Informative Path Planning for Search and Rescue Using a UAV

Ajith Anil Meera

Technische Universiteit Delft

TU Delft Delft University of Technology

Challenge the future

INFORMATIVE PATH PLANNING FOR SEARCH AND RESCUE USING A UAV

by

Ajith Anil Meera

in partial fulfillment of the requirements for the degree of

Master of Science
in Mechanical Engineering

at the Delft University of Technology,
to be defended publicly on Thursday August 23, 2018 at 14:00 CEST.

Supervisor:	Prof. Dr. Roland Siegwart	ETH Zurich
	Marija Popovic	ETH Zurich
	Alexander Millane	ETH Zurich
Thesis committee:	Prof. dr. ir. Martijn Wisse	TU Delft
	Dr. Wei Pan	TU Delft
	Dr. Javier Alonso Mora	TU Delft
	Dr. Peyman Mohajerin Esfahani	TU Delft

This thesis is confidential and cannot be made public until February 23, 2020.

An electronic version of this thesis is available at http://repository.tudelft.nl/.

ETH Zürich
Autonomous Systems Lab

TU Delft Delft University of Technology

ABSTRACT

Target search in an obstacle filled environment is a practically relevant challenge in robotics that has a huge impact in the society. The wide range of applications include searching for victims in a search and rescue operation, detecting weeds in precision agriculture, patrolling borders for military and navy, automated census of endangered species in a forest etc. An efficient target search algorithm provides a data acquisition platform with least human intervention, thus improving the quality of life of humans. This thesis aims at introducing a general path planning algorithm for UAVs flying at different heights in an obstacle filled environment, searching for targets in the ground field. An adaptive informative path planning (IPP) algorithm is introduced that simultaneously trade off between area coverage, field of view, height dependent sensor performance and obstacle avoidance. It plans under uncertainties in the sensor measurements at varying heights, and is robust against wrong target detections. It generates an optimal fixed horizon plan in the form of a 3D minimum-snap trajectory that maximizes the information gain in minimum flight time by providing maximum area coverage, without any collision with the obstacles. The resulting planner is modular in terms of the mapping strategy, environment complexity, different target, changes in the sensor model and optimizer used. The planner is tested against varying environmental complexities, demonstrating its capability in handling a wide range of possible environments. The planner outperforms other planners like non-adaptive IPP planner, coverage planner and random sampling planner, by demonstrating the fastest decrease in map error while flying for a fixed time budget. A proof of concept for the algorithm is provided through real experiments by running the algorithm on a UAV flying inside a lab environment, searching for targets lying on the ground. All the targets were successfully found and mapped by the algorithm, demonstrating its applicability in a real-life target search problem.

Ajith Anil Meera
Delft, August 2018

Contents

1

INTRODUCTION

This chapter aims at providing an introduction to the target search problem in a search and rescue operation. Motivated by the huge impact that an efficient target search algorithm can make on the society, the requirements of the algorithm are defined along with the problem statement for the thesis. This chapter lays the theoretical foundation for the approach followed in the thesis.

MOTIVATION

The increasing impact of natural and man-made disasters on the urban population has led to a growing demand for efficient search and rescue operations that can search a calamity zone and find all the victims in the least amount of time. However, the random nature of these disasters makes the search and rescue operation a challenging problem. In many cases, the size of the affected area is so large that finding all the victims within a given amount of time is not possible for the rescue team. Moreover, most of the calamity regions are dangerous for humans to directly intervene. These limitations have given way to robotic systems that now act as an extension for the search and rescue team to reach out to locations inaccessible to humans at a much faster rate. These systems include snake-like robots like *IRS Soryu*, crawler robots like *Terminatorbot* [1], hexapod robots like *RHex* [2], Unmanned Aerial Vehicles (UAVs) etc. They have successfully been used even in large scale disasters like the terrorist attack on the World Trade Centre, hurricane Katrina, etc [3]. However, the control and inspection tasks using these systems are done manually by humans, limiting their widespread use by emergency response teams worldwide as a standard protocol. Making these systems autonomous and reliable would decrease human intervention, thus improving the efficiency of the workforce. Rescue operations are a race against time, and each saved minute could be exploited to save more victims. Therefore, the research towards autonomous search and rescue operations is vital.

According to the 2005 World Disaster report [4], over 900,000 people have been killed from 1995 to 2004 by urban disasters. Almost 80% of the survivors of urban disaster are surface victims [5], who lie on the surface of the rubble and can easily be observed by a aerial vehicles flying over the disaster site. Therefore, deploying a UAV to autonomously scan a disaster area would increase the quality of the rescue operation. With the advent of portable fast computing machines, UAVs are now capable of running more complex algorithms. The agile manoeuvering capabilities along with its rapid scanning capabilities makes the UAV one of the best and fastest data acquisition platforms available. Therefore, a UAV based initial search is the best way to find most number of victims in a disaster site, in a minimum amount of time.

A UAV based search operation is constrained with a low flight time due to battery restrictions. Moreover, the UAV should be capable of ensuring its own safety during the flight. The lightweight and fragile structure prone to destruction via collisions, the low endurance or performance in tough environments like proximity to a burning buliding, etc, pose serious problems for its operation on a disaster site. Therefore, an algorithm that not only scans for humans in an area, but also incoorporates battery restrictions, collision avoidance and risk avoidance, is essential for safe performance.

The search and rescue algorithm for a UAV can be formulated as a constrained optimization problem where the objective is to maximize the target detection, given the budget constraints for flight time or battery power. The output of the algorithm should be an optimal path for the UAV to fly and take measurements. This motivates the problem to be an Informative Path Planning (IPP) problem, which is discussed in the next

section.

INFORMATIVE PATH PLANNING

The IPP problem seeks to maximize the information gathered from the surroundings, subjected to a mobility-related budget constraint. The IPP problem can be formulated as:

$$P^* = \underset{P \in \psi}{\arg\max}\ O[\text{MEASURE}(P)], \tag{1.1}$$

$$s.t.\ cost(P) \leq B,$$

where P is the path out of all possible paths ψ, O defines the information objective function or the information quality of the data collected, $measure(P)$ obtains the discrete measurement along the path P, $cost(P)$ provides the corresponding travel cost (e.g. time, fuel, or energy) and B is the cost budget [6]. IPP aims at choosing an optimal information-rich path within a path budget so as to cover the target area. For a search and rescue scenario, the objective function would represent the measure of the quality of information denoting the presence or absence of a victim in the environment.

A clear distinction exists between the objectives of IPP, shortest-distance path planning [7] and coverage path planning [8]. While coverage path planning aims at covering the target area completely [8], IPP aims at choosing an optimal information-rich path within a path budget so as to cover the target area. Unlike the shortest-distance path planning problems, IPP is not goal-directed and typically chooses a path with larger path length. Shortest path planning problems can make use of algorithms like Dijkstra's algorithm because of the modularity of the objective function (usually Euclidean distance). This means that the sum of the objective value of two segments is equal to the objective value of the concatenation of the two segments. Many information theoretic objective functions exhibit submodularity - intuitively, the property of diminishing returns [9]. This encodes the property that the measurements are less informative if other measurements were taken close-by. Therefore, the information gathered in the future is strongly dependent on the prior trajectories of the robot, which is not the case with modular objectives [10]. This property differentiates IPP from other path planning algorithms in terms of the objective function.

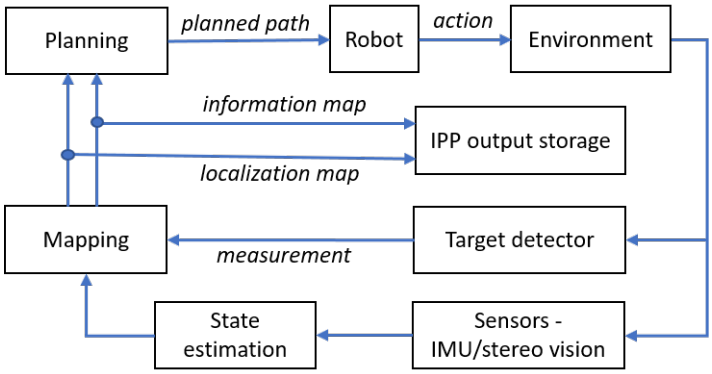


Figure 1.1: System architecture for informative path planning.

The nature of the gathered information can range from phenomenon monitoring like temperature, pressure, algae growth etc, to application specific searches like search for victims, military surveillance etc. The information thus gained could be of critical use to the community in terms of research, industrial application or social benefits. The information is represented in the form of an information map, which is the resulting output of an IPP planner. A system architecture of an IPP planner is shown in Figure 1.1. The target detector collects information from the environment, identifies the target, and fuses it with the map, depending on the UAV position. The planner keeps track of two maps, one for localization and the other one for information storage. These maps are used by the planner in deciding the next best path in the environment that can maximize the information gain and improve the quality of the map. The robot moves along the planned path and take measurements during its flight at a measurement frequency. The objective of the planner is to generate the UAV path that would eventually result in a map that is close to the ground truth. The error in the information map would be higher if the measurement uncertainties of the target sensor is not considered while mapping and planning. Therefore, it is imperative that the measurement uncertainty is modelled accurately.

MEASUREMENT UNCERTAINTY

The measurement outcome from the target sensor could be of two types depending on the phenomenon being observed. If the phenomenon being observed only results in discrete measurements like the presence or absence of a human at a location in the environment, then the outcome can be treated as a discrete random variable. Similarly, if the phenomenon being observed takes a continuous value like temperature, pressure or radiation measurements, then the outcome can be treated as a continuous random variable. The measurement uncertainty related to a continuous random variable can be expressed as the variance of the normal distribution by approximating the sensor to follow a Gaussian noise model. The discrete random variable for a classifier based target sensor can be approximated to follow a Bernoulli distribution. A Bernoulli distribution is a discrete probability distribution which takes the value 1 with probability p and the value 0 with the probability $1-p$. The expected value of a Bernoulli random variable X is given by $E[X] = p$, while the variance representing the uncertainty is given by $Var[X] = p(1-p)$. Uncertainties related to the measurement outcome should be incorporated into the mapping and planning strategies depending on the nature of the target sensor used.

REQUIREMENTS

This thesis aims to introduce a general 3D path planning algorithm for target search using a UAV that can search, map and plan in an obstacle filled environment with uncertain measurements from the sensor. The algorithm should meet the following requirements:

- Provide complete coverage of an area,
- Find all the targets in the field,
- Adhere to the time budget constraint of the UAV,
- Plan a dynamically stable 3D path,
- Ensure collision avoidance with the obstacles,
- Avoid high risk areas in the environment.

CONTRIBUTIONS

The main contributions of the thesis include:

- The introduction of a modular 3D path planning algorithm that solves a target search problem using UAV in an obstacle filled environment by accounting for measurement uncertainties.
- The introduction of a layered optimization approach for adaptive IPP, that optimizes the search space using Bayesian optimization, and optimizes the UAV path using standard optimizers for maximal information gain. The adaptive planner is robust against wrong detections and is beneficial for a target search problem.

This chapter dealt with introducing the target search problem in a search and rescue operation, motivated by the huge impact that the algorithm can make in the society. The IPP scheme that cater to the requirements of the algorithm was summarized. The next chapter provides a brief introduction to different types of IPP methods found in literature, and compares them against the method followed in this thesis.

2

RELATED WORK

IPP has been used for solving a wide range of real life problems including environment monitoring [11–13], exploration [14–16] and sensor placement problem [17, 18]. While in the operations research community, a discrete version of IPP is known as Orienteering Problem [19] and Travelling Salesman Problem [20], in robotics community it was a subset of a large class of active perception [21] problems. With the advancements in fast computing machines and portable on-board processing devices, IPP literature has seen an exponential growth in academia. These advancements have enabled the use of IPP in a continuous domain to plan paths for a robot that gathers data from the environment. IPP can be broadly classified based on three criteria: adaptiveness, greediness (myopic property) and continuity. An IPP method could be adaptive or non-adaptive, myopic or non-myopic, and continuous or discrete. The IPP algorithm introduced in this thesis is adaptive, non-myopic and continuous. The following sections discuss each classification and contrast each of it against the approach followed in this thesis, demonstrating that the algorithm is the most suitable IPP algorithm for target search using UAV in a 3D environment.

ADAPTIVE AND NON-ADAPTIVE IPP

The IPP algorithm can be simplified by assuming the independence between the information theoretic objective function and the actual value of the measurement taken at a location. This renders the IPP algorithm with a computationally cheaper way of solving the IPP problem - by planning the path $a\ priori$. The objective function depends only on the sampling location and not on the measurement taken at the location. This class of IPP problem, which is more explorative in nature, is known as a non-adaptive (offline) IPP problem [22–24]. However, many mobile sensing applications require an exploitative and adaptive (online) planning scheme that accounts for the latest measurements from the underlying field [25–28]. For a search and rescue robot, multiple measurements are beneficial when a human target is detected than when it isn't. Such adaptive planners could combine exploration and exploitation in their planning scheme. Although adaptive IPP is more computationally expensive than non-adaptive IPP, it performs better in terms of information gain [29–31]; the benefits of adaptivity are well known in literature [32]. It is proved that if a problem satisfies adaptive sub-modularity, then a simple adaptive greedy algorithm is guaranteed to be competitive with the optimal policy [33]. Therefore, it is evident that an adaptive planner is more suitable than a non-adaptive planner for target detection using UAV. The planner introduced in this thesis is an adaptive planner, which theorectically yields better performance than a non-adaptive planner.

MYOPIC AND NON-MYOPIC IPP

Early methods in IPP concentrated on approximation algorithms that yield sub-optimal solutions. Greedy strategies [34] that choose the locations that have the maximum information gain at the current instant and do not consider future information gain are called myopic IPP methods [27, 35]. Greedy approaches could also have a finite look ahead where the maximal information gain locations are greedily selected at each step in the planning horizon. The submodularity property of the objective function was proved to significantly reduce the computation time for greedy algorithms [36]. The recursive greedy algorithm [37] is one such popular IPP approach that recurses over possible points in the path to provide near-optimal solutions at quasi-polynomial computation time. [18] has shown that for the sensor placement problem, the pure greedy

algorithm gives results that are within 95% of the optimal solution with a risk of arbitrarily bad solutions. Greedy approaches with limited look ahead time [35, 38] have shown to have shorter computation time but do not provide any optimality guarantee due to the limited planning horizon. A myopic strategy is a good option in situations where it is the characteristic of the monitored target that it mainly occurs in large colonies; the algae or fish monitoring system can look for large colonies or schools myopically. Although myopic strategies might work for some cases in the IPP setting because of its maximal information gain objective, it runs with a risk of obtaining bad solutions.

Non-myopic IPP approaches, on the other hand, have a look ahead and plan to optimize future information gains without greedily selecting locations at each step in the planning horizon [39–42]. In real-world applications, non-myopic methods were shown to outperform myopic ones in terms of running time and information gain [40]. The planner in this thesis considers a finite horizon greedy solution as the intial solution for the optimization routine. Starting with an initial myopic solution, the optimizer evaluates an optimal non-myopic solution for the problem. This approach ensures the performance guarantee of a myopic solution which then gets refined by the optimizer to yield an even better non-myopic solution.

CONTINUOUS AND DISCRETE IPP

Early works discretized the search space before solving the IPP problem, while recent works focus on solving it in a continuous space. Most of the discrete IPP solutions use a graph or tree based combinatorial method. The recursive greedy algorithm [22, 37, 43], and the branch and bound technique [42] stand out in the discrete IPP framework. Branch and bound method was shown to significantly reduce the computation time realtive to the brute force method, therefore enabling the use of larger graphs.

The main disadvantage of discrete IPP is that it is not scalable due to the large graphs and trees that arise; it consumes higher memory and at the same time makes optimization hard and slow. Moreover, the choice of density of the graph largely influences the optimization procedure; fine discretization makes the optimization slow, while course discretization makes the solution sub-optimal. Furthermore, continuous IPP provide smooth trajectories for UAVs which are beneficial for its smooth flight dynamics, whereas discrete IPP provide only distinct via-points in space with no further path information. Therefore, continuous IPP is more suitable than discrete IPP for target detction using UAV. The planner introduced in this thesis is a continuous IPP planner which generate 3D minimum snap polynomial trajectories that cater to the flight dynamics of the UAV. The discrete IPP methods on the other hand generate straight line paths with sharp turns that induce jerks during the flight, thus decreasing the quality of the measurement.

Continuous IPP consists of a large number of approaches and hence demands further classification for a detailed analysis and comparison. Moreover, the approach followed by the thesis belongs to the continuous IPP, therefore demanding a detailed study. Continuous IPP can be classified into three major groups: control or policy learning, sampling-based IPP and trajectory optimization. The following subsections define each of these approaches and compare them against the approach followed in this thesis.

CONTROL OR POLICY LEARNING

Control learning IPP learns a particular strategy that maximizes the information gain from the environment. The curiosity based visual exploration scheme for Autonomous Underwater Vehicles (AUVs) [44], mutual information based control policy for SLAM [15], reactive control policy [45] are some examples. A provably stable adaptive controller that can learn the locations of dynamic events in the environment was introduced by [46]. Finite-horizon model predictive control methods for IPP [47–49] work better than myopic methods, but do not provide performance guarantees beyond the horizon depth.

The problem of policy learning in environment monitoring IPP can easily be formulated as a Partially Observable Markov Decision Process (POMDP) [50]. POMDP methods have been used in very small problems in IPP [51]. [52] showed that POMDP has potential in search and rescue operations using UAVs when compared to greedy strategies, but at the expense of high computation time. However, using general POMDP solvers in a continuous IPP problem will lead to large numbers of continuous variables, which would lead to an even larger belief space, making it almost intractable for solving in real-time in a UAV with on-board processing. Although reinforcement learning techniques demonstrate some hope in IPP frameworks [53–55], they are far from being applied in a real-time active perception problems. Therefore, policy learning is not the most suitable solution for target search using UAV. The approach in this thesis differs from control or policy learning by the fact that there is no policy learning involved. Instead a standard optimizer is used to optimize the objective function to perform an active perception task. This renders the algorithm with the capability to perform

online planning and active perception without any large training time.

SAMPLING-BASED IPP

Another approach to learn a phenomenon is to randomly sample locations in the environment to make measurements and then connect the samples using trees or graphs. This representation is then used to evaluate the most information rich path for the robot. This type of sampling-based strategies have an advantage of quickly exploring the search space to achieve a feasible solution [56]. This property has been exploited by a large amount of IPP literature. The iRRT [57] algorithm based on the Randomly exploring random tree (RRT) algorithm can be used to solve tracking-based information gathering problems without any asymptotic optimality guarantees. Combining ideas from iRRT, RRT and branch and bound technique, the Rapidly-exploring Information Gathering (RIG) algorithm [24] solved continuous IPP in a 2D setting with asymptotic optimality, outperforming the branch and bound method. By combining Gaussian Process and RRT* algorithm, [58] achieved 1200% improvement in RMSE compared to random and greedy strategies in an indoor monitoring task. Combining RRT planner with genetic algorithm [59] yielded better information gathering capability. Other sampling based IPP methods include CE-IPP [60], which uses randomly exploring random graphs (RRG) and cross entropy (CE), asymptotically near-optimal RAOr algorithm [61], ReASC algorithm [28], which outperforms myopic algorithms, BRM [62] algorithm, based on probabilistic roadmaps, CA-RRT [63] which works with a known cost map (hence unsuitable for exploration tasks), RRC [64] etc. Sampling based IPP has also been used for information gathering in exploration-based IPP [14, 65]. Energy-optimal paths for an AUV under strong ocean currents were planned by [66] using simulated annealing and swarm optimization. However, their model does not account for uncertainties while decision making.

Promising solutions were provided by recent papers by utilizing the advantages of a Monte Carlo Tree Search (MCTS) algorithm into the IPP problem [67–69]. They provide anytime solutions that are robust to uncertainties in a Bayesian framework [70]. However, the applications concentrate on ground robots and hence deal with a two-dimensional search space.

Despite its success, sampling-based IPP methods have a disadvantage in high dimensional search space due to their probabilistic completeness. The method finds a solution if one exists, given sufficient runtime of the algorithm - which could be infinite [71]. This might not be readily feasible for applications like search and rescue where time is a crucial element [3]. Therefore, sampling-based IPP is not the most suitable solution for target search using UAV. However, the approach followed by this thesis makes use of a finte horizon random sampling based IPP method to evaluate the initial myopic solution for the optimizer. This strategy takes advantage of the quick search-space exploration capability of a sampling based IPP method to reach an initial solution, within few iterations. Therefore, the algorithm makes use of the advantage of sampling based IPP methods and later improvizes it using standard optimizers to yield an even better solution.

TRAJECTORY OPTIMIZATION

In IPP literature, some recent methods stand out by adopting a different approach in solving continuous IPP problems in a 3D setting. These methods parametrize the path of the UAV in a 3D environment and use advanced hyper-parameter optimization algorithms to tune these parameters such that the resulting path is optimal for information gathering. They are shown to successfully solve more complex problems than the previous IPP methods and are therefore promising.

Bayesian optimization (BO) [72] is one such method, a scalable and parameter-free algorithm [73] which is widely used to find the maximum of expensive cost functions. It sets a prior over the objective function and combines it with evidence to obtain a posterior function. The next best measurement location is then selected based on the utility that takes into account both exploration (sampling from areas of high uncertainty) and exploitation (sampling areas likely to offer improvement over the current best observation [74]). BO has been successfully used in IPP; environment monitoring [75] by optimizing parameters of a continuous spline path [76], reward based active classification for agricultural mapping [77], active perception and smooth navigation [78]. Very recent works [79, 80] have solved the continuous IPP problems in a POMDP framework using BO and MCTS. However, the action space is still discrete and limited, and the work was done in a 2D trajectory space. BO in a POMDP framework was solved in a continuous action and state space by [81]. However, the method was done for a ground robot in a 2D search and trajectory space. Real-time capabilities of these methods in a 3D search and trajectory space are unclear.

Another approach is to use standard evolutionary strategy based optimization techniques to plan the robot trajectory. One such method is Covariance Matrix Adaptation Evolution Strategy (CMAES) which can be used for continuous polynomial trajectory fitting in constrained spaces [82]. CMAES is one of the most

Table 2.1: Overall review of major IPP papers

	Reference	Algorithm	Search space	Traj. space	
Discrete IPP	[37]	Recursive greedy	Graphs & trees	-	-
	[42]	Branch and bound	2D	2D	
Control strategy	[84, 85]	CSQMI based control	3D	2D	
	[53]	iGP-SARSA(λ)	2D	2D	
Sampling based IPP	[24]	RIG	2D	2D	
	[14]	Next best view planner	3D	3D	
	[67]	MCTS	2D	2D	
	[60]	CE-IPP	2D	2D	
Trajectory optimization	[76]	BO	2D	2D	
	[6]	CMA-ES	2D	2D	
	[13]	CMA-ES	2D	3D	

powerful evolutionary algorithms for real-valued optimization and has recently been shown to successfully work on 3D trajectory space in the continuous IPP setting [6, 12, 13]. The method was shown to outperform classic IPP algorithms like branch and bound, BO and RIG in terms of information gain and computational time, thus demonstrating the capabilities of the method in active perception tasks for UAVs. However, it is applicable only for navigation in convex and obstacle-free environments. Moreover, being a quasi parameter free [82] and a derivative free method, CMAES is a black box optimizer [83]. This implies that no knowledge about the way in which the best path is chosen by the optimizer would be unknown to the user, making it hard to reason about the result obtained.

In summary, trajectory optimization methods are suited for UAVs because they take into consideration the continuity and smoothness of the trajectory, which is imperative for the flight dynamics of the UAV. These methods employ state-of-the-art optimization techniques that are global optimizers. However, they are more of "black-box" optimizers with less control parameters, when compared to sampling-based IPP or control-learning IPP. This thesis makes use of a trajectory optimization strategy to plan a 3D polynomial trajectory by using a finite horizon sampling based myopic solution as an initial solution for the optimizer. Both CMAES and BO were used as the optimizer, demonstrating the modularity of the algorithm.

This chapter dealt with providing a brief introduction of IPP literature. The most suitable strategy for target search using UAV was found to be a non-myopic and adaptive strategy. Trajectory optimization in a continuous IPP setting was found to be the most suitable IPP method. Therefore, the thesis adopts a continuous IPP strategy by selecting a continuous path and optimizing it for maximum information gain. The search space is considered to be a 2D ground plane, while the trajectory space is considered to be a 3D environment with obstacles. Table 2.1 gives a an overall review of the major IPP papers along with the algorithm and the dimension of search space and trajectory space.

The layout of the thesis is as follows. The modelling of the field, environment, sensor etc, will be covered in Chapter 3, while the mapping strategies used by the algorithm will be detailed in Chapter 4. Chapter 5 describes the main planning strategy used by the algorithm, and illustrates its working with the help of examples. The algorithm will then be benchmarked against other planners in Chapter 6. The algorithm will be tested against different environments, optimizers, objectives etc in this chapter. Chapter 7 will deal with the real experiments with a UAV flying in a lab environment, running the algorithm. Finally, Chapter 8 will provide a conclusion to the thesis, and will provide the possible future works in the algorithm.

3

MODELLING

Modelling the real-world is imperative for planning to act sensibly in it. Active perception tasks often require this model of the world to optimize the next best location for data acquisition. To know where to observe, it is important to know what could be observed. This is essentially captured by a representative model of the phenomenon. This section deals with modelling various aspects of the planner including the target or phenomenon under observation, the environment through which the UAV has to fly and the human detection sensor.

SIMULATION ENVIRONMENT

The algorithm is completely implemented in C++ in a Robotics Operating System (ROS) environment. *Gazebo* is used to model the environment (shown in Figure 3.1) in which the model of an AscTec Firefly flies around through the planned path and captures images. $Rviz$ is used as a visualization platform for the planned path. $MATLAB$ is only used for plotting saved data and for generating the necessary plots from the results.

Figure 3.1: A 30mx30m environment in *Gazebo* with a building of dimension 10mx4mx26m.

MODELLING THE FIELD

The phenomenon being observed by the UAV in an environment is generally expressed in the form of a map. A map is updated by using the measurements gathered by the UAV from the environment, and provides relevant information for planning and decision making. The environment map can be discretized and represented as a mesh grid. Each cell in the map contains some information about the presence of the target in it. If the presence of a target in a cell could provide some information about the presence of the target in the neighbouring cells, the cells in the grid are correlated. This property is applicable for phenomena which occurs as a cluster - school of fish in sea, algae growth etc. The cells in the grid can be independent of each other if the presence of a target in the cell does not influence the presence of target in the neighbouring cells. Depending on the underlying assumption of independence or interdependency between each cell in

the map, the map can be represented as an Occupancy Grid (OG) or as a Gaussian Process (GP) respectively. In this thesis, both the modelling approaches were considered. The GP assumes a continuous and correlated model of the field, while OG assumes an independent and discrete model of the field. Correlations within the phenomenon being observed motivates a GP model of the field, while an uncorrelated phenomenon motivates the use of an OG map. The following sections introduce the implemented modelling strategies.

GAUSSIAN PROCESS

GP is a nonparametric Bayesian technique that places a prior distribution over the space of function to be evaluated. It uses the set of observed samples to estimate the value of the unobserved areas in the input space [86]. A GP models spatial correlations in the map in a probabilistic and non-parametric manner [87]. The target map is assumed to be a continuous function in 2D space $\zeta : \varepsilon \rightarrow \mathbb{R}$, where $\varepsilon \subset \mathbb{R}^2$ is a point on the map ζ under consideration. GP is fully characterized by the mean $\mu = E[\zeta]$ and the covariance $P = E[(\zeta - \mu)(\zeta^T - \mu^T)]$. The continuous 2D function of the environment being mapped can therefore be expressed as $\zeta \sim GP(\mu, P)$. This representation of the field embeds our best estimate of the field being observed in the form of the mean μ, and the correlation between different points on the map in the form of P. μ can be interpreted as the best estimate of the probability of target occupancy in the cell. The covariance matrix P captures the interdependency between different points in the map. This is particularly important in cases where the phenomenon being observed usually appears in the form of clusters. Groups of victims at a particular location in a disaster site and highly concentrated metallic content in an underground mine detection task are few examples for correlated data in the field being observed.

In monitoring tasks where the prior information about the field is available, the information can be blended into the GP model as an initialization in the form of a pre-trained kernel $K(X, X)$, which is the $n \times n$ matrix of covariances evaluated at all the pairs of training points. With the fixed-size environment discretized at a certain resolution with n training locations $X \subset \varepsilon$, we can predict the prior map in n^* test locations $X^* \subset \varepsilon$. Each element $k(x_i, x_j)$ of $K(X, X)$ is defined by the covariance function or the kernel, which is the key ingredient of the GP. The kernel encodes all the assumptions about the map ζ that is being modelled. If the cells x_i and x_j are close to each other, then their map values are assumed to be similar to each other. This similarity between the cells are embedded in the kernel. The most frequently used kernels in literature are squared exponential, Matern kernel and periodic covariance function. The isotropic Matern 3/2 kernel function common in geostatistical analysis and spatial statistics, is used to describe the field. It is defined as [87]:

$$k_{Mat3}(x, x^*) = \sigma_f^2 \Big(1 + \frac{\sqrt{3}d}{l}\Big) \exp\Big(-\frac{\sqrt{3}d}{l}\Big), \tag{3.1}$$

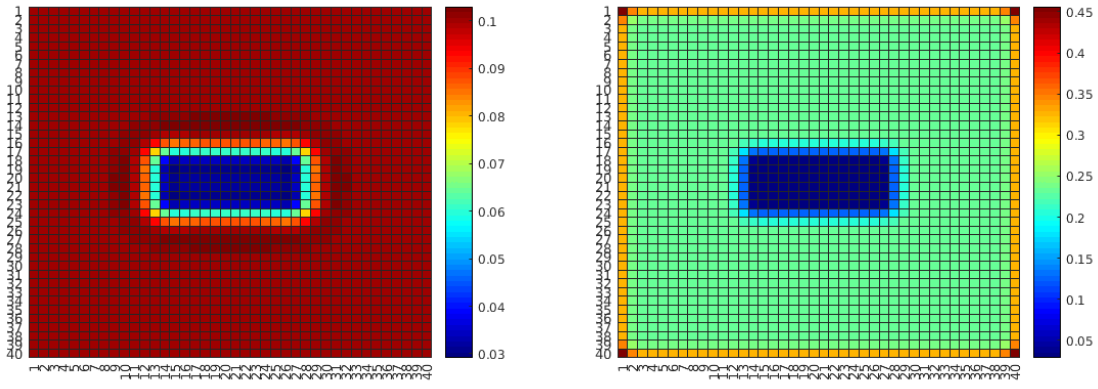
where l and σ_f^2 are the hyperparameters representing the lengthscale and signal variance respectively, and d is the Euclidean distance between inputs x and x^*. l represents the smoothness of the function describing the map; small l represents a function which changes quickly while large l represents a very smooth function. σ_f^2 is the scaling factor that determines the variation of the function value from the mean. The set of fixed hyperparameters $\theta = \{\sigma_n^2, \sigma_f^2, l\}$ represents the control relations in the GP and can be optimized using various techniques like maximizing the log-marginal likelihood of the data [87]. The training can be done with previously acquired maps to match the properties of the map ζ at the required resolution.

For a search and rescue operation, there is no prior information available about the field and hence, the values at $x_i \in X$ are initialized with a constant prior mean. However, from the available real map of the environment from *Google Maps* for example, certain regions in the field which are occupied by the buildings could be identified. These regions are least likely to contain humans. The search is restricted within surface victims who lie on the field so that they are visible from a flying UAV. Therefore, a very low initial constant prior can be used at these locations. The covariance is then calculated using the classic GP regression equation [88]:

$$P = K(X*, X*) - K(X*, X)[K(X, X) + \sigma_n^2 I]^{-1} K(X^*, X)^T, \tag{3.2}$$

where $K(X, X^*)$ denotes the $n \times n^*$ matrix of the covariances evaluated at all pairs of training and test points, P is the posterior covariance, and σ_n^2 is a hyperparameter representing the noise variance. The diagonal elements of covariance matrix are the variances of each point on the map, which is the uncertainty associated with the mean at that point on the map. Therefore, trace of the covariance matrix P represents the total variance or uncertainty of the map.

Figure 3.2 shows the initial GP mean and variance for an environment shown in Figure 3.1 with a grid resolution of $0.75mx0.75m$. It can be observed from Figure3.2a that the mean is lower at cells closer to the



(a) GP mean

(b) GP variance

Figure 3.2: Field shown in Figure 3.1, initialized with a GP prior $\mu = 0.1$ at unknown locations

obstacle, than those that are far. This results from the continuous model of the field that uses a GP. It can also be observed that the value of variance at the corners and edges are higher due to higher uncertainty at boundaries, which is the characteristic of a GP. The uncertainty at the location where there is a building is observed to be low, which can be accounted to the assumption that surface victims are not present inside the building.

It should be noted that the GP has a disadvantage of increasing computation complexity with observations taken. The computation complexity of GP is cubic in the number of points observed $O(n^3)$ [89], and hence is not preferred if large number of observations are to be made on the field.

OCCUPANCY GRID

The most commonly used map representation for planning in robotics is the occupancy map. The field can be represented as a 2D occupancy grid [90] m where each cell is associated with a Bernoulli random variable that represents the probability of target occupancy. The probability distribution of the map is given by the product over the cells as shown in Equation 3.3, where $p(m_i)$ is the probability of the cell to be occupied by the target (human), and $1 - p(m_i)$ is the probability of the cell to not be occupied by the target.

$$p(m) = \prod_i p(m_i) \tag{3.3}$$

The cells m_i are assumed to be independent of each other, which is not an accurate assumption due to the presence of the same target in multiple cells. However the factorization simplifies the estimation of posterior probability of the map, enabling the use of a binary Bayes filter to estimate the occupancy probability for each grid cell. The log-odds representation is used to denote the probability that each grid cell is occupied, and is given in Equation 3.4.

$$l(m_i) = \log \frac{p(m_i)}{1 - p(m_i)} \tag{3.4}$$

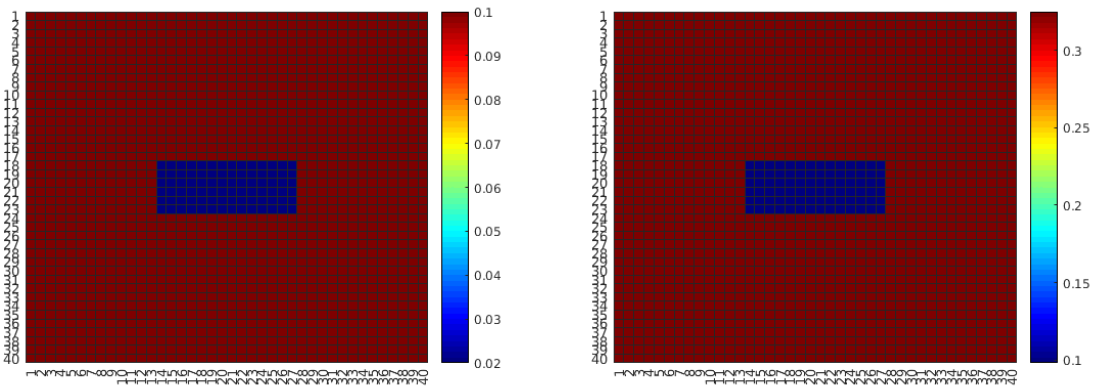
The probability can be retrieved back by using Equation 3.5.

$$p(m_i) = 1 - \frac{1}{1 + e^{l(m_i)}} \tag{3.5}$$

This approach decreases the computational complexity of the occupancy grid update and at the same time increases the precision of the calculation due to smaller round-off errors. Similar to the previous section, an unknown prior can be set as an initialization for the occupancy probability. The known locations in the environment like the buildings can be initialized with a low probability for human occupancy.

The information entropy or Shannon's entropy is defined as the average rate at which information can be generated by a stochastic source of data [91]. It defines the uncertainty related to the estimate for occupancy of the grid and is defined as:

$$H = \sum_{m \in M} p \log p + (1 - p) \log(1 - p), \tag{3.6}$$



(a) Human occupancy probability

(b) Entropy of the field

Figure 3.3: Field shown in Figure 3.1, initialized with a OG prior $\mu = 0.1$ at unknown locations

where p is the probability for the cell to be occupied by the target. Entropy is the minimum if the occupancy probability is either 1 or 0, and is maximum if the occupancy probability is 0.5. Therefore, entropy represents uncertainty associated with the current occupancy probability of the cell.

Figure 3.3 shows the initial OG values and the entropy for an environment shown in Figure 3.1 with a grid resolution of $0.75m$x$0.75m$. It can be observed from Figure 3.3a that the occupancy probability is discrete and does not decrease with respect to distance from the obstacle, owing to the independence of the cell occupancies. From Figure 3.3b, it can be observed that entropy is the lowest at the building as we are certain that there are no humans there, while it is maximum at other locations.

ENVIRONMENT MODELLING

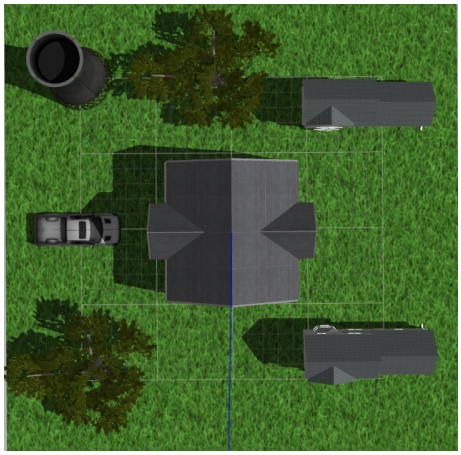
The environment in which the UAV flies is often complex and may feature various obstacles like buildings, trees etc. In addition, there are regions in the environment which are detrimental to the flight, regions on fire for example. Modelling the environment is therefore crucial to avoid such regions and ensure flight safety. Moreover, obstacles in the environment limit the field of view (FoV) of the UAV and hence should be considered while mapping. A sample urban environment with housing in close proximity to a chemical/nuclear reactor is shown in Figure 3.4. During a man-made disaster at the reactor, the UAV should not cross into the critical radius of the danger zone. It is however supposed to map the area in the danger zone that is not occupied by the reactor. This section deals with modelling the environment by taking these factors into account so as to aid the planning and mapping strategies described in the coming chapters. As illustrated in Figure 1.1, there are two map representations: one is the 2D information map of the field that was detailed in the previous section, while the other is the 3D obstacle map that will be discussed in this section.

OBSTACLES AND DANGER ZONES

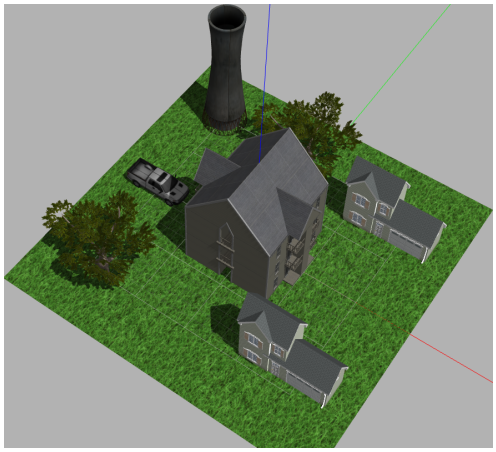
The obstacles in the environment are assumed to be opaque and rigid. This implies that the obstacles completely limit the FoV of the UAV and at the same time provide complete rigidity in case of a collision. The danger zone is defined by a central location and a critical radius. The region is assumed to be a high cylindrical obstacle during the path planning, while the region is treated as transparent while mapping. For simplicity, the obstacles are approximated to the closest volumetric shape. The buildings are modelled as a cuboid that can enclose them, while the trees are modelled as an enclosing sphere. Figure 3.5 shows the $Rviz$ visualization of the environment. Note that the size of the reactor in Figure 3.5b is slightly larger than in Figure 3.5a because of the critical radius. Additionally, the boundaries of the environment and the ground are treated as a plane obstacle, which is not shown in Figure 3.5b.

ESDF MAPS

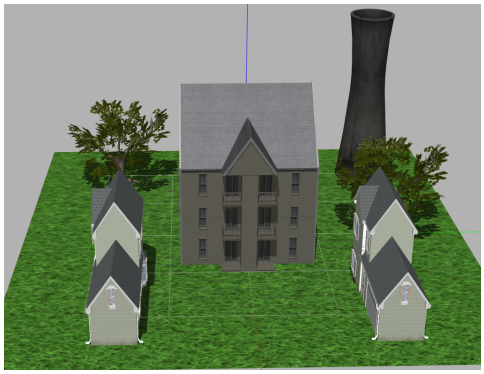
A UAV flying in an obstacle filled environment must know the information about the closest obstacle to it, so as to avoid a head-on collision. A well known representation in robotics is the Euclidean Signed Distance Function (ESDF) which is defined as the distance from a point in the free space of the environment, to its



(a) Top view

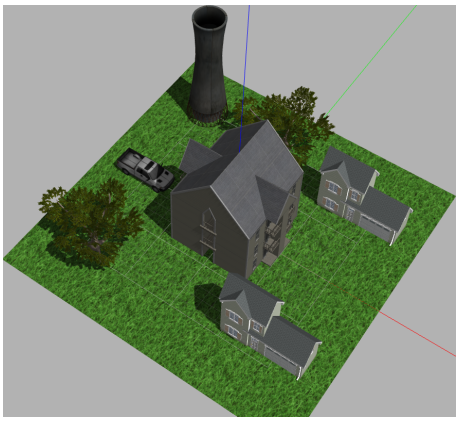


(b) Isometric view

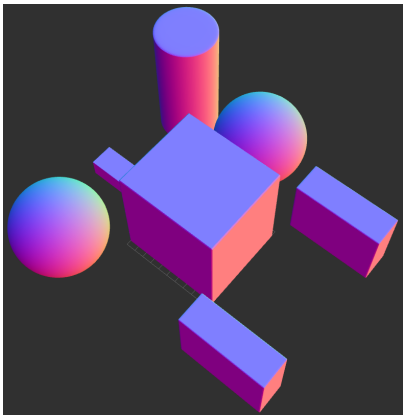


(c) Front view

Figure 3.4: An urban environment with a dangerous reactor near the housing in *Gazebo*



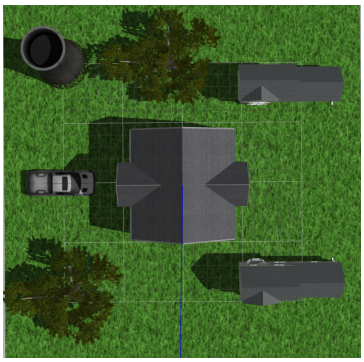
(a) The environment in *Gazebo*.



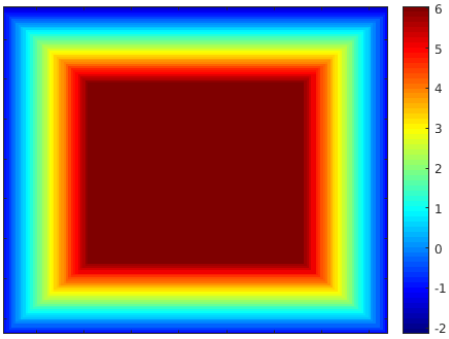
(b) *Rviz* visualization of environment.

Figure 3.5: Modelling the environment using simple geometric shapes

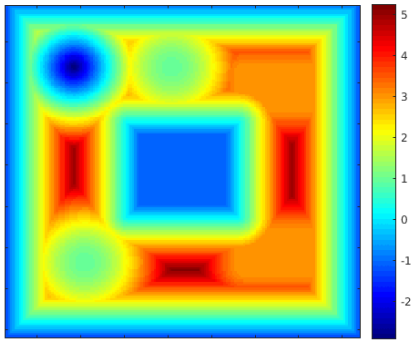
closest obstacle. In case of points inside an obstacle, it is the distance to the closest free space in the environment, with a negative sign. Therefore, the ESDF measure is positive if the point is outside the obstacles, while it is negative inside an obstacle, and zero on the boundary of an obstacle. In order to decrease the computation time for planning, we create an ESDF map of the environment as a preprocessing step, and query from it while planning. This representation of the environment is particularly convenient to use because checking the sign of the ESDF value at a point determines if the point is in the free space or inside an obstacle.



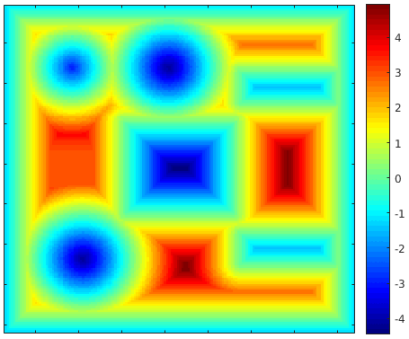
(a) Top view.



(b) ESDF map at height = 20m.



(c) ESDF map at height = 10m.



(d) ESDF map at height = 5m

Figure 3.6: ESDF map plotted on horizontal planes at different heights.

The *voxblox* package [92] is used to generate the ESDF map for the sample environment. Figure 3.6 shows the ESDF map slices at different heights. The red region is the region which is farthest from any obstacle, while blue regions are inside an obstacle. It can be noted that as the height decreases, the obstacles in the environment appear in the ESDF map as expected. A UAV planning a path using such a map would prefer to stay close to the red regions to avoid any possible collisions. The next section deals with the usage of ESDF maps to generate a penalty function to model collisions for planning.

ARTIFICIAL POTENTIAL FIELD

Artificial potential fields [93] are a well known approach in robotics for global path planning in an obstacle filled environment. We make use of potential fields to create a penalty function for the proximity of the planned path to an obstacle or boundary, by also considering the UAV dimesions. We use two most frequently used potential fields in literature: hard constraint and logistic function. The hard constraint penalty function penalizes in case of a collision and does not penalize if there is no collision as given in Equation 3.7, where $x = ESDF(UAV\_pose)$ is the ESDF value queried at the position of UAV, $C_{collision}(x)$ is the penalty function, and x_{UAV} is the size of the UAV.

$$C_{collision}(x) = \begin{cases} 0 & \text{if } x \geq \frac{x_{UAV}}{2}, \\ -1 & \text{otherwise.} \end{cases} \tag{3.7}$$

The logistic function penalizes collision in a softer way depending on how close the UAV is to the obstacle. A combination of a logistic function and a linear function is used so that the penalty increases closer to the obstacle, while it decreases smoothly outside the obstacle to zero. The penalty function is defined as given in Equation 3.8.

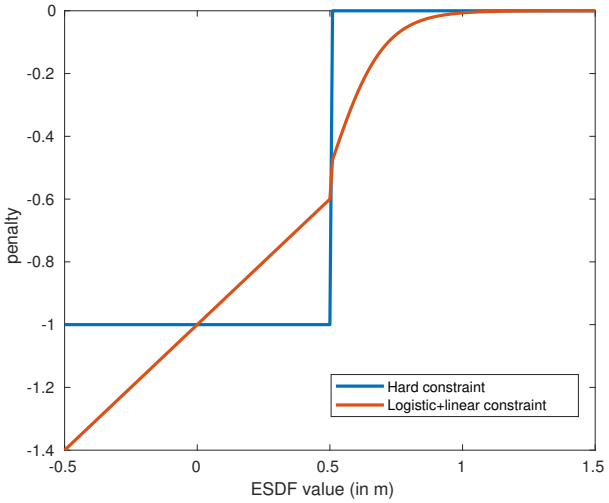
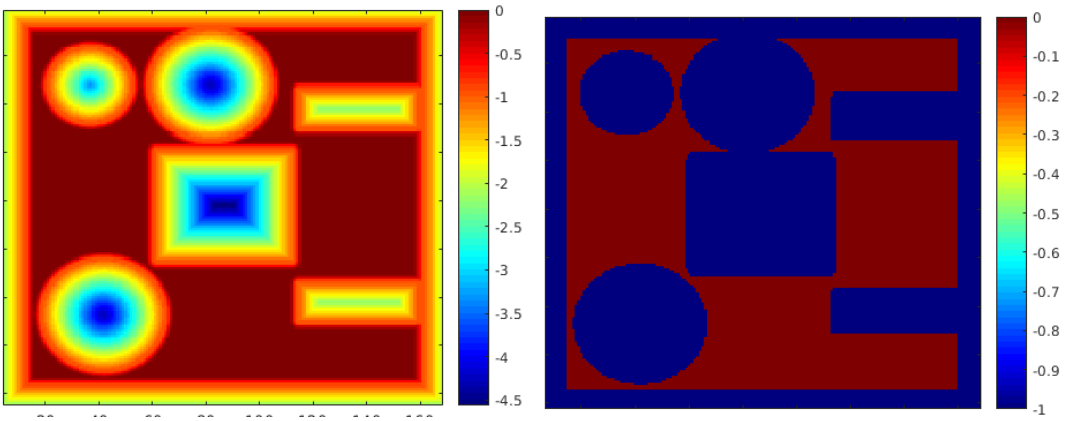
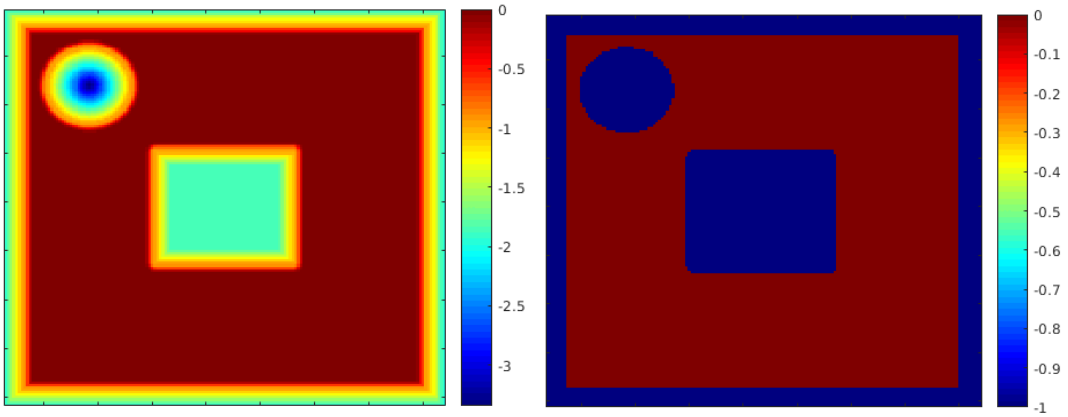


Figure 3.7: Artificial potential field with $\frac{x_{UAV}}{2} = 0.5m$



(a) Logistic+linear penalty. (b) Hard constraint penalty.

Figure 3.8: Artificial potential fields corresponding to the ESDF map in Figure 3.6d.



(a) Logistic+linear penalty. (b) Hard constraint penalty.

Figure 3.9: Artificial potential fields corresponding to the ESDF map in Figure 3.6c.

$$C_{collision}(x) = \begin{cases} -1 + 0.8x & \text{if } x \leq \frac{x_{UAV}}{2}, \\ \frac{1}{1+exp^{-10\left(x-\frac{x_{UAV}}{2}\right)}} - 1 & \text{otherwise.} \end{cases} \tag{3.8}$$

Figure 3.7 shows the comparison between the nature of both potential fields used. It can be observed that the penalty is zero if the UAV is away from the closest obstacle, while the penalty is negative otherwise, representing a punishment for collision. It should be noted that the Equations 3.7 and 3.8 represent a discontinuous function, and the discontinuity can be observed at $x = \frac{x_{UAV}}{2}$ in Figure 3.7.

Figure 3.8 shows the artificial potential field for the ESDF map given in Figure 3.6d. It can be observed from Figure 3.8a that the penalty increases inside the obstacle, while it stays the same for Figure 3.8b. It can be noted that flying around at a height of 5m yields high penalty because of very small gaps to pass in between the obstacles. However, flying around at a height of 10m yields much lower penalty as can be observed from Figure 3.9. Therefore, when collision avoidance is considered, there is a preference for the flying height of the UAV depending on the environment. In the following section, the human detection sensor is modelled for varying heights. The objective is to determine if the sensor prefers a certain height for best performance.

SENSOR MODELLING

The model of a sensor is essential to determine the quality or usefulness of the observations while mapping, and to determine the location of maximum information gain while planning. Without a sensor model, the quality of information to be gained from a location in the environment would be unknown. It is useful in predicting measurements and making decisions to yield the best path for data acquisition. Most often, the sensor measurements are noisy and have an uncertainty associated with each measurement. The formal definition of measurement uncertainty with respect to continuous and discrete random variable is defined in Section 1.3. Accounting for such uncertainties is crucial for a planning algorithm to ensure robustness. For example, if a sensor which gives measurements with high uncertainty is used, the algorithm should adapt such that multiple measurements are taken at the same location. Therefore, sensor modelling is crucial for planning. This section deals with modelling a human detection sensor and evaluating the correlation between the sensor performance and the altitude, so that it can be used for planning and mapping in the upcoming chapters.

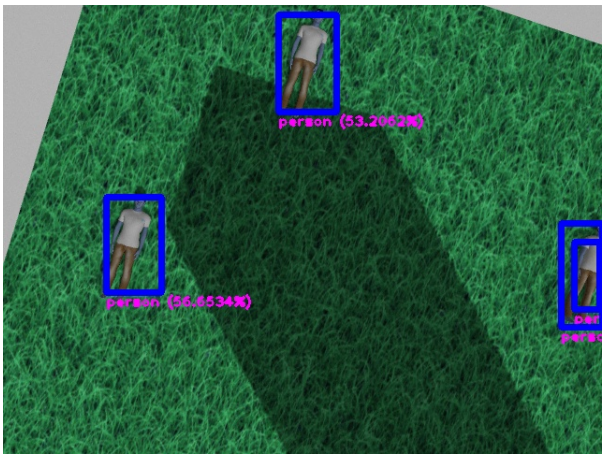
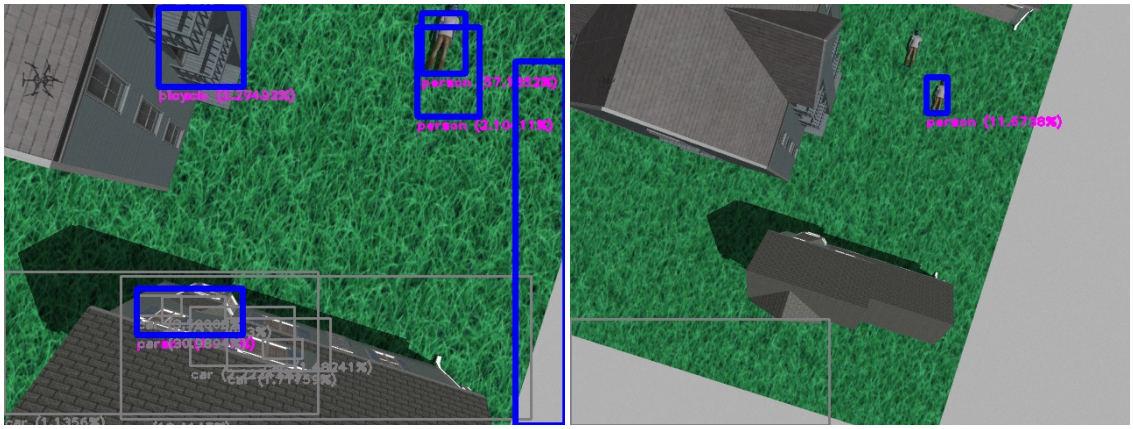


Figure 3.10: True positive detections by YOLO.

HUMAN DETECTION SENSOR

Currently available human detection sensors are based on different types of information - temperature or images for example. An image based human detection framework was selected to detect and locate the humans in the scene. However, the algorithm is modular and can incorporate other sensors, given that it provides the detected locations. Single Shot MultiBox Detector (SSD) [94] and You Look Only Once (YOLO) [95] are the state-of-the-art deep learning based object detectors that can detect several objects in the image including humans. Histogram of Oriented Gradients (HOG) [96] is a widely used human detector that uses a gradient



(a) False positive detections by YOLO. (b) False negative detections by YOLO.

Figure 3.11: The YOLO human detection marked with blue bounding box

based descriptor to train a linear Support Vector Machine (SVM) based human classifier. The human detectors currently available are biased towards humans in the upright position. However, in a search and rescue operation, they could be found in abnormal poses, and might be occluded by the debris. Therefore, human detectors trained on disaster dataset would be the most suitable detector for the application. However, there is a literature gap in this area.

The human detectors available as opensource C++ packages were experimented on the images captured from Gazebo. YOLO Tiny 2.0 demonstrated the best performance on the images with the least processing time. Therefore, YOLO Tiny was selected as the human classifier for the thesis. It should be noted that the planner is modular and hence can be replaced with a different sensor. Figure 3.11 shows the detected bounding box around the human mannequins in an image captured by a flying UAV. It can be noted that in Figure 3.10, the detections are near perfect which are true positive detections. However, YOLO is not perfect and might give false positives and false negatives as shown in Figure 3.11a and 3.11b respectively.

Some of the false negative detections by YOLO can be filtered out based on the size of the detected human. For example, if a bus is detected as a human, a size matching will be performed and the detection will be labelled as a non-human one. The size of the detection can be evaluated by performing the coordinate transform from the image plane to the ground plane. Figure 3.12 shows the effectiveness of this method in removing the majority of false positive human detections.

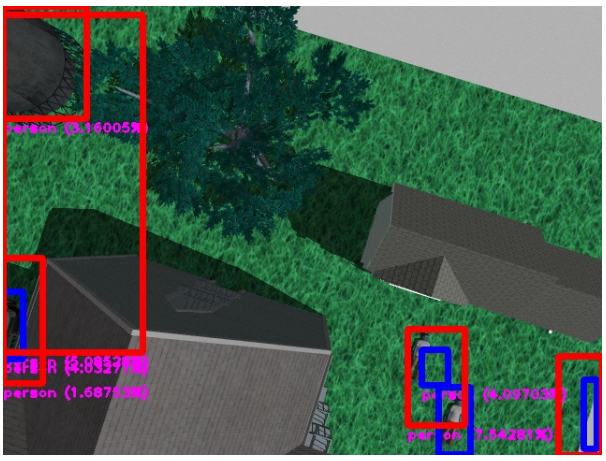


Figure 3.12: Giant human detections by YOLO marked by red are classified as non-human detections.

PERFORMANCE OF YOLO AT DIFFERENT ALTITUDES

The performance of the classifier should be evaluated at different altitudes for planning a path for data acquisition in 3D space. An increase in altitude increases the coverage, while decreasing the size of human in the

captured image. Therefore, there should be a tradeoff between accuracy of the sensor and the coverage while planning. This section deals with formulating a height dependent sensor model for mapping and planning.

It can be observed from the previous section that YOLO is not a perfect human classifier. There is an uncertainty associated with each output from YOLO. Images captured from low altitudes contain humans with high resolution and is less noisy. However, images captured from high altitudes contain humans with low resolution, and is more noisy. Previous modelling approaches for a height-dependent sensor model uses artificial uncertainty curves [12], assuming that the uncertainty increases with respect to height and saturates beyond a certain height. We use the same model to determine the uncertainty of measurement at a given height, given by,

$$\sigma_{s,i}^2 = A(1 - e^{-Bh}), \tag{3.9}$$

where $\sigma_{s,i}$ is the noise variance expressing uncertainty in measurement, h is the measurement height and A and B are positive constants. The behavior of the image captured from a UAV at different altitudes is essentially captured by Equation 3.9.

The underlying model of YOLO is a deep neural network which is essentially a black box classifier. Therefore, it is not a trivial task to model the way in which YOLO performs. Although YOLO is supposed to be a scale invariant object detector, it fails to classify the humans correctly in an image taken from beyond a particular height in the environment. Moreover, the classifier was observed to not detect the humans correctly when the image was taken at a very low height. The most probable reason for this is the absence of such images in the training dataset. This intuition from the real experiments with the classifier led to the idea that the performance of YOLO is in fact height dependent.

The most frequently used performance metric for an image based classifier is precision, recall and F1 score. Precision is the fraction of correct human detection among all the detctions made by the classifier, while recall is the fraction of correct human detections among all the humans in the ground truth. F1 score combines precision and recall into a single metric, which is the harmonic mean of both measures.

$$precision = \frac{true\ positive}{true\ positive + false\ positive} \tag{3.10}$$

$$recall = \frac{true\ positive}{true\ positive + false\ negative} \tag{3.11}$$

$$F1\ score = \frac{2 \times precision \times recall}{precision + recall} \tag{3.12}$$

It should be noted that the ground truth is based on the location of humans placed in the field. Therefore, the performance metrics are evaluated after transforming the pixels from the image plane to the ground plane and then comparing it against the ground truth of the discretized field. This implies that the number of grid points in the field that are visible from a particular height increases with height, even though the number of pixels in the images remain the same.

We propose to model the performance of the classifier based on experiments on a realistic simulated environment. 100 flights were simulated in the environment shown in Figure 3.1 with a human mannequin placed below the UAV. The UAV takes off from a height of 2m, directly above the human mannequin and flies to a height of 26m vertically, while capturing images. F1 score is evaluated after transforming each image to a sub map on the field. Figure 3.13 shows the mean F1 score along with one standard deviation shown in red. It can be observed that the performance of the classifier increases with height, reaches a maximum and then decreases to zero, staying zero beyond the saturation height. Therefore, it can be concluded that the performance of the YOLO as a human classifier is height-dependent, and that there is a saturation height beyond which the classifier fails completely. The classifier has a preferred height for operation that maximizes its performance. Therefore, it is crucial that the planner simultaneously trades off between accuracy, coverage and obstacle avoidance to plan the path.

Based on the experimental results shown in Figure 3.13, a height-dependent inverse sensor model for detecting humans correctly while using YOLO was proposed as a normal distribution till the saturation height:

$$P(human = 1 | z = 1) = \begin{cases} P_{max} e^{-0.5\left(\frac{h - h_{opt}}{\sigma_1}\right)^2} & \text{if } h < h_{sat}, \\ 0.5, & \text{otherwise} \end{cases} \tag{3.13}$$

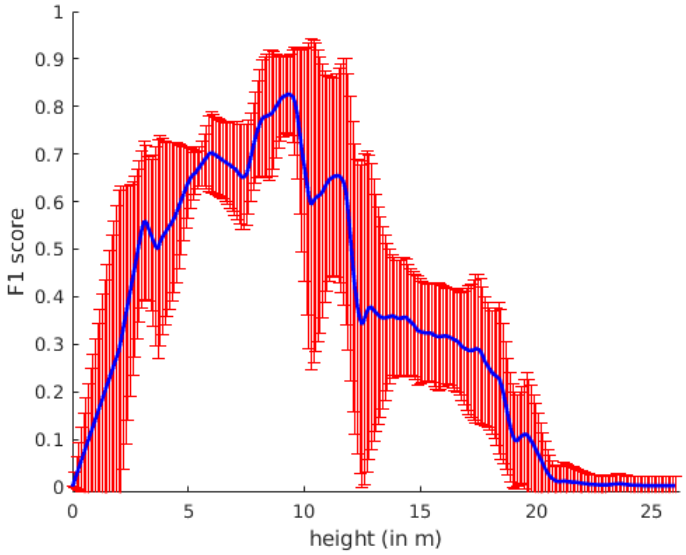


Figure 3.13: F1 score of YOLO on images captured at different heights. The plot shows the mean and standard deviation of 100 flight experiments.

where $P(human = 1|z = 1)$ is the probability for the cell to contain a human if the prediction is a human at an altitude h, h_{opt} is the height of maximum F1 score in Figure 3.13, h_{sat} is the saturation height at which F1 score drops to minimum, σ_1 is the standard deviation of the F1 score curve approximated as a normal distribution, and P_{max} is the maximum probability achievable. Similarly, a height-dependent inverse sensor model for YOLO to misclassify humans is also proposed:

$$P(human = 1|z = 0) = \begin{cases} 1 - \left[(1 - P_{min})e^{-0.5\left(\frac{h-h_{opt}}{\sigma_1}\right)^2}\right], & \text{if } h < h_{sat} \\ 0.5, & \text{otherwise} \end{cases} \tag{3.14}$$

where $P(human = 1|z = 0)$ is the probability for it to be a human if the prediction is not human at altitude h, and P_{min} is the least probability misclassify humans, that the sensor can attain. The curves for both the sensor models are shown in Figure 3.14 where $P_{max} = 0.95$, $h_{opt} = 10m$, $h_{sat} = 20m$, $\sigma_1 = 10$ and $P_{min} = 0.1$.

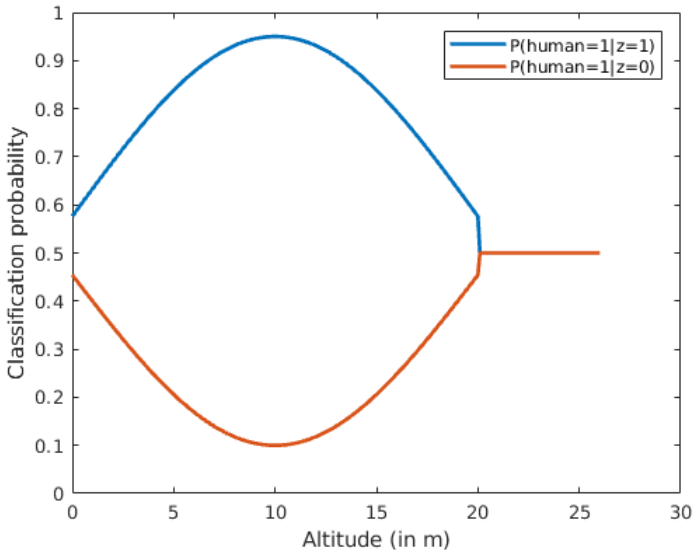


Figure 3.14: Proposed height dependent inverse sensor model for YOLO.

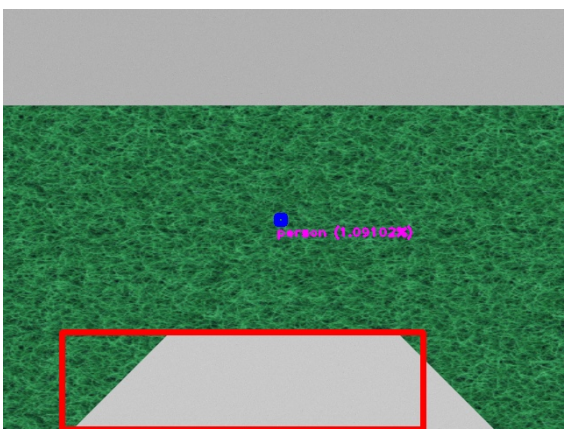
The sensor works best when it simultaneously produce the highest correct human classification and the lowest human misclassifications. The sensor performs worst when both the probabilities approach unknown classification probability of 0.5. It can be observed from Figures 3.13 and 3.14 that the sensor performs the best at almost the same altitude of 10m. Moreover, in both figures, the sensor performs worst beyond a saturation altitude or at a very low altitude. Therefore, the sensor model shown in Figure 3.14 closely represents the experimental results in Figure 3.13 and can be used for planning. However, it should be noted that the constants in the sensor model formulated in this section should be adapted to the environment under consideration. The results in Figure 3.14 have a close resemblance to the artificial probability curves for an agricultural weed classifier in [13], except for the decreasing performance of YOLO that was observed in the experiments when the altitude is very low. The proposed sensor model is directly used by the mapping and planning strategies in the next chapters.

In summary, this chapter dealt with the modelling the various components that are necessary for the planner. The field was modelled using two methods: GP and OG. A GP represents a correlated field while an OG represents a non-correlated field. The environment was then modelled using ESDF maps to represent the obstacles and danger zones. Artificial potential functions were used based on the ESDF maps to model the penalty for collision with the obstacles. The YOLO object detector was used as a human classifier, for which a height-dependant sensor model was formulated based on simulated experiments of UAV flight. It was observed that both collision avoidance and sensor performance have a preferred altitude for flight, based on the environment. This motivates the use of an IPP based approach for target detection since the information gain from the environment is strongly dependent on the altitude. The next chapter deals with the mapping strategy used by the algorithm so as to fuse the measurements taken by the UAV into the map so as to bring out a meaningful representation of humans in the field.

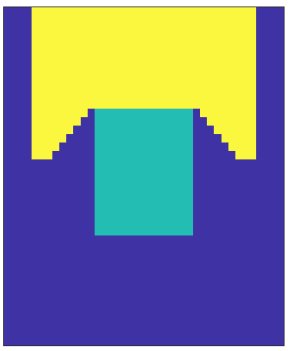
4

MAPPING

The map captures all the information that is required for planning, and is very valuable for many applications, thereby demanding accuracy in its estimation. Mapping is the process of fusing the sensor measurements onto a map that we are interested to build. In our case the map is the representation of the field with the presence and absence of humans clearly indicated in it. The objective is to fuse the output of YOLO into the field in a probabilistic manner so that the uncertainty related to each measurement is also taken into account while mapping. Updating the map should increase the belief in the map so that it matches with the ground truth of humans in the field, provided a good sensor performance. The output of the algorithm is a map that will be passed to the search and rescue team at the end of the flight, so that the detected victims can be given immediate medical attention. Therefore, it is imperative that the output of the planner is as accurate as possible, which mainly depends on the mapping strategy followed. Therefore, mapping serves as a crucial step in automated search and rescue using UAV. This chapter deals with evaluating the FoV of the UAV flying in an environment filled with obstacles, and then updating the map from the measurements acquired.



(a) Measurement from the camera at altitude = 22m.



(b) FoV shown in yellow and obstacle in green.

Figure 4.1: Identification of the qualifying cells for map updation in the field.

FIELD OF VIEW

In an obstacle rich environment, the UAV has a restricted visibility of the field. Determination of FoV is therefore necessary for proper mapping and planning. The field is first discretized into cells of uniform size. All the pixels in the captured image is then projected onto the ground plane, including those pixels representing the obstacles (grey portion in Figure 4.1a). It is then transformed into the corresponding grid coordinate in the field. This yields a subset of the field which is currently visible by the UAV assuming that there are no obstacles. The procedure to evaluate the FoV involves filtering out those grid points whose visibility is blocked

by the obstacle. This problem is generally solved in robotics using a ray casting method where the line joining the UAV location and the grid point on the ground is finely discretized and checked for collision with any obstacle. However, the computational complexity of the ray casting method on an image of dimension $m \times n$ pixels and a discretization size of p points (practically more than 250 for 30m×30m environment) on the line joining the UAV position and the projected pixel on the ground, is $O(mnp)$. A computationally faster alternative using ESDF maps is proposed. Let r_{UAV} be the radius of the smallest sphere that can enclose the UAV, $R_g(x_g, y_g, z_g)$ be the projected pixel on the ground, and $R_0(x_0, y_0, z_0)$ be the position of the UAV. The next point can be sampled from the line connecting R_0 and R_g by using the ESDF value at the previous point to move along that line from R_0 towards R_g. This is possible because the ESDF value represents the minimum distance to an obstacle and hence we can be sure that there is no obstacle close to it within a radius equal to the ESDF value. Therefore skipping all the points in between and jumping towards R_g along the line would not result in a collision. In a loop, it only takes a few iterations i (practically 4) to determine if the ground is visible or not, compared to 250 for the ray casting method. The computational complexity is drastically reduced in case of planning where the FoV should be calculated on at least k (practically more than 40) images to plan a path. Therefore the computational complexity is reduced from $O(kmnp)$ to $O(kmni)$, where $i << p$, when compared to the discretized ray casting method. The pseudocode to check the ground visibility is given in Algorithm 1.

Algorithm 1 Function to check if a ground point is visible from a UAV.

function ISGROUNDVISIBLE(R_0,R_g,ESDF)
  Initialize $distIncr = 0.05$
  if UAV is already close to obstacle **then**
    $distIncr = r_{UAV}$ ▷ throw the point out of the obstacle
  Initialize u = $\frac{distIncr}{||R_0 - R_g||}$ ▷ scale the increment
  while u<1 **do** ▷ until we reach the ground point
    $next\_point = (1-u)R_0 + uR_g$
    $min\_dist = ESDF(next\_point)$
    if $next\_point$ has low ESDF value **then**
      if $next\_point$ is not close to ground **then**
        return false
      else
        return true
    Increment u by $\frac{min\_dist}{||R_0 - R_g||}$ ▷ jump towards the ground point

The pixels are successfully projected onto the map, and those points on ground that are blocked by the obstacles are filtered out. However, we do not want to update the cells in the field if only a few pixels project into the cell. It should be updated only if a number of pixels reach a consensus over the measurement. This uncertainty happens on the pixels that are at the corners and edges of the image. Therefore, hashing operation is performed on each pixel by considering the grid cells as a bucket to fill in the pixels. Once the number of pixels mapping onto the cell is beyond a particular threshold value, the cell is considered as qualified to be updated, else the cell is discarded from being updated.

The algorithm is applied to identify the qualified grid points in the field for updation, and an example is shown in Figure 4.1. Figure 4.1a shows the image captured by the UAV where the grey section is an obstacle. Figure 4.1b shows the FoV of the UAV in yellow. It can be observed that grass in Figure 4.1a has the same pattern as the yellow patch in Figure 4.1b. This shows that the FoV calculation is correct and that Algorithm 1 works correctly as expected. The next section deals with fusing the sensor information in the FoV onto the map, for both continuous and discrete mapping scenarios.

SEQUENTIAL DATA FUSION

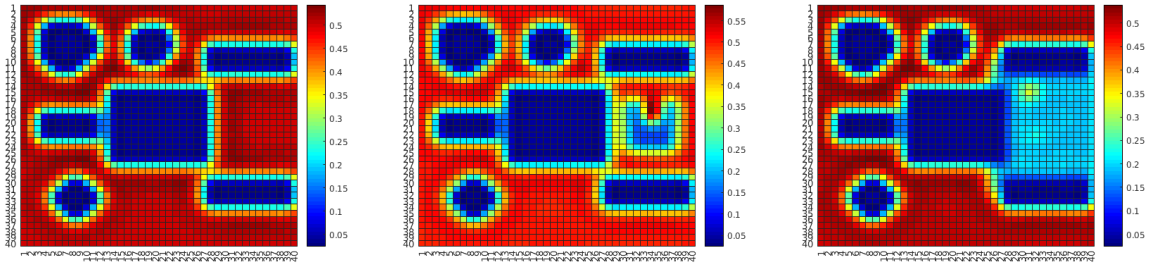
In the previous section, an algorithm to evaluate the FoV of the UAV was presented. Fusing the data in the FoV onto the map in a probabilistic manner, while considering the sensor uncertainties described in Section 3.4, is important to produce an accurate map of the field. This section presents two types of sensor fusion algorithms: one for the continuous mapping for the GP detailed in Section 3.2.1 and the other for a discrete mapping for the OG detailed in Section 3.2.2.

KALMAN FILTER UPDATE

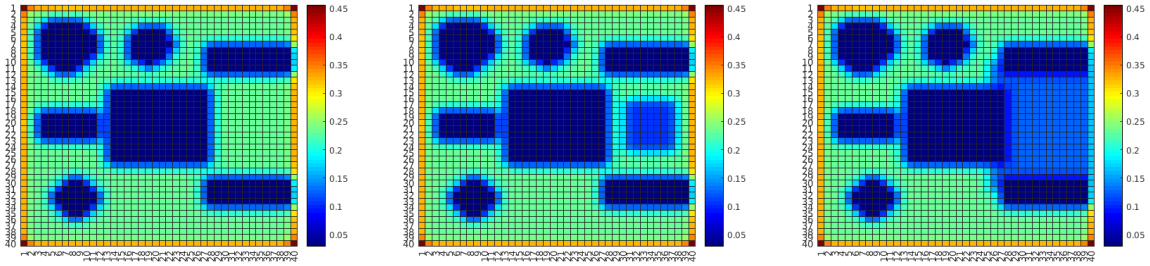
This section deals with the data fusion for a GP using recursive filtering based on Kalman Filter (KF). Given a uniform initial mean μ^- and spatial correlations P^- given in Equation 3.2, the map $p(\zeta|X) \sim GP(\mu^-, P^-)$ is used as a prior onto which new sensor measurements are fused. Let $z = [z_1, ..., z_m]^T$ denote the m independent measurements for the human occupancy given by YOLO at all the points $[x_1, ..., x_m]^T \subset X$ in the FoV evaluated in Section 4.1. The maximum *a posteriori* estimator is used to fuse the measurements z with the prior map $p(\zeta|X)$, as formulated as:

$$\arg\max_{\zeta} p(\zeta|z, X) \tag{4.1}$$

(a) Environment with 2 humans. (b) YOLO at height = 5m. (c) YOLO at height = 15m.



(d) Initial GP mean. (e) GP mean after fusing figure 4.2b (f) GP mean after fusing figure 4.2c



(g) Initial GP variance. (h) GP variance after fusing Figure 4.2b. (i) GP variance after fusing Figure 4.2c.

Figure 4.2: KF update using YOLO output at different heights on a GP map.

The posterior density $p(\zeta|z, X) \propto p(z|\zeta, X) \times p(\zeta|X) \sim GP(\mu^+, P^+)$ is computed directly by applying the KF update equations [88]:

$$\mu^+ = \mu^- + Kv \tag{4.2}$$

$$P^+ = P^- - KHP^-, \tag{4.3}$$

where $K = P^- H^T S^{-1}$ is the Kalman gain and $v = z - H\mu^-$ and $S = HP^- H^T + R$ are the measurement and covariance innovations. H is an $m \times n$ matrix representing a linear sensor model that intrinsically selects the FoV $\{\zeta_1, ..., \zeta_m\}$ observed through z, and R is a diagonal $m \times m$ matrix of the height dependent variances $\sigma^2_{s,i}$ associated with each measurement z_i. Therefore, H controls which cells in the map should be updated, and

R controls the influence of these measurement on map updation. If the image is captured at low altitudes, the associated uncertainty in measurement is lower from Equation 3.9, which has higher influence on the map updation. An image captured at high altitudes has a lower influence on the map updation due to the high uncertainty associated with the measurement. Each time a new measurement is registered, Equation 4.2 and 4.3 are used to sequentially fuse the measurement onto the GP map.

Figure 4.2 shows the KF update procedure using the YOLO output on the GP map. The environment is the same as Figure 3.11, with 2 human mannequins placed in it as shown in Figure 4.2a. Figure 4.2d and 4.2g shows the initial mean and variance of the GP. The UAV flies to a height of 5m and 15m to capture images which YOLO uses to generate Figure 4.2b and 4.2c. KF update is done to fuse the YOLO output to the intial GP map. The mean after KF update on the initial mean at 5m and 15m are shown in Figure 4.2e and 4.2f respectively. Similarly, the variances are shown in Figure 4.2h and 4.2i.

It can be observed from Figure 4.2e and 4.2f that the non-human detections are updated with low value (blue color), while the human detections are updated with a high value (red/green) color. This demonstrates, how our mapping strategy is succeeding in revealing the field after taking a measurement. However, it can be noted that the false positives and false negatives are also fused as such into the map. Since YOLO failed to find a human in Figure 4.2c, the corresponding location is updated as non-human in Figure 4.2f. Therefore, looking only once at a particular location is not a good strategy towards building an accurate map.

It can be observed from Figure 4.2h and 4.2i that the variance decreases as more measurements are taken. However, a slight difference can be noticed in the amount of variance reduction. The variance reduced to a lower value in Figure 4.2h (darker blue color), while it is decreased to a slightly higher value in Figure 4.2i (lighter blue color). This is because of our height-dependent sensor model given in Equation 3.9. A measurement made at a lower height reduces the uncertainty or variance of the map more than that can be reduced at a higher height. The obstacle cells are not updated in Figure 4.2f and 4.2i, although they are found in Figure 4.2c. This demonstrates that our FoV Algorithm 1 works as expected by not updating the obstacle cells and the cells behind it.

This section demonstrated the mapping strategy used to fuse the measurement onto the GP map, considering the measurement uncertainty and the FoV. In the next section, the mapping strategy to fuse the YOLO output onto a OG in a probabilistic manner using Bayesian inference will be discussed.

BAYESIAN INFERENCE

This section aims at defining the mapping strategy for OG in a probabilistic manner so that the YOLO output can be fused into the map. The standard OG mapping strategy [97] uses Bayesian inference to fuse the sensor data onto a OG map in a probabilistic manner. For each cell $m_i \in M$ in the FoV at time t, we perform a log-likelihood update, given an observation z from YOLO, as:

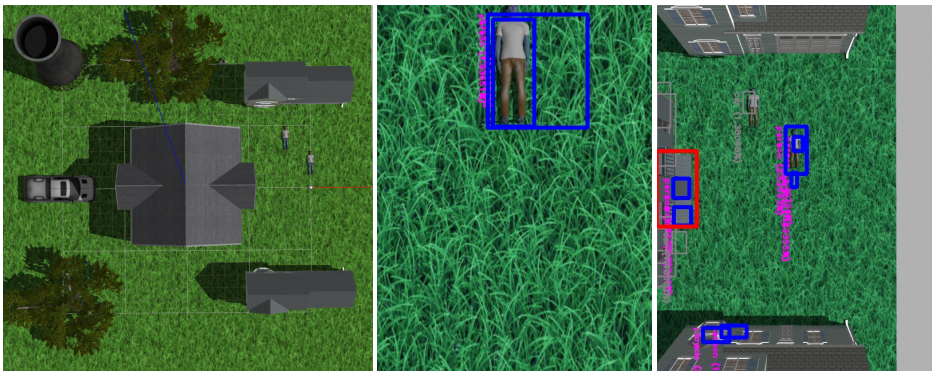
$$l(m_i|z_{1:t}, x_{1:t}) = l(m_i|z_{1:t-1}, x_{1:t-1}) + l(m_i|z_t, x_t) - l(m_i), \tag{4.4}$$

where the first term in the right hand side is the recursive term, the second term is the log-likelihood inverse sensor model and the third term is the map prior. The inverse sensor model is the log odds of the probability $P(human = 1|z = z)$ and is given by Equations 3.13 and 3.14. If YOLO detects a human in one of the cells in the FoV, the corresponding cell is updated with the probability $P(human = 1|z = 1)$ corresponding to the UAV altitude using Equation 3.13, for the inverse sensor model. If YOLO detects no human in the cell, the corresponding cell is updated with the probability $P(human = 1|z = 0)$ using Equation 3.14 for the inverse sensor model. After taking repeated measurements at the same location, the map is expected to converge to the ground truth.

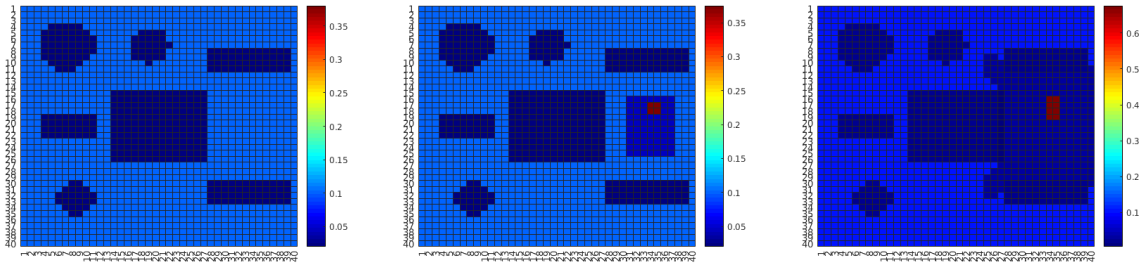
Figure 4.3 shows the Bayesian update of the OG using YOLO outputs at different heights. The OG is initialized with a prior probability of 0.1, assuming that the field doesn't have any humans. It can be observed from Figure 4.3e and 4.3f that the YOLO update is correctly fused into the map. The entropy of the map is decreased after mapping, in the regions where no humans are detected. This is due to the increasing confidence of not having a human in the cell, given that our prior was a probability of 0.1. However, the entropy increases at the region where a human is detected. This can be accounted for a higher human occupancy in a cell where our initial belief was to not have a human. This dilemma increased the entropy at those locations in Figure 4.3h and 4.3i. Since the objective is to decrease the entropy of the map so that the uncertainty is minimized, multiple measurements at those locations are necessary.

A clear distinction can be observed between the map update followed by OG and GP by comparing Figure 4.2e and 4.3e. The surrounding cells of the detected humans are also updated as humans in a continuous manner in Figure 4.2e, while only those cells where a human is detected is updated in 4.3e. This can be

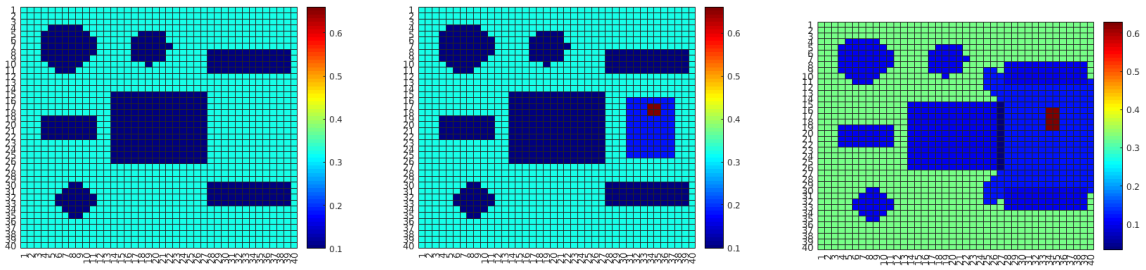
attributed to the interdependency and independence of the cells in GP and OG respectively. Therefore, both the mapping strategies work as expected.



(a) Environment with 2 humans. (b) YOLO at height = 5m. (c) YOLO at height = 15m.



(d) Initial OG occupancy probability. (e) OG probability fusing Figure 4.3b. (f) OG probability fusing Figure 4.3c.



(g) Initial OG entropy. (h) OG entropy after fusing Figure 4.3b. (i) OG entropy after fusing Figure 4.3c.

Figure 4.3: Bayesian update using YOLO output at different heights on a OG.

This chapter dealt with the mapping strategies followed by the GP and OG fields in fusing the sensor measurements sequentially. The maps were updated in a probabilistic manner by taking into account the uncertainty of the measurement, on both the OG and GP. The concepts detailed in this chapter will be used to plan the most informative path for the UAV in the next chapter. The next chapter deals with planning a 3D path to find all the human victims in the field in the least possible time.

5

PLANNING

The objective of this chapter is to introduce a path planning algorithm such that the resultant path is the most informative one to find humans in the field, in the least possible time. Moreover, it should be collision free, and should cater to the dynamics of a smooth and jerk free path. This chapter treats this problem as an optimization problem with information gain, coverage and collision avoidance as the objective, and flight time as the constraint. Section 5.2 deals with converting the problem into an optimization problem, while Section 5.1 deals with parameterizing the polynomial path to be planned. The initial solution required for the optimization is evaluated in Section 5.4. The uncertainty reducing objective function that is used for the optimization process is discussed in Section 5.3. This section introduces two informative planning strategies: one for continuous mapping using a GP, and the other for discrete mapping using a OG.

TRAJECTORY PARAMETRIZATION

A polynomial trajectory ψ is parameterized by a sequence of N control waypoints to be visited by the UAV, defined as $C = [c_1, ..., c_N]$, where the first waypoint c_1 represents the current UAV location. The polynomial trajectory connects these control points using $N-1$ k-order spline segments for minimum-snap dynamics as given in [98]. The trajectory ψ represents the fixed-horizon path for the UAV for data gathering. A sample trajectory in a complex obstacle filled environement from Figure 3.5a is shown in Figure 5.1. Depending on the measurement frequency of the sensor, the measurement locations along the path ψ can be evaluated, based on which the predictions about the informativeness of the path can be estimated.

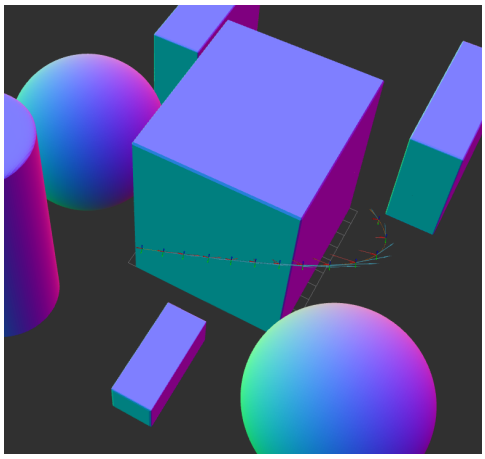


Figure 5.1: A sample minimum-snap polynomial trajectory in $Rviz$. The path avoids all obstacles.

PLANNING AS AN OPTIMIZATION PROBLEM

The definition of the standard IPP problem given in Equation 1.1 requires an information theoretic objective function for optimization. However, we require an additional term for collision avoidance in the objective function. Therefore, we reformulate the objective function as,

$$P^* = \underset{P \in \psi}{\arg\max} \frac{k_1 O_{informative} + k_2 C_{collision}}{t_{flight}}, \tag{5.1}$$

where $O_{informative}$ is the information-theoretic objective, t_{flight} is the flight time, $C_{collision}$ is the penalty for collision defined in Section 3.3.3, and k_1 and k_2 are scaling constants. It should be noted that $C_{collision}$ is always non-positive and hence it reduces the objective, while the information gain is always positive and hence always increases the objective. Therefore, the objective function is maximum when $C_{collision}$ is zero, $O_{informative}$ is the maximum, and t_{flight} is the minimum.

Algorithm 2 Function to solve the IPP problem for target search using a UAV.

```
1: function IPP_SaR()
2:     Create ESDF map
3:     Initialize the field
4:     if not landing condition then          ▷ Replan until the map cannot be further improved.
5:         C = OptimizePath(R_0, ESDF)
6:         Fly along the path
7:         Take measurements
8:         Update map at measurement locations
9:         Update R_0
```

The objective function can be optimized by standard state-of-the-art optimizers like the CMAES or Bayesian Optimization detailed in Section 2.3.3, such that it results in a 3D collision free, polynomial path as described in Section 5.1. It is important that these optimizers can handle constrained optimization such that the resulting path completely lies inside the environment. We use a two step optimization technique, where a suboptimal finite horizon greedy intial solution is evaluated and is used as an initialization point for the global optimizers. Algorithm 2 describes how the IPP problem for search and rescue is solved, while Algorithm 3 demonstrates our optimization strategy. Section 5.4 describes the coarse layered search that is used to evaluate a greedy initial solution for the optimization.

Algorithm 3 Function to perform optimization.

```
1: function OPTIMIZEPATH(R_0, ESDF)
2:     do
3:         C_initial = GreedyLayeredSearch(R_0, ESDF).     ▷ Greedy layered search
4:         C = CMAES(C_initial, ESDF)                      ▷ Black-box optimization
5:     while not isCollisionFreePath(C, ESDF)
6:     return C
```

UNCERTANITY REDUCTION AS THE OBJECTIVE

Many IPP approaches use uncertainty reduction as the objective to solve the IPP problem [12, 13]. This is inspired by the concept that the less uncertain we are about our map estimate, the better the output map. The information gain in these cases is the net uncertainty reduction of the map, assuming that a measurement is made at a location. For a GP map, the information gain is defined as the reduction in variance of the map:

$$O_{informative} = Tr(P^-) - Tr(P^+), \tag{5.2}$$

where $Tr(.)$ denotes the trace of the matrix, P^- is the prior covariance, and P^+ is the posterior covariance given by Equation 4.3. For a OG map, the information gain is defined as the reduction in entropy of the map:

$$O_{informative} = H(M^-) - H(M^+), \tag{5.3}$$

where $H(M^-)$ and $H(M^+)$ are the prior and posterior entropy of the map, which was defined in Equation 3.6.

Algorithm 4 Function to check if the path is collision free.

```
1: function isCollisionFreePath(C, ESDF)
2:     sampled_points = SampleFromTrajectory(C, sampling_frequency)
3:     for each point in sampled_point do
4:         if ESDF(sampled_point) < r_UAV then
5:             return false
6:     return true
```

COARSE LAYERED GREEDY SEARCH

This section deals with evaluating an initial path for the standard optimizers. We follow a greedy/myopic strategy where we evaluate the next best viewpoint $best\_pose$ by randomly sampling the viewpoints R_{rand} from different horizontal planes, and simulating a predicted map update for a measurement at that viewpoint. For a GP, the predicted map update is done such that the mean is kept constant, and the covariance is updated as per Equation 4.3, according to the sensor model at that height given by Equation 3.9. This follows the assumption that the best estimate of what we are going to observe next is actually the GP mean at that location. However, the uncertainty at that location would be reduced according to the sensor model. For an OG, the map is updated depending on the current occupancy of the cell. The expected value of a Bernoulli distribution is the occupancy probability of the cell, as mentioned in Section 1.3. If a cell has an occupancy probability of more than 0.5, the predicted map update is done as per the equation 4.4, with the sensor model given in Equation 3.13, assuming that a human is going to be detected. While the cell is updated with the sensor model given in Equation 3.14, if the occupancy probability is less than 0.5. This follows the assumption that the best estimate of what is going to be observed is the current occupancy probability itself. The update prediction is conditioned on the most probable value, given the current state of the cell.

Algorithm 5 Function to perform the greedy search.

```
1: function GreedyLayeredSearch(R_0, ESDF)
2:     Initilize empty queue C_initial.
3:     Insert R_0 into C_initial.
4:     for i = 1 : N − 1 do
5:         next_best_viewpoint = NextBestViewPoint(C_initial[i], ESDF)
6:         Insert next_best_viewpoint into C_initial.
7:         Update map at next_best_viewpoint.
8:     return C_initial
```

Equally spaced n_{layer} number of horizontal planes are defined from which n_{rand} number of points R_{rand} are randomly selected such that all those points are directly visible from the simulated current UAV pose $R_{current}$. This can be achieved by adapting the algorithm 1 such that instead of ground visibility, the visibility of R_{rand} from $R_{current}$ can be checked. A simulated map update is performed on all these viewpoints, and the information gain $O_{informative}$ associated with the measurement is calculated for all the points. The information gain could be a decrease in variance for a GP map as given in Equation 5.2, or a decrease in entropy for an OG map as given in Equation 5.3. The viewpoint that gives the best $O_{informative}$ out of all the n_{rand} randomly chosen points is chosen as the next best viewpoint. Algorithm 6 shows how the next best viewpoint can be evaluated.

The best viewpoint thus calculated is then inserted into the queue of initial control points $C_{initial}$, where the first member of the queue is the current pose of the UAV. The process is iterated until all N initial control points are evaluated. It should be noted that the fourth best viewpoint is evaluated on the map that has already been sequentially updated with the predicted map update at the third and second best viewpoints in $C_{initial}$. This makes our planner a fixed horizon planner, where we take into account all the possible measurements that could sequentially update the map during the flight. The algorithm to perform the layered greedy search is described in Algorithm 5.

This section dealt with evaluating the initial control points for a myopic solution that can initialize the standard optimizers. These points effectively represent a polynomial path presented in Section 5.1, and provides a greedy solution to the IPP problem. The sampling based coarse search helps in quickly searching the visible environment and reaching a myopic solution. This aids the performance of the optimizer in fine tun-

Algorithm 6 Function for greedy next best global viewpoint selection.

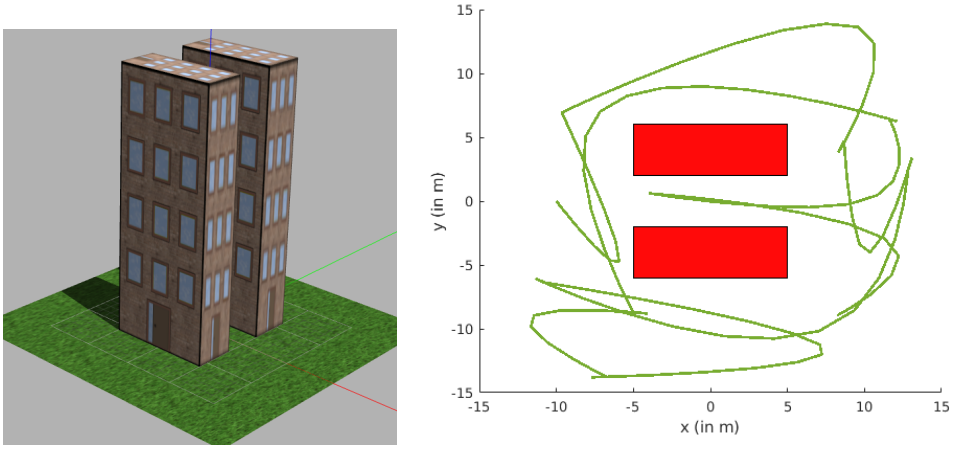
function NEXTBESTVIEWPOINT($R_{current}$,ESDF)
  Initialize $max\_objective$ to 0.
  Initialize $best\_pose$ to $R_{current}$
  Define n_{layer} equally spaced horizontal planes.
  for i=1 to n_{rand} **do**
    do
      $n_{plane} = i$ mod n_{layer} ▷ plane number
      Randomly sample a point R_{rand} from plane n_{plane}.
    while not isPointVisible(R_{rand},$R_{current}$, $ESDF$)
    Simulate a map update at R_{rand}.
    Calculate $O_{informative}$
    if $O_{informative}$>$max\_objective$ **then** ▷ if better viewpoint is found
      Set $max\_objective = O_{informative}$
      Set $best\_pose = R_{rand}$
  return $best\_pose$

ing the solution to evaluate a non-myopic solution. The next section deals with finding an optimal solution for the problem using standard optimizers.

PATH OPTIMIZATION

The aim of this section is to show that it is possible to solve the target search problem formulated in the thesis using state-of-the-art optimizers, by refining the initial solution evaluated by the coarse greedy search given in previous section. The objective function formulated in Equation 5.1 is however a very complicated objective function that is highly non-linear and is influenced by a number of factors like the sensor model, type of map, obstacle shapes and sizes, penalty function for collision avoidance, FoV at a location, minimum-snap dynamics of the path, initial greedy search solution, measurement frequency of the sensor, current state of the field, length of the horizon etc. Even the basic version of a discrete IPP problem is a hard optimization problem, which is often proposed to be at least NP-complete [18] or NP-hard [23]. This motivates the use of state-of-the-art black-box optimizers, that do not have many tuning parameters. Therefore, we use two state-of-the-art optimizers to solve the problem: CMAES and BO.

Algorithm 3 shows the two steps in optimizing the path. The output path from the optimizer is checked for possible collisions to ensure safety of the UAV. Algorithm 4 demonstrates how the collision check for a path works. The optimization is repeated in case it results in a path that is not collision free.

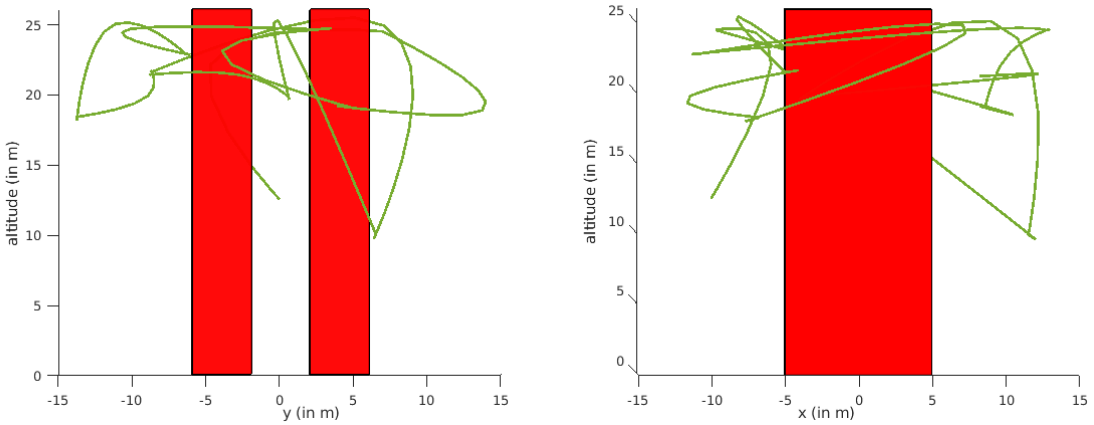


(a) Environment with two buildings.

(b) Top view of the path taken.

Figure 5.2: Paths planned by a UAV for a flight time of 130s in the environment in Figure 5.2a.

The algorithm was run on a UAV in the *Gazebo* environment with dimensions $30m \times 30m \times 26m$ as shown in Figure 5.2a. The UAV flight started at location $(-10, 0, 13)$ and lasted for 130s. A GP map was used for the field, and uncertainty reduction was used as the objective, which was then solved by the CMAES. A new path is planned once the UAV has completed the previously planned path and has taken measurements. These paths were recorded during the flight and was plotted in *Matlab*, which is shown in Figure 5.2b. It can be observed that all the planned paths for the UAV are collision free. Moreover, the paths adhere to the boundary constraints and do not cross the environment dimensions. The gap between the two buildings was also explored and mapped by the algorithm, demonstrating the effectiveness of the algorithm in searching for victims in between them as well. The paths are planned such that measurements can be taken at different accuracies. This behaviour arises from different tradeoffs that are handled by the optimizer within the planner. Multiple measurements at varying heights should be taken at the same location to increase the map accuracy. Figure 5.3a illustrates that the planned path considers different heights for data acquisition, depending on the current state of the GP field. Therefore, the influence of the height-dependent sensor model is clearly visible in the planned path.



(a) Front view of the path.

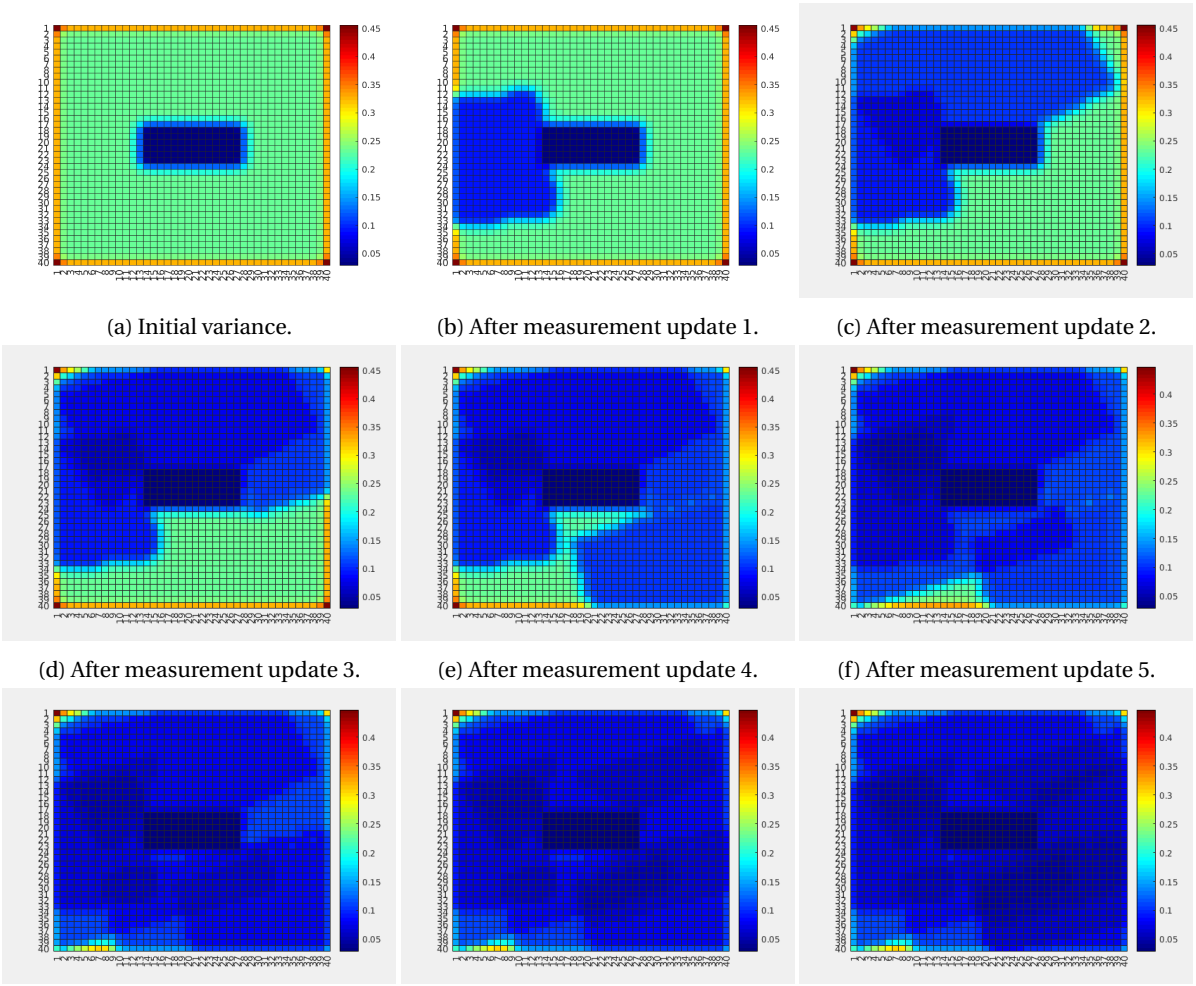
(b) Side view of the path.

Figure 5.3: 3D trajectories generated by the planner in the environment in Figure 5.2a.

Figure 5.4 shows the variance reduction by the IPP algorithm running on a UAV flying in the environement shown in Figure 5.5a. The obstacle is as tall as the maximum height of the environment so that the UAV can only fly around the obstacle and not above it. It can be observed that the variance of the map reduces as expected after each measurement. Moreover, no obstacle or boundary collisions were observed during the flight. Therefore, the optimizer successfully solves the IPP objective function and provides a path that leads to the reduction of uncertainty in the GP map. The output of the algorithm after a flight for 130s is shown in Figure 5.5b. It can be observed from Figure 5.5 that the humans were correctly detected by the algorithm, demonstrating the success of the framework.

The same algorithm was run with the OG map on the same environment, and the reduction in entropy of the map is shown in Figure 5.6. It can be observed that the entropy of the map decreases after each measurement, demonstrating the success of the IPP algorithm in planning an informative path using an OG map. The output map after a flight time of 130 seconds is shown in Figure 5.7b. It can be noted that all the humans in the map are correctly identified.

It can be observed from Figure 5.6 that the detected humans appear as high values in the entropy map. This is because the prior for the OG map is set as 0.1, assuming that there are no humans in the field. However, as new measurements are made, the occupancy probability increases from 0.1. This results in a net increase in entropy at those locations. Therefore, the optimizer is rewarded if more measurements are made at those locations where humans were detected. After several observations, the occupancy probability approaches 1, which results in a very low entropy, making those locations no longer interesting for the optimizer. This type of planning which is dependent on the measurements taken, is known as adaptive planning. Therefore, setting a low prior to the OG map, and optimizing the IPP objective for entropy reduction, makes the planner adaptive in nature during the first few measurements. The GP based planner is not adaptive as can be observed from Figure 5.4. The detected humans do not influence the objective function. It tries to reduce the

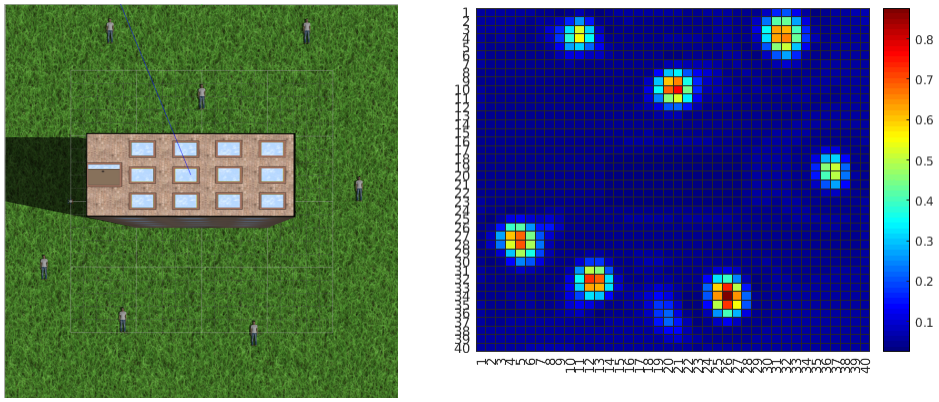


(a) Initial variance. (b) After measurement update 1. (c) After measurement update 2.

(d) After measurement update 3. (e) After measurement update 4. (f) After measurement update 5.

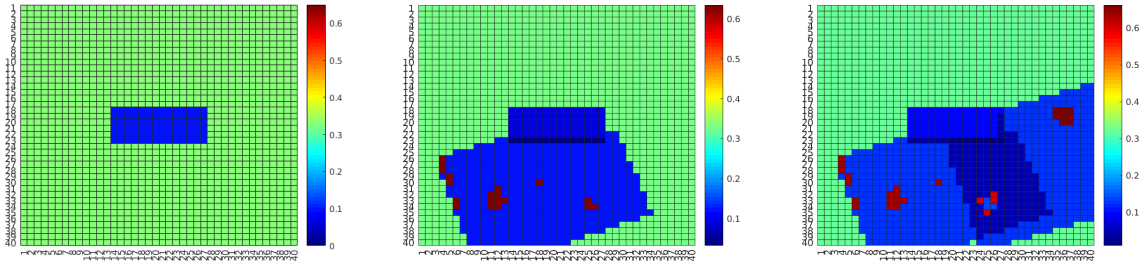
(g) After measurement update 6. (h) After measurement update 7. (i) After measurement update 8.

Figure 5.4: Variance reduction of the map at different stages of the flight.

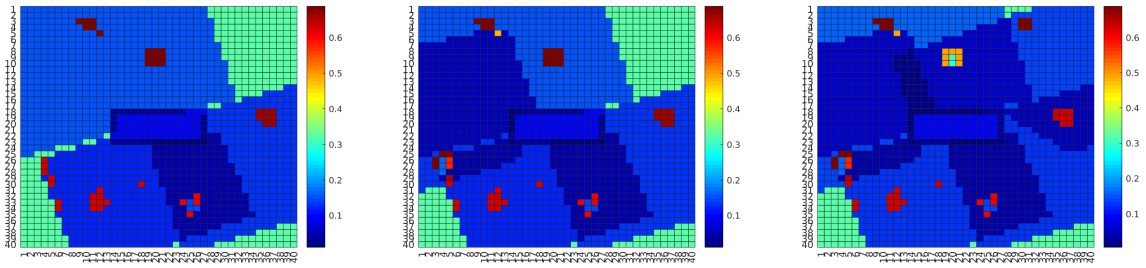


(a) Environment with 7 scattered men. (b) GP map after a flight for 130 seconds.

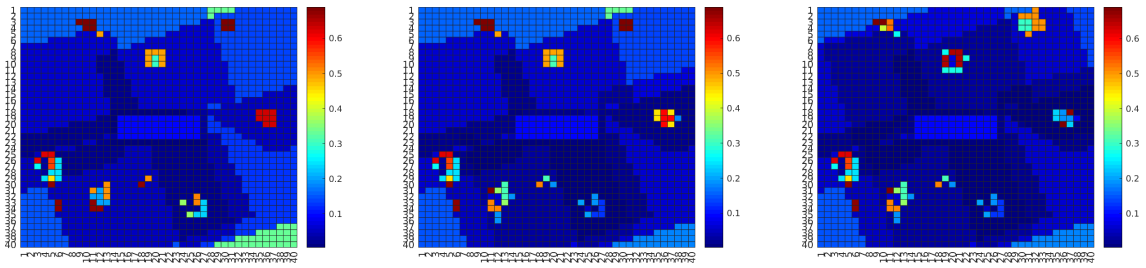
Figure 5.5: Comparison of the output GP map from the IPP planner with the real environment.



(a) Initial entropy. (b) After measurement update 1. (c) After measurement update 2.

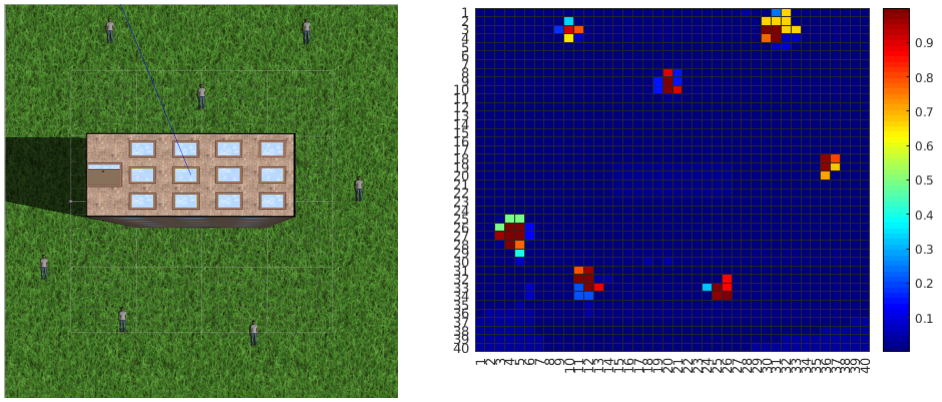


(d) After measurement update 3. (e) After measurement update 4. (f) After measurement update 5.



(g) After measurement update 6. (h) After measurement update 7. (i) After measurement update 8.

Figure 5.6: Entropy reduction of the map at different stages of the flight.



(a) Environment with 7 scattered men. (b) OG map after a flight for 130s.

Figure 5.7: Comparison of the output OG map from the IPP planner with the real environment.

uncertainty of the field without considering the detected humans while planning the next path. However, as discussed in Section 2.1 an adaptive planner demonstrates better performance than a non-adaptive planner. In the next section, a layered optimization approach is introduced, which considers the measured detections for planning the next path.

LAYERED OPTMIZATION FOR ADAPTIVE PLANNING

Consider a scenario where the victims are located in a specific area of the field, instead of being scattered around like in Figure 5.5a. The GP based planner that aims to reduce the variance would cover the area by looping around the environment multiple times. However, this is not an efficient strategy for search and rescue operations, where flight time is of prime importance. The planner should instead aim to cover the environment once, and then allocate rest of its flight time to search intensively in the region where humans were detected. This is known as an exploitation strategy. However, the planner should not concentrate on looking at the detections by avoiding the unexplored regions. This results in possibly missing out some victims from being detected. Therefore, there should be a fine balance between exploration and exploitation strategies within the planner. The variance reduction is a pure exploration strategy, which should be blended with an exploitative strategy for efficient performance. Therefore, a layered optimization approach is introduced in this section that balances the exploration-exploitation tradeoff. In this approach, the path is optimized using the CMAES, which simulataneously optimizes the field by using BO.

In BO, the optimal solution of an unknown and costly to evaluate objective function is evaluated by building a surrogate model of the objective function. The next location in the search space to sample is then identified by optimizing this surrogate model by using standard optimizers [99]. This process is repeated until the model of the objective function is built with satisfactory uncertainty. The surrogate model in literature is known as an acquisition function. The advantage of using an acquisition function is that it is much easier to compute and optimize, than the objective function itself. Moreover, the acquisition functions have a tuning parameter that can be tuned for the application at hand for exploration-exploitation tradeoff. The most common acquisition functions are:

- Probability of Improvement (PI) [100]

$$PI(x) = \Phi\Big(\frac{\mu(x) - g(x_{best}) - \zeta}{\sigma(x)}\Big); \tag{5.4}$$

- Expected Improvement (EI) [100]

$$EI(x) = \sigma(x)[Z\Phi(Z) + \phi(Z)], \tag{5.5}$$

where

$$Z = \frac{\mu(x) - g(x_{best}) - \zeta}{\sigma(x)}$$

- Upper confidence bound (UCB) [101]

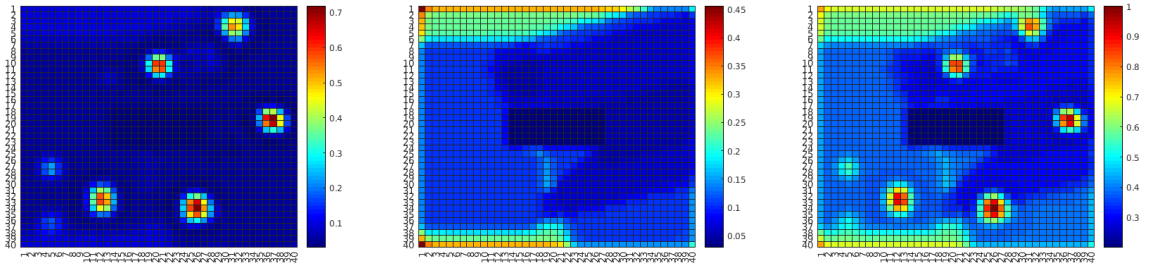
$$UCB(x) = \mu(x) + \kappa\sigma(x), \tag{5.6}$$

where x_{best} is the location of the best sample gathered so far, ζ, κ are exploration-exploitation tuning parameters, ϕ is the normal probability density function and Φ is the normal cumulative distribution function. If the variance is lower than the required level of certainty for a measurement, the aquisition is set to zero.

The acquisition function, therefore, is capable of binding the mean and variance of the GP, into a single function. The idea is to calculate the acquisition function of the field using the mean and variance of the GP field, and use this as the information theoretic objective for IPP. This approach would give an acquisition function which is high at those areas where the mean is high, and also where the variance is high. The UAV should focus not only on areas with high uncertainty, but also regions with high mean. This idea is illustrated in Figure 5.8, where the mean is given in Figure 5.8a and the covariance in Figure 5.8b, while the corresponding UCB acquisition function is given by Figure 5.8c. It can be observed that the acquisition function contains information about both the mean and the variance.

A UAV should plan a path such that it observes the region with highest value of aquisition function. This measure is quantized by introducing a term acquisition view (AV), which is defined as the sum of all acquisition values within the FoV of the UAV, formulated as:

$$AV = \sum_{x \in FoV} UCB(x). \tag{5.7}$$



(a) GP mean. (b) GP variance. (c) UCB acquisition function.

Figure 5.8: The UCB acquisition function that binds GP mean and variance into one function.

However, maximizing the AV would result in a strategy where the UAV prefers to stay at the maximum height possible so as to increase the FoV. The preference for a certain flying height for the best sensor performance, shown in Figure 3.13 is exploited to evaluate the information-theoretic objective function:

$$O_{informative} = AV \times \left[\frac{1}{\sigma_1 \sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{h-h_{opt}}{\sigma_1}\right)^2} \right], \tag{5.8}$$

where h is the altitude, h_{opt} is the height of best sensor performance, and σ_1 is the standard deviation of the curve in Figure 3.13 assuming that it is an approximate normal distribution centred around h_{opt}.

Let θ_1 and θ_2 be the FoV angles of the camera. The dimension of the rectangular FoV of the camera at a height h, assuming that there are no obstacles, is $2h\tan\frac{\theta_1}{2}$ and $2h\tan\frac{\theta_2}{2}$. Therefore the area of the FoV is given by:

$$Area(FoV) = 4h^2 \tan\frac{\theta_1}{2} \tan\frac{\theta_2}{2} \tag{5.9}$$

Assuming a constant unit acquisition function in the FoV, $Area(FoV)$ can be considered as the AV of a uniformly discretized grid. Substituting Equation 5.9 in 5.8 and ploting the result with respect to h, for $h_{opt} = 10m$, $\sigma_1 = 7$, $\theta_1 = \frac{\pi}{4}$ and $\theta_2 = \frac{\pi}{3}$ results in the curve shown in Figure 5.9.

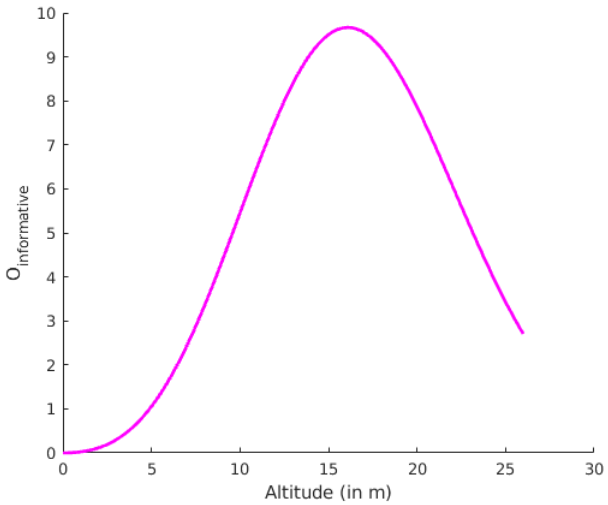


Figure 5.9: Information-theoretic objective using acquisition function for a simple example.

It can be observed from Figure 5.9 that the objective is maximized at a height different from that of h_{opt}. There is a fine trade-off between the sensor performance and the FoV that results in a higher preferred height. Therefore, the new objective function based on acquisition function accounts for the sensor model and prefers a particular height for data acquisition that balances sensor performance and FoV. Although, the example under consideration is a simple one without considering any obstacles, and with a uniform acquisition function, it does expose the idea of a preferred height for the planner for information gathering.

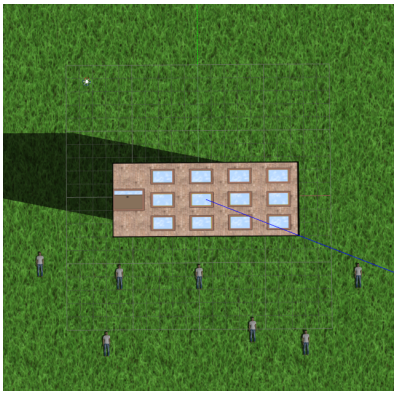
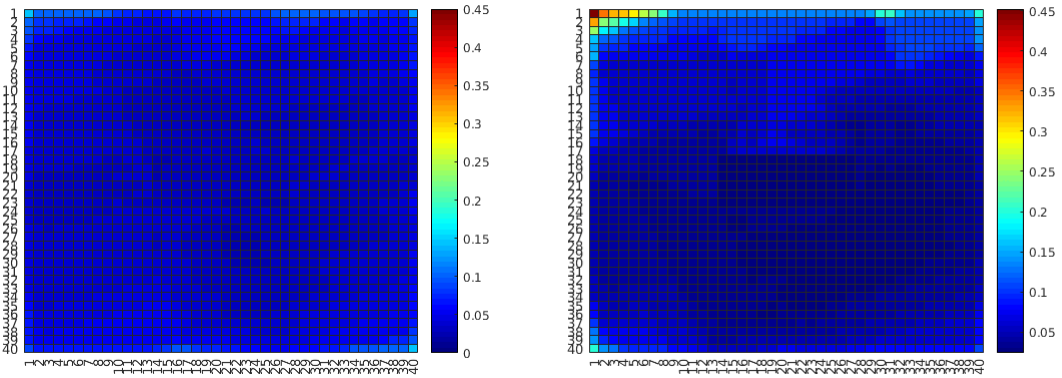


Figure 5.10: Environment with 7 men placed on one side of the field.

Both planners, one adaptive planner based on acquisition function as the objective, while the other one a non-adaptive planner based on variance reduction as the objective, were tested on the environment shown in Figure 5.10. The variances after 120s of flight are shown in Figure 5.11. Figure 5.11a and 5.11b show the GP variance for non-adaptive and adaptive planner respectively. Both planners successfully found all 7 victims in the field, but with different uncertainties about their detections. It can be observed that the resultant variance is mostly uniform in Figure 5.11a, while in Figure 5.11b, the variance at locations where the humans were detected is the minimum and the variance at locations where there is no human is the maximum. The planner with variance reduction objective was covering the field repeatedly without considering the detections into planning, while the planner with acquisition based objective scanned the environemnt once and then continued scannning those region where the humans were detected. This adaptive strategy in path planning resulted in a very low variance at locations where there are humans. At the end, the acquisition based planner is more certain about all the human detections than the variance based planner. The real life benefits of an adaptive planner is that the target detections are more reliable, increasing its employability in wide range of applications. It would be of particular importance where the decisions made based on the output map of the planner has high consequences. A terrorist breach at the international border detected by an IPP algorithm has high political consequences, and hence demands a highly accurate detection.

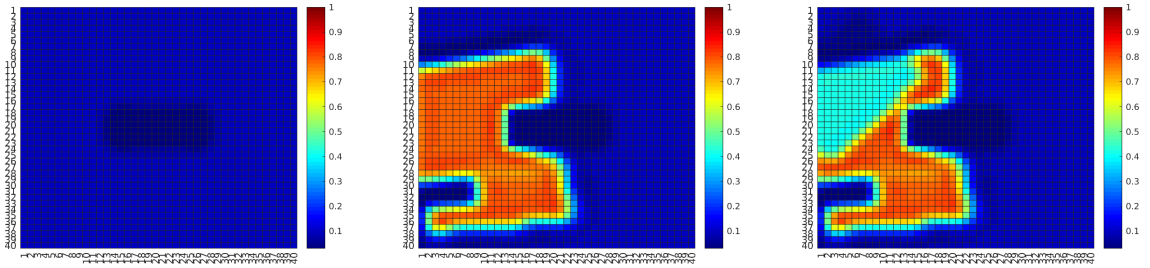


(a) Variance reduction as the objective.

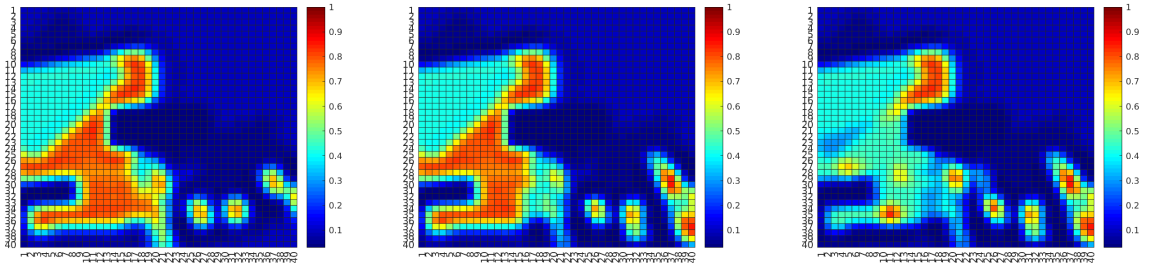
(b) Acquisition based objective.

Figure 5.11: GP variance after 210 seconds of flight with different information objective.

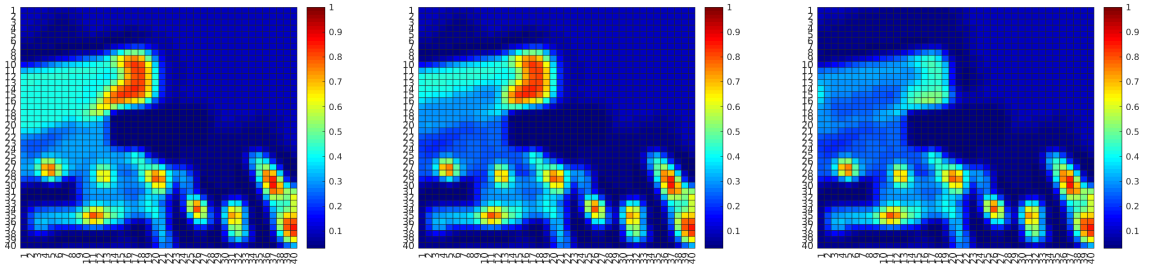
An adaptive strategy is particularly beneficial, considering the robustness of the path planning algorithm. After a false positive detection, an adaptive strategy tends to repeat measurements at those locations to confirm the presence of the human. The planner takes multiple measurements on all the detections. The filtering out process discussed in Section 3.4.1 in which large human detections were ignored, was removed to demonstrate the robustness of the algorithm under a poor sensor. Without the filter, large detections are allowed from the YOLO, resulting in very poor sensor performance. However, the planner is modular and can handle changes in the sensor model. The parameters of the sensor model were updated and the simulation



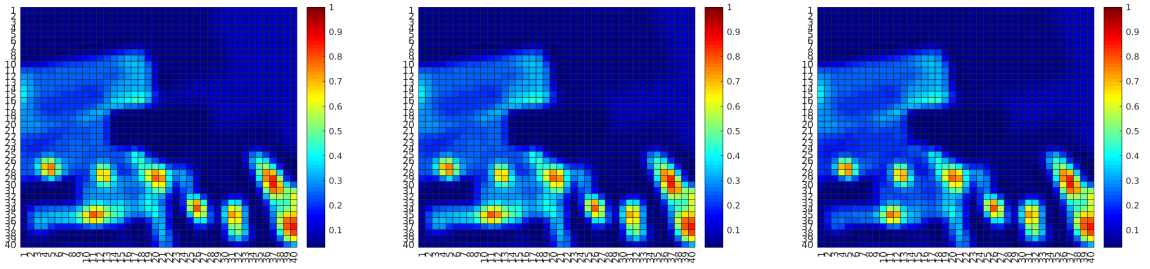
(a) Initial GP mean. (b) After measurement update 1. (c) After measurement update 2.



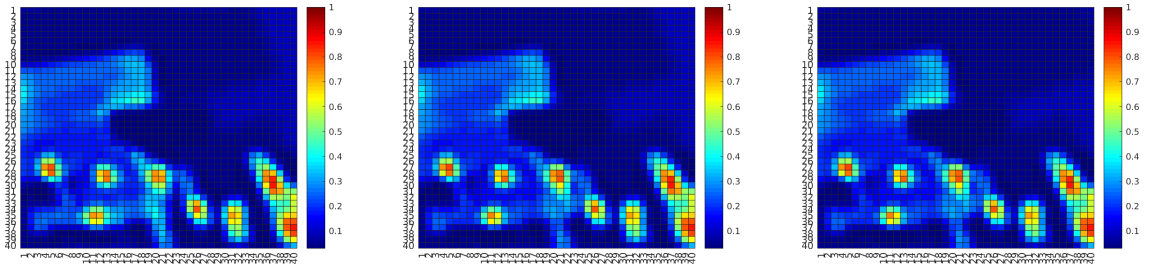
(d) After measurement update 3. (e) After measurement update 4. (f) After measurement update 5.



(g) After measurement update 6. (h) After measurement update 7. (i) After measurement update 8.



(j) After measurement update 9. (k) After measurement update 10. (l) After measurement update 11.



(m) After measurement update 12. (n) After measurement update 13. (o) After measurement update 14.

Figure 5.12: Robustness demonstrated by the acquisition based adaptive IPP planner, in case of false positive detections from a poorly performing sensor.

was run with the adaptive planner on the environment shown in Figure 5.10. Figure 5.12 shows the GP mean after each measurement update. It can be observed that a large false positive detection was made in Figure 5.12b. However, repeated sampling on the detected location, enabled by the adaptive sampling strategy, helped in recovering from the erroroneous map update. Figure 5.12o shows the GP mean after a flight time of 150 seconds. The map has almost recovered from the large false positive detection, while managing to successfully detect all humans in the field. However, a poor sensor brings new false positives that the planner tries to combat through repeated measurements. Therefore, the adaptive strategy using acquisition function for planning is more advantageous than a non-adaptive strategy in a search and rescue operation.

This chapter dealt with the core path planning algorithm used. The IPP problem was formulated as an optimization problem, which was then solved using the CMAES and was validated. The possibility of uncertainty reduction as the objective was explored in both continuous and discrete mapping using variance reduction and entropy reduction respectively. A layered optimization strategy was then introduced for adaptive planning to balance the exploration-exploitation tradeoff. The layered optimization that uses BO surrogates to optimize the field, and the CMAES to optimize the path was found to result in a map with better confidence on all human detections. The robustness of an adaptive strategy for planning was illustrated by using a poor human sensor and showing that taking repeated measurements on the detections helps to eliminate the false positive detections from the map. The next chapter deals with benchmarking the algorithm against state-of-the-art planners by running multiple simulations with varying experimental conditions of realistic scenarios.

6

SIMULATION RESULTS

The previous chapter presented the theory for the proposed IPP planner, while the algorithm is evaluated in this chapter. This chapter deals with testing the planner against different environmental conditions and benchmarking the planner against other state-of-the-art planners in a realistic scenario.

FLIGHT PARAMETERS

The environment chosen by each set of experiment is defined at the beginning of each section. The following flight parameters can be assumed for the experiments unless another parameter value is explicitly mentioned. The flight parameters include a reference speed [98] of $5m/s$ and a reference acceleration [98] of $3m/s^2$ with 3 control points defining the parametrized trajectory. The sensor model given in Equation 3.9 is used for the GP mapping with $A = 1$ and $B = 0.05$, while the one given in Equation 3.13 and 3.14 is used for the OG mapping with $h_{sat} = 25.5m$, $P_{max} = 0.95$, $P_{min} = 0.8$, $h_{opt} = 15m$ and $\sigma_1 = 20$. A minimum flight height of 2m and maximum flight height of 26m were set. The field is of size $30m \times 30m$, discretized such that the dimension of each cell is $0.75m \times 0.75m$. Each flight trial starts at the location $(-10m, 0m, 13m)$ in the map. The FoV of the camera is $(45°, 60°)$. The measurement frequency of the sensor is set to a low value of 0.15Hz to demonstrate that the algorithm works well even with a low measurement frequency.

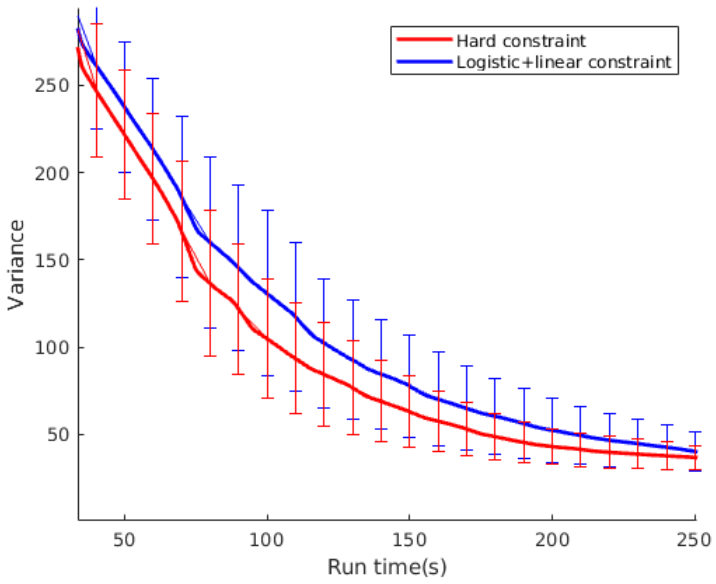


Figure 6.1: Influence of different penalty functions on uncertainity reduction of GP field.

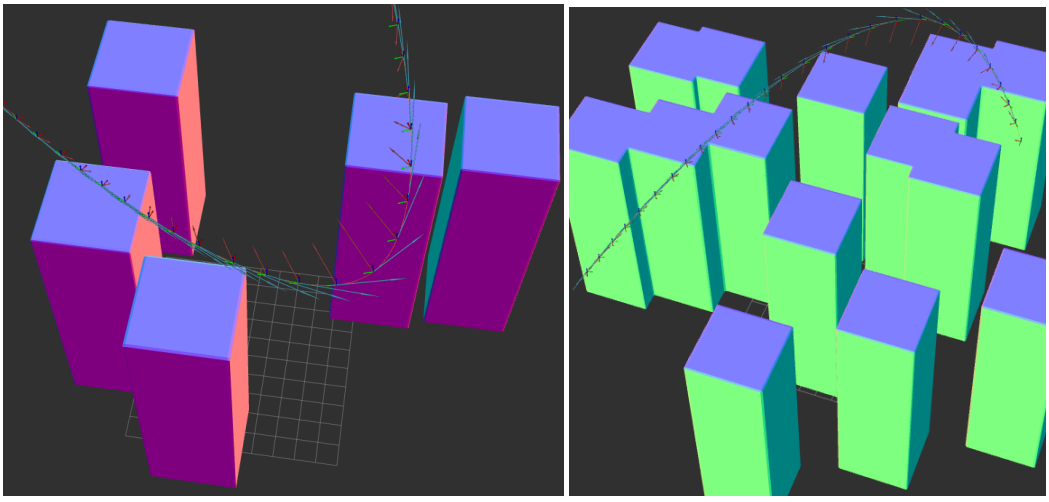
PENALTY FUNCTION

The two types of potential fields as defined in Section 3.3.3 were considered for collision avoidance. This section deals with finding the most suitable penalty function that can be applied for collision avoidance so that the CMAES can deliver the best path in the least amount of time. Two sets of simulations, each containing 100 flight trials and lasting 250s were performed on the environment shown in Figure 5.10, using uncertainty reduction as the objective. The difference between two sets of simulations is the penalty function used; one uses a hard constraint given by Equation 3.7, while the other uses logistic constraint given by Equation 3.8. Uncertainty in the GP map was recorded and plotted against run time, and is shown in Figure 6.1.

It can be observed from Figure 6.1 that the CMAES is capable of solving the IPP problem computationally faster by using a hard constraint as the collision penalty function. This might be because of the fact that a complicated penalty function might be computationally expensive for the CMAES to optimize, when compared to a simple hard constraint penalty function. Both penalty functions successfully planned collision free paths, although they differ slighly in their computation time. The uncertainty for hard constraint drops faster than that of the logistic constraint. Moreover, the average standard deviation of the 100 variance vs run-time plots for hard constraint is 18.864, compared to 24.473 for logistic constraint. The mean and standard deviation of the hard constraint penalty function are lower than that of the logistic function. Therefore, the hard constraint was selected to model the collision penalty in the planner.

ENVIRONMENT COMPLEXITY

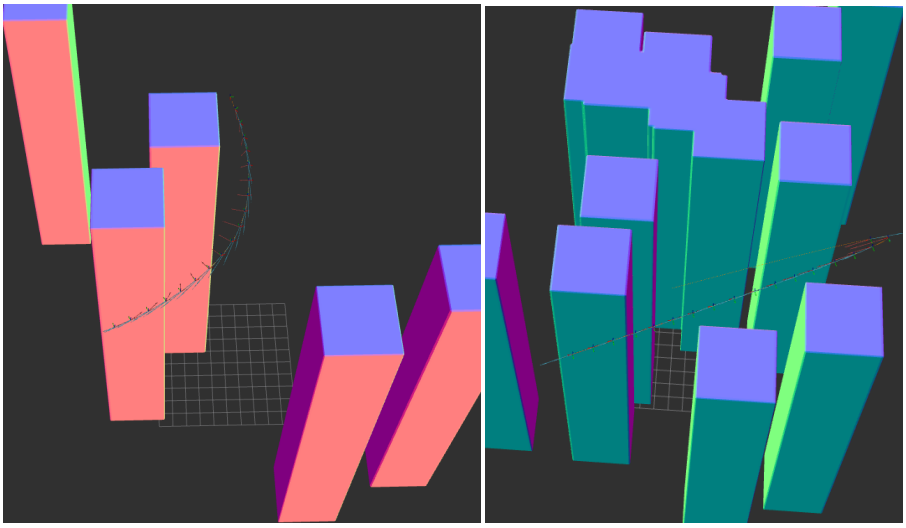
In previous chapters, different environments were used to demonstrate the planning algorithm. The planner should be capable of operating in different environment complexities. Two types of urban infrastructure is considered: Delft being a small city would be a simpler environment compared to Rotterdam, which has closely packed high-rise buildings. These real life scenarios are bought into the simulation environment through a simple and a complex environment. A simple environment has low obstacle density and consists of obstacles with small height, that do not heavily obstruct the UAV flight. While a complex environment would contain high rise buildings, packed in the environment with a high density. It complicates the obstacle avoidance task and limits the FoV. This results in a longer flight time to build a map with sufficient accuracy. This section deals with analyzing the performance of the planning algorithm with respect to different environment complexities.



(a) 5 small obstacles at random locations. (b) 15 small obstacles at random locations.

Figure 6.2: Planned path in an environment with obstacle height=13m and environment height=26m.

Two types of environments are considered: one with obstacles of height 13m and the base of dimension $4m \times 4m$, and the other with obstacles of height 26m and base dimensions $4m \times 4m$. The UAV is able to fly over the obstacles in the first environment, while not in the second environment. The planned path in the first environment is shown in Figure 6.2, while on the second environment is shown in Figure 6.3. It can be observed that the UAV prefers flying over the obstacles due to better collision avoidance and better FoV in Figure 6.2, while it flies in between the obstacles in Figure 6.3. Different obstacle densities were simulated to



(a) 5 big obstacles at random locations. (b) 15 big obstacles at random locations.

Figure 6.3: Planned path in an environment with obstacle height=26m and environment height=26m.

test the algorithm. The locations of the obstacles are randomly selected in each test flight. Figure 6.4 shows the average result along with the error bars for 100 flights in each scenario. It can be observed that as the number of obstacles increases, the initial variance decreases. This is due to the assumption that humans are not present on the ground where there is an obstacle. Therefore, the initial variance of the field decreases with the number of obstacles. It can be observed from Figure 6.4 that all the environments result in approximately the same variance after a run time of 300s. This is due to the fact that the UAV flies over the buildings and increases its FoV, which helps in quickly reducing the variance.

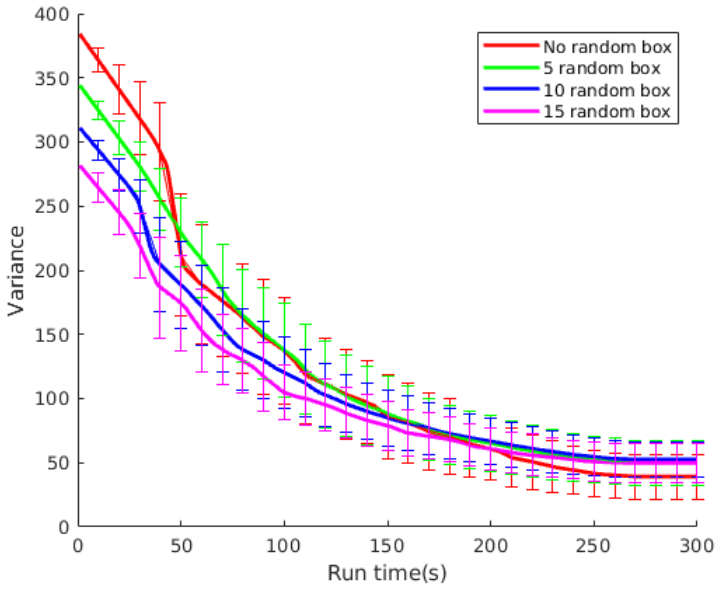


Figure 6.4: Effect of obstacle density on an environment with small buildings.

Similar experiments were conducted on high-rise buildings that are randomly placed on the field before each test flight. 100 such test flights were performed and the results are shown in Figure 6.5. Similar to Figure 6.4, the initial variance decreases with the number of obstacles. However, as the number of obstacles increases, the planner finds it increasingly difficult to decrease the variance of the field beyond a certain limit. The main reason for it is the restricted FoV as the UAV cannot fly over the obstacles. Moreover, certain

regions in the environment remain completely inaccessible for mapping, depending on the complexity of the environment.

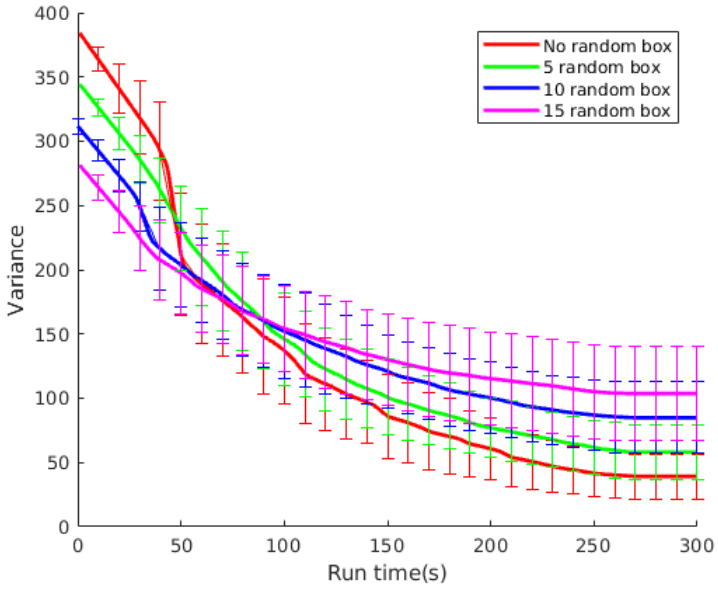


Figure 6.5: Effect of obstacle density on an environment with big buildings.

This section dealt with analyzing the performance of the algorithm with respect to increasing environment complexity. The results of the simulated test flights under various test conditions illustrate that the algorithm demonstrates similar performance after some time if the environment consists of medium height obstacles over which the UAV can fly. However, a decreasing performance was observed as the the environment complexity increases if the obstacle heights are too high. This section therefore validates the algorithm in randomly selected environments of varying complexity, demonstrating that it is applicable not just for a few selected environments. The competitiveness of the algorithm compared to other methods can be demonstrated through benchmarking, for which an evaluation metric is necessary. The next section deals with choosing the appropriate evaluation metric for acquisition based adaptive strategy.

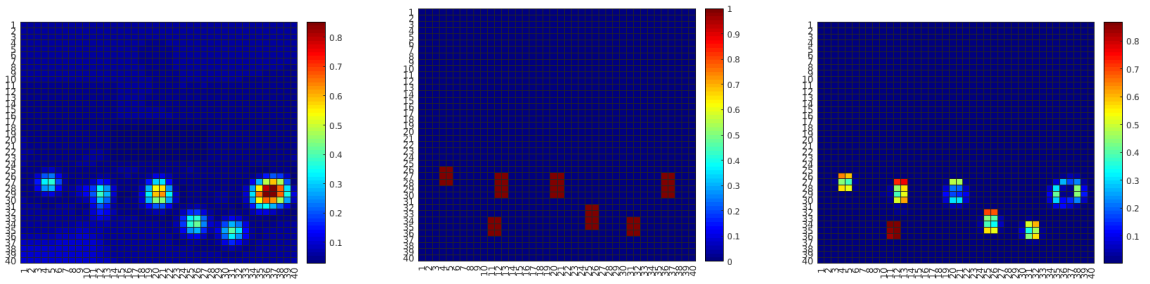
EVALUATION METRIC

In the previous sections, variance was used as the evaluation metric to compare the performance of the algorithm against different conditions. Reduction of uncertainty is a good performance metric if variance reduction is the objective of the planner. However, it might not be the most appropriate evaluation metric for an acquisition based planner, because the planner cares about both detections and variance in the interesting regions. Moreover, the end goal of a search and rescue operation is not to reduce the uncertainity of the field, but to find all victims in the field with accuracy. The final map should represent the ground truth, as closely as possible. Therefore, an appropriate measure is to evaluate the error between the map and the ground truth. Formally, it is defined as the Root Mean Squared Error (RMSE). RMSE is usually a very small number and hence a scaled measure of Root Square Error (RSE) is used, which is defined as:

$$RSE = \sum_{\forall \epsilon \in M} \Big[M(\epsilon) - GT(\epsilon) \Big]^2, \tag{6.1}$$

where ϵ is a grid cell in the map M, and GT is the ground truth map. Figure 6.6 shows the calculation of squared error from GP mean and ground truth of the environment shown in Figure 5.10. It can be observed that the squared error corresponding to the undetected human in the bottom left side of Figure 6.6a is very high in Figure 6.6c. Moreover, the extra grids detected as humans in the bottom right side of Figure 6.6a also contributes to the squared error in Figure 6.6c. Therefore, the RSE would equally weigh the contribution from false positive errors and false negative errors in the map, making it a good metric to capture the error in the map. However, it is important to set the GP prior to the most probable value of the ground truth, to obtain

the correct RSE characteristics. If the GP prior is set with a high value in an environment with a ground truth having humans sparsely located, all the cells would have a high squared error, resulting in a high RSE. The major contribution of the RSE will be from the unobserved spaces in the environment. This would result in an RSE measure that is biased towards unobserved spaces. However, setting a lower prior would be less biased towards unobserved space. The main contribution to the RSE would be coming from the unobserved cells with humans and the false positives in the map. Therefore, setting a high GP prior results in a major contribution from the unobserved cells, while setting it to a low value results in a major contribution from unobserved cells with humans and the false positive detections in the map. Therefore, the GP prior is set to the most probable value, which is a low value, assuming that most of the field is empty.



(a) GP mean. (b) Ground truth. (c) Squared error.

Figure 6.6: Squared error calculated from GP mean and ground truth.

Metrics like the Weighted RMSE (WRMSE) [75] could also be used as an evaluation metric. WRMSE gives a higher weight for the errors made at locations where the ground truth is high. However, this is equivalent to biasing the metric towards false negatives or unobserved spaces with humans, and neglecting the false positive errors to some degree. In a search and rescue operation, false positives and false negatives are both expensive and important. RMSE treats false positive errors and false negative errors equally. Therefore, RMSE is preferred over WRMSE as an evaluation metric for benchmarking, presented in the next sections.

Algorithm 7 Function for random sampling planner.

```
1:  function random_sampling()
2:      Create ESDF map
3:      Initialize the field
4:      if not landing condition then                ▷ Replan until the map can be made no better
5:          do
6:              Initialize C to R_0.
7:              Randomly sample a point R_rand from the environment.
8:              Insert R_rand to C.
9:          while not isCollisionFreePath(C, ESDF)
10:         Fly through the straight line path.
11:         Take measurements.
12:         Update map at measurement locations.
13:         Update R_0.
```

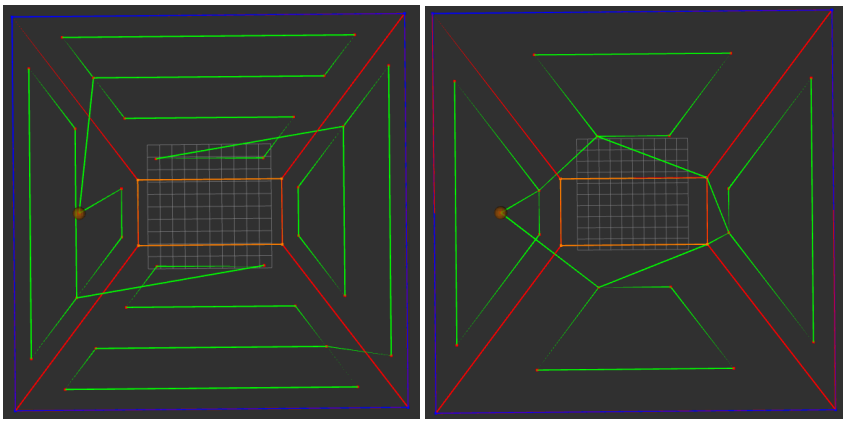
BENCHMARKS

This section aims at evaluating the performance of the IPP planner against two benchmarks: random sampling and coverage planner. A simple environment with an obstacle at the centre is considered. Multiple flight trials were simulated with different planners, and the RSE is evaluated after each measurement update. The results of each planner are then compared.

RANDOM SAMPLING

Random sampling is a good strategy to many problems in robotics. Therefore it was selected as a benchmark for comparison. A randomly sampled point from the 3D environment R_{rand} that is visible from the current

UAV location R_0 is selected as the next location to visit. Measurements are taken during the flight towards that point. The algorithm is summarized in algorithm 7.

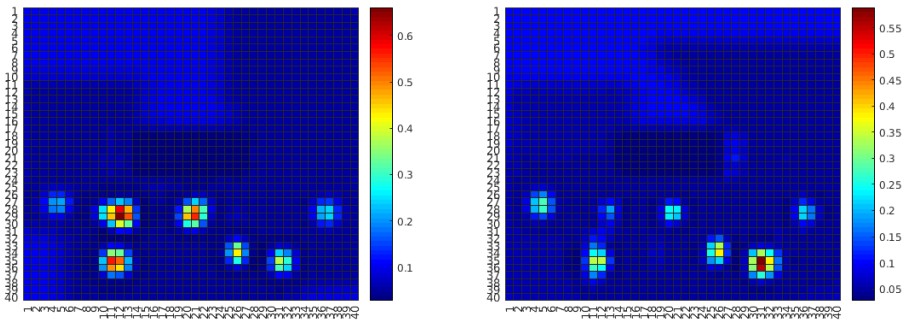


(a) Coverage planner at altitude = 5m. (b) Coverage planner at altitude = 15m.

Figure 6.7: Path planned by coverage planner at different heights.

COVERAGE PLANNING

Coverage Path Planning (CPP) aims to determine a path that passes over all the points of an area of interest while avoiding obstacles [8]. This is a standard solution that is widely used in robotics to provide coverage for an area in applications like painting robots, automated harvesters, inspection problems, etc. The choice of coverage planner for a search and operation is apt, in the sense that the objective is to cover a given area and find all victims in the field. Coverage planning can be applied to a fixed altitude in the environment. Figure 6.7 shows the path generated by the coverage planner in the environment given in Figure 5.10. The start and end locations of the UAV are set to be $(-10m, 0m, h)$ for both the cases, where h is the altitude at which coverage is done. All the parameters of the flight were kept the same as that of the IPP planner. The orange box at the centre of the Figure 6.7 represents the obstacle inflated to a size equal to half the length of the UAV. The green curve is the generated path. It can be observed that as h increases, the paths are further away from each other. This is attributed to the increasing ground visibility with respect to altitude. While flying through the generated path, the UAV takes measurements at a constant rate to find all the human victims.



(a) Coverage result at altitude = 10m. (b) Coverage result at altitude = 20m.

Figure 6.8: Quality of coverage planner result at different altitudes.

The performance of the coverage planner may vary with altitude, assuming no overlap in the measurement area. Figure 6.8 shows the end result of coverage planner after 150 seconds of flight. It can be observed that GP mean is significantly lower than 1. Moreover, at an altitude of 20m, some of the detections are not strong. Multiple measurements at same location is crucial towards increasing the accuracy of the detection, which a coverage planner lacks. The coverage planner was experimented at different heights by keeping all other parameters constant. Figure 6.9 shows the decrease in RSE of the map with respect to flight time of the

UAV, at different heights. The altitudes higher than h_{opt} were used to analyze the performance of the planner. It can be observed that as altitude increases, the performance of the coverage planner decreases.

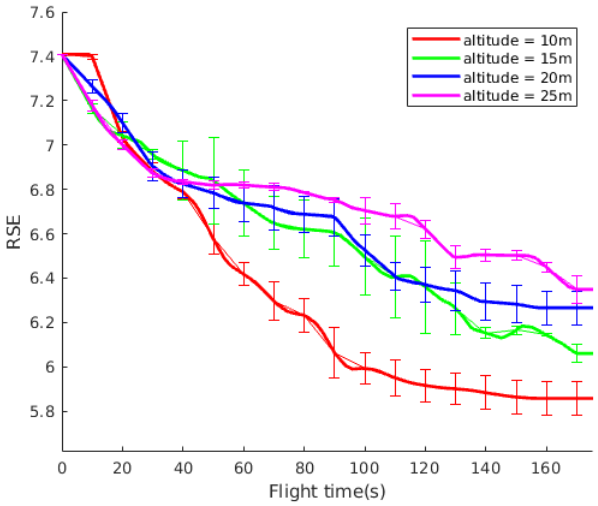


Figure 6.9: Effect of altitude on coverage planner.

Coverage planner would take a very lengthy path at lower heights, as shown in Figure 6.7a, making it impossible to cover the area in the budget flight time. The choice of altitude for coverage planner is motivated by the budget flight time. Therefore, the lower heights that does not ensure coverage were not considered. The height at which the coverage planner performs the best (10m) is selected for the benchmarking in the coming sections.

BENCHMARKING

This section deals with benchmarking the IPP planner against different objectives (adaptive and non-adaptive), planners (random sampling and coverage planner), optimizers (BO and CMAES), and sensors (classifiers and non-classifiers). The objectives under consideration are uncertainty reduction and the acquisition based objective. The planner is then benchmarked against random sampling method and coverage planner to evalaute its performance.

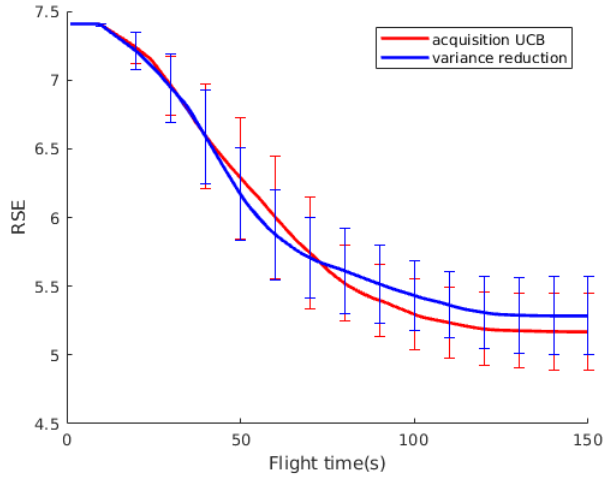


Figure 6.10: Comparison of the performance of adaptive and non-adaptive strategies.

ADAPTIVE AND NON-ADAPTIVE STRATEGIES

This section deals with demonstrating the effectiveness of using an acquistion based objective function in target search problems, when compared to uncertainty reduction as the objective. Two sets of flight experiments were conducted with the same environment and flight parameters except for the difference in the objective function: one planner used variance reduction as the objective, while the other planner used UCB acquisition based objective function. This results in one adaptive planner and one non-adaptive planner. The environment shown in Figure 5.10 was used for the flights. As illustrated in Section 5.6, the adaptive planner scans the area and then concentrates on regions of high detections, while a non-adaptive planner circles around the environment aiming to reduce uncertainty. The average result of 25 experiments each, along with the standard deviation are shown in Figure 6.10. It can be observed that both planners behave the same at the beginning and then saturates at different values. The adaptive strategy concentrates on the detections and decreases the error in the map through repeated measurements, while the non-adaptive strategy fails to do this. Therefore, an adaptive strategy performs better than a non-adaptive strategy in reducing the RSE of the map, while flying in an environment where the victims are located in some specific region of the field. However, if the victims are uniformly scattered in the environment, an adaptive planner performs like a non-adaptive planner.

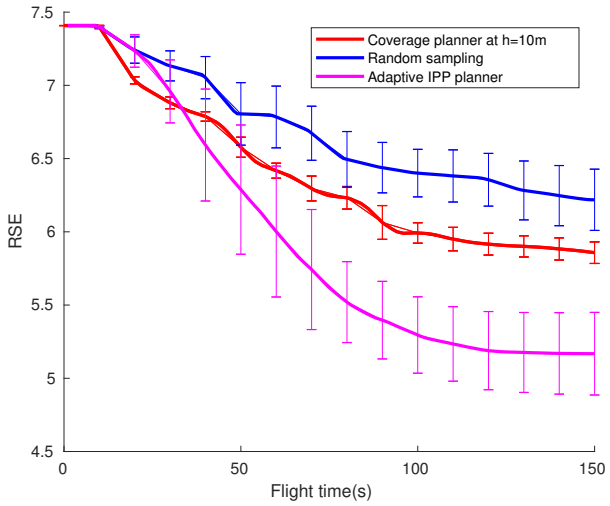


Figure 6.11: Comparison of differrent planners with GP map.

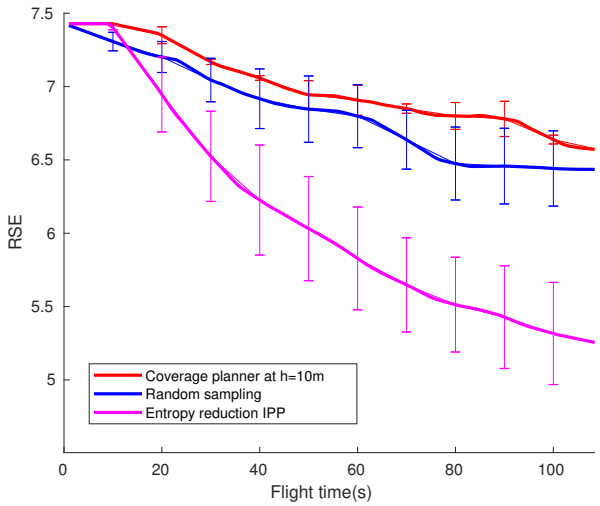


Figure 6.12: Comparison of differrent planners with OG map.

PLANNERS

Flight simulations were performed with the same flight parameters and same the environment as shown in Figure 5.10, to analyze the performance of different planners. A GP based continuous mapping was used for evaluation. Figure 6.11 summarize the results of 25 flights each with different planners: coverage planner at the best performing height (10m), random sampling and adaptive IPP planner. It should be noted that the trajectory of coverage planner is 2D, while that of random sampling and IPP planners are 3D. It can be observed from Figure 6.11 that the adaptive IPP planner quickly reduces the RSE of the map compared to other planners, demonstrating a very competitive performance.

Figure 6.12 shows the results of a similar experiment conducted on an OG map. It can be observed that entropy reduction based IPP planner performs better than other planners like random sampling and coverage planner. Therefore, the IPP planner outperforms other planners both for continuous and discrete mapping methods.

OPTIMIZERS

In this section, the planner is shown to be modular with respect to different optimizers, that can still solve the problem. The adaptive IPP planner used in the previous experiments use CMAES as the optimizer. However, BO is also a very competitive state-of-the-art optimizer, which was shown to solve an IPP problem [76]. $BayesOpt$ package [102] was used for BO, while $libcmaes$ package [103] was used for CMAES. The optmizer that solves the problem without the need to tune many hyperparameters would be preferred over the optimizer that is hard to tune when the planner is modified slightly.

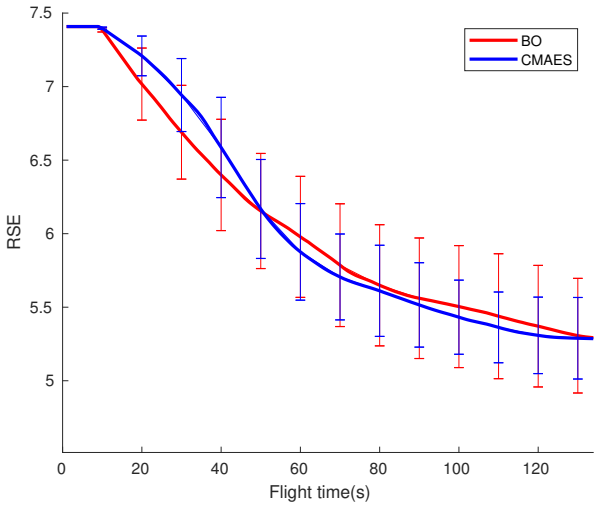


Figure 6.13: Performance of the optimizers with variance reduction as the objective.

Different variants of CMAES and BO were experimented, with variance reduction as the objective. The CMAES is quasi parameter free [82] and has two parameters: the step size and the number of offsprings, while the BO has more than 20 parameters. Moreover, the influence of each parameter on the flight performance can be evaluated only after hours of repeated test flights. This restricts the tuning of all BO parameters. Therefore, default parameters of the $BayesOpt$ package were used, and the parameters common to both the optimizers like the number of initial samples and number of iterations were set to the same value so that both of them consume the same computation time. The best results obtained from both the optimizers were considered and is shown in Figure 6.13. It can be observed that both the optimizers are successful in solving the IPP problem. However, the CMAES can be found to be performing slightly better than BO. The final half of the plot is the relevent part, since, this part is likely to produce the final map that is to be used by the search and rescue team. The average standard deviation of flights with BO running is 0.3507, while that of CMAES is 0.2501. Therefore, CMAES optimizer results in a flight that has a slightly better mean and and standard deviation, when compared to BO for near-default parameters.

The hyperparameters of both optimizers were kept the same and the acquisition function based objective function was used. Figure 6.14 shows the results of 25 flight trials. BO-BO represents the layered optimization approach, where the field is optimized by BO-UCB based acquisition and the path is optimized by BO.

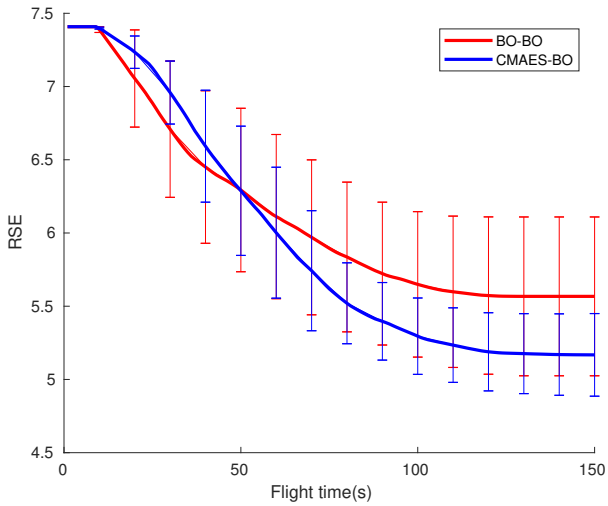


Figure 6.14: Performance of different optimizers solving the IPP problem with layered optimization.

CMAES-BO represents the layered optimization approach, where the field is optimized by BO-UCB based acquisition and the path is optimized by CMAES. The resulting flights have an average standard deviation of 0.2707 for CMAES-BO, and 0.4628 for BO-BO. From Figure 6.14, it can be observed that BO finds it hard to reach the global optimum towards the end for an acquisition based objective, while the CMAES decreases the resultant RSE to a lower value. It might be possible for the BO to solve the problem by tuning its hyperparameters. However, the aim of this section is to find the optimizer that is robust to changes in the objective functions used in the IPP framework, without having to tune its hyperparameters each time. The implication of this is a generalized algorithm that is robust, and can be applied to different IPP problems without many internal changes, thus improving the ease of its use from a user's point of view.

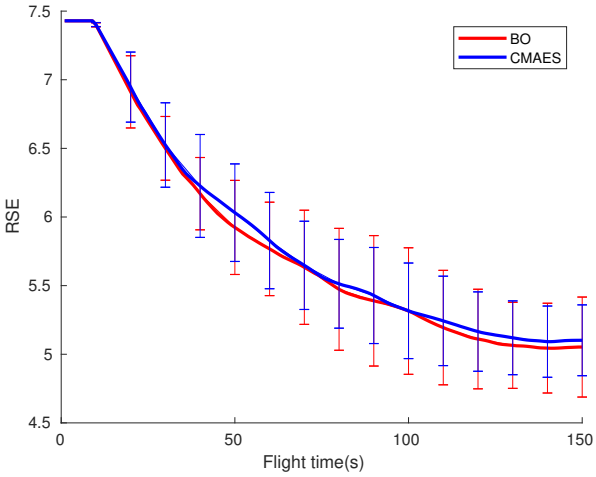


Figure 6.15: Performance of the optimizers with entropy reduction as the objective.

The hyperparameters of both optimizers were kept the same as in the previous simulations, and the entropy reduction for discrete mapping is used as the objective. Figure 6.15 shows the results of 25 flight trials. It can be observed that both optimizers are able to decrease the RSE to similar values, with BO having a slightly better mean. However, CMAES is able to generate flights with very similar performance with an average standard deviation of 0.2869, when compared to 0.3266 by BO. Therefore, from Figure 6.13, 6.14 and 6.15 it can be concluded that CMAES solves the optimization problem better than BO, without having to change any hyperparameters. This conclusion is in line with the findings in [6], which demonstrated better performance of the CMAES compared to the BO for an information-theoretic objective. It should be noted that the nature of

the objective function is dependent heavily on the environment and current state of the field. A reliable optimizer that can demonstrate good performance under changes in objective function is preferred for a more general and modular planner. Therefore, CMAES is a reliable choice for target search problem, compared to BO.

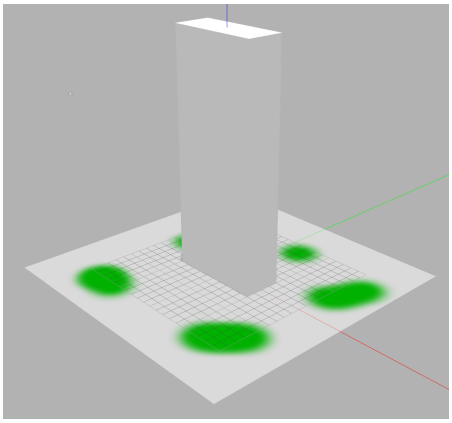


Figure 6.16: Environment with green blobs on the field.

SENSORS

This section deals with demonstrating the modularity of the algorithm in terms of the sensor used. The aim is to prove that the planner performs better than other planners for general non-classifier based sensor as well. An image based color detection sensor that can detect the amount of green color in the captured image is used for the simulation. Unlike the neural network based classifier, the color sensor gives a continuous value as the output instead of a binary value. The measurement uncertainty is assumed to have a Gaussian based noise variance model. The height dependent sensor model given in Equation 3.9 is used as the sensor model for planning and mapping with $A = 0.05$ and $B = 0.2$. Variance reduction in the GP map is used as the objective, and variance is used as the evaluation metric. The obstacles and the environment boundaries are the same as that of previous sections. The field is changed to a number of green blobs scattered on a white background, as shown in Figure 6.16. This could represent an algae growth on the surface of a canal in Amsterdam, surrounding a building. The aim of the planner is to find the green colored target, by reducing the uncertainty of the field.

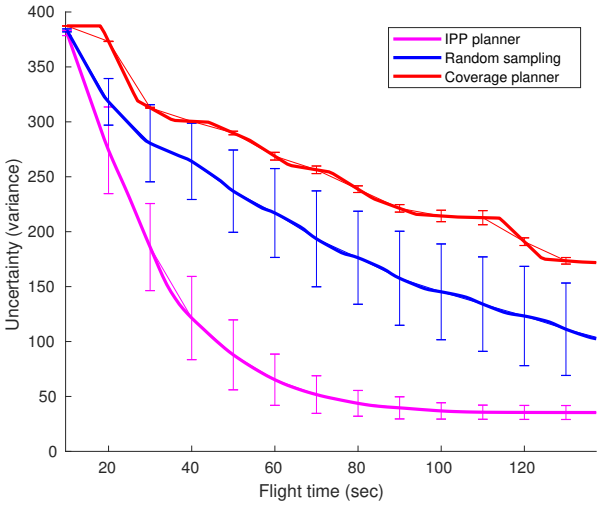


Figure 6.17: Performance of the planners with variance reduction as the objective for environment in FIgure 6.16.

100 flight trials each were performed with the IPP planner, coverage planner and random sampling, and the results are shown in Figure 6.17. It can be observed that the IPP planner reduces the variance of the field faster than both other planners. Therefore, it can be concluded that the algorithm outperforms other planners while using a classifier based sensor, as well as while using a non-classifier based sensor. This generalizes the planner so that it is employable in a wide range of environment monitoring tasks like temperature monitoring, gas concentration detection etc.

This chapter dealt with benchmarking the algorithm against various environment conditions, planners, optimizers, sensors and applications. It was found that using the hard constraint penalty function is favourable compared to the logistic function. The experiments with different environment complexities conclude that the algorithm is capable of handling a number of environments with randomly placed obstacles of different densities. It was found that the adaptive strategy for IPP planner works better than the non-adaptive strategy. The IPP planner demonstrated competitive results when compared to coverage planner and random sampling. Different optimizers were experimented and the CMAES was found to be the most appropriate optimizer for IPP based planner. Finally, a non-classifier based sensor was used to demonstrate that the planner still performs better than the other planners in terms of the RSE measure. Therefore, it can be concluded that the algorithm demonstrate competitive results in simulation. It is imperative to show that the algorithm is still applicable outside the simulation world, to convince the users about its real-life applicability. The next section deals with real experiments on the UAV flying inside a lab setup with obstacles.

7

EXPERIMENT RESULTS

This section aims at providing a proof of concept of the IPP based planner through real experiments. The algorithm is applied on a UAV that flies in a real environment in search of real humans lying on the floor. The goal of the UAV is to fly in the environment such that it doesn't collide with obstacles or cross the environemnt boundary, and at the same time find all the humans in the field. The following sections describe the experimental setup and the results of the test flight.

HARDWARE

A PX4 based custom made UAV called Loon from Autonomous Systems Lab was used as the platform for the test flight. It uses a dual core Intel nuc-i7, 3.5GHz processor with mRo Pixhawk as the flight controller. The radius of the sphere that can enclose the UAV is 90cm and is approximated as 1m by the planner to account for a factor of safety while flying. A downward looking camera, Chameleon 3 (model:CM3-U3-13Y3C-CS) was used to capture images of size 1280 × 1024 pixels. The camera captures the images at a maximum frame rate of 149 FPS and publishes the image as a ROS topic for the planner to use. YOLO2 Tiny was found to perform the human detection while in flight, at a measurement frequency of 1Hz. All the computations, including the working of YOLO was performed onboard. A Vicon based motion capture system was used to estimate the odometry of the UAV. Figure 7.1 shows the UAV used for the experiment along with its components.

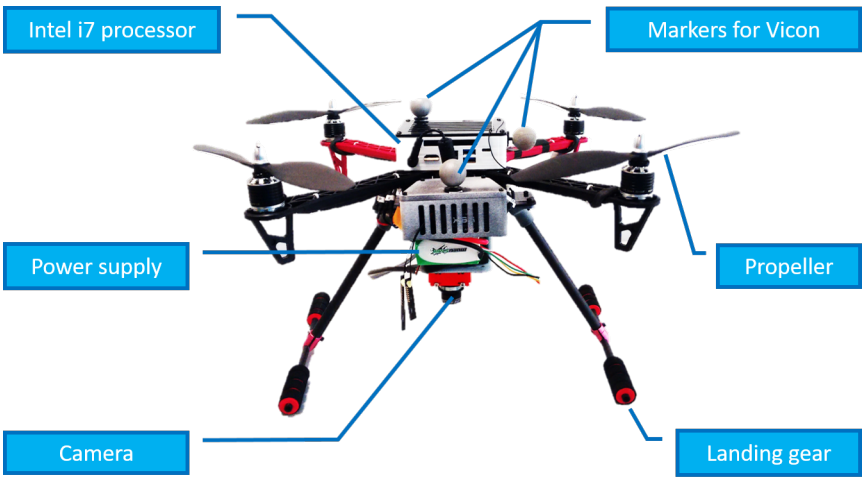
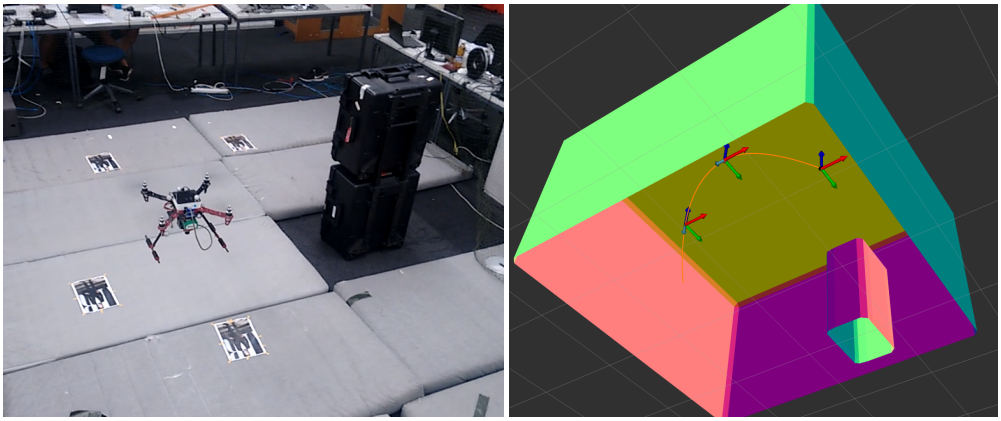


Figure 7.1: Components of the UAV used for the experiment.

ENVIRONMENT SETUP

The lab environment under consideration is a 3D space of dimensions $3.4 \times 3.4 \times 2m$. The relatively large size of the UAV when compared to the lab environment restricts the size and number of obstacles that can be



(a) Environment with a black obstacle. (b) Bottom view in *Rviz*.

Figure 7.2: The UAV flying in an environement with one obstacle and four human images on the ground.

placed in the environment. A cuboidal obstacle of size $0.5 \times 0.7 \times 1.3m$ was placed at the location $(0.57m, 1.15m)$. Four images of a human were printed to the size of an A4 sheet and were placed at different locations on the ground. Real sized human images/mannequins were not used due to the restrictions in the maximum height of the environment. Figure 7.2 shows the flight of the UAV in an environment with a black obstacle and four images of humans spread across the floor.

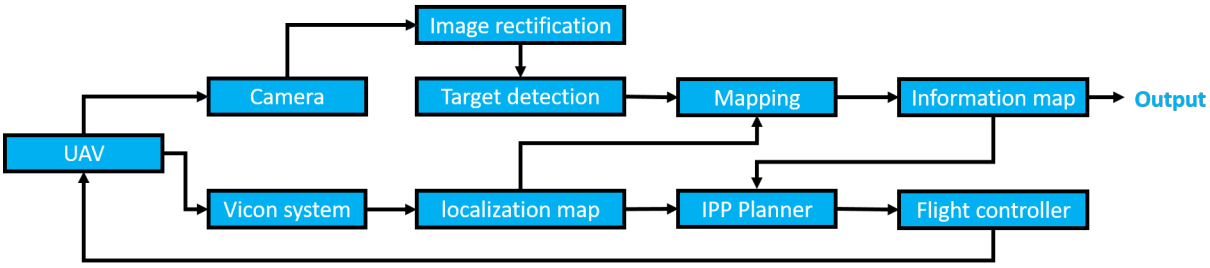


Figure 7.3: Flow chart for the flight experiment using UAV.

SOFTWARE SETUP

The Chameleon 3 camera output has huge distortions due to the fish-eye effect. Therefore, image rectification is important to reduce the error while mapping the image onto the ground. The intrinsic matrix of the camera was evaluated using the *Kalibr* camera calibration toolbox [104], which was later used to perform image rectification using the *image_undistort* ROS package. The resulting rectified image is then passed on to YOLO for target detection and mapping. Mapping from the image to the ground plane requires an extrinsic matrix which is a transformation matrix from the camera frame of reference to the Vicon system. It was evaluated manually through general coordinate transformation equations. The schematic of the experiment is shown in Figure 7.3. It can be observed that the UAV odometry and the image captured are the flight dependent inputs to the system, which are then processed to achieve the information map, which acts as the final output of the flight.

YOLO2 Tiny was tested against different heights by flying the UAV vertically above a human image. One of the YOLO target detection output, recorded while in flight is shown in Figure 7.4. It was observed that YOLO fails to detect the human beyond an altitude of 1m, while it performs the best around a height of 0.6m. The probability of a human detection to be correct was found to be very high, while the probability of missing out human detection was also found to be high. Therefore, a good true positive inverse sensor model and a bad false negative inverse sensor model were assumed for the sensor, which is given by Equation 7.1 and 7.2 respectively.

$$P(human = 1|z = 1) = \begin{cases} 0.95e^{-0.5\left(\frac{h-0.6}{0.75}\right)^2} & \text{if } h < 1, \\ 0.5 & \text{otherwise.} \end{cases} \tag{7.1}$$

$$P(human = 1|z = 0) = \begin{cases} 1 - \left[0.7e^{-0.5\left(\frac{h-0.6}{0.75}\right)^2}\right] & \text{if } h < 1, \\ 0.5 & \text{otherwise.} \end{cases} \tag{7.2}$$

Figure 7.4: YOLO output of an image captured by the UAV while in flight.

The performance of YOLO depends on the yaw angle of the UAV while the image is captured. This can be attributed to the lack of humans at different orientations in the training dataset of YOLO. Therefore, the polynomial trajectories for the UAV are generated such that the resultant yaw angle of the UAV is always constant while in flight. This measure is also crucial towards an easy safety piloting, in case of unexpected flight behavior of the UAV which could result in a hardware damage. The reference speed of the UAV is set to a low value of $0.2 m/s$, and the reference acceleration was set to $0.5 m/s^2$ to avoid motion blur effects.

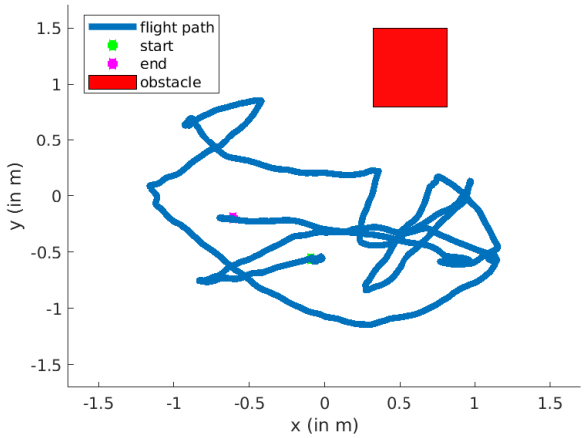
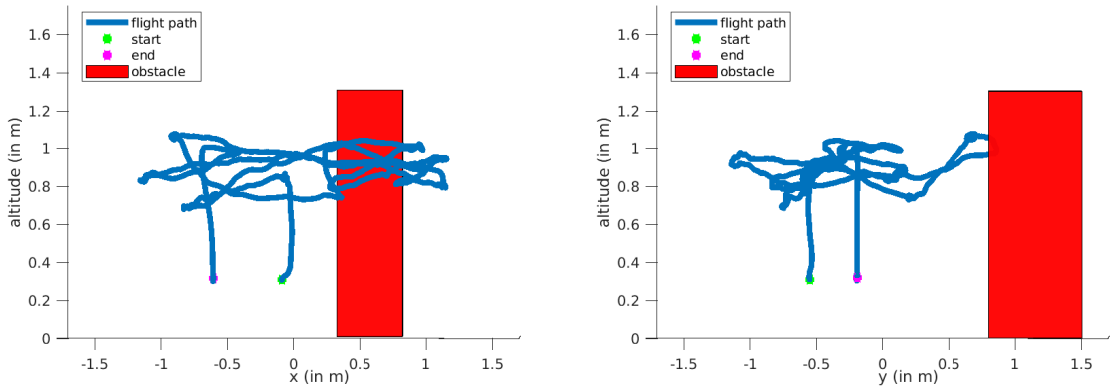


Figure 7.5: Top view of the path taken by the UAV.

FLIGHT RESULTS

The IPP planner was used to plan trajectories for the UAV and it was commanded to follow the trajectory while taking mesurements. The measurements made are then fused onto the discrete map. At the end of each trajectory, replanning is performed and the flight is resumed until the flight time budget of 150s is exhausted. The data related to the test flight are saved for further evaluations. The UAV took reliable paths that never collided with obstacles. The top view of the path taken by the UAV is shown in Figure 7.5. It can be observed that the UAV stays away from the obstacles and the environment boundaries. All points on the path are at least 0.5m away from the closest obstacle or boundary, which is the radius of the UAV.



(a) Front view.

(b) Side view.

Figure 7.6: Path taken by the UAV in the environment.

Figures 7.6a and 7.6b show the front view and top view of the path. It can be observed that although the maximum height of the environment is 2m, the UAV is restricted within an altitude of 1m. This is due to the role of saturation height $h_{sat} = 1m$ in planning. The sensor model given by Equations 7.1 and 7.2 considers poor performance beyond h_{sat}. Therefore, the paths are planned such that the optimizer avoids those paths that does not provide any information gain. It can also be noted that the planned path is in a 3D space and not on a 2D plane as that of a coverage planner, enabling variable resolution mapping. The UAV preferred a higher altitude at the beginning of the flight for better field of view. However, after covering the area, it preferred lower heights for better resolution. It can also be observed that the planner prefers an altitude that is higher when compared to the optimal height for sensor performance, $h_{opt} = 0.6m$, demonstrating the trade-off between sensor performance and FoV while planning.

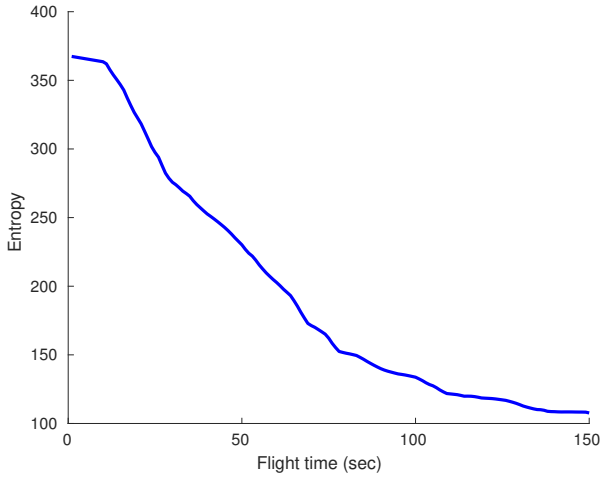
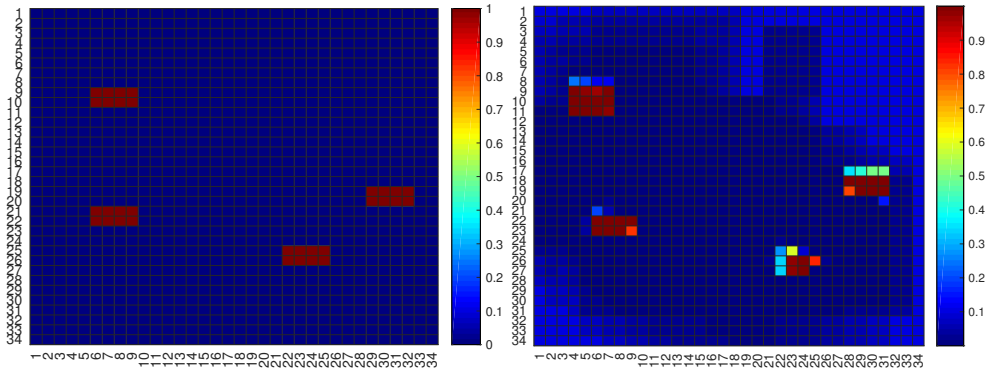


Figure 7.7: Entropy reduction while in flight.

The objective used for planning was entropy reduction. Figure 7.7 shows the reduction in entropy with respect to the flight time. Therefore, the optimizer is capable of performing entropy reduction, and the goal of the planner is achieved through the multi-resolution mapping procedure. Figure 7.8 demonstrates the success of the algorithm in the target search task. The close resemblance of the ground truth of human occupancy shown in Figure 7.8a, and the state of the map after a flight time of 150s as shown in Figure 7.8b, shows the ability of the algorithm to find all targets accurately. It can also be observed from Figure 7.8b that the top right portion remains unmapped. This is an inaccessible space in the environment into which the UAV does not fit. Therefore, it can be concluded that the algorithm is capable of covering the entire field except for the inaccessible regions in the field constrained by the UAV size. The algorithm internally identifies the

inaccessibility of the region and prevents it from violating the collision avoidance rule although it might lead to a higher information gain. This demonstrates the top priority of the algorithm towards collision avoidance when compared to the information gain, which is of utmost importance for a data acquisition system.



(a) Ground truth of human occupancy.

(b) Map after a flight time of 130 seconds.

Figure 7.8: Comparison of the ground truth and the human occupancy map after the flight.

This chapter demonstrated a proof of concept for the IPP algorithm in solving a target detection task, by implementing it on a UAV and flying it in a real environment. The optimizer successfully evaluated the paths by trading off between sensor performance, information gain and coverage, with collision avoidance as its top priority. The entropy reduction during the flight demonstrated that the mapping strategy is successfull in delivering what the planner intended to perform in its objective. Moreover, the resulting map closely resembles the ground truth. Therefore, the algorithm is useful in the target detection task and performs realiably in real experiments.

8

CONCLUSION

The target search problem using a UAV was converted into a general optimization problem that optimizes an information-theoretic objective to find all the targets in the field. The IPP planner generates a fixed horizon plan in the form of an optimal 3D minimum-snap trajectory in an obstacle filled environment for maximum information gain, minimum flight time, maximum coverage and no collisions. The planner caters for uncertainties and errors in sensor measurements in a probabilistic manner. An adaptive information-theoretic objective based on acquisition functions in BO was introduced for continuous mapping, which was shown to outperform the existing objectives like uncertainty reduction in terms of map error. It also outperforms other benchmarks like coverage planner and random sampling in terms of RSE reduction. The planner was tested against varying environmental complexities, demonstrating the capability of the algorithm to function in wide range of environments. The algorithm is modular in terms of mapping strategies, sensor selection, environment complexities, optimizers used, type of sensor and changes in sensor model. However, it has two main shortcomings. Firstly, it assumes a known and static environment, which might not be true in all scenarios. Secondly, the planner assumes a 2D non-temporal ground field. The algorithm can be improved to fuse the sensor data from a distance sensor so that it can dynamically update the obstacle map. It can also be improved to search for spatio-temporal phenomena. Therefore, the future work on the algorithm includes, planning in a dynamic environment while searching for a target in a temporal 3D field. The algorithm can also be extended into a distributed algorithm that performs the planning and mapping using a collaboration of multiple UAVs, covering a very large area.

BIBLIOGRAPHY

[1] R. M. Voyles and A. C. Larson, *Terminatorbot: a novel robot with dual-use mechanism for locomotion and manipulation,* IEEE/ASME Transactions on Mechatronics **10**, 17 (2005).

[2] D. Campbell and M. Buehler, *Stair descent in the simple hexapod'rhex',* in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, Vol. 1 (IEEE, 2003) pp. 1380–1385.

[3] R. R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. M. Erkmen, *Search and rescue robotics,* in *Springer Handbook of Robotics* (Springer, 2008) pp. 1151–1173.

[4] I. F. of Red Cross, R. C. Societies, and C. for Research on the Epidemiology of Disasters, *World Disasters Report* (Kumarian Press, 2005).

[5] N. F. P. Association *et al., NFPA 1670, standard on operations and training for technical rescue incidents* (The Association, 2000).

[6] G. Hitz, E. Galceran, M.-È. Garneau, F. Pomerleau, and R. Siegwart, *Adaptive continuous-space informative path planning for online environmental monitoring,* Journal of Field Robotics **34**, 1427 (2017).

[7] Y. K. Hwang and N. Ahuja, *Gross motion planning—a survey,* ACM Computing Surveys (CSUR) **24**, 219 (1992).

[8] E. Galceran and M. Carreras, *A survey on coverage path planning for robotics,* Robotics and Autonomous systems **61**, 1258 (2013).

[9] A. Krause and C. Guestrin, *Near-optimal observation selection using submodular functions,* in *AAAI*, Vol. 7 (2007) pp. 1650–1654.

[10] A. Krause and C. Guestrin, *Submodularity and its applications in optimized information gathering,* ACM Transactions on Intelligent Systems and Technology (TIST) **2**, 32 (2011).

[11] J. Delmerico, E. Mueggler, J. Nitsch, and D. Scaramuzza, *Active autonomous aerial exploration for ground robot path planning,* IEEE Robotics and Automation Letters **2**, 664 (2017).

[12] M. Popovic, T. Vidal-Calleja, G. Hitz, I. Sa, R. Siegwart, and J. Nieto, *Multiresolution mapping and informative path planning for uav-based terrain monitoring,* arXiv preprint arXiv:1703.02854 (2017).

[13] M. Popović, G. Hitz, J. Nieto, I. Sa, R. Siegwart, and E. Galceran, *Online informative path planning for active classification using uavs,* in *Robotics and Automation (ICRA), 2017 IEEE International Conference on* (IEEE, 2017) pp. 5753–5758.

[14] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, *Receding horizon" next-best-view" planner for 3d exploration,* in *Robotics and Automation (ICRA), 2016 IEEE International Conference on* (IEEE, 2016) pp. 1462–1468.

[15] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, *Information based adaptive robotic exploration,* in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, Vol. 1 (IEEE, 2002) pp. 540–545.

[16] T. Kollar and N. Roy, *Efficient optimization of information-theoretic exploration in slam.* in *AAAI*, Vol. 8 (2008) pp. 1369–1375.

[17] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg, *Near-optimal sensor placements: Maximizing information while minimizing communication cost,* in *Proceedings of the 5th international conference on Information processing in sensor networks* (ACM, 2006) pp. 2–10.

[18] A. Krause, A. Singh, and C. Guestrin, *Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies,* Journal of Machine Learning Research **9**, 235 (2008).

[19] P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden, *The orienteering problem: A survey,* European Journal of Operational Research **209**, 1 (2011).

[20] E. Lawer, J. K. Lenstra, A. R. Kan, and D. Shmoys, *The traveling salesman problem: a guided tour of combinatorial optimization,* (1985).

[21] R. Bajcsy, *Active perception,* (1988).

[22] J. Binney, A. Krause, and G. S. Sukhatme, *Optimizing waypoints for monitoring spatiotemporal phenomena,* The International Journal of Robotics Research **32**, 873 (2013).

[23] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser, *Efficient informative sensing using multiple robots,* Journal of Artificial Intelligence Research **34**, 707 (2009).

[24] G. A. Hollinger and G. S. Sukhatme, *Sampling-based robotic information gathering algorithms,* The International Journal of Robotics Research **33**, 1271 (2014).

[25] P. Dames, M. Schwager, V. Kumar, and D. Rus, *A decentralized control policy for adaptive information gathering in hazardous environments,* in *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on* (IEEE, 2012) pp. 2807–2813.

[26] Z. W. Lim, D. Hsu, and W. S. Lee, *Adaptive informative path planning in metric spaces,* The International Journal of Robotics Research **35**, 585 (2016).

[27] S. Kemna, D. A. Caron, and G. S. Sukhatme, *Adaptive informative sampling with autonomous underwater vehicles: Acoustic versus surface communications,* in *OCEANS 2016 MTS/IEEE Monterey* (IEEE, 2016) pp. 1–8.

[28] G. A. Hollinger, *Long-horizon robotic search and classification using sampling-based motion planning.* in *Robotics: Science and Systems*, Vol. 3 (2015).

[29] K. H. Low, J. M. Dolan, and P. K. Khosla, *Information-theoretic approach to efficient adaptive path planning for mobile robotic environmental sensing.* in *ICAPS* (2009) pp. 233–240.

[30] A. Singh, R. Nowak, and P. Ramanathan, *Active learning for adaptive mobile sensing networks,* in *Proceedings of the 5th international conference on Information processing in sensor networks* (ACM, 2006) pp. 60–68.

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

[32] B. C. Dean, M. X. Goemans, and J. Vondrdk, *Approximating the stochastic knapsack problem: The benefit of adaptivity,* in *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on* (IEEE, 2004) pp. 208–217.

[33] D. Golovin and A. Krause, *Adaptive submodularity: A new approach to active learning and stochastic optimization.* in *COLT* (2010) pp. 333–345.

[34] P. R. Goundan and A. S. Schulz, *Revisiting the greedy approach to submodular set function maximization,* Optimization online , 1 (2007).

[35] A. Singh, F. Ramos, H. D. Whyte, and W. J. Kaiser, *Modeling and decision making in spatio-temporal processes for environmental surveillance,* in *Robotics and Automation (ICRA), 2010 IEEE International Conference on* (IEEE, 2010) pp. 5490–5497.

[36] M. Minoux, *Accelerated greedy algorithms for maximizing submodular set functions,* in *Optimization techniques* (Springer, 1978) pp. 234–243.

[37] C. Chekuri and M. Pal, *A recursive greedy algorithm for walks in directed graphs,* in *Foundations of Computer Science, 2005. FOCS 2005. 46th Annual IEEE Symposium on* (IEEE, 2005) pp. 245–253.

[38] G. Hitz, A. Gotovos, M.-É. Garneau, C. Pradalier, A. Krause, R. Y. Siegwart, *et al., Fully autonomous focused exploration for robotic environmental monitoring,* in *Robotics and Automation (ICRA), 2014 IEEE International Conference on* (IEEE, 2014) pp. 2658–2664.

[39] A. Singh, A. Krause, and W. J. Kaiser, *Nonmyopic adaptive informative path planning for multiple robots.* in *IJCAI*, Vol. 3 (2009) p. 2.

[40] A. Meliou, A. Krause, C. Guestrin, and J. M. Hellerstein, *Nonmyopic informative path planning in spatio-temporal models,* in *AAAI*, Vol. 10 (2007) pp. 16–7.

[41] J. L. Nguyen, N. R. Lawrance, and S. Sukkarieh, *Nonmyopic planning for long-term information gathering with an aerial glider,* in *Robotics and Automation (ICRA), 2014 IEEE International Conference on* (IEEE, 2014) pp. 6573–6578.

[42] J. Binney and G. S. Sukhatme, *Branch and bound for informative path planning,* in *Robotics and Automation (ICRA), 2012 IEEE International Conference on* (IEEE, 2012) pp. 2147–2154.

[43] J. Binney, A. Krause, and G. S. Sukhatme, *Informative path planning for an autonomous underwater vehicle,* (2010).

[44] Y. Girdhar and G. Dudek, *Modeling curiosity in a mobile robot for long-term autonomous exploration and monitoring,* Autonomous Robots **40**, 1267 (2016).

[45] R. A. MacDonald and S. L. Smith, *Reactive motion planning in uncertain environments via mutual information policies,* in *International Workshop on the Algorithmic Foundations of Robotics (WAFR)* (2016).

[46] D. E. Soltero, M. Schwager, and D. Rus, *Generating informative paths for persistent sensing in unknown environments,* in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on* (IEEE, 2012) pp. 2172–2179.

[47] A. Ryan and J. K. Hedrick, *Particle filter based information-theoretic active sensing,* Robotics and Autonomous Systems **58**, 574 (2010).

[48] A. Jones, M. Schwager, and C. Belta, *A receding horizon algorithm for informative path planning with temporal logic constraints,* in *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (IEEE, 2013) pp. 5019–5024.

[49] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte, *Coordinated decentralized search for a lost target in a bayesian world,* in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, Vol. 1 (IEEE, 2003) pp. 48–53.

[50] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics* (MIT press, 2005).

[51] V. Myers and D. P. Williams, *A pomdp for multi-view target classification with an autonomous underwater vehicle,* in *OCEANS 2010* (IEEE, 2010) pp. 1–5.

[52] S. Waharte and N. Trigoni, *Supporting search and rescue operations with uavs,* in *Emerging Security Technologies (EST), 2010 International Conference on* (IEEE, 2010) pp. 142–147.

[53] J. J. Chung, N. R. Lawrance, and S. Sukkarieh, *Gaussian processes for informative exploration in reinforcement learning,* in *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (IEEE, 2013) pp. 2633–2639.

[54] L. A. Bush, B. Williams, and N. Roy, *Computing exploration policies via closed-form least-squares value iteration,* in *International Conference on Planning and Scheduling* (2008).

[55] T. Kollar and N. Roy, *Trajectory optimization using reinforcement learning for map exploration,* The International Journal of Robotics Research **27**, 175 (2008).

[56] S. M. LaValle and J. J. Kuffner Jr, *Randomized kinodynamic planning,* The International Journal of Robotics Research **20**, 378 (2001).

[57] D. Levine, B. Luders, and J. P. How, *Information-rich path planning with general constraints using rapidly-exploring random trees,* (2010).

[58] A. Viseras, D. Shutin, and L. Merino, *Online information gathering using sampling-based planners and gps: an information theoretic approach,* (2017).

[59] J.-y. Kwak and P. Scerri, *Path planning for autonomous information collecting vehicles,* in *Information Fusion, 2008 11th International Conference on* (IEEE, 2008) pp. 1–8.

[60] J. Suh, K. Cho, and S. Oh, *Efficient graph-based informative path planning using cross entropy,* in *Decision and Control (CDC), 2016 IEEE 55th Conference on* (IEEE, 2016) pp. 5894–5899.

[61] S. Arora and S. Scherer, *Randomized algorithm for informative path planning with budget constraints,* in *Robotics and Automation (ICRA), 2017 IEEE International Conference on* (IEEE, 2017) pp. 4997–5004.

[62] R. He, S. Prentice, and N. Roy, *Planning in information space for a quadrotor helicopter in a gps-denied environment,* in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on* (IEEE, 2008) pp. 1814–1820.

[63] J. Suh and S. Oh, *A cost-aware path planning algorithm for mobile robots,* in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on* (IEEE, 2012) pp. 4724–4729.

[64] X. Lan and M. Schwager, *Planning periodic persistent monitoring trajectories for sensing robots in gaussian random fields,* in *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (IEEE, 2013) pp. 2415–2420.

[65] C. Stachniss, G. Grisetti, and W. Burgard, *Information gain-based exploration using rao-blackwellized particle filters.* in *Robotics: Science and Systems*, Vol. 2 (2005) pp. 65–72.

[66] J. Witt and M. Dunbabin, *Go with the flow: Optimal auv path planning in coastal environments,* in *Australian Conference on Robotics and Automation*, Vol. 2008 (2008).

[67] A. Arora, R. Fitch, and S. Sukkarieh, *An approach to autonomous science by modeling geological knowledge in a bayesian framework,* arXiv preprint arXiv:1703.03146 (2017).

[68] A. Arora, P. M. Furlong, R. Fitch, T. Fong, S. Sukkarieh, and R. Elphic, *Online multi-modal learning and adaptive informative trajectory planning for autonomous exploration,* in *Field and Service Robotics* (Springer, 2018) pp. 239–254.

[69] G. Best, O. M. Cliff, T. Patten, R. R. Mettu, and R. Fitch, *Decentralised monte carlo tree search for active perception,* in *Proc. of WAFR* (2016).

[70] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, *A survey of monte carlo tree search methods,* IEEE Transactions on Computational Intelligence and AI in games **4**, 1 (2012).

[71] M. Elbanhawi and M. Simic, *Sampling-based robot motion planning: A review,* IEEE Access **2**, 56 (2014).

[72] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, *Boa: The bayesian optimization algorithm,* in *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1* (Morgan Kaufmann Publishers Inc., 1999) pp. 525–532.

[73] M. Pelikan, K. Sastry, and D. E. Goldberg, *Scalability of the bayesian optimization algorithm,* International Journal of Approximate Reasoning **31**, 221 (2002).

[74] E. Brochu, V. M. Cora, and N. De Freitas, *A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning,* arXiv preprint arXiv:1012.2599 (2010).

[75] R. Marchant and F. Ramos, *Bayesian optimisation for intelligent environmental monitoring,* in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on* (IEEE, 2012) pp. 2242–2249.

[76] R. Marchant and F. Ramos, *Bayesian optimisation for informative continuous path planning,* in *Robotics and Automation (ICRA), 2014 IEEE International Conference on* (IEEE, 2014) pp. 6136–6143.

[77] K. C. Vivaldini, V. Guizilini, M. D. Oliveira, T. H. Martinelli, D. F. Wolf, and F. Ramos, *Route planning for active classification with uavs,* in *Robotics and Automation (ICRA), 2016 IEEE International Conference on* (IEEE, 2016) pp. 2563–2568.

[78] J. R. Souza, R. Marchant, L. Ott, D. F. Wolf, and F. Ramos, *Bayesian optimisation for active perception and smooth navigation,* in *Robotics and Automation (ICRA), 2014 IEEE International Conference on* (IEEE, 2014) pp. 4081–4087.

[79] P. Morere, R. Marchant, and F. Ramos, *Sequential bayesian optimisation as a pomdp for environment monitoring with uavs,* arXiv preprint arXiv:1703.04211 (2017).

[80] R. Marchant, F. Ramos, S. Sanner, *et al.*, *Sequential bayesian optimisation for spatial-temporal monitoring.* in *UAI* (2014) pp. 553–562.

[81] R. Martinez-Cantin, N. de Freitas, E. Brochu, J. Castellanos, and A. Doucet, *A bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot,* Autonomous Robots **27**, 93 (2009).

[82] N. Hansen, *The cma evolution strategy: a comparing review,* in *Towards a new evolutionary computation* (Springer, 2006) pp. 75–102.

[83] A. Auger, N. Hansen, J. M. P. Zerpa, R. Ros, and M. Schoenauer, *Experimental comparisons of derivative free optimization algorithms.* SEA **5526**, 3 (2009).

[84] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. Kumar, *Information-theoretic planning with trajectory optimization for dense 3d mapping.* in *Robotics: Science and Systems* (2015).

[85] B. Charrow, S. Liu, V. Kumar, and N. Michael, *Information-theoretic mapping using cauchy-schwarz quadratic mutual information,* in *Robotics and Automation (ICRA), 2015 IEEE International Conference on* (IEEE, 2015) pp. 4791–4798.

[86] C. E. Rasmussen, *Evaluation of Gaussian processes and other methods for non-linear regression.* (Citeseer, 1999).

[87] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning,* the MIT Press **2**, 4 (2006).

[88] S. Reece and S. Roberts, *An introduction to gaussian processes for the kalman filter expert,* in *2010 13th International Conference on Information Fusion* (2010) pp. 1–9.

[89] D. Barber, *Bayesian reasoning and machine learning* (Cambridge University Press, 2012).

[90] A. Elfes, *Using occupancy grids for mobile robot perception and navigation,* Computer **22**, 46 (1989).

[91] C. E. Shannon, *A mathematical theory of communication,* ACM SIGMOBILE Mobile Computing and Communications Review **5**, 3 (2001).

[92] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, *Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning,* in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on* (IEEE, 2017) pp. 1366–1373.

[93] C. W. Warren, *Global path planning using artificial potential fields,* in *Proceedings, 1989 International Conference on Robotics and Automation* (1989) pp. 316–321 vol.1.

[94] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, *Ssd: Single shot multibox detector,* in *Computer Vision – ECCV 2016*, edited by B. Leibe, J. Matas, N. Sebe, and M. Welling (Springer International Publishing, Cham, 2016) pp. 21–37.

[95] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, *You only look once: Unified, real-time object detection,* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016) pp. 779–788.

[96] N. Dalal and B. Triggs, *Histograms of oriented gradients for human detection,* in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, Vol. 1 (IEEE, 2005) pp. 886–893.

[97] S. Thrun, *Learning occupancy grid maps with forward sensor models,* Autonomous robots **15**, 111 (2003).

[98] C. Richter, A. Bry, and N. Roy, *Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments,* in *Robotics Research* (Springer, 2016) pp. 649–666.

[99] D. R. Jones, M. Schonlau, and W. J. Welch, *Efficient global optimization of expensive black-box functions,* Journal of Global optimization **13**, 455 (1998).

[100] D. R. Jones, *A taxonomy of global optimization methods based on response surfaces,* Journal of global optimization **21**, 345 (2001).

[101] D. D. Cox and S. John, *Sdo: A statistical method for global optimization,* Multidisciplinary design optimization: state of the art , 315 (1997).

[102] R. Martinez-Cantin, *Bayesopt: A bayesian optimization library for nonlinear optimization, experimental design and bandits,* The Journal of Machine Learning Research **15**, 3735 (2014).

[103] E. Benazera, *libcmaes: Multithreaded c++ 11 implementation of cma-es family for optimization of nonlinear non-convex blackbox functions, 2014,* Google Scholar .

[104] J. Maye, P. Furgale, and R. Siegwart, *Self-supervised calibration for robotic systems,* in *2013 IEEE Intelligent Vehicles Symposium (IV)* (2013) pp. 473–480.