PATH PLANNING ALGORITHMS FOR ADVERSE WEATHER CONDITIONS

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Science

by

Abhishek Puthige

May 2022

ABSTRACT

The treasure hunt problem was introduced to describe the problem of planning the path and measurements of a sensor installed on a ground robot, in order to classify multiple targets in an obstacle-populated environment. The use of conventional path-planners like probabilistic roadmaps (PRM) for this purpose requires prior knowledge of the workspace while other online path-planning algorithms rely on the sensor's ability to form a global map and localize itself in it. However, in unknown workspaces, under environmental pressures like fog, the information captured by the sensor is extremely local leading to a non-convergent global map, which subsequently limits the functioning of the algorithms. Artificial systems implement decision and control policies to optimize a given cost function on one hand, humans use satisficing decision strategies to overcome the limitations of partial information on the other. Satisficing strategies lead to solutions that are not always optimal for a given system, but which are good enough to meet all its needs at a certain level given the constraints on resources. To overcome the limitations of the current artificial systems, this work aims to create the building blocks for an adaptive heuristic path planner which efficiently tackles the treasure hunt problem in unknown workspaces under environmental pressures. Two different path planners that mimic satisficing strategies have been simulated. With the results of this work, adaptive heuristic path planners can further be developed which will improve autonomous area exploration and efficiently solve the treasure hunt problem.

BIOGRAPHICAL SKETCH

Abhishek Puthige is an M.S. student in the Laboratory for Intelligent Systems and Controls (LISC) at Cornell University. He received his B.Tech degree in Mechanical Engineering from PES University. His research interests include vehicle intelligence, robotics, and machine learning.

This document is dedicated to all Cornell graduate students.

ACKNOWLEDGEMENTS

I would like to thank Dr. Silvia Ferrari for her kind guidance and immense support for my endeavours during my time at Cornell University. It was a great honor to conduct my research under her guidance. I would also like to thank Dr. Fengqi You as my committee member for his valuable advice on my research. I would also like to thank Yucheng Chen, Jane Shin, and Shi Chang for their help with my research and defense. I would also like to thank all LISC members, who provided me invaluable support on both my work and life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION AND BACKGROUND

Autonomous mobile robots are being increasingly deployed in a variety of applications like robo-taxis, urban surveillance, and landmine detection. The navigation task of such robots can be tessellated into perception, path-planning, and motion control given an objective. In the context of an autonomous mobile robot, perception is defined as the sensory experience of the world, which includes recognizing and interpreting the sensor data [1]. Motion control refers to controlling the robot to manipulate it and produce favorable changes in its state. Path planning refers to estimating a sequence of actions that transform a robot's initial state to a desired final state with the aim of achieving a pre-determined objective. It is becoming an ever-increasing part of our lives with the advances being made in the domain of autonomous technologies. These technologies use path planning in many different ways with different optimization objectives, such as energy and terrain considerations in the mars rover [2], and traffic, distance, and the local environment in autonomous vehicles [3–6]. Due to a myriad applications of path-planners, several solutions have been developed to solve these problems [7–12]. These formulations depend on several factors like the type of the system and its constraints and the optimization objective. For instance, the indoor navigation of a robot problem can either be represented by a graph with nodes and edges corresponding to the configuration space in the environment [13–15] or as a potential gradient problem in which there is an attractive force from the target and a repulsive force from the obstacles [16, 17]. Despite the many possible formulations, these algorithms suffer from the drawback of either requiring the knowledge of the global environment as *a priori* or require to be able to simultaneously map the environment and localize themselves. In the presence of environmental pressures like fog, the ability of vision based sensors to perceive the world around them decreases, due to which the information available is hyper-local, making it challenging to localize the robot and map the world around it. Thus, the path planning problem of navigating a robot in an unknown workspace under adverse environmental conditions requires a different approach.

Satisficing is a set of choice strategies for systems dealing with real-world problems of noisy data, insufficient intelligence, time and computational pressures, and other constraints [18]. Satisficing strategies lead to solutions that are not always optimal for a given system and environment, but which are good enough to meet all of its needs at a certain level [19–21]. Current artificial systems often implement decision and control policies that are designed to optimize a given cost function. In contrast, biological organisms use satisficing strategies because of the available partial or no prior data about the history, present context, and potential outcomes of decisions. Thus, artificial systems may fail to return a feasible policy under real world constraints, like those of time, computation, and weather conditions, unlike biological organisms which are capable of producing fast but "good enough" decisions that trade-off the benefits of accumulating exhaustive data against the potential costs of missing an impending deadline. Moreover, the optimal strategies based on small-world assumptions can lead to failures or disasters when applied to our large-world which is filled with uncertainty and noisy information [22]. The implementation of such satisficing strategies in autonomous systems teamed with humans could also facilitate cooperation.

Heuristics are quick decision-making systems that ignore part of the information to save time and effort. A significant feature of a successful satisficing method is that it finds ways to circumvent the traditional speed-accuracy trade-off, resulting in a marginal loss in accuracy or optimality in exchange for a large gain in speed and resource conservation [23–26]. Studies have also shown that satisficing in humans is mediated by the use of heuristics [23, 24]. For the purpose of this research, heuristic path planners are defined as those planners which are capable of learning the way in which humans employ satisficing strategies [27] while attempting to navigate an unknown environment with limited sensory information.

Ferrari S., first proposed the treasure hunt problem in [28] to describe the fundamental problem of sensor path planning. Traditionally, robot path planning algorithms have been designed to determine an optimal path, given the initial and final configurations and kinodynamic constraints [29, 30]. However, sensor path-planning typically addresses a different paradigm with the aim of obtaining measurements from either a subset or all of the possible targets in the workspace [31–33], making many of the approaches commonly used in robot path planning inapplicable to sensor path planning. The treasure hunt problem that we are motivated to solve is a target classification problem introduced in [34] to describe the problem of planning a path and measurements of a sensor installed on a ground robot, in order to classify multiple targets in an obstacle-populated environment, where spatially distributed targets are represented by hidden hypotheses variables which can never be observed; but only inferred from a set of correlated cues.

To overcome the limitations of the methods mentioned above and to develop adaptive heuristic path planners aimed at solving the treasure hunt problem efficiently, this thesis presents the required building blocks for a path planning problem with obstacle avoidance, while allowing measurements on the necessary targets under environmental pressures. As chapter 3 would elaborate, the building blocks consist of random walk and area coverage algorithms which can be fused together intelligently to create an adaptive algorithm capable of creating satisficing strategies, similar to those used by biological organisms like humans, for a robot navigating an unknown environment under external pressures like fog, time, and ”money”. The chapter further shows the simulations of the aforementioned strategies in the Webots® simulator and analyses their performance on the treasure-hunt problem. Further, it also talks about the dataset created for training detectors and classifiers under foggy conditions, while analysing the performance of existing classifiers under the influence of fog and the target distance. Finally, the thesis outlines the potential future steps that can be taken to create an adaptive heuristic path planner.

CHAPTER 2

PROBLEM FORMULATION AND ASSUMPTIONS

This thesis considers the problem of planning the path of a directional sensor onboard an unmanned ground vehicle (UGV) to classify a set of targets in an unknown obstacle-populated workspace under environment pressures like fog. The workspace, which is denoted by, $\mathcal{W} \subset \mathbb{R}^2$ is assumed here to be a compact subset of a Euclidean space, populated with n fixed targets, at unknown locations, denoted by $\mathcal{T}_1, ..., \mathcal{T}_n$. The probabilistic model of sensor measurements and classification has been learned from data and expert knowledge using a Bayesian Network (BN). BN represents a joint probability mass function (PMF) by a directed acyclic graph (DAG), $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. The node set $\mathcal{N}$ consists of M discrete feature random variables $X_1, ..., X_M$ and a categorical random variable Y. Each feature random variable $X_j \in \mathcal{N}$ is associated with a finite set of mutually exclusive states $\mathcal{X}_j = \{x_1, ..., x_J\}$, and the states of the categorial random variable Y is $\mathcal{Y} = \{y_1, y_2\}$. Each node in $\mathcal{N}$ is associated with a conditional probability table (CPT) of BN parameters. The set of edges $\mathcal{E}$ represents conditional dependencies between nodes in $\mathcal{N}$, and expresses the joint PMF as

$$p(\mathcal{N}) = p(X_1, ..., X_M, Y) = p(Y|pa(Y)) \prod_{j=1}^{M} p(X_j|pa(X_j)) \tag{2.1}$$

where $pa(X_j)$ is the parent set of X_j , such that $\forall X_i \in pa(X_j)$ there exists a directed arc $(i, j) \in \mathcal{E}$ and similarly for $pa(Y)$.

$\mathcal{W}$ is also populated with n static obstacles $\mathcal{B}_1, ..., \mathcal{B}_n$ whose geometry and positions are not known a priori. The UGV's geometry is described by a rigid object $\mathcal{A} \subset \mathbb{R}^2$ that is a compact subset of $\mathcal{W}$. The UGV is equipped with a directional sensor, such as a camera, with a field of view (FoV) denoted by $\mathcal{S}$. A configuration vector $\mathbf{q} = [x, y, \theta]^T \in C$ specifies the position of the UGV's geometrical center and orientation of the UGV with respect to a fixed Cartesian frame $\mathcal{F}_W$ embedded in $\mathcal{W}$ with origin O_W, where configuration space $C \subset \mathcal{W} \times (-\pi, \pi]$ denotes the space of all possible values of the configuration vector $\mathbf{q}$. $\mathbf{q}$ also specifies a moving Cartesian frame $\mathcal{F}_A$ embedded in $\mathcal{A}$ with origin O_A defined at the position of UGV. The UGV is assumed to obey the a unicycle robot kinematics. The control input is $\mathbf{u} = [v\ \omega]^T \in \mathcal{U} = \{v, \omega \,|\, 0 \leq v \leq v_m, 0 \leq \omega \leq \omega_m\}$, where $v_m, \omega_m \in \mathbb{R}^+$ are the maximum permissible velocity and angular velocity respectively. The unicycle model is

$$\dot{\mathbf{q}}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} cos\theta(t) & 0 \\ sin\theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} = \mathbf{g}[\mathbf{q}(t)]\mathbf{u}(t) \tag{2.2}$$

A C-obstacle is a subset of C that has collisions with at least one obstacle, denoted as $C\mathcal{B}_i \equiv \{\mathbf{q} \in C \,|\, \mathcal{A}(\mathbf{q}) \cap \mathcal{B}_i \neq \emptyset\}$, where $\mathcal{A}(\mathbf{q})$ denotes the subset of $\mathcal{W}$ occupied by $\mathcal{A}$ with UGV configuration $\mathbf{q}$. Then, the union of all C-obstacles obtained from B is defined as the C-obstacle region, i.e., $C\mathcal{B} = \bigcup_i C\mathcal{B}_i$. The free configuration is the complement of the C-obstacle region, i.e., $C_{free} = C \setminus C\mathcal{B}$ [35]. The robotic sensor is free to rotate and translate in this free configuration

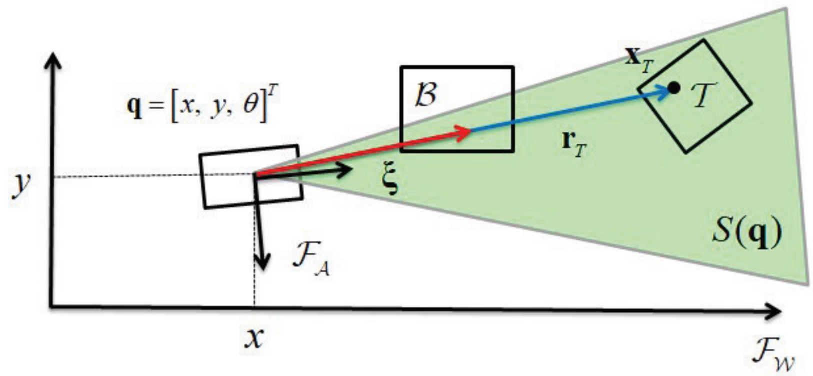

Figure 2.1: Line of sight visibility without fog.

space, while remaining fixed on the UGV.

A directional sensor is influenced by occlusions caused by obstacles in its line-of-sight (LoS), in front of the target of interest $\mathcal{T}_i$, and the density of fog in the environment. The FOV of a sensor with dynamic state $\mathbf{q}$ operating in an environment with fog density F can be modeled as a compact subset $\mathcal{S}(\mathbf{q}, F)$ of $\mathcal{W}$ occupied by $\mathcal{S}$ when the UGV is at configuration $\mathbf{q}$. Let $\mathbf{x}_T$ be the position of an interest point in target $\mathcal{T}_i$. The coordinate of the point of interest in $\mathcal{F}_A$ is $\mathbf{r}_T = \mathbf{x}_T - \mathbf{q}$. $\mathbf{x}_T$ is in the LoS of the sensor at $\mathbf{q}$ if there are no points in the obstacle region $\mathcal{B}$ that are co-directional with $\mathbf{r}_T$ and closer to $\mathbf{q}$ than $\mathbf{x}_T$ and is within the visibility range, V_F for a given fog density F , or

$$\nexists\, \boldsymbol{\xi} \in \mathcal{B} \text{ s.t. } \boldsymbol{\xi} \cdot \mathbf{r}_T = ||\boldsymbol{\xi}||||\mathbf{r}_T|| \text{ and } ||\boldsymbol{\xi}|| < ||\mathbf{r}_T|| \text{ and } ||\boldsymbol{\xi}|| < ||V_F|| \tag{2.3}$$

where $\boldsymbol{\xi}$ is defined with respect to $\mathcal{F}_A$, and it is assumed that $\mathcal{T}_i \cap \mathcal{B} = \emptyset$. Target $\mathcal{T}_i$

is visible to the sensor if $\mathcal{S}(\mathbf{q}) \cap \mathcal{T}_i \neq \emptyset\}$ and $\mathcal{T}_i$ is in the LoS of the sensor. C-target of target $\mathcal{T}_i$ is defined as a subset of C such that $C\mathcal{T}_i = \{\mathbf{q} \in C \mid \mathcal{S}(\mathbf{q}, F) \cap \mathcal{T}_i \neq \emptyset\}$ and the LoS test is satisfied for all obstacles. Under the influence of fog, passing the LoS test does not guarantee the target visbility. Therefore, the path planning problem for visiting targets under foggy conditions can be defined as:

"Plan a trajectory in C_{free} such that there exists a C-target $C\mathcal{T}_i$ in the trajectory given the environmental pressures and sensor measurements."

CHAPTER 3

SIMULATIONS AND RESULTS

This chapter introduces the workspaces designed for the treasure-hunt problem. In addition to this, each of the designed workspaces have different possible target layouts which correspond to different target densities. The different path planning approaches used have been simulated and tested on a subset of the designed workspaces. In addition to this, simulations have been carried out to test the performance of the classifiers against changing levels of fog and target distances. Further, information gain from each additional level of classification has also been estimated. All of the simulations have been carried out in the Webots® simulator.

3.1 Sensor parameters and simulation environment design

This section includes details about the different sensor parameters and the details about the different designs of the different workspaces which have been created.

3.1.1 Sensor parameters

The sensor used for observing the targets is a camera. The open angle of the camera, also known as half of the sensor field-of-view, α is $\pi/4$. Theoretically, the detection range of the camera is infinite. However, the detection has been contracted to be bounded between r_{min} and r_{max} which are 0.1 meters and 2.0 meters respectively. These parameters have been determined from the sensor applied in the immersive satisficing experiments which have been carried out previously by Ferrari [36, 37]. These sensor specifications also support the on-board computer vision algorithms for target feature extraction and classification.

3.1.2 Environment design

The environment design is predominantly influenced by the following parameters, the size of the workspace, the layout of the obstacles in the worksapce, the number of targets or the target density in the workspace, their types, and the level of fog. The design for each of which has been described below.

Workspace design

The workspace design consists of two main parameters - the dimensions of the workspace and the layout of the obstacles in the workspace. The dimensions of all the workspaces designed have been kept consistent with those of the human active satisficing experiment, which are 9 metres in length and 9 metres in width. This has been done in order to facilitate comparison of results of the heuristic path planners developed and the human experiments conducted to study their satisficing strategies.

In this thesis, three workspace designs have been proposed. They have been designed to resemble a typical house, a warehouse, and a maze. Figure 3.1 shows the workspace that's been designed to resemble a typical house. It can be seen that the workspace has been divided into four quadrants to resemble the division of a house into four rooms. The bottom-left room is designed to resemble a typical bedroom with a bed, an arm-chair, a study desk, and a wardrobe. Next to it,is a dining room with a table and a potted plant. In the top-left corner of the workspace, a living-room can be found which includes a couch, a show-case, another storage shelf, and a painting. The final room represents a room for gatherings which includes four chairs and a painting. The lighting for each of the rooms is provided by a ceiling light of white color in the middle of the room to ensure even distribution of light. The color of the lights play an important role in our simulations as changing them would lead to a hue being cast on all the objects which would affect the results of feature classifiers.

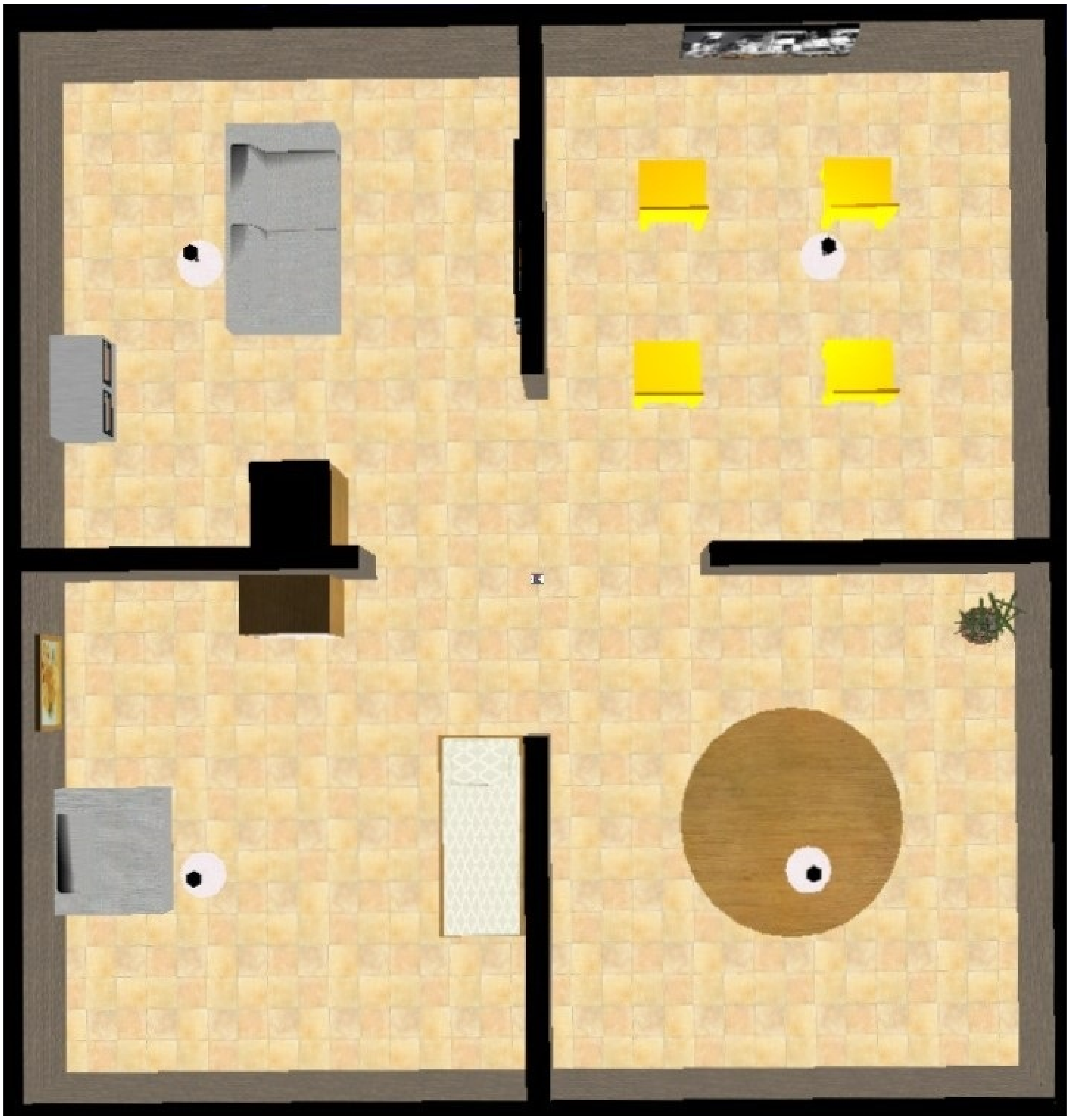

Figure 3.1: Overview of Workspace 1 representing a home setting

Figure 3.2 represents a warehouse with an attached office. This workspace has been tessellated into three sections. Two of the smaller, equally size sections are designed to resemble a shipment receiving room in a warehouse, and an office. The larger section is designed to resemble a storage section with shipment boxes and barrels placed all around. Similar to the first workspace design, shown in Figure 3.1, this workspace is also illuminated by four evenly distributed ceiling lights of white color to ensure even lighting conditions.

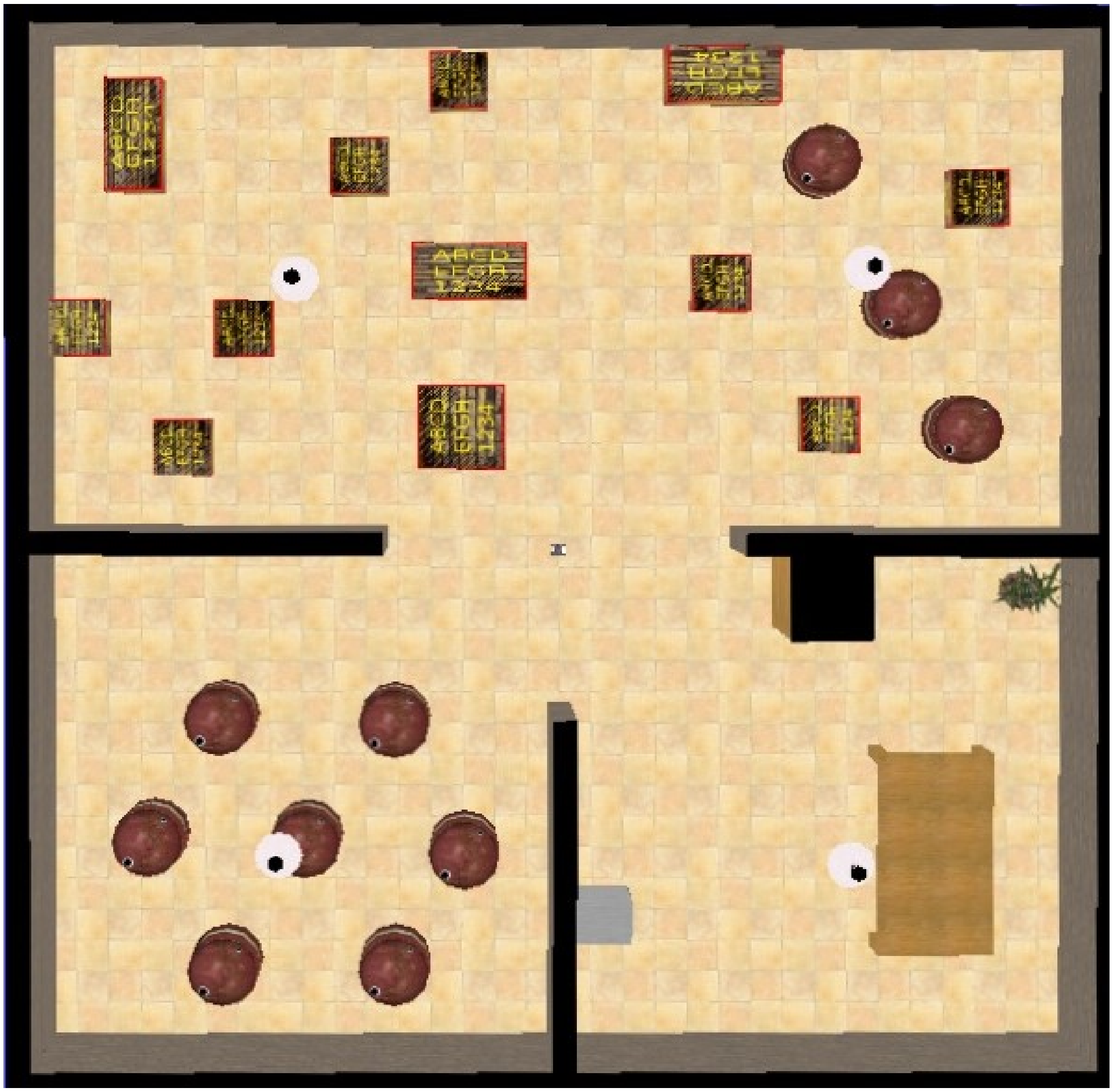

Figure 3.2: Overview of Workspace 2 representing a warehouse setting

Figure 3.3 represents a maze. The thin walls used in the maze, present a unique challenge such that their accurate detection by sensors like lidar or radar can be challenging when viewed from the side. In addition to using thing walls in the maze, obstacles like shipping crates and potted plants are also present in this environment. These obstacles are placed in a manner that they provide the robot with a narrow path to move from one place to another. Such conditions have known to be challenging for algorithms like probabilistic roadmaps (PRM) and rapidly-exploring random trees (RRTs). These cases can be used to analyze the performance of heuristic path planners where modern path planners are known to struggle.

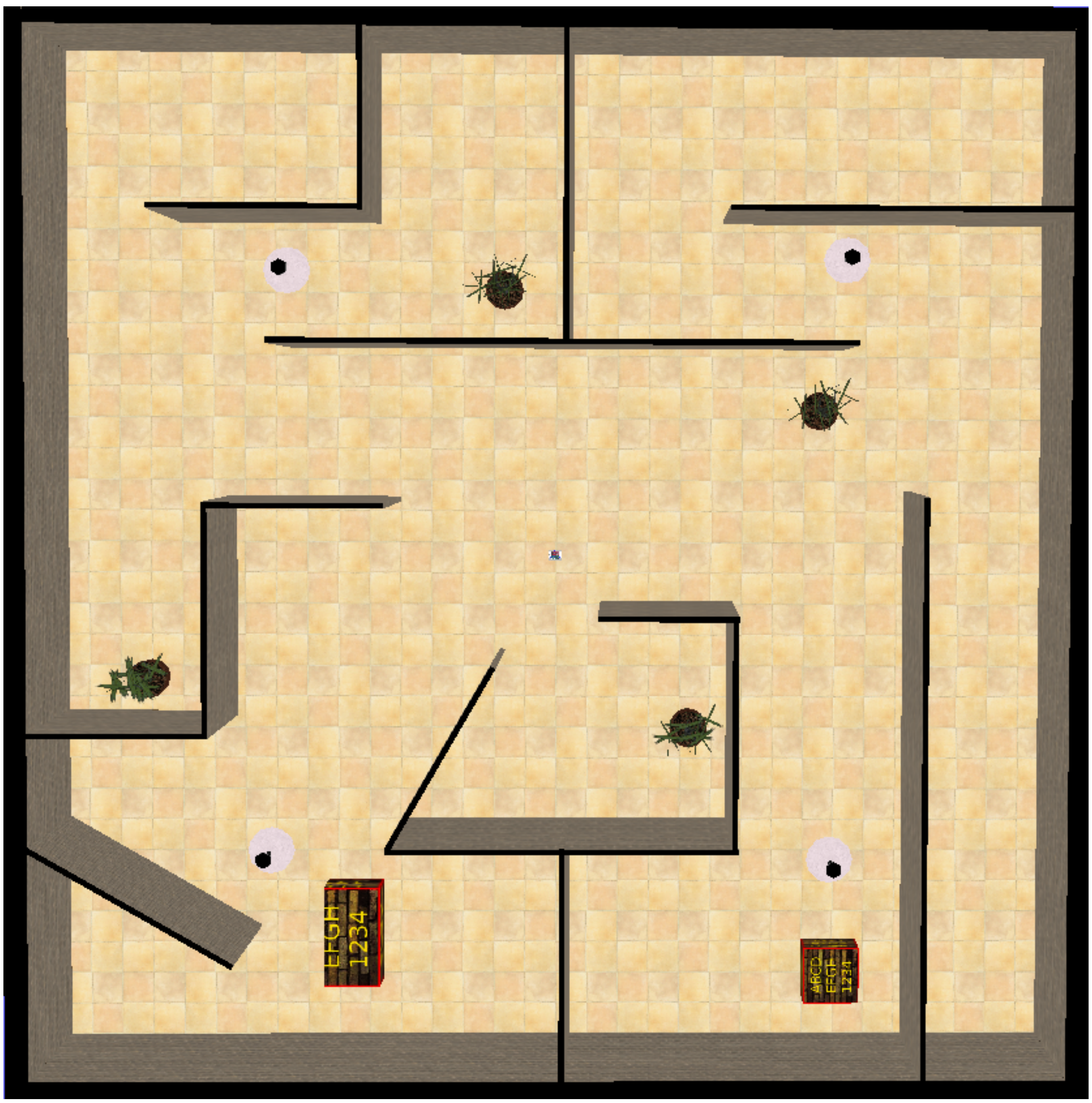

Figure 3.3: Overview of Workspace 3 representing a maze

Fog densities

In this thesis, a significant emphasis has been placed on the effects of fog on the functioning of path-planners and feature classifiers. In order to study these effects, it was imperative to characterize the different fog densities.In the Webots® simulator, the levels of fog are defined by visibility range, such that a higher visibility range corresponds to a lower fog level. The simulator offers three different types of fog, ”LINEAR”, ”EXPONENTIAL”, and ”EXPONENTIAL2”. Each of these levels corresponds to the function used for blending the fog with ”LINEAR” being most artificial looking fog and ”EXPONENTIAL2” using a squared exponential blending function to return the most natural looking fog. In these simulations ”EXPONENTIAL2” setting has been used for fog. Fog has been characterized into five different levels which are no fog, haze, light fog, medium fog, and dense fog. The visibility ranges corresponding to the levels of fog has been tabulated in Table 3.1

Fog level	**Visibility (metres)**
No fog	∞
Haze	1.6
Light fog	1.2
Medium fog	0.8
Dense fog	0.4

Table 3.1: Fog densities and the corresponding visibility levels

Figure 3.4 shows a watermelon placed at 100cm from the camera at varying levels of fog. It can be seen that the watermelon becomes invisible to the camera under the dense fog conditions.

Figure 3.4: Snapshot of a watermelon at 5 different fog densities

Target layout

The layout of targets has the potential to greatly impact the number of targets that have been discovered by the robot under any given initial conditions. All the proposed target layouts have been designed such that they are largely uniformly distributed across all the workspace. For each of the three worksapces, four different target layouts have been proposed, which correspond to different target densities. The sparse target configuration has only three targets distributed across the workspace, while the few targets configuration has five. The medium targets configuration has been designed to have thirteen targets in the workspace and the many targets configuration is designed to have fifteen targets distributed across the workspace. Sections a-d in Figures 3.5 - 3.7 contain the target four different target layouts for each of the workspaces, in increasing order of the number of targets in the worksapce.

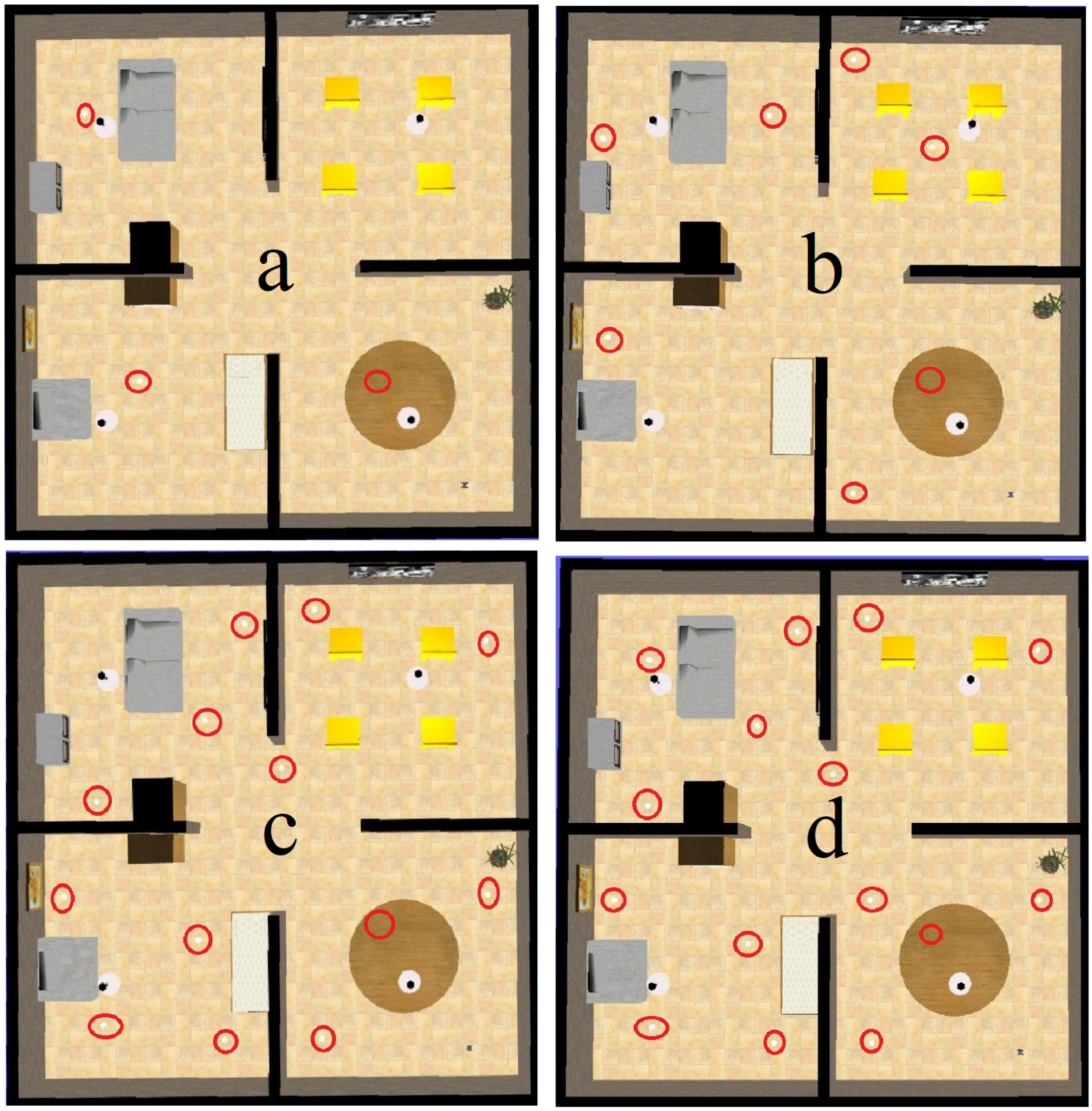

Figure 3.5: Different target layouts for workspace 1

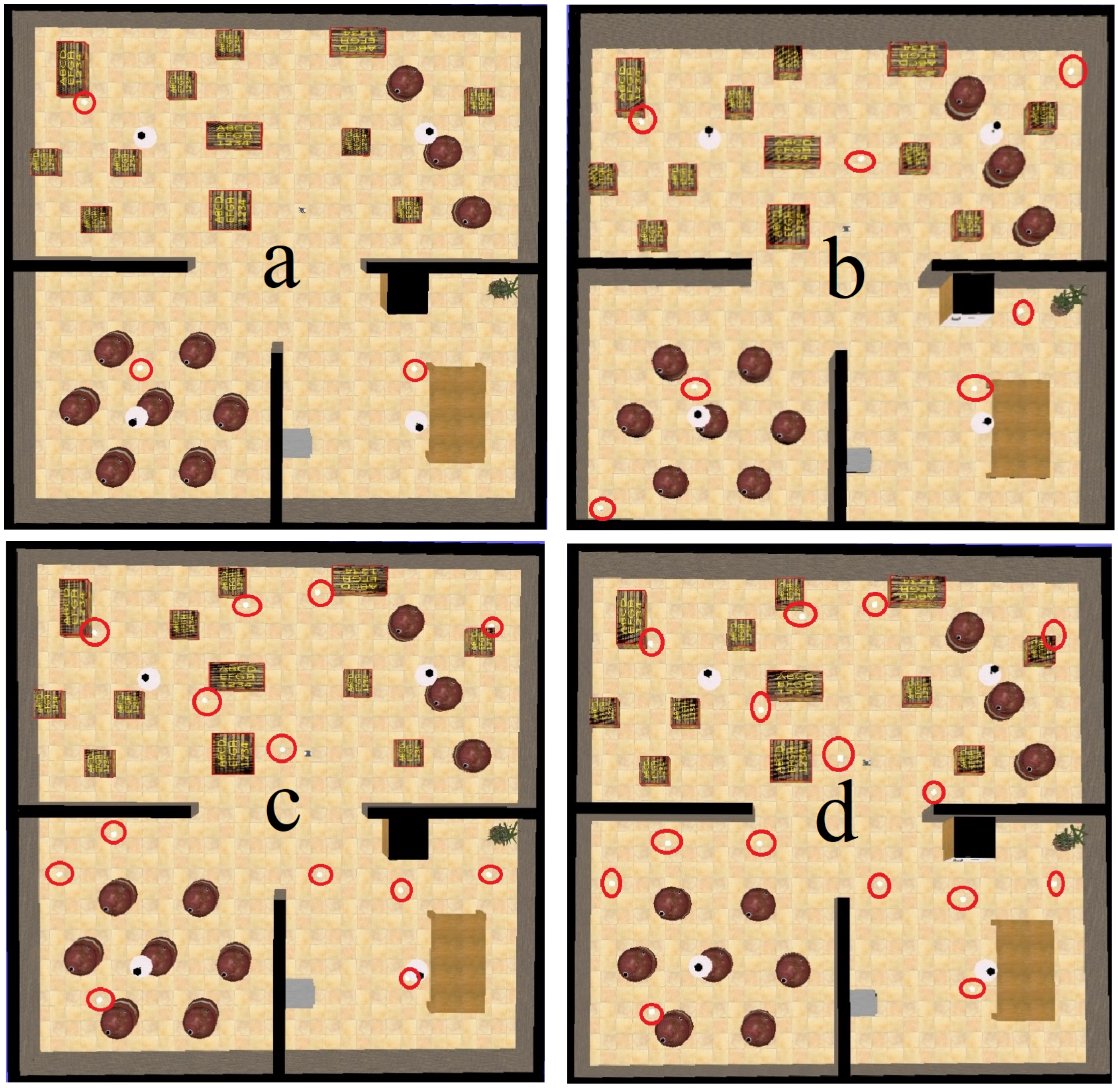

Figure 3.6: Different target layouts for workspace 2

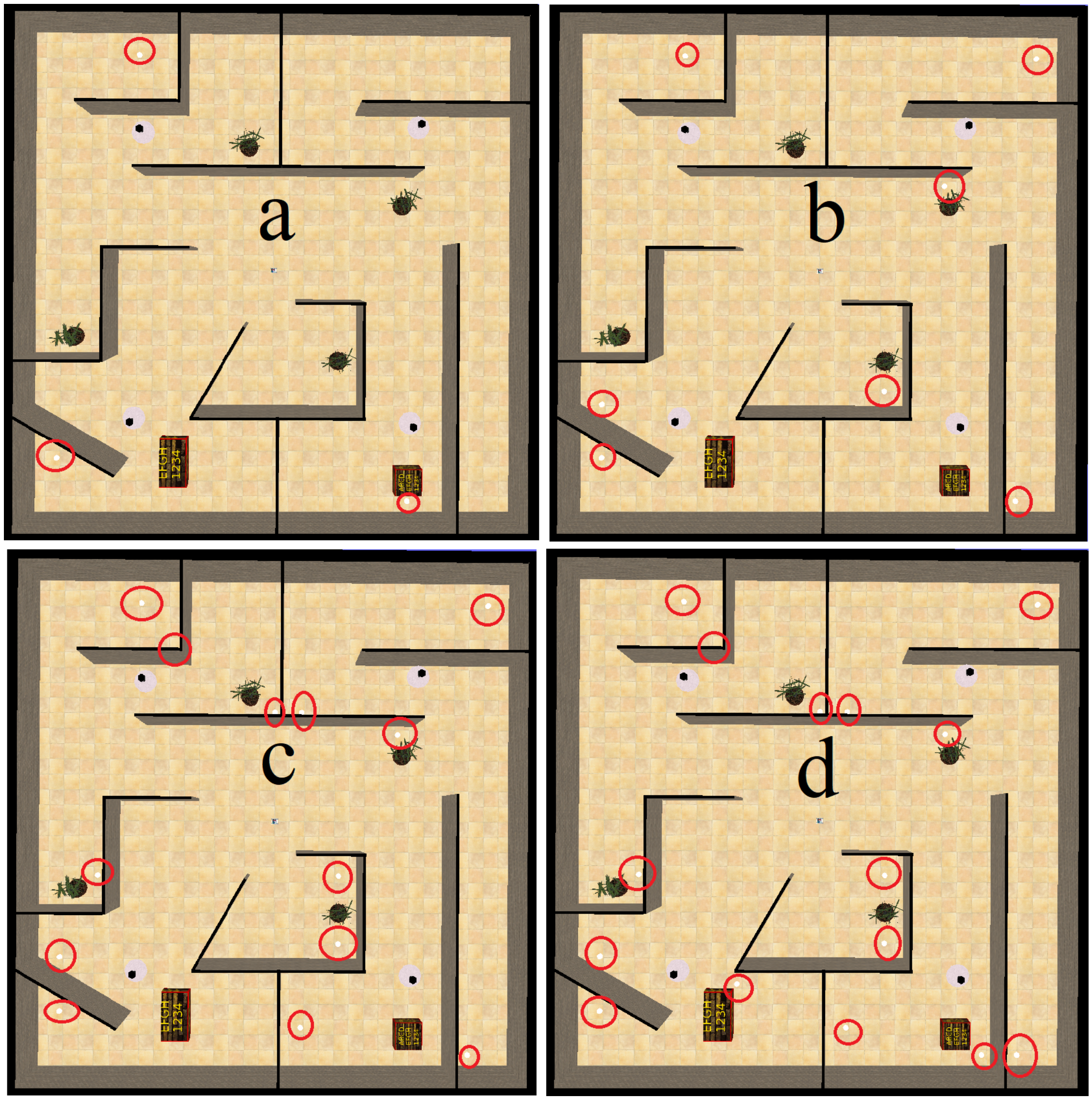

Figure 3.7: Different target layouts for workspace 3

Target features

In the simulations, each target is initially shown as a cylinder of gray color with a radius of 0.05 metres and a height of 0.1 metres. Each target is known to have three levels of features. The first level is the shape of the target, which could either be a sphere or a box. The second level of features is the color of the object which could either be green or orange, if the target is known to be spherical or brown or black, if the target is known to be a box. The final level of features is the texture of the target such that they resemble either a plain surface or a striped surface. In the case of a green sphere, the potential target identities are apple or watermelon. For an orange sphere, the potential target identities are orange or basketball, while the potential target identities for a brown box are cardboard box or wooden box, and the possible target identities for a black box are either a computer or a book. Figure 3.8 shows all the possible target features. In the first column are all the target features whose parent is a box, from top-to-bottom, they are black box, brown box, computer, book, cardboard box, and wooden box respectively. In the second column are all the features whose parent is a sphere, from top-to-bottom, they are green sphere, orange sphere, watermelon, apple, orange, and basketball respectively.

3.2 Simulations in MATLAB® and Webots® environments

This section describes in detail the steps for revealing different target features, the algorithms proposed for addressing the treasure-hunt problem and the workspace designs used for testing the performance of the proposed algorithms.

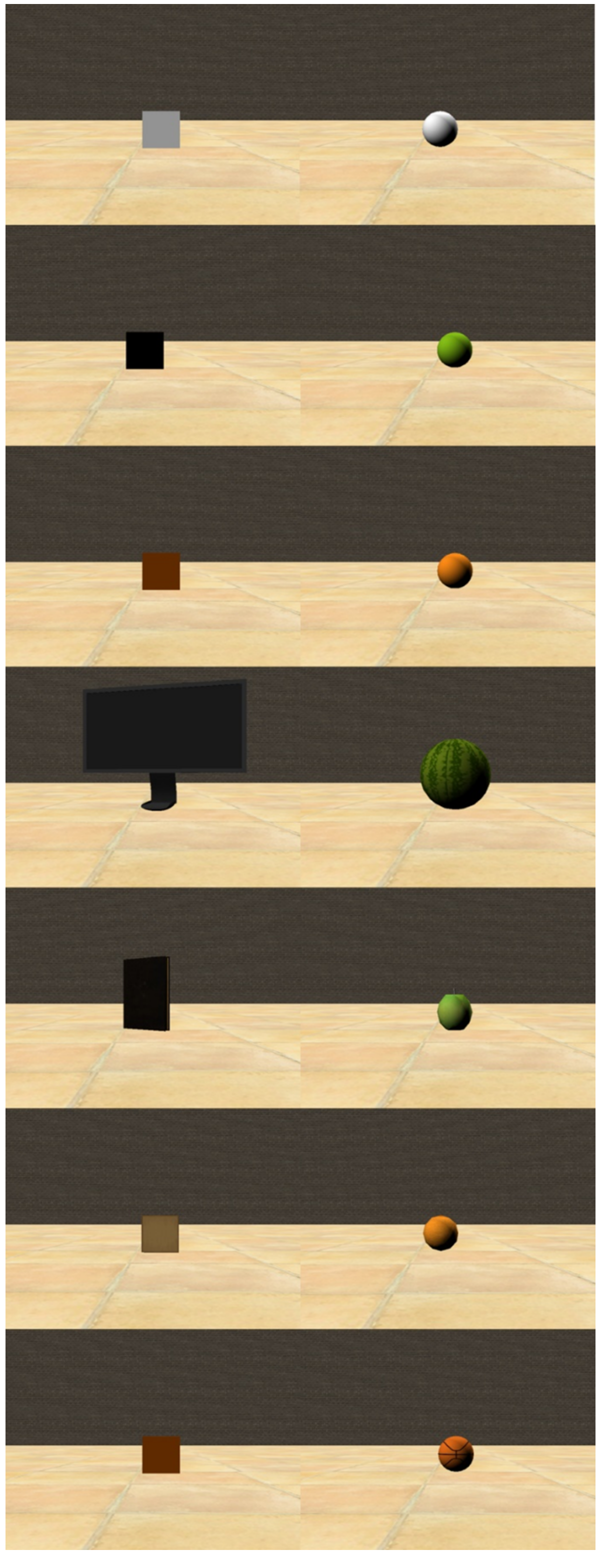

Figure 3.8: All possible target features

3.2.1 Evaluation of performance of the target feature classifiers

The process of evaluating the performance of the target feature classifiers can be broken down into two steps. The first being the updation of the target cues to reveal more features and the second being the updation of environmental variables like the level of fog, the target distance and its orientation. The details of each of the two steps have been described in greater detail in the sub-section below.

Revealing target features

As mentioned previously, each target has three levels of features. The first corresponds to its shape, while the second and third correspond to color and texture respectively. There are a total of eight possible identities which have been enumerated as shown in Table 3.2. Based on the enumeration of each target, the shape, color, and texture are displayed every time the target features are updated.

Target identity	Shape	Color	Texture
1	Sphere	Green	Apple surface
2			Watermelon stripes
3		Orange	Orange surface
4			Basketball stripes
5	Box	Brown	Cardboard box surface
6			Wooden box stripes
7		Black	Computer surface
8			Book stripes

Table 3.2: Target identities and their corresponding features

Estimating classification performance

In order to evaluate the performance of the classifiers, it is important to have working knowledge of the different classifiers. In our case, we only use binary classifiers. Each of these binary classifiers is a support vector machine (SVM) trained on the histogram of oriented gradients (HOG) features extracted from the entire image. For classifying all the possible target features, seven binary classifiers are being used, as listed in Table 3.3.

Target feature	Binary classifier outputs
Shape	Sphere-Box
Color	Orange-Green
	Brown-Black
Texture	Apple surface - Watermelon stripes
	Orange surface - Watermelon stripes
	Cardboard box surface - Wooden box stripes
	Computer surface - Book stripes

Table 3.3: List of binary classifiers for different features

For the purposes of this work, the target distance, $\mathbf{r}_T$ has been defined as the horizontal distance between the imaging sensor and the center of the target as shown in Figure 2.1

To understand the effects of fog densities and the target distance, the performance of the classifiers has been evaluated over nine different, uniformly distributed target distances with five different levels of fog. The list of the target distances can mathematically be defined as $\mathbf{r}_T = \{t \in \mathbb{R} \mid t = 0.2i,\ 2 \leq i \leq 10\}$ At each combination of target distance and fog level, ten batches of a hundred trials have been simulated and classified. Classifying one trial implies that all three features of the object have been classified independently. For classifying a given target, the simulation methodology is as follows - initially, each target is of an unknown shape represented by a cylinder in the simulations. The target is then updated to reveal the shape and the shape classifier is called. Then, the target is updated to reveal its color, in addition to the already revealed shape. Now, depending on the ground truth of the target identity, one of the two color classifiers is called. After classifying the object color, the target is updated for a third time to reveal its texture. Now, again, depending on the ground truth of the target texture, the appropriate texture classifier is called. Once the target texture has been classified, it is then reset to a cylinder to run another trial. This process ensures that the performance of the preceding classifier does not affect the results of the current classifier.

Dataset for training object detectors and classifiers

By simulating the different objects to analyze the performance of the classifiers, we have also been able to create a dataset of all the different objects at their various feature levels, orientations, target distances, and fog densities. Cumulatively, this becomes a dataset of 108,000 images. The dataset can be broken down into three mutually exclusive categories - shape, color, and texture. Under the shape category, there exist 36,000 images of spheres and boxes of smooth texture and gray color. Out of these 36,000 images, 17,900 are spheres and 18,100 are boxes. There are 36,000 more images for the purposes of color classification which consists of 8831 green spheres of smooth texture, 9069 orange spheres of smooth texture, 9150 brown boxes of smooth texture, and 8950 black boxes of smooth texture as well. The dataset for texture classification consists of another

36,000 images. This portion of the data consists of 4460 Apples, 4371 Watermelons, 4585 Basketballs, 4484 Oranges, 4558 Cardboard boxes, 4592 wooden boxes, 4435 books, and 4515 computers. The dataset does not contain any images for targets beyond 140cm in the case of medium fog and beyond 80cm for the case of dense fog because at those distances, the targets become invisible due to the fog pressure.

3.2.2 Path planning algorithms for adverse weather conditions

Adverse weather conditions like fog or heavy rain produce challenging conditions for vision based sensors like cameras and lidar wherein the effective range of the sensor is reduced. The effective range of the sensor plays an important role in the robot being able to localize itself in the environment. With extremely limited range of a sensor, it is very challenging to localize the robot. The algorithms discussed in this section are random-walk and area-coverage algorithms, which have been implemented in a manner that they search the unknown workspace for targets of interest while avoiding the obstacles.

Simulations of random-walk algorithm

In the random walk algorithm, the robot keeps moving straight unless it encounters an obstacle. When the robot encounters an obstacle, it turns away from it by an arbitrary amount until it clears the obstacle and then continues moving forward. When the robot detects a visible target, the robot stops to perform target classification and once all the features have been classified, the target node is deleted from the simulation and the robot continues to move forward. Clicking here loads a video of the bird's eye view of the robot being controlled by the random walk algorithm without the presence of fog in order to visualize the robot motion. Clicking here loads a video of the robot preforming the random walk motion under medium fog conditions. For this video, only the robot's first-person-view (FPV) is available due to the presence of fog. The results in this thesis analyze the performance of this algorithm in workspaces 1 and 2 under the medium fog conditions.

Simulations of area-coverage algorithm

In the area coverage algorithm, the robot preforms a sweeping/zig-zag motion. The robot keeps travelling straight until it encounters an obstacle. All the obstacles in the workspace can be classified under two broad categories; workspace corners or not. If the obstacle encountered is a workspace corner, which is defined as having an obstacle in the front and either on the left or the right sides of the robot, the robot will turn away from the corner and continue in a striaght line. However, if the obstacle encountered is not a corner, it will turn around and continue the sweeping motion. By doing this, the direction of sweep changes by $\pi/2$ radians every time a corner is detected, whereby the robot visits the previously unexplored parts of the workspace. This method explores a given section of the workspace more thoroughly while trading off the ability to explore more sections of the workspace in a limited amount of time as compared to the previous random-walk algorithm. Clicking here loads a video of the bird's eye view of the robot being controlled by the area coverage algorithm without the presence of fog in order to visualize the robot motion. Clicking here loads a video of the robot preforming the area coverage motion under medium fog conditions. For this video, only the robot's first-person-view (FPV) is available due to the presence of fog. The results in this thesis analyze the performance of this algorithm in workspaces 1 and 2 under the medium fog conditions.

3.3 Results

This section talks about the results from the above simulations and the expected entropy reduction study.

3.3.1 Performance of different feature classifiers

The performance of feature classifiers can be divided into three sections depending on the features, shape, color, and texture namely. The results from 360 batches of simulations for each of which have been shown in the following subsections -

Performance of shape classifier

From Figure 3.9, it can be seen that the average values of the accuracy of classifications under the no fog conditions are at 100% upto a target distance of 80cm. With an increasing fog pressure, it can be noted that the classifier accuracy drops to its worst-case performance of about 50%. This indicates that the classifier has potentially not been trained with targets at distances greater than 80cm. It can be seen that the optimal target distance for shape classification, if the visibility is greater than 0.8m, is 80 cm. The results for target distances greater than 80cm under dense fog conditions and the results for target distances greater than 140cm under medium fog conditions have not been plotted due the targets being completely covered by fog and being invisible.

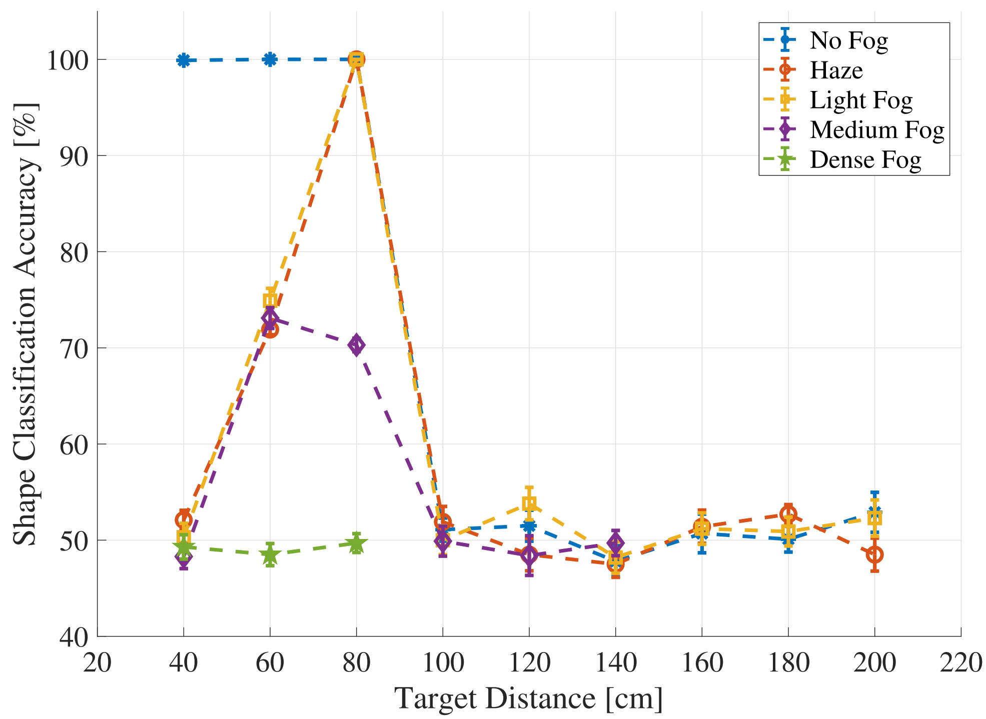

Figure 3.9: Change in performance of shape classifiers with varying fog densities and target distances

Performance of color classifiers

From Figure 3.10, it can be noted that, in general, the accuracy of the classifications seems to be following a downward trend. However, intermittently, it can be noted that there is a rise in the classification accuracy with an increase in the target distance. With an increase in the density of fog, the accuracy of the classification of the target classifier seems to be reducing. However, it can be noted that the classifications under foggy conditions out-perform the classifications under the no-fog condition when the target distance is greater than 120cm. This could be attributed to the brown walls in the simulation, visible only under no fog conditions, leading to incorrect color classification.

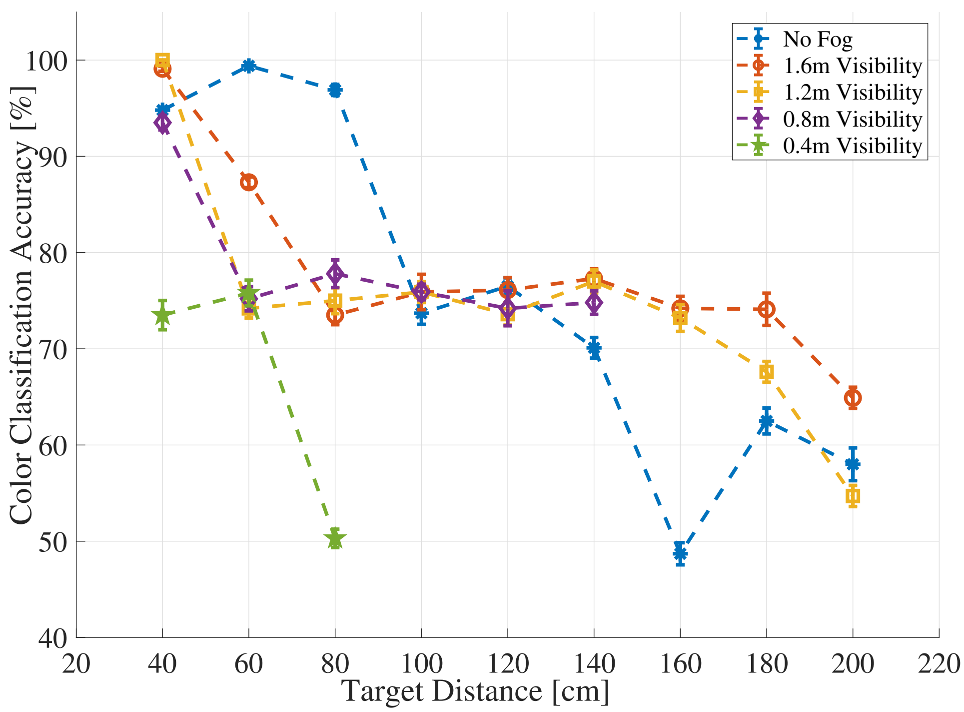

Figure 3.10: Change in performance of color classifiers with varying fog densities and target distances

Performance of feature classifiers

From Figure 3.11, it can be noted that the accuracy of texture classification reduces as the distance and fog level increases as expected. It can also be noted that the worst-case performance of this classifier is slightly lower than 50%. In the presence of fog, it can be noted that the best classifier performance was achieved when the target was closest to the camera. However, the best classification performance has been observed to be achieved at a target distance of 60cm under the no fog conditions. This could potentially be the result of some of the targets, such as the computers and books, being larger than the camera's FoV when they have been placed at a target distance of 40cm.

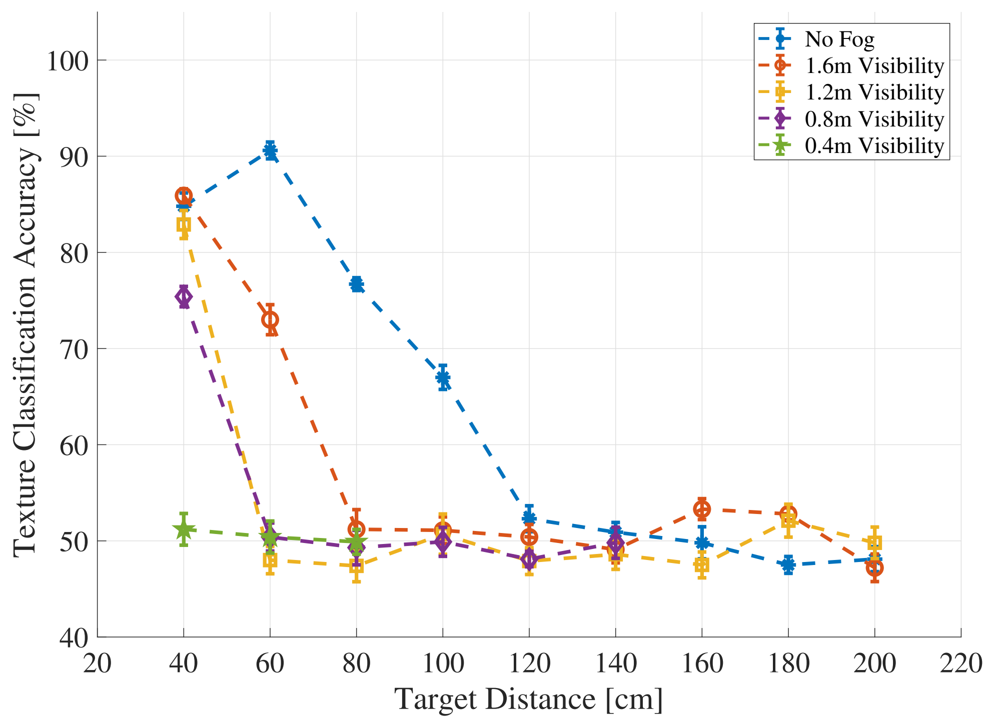

Figure 3.11: Change in performance of texture classifiers with varying fog densities and target distances

3.3.2 Expected entropy reduction study

In information-driven sensor planning, the planner navigates to collect valuable sensor measurements for target classification. Further, the process of measuring the updated cues and attempting to classify them has additional computational costs. Thus, it is important to estimate the quality of the measurements before obtaining them, such that the cost of those measurements do not exceed the information gained from them. Expected entropy reduction (EER) evaluates a measurement's reward by the ability to reduce the uncertainty in the classification variable. Entropy reduction is formulated using conditional entropy. The conditional entropy of a discrete and random variable Y given another variable Z is described by the expected value of the entropies of the conditional distributions over the range of the conditioning random variable [**?**]

$$H\left[Y|Z\right] = -\sum_{z}\sum_{y} P(y,z)\log_2 P(y|z) \tag{3.1}$$

where $H[\cdot]$ denotes the Shannon entropy, and $\sum_y$ denotes the marginalization over the range of Y.

Then entropy reduction, which is shown to be additive in [**?**], describes the reduction in uncertainty brought by a set of new measurements Z_j with prior information about Z_i as

$$\Delta\hat{H}[Y;Z_j|Z_i] = H[Y|Z_i] - H[Y|Z_i,Z_j] \tag{3.2}$$

The entropy reduction will represent the information value brought by a new set of measurements M_i. As mentioned above, it is important to note that the actual entropy reduction $\Delta\hat{H}[Y_i; M_i|\mathcal{F}_i]$ can not be determined without knowing the result of the measurement set M_i. Instead, the expected entropy reduction is applied to address this issue. Given a set of prior knowledge $\mathcal{F}_i$, the information value of a target is represented in terms of the EER brought by measuring all of its remaining features

$$EER(\mathcal{F}_i) = \begin{cases} H[Y_i|\mathcal{F}_i] - E_{M_i}H[Y_i|\mathcal{F}_i, M_i] & \text{if } l_i < L \\ \\ 0 & \text{if } l_i = L \end{cases} \tag{3.3}$$

where $E_{M_i}H[Y_i|\mathcal{F}_i, M_i] = \sum_{k=l_i+1}^{L} \sum_{q=1}^{N_k} [H[Y_i|x_{i,k} = x_{i,k}^q] P(x_{i,k} = x_{i,k}^q|\mathcal{F}_i)]$. The conditional entropy $H[Y_i|\mathcal{F}_i]$ and $H[Y_i|x_{i,k} = x_{i,k}^q]$ are computed using the definition in Eq.3.2 and the posterior PMF where all the probabilities have been acquired using the BN CPTs. The conditional probabilities of the different target features has been shown in Figure 3.13 and information gain calculated using this information has been shown in 3.4

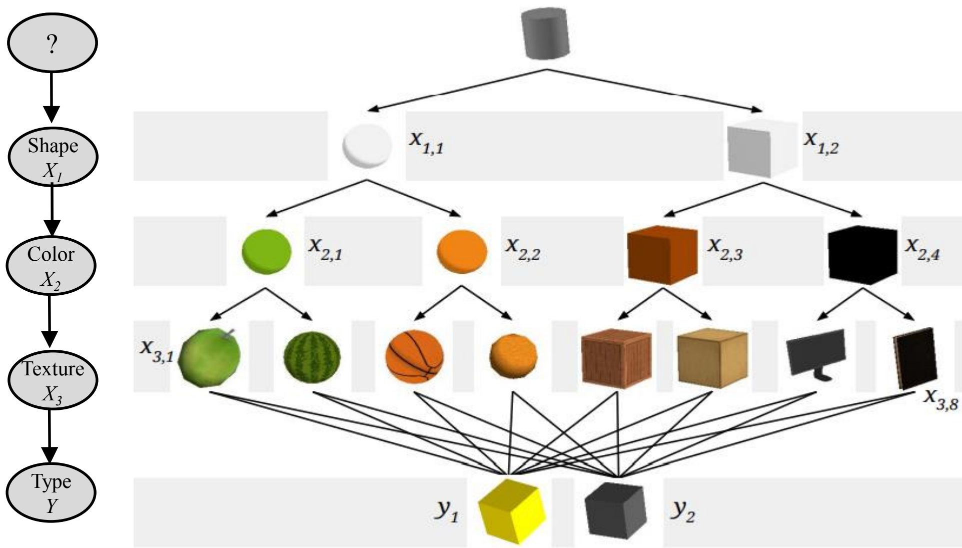

Figure 3.12: Bayesian Network for showing target features and treasure classifications

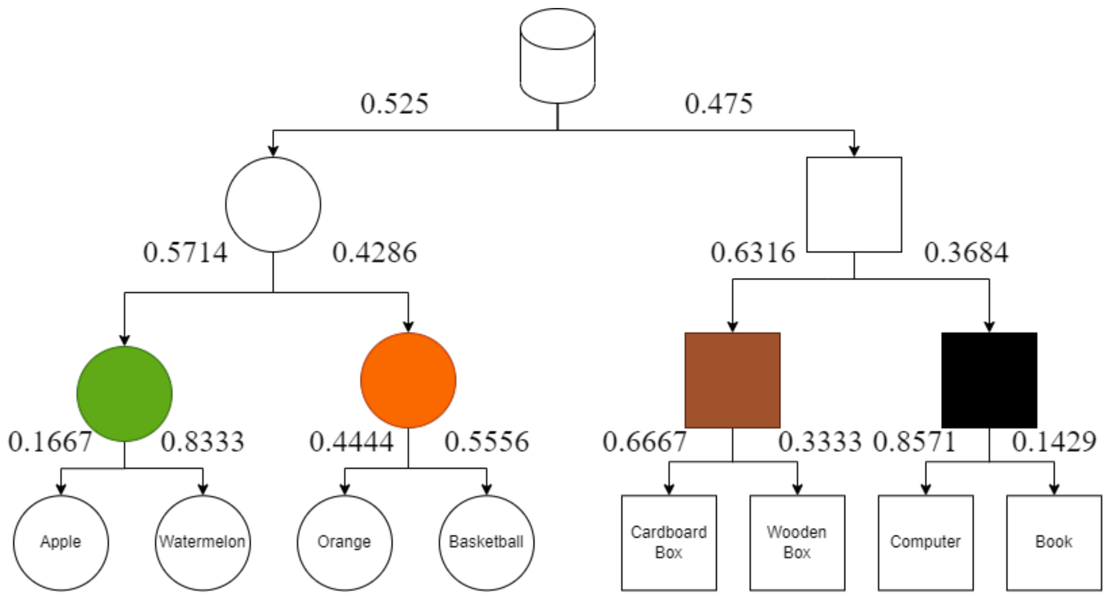

Figure 3.13: Conditional probabilities used for estimating information gain

<table>
<tr><th>Classification Level</th><th colspan="4">Information Gain</th></tr>
<tr><td>Shape</td><td colspan="4">0.0049</td></tr>
<tr><td>Color</td><td colspan="2">0.372</td><td colspan="2">0.0788</td></tr>
<tr><td>Texture</td><td>0.0623</td><td>0.5865</td><td>0.0593</td><td>0.2705</td></tr>
</table>

Table 3.4: Calculated information gain form each additional level of classification

From Table 3.4, it can be noted that there is a very low amount of information gained from the shape classifiers. The color classifiers also, have a low information gain. From the information gain of texture classifiers, it can be noted that classifying the texture of an orange sphere and that of a black box lead to relatively high information gains while classifying the texture of a green sphere and a brown box leads to low information gain. This knowledge can be further used in the path planners when performing target classification in order to analyze if the information gain from the additional classification step is greater than the cost of performing the said measurement and classification in order to more efficiently use the available resources.

3.3.3 Performance of path planning algorithms and classifiers

This section goes over the performance metrics being used to compare the overall performance , in the simulations, which includes the performance of the classifiers and that of the path planners. The feature and treasure classifications have been compared to estimate the feature classification performance and the number of targets detected and distance travelled are to estimate the performance of the path planners. Ideally, it is expected for the classification accuracies to be high while the distance travelled would be lower with a high number of target detections. The following are the performance metrics observed for the each of the eight simulations in the two workspaces.

Number of targets discovered

For each case, the following Figures 3.14 and 3.15 represent the number of targets detected for each of the different algorithms against the total number of targets present in the workspace.

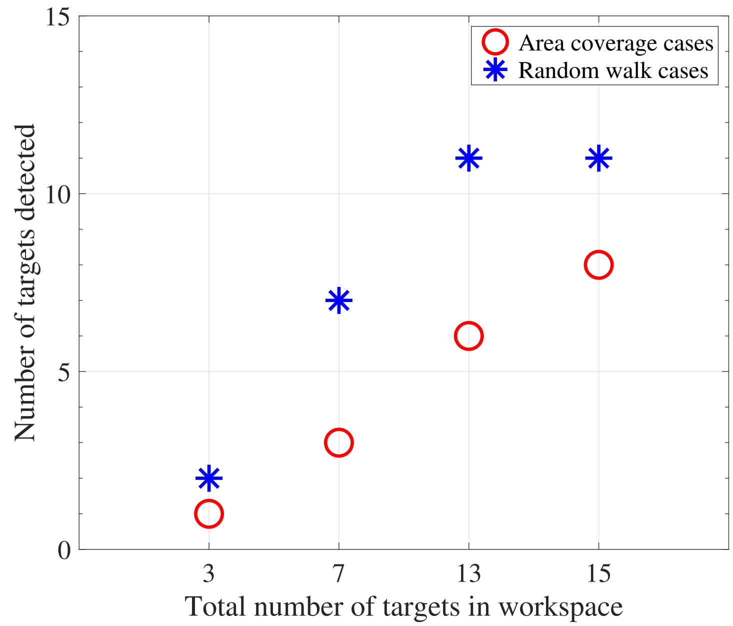

Figure 3.14: Number of targets detected in workspace 1 for each of the planning algorithms for different simulations

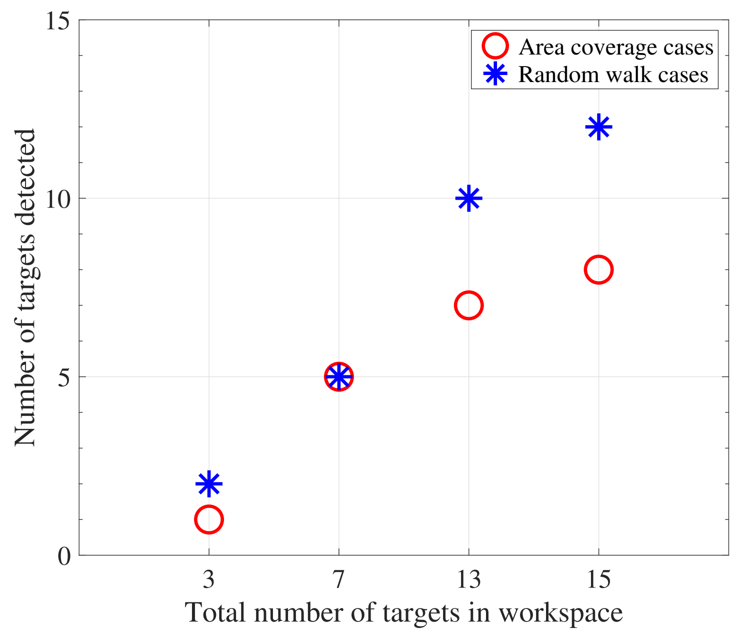

Figure 3.15: Number of targets detected in workspace 2 for each of the planning algorithms for different simulations

From the above Figures 3.14 and 3.15, it can be noted that, on average, random walk tends to detect more targets as compared to the area coverage algorithm. This is so because the area coverage algorithm's performance is constrained by the amount of time the simulations are run for.

Distance travelled

At each timestep, the distance travelled by the robot is updated. In addition to looking at just the distance travelled by the robot, looking at the distance travelled per target discovered would help us understand which algorithm works better. Table 3.5 shows the distance travelled by the robot and the number of targets discovered by each of the algorithms in the first workspace and table 3.6 shows the same for the second wrokspace.

Planner	Total Targets	Distance Travelled (m)	Targets Detected
Area coverage	3	310.61	1
	7	224.18	3
	13	246.38	6
	15	205.78	8
Random walk	3	274.66	2
	7	164.87	7
	13	291.69	11
	15	236.86	11

Table 3.5: Distance travelled and the number of targets discovered in workspace 1 for different number of total targets

Figure 3.16 plots the distance travelled per target discovered in workspace 1 for random walk and area coverage algorithms.

Planner	Total Targets	Distance Travelled (m)	Targets Detected
Area coverage	3	271.34	1
	7	265.94	5
	13	134.70	7
	15	216.25	8
Random walk	3	106.85	2
	7	107.27	5
	13	219.49	10
	15	226.57	12

Table 3.6: Distance travelled and the number of targets discovered in workspace 2 for different number of total targets

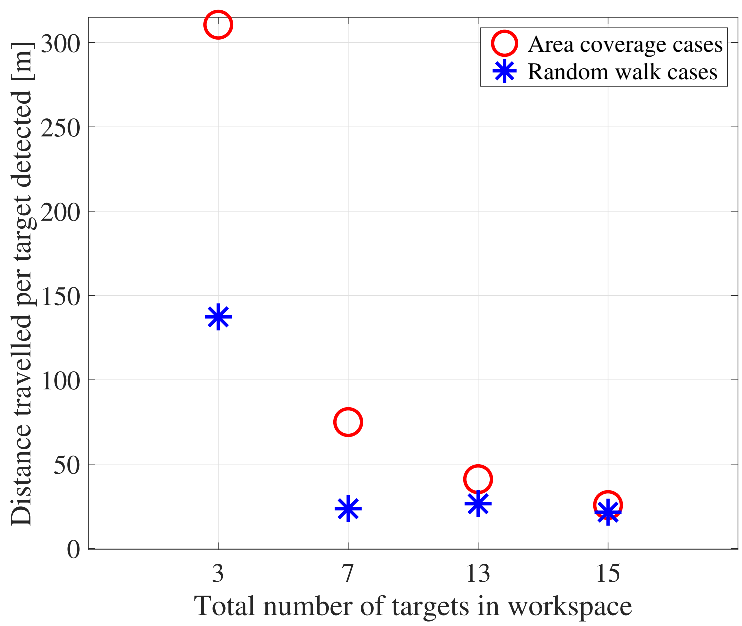

Figure 3.16: Distance travelled per target discovered for Workspace 1

Figure 3.17 plots the distance travelled per target discovered in workspace 2 for random walk and area coverage algorithms.

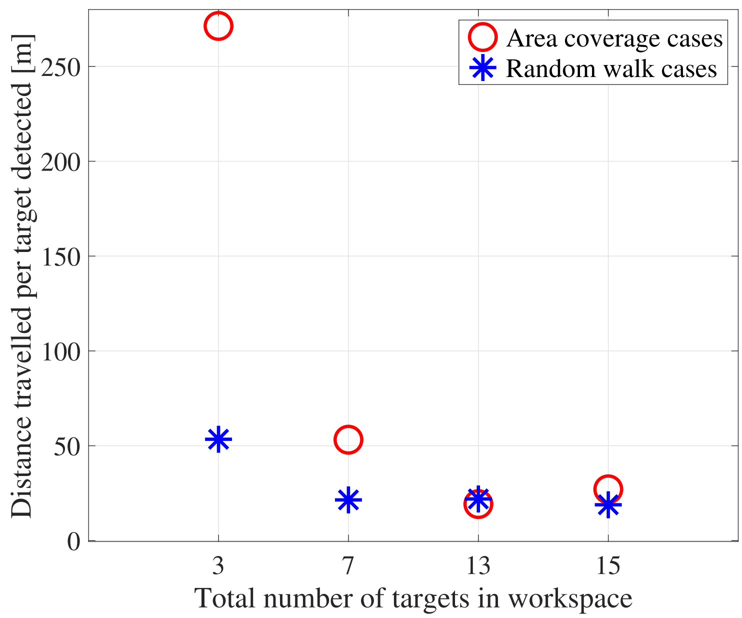

Figure 3.17: Distance travelled per target discovered for Workspace 2

From Figures 3.16 and 3.17 we can see that the distance travelled per target detected tends to decrease as the number of targets in the workspace increase. This is because the total distance travelled by the robot in either of the planners remains largely consistent across the different simulations because this depends a lot maximum speed of the robot, which is the same for both simulations.

Sequential classification results

In these simulations, every time an object is detected, we first try to classify its shape, and based on its result, we load the color classifier, and based on whose result, the texture classifier is loaded. In doing so, we simulate the classifications in a manner that we would do in the real world where the ground truth of the object is unknown. Thus if the first level of classification is incorrect, all the subsequent levels of classification will also be incorrect. The results of these sequential classifications can be seen in Figures 3.18 and 3.19.

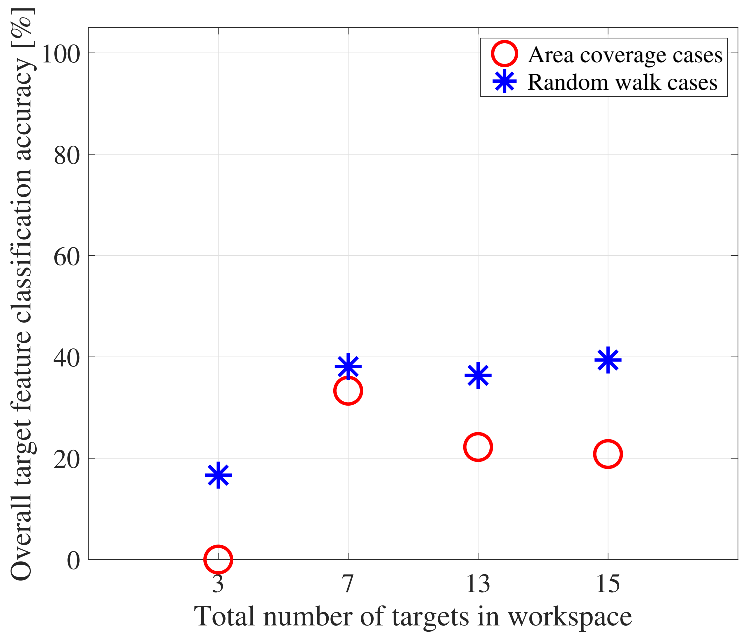

Figure 3.18: Overall classification accuracy for all target features detected in workspace 1

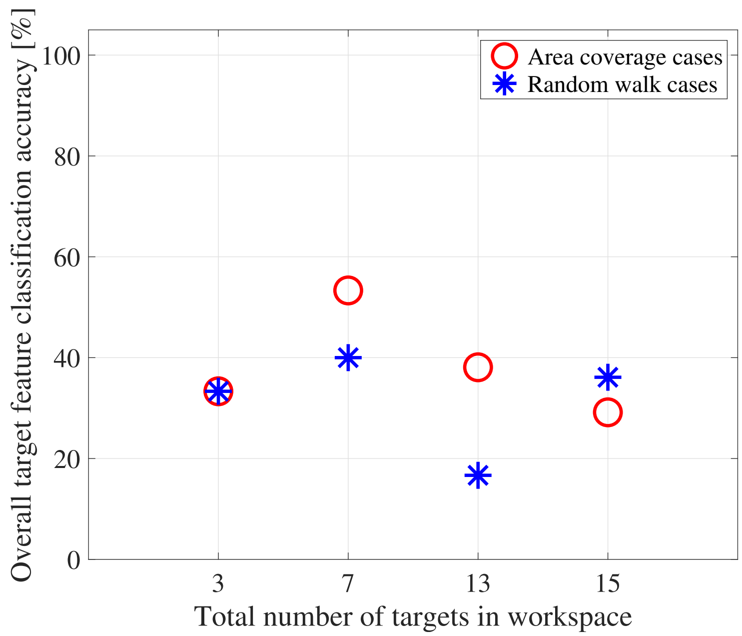

Figure 3.19: Overall classification accuracy for all target features detected in workspace 2

All the above mentioned accuracies are percentages of the detected objects classified. Furthermore, from the simulations it has been noted that the shape classifier is biased to predict a box, the box color classifier is biased to predict brown, and the texture classifier is biased to predict the surface texture of a cardboard box. This leads to a very large number of final predictions being cardboard boxes.

Treasure classification results

In order for an object to be correctly classified as a treasure or not, we check if all the target features (shape, color, and texture) have been correctly classified. Only then do we attempt to classify the detected object as a treasure or not. If any of the previous feature classifications is incorrect, it is assumed that the final treasure classification is also incorrect. Further, the results of treasure classification depend predominantly on the conditional probability table which gives us the probabilities of a detection being a treasure given its texture. The said conditional probability table has been shown in 3.7.

Target Name	**Treasure probability**
Apple	0.93
Watermelon	0.59
Orange	0.88
Basketball	0.05
Carboard box	0.69
Wooden box	0.39
Computer	0.14
Book	0.94

Table 3.7: Probabilities of treasure given texture of objects

Figure 3.20 plots the treasure classification accuracy for our simulations in workspace 1 for the different algorithms and the different number of targets and Figure 3.21 does the same for workspace 2. Please note that the different algorithms have been mentioned to indicate the source of the data and the treasure classification has no bearing on it.

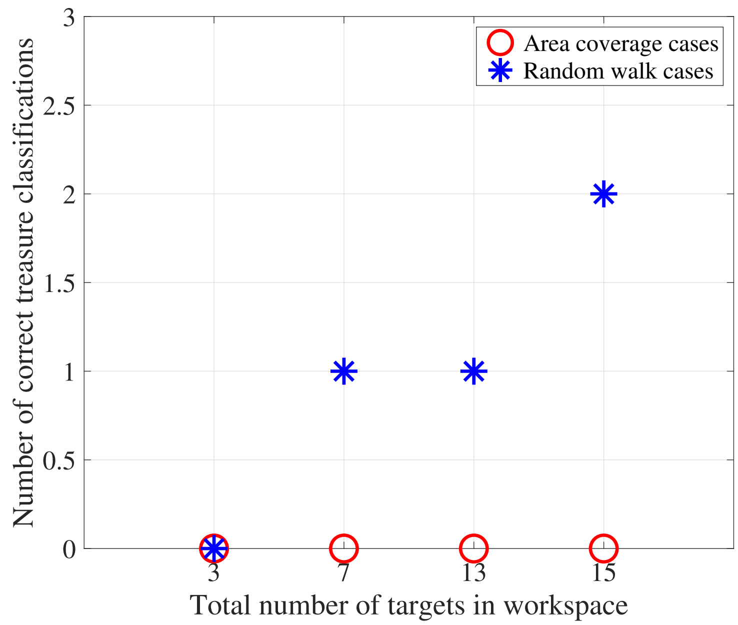

Figure 3.20: Treasure classification accuracy for workspace 1

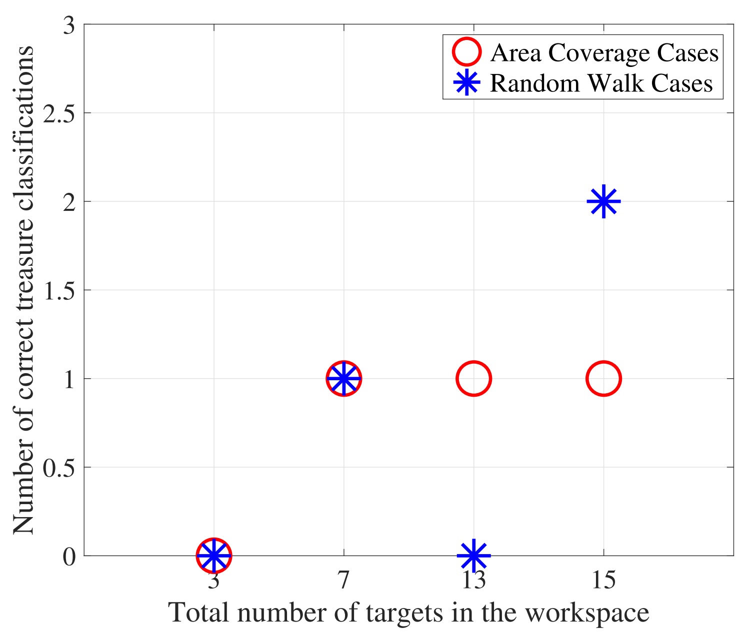

Figure 3.21: Treasure classification accuracy for workspace 2

From Figures 3.20 and 3.21, we can infer that the number of targets correctly classified as treasure or not is quite low. The nature of these results could be attributed to the propagation of incorrectly classified results from the shape and color classifiers, in addition to not being able to discover all the targets.

CHAPTER 4

CONCLUSION AND DISCUSSION

In this thesis, two sensor path planning methods have been proposed for the treasure-hunt problem, which overcome the shortcomings of many of the commonly used path planning algorithms. The first algorithm proposed was that of a random-walk where the robot travels straight until it encounters and obstacle and then turns around in a random direction and continues to do so until it either detects all the obstacles in the workspace or runs out of simulation time. The second method is that of area coverage where the robot travels in a sweeping/zig-zag motion until it detects a corner, in which case, it turns away from it and continues the sweeping motion to explore the workspace until it either discovers all the targets in the workspace or runs out of simulation time.

Three workspaces have been designed in the Webots® simulator to simulate a home, a warehouse, and a maze respectively. For each of these workspaces, four different target layouts have been proposed. In addition to this, any combination of the aforementioned workspaces and target layouts can be recreated at five different levels of fog which correspond to the following - no fog, haze, light fog, medium fog, and dense fog. The path planners proposed in this thesis have been simulated in the workspaces resembling a home and a warehouse and it has been noted that the random-walk planner tends to visit more targets in the given simulation time. However, without any time constraints, the area coverage algorithm would have eventually visited all the waypoints.

The performance of all of the feature classifiers as a function of fog density and target distance has been analyzed. The optimal target observation distance under no fog conditions was estimated to be about 80cm whereas under the presence of fog, it was found to be about 60cm. As the target distance increases further, the accuracy of the classifiers tends to reduce. Similarly, light fog conditions improved the performance of the classifier for nearby objects. However, the overall trend observed was that as the density of the fog increases, the performance of the feature classifier reduces. In doing so, we have also been able to create a dataset of images capable of being used for training future classifiers and object detectors for the treasure-hunt problem. Additionally, the information gain at each level of features has also been analyzed to enable future studies on object classification, its cost and trade-offs.

CHAPTER 5

FUTURE WORK

With the building blocks that have been presented in this thesis, the next steps would be to use them and to create an adaptive sensor path planner such that it switches between area coverage and random-walk strategies depending on the environment and the revealed cues. Further heuristics path planners can be developed using these building blocks such that they return satisificing policies under real-world constraints for the treasure-hunt problem. These new strategies can also be tested for robustness under other environmental pressures like rain, and other pressures like time and money. In addition to this, so far, all the algorithms used in solving the treasure hunt problem have assumed the target location to be known in some capacity. However, in order to relax this assumption, an object detector capable of recognizing the targets of interest can be trained on the dataset created. The said dataset can also be used to retrain the classifiers to improve the feature classification results.

BIBLIOGRAPHY

[1] Ming Xie. Fundamentals of robotics - linking perception to action. In *Series in Machine Perception and Artificial Intelligence*, 2003.

[2] Masahiro Ono, Thoams J Fuchs, Amanda Steffy, Mark Maimone, and Jeng Yen. Risk-aware planetary rover operation: Autonomous terrain classification and path planning. In *2015 IEEE aerospace conference*, pages 1–10. IEEE, 2015.

[3] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. Practical search techniques in path planning for autonomous driving. *Ann Arbor*, 1001(48105):18–80, 2008.

[4] Kazi Mahmud Hasan, Khondker Jahid Reza, et al. Path planning algorithm development for autonomous vacuum cleaner robots. In *2014 International Conference on Informatics, Electronics & Vision (ICIEV)*, pages 1–6. IEEE, 2014.

[5] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[6] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. Path planning for autonomous vehicles in unknown semi-structured environments. *The international journal of robotics research*, 29(5):485–501, 2010.

[7] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, May 2005.

[8] Purushothaman Raja and Sivagurunathan Pugazhenthi. Optimal path planning of mobile robots: A review. *International journal of physical sciences*, 7(9):1314–1320, 2012.

[9] James J Kuffner. Goal-directed navigation for animated characters using real-time path planning and control. In *International Workshop on Capture Techniques for Virtual Environments*, pages 171–186. Springer, 1998.

[10] Julien Pettre, Jean-Paul Laumond, and Daniel Thalmann. A navigation graph for real-time crowd animation on multilayered and uneven terrain. In *First International Workshop on Crowd Simulation*, volume 43, page 194. New York: Pergamon Press, 2005.

[11] Juan Cortés, Thierry Siméon, Vicente Ruiz de Angulo, David Guieysse, Magali Remaud-Siméon, and Vinh Tran. A path planning approach for computing large-amplitude motions of flexible molecules. *Bioinformatics*, 21(suppl_1):i116–i125, 2005.

[12] Nancy M Amato, Ken A Dill, and Guang Song. Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures. *Journal of computational biology*, 10(3-4):239–255, 2003.

[13] Hector Gonzalez, Jiawei Han, Xiaolei Li, Margaret Myslinska, and John Paul Sondag. Adaptive fastest path computation on a road network: a traffic mining approach. In *33rd International Conference on Very Large Data Bases, VLDB 2007*, pages 794–805. Association for Computing Machinery, Inc, 2007.

[14] Ing-Chau Chang, Hung-Ta Tai, Feng-Han Yeh, Dung-Lin Hsieh, and Siao-Hui Chang. A vanet-based a* route planning algorithm for travelling time- and energy-efficient gps navigation app. *International Journal of Distributed Sensor Networks*, 9(7):794521, 2013.

[15] F Benjamin Zhan and Charles E Noon. Shortest path algorithms: an evaluation using real road networks. *Transportation science*, 32(1):65–73, 1998.

[16] D Shah. Path planning for mobile robots using potential field method. *Master's Project, The University of Texas, Arlington, TX, USA*, 2018.

[17] Jerome Barraquand, Bruno Langlois, and J-C Latombe. Numerical potential field techniques for robot path planning. *IEEE transactions on systems, man, and cybernetics*, 22(2):224–241, 1992.

[18] Herbert Alexander Simon. *Models of bounded rationality: Empirically grounded economic reason*, volume 3. MIT press, 1997.

[19] Herbert A Simon. A behavioral model of rational choice. *The quarterly journal of economics*, 69(1):99–118, 1955.

[20] Herbert A Simon. Rational choice and the structure of the environment. *Psychological review*, 63(2):129, 1956.

[21] Herbert A Simon. Invariants of human behavior. *Annual review of psychology*, 41(1):1–20, 1990.

[22] Hannah Oh Tobias Egner Marc A. Sommer Pingping Zhu, Silvia Ferrari. Satisficing classification based on adaptive evidence accumulation strategy. *IEEE Transactions on Neural Networks and Learning Systems, Under review*.

[23] Gerd Gigerenzer and Wolfgang Gaissmaier. Heuristic decision making. *Annual review of psychology*, 62:451–482, 2011.

[24] Anja Dieckmann and Jörg Rieskamp. The influence of information redundancy on probabilistic inferences. *Memory & Cognition*, 35(7):1801–1813, 2007.

[25] Peter M Todd and Gerd Gigerenzer. Précis of simple heuristics that make us smart. *Behavioral and brain sciences*, 23(5):727–741, 2000.

[26] Gerd Gigerenzer and Daniel G Goldstein. Reasoning the fast and frugal way: models of bounded rationality. *Psychological review*, 103(4):650, 1996.

[27] TL Brink. Jw payne, jr bettman, ej johnson: The adaptive decision maker. cambridge university press, cambridge 521-41505-5 (hardback), 0-521-42526-3 (paperback), 1994.

[28] Silvia Ferrari and Chenghui Cai. Information-driven search strategies in the board game of CLUE ®. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(3):607–625, 2009.

[29] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[30] Jean-Claude Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, USA, 1991.

[31] Wenjie Lu, Guoxian Zhang, and Silvia Ferrari. An information potential approach to integrated sensor path planning and control. *IEEE Transactions on Robotics*, 30(4):919–934, 2014.

[32] Guoxian Zhang, Silvia Ferrari, and M Qian. An information roadmap method for robotic sensor path planning. *Journal of Intelligent and Robotic Systems*, 56(1):69–98, 2009.

[33] Yucheng Chen Zeyu Liu Min Zheng David Wijk Silvia Ferrari Jake Gemerek, Bo Fu. Directional sensor planning for occlusion avoidance. *IEEE Transactions on Robotics, Under review*.

[34] Chenghui Cai and Silvia Ferrari. Information-driven sensor path planning by approximate cell decomposition. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(3):672–689, 2009.

[35] Jean-Claude Latombe. *Robot motion planning*. Kluwer, 2010.

[36] Hanna Oh, Jeffrey M Beck, Pingping Zhu, Marc A Sommer, Silvia Ferrari, and Tobias Egner. Satisficing in split-second decision making is characterized by strategic cue discounting. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 42(12):1937, 2016.

[37] Hanna Oh-Descher, Jeffrey M Beck, Silvia Ferrari, Marc A Sommer, and Tobias Egner. Probabilistic inference under time pressure leads to a cortical-to-subcortical shift in decision evidence integration. *NeuroImage*, 162:138–150, 2017.