as possible goals. Using a threshold value closer to zero would lead to poor identification of goals where a person is not completely static but has slowed down, e.g. doors. Using the aforementioned frequency grid, where the most visited cell has $N_{\text{max}}$ visitations, we identify possible goal locations as those cells with $n_i > \frac{N_{\text{max}}}{2}$. If there are multiple adjacent states identified as goals, we take the mean position of those states as a single goal location. By implicitly learning social context cues, our method can also learn additional goals that cannot easily be identified from semantics, e.g. particular gathering points without identifiable objects at those locations. Note that the described goal estimation method can be supplemented with semantic information from the perception pipeline to use identified objects such as desks, sofas, and doors as possible goals, even if they were not visited during the observation time.

We demonstrate our occupancy analysis method on both the Oxford-IHM and THÖR datasets; the results are shown in Fig. 7 and indicate that the most commonly occupied grid states provide accurate estimates of the ground truth goal locations in each dataset. For both datasets, we analysed segments $\sim 5$ minutes long. For the THÖR dataset, we




**(a)** Including Device-Host Transfer Time

**(b)** Excluding Device-Host Transfer Time

**Figure 6.** A comparison of the compute times for composite distance fields and full PBA calculations across a range of voxelmap sizes. The plotted line (green) corresponds to the ratio between the bars, i.e. the resultant speed-up of using composite distance fields. While we see an order of magnitude speed-up in the underlying distance field generation, we find that the device-host transfer time dominates the update time, reducing our overall speed-up from $\sim 9.1 - 13.6\times$ to $\sim 1.7 - 2.1\times$.

tracked all the subjects marked as visitors (Rudenko et al. 2020a). On the Oxford-IHM dataset, all ground truth goals were identified, although with an offset due to the ground truth goals being objects that humans maintain a distance from, e.g. a person sits in front of a desk rather than on it. On the THÖR dataset, we identified three out of the five labelled goals; our method did not identify two of the goals for several reasons. Firstly, these goals were in areas with frequent motion capture track drops, a problem that does not occur with live robot sensor data. Secondly, these goals represented exit and entrance locations but without doors; as a result, people passed through without slowing down. One could argue that in this instance, these do not represent accurate goal locations since the people will continue to walk to their true goal locations.

For a given determination of $\mathcal{G}$, we want to determine the probability of each goal, $\boldsymbol{g}_k$, being the human's intended goal, $\boldsymbol{g}$, given that we have observed a history of the person's trajectory, $\boldsymbol{X}_h(t)$, where $t$ is the current time. We use $\boldsymbol{X}_h(t)$ to denote the matrix formed by the past $N$ vector measurements of the human's position, $\boldsymbol{x}_h(t_i) = [x_h(t_i), y_h(t_i)]$ for times $t_i < t$.

By framing the human intention recognition problem in this probabilistic manner, we can use Bayes' rule to derive a posterior distribution for goal locations as

$$p\big(\boldsymbol{G}|\boldsymbol{X}_h\big) \propto p(\boldsymbol{G})p\big(\boldsymbol{X}_h|\boldsymbol{G}\big), \tag{1}$$

where $p(\boldsymbol{G})$ is a distribution that encodes prior knowledge of goal probabilities and $p\big(\boldsymbol{X}_h|\boldsymbol{G}\big)$ is a conditional distribution that represents the *likelihood* of the recorded past human trajectory for a set of a given goal.

If there is no prior knowledge about the probability distribution for goals, i.e. no goal visitation history, the prior $p(\boldsymbol{G})$ is set as the uniform distribution. On the other hand, if we have an observed (or pre-recorded) history of motion data and perform the occupancy grid analysis described previously, $p(\boldsymbol{G})$ can be set as a categorical distribution where the prior probability for each identified goal is proportional to its number of visitations, $N_k$:

$$p(\boldsymbol{G} = \boldsymbol{g_k}) = \frac{N_k}{\sum_{l=1}^{K} N_l}. \tag{2}$$

Note that in practical applications, the prior goal distribution $p(\boldsymbol{G})$ can be initialised as a uniform distribution when a robot first begins to operate in an environment. As information about the environment and human movement is collected during operation, the prior distribution can be altered online after performing the grid occupancy analysis.

Lastly, we calculate the likelihood, $p\big(\boldsymbol{X}_h(t)|\boldsymbol{G}\big)$, i.e. given the human's recent history, what is the likelihood of each possible goal being the intended one? Intuitively, a person is likely to look at the object they want to reach or move towards in the near future, i.e. the intended goal. As shown by Kratzer et al. (2020), for a set of objects that represents possible goal locations in the environment, a person's gaze is a great predictor of intention. While we could build on this idea directly and use the difference in angle between a person's gaze and each object to determine the probability of each goal, determining a person's gaze in practice is challenging. Wearable gaze tracking equipment is shown to work very well (Kratzer et al. 2020); however, it is impractical to assume that this is available in everyday applications such as in a household environment. On the other hand, alternative methods that try to estimate the human gaze from images (Kellnhofer et al. 2019; Park et al. 2018) require significant computational resources to work in real-time and have degraded performance when a person is turned away from the robot.

For the aforementioned reasons, we use the human's estimated orientation, obtained from the history of positions, as a predictor of *intent*. While the use of estimated orientation as a motion cue may not be as effective as using a person's gaze, it does not require observability of the human's face and can suffice when other motion cues are unavailable due to hardware constraints. We estimate the human body orientation $\hat{\theta}_h(t_i)$ from the difference between

two subsequent positions

$$\hat{\theta}_h(t_i) = \arctan\left(\frac{y_h(t_i) - y_h(t_{i-1})}{x_h(t_i) - x_h(t_{i-1})}\right), \tag{3}$$

where we use the '$h$' subscript to denote human positions. The relative orientation between a particular goal location and a person is then given by

$$\delta\theta_{g_k}(t_i) = \arctan\left(\frac{y_{g_k} - y_h(t_i)}{x_{g_k} - x_h(t_i)}\right) - \hat{\theta}_h(t_i). \tag{4}$$

In practice, there is likely to be noise in individual estimates of the orientation either due to sensor measurements or because a person may briefly look away from their goal without changing their intent. As a result, the relative orientation $\delta\theta_{g_k}(t_i)$ can quickly vary between subsequent timesteps without an actual change in the person's body motion. Therefore we calculate the average over the past $N$ relative orientations

$$\overline{\delta\theta_{g_k}}(t_i) = \frac{1}{N}\sum_{j=0}^{N-1} \delta\theta_{g_k}(t_{i-j}), \tag{5}$$

We achieve more stable estimates for the relative orientation by aggregating the recent trajectory history. When the average relative orientation, $\overline{\delta\theta_{g_k}}(t_i)$, is zero, it implies that a person is moving in the direction of the goal $\boldsymbol{g_k}$. Conversely, when $\overline{\delta\theta_{g_k}}(t_i)$ is equal to $\pi$, it implies that a person is moving directly away from the goal $\boldsymbol{g_k}$. We thus formulate the likelihood $p\big(\boldsymbol{X}_h|\boldsymbol{G} = \boldsymbol{g_k}\big)$ by calculating the softmax function of the average of past $N$ relative orientations $\overline{\delta\theta_{g_k}}(t_i)$

$$p\big(\boldsymbol{X}_h|\boldsymbol{G} = \boldsymbol{g_k}\big) = \frac{e^{\lambda\overline{\delta\theta_{g_k}}(t_i)}}{\sum_{l=1}^{K} e^{\lambda\overline{\delta\theta_{g_l}}(t_i)}}, \tag{6}$$

where $\lambda$ is a constant that determines the sensitivity of the exponentiated cost. We use $\lambda = 1$ throughout the rest of this work. Following Eq. 1, the intended goal position is simply extracted by calculating the maximum a posteriori probability (MAP) estimate

$$\hat{\boldsymbol{g}}_{\text{MAP}} = \underset{\boldsymbol{G}\in\mathcal{G}}{\arg\max}\; p(\boldsymbol{G})p\big(\boldsymbol{X}_h|\boldsymbol{G}\big). \tag{7}$$

The estimated goal, $\hat{\boldsymbol{g}}_{\text{MAP}}$, has the corresponding probability, $p\big(\boldsymbol{G} = \hat{\boldsymbol{g}}_{\text{MAP}}|\boldsymbol{X}_h\big)$, that represents how sure we are that the estimated goal is the intended one.

## 7.2 Trajectory Optimisation

Once we have determined a person's intended goal, we want to anticipate their motion towards it in order to safely steer the robot away from a person and avoid potential collisions. We make several assumptions about human behaviour in our trajectory prediction method.

Our first assumption is that a person, unobstructed by other factors, will move according to a constant-velocity kinematic model; a person walking directly towards a goal will tend to maintain the same velocity. While we could employ a higher-order kinematic model, such as constant-acceleration, it is unlikely that a person will quickly change their movement speed under normal circumstances, and so a constant-velocity motion model is sufficient for modelling human motion in open spaces (Schöller et al. 2020). The second assumption that we make is that a person, unlike a moving inanimate object, is generally aware of obstacles in the environment and will try to avoid colliding with them. As such, a robot can use information that it has accumulated about the map of the environment as an environmental prior when predicting a person's trajectory. Our third assumption is that a person is aware of robots operating in the environment and will tend to avoid the space that it occupies.

Using these assumptions, we formulate the human trajectory prediction problem as non-linear trajectory optimisation; although primarily used for robot motion planning, in our implementation, we use GPMP2 (Mukadam et al. 2018) as a state-of-the-art trajectory optimisation method. Since human trajectory prediction and robot motion planning share similarities, we believe that GPMP2, with minor adaptations, is suitable for our prediction problem.

In GPMP2, the motion planning problem is framed as a probabilistic inference problem whereby the aim is to formulate the posterior density of a trajectory and solve for the maximum a posteriori (MAP) estimator, just as we did in the previous section. Using Bayes' rule, the posterior distribution of a trajectory, $\boldsymbol{x}$, given the likelihood on a collection of events, $\boldsymbol{e}$, is given by

$$p(\boldsymbol{x}|\boldsymbol{e}) \propto p(\boldsymbol{x})p(\boldsymbol{e}|\boldsymbol{x}), \tag{8}$$

where $p(\boldsymbol{x})$ represents the prior that encourages trajectory smoothness, while $p(\boldsymbol{e}|\boldsymbol{x})$ represents the probability of the events $\boldsymbol{e}$ occurring given $\boldsymbol{x}$. In the case of motion planning, $\boldsymbol{e}$ corresponds to binary events that a trajectory $\boldsymbol{x}$ is collision-free at a particular state.

In GPMP2, robot trajectories are represented as samples from a continuous-time Gaussian Process (GP), $\boldsymbol{x}(t) \sim \mathcal{GP}(\boldsymbol{\mu}(t), \boldsymbol{\mathcal{K}}(t,t'))$, where $\boldsymbol{\mu}(t)$ is the vector-valued mean trajectory and $\boldsymbol{\mathcal{K}}(t,t')$ is the matrix-valued covariance. By carefully choosing a structured kernel, one can show that the resultant precision matrix is exactly-sparse (Mukadam et al. 2018). Consequently, Mukadam et al. (2018) show that the probabilistic inference problem in Eq. 8 can be efficiently solved on a factor graph.

We adopt a similar factor graph formulation and adapt it for human trajectory prediction. Using a structured kernel as in Mukadam et al. (2018), the prior and the likelihood functions can be written as a product of functions

$$\begin{aligned} p(\boldsymbol{x}_h)p(\boldsymbol{e}|\boldsymbol{x}_h) &\propto f^{prior}\left(\boldsymbol{X}_h\right) f^{like}\left(\boldsymbol{X}_h\right) \\ &= \prod_i f_i\left(\boldsymbol{X}_{h,i}\right) \end{aligned} \tag{9}$$

where $\boldsymbol{X}_h = \{\boldsymbol{x}_{h,0}, \ldots, \boldsymbol{x}_{h,N}\}$ represents the set of future human positions along the predicted trajectory. The *factors* $\mathcal{F} = \{f_0, \ldots, f_M\}$ are functions that act on variable subsets of the trajectory. As shown by Kschischang et al. (2001), the posterior distribution can be represented by a bipartite factor graph $G = \{\boldsymbol{X}, \mathcal{F}, \mathcal{E}\}$, where $\mathcal{E}$ is the set of *edges* that connect *variable* and *factor* nodes.

To encourage *smoothness* in our predicted human trajectories and account for the tendency to move according to a constant-velocity motion model, we adopt the GP prior

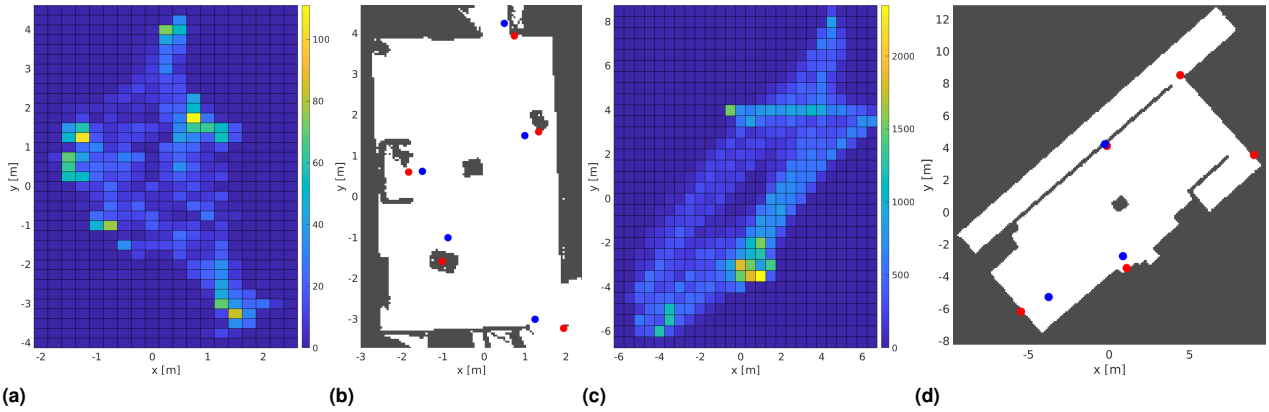



**Figure 7.** By performing our occupancy analysis method on recorded motion data, we can estimate a person's possible goals. We demonstrate this technique using the Oxford-IHM (Figs. 7a and 7b) and THÖR datasets (Figs. 7c and 7d). We monitor the frequency of visitations to each grid state after applying a velocity threshold, resulting in the heatmaps shown. The most occupied states provide a reliable estimate of possible goal locations. Figures 7b and 7d show maps of example environments from each dataset with the actual (blue) and estimated (red) goal locations.

proposed in Mukadam et al. (2018),

$$p(\boldsymbol{x}_h) \propto \exp\{-\frac{1}{2}\|\boldsymbol{x}_h - \boldsymbol{\mu}_h\|^2_{\boldsymbol{\mathcal{K}}_h}\}, \tag{10}$$

given in terms of the mean trajectory $\boldsymbol{\mu}_h$ and covariance $\boldsymbol{\mathcal{K}}_h$. We initialise the mean as a constant-velocity straight line, while the covariance is obtained by solving the Linear Time-Varying Stochastic Differential Equation (LTV-SDE) with constant-velocity model system matrices, as in Mukadam et al. (2018). Due to the structured kernel choice, this GP prior has a Markovian structure; as such, it can be written as a product of GP prior factors that depend only on two neighbouring states, $f^{gp}(\boldsymbol{x}_{h,i}, \boldsymbol{x}_{h,i+1})$.

In addition to GP priors that describe how our trajectory behaves, we want to impose knowledge of a person's start and intended goal states. In the context of the human trajectory prediction, the start state is the current position of a person in the environment, while the intended goal state is obtained by our intention recognition method described in Sec. 7.1. We encode start and goal states by using the following factors:

$$f^{start}(\boldsymbol{x}_{h,0}) = \exp\{-\frac{1}{2}\|\boldsymbol{x}_{h,0} - \boldsymbol{x}_{current}\|^2_{\boldsymbol{\Sigma}_{h,0}}\}, \tag{11}$$

$$f^{goal}(\boldsymbol{x}_{h,N}) = \exp\{-\frac{1}{2}\|\boldsymbol{x}_{h,N} - \boldsymbol{x}_{goal}\|^2_{\boldsymbol{\Sigma}_{h,N}}\}, \tag{12}$$

where $N$ represents the final support state of the trajectory. $\boldsymbol{\Sigma}_{h,0}$ and $\boldsymbol{\Sigma}_{h,N}$ are the isotropic covariance matrices for the start and goal states. Smaller values along the diagonals of these matrices result in higher costs for deviating from the specified start and goal states, encouraging the optimised human trajectory prediction to adhere to the start and goal state constraints.

The motion prior part of the factorisation in Eq. 9 thus becomes a product of factors that takes into account the current position of a person, their intended goal and the constant-velocity movement assumption

$$f^{prior}(\boldsymbol{X}_h) = f^{start}(\boldsymbol{x}_{h,0}) f^{goal}(\boldsymbol{x}_{h,N}) \prod_{i=0}^{N-1} f^{gp}(\boldsymbol{x}_{h,i}, \boldsymbol{x}_{h,i+1}). \tag{13}$$

The remaining product of factors represents the likelihood $f^{like}(\boldsymbol{X}_h)$ and encodes all other state-dependent costs and constraints. In the case of human trajectory prediction, we partition it into separate factors that encode collision avoidance with respect to the environment, $f^{obs}$, and collision avoidance with respect to the moving robot, $f^{robot}$. The likelihood thus becomes

$$f^{like}(\boldsymbol{X}_h) = \prod_{i=1}^{N-1} f_i^{obs}(\boldsymbol{x}_{h,i}) f_i^{robot}(\boldsymbol{x}_{h,i}). \tag{14}$$

For environment collision avoidance factors, we adopt the formulation from GPMP2 (Mukadam et al. 2018) which uses a hinge loss function on the Euclidean distance field of the environment to penalise states that are close to obstacles. As described in Sec. 8, in practice, this distance field is provided by the perception part of our pipeline and updated online; this is in contrast to previous works which pre-compute it Mukadam et al. (2018); Park et al. (2012).

For the robot avoidance factor, $f^{robot}$, we propose

$$f_i^{robot}(\boldsymbol{x}_{h,i}) = \exp\{-\frac{1}{2}\|\boldsymbol{h}(\boldsymbol{x}_{h,i})\|^2_{\boldsymbol{\Sigma}_r}\}, \tag{15}$$

where $\boldsymbol{h}(\boldsymbol{x}_{h,i})$ is the hinge loss function of the distance between a person and the robot at the current position, $\boldsymbol{x}_r$. The hinge loss function is defined as

$$\boldsymbol{h}(\boldsymbol{x}_{h,i}) = \begin{cases} \varepsilon_r - \|\boldsymbol{x}_{h,i} - \boldsymbol{x}_r\|_2 & \text{if } \|\boldsymbol{x}_{h,i} - \boldsymbol{x}_r\|_2 \leq \varepsilon_r \\ 0 & \text{if } \|\boldsymbol{x}_{h,i} - \boldsymbol{x}_r\|_2 > \varepsilon_r \end{cases}. \tag{16}$$

$\varepsilon_r$ is a tolerance parameter of our formulation which indicates how close a person is likely to get to a robot before altering their trajectory to avoid collision. If a person is

sufficiently far away from the robot, we assume that they will not change their behaviour. However, if the robot comes within the *safety distance*, $\varepsilon_r$, our assumption is that a person will change their behaviour to move away from the robot and avoid collision.

The complete factor graph that we propose for human trajectory prediction can be written as

$$p(\boldsymbol{x_h}|\boldsymbol{e}) \propto f^{start} f^{goal} \prod_{i=0}^{N-1} f_{i,i+1}^{gp} \prod_{i=1}^{N-1} f_i^{obs} f_i^{robot}, \quad (17)$$

If we cannot determine a person's intended goal, for instance, if we do not have a set of possible goals or the estimated goal's probability is low, we can omit the goal prior factor. Our proposed prediction method will then work as a constant-velocity model that considers collisions via the robot and obstacle factors. We perform inference on the factor graph using the Levenberg-Marquardt optimisation method implemented in GTSAM Dellaert (2012) with an initial damping parameter of $0.01$.

## 7.3 Evaluation

We evaluate the proposed trajectory prediction method on our human motion dataset described in Sec. 4 and on the THÖR public dataset of human motion trajectories (Rudenko et al. 2020a). On both datasets, we predict over four different prediction horizons: $1.6\,\mathrm{s}$, $3.2\,\mathrm{s}$, $4.8\,\mathrm{s}$, and $8.0\,\mathrm{s}$. These prediction horizons cover both short-term and long-term human motion prediction and have previously been used in multiple evaluation pipelines, including the ATLAS benchmark (Rudenko et al. 2021). We thus use them for retaining consistency with the existing body of work in human motion prediction evaluation.

*7.3.1 Oxford-IHM Dataset* Our dataset comprises three different sensor measurements on which we evaluate trajectory prediction performance: the Vicon motion capture data, a static RGB-D camera and an HSR's head-mounted RGB-D camera. The Vicon motion capture data serves as the ground truth against which we compare our predicted trajectories for each type of sensor data. Evaluation of prediction methodologies on the motion capture data gives an indication of their potential performance when given accurate, high-frequency streams of data for the robot's pose, person's pose and goal locations. On the other hand, evaluation of each method on the data obtained using static and robot-mounted RGB-D cameras indicates their respective prediction performance when operating in real-world environments. Predictions on these data sources account for errors in human position estimation arising from factors such as measurement noise, misdetections, and occlusions.

Further, we compare the performance of our proposed method on the motion capture data with two ablations: (1) without the intention recognition proposed in Sec. 7.1, (2) without the robot avoidance factor proposed in Eq. (15). These two ablation studies enable us to assess the respective impact of the two features on human motion prediction performance. For each data source, we compare the performance of our proposed method against two baselines: (1) Constant Velocity Model (CVM), (2) Linear Velocity Model (LVM), similar to the ATLAS benchmark (Rudenko et al. 2021). The *CVM* generates predictions by forward propagating the velocity of the person's last observed state, while the *LVM* model generates predictions by forward propagating the observed average velocity of the person. We evaluate trajectory prediction performance using commonly used geometric metrics: Average Displacement Error (ADE) and Final Displacement Error (FDE) (Rudenko et al. 2020b). ADE measures the average error across predicted trajectories and the ground truth trajectory, while FDE measures the error of the final predicted point. Since motion capture and RGB - D measurements are inherently asynchronous, for evaluation on RGB-D data sources, we use GP interpolation (Mukadam et al. 2018) between states that are temporally closest to the ground truth measurements. For the two baselines methods, we use linear interpolation.

The results of evaluating across all 12 runs in the Oxford-IHM dataset, are shown in Tables 3 and 4. From the results, our proposed method outperforms the baseline methods for each type of sensor data and every prediction horizon. For the shortest prediction horizon ($1.6\,\mathrm{s}$), our proposed method performs similarly to the *CVM* baseline; this is expected since our proposed method has a smoothness factor that is initialised with a constant-velocity motion model. For short prediction horizons, goal and environmental factors have a marginal impact on human motion in absolute terms, meaning that simple kinematic models can often suffice. However, as we predict over longer horizons, our proposed method significantly outperforms the baselines, in line with our expectations, since the *CVM* and *LVM* models do not predict goal-oriented behaviour and disregard environmental cues. Without intent recognition, our method achieves similar performance to the *CVM* and performs significantly worse than our complete proposed method, demonstrating the importance of intention recognition for human motion prediction in indoor environments. Our proposed method performs similarly with and without using the robot avoidance factor, with only minor improvements being achieved with the robot avoidance factor. However, we believe that this factor may become more significant when operating in cluttered environments since it achieves better prediction in specific cases, for example, when the robot blocks a direct path to the intended goal. Figure 8 shows an instance of such a situation in the Oxford-IHM dataset.

Using data from the static and robot-mounted RGB - D cameras, we see similar performance trends to those achieved using motion capture measurements, albeit with a greater error. However, the dominant source of this error arises from our position estimation approach applied to the RGB-D images. By comparing the position estimates obtained using our image processing with ground truth measurements, we observe average position estimation errors of $15.9\,\mathrm{cm}$ and $30.4\,\mathrm{cm}$ respectively for the HSR RGB-D and Static RGB-D data sources. These results suggest that by further improving the position estimation method, similar trajectory prediction performance can be achieved on RGB-D sources to that achieved using high-frequency motion capture. While the static sensor had fewer measurement drops due to having the whole environment in its field of view, the robot-mounted camera achieved better performance since the robot was usually closer to

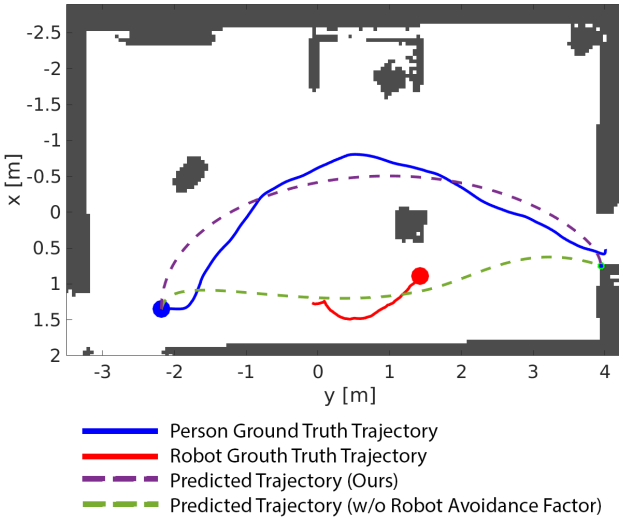



**Figure 8.** An instance in the Oxford-IHM dataset that highlights the impact of using a robot avoidance factor. Without the proposed robot avoidance factor, the predicted human trajectory significantly deviates from the ground truth and collides with the robot.

the person and thus more in accordance with the camera's recommended 'distance of use' for the camera. The results indicate that our proposed human trajectory prediction method works effectively with live sensor data and can be integrated within our proposed perception and motion planning framework as described in previous sections.

*7.3.2 THÖR dataset* We further benchmark our proposed method trajectory prediction on the THÖR dataset (Rudenko et al. 2020a) in which ten human subjects are tracked in an indoor environment with static obstacles and perform four different social roles that imitate typical activities found in populated spaces (e.g. offices). Enacting these roles results in various motion patterns, and nine out of the ten subjects exhibit goal-oriented behaviour. The dataset includes five labelled goals with known ground truth positions. The motion capture data provides ground truth trajectories against which we compare our predictions.

We use the ATLAS benchmark (Rudenko et al. 2021) to compare the performance of our proposed method against five different methods, including two baselines (*CVM* and *LVM*). The other three methods are local interaction models, namely the Social force model (*Sof*) (Helbing and Molnar 1995) and its two predictive extensions *Zan* (Zanlungo et al. 2011) and *Kara* (Karamouzas et al. 2009)). These models consider that multiple people are moving in the same environment and will anticipate and evade collisions with each other. As with the Oxford-IHM dataset, we evaluate trajectory prediction performance using the ADE and FDE.

The results of benchmarking for all subjects, across all four runs of the THÖR *One obstacle* experiment, are shown in Table 5. In contrast to the Oxford-IHM dataset, the THÖR dataset features fewer obstacles and a larger environment, resulting in more straight-line trajectories with constant velocity. Consequently, we achieve better performance than on the Oxford-IHM dataset. Our method is shown to outperform the baseline methods for all prediction horizons. While the *CVM* baseline achieves a similar level of performance on the shortest prediction horizon, our proposed method significantly outperforms on longer horizons for the reasons explained in Sec. 7.3.1 and in line with our expectations. Our method marginally outperformed the local interaction models (*Sof*, *Zan* and *Kara*) on the ADE metric, but was marginally worse on the FDE metric.

The relatively high standard deviations of the proposed method can be explained by the different social roles assigned to subjects on the THÖR dataset. For the *Lab Worker* and *Utility Worker* roles, the proposed method achieves superior performance in both ADE and FDE because these roles operate in a very goal-oriented manner with no social interactions. In contrast, subjects in the social role, *Visitor*, exhibited behaviours not currently modelled by our method, such as slowing down to interact with other people. For these roles, our proposed method had performed worse than the local interaction models which consider the social component of human behaviour. By not accounting for people slowing down, our proposed method often predicted people travelling further than the ground truth, mainly affecting the FDE performance.

*7.3.3 Parameters* Since we use GPMP2 as the backbone for trajectory optimisation, our trajectory prediction method depends on a similar set of parameters. Its parameter $Q_c$ specifies the uncertainty in the prior distribution and determines how heavily states are penalised for deviating away from the mean. $\Sigma_{obs}$ represents the obstacle cost weight with smaller values more strongly penalising collisions with obstacles. Since we introduce the robot avoidance factor in Eq. 15, we have the additional parameter, $\Sigma_r$, that we set to the same value as $\Sigma_{obs}$ throughout this paper, equally penalising collisions with the robot and the static environment. The parameter $\varepsilon$ represents a *safety distance* from static obstacles. For larger values of $\varepsilon$, optimised trajectories will deviate more from a straight line to maintain a larger distance from obstacles. The proposed robot avoidance factor has a similar parameter, $\varepsilon_r$, indicating a desired *safety distance* from the robot.

For our evaluation, we performed a grid search over parameters $Q_c$ and $\Sigma_{obs}$ for the Oxford-IHM and THÖR datasets. We found that good trajectory performance was achieved for parameters in the following ranges; $Q_c \in [0.01, 0.5]$ and $\Sigma_{obs} \in [0.02, 0.3]$. We used $Q_c = 0.2$ on Oxford-IHM and $Q_c = 0.05$ on the THÖR dataset. Since ground truth human trajectories on the Oxford-IHM dataset were less smooth than on THÖR due to its environment being smaller and more cluttered, a larger value of $Q_c$ was needed to better account for obstacle avoidance in tight spaces. On both datasets we used $\Sigma_{obs} = 0.1$ and $\varepsilon = 0.4$, resulting in collision-free trajectories for most predictions. Note that $\varepsilon$ also considers the radius of a person since the obstacle cost looks at the distance between the *centre* of a person and the nearest obstacle. The *safety distance* from the robot was set as $\varepsilon_r = 0.8$, double the distance from static obstacles because we account for the robot's radius and also that a person is likely to move farther from a moving robot than a static obstacle.

Due to the underlying continuous-time trajectory representation, we must define the total duration of a trajectory, corresponding to an estimation of the time required for a

| | | Prediction horizon (s) | | | |
|---|---|---|---|---|---|
| | Method | 1.6 | 3.2 | 4.8 | 8.0 |
| Vicon | LVM | $0.5 \pm 0.02$ | $1.08 \pm 0.04$ | $1.7 \pm 0.05$ | $2.98 \pm 0.07$ |
| | CVM | $\mathbf{0.28 \pm 0.03}$ | $0.75 \pm 0.04$ | $1.35 \pm 0.07$ | $2.75 \pm 0.11$ |
| | Ours | $\mathbf{0.28 \pm 0.02}$ | $\mathbf{0.66 \pm 0.06}$ | $\mathbf{0.99 \pm 0.09}$ | $\mathbf{1.54 \pm 0.11}$ |
| | Ours w/o Factor | $\mathbf{0.28 \pm 0.03}$ | $0.68 \pm 0.07$ | $1.04 \pm 0.09$ | $1.59 \pm 0.12$ |
| | Ours w/o Intent | $\mathbf{0.28 \pm 0.03}$ | $0.74 \pm 0.05$ | $1.25 \pm 0.08$ | $2.54 \pm 0.12$ |
| Static RGB-D | LVM | $0.78 \pm 0.03$ | $1.38 \pm 0.06$ | $2.02 \pm 0.09$ | $3.28 \pm 0.1$ |
| | CVM | $\mathbf{0.59 \pm 0.03}$ | $1.03 \pm 0.06$ | $1.65 \pm 0.1$ | $3.03 \pm 0.13$ |
| | Ours | $\mathbf{0.59 \pm 0.04}$ | $\mathbf{0.95 \pm 0.08}$ | $\mathbf{1.13 \pm 0.1}$ | $\mathbf{1.86 \pm 0.14}$ |
| HSR RGB-D | LVM | $0.66 \pm 0.03$ | $1.26 \pm 0.06$ | $1.89 \pm 0.09$ | $3.12 \pm 0.09$ |
| | CVM | $0.44 \pm 0.03$ | $0.91 \pm 0.05$ | $1.54 \pm 0.08$ | $2.99 \pm 0.10$ |
| | Ours | $\mathbf{0.43 \pm 0.03}$ | $\mathbf{0.81 \pm 0.07}$ | $\mathbf{1.13 \pm 0.1}$ | $\mathbf{1.71 \pm 0.13}$ |

**Table 3.** Average Displacement Error (ADE) on the Oxford-IHM dataset

| | | Prediction horizon (s) | | | |
|---|---|---|---|---|---|
| | Method | 1.6 | 3.2 | 4.8 | 8.0 |
| Vicon | LVM | $1.03 \pm 0.03$ | $2.29 \pm 0.07$ | $3.6 \pm 0.09$ | $6.09 \pm 0.18$ |
| | CVM | $\mathbf{0.64 \pm 0.04}$ | $1.82 \pm 0.08$ | $3.26 \pm 0.14$ | $6.41 \pm 0.2$ |
| | Ours | $0.65 \pm 0.04$ | $\mathbf{1.36 \pm 0.18}$ | $\mathbf{1.98 \pm 0.14}$ | $\mathbf{2.88 \pm 0.15}$ |
| | Ours w/o Factor | $0.65 \pm 0.04$ | $1.38 \pm 0.18$ | $2.02 \pm 0.14$ | $2.94 \pm 0.16$ |
| | Ours w/o Intent | $0.65 \pm 0.04$ | $1.82 \pm 0.08$ | $3.2 \pm 0.14$ | $6.11 \pm 0.21$ |
| Static RGB-D | LVM | $1.34 \pm 0.04$ | $2.6 \pm 0.08$ | $3.88 \pm 0.11$ | $6.38 \pm 0.2$ |
| | CVM | $\mathbf{0.92 \pm 0.05}$ | $2.1 \pm 0.07$ | $3.54 \pm 0.13$ | $6.59 \pm 0.22$ |
| | Ours | $\mathbf{0.92 \pm 0.05}$ | $\mathbf{1.63 \pm 0.14}$ | $\mathbf{2.24 \pm 0.16}$ | $\mathbf{3.13 \pm 0.18}$ |
| HSR RGB-D | LVM | $1.18 \pm 0.04$ | $2.45 \pm 0.09$ | $3.75 \pm 0.09$ | $6.17 \pm 0.18$ |
| | CVM | $\mathbf{0.80 \pm 0.04}$ | $1.98 \pm 0.1$ | $3.41 \pm 0.16$ | $6.51 \pm 0.16$ |
| | Ours | $0.81 \pm 0.04$ | $\mathbf{1.42 \pm 0.13}$ | $\mathbf{2.12 \pm 0.18}$ | $\mathbf{3.02 \pm 0.18}$ |

**Table 4.** Final Displacement Error (FDE) on the Oxford-IHM dataset

| | Prediction horizon (s) | | | | |
|---|---|---|---|---|---|
| | Method | 1.6 | 3.2 | 4.8 | 8.0 |
| ADE | CVM | $\mathbf{0.15 \pm 0.09}$ | $0.38 \pm 0.24$ | $0.71 \pm 0.45$ | $1.51 \pm 0.91$ |
| | LIN | $0.29 \pm 0.18$ | $0.60 \pm 0.38$ | $0.99 \pm 0.63$ | $1.84 \pm 1.08$ |
| | Sof | $0.18 \pm 0.10$ | $0.36 \pm 0.20$ | $0.60 \pm 0.35$ | $1.13 \pm 0.67$ |
| | Zan | $\mathbf{0.15 \pm 0.09}$ | $0.34 \pm 0.20$ | $0.59 \pm 0.36$ | $1.16 \pm 0.70$ |
| | Kara | $0.16 \pm 0.08$ | $0.35 \pm 0.19$ | $0.60 \pm 0.36$ | $1.16 \pm 0.69$ |
| | Ours | $\mathbf{0.15 \pm 0.09}$ | $\mathbf{0.33 \pm 0.26}$ | $\mathbf{0.57 \pm 0.41}$ | $\mathbf{1.12 \pm 0.83}$ |
| FDE | CVM | $0.28 \pm 0.18$ | $0.86 \pm 0.54$ | $1.64 \pm 1.05$ | $3.54 \pm 2.11$ |
| | LIN | $0.49 \pm 0.31$ | $1.20 \pm 0.75$ | $2.07 \pm 1.30$ | $3.97 \pm 2.27$ |
| | Sof | $0.29 \pm 0.16$ | $\mathbf{0.72 \pm 0.42}$ | $\mathbf{1.27 \pm 0.79}$ | $\mathbf{2.48 \pm 1.54}$ |
| | Zan | $\mathbf{0.26 \pm 0.16}$ | $\mathbf{0.72 \pm 0.43}$ | $1.31 \pm 0.82$ | $2.62 \pm 1.61$ |
| | Kara | $0.28 \pm 0.15$ | $0.73 \pm 0.42$ | $1.31 \pm 0.82$ | $2.59 \pm 1.59$ |
| | Ours | $0.28 \pm 0.17$ | $0.78 \pm 0.64$ | $1.41 \pm 0.99$ | $2.98 \pm 1.89$ |

**Table 5.** ADE and FDE on the THÖR dataset

person to reach their intended goal. We estimate this time by calculating the distance between the person's current position and their intended goal and dividing it by their current velocity. For estimated times shorter than the prediction horizon used for evaluation, we use the goal state to predict a person's position for timestamps after the estimated trajectory time.

## 8 Receding Horizon And Predictive Gaussian Process Motion Planner 2

This section describes how we integrate the methods and concepts discussed in previous sections within a single framework to be deployed on a physical robot. We build upon the integrated perception and motion planning framework described in Finean et al. (2021); we use the GPU-Voxels framework to maintain a voxelmap of the scene and compute distance fields (Hermann et al. 2014; Juelg et al. 2018), while motion planning is performed using GPMP2 in a receding-horizon manner. However, we introduce several extensions.

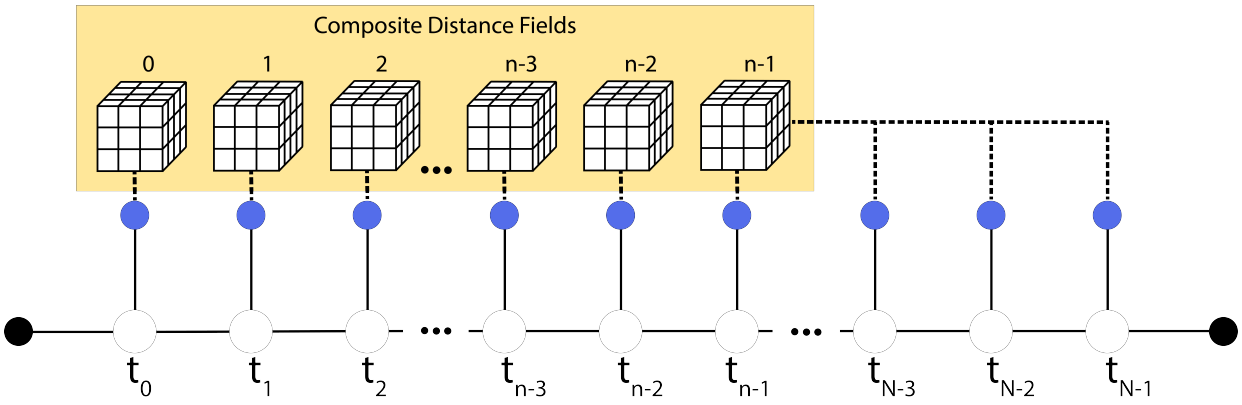



**Figure 9.** The assignment of composite distance fields to the obstacle factors (blue) in RHAP-GPMP2. Given a long time horizon of $N$ timesteps, we assign independent distance fields to the first $n$ timesteps, where $n$ is our dynamic obstacle prediction horizon. For time-indexed obstacle factors greater than $n$, we assign the $n^{\text{th}}$ distance field.

Firstly, as discussed in Sec. 3 and Sec. 5, we introduce image segmentation to remove dynamic obstacles prior to generating pointclouds for integration into the maintained voxelmap of the static scene. At this point, the voxelmap can undergo further filtering if necessary. In the presence of dynamic obstacles, we found it beneficial to filter out voxels that have fewer than five connected voxels; this reduced the instances of spurious voxels being designated as occupied in the voxelmap of the static scene.

Secondly, we propose the Receding Horizon And Predictive Gaussian Process Motion Planner 2 (RHAP-GPMP2). As described in Sec. 7.2, GPMP2 formulates the motion planning problem as probabilistic inference on a factor graph. In RHAP-GPMP2, we continuously monitor the validity of the current trajectory, re-estimate the expected time-to-goal, and re-optimise trajectories to re-evaluate their cost as we observe the environment. If the current trajectory becomes invalid or a re-optimised trajectory significantly lowers the cost, we generate a new factor graph for trajectory optimisation. We use a straight-line trajectory initialisation for the first optimisation and in recovery behaviours; otherwise, we re-use and re-optimise the previously planned trajectory to maintain smoothly executed trajectories. In previous work, we used a singular voxelmap that is maintained, converted to a distance field, and sent to all obstacle factors within the factor graph used for motion planning. However, RHAP-GPMP2 builds upon the concepts presented in Finean et al. (2020) and extends the motion planning work to time-configuration space planning. To achieve this, we maintain:

1. A static voxelmap of the scene
2. A distance field (static or maintained) for each dynamic obstacle (discussed in Sec. 6)
3. A sequence of $n$ composite distance fields.

The variable $n$ is determined by how far into the future we wish to incorporate predicted positions for moving objects in the scene. In this work, we use a time-discretisation between factor graph support states of $0.5\,\text{s}$ and so choose a value of $n = 20$, corresponding to a time horizon of $10\,\text{s}$. Each time-indexed obstacle factor in the factor graph is associated with a corresponding time-indexed composite distance field. For time-indices greater than $n$, we assign the composite distance field for time index $n$. Distance field assignment for RHAP-GPMP2 is illustrated in Fig. 9.

During each update loop, the static voxelmap is updated using the latest observations of the scene (with the dilated dynamic obstacle masks removed) and composite distance fields are generated using the latest trajectory predictions for dynamic obstacles in the scene, as predicted by our trajectory prediction module described in Sec. 3, Sec. 7.1, and Sec. 7.2. To integrate the human prediction module, we additionally calculate a 2D EDT of the environment by collapsing the maintained 3D voxel grid to 2D and using PBA to calculate the corresponding EDT on the GPU. The EDT is then transferred to the CPU for use in the trajectory prediction module.

## 9 Live Hardware Experiments

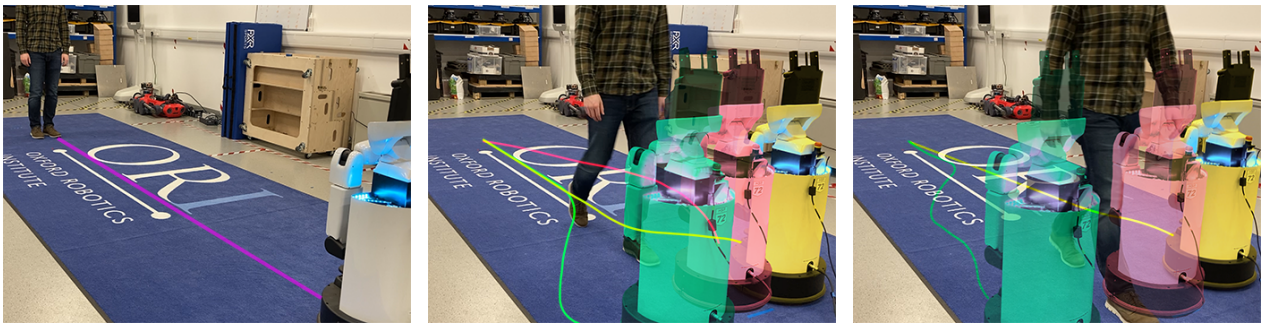
To demonstrate the robustness and capabilities of our whole integrated framework, we deploy our implementation on a physical Toyota Human Support Robot and explore both base-only and whole-body tasks across a range of dynamic scenarios as follows:

1. Change of Places
2. Change of Places with Obstacle
3. Multi-Goal
4. Narrow Passage
5. Change of Places with Half-Wall

Note that in base-only tasks, the motion planner still optimises in the high-dimensional space of whole-body motions. The baselines of interest for this work are: (1) *No Prediction* and (2) Prediction using *CVM*.

In (Finean et al. 2021), the robot was able to adapt to changes in the environment; however, the human's trajectory was not aimed directly towards the robot in any of the tasks. Hence, the robot was able to perform sufficiently well without prediction. In contrast, for most tasks presented in this work, the robot is required to move out of the way of a person in order to avoid collision.

In our hardware experiments, we perform calculations on an external laptop connected to the HSR via an Ethernet



**(a)** Start **(b)** *No Prediction* results in collision **(c)** Both predictive methods succeed

**Figure 10.** *Change of Places* – the robot is tasked with a base-only goal in front of a person. During execution, the person walks towards a goal behind the robot, requiring the robot to react and move out of the way. All trajectories initially follow a straight line (purple), but as the person approaches, the replanned trajectories for each method diverge as highlighted by the different colours for *No Prediction* (red), *CVM with Prediction* (yellow), *Proposed Method with Prediction* (green). We find that without accounting for the prediction trajectory of the person, the robot collides with the person. In contrast, our proposed framework can avoid collision and complete the task.

connection to provide sufficient bandwidth for the transfer of images. Hardware specifications for the laptop are: NVIDIA RTX 2070 Super GPU, 8-core Intel Core i9-10980HK CPU @ 5.30 GHz and 2667 MHz DDR4 RAM.

*Results* Due to the relatively high speed of the human motions in our hardware experiments, we found that the robot always ended up in collision without accounting for the human's predicted motion. We provide supplementary video footage of these experiments[‡] and describe the results of each experiment in the following subsections.

### 9.1 Change of Places - Prediction is Needed

In our simplest task, *Change of Places*, we do not provide static obstacles, and the robot is tasked with a base-only goal in front of a person. During the task, the person walks towards a goal behind the robot. The resultant task is illustrated in Fig. 10. We observed that in the *No Prediction* case, without predicting the human's trajectory, re-planned robot trajectories repeatedly become invalid, resulting in collision. Due to the straight-line nature of this task, we found that the *CVM* performed equally as well as our full prediction method. For brevity, in further tasks, we only consider the *CVM* baseline.

### 9.2 Change of Places with Obstacle - Limitations of the CVM

With the addition of a central obstacle to the previous task, both the robot and human must take curved geodesics. Results are illustrated in Fig. 11. This experiment highlighted the limitation of the *CVM* – when a person follows a curved path, the resultant prediction is tangential to the actual path. In this task, this erroneous prediction results in collision. In contrast, our method accurately predicted the person's trajectory, enabling the robot to follow a collision-free trajectory.

### 9.3 Multi-Goal - Robust to Intention Recognition

While the goal-based aspect of our prediction framework is more extensively evaluated in Sec. 7.3, we provide a hardware experiment in which the person is determined to have two potential goals: one behind the robot's starting position, the other at a hand-wash station across the robot's path. The robot is tasked with a whole-body motion to place a can on the table opposite. Across multiple trials, the robot successfully predicts the person's intended goal and adapts its motions appropriately to execute the task collision-free. Figure 12 illustrates these results.

### 9.4 Additional Capabilities

We additionally tested our approach in a *Narrow Passage* task and a variant of the *Change of Places with Obstacle* experiment in which the central obstacle was replaced with a wall across half of the room. In both tasks, the robot successfully adapted to the predicted trajectory of the person and moved to the side of the person's path before continuing towards the goal. Images from the two tasks are shown in Figures 1 and 13.

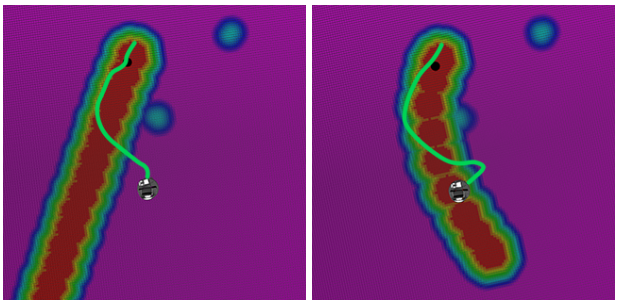
## 10 Discussion

One advantage of our proposed human trajectory prediction approach is that it bridges the gap between model-based and learning-based prediction methods. A major limitation of learnt models is their ability to transfer to scenarios that differ from those in which it was trained. In contrast, an appealing attribute of our human trajectory prediction method is that it can be readily deployed in any environment by defaulting to a constant-velocity model while retaining the ability to improve over time as a prior distribution is learnt over likely human goal intentions. An interesting direction for further research would be to explore the online-learning of human intentions further and incorporate scene semantics.

[‡]Supplementary video available at `https://youtu.be/gdC3mpZNjG4`

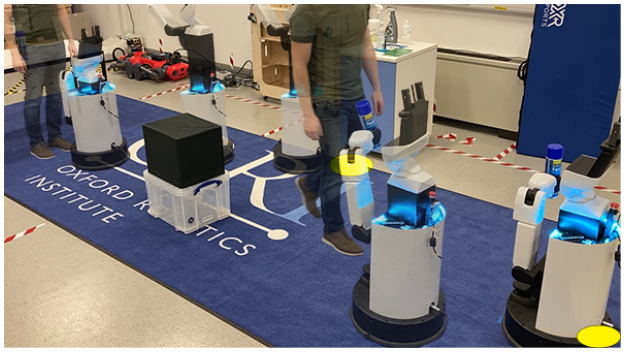


**(a)** *Change of Places with Obstacle*

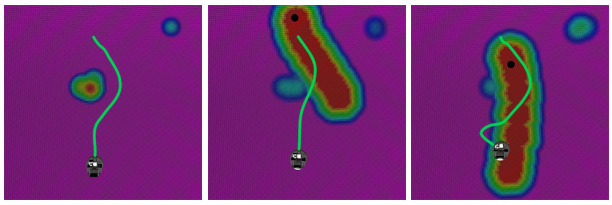


**(b)** CVM Prediction **(c)** Our Prediction

**Figure 11.** *Change of Places with Obstacle* – a person walks around a static obstacle towards the robot's starting location. Meanwhile, the robot is tasked with a whole-body goal to place a canister on top of a table on the other side of the room. Fig. 11a shows the trajectories taken by our method (green) and using the *CVM* (yellow). While our methods avoided the person, the *CVM* trajectory did not move out of the way in time, requiring the human to slow down to avoid a collision. The reason for this is explained by Figures 11b and 11c which show superposed 2D projections of the 3D composite distance fields used for motion planning at two given moments in time. While our prediction method more accurately predicts the human's trajectory, the *CVM* predicts a trajectory tangential to the actual one taken.



**(a)**



**(b)** **(c)** **(d)**

**Figure 12.** *Multi-Goal* – the robot is tasked with a whole-body goal to place a canister on a table on the other side of a static obstacle. A person in the scene has two possible goal locations - the first is behind the robot's starting location, the second is at a hand-wash station in front of the robot. This experiment demonstrated our framework's ability to adapt and update human trajectory predictions even when the human's intended goal is deemed to have changed. Figure 12b shows the initial planned robot trajectory superposed on an aerial view of the ground distance field. Figure 12c shows the updated trajectory as the human is deemed to be moving towards the hand-wash station while Fig. 12c shows the updated trajectory as the prediction module correctly identifies that the person's intended goal is the one behind the robot.

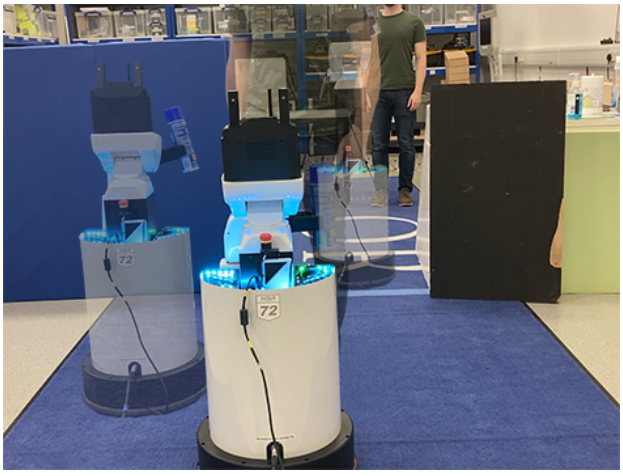


**Figure 13.** *Narrow Passage* – the robot is tasked with a whole-body goal to place a canister on top of a table that is on the other side of a narrow passage. During execution, a person walks through the narrow passage to act as a dynamic obstacle. Our method achieved a successful collision-free trajectory by re-planning to move to the side while the person walked past.

In (Finean et al. 2021), no additional filtering was applied to the maintained voxelmap to remove 'lingering' voxels that a moving obstacle may leave. In this work, we found that these lingering voxels provided a substantial disadvantage for motion planning compared to our proposed method; the proposed method uses dilated segmentation of dynamic obstacles, resulting in a cleaner static voxelmap. As such, to provide an appropriate *No Prediction* baseline in line with our previous work, we introduced the segmentation pipeline such that the human obstacle is tracked and composited into the singular distance field used for the motion planning.

While we obtained robust re-planning and collision avoidance behaviours across various tasks, there are several limitations in the presented work that are worth noting. Firstly, we do not explicitly model uncertainty in our current method of compositing the predicted positions of moving obstacles. Instead, we account for a margin-of-safety via the $\epsilon$ parameter within the GPMP2-based obstacle factors – $\epsilon$ determines the upper distance used for hinge-loss obstacle costs. While further exploration of this was beyond the scope of the presented work, we could enlarge the volume of the person/cylinder over the course of the prediction time

horizon to appropriately account for a growing uncertainty in future position as time increases.

In this work, we demonstrated that GPU implementations of predicted composite distance fields can provide a significant performance boost compared to calculating distance fields from scratch. However, as shown in Fig. 6, the key bottleneck for further composite distance field performance gains is the device-host transfer time. Future work could explore alternative approaches to minimise data transfer between device and host.

A natural limitation of our motion planning implementation is that the optimisation is prone to get stuck in local minima by only optimising a single trajectory. While this did not result in collisions in our experiments, the phenomenon is evident in trajectories such as Figures 11c and 12c – rather than planning to travel on the other side of the static obstacle to the human, the re-planned trajectory avoids the human but stays within the same homotopy class. To address this, one could consider maintaining and optimising multiple trajectories at a time in different homotopy classes, such as work by Kolur et al. (2019) and Merkt et al. (2021), however, this is likely to increase the planning time and limit the robot's ability to react.

It is worth noting that we use two different motion planning approaches in our proposed framework. For robot motion planning, we use predicted versions of the environment for each time step, while for human trajectory prediction, we only use the latest observation of the environment. Our reasoning for this is two-fold; firstly, the walking speed of a human is significantly higher than that of the robot's mobile speed, so a human has less need to account for the predicted trajectory of the robot. Secondly, from our experience, humans will readily travel much closer towards the path of a moving robot, while from the robot perspective, we need to retain a more cautious approach to collision avoidance.

## 11 Conclusion

To enable predictive whole-body motion planning in dynamic environments, we introduced several novel methods and integrated them within a novel framework that can account for the predicted trajectories of humans in a scene. We firstly proposed an intention-aware trajectory prediction model for humans in indoor environments and demonstrated state-of-the-art performance on both a publicly available dataset as well as our own goal-oriented dataset, the Oxford Indoor Human Motion (Oxford-IHM) dataset, that we make publicly available.

For predictive and reactive motion planning, we proposed the Receding Horizon And Predictive Gaussian Process Motion Planner 2 (RHAP-GPMP2), a receding-horizon motion planner that utilising predicted composite distance fields to embed the predicted trajectories of moving obstacles. To this end, we demonstrated the viability and effectiveness of composite distance fields in a GPU-based perception framework and show that composite distance fields can reduce distance field computation times by $89\,\%$ to $93\,\%$, underpinning our integrated framework's ability to avoid moving obstacles in real-world environments.

We verified our proposed framework on a physical Toyota Human Support Robot (HSR) and demonstrated that our system can use live sensor measurements to predict and incorporate the trajectories of humans in a robot's workspace, enabling it to avoid collisions when performing whole-body motion planning across a variety of challenging and dynamic environments.

### Funding

This research was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) through the University of Oxford Centre for Doctoral Training, Autonomous Intelligent Machines and Systems (AIMS) [grant references EP/L015897/1, EP/R512333/1], the European Regional Development Fund (DATACROSS) [grant reference KK.01.1.1.01.0009], and in part by the Human-Machine Collaboration Programme, supported by a gift from Amazon Web Services.

### Declaration of Conflicting Interests

The Authors declare that there is no conflict of interest.

### References

Alahi A, Goel K, Ramanathan V, Robicquet A, Fei-Fei L and Savarese S (2016) Social lstm: Human trajectory prediction in crowded spaces. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 961–971.

Amirian J, Hayet JB and Pettré J (2019) Social ways: Learning multi-modal distributions of pedestrian trajectories with gans. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. pp. 0–0.

Bartoli F, Lisanti G, Ballan L and Del Bimbo A (2018) Context-aware trajectory prediction. In: *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, pp. 1941–1946.

Batkovic I, Zanon M, Lubbe N and Falcone P (2018) A computationally efficient model for pedestrian motion prediction. In: *2018 European Control Conference (ECC)*. IEEE. ISBN 978-3-9524-2698-2, pp. 374–379. DOI:10.23919/ECC.2018.8550300.

Bernardin K and Stiefelhagen R (2008) Evaluating multiple object tracking performance: The clear mot metrics. *EURASIP Journal on Image and Video Processing* DOI:10.1155/2008/246309.

Best G and Fitch R (2015) Bayesian intention inference for trajectory prediction with an unknown goal destination. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5817–5823.

Brscic D, Kanda T, Ikeda T and Miyashita T (2013) Person tracking in large public spaces using 3-d range sensors. *IEEE Transactions on Human-Machine Systems* 43: 522–534. DOI:10.1109/THMS.2013.2283945.

Cao TT, Tang K, Mohamed A and Tan TS (2010) Parallel Banding Algorithm to compute exact distance transform with the GPU. In: *ACM SIGGRAPH I3D*. ISBN 9781605589381, pp. 83–90. DOI:10.1145/1730804.1730818.

Cao Z, Gao H, Mangalam K, Cai QZ, Vo M and Malik J (2020) Long-term human motion prediction with scene context. In: *European Conference on Computer Vision*. Springer, pp. 387–404.

Cao Z, Hidalgo G, Simon T, Wei SE and Sheikh Y (2021) OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43(1): 172–186. DOI:10.1109/TPAMI.2019.2929257.

Dellaert F (2012) Factor graphs and gtsam: A hands-on introduction. Technical report, Georgia Institute of Technology.

Dou M, Taylor J, Fuchs H, Fitzgibbon A and Izadi S (2015) 3d scanning deformable objects with a single rgbd sensor. IEEE. ISBN 978-1-4673-6964-0, pp. 493–501. DOI:10.1109/CVPR.2015.7298647. URL http://ieeexplore.ieee.org/document/7298647/.

Finean MN, Merkt W and Havoutis I (2020) Predicted composite signed-distance fields for real-time motion planning in dynamic environments. In: *International Conference Conf. Automat. Planning and Scheduling*.

Finean MN, Merkt W and Havoutis I (2021) Simultaneous Scene Reconstruction and Whole-Body Motion Planning for Safe Operation in Dynamic Environments. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Flores C, Merdrignac P, de Charette R, Navas F, Milanes V and Nashashibi F (2019) A cooperative car-following/emergency braking system with prediction-based pedestrian avoidance capabilities. *IEEE Transactions on Intelligent Transportation Systems* 20: 1837–1846. DOI:10.1109/TITS.2018.2841644.

Ghosh P, Song J, Aksan E and Hilliges O (2017) Learning human motion models for long-term predictions. In: *2017 International Conference on 3D Vision (3DV)*. IEEE, pp. 458–466.

Han L, Gao F, Zhou B and Shen S (2019) FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISBN 9781728140049, pp. 4423–4430. DOI:10.1109/IROS40897.2019.8968199.

Handa A, Whelan T, McDonald J and Davison AJ (2014) A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In: *IEEE International Conference on Robotics and Automation*. IEEE. ISBN 978-1-4799-3685-4, pp. 1524–1531. DOI:10.1109/ICRA.2014.6907054.

He K, Gkioxari G, Dollár P and Girshick R (2020) Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42(2): 386–397. DOI:10.1109/TPAMI.2018.2844175.

Helbing D and Molnar P (1995) Social force model for pedestrian dynamics. *Physical review E* 51(5): 4282.

Hermann A, Drews F, Bauer J, Klemm S, Roennau A and Dillmann R (2014) Unified GPU voxel collision detection for mobile manipulation planning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISBN 9781479969340, pp. 4154–4160. DOI:10.1109/IROS.2014.6943148.

Hermann A, Mauch F, Fischnaller K, Klemm S, Roennau A and Dillmann R (2015) Anticipate your surroundings: Predictive collision detection between dynamic obstacles and planned robot trajectories on the gpu. In: *2015 European Conference on Mobile Robots (ECMR)*. IEEE, pp. 1–8.

Janoch A, Karayev S, Jia Y, Barron JT, Fritz M, Saenko K and Darrell T (2011) A category-level 3-d object dataset: Putting the kinect to work. In: *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE. ISBN 978-1-4673-0063-6, pp. 1168–1174. DOI:10.1109/ICCVW.2011.6130382.

Juelg C, Hermann A, Roennau A and Dillmann R (2018) Fast online collision avoidance for mobile service robots through potential fields on 3D environment data processed on GPUs. In: *IEEE International Conference on Robotics and Biomimetics (ROBIO)*. ISBN 9781538637418. DOI:10.1109/ROBIO.2017.8324535.

Kalakrishnan M, Chitta S, Theodorou E, Pastor P and Schaal S (2011) Stomp: Stochastic trajectory optimization for motion planning. In: *IEEE International Conference on Robotics and Automation*. IEEE. ISBN 978-1-61284-386-5, pp. 4569–4574. DOI:10.1109/ICRA.2011.5980280.

Karamouzas I, Heil P, Van Beek P and Overmars MH (2009) A predictive collision avoidance model for pedestrian simulation. In: *International workshop on motion in games*. Springer, pp. 41–52.

Kellnhofer P, Recasens A, Stent S, Matusik W and Torralba A (2019) Gaze360: Physically unconstrained gaze estimation in the wild. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 6912–6921.

Kitani KM, Ziebart BD, Bagnell JA and Hebert M (2012) Activity forecasting. In: *European conference on computer vision*. Springer, pp. 201–214.

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

Kolur K, Chintalapudi S, Boots B and Mukadam M (2019) Online motion planning over multiple homotopy classes with gaussian process inference. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISBN 978-1-7281-4004-9, pp. 2358–2364. DOI:10.1109/IROS40897.2019.8967598.

Kratzer P, Bihlmaier S, Midlagajni NB, Prakash R, Toussaint M and Mainprice J (2021) Mogaze: A dataset of full-body motions that includes workspace geometry and eye-gaze. *IEEE Robotics and Automation Letters* 6: 367–373. DOI:10.1109/LRA.2020.3043167. URL https://ieeexplore.ieee.org/document/9286421/.

Kratzer P, Toussaint M and Mainprice J (2020) Prediction of human full-body movements with motion optimization and recurrent neural networks. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1792–1798.

Kschischang FR, Frey BJ and Loeliger HA (2001) Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory* 47(2): 498–519.

Lai K, Bo L, Ren X and Fox D (2011) A large-scale hierarchical multi-view rgb-d object dataset. In: *IEEE International Conference on Robotics and Automation*. IEEE. ISBN 978-1-61284-386-5, pp. 1817–1824. DOI:10.1109/ICRA.2011.5980382.

Lee Y and Park J (2020) CenterMask: Real-time anchor-free instance segmentation. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. ISBN 1911.06667v6, pp. 13903–13912. DOI:10.1109/CVPR42600.2020.01392.

Luo Y, Cai P, Bera A, Hsu D, Lee WS and Manocha D (2018) Porca: Modeling and planning for autonomous driving among many pedestrians. *IEEE Robotics and Automation Letters* 3: 3418–3425. DOI:10.1109/LRA.2018.2852793.

Mainprice J and Berenson D (2013) Human-robot collaborative manipulation planning using early prediction of human motion. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISBN 9781467363587, pp. 299–306. DOI: 10.1109/IROS.2013.6696368.

Merkt WX, Ivan V, Dinev T, Havoutis I and Vijayakumar S (2021) Memory clustering using persistent homology for multimodality- and discontinuity-sensitive learning of optimal control warm-starts. *IEEE Transactions on Robotics* 37(5): 1649–1660. DOI:10.1109/TRO.2021.3069132.

Mukadam M, Dong J, Yan X, Dellaert F and Boots B (2018) Continuous-time Gaussian process motion planning via probabilistic inference. *The Int. J. of Rob. Res.* 37(11): 1319–1340. DOI:10.1177/0278364918790369.

Munaro M and Menegatti E (2014) Fast RGB-D people tracking for service robots. *Autonomous Robots* 37(3): 227–242. DOI: 10.1007/s10514-014-9385-0.

Newcombe RA, Fitzgibbon A, Izadi S, Hilliges O, Molyneaux D, Kim D, Davison AJ, Kohi P, Shotton J and Hodges S (2011) KinectFusion: Real-time dense surface mapping and tracking. In: *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. IEEE. ISBN 978-1-4577-2183-0, pp. 127–136. DOI:10.1109/ISMAR.2011.6092378.

Oleynikova H, Burri M, Taylor Z, Nieto J, Siegwart R and Galceran E (2016) Continuous-time trajectory optimization for online uav replanning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2016-November. IEEE. ISBN 978-1-5090-3762-9, pp. 5332–5339. DOI:10.1109/IROS.2016.7759784.

Oleynikova H, Taylor Z, Fehr M, Siegwart R and Nieto J (2017) Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2017-September. ISBN 9781538626825, pp. 1366–1373. DOI: 10.1109/IROS.2017.8202315.

Palazzolo E, Behley J, Lottes P, Giguere P and Stachniss C (2019) ReFusion: 3D Reconstruction in Dynamic Environments for RGB-D Cameras Exploiting Residuals. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISBN 9781728140049, pp. 7855–7862. DOI:10.1109/IROS40897.2019.8967590.

Park C, Pan J and Manocha D (2012) ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments. In: *International Conference Conf. Automat. Planning and Scheduling*. ISBN 9781577355625, pp. 207–215.

Park JS, Park C and Manocha D (2019) I-Planner: Intention-aware motion planning using learning-based human motion prediction. *The Int. J. of Rob. Res.* 38(1): 23–39. DOI: 10.1177/0278364918812981.

Park S, Spurr A and Hilliges O (2018) Deep pictorial gaze estimation. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. pp. 721–738.

Pellegrini S, Ess A, Schindler K and Van Gool L (2009) You'll never walk alone: Modeling social behavior for multi-target tracking. In: *2009 IEEE 12th International Conference on Computer Vision*. IEEE, pp. 261–268.

Rehder E, Wirth F, Lauer M and Stiller C (2018) Pedestrian prediction by planning using deep neural networks. In: *IEEE International Conference on Robotics and Automation*. IEEE. ISBN 978-1-5386-3081-5, pp. 1–5. DOI:10.1109/ICRA.2018.8460203.

Ren S, He K, Girshick R and Sun J (2017) Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39: 1137–1149. DOI:10.1109/TPAMI.2016.2577031. URL `http://ieeexplore.ieee.org/document/7485869/`.

Rösmann C, Oeljeklaus M, Hoffmann F and Bertram T (2017) Online trajectory prediction and planning for social robot navigation. In: *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, pp. 1255–1260.

Rudenko A, Huang W, Palmieri L, Arras KO and Lilienthal AJ (2021) Atlas : a Benchmarking Tool for Human Motion Prediction Algorithms. *Robotics: Science and Systems (RSS) Workshop on Social Robot Navigation* .

Rudenko A, Kucner TP, Swaminathan CS, Chadalavada RT, Arras KO and Lilienthal AJ (2020a) Thör: Human-robot navigation data collection and accurate motion trajectories dataset. *IEEE Robotics and Automation Letters* 5(2): 676–682.

Rudenko A, Palmieri L and Arras KO (2018) Joint long-term prediction of human motion using a planning-based social force approach. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4571–4577.

Rudenko A, Palmieri L, Herman M, Kitani KM, Gavrila DM and Arras KO (2020b) Human motion trajectory prediction: A survey. *The Int. J. of Rob. Res.* 39(8): 895–935.

Runz M, Buffier M and Agapito L (2019) MaskFusion: Real-Time Recognition, Tracking and Reconstruction of Multiple Moving Objects. In: *Proceedings of the 2018 IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2018*. ISBN 9781538674598, pp. 10–20. DOI:10.1109/ISMAR.2018.00024.

Schöller C, Aravantinos V, Lay F and Knoll A (2020) What the constant velocity model can teach us about pedestrian motion prediction. *IEEE Robotics and Automation Letters* 5(2): 1696–1703.

Scona R, Jaimez M, Petillot YR, Fallon M and Cremers D (2018) StaticFusion: Background Reconstruction for Dense RGB-D SLAM in Dynamic Environments. In: *IEEE International Conference on Robotics and Automation*. ISBN 9781538630815, pp. 3849–3856. DOI:10.1109/ICRA.2018.8460681.

Sturm J, Engelhard N, Endres F, Burgard W and Cremers D (2012) A benchmark for the evaluation of RGB-D SLAM systems. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. ISBN 9781467317375, pp. 573–580. DOI: 10.1109/IROS.2012.6385773.

Sung J, Ponce C, Selman B and Saxena A (2012) Unstructured human activity detection from rgbd images. In: *IEEE International Conference on Robotics and Automation*. IEEE. ISBN 978-1-4673-1405-3, pp. 842–849. DOI:10.1109/ICRA.2012.6224591.

Treuille A, Cooper S and Popović Z (2006) Continuum crowds. *ACM Transactions on Graphics (TOG)* 25(3): 1160–1168.

Vasquez D (2016) Novel planning-based algorithms for human motion prediction. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3317–3322.

Whelan T, Kaess M and Fallon M (2012) Kintinuous: Spatially extended kinectfusion. *RSS Workshop on RGB-D: Advanced*

*Reasoning with Depth Cameras* : 7.

Whelan T, Leutenegger S, Salas-Moreno RF, Glocker B and Davison AJ (2015) ElasticFusion: Dense SLAM without a pose graph. In: *Robotics: Science and Systems*, volume 11. ISBN 9780992374716. DOI:10.15607/RSS.2015.XI.001.

Xie W, Liu PX and Zheng M (2021) Moving Object Segmentation and Detection for Robust RGBD-SLAM in Dynamic Environments. *IEEE Transactions on Instrumentation and Measurement* 70. DOI:10.1109/TIM.2020.3026803.

Yan Z, Duckett T and Bellotto N (2017) Online learning for human classification in 3d lidar-based tracking. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 864–871.

Yu C, Ma X, Ren J, Zhao H and Yi S (2020) Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In: *European Conference on Computer Vision*. Springer, pp. 507–523.

Yu R, Russell C, Campbell NDF and Agapito L (2015) Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video. IEEE. ISBN 978-1-4673-8391-2, pp. 918–926. DOI:10.1109/ICCV.2015.111. URL `http://ieeexplore.ieee.org/document/7410468/`.

Zanlungo F, Ikeda T and Kanda T (2011) Social force model with explicit collision prediction. *EPL (Europhysics Letters)* 93(6): 68005.

Zhang H and Parker LE (2011) 4-dimensional local spatio-temporal features for human activity recognition. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. ISBN 978-1-61284-456-5, pp. 2044–2049. DOI:10.1109/IROS.2011.6094489.

Zhang T and Nakamura Y (2020) PoseFusion: Dense RGB-D SLAM in Dynamic Human Environments. In: *Springer Proceedings in Advanced Robotics*, volume 11. pp. 772–780. DOI:10.1007/978-3-030-33950-0_66.

Zhang Z, Zhang J and Tang Q (2019) Mask R-CNN Based Semantic RGB-D SLAM for Dynamic Scenes. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, volume 2019-July. ISBN 9781728124933, pp. 1151–1156. DOI:10.1109/AIM.2019.8868400.

Zhi W, Ott L and Ramos F (2021) Probabilistic trajectory prediction with structural constraints. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 9849–9856.

Zhou B, Gao F, Wang L, Liu C and Shen S (2019) Robust and efficient quadrotor trajectory generation for fast autonomous flight. *IEEE Robotics and Automation Letters* 4: 3529–3536. DOI:10.1109/LRA.2019.2927938.

Ziebart BD, Maas AL, Bagnell JA, Dey AK et al. (2008) Maximum entropy inverse reinforcement learning. In: *Aaai*, volume 8. Chicago, IL, USA, pp. 1433–1438.

Ziebart BD, Ratliff N, Gallagher G, Mertz C, Peterson K, Bagnell JA, Hebert M, Dey AK and Srinivasa S (2009) Planning-based prediction for pedestrians. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 3931–3936.

**Statement of Authorship for joint/multi-authored papers for PGR thesis**

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

| | |
|---|---|
| Title of Paper | Motion Planning in Dynamic Environments Using Context-Aware Human Trajectory Prediction |
| Publication Status | Submitted for publication |
| Publication Details | M. N. Finean, L. Petrović, W. Merkt, I. Marković, and I. Havoutis. "Motion Planning in Dynamic Environments Using Human Trajectory Prediction" |

Student Confirmation

| | | | |
|---|---|---|---|
| Student Name: | Mark Nicholas Finean | | |
| Contribution to the Paper | Developed the concepts and ideas related to perception, image processing, motion planning, and composite-sdfs<br>Jointly developed concepts related to proposed method of human trajectory prediction<br>Implementation of concepts (code)<br>Data collection for composite-sdf analysis<br>Devised and conducted hardware experiments<br>Jointly devised and collected the associated dataset<br>Wrote the paper with co-authors | | |
| Signature |  | Date | 14/01/2022 |

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

| | | | |
|---|---|---|---|
| Supervisor name and title: Dr. Ioannis Havoutis | | | |
| Supervisor comments | | | |
| Signature |  | Date | 14/01/2022 |

# 7 Summary and Future Work

## Contents


**7.1 Predicted Composite Distance Fields . . . . . 112**
**7.2 Real-time Mapping and Motion Planning . . . . . 112**
**7.3 Active Gaze Control . . . . . 113**
**7.4 Integrating Mobile Manipulation with Human Trajectory Prediction . . . . . 113**
**7.5 Limitations and Future Work . . . . . 114**
7.5.1 Active Gaze Control . . . . . 114
7.5.2 Integrated Motion Planning, Perception and Prediction Framework . . . . . 115
**7.6 Conclusion . . . . . 117**

In this thesis, we encountered new challenges, proposed novel algorithms and frameworks, and ultimately achieved the ambitious goal that we set out at the start of this research — we demonstrated and released a state-of-the-art integrated framework for using physical robots to perform loco-manipulation tasks in dynamic environments and avoid moving obstacles. This final chapter summarises our contributions and discusses the limitations of our work. We will conclude by describing several future directions for this research.

## 7.1 Predicted Composite Distance Fields

Many trajectory optimisation-based motion planners use signed distance fields to provide occupancy information and local gradients that guide the optimisation. In Chapter 3, we introduced the concept of *predicted composite distance fields* (PCDFs) as a means of directly embedding the predicted occupancy of dynamic obstacles into a distance field and showed that our method can generate predicted versions of this environment representation significantly faster than computing them from scratch. Further, we provided the first extension of GPMP2 to operate in a receding-horizon manner with time-dependent distance fields at each variable node in the associated factor graph. We demonstrated the resultant framework in both simulation and hardware experiments.

## 7.2 Real-time Mapping and Motion Planning

The work presented in Chapter 3 motivated the development of a framework that integrates our proposed motion planning methods with live sensing capabilities and a fast perception framework. In Chapter 4, we sought to address this need. We first explored practical concerns such as whether 'signing' distance fields is beneficial in motion planning tasks, an important implementation issue that we could not find discussed in prior literature, and then investigated three state-of-the-art approaches for generating distance fields in environment reconstruction. We proceeded to use our findings to develop a hybrid system that links our receding-horizon implementation of GPMP2 [8] with a GPU-optimised perception pipeline.

This system constitutes an important milestone in this thesis as it enables GPMP2, a state-of-the-art optimisation based motion planner, to perform reactive whole-body motion planning in dynamic environments online. We validated our approach on a physical Toyota Human Support Robot with live sensor data and accomplished a variety of whole-body motion planning tasks in dynamic environments.

## 7.3 Active Gaze Control

In Chapter 5, we addressed the primary cause of failure in the work presented in Chapter 4 and identified the challenge of *Trajectory-Constrained Active Gaze Control* for collision avoidance. By strategically controlling the pose of a robot's onboard sensors, such as a head camera, we proposed that observations can be optimised to provide more effective collision avoidance and map exploration for a given robot trajectory. Furthermore, we detailed an optimisation-based approach to greedily determine the optimal camera poses throughout a robot's trajectory. Leveraging the GPU-based perception framework described in Chapter 4, we showed that our method outperforms benchmark approaches in both its success rate for generating collision-free trajectories and in map exploration. To further verify our method, we demonstrated its ability to perform in dynamic environments with a physical robot accomplishing a pick-up task. While we provide an early solution to this new research challenge, there are still a number of limitations to be explored in further work, as discussed later in this chapter.

## 7.4 Integrating Mobile Manipulation with Human Trajectory Prediction

In Chapter 6, we saw the development and integration of a number of contributions, culminating in multiple demonstrations of predictive human trajectory avoidance on a physical mobile manipulator across a range of whole-body motion planning tasks.

While in Chapter 3 we hypothesised that parallel processing could be used in conjunction with PCDFs to achieve further performance gains, in this work, we built upon the GPU-optimised framework described in Chapter 4 to perform distance

field compositions on a GPU. Benchmarking our approach against a state-of-the-art parallel EDT algorithm, PBA, we demonstrated that PCDFs significantly outperform when multiple distance fields are required for each environment update loop.

For human trajectory prediction, we identified the need for a new dataset and developed the Oxford Indoor Human Motion dataset – a motion-capture dataset in which people perform natural movements between goals in an indoor environment and are observed using RGB-D camera sensors from both an external and robot perspective.

We proposed a lightweight trajectory optimisation-based approach that uses probabilistic intention recognition to determine a person's intended goal position. Analysis on the THÖR [140] and Oxford-IHM datasets demonstrates our approach to outperform benchmark methods.

Finally, we proposed RHAP-GPMP2, a receding-horizon motion planning framework that incorporates the aforementioned contributions to enable motion planning that accounts for the predicted motion of humans in the robot's workspace. The resultant framework demonstrates the achievement of our thesis goal, enabling a mobile manipulator to perform whole-body tasks while avoiding the predicted trajectory of a human operator. While we would love to say "mission accomplished", there are still many areas to improve in future work on this topic; we detail them in the following section.

## 7.5 Limitations and Future Work

In this section, we detail the limitations that remain with our final presented framework and describe some of the potential avenues for further work.

### 7.5.1 Active Gaze Control

**Smoothness between poses** Particularly in the case of human support robots, robots need to exhibit natural, smooth movements to be well-perceived by humans [167]. A limitation of our method for gaze control is that by greedily determining optimal sensor poses, the resultant movements can appear jerky. Within our approach, we can attempt to alleviate this problem by considering a more limited

range of motion primitives in the optimisation; however, we sacrifice optimality by considering fewer motion primitives. A further research direction for this work would be to explore performing the optimisation over longer time horizons and encouraging *smoothness*. The main concern in pursuing such an approach is the likely increase in computation time, making the gaze controller less reactive to a changing robot trajectory and new observations of the environment. Thus, this remains an interesting avenue for further work in this newly-defined research area.

**Multi-Objective Tuning and Observability Guarantees** Further limitations in our presented approach are the manual *tuning* and the lack of *guarantees* that it provides regarding trajectory observance and collision avoidance. By carefully tuning the rewards ($c_1$, $c_2$, $c_3$), time threshold ($\tau_c$), and distance threshold ($\tau_s$), we can encourage certain types of robot behaviour. In the case of using $c_1 = 0$ and $c_2 = 0$, the sensor will exhibit purely exploratory behaviour; in contrast, by assigning $c_1$ a very large value of reward and using small values for $\tau_c$ and $\tau_s$, the robot will exhibit very conservative behaviour and try to observe voxels immediately in front of its direction of travel. In practice, the behaviour that we want a robot to exhibit is likely to be task-dependent; as such, the parameters used in our framework are likely to vary between tasks. Exploring an automated tuning of parameters was beyond the scope of this work but would be an interesting direction of research for further work.

### 7.5.2 Integrated Motion Planning, Perception and Prediction Framework

As we discussed in Chapter 1, this thesis focuses on the core capability of a robot performing a single action robustly in the face of a changing environment. In reality, we envisage our work being embedded in a more extensive system that performs more complex tasks.

**Local Minima** An inherent limitation of the motion planning part of our proposed framework is that we only optimise a single trajectory. Consequently, the optimisation is prone to local minima, resulting in the robot being unable to find a feasible trajectory. We consider this one of the significant limitations of our

framework in its current form because for robots to be deployed in real applications that interact with or work alongside humans, we require them to avoid collisions reliably. Previous works have sought to address this limitation by optimising for multiple trajectories at a time in different homotopy classes [49, 168], improving the likelihood of obtaining a feasible trajectory. Similar methods could be explored to extend the work presented in this thesis. However, the additional computation associated with optimising multiple trajectories will likely increase planning times and adversely affect the robot's ability to react to environmental changes.

**Semantic Human Trajectory Prediction Cues** In Chapter 6, we described a hybrid human trajectory prediction method that can incorporate learnt information about a person's possible intended goals. While we demonstrated this concept using occupancy analysis on observed motion data, incorporating semantic information was beyond this work's scope. As previous works have shown, scene semantics are likely to provide strong indicators of a person's intentions [123, 169] and could be a promising extension to our work.

**Predicting Articulated Motions** Regarding human motion prediction, we note that our proposed method presented in Chapter 6 focuses solely on predicting a human's position, represented by a 2D or 3D point. This approach is particularly well-suited to tasks involving navigation and in contrast to works that focus on the prediction of articulated movements in collaborative manipulation tasks [152, 153]. In future work, our framework could be adapted to additionally incorporate the predicted motion of articulated joints associated with a human. The underlying mechanism in our approach for dynamic obstacle collision avoidance is the PCDFs that represent each dynamic obstacle's pose and shape. Rather than using a primitive shape, more detailed models, including the predicted poses of individual joints, could be used to represent humans and other dynamic obstacles – there is a large body of work in the dense reconstruction of deformable objects that could be suitable for application here [170–174].

**Variable Mapping Resolution (Technical)** In the final hardware experiments presented in Chapter 6, we maintained large voxelmaps of the environment, each encompassing the entire environment in which the robot operates. These voxelmaps use a fixed resolution of 5 cm, a value that was chosen to balance voxelmap-related computation times with the desire for high-resolution maps for collision-avoidance. While this resolution is sufficient for tasks such as navigation or pick-and-place, higher resolution maps may be desirable for fine-manipulation tasks in more restricted and hazardous environments. In our current implementation, using high-resolution maps would significantly increase computation times and memory requirements.

**Fixed Workspace Perspective (Technical** While the framework we have presented can be applied in unknown environments, it is currently limited to operating within a fixed grid workspace centred on the robot's starting position. Further extensions to our work are necessary to enable an 'infinite' operating workspace, such as a 'moving-window' implementation whereby a robot-centred environment representation is maintained as the robot moves. Usenko et al. [175] use '3D circular buffers' for similar purposes in trajectory re-planning for MAVs, while Kintinuous [94] uses this approach to extend KinectFusion to dense mesh-based mapping over larger-scale environments. While useful for operating in large, multi-room environments, extending our method to use a sliding-window environment representation is likely to incur additional computation cost overheads.

## 7.6 Conclusion

In this thesis, we investigated the use of trajectory optimisation-based motion planning methods for challenging whole-body tasks in dynamic environments. By focusing on the union of perception and motion planning, we developed a practical framework that facilities real-time, GPU-optimised perception combined with fast trajectory optimisation-based motion planning. A challenge for such motion planners has been the prohibitively long time associated with repeatedly computing distance fields in dynamic environments; to this end, we proposed and demonstrated

significant performance gains can be achieved with our proposed method of *predicted composite distance fields*. In developing our integrated framework, we identified the novel problem of *Trajectory-Constrained Active Gaze Control* and proposed a greedy-optimisation based solution that can be applied to robots with movable sensors. In our final development step, we sought to provide a full robot stack that enables a physical robot to anticipate and avoid real people in its workspace. To achieve this, we developed and integrated a human trajectory prediction model that uses a probabilistic framework for determining a person's intended goal position and performs trajectory optimisation-based motion planning to predict the person's future movements. In line with the goal set out at the start of this thesis, we demonstrated our resultant framework on a physical human support robot and performed whole-body motion planning tasks in a variety of challenging environments in the presence of people walking in the robot's workspace.

We embarked on this research journey being guided by application, and it has been incredibly rewarding to see research come to life on a physical robot. We hope that the ideas and contributions presented in this thesis will assist the robotics community at large in further enabling robots to operate in real-world environments alongside humans.

# References


[1] OUTREACH@DARPA.MIL, "Team cerberus and team dynamo win darpa subterranean challenge final event", *DARPA*, Sep. 24, 2021. [Online]. Available: https://www.darpa.mil/news-events/2021-09-24a (visited on 09/24/2021).

[2] S. McDill and S. Orlofsky, "Boston dynamics' backflipping robot shows off new 'parkour' routine", *Reuters*, Aug. 17, 2021. [Online]. Available: https://www.reuters.com/technology/boston-dynamics-backflipping-robot-shows-off-new-parkour-routine-2021-08-17/ (visited on 08/17/2021).

[3] M. N. Finean, W. Merkt, and I. Havoutis, "Predicted Composite Signed-Distance Fields for Real-Time Motion Planning in Dynamic Environments", in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 31, 2020, pp. 616–624. [Online]. Available: https://ojs.aaai.org/index.php/ICAPS/article/view/16010.

[4] M. N. Finean, W. Merkt, and I. Havoutis, "Simultaneous Scene Reconstruction and Whole-Body Motion Planning for Safe Operation in Dynamic Environments", in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Sep. 2021, pp. 3710–3717. [Online]. Available: https://ieeexplore.ieee.org/document/9636860/.

[5] M. N. Finean, W. Merkt, and I. Havoutis, "Where Should I Look? Optimised Gaze Control for Whole-Body Collision Avoidance in Dynamic Environments", *IEEE Robotics and Automation Letters*, vol. 7, pp. 1095–1102, 2 Apr. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9661405/.

[6] M. N. Finean, L. Petrović, W. Merkt, I. Marković, and I. Havoutis, "Motion Planning in Dynamic Environments Using Human Trajectory Prediction", Jan. 2022. arXiv: 2201.05058. [Online]. Available: http://arxiv.org/abs/2201.05058.

[7] T. Yamamoto, T. Nishino, H. Kajima, M. Ohta, and K. Ikeda, "Human support robot (hsr)", in *ACM SIGGRAPH 2018 Emerging Technologies*, ACM, Aug. 2018, pp. 1–2.

[8] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots, "Continuous-time Gaussian process motion planning via probabilistic inference", *The International Journal of Robotics Research*, vol. 37, no. 11, pp. 1319–1340, 2018.

[9] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments", in *ICAPS 2012 - Proceedings of the 22nd International Conference on Automated Planning and Scheduling*, 2012, pp. 207–215.

[10] G. Santayana, *The Life of Reason: The Phases of Human Progress*. Project Gutenberg, 2021, ch. 7, p. 92.

[11] L. Zhang, Y. J. Kim, and D. Manocha, "A hybrid approach for complete motion planning", in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Oct. 2007, pp. 7–14.

[12] O. Andersson, O. Ljungqvist, M. Tiger, D. Axehill, and F. Heintz, "Receding-horizon lattice-based motion planning with dynamic obstacle avoidance", in *2018 IEEE Conference on Decision and Control (CDC)*, IEEE, Dec. 2018, pp. 4467–4474.

[13] E. W. Dijkstra, "A note on two problems in connexion with graphs", *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[14] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths", *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[15] M. Likhachev, G. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality", in *Advances in Neural Information Processing Systems*, 2004.

[16] A. Menon, B. Cohen, and M. Likhachev, "Motion planning for smooth pickup of moving objects", in *Proceedings - IEEE International Conference on Robotics and Automation*, 2014, pp. 453–460.

[17] A. Stentz, "The focussed D* algorithm for real-time replanning", in *14th International Joint Conference on Artificial intelligence (IJCAI)*, 1995.

[18] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain", *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 354–363, 2005.

[19] M. Phillips and M. Likhachev, "SIPP: Safe interval path planning for dynamic environments", in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 5628–5635.

[20] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime dynamic a*: An anytime, replanning algorithm", in *ICAPS 2005 - Proceedings of the 15th International Conference on Automated Planning and Scheduling*, 2005, pp. 262–271.

[21] Y. Yang, W. Merkt, V. Ivan, Z. Li, and V. Vijayakumar, "Hdrm: A resolution complete dynamic roadmap for real-time motion planning in complex scenes", *IEEE Robotics and Automation Letters*, vol. 3, pp. 551–558, 1 Jan. 2018.

[22] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning", *International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[23] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 566–580, 4 1996.

[24] S. Lindemann and S. LaValle, "Incrementally reducing dispersion by increasing voronoi bias in rrts", in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 2004, IEEE, 2004, 3251–3257 Vol.4.

[25] I. Garcia and J. How, "Improving the efficiency of rapidly-exploring random trees using a potential function planner", in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, CDC-ECC '05*, vol. 2005, Jan. 2006, pp. 7965–7970.

[26] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning", *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[27] J. Kuffner and S. LaValle, "Rrt-connect: An efficient approach to single-query path planning", in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, IEEE, 2000, pp. 995–1001.

[28] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library", *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, Dec. 2012.

[29] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives", *International Journal of Humanoid Robotics*, vol. 02, pp. 505–518, 04 Dec. 2005.

[30] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation planning on constraint manifolds", in *2009 IEEE International Conference on Robotics and Automation*, IEEE, May 2009, pp. 625–632.

[31] C. Suh, T. T. Um, B. Kim, H. Noh, M. Kim, and F. C. Park, "Tangent space rrt: A randomized planning algorithm on constraint manifolds", in *2011 IEEE International Conference on Robotics and Automation*, IEEE, May 2011, pp. 4968–4973.

[32] M. Stilman, "Task constrained motion planning in robot joint space", in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Oct. 2007, pp. 3074–3081.

[33] J. van den Berg, D. Ferguson, and J. Kuffner, "Anytime path planning and replanning in dynamic environments", in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, vol. 2006, IEEE, 2006, pp. 2366–2371.

[34] D. Ferguson, N. Kalra, and A. Stentz, "Replanning with rrts", in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, vol. 2006, IEEE, 2006, pp. 1243–1248.

[35] O. Adiyatov and H. A. Varol, "A novel rrt*-based algorithm for motion planning in dynamic environments", in *2017 IEEE International Conference on Mechatronics and Automation (ICMA)*, IEEE, Aug. 2017, pp. 1416–1421.

[36] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking", *International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[37] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization", in *Robotics: Science and Systems*, Citeseer, vol. 9, 2013, pp. 1–10.

[38] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "CHOMP: Covariant Hamiltonian optimization for motion planning", *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, Aug. 2013.

[39] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning", in *2009 IEEE International Conference on Robotics and Automation*, IEEE, May 2009, pp. 489–494.

[40] A. Byravan, B. Boots, S. S. Srinivasa, and D. Fox, "Space-Time functional gradient optimization for motion planning", *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 6499–6506, 2014.

[41] K. He, E. Martin, and M. Zucker, "Multigrid CHOMP with Local Smoothing", in *IEEE-RAS International Conference on Humanoid Robots*, vol. 2015-Febru, 2015, pp. 315–322.

[42] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning", in *2011 IEEE International Conference on Robotics and Automation*, IEEE, May 2011, pp. 4569–4574.

[43] M. Elbanhawi, M. Simic, and R. Jazar, "Randomized bidirectional b-spline parameterization motion planning", *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 406–419, 2016.

[44] Z. Marinho, B. Boots, A. Dragan, A. Byravan, G. J. Gordon, and S. Srinivasa, "Functional gradient motion planning in reproducing kernel hilbert spaces", in *Robotics: Science and Systems*, vol. 12, 2016.

[45] M. Mukadam, X. Yan, and B. Boots, "Gaussian Process Motion planning", in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2016, pp. 9–15.

[46] M. Mukadam, J. Dong, F. Dellaert, and B. Boots, "STEAP: simultaneous trajectory estimation and planning", *Autonomous Robots*, vol. 43, no. 2, pp. 415–434, 2019.

[47] E. Huang, M. Mukadam, Z. Liu, and B. Boots, "Motion planning with graph-based trajectories and Gaussian process inference", in *Proceedings - IEEE International Conference on Robotics and Automation*, 2017, pp. 5591–5598.

[48] K. V. Alwala and M. Mukadam, "Joint sampling and trajectory optimization over graphs for online motion planning", *arXiv*, Nov. 2020. [Online]. Available: `http://arxiv.org/abs/2011.07171`.

[49] K. Kolur, S. Chintalapudi, B. Boots, and M. Mukadam, "Online motion planning over multiple homotopy classes with gaussian process inference", in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Nov. 2019, pp. 2358–2364.

[50] L. Petrovic, J. Persic, M. Seder, and I. Markovic, "Stochastic optimization for trajectory planning with heteroscedastic gaussian processes", in *2019 European Conference on Mobile Robots (ECMR)*, IEEE, Sep. 2019, pp. 1–6.

[51] G. Ye and R. Alterovitz, "Demonstration-guided motion planning", in *Robotics Research : The 15th International Symposium ISRR*, H. I. Christensen and O. Khatib, Eds. Cham: Springer International Publishing, 2017, pp. 291–307.

[52] A. H. Qureshi and M. C. Yip, "Deeply informed neural sampling for robot motion planning", in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Oct. 2018, pp. 6582–6588.

[53] A. H. Qureshi, A. Simeonov, M. J. Bency, and M. C. Yip, "Motion planning networks", in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 2118–2124.

[54] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey", *Journal of Machine Learning Research*, vol. 10, pp. 1633–1685, 2009.

[55] N. Jetchev and M. Toussaint, "Fast motion planning from experience: Trajectory prediction for speeding up movement generation", *Autonomous Robots*, vol. 34, pp. 111–127, 1-2 Jan. 2013.

[56] O. Saha, P. Dasgupta, and B. Woosley, "Real-time robot path planning from simple to complex obstacle patterns via transfer learning of options", *Autonomous Robots*, vol. 43, pp. 2071–2093, 8 Dec. 2019.

[57] J. Dong, M. Mukadam, F. Dellaert, and B. Boots, "Motion planning as probabilistic inference using gaussian processes and factor graphs", in *Robotics: Science and Systems XII*, Robotics: Science and Systems Foundation, Jun. 2016.

[58] S. Anderson, T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch nonlinear continuous-time trajectory estimation as exactly sparse Gaussian process regression", *Autonomous Robots*, vol. 39, no. 3, pp. 221–238, 2015.

[59] M. Mukadam, "Structured Learning and Inference for Robot Motion Generation", PhD thesis, Georgia Institute of Technology, Jul. 2019.

[60] M. Kaess, A. Ranganathan, and F. Dellaert, "Isam: Incremental smoothing and mapping", *IEEE Transactions on Robotics*, vol. 24, pp. 1365–1378, 6 Dec. 2008.

[61] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "Isam2: Incremental smoothing and mapping using the bayes tree", *The International Journal of Robotics Research*, vol. 31, pp. 216–235, 2 Feb. 2012.

[62] M. Toussaint, "Robot trajectory optimization using approximate inference", in *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, vol. 382, ACM Press, 2009, pp. 1–8.

[63] T. P. Minka, "Expectation propagation for approximate bayesian inference", in *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, ser. UAI '01, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 362–369.

[64] F. Dellaert and M. Kaess, "Square root sam: Simultaneous localization and mapping via square root information smoothing", *The International Journal of Robotics Research*, vol. 25, pp. 1181–1203, 12 Dec. 2006.

[65] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm", *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[66] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction.", *Technical Report*, pp. 1–27, GT-RIM-CP&R-2012-002.

[67] M. Herbert, C. Caillas, E. Krotkov, I. Kweon, and T. Kanade, "Terrain mapping for a roving planetary explorer", in *Proceedings, 1989 International Conference on Robotics and Automation*, IEEE Comput. Soc. Press, 1989, pp. 997–1002.

[68] R. Triebel, P. Pfaff, and W. Burgard, "Multi-level surface maps for outdoor terrain mapping and loop closing", in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Oct. 2006, pp. 2276–2282.

[69] Y. Roth-Tabak and R. Jain, "Building an environment model using depth information", *Computer*, vol. 22, pp. 85–90, 6 Jun. 1989.

[70] H. Moravec, "Robot spatial perception by stereoscopic vision and 3d evidence grids", CMU Robotics Institute, Daimler Benz Research, Tech. Rep., 1996.

[71] A. Elfes and L. Matthies, "Sensor integration for robot navigation: Combining sonar and stereo range data in a grid-based representataion", in *26th IEEE Conference on Decision and Control*, IEEE, Dec. 1987, pp. 1802–1807.

[72] A. Elfes, "Using occupancy grids for mobile robot perception and navigation", *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[73] D. Fox, W. Burgard, S. Thrun, and A. Cremers, "A hybrid collision avoidance method for mobile robots", in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, vol. 2, 1998, 1238–1243 vol.2.

[74] A. Eliazar and R. Parr, "Dp-slam 2.0", in *IEEE International Conference on Robotics and Automation*, vol. 2004, IEEE, 2004, 1314–1320 Vol.2.

[75] J. Velagic, B. Lacevic, and N. Osmic, "Efficient path planning algorithm for mobile robot navigation with a local minima problem solving", in *2006 IEEE International Conference on Industrial Technology*, 2006, pp. 2325–2330.

[76] J. M. Santos, D. Portugal, and R. P. Rocha, "An evaluation of 2d slam techniques available in robot operating system", in *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, IEEE, Oct. 2013, pp. 1–6.

[77] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees", *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, Apr. 2013.

[78] D. Meagher, "Geometric modeling using octree encoding", *Computer Graphics and Image Processing*, vol. 19, pp. 129–147, 2 Jun. 1982.

[79] P. Sodhi, B.-J. Ho, and M. Kaess, "Online and consistent occupancy grid mapping for planning in unknown environments", in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Nov. 2019, pp. 7879–7886.

[80] E. Vespa, N. Nikolov, M. Grimm, L. Nardi, P. H. J. Kelly, and S. Leutenegger, "Efficient octree-based volumetric slam supporting signed-distance and occupancy mapping", *IEEE Robotics and Automation Letters*, vol. 3, pp. 1144–1151, 2 Apr. 2018.

[81] E. Vespa, N. Funk, P. H. J. Kelly, and S. Leutenegger, "Adaptive-resolution octree-based volumetric slam", in *2019 International Conference on 3D Vision (3DV)*, IEEE, Sep. 2019, pp. 654–662.

[82] M. Niesner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3D reconstruction at scale using voxel hashing", *ACM Transactions on Graphics*, vol. 32, no. 6, 2013.

[83] S. Lefebvre and H. Hoppe, "Perfect spatial hashing", *ACM Transactions on Graphics*, vol. 25, pp. 579–588, 3 Jul. 2006.

[84] D. A. Alcantara, A. Sharf, F. Abbasinejad, S. Sengupta, M. Mitzenmacher, J. D. Owens, and N. Amenta, "Real-time parallel hashing on the gpu", *ACM Transactions on Graphics*, vol. 28, pp. 1–9, 5 Dec. 2009.

[85] D. R. Canelhas, T. Stoyanov, and A. J. Lilienthal, "A survey of voxel interpolation methods and an evaluation of their impact on volumetric map-based visual odometry", in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2018, pp. 3637–3643.

[86] M. Sramek, "Fast surface rendering from raster data by voxel traversal using chessboard distance", in *Proceedings Visualization '94*, IEEE Comput. Soc. Press, 1994, pp. 188–195.

[87] S. Gibson, "Using distance maps for accurate surface representation in sampled volumes", in *IEEE Symposium on Volume Visualization*, IEEE, 1999, pp. 23–30.

[88] S. F. Frisken, R. N. Perry, A. P. Rockwood, and T. R. Jones, "Adaptively sampled distance fields", in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*, ACM Press, 2000, pp. 249–254.

[89] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm", in *Proceedings of the 14th annual conference on Computer graphics and interactive techniques - SIGGRAPH '87*, vol. 21, ACM Press, Aug. 1987, pp. 163–169.

[90] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots", *The International Journal of Robotics Research*, vol. 5, pp. 90–98, 1 Mar. 1986.

[91] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots", in *IEEE International Conference on Intelligent Robots and Systems*, 2019, pp. 4423–4430.

[92] B. Curless and M. Levoy, "A volumetric method for building complex models from range images", in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96*, ACM Press, 1996, pp. 303–312.

[93] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking", in *2011 10th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2011*, IEEE, Oct. 2011, pp. 127–136.

[94] T. Whelan, M. Kaess, and M. Fallon, "Kintinuous: Spatially extended kinectfusion", *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, p. 7, 2012.

[95] R. Wagner, U. Frese, and B. Bauml, "3d modeling, distance and gradient computation for motion planning: A direct gpgpu approach", in *2013 IEEE International Conference on Robotics and Automation*, IEEE, May 2013, pp. 3586–3592.

[96] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning", in *IEEE International Conference on Intelligent Robots and Systems*, vol. 2017-September, 2017, pp. 1366–1373.

[97] D. Kappler, F. Meier, J. Issac, J. Mainprice, C. G. Cifuentes, M. Wuthrich, V. Berenz, S. Schaal, N. Ratliff, and J. Bohg, "Real-time perception meets reactive motion generation", *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1864–1871, Mar. 2018.

[98] R. B. Rusu, I. A. Şucan, B. Gerkey, S. Chitta, M. Beetz, and L. E. Kavraki, "Real-time perception-guided motion planning for a personal robot", in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, 2009, pp. 4245–4252.

[99] L. P. Kaelbling and T. Lozano-Pérez, "Integrated task and motion planning in belief space", *The International Journal of Robotics Research*, vol. 32, pp. 1194–1227, 9-10 Aug. 2013.

[100] D. Maier, A. Hornung, and M. Bennewitz, "Real-time navigation in 3D environments based on depth camera data", in *IEEE-RAS International Conference on Humanoid Robots*, 2012, pp. 692–697.

[101] W. Merkt, Y. Yang, T. Stouraitis, C. E. Mower, M. Fallon, and S. Vijayakumar, "Robust shared autonomy for mobile manipulation with continuous scene monitoring", in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, vol. 2017-August, IEEE, Aug. 2017, pp. 130–137.

[102] V. Ivan, Y. Yang, W. Merkt, M. P. Camilleri, and S. Vijayakumar, "Exotica: An extensible optimization toolset for prototyping and benchmarking motion planning and control", *Studies in Computational Intelligence*, vol. 778, pp. 211–240, 2019.

[103] F. Husain, A. Colome, B. Dellen, G. Alenya, and C. Torras, "Realtime tracking and grasping of a moving object from range video", in *Proceedings - IEEE International Conference on Robotics and Automation*, IEEE, 2014, pp. 2617–2622.

[104] A. Hermann, S. Klemm, Z. Xue, A. Roennau, and R. Dillmann, "Gpu-based real-time collision detection for motion execution in mobile manipulation planning", in *2013 16th International Conference on Advanced Robotics (ICAR)*, IEEE, Nov. 2013, pp. 1–7.

[105] A. Hermann, F. Drews, J. Bauer, S. Klemm, A. Roennau, and R. Dillmann, "Unified gpu voxel collision detection for mobile manipulation planning", in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Sep. 2014, pp. 4154–4160.

[106] A. Hermann, J. Bauer, S. Klemm, and R. Dillmann, "Mobile manipulation planning optimized for gpgpu voxel-collision detection in high resolution live 3d-maps", in *ISR/Robotik*, 2014, pp. 1–8.

[107] C. Juelg, A. Hermann, A. Roennau, and R. Dillmann, "Fast online collision avoidance for mobile service robots through potential fields on 3d environment data processed on gpus", in *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, vol. 2018-January, IEEE, Dec. 2017, pp. 921–928.

[108] T. T. Cao, K. Tang, A. Mohamed, and T. S. Tan, "Parallel Banding Algorithm to compute exact distance transform with the GPU", in *ACM SIGGRAPH I3D*, 2010, pp. 83–90.

[109] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey", *The International Journal of Robotics Research*, vol. 39, pp. 895–935, 8 Jul. 2020.

[110] E. Rehder, F. Wirth, M. Lauer, and C. Stiller, "Pedestrian prediction by planning using deep neural networks", in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2018, pp. 1–5.

[111] C. Anderson, R. Vasudevan, and M. Johnson-Roberson, "Off the beaten sidewalk: Pedestrian prediction in shared spaces for autonomous vehicles", *IEEE Robotics and Automation Letters*, vol. 5, pp. 6892–6899, 4 Jun. 2020.

[112] D. Ridel, E. Rehder, M. Lauer, C. Stiller, and D. Wolf, "A literature review on the prediction of pedestrian behavior in urban scenarios", in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, Nov. 2018, pp. 3105–3112.

[113] B. T. Morris and M. M. Trivedi, "Understanding vehicular traffic behavior from video: A survey of unsupervised approaches", *Journal of Electronic Imaging*, vol. 22, p. 041 113, 4 Sep. 2013.

[114] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles", *ROBOMECH Journal*, vol. 1, pp. 1–14, 1 Dec. 2014.

[115] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, P. Jennings, and A. Mouzakitis, "Deep learning-based vehicle behavior prediction for autonomous driving applications: A review", *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, pp. 33–47, 1 Jan. 2022.

[116] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll, "What the constant velocity model can teach us about pedestrian motion prediction", *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1696–1703, 2020.

[117] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics", *Physical Review E*, vol. 51, pp. 4282–4286, 5 May 1995.

[118] J. Elfring, R. van de Molengraft, and M. Steinbuch, "Learning intentions for improved human motion prediction", *Robotics and Autonomous Systems*, vol. 62, pp. 591–602, 4 Apr. 2014.

[119] G. Ferrer and A. Sanfeliu, "Behavior estimation for a complete framework for human motion prediction in crowded environments", in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2014, pp. 5940–5945.

[120] I. Karamouzas, P. Heil, P. van Beek, and M. H. Overmars, "A predictive collision avoidance model for pedestrian simulation", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5884 LNCS, pp. 41–52, 2009.

[121] F. Zanlungo, T. Ikeda, and T. Kanda, "Social force model with explicit collision prediction", *EPL (Europhysics Letters)*, vol. 93, p. 68 005, 6 Mar. 2011.

[122] M. Luber, J. A. Stork, G. D. Tipaldi, and K. O. Arras, "People tracking with human motion predictions from social forces", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 464–469.

[123] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert, "Activity forecasting", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7575 LNCS, pp. 201–214, PART 4 2012.

[124] L. Ballan, F. Castaldo, A. Alahi, F. Palmieri, and S. Savarese, "Knowledge transfer for scene-specific motion prediction", in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham: Springer International Publishing, 2016, pp. 697–713.

[125] A. Treuille, S. Cooper, and Z. Popović, "Continuum crowds", *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 1160–1168, 2006.

[126] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking", in *2009 IEEE 12th International Conference on Computer Vision*, IEEE, 2009, pp. 261–268.

[127] J. Sun, Q. Jiang, and C. Lu, "Recursive social behavior graph for trajectory prediction", in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2020, pp. 657–666.

[128] P. Ghosh, J. Song, E. Aksan, and O. Hilliges, "Learning human motion models for long-term predictions", in *2017 International Conference on 3D Vision (3DV)*, IEEE, 2017, pp. 458–466.

[129] F. Bartoli, G. Lisanti, L. Ballan, and A. Del Bimbo, "Context-aware trajectory prediction", in *2018 24th International Conference on Pattern Recognition (ICPR)*, IEEE, 2018, pp. 1941–1946.

[130] C. Yu, X. Ma, J. Ren, H. Zhao, and S. Yi, "Spatio-temporal graph transformer networks for pedestrian trajectory prediction", in *European Conference on Computer Vision*, Springer, 2020, pp. 507–523.

[131] P. Kratzer, M. Toussaint, and J. Mainprice, "Prediction of human full-body movements with motion optimization and recurrent neural networks", in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 1792–1798.

[132] Z. Cao, H. Gao, K. Mangalam, Q.-Z. Cai, M. Vo, and J. Malik, "Long-term human motion prediction with scene context", pp. 387–404, Jul. 2020.

[133] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset", *The International Journal of Robotics Research*, vol. 32, pp. 1231–1237, 11 Sep. 2013.

[134] M.-F. Chang, D. Ramanan, J. Hays, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, and S. Lucey, "Argoverse: 3d tracking and forecasting with rich maps", in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2019, pp. 8740–8749.

[135] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "Nuscenes: A multimodal dataset for autonomous driving", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020.

[136] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark", in *2009 IEEE conference on computer vision and pattern recognition*, IEEE, 2009, pp. 304–311.

[137] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection", in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, IEEE, 2005, pp. 886–893.

[138] A. Ess, B. Leibe, K. Schindler, and L. V. Gool, "A mobile vision system for robust multi-person tracking", in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2008, pp. 1–8.

[139] M. Wulfmeier, D. Z. Wang, and I. Posner, "Watch this: Scalable cost-function learning for path planning in urban environments", in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 2016-November, IEEE, Oct. 2016, pp. 2089–2095.

[140] A. Rudenko, T. P. Kucner, C. S. Swaminathan, R. T. Chadalavada, K. O. Arras, and A. J. Lilienthal, "Thör: Human-robot navigation data collection and accurate motion trajectories dataset", *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 676–682, 2020.

[141] P. Kratzer, S. Bihlmaier, N. B. Midlagajni, R. Prakash, M. Toussaint, and J. Mainprice, "Mogaze: A dataset of full-body motions that includes workspace geometry and eye-gaze", *IEEE Robotics and Automation Letters*, vol. 6, pp. 367–373, 2 Apr. 2021.

[142] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, "Planning-based prediction for pedestrians", in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Oct. 2009, pp. 3931–3936.

[143] D. Vasquez, "Novel planning-based algorithms for human motion prediction", in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016, pp. 3317–3322.

[144] C. Rösmann, M. Oeljeklaus, F. Hoffmann, and T. Bertram, "Online trajectory prediction and planning for social robot navigation", in *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, IEEE, 2017, pp. 1255–1260.

[145] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert, "Activity forecasting", in *European conference on computer vision*, Springer, 2012, pp. 201–214.

[146] A. Rudenko, L. Palmieri, and K. O. Arras, "Joint long-term prediction of human motion using a planning-based social force approach", in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 4571–4577.

[147] S. V. Albrecht, C. Brewitt, J. Wilhelm, B. Gyevnar, F. Eiras, M. Dobre, and S. Ramamoorthy, "Interpretable goal-based prediction and planning for autonomous driving", in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2021, pp. 1043–1049.

[148] C. L. Baker, R. Saxe, and J. B. Tenenbaum, "Action understanding as inverse planning", *Cognition*, vol. 113, pp. 329–349, 3 Dec. 2009.

[149] F. Eiras, M. Hawasly, S. V. Albrecht, and S. Ramamoorthy, "A two-stage optimization-based motion planner for safe urban driving", *IEEE Transactions on Robotics*, vol. 38, pp. 822–834, 2 Apr. 2022.

[150] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey", *Robotics and Autonomous Systems*, vol. 61, pp. 1726–1743, 12 Dec. 2013.

[151] R. Möller, A. Furnari, S. Battiato, A. Härmä, and G. M. Farinella, "A survey on human-aware robot navigation", *Robotics and Autonomous Systems*, vol. 145, p. 103 837, Nov. 2021.

[152] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion", in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Nov. 2013, pp. 299–306.

[153] J. S. Park, C. Park, and D. Manocha, "I-planner: Intention-aware motion planning using learning-based human motion prediction", *The International Journal of Robotics Research*, vol. 38, pp. 23–39, 1 Jan. 2019.

[154] C. Connolly, "The determination of next best views", in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2, Institute of Electrical and Electronics Engineers, 1985, pp. 432–435.

[155] R. Border, J. D. Gammell, and P. Newman, "Surface edge explorer (see): Planning next best views directly from 3d observations", in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2018, pp. 1–8.

[156] M. Krainin, B. Curless, and D. Fox, "Autonomous generation of complete 3d object models using next best view manipulation planning", in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 5031–5037.

[157] C. Collander, W. J. Beksi, and M. Huber, "Learning the next best view for 3d point clouds via topological features", in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2021, pp. 12 207–12 213.

[158] C. Potthast and G. S. Sukhatme, "A probabilistic framework for next best view estimation in a cluttered environment", *Journal of Visual Communication and Image Repr.*, vol. 25, no. 1, pp. 148–164, 2014.

[159] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon next-best-view planner for 3D exploration", in *Proceedings - IEEE International Conference on Robotics and Automation*, 2016, pp. 1462–1468.

[160] N. Smith, N. Moehrle, M. Goesele, and W. Heidrich, "Aerial path planning for urban scene reconstruction", *ACM Transactions on Graphics*, vol. 37, pp. 1–15, 6 Dec. 2018.

[161] J. Nunez-Varela, B. Ravindran, and J. L. Wyatt, "Where do i look now? gaze allocation during visually guided manipulation", in *2012 IEEE International Conference on Robotics and Automation*, IEEE, May 2012, pp. 4444–4449.

[162] D. Fox, W. Burgard, and S. Thrun, "Active markov localization for mobile robots", *Robotics and Autonomous Systems*, vol. 25, pp. 195–207, 3-4 Nov. 1998.

[163] G. Costante, C. Forster, J. Delmerico, P. Valigi, and D. Scaramuzza, "Perception-aware path planning", *arXiv preprint arXiv:1605.04151*, Jan. 2016. arXiv: `1605.04151`. [Online]. Available: `http://arxiv.org/abs/1605.04151`.

[164] G. Lidoris, K. Kühnlenz, D. Wollherr, and M. Buss, "Combined trajectory planning and gaze direction control for robotic exploration", in *Proceedings - IEEE International Conference on Robotics and Automation*, 2007, pp. 4044–4049.

[165] J. F. Seara, K. H. Strobl, and G. Schmidt, "Information management for gaze control in vision guided biped walking", *IEEE International Conference on Intelligent Robots and Systems*, vol. 1, pp. 31–36, 2002.

[166] D. Droeschel, D. Holz, J. Stückler, and S. Behnke, "Using time-of-flight cameras with active gaze control for 3d collision avoidance", in *2010 IEEE International Conference on Robotics and Automation*, IEEE, May 2010, pp. 4035–4040.

[167] M. Shidujaman, S. Zhang, R. Elder, and H. Mi, "'roboquin": A mannequin robot with natural humanoid movements", in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, IEEE, Aug. 2018, pp. 1051–1056.

[168] W. X. Merkt, V. Ivan, T. Dinev, I. Havoutis, and S. Vijayakumar, "Memory clustering using persistent homology for multimodality- and discontinuity-sensitive learning of optimal control warm-starts", *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1649–1660, 2021.

[169] A. Syed and B. T. Morris, "Sseg-lstm: Semantic scene segmentation for trajectory prediction", IEEE, Jun. 2019, pp. 2504–2509.

[170] R. D. Charette and S. Manitsaris, "3d reconstruction of deformable revolving object under heavy hand interaction", arXiv: 1908.01523. [Online]. Available: http://arxiv.org/abs/1908.01523.

[171] M. Dou, J. Taylor, H. Fuchs, A. Fitzgibbon, and S. Izadi, "3d scanning deformable objects with a single rgbd sensor", in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 07-12-June-2015, IEEE, Jun. 2015, pp. 493–501.

[172] H. Kawasaki, R. Furukawa, R. Sagawa, and Y. Yagi, "Dynamic scene shape reconstruction using a single structured light pattern", in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2008, pp. 1–8.

[173] A. Jordt and R. Koch, "Fast tracking of deformable objects in depth and colour video", in *Proceedings of the British Machine Vision Conference 2011*, British Machine Vision Association, 2011, pp. 114.1–114.11.

[174] R. Yu, C. Russell, N. D. F. Campbell, and L. Agapito, "Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video", in *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Dec. 2015, pp. 918–926.

[175] V. Usenko, L. von Stumberg, A. Pangercic, and D. Cremers, "Real-time trajectory replanning for mavs using uniform b-splines and a 3d circular buffer", in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 2017-Septe, IEEE, Sep. 2017, pp. 215–222.