The M-Space Feature Representation for SLAM

John Folkesson, *Member, IEEE*, Patric Jensfelt, *Member, IEEE*, and Henrik I. Christensen, *Member, IEEE*

***Abstract*—In this paper, a new feature representation for simultaneous localization and mapping (SLAM) is discussed. The representation addresses feature symmetries and constraints explicitly to make the basic model numerically robust. In previous SLAM work, complete initialization of features is typically performed prior to introduction of a new feature into the map. This results in delayed use of new data. To allow early use of sensory data, the new feature representation addresses the use of features that initially have been partially observed. This is achieved by explicitly modelling the subspace of a feature that has been observed. In addition to accounting for the special properties of each feature type, the commonalities can be exploited in the new representation to create a feature framework that allows for interchanging of SLAM algorithms, sensor and features. Experimental results are presented using a low-cost web-cam, a laser range scanner, and combinations thereof.**

***Index Terms*—Feature representation, localization, mapping, mobile tobots, simultaneous localization and mapping (SLAM), navigation.**

I. Introduction

THE PROBLEM of simultaneous localization and mapping has been widely studied. Central to the problem is how the estimation algorithm represents the map. Dense representations[1] that store partially processed sensor data give a very detailed map [1]–[7]. As more data is retained, this can be an advantage when, for instance, a robot needs to match sensor readings from a totally unknown location with the map. A drawback of this approach is that the amount of data stored in the map becomes very large.

An alternative is to use the sensor data to form a hypothesis about the existence and location of features in the environment. Only the features and the accumulated evidence about them is stored. This leads to an abstraction of the environment that ideally will capture the essential information gathered from the sensors. Thus, a feature map can be considered a form of data compression. The abstraction of the environment could also lead to an understanding of the data on a deeper level. It is preferred, even if not necessary, for the features to be landmarks recognizable by humans.

One advantage of the data reduction is that the resources that would have been used to store and manage the large data can now be used to better interpret the data. Thus, algorithms of greater complexity can be applied to try to estimate the map.

The selection of features is based on the task, the sensors, and the environment. One task that is fundamental to a SLAM map is the task of localization. Without the ability to localize the robot, using the map, it would be impossible to combine new sensor data with older information.

As a result, one requirement of a map feature is that it be able to constrain some of the pose dimensions of the robot. For the case of a wheeled robot operating in a planar environment, the distance to the floor is useless as a feature measurement since the robot is already constrained to a level surface. On the other hand, a doorway is a useful feature.

A single feature will partially constrain the robot pose in the full space of robot poses. In general, a combination of several features are required to fully constrain the pose. For example, in 2-D, three differing bearing angles to points are sufficient.

The feature selection is also driven by the need to extract information about the geometry of the features from the sensor data. This then is a requirement that the sensor measurements can constrain some of the geometry of features.

Thus, the available sensors limit the feature selection. Moreover, just as a single feature may not fully constrain the robot pose, a single sensor reading may not be enough to fully constrain the geometry of the feature. This leads to the situation of partially observed features. In some cases, the partial observability depends on the frame of reference of the sensor data. As the robot moves around it can more fully observe the features. For example, the angle of a wall might be measured first. Then, as the robot travels along the wall the length of the wall is observed. Any single measurement might only observe a subspace of the complete feature description.

This partial observability is a fundamental property of map features. Coming back to representations of features, the representation of the evidence gathered about features should be able to represent the lack of evidence for the unobserved parts of the features as well as the evidence for the observed parts.

The observable subspace needs to be explicitly represented for a practical reason. Simultaneous localization and mapping (SLAM) estimation might process different parts of the data using different estimated states. Without explicit representation, the observable subspace might shift in the full feature space throughout the calculation. This can lead to inconsistent estimates.

Another factor in feature selection is the environment itself. It can be a drawback of feature-based mapping that features

Manuscript received March 14, 2006; revised October 29, 2006. This paper was recommended for publication by Associate Editor G. Sukhatme and Editor L. Parker upon evaluation of the reviewers' comments. This work was supported by the Swedish Foundation for Strategic Research through the Centre for Autonomous Systems.

J. Folkesson is with the Department of Mechanical Engineering, Massacusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: johnfolk@mit.edu).

P. Jensfelt is with the Center for Autonomous Systems, Numerical Analysoch Datalogi, Kungliga Tekniska Högskolan, S-100 44 Stockholm, Sweden (e-mail: patric@nada.kth.se).

H. I. Christensen is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: hic@cc.gatech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2007.903807

[1]SLAM methods that employ dense representation include scan matching, image matching, and occupancy grid methods.

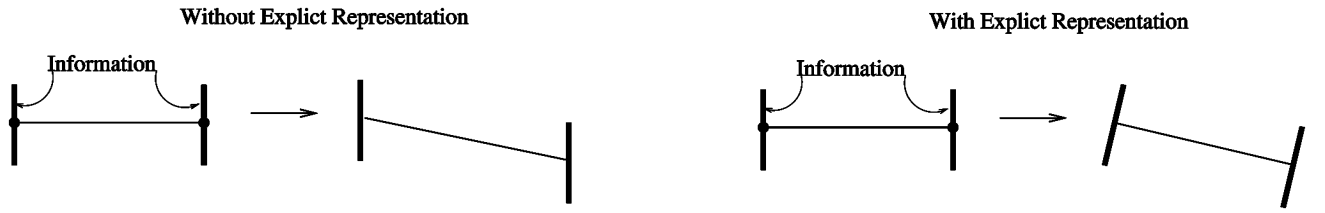

Fig. 1. Here, we illustrate how inconsistency can arise. The left pair of figures shows a line feature. The information gathered on the line is only perpendicular to the line. At a later stage of SLAM the line has rotated but the information has not. This then incorrectly indicates some information tangent to the line. By explicitly representing the observable subspace, we are able to have the information rotate with the feature (right pair of figures).

are normally selected for a specific environment based on prior knowledge of that environment. For a park, trees might be used [8], for an office building walls are more appropriate. Using vision some rather general image features can be employed [9], [10]. Some of the really early work on SLAM used features, e.g., [11]–[13].

One other issue regarding features is the initialization of the features in the map. If the features are initialized too quickly poor features may be used resulting in worse navigation. The use of delayed initialization was investigated in [14].

A. Outline

As pointed out in [15] most feature representations use completely different parameterizations for each type of feature. The result is a representation that lacks generality. The motivation for our work is to look for a representation that can: 1) use the information from the measurements of the features to improve the estimated feature coordinates while not changing the unobserved parts of the feature (preserving invariants); 2) represent both size and position information; 3) represent connections between features such as a corner between two walls; and 4) represent uncertainties in a frame attached to the feature.

In Section II, we introduce the basic mathematical analysis of using features to do SLAM. In Section III, we survey some of the different feature representations that have been used. In Section IV, we describe the basis for our new representation and provide some feature examples in Section IV-D. Section V outlines a SLAM EKF implementation using the new representation. Section VI deals with the cases of connected features such as when two walls form a corner. To highlight some of the benefits of the representation, we present experimental results in Section VII. Finally, we summarize and draw conclusions in Section VIII.

II. Feature Math

In Section I, we discussed how the feature measurements constrain the robot pose and parts of the feature coordinates. Here, we formalize the problem. The sensors provide data from which a vector of measurements is extracted, denoted by $\mathbf{v}$. At the same time there are features that have given rise to these measurements. These features will have some parameterization of their geometry,[2] which we will denote by $\boldsymbol{\lambda}$. Finally, the robot pose coordinates are denoted by $\mathbf{x}_r$.[3]

[2]By feature geometry we mean location, orientation, and spacial extent.

[3]The robot pose here could contain offsets to the various sensors and the number of coordinate dimensions could thus be greater than 6.

To make inferences about the feature and robot coordinates based on the measurements, we need to define a vector function that relates all three. We refer to this function as an innovation, $\eta(\mathbf{v}, \mathbf{x}_r, \boldsymbol{\lambda})$. The Jacobian matrix of this innovation is

$$\begin{pmatrix} J_{\eta v} & J_{\eta r} & J_{\eta \lambda} \end{pmatrix} = \begin{pmatrix} \frac{\partial \eta}{\partial \mathbf{v}} & \frac{\partial \eta}{\partial \mathbf{x}_r} & \frac{\partial \eta}{\partial \boldsymbol{\lambda}} \end{pmatrix}. \tag{1}$$

We can now be more formal about what we mean by measurements constraining the robot pose. The measurements will constrain the robot pose coordinates on the subspace of $\mathbf{x}_r$ spanned by $J_{\eta r}$. In other words, if there are directions that the robot pose can change without causing any change to the innovation, then these directions are not constrained by the measurement. Such directions are characterized by being in the null space of $J_{\eta r}^T J_{\eta r}$. Similarly, the observable subspace of the feature coordinates $\boldsymbol{\lambda}$ is spanned by $J_{\eta \lambda}$. We will now show how these two matrices are related to one another by the transformation properties of the feature coordinates.

The sensor measurements are taken in a reference frame attached to the robot. Thus, the innovation function in this frame will depend on the transformed feature coordinates only

$$\eta(\mathbf{v}, \mathbf{x}_r, \boldsymbol{\lambda}) = \eta(\mathbf{v}, \mathbf{x}_o) \tag{2}$$

where $\mathbf{x}_o = T(\boldsymbol{\lambda} | \mathbf{x}_r)$ are the feature coordinates transformed to the sensor frame of reference. It is the Jacobians of this transformation that give us the relationship between $J_{\eta r}$ and $J_{\eta \lambda}$

$$J_{\eta r} = J_{\eta o} J_{or}, \quad J_{\eta f} = J_{\eta o} J_{o\lambda}, \quad J_{\eta r} = J_{\eta f} J_{o\lambda}^{-1} J_{or}. \tag{3}$$

The approximations used by SLAM estimation algorithms often involve linearization of these Jacobians with respect to the feature and robot pose coordinates. Consistency problems occur when the Jacobians used at an earlier iteration of a SLAM algorithm no longer lie in the observable subspace of the features. This situation can arise when these features are rotated with respect to their earlier orientations, see Fig. 1. We have shown how the constrained subspace of robot poses and the observable subspace of features are related to one another by the transformation of the feature coordinates. Both this central importance of transformation rules and the existence of an observable subspace will guide us in defining our feature representation.

III. Feature Representations

A number of different types of features have been used for mapping. Depending on the type of application the model of the environment is 2-D or 3-D. For most indoor applications a 2-D representation is used. Navigation in cluttered environments often requires a 3-D representation. When taking the step

outdoors the world is less structured and it becomes increasingly likely that the ground is not flat which also calls for a 3-D model. Some commonly used features are as follows.

- *Point*: The point is probably the most commonly used feature. It is simple to work with, yet powerful. When working in 2-D, vertical edges become points. In outdoor applications, point features have successfully been used to represent, e.g., tree trunks [8].
- *Line*: In indoor man-made environments there are typically many walls. These appear as lines when measured with 2-D range sensors. These line segments are, therefore, natural to use as features. Vertical edges are also common in man-made environments. This type of line feature is typically extracted using visual information as in [16].
- *Plane*: When taking the step from 2-D to 3-D what was a line in 2-D often becomes a plane in 3-D. Planes have been used in, for example, [17] and [18].

A. Wall Feature Representations

In this section, we will look closer at the wall feature as it can be used to illustrate some of the good and bad aspects of the numerous different representations that have been suggested in the literature. To limit the scope we concentrate on representations for the 2-D case in which a wall appears as a line segment in the x–y plane.

Before listing different representations some characteristics of the wall are given. Four parameters are needed to fully specify the line segment. One key issue of a wall is that the sensors may not be able to detect its end points. Even if the end points are within the range of the sensors they are often hard to detect due to occlusions. This implies that a measurement typically only constrains the position of the sensor to a certain distance and relative angle w.r.t. the wall. In other words, all dimensions of a wall are typically not constrained by one single measurement.

1) Slope and Intersection: The slope and intersection representation $y = k_y x + m_y$ is the first that comes to mind. However, it suffers from a major disadvantage in that the parameter m_y has a hyperbolic dependence on the rotation angle of the line. This nonlinear and singular behavior must be dealt with in an *ad hoc* manner and adds complications and approximations to the SLAM estimate. The extent of the wall is not represented either.

2) End Points: The most straight forward representation is to use the two end points to parameterize the wall. One disadvantage with this is that one typically does not measure the end points of a wall but rather the distance to and orientation of the wall. This means that all four dimensions of the wall cannot be properly initialized. An *ad hoc* solution to this using very large uncertainty for the end-points along the wall will not behave well during estimation.

3) Distance and Direction $(\rho\alpha)$: Starting from the measurements, one of the most natural representations is to use the perpendicular distance to the wall and the orientation of the wall. This results in a description of an infinite line. This has been used in, for example, [16]. There are two main problems with this representation. The first one is that walls are not infinite in reality. This becomes a problem in large areas where there will be lines everywhere. The second problem is the so called "lever-arm" effect [19] that results from the dependence of the representation on the choice of origin. Small changes in the orientation angle will be able to move parts of the line by significant distances.

4) Complementing $\rho\alpha$: To limit the extent of the wall, two more parameters can be added: the length and the tangential position of the wall. The tangential position is defined as the distance from the center of the wall to the normal passing through the origin. This representation still suffers from the lever-arm effect.

5) Center Point, Length, and Orientation: Another way to represent a wall is to specify the center position, the orientation, and the length of the wall. The main disadvantage of this representation according to [19] is the strong coupling that it creates between the parameters. This representation also suffers from the problem of not being fully observable in all cases.

6) SP-Model: The SP-model [15] offers a solution to the lever-arm effect. A local coordinate system is attached to the center of the wall and the wall is described by the transformation to this local frame. This allows for a representation of the uncertainty of the wall location in this local frame. A drawback of the SP-model is that it does not handle information along the direction of the wall as easily. We will come back to describe the SP-model in more detail in Section III-B.

7) Summary: To summarize, there are many different representations. The previous is just a collection of some of the most common ones for walls. It is clear that all of the previous representations have their pros and cons, and this is the reason for the large selection of them. Looking back at Section I-A and our initial requirements, we see that we can discard the representations that use the distance and orientation of the line in global coordinates since these represent the uncertainties in a global frame leading to the lever-arm effect. The arguments against the end point model are mainly that it requires more computations for matching and update and that it requires four parameters to be updated even though not all dimensions might be constrained, e.g., if only the distance and orientation are known.

B. SP-Model

The feature representation we propose in this paper is similar to the symmetries and perturbation (SP)-model [15] and we will, therefore, provide a summary of some of its key points. For a full description we refer to [20]. One of the take home messages given by the SP-model literature is that SLAM is to a large extent about making different types of frame transformations. The map features are in the map frame, the robot has its own moving frame, and the sensors as well. When performing predictions, matchings, and updates one constantly moves between these different frames. A good framework should make these operations simpler and be as general and reusable as possible.

The symmetry in the name comes from the symmetries that are often encountered in features. For example, given an infinite line there is a translational symmetry along the direction of the line, a rotational symmetry around the line, and a reverse symmetry. This means that one cannot detect translations along the line axis, not rotations around it and not if the line is turned around 180 deg.

A distinction is made between the symmetries in the feature itself and the observations. Feature symmetries correspond to

the degrees-of-freedom (DOF) that are not determined by the feature, such as the three rotation DOFs for a point. The observation symmetries depend on the type of feature and the type of observation.

A characteristic property of SP-model is that each feature element has its own local reference frame. The frame of reference is chosen with the axes along the directions of symmetry. A line for example has the x-axis along the direction of the line. A plane will have a normal that coincides with the z-axis. The main advantage of using a local frame is that the description of the uncertainty can be made independent of the global position of the features (compare the lever-arm effect from Section III-A). The local frames also help to make frame transformations and differentiations thereof more standardized.

Another key concept in the SP-model is the so called *binding matrix* B. The binding matrix is a row-selection matrix. The self-binding matrix selects the DOFs that are not part of the motion symmetry, i.e., the DOFs of a feature that are constrained and have probabilistic information attached to them.

Now to the perturbation part of the SP-model name. The location of a feature is given by a location vector $\mathbf{x}_{\mathrm{SP}}$. The estimation of this location vector is composed of two parts: an estimated location vector, $\hat{\mathbf{x}}_{\mathrm{SP}}$ and a differential location vector $\mathbf{d}_{\mathrm{SP}}$ defined as

$$\mathbf{x}_{\mathrm{SP}} = \hat{\mathbf{x}}_{\mathrm{SP}} \oplus \mathbf{d}_{\mathrm{SP}}. \tag{4}$$

The location vector gives the transformation to the local frame in the base frame. The differential location vector gives the location of the feature relative to the local frame. With this setup the differential location will be small and lever-arm effects will be kept to the minimum. Some of the dimensions of $\mathbf{d}_{\mathrm{SP}}$ will correspond to the motion symmetries of the feature and are zero. The rest of the dimensions form the perturbation vector $\mathbf{p}_{\mathrm{SP}}$, which is related to the differential location vector via the binding matrix according to

$$\mathbf{d}_{\mathrm{SP}} = B_{\mathrm{SP}}^T \mathbf{p}_{\mathrm{SP}} \tag{5}$$

$$\mathbf{p}_{\mathrm{SP}} = B_{\mathrm{SP}} \mathbf{d}_{\mathrm{SP}}. \tag{6}$$

The estimate of the perturbation vector is denoted by $\hat{\mathbf{p}}_{\mathrm{SP}}$ and has an associated uncertainty expressed by the covariance matrix C_{SP}. Each feature is described by a triplet $\{\hat{\mathbf{x}}_{\mathrm{SP}}, \hat{\mathbf{p}}_{\mathrm{SP}}, C_{\mathrm{SP}}\}$. Note that although the location vectors all have dimension 6, the perturbation vector, and thus the covariance matrix can be of lower order. A point, for example, will have dimension 3 for $\mathbf{p}$ and C. The selection of dimensions is taken care of by the binding matrix.

To summarize, the SP-model offers a nice solution to many of the representational problems. It provides a general way to deal with features which can be exploited in frameworks such as presented in [21]. Another advantage is that the binding matrices offer a machinery for making partial observations of a feature. This is useful, for example, when observing a single point on a line. A limitation with the SP-model is that one has to attach a frame to all features. For some types, such as lines, it is difficult to model the extent, e.g., the length, in a probabilistic way within the SP-model framework. In [20], the length of lines is estimated and modeled but it relies on always detecting both end points at the same time and making a *direct measurement* of the length. An *indirect measurement* is not possible as the origin of the reference frame cannot be observed as it is not attached to anything observable, just defined to be in the middle of the line. To go further and also store the actual location of the end points also requires some work. End points of lines are handled by so called semi-planes. They have their own reference frame, aligned with the one for the line, have no symmetries but are highly correlated with the corresponding line feature. One of the goals with the SP-model was to avoid redundancy in the representation. A line is fully specified by four parameters. The SP-model estimates two parameters when the line has unknown extent but eight parameters $(3+3+2)$ when both end points are estimated. The problem occurs when a symmetry "breaks," e.g., when observing the end of a line and the translational symmetry is removed. When the reference frame is attached to the middle of a line there is no way to observe the origin of the frame and thus nowhere to relate the end points.

IV. M-Space Representation

In this section, we introduce a new feature representation that like the SP-model attaches a local frame to each feature element and allows for a generic treatment of many types of features.

The new representation is called the M-space representation, where M-space denotes the measurement subspace as explained in the following. Here, we propose to use a set of point coordinates as the basis for the representation. We deal with the following three types of coordinates:

- *3-D coordinates* $\mathbf{x}^{3D}$ are the general 3-D coordinates (x, y, z);
- *2-D coordinates* $\mathbf{x}^{2D}$ are used when dealing with features in a 2-D representation, i.e., when the features are assumed to be infinite in the vertical direction.
- *Scalar coordinates* $\mathbf{x}^{S}$ can be used to model nonposition information such as, for example, the radius of a tree trunk.

Combinations of any number of the three types of coordinates can be used to parameterize a certain feature. By using this parameterization, we have a generic treatment of coordinate transformations. These transformations are central to geometric estimation and by making them generic we can have code common to all features to do the most complicated parts of the estimation algorithms.

The other important part of SLAM is the estimate of the uncertainty in the coordinates. For this we will first introduce some notation. Let s denote the sensor frame in which measurements are typically given. Let r denote the robot frame and m the map (or global) frame. A feature coordinate $\mathbf{x}_f$ expressed in the sensor frame is denoted $\mathbf{x}_{s,f}$ and in the map frame $\mathbf{x}_{m,f}$. With this notation, we can write the transformation rules for the different coordinates as

$$\mathbf{x}_{s,f}^{3D} = R_{m,s}^{3D} \left(\mathbf{x}_{m,f}^{3D} - \mathbf{x}_{m,s}^{3D} \right) \tag{7}$$

$$\mathbf{x}_{s,f}^{2D} = R_{m,s}^{2D} \left(\mathbf{x}_{m,f}^{2D} - \mathbf{x}_{m,s}^{2D} \right) \tag{8}$$

$$\mathbf{x}_{s,f}^{S} = \mathbf{x}_{m,f}^{S} \tag{9}$$

where $R_{m,s}$ is the rotation matrix from the map frame to the sensor frame. Note that the 2-D rotation matrix $R_{m,s}^{2D}$ is not a normal rotation matrix. It accounts for 3-D motion of the robot

assuming that the features are the 2-D projection of vertical structures as an edge (see Appendix I for more details).

With this said, we can introduce the M-space concept. The measurement subspace, or M-space, is an abstraction of the measured subspace of the feature space that reflects symmetries and constraints. The idea is that the features are parameterized to fully specify their location and extent (the feature space) but that they can be initialized in a subspace corresponding to the information provided by the sensors.

For example, when representing a line segment the extent is accommodated for in the representation even though only the distance to and the orientation of the line is known initially. We cannot represent the uncertainty with regard to changes in the coordinates along the length of the line by a Gaussian distribution. However, the uncertainty regarding changes perpendicular to the line and regarding the orientation can be approximated by a Gaussian. Let $\delta\mathbf{x}_p$ denote the M-space corresponding to a small change in feature coordinates $\delta\mathbf{x}_f$. Here, the subscript p stands for small perturbations in the M-space. The actual values of the M-space coordinates $\mathbf{x}_p$ are never needed or considered. It is only the changes to them that enter into the estimates. These changes are used to make adjustments to the feature coordinates $\mathbf{x}_f$. The uncertainty estimate is an estimate of the distribution of $\delta\mathbf{x}_p$ values around a mean of 0. The adjustments to the feature coordinates are made to maintain this 0 mean. No recentering step like in the SP-model is required with this view of the uncertainty. The uncertainty is defined in a frame attached to the feature and can be projected into the global frame using the current global coordinates of the feature. The statistics are represented in an analytic way rather than in the strict geometric sense of the SP-model. In most cases, the differences are in the second-order corrections to the covariances.

A. Projection Matrix

The relation between the feature space coordinates and the M-space coordinates is defined by a projection matrix $B(\mathbf{x}_f)$ similar to the binding matrix in the SP-model. The projection matrix relates small changes $\delta\mathbf{x}_p$ to small changes $\delta\mathbf{x}_f$. An important difference to the binding matrix is that the projection matrix is a function of the individual feature and changes with time. The rather involved recentering step in the SP-model is replaced by reevaluating the projection matrices. The fundamental relations between $\delta\mathbf{x}_p$ and $\delta\mathbf{x}_f$ are

$$\delta\mathbf{x}_p = B(\mathbf{x}_f)\delta\mathbf{x}_f \tag{10}$$

$$\delta\mathbf{x}_f = \widetilde{B}(\mathbf{x}_f)\delta\mathbf{x}_p \tag{11}$$

$$I_{pp} = B(\mathbf{x}_f)\widetilde{B}(\mathbf{x}_f) \tag{12}$$

where we refer to $\widetilde{B}(\mathbf{x}_f)$ as the dual of $B(\mathbf{x}_f)$.

B. Feature Initialization and Growing Dimensions

A common issue in feature-based SLAM is that one cannot initialize a feature after the first observation. A single observation typically does not contain enough information to do so reliably. Among the reasons behind this we find the following.

- The entire feature is not detected at once.
  - — In the case of a line, the end points might not have been detected if the line is partially occluded or long.
  - — Using monocular vision only the bearing to the feature can be initialized from a single image.
- Measurements are noisy. Even though a feature is fully observed it is good practice to get a second opinion from new measurement data to reject false measurements.

The M-space representation offers a solution to these problems by allowing the M-space dimensionality to change over time. Features are typically initialized with zero M-space dimensions and with time, as more information is gathered, more dimensions will be added. Returning once again to the wall example, the life cycle of the wall might be one of the following:

1) *First Detection*: feature initialized with 0 M-space dimensions.
2) *Reobserved N Times*: the wall existence and quality is confirmed. The distance and orientation of the wall added to the M-space.
3) *Start Point Detected*: M-space dimensionality goes up to 3.
4) *End Point Detected*: the feature reaches full dimensionality of 4.

The importance of the ability to let the dimensions of a feature grow over time is well illustrated by a horizontal line feature observed by a camera. A single image does not contain information to pinpoint the location of a feature. The assumption that the line feature is horizontal implies that a single observation will be enough to provide information about the relative orientation of the robot. That is, even if the robot moves parallel under the line and is unable to use triangulation to fix the position of the line in space the observations of the line can help reduce the angular uncertainty of the robot. This is useful in, for example, a corridor where the motion often is parallel to the linear structures found in the ceiling.

TABLE I
PARAMETERIZATION OF SOME DIFFERENT FEATURES. ALSO SHOWN ARE THE MINIMUM AND MAXIMUM DIMENSIONS OF THE M-SPACE WHEN THE CORRESPONDING FEATURE IS INITIALIZED

Feature	Parameterization	min(dim(M))	max(dim(M))
Point	$\mathbf{x}_f = \{\mathbf{x}^{3D}\}$	3	3
Wall	$\mathbf{x}_f = \{\mathbf{x}^{2D}_{start}, \mathbf{x}^{2D}_{end}\}$	2	4
HLine	$\mathbf{x}_f = \{\mathbf{x}^{3D}_{start}, \mathbf{x}^{3D}_{end}\}$	1	3
Pole	$\mathbf{x}_f = \{\mathbf{x}^{S}, \mathbf{x}^{2D}\}$	3	3

C. Enforcing Constraints

The coordinates that parameterize the feature will sometimes have constraints on them. A line detected on the ceiling mentioned in Section IV-B is one example. The horizontal constraint says that the z-coordinate of the two line end points must be the same. Assuming that they are initialized at the same height the projection matrix can be used to ensure this. Another form of constraint is when two different features share a common coordinate. This will be dealt with in more detail in Section VI.

D. Examples of Feature Parameterizations

To make the discussion from Section IV-C more concrete, we will show how to parameterize four different feature types that we use later for the experimental evaluation. Table I summarizes these.

TABLE II
B-MATRICES FOR DIFFERENT TYPES OF FEATURES WHEN THEY HAVE REACHED THEIR FULL DIMENSION. THE PARAMETER γ IS THE NORMAL TO THE LINE IN THE $x-y$ PLANE

Feature	B-matrix
Point	$B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
Wall	$B = \begin{pmatrix} \frac{\cos\gamma}{L\sqrt{2}} & \frac{\sin\gamma}{L\sqrt{2}} & \frac{-\cos\gamma}{L\sqrt{2}} & \frac{-\sin\gamma}{L\sqrt{2}} \\ \frac{\cos\gamma}{\sqrt{2}} & \frac{\sin\gamma}{\sqrt{2}} & \frac{\cos\gamma}{\sqrt{2}} & \frac{\sin\gamma}{\sqrt{2}} \\ -\sin\gamma & cos\gamma & 0 & 0 \\ 0 & 0 & -\sin\gamma & \cos\gamma \end{pmatrix}$
HLine	$B = \begin{pmatrix} \frac{\cos\gamma}{L\sqrt{2}} & \frac{\sin\gamma}{L\sqrt{2}} & 0 & \frac{-\cos\gamma}{L\sqrt{2}} & \frac{-\sin\gamma}{L\sqrt{2}} & 0 \\ \frac{\cos\gamma}{\sqrt{2}} & \frac{\sin\gamma}{\sqrt{2}} & 0 & \frac{\cos\gamma}{\sqrt{2}} & \frac{\sin\gamma}{\sqrt{2}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & \frac{1}{\sqrt{2}} \end{pmatrix}$
Pole	$B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

E. Point Feature

The point feature is the simplest of the features to describe. It is parameterized by a single 3-D point

$$\mathbf{x}_f = \{\mathbf{x}^{3D}\}.$$

The M-space coordinates are the same as the feature coordinates which gives a B-matrix that is the identity matrix (see Table II). In this paper, we use point features to model lamps mounted in the ceiling. For initialization information, we store the position of the camera and the direction to the lamp in that frame for each observation. It is not until the robot has moved enough to allow for triangulation that the points can be initialized. The point feature is initialized directly to three dimensions (its full dimension).

F. Wall Feature

Wall features are detected using a laser scanner and are parameterized by two 2-D points

$$\mathbf{x}_f = \left\{\mathbf{x}^{2D}_{\text{start}}, \mathbf{x}^{2D}_{\text{end}}\right\}$$

the end points of the wall.

This parameterization comes from the fact that the line segment found in the scan is assumed to be from a vertical plane, the wall. The wall is initialized as having two dimensions in the M-space, corresponding to the perpendicular distance to and direction of the line segment. This results in the first two rows in the B-matrix shown in Table II. The third row in the B-matrix in Table II corresponds to the start point and the fourth to the end point. The dimensionality of a wall can thus be 0 (not initialized), 2 (no end points), 3 (one end point), and 4 (two end points). Given the endpoints of a line in 2-D it is possible to find a normal vector to the line

$$\mathbf{Normal}^{\text{Line2D}} = (\cos\gamma, \quad \sin\gamma). \tag{13}$$

There are two possible normal vectors. One can chose the vector that points away from the wall surface. This defines γ unambiguously. The length of a line L given its endpoints is also known. The projection matrix for the case when the M-space corresponds to the distance to and the orientation of the line is given by

$$B^{\text{Line2D}} = \begin{pmatrix} \frac{\cos\gamma}{L\sqrt{2}} & \frac{\sin\gamma}{L\sqrt{2}} & \frac{-\cos\gamma}{L\sqrt{2}} & \frac{-\sin\gamma}{L\sqrt{2}} \\ \frac{\cos\gamma}{\sqrt{2}} & \frac{\sin\gamma}{\sqrt{2}} & \frac{\cos\gamma}{\sqrt{2}} & \frac{\sin\gamma}{\sqrt{2}} \end{pmatrix}. \tag{14}$$

Notice how the first row in the projection matrix corresponds to rotation around its center, that is to say changes to γ. The scaling with L is chosen so that $\delta\mathbf{x}_p$ will be directly related to the measurement angle and not dependent on the length of the line. The second row is the projection of the movements of the center point of the wall in the normal direction. A dual of this projection matrix that satisfies (12) is given by

$$\widetilde{B}^{\text{Line2D}} = \begin{pmatrix} \frac{L\cos\gamma}{\sqrt{2}} & \frac{L\sin\gamma}{\sqrt{2}} & \frac{-L\cos\gamma}{\sqrt{2}} & \frac{-L\sin\gamma}{\sqrt{2}} \\ \frac{\cos\gamma}{\sqrt{2}} & \frac{\sin\gamma}{\sqrt{2}} & \frac{\cos\gamma}{\sqrt{2}} & \frac{\sin\gamma}{\sqrt{2}} \end{pmatrix}^T. \tag{15}$$

G. Horizontal Line Feature

The horizontal line feature (HLine) is parameterized by two 3-D points

$$\mathbf{x}_f = \left\{\mathbf{x}_{\text{start}}^{3D}, \mathbf{x}_{\text{end}}^{3D}\right\}$$

the end points of the line. This feature illustrates how the M-space representation can constrain the two points that have the same height. Another advantage is that the HLine can be initialized almost immediately with 1 dimension corresponding to the direction of the line, before its location is known. Table II shows the B-matrix for the HLine when it has reached the full dimension. The angle γ here is the same as for a wall feature and represents the angle of the wall in the horizontal plane. Initially only the first row is used, corresponding to the direction of the line. When the position of the line can be triangulated the dimension goes up to three. Note that the position tangential to the line is not measured in our work as it could not be done reliably.

H. Pole Feature

The vertical pole feature (Pole) is parameterized by a radius and a 2-D point. The radius acts like a scalar under transformations. The center point is idealized by treating the pole as having infinite height. This then defines the transformation. As it is hard to imagine a measurement of the pole that does not depend on both the radius and center it would not be possible to initialize less than three M-Space dimensions for pole features.

V. SLAM

To illustrate the use of the M-space representation for SLAM, we present an approach to implementing an EKF SLAM algorithm where the map estimation is separated from the details of the maintaining the features. Although we have chosen the EKF algorithm for illustration, the M-space feature representation is orthogonal to the choice of SLAM algorithm. Thus there is a homogeneous handling of all types of features and an interchangeability of SLAM algorithms.

A. Measurements

Let $\mathbf{v}$ denote a measurement and $\hat{\mathbf{x}}_o = \hat{\mathbf{x}}_{s,f}$ denote the predicted feature coordinates in the sensor frame (o for observation). Note that $\hat{\mathbf{x}}_o$ is not the measurement prediction. Furthermore, let $\boldsymbol{\eta}(\hat{\mathbf{x}}_o, \mathbf{v})$ be the innovation function with expected value of $\mathbf{0}$. For small deviations $\delta\mathbf{v}$ and $\delta\mathbf{x}_o$ from the measurement and predicted feature coordinates we have

$$\delta\boldsymbol{\eta} = J_{\eta v}(\hat{\mathbf{x}}_o, \mathbf{v})\delta\mathbf{v} + J_{\eta o}(\hat{\mathbf{x}}_o, \mathbf{v})\delta\hat{\mathbf{x}}_o \tag{16}$$

where $J_{\eta v}$ and $J_{\eta o}$ are Jacobians of η with respect to $\mathbf{v}$ and $\hat{\mathbf{x}}_o$, respectively.

We can express $\hat{\mathbf{x}}_o$ in terms of entities that are estimated during the SLAM process, the robot pose in the global frame $\mathbf{x}_{m,r}$ and the feature coordinates in the map frame $\mathbf{x}_{m,f}$. Furthermore, the sensor pose w.r.t. the robot, $\mathbf{x}_{r,s}$ is either assumed known or also estimated during SLAM. We find

$$\mathbf{x}_{m,s} = \mathbf{x}_{m,r} \oplus \mathbf{x}_{r,s} \tag{17}$$

$$\mathbf{x}_o = \ominus\, \mathbf{x}_{m,s} \oplus \mathbf{x}_{m,f} \tag{18}$$

where $\oplus$ is the compound operator and $\ominus$ is the inverse compound operator (see [11]). Using a first-order approximation, we get

$$\delta\mathbf{x}_o = \frac{\partial \mathbf{x}_o}{\partial \mathbf{x}_{m,s}}\delta\mathbf{x}_{m,s} + \frac{\partial \mathbf{x}_o}{\partial \mathbf{x}_{m,f}}\delta\mathbf{x}_{m,f} \tag{19}$$

$$\delta\mathbf{x}_{m,s} = \frac{\partial \mathbf{x}_{m,s}}{\partial \mathbf{x}_{m,r}}\delta\mathbf{x}_{m,r} + \frac{\partial \mathbf{x}_{m,s}}{\partial \mathbf{x}_{r,s}}\delta\mathbf{x}_{r,s}. \tag{20}$$

Introducing

$$J_{os} = \frac{\partial \mathbf{x}_o}{\partial \mathbf{x}_{m,s}}$$

$$J_{sr} = \frac{\partial \mathbf{x}_{m,s}}{\partial \mathbf{x}_{m,r}}$$

$$J_{ss} = \frac{\partial \mathbf{x}_{m,s}}{\partial \mathbf{x}_{r,s}}$$

$$J_{of} = \frac{\partial \mathbf{x}_o}{\partial \mathbf{x}_{m,f}}$$

and substituting (20) and (11) into (19), we can write

$$\delta\hat{\mathbf{x}}_o = \begin{pmatrix} \underbrace{J_{os}J_{sr}}_{\text{robot pose}} & \underbrace{J_{os}J_{ss}}_{\text{sensor pose}} & \underbrace{J_{of}\widetilde{B}_f}_{\text{features}} \end{pmatrix} \begin{pmatrix} \delta\mathbf{x}_{m,r} \\ \delta\mathbf{x}_{r,s} \\ \delta\mathbf{x}_{\mathbf{p}} \end{pmatrix}. \tag{21}$$

Note that the Jacobians J_{sr} and J_{ss} do not depend on the kind of feature that is used. They depend only on the transformation to the sensor frame. Furthermore, the Jacobians J_{os} and J_{of} depend only on the three sets of coordinates, (i.e., 3-D, 2-D, and scalars) and the sensor transform. It has the same form for all features and measurements and thus also represent generic calculations.

Going back to (16), we see that it is only the definitions of η, $J_{\eta v}$, $J_{\eta o}$, and B_f that depend on the type of feature. All the other expressions look the same for all features. By exploiting this, SLAM can be implemented separately from the details of the type of feature. In an object oriented setting each feature type has to provide an implementation for η, $J_{\eta v}$, $J_{\eta o}$, and B_f, whereas the rest of the expressions are common.

In Section II, we presented the Jacobians of a general feature parameterization $\boldsymbol{\lambda}$. Comparing terms between (1) from Section II and (21) and (16), we can now see that

$$J_{o\lambda} = J_{of}\widetilde{B}_f B_f. \tag{22}$$

In (21), we have factored out the B_f term from the Jacobian, which has the effect of projecting them to the M-space. Similarly by including the $B_f(\mathbf{x}_f)$ with $\delta\mathbf{x}_f$, we project the perturbations into the M-space.

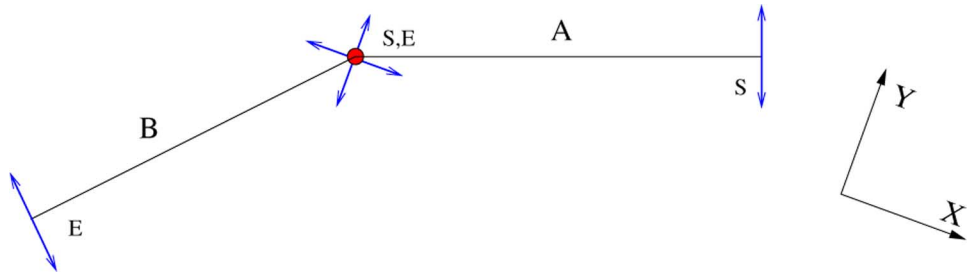

Fig. 2. Two walls with a shared corner point. The nonshared points have not yet been observed and thus only have information orthogonal to the walls. This corresponds to the uncertainty from before the point was shared.

B. EKF Implementation

In EKF SLAM one alternates prediction and updates steps in an iterative algorithm. The predict step calculates the estimated motion of the robot and adjusts the state covariance matrix C_{state}. Then the update step calculates the changes to the state and C_{state} based on the feature measurements.

To implement the EKF in the M-space we need to include the M-space perturbations in the update step. The covariance matrix will now be an estimate of Gaussian distribution in the perturbations around the current feature positions projected into the M-space. Thus, no explicit state vector is computed but rather perturbations in the M-space which are then projected onto the feature coordinates using the $\widetilde{B}$ matrices. The Kalman state perturbations include the feature M-Space perturbations

$$\delta x_{\text{state}} = \begin{pmatrix} \delta \mathbf{x}_r \\ \delta \mathbf{x}_p \end{pmatrix}. \tag{23}$$

The updated robot pose state is then $\mathbf{x}_r(t) + \delta \mathbf{x}_r(t)$. We use the B matrix to project the change in the M-Space to changes to the feature parameters

$$\delta x_f = \widetilde{B} \delta \mathbf{x}_p. \tag{24}$$

Here, $\widetilde{B}(\mathbf{x}_f)$ is calculated every time that $\mathbf{x}_f$ changes. The predict step uses the pose prediction at time t

$$\mathbf{x}_r(t) = f(\mathbf{x}_r(t-1), \mathbf{x}_d(t, t-1)) \tag{25}$$

where $\mathbf{x}_d(t, t-1)$ are the incremental dead-reckoning coordinates and f is a function describing the dead-reckoning motion. The robot pose block C_{rr} of the state covariance matrix C_{state} will also change

$$C_{rr}(t) = J_{rr} C_{rr}(t-1) J_{rr}^T + J_{rd} C_d(t, t-1) J_{rd}^T. \tag{26}$$

Here, we assume the $\mathbf{x}_d(t, t-1))$ are statistically independent of the $\mathbf{x}_r(t-1)$. The Jacobians of the function f are denoted by J_{rr} and J_{rd}. The C_d is the covariance of $\mathbf{x}_d(t, t-1))$. The off diagonal block for the covariance of the robot pose with the M-space perturbations C_{rp} will change by

$$C_{rp}(t) = J_{rr} C_{rp}(t-1). \tag{27}$$

The update step is given by the standard EKF update equations. The only difference is the definition of the linearized state.[4] The update step incorporates the feature observations using the matrix K

$$K = C_{\text{state}}(t) J_{\eta,\text{state}}^T S^{-1} \tag{28}$$

where S is

$$S = \left(J_{\eta,\text{state}} C_{\text{state}} J_{\eta,\text{state}}^T \right) + C_\eta(t) \tag{29}$$

$$J_{\eta,\text{state}} = J_{\eta o} \begin{pmatrix} J_{or} & J_{of} \widetilde{B} \end{pmatrix} \tag{30}$$

$$\delta \mathbf{x}_{\text{state}}(t) = -K \eta(\mathbf{x}_{\text{state}}(t)). \tag{31}$$

The updated covariance of the robot $C_r(t) + \delta C_r(t)$

$$\delta C_{\text{state}}(t)' = -K J_{\eta,\text{state}} C_{\text{state}}(t). \tag{32}$$

We see that it is only in (30) that the feature representation plays a role. The $\widetilde{B}$ here is calculated around the current feature state. The filter maintains the robot pose estimate. It also maintains covariance estimates of the errors in the pose and the M-Space perturbations. There is no state vector for the features in the filter. Instead the features maintain the full $\mathbf{x}_f$ while the filter provides adjustments to those coordinates as shown before.

So to summarize an EKF cycle we carry out the following:

1) predict a new robot pose using (25);
2) adjust the covariance using (26) and (27);
3) update the state using (31) and (24);
4) update the covariance using (32).

VI. Handling Shared Coordinates

As mentioned before in Section IV, part of $\mathbf{x}_f$ can be shared between two different features, i.e., one or more of the coordinates can be common. Thus, a wall described by two 2-D endpoints could share a corner point with another wall. This will force the corner to always be consistent with measurements of both walls.

When a coordinate is shared between two features it is natural to seek a representation for the uncertainty that is not tied to one of the features. Therefore the corresponding submatrix in the B matrix is set to the identity matrix which corresponds to representing the uncertainty in, for example, x and y rather than the distance orthogonal to the wall and its orientation. Fig. 2 shows an example where two walls share a corner point. The arrows mark the direction of the M-space coordinates. When two walls are joined with a shared point the M-space coordinates are redefined. Assuming that none of the other endpoints are known, the two walls would have had M-space coordinates corresponding to distance and orientation. The two nonshared endpoints get

[4]Our notation may bother some readers. Often these equations are written using P's for our C's, H for our $J_{\eta,\text{state}}$ and W for our K.

Fig. 3. This shows maps of our laboratory made with four different robots using the same configuration parameters for the EKF SLAM program. From left to right: a Performance PeopleBot, a Pioneer2DX, a PowerBot, and a custom built robot. The lighter lines are the hand made map shown for reference. The darker lines are the SLAM map. The building is 13 × 39 m.

uncertainty orthogonal to wall and the M-space coordinates for the shared point coincide with the feature coordinate point (i.e., x, y). For example in an EKF SLAM algorithm, the end point of A and the start point of B will map to the same rows of the covariance matrix. The $\widetilde{B}$ corresponding to wall A from Fig. 2 is given by

$$\widetilde{B} = \begin{pmatrix} \frac{L}{L_s}\cos\gamma & 0 & 0 \\ \frac{L}{L_s}\sin\gamma & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{33}$$

where L is the current length of the wall and L_s is the length of the wall at the time when the two lines where joined with a shared point. The scaling by L/L_s serves to compensate for the decrease in angular uncertainty that results when the wall is extended. Without this compensation the wall would appear to become more certain in angle just by sliding the end point along the wall. This is a conservative approximation to the extent that the perpendicular uncertainty is due to measurements of the perpendicular distance to the wall.

We can thus represent physical connections between the simple features to form more complicated composite features. This is done without the need for any explicit enforcement of constraints.

VII. Experimental Verification

In this section, we show experimental results to illustrate that M-space representation is practical for building maps with real data. We will use both laser scanners, cameras, and combinations thereof. To show the platform independence of our representation and implementations, we tried our EKF on four different indoor robots in our laboratory. The results are shown in Fig. 3. Each of these robots was equipped with a SICK laser scanner LMS-200. The SLAM program configuration was the same for each run except for slight differences in the odometry model parameters. Although we collected 32-m laser scans on some of the robots, we limited them to 8 m to be comparable to the other robots.

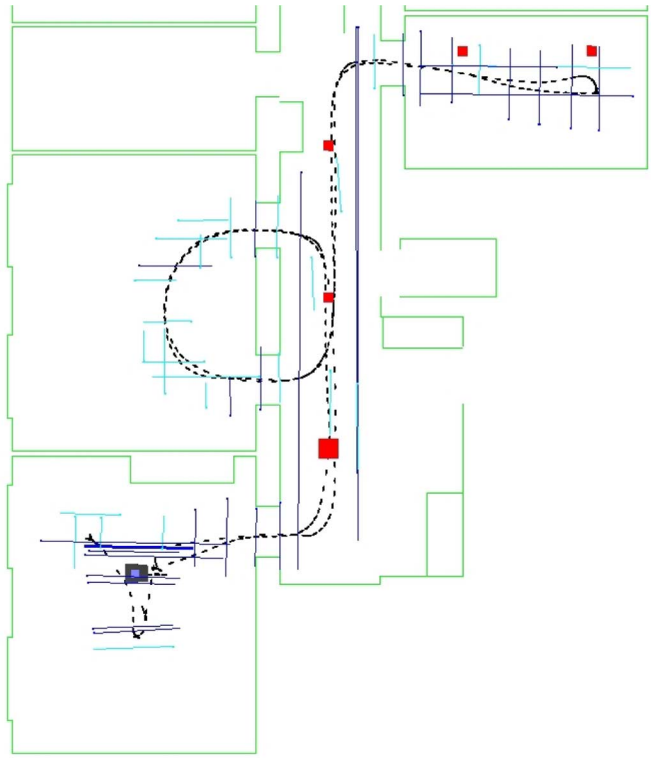

Fig. 4. This is an EKF map made of the vision features on the ceiling of our laboratory, lines, and lamps. The true map of the walls in the laboratory is also shown as a reference. One can see that the estimated robot path passes through the center of the doorways and that the rectangular nature of the ceiling lines is maintained indicating good accuracy. The lighter lines have M-space dimension of 1 (direction only) while the darker lines have 3. The squares show the position of the ceiling lamps which were additional features.

Notice that some walls shared endpoints with adjacent walls forming a corner. These corners had one 2-D point that, when updated, changed both walls. In other words, the corner constraints were explicit in the representation.

To illustrate the M-space features with another type of sensor, we mounted a Philips web camera pointing straight up at the ceiling on a custom robot. We then used the camera and odometry for doing SLAM while using the SICK to first check our accuracy using only the camera and then to show that we could combine data from different sensors for building the map. The ability to integrate in SLAM different sensor and feature types has been previously demonstrated in [22].

The images were 320×240 at 10 Hz. The features were found by using the OpenCV library. Lines and points were extracted. Lines were extracted using the Hough Transform. For point features, we used the center of lamps of circular shape that were detected based on their intensity. The height of the linear structures above the camera varied between 1.5 and 2.5 m.

The ability to use the lines for orientation almost immediately was of great benefit in the corridor where the robot moved parallel to the line for a long distance before turning into a room and finally being able to triangulate the line's position. This helped the robot to stay localized. Figs. 4–6 show the result with

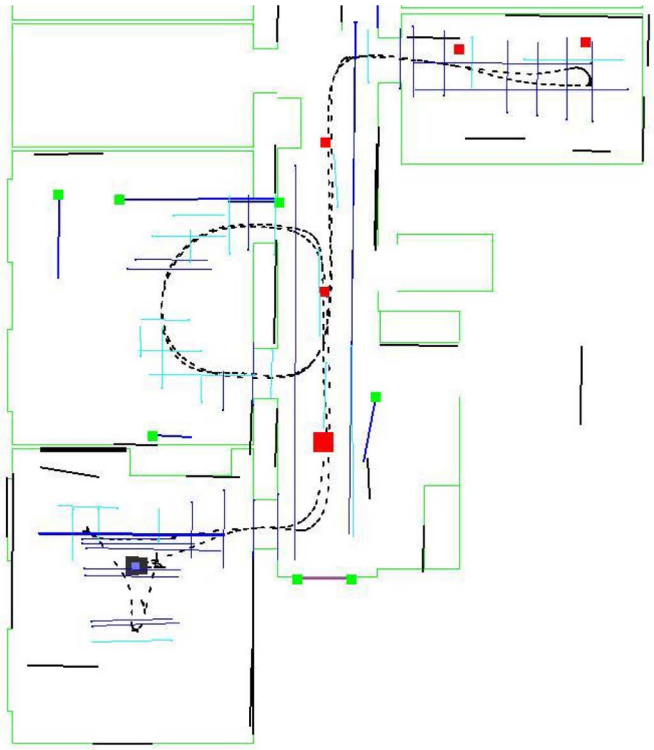

Fig. 5. EKF map obtained using both the camera images and the laser to detect the wall features. Here, we see that the M-space representation allows the combining of different sensor data into one map. Wall endpoints are shown (light squares) when they were used as part of the SLAM estimate. The shade here shows the M-Space dimension of the features. Light grey lines are ceiling lines, angle only (1-D). Walls without endpoints are shown in black (2-D). Dark grey lines with endpoints are walls and without endpoints are 3-D ceiling lines (3-D). The lamps are also 3-D.

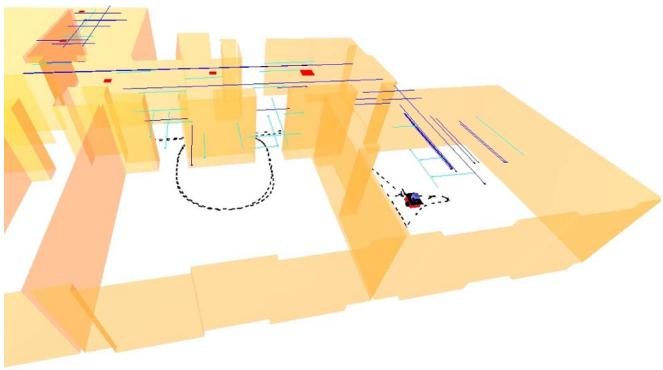

Fig. 6. Same EKF map as in Fig. 4 from another viewing angle. One can see that the features are also at the correct height. The ceiling height is higher in the rooms than in the hallway.

a custom built robot. Fig. 4 uses only the camera images and shows the lines and lamps found on the ceilings. Fig. 4 also uses the SICK laser to detect the walls. Notice that some of the lines found by the SICK are from furniture which is not parallel to the walls. There was no significant change in accuracy when adding the SICK. Robustness was presumably better.

For the Pioneer robot, we mounted a QuickCam web camera with wide-angle lens. This camera had a focal length of 283 pixels as compared to 503 for the Phillips camera used on the first data set. The wider field of view helped by holding the features in view for a longer period of time. Fig. 7 shows the results for a larger map using vision features only for SLAM. The laser scanner, not part of the SLAM estimation, lining up with the walls of the hand made map confirm that the robot remained localized correctly.

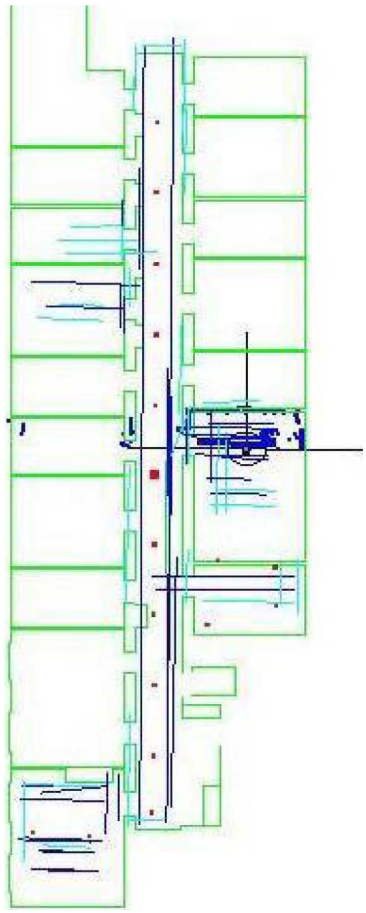

Fig. 7. This shows a map made using vision features with the pioneer robot over a larger area. Again, the walls are shown for reference only and were not part of the estimation process. The path starts and ends in the same room, proceeding up to the end of the corridor, then down and finally back up to the starting room.

VIII. Conclusion

The M-space feature representation allows us to represent all the geometric aspects of features in a generic way. The representation is split into two subspaces, the observable part, M-space, for which the measurements provide a Gaussian estimate of the uncertainty and the remaining dimensions which have uncertainties that cannot be estimated well by Gaussians.

Furthermore, by having the statistical uncertainties in the feature coordinates expressed in a frame attached to the features themselves we avoid the problems such as the lever-arm effect and linearizations that become invalid due to rotations of coordinates in the global frame.

By representing all the geometric information with a generic parameterization that has well defined transformation rules, the SLAM programs can be written without knowing the specific feature details. Different feature types can then be developed separately.

We have demonstrated the usefulness of this idea with an EKF SLAM implementation using two different types of sensors for the feature measurements, four different robots, and three types of features. The details of the feature detection, initialization, and preliminary matching were the same across the different robots.

Some other practical benefits specific to the experiments are shown as follows.

- The finite size of the features is part of the representation and does not need to be dealt with separately.
- The walls of the laser built map contained some corner points which were represented by common parameters for the two walls. Thus, no explicit adjustment was needed to hold the corner at the wall intersection, as in some other representations.
- We were able to represent the partially observable and constrained horizontal line features in the M-space. The M-space allows us to estimate the statistical variations in the initialized and measured directions without any changes to the uninitialized directions. This allows for partial initialization which can be very practical.

APPENDIX
3-D OBSERVATIONS IN A 2-D MODEL

When the ground plane is not flat a 2-D sensor such as a laser scanner will make observations that are not limited to a horizontal 2-D slice of the world. The same happens when the same sensor is mounted on a pan-tilt unit.

We model 2-D points as having an x and y but extend to plus/minus infinity in the z direction. The assumptions behind this are that objects extend from the ground and up and that the sensor does not tilt too far down so that it detects the ground. Give a certain rotation of the 2-D sensor, specified by the three Euler angles θ, ϕ, and ψ, the position of the point in the sensor frame can be calculated using the following "rotation" matrix:

$$R_s^{2D} = \begin{pmatrix} \frac{\cos\theta + \sin\theta \sin\phi \tan\psi}{\cos\phi} & \frac{\sin\theta - \cos\theta \sin\phi \tan\psi}{\cos\phi} \\ \frac{-\sin\theta}{\cos\psi} & \frac{\cos\theta}{\cos\psi} \end{pmatrix}. \quad (34)$$

Note that this is not a rotation in strict sense. One important observation here is that the distance between two points after the transformation will be different.

REFERENCES

[1] G. Weiss and E. v. Puttkamer, "A map based on laserscans without geometric interpretation," *Intell. Auton. Syst.*, pp. 403–407, 1995.

[2] F. Lu and E. Milios, "Optimal global pose estimation for consistent sensor data registration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1995, pp. 93–100.

[3] J. L. Crowley, F. Wallner, and B. Schiele, "Position estimation using principal components of range data," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1998, pp. 3121–3128.

[4] R. Sim and G. Dudek, "Learning visual landmarks for pose estimation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1999, pp. 1972–1978.

[5] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1985, pp. 116–121.

[6] B. Yamauchi, A. Schultz, and W. Adams, "Mobile robot exploration and map-building with continous localization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1998, pp. 3715–3720.

[7] S. Thrun, D. Fox, and W. Burgard, "A probalistic approach to concurrent mapping and localization for mobile robots," *Auton. Robots*, vol. 5, pp. 253–271, 1998.

[8] J. E. Guivant and E. M. Nebot, "Optimization of the simultaneous localization and map-building algorithm for real-time implementation," *IEEE Trans. Robot. Autom.*, vol. 17, no. 3, pp. 242–257, Jun. 2001.

[9] S. Se, D. G. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *Int. J. Robot. Res.*, vol. 21, no. 8, pp. 735–58, 2002.

[10] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. Int. Conf. Comput. Vision (ICCV)*, 1999, pp. 1150–57.

[11] R. Smith, M. Self, and P. Cheeseman, "A stochastic map for uncertain spatial relationships," presented at the 4th Int. Symp. Robot. Res., Cambridge, MA, 1987.

[12] P. Moutarlier and R. Chatila, "Stochastic multisensory data fusion for mobile robot location and environmental modelling," in *Proc. Int. Symp. Robot. Res.*, 1990, pp. 85–94.

[13] J. Leonard and H. Durrant-Whyte, "Mobile robot localization by tracking geometric beacon," *IEEE Trans. Robot. Autom.*, vol. 7, no. 3, pp. 376–382, Jun. 1991.

[14] J. J. Leonard and R. J. Rikoski, "Incorporation of delayed decision making into stochastic mapping," in *Experimental Robotics VII*. New York: Springer-Verlag, 2001, vol. 271, pp. 533–542.

[15] J. Tardós, "Representing partial and uncertain sensorial information using the theory of symmetries," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1992, pp. 1799–1804.

[16] K. Arras, N. Tomatis, and R. Siegwart, "Multisensor on-the-fly localization using laser and vision," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2000, pp. 462–476.

[17] O. Faugeras, N. Ayache, and B. Faverjon, "Building visual maps by combining noisy stereo measurements," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1986, pp. 1433–1438.

[18] J. A. Castellanos, J. Montiel, J. Neira, and J. D. Tardós, "The SPMAP: A probabilistic framework for simultaneous localization and map building," *IEEE Trans. Robot. Autom.*, vol. 15, no. 5, pp. 948–952, Oct. 1999.

[19] J. L. Crowley, P. Stelmaszyk, T. Skordas, and P. Puget, "Measurement and integration of 3-D structures by tracking edge lines," *Int. J. Comput. Vision*, vol. 8, no. 1, pp. 29–52, 1992.

[20] J. A. Castellanos and J. D. Tardós, *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*. Norwell, MA: Kluwer, 1999.

[21] P. Newman, J. Leonard, J. Tardós, and J. Neira, "Explore and return: Experimental validation of real-time concurrent mapping and localization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2002, pp. 1802–1809.

[22] J. A. Castellanos, J. Neira, and J. D. Tardós, "Multisensor fusion for simultaneous localization and map building," *IEEE Trans. Robot. Autom.*, vol. 17, no. 6, pp. 908–914, Dec. 2001.

John Folkesson (M'02) received the B.S. degree in physics from Queens College, New York, in 1983, and the M.S. and Ph.D. degrees in computer science from the Royal Institute of Technology (KTH), Stockholm, Sweden, in 2001 and 2005, respectively.

He is currently a Postdoctoral Fellow with the Massachusetts Institute of Technology (MIT), Cambridge. He has designed and implemented several autonomous robot systems and is one of the main contributors to the CURE robotics C++ libraries.

Patric Jensfelt (M'96) received the M.Sc. degree in engineering physics and the Ph.D. degree in automatic control from the Royal Institute of Technology (KTH), Stockholm, Sweden, in 1996 and 2001, respectively.

Between 2002 and 2004, he worked as a Project Leader in two industrial projects. He is currently a Senior Researcher with the Centre for Autonomous System and the co-PI for the European project "CoSy" at KTH. He is also one of the main contributors to the CURE robotics C++ libraries. His main research interests include the fields of mobile robotics, systems integration, and human robot interaction.

Henrik I. Christensen (M'86) received M.Sc. and Ph.D. degrees from Aalborg University, Aalborg, Denmark, 1987 and 1990, respectively.

He is the KUKA Chair of Robotics and the Director of the Center for Robotics and Intelligent Machines with the Georgia Institute of Technology, Atlanta. He also held positions with the Royal Institute of Technology, Stockholm, Sweden, Aalborg University, and Oak Ridge National Laboratory, Oak Ridge, TN. He was the founding coordinator of European Robotics Network 013 EURON (1999–2006). His research interests include systems integration, estimation, and human robot interaction. He has published more than 230 contributions across vision, robotics, and AI. He serves on the editorial board of IJRR, RAS, AI Magazine, Service Robotics, and Autonomous Robots. He is a consultant to agencies and companies across three continents.