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

Normalized graph cuts for visual SLAM

John G. Rogers and Henrik I. Christensen

Abstract— **Simultaneous Localization and Mapping (SLAM) suffers from a quadratic space and time complexity per update step. Recent advancements have been made in *approximating* the posterior by forcing the information matrix to remain sparse as well as *exact* techniques for generating the posterior in the *full SLAM* solution to both the trajectory and the map. Current approximate techniques for maintaining an online estimate of the map for a robot to use while exploring make capacity-based decisions about when to split into sub-maps. This paper will describe an alternative partitioning strategy for online approximate real-time SLAM which makes use of normalized graph cuts to remove less information from the full map.**

I. INTRODUCTION

The problem of Simultaneous Localization and Mapping (SLAM) has seen a flurry of publication over the past decade. Many of the core issues have been addressed in a research setting, yet a viable general-purpose implementation remains elusive. The absence of a general purpose reliable SLAM module is due to two main issues in moving from the research environment to the real world. The first main issue, which will be addressed in this paper, is the quadratic update complexity inherent in maintaining a fully dense covariance or information matrix. The second issue, which will be peripherally discussed in this paper, is the *data association* problem – if even a single feature is mistakenly confused for another, the entire map can be rendered inconsistent.

Most modern approaches to solving the SLAM problem follow one of two main themes. One approach stems from advancements made on the Structure from Motion (SFM) problem in the computer vision community. This approach is to optimize the representation of the perceived environment in an offline manner. This has been applied in robotics as the Smoothing and Mapping (SAM) problem [12], which is also known as *full SLAM* because the entire robot trajectory is optimized along with the landmark poses. The original approach from the robotics community has been to use a recursive filter algorithm to marginalize past poses in the interest of real-time operation, but at the expense of correlating all map features.

The original approach to the SLAM problem is to update a state vector and covariance matrix composed of the robot pose and the estimated landmark positions with the Extended Kalman Filter (EKF). This technique evolved out of the original successful application of the EKF for mobile robot localization with an *a priori* map in [7] and [5]. By placing the estimated landmark locations within the state vector and updating them simultaneously, the first SLAM implementation was reported by [25]. The recent summary papers by Durrant-Whyte and Bailey [13], [1] serve as an excellent survey of the history of the SLAM problem, from its initial beginnings up to the state-of-the-art today.

The full SLAM, or Smoothing and Mapping (SAM) problem allows for the use of a sparse representation for the covariance or information matrix by maintaining the entire robot trajectory within the state vector. Features are correlated when prior poses are marginalized out, in effect, the EKF does this with every prior pose and correlates all features resulting in a dense covariance matrix. Folkesson and Christensen developed GraphSLAM [15], which was able to close loops and avoided linearization error through the use of a nonlinear optimization engine. Loop closure was achieved by adding human-guided constraints between features and then reoptimizing. Dellaert *et. al.* [12] has developed the Square Root SAM algorithm which uses sparse Cholesky factorization to optimize a set of landmark measurements and the robot trajectory in an efficient manner. Further progress has been made on online solutions to the SAM problem which uses QR factorization for reordering the measurements to get optimal and online or incremental updates such as with incremental SAM (iSAM) [17], [18].

These algorithms represent excellent progress towards a large-scale solution to the SLAM problem; however, each of them requires an ever increasing amount of computation per update step. EKF updates suffer from quadratic complexity per update step, iSAM updates can be held to near linear complexity per update step with proper reordering. Sub-mapping strategies have been developed to *approximate* the solution to the SLAM problem while maintaining constant time update steps. This is accomplished by keeping the size of the sub-map at no larger than a fixed maximum number of features. Since the filter approximates the solution by assuming that features in other maps are independent, it can confine the update to the local map, which is smaller than a constant maximum size.

Sub-mapping strategies such as Atlas [4] and Hierarchical SLAM [14] maintain constant sized local maps and offer loop closure updates which are linear in the size of the loop, in terms of local map frames traversed. Other algorithms postpone global updates until they are needed and focus on local updates such as the Compressed EKF [16]. Techniques within the SAM community such as Tectonic SAM [22] are able to optimize sub-maps to generate exactly the same results as full Square Root SAM but with the efficiency of a constant sized map. Since each local map starts with

This work was supported by the ARL CTA MAST project 104953

J. G. Rogers is a Ph.D. student at Georgia Tech College of Computing jgrogers@cc.gatech.edu

H. I. Christensen is the KUKA Chair of Robotics at Georgia Tech College of Computing hic@cc.gatech.edu

the robot perfectly localized in these schemes, sub-maps offer some protection against the data association ambiguities in nearest-neighbor gating which would otherwise be more difficult with larger pose uncertainties. Loop closure in sub-map algorithms requires joint data association such as [21]; however, pose uncertainties are small enough within a local map to permit nearest-neighbor gating. This is a significant advantage for the visual SLAM technique used in this paper which relies on low uncertainties to achieve real-time performance while measuring visual features. These sub-mapping strategies make fixed, capacity-based decisions about when a sub-map should be made; this paper will demonstrate an alternative strategy based on *normalized graph cuts* from the image segmentation community which will provide a better approximation to the full map.

Techniques such as [3] also use normalized graph cuts to partition the maps, but they use a *sensed-space overlap* metric, which is a spatial separation based on proximity. The technique described in this paper is different in that it uses an information sharing metric for its edge weights. This technique uses a Hybrid Metric-Topological sub-mapping data structure instead of Atlas as shown in [2].

This paper will develop an alternative sub-mapping strategy by first discussing the theory behind the component technologies in Section II. In Section III the specific software implementation will be detailed, then in Section IV experimental results will be presented. Section V draws conclusions and briefly outlines the future direction of this work.

II. METHODOLOGY

This paper presents a new strategy for choosing partitions in an Atlas framework based on normalized graph cuts from the image segmentation community.The software described in this paper uses Andrew Davison's SceneLib [9], [10], [8][1]. The unified inverse-depth parameterization of Montiel *et. al.* [20] is used for landmark representation, and the Atlas framework of [4] is used to maintain local maps. The Joint Compatibility Branch and Bound (JCBB) algorithm of [21] is used for robust data association. Normalized graph cuts will be used to find partitions to separate local maps which balance map capacity while removing as little information as possible.

A. Monocular SLAM

Davison's early work in [9], [10], [8] showed that it is possible to perform real-time SLAM with a commodity monocular camera without the use of odometric or intertial motion feedback. The real-time performance is accomplished by projecting the covariance ellipse of the feature into the image, which, by taking the 99% confidence level yields a search region which can be examined to find the feature. Since this region is much smaller than the overall image, the template search is quick to yield an accurate measurement of the feature. This can be done at full frame-rate which allows the use of an impulse motion model without any odometric or intertial feedback.

[1]Andrew Davison has made the MonoSLAM software available at `http://www.doc.ic.ac.uk/~ajd/Scene/index.html`

B. Inverse Depth

The software used in this paper uses the core components of SceneLib, but the two-phase feature initialization is removed by using the unified inverse-depth parameterization [20]. In the inverse depth parameterization, the feature state in the filter is expanded from the 3 spatial coordinates into a spherical coordinate system consisting of the origin of the feature. which was the robot position when it was first observed, the horizontal bearing θ, the vertical bearing ϕ, and the *inverse depth* ρ.

Using this parameterization, the projection of the feature into the camera is shown in equation 1. Please refer to [20] for specific implementation details. This projection makes use of the inverse depth of the landmark, which allows it to be well-defined even when $\rho_i = 0$, when the depth is infinite.

$$h^C = R^{CW}\left(\rho_i \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} - r^{WC}\right) + \begin{pmatrix} \sin\theta_i \cos\phi_i \\ \cos\theta_i \cos\phi_i \\ \sin\phi_i \end{pmatrix}\Bigg) \tag{1}$$

C. JCBB

The public version of MonoSLAM uses a simple individual compatibility nearest neighbor data association gate. This is fine for small motions and accurate poses and unambiguous features, but in a more realistic setting in a large scale robot trajectory a more robust data association test is needed. The technique outlined in this paper makes use of the Joint Compatibility Branch and Bound (JCBB) algorithm due to Neira et al [21]. JCBB has been used with MonoSLAM before in [6]; however, this development has not been included in MonoSLAM so it was re-implemented for this work.

D. Atlas

The approach described in this paper uses the Atlas framework developed by [4], which is a technique for sub-mapping based on computing the minimum uncertainty path to each other frame using a modified form of Dijkstra's shortest path algorithm. Loop closure in Atlas is accomplished by the addition of a link between local frames of reference with a transformation which can be estimated through correspondences between features in common between these two map frames. Atlas is a technique for *approximating* the true posterior map while maintaining near constant-time performance. Local maps are formed when the capacity of the previous map is filled with a pre-defined maximum number of features. This capacity based partition choice forms a cut which could potentially remove more information than a more informed choice. A partitioning scheme which minimizes the information removed from the map will generate a more accurate map than a scheme which arbitrarily selects this partition. The choice of Atlas as a underlying framework is merely an implementation detail; the use of normalized graph cuts to find map partitions could be used with any sub-mapping framework.

E. Normalized Graph Cuts for Image Segmentation

Normalized graph cuts was used by Shi and Malik to provide better results than standard graph cuts on image segmentation tasks in the presence of outliers [24]. Minimizing the cost of the graph cut often consists of a highly unbalanced partition where very few features will appear in one of the sets, and the bulk will remain in the other. For making submaps, this unbalanced partition is undesirable because it will typically separate few features and fail to realize the benefit of submapping. Instead of minimizing the cost of the cut, Shi and Malik minimize the disassociation measure called the *normalized cut*(Ncut):

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

where $cut(A,B)$ is the sum of all the weights of the edges removed by this cut, and $assoc(A,V)$ is the sum of all the weights of all edges connecting the set A to all of the vertices in the graph. Finding a partition $(A,B), A \bigcup B = V$ which minimizes this $Ncut(A,B)$ is the goal. Unfortunately, finding this partition is NP-hard, but by relaxing the requirement that set membership is absolute, and allowing a vertex to be partly in one set and also partly in another, this can be approximated by an eigenvalue problem. For completeness, please refer to the source paper [24] for the details of this computation and proof.

In terms of robot map building, normalized graph cuts will be used to find better partitions of the state vector into subsets which follow the typical compact block-diagonal structure of the covariance matrix.

III. IMPLEMENTATION

For the implementation we have adopted the SceneLib package provided by Davison [10], [9], [8], [11]. This package allows for rapid prototyping and it provides a reference for evaulation of performance. The software used in this paper has applied many of the advancements made to the open-source SceneLib package which have been detailed in the literature since its release in 2006. These advancements include the inverse depth parameterization from [20], and the sub-mapping technique Atlas from [4].

A. Monocular SLAM implementation

The intended application for SceneLib is that of performing SLAM on a monocular hand-held camera; with the addition of an odometric motion model, SceneLib can be made to work more accurately. This enables the visual SLAM algorithm to keep the small search regions for features even during larger scale trajectories, which keeps the frame rate high.

Monocular SLAM typically requires a two-phase feature initialization process. The use of the inverse depth parameterization allows us to eliminate this two-phase initialization. This reduces the complexity significantly and also allows landmarks to be used immediately in the SLAM update. A large depth uncertainty is placed on a new landmark – one which includes infinite depth with significant probability. The filter will be able to use this feature for correcting heading errors when it is first observed, and once the feature converges to a better estimate it will also help correct spatial pose errors as well. The modular architecture of SceneLib made the inclusion of the inverse depth parameterization a simple process.

Despite the excellent performance of SceneLib in a small environment, the algorithm suffers from quadratic update complexity and will ultimately fail to maintain real-time operation when presented with a larger environment. This issue is addressed in the large-scale MonoSLAM implementation of [6] which uses submapping with MonoSLAM with an optimization step to align adjacent map frames based on shared features.

B. Atlas implementation

Atlas leaves the choice of the loop-closure strategy open to the user. The implementation in [4] finds landmark correspondences and finds a robust transformation between their local frames of reference using RANSAC. The robot used in [4] is confined to a 2D plane and it uses a laser scanner to collect landmark measurements. Unfortunately, a search for corresponding landmarks does not extend well to the visual SLAM world of 3D features and images. Visual features are unlikely to be shared between two local maps. To adapt the Atlas scheme to the visual SLAM domain, this paper has employed the simple technique of projecting the features from the reference frame with which the loop closure is to take place into our current camera pose. This is accomplished by computing the pose of the robot in the other reference frame using the minimum uncertainty projection from Atlas and also computing the uncertainty in the pose using the rule given by $J_1(T_{ab}^{-1}, x_v^b)$, which is the Jacobian with respect to the first parameter of the composed transformation between the two map frames and the robot's current pose x_v^b in frame b, J_2 is likewise defined as the Jacobian with respect to the second parameter. The covariance of the composite transformation can be computed using formula 2. More details on the Jacobians J_1 and J_2 can be found in [26]. These are the same rules which are used in Atlas to compute the uncertainty along a given set of transformations.

$$\Sigma_{ac} = \begin{array}{c} J_1(T_{ab}^{-1}, x_v^b)\Sigma_{ab}J_1(T_{ab}^{-1}, x_v^b)^T + \\ J_2(T_{ab}^{-1}, x_v^b)\Sigma_{bc}J_2(T_{ab}^{-1}, x_v^b)^T \end{array} \quad (2)$$

The current Atlas strategy for splitting maps is to build a map until a finite capacity is reached. At this point, a sub-map is created. This strategy is essentially equivalent to making arbitrary cuts in the information shared between features until a partition is formed. A better strategy is to partition the set of features into smaller maps based on a cut which minimizes the information lost. This strategy will make a final *a posteriori* map which is closer to the ideal unpartitioned map than the *a posteriori* map from capacity Atlas.

Adapting the normalized graph cuts [24] algorithm from the image segmentation community to be used for partitioning robot map features is accomplished by determining

that the affinity measure between features is the volume of the information which connects them. This is a simple computation from the information matrix, which is formed by inverting the covariance matrix $I = \Sigma^{-1}$.

The only terms of interest are the $I_{y_i y_j}$ terms which measure the information shared between two features y_i and y_j. The weight on the edge connecting feature y_i to feature y_j is set to $W_{ij} = \left|\det I_{y_i y_j}\right|$ which measures the total information shared between these two features, regardless of their parameterization. Now the normalized graph cuts algorithm can partition these two sets into balanced sets which have the minimum information shared along the cut. From [24], we set D to be the diagonal matrix with $D_{ii} = \sum_j W_{ij}$ and then we can solve the eigenvalue problem:

$$(D - W)y = \lambda D y$$

to find the real-valued eigenvector y corresponding to the second-smallest eigenvalue γ. Thresholding the elements of y gives a partitioning on the set of features: if $y[i] > 0$ then feature y_i is in set 1, and if $y[i] < 0$ then feature y_i is in set 0.

C. Map Partitioning

The transformation between the old map and the new map is initially set to the old robot pose and covariance in the old map, just as with capacity based Atlas. The robot pose in the new frame is set to zero, with zero covariance Σ_{xx}.

Unlike in capacity based Atlas, partitioning based on normalized graph cuts requires actually splitting up an active map two generate two sub-maps rather than simply starting a new map. Fortunately, this just involves re-expressing the features which will be carried into the new map frame in its coordinate system, and transforming the covariance matrix by this transformation.

Features y_i which are to be moved from frame a to b are transformed according to the rule $y_i^b = T_{ab} y_i^a$ Since y_i are expressed in a spherical "inverse depth" coordinates, this transformation involves re-expressing the origin in terms of the new coordinate system b and rotating the horizontal bearing θ and vertical bearing ϕ into the new representation. The inverse depth coordinate ρ remains unchanged.

Feature covariance elements $\Sigma_{y_i y_j}$ are set to zero and are removed if y_i and y_j fall into separate partitions. $\Sigma_{x y_i}$ is transformed according to the rule seen in equation 3

$$\Sigma_{x y_i}^b = \frac{\delta x^b}{\delta x^a} \Sigma_{x y_i}^a \frac{\delta y_i^b}{\delta y_i^a}^T \tag{3}$$

The only type of covariance submatrix which remains is $\Sigma_{y_i y_j}$ where y_i and y_j fall within the same partition (or are the same element if $i == j$) in which case the transformation rule is seen in equation 4.

$$\Sigma_{y_i y_j}^b = \frac{\delta y_i^b}{\delta y_i^a} \Sigma_{y_i y_j}^a \frac{\delta y_j^b}{\delta y_j^a}^T \tag{4}$$

Using these transformation rules, the new covariance and state vectors can be constructed for the partitioned sub-map. The robot is now able to proceed with its task using this new local map.

IV. EXPERIMENTAL RESULTS

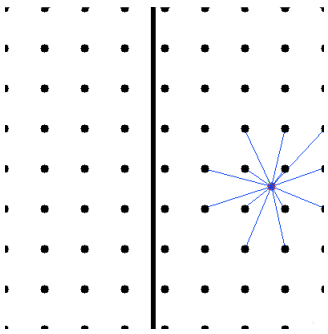

Fig. 1. Example proof-of-concept set up for Experiment 1.

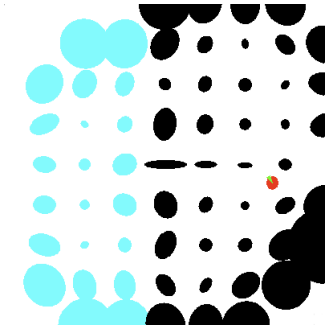

Fig. 2. Result of normalized graph cuts on proof-of-concept. Each sub-map is colored differently. The size of each blob corresponds to the associated uncertainty in its location.

A proof-of-concept simulation was created to validate the use of the normalized graph cuts algorithm for the application of splitting up robot maps. This simulated world is a typical 2D planar robot which makes range and bearing measurements on a set of features, which are located on a regular grid. An artificial "wall" has been placed in the center of the world which splits the features on the right half from the features on the left half as can be seen in figure 1. This wall prevents the robot from measuring features from the opposite side, which approximates the effect of a wall in the real world separating two rooms, whose features can be independently measured and may be a good candidate for partitioning into sub-maps.

In this initial proof of concept experiment, the partitioning algorithm was able to immediately segment the features according to the most natural partition suggested by the wall. This result can be clearly seen to correspond exactly to the wall in figure 2.

The input for the second experiment is camera data and odometry from a Pioneer PeopleBot moving through different trajectories in our lab. The results from two trajectories are shown below, the Foyer trajectory and the

Office trajectory. Three alternatives are compared in this experiment. The first alternative is to use a standard EKF without any sub-mapping. This cannot maintain real-time performance through a large sequence due to the quadratic update complexity in the number of features; however, it will be used as a baseline comparison. It should always generate a map which has less uncertainty than either of the sub-mapping techniques since no links are removed between features.

The second alternative which is evaluated in this experiment is to use a strict capacity limit of 30 features per sub-map. This alternative should be similar to the implementation of [6], with one minor detail changed – this comparison will use the same matching and correction technique for links between submaps as our new algorithm. There is no use of shared features and a global optimization step. The third alternative is the method described in this paper; normalized graph cuts sub-map partitioning. These alternatives are compared by the median landmark uncertainty at every frame, after the first partition is made.

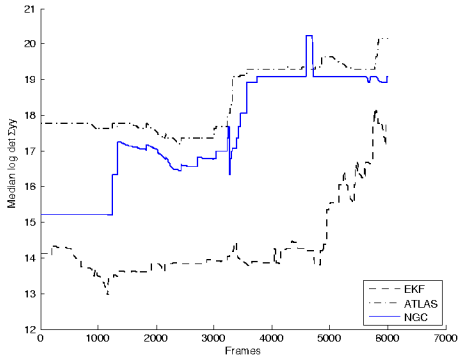

Fig. 3. Median feature uncertainty with various sub-mapping options. RIM Center "foyer trajectory"

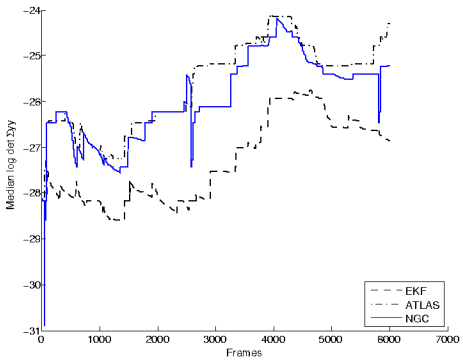

Fig. 4. Median feature uncertainty with various sub-mapping options. RIM Center "office trajectory"

The median landmark uncertainty is shown in figure 3 and figure 4. The initial frames are omitted due to the fact that all three methods are equivalent before the first partition, and the median landmark uncertainty is erratic when there are few landmarks in the map. The median landmark uncertainty is computed in the initial robot frame by composing the landmark and frame location uncertainty. Both trajectories consistently show that the normalized graph cuts partition choice results in a more confident map while maintaining consistency – without becoming more confident than a full EKF would. This result is not surprising since we have stated that Atlas makes partition choices arbitrarily, and so it cannot expect to perform better than a partition which removes the least information from the map possible by performing normalized graph cuts.

TABLE I
EXPERIMENTAL RESULTS, MEDIAN LANDMARK UNCERTAINTY AT END OF RUN * 1E-10

Run#	EKF	Atlas	NGC	Improvement%
1	.0369	.14	.0727	48.02%
2	.012	.187	.111	40.69%
3	.0435	2.97	2.87	3.49%
4	.00969	1.55	1.44	6.95%

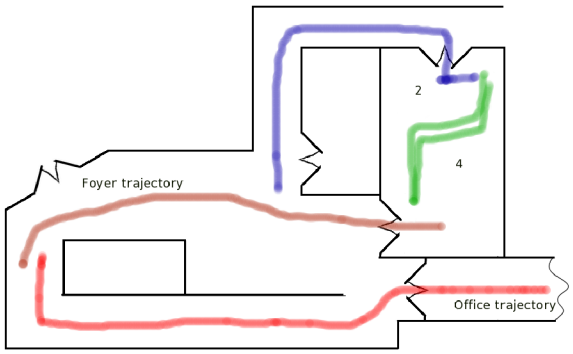

Fig. 5. Approximate path of robot during RIM center tests

Results from additional test runs can be seen in table I. Runs 1 & 2 are test runs in which the robot moves between a room and a hallway; both show significant improvement in landmark uncertainty. Runs 3 & 4 do not show much improvement; however, this is because these runs were carried out exclusively within one room. No architectural boundaries were available to provide superior landmark partitions, so Atlas' arbitrary cuts were just as good as the normalized graph cut. See figure 5 for a visualization of the approximate path of the robot in the RIM center tests. Runs 1 & 3 took place in our old building; they are comparable to Runs 2 & 4 respectively in terms of architectural boundaries crossed.

The normalized graph cuts implementation allows for significantly more certain feature localization compared to arbitrary capacity cuts. This result demonstrates that normalized graph cuts offers significant advantage over arbitrary cut choices as implemented in standard Atlas.

V. CONCLUSIONS AND FUTURE WORKS

A. Conclusions

Traditional approaches to SLAM are limited by a quadratic complexity. To address this problem, sub-mapping approaches have been suggested. These methods typically apply a heuristic for the partition selection. In this paper we have presented the use of normalized graph cuts as a methodology for the partitioning of sub-maps. The application of normalized graph cuts for map partitioning has been demonstrated in the context of monocular SLAM. The proposed method has been implemented in an extension to SceneLib. Experiments using both simulation and real data clearly indicate that this method generates maps superior to ad-hoc partitioning strategies. Consequently, the method provides a theoretically sound basis for SLAM using sub-maps and as such, directly addresses the complexity challenge.

B. Future Works

Other research groups have established excellent progress in sub-mapping with the *full SLAM*, so one might wonder why we would continue to pursue EKF based approaches to SLAM sub-mapping. The answer to this is that our technique enables a robot to not only partition out *independent* features in a map from one another, but also potentially establish which features are more *dependent*. Our long-term research goal is to establish a hierarchy of features to be used for SLAM, with simple visual features on the lowest level and objects on a higher level and perhaps rooms or buildings on the highest level, inspired by Kuipers' Spatial Semantic Hierarchy [19]. One way of determining which features might admit a higher level representation would be to use this normalized graph cuts algorithm to partition low level features according to shared information and then place a higher level representation on this set of features which will reduce the complexity of the mapping task. These covariant features offer little marginal improvement in the robot's localization task since they are highly correlated with the other features, so they can be safely grouped into a higher level construction. Single cluster partitioning from [23] could be used to identify objects or other highly correlated regions which could support segmentation and representation with a higher level object model.

VI. ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of the ARL CTA MAST project, and Alex Makarenko for his helpful suggestions.

REFERENCES

[1] T. Bailey and H. Durrant-Whyte. Simultaneous localisation and mapping (SLAM): Part II state of the art. *Robotics and Automation Magazine*, September 2006.

[2] J. L. Blanco, J. A. Fernandez-Madrigal, and J. Gonzales. Toward a unified Bayesian approach to hybrid metric-topological SLAM. In *IEEE Transactions on Robotics*, volume 24, April 2008.

[3] J. L. Blanco, J. Gonzales, and J. A. Fernandez-Madrigal. Consistent observation grouping for generating metric-topological maps that improves robot localization. pages 818–823, May 2006.

[4] M. Bosse, P. Newman, J. Leonard, and S. Teller. An *Atlas* framework for scalable mapping. *International Conference on Robotics and Automation*, 2003.

[5] R. Chatila and J.P. Laumond. Position referencing and consistent world modeling for mobile robots. *International Conference on Robotics and Automation*, 1985.

[6] L. A. Clemente, A. J. Davison, I. Reid, J. Neira, and J. D. Tardós. Mapping large loops with a single hand-held camera. *Robotics: Science and Systems*, 2007.

[7] J. Crowley. World modeling and position estimation for a mobile robot using ultra-sonic ranging. *International Conference on Robotics and Automation*, 1989.

[8] A. J. Davison. *Mobile Robot Navigation Using Active Vision*. Ph.d. thesis, University of Oxford, 1998.

[9] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. *International Conference on Computer Vision*, 2003.

[10] A. J. Davison and D. W. Murray. Simultaneous localisation and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[11] A. J. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

[12] F. Dellaert. Square root SAM: Simultaneous localization and mapping via square root information smoothing. In *Robotics: Science and Systems*, 2005.

[13] H. Durrant-Whyte and T. Bailey. Simultaneous localisation and mapping (SLAM): Part I the essential algorithms. *Robotics and Automation Magazine*, June 2006.

[14] C. Estrada, J. Neira, and J. D. Tardós. Hierarchical SLAM: Real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4):588–596, August 2005.

[15] J. Folkesson and H. Christensen. Graphical SLAM - a self-correcting map. *International Conference on Robotics and Automation*, 2004.

[16] J. E. Guivant and E. M. Nebot. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17(3), June 2001.

[17] M. Kaess, A. Ranganathan, and F. Dellaert. Fast incremental square root information smoothing. In *Internation Joint Conference on Artificial Intelligence*, 2007.

[18] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, (Forthcoming), 2008.

[19] B. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119(1-2):191–233, May 2000.

[20] J. M. M. Montiel, J. Civera, and A. J. Davison. Unified inverse depth parameterization for monocular SLAM. *Robotics: Science and Systems*, 2006.

[21] J. Neira and J. D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, Dec 2001.

[22] K. Ni, D. Steedly, and F. Dellaert. Tectonic SAM: Exact; out-of-core; submap-based SLAM. In *International Conference on Robotics and Automation*, 2007.

[23] E. Olson, M. Walter, S. Teller, and J. Leonard. Single-cluster spectral graph partitioning for robotics applications. In *Robotics: Science and Systems*, June 2005.

[24] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[25] R. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5(4):56–68, Winter 1987.

[26] J. D. Tardós, J. Neira, P. Newman, and J. Leonard. Robust mapping and localization in indoor environments using sonar data. *International Journal of Robotics Research*, 2002.