Fast Minimum Uncertainty Search on a Graph Map Representation

Henry Carrillo, Yasir Latif, José Neira and José A. Castellanos

***Abstract*—This paper addresses the problem of path planning considering uncertainty criteria over the belief space. Specifically, we propose a path planning algorithm that uses a novel determinant-based measure of uncertainty and a reduced representation of the environment, in order to obtain the minimum uncertainty path from a roadmap. Our proposal does not require *a priori* knowledge of the environment due to the construction of the roadmap via a graph-based SLAM algorithm. We report experimental results of our proposal in four datasets that show its feasibility to obtain the minimum uncertainty path towards an autonomous navigation framework and we also show an improvement in the computation time with respect to the state of the art.**

I. Introduction

Path planning solely in the $\mathcal{C}_{\text{free}}$ (*i.e.* only taking into account geometric constrains) does not guarantee the safety of a robot navigating over those paths [1]–[3]. The above problem stems from the uncertainty generated by the inherent noise in the localization and control systems of a robot working in a real environment, which is the result of the imperfect data it gathers. Moreover, a robot navigates an environment in order to fulfil a task, and an initial condition to effectively complete any task is to accurately reach *a priori* initial position established in the workspace where this task will be performed. For example, a mobile manipulator aiming at opening a door using visual servoing, needs to position its manipulator within the range of the doorknobs.

In order to overcome the above problem, several works, such as Mihaylova et al. [4], Gonzalez et al. [5] and Lambert et al. [3] have proposed the integration of the uncertainty in the path planning process. Recently Prentice and Roy [6] have proposed to plan over the so-called belief space. The use of the belief space in the proposal of different path planners such as He et al. [7], Prentice et al. [8] and Valencia et al. [9] have proved experimentally that taking into account the uncertainty in the planning process leads to an accurate and safe navigation process.

All the aforementioned path planners, which use the belief space, rely on metrics or criteria that measure or quantify the uncertainty or its dual, the information of certain configuration in the space. Among the most used metrics are: trace of the covariance matrix, entropy and mutual information [10]–[12]. In this paper, we devise a path planner that relies on a novel determinant-based metric to take into account the effects of uncertainty and a novel reduced graph representation that speeds up the search process. As a result, our planner can be seamlessly integrated with recently proposed graph-based SLAM algorithms such as iSAM [13], FrameSLAM [14], or RSLAM [15].

The reminder of the paper is structured as follows: Section II presents a brief overview of the uncertainty metrics commonly utilized in path planners that use the belief space and presents in more detail the metric used in our approach. In section III, we define the problem we are dealing with: "how to plan the minimum uncertainty path in a roadmap like structure?". Section IV reports our path planner that uses a novel graph-based reduced representation of the environment and a determinant-based criterion to quantify the uncertainty, and is designed to seamlessly integrate with a graph-based SLAM algorithm. Section V presents experimental trials of our approach in real datasets. Moreover, we compare our method against the state-of-the-art and show an improvement in the computation time. Finally, section VI presents some conclusions and possible future work.

II. Uncertainty Measures

Historically, the uncertainty metrics were first proposed in the Theory of Optimal Experiment Design (TOED) [16] [17] context and were named like an alphabet with the suffix optimality attached to them to denote the origin. These metrics or criteria coming from the TOED aim at capturing the idea of whether or not the uncertainty represented by a covariance matrix, $\mathbf{\Sigma}$, is large or small.

Formally, an uncertainty criterion has to define a function ϕ that maps a covariance matrix of size $l \times l$ to a scalar,

$$\phi : \mathbf{\Sigma} \rightarrow \mathbb{R} \tag{1}$$

This function has to be positive homogeneous, isotonic (*i.e.* order preserving) and concave [17].

A compendium of functions fulfilling the above requirements can be found in [16] or [17]. Among the most commonly used functions or uncertainty criteria, for a covariance matrix $\mathbf{\Sigma}$ with size $l \times l$ and eigenvalues λ_l, are:

- A-optimality criterion (*A-opt*) [18]: This criterion targets the minimization of the variance and it is defined as

This work has been supported by the MICINN-FEDER projects DPI2009-13710 and DPI2009-07130, and research grants BES-2010-033116 and BES-2010-036653, also by DGA-FSE (grupo T04).

H. Carrillo, Y. Latif, J. Neira and J. A. Castellanos are with the *Departamento de Informática e Ingeniería de Sistemas, Instituto de Investigación en Ingeniería de Aragón, Universidad de Zaragoza*, C/ María de Luna 1, 50018, Zaragoza, Spain. {hcarri, ylatif, jneira, jacaste}@unizar.es

TABLE I
SUMMARY OF UNCERTAINTY METRICS

Metric	Formulation
A-opt	$\text{trace}(\mathbf{\Sigma}) = \sum_{k=1}^{l} \lambda_k$
D-opt	$\exp\left(l^{-1} \sum_{k=1}^{l} \log(\lambda_k)\right)$
E-opt	$max(\lambda_k)$

follows,

$$\text{trace}(\mathbf{\Sigma}) = \sum_{k=1}^{l} \lambda_k \tag{2}$$

- D-optimality criterion (*D-opt*) [19]: This criterion aims at capturing the full dimension of the covariance matrix and at first glance it can be defined as

$$\det(\mathbf{\Sigma}) = \prod_{k=1}^{l} \lambda_k \tag{3}$$

- E-optimality criterion (*E-opt*) [20]: This criterion intends to minimize the maximum eigenvalue of the covariance matrix $\mathbf{\Sigma}$. The main advantage of this criterion is the simplicity of its computation, but it is a rough approximation of the error ellipsoid.

According to the TOED [16] [17] *D-opt* gives the most accurate approximation of the uncertainty enclosed in the covariance matrix, but in the context of active SLAM or planning under uncertainty [10]–[12] and [21], have shown that using the definition in (3) to compute the *D-opt* does not produce a meaningful metric (*i.e.* the value gets stuck at zero).

In [22] a novel computation form of the uncertainty criterion based on the determinant of the covariance matrix is presented. There the *D-opt* is computed as follows:

$$\exp\left(l^{-1} \sum_{k=1}^{l} \log(\lambda_k)\right) \tag{4}$$

that stems from the family of uncertainty criteria proposed by Kiefer in [23],

$$\phi_p(\xi) = [l^{-1}\text{trace}(\mathbf{\Sigma}^p(\xi))]^{1/p} \tag{5}$$

This family of uncertainty criteria is valid in the range of $0 < p < \infty$ for a covariance matrix ($\mathbf{\Sigma}$) of size $l \times l$ associated to a design ξ. Moreover, the case ϕ_1 and the boundary cases ϕ_0 and ϕ_∞ are the already known A, D and E-optimality criteria.

Using (4) instead of (3) in order to compute the *D-opt* in the planning under uncertainty or active SLAM context has the advantage of producing a meaningful uncertainty metric *i.e.* it does not get stuck at zero and evolves resembling the uncertainty encompassed in the covariance matrix. Furthermore, according to the TOED *D-opt* is the uncertainty criterion that by definition truly captures the complete dimension of the uncertainty, unlike the *A-opt* and *E-opt* that are approximations [16] [17] [24]. A summary of the uncertainty metrics is presented in Table I.

III. PATH PLANNING IN THE BELIEF SPACE

Assuming that the data structure representing the environment (*i.e.* a map) is a graph-like structure (*i.e.* a metric-topological representation), we could use the well-known Probabilistic Roadmaps (PRM) algorithm to generate a discrete graph in $\mathcal{C}_{\text{free}}$ (*i.e.* the set of configuration at which the robot does not intersect any obstacle [25]). Given a start configuration $\mathbf{X_{Start}}$ of a robot and a goal configuration $\mathbf{X_{Goal}}$, within the above discrete graph, we are aiming at finding the minimum uncertainty path between them.

Planning for the minimum uncertainty path cannot be done in the $\mathcal{C}_{\text{free}}$, because it cannot guarantee the safety we are aiming for. Therefore, it seems more natural to use another space such as the belief ($\mathcal{B}$) or information ($\mathcal{I}$) space [6] [8] [2]. Autonomous path planning in the belief or information space, as any autonomous path planner, relies heavily on metrics that quantify the cost of moving from one configuration to another. In the case of the belief space, the most common used metrics are the ones based on TOED (uncertainty) and on information theory. In both cases, and unlike the metrics used in the configuration space, the evolution of the uncertainty or information metrics are non-monotonic, and so far there is no optimistic heuristic that allows the use of the plethora of well-know and effective non-uniform path planner such as: A* [26] or D* [27] [28].

There is no doubt that imposing a discretization of the environment with the PRM algorithm, prevents any discrete search algorithm from guaranteeing that it will find the minimum uncertainty path. However, within a discrete graph built upon a tractable computing premise according to the PRM algorithm, is possible to find the minimum uncertainty path.

Another issue with the PRM algorithm is its requirement of *a priori* knowledge of the environment to produce the roadmap. This constraint limits the feasibility of the integration of the path planner in an autonomous robot framework. An approach based on a SLAM algorithm can produce a roadmap of the environment and overcomes the aforementioned issue. Specifically, we can use a graph-based SLAM algorithm such as: iSAM [13], FrameSLAM [14], or RSLAM [15], that does not need beforehand knowledge of the environment and can produce a good estimate of the environment enclosed in a pose/feature graph.

It is worth mentioning here that although the robot starts to plan assuming that the environment is static, this assumption will not hold as it executes several steps of the planned path. This happens when a previously traversable path is no longer available or new information such as loop closures update the belief of the robot about the map topology. Therefore planning algorithms should be able to perform fast replanning both to deal with changes in the environment as well as the changes in the belief of the robot, whenever the need arises.

Algorithm 1 FaMUS algorithm

Require:
- A pose graph map of the environment $\mathbf{G}$
- A initial pose n_s and a goal pose n_g.

Ensure:
- The path with the minimum accumulated uncertainty cost from the pose n_s to the pose n_g.

1: $\forall n_i \in \mathbf{G}$: calculate *D-opt*
2: $\forall n_i \in \mathbf{G}$: find reachable neighbours and add edges
3: $\mathbf{G_d} \leftarrow$ ReduceGraph $(\mathbf{G})$
4: $minPath_d \leftarrow$ DijsktraSearch(n_s,n_g,$\mathbf{G_d}$)
5: **return** ReconstructPath($minPath_d$,$\mathbf{G_d}$)

Algorithm 2 The graph reduction process: ReduceGraph(n_s,n_g,**G**)

Require:
- The original graph $\mathbf{G}$ with the associated cost for each node
- A initial pose n_s and a goal pose n_g.

Ensure:
- The decision graph $\mathbf{G_d}$

1: **for** $n_i \in \mathbf{G}$ **do**
2: **if** $\|n_i\| = 2$ **then**
3: $start \leftarrow i$
4: $cost \leftarrow 0$
5: **while** $\|n_i\| = 2, n_i \neq n_s, n_i \neq n_g$ **do**
6: $cost \leftarrow cost + f(n_i)$
7: $i \leftarrow i + 1$
8: **end while**
9: AddEdge(start-1,i,cost,**G**)
10: RemoveVertices(start $\rightarrow$ i-1, **G**)
11: **end if**
12: **end for**
13: **return G**

IV. Our approach

In this section, we present **FaMUS** (**Fa**st **M**inimum **U**ncertainty **S**earch): an algorithm capable of planning the minimum uncertainty path using a pose graph formulation of the SLAM problem. Similar approaches have been previously applied in a SLAM context by He et al. [7] using a cost function based on the trace of the covariance matrix and by Valencia et al. [9] using an entropy based cost function. Our proposal, in contrast, uses a cost function based on the *D-opt* over a reduced graph for fast minimum uncertainty path planning.

Algorithm 1 shows the pseudo-code for the FaMUS algorithm. It requires a pose graph map (*e.g.* produced by a graph based SLAM algorithms), a start ($\mathbf{X_{Start}}$) and goal ($\mathbf{X_{Goal}}$) pose within the graph. It ensures the path with minimum uncertainty from $\mathbf{X_{Start}}$ to $\mathbf{X_{Goal}}$ according to the *D-opt* (*c.f.* section II and (4)).

In a pose graph formulation of the SLAM problem, each robot position is represented as a vertex in a graph and the relationships between poses form the edges. Landmarks are used to generate the relationship between consecutive poses and as a result the graph at the end contains only the robot trajectory. Due to the nature of the SLAM problem, the graph has sparse connectivity since most of the constraints are sequential *i.e.* they link two consecutive poses in time. There are also non consecutive constraints that arise from loop closures when the robot revisits an already explored area. These constraints reduce the overall uncertainty in the trajectory. We argue that the pose graph formulation is good for path planning because it already provides information about potentially traversable paths since the robot has already been through this trajectory during mapping. The pose graph SLAM is inherently sparse because there are a lot more consecutive constraints than loop closures. We aim to exploit this sparsity in order to achieve speed-up in path planning.

A. Metric Calculation

For every vertex in the graph, we compute the marginalized covariance and calculate the metric. In a pose graph formulation, the marginal uncertainty of a node is a fusion of the uncertainty for all possible paths from the origin of the map [9]. In our approach, we only take the marginal covariance of the pose into account when calculating the *D-opt* criterion. For a more general case of pose-feature graph, a similar approach can be taken by taking into account the visible landmarks from the vertex under consideration.

For the marginalized covariance, we calculate the Eigenvalues and compute the metric given in (4). The cost of travelling from a vertex V_i to a vertex V_j is equal to the value of the metric calculated at V_j.

B. Increasing traversability

Inherently, the pose graph formulation results in a graph that is sparsely connected where different trajectories are only linked if a loop closing constraint is established between them. It might happen that even though two trajectories lie very close to each other, there are no edges between the corresponding vertices because the place recognition system was unable to close the loops. If such vertices lie within some distance of the current robot location, then the node is considered reachable from the current node and an edge is added to it. It should be noted that this threshold is very restrictive and is set smaller than the robot's diameter. Since a pose graph does not contain any information about the structure of the environment, it is possible that, for example, the two trajectories lie on the opposite side of a wall and hence are not reachable from each other. Therefore, we set this threshold to be very conservative in order to add edges that are already within the reach of the robot. We use a fast KD-tree based nearest neighbour search to find possible neighbours for every node in the graph.

C. Decision Points

Vertices that are only connected by sequential constraints have a degree two *i.e.* $\|V_i\| = 2$; one connecting it to the

previous pose in time, and one connecting it to the next. If a robot arrives at such a vertex from one direction, it can only move in the other direction. At these vertices, the robot has no choice but to continue travelling till it reaches a vertex with a degree greater than two. At that point, the robot can take one of the $n-1$ paths, where n is the degree of the vertex. We call such vertices "decision points", since the robot has to make a decision about which path to take. Formally, a decision point can be defined as

$$\exists V_i \in \mathbf{G} : \|V_i\| > 2 \tag{6}$$

Poses that participate in a loop-closure constraint form the decision points because they connect to more than two vertices. Between two decision points, the robot must follow the trajectory connecting them. The given start and end vertex in the graph are also considered decision points.

D. Decision Graph

The complexity of search algorithms such as depth-first or breadth-first depends on the number of vertices and number of edges in the graph being searched. While efficient implementations exist that exploit GPUs or multi-threads, we propose a method for reducing the number of vertices and edges in the graph in order to sequentially search the decision graph faster.

As discussed earlier, between two decision points the robot has to follow the sequential links. Therefore, we can connect the two decision points using new edges that encode the cost of the underlying sequential links, and remove the sequential links. The cost of travelling the same vertices whether encoded by the actual number of links or just a single link is therefore the same. We call this graph consisting of only decisions points as "Decision Graph". An example can be seen in Fig. 1(d-f). In (d) the reduction can be seen very clearly because most of the map consists of long corridors without any loop closures. All such corridors are reduced to single edges which connect decision points that are present at the end of the corridors. (The corners of the corridors becomes decision points because the robot turns slowly, generating a lot of poses at the same place which are then linked together by the neighbourhood search (*c.f.* Section IV-B)). Another example is in fig. 1-(f) where the arrow points to two trajectories that are reduced to single edges.

In order to have an insight into how the decision graph is constructed, let us consider a sequence of vertices $D_i[O_1 \dots O_n]D_{i+1}$, where there are n sequential links between the two decision points. Let the cost of travelling from D_i to O_1 be $c_{i,1}$ and that of travelling from O_n to D_{i+1} be $c_{n,i+1}$. We can calculate the cost $c_{1,n}$ as

$$c_{1,n} = \sum_{i=1}^{n} f(O_i) \tag{7}$$

where $f(\cdot)$ is the metric under consideration. This gives us the accumulated cost of travelling the sequential links. The cost of travelling from D_i to D_{i+1} is then calculated as $c_{i,i+1} = c_{1,n} + c_{n,i+1}$ (since $c_{i,1} = f(O_1)$). We can therefore add a new edge with this weight between the decision points and effectively eliminate n vertices and the corresponding edges.

We do this for all the decision points in the graph and as a result we get the decision graph, which contains only the decision points and the associated costs of travelling between them. We argue that the two representation are equivalent and there is no loss of information when constructing the decision graph from the original graph, therefore they are provably equivalent when used as input for path planning over belief space.

E. Searching over the decision graph

Given a start and a goal vertex, we can search over the decision graph to find the minimum uncertainty path. For the search, we use the classic Dijsktra algorithm. When the decision graph is constructed, we keep the start and the goal vertices as decision points as well.

The overall algorithm is presented in algorithm 1 and the method used for constructing the decision graph is given in algorithm 2. Since the search is carried out over the decision graph, the resulting path only contains the decision points (Algorithm 1, Line 4). For actually carrying out the motion, the robot needs to know which vertices to follow in order to travel between decision points. Therefore, we reconstruct the path in the original graph by adding the segments between decision point that the robot must traverse to reach one decision point from another *i.e.* line 5 of algorithm 1. This can be done efficiently by maintaining a lookup table for each pair of decision points that have a common edge in the decision graph.

V. Experiments

In this section we present the methodology and the results for a set of experiments carried out in order to validate and evaluate our minimum uncertainty path planning approach. We use for the experimentation, simulated and real data, from indoor and outdoor environments. Specifically for the real data, we use the Bicocca [29], New college [30], and Intel datasets and for the simulated counterpart, we use the Manhattan dataset. The Manhattan and Intel datasets are available with the downloaded version of g2o [31].

To obtain the graph based map needed by our approach, the FaMUS algorithm, we use the g2o graph optimizer [31] with the Gauss-Newton method and one iteration. We use g2o to efficiently calculate the marginal covariances needed to calculate the *D-opt* criterion. Fig. 1 shows the optimized map for the Bicocca, Intel and New college dataset used in the experiments.

In the remainder of this section, we show the result of the proposed algorithm using the above simulated and real datasets. Specifically, we will show:

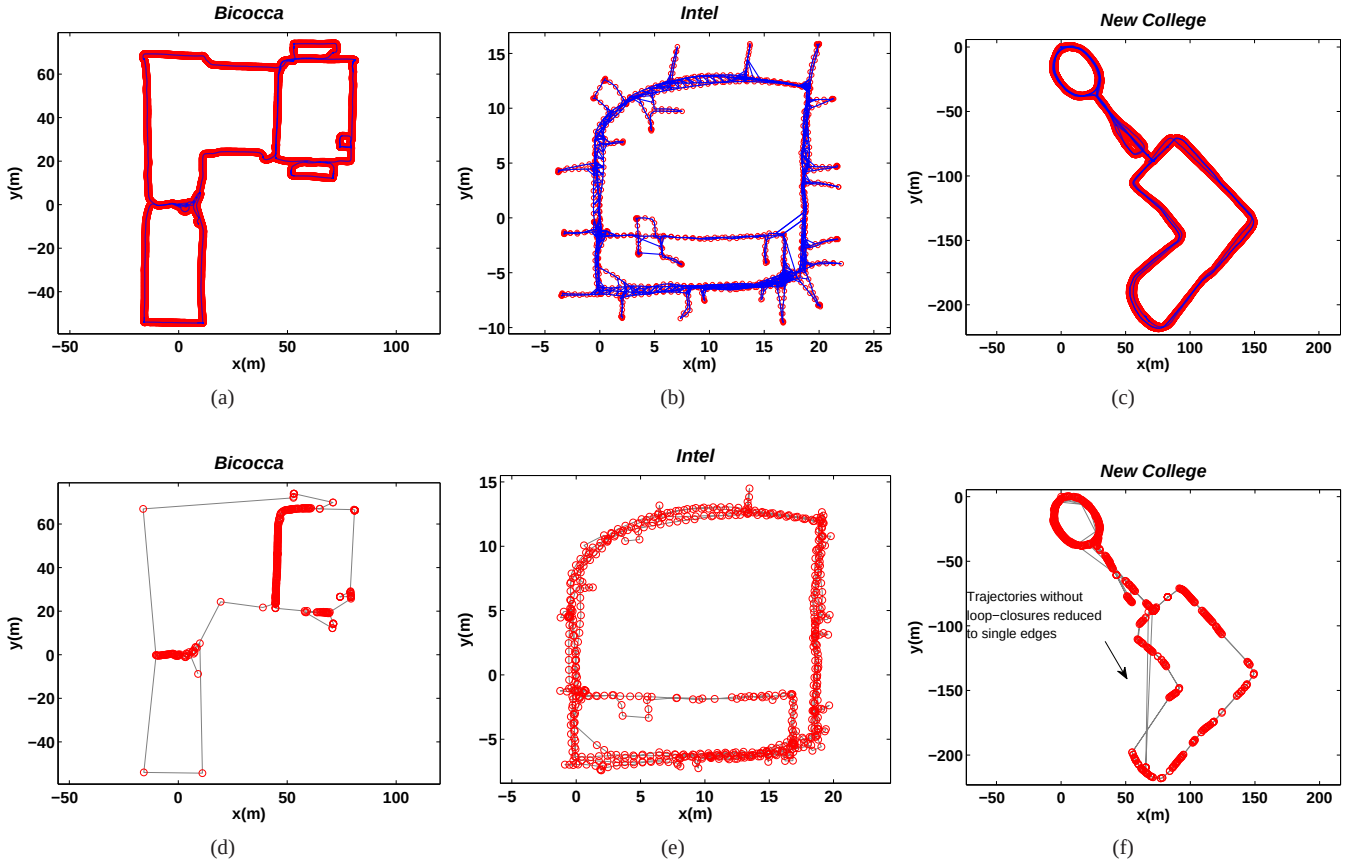


Fig. 1. (a)-(c) Graph map with full vertices (red) and edges (blue). (d)-(f) Graph map with reduced vertices (red) and edges (light grey). Figure best viewed in color.

- the ratio of reduction in the number of vertices and edges in the graph for each dataset due to the computation of the decision graph
- a MonteCarlo simulation that validates the resultant minimum uncertainty path of the FaMUS algorithm.
- a comparison between the accumulated uncertainty in the resulting path of the FaMUS algorithm and the shortest path algorithm.
- timing comparisons with previous approaches.

Finally, we also investigate experimentally how many times the minimum uncertainty path is different form the shortest path and on average, what is the overlap between them.

A. Graph reduction

A key aspect of our algorithm is the reduction of the vertices and edges in the graph to a provable equal representation used in order to increase the speed of searching the minimum uncertainty path.

For the used datasets, table II presents the percentual reduction in the number of vertices and edges in the input graph based map needed by the FaMUS algorithm.

The reduction achieved allows us to reduce the effective complexity of our algorithm in running time and hence it permits a small execution time, as is shown in section V-C.

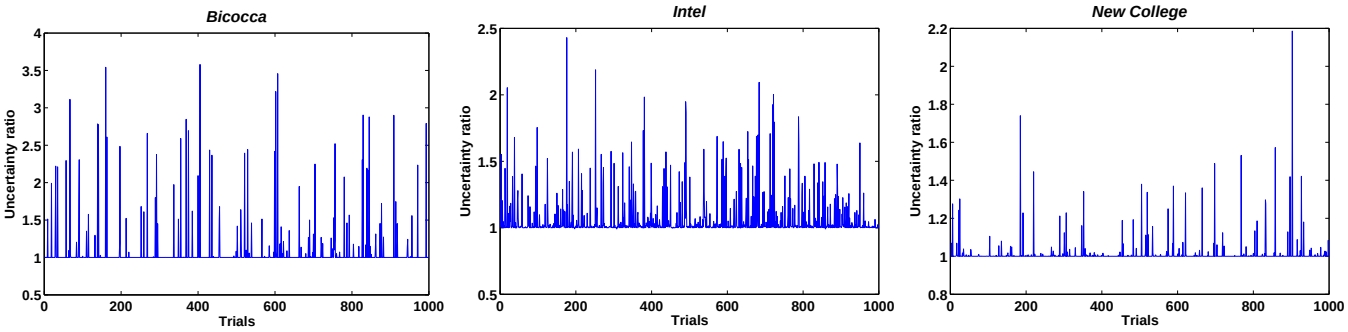
TABLE II
PERCENTAGE REDUCTION OF VERTICES AND EDGES BY THE FAMUS ALGORITHM

Dataset (name)	Vertices (Full/Redu.)	Edges (Full/Redu.)	Vertices Reduction	Edges Reduction
Bicocca	8358/980	8513/6936	88.27%	18.52%
Intel	943/623	1837/1527	33.93%	16.87%
Manhattan	3500/2469	5598/4863	29.45%	13.12%
New College	12816/1055	13171/2624	91.76%	80.07%

B. H_0: Are the minimum uncertainty path and the shortest necessarily equal

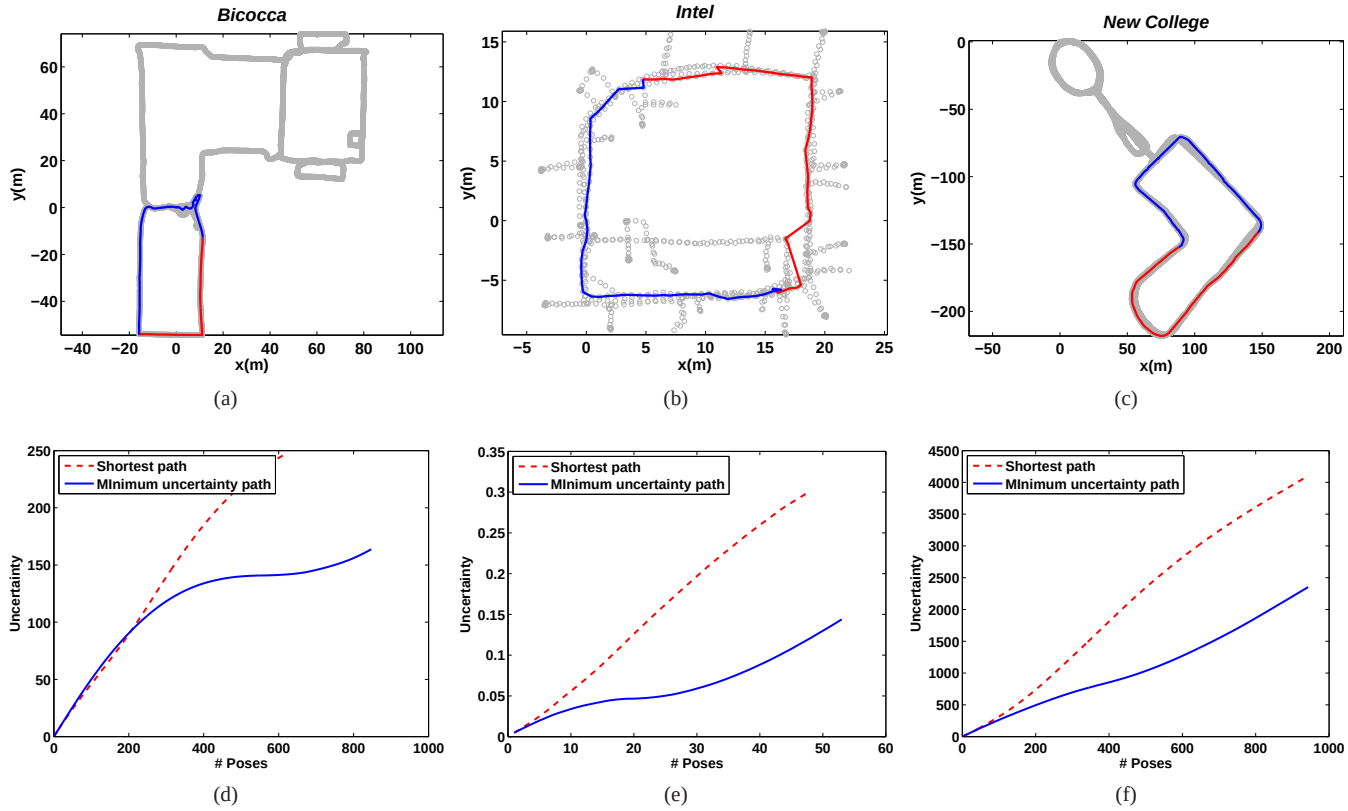
One of the main reasons to plan in the belief space is that the shortest path may be unsafe regarding the uncertainty, in the task of planning from point A to point B. Here we put to test this hypothesis by comparing the final accumulated uncertainty at the end point after following the shortest path and the minimum uncertainty path generated by the FaMUS algorithm.

We ran the above experiment 1000 times in each dataset, therefore obtaining 4000 trials in total. We select randomly each time the start and goal points with a separation of X meters in the euclidean sense. To make the results of the



(a) (b) (c)

Fig. 2. (a)-(c) Uncertantinty ratio of the accumulated uncertainty for 1000 trials. In each trial the start and goal node are selected at random. The accumulated uncertainty is measured at the goal vertex of the shortest path and the path generated by the FaMUS algorithm. A ratio greater that one means lower accumulated uncertainty in the path generated by the FaMUS algorithm.



(a) (b) (c)

(d) (e) (f)

Fig. 3. (a)-(c) Shortest path (red) and minimum uncertainty path (blue) generated by the FaMUS algorithm.(d)-(f) The corresponding accumulated uncertainty for the above paths. Figure best viewed in color.

experiment visually interpretable and to avoid misleading statistics, we show (Fig. 2) the ratio between the uncertainties of the paths generated by the FaMUS algorithm and the shortest path. Also in table III we count how many times the path are the same, completely different and how many nodes are overlapped on average in each dataset.

Fig. 3 shows the resulting paths for some cases in the above experiment. It is worth mentioning here that the FaMUS algorithm follows the classical active SLAM behaviour demonstrated in the literature [32] *i.e.* it prefers paths that have loop-closings in order to exploit previously visited places to keep the uncertainty low, even though they may result in longer path lengths.

Finally, it is worth mentioning that as expected, all the values in Fig. 2 are above one (*i.e.* we are actually obtaining the minimum uncertainty path).

C. Timing comparisons

Table IV summarizes the average computation time of 1000 experiments for each dataset. The experiment carried

TABLE III
PATHS GENERATED BY THE FAMUS ALGORITHM VS THE SHORTEST PATH

Dataset	= paths	!= paths	% overlap
Bicocca	261	51	87.35%
Intel	170	74	62.59%
Manhattan	146	37	70.22%
New College	215	21	87.79%

TABLE IV
TIMING PERFORMANCE OF THE FAMUS ALGORITHM

Dataset	# Vertices	# Edges	Time (ms)
Bicocca	8358	8513	1397.2
Intel	943	1837	215.180
Manhattan	3500	5598	839.97
New College	12816	13171	2143.2

out is described in section V-B. The FaMUS algorithm outperforms the state of the art [9] in this regard. *e.g.* for a graph with 13171 vertices and 12816 edges we calculate the minimum uncertainty path it in 2.14 seconds on average, for approximately 700 vertices [9] does it in 5 seconds. *i.e.* for 94.68% more vertices the FaMUS algorithm does it in 57.13% less time.

The timing given here were obtained using a research implementation of the FaMUS algorithm in C++ on a 2.8 GHz core 2 Duo Intel processor with 8 GB of memory under Ubuntu 10.04 LTS.

Finally, it should be pointed out that for future comparisons, public dataset and not *ad-hoc* examples should be used for timing comparison. This is because timing performance evaluation of this type of algorithms depends on the topology of the graph (edges and vertices), thus a fixed, open and reproducible testbed is desirable.

VI. DISCUSSION

In this paper, we have proposed a fast path planning algorithm capable of obtaining the minimum uncertainty path according to a reduced representation of the environment using a determinant-based criterion. In the literature, to the best knowledge of the authors of this article, this is the first use of the determinant-based criterion to quantify the uncertainty of the robot and environment in a path planning under uncertainty context, therefore accurately capturing the complete dimension of the uncertainty according to the TOED [16] [17].

Since the complexity of search algorithms is dependant on the number of vertices and edges in the graph, a significant speed up can be achieved by reducing the size of the graph. In this work, we have proposed a novel representation termed "decision graph" which enables searching over only those vertices in the graph where the robot can make a decision between at least two paths. This representation is provably equivalent to searching over the whole graph in the sense that both produce the same minimum uncertainty paths. This means that although the decision graph has fewer vertices and edges than the original graph, there is no loss of information when constructing the reduced representation.

The FaMUS algorithm proposed here generates the minimum uncertainty path given a pose graph of the environment. If the environment is static, as is assumed, the robot will be able to reach the goal and have the minimum uncertainty at the goal node. But this is a very strong assumption because real world environments are highly dynamic. As the robot moves through the planned path, it gathers new information that may alter the belief of the robot and may lead to a new path with lower uncertainty. Therefore, the robot should be able to carry out fast replanning in order to deal with the dynamic nature of the world. Our method, while generating the minimum uncertainty path under the *D-opt* criterion, is fast enough to replan in real time.

The proposed algorithm produces, via an exhaustive search, the minimum uncertainty path from an initial configuration of the robot until the goal configuration. We use an exhaustive search because the minimum uncertainty path in the belief space cannot be guaranteed in a greedy search procedure due to the non-monotonic evolution of the uncertainty. Moreover, the non-monotonic evolution prevents the direct use of well known non-uniform path planning algorithm such as A* or D*. The proposal of optimistic heuristics of the uncertainty should be the focus of future work in the path planning under uncertainty research if we desire to use the incremental, computationally tractable yet complete path planning solutions of [27], [28] and [33], among others.

Comparing our algorithm with respect to the state of the art, *i.e.* Valencia et al. [9], our algorithm proposal differs from their work in two distinctive regards, namely the size of the processed graph and the generality of the input graph map representation. Concerning the first one, we devise a decision graph which is equivalent in terms of response to the full input graph but less costly, computationally speaking, to evaluate. The work of Valencia et al. [9] does not perform this or any related step. Comparing the rest of the algorithm *i.e.* the graph search procedure, our work and theirs rely on the same algorithm techniques (*e.g.* Dijsktra).

Apropos the second difference, our approach is more general with regards to the range of SLAM algorithms it can operate on, the only restriction being that we have access to the covariance at each node (as well as landmark if the algorithm maintains it). We do not limit the graph input to a particular class such as the produces by PoseSLAM as Valencia et al. [9]. In this regard, we also see as strength the ability of our algorithm to use closed form equations to compute the uncertainty from the covariance matrix without imposing any distribution to the errors, in contrast to Valencia et al. that use a Gaussian distribution.

To conclude, we have presented a path planning algorithm in the belief space that generates the minimum uncertainty path according to the *D-opt* criterion by speeding up the

search using a reduced graph representation of the environment termed "decision graph" which exploits the sparsity inherent in the pose graph SLAM formulation. As future work, we are aiming at developing an active selection procedure of the nodes of the reduced graph in order to further reduce the uncertainty in the navigation. Also, we aim at performing field test, in heterogeneous environment, of the proposed algorithm.

REFERENCES

[1] J.-P. Laumond, *Robot Motion Planning and Control*. Berlin: Springer-Verlag, 1998, available online at http://www.laas.fr/∼jpl/book.html.

[2] J. Barraquand and P. Ferbach, "Motion Planning with Uncertainty: The Information Space Approach," in *Proceedings IEEE International Conference on Robotics & Automation*, 1995, pp. 1341–1348.

[3] A. Lambert and D. Gruyer, "Safe path planning in an uncertain-configuration space," in *IEEE Int. Conf. on Robotics and Automation*, 2003, pp. 4185–4190.

[4] L. Mihaylova, J. D. Schutter, and H. Bruyninckx, "A Multisine Approach for Trajectory Optimization Based on Information Gain," in *Proceedings IEEE International Conference on Robotics & Automation*, 2002, pp. 661–666.

[5] J. Gonzalez and A. Stentz, "Planning with Uncertainty in Position: An Optimal and Efficient Planner," in *IEEE / RSJ Int. Conf. on Intelligent Robots and Systems*, August 2005, pp. 2435 – 2442.

[6] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in linear pomdps by factoring the covariance," in *Proceedings of the 13th International Symposium of Robotics Research*, Hiroshima, Japan, November 2007.

[7] R. He, S. Prentice, and N. Roy, "Planning in information space for a quadrotor helicopter in a GPS-denied environment," in *IEEE Int. Conf. on Robotics and Automation*, 2008, pp. 1814–1820.

[8] S. Prentice and N. Roy, "The Belief Roadmap: Efficient Planning in Belief Space by Factoring the Covariance," *The International Journal of Robotics Research*, 2009.

[9] J. Valencia, R. Andrade Cetto and J. Porta, "Path planning in belief space with Pose SLAM," in *IEEE International Conference on Robotics and Automation*, vol. 3, May 2011, pp. 78–83.

[10] J. de Geeter, J. de Schutter, H. Bruyninckx, H. van Brussel, and M. Decreton, "Tolerance-weighted L-optimal experiment design for active sensing," in *IEEE / RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 3, Oct 1998, pp. 1670–1675.

[11] L. Mihaylova, T. Lefebvre, H. Bruyninckx, K. Gadeyne, and J. De Schutter, "A Comparison of Decision Making Criteria and Optimization Methods for Active Robotic Sensing," in *Numerical Methods and Applications*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2003, vol. 2542, pp. 316–324.

[12] R. Sim and N. Roy, "Global A-Optimal Robot Exploration in SLAM," in *IEEE Int. Conf. on Robotics and Automation*, Apr 2005, pp. 661 – 666.

[13] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental Smoothing and Mapping," *Robotics, IEEE Transactions on*, vol. 24, no. 6, pp. 1365 –1378, Dec 2008.

[14] K. Konolige and M. Agrawal, "FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping," *Robotics, IEEE Transactions on*, vol. 24, no. 5, pp. 1066 –1077, oct. 2008.

[15] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "RSLAM: A System for Large-Scale Mapping in Constant-Time Using Stereo," *International Journal of Computer Vision*, vol. 94, pp. 198–214, 2011, 10.1007/s11263-010-0361-7.

[16] A. Pázman, *Foundations of Optimum Experimental Design (Mathematics and its Applications)*. Springer, 1986.

[17] F. Pukelsheim, *Optimal Design of Experiments (Classics in Applied Mathematics)*. Society for Industrial and Applied Mathematics, 2006.

[18] H. Chernoff, "Locally Optimal Designs for Estimating Parameters," *The Annals of Mathematical Statistics*, vol. 24, no. 4, pp. pp. 586–602, 1953.

[19] A. Wald, "On the Efficient Design of Statistical Investigations," *The Annals of Mathematical Statistics*, vol. 14, no. 2, pp. pp. 134–140, 1943.

[20] S. Ehrenfeld, "On the Efficiency of Experimental Designs," *The Annals of Mathematical Statistics*, vol. 26, no. 2, pp. pp. 247–255, 1955.

[21] T. Lefebvre, H. Bruyninckx, and J. De Schutter, "Task Planning With Active Sensing For Autonomous Compliant Motion," *The International Journal of Robotics Research*, vol. 24, no. 1, pp. 61–81, 2005.

[22] H. Carrillo, I. Reid, and J. A. Castellanos, "On the Comparison of Uncertainty Criteria for Active SLAM," in *IEEE International Conference on Robotics and Automation, ICRA'12*, May 2012.

[23] J. Kiefer, "General Equivalence Theory for Optimum Designs (Approximate Theory)," *The Annals of Statistics*, vol. 2, no. 5, pp. pp. 849–879, 1974.

[24] V. Fedorov, *Theory of Optimal Experiments (Probability and mathematical statistics)*. Academic Press Inc, 1972.

[25] H. Choset, W. Burgard, S. Hutchinson, G. Kantor, L. E. Kavraki, K. Lynch, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT Press, June 2005.

[26] P. Hart, N. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, Feb. 1968.

[27] A. Stentz, "The Focussed D* Algorithm for Real-Time Replanning," in *Proceedings of the International Joint Conference on Artificial Intelligence*, August 1995.

[28] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 354–363, 2005.

[29] RAWSEEDS, "Robotics advancement through Webpublishing of sensorial and elaborated extensive data sets (project FP6-IST-045144)," 2009, http://www.rawseeds.org/rs/datasets.

[30] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The new college vision and laser data set," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, May 2009. [Online]. Available: http://www.robots.ox.ac.uk/NewCollegeData/

[31] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[32] C. Stachniss, G. Grisetti, and W. Burgard, "Information Gain-based Exploration Using Rao-Blackwellized Particle Filters," in *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[33] J. van den Berg, D. Ferguson, and J. Kuffner, "Anytime Path Planning and Replanning in Dynamic Environments," in *IEEE Int. Conf. on Robotics and Automation*, May 2006, pp. 2366 – 2371.