Interactive multi-objective path planning through a palette-based user interface

Meher T. Shaikh, Michael A. Goodrich, Daqing Yi, and Joseph Hoehne

Brigham Young University, Provo, UT, USA

ABSTRACT

In a problem where a human uses supervisory control to manage robot path-planning, there are times when human does the path planning, and if satisfied commits those paths to be executed by the robot, and the robot executes that plan. In planning a path, the robot often uses an optimization algorithm that maximizes or minimizes an objective. When a human is assigned the task of path planning for robot, the human may care about multiple objectives. This work proposes a graphical user interface (GUI) designed for interactive robot path-planning when an operator may prefer one objective over others or care about how multiple objectives are traded off. The GUI represents multiple objectives using the metaphor of an artist's palette. A distinct color is used to represent each objective, and tradeoffs among objectives are balanced in a manner that an artist mixes colors to get the desired shade of color. Thus, human intent is analogous to the artist's shade of color. We call the GUI an "Adverb Palette" where the word "Adverb" represents a specific type of objective for the path, such as the adverbs "quickly" and "safely" in the commands: "travel the path quickly", "make the journey safely". The novel interactive interface provides the user an opportunity to evaluate various alternatives (that tradeoff between different objectives) by allowing her to visualize the instantaneous outcomes that result from her actions on the interface. In addition to assisting analysis of various solutions given by an optimization algorithm, the palette has additional feature of allowing the user to define and visualize her own paths, by means of waypoints (guiding locations) thereby spanning variety for planning. The goal of the Adverb Palette is thus to provide a way for the user and robot to find an acceptable solution even though they use very different representations of the problem. Subjective evaluations suggest that even non-experts in robotics can carry out the planning tasks with a great deal of flexibility using the adverb palette.

Keywords: human-robot interaction, multi-objective decision making, user interface, supervisory control

1. INTRODUCTION

Consider a problem where a human uses supervisory control to manage robot path-planning by evaluating multiple paths generated by an algorithm and assigning a robot to execute one of the paths. Given a set of paths, the task of choosing among these multiple paths places a burden on a human operator, as the human may find it difficult to compare the paths against each other. This triggers the need of a robust and intuitive interface that can act on the output of well established path-planning algorithms, and allow the user to select the most desired path in a way that keeps human workload within acceptable bounds.

This paper proposes a novel human-robot interface called as an *Adverb Palette (AP)* to help the operator issue commands to the robot to take a specific path from the many available paths. Figure 1 shows the adverb palette. On the left side of the interface, the map shows in gray all potential paths that a robot can take, and the right side of the interface provides an area that can be used by the human to find tradeoffs among the paths. Based on the command issued on right side panel, one of the gray paths gets highlighted on the left panel.

An *adverb* encodes the objective associated by a verbal command given by the human to the robot. For example consider a command, "Go from point A to point B quickly." The adverb "quickly" in the command indicates minimizing path length, in other words asking the robot to take the shortest path from point A to point B. For a command, "Go from point A to point B quickly and safely," two objectives need to be minimized: path length and risk of being exposed to threats in the environment, respectively. *AP* is an interface where such commands can be interactively evaluated by a user. The execution of the command by the robot, and robot's performance evaluation is not in the scope of this work and is left for future work. The goal of this paper is to present th adverb palette interface and subjectively compare it to two other interfaces.

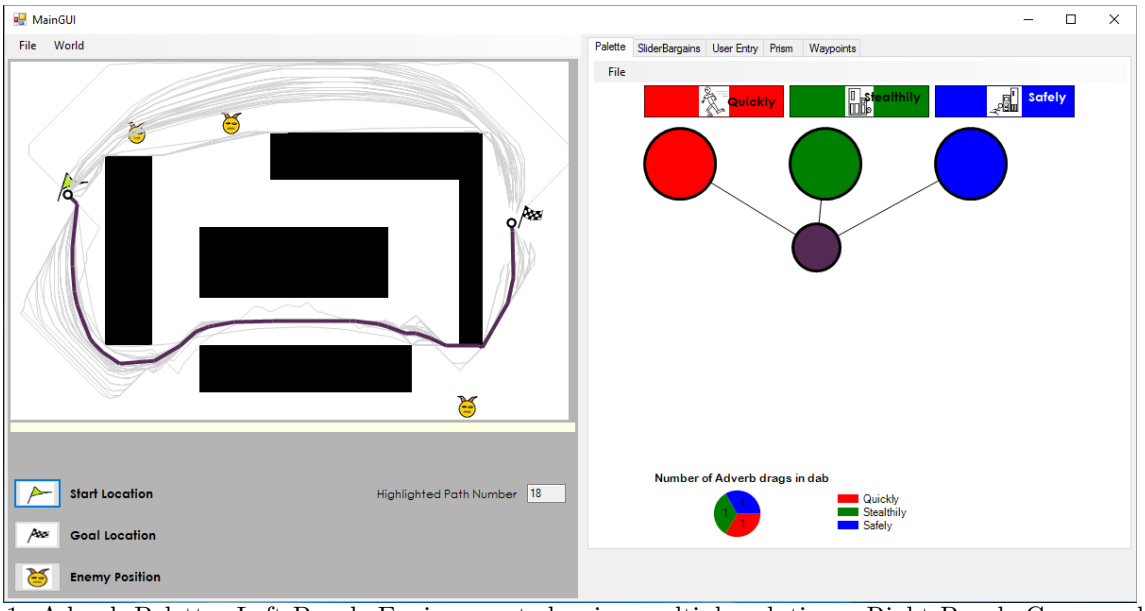

Figure 1: Adverb Palette: Left Panel: Environment showing multiple solutions. Right Panel: Command interface.

Although there exist many algorithms for multi-objective optimization (see, for example,[1–6]) we use the MORRF* algorithm[4] as it has demonstrated both effectiveness and efficiency in generating Pareto optimal solutions. A solution is Pareto optimal if there is no other solution that is better for every objective. Figure 2 shows the Pareto optimal paths discovered by the MORRF* algorithm in a simple world with two objectives to minimized. Each point in the curve represents a path and its associated cost of *objective 1* and *objective 2*. The blue square in the top left corner represents a path where the cost of objective 1 is minimum, and similarly the blue square at the bottom right corner represents a path for which the cost of *objective 2* is minimum.

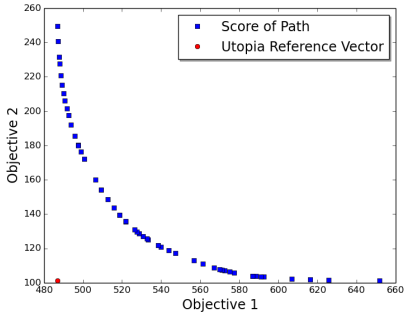

Figure 2: Path Planning with MORRF* for two objectives.

We quote the formal definition of the path-planning problem from[4] as follows: Consider a bounded, connected open set $X \subset \mathbb{R}^d$, an obstacle space X_{obs}, an initial state x_{init}, and a goal region x_{goal}. Consider the set of K objectives determined by a vector function $\boldsymbol{c}(\cdot) = [c_1(\cdot), \ldots, c_K(\cdot)]^T$ defined by $\boldsymbol{c} : \mathbb{X} \to \mathbb{R}^K$. Denote the obstacle-free space by $X_{\text{free}} = X \setminus X_{\text{obs}}$. Note that $\boldsymbol{c}$ is defined for all points in X in free space.

Again quoting from,[4] the solutions that satisfy the following equation are Pareto Optimal.

$$\arg\min_{x} \max_{1 \le k \le K} \{\lambda_k \left(|c_k(x) - \boldsymbol{z}_k^{\text{utop}}|\right)\} \tag{1}$$

where $\boldsymbol{\lambda} = [\lambda_1, \cdots, \lambda_K]^T$ is a weighting vector such that $\sum_{k=1}^{K} \lambda_k = 1$, $\boldsymbol{z}^{\text{utop}} = [z_1^*, \cdots, z_K^*]^T$ denotes the Utopia reference vector, and finally x denotes a potential solution. For details see.[4]

Given the Pareto optimal solution set that satisfy Equation 1, where each solution represents a path that goes from point A to point B, the goal is to enable a user to find a tradeoff that best expresses his or her intent. Expressing intent has two subproblems to be solved:

1. Design an interface to construe human intent, and,
2. Design an algorithm that translates from the interface input to one of the Pareto optimal paths that most closely matches human intent.

2. RELATED WORK

The design and use of Adverb Palette (AP) relates to user interface design, multi-criterion/attribute/objective decision making, human-machine systems, human-factors, human-robot interaction, ecological interface design, cognitive engineering systems etc.

Making trade-offs in decision-making is known as multiple criterion decision-making,[7] multiple attribute decision-making,[8] etc. The goal however remains the same as to make preference decisions over available alternatives, or in other words, to choose from among a finite set of discrete alternatives.[9] This paper uses three objectives: minimizing distance from the robot's start location to a goal location, avoiding exposure of the robot to one or more enemies, and avoiding collisions with obstacles.

A great deal of emphasis has been on designing powerful and easy to use interfaces.[10–17] Many in the field also elaborate on the challenges and complexity of practical design problems. The AP is closely related to ecological interface design,[11] which is based on a taxonomy of skills, rules, and knowledge used in cognitive control.[18] An ecological interface should not contribute to the difficulty of task, and at the same time it should support the entire range of activities that the operators are faced with. The term ecological (relation between organism and the the environment) corresponds to the operator and the work environment. In our scenario, the work environment is the n-dimensional space that the robot is going to navigate from one location to another, and the AP is at the disposal of an operator to make effective path planning decision. To make the robotic-path planning task easier and intuitive for the operator, we have used the metaphor of a palette. In the current work, the three objectives that we consider are represented by the colors red, green and blue respectively. Colors are a strong stimuli,[19] though the interfaces in this paper would not work for color blind individuals.

Although teams of humans and robots working as peers may be forthcoming, most robots are managed using supervisory control.[20] For example, in search and rescue operation where the robot may be in unstructured and unfamiliar environments,[14,21] strategic decision-making may be necessarily performed by an operator. Designing interfaces for supervisory control is one element of the field of human-robot interaction (HRI).[22] Designing intuitive and efficient interfaces has been a challenging issue in HRI.[23,24] However, significant research on HRI is inspired by the principles, and levels of autonomy (LOA) given by.[25] According to types and levels of human interaction,[25] a design involving human-machine interaction varies according to level of automation required. Studies has also been conducted in adjusting the autonomy responding to the environment and workload changes.[26] Considering the given LOA, AP allows the user to make decisions on paths, and then delegate the task to the robot. Note that recent work has identified a critical need to move past the limitations of LOAs on human-robot teaming.[27]

As previously mentioned, robots are transitioning from functional tools to interactive teammates.[28–31] Robust level of robot intelligence will cause HRI to evolve beyond command and control methods. Human mental models[30,31] for human-robot teams dictate how humans expect a robot to plan and execute tactical movement commands under constraints like "navigate quickly", "navigate stealthily", and "navigate safely". AP provides a medium to explicitly express the human expectations on the interface with the help of adverbs in order to plan the path for robot.

3. ADVERB PALETTE

The *Adverb Palette (AP)* is a mouse-based interactive graphical user interface designed for motion-planning for robots. It provides selection and visualization of possible routes/paths that a robot can take to go from a start location x_{init} to a goal region x_{goal}, given the configuration space. The *AP* interface helps the user to blend objectives in a way a painter blends colors on a palette. A blend/mixture of objectives corresponds to one path from the available Pareto optimal paths. As shown in Figure 1, the Adverb Palette has two parts: the *map* in the left panel that aids visualization and the *command interface (CI)* in the right panel through which the user can balance different adverbs. In short, the right panel is the command area for the user actions, and the map is the area that gets updated by highlighting the path according to the user's action on the *CI*. We will explore two types of AP interfaces.

Consider an *AP* interface that supports three adverbs: Quickly, Stealthily, and Safely, symbolized by colors *red*, *green* and *blue*, respectively on the CI.

- Quickly: A command to the robot to consider a path which has shortest distance.
- Stealthily: A command to the robot to consider a path that avoids being viewed by enemies.
- Safely: A command to the robot to consider a path that stays away from obstacles.

We have developed the adverb palette and a complementary interface that uses a different CI which we call the *sliders* interface. We have also implemented a baseline command interface *waypoints input*. The map is common to all these options. In each case, the map shows all the routes (paths) in gray, and with no user action a highlighted path is displayed that gives equal preference to all the adverbs. Each of the options has a different way of balancing the adverbs for a particular tradeoff path. The following sections briefly describe each option.

3.1 Palette

The *palette* AP displays three initial circles called the "primary dabs," one for each adverb (objectives). The user can select a path that uses only one objective by clicking on one of the primary dabs; i.e. to take the shortest path, most stealthy path, or most safe path by clicking *quickly*, *stealthily*, or *safely*, respectively. The user can also issue a mixture of the above adverbs by dragging and dropping adverbs into the white area of the *CI* to create smaller circles called "paint dab" that blend colors, similar to the way a painter mixes the colors in her paint dab to get a desired shade. Line segments connect the primary and paint dabs, creating a tree structure that allows the human to see the proportions of each objective.

For example, the user can command a robot to go from location A to location B using a path that is both quick and stealthy, which is represented numerically as " 50% quickly, 50% stealthily, 0% safely ". This numerical mixture can be made by dragging the adverb *quickly* into a new paint dab, and later on dragging the adverb *Stealthily* onto it. Blending in multiple adverbs (colors) is thus equivalent to making trade-offs with multiple objectives. The default magenta paint dab in Figure 1 is thus an example of a mixture " 33.33% quickly, 33.33% stealthily, 33.33% safely". Such a mixture may be desired if there is a command for the robot to move from location A to B such that it should move fast, should stay away from both obstacles and the enemy.

The pie graph on the lower left area in the *CI* shows the proportion of each objective in a particular paint dab. The paint dab thus represents a human command. By mixing different paint dabs, a user can visualize the consequences of different commands. Figure 3 shows an example command where the user desires a path that is quick and safe but does not care about being seen by enemies.

We now discuss the algorithm used to translate the user commands into a tradeoff between objectives. Let K denote the number of objectives and let the user's action be denoted by x. The *CI* highlights the paths on the map for each potential solution x. Let dab_{d} represent any paint dab on *CI*. Let n_i be the number of times the user has dragged adverb i on dab_{d}, where $0 > i \leq K$. The total number of drags a user makes for dab_{d} is:

$$n = \sum_{i=1}^{K} n_i \tag{2}$$

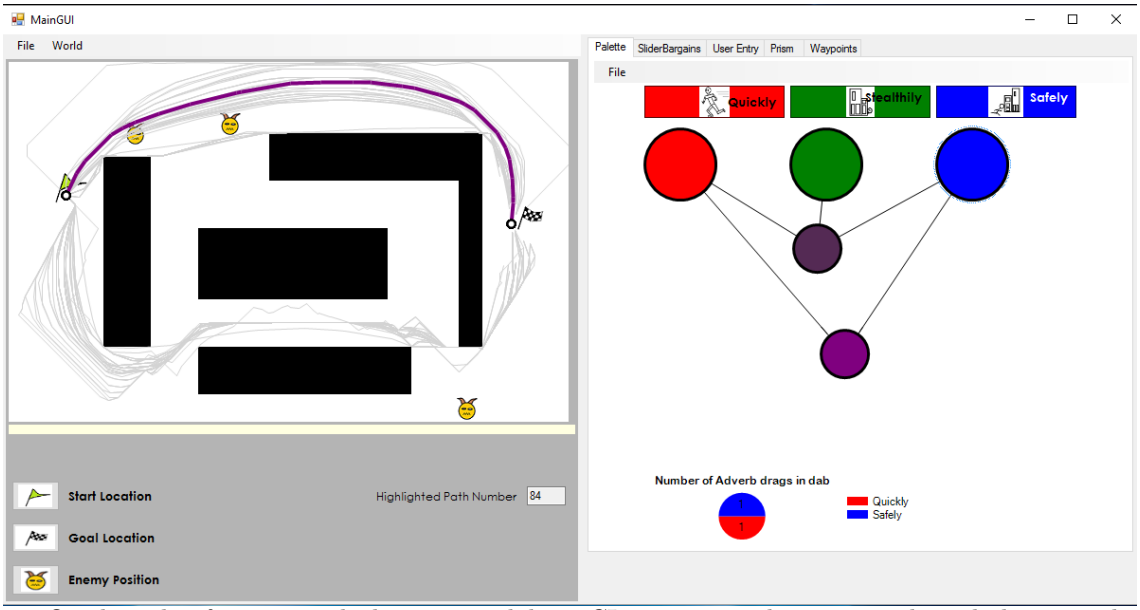

Figure 3: Quick and safe command: lowermost dab in CI represents the command, and the map shows the corresponding path (highlighted magenta)

Based on the number of adverb drags the user makes on the paint dab, the *user's intent*, also referred to as *human intent*, can be represented as a vector $\vec{h}_n$ as:

$$\vec{h}_n = [h_{1_n}, h_{2_n}, ..., h_{K_n}]^T \tag{3}$$

where h_{i_n} is computed as n_i/n. Therefore, $\sum_{i=1}^{K} h_{i_n} = 1$.

In Section 7, we will discuss the mapping between the human intent and the path that best matches the intent.

3.2 Sliders

Figure 4 shows the *slider* interface of the *CI*. Here the user adjusts the trackbars to get to a desired mixture, and the corresponding path from the left panel is selected. The three sliders represent the three adverbs. The user can issue any of the three primary commands to the robot i.e. to take the shortest path, most stealthy path, or most safe path by sliding the *red*, *green*, or *blue* slider to the maximum units, respectively.

If max_{scale} is the maximum number of units considered for each slider, then at any point of time, the sum of the units on each slider do not exceed the value max_{scale}. Therefore, if max_{scale} is 100, and if the *red*, *green*, or *blue* sliders are at say $33, 33, 34$ units respectively, then moving the *blue* slider to 60 units will cause a change to the slider units to $20, 20, 60$ units respectively. The adjustment thus guarantees that at any point of time the mixture represents a percentage of each of the adverbs. Unlike the palette, the user can discover paths while moving a slider, and settle down to certain position if she desires it; if the user moves one slider the other two sliders get updated automatically and the corresponding path gets shown on the map.

Let s_i is the number of units on slider i , and let max_{scale} be the maximum number of units considered for each slider. The maximum units are determined by the objective values for the set of paths returned by the MORRF* algorithm; the maximum unit is the cheapest path for that objective returned by MORRF* and the minimum unit is the most expensive path returned by MORRF*. The *human intent* can be represented as a vector $\vec{h}_s$ as:

$$\vec{h}_s = [h_{1_s}, h_{2_s}, \cdots, h_{K_s}]^T \tag{4}$$

where h_{i_s} is computed as s_i/max_{scale}. Therefore, $\sum_{i=1}^{K} h_{i_s} = 1$.

Let $\mathbb{M}(H, K)$ represent a matrix of slider values such that:

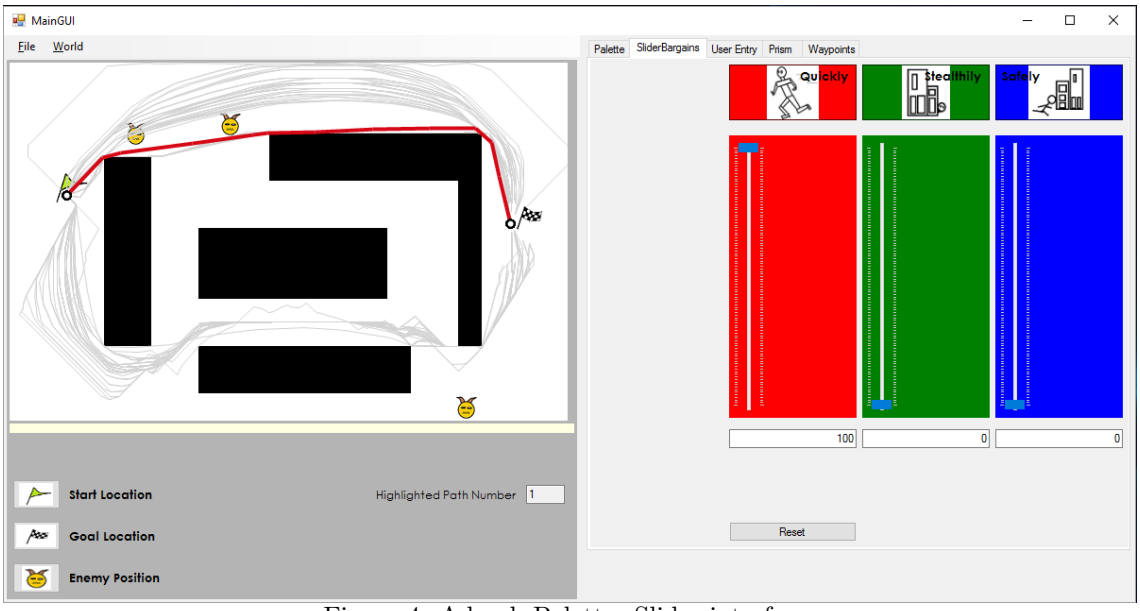

Figure 4: Adverb Palette: Slider interface.

- each row represents slider values, a unique combination of K-slider values treated as a vector $\vec{s}$
- each element in a row represents a slider value, where the slider value s_i is $s_i \in \mathbb{Z} : 0 \leq s_i \leq \max_{scale}$.
- the elements in each row add to the value max_{scale}
- H is the total number of rows in the matrix, where each row represents a unique combination of the K slider values (in other words there are H combinations), and
- K is the number of columns in the matrix. Each column represents one objective/slider.

In Section 7, we will discuss the mapping between the human intent and the path that best matches the intent.

3.3 Waypoints

The *waypoints* interface assists a user to construct her own path on the map by allowing her to provide location guidelines that the robot should visit while taking a path. Unlike the other two interfaces, the user here does not make a tradeoff among the available paths from the algorithm but instead makes her own path on the map. She can however compare her path with the best or worst with respect to an adverb based on the Pareto optimal paths' best and worst for that particular adverb. Figure 5 shows one of the paths constructed using the waypoints interface.

The interface allows to point to locations using 'Submit Waypoints' button. Then when the user is done submitting the main location points that she desire, she can commit these locations to form a path using 'Commit Waypoints' button. Any number of paths can be created using the mentioned button pair. Clicking on a paths' waypoint will give its corresponding path score. All paths can be cleared off to remove mess using 'Clear Waypoints' button. The path score is calculated in a way similar to how the MORRF* algorithm calculates the score for each adverb for a path.

The following sections detail the approach towards finding the costs associated with paths generated by the computer algorithm, and the one generated by the *waypoints* interface.

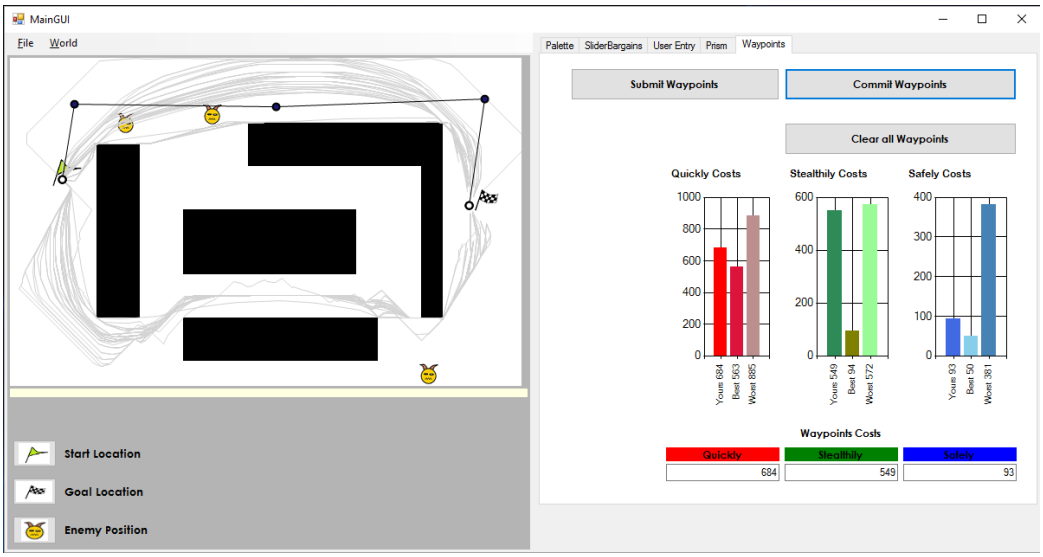

Figure 5: Waypoints Interface

4. COST FUNCTION FOR ADVERB 'STEALTHILY'

In the introduction, we discussed in brief the adverb 'stealthily'. Choosing a path that is stealthy means taking a path that is less likely to be detected or seen by the enemies that are posted in the region of interest. Therefore, we express the cost function in terms of the probability of the path being seen by enemy.

Let X_{E} be the set of the locations of n enemies, $X_{\mathrm{E}} = \{\mathbf{x}_{e_i} | \mathbf{x}_{e_i} \in X_{\mathrm{free}}\}$, and let the location of the robot be denoted by $\mathbf{x}_{\mathrm{rob}}$. We define the cost of stealthiness for any location of the robot $\mathbf{x}_{\mathrm{rob}} \in X_{\mathrm{free}}$, in terms of the probability of this point being seen. Let $C_{\mathrm{stealth}}(\mathbf{x}_{\mathrm{rob}}, X_{\mathrm{E}})$ denote the stealthily cost given the positions of the robots and the enemies. Defining this as the probability of being seen by any enemy gives:

$$C_{\mathrm{stealth}}(\mathbf{x}_{\mathrm{rob}}, X_{\mathrm{E}}) = P_{\mathrm{Seen}}(\mathrm{true}). \tag{5}$$

Equation 5 defines the cost in terms of the probability that the robot is seen by at least one enemy. We will now create a Bayesian network that allows us to compute $P_{\mathrm{Seen}}(\mathrm{true})$. Formally, we say that the agent has been seen if it has been seen by one or more enemies. Thus, we have a family of boolean random variables Seen_i, one for each enemy, and the Seen random variable is an accumulation of these.

This means that we will compute $P_{\mathrm{Seen}}(\mathrm{true})$ as the marginal distribution from the joint probability of all random variables as follows:

$$P_{\mathrm{Seen}}(\mathrm{true}) = \sum_{s_1,\ldots,s_n} P_{\mathrm{Seen},\mathrm{Seen}_1,\ldots\mathrm{Seen}_2}(\mathrm{true}, s_1, \ldots, s_n). \tag{6}$$

We now construct the Bayesian network by which this joint probability will be computed. We adopt a Noisy OR network because it matches our intention that the robot is seen if it is seen by any enemy [32, Chapter 14].

Computing the joint distribution will be done in two steps: First, we propose a simple Boolean network that models how the *Seen* random variable relates to the set of *Seen by enemy i* random variables. Second, we propose a second simple Boolean network that models how the *Seen by enemy i* random variable can be created from component parts.

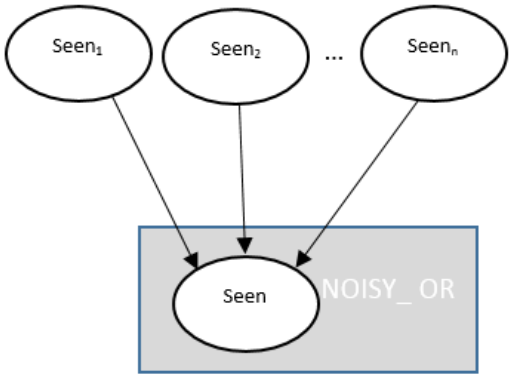

Figure 6: Probability of $\mathbf{x}_{\text{rob}}$ being seen by any enemy (modelled by Noisy_OR).

4.1 Seen by Any Enemy

Figure 6 illustrates a Bayesian network that models the probability of being seen as a function by any of the enemies. The Bayesian network uses a Noisy OR model. In the Noisy OR model, we construct a conditional probability table for $P_{\text{Seen}|\text{Seen}_i,\cdots\text{Seen}_n}(\text{true}|s_1,\cdots,s_n)$ for each $s_i \in \{\text{true},\text{false}\}$.

The algorithm for constructing this table assigns the probability of $n+1$ rows in the table, computing the probability of every other row from these initial assignments. The rows that are assigned values correspond to situations where one and only one enemy, say enemy number i, sees the robot and all other fail to see the robot. This means that values are assigned for

$$P_{\text{Seen}|\text{Seen}_1,\ldots\text{Seen}_n}(\text{true}|\text{false},\ldots,\text{false},\text{true},\text{false},\ldots,\text{false}) = p_i \tag{7}$$

where the *true* on the right side of the conditioning bar occurs at the position corresponding to random variable Seen_i. For example, Table 1 shows an example conditional probability table when there are 3 enemies in the environment. As illustrated in the figure, when there are n enemies then we need to specify n values.

Seen_1	Seen_2	Seen_3	p_i
T	F	F	p_1
F	T	F	p_2
F	F	T	p_3
F	T	T	$1-[(1-p_2)\times(1-p_3)]$

Table 1: Conditional Probability Table for three enemies: T=true F=false.

By convention, the probability of the Seen random variable being true given that all of its parent random variables are false is zero. Values at other positions are assigned as follows:

$$P_{\text{Seen}|\text{Seen}_1,\cdots,\text{Seen}_n}(\text{true} \mid s_1,\cdots,s_n) = 1 - \prod_{\{i:s_i=\text{true}\}} (1-p_i) \tag{8}$$

where the p_i is defined in Equation 7. The last row of Table 1 illustrates this situation.

Equations 7-8 define a noisy-OR in terms of tunable parameters p_i. For simplicity, we let $p_i = 1$ for all i. In the context of the cost associated with the "stealthily" adverb, this produces the effect of saying that there is an equal cost to the robot if one, two, or more enemies see the robot. Formally, when $\forall i p_i = 1$ implies that

$$P_{\text{Seen}|\text{Seen}_1,\ldots\text{Seen}_n}(\text{true}|s_1,\ldots,s_n) = \begin{cases} 1 & \text{if any } s_i = \text{true} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Equation 9 leads to a convenient form for computing the marginal probability P_{Seen},

$$
\begin{aligned}
P_{\text{Seen}}(\text{true}) &= \sum_{s_1,\cdots,s_n} P_{\text{Seen},\text{Seen}_1,\cdots\text{Seen}_n}(\text{true} \mid s_1,\cdots,s_n) \\
&= \sum_{s_1,\cdots,s_n} P_{\text{Seen}|\text{Seen}_1,\cdots,\text{Seen}_n}(\text{true} \mid s_1,\cdots,s_n) \prod_{i=1}^{n} P_{\text{Seen}_i}(s_i) \\
&= \left[\sum_{s_1,\cdots,s_n} \prod_{i=1}^{n} P_{\text{Seen}_i}(s_i) \right] - \prod_{i=1}^{n} P_{\text{Seen}_i}(\text{false}),
\end{aligned} \tag{10}
$$

where the first line is how you compute a marginal distribution from a joint distribution, the second line exploits the conditional independence assumptions of the noisy-OR Bayesian network, and the last line follows from the fact that the conditional is only one or zero.

4.2 Detection Likelihood of a robot by an enemy e_i

Consider three factors that affect whether the robot at location $\mathbf{x}_{\text{rob}}$ can be seen by an enemy: the distance of $\mathbf{x}_{\text{rob}}$ to $\mathbf{x}_{\text{e}_\text{i}}$ encoded as detection range, the visibility of $\mathbf{x}_{\text{rob}}$ from $\mathbf{x}_{\text{e}_\text{i}}$ considering objects in the world, and visibility of $\mathbf{x}_{\text{rob}}$ from $\mathbf{x}_{\text{e}_\text{i}}$ considering the environment as terrain. The resulting effect after considering all the three factors for an individual enemy yields *detection likelihood* of $\mathbf{x}_{\text{rob}}$ by $\mathbf{x}_{\text{e}_\text{i}}$. This detection likelihood serves as the P_{Seen_i}, i.e. the probability of $\mathbf{x}_{\text{rob}}$ being detected or seen by $\mathbf{x}_{\text{e}_\text{i}}$. The factors contributing to this detection likelihood are computed as follows:

4.2.1 Detection range

A distant enemy is less likely to have an harmful effect at $\mathbf{x}_{\text{rob}}$ than an enemy that is standing next to it. This aspect is captured in *detection range* as $P_{\text{dRange}_i}(\text{Seen}_i; \boldsymbol{x}_{\text{rob}}, \boldsymbol{x}_e)$ which is calculated as a function of euclidean distance between the $\boldsymbol{x}_{\text{rob}}$ and $\boldsymbol{x}_e$, $\|\boldsymbol{x}_{\text{rob}} - \boldsymbol{x}_e\|$, in d-dimensional space, where $\boldsymbol{x}_{\text{rob}} \neq \boldsymbol{x}_e$.

$$
P_{\text{dRange}_\text{i}}(\text{Seen}_i; \boldsymbol{x}_{\text{rob}}, \boldsymbol{x}_e) = \begin{cases} 1 & \text{if } 0 \leq \|\boldsymbol{x}_{\text{rob}} - \boldsymbol{x}_e\| \leq \delta \\ m(\|\boldsymbol{x}_{\text{rob}} - \boldsymbol{x}_e\|) + b & \text{if } \delta < \|\boldsymbol{x}_{\text{rob}} - \boldsymbol{x}_e\| \leq D \\ 0 & \text{otherwise} \end{cases} \tag{11}
$$

where δ is the minimum distance at or below which the detection of $\boldsymbol{x}_{\text{rob}}$ by $\boldsymbol{x}_e$ is maximum, hence 1, and D is the maximum possible distance between two points across the configuration space. As $\|\boldsymbol{x}_{\text{rob}} - \boldsymbol{x}_e\|$ reaches D, the likelihood of detection approaches zero. Therefore, given this definite detection range, the probability of detection for any distance that lie between δ and D follows the equation of line formed between points $(\delta, 1)$ and $(D, 0)$ with O as origin, m representing its slope, and b being the y-intercept . Figure 7 illustrates the detection range curve.

4.2.2 Visibility

Two points in the world are said to be mutually visible to each other if a straight line segment can be drawn between them and none of the obstacle points lie on the line segment. Thus, if there lies no obstacle in between $\boldsymbol{x}_{\text{rob}}$ and the $\boldsymbol{x}_e$ then $\boldsymbol{x}_{\text{rob}}$ is visible to $\boldsymbol{x}_e$. *Visibility* between the two points is hence expressed as $P_{visible_i}(\text{Seen}_\text{i}; \text{x}_{\text{rob}}, \text{x}_\text{e}, \text{X}_{\text{obs}}) \in \{0, 1\}$ given by:

$$
P_{\text{visible}_i}(\text{Seen}_i; \boldsymbol{x}_{\text{rob}}, \boldsymbol{x}_e, X_{\text{obs}}) = \begin{cases} 1 & \text{if } \boldsymbol{x}_{\text{rob}} \text{ is visible from } \boldsymbol{x}_e \\ 0 & \text{otherwise} \end{cases} \tag{12}
$$

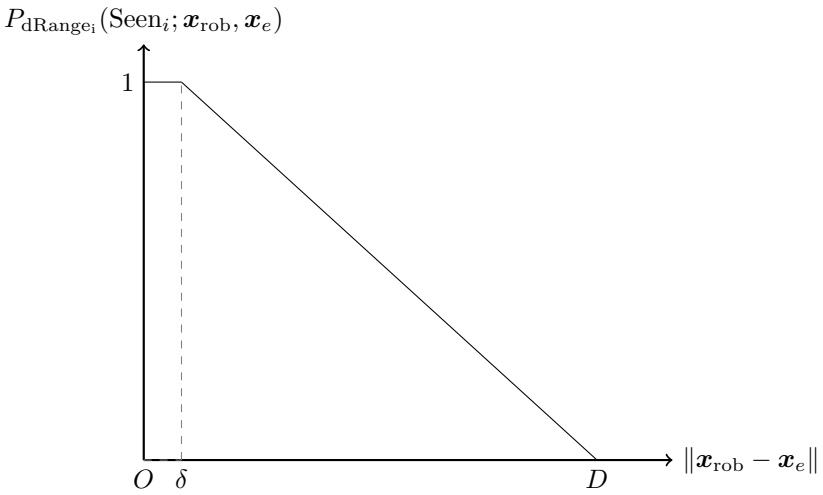

Figure 7: Detection range curve.

4.2.3 Viewshed

The navigation environment for a robot may be a rough natural terrain instead of a flat surface. Because of the characteristics of terrain such as hill tops or valleys, to an enemy the robot's position may be visible or it may be occluded by terrain. In a configuration space that has terrain characteristics, which points are visible from a given point is captured by a concept called *viewshed analysis* in literature.[33,34] The viewshed is computed based on a digital representation of the terrain called a *Digital Elevation Model*. Viewshed, $VS_{\boldsymbol{x}_p}$, of any point $\boldsymbol{x}_p$ is the set of all the points on the terrain that are in the line-of-sight of $\boldsymbol{x}_p$. For a stealthy path we desire $\mathbf{x}_{\text{rob}}$ to be **not** in the viewshed of the enemy point. We assume that the viewshed of each of the enemy points is available with us, and hence do not digress to show viewshed calculations.[35] Given the viewshed of $\mathbf{x}_{\text{e}_i}$, $VS_{\mathbf{x}_{\text{e}_i}}$, we represent the viewshed component of $\mathbf{x}_{\text{rob}}$ w.r.t. $\mathbf{x}_{\text{e}_i}$ as:

$$P_{\text{viewshed}_i}(\text{Seen}_i; VS_{\mathbf{x}_{\text{e}_i}}, \mathbf{x}_{\text{rob}}) = \begin{cases} 1 & \text{if } \mathbf{x}_{\text{rob}} \in VS_{\mathbf{x}_{\text{e}_i}} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

4.2.4 Fusion of detection range, visibility, and viewshed

Recall that we use Boolean network to model *Seen by enemy i* from component parts. Once we know the three components P_{dRange_i}, P_{visible_i}, and P_{viewshed_i} of $\mathbf{x}_{\text{rob}}$ being detected by $\mathbf{x}_{\text{e}_i}$, we combine them by using a NOISY-AND network. This allows us to compute P_{Seen_i} as the product of P_{dRange_i}, P_{visible_i}, and P_{viewshed_i}. In other words, the robot is seen by $\mathbf{x}_{\text{e}_i}$ and has a probability greater than zero only when all the three components yields results greater than zero. Therefore,

$$P_{\text{Seen}_i} = P_{\text{dRange}_i} * P_{\text{visible}_i} * P_{\text{viewshed}_i} \tag{14}$$

4.3 Stealthily cost for a path

Substituting P_{Seen_i} of Equation 14 for each of the enemy in Equation 10 we obtain $P_{\text{Seen}}(\text{true})$ that produces the stealthy cost $C_{\text{stealth}}(\mathbf{x}_{\text{rob}}, X_{\text{E}})$. The stealthy cost can thus be computed for every possible point of robot location in the configuration space. The stealthy cost of a path (starting from the initial state to the goal state) can be determined as the sum of the costs of individual points constituting the path. In short, if the path has significant number of points that have a high probability of being seen by the enemy, then the robot should avoid such paths if one desires stealthiness.

Figure 8 illustrates a world with three enemies and its corresponding stealthily objective function. The figure assumes a flat earth (i.e., viewshed is all points). Referring to Figure 8b, if the robot has to travel from the top left corner to the bottom right corner of the configuration space, a path that goes between the obstacles and

(a) World with three enemies.

(b) $C_{\text{stealth}}(\mathbf{x}_{\text{rob}}, X_{\text{E}})$

Figure 8: Stealthily objective function

lower part of the space is more stealthy than a path that goes through the left side of the left obstacle in the space.

5. COST FUNCTION FOR ADVERB 'SAFE'

To go from an initial state to the goal state, we would want the robot to take a collision free path w.r.t obstacles. A path that goes very close to the obstacles is unsafe. With this intuition, we represent the cost of safety associated with any location of the robot $\mathbf{x}_{\text{rob}}$ in the configuration space as a function of inverse distance between $\mathbf{x}_{\text{rob}}$ and the nearest obstacle in that space. Let $C_{\text{safe}}(\mathbf{x}_{\text{rob}}, X_{\text{obs}})$ denote the safety cost given the positions of the robot and the obstacles. Defining this as a function of distance to the nearest obstacle, we get:

$$C_{\text{safe}}(\mathbf{x}_{\text{rob}}, X_{\text{obs}}) = \begin{cases} 1 & \text{if } \exists \mathbf{x}_{\text{obs}} \text{ such that } \|\mathbf{x}_{\text{rob}} - \mathbf{x}_{\text{obs}}\| \leq \eta \\ 1/(\min_{\mathbf{x}_{\text{obs}} \in X_{\text{obs}}}\{\|\mathbf{x}_{\text{rob}} - \mathbf{x}_{\text{obs}}\|\}) & \text{if } \eta < \|\mathbf{x}_{\text{rob}} - \mathbf{x}_{\text{obs}}\| \leq D \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

where $\|\mathbf{x}_{\text{rob}} - \mathbf{x}_{\text{obs}}\|$ equals the euclidean distance between $\mathbf{x}_{\text{rob}}$ and $\mathbf{x}_{\text{obs}}$, η is the minimum distance at or below which the safety cost for $\mathbf{x}_{\text{rob}}$ is maximum, and finally $\mathbf{D}$ is as defined in subsection 4.2.1. If a path has many points for which $C_{\text{safe}}(\mathbf{x}_{\text{rob}}, \mathbf{x}_{\text{obs}})$ is high, i.e many path points are in the close proximity of obstacles, then the path is considered as a unsafe path.

5.1 Safety cost for a path

Using Equation 15 the safety cost can be computed for every possible point of robot location in the configuration space. Figure 9 illustrates the safety cost for every point in the configuration space w.r.t the given obstacles. As in the case of stealthily cost, the safety cost of a particular robotic path is the accumulation of the safety cost of individual points that make the path. Referring to Figure 9, if a robot has to travel from the top left corner to the bottom right corner, then a path that goes through in between the two obstacles (the vertical middle region of configuration space) would be relatively **unsafe** compared to a path that goes either from the left side or the right side of the environment.

6. COST FUNCTION FOR ADVERB 'QUICKLY'

The adverb 'quickly' is associated with minimizing path length. Assuming that the robot traverses every point in the configuration space with an equal cost, the 'quickly' cost is the Euclidean distance between the start and the goal position such that the obstacles do not intercept the path. Consider a path that is formed by **k** straight line segments, then the total path length is the summation of the euclidean distances of these individual line segments. Figure 10 shows two path options going from start location **A** to goal location **B**. It can been that the path distance of the path formed by points **ALMNOB** is less than the path distance formed by **AXYB**, hence the orange path is comparatively quicker than the blue path.

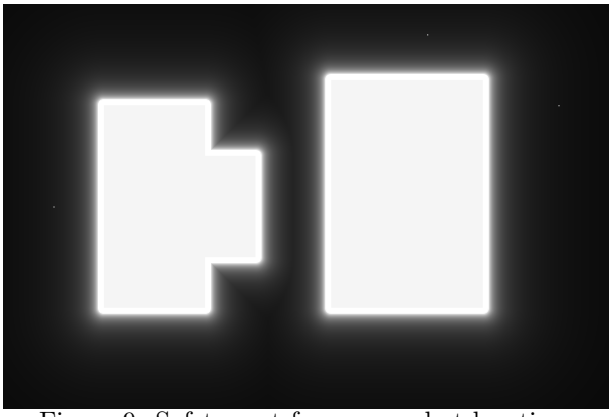

Figure 9: Safety cost for every robot location.

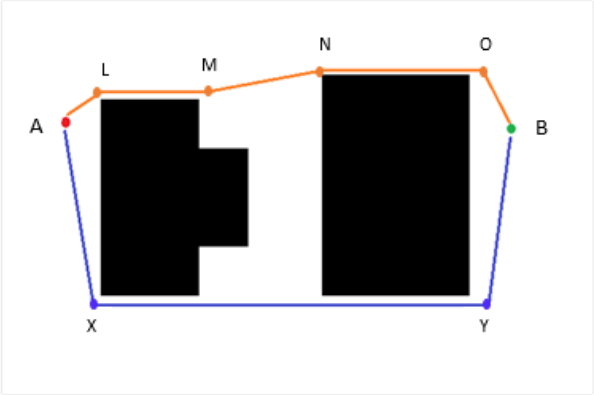

Figure 10: Path **ALMNOB** is quicker than the path **AXYB**.

7. COSINE SIMILARITY FOR PATH SELECTION

The *palette* and *sliders* interfaces produce a human intent vector denoted by $\vec{h} = [h_1, h_2, ..., h_K]^T$ where $\sum_{i=1}^{K} h_i = 1$. Each Pareto optimal path given by the MORRF* algorithm can also be represented as a vector, which we will soon discuss. Thus we have two vector representations, a human intent vector and a path vector. For a given human intent vector $\vec{h}$ we compute the cosine similarity with each of the available path vector. The path that shows the highest similarity to the human intent is the path that most closely matches the user's intent.

7.1 Path vector

Recall from Figure 2 that each point in the figure represents a Pareto optimal path, and the X-axis and the Y-axis show the cost for *objective 1* and *objective 2*, respectively. The path at the top left corner has the minimum cost for *objective 1* but is expensive in terms of *objective 2*. Similarly, a path that is in the middle of the curve provides a balance between both the objectives.

We convert the cost vector to a payoff vector so that we can compare the path with the positive human intent vector. Let there be S total solutions/paths and let the costs associated with a path $s_j \in S$ and objective $k \in \{1, \ldots, K\}$ be denoted by $c_k(s_j)$. The cost vector for path s_j is

$$\vec{c}(s_j) = [c_1(s_j), c_2(s_j), \cdots, c_K(s_j)]^T. \tag{16}$$

For each path vector, we convert the path costs to path payoffs by multiplying the path cost vector in Equation 16 by -1 yielding payoffs for each objective $p_k(s_j) = (-1)c_k(s_j)$ and a payoff vector of

$$\vec{p}(s_j) = (-1)\vec{c}(s_j). \tag{17}$$

Next, from among all the solutions S, we determine the minimum and the maximum payoff values for each of the objectives.

The payoff values in Equation 17 can be normalized to the bounds $[0.0, 1.0]$ using the formula

$$\hat{p}_k(s_j) = \frac{p_k(s_j) - \min_{s_\ell \in S}\{p_k(s_\ell)\}}{\max_{s_\ell \in S}\{p_k(s_\ell)\} - \min_{s_\ell \in S}\{p_k(s_\ell)\}}.$$

The corresponding normalized vector is given by

$$\boldsymbol{p}(s_j) = [\hat{p}_1(s_j), \hat{p}_2(s_j), \cdots, \hat{p}_K(s_j)]^T. \tag{18}$$

7.2 Cosine Similarity

The cosine similarity between a path vector, $\boldsymbol{p}(s_j)$, and the human intent vector is $\boldsymbol{h}$ is:

$$\text{CosineSimilarity}(\boldsymbol{h}, \boldsymbol{p}(s_j)) = \frac{\boldsymbol{h} \cdot \boldsymbol{p}(s_j)}{\|\boldsymbol{h}\|\|\boldsymbol{p}(s_j)\|} = \frac{\sum_{k=1}^{K} h_k p_k(s_j)}{\sqrt{\sum_{k=1}^{K} h_k^2}\sqrt{\sum_{k=1}^{K} p_k^2(s_j)}} \tag{19}$$

For certain user command $\boldsymbol{h}$, if some path $\boldsymbol{p}(s_j)$ ends up with same orientation, then they have the cosine similarity of 1, and if they are at, say, 90° apart then they end up with the cosine similarity of 0 indicating that they have nothing in common. Figure 11 shows comparison of path $\boldsymbol{p}(s_1)$ and $\boldsymbol{p}(s_2)$, w.r.t an example command $\boldsymbol{h}_{\text{ex}}$ on AP. The orange points are example paths on the Pareto front. Each of the blue point is an example of user's intent made through AP. It can be seen that since θ_2 is smaller than θ_1 , CosineSimilarity$(\boldsymbol{h}_{\text{ex}}, \boldsymbol{p}(s_2))$ is larger than that of CosineSimilarity$(\boldsymbol{h}_{\text{ex}}, \boldsymbol{p}(s_1))$. Thus, path $\boldsymbol{p}(s_2)$ will serve as a better path for $\boldsymbol{h}_{\text{ex}}$ than the path $\boldsymbol{p}(s_1)$. We thus formulate the best path for $\boldsymbol{h}$ as s^* as:

$$s* = \arg\max_{s_j \in S} \text{CosineSimilarity}(\boldsymbol{h}, \boldsymbol{p}(s_j)) \tag{20}$$

For a given intent vector, this best path is rendered on the map for both the *sliders* and *palette* interfaces.

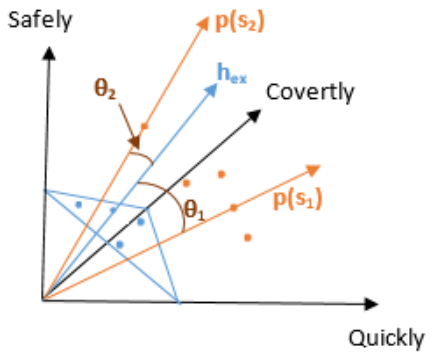

Figure 11: Path Comparison w.r.t example human intent vector $\boldsymbol{h}_{\text{ex}}$.

8. SUBJECTIVE EVALUATION

We have provided three interface designs. Each design can be extended to more than three objectives. This can be seen that from the fact that more basic color dabs can be added to the *palette* interface, as well as more control trackbars can be added to the *sliders*. Of course, blending more than three colors in the palette can cause ambiguities, so blending would need to be supplemented with something like textures.

While issuing a command through the *palette* interface the user can easily view the blend by observing the pie graph associated with the paint dab. Similarly, in *sliders* interface she can view the preference of each of

the adverbs in the textboxes below the sliders. By mapping the intent vector to a path vector via the *palette* or *sliders* interface, every user action updates the map immediately thereby giving the idea to the user of the immediate consequences of her action/command.

Each interface has it benefits and limitations. On the *palette* it is possible to have multiple user created paint dabs, each representing a particular command. Such a history is not available with *sliders*, because the moment one of the sliders is moved, the map gets updated to show the recent path according to the current user's action. On the other hand, while moving one of the adverb sliders it is possible to discover different paths associated with the different adverb values till certain adverb value is reached. The palette is devoid of this.

The user can also provide waypoints on the map to obtain a certain path. The resulting score for this waypoints path is calculated based on the cost of each path point given in subsections 4.3 and 5.1

9. SUMMARY AND FUTURE WORK

We have presented here three interfaces for exploring tradeoffs between robot paths with three objectives. In the future we plan to conduct a user study to measure which of the interfaces the user like, which is more easier to operate, and would capture time statistics with respect to each interface.

ACKNOWLEDGMENTS

This work is supported by the Army RCTA program. All results and conclusions are the responsibility of the authors and do not necessarily reflect the opinions of the funding source.

REFERENCES

[1] LaValle, S. M., "Rapidly-exploring random trees a new tool for path planning," (1998).

[2] Bruce, J. and Veloso, M., "Real-time randomized path planning for robot navigation," in [*Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference*], **3**, 2383–2388, IEEE (2002).

[3] Bry, A. and Roy, N., "Rapidly-exploring random belief trees for motion planning under uncertainty," in [*Robotics and Automation (ICRA), 2011 IEEE International Conference*], 723–730, IEEE (2011).

[4] Yi, D., Goodrich, M. A., and Seppi, K. D., "MORRF*: Sampling-based multi-objective motion planning," in [*Proceedings of the 24th International Conference on Artificial Intelligence*], 1733–1739, AAAI Press (2015).

[5] Kavraki, L. E., Švestka, P., Latombe, J.-C., and Overmars, M. H., "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on* **12**(4), 566–580 (1996).

[6] Ahmed, F. and Deb, K., "Multi-objective optimal path planning using elitist non-dominated sorting genetic algorithms," *Soft Computing* **17**(7), 1283–1299 (2013).

[7] Wallenius, J. and Zionts, S., [*Multiple criteria decision making: From early history to the 21st century*], World Scientific (2011).

[8] Hwang, C.-L. and Yoon, K., [*Multiple attribute decision making: Methods and applications a state-of-the-art survey*], vol. 186, Springer Science & Business Media (2012).

[9] Gal, T., Stewart, T., and Hanne, T., [*Multicriteria decision making: Advances in MCDM models, algorithms, theory, and applications*], vol. 21, Springer Science & Business Media (2013).

[10] Bennett, K. B. and Flach, J. M., [*Display and interface design: Subtle science, exact art*], CRC Press (2011).

[11] Vicente, K. J. and Rasmussen, J., "Ecological interface design: Theoretical foundations," *Systems, Man and Cybernetics, IEEE Transactions on* **22**(4), 589–606 (1992).

[12] Bennett, K. B., Posey, S. M., and Shattuck, L. G., "Ecological interface design for military command and control," *Journal of Cognitive Engineering and Decision Making* **2**(4), 349–385 (2008).

[13] Hall, D. S., Shattuck, L. G., and Bennett, K. B., "Evaluation of an ecological interface design for military command and control," *Journal of Cognitive Engineering and Decision Making* **6**(2), 165–193 (2012).

[14] Kadous, M. W., Sheh, R. K.-M., and Sammut, C., "Effective user interface design for rescue robotics," in [*Proceedings of the 1st ACM SIGCHI/SIGART Conference on Human-robot interaction*], 250–257, ACM (2006).

[15] Chen, J. Y., Haas, E. C., and Barnes, M. J., "Human performance issues and user interface design for teleoperated robots," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* **37**(6), 1231–1245 (2007).

[16] Baker, M., Casey, R., Keyes, B., and Yanco, H. A., "Improved interfaces for human-robot interaction in urban search and rescue.," in [*SMC (3)*], 2960–2965, Citeseer (2004).

[17] Adams, J. A., "Critical considerations for human-robot interface development," in [*Proceedings of 2002 AAAI Fall Symposium*], 1–8 (2002).

[18] Rasmussen, J., "Skills, rules, and knowledge; Signals, signs, and symbols, and other distinctions in human performance models," *Systems, Man and Cybernetics, IEEE Transactions on* (3), 257–266 (1983).

[19] Mahnke, F. H., [*Color, environment, and human response: An interdisciplinary understanding of color and its use as a beneficial element in the design of the architectural environment*], John Wiley & Sons (1996).

[20] Sheridan, T. B., [*Telerobotics, automation, and human supervisory control*], MIT press (1992).

[21] Lin, L. and Goodrich, M. A., "Sliding autonomy for UAV path-planning: Adding new dimensions to autonomy management," in [*Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*], 1615–1624, International Foundation for Autonomous Agents and Multiagent Systems (2015).

[22] Goodrich, M. A. and Schultz, A. C., "Human-robot interaction: A survey," *Foundations and trends in human-computer interaction* **1**(3), 203–275 (2007).

[23] Fang, H., Ong, S., and Nee, A., "Novel AR-based interface for human-robot interaction and visualization," *Advances in Manufacturing* **2**(4), 275–288 (2014).

[24] Driewer, F., Sauer, M., and Schilling, K., "Discussion of challenges for user interfaces in human-robot teams.," in [*EMCR*], Citeseer (2007).

[25] Parasuraman, R., Sheridan, T. B., and Wickens, C. D., "A model for types and levels of human interaction with automation," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* **30**(3), 286–297 (2000).

[26] Goodrich, M. A., McLain, T. W., Anderson, J. D., Sun, J., and Crandall, J. W., "Managing autonomy in robot teams: Observations from four experiments," in [*Proceedings of the ACM/IEEE International Conference on Human-robot interaction*], 25–32, ACM (2007).

[27] Johnson, M., Bradshaw, J. M., Feltovich, P. J., Jonker, C. M., van Riemsdijk, B., and Sierhuis, M., "The fundamental principle of coactive design: Interdependence must shape autonomy," in [*Coordination, organizations, institutions, and norms in agent systems VI*], 172–191, Springer (2011).

[28] Phillips, E., Ososky, S., Grove, J., and Jentsch, F., "From tools to teammates toward the development of appropriate mental models for intelligent robots," in [*Proceedings of the Human Factors and Ergonomics Society Annual Meeting*], **55**(1), 1491–1495, SAGE Publications (2011).

[29] Fong, T., Thorpe, C., and Baur, C., "Collaboration, dialogue, human-robot interaction," in [*Robotics Research*], 255–266, Springer (2003).

[30] Talone, A. B., Phillips, E., Ososky, S., and Jentsch, F., "An evaluation of human mental models of tactical robot movement," in [*Proceedings of the Human Factors and Ergonomics Society Annual Meeting*], **59**(1), 1558–1562, SAGE Publications (2015).

[31] Phillips, E., Ososky, S., and Jentsch, F., "An investigation of human decision-making in a human—robot team task," in [*Proceedings of the Human Factors and Ergonomics Society Annual Meeting*], **58**(1), 315–319, SAGE Publications (2014).

[32] Russell, S. and Norvig, P., [*Artificial intelligence: A modern approach*] (1995).

[33] Kim, Y.-H., Rana, S., and Wise, S., "Exploring multiple viewshed analysis using terrain features and optimisation techniques," *Computers & Geosciences* **30**(9), 1019–1032 (2004).

[34] STUCKY, J. L. D., "On applying viewshed analysis for determining least-cost paths on digital elevation models," *International Journal of Geographical Information Science* **12**(8), 891–905 (1998).

[35] Wang, J., Robinson, G. J., and White, K., "A fast solution to local viewshed computation using grid-based digital elevation models," *Photogrammetric Engineering and Remote Sensing* **62**(10), 1157–1164 (1996).