CARNEGIE MELLON UNIVERSITY

SENIOR UNDERGRADUATE THESIS

A mixed initiative approach to survivable path planning using imprecise information

---

Author:
Rohith Krishnan PILLAI

Supervisor:
Dr. Gianni DI CARO

A thesis submitted in fulfillment of the requirements for the degree of Bachelors of Science

in the

Qatar Computer Science Department

Declaration of Authorship

I, Rohith Krishnan PILLAI, declare that this thesis titled, "A mixed initiative approach to survivable path planning using imprecise information" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

"I ultimately got into robotics because for me, it was the best way to study intelligence."

Sebastian Thrun

CARNEGIE MELLON UNIVERSITY

Abstract

Dr. Gianni DI CARO
Qatar Computer Science Department

Bachelors of Science

A mixed initiative approach to survivable path planning using imprecise information

by Rohith Krishnan PILLAI

Safe robot navigation in human-robot teams during post-disaster scenarios such as after an earthquake, is a challenge for robots due to the cluttered, potentially unknown and hazardous environment. However, the human teammate's advanced cognitive abilities can be exploited to aid in safe robot navigation in the difficult to navigate environment. This research aims to create a mixed initiative path planning system that receives the human's advisory information that models potentially undetected hazards in the environment and integrates it to the robots world view to plan paths that are the most survivable - a path with low danger and high probability of existence. Current research mostly focuses on solely autonomous planning using bayesian methods for map creation and updation, and uses precise spatial information through gestures and speech to provide human input to robots. However, we consider human input that is inherently high-level and imprecise due to the use of ambiguous language. Additionally we also use imprecise probabilities for map creation and updation instead of bayesian probabilities due to difficulties in having a generalizable model of the "human sensor" that can refine our estimates over time. We discuss a method that uses a novel map representation based on imprecise probabilities, integrates imprecise human input modeling potentially undetected hazard to robot's world map, and plans survivable and efficient paths using a modified rapidly exploring random tree (RRT) algorithm.

Acknowledgements

I would like to take this opportunity to thank Dr. Gianni Di Caro for always being there to help me out when I ran into problems and for being the perfect person to go to bounce off ideas with. I would also like to thank him for the long hours that we spent together on this topic and how to tackle it and for his continued support without which this would not be possible.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

The ability to perform a safe and effective navigation is at the core of many mobile robotics tasks. It is necessary that the robot is able to move from one location in the world to another to fulfill their mission. Although navigation is commonly intuitive for humans, navigation in robots require some effort to compute and a number of different approaches have been proposed over the years including potential field methods to velocity-obstacle methods etc. Hence safe robot navigation becomes challenging especially in cases when the environment becomes extremely cluttered, hazardous, potentially unknown etc. Such conditions are indeed very common in post-disaster scenarios such as when navigating in a building collapsed or partially collapsed after an earthquake, a flood, or a fire. In these cases the debris can range from small to very large sizes and the number of hazards for the robot are also very high.

In this research we consider situations where humans and robots are working together in human-robot mixed teams. Human robot teams are set to become more common in the future and especially in post-disaster scenarios such as search and rescue following a earthquakes and fires. The potentials of human-robot teams greatly help increase team effectiveness in tasks such as search and rescue as both the robots and humans can collaborate to increase efficiency in terms of coverage, better on site control of robot systems, and making robots carry out mission tasks that are simply too dangerous for human teammates etc. In such cases it is to be noted that although the humans and robots in the team might have different missions, the availability of the human in proximity to the robot means that robot behavior could be modulated or even have the human teammate be an advisor that can potentially help in case of challenging situations. This is especially the case with regards to hazards that the robots sensors may fail to detect. We use this premise to extend it to the idea of humans helping robots to better understand their surrounding for improving both map building and safe navigation and planning.

Typical post-disaster scenarios have a world space that tend be partially unknown to the robot, highly cluttered and contain hazardous features which makes navigating such an environment difficult for local navigation approaches mentioned before as the robot sensors cannot always guarantee safe navigation. For instance, the presence of a puddle of water on the ground for ground robots, and really thin electrified wires for aerial robots might go undetected leading the robot to make navigation decisions that might not be seen as harmful to the robot in the robot's world model but however it is in the real world.

However, human team mates with their superior cognitive abilities are able to observe and create models of possible obstacles and hazards in their surrounds pretty easily, and communicating such information to the robots should allow the

robots to plan paths during navigation that are much safer and avoid these obstacles and hazards that were pointed out to them. For example, when a robot fails to detect a hole in the floor in it's surroundings and plans a path that runs through it. In this case the human teammate can notice the robot moving towards the apparent hazard and is able to point this out to the robot. Then the robot could in turn update it's world model to one that prevents running into the hole in the floor. In this case the human actively took the initiative to provide the additional advisory information about the hazard , hole in the floor, to the robot. On the other hand the robot can also possibly take the initiative and ask the human for information when the robot is unable to find any possible path that is survivable to it's target. This is an example of mixed initiative system which according to Jiang and Arkin, 2015 is defined more generally as "a collaboration strategy for human-robot teams where humans and robots opportunistically seize (relinquish) initiative from (to) each other as a mission is being executed".

More generally in these scenarios, the human-robot team has to navigate through the environment in a manner that can ensure the safety of the team members, both human and robot, and also through paths that actually exists and are traversable for the robot. In particular we want the robot agent to be able to navigate in a manner that is both survivable and efficient, defined as follows.

- Survivable: A path exists between the two points in space that can be safely navigated by the robot system, without harming the robot. The survivability metric for a path involves the max risk/probability of occupancy of all the poses given by the path.
- Efficient: The amount of time taken to navigate the path and the smoothness of the path with respect to an optimal baseline.

1.2 Challenges in robot navigation in human robot teams

However there are a few challenges associated this type of a mixed initiative system for safe navigation. The first challenge is that the robot itself might not be able to detect the hazards or obstacle well enough to be able to safely avoid it during navigation, not even over time. This could mean that without any external input any path computed through this region has a high chance of being non-survivable. Hence any hazard or obstacle that is not modeled could possibly harm the robot, and therefore timely providing the advisory information is very important.

Another challenge in mixed initiative based path planning involves the sporadic nature of the advisory information. This is due to the fact that it is not possible nor reasonable for the human to always keep an eye on the robot and regularly provide advisory information as the human would most likely be busy with their mission. Hence the input only arrives in some extreme situations or when the human notices the robot is acting in a manner that is indicative of not being aware of a hazard in its surroundings. Therefore we need to be able to need to be to find survivable paths with the irregular frequency of advisory information. Since there might be no way to prevent running into a hazard if neither the human nor the robot is aware of it, in this research we assume that the human is able to provide the advisory information at some point prior to any possible run-ins with a hazard/obstacle.

The most difficult challenge is the fact that the language that is involved in the communication of the advisory information is inherently imprecise and ambiguous. Here the ambiguity of the advisory information is the result of the inherent nature of

natural conversational language that is commonly not extremely precise in describing spatial relations, in the sense that no precise geometric information is usually provided regarding the spatial context. For example, consider the example of the robot being given human advisory information regarding the hole in the floor. In this case the human input is likely going to be along the lines of "hey watch out for the hole in the ground pretty close, in front and to the left of you". This is in stark contrast to an input that says "hey there is a hole in the ground exactly a square in shape, at 35.5 degrees and 2.34 meters away from you". Comparing the former and the latter it is possible to see that the former is most likely the format of sharing information about an obstacle or hazard to the robot as opposed to the extremely detailed input of the latter (that could, for instance, be provided to the robot by an external artificial sensor system). It is important to see that the former is however very ambiguous about providing much information about the characteristics of a hazard that we to model such as the size of the hazard, it's shape (which could be irregular and hence extremely difficult to describe such as the shape of a puddle of water on the floor), the distance from the robot, and it's orientation with respect to the robot. Hence the system needs to create a quantitative probability model, by converting from the imprecise information provided to some sensible ranges in distance, shape, size etc. in order to effectively path plan and perform safe navigation. However, there is no way to create a prior model and revise it over time using Bayesian methods to reduce our uncertainty as there is no way to model a generalized "human sensor". Hence we cannot use Bayesian methods and have to resort to using imprecise probability to model the hazard. Imprecise probabilities are in the simplest sense a range of the probability for an event occurring, flattened with just an upper/max and lower/min probability in the range representing the probability of the event itself. The use of imprecise probabilities and its detailed justification can be found on chapter 2. Hence the imprecise information has to be modeled into an imprecise probability and it is not obvious how this should be carried out.

Although the idea of having a human teammate provide advisory information to improve safe navigation of the robot might be practical, it certainly has a few challenges that need to be addressed. The hardest of which is the not so obvious method of quantifying and building a model of the hazard from very vague and imprecise information using imprecise probabilities. The other challenges such as the sporadic frequency of the human input and the fact that the robot itself might not be able to model the hazard means that we need a reliable method for integrating the model and planning a survivable path on it.

1.3 Related Work

This work is related to two different topics, which are human robot interaction and survivable path planning. Human robot interaction is a key point in our research as it is needed to be able to convert the human advisory information into a model that can be used in path planning. There is a lot work currently in human robot interaction that tries to identify methods to make feedback from robots more apparent, find new methods to communicate more intuitively with the robot, and even about making the robots intentions known and transparent to the humans. Goodrich and Schultz, 2007 captures some of the considerable amount of work in this area. If we consider the relevant prior work in human robot interaction about the methods to model spatial information, then most of these look at methods that provide precise spatial information to the robot, such as Skubic et al., 2004 who use a multi-modal

system with gestures and speech to communicate the location or direction of movement or obstacles to the robot. The paper also studies more about extracting the best fitting spatial descriptions of a world or obstacle from its grid map representation, and discusses ways that it could be used in human and robot dialog. Both Skubic et al., 2004 and Lei et al., 2014 use speech based systems and or multi-modal systems for providing the robot with commands. We only consider using speech based human input in our system as it is most likely that the human teammate has their hands busy during scenario such as search and rescue. This input is also however not commands to the robot but rather only additional advisory information to model potentially undetected hazards by the robot. However most importantly our research differs in the fact that we deal with imprecise rather than precise spatial information available from the human. This due to the fact that we only receive imprecise spatial information through the speech from the human and there are no other ways to clarify or to reinforce the information provided. This is also why we have to resort to working with imprecise probabilities rather than the Bayesian probabilities which are updated over time.

The second subfield that our research is related to is of survivable path planning. Some recent research in the field uses human movements in the surroundings to predict the location of certain objects such as doors for navigation as shown by Oh et al., 2015. Similarly other research such as Cho, Park, and Lee, 2015 and Kim and Lee, 2013 look at the aspect of survivable path planning in terms of scenarios in the battle field where the robot has to stay safe and avoid line of fire or being taken out by the enemy. Again this is in stark contrast to search and rescue situations where there are not too many wide open areas to navigate through but rather cluttered spaces with many possible hazards that unlike the prior might not be detected by the sensors. Other works mostly focus on multi-robot formation survivability across adversarial environments Shapira and Agmon, 2015, by picking the multi-robot formation that is most likely to withstand disturbances while navigating through a certain map. While this may be of use in search and rescue, it still does not address the issue of planning in partially unknown and cluttered environments. Also to be noted is that we consider the case on single robot navigation and hence don't enjoy the same redundancy and distribution of a multi-robot system. However, all these different works use survivability in a different scenario to our post-disaster scenario and also all employ Bayesian methods in order construct and path plan on the map. In our research we avoid using Bayesian methods and instead opt for using imprecise probabilities for building a map representation of our hazards and world map for the robot.

There is also some theoretical work on planning for survivability or traversability especially in terms of graph based problems. Some of these include the Canadian traveler problem that tries to find a path from one node to the target but the probability that edge exist or not is only discovered on arrival. Problems such as the stochastic shortest path and robust path optimization are two path based problems that are also related to the idea of finding a path between node given that the edges costs are probabilistic.

1.4 Research problem statement

In light of the challenges mentioned and the related works in this field, the research question that we tackle in this thesis can be summarized as:

How can robots use externally provided, sporadic imprecise information to perform safe navigation by adaptively building maps and planning paths that are survivable and efficient?

The main research question can be further broken down into three sub questions that we need to answer:

1. How to interpret human "imprecise sensor" by using imprecise probabilities?
2. How to integrate imprecise probabilities map with the robot sensor map to create a new imprecise probability map?
3. How to plan on this map to find the most survivable paths?

1.5 Research contributions

By addressing the three challenges above, we can single out threes main research contributions:

1. representing human input as imprecise sensors and the use of imprecise probabilities to model it;
2. building maps using imprecise probabilities;
3. planning for the most survivable path using a modified RRT based planner (2.1);
4. building an entire system chain that integrates the above three aspects

The next chapters elaborate on the background and methodology of each of these three contributions and the implementation of it using tools such as ROS and Gazebo.

Chapter 2

Background

2.1 Path Planning Requisites

Path planning refers to computing a sequence of robot poses in its configuration space so as to identify a feasible motion for the robot for going from an initial configuration to a target configuration. For instance, finding a way to move from the arm flexed configuration to the arm relaxed configuration for a robotic arm. In our case, considering mobile robots such as ground or flying robots, path planning refers to finding a sequence of poses in the world through which the robot can move to go from a starting location, most usually the current robot location to a target location, usually where the mission task has to be committed.

However, in order to plan paths between positions in space it is imperative that we have both a representation of the world around the robot and also knowledge of where the robot is in space. In absence of prior knowledge, this can be done by the robot itself by a procedure known as simultaneous localization and mapping (SLAM), which aims to continuously localize or figure out the robots pose with respect to its surroundings or the world frame and then map to fill in the structure and arrangement of the objects in the world around it. SLAM is a very well studied procedure and uses methods such as extended Kalman filters and Bayesian filters(e.g. FastSlam Montemerlo et al., 2002), and are used extensively by mobile robots especially once placed in potentially unknown environments. Although there are many map representations that can be used, the most common and basic representation is of the occupancy grid Elfes, 1990. The occupancy grid is a discretization of the world into multiple cells which are of a fixed size, similar to pixels in image. The map is comprised of an 2D array of these cells with each cell containing a value that represents the probability of the cell's counterpart in the real world being occupied, due to some part of an obstacle falling within this cell. The idea of the occupancy grid can also be extended to 3D where each cell is called a voxel and similarly contains the probability of the cell being occupied. The occupancy gird can be represented more efficiently by quadtrees or octrees, which recursively divides the space into 4 and 8 respectively. This creates a tree structure whose depth increases the resolution of the pixel or the voxel. In this representation hence large empty occupancy gird spaces can be represented by the value of a single node with the least depth in the tree over the area. It also allows easy computation in viewing the same map with various resolutions simply by changing the tree depth. In our research we use the Octomaps ROS package Hornung et al., 2013 to conduct SLAM and create a 2D occupancy grid of our sensor based map. Path planning can be accomplished using many different techniques such as potential field based methods, graph based methods, road-map based methods, and sampling based methods etc. In this research we use a modification on a sampling based method called Rapidly Exploring Random Trees (RRT) LaValle, 1998. We use this method as RRTs work very well in practice and they are

very fast at finding a path, although it does have the problems that come with being a sampling based approach of not being able to guarantee that a path will be found, although they have probabilistic asymptotic guarantees. The RRT works by building a tree over many iterations until the target point is added to the tree. At each iteration a random point in the configuration space is is picked and it's closest node in the tree is found. If the random point is within a certain range (δ) of the closest node the random point is added to the tree. If not then a point that is δ away but in the direction of the random point is added to the tree. In this way the tree if left to grow unbounded will fill up any free space creating a road-map. For this reason RRTs are also considered to be a road-map based approach to path planning. The figure 2.1 shows an example output from an RRT.

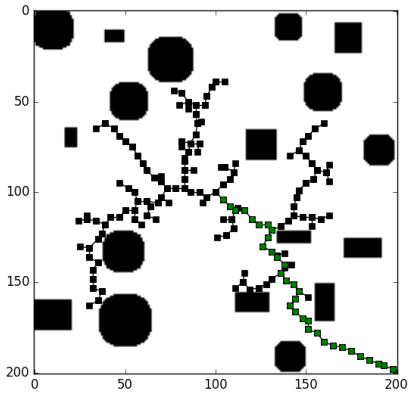

FIGURE 2.1: An example of an output from an RRT showing the tree in black and the path in green.

It is important to distinguish between global and local planning. Global planners usually plan a general path between two points where the target could be very far away. The global planner is returns a set of waypoints that breaks down the global path to smaller paths. The local planner receives the waypoints and planning paths between the two waypoints by taking into consideration the motion constraints of the particular robot as well as employing obstacle avoidance techniques during local navigation. In our research we aim to build a global planner that finds the most survivable path between the robot and its target, and so the final output of the system is a set of waypoints describing the path to be locally planned for by a local planner that accommodates for the robots motion constraints.

2.2 Imprecise Probabilities

As we have pointed out, the information that we get as input from the human teammate is inherently imprecise and to model this imprecise information using Bayesian methods presents some issues. This method usually works well for other sensors such as laser sensors in order to create an occupancy grid and build a map of the world, but extending it to model human input has many issues. To understand why this is the case we need to inspect the way Bayesian methods are used in map building and updating.

Bayesian map building and updating requires two key components:

1. a priori probabilities

2. a way to update the probabilities over time

The method to update the probabilities involves updating the occupancy probability of a cell in a map given a new piece of information or evidence. Usually the evidence providing entity is a sensor with its own known probability distribution model. Then the update of the prior probabilities is done using the Bayes formula 2.1 shown below.

$$P(D|S_m) = \frac{P(S_m|D) \cdot P(D)}{P(S_m)} \tag{2.1}$$

In this case the random variables D represents the probability that a specific cell in the occupancy grid is occupied and S_m is the observed value returned by the sensor. With the update, the probability of the cell being occupied given the new observation by the sensor is calculated as the product of the probability of the sensor returning that particular value given the cell is occupied times the probability of the cell being occupied based on some a priori model, divided by the probability that the the sensor would even return the observed value.

However, assuming that we can satisfy the first requirement of having prior probabilities for the map, this method of updating probabilities cannot really be used with humans providing imprecise information. This is because even though the information is provided by a human, there is no way to make a reliable model for the human that generalizes to different people. Even with a single person we would need to empirically build this model. This means that we do not have a known probability distribution for the observation made by the human nor for the probability distribution of the human observation given the occupancy value of the cell. This prevents us from being able to use traditional Bayesian methods for building and updating our maps, and forces us to use an alternative method that does not rely on such such known probability distributions models. An example of trying to model a general "human sensor" specifically for the quantification of proximity to a hazard please look at experiment 4.3.1.

Our choice to overcome the mentioned issues, has been to use imprecise probabilities as an alternative to Bayesian probabilities. Imprecise probabilities allows us to model the inherent imprecise nature of the human information, by not forcing us to fix a precise probability to an event, but rather a range. In the crudest sense, imprecise probabilities, can be thought of as interval probabilities with an upper estimated probability and a lower estimated probability associated with each event (Augustin et al., 2014). On one hand a Bayesian probability value can be thought of as the most likely value for the probability of an event as shown by the distribution curve in 2.2, while the imprecise probability is a uniformly distributed uncertainty between the lower and upper probability values. Imprecise probabilities also provide us a way to automatically capture the confidence in the probability of an event through the range in upper and lower probability, which is not possible with Bayesian probabilities. The closer the upper and lower probabilities are to each other the more the confidence in the probability, a fact that we can use to our advantage when working with imprecise probabilities.

Imprecise probabilities are a generalization on Bayesian probability, as it converges to Bayesian probability when the upper and lower probabilities are the same. The upper probability is considered to be an estimate of the plausibility of a certain event, where as the lower probability is considered to the be the certainty of an event as described by Miranda, 2008. Unlike with Bayesian probabilities where we have one value for the probability of occupancy per cell, we need a different representation for imprecise probabilities. Therefore we use two separate occupancy maps

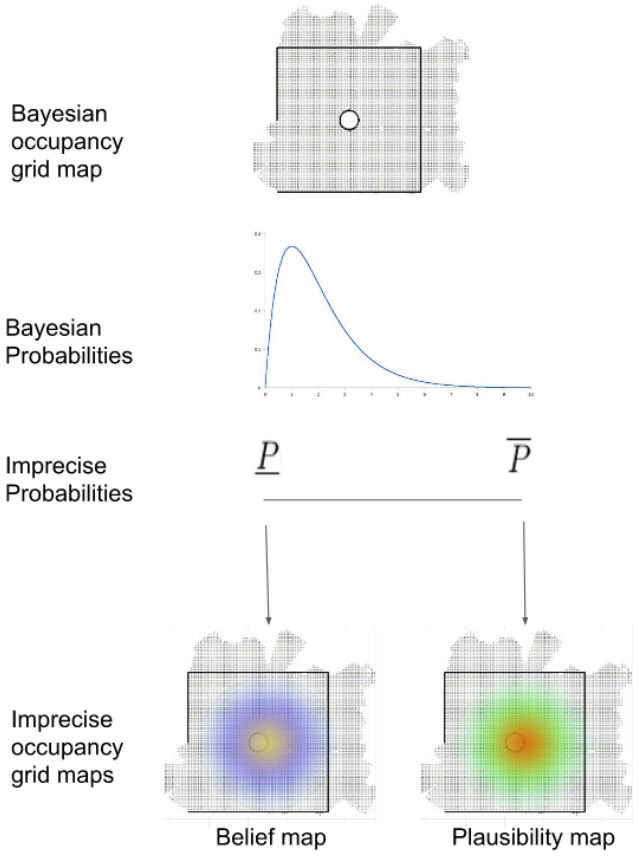

FIGURE 2.2: How imprecise probabilities and their maps (bluish, reddish) differ from Bayesian probabilities and map (gray)

when representing imprecise probability occupancy grids, the belief map with all the lower probabilities of the cell being occupied and the plausibility map with the upper probabilities of the cell being occupied.This can be seen as the two maps at the bottom of the figure 2.2 with the reddish colors representing the upper and the bluish colors the lower probability.

The imprecise probability maps are updated by using an autoregressive moving average (ARMA) based sliding window moving average. Chapter 3 contains a detailed explanation of the imprecise probability updation procedure.

2.3 Tools used: ROS and Gazebo

In this research we create a system that allows for survivable path planning using imprecise information built on the ROS platform. The Robot Operating System (ROS) is an open source middle-ware created for robotics that allows the creation of robotics applications that can work on a variety of different robots. Hence building a ROS package for survivable path planning incorporating human information helps to create an application that is very versatile and portable and can enjoy the reuse of existing tools.

ROS works based on a publish-subscribe model. A package created in ROS works by individual programs called nodes that subscribe to certain topics, work on the information it collects and possibly publish to a different topic. A package have many different nodes which communicate with each other by passing information through topics. A topic is a stream of broadcast messages of a defined type. For example, occupancy grids can be a type of message in ROS. Nodes can choose to receive the messages broadcast on a certain topic by subscribing to it, and similarly they can send messages on a certain topic by publishing messages on the topic. A node can be run by using the command `rosrun` $<$ `package name` $><$ `node name` $>$. It is possible to have multiple nodes run at the same time with certain parameters by creating and running a launch file. A launch file can be run by using the command `roslaunch` $<$ `package name` $><$ `launch file name` $>$.

In order to test the package that we are building, we use the Gazebo simulator. The simulator allows us to test our programs to and to see the actions of the program on the robot etc. The Gazebo simulator can be launched with many different worlds configurations that are defined by an XML file with the .world extension that specifies the entities and their characteristics and pose in the world. Moreover, the rviz tool allows to display data in real-time, such as visualize the data from different topics like the occupancy grid. Similarly the octovis tool allows us to visualize in 3D the octomap that we create and save using the Octomaps package.

2.3.1 System overview

The architecture of the ROS package is as shown in the figure 2.3. There are a total of five different nodes in the package that are implemented, and we use a custom launch file to set the parameters of octomap and run the octomap node from the the launch file. The gazebo provides the sensor data and the odometry and the creates the world in which we can move the robot around. The local planner is a placeholder for how any local planner can be integrated into the system by subscribing to the global planning map to create the local cost map and using the waypoints from the global planner to plan local paths between the waypoints. The detailed list and explanations of the purpose of each node/component is as given below.

- `genModels`: Creates a set of worlds that are of a specific density of coverage with respect to obstacles, and also generates the accompanying world file and initial map file which is later used by `updateIPMAP` to build and update the imprecise probability maps.
- `textInput`: Provides a text based interface to inputing spatial specifications of a human input which is then sent without processing to `hi2IPMAP`.
- `hi2IPMAP`: Translates the spatial specifications received into numerical ranges for the direction, range and also models the human input by creating the upper and lower probability maps for the human input. This is then sent to the `updateIPMAP` node to be integrated.
- `updateIPMAP`: This node is one of the most important nodes in the package as it handles the updation of the current imprecise probability maps using the projected map received from the octomap node and as well as the integration of the human imprecise probability maps to the current maps. This node is also creates and publishes the most up to date planning map for the `iispPlanner`.
- `iispPlanner`: This node needs to be given the target position to get to and the maximum risk that the navigation should accept. It then uses the planning map to find the most survivable path using our modified RRT algorithm from the current robot pose to the target position and publishes the waypoints of the global path, if one is found. If no such path exists then it asks the human whether the try again by possibly increasing the max risk threshold.

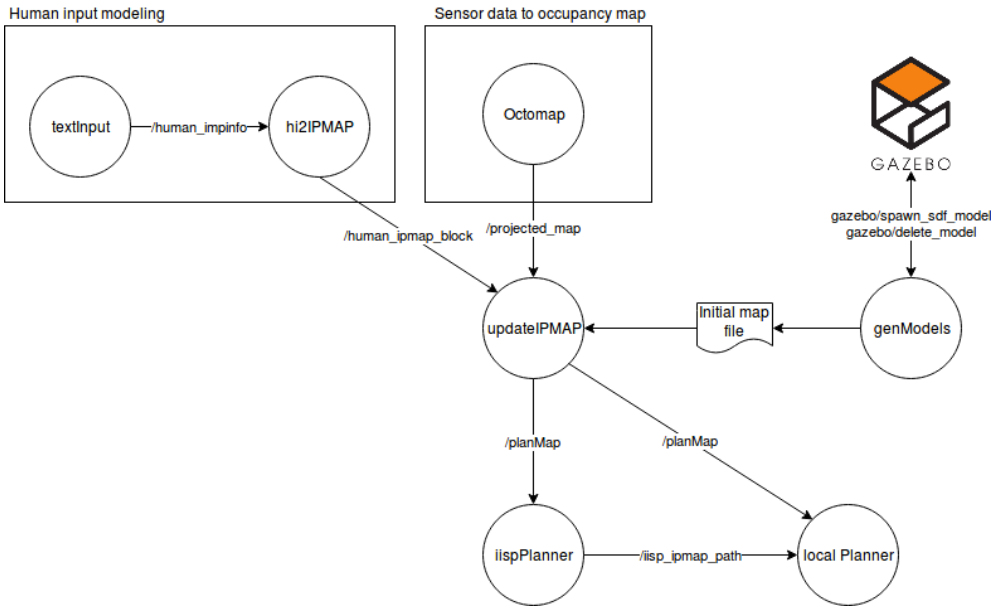

FIGURE 2.3: The architecture and the relations between the nodes in the package are shown. The circles are individual nodes in the package, and the topics are on the lines between them.

Chapter 3

Map creation and update based on robot sensors

3.1 Introduction

This chapter discusses the method used in the creation and updation of the imprecise probability maps, the belief map and the plausibility map. There are three main components in this procedure: the generation of the initial maps that the robot has, the SLAM map created by the robot using its sensors and the final imprecise probability maps that are updated.

3.2 Generating initial maps

In post disaster scenarios it is common that some maps of the area prior the disaster are available, although we cannot know how much of the initial map is still reliable as the disaster such as a earthquake can easily caused many possible landmarks and structures to have collapsed. However the initial map still provides a good starting point, although we need to update it with data that we get from our sensors. Also since we are planning global paths in the world we need some initial map to both locate the target and to plan the global path from the current robot location to it. For the research, we will be using some synthetically constructed world using gazebo to abstract some cluttered and potentially unknown situation.

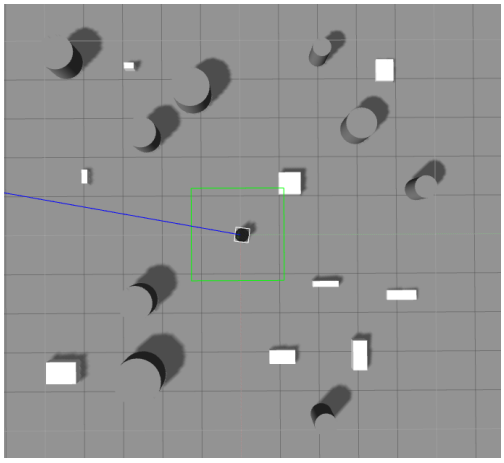

FIGURE 3.1: Gazebo simulation of an auto generated world map with density 0.1

To generate the initial maps of the world the node `genModels` first creates a world file with the robot at the origin and multiple convex obstacles modeled by either

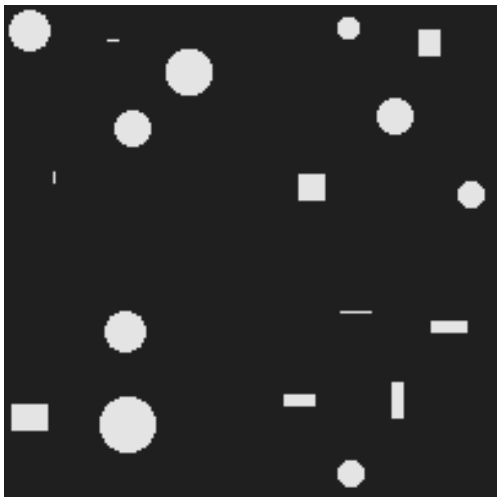

FIGURE 3.2: Auto generated occupancy grid map of world 3.1

a cylinder or a cube placed randomly in different places in the world. The node is provided a density range and a bounding box using which the worlds will be generated to fit. The density of the world is determined by the percentage of area that makes up the obstacles versus the total area of the bounding box that makes up the "mapped" section of the world. The bounding box determines the dimensions of the occupancy grid that is created of the auto generated world. The figure 3.1 shows a world that is randomly generated by `genModels` and figure 3.2 the auto-generated occupancy grid of the same world. The generated occupancy grid map's meta information is stored in a yaml file which defines the resolution of the map, it's origin with respect to world frame, and also the location of the pgm binary file containing the data of the map. This map is then later used by other nodes such as the `updateIPMAP` node that updates the current versions of the imprecise probability maps of the world.

3.3 Map creation from sensor data

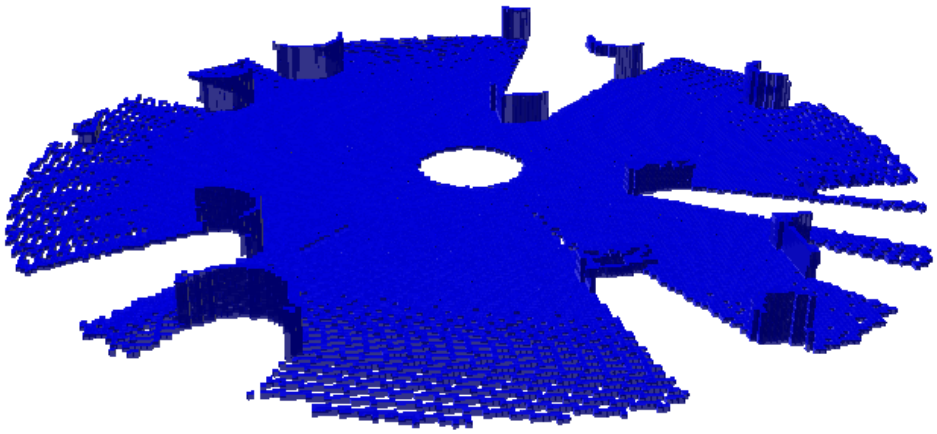

FIGURE 3.3: Octomap created using point clouds from depth camera

The sensors of the robots such as the depth camera are used to create a sensor map from the SLAM procedure. The ROS package Octomap is used to create the

projected 2D occupancy grid of the world. The world that was created in the previous section is mapped using SLAM and an octomap representation of the map is created as shown in figure 3.3. The depth camera of the robot is used to produce point clouds, which are points in space that are treated as occupied. Then these point clouds are used to create the octomap representation of the world around the robot. However during the creation of the point cloud the points above the robot's height is ignored as the robot will not run into the obstacle that might be possibly hanging from the ceiling of the building. The octomap is finally then projected on to the 2D ground plane to create the occupancy grid map of the world as shown in figure 3.4. The probabilities of occupancy are given in a range of 0 to 100 for each cell of the grid. This map is published by the octomap node at a frequency of 2.5 hz, and is subscribed by the `updateIPMAP` node that goes on to update the current imprecise probability maps.

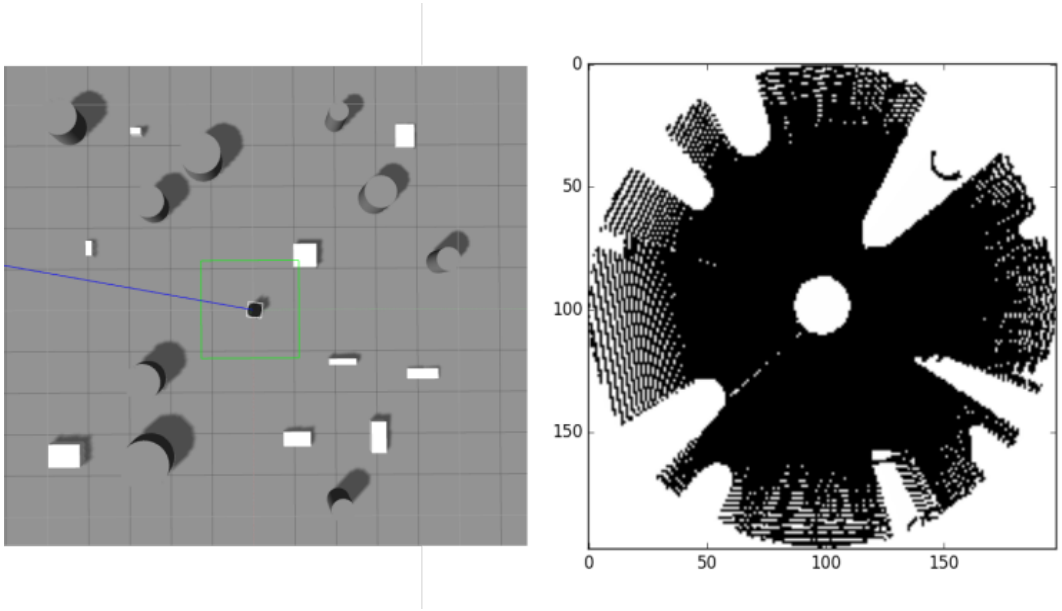

FIGURE 3.4: The world layout (left) and Occupancy grid map created by projection to ground plane(right)

3.4 Map updation from sensor map

The 2D occupancy grid map created from the sensors during SLAM is received by the `updateIPMAP` node and the node then updates the probabilities of the upper and lower probability maps by using an autoregressive moving average (ARMA) technique. This is the method that we adopt for updating the probabilities because we cannot use bayesian methods to update our prior probabilities from our initial map as we do not have a probability distribution from the sensor model directly but rather only get the probabilities from the occupancy grid created by it. To make an ARMA up date we only need to update the probabilities with a sliding window moving average as the Gaussian white noise can be assumed to be part of the sensor input. Hence to do the updates we use the formula 3.1, where $C_{ij,t}$ is the occupancy grid value of the cell at row i column j at time t, M_{ij} is the number of sensor maps that contributed to the cell C_{ij} from time $t-1$ to $t-k$, k is the number of inputs in the sliding window, I_{ij} is value of cell C_{ij} of the current map we have, and $C_{ij,t-k}$ is

the value of cell C_{ij} at time $t-k$.

$$C_{ij,t} = \frac{1}{M_{ij}+1}(I_{ij} + \sum_{k=1}^{M_{ij}} C_{ij,t-k}) \tag{3.1}$$

Using this update formula we revise the cell values for the upper and the lower imprecise probabilities. In our implementation we use a sliding widow size of 9. The result of this can be seen in the figure 3.5. The probabilities of occupancy of the map are given by the gray-scale value of the cell, and hence a darker color represents a higher probability of occupancy. If there is no human information that was passed to the node at this point then the current imprecise probability maps are converted to a planning map and then our modified RRT algorithm is used to find the most survivable path as explained in chapter 5.

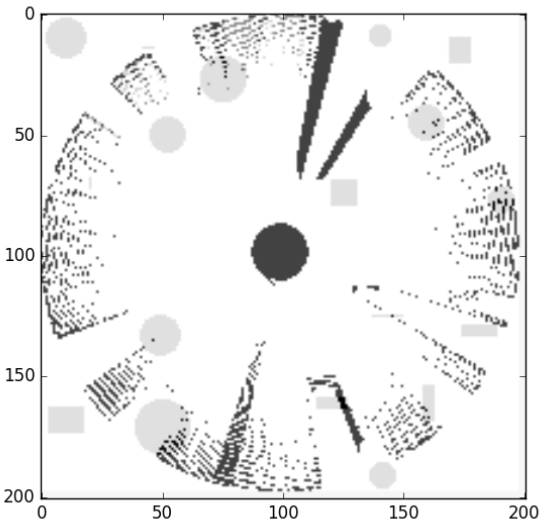

FIGURE 3.5: Occupancy grid map of upper probabilities/ plausibility map

Chapter 4

Modeling using imprecise human inputs

4.1 Introduction

While in the previous chapter, the use of robot's sensor inputs where used to create and update imprecise probabilities maps, in this chapter we discuss the modeling of the imprecise human input and its integration into the world map. The process of incorporating the human input involves up to three different nodes in the package, and is a key aspect of the survivable path planning system.

4.2 Parsing human advisory information

The human input is introduced by speech to the robot and we use an off the shelf API of the Google speech recognition system to convert all our input from natural language to the spatial specification data structure. Due to relatively low accuracy and high latency of the speech recognition system, we proceeded to provide the human input in an spatial specification parsed format. The spatial specification is a custom data structure that contains the following important information needed to capture the qualities of the imprecisely described hazard or obstacle.

- Object: This is the type of object and usually contains information regarding the size of the obstacle as well. The possible currently recognizable sizes are small and large which are modeled to be 0.25m and 1.0m respectively. Not providing any clues regarding the size of the obstacle/hazard makes the node assume medium size with the 0.5m long. These numbers were chosen a they were the ranges of sizes that obstacles in the synthetically generated world, and hence we assume that the obstacle modeled would be in a similar size range as well.
- Range: This is the distance that the obstacle/hazard is from the robot. There are two currently recognizable ranges, close and far. Please see the 4.2 for how this is interpreted into a numerical range.
- Direction: This is the direction in which the obstacle/hazard is located in the 8 region template as shown in figure 4.1 with respect to the robots current orientation.
- Message: This is any additional message that human would like to give. This field is used to allow for new vocabulary to be tested out, and or show the whole human input message for further modification of the model. Usual messages include "watch out!", "careful!" etc.

4.3 Converting spatial specifications to numerical ranges

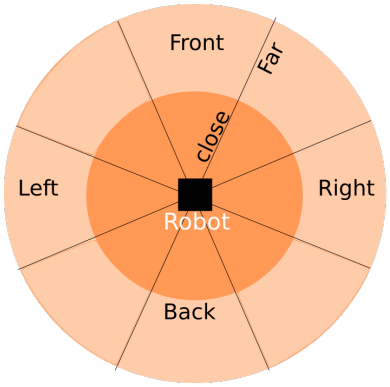

FIGURE 4.1: Template for organizing spatial regions around the robot

Even with the human input parsed into the spatial specifications it needs to be converted into numerical ranges before the model can be used to update the imprecise probability maps. The primary method of converting the spatial specification to numerical values is template matching. The figure 4.1 shows the 8 directions that are used when defining the location of the obstacle/hazard with respect to the robot, and these directions are replaced by Cartesian unit vectors by the `hi2IPMAP` node. For the sake of simplicity, we have used 8 slices, but more can be used, for increased precision. More slices, means more "priors", which precisely what we want to avoid. The size determined from the imprecise information is used to scale the unit convex model that is detailed in section 4.4. However, to convert the distance of the robot from the obstacle/hazard needed more context and cannot be easily done especially from just the imprecise human information as a being close or far. Since the notion of distance is very subjective we held an experiment to get empirical values to create a general model in order to be able to convert the imprecise subjective term of close and far to quantifiable value. Once these spatial specifications have been converted to numerical values it can then be integrated into the imprecise probabilities maps.

4.3.1 Experiments on proximity

FIGURE 4.2: The experimental setup with the view from the participants point of view

With the human input being inherently imprecise we need a method or model to quantity the proximity of the robot to different obstacles/hazards from the imprecise

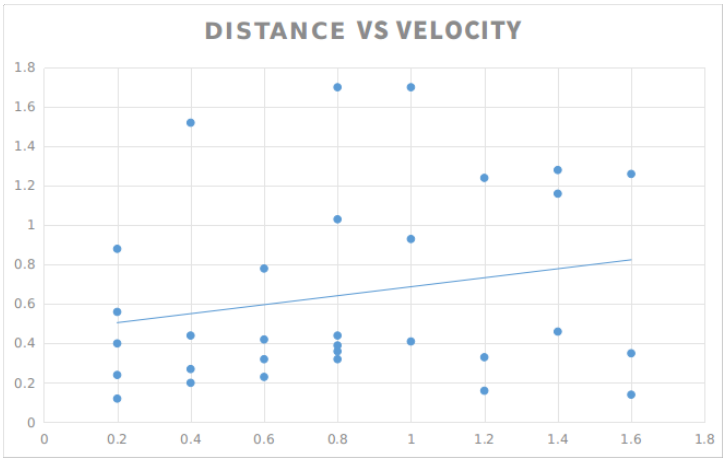

FIGURE 4.3: The graph shows the positive correlation between distance of the robot to the obstacle and the robot velocity

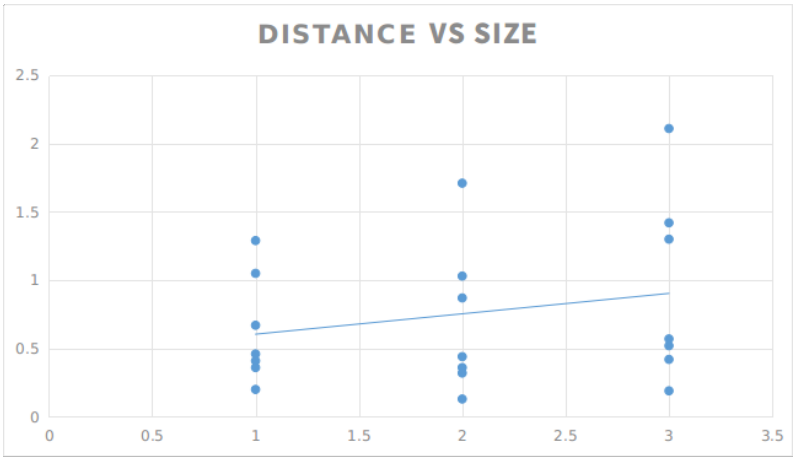

FIGURE 4.4: The graph shows the positive correlation between distance of the robot to the obstacle and the size of the obstacle

information. Hence, we aimed to collect empirical data to be able to build a model to quantify the ranges of distance from the spatial specification. However, instead of having a fixed value for each modeled distance, it may be possible that other pieces of information such as the robot's current velocity and the obstacle's size that might affect the distance of what is termed "close". Therefore, the major question that the experiment tries to answer is does the velocity of the robot or the size of the obstacle change the "close" distance across a general audience. For the sake of the experiment, each of the participants were given the definition of close to be the least distance to obstacle needed by the robot to still be on a collision-free path at a constant velocity. The experiment setup from the participants point of view is shown in figure 4.2. After reading out the definition of "close" to the participants as a prompt, the robots were made to run straight towards the obstacle at a velocity that was randomly chosen from the range of 0.2 to 1.6 m/s with the medium sized box as the obstacle. The participants were given the instruction to stop the robot as soon as they thought that the robot was close to the obstacle given by the boxes. There were 3 sizes of the boxes used as well with a constant velocity of 0.8 m/s, to see the effect of obstacle size to the "close" proximity. With a 32 data points from 8 different participants we got the following results as shown by the figures 4.3 and 4.4.

From the two resulting graphs we can see that there exist some positive correlation between the "close" distance and the velocity of the robot. As the robot increases it's velocity the "close" distance seems to increase. This can also be seen for the "close" distance and the size of the obstacle, although there does seem to be a large variance in distances for both. Using this observation we tried to create a model for the translating the "close" term with respect to the current velocity of the robot by using the line of best fit and this is used to calculate the distance from the obstacle during human input merging to the map in `updateIPMAP`. The distance versus size data was not very helpful as we were dealing with categorical values for the sizes which are not very generalizable.

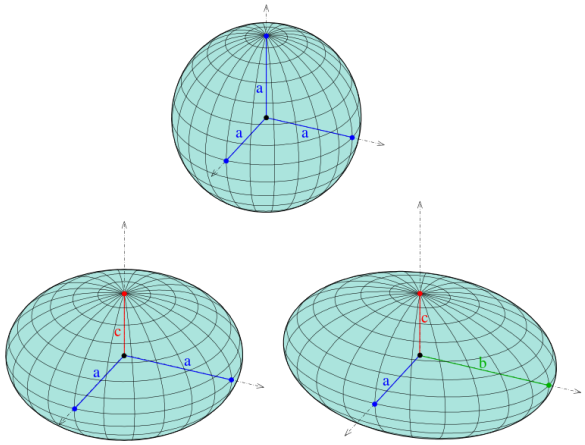

FIGURE 4.5: An ellipsoid convex embedding

4.4 Modeling the hazard using a convex shape

Now that we have some numerical and quantified parameters, we need to be able to create an imprecise probability model to be used to integrate into our current IP

maps. Although we converted the size, orientation, and distance to numerical values that we can work with, the shape of the obstacle is something that we cannot determine from the human input, especially if no such information is explicitly provided. In such a case it might be better to model the obstacle as a symmetrical shape as our uncertainty is high about its shape properties and also there does not exist any preferential direction of error. Hence we use a ellipsoid to model the probability of the obstacle from the human input. We used an ellipsoid because its parametric form allows us to easily modulate its eccentricity if we do receive such additional information, it can be compacted in a pair of parameters, and the convexity of the ellipsoid is amenable and "convenient" for path planning. Since we are modeling an uncertain obstacle, the model we make must have a confidence that reduces towards the edges of the obstacle, and to model this we use the lower hemisphere of the ellipsoid to model the upper probabilities and the upper hemisphere to model the lower probabilities. This makes sure that even though the median probability at all points in the obstacle is at 0.5, the confidence in the probabilities decreases towards the edges and increases towards the center of the model. The figure 4.6 shows the upper and lower probabilities of the obstacle modeled. Notice from the image, the graduation in color from the center for both the imprecise probability obstacles, where the darker colors indicate values close to 1.

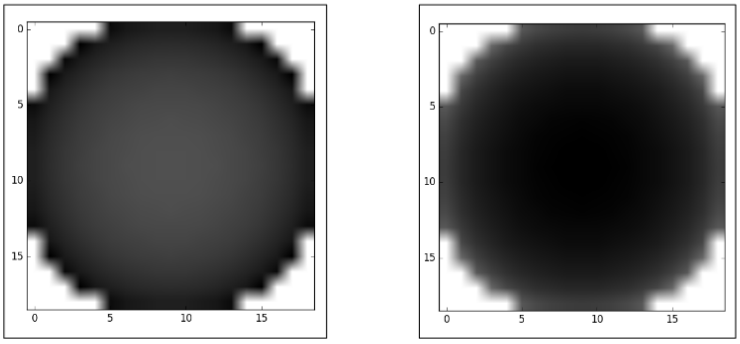

FIGURE 4.6: Template for organizing spatial regions around the robot using a convex embedding

4.5 Integrating the human input into IP map

The modeled human input is received by the `updateIPMAP` node as imprecise probability maps. This model however is then needed to update our current imprecise probability maps. This update is done through by first finding the location of the robot on the map and then finding the intended location of the human input provided obstacle on the map and then replacing the upper probability values with max value between the current cell value and the corresponding cell value from the modeled human input upper probability map. Similarly the the lower probability values are replaced with with min value between the current cell value and the corresponding cell value from the modeled human input lower probability map. This update criteria ensure consistency in the definition or imprecise probabilities and also helps integrate the human input into the imprecise probability maps as shown in figure 4.7.

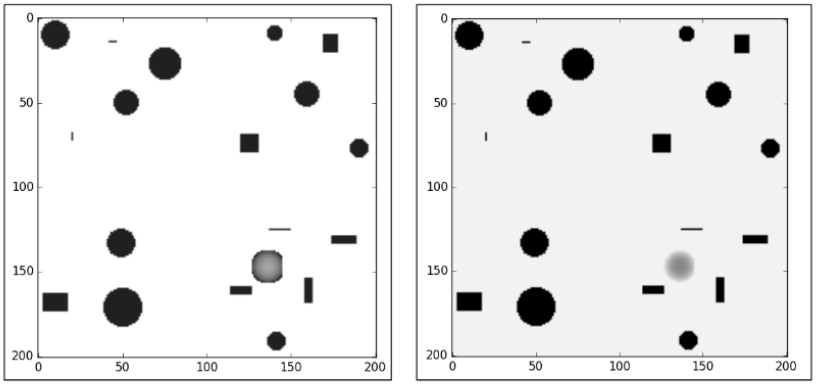

FIGURE 4.7: The imprecise probability maps with integrated human input. Upper probabilities on left, lower probabilities on right.

Chapter 5

Path planning

5.1 Introduction

The output from the updated imprecise probability maps are used to find the most survivable paths in the map for the robot to navigate the environment safely. There are two main components in this task, the first of which is to create a unified planning map from the imprecise probability map. The second component is the method of finding the path from the current cell to the target cell using a modified RRT to find the most survivable path in the planning map.

5.2 Creating the planning map from IP maps

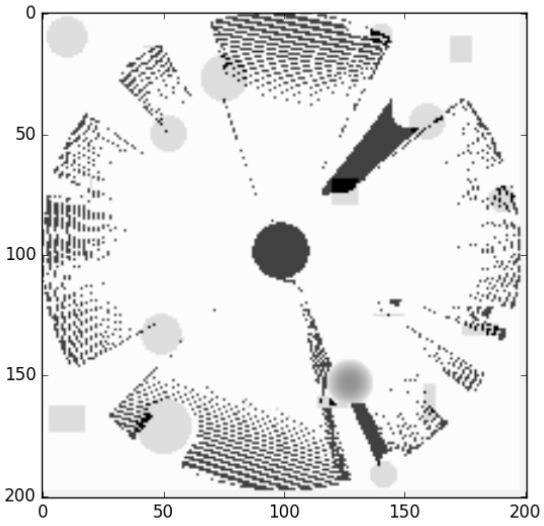

FIGURE 5.1: The planning map with integrated human input integrated in the middle of the bottom right of the map

The imprecise probability maps are good representations for updating and keeping track of the obstacles, hazards, and to integrate human input into. However in order to be able to path plan we need a unified map. This is because path planning with two different sets of values can quickly become cumbersome and would require a metric for survivability be extracted from the upper and lower probabilities of a cell. Having one map that then represents the survivability metric for each cell makes this process much faster as the pre-computation has been done, and it also

makes it possible to modify any existing path planning algorithm to make it suitable to calculate the most survivable paths easier. For these reasons we create the planning map from the two imprecise probability maps by the following formula 5.1.

$$P = \underline{P}(1 - (\overline{P} - \underline{P})) \tag{5.1}$$

This equation wraps up the survivability of the cell as being the lower probability which our belief by our confidence in the probability values of this cell given by the complement of the difference in probabilities between the upper and lower probabilities. The creation of the planning map from the imprecise probability map representation provides the map as given in figure 5.1. This is created by the `updateIPMAP` node and published for the survivable path planner to use.

5.3 Modified RRT for survivable path planning

In order to find the most survivable path from the planning map we use a sampling based planning algorithm of a modified RRT. The algorithm of the modified RRT is shown in figure 5.2. In practice we dilate the obstacles in the map by the radius of the robot to first find the configuration space of the robot before starting to plan. This ensures that we only sample points that the robot can safely be positioned at. Similarly the local planner also dilates its obstacle to create a local cost-map for navigation.

```
T.init()
currentRisk = min(map)
While(PathNotFound):
    point = sampleMap(currRisk ,targetBias)
    If currentRisk >= maxRisk: break
    If point==None or  iter >=maxIter:
        currentRisk += stepSize
        Iter = 0
    C = FindCloset(T, point)
    If dist(point, C) > delta:
        point = pointInDirection(point)
    If !collision(C, point):
        T.add(point, parent = C)
        If point == target: break
    Iter+=1
```

FIGURE 5.2: The modified RRT pseudoscode

The modified RRT works along the same basic principle as the basic RRT algorithm. We first introduce a max risk term that is one of the planners global parameters along with the global target position. The max risk term denotes the max value of survivability that the planner is willing to sample before considering the cell unsurvivable. We start the RRT planner by initializing the RRT tree T by adding the current starting point of the robot. The tree will contain all the nodes that are sampled and have survivable path to them from the current position. The current risk to start out is set to the lowest possible risk from the whole map, and as we use up all the lowest risk cells we will have to increase our risk by the stepSize variable, until we either find a path to the target or we reach the max risk term threshold. At each iteration we first sample from the map by looking at only cell with a risk less than or equal to the current risk and use this as our random point. However for every 5th

sampling we make the random point equal the target cell as we want to encourage exploring the tree in the general direction towards the target. Then the closest node in the tree to the random point is found and if it is within delta L2 metric distance away and has no obstacle between them, then it is added to the tree. If not then a point delta away along the closest node, random point vector is checked for collision with any obstacles between them and if not then added to the tree. Otherwise we move onto the next iteration. If we reach the max number of iterations then we make sure to reset it and to increase our current risk level so as to continue exploring the next most survivable cells in the map. Hence by systematically sampling for the most survivable cells first we can nearly guarantee to find a path that is the most survivable.

Sample results for the world scenario considered so far using the modified RRT algorithm is shown in the figures 5.3 and 5.4. The black nodes represent the tree that was built and the green nodes are the nodes in the tree which are part of the path to the target. Both these paths were found within the specified max iteration limit of 1000 iterations and their respective waypoints for the these paths were published to allow the local planner to plan the paths between the way points by taking into account the constraints of the robot itself and using obstacle avoidance to prevent running into any objects.

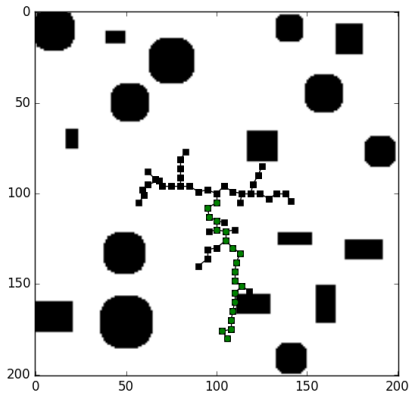

FIGURE 5.3: The planning map with global survivable path planned to target position (4,0.35)

While the above figures show the scenario when a path is found to the target, if there was no path found then the node actively asks the human explicitly if any path exists, and if the answer is yes the human is asked if he wants to increase the max risk level and continue path planning to find a new path to the target or simply stop and return to previous robot position. In this way the robot is able to get the information it needs to continue to path plan and to get to the target position to finish its mission.

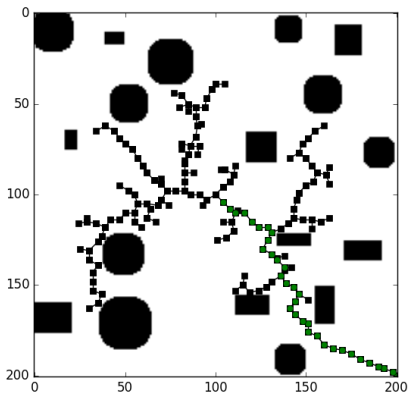

FIGURE 5.4: The planning map with global survivable path planned to target position (5,5)

Chapter 6

Summary and Conclusion

6.1 Summary of the research and Conclusion

This research was about trying to devise a method that could help improve the safety of navigation of individual robots in a human-robot team in the backdrop of post-disaster scenarios such as a building collapse. The main issues that makes navigation hard in these scenarios are that the environment tends to be very cluttered, hazardous to our robot and potentially unknown. However, in this research we considered the specific idea of human teammates in the human robot team, trying to help out the robots in modeling potentially un-modeled hazards in their neighborhood by passing on this information to the robots through speech in natural language. This provided many challenges in our hazard modeling task such as the creation of the quantitative model of the hazard from inherently imprecise human input, and creation of the maps and how to update it given that we don't have a model for the human sensor. Finally planning in this map to find the most survivable paths was also one of challenges. We have tackled these challenges by using a novel map representation using imprecise probabilities, and created and updated our current maps using an ARMA based technique. We modeled the undetected hazards and integrated the human input to the map using imprecise probabilities and the metrics defined. We then created a modified RRT algorithm to find the most survivable paths in our maps. To bring it all together we have created a ROS package that integrates all these different processes so that we can create a mixed initiative survivable path planner that can receive and integrate human inputs for finding the most survivable paths in the map.

6.2 Future work

There is still much more that can be done to improve the methods that we have used. To continue there needs to some additional extensive testing conducted using a local planner that is integrated with the ROS package. Then the tests have to be conducted in both simulations and in real life to see the measured effectiveness of our method as opposed to already existing methods that are human advisory information agnostic. The ROS package should also be given a good speech recognizer to parse the human input into the spatial specification. The research has been one of a possibly few that have used imprecise probabilities in path planning, and this is sub area that needs to be studied deeper. Imprecise probabilities have been helpful in capturing the human input in our research and has some advantages over Bayesian probability such as the automatic confidence level on the probability that is not available in Bayesian probabilities. Hence, the use of imprecise probabilities in path planning and more robotics application is certainly worth looking into as well.

Bibliography

Augustin, Thomas et al. (2014). *Introduction to imprecise probabilities*. John Wiley & Sons.

Cho, Beom-Seok, Se-Hong Park, and Min-Cheol Lee (2015). "Visibility and survivability map based path planning and its simulation". In: *Ubiquitous Robots and Ambient Intelligence (URAI), 2015 12th International Conference on*. IEEE, pp. 482–484.

Elfes, Alberto (1990). "Occupancy grids: A stochastic spatial representation for active robot perception". In: *Proceedings of the Sixth Conference on Uncertainty in AI*. Vol. 2929, p. 6.

Goodrich, Michael A and Alan C Schultz (2007). "Human-robot interaction: a survey". In: *Foundations and trends in human-computer interaction* 1.3, pp. 203–275.

Hornung, Armin et al. (2013). "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees". In: *Autonomous Robots*. Software available at http://octomap.github.com. DOI: 10.1007/s10514-012-9321-0. URL: http://octomap.github.com.

Jiang, Shu and Ronald C Arkin (2015). "Mixed-Initiative Human-Robot Interaction: Definition, Taxonomy, and Survey". In: *Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on*. IEEE, pp. 954–961.

Kim, Min-Ho and Min-Cheol Lee (2013). "UGV optimal path planning algorithm by using 3D survivability map". In: *Ubiquitous Robots and Ambient Intelligence (URAI), 2013 10th International Conference on*. IEEE, pp. 325–327.

LaValle, Steven M (1998). "Rapidly-exploring random trees: A new tool for path planning". In:

Lei, Ze et al. (2014). "Artificial robot navigation based on gesture and speech recognition". In: *Security, Pattern Analysis, and Cybernetics (SPAC), 2014 International Conference on*. IEEE, pp. 323–327.

Miranda, Enrique (2008). "A survey of the theory of coherent lower previsions". In: *International Journal of Approximate Reasoning* 48.2, pp. 628–658.

Montemerlo, Michael et al. (2002). "FastSLAM: A factored solution to the simultaneous localization and mapping problem". In: *Aaai/iaai* 593598.

Oh, Jean H et al. (2015). "Toward Mobile Robots Reasoning Like Humans." In: *AAAI*, pp. 1371–1379.

Shapira, Yaniv and Noa Agmon (2015). "Path planning for optimizing survivability of multi-robot formation in adversarial environments". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, pp. 4544–4549.

Skubic, Marjorie et al. (2004). "Spatial language for human-robot dialogs". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 34.2, pp. 154–167.