Risky Planning: Path Planning over Costmaps with a Probabilistically Bounded Speed-Accuracy Tradeoff

Liz Murphy
School of Engineering Systems, Queensland University of Technology
Email: liz.murphy@qut.edu.au

Paul Newman
Oxford University Mobile Robotics Group
Email: pnewman@robots.ox.ac.uk

***Abstract*— This paper is about generating plans over uncertain maps quickly. Our approach combines the ALT (A* search, landmarks and the triangle inequality) algorithm and risk heuristics to guide search over probabilistic cost maps. We build on previous work which generates probabilistic cost maps from aerial imagery and use these cost maps to precompute heuristics for searches such as A* and D* using the ALT technique. The resulting heuristics are probability distributions. We can speed up and direct search by characterising the risk we are prepared to take in gaining search efficiency while sacrificing optimal path length. Results are shown which demonstrate that ALT provides a good approximation to the true distribution of the heuristic, and which show efficiency increases in excess of 70% over normal heuristic search methods.**

I. INTRODUCTION

Typical approaches to path planning for field robots involve simplifying the robot's environment into a discretized costmap, where the cost of each grid cell is a scalar value proportional to the estimated cost of traversing the terrain thought to lie at that location. In the absence of perfect sensing and classification, this cost is actually an uncertainty distribution over the terrain associated with a particular cell. In [1] we argued that the typical 'scalar' approach to costmap creation throws away useful information, and presented a framework for producing probabilistic cost maps from aerial imagery.

In this work we capitalize on the probabilistic nature of these costmaps to 'speed up' path planning by providing guaranteed bounds on the sub-optimality of paths produced when we trade the accuracy of the results for the speed of finding a potential path. We achieve this by making the *heuristic* that controls the search probabilistic. The A* and D* family of algorithms, commonly used in field robotics, both belong to the heuristic search family. The heuristic is used to estimate how far a particular node is away from the goal (in A*, in D* the direction of search is reversed so the heuristic estimates the distance to the start) and is used in conjunction with the known distance from the start to estimate the shortest path from start to goal via that node. This total distance is used to prioritise the node against all other nodes in the search in terms of its likely contribution to finding the shortest path and determines which of the nodes under consideration is chosen for expansion next. By focussing the search towards the goal, heuristic searches find optimal paths more efficiently than comparable algorithms such as Dijkstra's search.

The more accurate the heuristic is, the less time we spend searching. A recent technique [2] precomputes distances between every cell in the map to and from a small set of *Landmarks* distributed throughout the map. These precomputed distances are used in conjunction with the triangle inequality to provide a lower bound on the heuristic distance for arbitrary node-goal pairs during an A* search. In our case this precomputation of path lengths involves the addition and subtraction of gaussian cell costs, which leads to a distribution which approximates the one we are interested in. This approximation governs the distance between the node and the goal.

Moreoften than not in robotic path planning we are interested in finding a 'good' path rather than the shortest possible path. *Anytime* algorithms recognize this and a suite of solutions have been developed to provide good trajectories to the robot quickly and improve on them if time allows [3] [4]. These algorithms work by relaxing the admissibility criterion - which normally requires that the heuristic estimate of the goal distance *always* be less than or equal to the true distance. Figure 1 shows the quandary that the admissibility criterion introduces into our probabilistic planning domain. Let $p(f_n)$ be the probability density function over the cost of traversing a path from start to goal through node n. Here, $p(f_{n_1})$ has the lower mean, but it is quite possible that the actual path cost $p(f_{n_2})$ through node n_2 is somewhat lower due to the longer tail of its distribution. Intuitively we would opt to expand n_1 as the shape of the density functions renders $f(n_1) > f(n_2)$ unlikely. However, an admissible heuristic would force us to expand n_2 as the longer tail of this distribution means that to act otherwise would risk overestimating the cost to the goal. We propose to solve this quandary by using the heuristic to quantify the level of risk we wish to take that the paths returned by the planner will be suboptimal.

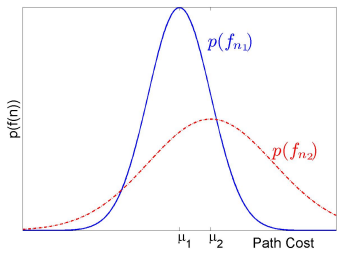

Fig. 1: Which node to expand next? Node n_1 with cost function f_1 promises a lower mean, but f_2 indicates n_2 has significant probability of offering a lower cost than n_1.

Pearl [5] [6] developed the R^*_δ algorithm to deal with cases in which the user wishes to invoke probability distributions in the admissibility criterion. The premise is simple - if we

This work was supported by The Australian Federation of University Women Queensland Branch

know the probability distribution governing the heuristic, we know how often, on average, it will overestimate the distance between the node and the goal and therefore we know how often it will overestimate the shortest path. The algorithm allows the specification of a level, δ which bounds the risk of returning a suboptimal path.

In this paper we show how the R^*_δ algorithm can be applied to probabilistic costmaps. We use ALT [2] to provide a good base estimate of the probability distribution governing the heuristic and learn a scaling parameter to improve this estimate on the fly. The results show that for minimal precomputation we can obtain 73% improvement in search efficiency and quantify the risk of the path being longer than optimal. Together with the work in [1], this paper provides an end-to-end framework for incorporating uncertainty into path planning for mobile robots.

II. RELATED WORK

Many modern planning algorithms [7] [8] [9] build on the idea of A* search [10]. A^* is a best-first graph search algorithm which finds the shortest path between a start and a goal node. Central to the A* algorithm and its variants is a distance-plus-cost heuristic which determines the order in which nodes of the graph being searched are visited. This heuristic itself has two parts, one is the cost of travelling from the starting node to the current node n under examination, usually denoted $g(n)$. The second part, denoted $h(n)$ is a heuristic estimate of the distance from n to the goal. The sum $f(n) = g(n) + h(n)$ determines the priority of the node n on the OPEN queue of nodes awaiting expansion.

Typically in the A* paradigm the heuristic must be consistent (monotonic) and admissible, meaning it never overestimates the cost of reaching the goal. In the extreme, when $h = 0$ for every node, A^* becomes Dijkstra's algorithm. Conversely, if we have perfect knowledge of the problem space and can tailor the heuristic to make use of this knowledge then $h = h_{optimal}$ and fewest nodes are expanded. It is the admissibility condition that guarantees that A* will find an optimal solution path if it exists. However, it has the unfortunate by-product of frequently leading the search to spend large amounts of time deliberating between roughly equal solution paths and does not give us the option of terminating the search with an acceptable but not optimal solution path. In *Bandwidth Search* [11] Harris showed that if the heuristic overestimation is bounded by ε then the resulting cost cannot exceed $(1 + \varepsilon)C^*$, where C^* is the optimal path cost. The concept of *ε-admissibility* has more recently been used in [4] to produce bounded suboptimal solutions which produce increasingly better solutions to planning problems in a given time window.

The idea of placing risk bounds on heuristics was introduced by Pearl [5] [6] who sought to invoke likelihood considerations into the admissibility guarantee. The resulting R^*_δ algorithm is a variant of $A*$ that relaxes Harris' $\varepsilon - admissibility$ condition even further, to allow the use of a precise estimator that may occasionally overestimate h^* by more than ε, such as is the case when the arc costs of a graph are known to be drawn from a probability distribution.

In [5] the performance of R^*_δ was demonstrated on an instance of the Travelling Salesman Problem with statistics generated from multiple previous searches. Pearl used regression techniques to fit parameters to a best fit model $\hat{h}_{opt}$, a heuristic that accurately estimates the future cost of the search but occasionally overestimates this distance. The true minimum arc cost h is assumed to be normally distributed with mean $\hat{h}_{opt}$ and variance σ^2 estimated from the statistics.

Obviously, the computation of heuristics for a Travelling Salesman search bears little relevance to problem domains encountered by a mobile robot. What we wish to demonstrate in this paper is the effectiveness of risk bounded search in searching terrain that may be traversed by a mobile robot. In [1] we demonstrated how probabilistic maps can be obtained. In the work that follows we will demonstrate how these probabilistic maps can be used to construct probabilistic heuristics for use with R^*_δ search.

When applying A^* search to large graphs such as those created by an overhead map it is desirable to restrict the point to point search to examining only relevant areas of the input graph. Recently there has been a focus on preprocessing the graph to obtain better heuristics. The ALT (so named because of its use of A^*, Landmark and the Triangle inequality) algorithm [2] [12] selects a small set of vertices as landmarks and uses precomputed distances from every node to these landmarks in evaluating the A* heuristic. The LPI (Landmark Pathfinding between Intersections) algorithm [13] also uses landmarks to precompute heuristics, but solution paths are restricted to follow shortest paths stored between landmarks. Hierarchical Terrain representation for Approximately shortest Paths (HTAP) [14] works by precomputing a hierarchy of abstracted graphs. At each level of the hierarchy a path is found between start and goal and this is used to constrain subsequent higher-resolution searches.

III. RISK BOUNDED SEARCH

The first stage of our risk-bounded search strategy is to obtain an input graph where the arc costs are probability distributions. The nature of the problem domain is such that negative arc costs are not permitted. The reason for this is two fold - with negative edge costs the optimality guarantees of A^* and its variants are voided. In addition, we are making the assumption that it will always cost something for a robot to traverse terrain.

We assume we have a set of arc costs obtained using a the approach in [1], where the cost of a grid cell is obtained by multiplying the probability of its class membership with the probability density associated with the cost of that class in that particular area. We then use ALT to precompute probabilistic heuristics. Consider the landmark L in Figure 2, if $d(\cdot)$ defines the distance *to* L, then by the triangle inequality:

$$d(u) - d(v) \leq dist(u, v) \tag{1}$$

else if $d(\cdot)$ defines the distance *from* L, we have

$$d(v) - d(u) \leq dist(v, u). \tag{2}$$

The largest lower bound over all landmarks is used to select

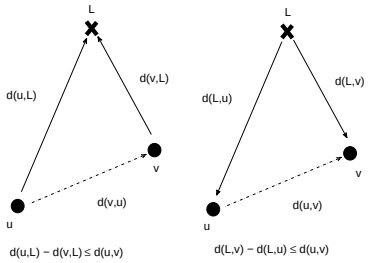

Fig. 2: Landmarks: The triangle inequality provides a lower bound on the distances between nodes in the graph, the maximum of these distances over all available landmarks is used to provide a feasible lower bound for any location in the graph.

the heuristic which will guide the A^* search. The mean of the Gaussian distributions is used in conjunction with Dijkstra's search to precompute distances to and from every point in the map to the selection of landmarks. The variances over the path so calculated are also summed. Neighbouring cell costs are added together leading to a cost over the entire path from node n to the Landmark (L_k) of K landmarks:

$$\hat{L}_{k\in K}(n) = \mathcal{N}\left(\sum_{i\in path} u_i, \sum_{i\in path} \sigma_i^2\right) \tag{3}$$

We use this estimate of the distance between node n and the landmark k in conjunction with equations (1) and (2) to estimate the distance from the node to the goal. The 'most accurate' lower bound is provided by taking the maximum heuristic estimate over the K landmarks we are evaluating.

$$\hat{h}_{ALT}(n) = \max_{k\in K}\left(\hat{L_k}(n) - \hat{L_k}(g), \hat{L_k}(g) - \hat{L_k}(n)\right) \tag{4}$$

Note that this distribution is an approximation to the distribution we are actually interested in, over the distance between the node and the goal. As Figure 3 illustrates, depending on where the landmark is located in relation to the node and the goal influences how accurate the landmark's estimation of the goal distance is.

We account for this by scaling the variance of the heuristic estimate by a factor φ, which is dependent on the ratio of the lengths of the sides of the triangle used to compute the estimate. This ratio is passed through a sigmoidal function

$$c = \frac{\min\left[d(u), d(v)\right]}{\max\left[d(u), d(v)\right]} \tag{5}$$

$$\varphi = \frac{1}{1 + e^{-10*(c-0.5)}} + 1 \tag{6}$$

which has the effect of scaling out the variance by a factor of between 1 and 2. For the examples in Figure 3 the variance of the Landmark 3 (LM3) case, which produces a perfect estimate of the distance, would be inflated by 1.25. The Landmark 1 case which grossly underestimates the distance between the two points would be inflated by 2.0.

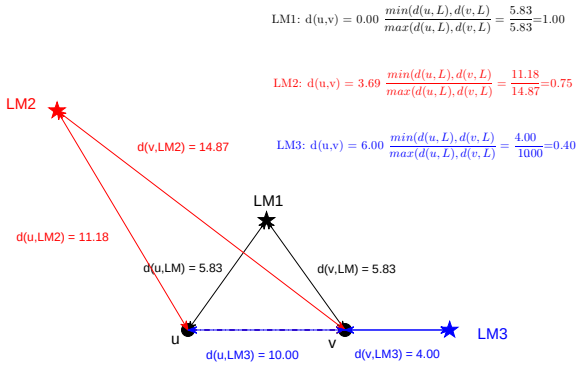

Fig. 3: Three different landmarks produce widely varying estimates for the true distance (6.00) between nodes u and v. While they all produce valid lower bounds, the correct distance is provided only by landmark $LM3$ which correspondingly exhibits the smallest ratio of min to max distances between u, v and the landmark. The problem of the accuracy of the lower bound being dictated by the relative position of the node, goal and landmark is addressed by pushing this ratio through a sigmoidal function to 'scale up' estimates such as those provided by $LM1$ in this instance, while leaving correct estimates (such as $LM3$) relatively untouched.

In order to guarantee the probabilistic bounds detailed later in this section, we also need to correct for the consistent overestimation of the mean by the ALT heuristic. We introduce small scaling parameters for both the mean (τ) and the variance (incorporated into φ) and learn these parameters by conducting multiple searches and comparing the results to optimal paths found by A* search using an admissible (euclidean) heuristic.

$$\hat{h} = \mathcal{N}(\mu_{\hat{h}_{ALT}} - \tau, \varphi(\sigma^2_{\hat{h}_{ALT}})) \tag{7}$$

Learning is done once per environment. In our experience the searches converge to a 'good' estimate within approximately 15 searches.

In analysing the risk we adapt the analysis provided in [5]. If we take $h(n)$ to be the minimum arc cost from node n to the goal, we know that equation 7 has provided for us an estimate of the minimum arc cost in the form of a probability density function $\hat{h}(n)$. We can choose to interpret this as the likelihood of h given our precomputed estimates $\hat{h}$, and denote the likelihood by $p(h \mid \hat{h})$. In the course of carrying out an $A*$ search we observe $\hat{g}(n)$, which is the best known approximation to $g(n)$ - the minimum cost of navigating from the start s to node n. Note that this is a scalar value. Knowledge of $\hat{g}$ induces a conditional density function on $f^\dagger(n)$, the cost of a path from the start to goal via n.

$$f^\dagger(n) = \hat{g}(n) + h(n) \tag{8}$$

This leads to a probability distribution over $f^\dagger$:

$$p(f^\dagger \mid \hat{g}, \hat{h}) = \hat{g} + p(h \mid \hat{h}) \tag{9}$$

Every optimal solution path must be a continuation of a path $T(n)$ which passes through some node currently contained on the OPEN list. Therefore we can say that OPEN always contains a/some node/s for which $f^\dagger(n) = f(n) = C_{opt}$, where C_{opt} denotes the optimal shortest path length.

Termination conditions for a normal, scalar-driven A^* search are straightforward. The search continues until the

goal node has the lowest f-value of any node in the OPEN list. Now that our OPEN list is ordered by probability distributions, the decision as to when to terminate the search becomes more complicated.

Figure 4 illustrates the dilemma. Suppose the best path found so far has a cost C, shown by the straight vertical line on the graph. The node n at the top of the OPEN list has an f-value given by the gaussian distribution shown. Expanding n shows some promise of coming up with a path shorter than C, but on average n would produce a longer path as the mean of the distibution $f^\dagger$ is greater than C. The decision as to whether to explore n or not thus involves evaluating the *risk* of terminating the search at cost C. Under

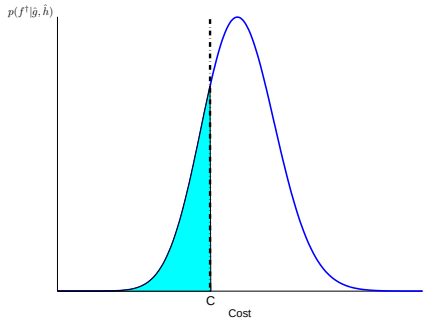

Fig. 4: Under R^*_δ search, if the true cost of node n lies in the shaded region and we terminate the search at Cost C, we risk not finding the shortest path.

the R^*_δ paradigm, the risk of missing futher cost reduction is characterized by a Risk Function $R(C)$ which depends on both C and $p(f^\dagger \mid \hat{g}, \hat{h})$. It is a non-decreasing function of C. We have chosen to investigate two types of Risk Functions:

1) *Probability of Suboptimal Termination*

$$R_{ST}(C) \triangleq p(C - f^\dagger > 0) = \int_{y=-\infty}^{C} p(y \mid \hat{h}, \hat{g}) dy \tag{10}$$

2) *Expected Risk*

$$R_{ER}(C) = \int_{f^\dagger=-\infty}^{C} (C - f^\dagger) p(f^\dagger \mid \hat{h}, \hat{g}) df \tag{11}$$

An R^*_δ search imposes the requirement that the underlying $A*$ search will continue until no node on the OPEN list has a risk associated with it that is greater than some level δ. This introduces the notion of *δ-risk admissibility*, guaranteeing that the search always terminates at a cost C such that $R(C) \leq \delta$ for all nodes left on OPEN.

This means that instead of using f-values to order the OPEN list, we use a *threshold cost function* $C_\delta(n)$ which is given by the solution to the equation

$$R(C) = \delta. \tag{12}$$

When using the *Probability of Suboptimal Termination* risk functional, we choose δ to be the probability of obtaining a suboptimal solution which we are prepared to accept. For instance, if we were prepared to risk obtaining a suboptimal solution 5 out of every 100 iterations of a particular problem instance we would set δ to 0.05. When using the *Expected Risk* functional, δ represents the additional cost of the path over that of the optimal path which we are prepared to accept. As equation (17) will show, it is often preferable to express this as a percentage of optimal path length.

Once we have chosen our risk functional and the level of risk we are prepared to accept, we need to translate this into a *threshold cost function* value $C_\delta(n)$ - that can be computed from $\hat{g}$ and $p(h \mid \hat{h})$ to be used in ordering the OPEN list in our A*-like search.

For the $R_{ST}(C)$ risk functional, it makes sense to use the properties of the Gaussian distribution and write δ as an expression of the distance from the mean.

$$C_\delta(n) = \begin{cases} \mu & \delta = 0.5 \\ \mu - \sigma & \delta = 0.159 \\ \mu - 2 * \sigma & \delta = 0.023 \end{cases} \tag{13}$$

Evaluating the $R_{ER}(C)$ functional for a gaussian leads to the following piecewise-linear approximation:

$$R_{ER}(C) = \begin{cases} 0 & C < \mu - \frac{2\sigma}{\sqrt{2\pi}} \\ \frac{\sigma}{\sqrt{2*pi}} + \frac{1}{2}(C - \mu) & \mu - \frac{2\sigma}{\sqrt{2\pi}} < C < \mu + \frac{2\sigma}{\sqrt{2\pi}} \\ C - \mu & \mu + \frac{2\sigma}{\sqrt{2\pi}} < C \end{cases} \tag{14}$$

which can be solved for values of $C_\delta(n)$

$$C_\delta = \begin{cases} \mu + 2\left(\delta - \frac{\sigma}{\sqrt{2\pi}}\right) & 0 < \delta < \frac{2\sigma}{\sqrt{2\pi}} \\ \mu + \delta & \frac{2\sigma}{\sqrt{2\pi}} < \delta \end{cases} \tag{15}$$

As alluded to earlier it is more useful to express the risk as a percentage of the solution cost

$$\delta = \frac{R(C')}{C'} \tag{16}$$

which leads to the following expression for C'_δ

$$C'_\delta = \begin{cases} \frac{\frac{2\sigma}{\sqrt{2\pi}} - \mu}{(2\delta - 1)} & 0 < \delta < \frac{\frac{2\sigma}{\sqrt{2\pi}} - \mu}{2\left(\frac{2\sigma}{\sqrt{2\pi}} + \mu\right)} + \frac{1}{2} \\ \frac{\mu}{(1-\delta)} & \frac{\frac{2\sigma}{\sqrt{2\pi}} - \mu}{2\left(\frac{2\sigma}{\sqrt{2\pi}} + \mu\right)} + \frac{1}{2} < \delta. \end{cases} \tag{17}$$

IV. Results

We implemented the risk heuristic searching algorithms in C++ and ran them on a standard PC with 4G RAM and a dual core 1.8GHz processor. Landmarks were generated using the *planar landmark selection* method [2] [15]. This method stems from the observation that placing a landmark behind the destination tends to generate good results in geometric graphs.

The graph in Figure 5 compares the results obtained over a 65×65 grid graph with various approximations to the heuristic. Eight landmarks were used in precomputing the search. Each data point in the graph is the average result obtained over repeated searches between the same start and goal node after sampling 50 distinct instances from the probabilistic cost map distribution. The 30 data points represent 30 different start/goal combinations. The three graphs show the results of searching using the mean minus 2 standard deviations (where 97.7% of searches should return the optimal path), the mean minus one standard deviation (84.1% should be optimal) and the mean (50% optimal). The

graphs clearly show that our method of scaling the heuristic produces a distribution nearly identical to the true distance between the node under consideration and the goal - because with the scaled mean and variance we obtain the expected performance in terms of the number of optimal solutions for all three cases.

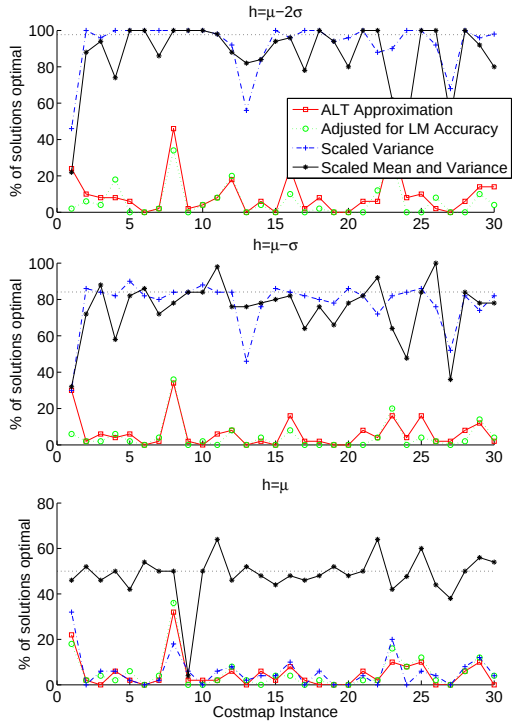

Fig. 5: Approximating the heuristic distribution. The dotted horizontal lines show expected % accuracy for each of the $h = \mu - 2\sigma$, $h = \mu - \sigma$ and $h = \mu$ bounds. The graphs show that scaling the heuristic estimate obtained via ALT produces many-fold improved results over the raw estimate. The bottom graph illustrates the importance of shifting the mean to account for consistent (but slight) overestimation by ALT.

To test the operation of the algorithm itself we generated a 256×256 random fractal terrain map (Figure 6) for both the mean and variance of the cells. This was done to ensure that there would be distinctive areas of high and low variance in the map - rather than randomly scattered throughout - and that this would reflect real world situations where certain regions would be better known than others. We placed 16 landmarks using planar landmark selection and compare the results of our risk based searches against that of A* guided by a euclidean distance heuristic.

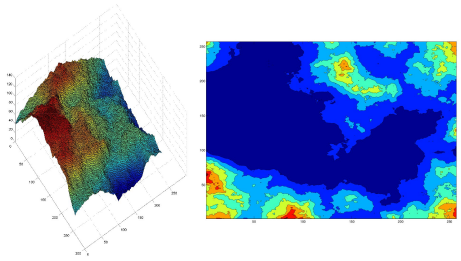

Fig. 6: The mean and variance of the random fractal terrain map used for the experiment

Precomputing the landmarks took an average of 2.58 seconds per landmark - this involved running a Dijkstra search in both forward and reverse directions over the graph, with the edge lengths set to Gaussian values. By way of comparison, A* searches with euclidean distance over the graph took an average of 2.16 seconds. So precomputation took approximately 19 times as long as one search.

We performed 60 different start to goal searches, and at each iteration generated 50 different samples from the probabilistic cost map to test the operation.

Figure 7 shows the results of the R^*_δ search with varying levels of suboptimal risk allowed. Overall, the $\mu - 2\sigma$ heuristic achieved 96% accuracy; $\mu - \sigma$ achieved 81.2% and μ achieved 50.2%, close to the expected values of 97.7%, 84.1% and 50.0% respectively. Figure 8 illustrates the efficiency savings of these heuristic values - on average they require 95.4%, 97.3% and 27.43% of the search effort respectively and produce normalized path lengths of $1.0002, 1.0006$ and 1.002 multiples of that of the A* search. Due to the probabilistic nature of the risk based search, $h = \mu - \sigma$ performs worse than $h = \mu$. However, our results show that using the $h = \mu$ search heuristic guarantees the shortest path distance 50% of the time and expands only 27% of the cells that an equivalent A* search would do - an efficiency saving of 73%.

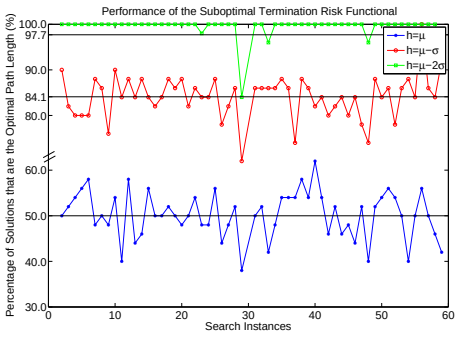

Fig. 7: Performance of R^*_δ with suboptimal termination

Figure 9 shows the results of running 60 iterations of the search with varying levels of expected risk allowed. Here, δ is used to bound the percentage over which the path length returned by R^*_δ exceeds the optimal path length. The graph shows that even with δ set to 40%, on average the path length is only 1.77% in excess of that achieved by A* with an admissible heuristic, and at worst it is 18.5% longer. However it only expands 78.7% of the nodes, an efficiency saving of 22.3% while bounding the length of the resultant path. Table I shows the planning times of the two heuristic families compared with that of A* search with the standard euclidean heuristic. Note that the more complicated nature of our heuristics results in longer planning times for all but the $h = \mu$ suboptimal termination heuristic. However, as the heuristics examined in this paper cause A* search to expand less nodes, we would expect the planning times to decrease relative to that of the euclidean heuristic as the size of the costmap grows.

V. Conclusions

This paper introduced a tradeoff between speed and accuracy for Path Planning over probabilistic costmaps. We see

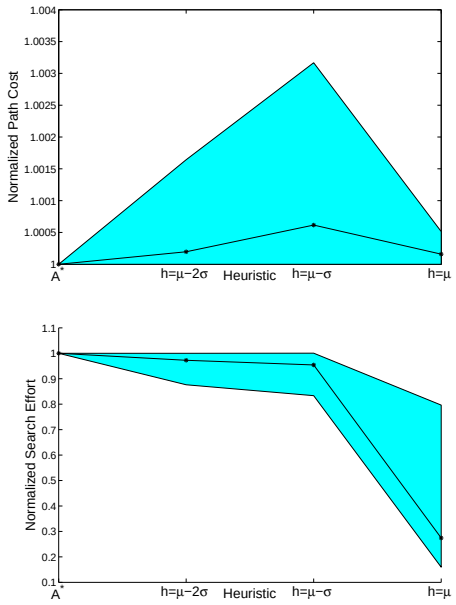

Fig. 8: The results of running R_δ search with three different risk functional levels as per Equation (13). The top plot is the normalized path cost with the solid line representing the mean of the normalized path cost and the filled section denoting the spread of values obtained within one standard deviation of the mean. The bottom plot is the normalized search effort in terms of nodes expanded. The solid line denotes the average search effort, the filled section denotes one standard deviation from the mean search effort.

	Time (s)
Suboptimal Termination	
μ	1.3134
$\mu - \sigma$	3.5597
$\mu - 2\sigma$	3.5530
Expected Termination	
$\delta = 5\%$	3.5955
$\delta = 10\%$	3.5816
$\delta = 20\%$	3.5400
A* search with euclidean heuristic	1.7623

TABLE I: Planning Times of the various heuristics on a standard PC with 4G RAM and dual core 1.8GHz processor

this as being particularly useful in field robotics applications where often a 'good' path is as acceptable as the shortest path, and coarse, uncertain prior data such as aerial maps - from which to precompute a probabilistic heuristic - are freely available. The heuristic is valid for the duration of planning over that particular environment, and allows the user to specify how much risk they wish to take on in terms of the quality of the paths returned. The results show efficiency gains of up to 73%, and that the precomputation effort is minimal especially if multiple traverses will be made over the same terrain.

In future we plan to examine the placement of the landmarks. In this work we used a planar landmark selection method which scatters landmarks around the boundary of the the map. In this approach landmarks simply equated to grid cells located in the map rather than physical objects with salient characteristics. While this worked well in normal scalar grid maps, it would be worthwhile investigating the effect of choice of landmark on the accuracy bounds of the R^*_δ search, i.e. is it better for our purposes to place landmarks in areas of well known terrain?

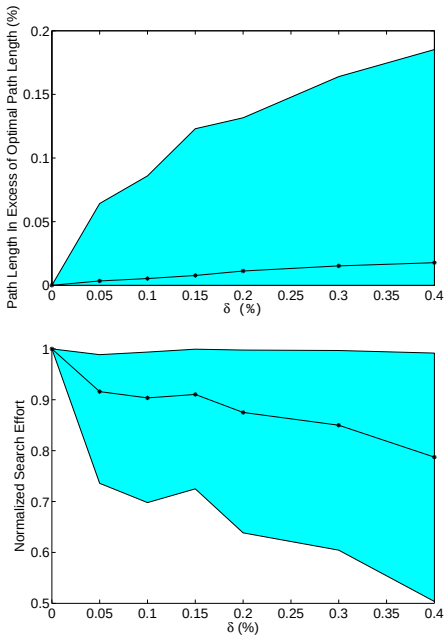

Fig. 9: The results of running R^*_δ with 6 levels of expected risk

REFERENCES

[1] E. Murphy and P. Newman, "Planning paths from overhead imagery," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Anchorage, AK, May 2010.

[2] A. V. Goldberg and C. Harrelson, "Computing the shortest path: A* search meets graph theory," in *16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '05)*, Vancouver, Canada, January 2005.

[3] M. Likhachev, G. Gordon, and S. Thrun, "Ara*: Anytime a* search with provable bounds on sub-optimality," in *Conference on Neural Information Processing Systems (NIPS)*, 2003.

[4] M. Likhachev, D. Ferguson, G. Gordon, A. T. Stentz, and S. Thrun, "Anytime dynamic a*: An anytime, replanning algorithm," in *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, June 2005.

[5] J. Pearl and J. H. Kim, "Studies in semi-admissible heuristics," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, no. 4, pp. 392–399, July 1982.

[6] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, 1984.

[7] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proceedings of the International Conference on Robotics and Automation*, 1994, pp. 3310–3317.

[8] S. Koenig and M. Likhachev, "D*lite," in *Eighteenth national conference on Artificial intelligence*. Menlo Park, CA, USA: American Association for Artificial Intelligence, 2002, pp. 476–483.

[9] D. Ferguson and A. T. Stentz, "The field d* algorithm for improved path planning and replanning in uniform and non-uniform cost environments," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-05-19, July 2005.

[10] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions of Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, July 1968.

[11] L. R. Harris, "The bandwidth heuristic search," in *International Joint Conference on Artificial Intelligence*, Stanford, USA, 1973, pp. 23–29.

[12] A. V. Goldberg, "Point-to-point shortest path algorithms with preprocessing,," in *Current Trends in Theory and Practice of Computer Science (SOFSEM)*, Harrachov, Czech Republic, 2007.

[13] K. Grant and D. Mould, "Lpi: Approximating shortest paths using landmarks," in *ECAI 2008 - Workshop on AI and Games*, Patras, Greece, July 2008.

[14] D. Mould and M. C. Horsch, "An hierarchical terrain representation for approximately shortest paths," in *8th Pacific Rim International Conference on Artificial Intelligence*, Auckland, New Zealand,, August 2004.

[15] A. V. Goldberg and R. F. Werneck. (2009, September) Selecting landmarks in shortest path computations. Patent Application Publication. Microsoft Corporation.