Map Building using Cheap Digital Cameras

Trevor Taylor, Wageeh W. Boles, Shlomo Geva
Queensland University of Technology
{t.taylor|w.boles|s.geva}@qut.edu.au

Abstract

Cheap digital cameras are readily available. They can be mounted on robots and used to build maps of the surrounding environment. However, these cameras suffer from several drawbacks such as a narrow field of view, low resolution and limited range due to perspective. These limitations can cause traditional approaches to Simultaneous Localization and Mapping to fail due to insufficient information content in the visual sensor data. This paper discusses these issues and presents a solution for indoor environments.

1. Introduction

Visual SLAM Simultaneous Localization and Mapping (VSLAM) has received a lot of attention in recent years [6, 20, 21]. Several different approaches have been tried with differing degrees of success.

One of the fundamental issues that makes Visual SLAM a difficult problem is perspective. Humans understand perspective as a result of accumulated experience. As small children we learn how to distinguish nearby objects from those far away, and we accept the fact that railway lines appear to converge in the distance even though they are parallel.

However, humans are generally very poor at estimating distances beyond a few meters. It is quite clear that we cannot produce accurate metric measurements from what we see. One reason is that we do not compute distances in a metric sense, but another problem is that perspective makes estimating large distances virtually impossible.

Using a cheap camera adds to the problems of building a map because they typically have narrow fields of view, relatively low resolution, and in some cases poor frame rates.

This paper discusses these issues in terms of the information content requirements for successful SLAM and shows that cheap cameras make it difficult to build maps reliably.

One possible solution to the problem for indoor environments is described. This entails imposing additional constraints on the problem to compensate for the lack of information from the visual sensor.

2. Related Work

Creating maps from video images is not a new area. Various methods have been used to obtain distance information from images, including stereo disparity [16], depth from focus [17], shape from motion [25], and so on.

The transformation that takes place during image capture results in the loss of spatial information. All points in 3D space along a ray from the camera out to infinity map to the same pixel in the image.

However, for example, if we impose the additional constraint that the pixel corresponds to a point on the floor, then the intersection of the ground plane and a ray from the camera is unique. This is the principle of Inverse Perspective Mapping (IPM) that enables us to use a camera as a range sensor [22].

Our approach is to identify the floor by segmenting the image and hence determine the boundaries of the floor. By applying Inverse Perspective Mapping it is possible to obtain a Radial Obstacle Profile that describes the location of all the surrounding obstacles as a linear array which is very similar to a scan from a Laser Range Finder (LRF) [23].

Much of the work on SLAM in recent years has used LRFs, such as GMapping [10] and DP-SLAM [8]. Compared with visual sensors, LRFs are highly accurate and, as shown below, have a larger range of operation than vision. However they have some significant disadvantages compared to vision – they operate entirely in a single plane, they cannot take advantage of the colour of objects and they cannot be used for object recognition in general.

Because localization is a key issue in SLAM, many systems rely on identifying features for matching between images. One well-known approach is SIFT (Scale-Invariant Feature Transform) developed by

0-7695-3067-2/07 $25.00 © 2007 IEEE
DOI 10.1109/DICTA.2007.8

Lowe [13]. SIFT has been used successfully for visual mapping [20].

However, SIFT does not work well in our environment. Fig. 1 shows two images of a corridor taken by one of our robots. Superimposed on the images is the output from sample code available from Lowe's web site that attempts to match features between two images.

In this example the robot rotated by about 5 degrees to the left. Several of the keypoints (denoted by small white arrows) are on the carpet and are clearly extraneous. Only one match was found, as indicated by the white line between the corresponding points in the two images.

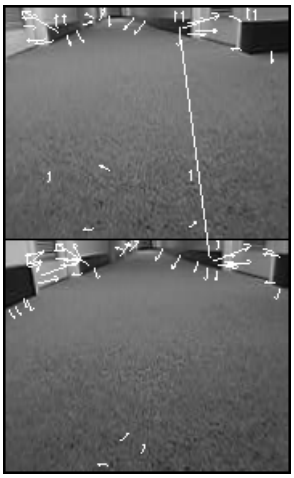

Figure 1. Matching of SIFT features

Another successful approach is MonoSLAM, developed by Davison [5], which applies an Extended Kalman Filter to features obtained with the Shi and Tomasi feature detector. This system uses a hand-held camera, but it works best with a wide-angle lens and a high-speed camera using a Firewire connection [6]. Such cameras do not meet our criteria for a cheap camera.

Smith extended Davison's work to use line features in images in addition to point features [21]. However, the examples shown were for a cluttered indoor environment where a large number of features could easily be obtained.

Line features in the map have also been used. For instance, a Hough transform applied to the map to obtain wall segments [18]. New information could then be examined to see if it coincided with an existing wall. This allowed the orientation of the robot to be adjusted to match existing walls. This work however used sonar, not vision.

More recently, P-SLAM [3] took a similar approach but developed it further. An Environmental-Structure Predictor was applied to the map so that the robot could predict what it might see next. Consequently, localization could be improved separately from traditional SLAM. The predictor used a variety of features, not just wall segments.

3. Visual Sensors

In our experimental work, we have used X80 robots from Dr. Robot in Canada. They are roughly 35cm in diameter with differential drive and are equipped with WiFi networking and built-in pan and tilt cameras (manufacturer unknown).

There are several factors that affect the quality of the information that can be obtained from a camera. For the X80 cameras, the resolution is only 176 by 144 pixels and it is fixed in the firmware. The frame rate is constrained by the architecture of the X80 robots with a theoretical maximum of 4 frames per second (fps), although we have never seen more than 2fps. The camera Field of View (FOV) is approximately 50.8°.

One solution would be to obtain better cameras for the robots. However, this ignores the problems and the cameras would not necessarily be cheap. These problems are discussed in this section.

3.1. Camera Deficiencies

As with all cheap cameras, there is a significant amount of lens distortion on an X80 camera. Therefore we calibrate the camera using Matlab code available on the Internet [1]. This involves taking several images of a calibration grid which can then be used to calculate the intrinsic parameters of the camera.

The Intel OpenCV Library [12] includes a routine that can undistort the camera image given the intrinsic parameters. This is applied to the raw camera images before any further processing takes place. Fig. 2 shows an example of lens distortion and the corrected image.

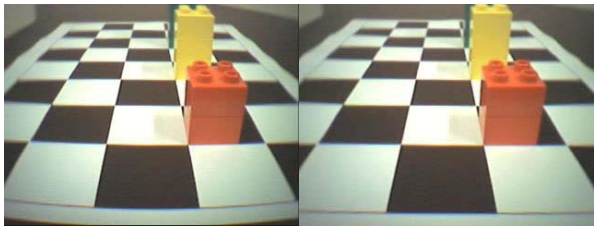

Figure 2. Correcting for lens distortion

Another problem is that the CCD in the camera module might not be mounted perfectly horizontally. This has a significant impact on range measurements because even a couple of pixels variation across the image can equate to tens of centimeters (see Fig. 3). It

is therefore necessary to estimate this side-to-side tilt angle (sometimes called the roll angle) from the camera extrinsic parameters. These parameters can also be calculated using the software from [1]. Images can be transformed by this angle to eliminate the tilt.

Cheap cameras typically have poor performance in low light. This problem can be corrected to some extent by adjusting the brightness of the image, but this is not always successful.

3.2. Perspective

The most significant effect of perspective is that the distance from the camera varies non-linearly as you move further up an image.

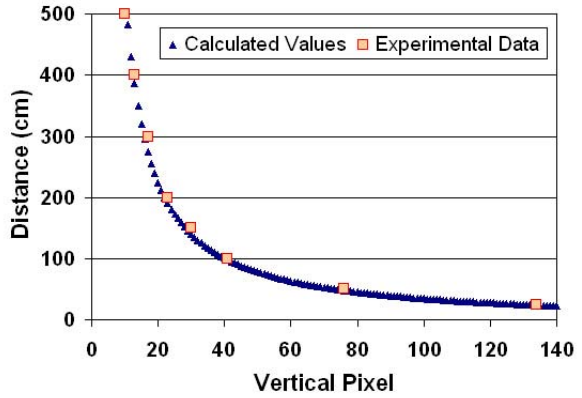

Figure 3. Range versus pixel (scanline) for a camera image

Consider Fig. 3 which shows the distance from the camera (across the floor) versus the vertical pixel coordinate (scanline). According to the usual convention for bitmap images, the top scanline in the image has a pixel y-coordinate of zero. The bottom scanline in this case is row 143.

The figure shows calculated range values for the camera on an X80 robot using Inverse Perspective Mapping as developed in [22]. The experimental data points validate the IPM calculations.

By the time the distance from the camera reaches 200cm the difference from one scanline to the next corresponds to over 10cm across the floor. The differences increase rapidly from there.

When drawing an occupancy grid map with a cell size of 5cm, the uncertainty in the location of obstacles makes the map unreliable beyond a couple of meters. In a probabilistic sense, distant obstacles become "smeared" over the map and eventually contribute very little information.

Therefore we usually set a cutoff for the visual range of between 2 and 4 meters. In effect, perspective places a limit on the maximum range of a visual sensor before the uncertainty in the range data makes it useless.

Note that the resolution of the camera has an impact on the maximum range. If the resolution was increased, the maximum range could also be increased because the point where the differences start to exceed the grid size would be further away. However, doubling the resolution does not double the maximum range, but it quadruples the number of pixels which affects the computational load for image processing.

Contrast this with a Laser Range Finder where the accuracy of the range data is within 1 or 2cm across the entire range of the LRF. This is well within a grid size of 5cm. Depending on the specifications of the LRF, the maximum range can be from 8m to over 100m.

3.2. Field of View

For a cheap CCD camera, such as a webcam, the Field of View is typically around 50-60°. This is very narrow and it makes teleoperating a robot quite difficult because we are used to having good peripheral vision.

In comparison, the human visual system has a FOV of around 160° for a single eye, and up to 200° when using both eyes [26]. Laser Range Finders typically have a FOV of 180°.

Combining the factors of limited range (due to perspective effects) and narrow FOV, the result is that a camera provides significantly less information about surrounding obstacles than a LRF.

Fig. 4 shows the difference in the areas covered by a LRF and a camera. It is immediately apparent that the information content of a LRF scan is significantly greater than for a camera.

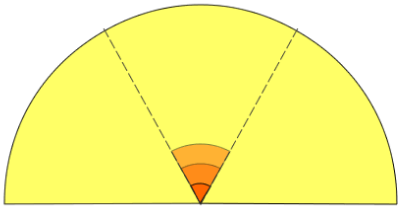

Figure 4. Comparison of field of view for a laser range finder and a camera

In Fig. 4 the semi-circle represents the area covered by a LRF with a 180° FOV and a 20m range. The smaller (dark) segments in the centre of the diagram are for a camera with a 60° FOV and ranges of 2, 4 and 6m. The diagram is drawn to scale.

The ratio of the areas covered is 75:1 for a LRF compared to a camera with only a 2m range. However, this is for a LRF with only a 20m range. There are

LRFs available with ranges of over 100m. Clearly this is a substantial advantage when building a map.

3.3. Frame Rate

Frame rate also has an effect on the information content, but in a temporal sense rather than a spatial sense. The higher the frame rate, the smaller the changes between frames when the robot is moving. This helps in calculating optic flow, or conversely in tracking the robot's motion.

Campbell demonstrated reliable tracking using a commercial webcam over short distances [2]. The robot used the texture of the floor to measure motions using optic flow. However, the system did not perform SLAM.

MonoSLAM [5] works with a hand-held camera. Davison indicated (in a personal communication) that a high frame rate is essential for the operation of his system. In our experience his system could not track the motion of the camera on an X80 robot at only 2 frames per second. This is not surprising, but it is a deficiency of the camera, not the MonoSLAM software.

MonoSLAM has one drawback which is that the scale of the map cannot be determined absolutely. As a result, the map will become stretched if the camera is moved more quickly. This problem can be overcome by using map matching to recognize loop closure and then re-scaling the map to fit [4].

Our system, on the other hand, is calibrated for actual range measurements so it produces maps that are to scale.

4. SLAM

In essence, the fundamental problem of SLAM is determining where you are – the Localization step. In comparison the Mapping step is relatively easy.

Odometry information based on wheel encoders is notoriously unreliable [15]. Without accurate information about a robot's motions it is not possible to draw an accurate map. Therefore SLAM attempts to use accumulated knowledge about the environment (in the form of a map) to improve the prediction of the robot's real pose (the position and orientation).

Fig. 5 shows an occupancy grid map drawn using data obtained by an X80 robot as it explored one of the buildings on our campus. This map was drawn based solely on wheel odometry information.

In Fig. 5a the usual convention is used where white represents free space, black cells are obstacles and shades of grey indicate various degrees of uncertainty about the occupancy of cells. The grid size in this case was 5cm and the map covers an area of roughly 20m by 20m.

For comparison, Fig. 5b shows the actual floor plan which was measured to within 1cm. The round objects are concrete pillars.

Errors due to rotations have the most significant impact on the resulting map because a small angular error quickly equates to a large position error if left unchecked. This is quite obvious in Fig. 5a because the corridors are not straight and most of them do not meet at right angles.

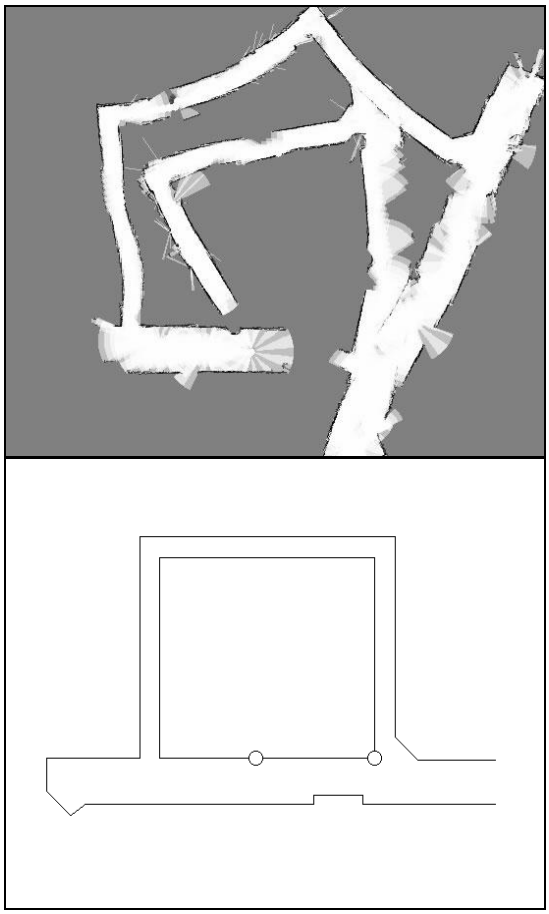

Figure 5. Maps (a) drawn using odometry (b) actual floor plan

Applying a Particle Filter in the form of the GridSLAM algorithm [24] improves the map noticeably, but it is still not correct. An example is shown in Fig. 6 using 50 particles.

Notice in Fig. 6 that the last leg of the route at the top of the map is not clearly defined. This map is an average of the 50 particles and the lack of definition indicates that the particles were diverging.

It might be argued that 50 particles are insufficient for the task and that more particles would solve the

problem. However, we have found that even 500 particles are not enough to guarantee stability, and with that many particles the computations take so long that they have to be done offline. In any case, the point is that we have developed a solution (outlined below) that works with only 50 particles.

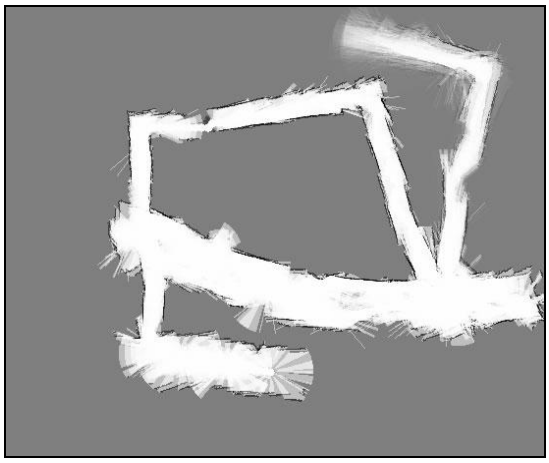

Figure 6. Map drawn using GridSLAM with 50 particles

The basic problem here is map correlation which is used for calculating the particle weights. There is not enough information available from the camera to reliably localize the robot against the map.

The narrow FOV and short range discussed above cause this problem. It should also be born in mind that for much of the time the robot is exploring and so it is seeing new areas and therefore there is not much to compare against in the existing map.

Much of the recent work on SLAM has used scan correlation to improve the robot's pose estimates [11]. This is feasible with a LRF, but not as useful for a camera. In our case, we obtain a data set similar to a laser scan by tracing rays through the image. However, the camera only has a narrow FOV and low frame rate, so scan matching does not work for many of the images.

5. Proposed Solution

For indoor environments, we can take advantage of human beings' propensity for building structures with straight walls and square corners.

Corridors are a common feature in all office buildings. Many solutions have been used for reactive corridor navigation ranging from simple wall following (keeping a constant distance from the wall) [9] to balancing the optic flow on either side (a technique learnt from bees) [19].

These approaches make the implicit assumption of parallel walls. We take this further and assume that walls are, in general, either parallel or orthogonal to one another.

When the robot is first started, it locates the longest wall segment in view and uses this to determine its initial orientation. For instance, if it is directly facing a wall then this will become the East-West axis of the map.

As the robot moves around, it continually extracts wall segments from the images. This process is called incremental localization and it works as follows:

1. Segment the floor out of the image (Fig. 7a) to obtain a floor contour (Fig. 7b);
2. Use the Douglas-Peucker algorithm [7] to obtain a set of straight edges (Fig. 7c);
3. Select lines (wall edges) that are longer than a specified threshold;
4. Transform these lines to the world coordinate system;
5. Compare the alignment of each of the wall segments with the cardinal orientations of North-South and East-West in the map;
6. If the difference between a wall segment and a standard orientation is less than a threshold, add this difference to the sum of differences; and
7. Adjust the robot's estimated orientation based on the average differences.

In step 3, the threshold is determined experimentally. By ensuring that a reasonable length of wall must be visible in the image, small amounts of clutter, such as furniture, rubbish bins, etc., can be ignored. Provided that the robot can see the wall from time to time it will be able to re-align itself.

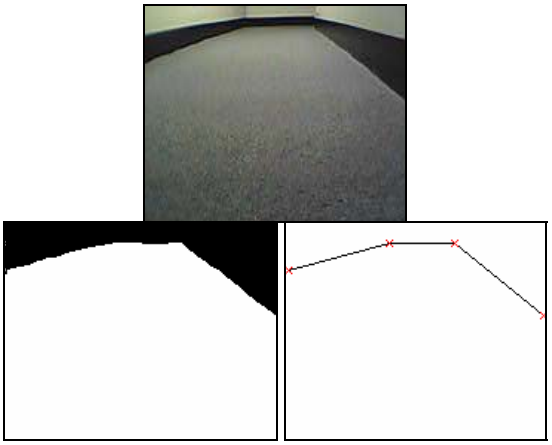

Figure 7. Extracting wall segments (a) original image (b) segmented image (c) line segments

Image segmentation is used to obtain the local free space map for incorporation into the global map. It is performed in two steps. Firstly, the most significant edges in the image are obtained using a Canny edge detector. Then a flood fill is performed with a floating range and using the edges as a mask. The result is shown in Fig. 7b.

The edges are used to constrain the fill and in most cases they represent the wall boundaries. Martin [14] found that in an indoor environment with a floor that is roughly a uniform colour, edges are the best indicators for wall boundaries. However, sometimes the edges are not detected reliably so a threshold on the flood fill prevents the fill from spreading across the entire image.

5.1. Segmentation Issues

From time to time, the flood fill "leaks" through a gap in the detected edges. This results in small streaks that are visible in the map. (See Fig. 6 for instance.) However, they are too narrow for the robot to pass through and therefore they are ignored by the exploration algorithm.

In a few cases, there are larger gaps in the walls in Fig. 6. These occurred because the doors were a similar colour to the carpet. The robot uses a sonar sensor as a safety measure to ensure that it does not try to drive though a closed door that it has mistakenly seen as part of the floor.

A more significant problem is that the camera does not operate well in low light. Consequently the robot sometimes fails to detect portions of the floor.

The dark spot in the upper corridor of Fig. 8 at the top left labeled (a) was a result of one of the overhead fluorescent lights not working. The robot almost lost sight of the floor completely.

In these situations the pixel values become dark grey and approach black. The seed value for the flood fill is based on a patch of floor that the robot saw in front of it when it first started up. (This allows it to handle different coloured floors and some variations in lighting.) However, even with a floating range, the seed will not be matched if the image is almost black.

These issues affect map quality, but have not been found to have any adverse effects on the algorithm discussed here.

5.2. Incremental Localization

The output from the Douglas-Peucker algorithm is a polygonal approximation to a contour which consists of straight lines. In the case of Fig. 7c these represent three different walls. Each one can provide information about the orientation of the robot. Therefore they are averaged together using a weighted average based on the length of the line segment – the longer a segment is, the more reliable it is presumed to be.

When the incremental localization algorithm is applied to the same data as the test run shown in Fig. 5, the map is improved, as shown in Fig. 8. This map is substantially correct.

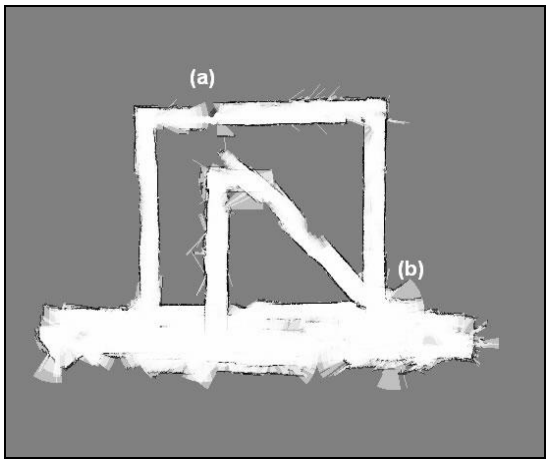

Figure 8. Map produced by applying constraints to wall orientations

However, the map in Fig. 8 is still wrong in one major area. The path of the robot has been incorrectly estimated at the bottom right-hand intersection, labeled (b), resulting in a fictitious corridor being created. This 45° corridor should overlay the North-South one. (This happens to correspond to a 45° wall which might be the reason for the error.)

When the robot reached the end of this corridor after traveling North, it found a 90° corner. Unable to reconcile this, it created a new corridor heading South which should instead overlay the one heading West.

Note that once the map has diverged substantially from reality, as in the middle of Fig. 8, SLAM might not be able to recover. This happens because the robot thinks it is exploring a new area and there is nothing in the map to localize against.

It would require a global search of the map to find the robot's actual pose. Particle Filters inherently operate only in a local area. The size of this area depends on the variances incorporated into the motion model, which affect the distribution of the particles.

Combining GridSLAM and the incremental localization technique into a single algorithm generated the final map shown in Fig. 9. Apart from the problem noted in Fig 8. at point (a) with the lighting and spurious gaps in the walls, the map is correct. (Compare it to Fig 5a.)

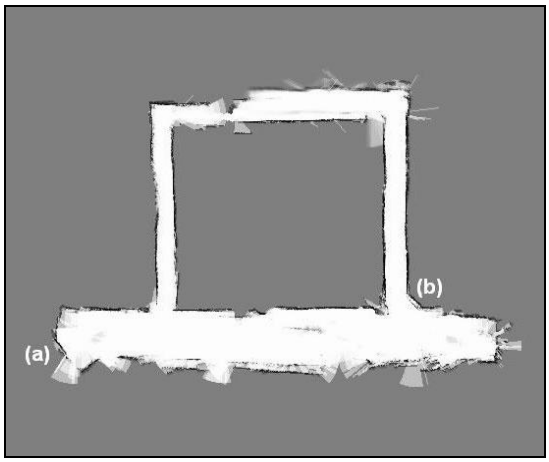

Figure 9. Map using GridSLAM and wall orientation constraints

It is worth noting that there are two areas in the map where the walls are not aligned with the cardinal directions. At the bottom-left of the map, labeled (a), the wall makes a 'V' in the map. This is correct. (Although a door was mistaken for free space.)

At the right-hand bottom intersection, labeled (b), another 45° wall is visible. These "diagonal" walls show that the combination of incremental localization and SLAM is tolerant of short stretches of wall that do not comply with the assumption of orthogonality.

Note that the East-West corridor at the bottom of the map, from (a) to (b), is in fact 2½ times the width of the other three corridors. This is reflected in the map, which shows that the measurements are reasonably accurate.

Our testing has so far been in corridors that are largely uncluttered. However, wall segments are not used for localization unless they exceed a specified length. This prevents the robot from becoming disoriented due to small objects along the walls.

6. Conclusions

This paper has explained how the combination of perspective effects, low frame rate and a narrow field of view from a cheap camera make visual SLAM very difficult and can even cause SLAM to fail.

These problems can be alleviated to some extent by using better cameras. However, they cannot be completely eliminated. In particular, perspective will always impose an upper limit on the range of a visual sensor.

One possible solution has been shown which relies on the walls in indoor environments being parallel or orthogonal to one another. This additional constraint compensates for the lack of information content in the video images and makes SLAM more reliable.

Note that this solution provides the robot with absolute orientations. However, it does not address the issue of position. Therefore the estimation of distances is totally dependent on the SLAM algorithm. This requires further work.

For pure translations, such as moving forwards, wall segments might be used to estimate the distance moved. However, the effects of perspective mean that motion estimates become less reliable the further the wall is away from the robot. Therefore a method will need to be developed that allows adjustments to the estimated position taking into account the reliability of the sensor information. One possible approach might be to use scan matching as in [10] but with an inverse weighting based on the range from the robot.

This solution is only suitable for a limited range of environments. Finding alternative solutions is the subject of ongoing research.

7. References

[1] J.-Y. Bouguet, "Camera calibration toolbox for Matlab," retrieved 19-Mar, 2006, from http://www.vision.caltech.edu/bouguetj/calib_doc

[2] J. Campbell, R. Sukthankar, I. R. Nourbakhsh, and A. Pahwa, "A Robust Visual Odometry and Precipice Detection System Using Consumer-grade Monocular Vision," presented at IEEE Conference on Robotics and Automation (ICRA), pp. 3421–3427, 2005.

[3] H. J. Chang, C. S. G. Lee, Y.-H. Lu, and Y. C. Hu, "P-SLAM: Simultaneous Localization and Mapping With Environmental-Structure Prediction," *IEEE Transactions on Robotics*, vol. 23, pp. 281-293, 2007.

[4] L. A. Clemente, A. J. Davison, I. D. Reid, J. Neira, and J. D. Tardós, "Mapping Large Loops with a Single Hand-Held Camera," presented at Robotics: Science and Systems (RSS), Atlanta, GA, available online, 2007.

[5] A. J. Davison and D. W. Murray, "Mobile Robot Localisation Using Active Vision," presented at 5th European Conference on Computer Vision (ECCV), Freiburg, Germany, pp. 809-842, 1998.

[6] A. J. Davison, Y. G. Cid, and N. Kita, "Real-time 3D SLAM with Wide-Angle Vision," presented at 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles (IAV), Lisbon, Portugal, CD-ROM, 2004.

[7] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Canadian Cartographer*, vol. 10, pp. 112-122, 1973.

[8] A. Eliazar and R. Parr, "DP-SLAM: Fast, Robust Simultaneous Localization and Mapping Without Predetermined Landmarks," presented at International Joint Conference on Artificial Intelligence, pp. 1135-1142, 2003.

[9] S. Fazil and L. Kleeman, "Wall following and obstacle avoidance results from a multi-DSP sonar ring on a mobile robot," presented at IEEE International Conference on Mechatronics and Automation, vol. 1, pp. 432-437, 2005.

[10] G. Grisetti, C. Stachniss, and W. Burgard, "Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters," *IEEE Transactions on Robotics*, vol. 23, pp. 34-46, 2007.
[11] D. Hähnel, W. Burgard, D. Fox, and S. Thrun, "An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," presented at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 206-211, 2003.
[12] Intel, "Open Source Computer Vision Library," retrieved 2-Aug, 2005, from http://www.intel.com/technology/computing/opencv/
[13] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, pp. pp 91-110, 2004.
[14] M. C. Martin, "Genetic Programming for Robot Vision," presented at From Animals to Animats 7: The Seventh International Conference on the Simulation of Adaptive Behavior, pp. 256-265, 2002.
[15] R. R. Murphy, *Introduction to AI Robotics*. Cambridge, MA: MIT Press, 2000.
[16] D. Murray and J. J. Little, "Using Real-Time Stereo Vision for Mobile Robot Navigation," *Autonomous Robots*, vol. 8, pp. 161-171, 2000.
[17] I. R. Nourbakhsh, D. Andre, C. Tomasi, and M. R. Genesereth, "Mobile robot obstacle avoidance via depth from focus," *Robotics and Autonomous Systems*, vol. 22, pp. 151-158, 1997.
[18] J. M. Pérez Lorenzo, R. Vázquez-Martín, P. Núñez, E. Pérez, and F. Sandoval, "A Hough-based Method for Concurrent Mapping and Localization in Indoor Environments," presented at IEEE Conference on Robotics, Automation and Mechatronics (RAM), Singapore, vol. 2, pp. 840-845, 2004.
[19] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi, "Divergent stereo for robot navigation: learning from bees," *IEEE Computer Vision and Pattern Recognition*, pp. 434-439, 1993.
[20] S. Se, D. Lowe, and J. J. Little, "Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks," *International Journal of Robotics Research*, vol. 21, pp. 735-758, 2002.
[21] P. Smith, I. Reid, and A. Davison, "Real-Time Monocular SLAM with Straight Lines," presented at British Machine Vision Conference (BMVC), 2006.
[22] T. Taylor, S. Geva, and W. W. Boles, "Monocular Vision as a Range Sensor," presented at International Conference on Computational Intelligence for Modelling, Control & Automation (CIMCA), Gold Coast, Australia, pp. 566-575, 2004.
[23] T. Taylor, S. Geva, and W. W. Boles, "Early Results in Vision-based Map Building," presented at 3rd International Symposium on Autonomous Minirobots for Research and Edutainment (AMiRE), Awara-Spa, Fukui, Japan, pp. 207-216, 2005.
[24] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.
[25] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method," *International Journal of Computer Vision*, vol. 9, pp. 137-154, 1991.
[26] B. A. Wandell, *Foundations of Vision*. Sunderland, MA: Sinaur Associates, 1995.