Robotics and Autonomous Systems 61 (2013) 749–767

Contents lists available at SciVerse ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Building 3D visual maps of interior space with a new hierarchical sensor fusion architecture

Hyukseong Kwon [1], Khalil M. Ahmad Yousef*, Avinash C. Kak

School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA

HIGHLIGHTS

- Our proposed system creates high quality 3D maps of interior space.
- We use a hierarchical approach to fusing vision and range data.
- We show how the uncertainties can be managed with interval based logic.
- Interval based logic allows fusion of information at different abstraction levels.

ARTICLE INFO

Article history:
Received 17 January 2012
Received in revised form
7 January 2013
Accepted 19 April 2013
Available online 9 May 2013

Keywords:
Sensor fusion
Hierarchical map building
SLAM
Dense visual map

ABSTRACT

It is now generally recognized that sensor-fusion is the best approach to the accurate construction of environment maps by a sensor-equipped mobile robot. Typically, range data collected with a range sensor is combined with the reflectance data obtained from one or more cameras mounted on the robot.

In much of the past work on sensor fusion in hierarchical approaches to map construction, the fusion was carried out only at the lowest level of the hierarchy. As a result, in those approaches, only the fused data was made available to the higher levels in the hierarchy. This implied that any errors caused by sensor fusion would propagate upwards into the higher level representations of an interior map. *Our work, on the other hand, checks for consistency between the data elements produced by the different sensors at all levels of the hierarchy.* This consistency checking is carried out with the help of an interval-based representation of uncertainties in the sensor data.

In addition to demonstrating that our approach to the fusion of range and image data results in dense 3D maps of the interior space, we also provide validation of our overall framework by presenting a set of loop closure results. These results demonstrate that our overall errors in the maps remain small (within 0.91% of the distance traveled for map construction) even when the robot has to traverse over large loops inside a building.

1. Introduction

For a mobile robot to figure out where it is and to determine how it should get to its destination, it needs some kind of a model of the environment. In some cases, such models of space – usually called maps – can be constructed from prior knowledge of the layout of the environment, its geometry, its topology, the reflectance properties of its surfaces, and, as recently demonstrated, even functional properties. But there exist many important situations where this type of prior knowledge is not available, at least not available at the level of detail that would be needed by a robot trying to navigate around in the environment. When such is the case, one of course has the option of actually generating the maps manually. But doing so can be difficult and expensive. An easier alternative consists of letting the robot make its own maps of the interior. When done properly, such a process can result in maps that directly take into account the different properties of the sensors that would subsequently be used for navigation. Incorporating sensor limitations into maps generated by other means, say manually or from CAD models of the interior, is highly non-trivial.

For a robot engaged in map building, it must obviously know its position in a world frame so that the spatial elements constructed from the current sensed data can be properly situated in the world frame. This implies that an effort at map building cannot be separated from self-localization. The joint problems of map building and localization are sometimes referred to by the acronym SLAM that stands for "Simultaneous Localization And Mapping" [1–3]. Since map building cannot be carried out without localization, in

* Corresponding author. Tel.: +1 7654943456; fax: +1 765 494 6440.
E-mail addresses: hyukseong.kwon@usafa.edu (H. Kwon), kahmadyo@purdue.edu, khalil.yousef@gmail.com (K.M. Ahmad Yousef), kak@purdue.edu (A.C. Kak).
[1] Present address: 2354 Fairchild Dr. 2F6, US Air Force Academy, CO 80840, USA.

0921-8890/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.robot.2013.04.016

the rest of this paper when we mention "map building", we would be referring to SLAM.

Early researchers on map building used just single sensors for the data. Typically, the sensors used were either of the type that looked straight ahead to determine the range to the structures in the environment, or that recorded reflectance and texture properties of the surfaces in the environment. The former type of sensors included ultrasonic and laser sensors that yielded the range measurements [4–9], and the latter type included monocular and stereoscopic camera systems [10–18].

More recently, the focus in the map-building community has shifted predominantly to the building of high-quality interior maps by fusing information from multiple disparate sensors. This work can be categorized in the following manner:

1. Mapping image *features* from a single camera onto the range data [19,20];
2. Mapping image *features* extracted with a stereo camera system onto the range data [21–24]; and
3. Constructing dense 3D maps by wholesale mapping of single-camera images onto the range data [25–27].

These three different approaches yield increasingly dense 3D maps of the indoor environment.

As an exemplar of the work that falls in the first category, the contribution by Castellanos et al. [20] is based on the projection of vertical line features extracted from a camera onto the range data from a 2D laser rangefinder. The chosen vertical line features are typically at the corners in the indoor environment. The authors used an extended Kalman filter (EKF) to merge the local maps and for robot self-localization.

To briefly describe a contribution that falls in the second category, the authors of [21] extract line features from stereo images and then map them onto the range data for 3D map building. Stereoscopic vision allows these authors to place non-vertical lines of the indoor scene in the 3D maps constructed by the system. These authors have also used the EKF for merging local maps. In another contribution in the second category, a somewhat different approach to the merging of stereoscopic images with range data has been taken by Baltzakis et al. [23]. These authors simply harvest the 3D features extracted from the stereoscopic images and from the range data and, subsequently, use these features in map construction. For an even more recent contribution in the second category, Ahn, Choi, Doh, and Chung [24] extract the point and line features from sonar data; these features are then used to key into a database of visually recognizable objects. Confirmation of the presence of the objects is achieved by using an EKF-based algorithm to fuse the sonar features with the features extracted from a pair of stereo cameras. However, note that the main goal of sensor fusion in the contribution in [24] is to achieve the high accuracy in robot localization, as opposed to constructing accurate high-density maps of the interior space.

Finally, to further elaborate a contribution that falls in the third category, a special feature of the work carried out by Liu et al. [27] is that the system makes an optimum choice for the intensity level to be associated with a surface whose position and orientation may be determined from the range data. In particular, the authors use the expectation–maximization algorithm to combine several intensity images and range data in order to select the best pixel value to be associated with a point on a 3D surface in the map.

In all of the work cited above that used multiple sensors, either there was no hierarchy of data objects involved in the map building process – so sensor fusion could be carried out straightforwardly at the only data abstraction level permitted – or, when the map building process entailed creating data objects in a hierarchical fashion, sensor fusion was carried out only at the lowest level of the hierarchy.

Our work is based on the premise that, first, it is necessary to use a hierarchical representation of space when maps are desired for large interior spaces, and, second, the sensor fusion must be carried out at all levels of the data object hierarchy. It should be possible for sensor-fusion-created spatial assertions to be undone if the higher-level objects in the hierarchy cannot satisfy the geometrical constraints placed on them.

Using a hierarchical representation of space and using an interval-based representation of uncertainty that allows sensor fusion to be carried out at all levels of the hierarchy, the work we present in this paper shows that a mobile robot can construct high-quality (meaning dense and preserving reflectance properties of the surfaces) maps of the interior space, while at the same time maintaining with high accuracy its localization information, by combining simple 2D range measurements acquired quickly with a laser sensor with the 3D reconstructions from a stereo head.

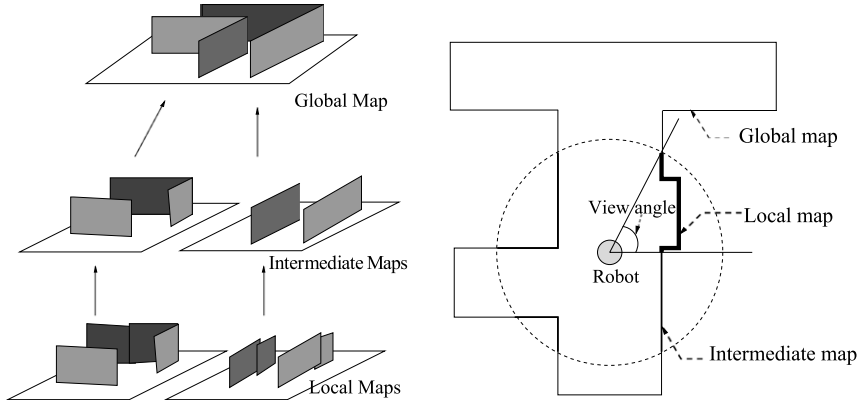
At the lowest level of processing in the SLAM presented in this paper, our system extracts line features from the range data produced by a laser scanner that scans the space horizontally in a plane close to the floor. These line features correspond typically to the flat vertical surfaces such as walls in the environment. The system then associates 2D camera images of these surfaces with the range line features. This process can be envisioned as associating a vertical surface with a range line and then texture mapping the surface with the camera images. The locations of the vertical surfaces are further adjusted with the help of 3D vertical lines extracted from stereo images that correspond to jump discontinuities identified in the range data.

In recent years the loop closure error [28–31] has become an important indicator of the accuracy of a SLAM framework. Loop closure involves letting a robot wander in an environment containing a loop for the purpose of map construction. We want to know that when the robot arrives back at a location it has seen previously, does it recognize that location and what is the positional error associated with that recognition? Although detecting previously visited positions and fixing loop closure errors is not within the main scope of this paper, we will nevertheless present several results that demonstrate the loop closure error associated with our framework. We also provide a comparison between our results and those obtained with a state-of-the-art SLAM technique called GMapping [32].

The remainder of this paper is organized as follows: Section 2 that follows will present the overall sensor fusion architecture that we use for SLAM. Subsequently, Sections 3–5 will present how the environment maps are built at the local level, at the intermediate level, and, at the global level, respectively. Section 6 will present some sample experimental results that demonstrate the ability of our SLAM framework to construct dense high-quality maps of the interior space. We will take up the subject of loop closure errors separately in Section 7. In that section, we will show how the loop closure error varies with the length of loop traversal. Finally, we will conclude in Section 8 where we also present some ideas on how this work can be extended.

2. A hierarchical architecture for sensor fusion for SLAM

The approach to SLAM presented in this paper consists of a hierarchy of map-building procedures: (1) Local Map Building (LMB), (2) Intermediate Map Building (IMB), and (3) Global Map Building (GMB). LMB consists primarily of extracting line features from the range data and attaching 2D camera images with the vertical surfaces corresponding to the range lines. IMB consists of stitching together the local maps for different rotational orientations of the robot (at generally the same translational position) for a 360° reconstruction of the interior space for a given translational position of the robot. For GMB, the robot integrates together the intermediate maps constructed at different translational positions of



(a) Representative hierarchy of our map building. (b) Coverage for each map building level.

Fig. 1. Hierarchical map building.

the robot.[2] Fig. 1(a) is an iconic representation of the three mapping levels used in our SLAM, these being Local, Intermediate, and Global. A planar view of the spatial coverage associated with each level is shown in Fig. 1(b). Algorithm 1 presents the main operations that are carried out at the three levels.

```
Start
    Initialization
    while need_more_maps do
        while ∼ all_orientation_covered do
            LMB
            Robot position adjustment
            Execute turning motion for next LMB
        end
        Merge local maps into an intermediate map
        Adjust robot position
        Merge intermediate maps for GMB
        Adjust robot position
        Execute translation motion for next IMB
    end
end
```

Algorithm 1: A top-level algorithmic view of the map building process.

We now draw the reader's attention to the data and uncertainty flow diagram of Fig. 2. Shown at the right of this figure are the same three abstraction levels we showed in Fig. 1—Local, Intermediate, and Global. The next two subsections separately explain the data flow and the uncertainty flow in the overall diagram of Fig. 2.

2.1. Data flow in the sensor fusion architecture

With regard to the data flow shown in Fig. 2, within each level or when crossing level boundaries, the data always flow from more reliable processing modules to less reliable processing modules. *This is based on the rationale that the outputs produced by the more reliable processing modules can aid in the interpretation of the information generated by the less reliable processing modules*. Data *reliability* depends on the uncertainty range that we must place on each data element at the output of a module. For example,

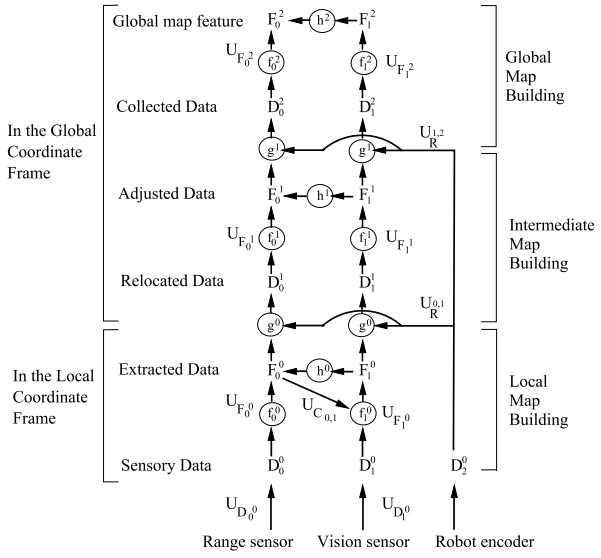


Fig. 2. Data and uncertainty propagation in the our map building.

range data are more reliable than vision data because of the smaller standard deviation associated with the range points (and with the straight lines that are fit to the range point) as compared to the standard deviations that we must associate with stereo-triangulated points and the stereo-extracted lines. Though the robot-wheel-encoder-estimated values are highly related to the feature locations obtained from the vision sensors and the range sensor, the encoder values themselves cannot be adjusted in the Global Coordinate Frame without the help of the other sensory information. Therefore, the inputs from the robot wheel encoder are located in the third column in our sensor fusion model. These values are used at the IMB level for finding the approximate location of the robot in the Intermediate Coordinate Frame (which is the same as the Global Coordinate Frame).

With regard to the notation shown in Fig. 2, we associate Level 0 with "LMB", Level 1 with "IMB", and Level 2 with "GMB". The input to each level is referred to as *data* and denoted by the symbol D. The output of each level is referred to as *features* and denoted by the symbol F. The function or the process that transforms D into F in each level is represented by the small letter f. Obviously, the output F for each level becomes the input D for the next level, except for a transformation that will be represented by the small letter g and the data reliability function denoted by h. As we will explain later, the role of the h functions is to check for consistency between the information produced by the different sensors.

[2] It is important to mention that, for the SLAM framework presented in this paper, the robot operates in stop–sense–move mode. That is, the sensory information is captured only when the robot has stopped for a moment between successive moves. With regard to the motions, the robot either executes pure translations or pure rotations at any given time.

With regard to the subscripts and the superscripts on D and F, in the symbol D_i^j, the subscript i stands for the type of data and the superscript j for the level in the hierarchy. The output of the data extraction/aggregation process at each level is represented by F_i^j where the subscript i again stands for the type of sensory data and the superscript j the level.

Keeping in mind the general notation presented above, we now explain the nature of the various entities in the data flow presented in Fig. 2:

- The entity D_0^0 that is seen in the "Range sensor" flow line is a list of range points collected from one position of the robot by scanning with the range sensor (we use the SICK-LMS 200 range sensor mounted on the robot as shown in Fig. 3(a)). These points are all confined to a horizontal plane parallel to the floor (about 8 in. off the floor) in a 180° angular field of view. Each range scan is limited to a maximum distance of 16 m from the robot and has an angular resolution of 0.5°, which results in 361 range point measurements. Fig. 4(b) shows an example of a range scan taken in a hallway system that is graphically depicted in Fig. 4(a). Each item in the list of range points is a pair (x, y) of the coordinate values with respect to the origin of the Robot Coordinate Frame. The calibration between the Range Coordinate Frame and the Robot Coordinate Frame is provided by the manufacturer of the robot. The entity D_1^0 in the "Vision sensor" flow line is a pair of images from the stereo head that is mounted on the robot as shown in Fig. 3(a). Each of these images exists in the Camera Coordinate Frame that is also shown in the figure. And, D_2^0 that is seen in the "Robot encoder" flow line is the robot position obtained from the robot wheel encoders. The robot position is expressed by the translational coordinates (x, y) and the orientation angle ϕ with respect to the initial location and orientation of the robot (world frame). In Fig. 3(b) we show the relative locations of the Camera/Range/Robot Coordinate Frames.
- The entity F_0^0 that you see in the "Range sensor" flow line is the range-data output for the 0th level. It consists of a set of straight lines fit to the range data D_0^0. Fig. 4(c) shows an example of straight line fitting to the range data that is shown in Fig. 4(b).
- The entity F_1^0 that you see in the "Vision sensor" flow line is the stereo vision output for the 0th level. Since the stereo process is capable of outputting different types of objects, F_1^0 is in actuality a union of three different types of objects: vertical line features, visual point features, and wall images. In order to distinguish between these three types of objects, we will use the notation $F_{1,0}^0$ to represent the stereo-extracted vertical line features that, in the absence of uncertainties, should correspond to the discontinuities in the parallel-to-the-floor straight lines extracted from the range data. Fig. 4(f) shows an example of the stereo-extracted vertical line features from the stereo image pair taken at the robot location shown in Fig. 4(a). Fig. 4(g) shows an example of the projection of range-data jump discontinuities into one image of a stereo pair of images. Fig. 4(h) shows the same projection of the range-data jump discontinuities, but also shows the stereo-extracted vertical line features. We use the notation $f_{1,0}^0$ to represent the *function* that extracts the vertical line features from the stereo images and the corresponding wall planes. By the same token, we will use the notation $F_{1,1}^0$ to represent the stereo-extracted landmark points after they are projected onto the wall plane. Fig. 4(j)–(l) shows examples of these extracted landmarks and their projection on the wall planes and wall images.

  Finally, we use the notation $F_{1,2}^0$ to represent the wall images that result from the projection of the left or right photometric image onto the wall plane. Fig. 4(m) shows an example of the wall images and the overall constructed local map. Obviously, the accuracy of the map building process can be gauged by the accuracy of the projections of the line and point features on the wall plane. To give the reader a sense of this accuracy, Fig. 4,(d) and (e), shows, respectively, the projections of the range point measurements and the fitted lines onto the stereo images. These projections speak to the high accuracy with which the local maps are constructed.

  It is important to realize that the concept of a wall plane used in the above explanation is not a static concept. During the map building process, the merging of a local map with the intermediate map can result in the readjustment of the position and the orientation of the wall plane. Obviously, as the wall plane changes, all of the projections on the wall plane will also change. The ultimate test of this adjustment is that when the line features, point features, and the photometric images are projected onto the wall plane, they must all be in congruence.

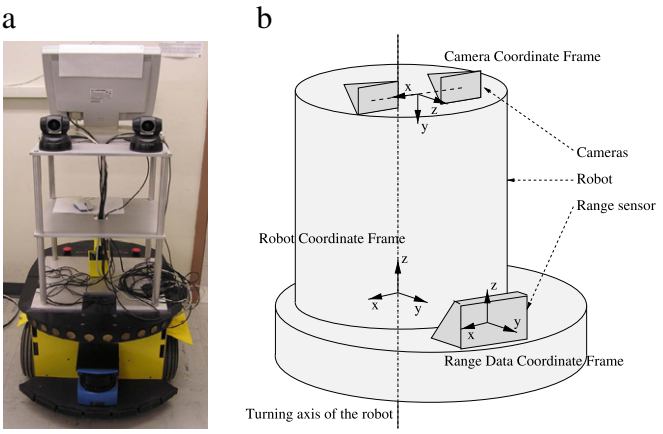


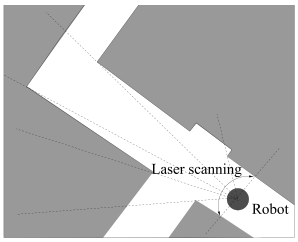
Fig. 3. (a) A 2D *SICK-LMS 200* laser range finder and a pair of *Sony EVI-D100* stereo cameras installed by us on the PowerBot robot from *ActivMedia*. (b) Relative locations of the Camera/Range/Robot Coordinate Frames.

The above explanation takes care of the 0th level of the data flow in the hierarchy of Fig. 2. What follows is an explanation of the other levels in that figure:

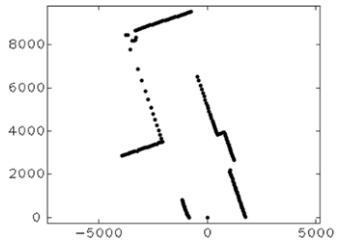
- All of the data entities in Level 0 of the hierarchy of Fig. 2 are defined in the Local Coordinate Frame of the robot. Level 1 situates the Level 0 data entities in the Intermediate Coordinate Frame. This then constitutes the input to Level 1 of the hierarchy. More specifically, D_0^1 represents the set $\bigcup_{i=1}^{k} \{F_0^0\}_i$ where k is the number of the local maps at each Intermediate Coordinate Frame, and D_1^1 represents the set $\bigcup_{i=1}^{k} \{F_1^0\}_i$.
- The output of Level 1 is F_0^1 for the range sensor and F_1^1 for the stereo fusion sensor. This output is the merged map features in the Intermediate Coordinate Frame.
- The entities D_0^2 and D_1^2 are the same as $\{\{F_0^1\}_k, \{F_0^2\}_{k-1}\}$ and $\{\{F_1^1\}_k, \{F_1^2\}_{k-1}\}$ respectively.
- The entity F_0^2 represents the global range line map and the entity F_1^2 the global 2D wall images. The 2D wall images in F_1^2 correspond to the parallel-to-the-floor range lines in F_0^2.

We will now focus on the within-level functions shown in Fig. 2. As already mentioned, these functions extract the output F from the input D at each level. To be consistent with how the subscripts and the superscripts are interpreted, f_0^j denotes the jth function in the "Range sensor" flow line. Along the same lines, f_1^j denotes the jth function in the "Vision sensor" flow line. To present in greater detail what is accomplished by each of the functions:

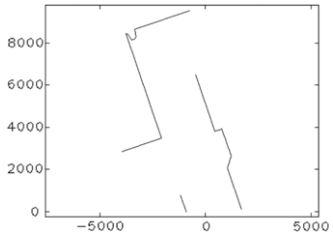
- The function f_0^0 carries out range line extraction from the point range data (see Fig. 4(c)). Straight lines are fit to range data D_0^0 after we segment the data on the basis of large jump



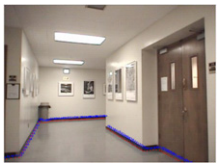
(a) A robot in the environment.



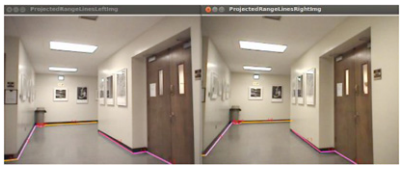
(b) Range point features D_0^0.



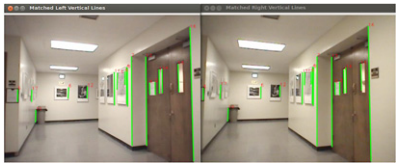
(c) Range line features F_0^0.



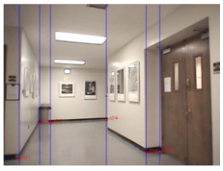
(d) The range points D_0^0 projected on the left image of the stereo head D_0^1.



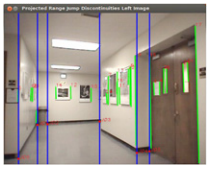
(e) The range lines F_0^0 projected on the stereo image pair D_0^1.



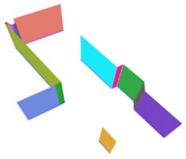
(f) Stereo extracted and matched vertical line features $F_{1,0}^0$.



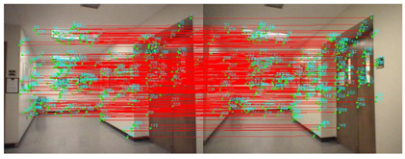
(g) Projection of range jump discontinuities into one image of a stereo pair.



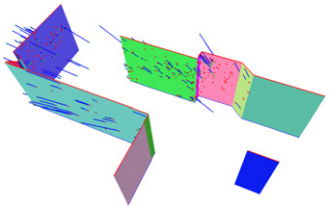
(h) Projection of the range jump discontinuities together with $F_{1,0}^0$ features.



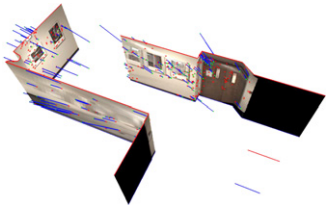
(i) Generated wall planes.



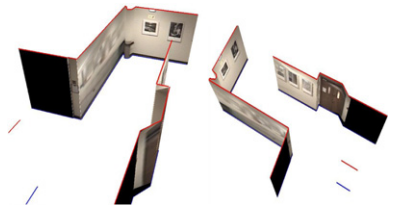
(j) Stereo extracted and matched SURF point features $F_{1,1}^0$.



(k) Triangulated SURF points $F_{1,1}^0$ (green) and their projection (red) on the wall planes connected through blue lines.



(l) Triangulated SURF points $F_{1,1}^0$ (green) and their projection (red) on the wall images connected through blue lines.



(m) Two views of LMB showing the wall image features $F_{1,2}^0$ after projecting them onto the wall planes.

Fig. 4. An example of the data entities that go into the construction of a local map. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

discontinuities. In the experiments we describe in Sections 6 and 7, a distance threshold of 50 cm and a change in slope by at least 30° to consecutive range points is used to identify jump discontinuities. In the same experiments, the minimum number of range points required to instantiate a line fitting was set to 3. A straight line is fit to each segment using a least-squares algorithm. A line is accepted only if the fit error is below a threshold. Should this error exceed the threshold for a segment, the segment is bisected and the process repeated for each portion separately.

- The function f_1^0 carries out stereo extraction of landmarks and line features and then projects the camera images on the wall planes corresponding to the range lines generated by f_0^0. Fig. 4(j), (f) and (m) illustrates our implementation of this function.
- The functions f_0^1 and f_1^1 adjust the coordinates associated with the features extracted in Level 0 to account for robot motion errors. This is done by enforcing geometrical consistency constraints. Note that, the features output by Level 0 are in the Local Coordinate Frame of the robot. Subsequently, they are relocated in the Intermediate Coordinate Frame by the Level 0 to Level 1 transformation represented by the function g^0 shown in Fig. 2.
- The functions f_0^2 and f_1^2 merge the individual line features at the output of Level 1 into larger features that may cover the space that was examined through multiple intermediate maps in the Global Coordinate Frame.

This sensory fusion step in Level j, as represented by the function f_i^j, can be expressed in the following form:

$$F_i^j = f_i^j\left(D_i^j\right). \tag{1}$$

In line with what we mentioned earlier about the role played by the functions denoted as f_1^j, it is in reality a composition of three functions: $f_{1,0}^j$, $f_{1,1}^j$, and $f_{1,2}^j$. This implies that F_1^j is in actuality a union of the outputs $F_{1,0}^j$, $F_{1,1}^j$, and $F_{1,2}^j$. The relationships between the component functions and these individual outputs can be expressed in the following form:

$$F_{1,m}^j = f_{1,m}^j\left(F_0^j, D_1^j\right) \tag{2}$$

where $m \in \{1, 2, 3\}$. Note that, the superscript j denotes the level in the hierarchy.

Now, we will focus on the relationships between the output of each level and the input to the next level as the data flows up the hierarchy. These relationships are expressed via the g functions shown in Fig. 2. More specifically,

$$D_i^{j+1} = g^j\left(F_i^j\right) \tag{3}$$

where g^j represents a coordinate transformation. For example, g^0 represents the coordinate transformation from the Local Coordinate Frame in which the data is collected in Level 0 to the Intermediate Coordinate Frame in which a map is accumulated as the robot rotates in order to collect different local maps. Also, g^1 represents the coordinate transformation from the Intermediate Coordinate Frame to the Global Coordinate Frame.

Finally, the function h^j checks for consistency between the data elements produced by the different sensors, range and vision, on the basis of their interval-based uncertainties, as described in the next subsection. It is expressed as in the following form:

$$F_0^j = h^j\left(F_0^j, F_1^j\right). \tag{4}$$

The implementation of h^j involves reconciling the uncertainties associated with the features. The next subsection, which focuses on how the uncertainties are represented, presents in Eq. (6) a formula that constitutes the implementation of h^j.

2.2. Uncertainty flow in the sensor fusion architecture

We believe that SLAM would be impossible without a well principled approach to the management of the uncertainties that are always present. At the lowest level of the hierarchy of Fig. 2, at the least the system must contend with the uncertainties associated with robotic motions during the course of map building. Since the actual position of the robot is highly likely to be different from its commanded motion, that fact must be taken into account when different local maps or intermediate maps are merged together. In addition to the uncertainties related to robotic motions, we must also deal with the noise in the sensor data.

We have used an interval based approach to the management of uncertainties in our SLAM. There is much tradition to using interval based approaches for modeling uncertainty, especially when it is difficult to assume any specific probability distributions for the parameters [33–36]. In support of using interval based uncertainties, here is a quote from [35]: "Such models are useful for modeling complex sensing devices, such as cameras (where the observations are accurate up to the pixel resolution), as well as networks of heterogeneous sensors where it is difficult to obtain an exact sensing model for each sensor". This quote speaks directly to the conditions that prevail in our experiments—it is extremely difficult to estimate the probability distribution of the uncertainties associated with the fusion of observations from different sensors. It is also non-trivial to fit probabilistic models to the uncertainties associated with robotic motions.

Our interval based approach to the management of uncertainty differs from other well known approaches for doing the same. By and large, all other approaches are based on Gaussian modeling of the uncertainty variables. For example, there is the approach by Smith et al. [37] in which an extended Kalman filter is used to keep track of the uncertainties through the covariance matrices of the relevant variables. For another approach to uncertainty management, Beeson et al. [38] have proposed a hierarchical map building method using the dual notions of metrical and topological maps. For the metrical maps, their method uses Gaussian modeling and extended Kalman filtering to keep track of the uncertainty variables and to merge them across the different levels of the map hierarchy. However, for the topological map, the merging of the uncertainties is carried out through a maximum-likelihood based framework. Yet another uncertainty management approach is incorporated in the Atlas framework by Bosse et al. [39]. This approach is based on simultaneous localization and mapping that again uses Gaussian modeling of the uncertainty variables. Uncertainties are set to the volume of the n-sigma hyper-ellipsoid of the probability distribution around each detected feature that is incorporated in the map. Comparing all these prior approaches to ours, a most significant difference is a consequence of the fact that our interval-based approach implies that the variables are uniformly distributed over their associated uncertainty intervals. We believe that this makes our approach more sensitive to even small overlaps in the uncertainty intervals when we combine the uncertainties associated with multiple variables.

In what follows, we define four different types of uncertainties, all based on the interval representation of uncertainty:

- $U_{D_i^0}$: *Sensing Uncertainty* that is associated with the data D_i^0 obtained through the ith sensor. For the range sensor, we assume a worst-case constant value for the uncertainty interval for all range measurements in a given hallway system. In general, the uncertainty associated with a range measurement is roughly linearly dependent on the distance to the target. The worst-case value for this uncertainty would correspond to the situation when the robot is facing straight ahead along the long direction of a hallway. For a hallway that is roughly 3 m wide, a value of 7 cm works out well for the uncertainty interval for the range sensor on our robot. For a stereo vision sensor, it is related to the stereo calibration errors and the distance between the stereo head and the wall plane whose images are being subject to stereo imaging. The uncertainty interval $U_{D_i^0}$ is used for determining the uncertainly interval $U_{F_i^0}$ that is described next.
- $U_{F_i^j}$: *Feature Uncertainty* that is associated with feature extraction or with feature location adjustment. Recall that features at Level j are extracted from the data D_i^j and that the features thus obtained at level j are represented by F_i^j. Later in this section we will mention how these uncertainties are obtained from the statistical parameters associated with the range and vision data.
- $U_{C_{i,i+1}}$: *Calibration Uncertainty* is the uncertainty related to the calibration errors between two different sensors, the ith sensor and the $(i+1)$th sensor. More specifically, we are talking about the uncertainty related to the calibration between the range and the vision sensors, $U_{C_{0,1}}$. The camera coordinate frame of the stereo head and the range data coordinate frame of the range sensor are related by a rotation matrix and translation vector. The standard deviation of the length of that translation vector is what defines $U_{C_{0,1}}$.[3]
- $U_{R^{j,j+1}}$: *Robot Motion Uncertainty* is the uncertainty related to the movement of the robot. Referring back to Fig. 2, the notation $U_{R^{0,1}}$ applies to the motion uncertainties associated with local map construction. Recall, each local map is constructed with a fixed rotational orientation of the robot. This orientation must be changed to create the next local map. The rotational uncertainty associated with this change in robot orientation is represented by $U_{R^{0,1}}$. By the same token, after an intermediate map is

[3] With regard to the procedures used for calibration, we use Zhang's algorithm [40] and the MATLAB Camera Calibration Toolbox available from [41] for the calibration of the cameras in the stereo head. For the calibration between the range sensor and the stereo cameras, we developed our own routines that are not presented here on account of space limitations.

constructed from all the local maps at a fixed translational position of the robot, the robot must execute a translational motion to get to the next position for the next set of local maps. The motion uncertainty associated with this translational motion is represented by $U_{R^{1,2}}$. In general, the motion uncertainties are caused by the slippage in the wheels, the result being that the robot positions as reported by the encoders will often be erroneous. As the reader would expect, the robot motion uncertainty grows differently for straight-line motions and turning motions. To estimate this uncertainty (either $U_{R^{0,1}}$ or $U_{R^{1,2}}$), we first find the relationship between the robot's real motions and the wheel encoder readings. To do this, we first locate the center of the Robot Coordinate Frame as the self-turning axis of the robot, and we set the direction of y-axis as the direction of the forward straight line motion of the robot as shown in Fig. 3(b). With the Robot Coordinate Frame as established above, the relationship between the uncertainties in the actual position/orientation of the robot vis-a-vis the commanded motions (either straight-line or turning) is determined using the same experimental procedures that were presented previously in [42].

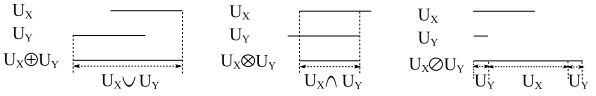
Fig. 2 shows how each type of uncertainty flows or propagates in our sensor fusion model. A more detailed depiction of how the uncertainties flow in Fig. 2 will be presented later in this section after we have told the reader about the operators that are used for manipulating uncertainty variables. These operators, presented visually in Fig. 5, can be used for combining the uncertainties as they propagate upwards in the flow diagram of Fig. 2. Each operator is a binary operator; its operands will be denoted X and Y. The meaning to be accorded to the operands for all three operators shown in Fig. 5 is as follows:

- X will be the data object whose uncertainty we want to keep track of during map building. For example, X could represent a line feature extracted from the range data.
- Y will represent the various kinds of "events", "processes", or other sensory observations that can increase or decrease the uncertainty associated with X. For example, as the robot moves, the uncertainty associated with a line feature extracted from the range sensor may increase in the local coordinate frame of the robot on account of the slippage in the wheels.

Here is a brief description of what is achieved by each of the three operators depicted in Fig. 5:

- $U_X \oplus U_Y$: This operator applies when the phenomenological considerations call for including all the uncertainties associated with X and Y in the updated uncertainty associated with X. We can say that the new uncertainty of X will be the union of the two prior uncertainties associated with X and Y, as shown in Fig. 5(a).
- $U_X \otimes U_Y$: This operator applies when the phenomenon represented by Y is able to decrease the uncertainty associated with X. If the underlying physical processes can guarantee that the new uncertainty associated with X will be limited to what corresponds to Y, we can say that the new uncertainty of X will be the intersection of the two prior uncertainties, as shown in Fig. 5(b).
- $U_X \oslash U_Y$: This operator applies if Y will increase the uncertainty interval associated with X at both ends of the interval. How X's uncertainty is increased is depicted in Fig. 5(c). We expand the uncertainty interval U_X in both directions by U_Y.

All the uncertainty update procedures are carried out in the *largest common subspace*, which is the common subspace of highest dimensionality that can be formed from the spaces associated with the two operands. For example, when the spaces associated with the operands X and Y are (x, y, t) and (y, z, t), the largest common subspace is (y, t).



(a) $\oplus$ operator. (b) $\otimes$ operator. (c) $\oslash$ operator.

Fig. 5. A visual depiction of the three operators used for combining interval based uncertainties.

Going back to Fig. 2, there are two entry points for the uncertainties: the uncertainties associated with the features extracted from the range data and the uncertainties associated with the features extracted from the stereo vision data. With regard to the range data, we associate uncertainties with the straight lines that are fit to the data. (Recall that the range data is collected in a horizontal plane parallel to the floor.) Ideally, these straight lines should correspond to the geometrically straight features in a hallway, but in practice the data collected is noisy and the least-mean-squares fit to the data an approximation to the true location and orientation of the actual feature. If for a given position of the robot, $\sigma_{F_0^0}$ is the standard deviation of the shortest distances from the range points to the fitted line represented symbolically by F_0^0, the uncertainty interval $U_{F_0^0}$ associated with range data collected at that position of the robot is set as this standard deviation. As mentioned, the other entry-point uncertainty relates to the uncertainty associated with the features extracted from the stereo vision, $U_{F_1^0}$. Recall that F_1^0 is a union of the stereo-extracted point features $F_{1,1}^0$, the stereo-extracted vertical lines $F_{1,0}^0$, and the wall images $F_{1,2}^0$, and, therefore, $U_{F_1^0}$ is some function of $U_{F_{1,0}^0}$, $U_{F_{1,1}^0}$, and $U_{F_{1,2}^0}$. What helps here is that we can ignore the contributions $U_{F_{1,0}^0}$ and $U_{F_{1,2}^0}$ when we propagate the uncertainties to higher levels in the hierarchy since the uncertainties associated with the wall images are directly taken into account when we reconcile the range lines (the wall images are required to pass through the range lines) and since the uncertainty associated with the stereo-extracted vertical lines are used only for fixing the boundaries of the wall images in local maps.

Therefore, we can write:

$$\begin{aligned} U_{F_1^0} \approx U_{F_{1,1}^0} &= \hat{U}_{F_0^0} \otimes u_1^0\left(F_{1,1}^0 | \hat{U}_{F_0^0} \oplus U_{D_1^0}\right) \\ \hat{U}_{F_0^0} &= \left(U_{F_0^0} \oslash U_{C_{0,1}}\right) \end{aligned} \tag{5}$$

where u_1^0 is the operator whose purpose is to estimate the uncertainty interval associated with its argument. So $u_1^0(X)$ estimates the uncertainty interval associated with X and $u_1^0(X|U)$ the uncertainty interval associated with X given the uncertainty interval U that represents all inputs that go into the calculation of $F_{1,1}^0$. As shown, this conditioning uncertainty interval U must be a composition of the uncertainties associated with all the inputs that go into calculating $F_{1,1}^0$.

To provide a phenomenological underpinning for Eq. (5), since the calibration uncertainty $U_{C_{0,1}}$ only amplifies the "noise" in the positional information, it makes intuitive sense to expand the uncertainty $U_{F_0^0}$ by $U_{C_{0,1}}$ and to then combine the result with the uncertainty $U_{D_1^0}$ derived from the stereo image sensing procedure. Note that, a visual feature is accepted in a local map provided the intersection between the uncertainty from the range data and that from the vision data, after each is expanded by the Calibration Uncertainty, is non-null. This is illustrated in Fig. 6. In this figure, the stereo vision uncertainty associated with a 3D landmark is represented by a spherical ball whose radius is equal to $U_{D_1^0}$. The

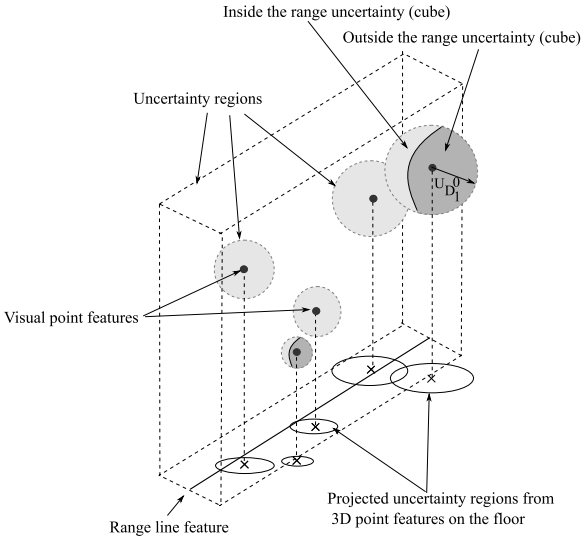


Fig. 6. A visual point feature extracted by the stereo algorithm is accepted when its uncertainty interval overlaps the uncertainty interval associated with the corresponding wall plane that is fit to a range line.

dark straight line at the bottom of the 3D rectangular enclosure is the range line that is fit to the individual points recorded by the range sensor. The width of the rectangular enclosure is twice the standard deviation associated with the least squares fit to the range points. The rectangular enclosure represents the 3D uncertainty bound associated with the range data; it is the uncertainty in the localization of the wall plane. Note that, the bottom of the 3D uncertainty enclosure coincides with the floor of the hallway. The points marked '×' in the figure are the projections of the 3D landmarks extracted by the stereo vision sensor. Such a 3D point is accepted as a feature for a local map if its uncertainty ball has non-null intersection with the rectangular 3D enclosure associated with the range line.

As the reader can tell from the right to left arrows in Fig. 2, after we have computed the uncertainty $U_{F_1^0}$ associated with the output of the vision sensors, it is used, through the function h^0, to update the previously computed uncertainty interval $U_{F_0^0}$ as shown in the following equation:

$$U_{F_0^0} = U_{F_0^0} \otimes U_{F_1^0}. \tag{6}$$

The purpose of $U_{F_1^0}$ is to restrict the locations of the visual features and also to refine $U_{F_0^0}$ if the uncertainty interval of $U_{F_1^0}$ is smaller than that of $U_{F_0^0}$, thus assuring the consistency between the different data elements produced by range and vision sensor. If the deviation of the visual point features from the fitted plane is smaller than that of the range point features with respect to the fitted range line, we can assume that the stereo reconstruction in relation to the corresponding wall plane is quite accurate, and we can then propagate that uncertainty to Level 1. Fig. 7 explains in greater detail the various steps that go into the calculation of $U_{F_1^0}$ and the update of $U_{F_0^0}$. Note that, Eq. (6) and the uncertainty processing steps explained by this figure constitute an implementation of the h^j function in Eq. (4). The range data uncertainty gives us the uncertainty interval A shown in the first part of the figure at the left. This interval is expanded by the uncertainty related to the calibration between the range sensor and the stereo vision sensor $U_{C_{0,1}}$. The interval B shown in (b) is the new expanded uncertainty interval. Note that this is the uncertainty interval that would be associated with the wall plane drawn through the range line. We also construct a vertical plane from the stereo point features. The uncertainty associated with this plane can either be as shown in (c) or as shown in (d). If the uncertainty interval associated with the plane drawn through the stereo landmarks is as depicted by C_1, as shown in (c), then the overall uncertainty associated with the wall plane shrinks to correspond to C_1. On the other hand, if the uncertainty interval associated with the plane drawn through the stereo landmarks is as depicted by C_2 as shown in (d) – note that C_2 is larger than B – then the uncertainty interval associated with the range line stays at B.

We will now discuss how robot motions influence the various uncertainty intervals. Fig. 8(a) shows feature location uncertainty $U_{F_0^0}$ before and after a straight line motion. The expansion of the uncertainty intervals at the end of the motion is owing to the Robot Motion Uncertainty. If the error vector from the straight line motion is $\mathbf{e}_s$ for a unit distance, the motion error vector for an arbitrary distance $\mathbf{d}$ would be $\mathbf{d} \cdot \mathbf{e}_s$. The vector $\mathbf{d} \cdot \mathbf{e}_s$ is decomposed into two orthogonal components ($\mathbf{a}$ and $\mathbf{b}$ in Fig. 8(a)) with respect to the slope of the corresponding range line feature. One of those two components is parallel to the range line and the other is perpendicular. These two component vectors make the bound of the uncertainty increase as shown in Fig. 8(a).

Fig. 8(b) of the figure shows the same for the robot's turning motion. If the error vector from the turning motion is $\mathbf{e}_t$ for a unit angle, then the error vector for an arbitrary turning angle $\mathbf{t}$ is given by $\mathbf{t} \cdot \mathbf{e}_t$. We decompose $\mathbf{t} \cdot \mathbf{e}_t$ into $\mathbf{a}$ and $\mathbf{b}$ as shown in Fig. 8(b). The remaining procedure to increase the bound of the uncertainty is the same as for the case of straight line motion.

The uncertainty transformations depicted in Fig. 8 correspond to the transition from F_0^0 to D_0^1 in the left upward propagation of uncertainty intervals in Fig. 2.

Recall that each local map is situated in an intermediate map with respect to some coordinate frame, the transformation that locates a specific local map in the intermediate map being given by g^0. Usually, the coordinate frame for the intermediate map corresponds to the coordinate frame used for the very first local map. In order to look for a match between a feature in a local map and a feature in the intermediate map to which the local map belongs, the applicable uncertainty intervals are combined for the local map feature in order to construct uncertainty bounds on the search. Additionally, when such a match is discovered, the local map feature is "merged" with the intermediate map feature and the uncertainty of the intermediate map feature updated with the uncertainty of the local map feature. In terms of the notation shown in Fig. 2, $U_{F_0^0}$ is the uncertainty interval associated with a range line feature in a local map. $U_{F_0^1}$ is the uncertainty interval associated with the corresponding range line in the intermediate map (base range line). When we merge the new range line in the local map with the range line in the intermediate map, we update the uncertainty of the latter through an approach that we call the "projected interval-based uncertainty update". This process of updating the uncertainty interval at the IMB level is depicted in Fig. 9(a) and in Eq. (7).

$$U_{F_0^1} = U_{F_0^1} \otimes proj_{F_0^1}(U_{F_0^0}) \tag{7}$$

where $proj_{F_0^1}(U_{F_0^0})$ stands for the projection of the uncertainty interval $U_{F_0^0}$ onto a unit vector along the perpendicular to the range line F_0^1.

By the same token, the update of the uncertainty interval associated with a visual point feature in the IMB is given by Eq. (8). Fig. 9(b) illustrates this equation. Note that Eq. (8) is similar to Eq. (7) except that it is missing the "*proj*" operator. This is because we are now dealing with a feature without any orientational information.

$$U_{F_1^1} = U_{F_1^1} \otimes U_{F_1^0}. \tag{8}$$

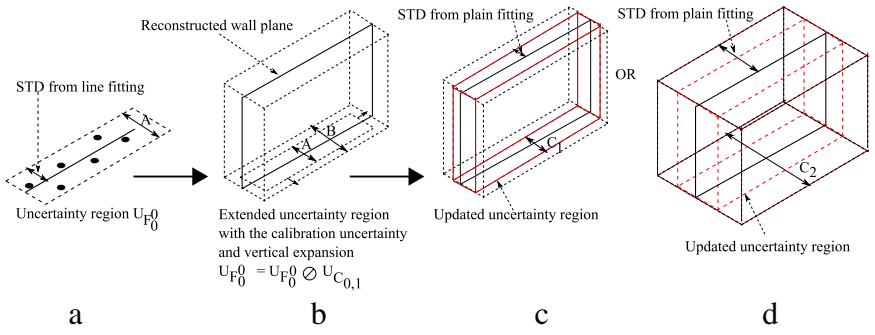
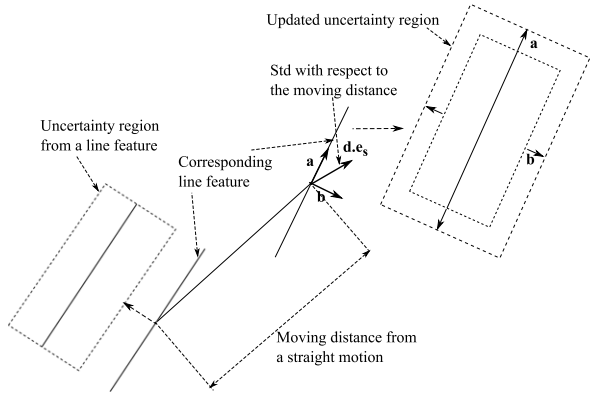
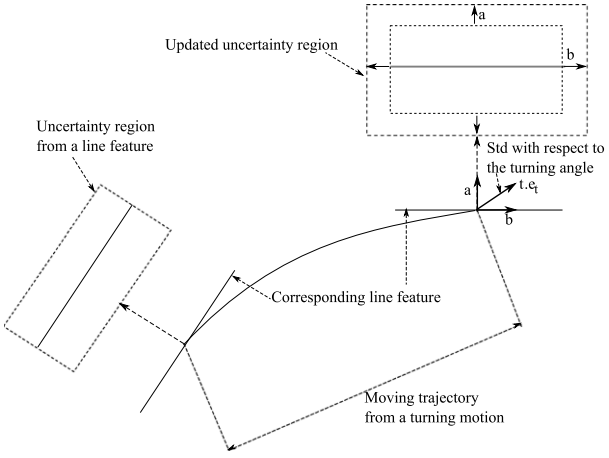


Fig. 7. Uncertainty propagation in the Local Map Building level.



(a) Uncertainty bound change in relation to the robot's straight line motion.



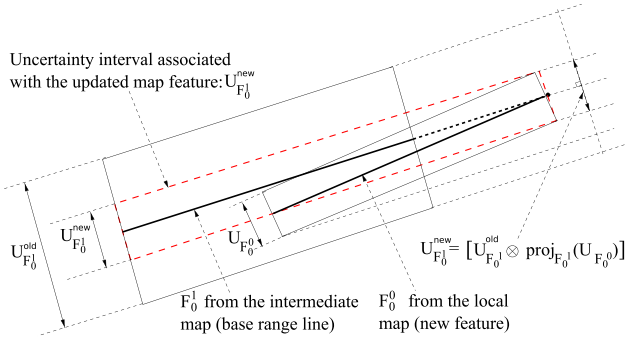
(b) Uncertainty bound change in relation to the robot's turning motion.

Fig. 8. Updated uncertainties in relation to the robot's straight and turning motion.

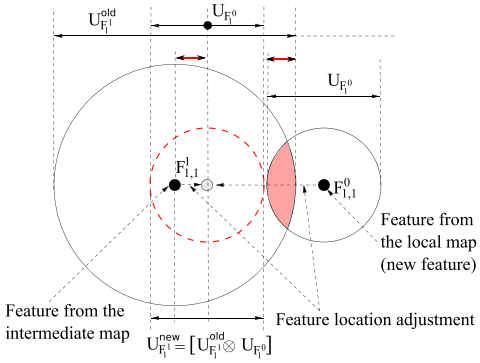
This procedure of merging local map features and the intermediate map features along with the uncertainty update process also occurs in the GMB process with the intermediate map features and the global map features.

3. Local Map Building

For Local Map Building, the very first step that takes place is the collection of range data D_0^0 with a laser scanner. The collected range data consist of points that usually fall on lines along the walls in the corridor. The next step is to fit straight lines to these range points. Thus, the step labeled "Line feature extraction" in box 2 of Fig. 10 is for fitting straight lines to the range values based on the function f_0^0 mentioned earlier.



(a) Uncertainty update for range features in IMB.



(b) Uncertainty update for a visual point feature in IMB.

Fig. 9. Uncertainty update for the IMB level.

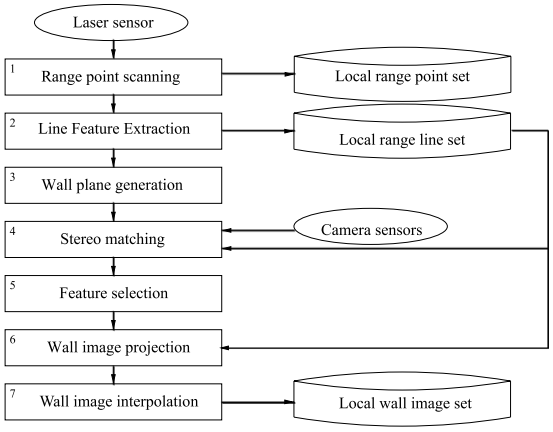


Fig. 10. Overview of the Local Map Building level.

Then, by the step labeled "Wall plane generation" in box 3 of Fig. 10, a vertical wall plane containing the straight line segment extracted from the range data is instantiated. This is done with the assumptions that the walls in the environment are assumed vertical to the floor and the floor constitutes a horizontal plane.[4] One of the stereo images is then chosen, based on the current position and direction of the robot and the field of view of each of the left and right cameras, to be projected onto the wall plane. As these projections are "accumulated" on the wall plane, we end up with a wall image. The computational steps "Stereo matching" and "Feature selection" in boxes 4 and 5 of Fig. 10 are meant to accomplish what the names imply. SURF [43] descriptors are extracted from the stereo images and matched together to compute the Fundamental Matrix F, which is then used to find the transformations required to rectify both of the stereo images. 3D point and 3D vertical line features are subsequently extracted from the rectified images through triangulation and through the use of the extended Hough transform (see Fig. 4(f)). It is noteworthy to mention that in order to affix the stereographically reconstructed images to the wall planes correctly (box 6), we need accurate calibration between the laser range sensor and the stereo cameras on the robot (see Fig. 4(d) and (e)). Thus, the success of this step is highly dependent on the uncertainty $U_{C_{0,1}}$ mentioned earlier. In box 7, an interpolation procedure is used to uniformly sample the projection of the wall images onto the wall planes. The wall planes are assumed to be of a predefined height. In the experiments reported in Sections 6 and 7, we assume the height of the ceiling to be a constant (240 cm), although small variations from this height have never caused any problems in our experiments. In the rest of this section, we will explain in greater detail some of the steps listed in Fig. 10.

3.1. The data fusion from the range data to the vision data

Fig. 11 shows the overview of the stereo reconstruction algorithm using the range data. Note that, this overview is the detailed version of Step 4 though Step 6 in Fig. 10. First of all, we transform the pair of the original stereo images into the rectified images with the stereo calibration information obtained by applying Zhang's algorithm [40] (Step 1). For the next step we first determine whether or not the robot has detected hallway corner points and/or jump discontinuities in the range data collected at the position of the robot. If such points are detected, that implies that the camera images correspond to more than one wall plane in the hallway. In this case, vertical lines corresponding to such points in the range data are projected into the rectified camera images (see Fig. 4(g)). This constitutes a vertical partitioning of the rectified images as represented by Step 2 in Fig. 11. Obviously, if corner points and/or jump discontinuities are not detected, the full rectified images constitute the vertical partitions.

Note that when the vertical lines corresponding to the detected jump discontinuities are projected into the rectified camera images, they may not correspond exactly to the visible vertical lines in the images. There can be a discrepancy between the vertical line projected from the range data and the visible vertical line at the corner point (see Fig. 4(h)). This discrepancy is caused mostly by the differences in the sampling resolutions of the camera images, on the one hand, and the range point data, on the other. This error is minimized by adjusting the relative locations and orientations between the range features and the visual features.

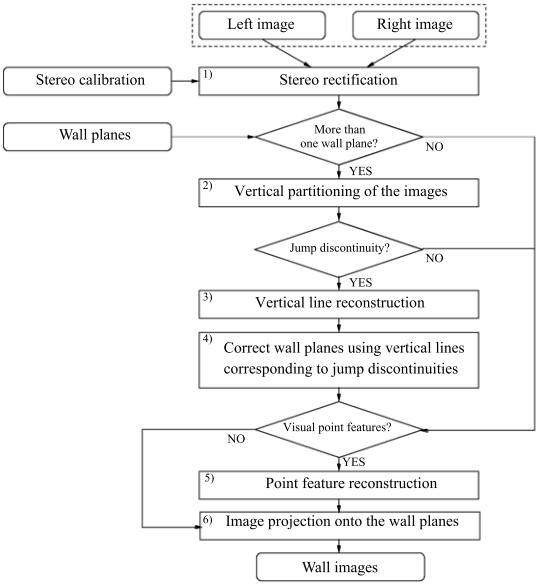


Fig. 11. Overview of stereo reconstruction with the help of range data.

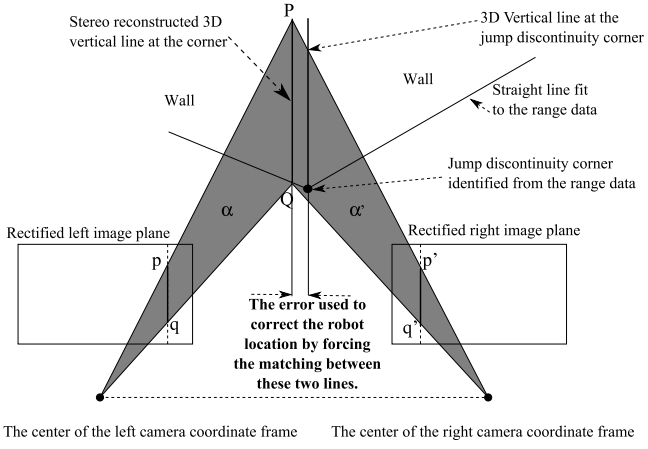


Fig. 12. Since the resolution of the stereo vision sensors far exceeds that of the range sensor as shown in the next subsection, hallway vertical edges are localized with greater precision with the vision sensors. This higher-precision stereo localization of the edges can then be used to adjust the location parameters of the features extracted from the range sensors and the location of the robot.

The adjustments are carried out by matching the vertical lines at the jump discontinuities and the stereo-reconstructed vertical lines (these correspond to Steps 3 and 4 in Fig. 11). As we explain in the next subsection, the stereo vision localizes the wall edges with far greater accuracy than the range sensors. So this adjustment constitutes a correction for not only the information supplied by the range sensor, but also for the localization parameters of the robot. Fig. 12 provides a visual explanation of these adjustments. Subsequently, in Step 5, point features are matched between the stereo images to determine whether or not they belong to the wall plane partition under consideration. Finally, by projecting each image partition onto the corresponding wall plane (Step 6), the wall images are attached to the wall planes (see Fig. 4(m)).

3.2. Getting a fix on wall boundaries in the vicinity of jump discontinuities

Since the map building process requires us to project camera images onto the walls, it is important for our system to determine the horizontal extent of each wall plane. When two walls meet

[4] Our mapping approach may not produce acceptable results when the space being mapped has few planar walls, if the walls are too curved, or if there are objects/people moving around during map construction. We have discussed these "failure modes" of our work in Section 6.1.

at a corner and both wall planes are visible to the range sensor, fixing the location of the corner is easy because it is trivially the intersection of the range lines corresponding to the wall planes.

However, when only one of the two wall planes which are visible to the range sensor meets the corner, getting a fix on the corner point is more challenging. Consider the blocked area depicted in Fig. 13(a). Such corner points will give rise to jump discontinuities in the range line for that section of the hallway.

The problem of localizing a corner point at a jump discontinuity is made difficult by the fact that, regardless of the angular resolution of the range sensor, when walls are oblique to the line of sight of the range sensor or walls are far from the robot, the actual sampling points on the walls may not be spaced sufficiently close for accurate localization of the corner. Our range sensor has an angular resolution of 0.5°. When the robot is, say, 8 m from a wall that is at, say, 90° to the line of sight of the range sensor, the 0.5° resolution translates into the successive range samples on the wall being 6.98 cm apart. That is fairly low resolution if the goal is to localize a corner accurately.

This problem with hallway corner localization can be mitigated by using the higher resolution provided by the vision sensors—provided, of course, the corner is visible as a vertical line feature in the camera images and provided that we can solve the stereo correspondence problem for the line feature. The cameras mounted on the robot have a field of view of roughly 64° and we typically have 640 pixels in each scan line. This translates into an angular horizontal resolution of roughly 0.1°, which means that the successive pixels sample the wall points 1.396 cm apart. By the same analysis, the vertical resolution on the wall is found to be 1.5 cm. So, under the very best circumstances, the stereo process may localize a corner in the hallway with 1.396 cm horizontal resolution. (Though this example ignores the intrinsic parameters of the camera and the relative location of the camera with respect to the range sensor, it still provides an approximate comparison for the different sensory resolutions in our system.)

In the rest of this section, we will describe how we use stereo vision to carry out a precise localization of the hallway corner that gives rise to a jump discontinuity in the range data (see Fig. 13(b)). This localization consists of the following four steps:

- We first backproject the portion of the view angle that corresponds to the jump discontinuity of the range data into each of the stereo images.
- We find the vertical lines in the backprojected regions in both the image planes using the extended Hough transform [44]. Then by using a similarity measure, we find the corresponding pair of vertical lines in the two images. Triangulation on the corresponding pair of lines yields a set of 3D vertical lines in the hallway, $F^0_{1,0}$. Should the set include more than one vertical line, we use the following step to select the line that corresponds to the jump discontinuity.
- We extend the wall planes neighboring the jump discontinuity until both the extended wall planes can be seen by the stereo vision. This step determines the possible regions on the extended wall planes for the placement of the 3D lines reconstructed from stereo vision.
- In order to obtain the updated boundary of the left wall plane near the jump discontinuity, we choose the rightmost 3D vertical line in the uncertainty bound of the left wall plane with respect to the Camera Coordinate Frame as the vertical line corresponding to the jump discontinuity in the range data. (The uncertainty bound associated with a wall plane was defined previously in Section 2.) On the other hand, In order to obtain the updated boundary of the right wall plane near the jump discontinuity, we choose the leftmost 3D vertical line in the uncertainty bound of the right wall plane. This is illustrated in Fig. 13.

Fig. 14 shows two typical experimental results that illustrate the benefits of the wall plane boundary correction described above.

4. Intermediate Map Building

Fig. 15 presents an overview of the IMB process. First, the Intermediate Coordinate Frame is initialized by setting it equal to the first Local Map Coordinate frame (Step 1). Subsequently, LMB for the current rotational orientation of the robot can proceed as explained in the previous section (Step 2). For additional local maps taken at different rotational orientations of the robot (but with the robot at roughly the same translational position), the robot orientation and translational information obtained from the wheel encoder is used to locate the newly acquired local map features in the Intermediate Coordinate Frame (Step 3). However at this point, due to the uncertainties associated with robot motions, the local map features and the corresponding intermediate map features will not be exactly aligned. This entails correcting the map feature information and the robot location information by adjusting the orientations and the positions of the corresponding features.

As a first step in establishing feature correspondences between the local map features and the intermediate map features, the system seeks out the set of range line candidates for matching between two maps—based on the slopes, relative locations of the range line features and the intersection of their uncertainty regions (Step 4). After such range-data line correspondences have been established, the local map range-data lines are added to the intermediate map (Step 5). Using these range line correspondences, we adjust the orientation of the robot and the map features with respect to the Intermediate Coordinate Frame (Step 6).

With the relationship thus established between the corresponding range line features at each robot position, we can apply one of two different types of adjustments to the locations of the robot and the map features. If the corresponding range lines are more or less parallel to one another, such adjustments can only be made in a direction that is perpendicular to the range lines. However, when intersecting range lines are available, the adjustments can be made to both the coordinates of the robot position and the features in a constructed map (Step 8).

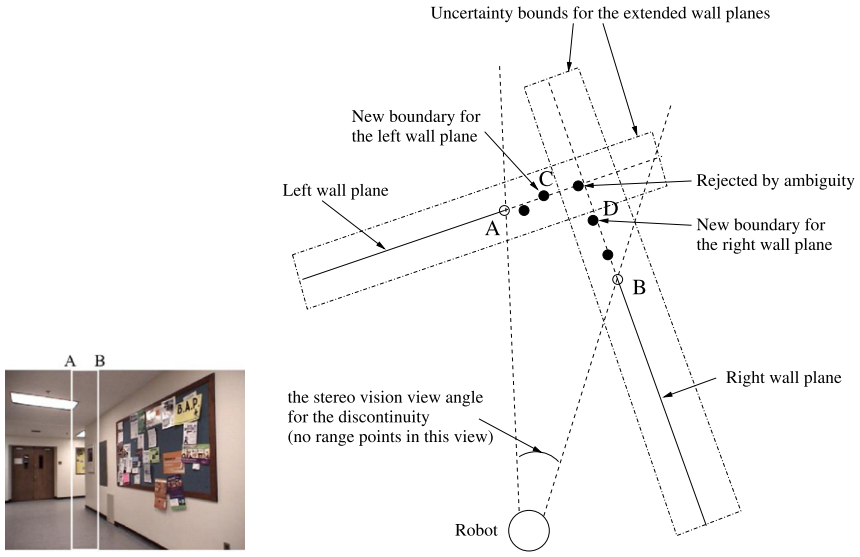
This brings us to the IMB steps that involve both stereo vision and range data. By matching corresponding point features from the local wall images and the intermediate wall images (Step 9), we estimate the average of the location differences between the corresponding point features from the two maps. When adjusting the locations of the features with this difference, we can apply additional corrections to the location of the robot (the translation components) and to the locations of the features with respect to the Intermediate Coordinate Frame (Step 10).

At the end of each iteration of the IMB process that involves merging of the local map features with the intermediate map features (Step 11), the resulting updated feature locations are stored in the intermediate map for GMB. Finally, when there is no longer a need to build any additional local maps, the IMB process also comes to a stop. At that point, the robot engages in the GMB process (Step 13).

Note that the explanations in this section for the map feature relocation, the map feature merging, the robot location correction, and the uncertainty bound update rule are also applied to the GMB process.

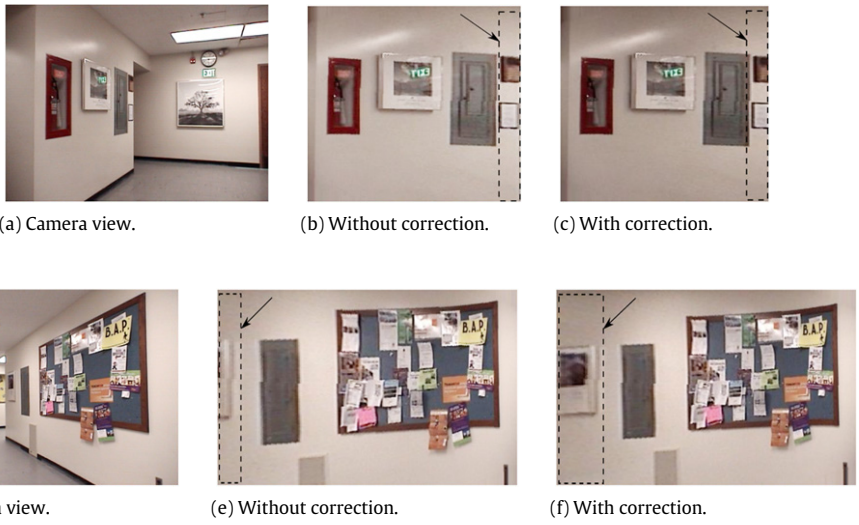
4.1. Finding feature correspondences between a Local Map and an Intermediate Map

As the robot rotates at roughly the same translational position in order to collect different local maps, the robot makes sure that there is an angular overlap between successive rotational orientations of the robot. This step, which helps the robot situate a new Local Coordinate Frame within an Intermediate Coordinate



(a) Example image. (b) Jump discontinuity and vertical lines.

Fig. 13. Finding the best vertical lines for the jump discontinuity.



(a) Camera view. (b) Without correction. (c) With correction.

(d) Camera view. (e) Without correction. (f) With correction.

Fig. 14. Examples of the corrected wall images using vertical line matching.

Frame, involves finding features matches between a new local map and the intermediate map accumulated so far.

Let us assume that we have a new observation Z of a feature from a new local map and we have n possible candidates, X_t, $t = 0 \cdots n-1$, in the current intermediate map that could potentially be matched with Z within the corresponding uncertainty bounds (U_t). Then we can find the best correspondence X^* for the observation Z through the maximization of the probability that the global feature that corresponds to Z is X_t. This is expressed by [45]:

$$X^* = \arg\max_{X_t} \Pr(X_t \mid Z, U_t). \tag{9}$$

With Bayes' rule, this equation can be transformed into the following equivalent form:

$$X^* = \arg\max_{X_t} \eta \Pr(Z \mid X_t, U_t) \Pr(X_t \mid U_t) \tag{10}$$

where η is the normalization factor. As long as the feature X_t is in the uncertainty bound U_t, we can set $\Pr(X_t \mid U_t)$ to the same value for all t. This implies that the above is equivalent to:

$$X^* = \arg\max_{X_t} \Pr(Z \mid X_t, U_t). \tag{11}$$

In order to obtain $\Pr(Z \mid X_t, U_t)$, we suggest a cosine similarity measure $\mathrm{Sim}_\nu(Z \mid X_t, U_t)$ where ν is the matching criterion within the uncertainty U_t. The probability $\Pr(Z \mid X_t, U_t)$ can be expressed as

$$\Pr(Z \mid X_t, U_t) = \frac{\mathrm{Sim}_\nu(Z \mid X_t, U_t)}{\sum_{t=1}^{n} \mathrm{Sim}_\nu(Z \mid X_t, U_t)} \tag{12}$$

therefore our matching criteria is equivalent to,

$$X^* = \arg\max_{X_t} \mathrm{Sim}_\nu(Z \mid X_t, U_t). \tag{13}$$

4.2. Relocation of the local map features and the robot in the Intermediate Coordinate Frame

With the approximate knowledge of the robot position/orientation as provided by the wheel encoders, the local map features can be positioned (albeit only approximately initially) in an intermediate map. The location parameters associated with these features

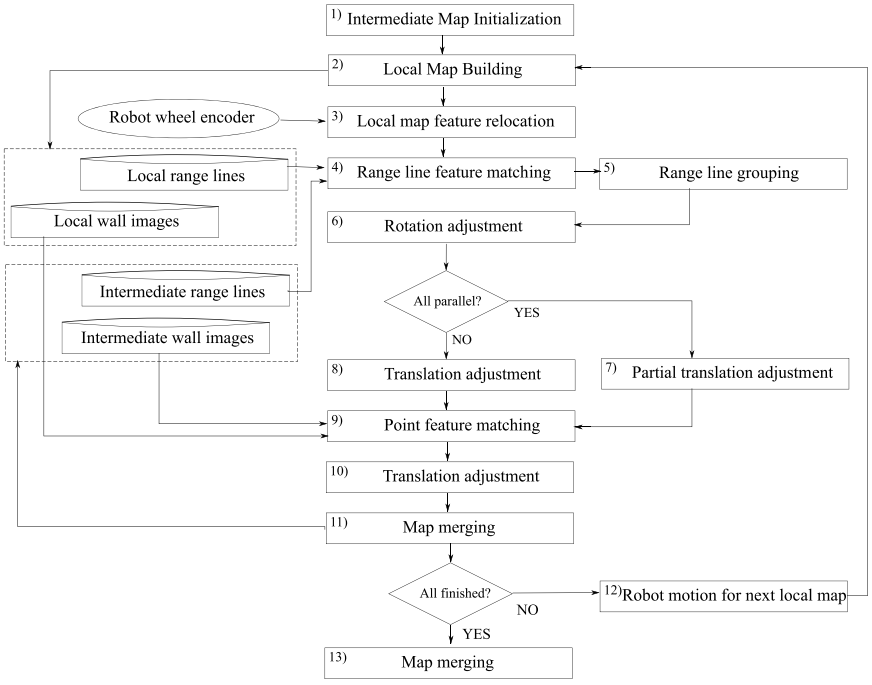


Fig. 15. Overview of Intermediate Map Building.

must subsequently be corrected by enforcing the geometrical constraints available at the intermediate map level.

Given the matched sets of corresponding features from the local map and the intermediate map, the current estimate of the orientation of the robot is refined, and from the corner points, the x, y location of the robot is made more precise.

However, if the robot finds itself in a long hallway or encounters jump discontinuities, it is possible that there are insufficient number of corner points to estimate the translational error for the new map features. By extending the range lines that are not parallel or not close to parallel, we obtain the intersections for each pair of lines, which we call "virtual corner points". The vector from the intersection of the local map features to those of the intermediate map features should be the translation error for the map features and the robot locations. By averaging those vectors, we can adjust the location of the new map features and the location of the robot. We should also mention that the previous correction with regard to the robot location constitutes only one part of the correction procedure to obtain higher localization accuracy of the robot. The other part of the correction procedure was mentioned earlier in Section 3.1 in which the vertical lines at jump discontinuities in the range data are matched with the stereo-reconstructed 3D vertical lines. These steps were previously illustrated in Fig. 12.

4.3. Wall image merging

For each merged intermediate range line feature, we have the corresponding wall image. In order to find the face (normal direction) of the merged wall image, we choose one of the wall images as a "base wall image" and then merge all other wall images into the wall plane of the base wall image. The base wall image is chosen on the basis of its yielding the highest average number of projected pixels on the corresponding wall plane on a per unit area basis. Obviously, the farther or more oblique a wall plane is in relation to the camera system, the smaller the number of interesting pixels on a unit-area basis in the wall image.

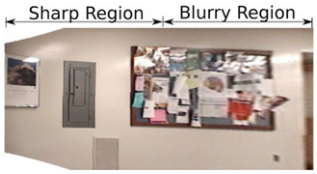
With regard to the merging of images with the base wall image, the merging process uses acquired features extracted using SURF. The main challenge here is dealing with the fact that, in general, the images will be of uneven quality on account of image-to-image variations in illumination, variations in viewpoint with respect to the orientation of the wall, etc. Therefore, care must be exercised in the selection of pixels used for the merging process. From the pixels that exist in the overlapping regions of the images, we choose those that are in close proximity to other pixels. Close proximity usually implies higher resolution and, thus, higher local image quality.

Fig. 16 shows an example of the result of wall image merging when each of the two images has its own sharp and blurry regions in the overlapping region. Fig. 17 shows an example of the result of wall image merging between several local maps into an intermediate map. It is important to mention that the visual quality of the resulting map as in Fig. 17 is only guaranteed provided that there are sufficient number of feature correspondences required to find the different transformations needed for stitching of the maps. Fig. 18 shows an example when this condition is not satisfied. In such a case, we rely on the placement of the wall images corresponding lines in the intermediate/global map to do the merging.

For the final step in the IMB procedure, we attach the global wall image onto the corresponding global wall plane while we align the left-upper pixel of the base wall image on the original location in the merged wall plane.

5. Global Map Building

The global map is a merged collection of the intermediate maps that are built at different locations in the environment. The GMB procedure consists of the following two steps: (1) merging of the latest intermediate map into the global map constructed so far while maintaining the consistency of the corresponding map features. This step is the same as the merging of the latest local map in an intermediate map, the only difference being that whereas the different intermediate maps for global map construction involve different translational positions of the robot, the different local maps that go into an intermediate map involve different rotational orientations of the robot. (2) determination of the next map building location with respect to the open passages that are available to the robot. For selecting the next map building location, the robot uses the depth-first heuristic for the simply reason it seems



(a) Wall image from the left view.

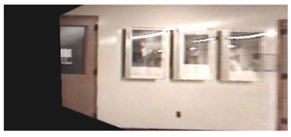
(b) Wall image from the right view.

(c) Merged wall image.

Fig. 16. Wall image merging using SURF matching and our own weights.

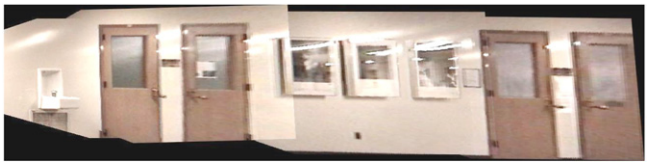
(a) Local map 1.

(b) Local map 2.



(c) Local map 3.

(d) Local map 4.



(e) Intermediate.

Fig. 17. An example of wall image merging between several local maps.

more natural to the exploration of the hallways typically encountered in institutional buildings.

6. Experimental results on the visual quality of the constructed maps

It should be obvious that the visual quality of the maps constructed by the SLAM framework presented here would depend critically on how the uncertainties in the different sensor are managed and how the more reliable sensors are given priority in establishing bounds on the uncertainties associated with the features extracted from the less reliable sensors. One can also expect the visual quality to depend on the various environment conditions such as the illumination and contrast variations and the availability of sufficient salient features required to find the different transformations to do the stitching process.

Some of the results we presented earlier bear testimony to the visual quality of the maps as several local maps are merged together to form intermediate maps (see, for example, Fig. 17 that was presented earlier in Section 4.3). As stated earlier, the high visual quality of those results depends on whether or not a sufficient number of feature correspondences can be established between the different local maps. *But that is to be expected.* Even human beings can get disoriented in long featureless corridors.[5]

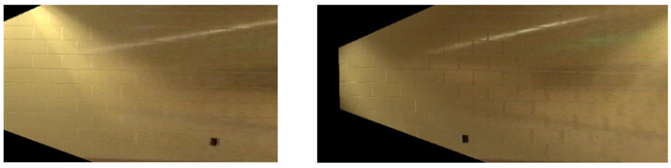
To give the reader a sense of the visual quality of the maps at a more global level, we present Fig. 19(a) as a 3D map constructed by our robot for the hallways immediately outside our laboratory at Purdue. This map was reconstructed using a *PowerBot* robot from *ActivMedia* shown in Fig. 3(a). The robot comes equipped with one SICK LMS-200 2D laser range sensor and a PC type computer consisting of a Pentium 4 processor with 1 GB RAM running at 2.4 GHz. Additionally, we installed two Sony EVI-D100 cameras on a customized superstructure mounted on the robot.[6]

To give the reader a sense of the size of the space mapped in Fig. 19(a), the longer straight hallway in the map is about 16 m long. Fig. 19(b) and (d) presents a sampling of the intermediate and the local maps that went into the reconstruction shown in Fig. 19(a).

A total of 54 local maps and 18 intermediate maps were used in the construction of the global map shown in Fig. 19(a). Each intermediate map is constructed from three local maps recorded at three different orientations, roughly 45° apart, of the mobile robot. We determined experimentally that this small number of local maps is sufficient to construct intermediate maps (that are eventually integrated into a global map). The translational distance between two successive positions for constructing the intermediate maps is roughly 2.5 m. The actual corrected trajectory and positions of the robot are shown in Fig. 20. The red circles represent the translational positions of the robot, with an intermediate map being constructed at each such position, and the radially outward

[5] When they do not, it because of higher level cognitive cues, such as the distance traveled along a corridor in relation to its length in their mental maps of the corridor.

[6] Please note that we do not use any of the robot's vendor-supplied software packages, such as Mapper3, in our map building routines. That is because these packages are not suitable for our real time implementation and our interval-based uncertainty calculus.



(a) Local map. (b) Another local map.

Fig. 18. An example where we cannot do the merging of the wall images directly because of the lack of feature correspondences.

Table 1
Average processing time (APT) and average number of features (ANF) for the local and intermediate maps that went into the reconstruction of the global map shown in Fig. 19(a).

Feature type	LMB ($j = 0$)		IMB ($j = 1$)	
	ANF	APT (s)	ANF	APT (s)
Range data D_0^j	361	0.001	1083	0.003
Fitted range lines F_0^j	8	0.079	9	0.251
Visual 3D lines $F_{1,0}^j$	13	0.776	13	2.428
Visual 3D points $F_{1,1}^j$	278	3.466	278	9.398
Visual wall images $F_{1,2}^j$	8	0.313	9	0.911
Total APT		4.635		12.991

Table 2
Average uncertainty bounds of the features per map type.

Uncertainty type	LMB ($j = 0$) Value (cm)	IMB ($j = 1$) Value (cm)
$U_{F_0^j}$	21.72	28.1
$U_{F_1^j}$	20.4	22.7

lines at each position show the rotational orientations of the robot for recording the local maps that are integrated into an intermediate map.

Table 1 lists the average number of features (ANF) collected for the local and the intermediate maps and their associated average processing time (APT). For the global map, the final number of the 2D range lines or wall images was 39, the total number of the original point range measurements was 19 494, and the total processing time was 265.37 s. Note that the number of features and the time taken for constructing a local map, an intermediate map, and the global map depend, on the one hand, on the visual and layout complexity of the interior space that robot is mapping, and, on the other hand, on the size of the space. In Table 2 we list the final average uncertainty bounds for the features in the local and the intermediate maps. Note that, the uncertainty intervals associated with the range lines and the wall images are the same. The final average uncertainty interval associated with the wall images and the range lines in the global map was found to be about 40.6 cm.

6.1. Failure modes in our map building system

The robotic mapping framework we have presented is based on certain key assumptions that, if violated, can result in the degradation of the quality of the constructed maps. One of the key assumptions is that the space to be mapped is delineated by planar walls and that there are no moving objects in the environment within sensing range of the robot. Typical corridors constitute a primary example of the interior space that our mapping framework can handle. If the walls are curved or if the space contains pillars and such, our mapping framework will try to create a planar approximation to the curved vertical surfaces. Such approximations break down for obvious reasons when the vertical surfaces are strongly curving. Figs. 4(m) and 21(a), which show a round trash can be constructed as a part of the scene, demonstrate that our system can handle a good amount of curvature in the vertical surfaces. For an opposite case when our system performs poorly in the presence of non-planar surfaces, Fig. 21(b) shows the reconstruction of a scene with a human standing by a doorway. The fact that our system also cannot handle objects moving about within the sensing range of the robot during map construction is illustrated in Fig. 21(e).

7. Experimental evaluation of loop closure errors and the overall localization accuracy

The visual quality of the reconstructed structures as illustrated in the previous section is obviously important for establishing the validity of our hierarchical sensor fusion architecture. At the same time, it is equally important to give the reader a sense of the localization accuracy and the loop closure errors in our system. The localization accuracy of the robot for the reconstructed map shown in Fig. 19 is approximately (−0.12 m, 0.05 m, −0.6°) in terms of the final robot pose (x, y, ϕ). This is computed by comparing the final map with the ground truth obtained from the architectural blueprint of the mapped area.

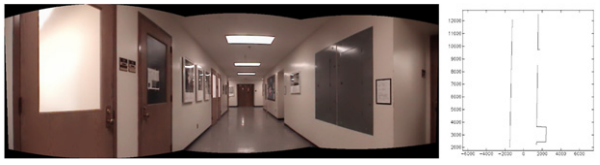
As we mentioned in the Introduction, over the years, the loop closure error has become an important metric of the overall accuracy of a SLAM system. The topic of loop closure for SLAM robots is a full-fledged research subject into itself, as it involves issues such as view invariance of landmarks, recognition of landmarks and local maps visited earlier, estimation of the positional discrepancy between the actual position of the robot and where it thinks it is after the robot has determined it is seeing a previously visited location, and, finally, techniques for closing the loop. The scope of our paper does not include those issues, our focus being merely to present a new approach to the hierarchical management of multisensor uncertainties to improve the quality of data fusion for the purpose of making maps of high visual quality. Nonetheless, it is important for us to demonstrate the loop closure errors associated with our SLAM procedures.

Since the magnitude of loop closure errors can be expected to be proportional to the length of loop traversals, we believe it is important to show how these errors vary as the hallway traversals become longer. Toward that end, we present in the rest of this section our loop-closure error results in three different ways: (1) loop-closure error for a small loop; (2) the same for a large loop; and, finally, (3) a table that illustrates such errors as a function of the length of the loop. For additional experimental support, we provide a comparison between our results and the results obtained with the GMapping system [32]. The GMapping system is a state-of-the-arts open source technique for SLAM. It is based on using the Rao–Blackwellized particle filter in which each particle carries an individual map of the environment. The source code for GMapping is available from the OpenSlam website.

Fig. 22 shows our typical loop closure errors in a relatively small hallway system presented schematically in Fig. 22(a). The total path length around the loop in this case is about 100 m and the loop closure error, shown in Fig. 22(b), is approximately (0.91 m, −0.62 m, 1.2°). Fig. 22(c) and (d) shows the results obtained with GMapping for the same environment, the former with 30 particles and the latter with 60 particles. The data used in all these map

(a) Global map.

(b) One of the intermediate maps.



(c) The images (stitching view of three local maps images) and the line map that used to construct the IMB shown in Fig. 19(b).

(d) Some local maps.

Fig. 19. An example of a global map constructed by the SLAM system presented in this paper. Also shown is a sampling of the intermediate and the local maps that went into the reconstruction.

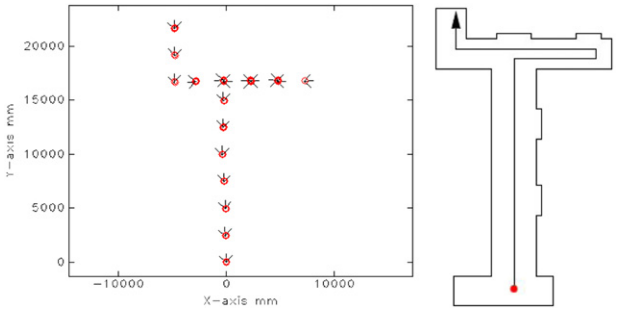
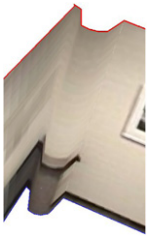
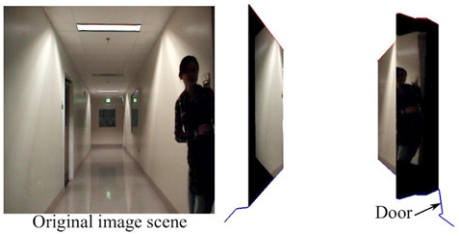


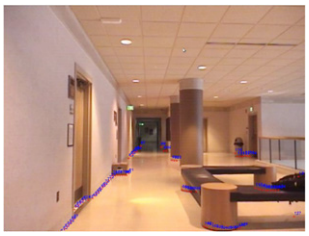
Fig. 20. Data collection points for the local maps during a run of the experiment.



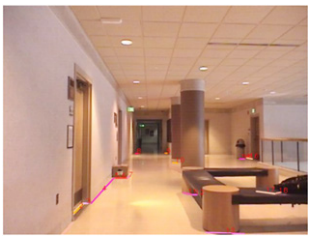
(a) Zoomed view of the trash can in Fig. 4(m).



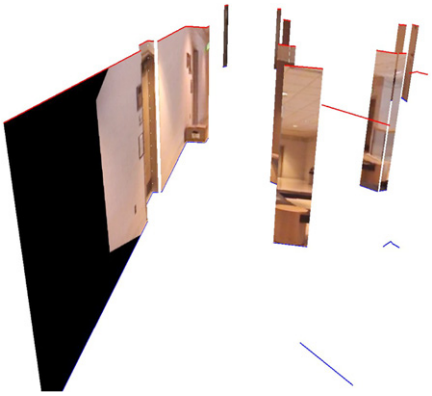
(b) An example of a person reconstructed as a part of the recovered map.



(c) Detected range points.



(d) Extracted range lines.

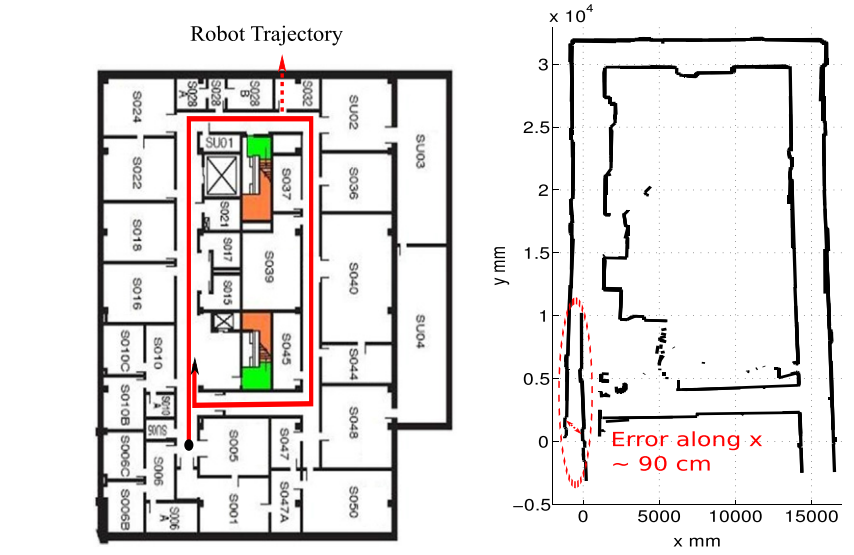


(e) Reconstructed map.

Fig. 21. The affects of the presence of curved walls and moving or movable objects in our map building system.

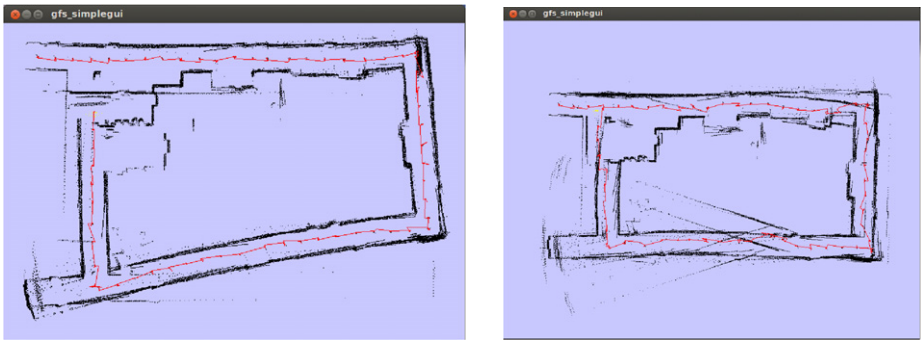
constructions consist of sets of 3 local maps taken 45° apart at each translational position of the robot and with an interval of roughly 2.5 m between successive locations at which the local maps are recorded. As the reader will recall, in our framework, the three local maps at each translational position are combined into an intermediate map. Except for the number of particles used, the GMapping code was run with default settings. The number of particles used in Fig. 22(d), 60, produced the best result with GMapping. Comparing our results in Fig. 22(b) with the best result produced by GMapping as shown in Fig. 22(d), it would be fair to claim that our results are somewhat "cleaner" than those obtained with GMapping.

We will now show the comparative results for the case when the size of the space is much larger than was the case in the previous study. Fig. 23(b) shows the 2D line map reconstruction with our framework for the interior space that is presented schematically in Fig. 23(a). The length of the loop in this case is



(a) MSEE-BLDG subbasement blueprint.

(b) 2D line map and loop closure error.



(c) GMapping result using the default settings (30 particles were used).

(d) GMapping result using the default settings except that 60 particles were used.

Fig. 22. Loop closure error in a small-sized indoor environment: MSEE-building subbasement.

roughly 250 m. The loop closure error with our framework in this case is about (−1.8 m, 0.55 m, −2.5°). Fig. 23(c) and (d) shows the GMapping results for the same environment with the same data that was collected for the result shown in Fig. 23(b). It is obvious that our results are superior to those obtained with GMapping. We noticed that, the problem that GMapping runs into is that the large translational intervals (2.5 m) that we use between successive data collection points causes GMapping to base its particle filter on a proposal distribution that uses odometry motions. However, such can be expected to be erroneous on account of the differential slippage between the wheels. It is possible to get better quality map constructions with GMapping but at the cost of vastly more data. Fig. 24 shows higher quality GMapping reconstruction that is based on 9 local maps 45° apart, with the translational positions being only 50 cm apart (as opposed to our framework needing only 3 local maps at each translational position that are 2.5 m apart). Even though the GMapping result in Fig. 24 is good, note that it still suffers from a serious defect: the overall orientation of the reconstructed map is about 45° off from its true orientation. Besides using vastly more data, the result shown in Fig. 24 also uses a large number of particles—90 to be exact.

To give the reader an idea of how the loop closure error varies with the distance traveled in the hallways for our framework, Table 3 shows the loop closure error as a function of the length of the loop for four different experiments.

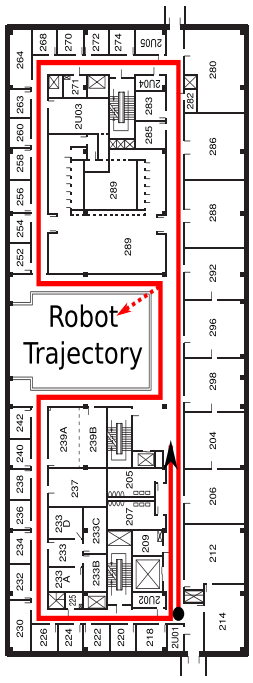
Table 3
Loop-closure error as a function of the length of the hallways traveled.

Hallway length (m)	Loop closure error		
	x (m)	y (m)	ϕ (°)
60	−0.52	0.48	−0.9
100	0.91	−0.62	1.2
190	−1.4	0.53	−2.1
250	−1.8	0.55	−2.5

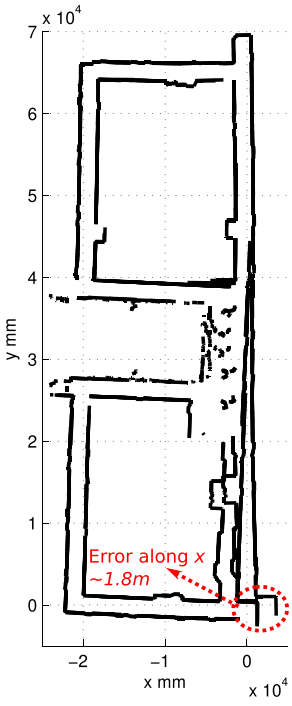
8. Conclusion and discussion

In this paper, we have proposed a new hierarchical approach to fusing data from disparate sensors, vision and range in our case, for constructing 3D maps of high visual quality and for achieving high-precision robot localization in the constructed maps. The core part of our contribution lies in how the uncertainties are managed with interval based logic. This logic allows our system to fuse information extracted from the sensors at different levels of abstraction. Our uncertainly management logic also allows our system to automatically place greater trust in a sensor if, based on the uncertainties associated with the features extracted from the sensor data, it can be considered to be more reliable.

Our map building architecture works well in environments that consist of planar walls (or curved walls that can be approximated by consecutive strips of planar walls). One way to extend our

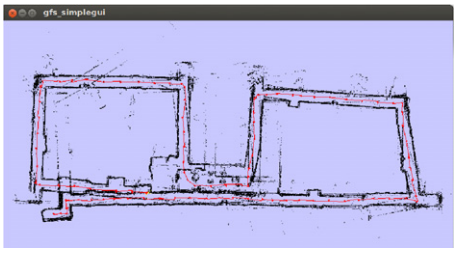


(a) MSEE-BLDG second floor blueprint.



(b) 2D line map and loop closure error.

(c) GMapping result (30 particles were used).



(d) GMapping result (60 particles were used).

Fig. 23. Loop closure error in a large-sized indoor environment: MSEE-building second floor.

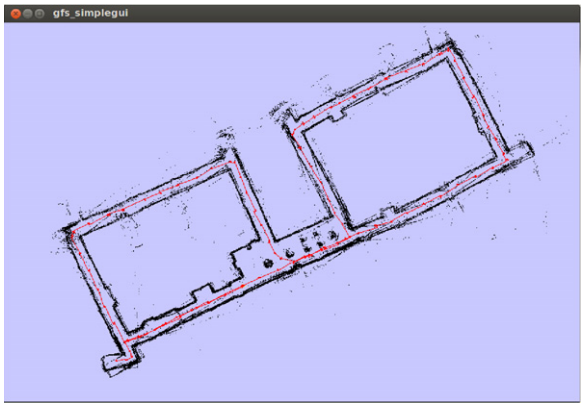


Fig. 24. GMapping result (90 particles were used) using the new data for the MSEE-building second floor.

research would be to remove the assumption of planarity that we associate with the interior structures that need to be mapped. Another important research direction in the future would be to incorporate in our map building logic different procedures for loop detection and loop-closure error elimination.

References

[1] S. Thrun, Robotic mapping: a survey, Tech. Rep., Carnegie Mellon University, 2002.
[2] H. Durrant-Whyte, T. Bailey, Simultaneous localization and mapping (SLAM): part I the essential algorithms, IEEE Robotics and Automation Magazine 13 (2) (2006) 99–108.
[3] H. Durrant-Whyte, T. Bailey, Simultaneous localization and mapping (SLAM): part II state of the art, IEEE Robotics and Automation Magazine 13 (3) (2006) 108–117.
[4] A. Elfes, Sonar-based real-world mapping and navigation, IEEE Journal of Robotics and Automation 3 (3) (1987) 249–265.
[5] J. Leonard, H. Durrant-Whyte, Simultaneous map building and localization for an autonomous mobilerobot, in: IEEE/RSJ International Workshop on Intelligent Robots and Systems, 1991, pp. 1442–1447.
[6] F. Lu, E. Milios, Globally consistent range scan alignment for environment mapping, Autonomous Robots 4 (1997) 333–349.
[7] W. Burgard, D. Fox, H. Jans, C. Matenar, S. Thrun, Sonar-based mapping of large-scale mobile robot environments using em, in: Proc. of the 16th International Conference on Machine Learning, 1999.
[8] J. Castellanos, J. Montiel, J. Neira, J. Tardos, The SPmap: a probabilistic framework for simultaneous localization and map building, IEEE Transactions on Robotics and Automation 15 (5) (1999) 948–962.
[9] S. Pfister, S. Roumeliotis, J. Burdick, Weighted line fitting algorithms for mobile robot map building and efficient data representation, in: IEEE International Conference on Robotics and Automation, 2003, pp. 1304–1311.
[10] C. Taylor, D. Kriegman, Structure and motion from line segments in multiple images, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (11) (1996) 1021–1032.

[11] A. Davison, Real-time simultaneous localisation and mapping with a single camera, in: IEEE International Conference on Computer Vision, 2003, pp. 1403–1410.
[12] K. Wnuk, F. Dang, Z. Dodds, Dense 3D mapping with monocular vision, in: International Conference on Autonomous Robots and Agents, 2004, pp. 212–217.
[13] N. Ayache, O. Faugeras, Maintaining representations of the environment of a mobile robot, IEEE Transactions on Robotics and Automation 5 (6) (1989) 804–819.
[14] J.J. Little, J. Lu, D.R. Murray, Selecting stable image features for robot localization using stereo, in: IEEE/RSJ International Workshop on Intelligent Robots and Systems, 1998, pp. 1072–1077.
[15] L. Iocchi, K. Konolige, Visually realistic mapping of a planar environment with stereo, in: International Conference on Experimental Robotics, 2000.
[16] A.J. Davison, D. Murray, Simultaneous localisation and map-building using active vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7) (2002) 865–880.
[17] W.-C. Chang, S.-A. Lee, Real-time feature-based 3D map reconstruction for stereo visual guidance and control of mobile robots in indoor environments, in: IEEE International Conference on Systems, Man and Cybernetics, 2004, pp. 5386–5391.
[18] J.M. Saez, F. Escolano, A global 3D map-building approach using stereo vision, in: IEEE International Conference on Robotics and Automation, 2004, pp. 1197–1202.
[19] M. Beckerman, A Bayes-maximum entropy method for multi-sensor data fusion, in: IEEE International Conference on Robotics and Automation, 1992, pp. 1668–1674.
[20] J.A. Castellanos, J. Neira, J.D. Tardos, Multisensor fusion for simultaneous localization and map building, IEEE Transactions on Robotics and Automation 17 (6) (2001) 908–914.
[21] P. Weckesser, R. Dillmann, U. Rembold, Sensor-fusion of intensity- and laser range-images, in: IEEE/SCIE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems, 1996, pp. 501–508.
[22] P. Grandjean, A.R.D.S. Vincent, 3-D modeling of indoor scenes by fusion of noisy range and stereodata, in: IEEE International Conference on Robotics and Automation, 1989, pp. 681–687.
[23] H. Baltzakis, A. Argyros, P. Trahanias, Fusion of range and visual data for the extraction of scene structure information, in: Intl. Conf. on Pattern Recognition, 2002, pp. 7–11.
[24] S. Ahn, J. Choi, N. Doh, W. Chung, A practical approach for EKF-SLAM in an indoor environment: fusing ultrasonic sensors and stereo camera, Autonomous Robots 24 (3) (2008) 315–335.
[25] P. Biber, H. Andreasson, T. Duckett, A. Schilling, 3D modeling of indoor environments by a mobile robot with a laser scanner and panoramic camera, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004, pp. 3430–3435.
[26] I. Stamos, P. Allen, Integration of range and image sensing for photo-realistic 3D modeling, in: IEEE International Conference on Robotics and Automation, 2000, pp. 1435–1440.
[27] Y. Liu, R. Emery, D. Charabarti, W. Burgard, S. Thrun, Using em to learn 3D models of indoor environments with mobile robots, in: 18th International Conference on Machine Learning, 2001, pp. 329–336.
[28] K. Ho, P. Newman, Detecting loop closure with scene sequences, International Journal of Computer Vision 74 (3) (2007) 261–286.
[29] K. Ho, P. Newman, Loop closure detection in SLAM by combining visual and spatial appearance, Robotics and Autonomous Systems 54 (9) (2006) 740–749.
[30] P. Newman, K. Ho, SLAM-loop closing with visually salient features, in: ICRA 2005. Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005, IEEE, 2006, pp. 635–642.
[31] K. Granström, J. Callmer, F. Ramos, J. Nieto, Learning to detect loop closure from range data, in: A. Bicchi (Ed.), Proceedings of 2009 IEEE International Conference on Robotics and Automation, 2009, pp. 15–22.
[32] G. Grisetti, C. Stachniss, W. Burgard, Improved techniques for grid mapping with rao-blackwellized particle filters, Robotics, IEEE Transactions on 23 (1) (2007) 34–46.
[33] R.P. Broadwater, H.E. Shaalan, W.J. Fabrycky, R.E. Lee, Decision evaluation with interval mathematics: a power distribution system case study, IEEE Transactions on Power Delivery 9 (1) (1994) 59–67.
[34] K. Briechle, U.D. Hanebeck, Localization of a mobile robot using relative bearing measurements, IEEE Transactions on Robotics and Automation 20 (1) (2004) 36–44.
[35] V. Isler, R. Bajcsy, The sensor selection problem for bounded uncertainty sensing models, in: Fourth International Symposium on Information Processing in Sensor Networks, 2005, pp. 151–158.
[36] C. Detweiler, J. Leonard, D. Rus, S. Teller, Passive mobile robot localization within a fixed beacon field, in: 2006 Workshop on the Algorithmic Foundations of Robotics, 2006.
[37] R. Smith, M. Self, P. Cheeseman, Estimating uncertain spatial relationships in robotics, in: Autonomous Robot Vehicles, vol. 1, 1990, pp. 167–193.
[38] P. Beeson, J. Modayil, B. Kuipers, Factoring the mapping problem: mobile robot map-building in the hybrid spatial semantic hierarchy, The International Journal of Robotics Research 29 (4) (2010) 428–459.
[39] M. Bosse, P. Newman, J. Leonard, S. Teller, Simultaneous localization and map building in large-scale cyclic environments using the atlas framework, The International Journal of Robotics Research 23 (12) (2004) 1113–1139.
[40] Z. Zhang, A flexible new technique for camera calibration, Tech. Rep. MSR-TR-98-71, Microsoft Research, December, 1998.
[41] J.-Y. Bouguet, Camera calibration toolbox for matlab, August 2005. Available from http://www.vision.caltech.edu/bouguetj/calib_doc.
[42] A. Kosaka, A. Kak, Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties, CVGIP: Image Understanding 56 (3) (1992) 271–329.
[43] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (SURF), Computer Vision and Image Understanding 110 (3) (2008) 346–359. http://dx.doi.org/10.1016/j.cviu.2007.09.014.
[44] D. Lagunovsky, S. Ablameyko, Straight-line-based primitive extraction in grey-scale object recognition, Pattern Recognition Letters 20 (10) (1999) 1005–1014.
[45] S. Thrun, W. Burgard, D. Fox, A probabilistic approach to concurrent mapping and localization for mobile robots, Machine Learning 31 (1998) 29–53.

Hyukseong Kwon was born in Seoul, Korea. He received the Bachelor of Science degree in Electronic Communication Engineering from Hanyang University in 1996, the Master of Science degree in Electrical Engineering from University of Southern California in 2001, and his Ph.D. in Electrical and Computer Engineering from Purdue University in 2008. He is now serving as a research associate at the US Air Force Academy. His research interests include computer vision, image processing, visualization, robotics, UAVs, and sensor fusion.

Khalil M. Ahmad Yousef was born in United Arab Emirates and grew up and raised in Jordan. He received the Bachelor of Science degree in Electrical and Computer Engineering from the Hashemite University in 2005 and the Master of Science degree in Computer Engineering from Jordan University of Science and Technology in 2008. Since January 2008, he has been pursuing his Ph.D. in the school of Electrical and Computer Engineering, Purdue University where he is affiliated with Robot Vision Laboratory. His research interests include computer vision, image processing, and robotics.

Avinash C. Kak is a Professor of electrical and computer engineering at Purdue University, West Lafayette, IN. His research and teaching include sensor networks, computer vision, robotics, and high-level computer languages. He is a coauthor of Principles of Computerized Tomographic Imaging, which was republished as a classic in applied mathematics by SIAM, and of Digital Picture Processing, which is also considered by many to be a classic in computer vision and image processing. His recent book Programming with Objects (Wiley, 2003) is used by a number of leading universities as a text on object oriented programming. His latest book Scripting with Objects, also published by Wiley, focuses on object-oriented scripting. These are two of the three books for an "Objects Trilogy" that he is creating. The last, expected to be finished sometime in 2013, will be titled Designing with Object.