An *Atlas* Framework for Scalable Mapping*

Michael Bosse Paul Newman John Leonard Seth Teller
Massachusetts Institute of Technology
{ifni,pnewman,jleonard,seth}@mit.edu

October 18, 2002

Abstract

This paper describes *Atlas* , a new framework in which existing small-scale simultaneous localization and mapping (SLAM) algorithms can be used to perform real-time mapping and navigation in large environments. Our approach combines the advantages of topological and metrical representations. The representation is a graph of coordinate frames, with each vertex in the graph representing a local frame, and each edge representing the transformation between adjacent frames. In each frame, we build a map that captures the local environment and the current robot pose along with the uncertainties of each. Each map's uncertainties are modeled with respect to its own frame. Probabilities of entities with respect to arbitrary frames are generated by following a path formed by the edges between adjacent frames, computed via Dijkstra's shortest path algorithm. Loop closing is achieved via an efficient map matching algorithm. We demonstrate the technique running in real-time in a large indoor structured environment (2.2 km path length) with multiple nested loops using both laser and ultrasonic ranging sensors. We also present results involving the execution of multiple passes through a modest sized environment. This experiment illustrates steady state performance and the increase in certainty in both local regions and edge estimates achieved by revisiting mapped areas.

1 Introduction

A scalable concurrent mapping and navigation algorithm for a mobile robot should be able to operate in an environment of any size. For a practical implementation of such an algorithm, the total computation must be proportional to the total amount of sensor measurements and independent of the size and complexity of the mapped region. This then implies constant time performance per iteration.

This paper describes *Atlas* — a technique that manages a graph of multiple local maps of limited size and expends constant time per observation processed. *Atlas* maintains a finite set of robot location hypotheses to determine which map, if any, best fits the perceived local environment. We demonstrate the technique running in real time in a large indoor structured environment (2.2 km path length) with multiple nested loops using both laser and ultrasonic ranging sensors. We also present results illustrating both inter- and intra-map convergence.

We aim to develop a recursive algorithm enabling a mobile robot to map and localize within an unknown environment of any size. To realize this goal we seek to minimize the computation required per sensor observation and its dependency on the complexity and size of the environment. Hence we require a scalable method for egomotion and environment capture — an important goal in computer vision and robotics.

*M. Bosse and S. Teller are at the Laboratory of Computer Science at MIT, and P. Newman and J. Leonard are at the Department of Ocean Engineering at MIT.

There is now a substantial corpus of knowledge in the fields of Simultaneous Localization and Mapping (SLAM) and Structure from Motion (SFM) for local, small-scale regions [Fau93, BZ99, Thr01, PKVG00, WC00, Jen01]. While such methods have been applied with success in small, controlled environments, they struggle when applied to larger environments. Many application domains require mapping of extended environments (thousands of square meters to tens of square kilometers in area) with long sensor excursions and closing of large loops [TAB+01].

Before proceeding, we make the following motivating observations and discuss their implications.

- Observations of spatially distant map elements can be largely decoupled due to lack of co-observability [GN01].
- Methods whose complexity growth is unbounded are untenable [SSC90].
- Many applications demand only egomotion estimates with respect to the local environment [Dav98, Kui00, GN01].
- The error integrated around a long loop may be large enough to prevent the recognition of an already mapped region — i.e., loop closing is hard [Thr01].
- We expect to encounter featureless regions in which navigation must rely on dead reckoning alone [New02].

Building a single monolithic map results in an un-mangaged growth of complexity and computational burden. This in combination with the fact that spatially distant features can be decoupled has led several researchers [CK97, LF01, BN01, WDDW02, TNNL02] to suggest partitioning the problem into sub-regions or sub-maps. Indeed the issue of complexity growth is also tackled by this approach. Given our desire to map a region of unlimited size, an un-managed growth of complexity with the size of the environment will prohibit a real-time implementation. Furthermore if a single high dimensional pdf is used to represent the entirity of the robot's environment (for example a single Gaussian pdf in a feature based approach) adverse non-linear effects stemming from angular error grow with environment size. These effects can be mitigated by using multiple lower dimension pdfs with a smaller spatial extent. It remains unclear, however, how to manipulate a set of sub-maps in a consistent manner.

Given that many applications require only a representation of the local region [NBL02, BK89], we may derive our output from a single sub-map without recourse to an absolute global coordinate frame.

Since it is often not available, we assume no external infrastructure and limit ourselves to purely proprioceptive and vehicle-relative sensing. For example, we permit accelerometers, gyros, odometry, radar/laser/sonar rangers, and cameras, but not GPS or a magnetic compass.

Although loop closing is difficult, it is a prerequisite for a useful system. Without detection of loop closure, revisited regions will be mapped multiple times. Similarly, it is important to be able to relocalize within an existing map after transiting a feature-sparse region. This difficulty arises due to the lack of absolute global sensor measurements, such as GPS fixes, which are drift-free.

1.1 Contributions

A complete implementation of large-scale mapping requires the integration of feature representation and detection, uncertainty modeling, data association and scaling techniques. This paper focuses on scaling. We describe a method in which existing small-scale mapping algorithms can be applied to large-scale problems via a new framework called *Atlas*.

The *Atlas* framework has the following novel aspects:

- We maintain no single, global coordinate frame but rather an interconnected set of local coordinate frames. Each frame contains a map, which we refer to as a *map-frame*. We consider two coordinate frames to be connected (adjacent) if their map-frames possess shared structure; for example, common mapped features. This adjacency is represented by a coordinate transformation between frames.
- After recognizing the closure of an extended loop, we do not constrain the composition of adjacency transformations to be the identity transformation. This is essential to achieving constant time performance in the *Atlas* framework since no global updates are required. (Nevertheless, the identity constraint can be applied off-line to refine the global arrangement of the multiple coordinate frames; this is illustrated in Figure 10 where the constraint has been applied for visualization purposes. This will be discussed further in Section 3.7.)
- The spatial extent of the map-frames is not predefined but rather determined by an intrinsic metric of the contained map. The same metric is used to invoke either a transition to an adjacent frame or the genesis of a new one.
- The computational cost of the algorithm is constant for processing the data from each time step. A limit is placed on the per-map computation by defining a measure of complexity for each map-frame, which is not allowed to exceed a threshold (the map capacity). Rather than operate on a map of ever-increasing complexity, the *Atlas* framework simply switches its focus to a new or adjacent map-frame. Additionally, the computational cost of deciding whether to transition to an adjacent map-frame or spawn a new one is also bounded; thus the total computation is constant time.

1.2 Roadmap

In Section 2 we provide an overview of the *Atlas* framework, which highlights its key components. Section 3 discusses these components in detail, illustrating them with the example of a feature-based implementation, for which results using sonar and laser are given in Section 4.

2 Atlas Framework

In essence, the *Atlas* framework is just a graph of coordinate frames. Each vertex in the graph represents a local coordinate frame, and each edge represents the transformation between adjacent local coordinate frames.

In each local coordinate frame, we build a map that captures the local environment and the current robot pose along with the their uncertainties. We refer to this couple as a map-frame.

Throughout this paper,we provide an example utilizing a feature-based representation of the environment. In this case, the estimate of the location of the robot and the features is a point in a high-dimensional parameter space estimated by a Kalman Filter [SSC90]. For completeness, a brief summary of this technique is provide in Appendix A. Our choice of representation is made without prejudice. The local map could also be represented with other methods, such as a particle filter or an occupancy grid [DFBT99, MC89].

We entertain the possibility that any one of several map-frames can best explain the recent observations of the robot. Hence we maintain a set of *map-frame hypotheses*, each independently processing the sensor measurements in turn. Figure 1 depicts a graph of map-frames in which two hypotheses are maintaining the robot position.

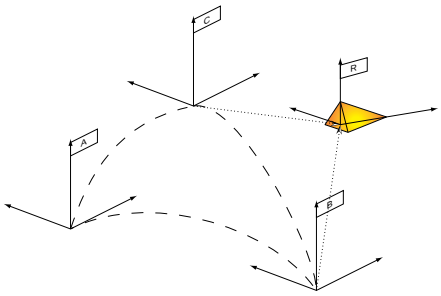


Figure 1: A graph of map-frames in which two hypotheses (B and C) maintain the robot position (R).

The uncertainty of the edges (adjacency transformations) in the *Atlas* graph are represented by a Gaussian random variable. In contrast to Chong and Kleeman [CK97], the graph's edges are undirected. In other words, there is no preferred direction to an edge; the reverse direction of an edge contains the inverse transformation.

In contrast to Decoupled Stochastic Mapping [LF01], each map's uncertainties are modeled with respect to its own frame. In other words, the pdf's representing uncertainties of the robot and the map are conditioned on the local coordinate frame.

We can still, however, generate probabilities of entities with respect to arbitrary map-frames by following (transforming by) a path formed by the edges between adjacent map-frames. We compute such paths via Dijkstra's shortest path algorithm [Dij59] using the uncertainties of the transformations as a statistical distance metric.

2.1 Requirements

The successful implementation of the *Atlas* framework requires the following two conditions to hold.

Condition 1: The *Atlas* framework needs a good local navigation method that can produce a consistent map.

Condition 2: The navigation uncertainty acquired around a loop before its closing must be small enough to avoid ambiguous map-matches.

The quality of the navigation and mapping produced by the *Atlas* framework can be no better than that provided by the local mapping scheme. The *Atlas* framework simply extends existing techniques so that they can be applied to large-scale regions. Condition 1 is a reflection of this point.

Condition 2 implies that ambiguous matches are tolerable as long as the predicted robot position is accurate enough to distinguish between multiple matches. This requirement limits the length of loops that the *Atlas* framework can handle based on the balance between the local navigation uncertainty accumulation rate and the repetitiveness of the environment.

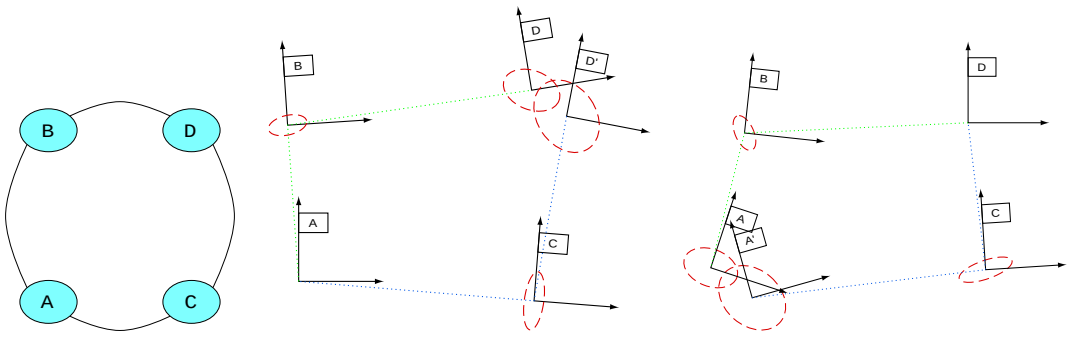


Figure 2: The Dijkstra Projection using two different source nodes. (a) depicts the topological arrangement of the *Atlas* graph. (b) uses map-frame A as the source of the projection. (c) uses map-frame D as the source. The ellipses on the coordinate frames represent the accumulated projection error. The shortest path from map-frame A to D is clearly via map-frame B.

3 Atlas Components

This section provides a detailed description of the six core concepts of the *Atlas* framework: uncertainty projection, competing hypotheses, creation of new map-frames (Genesis), closing loops (Map-Matching), instantiating and evaluating new hypotheses in adjacent map-frames (Traversal), and transformation edge refinement. These six components are now discussed.

3.1 Uncertainty Projection

Atlas edges contain the information necessary to relate two map-frames. The uncertainty of the transformation edge is used to project a stochastic entity (such as the robot position) from one map-frame into another. However, if the map-frames are not adjacent, these transformations (and their uncertainties) must be composed along a path of edges that link the *Atlas* vertices. There may be more than one path from one vertex to another. Since these cycles are are not constrained online, distinct paths will not in general produce the same composite transformation. In Figure 2(a), frame D is reachable from A via B or C, resulting in the two possible projections of frame D relative to A, shown in Figure 2(b).

To resolve this ambiguity we use Dijkstra's shortest path algorithm [Dij59] to find a unique path between the nodes. Instead of using a physical distance to compute the length of the path, we use a statistical metric, ρ, based on the uncertainty of the transformation in *Atlas* edges. The metric we choose is the determinant of the covariance matrix of the composite transformation.

$$T_a^c = T_a^b \oplus T_b^c \tag{1}$$

$$\Sigma_{ac} = \mathrm{J}_1\left(T_a^b, T_b^c\right)\Sigma_{ab}\mathrm{J}_1\left(T_a^b, T_b^c\right)^T + \mathrm{J}_2\left(T_a^b, T_b^c\right)\Sigma_{bc}\mathrm{J}_2\left(T_a^b, T_b^c\right)^T \tag{2}$$

$$\rho = \|\Sigma_{ac}\| \tag{3}$$

We define the Dijkstra Projection of an *Atlas* graph with respect to a given source vertex as the global arrangement of frames using compositions along Dijkstra shortest paths. This projection has the property of transforming the *Atlas* graph into a tree of transformations with the source map-frame as the root. We measure the nearness of any map-frame to the source frame as ρ computed from the compositions of transformations up the tree to the root.

3.2 Competing Hypotheses

At any given time, there are several competing map-frame hypotheses that attempt to explain the current robot pose and sensor observations. There can only be one hypotheses per map-frame. We verify the validity of each map-frame's hypothesis by monitoring a performance metric $q \in [0 \rightarrow 1]$ for a few time steps.

The metric q depends on the hypothesis's map-frame $\mathcal{M}_i$, robot pose estimate $\mathbf{x}_i$, and recent sensor measurements Z.

$$q_i \quad = \quad q(\mathcal{M}_i, Z, \mathbf{x}_i) \quad \in \quad [0 \rightarrow 1] \tag{4}$$

For example, a suitable form of q for a feature-based approach is

$$q \quad = \quad \alpha \left(1 - \frac{\|\Sigma_{\mathbf{x}_i}\|}{\|\Sigma_{\mathbf{x}_\star}\|}\right) + (1-\alpha)\frac{\eta_a}{\|Z^k\|} \tag{5}$$

where η_a is the number of matched observations, and $\|Z\|$ is the total number of recent observations. The parameter $\alpha \in [0 \rightarrow 1]$ reflects the relative importance placed on the robot uncertainty and successful explanation of sensor data. If in 2D, we specify maximum acceptable uncertainties in location and orientation, $\sigma_x, \sigma_y, \sigma_\theta$, then $\|\Sigma_{\mathbf{x}_\star}\|$ is the product $\sigma_x^2\sigma_y^2\sigma_\theta^2$.

At any given time, there may be several active hypotheses in an *Atlas* graph. Each map-frame can support only one hypothesis at a time, and the maximum number of total hypotheses, $\mathcal{H}_m$, is fixed so that the computational requirements remain bounded. If the number of potential hypothesis is greater than $\mathcal{H}_m$, then they are instantiated only when existing hypothesis are terminated. In practice this will only occur in highly interconnected regions of the *Atlas* graph.

3.3 Genesis

Since we bound the complexity (for example the number of features) of each map, when we enter unexplored regions we need to create new local map-frames. Genesis process adds a new vertex and edge to the *Atlas* graph. Mathematically, the generation of a new map-frame $\mathcal{M}_j$ and robot pose $\mathbf{x}_j$ via Genesis is a function of an old map-frame $\mathcal{M}_i$ and robot pose $\mathbf{x}_i$:

$$(\mathcal{M}_j, \mathbf{x}_j) \; = \; g(\mathcal{M}_i, \mathbf{x}_i). \tag{6}$$

The process of Genesis encapsulated by the function g is broken down as follows:

1. The current robot pose defines the origin of a new frame. Thus, the transformation from the old to the new frame is simply the robot's position in the old frame at the time the new frame is created.
2. The robot pose is initialized to zero in the new frame.
3. The uncertainty of the transformation is set to the uncertainty of robot pose in the old frame.
4. The uncertainty of the robot pose in the new frame is zero by definition. All of the uncertainty of the robot pose at the time of Genesis is captured by the uncertain transformation.

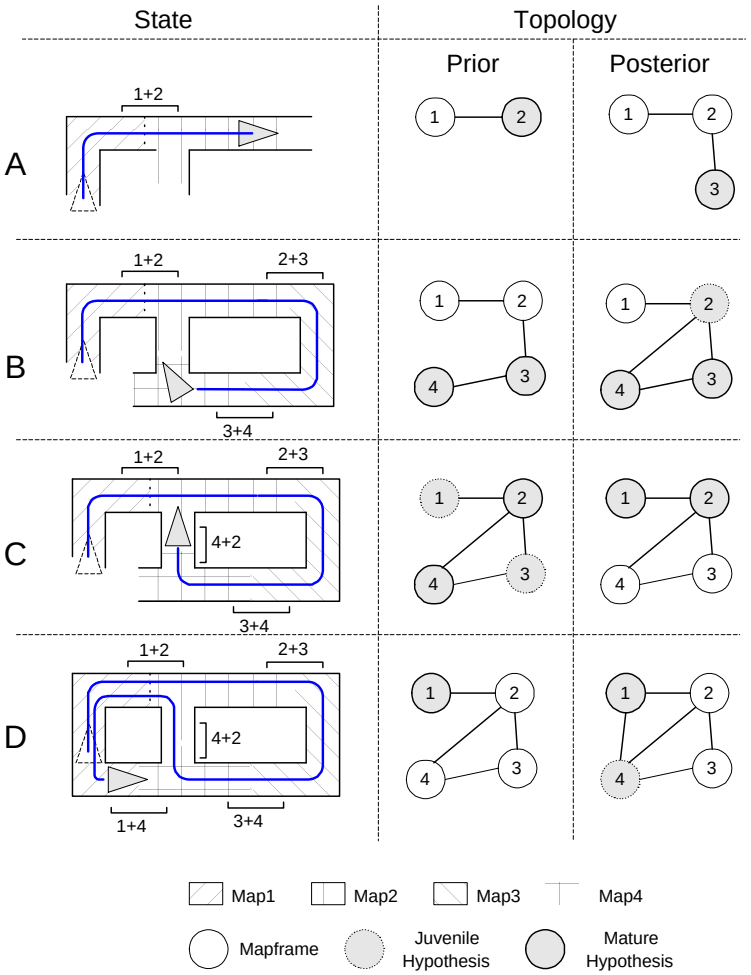


Figure 3: The spatial and topological evolution of an Atlas. This set of diagrams highlight key aspects of the Atlas framework. The environment depicted is fictional and the robot path is construed to be illustrative. The right hand column of this diagram uses shading to indicate the map-frames which succesfully represent the local region. Note that contrary to that suggested by the shading, the maps do in fact overlap. This intersection is denoted by labeled brackets on the spatial diagrams. Note also that the extent of a map-frames is not defined or bounded by area covered but by the set of mapped features *within* it.

A Genesis: The vehicle has built two maps (1,2). Map 1 no longer explains the surroundings and is inactive. Map 2 is "full" and so the genesis of Map 3 occurs. An edge is built between 2 and 3. Map 3 immediately becomes the dominant hypothesis.

B Map Matching: Mature hypotheses are running in both maps 3 and 4. The Dijkstra projection suggests that map 4 may be "close" to map 2. The map matching algorithm confirms this conjecture and a new link is created between 4 and 2. This is loop closure. Map-frame 2 is now adjacent to a mature frame (4) and so shortly after the edge creation a juvenile hypothesis is attached to it.

C Hypothesis Cull: The robot has recently traversed into map 2 (from 4) which has also become dominant. Juvenile hypothesis have been installed in the adjacent frames 1 and 3. A short time later the juvenile hypothesis in map 3 has been terminated - it cannot adequately explain the central corridor. The previously mature hypothesis in map 4 has also been culled for the same reasons. The juvenile hypothesis in map 1 however has been promoted to mature status. At this point the estimate of the transformation between frames 1 and 4 can be updated using an observation constructed from the fact that the vehicle is simultaneously in map 1 and 4.

D Loop closure: Initially only one mature hypothesis exists (in map1). Unlike in case **A** genesis is not imminent (low vehicle uncertainty and map is not full). Instead the map matching algorithm conjectures that maps 1 and 4 may be adjacent. The map matching algorithm confirms this and another new link is created. This completes the topological and spatial description of the environment.

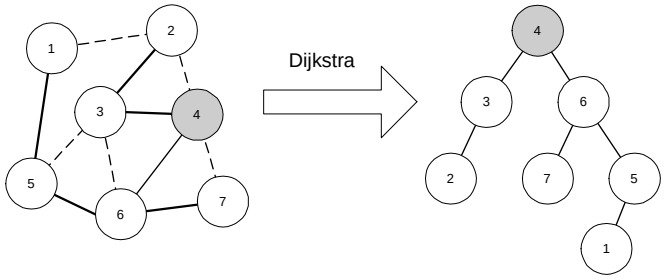


Figure 4: The Dijkstra projection from a given node in a graph transforms the graph into a tree with the source node as a root. Here we are taking node 4 as a source. Solid lines correspond to links that are used in the tree representation. Note how in this example the uncertainty between link 4 and 7 is larger than that accrued via traversing links 4-6 and then 6-7. Hence there is no direct link between 4 and 7 in the tree.

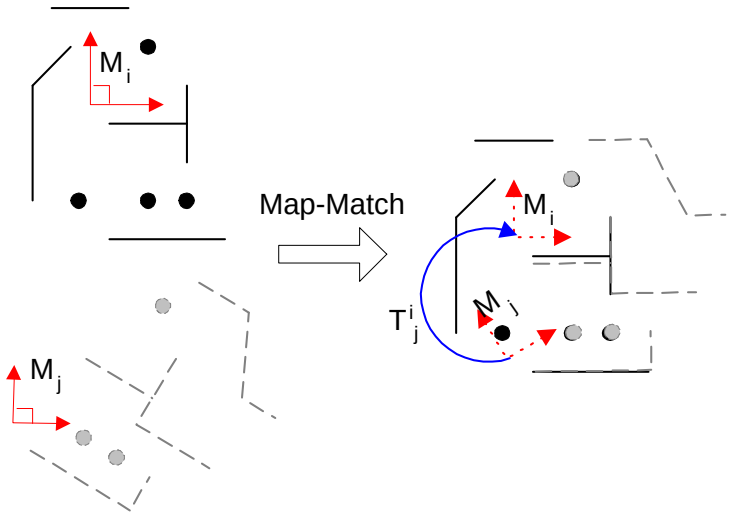
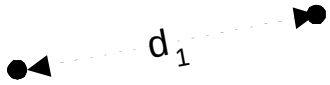
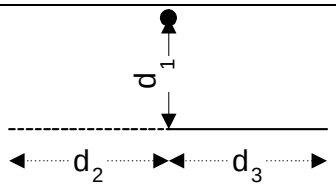
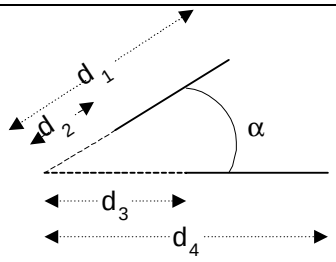


Figure 5: Map-Matching as a search for a transformation between maps that maximally aligns common mapped features. Here two maps i and j share features and a good map match can be found between them. Note how only a subset of features are matched.

Table 1: Defining map signature elements between pairings of line and point features

Pairing	Geometry	Element
Point-Point		$[d_1]$
Point-Line		$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}$
Line-Line		$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ \alpha \end{bmatrix}$

3.4 Map-Matching

Genesis creates new maps to explain unexplored areas. However, eventually the robot will revisit an area that it has already explored. The Map-Matching process detects these situations and forms an edge in the *Atlas* graph between two unconnected vertices (map-frames).

We can describe the Map-Matching process as a search for a coordinate transformation that aligns overlapping map-frames. The Dijkstra projection is used to form the prior estimate of the transformation between two map-frames, but its uncertainty may be very large. Too large, in fact, to be able to rely on the simple strategy of nearest neighbor feature gating for data association. Thus we need a method that is robust to large initial errors in the transformation.

In general terms, Map-Matching is comprised of two steps:

1. A scalar $m_{ij} = m(\mathcal{M}_i, \mathcal{M}_j)$ identifies common structure between two maps $\mathcal{M}_i$ and $\mathcal{M}_j$.
2. A function $t(\mathcal{M}_i, \mathcal{M}_j)$ forms an estimate of the transformation T_i^j and its covariance Σ_{ij} between maps $\mathcal{M}_i$ and $\mathcal{M}_j$.

The exact form of m used for determining common structure is not dictated by the *Atlas* framework. For the example using feature-based maps, we define the operation $\mathcal{M}_i \cap \mathcal{M}_j$ as the search for correspondence between features in $\mathcal{M}_i$ and $\mathcal{M}_j$. We define η_m as the minimum number of correspondences required for m_{ij} to be considered a positive Map-Match.

Following a positive Map-Match between map-frames $\mathcal{M}_i$ and $\mathcal{M}_j$, we compute the estimate of the transformation T_i^j between the coordinate frames of $\mathcal{M}_i$ and $\mathcal{M}_j$, as well as its uncertainty, Σ_{ij}. Note

that t does not depend on the robot pose. This is crucial since the navigation errors accumulated around a loop may be larger than simple data association of sensor measurements can handle.

Repetitive structure in the environment may cause ambiguous edges to be formed by Map-Matching. The degree of repetition can be assessed by Map-Matching a map with itself. Only structure elements that match uniquely within a map should be used to evaluate a match to another map.

We now describe a two stage implementation of the Map-Matching process for a our feature-based example summarized as follows:

1. A signature for both maps is constructed which is an ordered list of elements describing properties of the map that are invariant to translation and rotation of its coordinate frame. A comparison operator is defined over two signatures which yields a set of correspondences between elements in each list.[1]

2. Each map is matched with itself to identify repetitive structure. Any elements that correspond to other elements in the same map are removed from the maps signature. This dramatically reduces the likelihood of false map matches due to repetitive structure by focusing on the unique elements of each local environment.

3. The signatures of both maps are now compared. Each element to element correspondence defines a potential alignment transformation from $\mathcal{M}_i$ to $\mathcal{M}_j$.

4. Each potential alignment transformation is applied to $\mathcal{M}_i$. The validity of each transformation is evaluated by counting the number, η, of feature pairs it brings into alignment with nearest neighbor gating. This is the implementation of function m defined above.

5. The correspondences from the best (largest η) potential transformation with $\eta > \eta_m$ are used to refine the transformation and its covariance. Each correspondence defines a constraint on the transformation. The combined set of η constraints are solved in weighted least-squares sense using the covariances of the feature estimates within each map to form the weights. This process also yields the covariance of the transformation. This step is the implementation of function t as defined above.

In this implementation, our maps consist of 2D point and line features. The elements we use to build a map signature are comprised of pairings of non-parallel lines, point-line pairs and point-point pairs drawn from the map. For each pairing, the signature element consists of distances and/or angles that are independent of the map frame's orientation and location. Table 1 defines the invariants used for the three species of pairings.

The number of signature elements to compare when matching maps is of $O(n^2)$ which may lead to $O(n^4)$ matches that need to be performed. However, we can reduce the number of matches that need to be tested to $O(n^2)$ by sorting the signature elements into a canonical order, which then reduces the total computational burden to $O(n^2 \log n)$.

The approach we have adopted for Map-Matching is not unique. For example the Joint Compatibility test with branch and bound technique suggested by [NT01] could also be utilized. The freedom to choose a Map-Matching strategy highlights the modularity of the *Atlas* framework.

[1] The sorting and comparison of elements requires the definition of "less than", "proximity" and "equivalence" tests between elements.

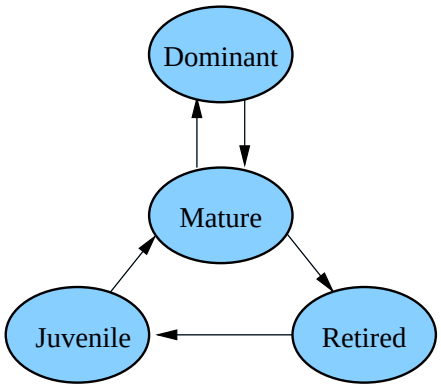


Figure 6: *Atlas* hypothesis state transition diagram. The dominant hypothesis is used to provide an output from the algorithm. It is simply the most successful of all the mature hypotheses. Mature hypothesis are able to extend maps and spawn juvenile hypotheses in adjacent map frames. When a mature hypothesis fails to describe the robot's environment adequately (the robot has moved away for example) it is retired. It can be reinstated as a juvenile at some time in the future by an adjacent mature hypothesis. Juveniles are not allowed to modify their map. If after a probationary period a juvenile's map is failing to explain sensor data it is deleted. However a successful juvenile is promoted to mature status.

3.5 Traversal

Once the robot has mapped an area, it can reuse previously built maps when the current frame is no longer adequate. When the robot leaves the area around which a map-frame was created, it will fail to match sensor measurements, thus its performance metric q (equation 4) will tend to zero, indicating poor performance. Therefore in a continuous attempt to find the best explanation of the sensor measurements, we run hypotheses in adjacent map-frames by projecting the current robot pose and uncertainty across the transformation edges in the local neighborhood of the current *Atlas* vertex.

There are four types of hypotheses, which we label as juvenile, mature, dominant and retired. See Figure 6 for a state transition diagram. All types, except retired ones, process sensor measurements and evaluate the same performance metric q. Mature hypotheses can extend their maps (increase the map's complexity) and spawn new hypotheses. Juvenile hypotheses can only process sensor data with regard to the existing map. Juvenile hypotheses are restricted from extending their maps because they are used to test how well a particular map explains the sensor data and not whether a new map could be built from the data.

A juvenile hypothesis can "mature" when after a probationary period its performance metric q becomes greater than any other mature hypothesis. If at the end of this probationary period a juvenile hypothesis' quality does not warrant promotion it is simply deleted.

The mature hypothesis with the best performance metric is considered the dominant hypothesis. The dominant hypothesis is used for publishing current robot pose and local features to clients of the *Atlas* framework. In other words, it is the output of the framework.

Mature hypotheses that fail to perform well are saved and "retired". A retired hypothesis may be reactivated at a later time as a juvenile.

If we have only one mature but failing hypothesis, then a new one must be created to explain current sensor data. This is done by Genesis (Section 3.3). This situation will occur when the robot moves into an unexplored area.

The initialization of a juvenile hypothesis and its graduation to maturity warrant further description. When creating a juvenile hypothesis in a retired map-frame $\mathcal{M}_i$ we reinitialize its robot pose $\mathbf{x}_i$ using the robot pose $\mathbf{x}_j$ from an adjacent map-frame $\mathcal{M}_j$. (See Figure 7.) First we seed the hypothesis with a robot pose $\mathbf{x}_j^\star$ projected into frame i.

$$\mathbf{x}_i^\star = T_i^j \oplus \mathbf{x}_j \tag{7}$$

$$\Sigma_{\mathbf{x}_i}^\star = \mathrm{J}_1\left(T_i^j, \mathbf{x}_j\right) \Sigma_{ij} \mathrm{J}_1\left(T_i^j, \mathbf{x}_j\right)^T + \mathrm{J}_2\left(T_i^j, \mathbf{x}_j\right) \Sigma_{\mathbf{x}_j} \mathrm{J}_2\left(T_i^j, \mathbf{x}_j\right)^T \tag{8}$$

Where $\mathrm{J}_1(\cdot,\cdot)$ and $\mathrm{J}_2(\cdot,\cdot)$ are the Jacobians of the transformation composition operators as defined in Appendix B.

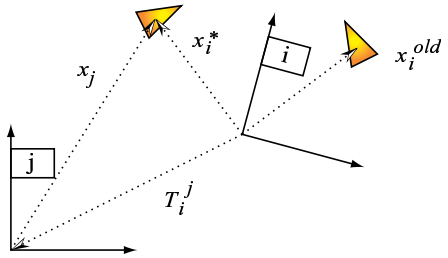


Figure 7: Seeding the robot position for juvenile hypothesis in frame i using the current pose in the adjacent map-frame j. The retire hypothisis attached to frame i has the robot location at $\mathbf{x}_i^{old}$. The hypothesis is rejuvenated to have the robot pose of $\mathbf{x}_i^\star$.

The hypothesis now enters a bootstrapping phase, in which a consistent initialization of the vehicle into the juvenile hypothesis is sought. Sensor measurements, interpreted with the seeded robot pose, $\mathbf{x}_i^\star$, are accumulated. This continues until we have collected enough measurements, Z, to solve explicitly for the robot pose independently of $\mathbf{x}_i^\star$; we call this function w. This approach conserves the statistical independence of map-frames.

$$(\mathcal{M}_i, \mathbf{x}_i^{\text{new}}) = w(\mathcal{M}_i, Z) \tag{9}$$

If an explicit solution to w cannot be computed because of lack of explained sensor measurements, then the hypothesis is invalid and terminated. Otherwise we have a tenable juvenile hypothesis.

3.6 Edge Refinement

When there is more than one mature hypothesis, we can refine the estimate of the transformation between them — the *Atlas* graph edge. Consider the case of two mature hypotheses in adjacent map-frames, $\mathcal{M}_i$ and $\mathcal{M}_j$. Both maintain an estimate of the current robot pose, $\mathbf{x}_i$ and $\mathbf{x}_j$ respectively. In combination they form a measurement of the transformation T_i^j:

$$T_i^j \oplus \mathbf{x}_j = \mathbf{x}_i \tag{10}$$

$$T_i^j = \mathbf{x}_i \ominus \mathbf{x}_j \tag{11}$$

From Equation 11, we can write the covariance of the observation, Σ_{ij}, as a function of the robot uncertainties $\Sigma_{\mathbf{x}_i}$ and $\Sigma_{\mathbf{x}_j}$.

$$\begin{aligned} M_i &= \mathrm{J}_1\left(\mathbf{x}_i, \ominus\mathbf{x}_j\right) \\ M_j &= \mathrm{J}_2\left(\mathbf{x}_i, \ominus\mathbf{x}_j\right) \mathrm{J}_{\ominus}\left(\mathbf{x}_j\right) \\ \Sigma_{ij} &= M_i \Sigma_{\mathbf{x}_i} M_i^T + M_j \Sigma_{\mathbf{x}_j} M_j^T \end{aligned} \tag{12}$$

We update the prior estimate $T_i^{j^-}$ (the existing graph edge) with the observation T_i^j to form the refined estimate $T_i^{j^+}$, and its uncertainty Σ_{ij}^+. Since we do not maintain the cross-covariances between robot estimates in different maps, we advocate the use of Covariance Intersection [JU97] to perform the update.

$$\Sigma_{ij}^+ = \left[\omega\left(\Sigma_{ij}\right)^{-1} + (1-\omega)\left(\Sigma_{ij}^-\right)^{-1}\right]^{-1} \tag{13}$$

$$T_i^{j^+} = \Sigma_{ij}^+\left[\omega\Sigma_{\mathbf{x}_i}^{-1}\mathbf{x}_i + (1-\omega)\Sigma_{\mathbf{x}_j}^{-1}\mathbf{x}_j\right] \tag{14}$$

Where

$$\omega = \arg\min_{\omega} \left\|\Sigma_{ij}^+\right\|. \tag{15}$$

If the uncertainty in local maps decreases, then with each map transition we can also improve our estimate of the transformation between frames. Therefore not only do we achieve local convergence, but inter-map certainty improves as well.

3.7 Obtaining a global map

We are often motivated to provide a single global map of the robot's environment. For example in Section 4 we compare an estimated map with an architectural drawing. This "globalized" representation is a result of a post processing procedure to find a global projection of each map-frame. In other words, we wish to find the position and orientation of each map-frame with respect to a single frame. We choose to reference all maps to the first map-frame created, frame 0.

The Dijkstra projection does this when using map-frame 0 as the source, however it only uses a minimal subset of the edges in the graph. We wish to find a projection that incorporate all the edges.

When there are loops in the graph, there will be a disparity $\nu_{i,j}$ between the transformation T_i^j stored in the *Atlas* graph edge and the transformation derived from the global poses of each frame (T_0^i and T_0^j respectively).

$$\nu_{ij} = T_i^j \oplus T_j^0 \oplus T_0^i \tag{16}$$

We seek to find the global arrangement $\mathcal{T}^*$ of all N frames $\mathcal{T} = \{T_0^1 \cdots T_0^N\}$ that minimizes this error over all edges. This can be posed as a non-linear least squares optimization problem:

$$\mathcal{T}^* = \arg\min_{\mathcal{T}} \sum_{ij} \|\nu_{ij}\|^2 \tag{17}$$

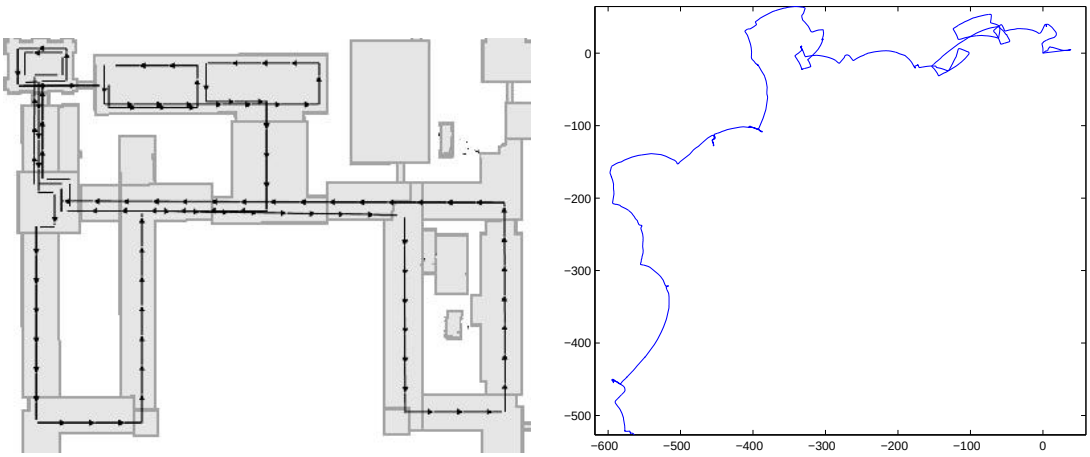
We use the Dijkstra projection to compute the initial global arrangement, and the optimization typically converges in less than 5 iterations using the Matlab optimization toolbox.

4 Results

4.1 Mapping the "Infinite Corridor"

This section describes and presents results of a lengthy experiment conducted within and around MIT's "Infinite Corridor". These results constitute a compelling demonstration of the constant-time, large-scale properties of the *Atlas* framework. The experiments utilized a standard B21 mobile robot equipped with SICK scanning laser and a ring of 24 Polaroid ultrasonic sensors. The results presented here, using both sonar and laser sensors along with odometry, are from a real-time C++ implementation of the *Atlas* framework using an Extended Kalman Filter for local navigation. Onboard odometry was the only other source of information used.

Figure 8(a) shows the topological path of the vehicle superimposed on an architectural drawing of the MIT main campus. The route was chosen to highlight the key capabilities of the *Atlas* framework — constant time processing and re-use of previously mapped regions. Firstly, the total path length and experiment duration was substantial — driving a little under 2.2 km in 2.5 hours. Secondly, the route contains nested loops of various sizes and topologies. Successful processing of this data set necessarily involves "loop closing" — a notoriously hard aspect of the SLAM problem.



(a) (b)

Figure 8: (a) The manually drawn topology of the driven route overlaid on an architectural drawing of part of the MIT campus. The large east-west passage (a.k.a. the "Infinite Corridor") is about 250m long. (b) The trajectory derived from odometry alone.

Figure 8(b) shows the dead-reckoned path resulting from simply integrating the odometry data, which is clearly a poor estimate of the true trajectory. This however is in stark contrast to the outcome of processing the data set using an implementation of the *Atlas* framework with laser and odometry data as shown in Figure 10(a). In all, 125 map-frames were built, each containing a maximum of 10 mapped line segments. The constant time property of the *Atlas* framework when applied to the large-scale SLAM problem is shown in Figure 9. The figure shows the instantaneous sum of kernel and user time for the *Atlas* process as well as its smoothed value. Note that as more features are mapped and more map-frames are created the mean processor load stays constant. Figure 9 also plots the numerical label of the dominant map-frame with time. During map-frame Genesis a counter is incremented and the newly created map is labeled with its value. As more and more new ground is covered, the value of the dominant map ID will tend to increase. When however the robot returns to a previously mapped area, the dominant map ID will decrease if loop closure is successful. For example, approximately one hour

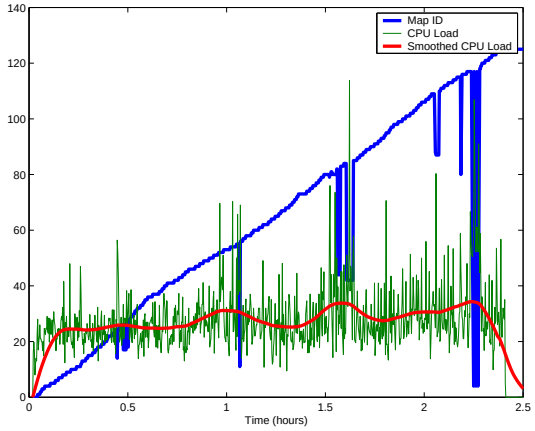


Figure 9: Processor load and current map ID for laser data run. The general linear increase in current map ID is indicative of the mapping of new areas. The occasional "fall-back" to a map with a lower ID represents succesful loop closing —the re-use of an existing map. Crucially the average processor load (thick grey line) stays constant. The occasional local increases in average CPU load preceeds loop closure events — times at which the number of active hypotheses increase. Since a limit is placed on the maximum number of active hypothesis, this effect is bounded and does not impinge on theconstant time nature of *Atlas* .

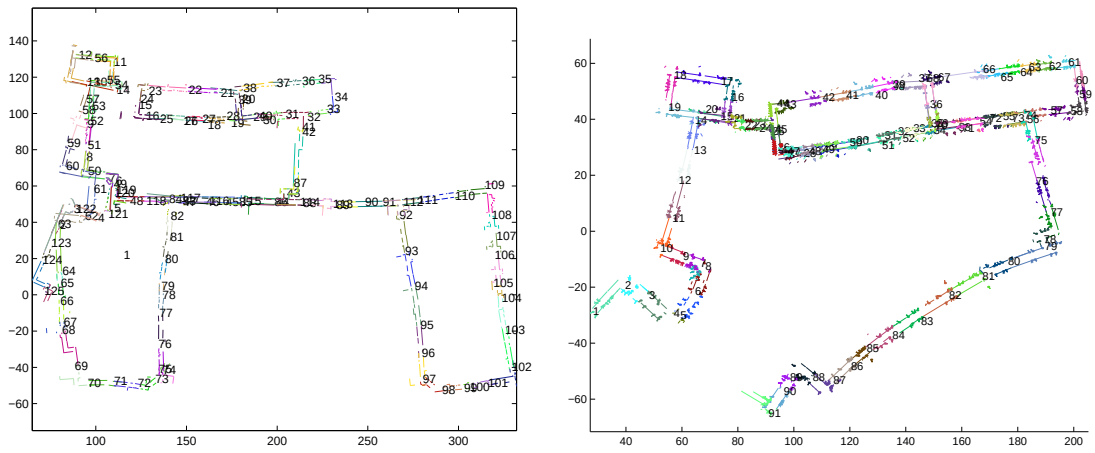
into the experiment the robot returned to an area first mapped 45 minutes earlier. Similarly, after 2hrs 15 minutes the vehicle returned to a region mapped just five minutes after the experiment began.

Figure 10(b) shows results using data from the same experiment but using the Polaroid ultrasonic rangers instead of the laser scanner. The local navigation method used is a combination of Delayed Decision Making [LRNB02], RANSAC [FB80] and wide beam sonar interpretation [LDW92]. The key idea of this approach is to incorporate temporal as well as spatial correlations in the stochastic mapping process. This enables map features to be consistently initialized using data from multiple vantage points, in a process of deferred feature initialization. Updates to the map and the vehicle trajectory can also be performed with groups of data acquired from multiple positions. This allows perceptual grouping techniques developed in computer vision, such as random sample consensus, to be used for correspondence.

The corrective effect of sonar data on angular error is less than that of laser data. This effect prevented the processing of the full data set, resulting in a smaller coverage than for the laser-driven case (although a greater number of features were created). When traversing down a long corridor using sonar alone the growth in angular error was such that condition 2 in section 2.1 was violated — the accumulated uncertainty around the loop that should have been closed was large enough to cause ambiguous map matches. At this point the *Atlas* was halted and results saved.

Figures 11(a) and (b) are similar to Figure 2 and illustrate with real data the effect of representing uncertainty around a loop of map-frames using different Dijkstra projections. Figure 11(a) uses the map in the top left as the source whereas Figure 11(b) shows exactly the same map-frames using only the bottom left one as a source. Note how uncertainty is always greatest across the loop from the source corresponding to the greatest relative Dijkstra distance.

To our knowledge this is the most extensive indoor demonstration of real-time, large-scale SLAM. The experiment was terminated only when the majority of the connected corridors had been explored, resulting in the mapping of 1154 features in 125 maps.



(a) (b)

Figure 10: A visualization of the complete set of numbered map-frames. For the purposes of visualization, a global projection has been applied as described in Section 3.7 (a) for the laser run and (b) for the sonar run.

4.2 Execution of multiple loops

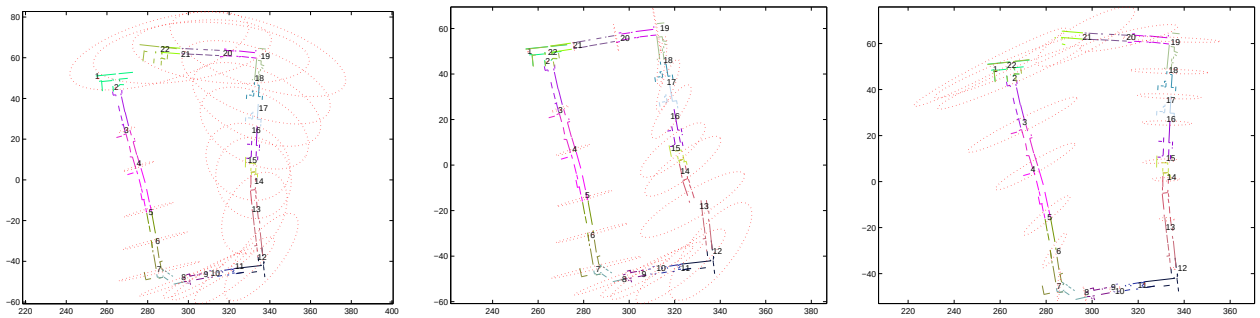
We now present results from a second experiment that illustrates the steady state performance of the *Atlas* framework while navigating in a limited area for a long time. The robot executed multiple "figure eight" maneuvers within the environment depicted in Figure 12(a). Each corridor was traversed several times in both directions. The total path driven was 690 meters taking 45 minutes to complete. Figure 12(b) is the "dead-reckoned" path using odometry data. Figure 13 depicts the amount of time spent in each map-frame. About halfway through the run, the algorithm reaches a steady state and does not create any additional maps — the largest map ID no longer increases. Note that the average number of active map-frames (containing a mature hypothesis) at any instant reaches a steady state value. Figure 14 illustrates the increase in certainty of inter map-frame transformations as areas are revisited. This manifests itself as a monotonically decreasing determinant of the transformation estimate covariance.

5 Conclusions and Further Work

We have presented *Atlas* , a general framework for large scale mapping and navigation. We have presented results using the framework with feature based local navigation in a large populated indoor environment. Two experiments were run in this environment using laser and then sonar sensing. In both cases the required computation was shown to be independent of the total mapped area and proportional to the total number of observations.

An additional experiment consisting of multiple loops was run in a smaller area. This illustrated the reduction in uncertainty in the relationship *between* local maps which is achieved when an area is revisited. This experiment also serves to illustrate the "steady-state" re-use of existing maps.

The fact that we do not embed estimates of transformations between map-frames in their maps warrants further discussion. We store the estimates externally in the edges of the *Atlas* graph. The reasoning behind this architecture is both philosophical and practical. Firstly, local maps are for estimating local entities – things that are contained within a single coordinate frame. Transformations between map-fames do not fulfill this criterion. Secondly, *Atlas* is intended to be a framework. Requiring the



(a) (b) (c)

Figure 11: Representing combined map-frame uncertainties from diametrically opposing locations. (a) All map-frames projected relative to map 1 before loop closure, and (b) after loop closure. (c) All map-frames projected relative to map 11.

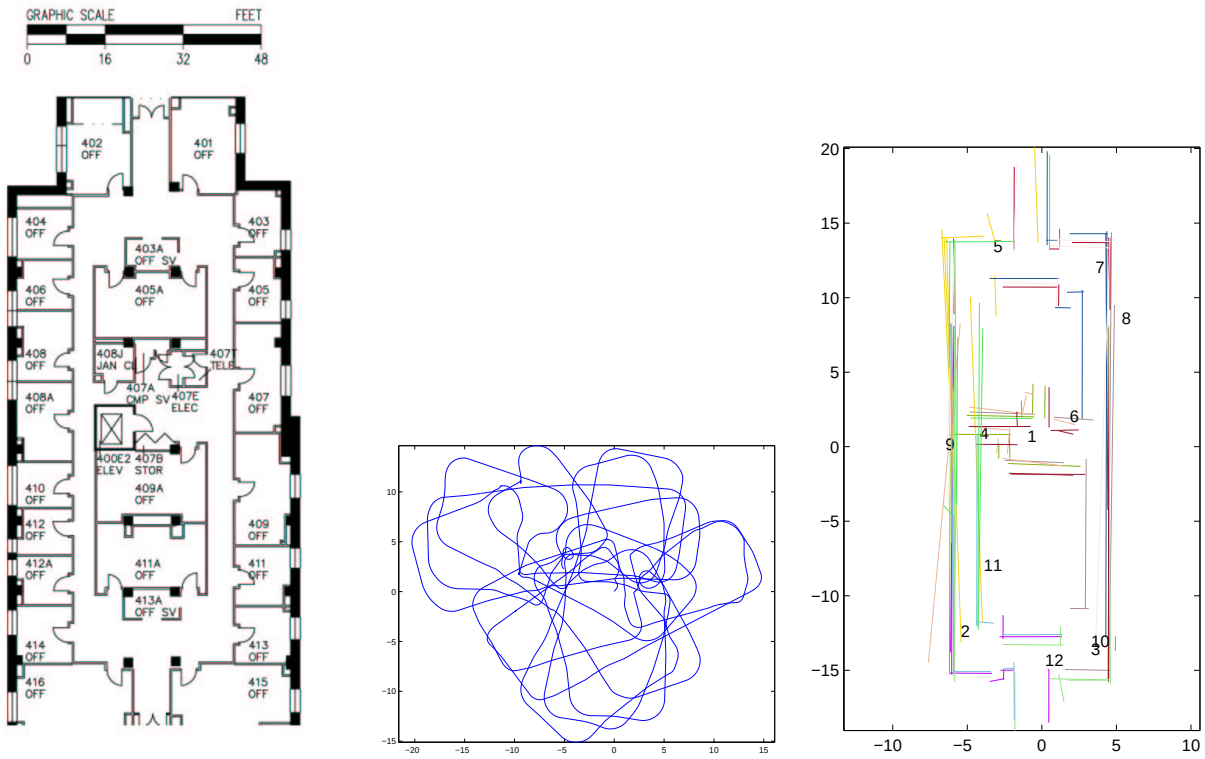
Table 2: Atlas parameter settings for the presented results.

Description	**Parameter**	**Value**
minimum probationary period for a juvenile hypothesis	τ_j	3 sec
maximum number of active hypotheses	$\mathcal{H}_m$	5
maximum number of features in map		10
maximum local uncertainty in robot	$\sigma_x^\star, \sigma_y^\star$ $\sigma_\theta^\star$	0.2 m 5 deg
minimum number of matched features	η_m	4

local maps to maintain estimates of edges imposes an unnecessary constraint on the local navigation algorithm. The third issue concerns map/edge correlation maintenance. Inserting edge estimation into each map will result in correlating local maps and edges. Maintaining consistency of these correlations across all connected map-frames would be a additional complication. This further endorses our policy to only keep local information in map-frames.

The following list summarizes the salient qualities of the *Atlas* frame work discussed in previous sections.

- The computational complexity of the *Atlas* framework is independent of the number of mapped features and the total area mapped — *Atlas* is "constant time".
- We maintain no single preferred global coordinate frame. Instead we use multiple local frames.
- The extent and geometry of local maps is not predefined. Instead they are dictated at run time by the local navigation and mapping performance.



(a) (b) (c)

Figure 12: (a)The architectural plan and (b) the dead-reckoned path for the multiloop data set. The total path length is 690 meters for a duration of 45 minutes. (c) The globally aligned set of map-frames at the end of the experiment.

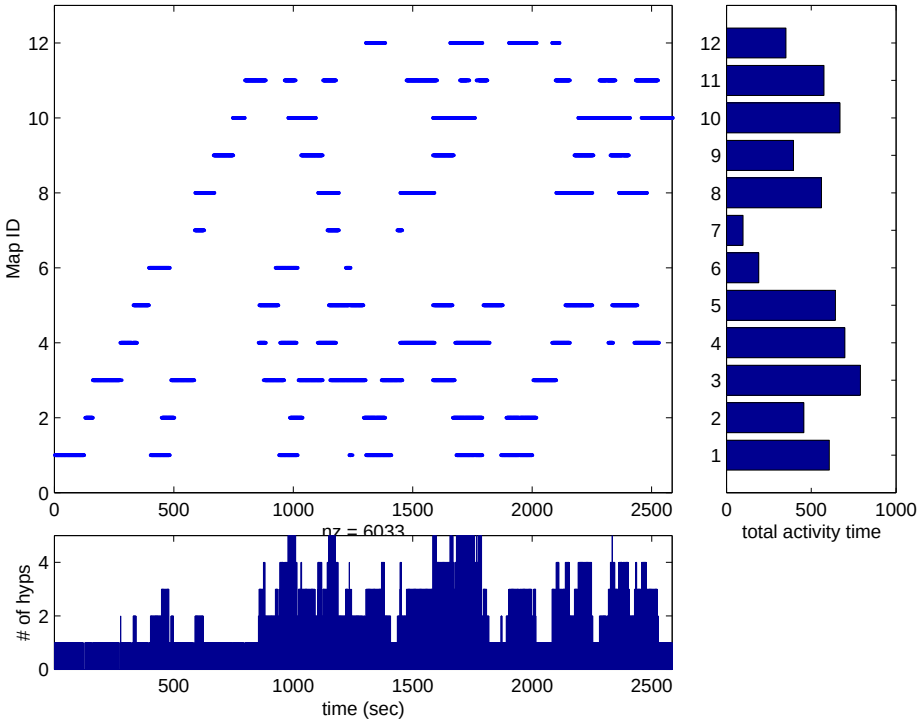
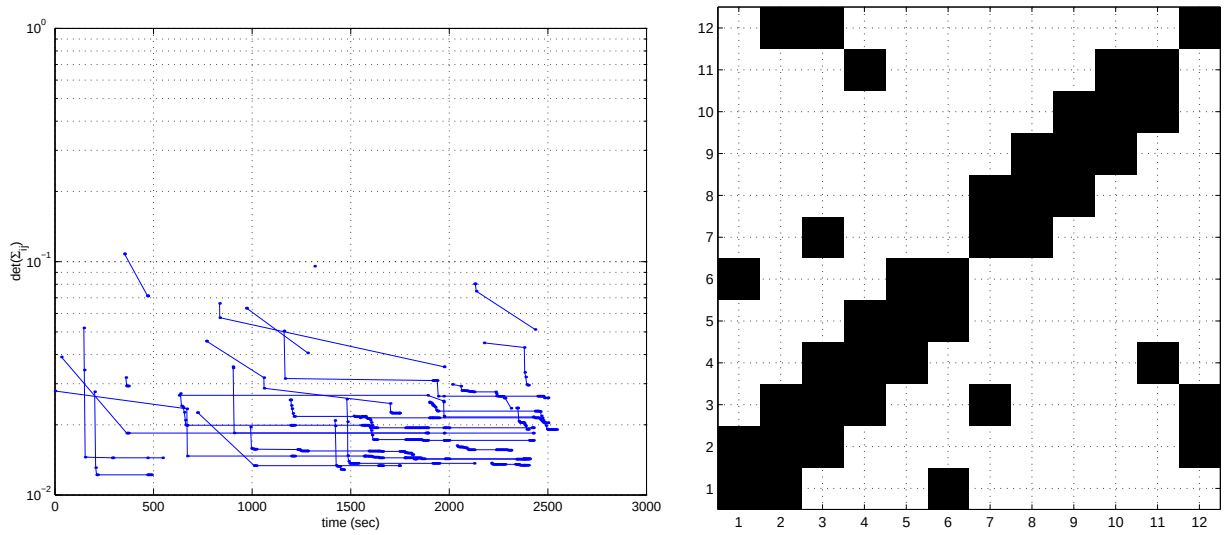


Figure 13: Map-frame genesis and activity. The generation of new map-frames drops off as the area becomes fully mapped and maps are re-used. Each dash in the central figure corresponds to a period in which a particular map frame was supporting an active, successful hypothesis. In this implementation map-frames are not deleted and so frames with erroneous maps might never be selected for re-use. Map 6 and 7 are such a cases — the right hand figure indicates correspondingly small amounts of activity time. The lower figure shows how the average number of active hypotheses remains constant throughout the experiment.

- The processes governing transitions between and creation of maps are fully automated.
- No constraints are placed on the scheme adopted for local navigation other than an ability to estimate robot position and uncertainty in the local frame. (We have used the example of a feature based approach through the paper.)
- *Atlas* maintains the adjacency and geometric relationships between regions as edges in a graph. The nodes of this graph are the local maps.
- New graph edges are created when commonly mapped regions are identified in two local maps. This process is called Map-Matching.
- The *Atlas* framework continually strives to re-use existing local maps to explain sensor data. It instantiates a hypotheses in a limited number of neighboring maps to find the best explanation.
- The certainty in the relationship between frames can be increased when the vehicle re-enters a previously mapped region. This is a direct consequence of increasing vehicle certainty in the local map.
- We do not apply online enforcement of loop constraints.



(a) (b)

Figure 14: (a) Plot of the determinant of the *Atlas* edge transformation covariances. After creation, the updating of edges yeilds a monotonic decrease in their uncertainty. Each trajectory in the figure corresponds to a different edge in the graph. (b) A plot of map-frame adjacencies. A black square at coordinate (i,j) indicates the existence of an edge between frame i and j. Broadly, Genesis is reponsible for the creation of the diagonal entries whereas map matching and loop-closing produce off diagonals.

- It is possible to "prettify" the global arrangement of maps in off-line optimization procedure. This is not necessary for sustainable online navigation and mapping.

We hope to extend the 2D implementation described in this paper to a full 3D version for use with an omni-directional video camera mounted on the B21 robot. We are also interested in a statement of the relationship between the convergence properties of the *Atlas* framework and a conventional "full covariance" solution (notwithstanding angular error) as proposed by [SSC90, DNDW$^+$01].

We believe that the *Atlas* framework constitutes a powerful tool in tackling the large scale SLAM problem.

Appendix A — Feature-based Local Navigation and Mapping

For the results presented in this work we chose to adopt a common approach to the SLAM problem - feature based stochastic mapping using an Extended Kalman Filter. There is a rich corpus of literature on this subject [SSC90]. For completeness, the EKF derived feature based SLAM algorithm is summarized below.

We seek an estimate $\hat{\mathbf{x}}(i|j)$ at time i using vehicle relative observations of features made up until time j of a state vector defined to be concatenation of vehicle and n feature locations (the map) and its

covariance $\mathbf{P}(i|j)$.

$$\begin{aligned}\hat{\mathbf{x}}(i|j) &= \left[\hat{\mathbf{x}}_v(i|j)^T, \hat{\mathbf{x}}_1(i|j)^T \cdots \hat{\mathbf{x}}_n(i|j)^T\right]^T \\ &= \left[\hat{\mathbf{x}}_v(i|j)^T, \hat{\mathbf{x}_\mathbf{m}}(i|j)^T\right]^T \\ \mathbf{P}(i|j) &= \begin{bmatrix}\mathbf{P}_{vv}(i|j) & \mathbf{P}_{vm}(i|j) \\ \mathbf{P}_{vm}(i|j)^T & \mathbf{P}_{mm}(i|j)\end{bmatrix}\end{aligned}$$

The state vector to be estimated $\hat{\mathbf{x}}$ evolves according to a non linear process model:

$$\mathbf{x}(k+1) = \mathbf{F}(\mathbf{x}(k), \mathbf{u}(k+1)) + \mathbf{v}_v(k+1).$$

Here $\mathbf{v}$ is a process noise vector which is assumed to be a zero mean, temporally uncorrelated random sequence with covariance given by:

$$\begin{aligned}\mathbf{E}[\mathbf{v}(k)] &= \mathbf{0} \\ \mathbf{E}[\mathbf{v}(i).\mathbf{v}(j)^T] &= \begin{cases}\mathbf{Q}(k) & \text{if } i = j = k, \\ \mathbf{0} & \text{otherwise.}\end{cases}\end{aligned}$$

If we assume that features do not move, the process model can be simplified to

$$\mathbf{x}(k+1) = \begin{bmatrix}\mathbf{F}_v(\mathbf{x}_v(k), \mathbf{u}(k+1)) \\ \mathbf{x}_\mathbf{m}(k)\end{bmatrix} + \begin{bmatrix}\mathbf{v}_v(k+1) \\ \mathbf{0}\end{bmatrix}.$$

Observations of features are modeled using a non-linear relationship between state and measurements

$$\hat{\mathbf{z}}(k) = \mathbf{H}(\mathbf{x}(k)) + \mathbf{w}(k).$$

The terms $\mathbf{w}$ is an observation noise vectors which is assumed to be a zero mean, temporally uncorrelated random sequence with covariance given by:

$$\begin{aligned}\mathbf{E}[\mathbf{w}(k)] &= \mathbf{0} \\ \mathbf{E}[\mathbf{w}(i).\mathbf{w}(j)^T] &= \begin{cases}\mathbf{R}(k) & \text{if } i = j = k, \\ \mathbf{0} & \text{otherwise.}\end{cases}\end{aligned}$$

The prediction equations for the EKF are written as follows

$$\begin{aligned}\hat{\mathbf{x}}(k+1|k) &= \mathbf{F}(\hat{\mathbf{x}}(k|k), \mathbf{u}(k+1), k+1) \\ \mathbf{P}(k+1|k) &= \nabla\mathbf{F}_{\hat{\mathbf{x}}}\mathbf{P}(\mathbf{k}|\mathbf{k})\nabla\mathbf{F}_{\hat{\mathbf{x}}}^{\mathbf{T}} + \mathbf{Q}(\mathbf{k}+\mathbf{1}) \\ \hat{\mathbf{z}}(k+1|k) &= \mathbf{H}(\hat{\mathbf{x}}(k+1|k), k+1)\end{aligned}$$

The term $\nabla\cdot_{\hat{\mathbf{x}}}$ is understood to be the Jacobian of $(\cdot)$ with respect to $\mathbf{x}$ evaluated at $\hat{\mathbf{x}}(k+1|k)$.

Following the prediction, the new observation $\mathbf{z}(k+1)$ is fused with the prior estimate by application of the following equations

$$\begin{aligned}\hat{\mathbf{x}}(k+1|k+1) &= \hat{\mathbf{x}}(k+1|k) + \mathbf{W}(k+1)\nu(k+1) \\ \mathbf{P}(k+1|k+1) &= \mathbf{P}(k+1|k) - \mathbf{W}(k+1)\mathbf{S}_{\nu\nu}\mathbf{W}^T(k+1)\end{aligned}$$

where

$$\begin{aligned}\nu(k+1|k) &= \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k) \\ \mathbf{S}_{\nu\nu} &= \nabla\mathbf{H}_{\hat{\mathbf{x}}}\mathbf{P}(\mathbf{k+1|k})\nabla\mathbf{H}_{\hat{\mathbf{x}}}^{\mathbf{T}} + \mathbf{R}(\mathbf{k+1}) \\ \mathbf{W}(k+1) &= \mathbf{P}(k+1|k)\nabla\mathbf{H}_{\hat{\mathbf{x}}}^{\mathbf{T}}\mathbf{S}_{\nu\nu}{}^{-\mathbf{1}}(\mathbf{k+1})\end{aligned}$$

The nature of the SLAM problem results in sparse Jacobians of the observation and process models. Under the assumption of static features $\nabla\mathbf{F}_{\mathbf{x}}$ becomes

$$\nabla\mathbf{F}_{\mathbf{x}} = \begin{bmatrix} \nabla\mathbf{F}_v & \mathbf{0} & \cdots & \cdots \\ \mathbf{0} & \mathbf{I} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I} \end{bmatrix}.$$

The observation of feature i is a function of only the vehicle state and the feature being observed. Hence the observation Jacobian is highly sparse and can be written in block form as:

$$\nabla\mathbf{H}_{\mathbf{x}} = [\nabla\mathbf{H}_{\mathbf{v}} \cdots \mathbf{0} \cdots \nabla\mathbf{H}_{\mathbf{i}} \cdots \mathbf{0}] \tag{18}$$

where $\nabla\mathbf{H}_{\mathbf{i}}$ is the derivative with respect to the i^{th} feature and $\nabla\mathbf{H}_{\mathbf{v}}$ is the derivative with respect to the vehicle pose. With n features, the sparseness of these matrices allows the prediction step to be performed $\mathcal{O}(n)$ and the update in $\mathcal{O}(n^2)$.

An important aspect of this algorithm is that the dimension of the state vector is not constant - it grows as more features are observed and mapped. An observation $\mathbf{z}$ of an unmapped feature can be used to initialize a new feature via an initialization function $\mathbf{g}(\hat{\mathbf{x}}, \mathbf{z})$.

$$\begin{aligned}\hat{\mathbf{x}} &\Rightarrow \left[\hat{\mathbf{x}}^T, \mathbf{g}(\hat{\mathbf{x}}, \mathbf{z})^T\right]^T \\ \mathbf{P} &\Rightarrow \begin{bmatrix} \mathbf{P} & \mathbf{P}\nabla\mathbf{G}_{\mathbf{x}}^{\mathbf{T}} \\ \nabla\mathbf{G}_{\mathbf{x}}\mathbf{P} & \nabla\mathbf{G}_{\mathbf{x}}\mathbf{P}\nabla\mathbf{G}_{\mathbf{x}} + \nabla\mathbf{G}_{\mathbf{z}}\mathbf{R}\nabla\mathbf{G}_{\mathbf{z}} \end{bmatrix}\end{aligned}$$

The association of measurements to features can be accomplished via the commonly applied nearest neighbor gating technique as described in [DNDW$^+$01, BSF88].

Appendix B — Jacobians of transformation compositions

In the case of two dimensions, we can represent a coordinate frames transformations as translation followed by a rotation, parameterized as:

$$T_i^j = \begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix}$$

which describes the transformation of coordinates in frame j to frame i.

We can compose transformations with the $\oplus$ operator as follows:

$$\begin{aligned}T_i^k &= T_i^j \oplus T_j^k \\ &= \begin{bmatrix} x_i + c_i x_j - s_i y_j \\ y_i + s_i x_j + c_i y_j \\ \theta_i + \theta_j \end{bmatrix}\end{aligned}$$

where c_i and s_i are $\cos(\theta_i)$ and $\sin(\theta_i)$, respectively.

When transforming the covariances of transformations, we need to evaluate the Jacobians of the transformation operator with respect to either of its arguments. The Jacobian of the T_i^k with respect to the *first* argument of $\oplus$ is $\mathrm{J}_1\left(T_i^j, T_j^k\right)$.

$$\mathrm{J}_1\left(T_i^j, T_j^k\right) = \begin{bmatrix} 1 & 0 & -s_i x_j - c_i y_j \\ 0 & 1 & c_i x_j - s_i y_j \\ 0 & 0 & 1 \end{bmatrix}$$

Likewise, the Jacobian of the T_i^k with respect to the *second* argument of $\oplus$is $\mathrm{J}_2\left(T_i^j, T_j^k\right)$.

$$\mathrm{J}_2\left(T_i^j, T_j^k\right) = \begin{bmatrix} c_i & -s_i & 0 \\ s_i & c_i & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The operator $\ominus$ forms the inverse transformation $T_j^i = \ominus T_i^j$ such that $T_j^i \ominus T_i^j = I$, where I is the identity transform. The Jacobian of T_j^i with respect to T_i^j is $\mathrm{J}_\ominus\left(T_i^j\right)$.

$$\mathrm{J}_\ominus\left(T_i^j\right) = \begin{bmatrix} -c_i & -s_i & s_i x_i - c_i y_i \\ s_i & -c_i & c_i x_i + s_i y_i \\ 0 & 0 & -1 \end{bmatrix}$$

References

[BK89] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *IEEE Trans. Systems, Man, and Cybernetics*, 19:1179–1189, September 1989.

[BN01] T. Bailey and E. Nebot. Localization in large-scale environments. *Robotics and Autonomous Systems*, 37(4):261–281, 2001.

[BSF88] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[BZ99] C. Baillard and A. Zisserman. Automatic reconstruction of piecewise planar models from multiple views. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 559–565, June 1999.

[CK97] K. Chong and L. Kleeman. Large scale sonarray mapping using multiple connected local maps. In *International Conference on Field and Service Robotics*, pages 538–545, ANU, Canberra, Australia, December 1997.

[Dav98] A. J. Davison. *Mobile Robot Navigation Using Active Vision*. PhD thesis, University of Oxford, 1998.

[DFBT99] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1999.

[Dij59] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Matematik*, 1:269–271, 1959.

[DNDW+01] M. W. M. G. Dissanayake, P. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on Robotic and Automation*, 17(3):229–241, June 2001.

[Fau93] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.

[FB80] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Technical Report 213, SRI International, March 1980.

[GN01] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. *IEEE Transactions on Robotic and Automation*, 17(3):242–257, June 2001.

[Jen01] P. Jensfelt. *Approaches to Mobile Robot Localization in Indoor Environments*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2001.

[JU97] S. J. Julier and J. K. Uhlmann. A non-divergent estimation algorithm in the presence of unknown correlations. In *Proceedings of the IEEE American Control Conference*, volume 4, pages 2369–2373, Albuguerque, NM, USA, June 1997.

[Kui00] B. J. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191–233, 2000.

[LDW92] J. J. Leonard and H. F. Durrant-Whyte. *Directed Sonar Sensing for Mobile Robot Navigation*. Boston: Kluwer Academic Publishers, 1992.

[LF01] J. Leonard and H. Feder. Decoupled stochastic mapping. *IEEE J. Ocean Engineering*, 26(4):561–571, 2001.

[LRNB02] J. J. Leonard, R. J. Rikoski, P. M. Newman, and M. C. Bosse. Mapping partially observable features from multiple uncertain vantage points. *Int. J. Robotics Research*, 2002. To Appear.

[MC89] H. P. Moravec and D. W. Cho. A Bayesian method for certainty grids. In *AAAI Spring Symposium on Robot Navigation*, pages 57–60, 1989.

[NBL02] P. Newman, M. Bosse, and J. Leonard. Feature based exploration. Submitted for publication for ICRA 2003, September 2002.

[New02] P. Newman. Subsea range only SLAM. Submitted for publication for ICRA 2003, September 2002.

[NT01] J. Neira and J.D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Trans. on Robotics and Automation*, 17(6):890–897, 2001.

[PKVG00] M. Pollefeys, R. Koch, M. Vergauwen, and L. Van Gool. Automated reconstruction of 3d scenes from sequences of images. *ISPRS Journal Of Photogrammetry And Remote Sensing*, 55(4):251–267, 2000.

[SSC90] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I. Cox and G. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer-Verlag, 1990.

[TAB+01] S. Teller, M. Antone, Z. Bodnar, M. Bosse, S. Coorg, M. Jethwa, and N. Master. Calibrated, registered images of an extended urban area. In *Computer Vision and Pattern Recognition*, volume 1, pages 813–820, Kauai, December 2001.

[Thr01] S. Thrun. A probilistic online mapping algorithm for teams of mobile robots. *Int. J. Robotics Research*, 20(5):335–363, May 2001.

[TNNL02] J.D. Tardós, J. Neira, P.M. Newman, and J.J. Leonard. Robust mapping and localization in indoor environments using sonar data. *Int. J. Robotics Research*, 2002. To appear.

[WC00] O. Wijk and H. Christensen. Triangulation based fusion of sonar data with application in robot pose tracking. *IEEE Trans. Robotics and Automation*, 16(6):740–752, December 2000.

[WDDW02] S.B Williams, G. Dissanayake, and H. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 406–411, 2002.