The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Efficient Path Planning in Deformable Maps

Mark A. Whitty, *Student Member, IEEE* and José E. Guivant, *Member, IEEE*

***Abstract*—This paper presents a framework for efficient path planning in a deformable map. A roadmap and local cost maps are combined and integrated into a generic SLAM process to provide fast path querying for multiple sources and multiple destinations. Analysis of a simple deformation metric shows the ability of the framework to efficiently maintain a consistent plan during major map adjustment by updating the roadmap and selected local cost maps. Results from simulation verify the effectiveness of the framework in handling deformable maps in an efficient manner.**

I. Introduction

Recent work in autonomous exploration using SLAM has revealed the need for efficient methods of path planning in large-scale environments. Whereas classical path planning techniques have been highly developed for static and deterministic environments, reactive navigation techniques are typically implemented for real time applications. However, recent work in sampling based methods [1] has generated techniques able to replan in real time while guaranteeing probabilistic completeness. The application of these methods concurrently with SLAM has only recently been investigated [2] but is poised to offer results which will lead to greater autonomy in mobile robotics.

Early research in SLAM stems from work [3] by Smith *et al.*, which provided a stochastic framework for estimating uncertain relative spatial relationships. It soon became apparent that solving the full SLAM problem was both mathematically rigorous and computationally intensive [4], particularly in larger environments. More recent methods [5], [6], have shown a substantial increase in efficiency, frequently by breaking the SLAM representation into a hybrid of metric and topological components. The Atlas [7], Network Coupled Feature Maps [8] and Constant Time SLAM [9] techniques are example of submapping methods which are locally referenced and couple their submaps using an adjacency graph. However loop consistency in the graph is not maintained by these techniques, meaning the global map is not optimal [10]. Hierarchical SLAM [11] enables global consistency during loop closure and has been demonstrated to operate in real-time for loops of 200-350m. While path planning techniques for static environments have been extensively developed and successfully demonstrated [12], the ability to replan in a map which is non-static is still a challenging problem. At least three major approaches exist; geometric decomposition, topological decomposition and configuration space sampling.

By geometrically decomposing the environment to varying degrees of fidelity, efficient path planning for a single fixed goal state is achievable. Stentz [13] and Likhachev *et al.* [14] amongst others provided mechanisms to replan to adapt to unknown obstacles in this case. These mechanisms proved very effective in practical application [15]. By generating a single policy over the global environment, these techniques limit their ability to be reused by other agents in achieving alternate goals.

Constructing a topological map for path planning directly from sensor input was achieved by Choset and Natagani [5] using their Generalised Voronoi Graph, in an approach similar to that proposed by Kuipers and Byun [16]. In large open spaces beyond the sensor range, this approach did not handle motion uncertainty well, prompting the idea of coastal navigation [17]; where a path proximate with obstacles is used to maintain view of sufficient landmarks for improved localization accuracy. The concept of manifolds was expounded by Howard *et al.* [18] where the motion of an agent is considered in a manifold instead of a 2D world, maintaining consistency as per a topological map.

Current configuration space sampling techniques are the progeny of Probabilistic Road Maps (PRM) introduced by Kavraki *et al.* [19]. Collision with obstacles is checked at sample points between configurations and is thus restricted to a binary characterization of space; usually free and occupied. Roadmap style path planners were extended to continuous cost spaces by Jaillet [20] using Rapidly-exploring Random Trees (RRT). Efficient reuse of subtrees was exploited by Zucker [21] for replanning. The ability to handle multiple-source and multiple-destination queries is a feature of the configuration space sampling techniques.

All of the approaches described above are not configured to efficiently handle the type of map deformation present in events such as SLAM loop closure on a large scale. Instead, they rely on the environment being static on a global scale while handling local map changes in a very efficient manner.

Manuscript received March 9, 2009. This work was supported in part by the UNSW Research Excellence Awards.

M. A. Whitty is with the School of Mechanical and Manufacturing Engineering, University of New South Wales, Sydney, Australia, 2052, (phone: 61-2-93854125; e-mail: m.whitty@unsw.edu.au

J. E. Guivant is with the School of Mechanical and Manufacturing Engineering, University of New South Wales, Sydney, Australia, 2052, (phone: 61-2-93859820; e-mail: j.guivant@unsw.edu.au).

This paper presents a framework for efficient path planning and replanning in deformable maps by integrating a roadmap and local cost maps analogous to the structure of hybrid metric-topological SLAM. One advantage of the hybrid structure of this framework is the ability to provide fast multiple-source multiple-destination querying capability across the portion of the map present at any instant. A second advantage is the maintenance of a globally consistent path planner during major map adjustments. A third advantage is a computation reduction driven by a simple mechanism for delaying local cost map recalculation. Finally, the framework is of sufficiently general construct that most existing planning algorithms can be integrated to generate paths across cost surfaces. Thus it is applicable to exploration, navigation between specific locations and encourages the pursuit of tasks requiring a higher degree of autonomy than currently available.

The remainder of this paper is organized as follows. Section II presents the fundamentals of SLAM and environment representation along with a brief note on deformation monitoring. A novel framework for path planning in deformable maps is presented in Section III as the major contribution of this paper. Section IV provides several studies which verify the key concepts of the framework. A preliminary version of a path planner that implements the framework is described in Section V with the results of this implementation in Section VI. Section VII provides the conclusions of this paper.

II. SLAM AND DEFORMATION MONITORING

A. SLAM and environment representation

This section presents a brief review of the fundamentals of SLAM which form one entity of the proposed framework. A byproduct of both the need to provide accurate environment representations, or maps, and the need to maintain localization accuracy, SLAM is a process which relies on the inherent correlation between those sub-processes [3]. For larger environments, small independent submaps are combined to reduce both the computation and storage requirements of SLAM.

One framework for combining SLAM landmark maps with dense metric sensory information was proposed by Nieto *et al.* [22]. It uses the concept of Unidirectional Information Flow to decouple landmark-based SLAM from the estimation of dense maps. This work was mirrored by Chakravorty and Saha [23] who demonstrated the separation of mapping and planning in a control system framework. By assuming the dense map does not provide any information to the robot pose and landmark maps, dense feature maps can be associated with local regions, ostensibly those delineated by the submapping methods.

B. Deformation monitoring

While the effect of deformation on maps is widely understood, it is a specific feature of this deformation which provides the key to the efficiency of the framework presented in this paper. This feature was most eloquently expressed by Frese [24] when referring to a map as having 'certainty of relations despite uncertainty of positions'. Frese went on to describe the uncertainty of an aspect f of a map estimate $\hat{x}$ as the standard deviation of its covariance,

$$\delta_{\hat{x}} = \sqrt{P(f(\hat{x}))}\text{, where} \tag{1.1}$$

$$P(f(x)) = JP_{\hat{x}}J^T \tag{1.2}$$

and $P_{\hat{x}}$ is the covariance of $\hat{x}_k$. After calculating the Jacobian, J, of f with respect to the state vector $\mathbf{X} = [x_1, y_1, ...x_L, y_L]$ it is evaluated at the estimated mean $\hat{x}_k = [\hat{x}_{1,k}, \hat{y}_{1,k}, ...\hat{x}_{L,k}, \hat{y}_{L,k}]$ at time step k where L is the number of elements used to produce the map aspect.

III. PLANNING IN DEFORMABLE MAPS

This section contains the major contribution of this paper namely the introduction of a framework for efficient planning and replanning in deformable maps. Four sub-sections present the key aspects of this framework, namely:

- Structure of the framework
- Constructing a roadmap and local cost maps
- Multiple-source multiple-destination queries
- Handling map updates

A. Structure of the framework

Figure 1 shows how the framework consists of two major entities. The first is a localization and mapping entity which is assumed to provide a consistent representation of the environment. The second is a planning entity, which itself contains two components.

In a clear analogy with hybrid metric-topological SLAM, the planning entity's first component, referred to as a *roadmap*, is a graph connecting reachable locations in the configuration space. The second, referred to as the *local planner*, consists of a set of local cost maps.

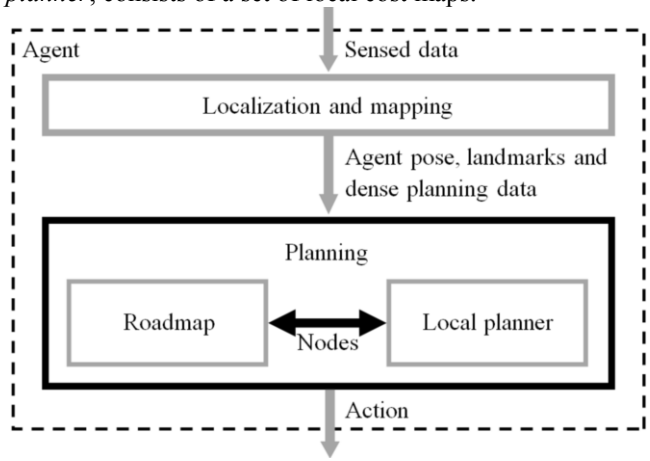

Fig. 1. Framework for planning in deformable maps.

Fig. 2 illustrates how the two components are integrated through the use of nodes, which are entities that exist as vertices in the roadmap but also exist as a fixed location in a single instance of a local cost map.

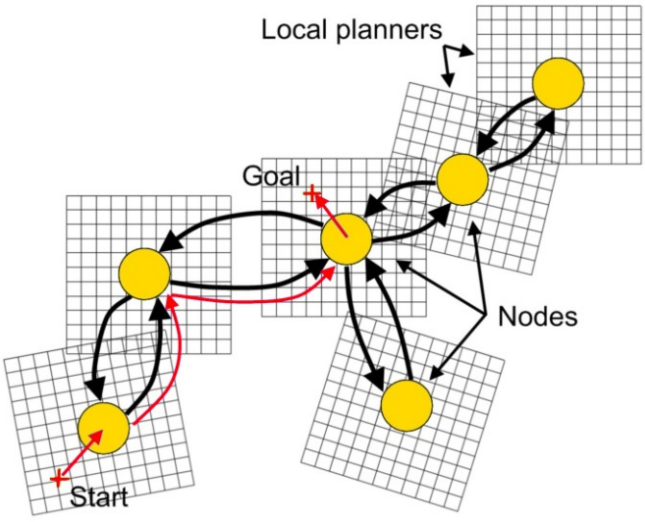

Fig. 2. Combining a roadmap with local planners.

The only information passed is the SLAM state vector and covariance matrix and a data structure containing consistent dense information, referred to as the *dense planning data*. Some examples of dense planning data are presented by Nieto et al. [22] in their HYbrid Metric Map approach.

B. Constructing a roadmap and local cost maps

A high level roadmap is constructed node by node as the agent initially explores its environment. When nearing unknown spaces near the edge of the current local cost map, a new local cost map is initiated following the algorithm in Table I. To maintain the unidirectional information flow, we do not add the position and orientation of the local cost map to the SLAM state vector, but rather express this pose as a function of a set of proximate landmarks known as the *neighboring landmarks*.

Table I. Initial cost map initialization

Algorithm
if distance to nearest node > threshold
Create a new local cost map
Set local cost map orientation to match global agent pose
Set local cost map origin to centroid of *k* neighboring landmarks
For each of *k* neighboring landmarks
Store landmark position in local coordinates with map
For each of *n* monitoring landmarks
Store landmark position in global coordinates with map
Select node position in local cost map
Create a new vertex in roadmap corresponding to this node
Calculate local cost map policy over local dense planning data
For each node overlapped by the new local cost map
Connect corresponding vertices with traversal cost on edge
For each node overlapped by the new local cost map
Connect corresponding vertices with traversal cost on edge

We assume the neighboring landmarks will be highly correlated, and store their indices and original positions in the local coordinate system generated as part of the local cost map. Thus the local cost map origin is defined as a function of a set of correlated landmarks in the SLAM process, which gives it a global pose that will deform with the landmarks. One, location in the local cost map frame is assigned the location of the node associated with this cost map and is also added as a new vertex in the roadmap. Additionally, a set of nearby *monitoring landmarks,* Ω, is defined and stored to aid replanning.

On initialization, each local cost map first extracts the overlapping dense planning data and then initiates a local planner. The local planner generates a locally optimal policy over the local region which is used to connect the roadmap to any point in the configuration space spanned by the local cost map.

C. Multiple-source multiple-destination queries

The hybrid structure of this framework provides the foundation for fast multiple-source multiple-destination querying capability across the known region spanned by the local cost maps. For any given source point and destination points in the region spanned by the local cost maps, the query procedure is illustrated in Fig. 2 and described by the algorithm in Table II.

Table II. Multiple-source multiple-destination querying

Algorithm
Find closest local cost maps which overlap the start and goal points
Do shortest path query between the corresponding nodes
Lookup paths corresponding to graph edges
Concatenate paths
Refine path according to computation time available

First determine which local cost maps overlap the given points. Then inspect each of the corresponding policies and choose the source and destination policies which minimize the traversal cost to and from the node. Perform a shortest path query on the roadmap between the corresponding vertices according to the costs associated with each edge. Once the shortest path through the roadmap has been found, the agent simply concatenates the corresponding sequence of local cost map policies which provides the complete path.

D. Handling map updates

As the SLAM process progresses in an unknown environment, the local cost maps will move but will remain rigid despite major map adjustments. This idea is further explored in Section IV and is well known in the SLAM community [24]. As was indicated in the literature review, it does not appear to have been considered in the context of planning and thus analysis of such is the major contribution of this paper. If a local cost map needs to be updated globally, its position and orientation are found by Procrustes shape analysis of the previous locally stored neighboring

landmark positions and the current global neighboring landmark positions provided by the SLAM process.

Despite the rigid body motion of local cost maps, there is still the possibility that local deformation will cause a local cost map policy to become inconsistent with its underlying dense planning data. As shown in Section III. C., deformation is monitored using a set of proximate landmarks associated with each local cost map. These may or may not be coincident with the neighboring landmarks used to maintain the local cost map's pose.

The geometry of the set of monitoring landmarks is tracked using a *deformation metric*. Deformation metrics are discussed in more detail in Section IV and are used to determine the extent to which non-rigid body motion has occurred locally. Once the deformation metric changes by a fixed proportion of its own value the amount of deformation is sufficient to recalculate the local policy. The positions of the monitoring landmarks are then reset as per Table III.

Table III. Planner update procedure

Algorithm
On each major SLAM update
For each local cost map proximate to agent
Recalculate policy based on current dense planning data
Check and update proximate roadmap edges
For every local cost map
Adjust global position and orientation according to neighboring landmarks
Calculate current δ_F and lookup previous δ_F'
If $\frac{\delta_F' - \delta_F}{\delta_F'} > threshold$
Recalculate local policy
Update roadmap if necessary
Set $\delta_F' = \delta_F$
If agent near local cost map boundary
Initialize new local cost map

Of course, an agent observing the environment will be updating the dense planning data underlying the local cost maps which span the agent's observable range. The corresponding local policies thus need to be recalculated on a regular basis. If the roadmap's edge costs are extracted from the local cost maps, then the relevant edge costs will also need to be updated, however this can be delayed until the next path query.

IV. Deformation Monitoring

While effective techniques for determining the final quality of a dynamically constructed map have been the focus of recent papers, few results in quantifying the type and extent of map deformation are in evidence. While it is not essential to have SLAM as the underlying map pose maintenance construct, the efficiency of this framework relies strongly on local rigid body motion of the map and the use of deformation metrics to determine when and where replanning is necessary. Both of these characteristics can be shown to be satisfied when using an underlying SLAM process.

A. Deformation metrics

A simple measure of the non-linear deformation experienced by a map is obtained by considering a distance measure, F, based on the set of so called monitoring landmarks, Ω. A logical distance measure is to take the set of Euclidean distances between pairs of the n monitoring landmarks:

$$\begin{aligned} F\left(\{(x_i,y_i)\}_{i\in\Omega}\right) &= \begin{bmatrix} f_{1,2} & f_{1,3} & \dots & f_{n-1,n} \end{bmatrix}^T \\ f_{i,j} &= \left\| (x_j,y_j) - (x_i,y_i) \right\|_2 \\ &= \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \end{aligned} \tag{4.1}$$

The uncertainty, δ_F, of this distance measure is the standard deviation of its covariance, P_F, which can be obtained from

$$P_F = J \cdot P_X \cdot J^T \tag{4.2}$$

where P_X is the covariance of the monitoring landmarks. The trace of δ_F provides a suitable scalar output for the deformation metric.

B. Localized rigid body motion of the map

To illustrate the effect of local rigid body motion, a simulation was run to visualize the type and extent of map deformation which may occur during SLAM. A set of proximate monitoring landmarks was associated with each of a sample subset of landmarks. Fig. 4 shows the deformation metric for each set of monitoring landmarks on each successive SLAM update.

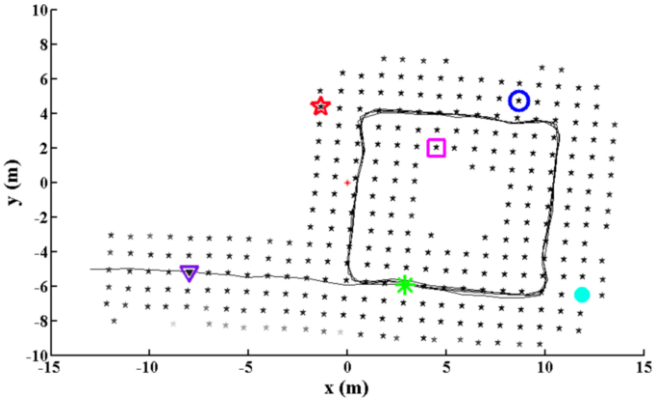

Fig. 3. SLAM simulation for testing deformation metrics.

Fig. 3 shows the path of the agent, starting at the bottom left, which traversed the loop four times in the simulation. Colored icons indicate the positions of landmarks in the sample subset, which also correspond to the colors in Fig. 4. From this figure, it is very easy to see that the purple triangle landmark which entered the sensor's field of view on one occasion decreased its local uncertainty monotonically while the agent was nearby. After the agent had passed, the local uncertainty remained constant, even though the remainder of the map was changing shape. This clearly supports the hypothesis that the rigidity of a local region is independent of deformation in distant locales. This is in agreement with

the formulation by Masson *et al.* [25], where the 'decorrelation effect' justified the use of local map representations for DenseSLAM.

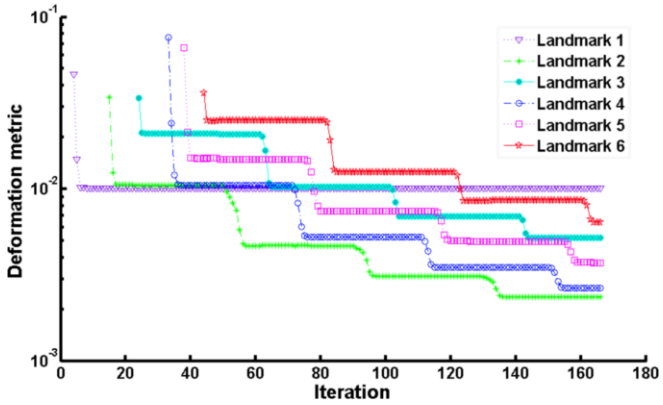

Fig. 4. Semi-log plot showing the deformation metrics for the sample subset of landmarks in Fig. 3.

The same effect is seen in the remaining landmarks of Fig. 4, where the local deformation metric remains constant when the agent is not close by. When the agent approaches the landmark, a sharp decrease in the uncertainty is experienced.

V. Implementation

This section presents a preliminary version of a two dimensional path planner which implements the planning entity required by the framework presented in Section III. The local planner consists of a simple rectangular grid that has its pose defined as a function of the neighboring landmarks. The local cost map extracts dense planning data from the underlying localization and mapping entity, which in this case is simply a point cloud. The average number of points between two cells is added to the distance between cell centers to form a traversal cost.

Local cost maps are generated as the agent explores the environment. Nodes are positioned at the centre of a single grid cell. Dynamic programming is used to convert the traversal cost to a discrete policy centered at the node over all the cells in a local cost map. To enable the roadmap to be populated with sufficient edges, we constrain the positioning of new local cost maps to have their nodes overlapping existing local cost maps. The roadmap edge cost then simply becomes the traversal cost obtained by following the lowest cost policy from the centre of one node to the next. When no possible path exists between two cost maps at the local planner level no direct link is inserted in the roadmap.

Querying the map is performed by first identifying all the local cost maps which overlap the source and destination points. Then the corresponding policies are inspected and the lowest cost nodes are identified. A shortest path algorithm is used to find the path between these vertices in the roadmap. If a path is found, the concatenated sequence of policies is traversed by the agent. As was foreshadowed by Frese [24], all the local cost maps contain policies which are relative and not absolute globally. Thus an agent may navigate by following a sequence of local reference frames.

VI. Experimental Results

To emphasize the efficiency gains of the framework, a series of local cost map planners was simulated on a two dimensional space. Initially, a single rectangular cost map was use to find the path between two points. Then the number of local cost maps was increased from 1 to 16, while maintaining the same grid cell resolution and total map size. The time taken to construct the planner and query it were calculated. These experiments were conducted using Matlab on a dual core 2.6GHz processor. 5000 dense points were used to represent the dense planning data.

Fig. 5 shows the resultant paths. The red circles and lines indicate the vertices and edges of the roadmap. Green color indicates the paths from the start and goal points to the nodes and the paths between vertices in the roadmap. Note that the paths are required to travel through at least one node, which means the total path cost is not optimal in the global sense as can be clearly seen in Fig. 5(b). For clarity, the extent of each local cost map is not shown.

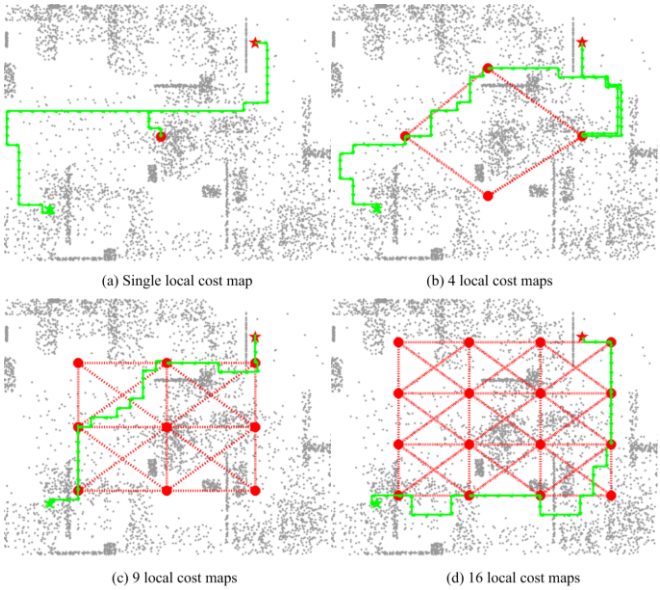

Fig. 5. Comparison of four amounts of local cost maps defined over the same 2D space. The red circles and lines delineate the roadmap. Green lines are the lowest traversal cost paths extracted from the roadmap and local cost maps. (a) contains a single local cost map while (b), (c) and (d) contain 4, 9 and 16 partially overlapping local cost maps respectively.

The results show how an agent is able to navigate through a densely populated area while following a sequence of locally defined paths which individually attempt to minimize the traversal cost. Considerable variation in the paths was found with differing amounts of local cost maps, due primarily to the intuitive placement of nodes in the center of evenly distributed local cost maps.

Table V shows that calculating the policy for the cost maps is the most computationally expensive part of the algorithm presented in Section V. By using a larger number of local cost maps, this time is substantially reduced, as computational time for each local cost map varies quadratically with its size. These proof of concept results are only intended to be a relative guide.

Table V. Comparison of planning and replanning speeds for different numbers of local cost maps.

	Time (s)				
Number of local cost maps	Extract dense planning data (s)	Calculate policy for all local cost maps (s)	Create roadmap (s)	Query (s)	Recalculate policy for one local cost map (s)
1	0.637	38.823	0.027	0.106	39.46
4	1.215	38.969	0.009	0.073	10.05
9	1.891	22.013	0.015	0.038	2.656
16	2.462	16.530	0.024	0.046	1.187

The largest advantage can be seen by considering the policy recalculation time normalized over the number of local cost maps. This decreases rapidly as the number of local cost maps increases, so if a change in deformation greater than the threshold occurs in only one local cost map, it is much more efficient to recalculate only that policy rather than a single policy over the whole region. Thus the effectiveness of the proposed framework is clearly verified.

VII. Conclusion

We have presented a framework for combining path planning with a generic SLAM implementation which enables fast multiple-source multiple-destination querying in a deformable map. The framework also allows efficient replanning during map deformation due to the relative positioning of local cost maps and use of a deformation metric to control replanning frequency.

Currently, work is ongoing to increase the efficiency of the local planner and integrate it fully with the SLAM implementation used to investigate the deformation metrics. Once completed, a full comparison with existing path planning methods can be undertaken. Further investigation into the behavior of deformation metrics is in progress and is expected to improve the efficiency of this framework even further. Ultimately we believe that this framework will provide the basis for higher levels of autonomy with cooperating agents.

Acknowledgment

The underlying EKF SLAM simulation was based on an implementation by Tim Bailey. Thanks to Udo Frese for helpful discussions regarding deformation monitoring.

References

[1] H. Choset, *Principles of Robot Motion: Theory, Algorithms, and Implementation.* Cambridge, MA: The MIT Press, 2005, pp. 242–246.

[2] Y. Huang, K. Gupta, "RRT-SLAM for motion planning with motion and map uncertainty for robot exploration," *Proc. 2008 IEEE Conf. on Robotics and Automation*, pp. 1077-1082, Sept. 2008

[3] R. Smith, M. Self, and P. Cheeseman, *Estimating uncertain spatial relationships in robotics*, in *Autonomous Robot Vehicles*, New York: Springer-Verlag, 1990, pp. 167-193.

[4] J. E. Guivant and E. M. Nebot, "Optimization of the Simultaneous Localization and Map Building Algorithm for Real Time Implementation," *IEEE Trans. Robotics and Automation,* vol. 17, no. 3, pp. 242-257, June 2001.

[5] H. Choset and K. Natagani, "Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization," *IEEE Trans. Robotics and Automation,* vol. 17, no. 2, pp. 125-137, April 2001.

[6] J. L. Blanco, J.-A. Fernandez-Madrigal, J. Gonzalez, "A New Approach for Large-Scale Localization and Mapping: Hybrid Metric-Topological SLAM," *Proc. 2007 IEEE International Conference on Robotics and Automation,* pp. 2061-2067, April 2007.

[7] M. Bosse, P. Newman, J. Leonard, and S. Teller, "Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework," *International Journal of Robotics Research*, vol. 23, no. 12, pp. 1113–1140, 2004.

[8] T. Bailey, "*Mobile Robot Localisation and Mapping in Extensive Outdoor Environments,"* PhD thesis, University of Sydney, Australian Centre for Field Robotics, 2002.

[9] J. Leonard and P. Newman, "Consistent, Convergent, and Constant-Time SLAM," *Proc. 2003 International Joint Conference on Artificial Intelligence*, vol. 18, pp. 1145-1150, 2003.

[10] J. A. Castellanos, J. Neira, and J. D. Tardos, "Map building and SLAM algorithms," in *Autonomous Mobile Robots*, CRC Press, pp. 335-372, 2006.

[11] C. Estrada, J. Neira, and J. D. Tardos, "Hierarchical SLAM: Real-time accurate mapping of large environments," *IEEE Trans. Robotics*, vol. 12, no. 4, pp. 588-596, August, 2005.

[12] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, vol. 99, no. 1, pp. 21-71, 1998.

[13] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proc. 1994 International Conference on Robotics and Automation*, vol. 4, pp. 3310-3317, May 1994.

[14] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime Dynamic A*: An Anytime, Replanning Algorithm," in *Proc International Conference on Automated Planning and Scheduling (ICAPS),* pp. 262-271, 2005.

[15] D. Ferguson, T. Howard, M. Likhachev, "Motion planning in urban environments," *Proc. of IEEE Intelligent Vehicles Symposium*, pp. 1149-1154, 2008.

[16] B. Kuipers, P. Beeson, J. Modayil, and F. Savelli, "The Hybrid Spatial Semantic Hierarchy: Factoring the Mapping Problem," in *Proc. 2004 IEEE International Conference on Robotics and Automation,* vol. 5, pp. 4845-4851, 2004.

[17] N. Roy and S. Thrun, "Coastal navigation with mobile robots," *Advances in Neural Processing Systems*, vol. 12, pp. 1043-1049, 1999.

[18] A. Howard, G. S. Sukhatme, and M. J. Mataric, "Multirobot simultaneous localization and mapping using manifold representations," in *Proceedings of the IEEE*, vol. 94, no. 2, pp. 1360-1369, July 2006.

[19] L. E. Kavraki, P. Svetska, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robotics and Automation,* vol. 12, no. 4, pp. 566-580, August 1996.

[20] L. Jaillet, J. Cortes, and T. Simeon, "Transition-based RRT for Path Planning in Continuous Cost Spaces," in *Proc. 2008 IEEE Int. Conf. on Robots and Systems,* pp. 2145-2150, 2008.

[21] M. Zucker, J. Kuffner, and M. Branicky, "Multipartite RRTs for rapid replanning in dynamic environments," in *Proc. 2007 IEEE Int. Conf. on Robotics and Automation*, pp. 1603-1609, 2007.

[22] J. Nieto, J. E. Guivant, and E. M. Nebot, "DenseSLAM: Simultaneous localization and dense mapping," *International Journal of Robotics Research*, vol. 25, no. 8, pp. 711-744, August 2006.

[23] S. Chakravorty and R, Saha, "Simultaneous planning localization and mapping: A hybrid Bayesian / frequentist approach," *American Control Conference*, pp. 1226-1231, June 2008

[24] U. Frese, "A Discussion of Simultaneous Localization and Mapping," in *Autonomous Robots*, vol. 20, no. 1, pp. 25-42, 2006.

[25] F. Masson, J. Nieto, J. Guivant, E. Nebot, "Robust autonomous navigation and world representation in outdoor environments," in *Mobile Robots: Perception & Navigation*, S. Kolski, Germany: Pro Literatur Verlag, 2007, pp. 299-320.