Abstract: This paper presents an approach to Simultaneous Localization and Mapping (SLAM) based on monocular vision. Standard multiple-view vision techniques are used to estimate robot motion and scene structure, which are then integrated with minimal odometric information and used to build a global environment map. Preliminary experimental results are also presented and discussed.

Key-Words: Robot localisation, Mapping, Monocular vision, SLAM

1 Introduction

SLAM - Simultaneous Localisation And Mapping - is the process of tracking the pose of a mobile robot, relative to its environment, while simultaneously building a map of the environment itself. Whatever the high-level goal of the mobile robot, SLAM is a critical factor for successful navigation in a partially or totally unknown environment, and has therefore been a highly active research topic during the last few years (see e.g. [1] for a recent survey).

Implementing SLAM obviously requires some ability of the robot to perform range sensing on its environment. As a matter of fact, most existing approaches to SLAM make use of sonars or laser scanners [2, 3, 4, 5]. Vision based SLAM is both more computer-intensive and difficult to implement, as stable visual landmarks are hard to detect and track over time. Most vision based approaches use binocular or trinocular stereo just as a range finding device [6, 7, 8]; the use of monocular vision has only recently drawn some attention [9, 10, 11, 12, 13]. Finally, almost all the published approaches use recursive statistical techniques (Kalman or Bayesian filtering) which, though successful on the short term, suffer from error accumulation over time.

This paper presents the approach to SLAM we are currently developing, which is based on monocular vision. From each image of the sequence acquired while the robot is moving, features are extracted to serve as visual landmarks. Such features are tracked along the sequence and used to get local estimates of robot motion and world structure from triples of sufficiently spaced images. Consecutive overlapping triples are recursively registered and integrated with available odometry in order to obtain an estimate of robot pose and of world point coordinates in a global reference. The latter are used to build a 2D occupancy grid map. In case of cyclic environments, visual landmarks are used to detect and fix the accumulated error.

The paper also presents some preliminary results obtained with our Pioneer 2 AT mobile robot equipped with a Sony XC55 camera.

2 SLAM by monocular vision

Our approach can be summarised as follows:

- extraction of features from the images and their tracking along the sequence;
- estimation of local motion and structure from triples of key frames, i.e. images showing an amount of disparity sufficient for a reliable estimation;
- registration of the estimates from different triples into a same reference frame, and integration with independent odometric information;
- building of a global occupancy grid map from the visual measurements;
- possible correction of accumulated errors in case of cyclic environments.

2.1 Feature detection and tracking

From each frame of the image sequence acquired while the robot is moving, various kinds of features can be extracted to serve as visual landmarks: corners, edge segments, textured patches etc. Our current implementation uses Shi-Tomasi corners [14], i.e. small textured image patches, whose centers yield pointwise measurements useful for motion/structure estimation. A significant advantage of Shi-Tomasi features is that their definition implicitly provides an efficient frame-to-frame tracking algorithm; other approaches may require independent feature extraction from each image and a costly search for matching pairs. Moreover,
since the tracking algorithm allows for affine distortion, such features can be successfully followed over large relative displacements in the image.

Features are extracted in the first frame of the sequence, and thereafter at more or less regular intervals; the frames from which new features are extracted are called key frames. The spacing between key frames is chosen to provide enough image disparity for a reliable estimation of motion and structure.

### 2.2 Estimation of local motion and structure

For each triple of key frames a local estimate of viewing geometry and 3D structure is obtained by standard projective geometry techniques.

An estimate of the viewing geometry of the triple, i.e. the trifocal tensor $T$ of the three views [15], is first computed by a robust Least Median Squares algorithm [16, 17]. To this extent, the six-point algorithm from [18] is applied to some suitable number $N$ of randomly chosen six point subsets $S_i$, $i = 1, \ldots, N$ of the points appearing in the three frames, yielding $N$ estimates $T_i$. For each $T_i$, a measure of fitting error in the image plane is evaluated on all matched points, and a robust statistics (median) of this error is computed. The $T_i$ with the lowest median error is kept.

This initial estimate is fed as starting point to a bundle adjustment procedure, which finds an optimal estimate of both viewing geometry and 3D structure by minimising the reprojection error on the image plane.

The final output of this step is therefore an estimate of the poses of the camera at the second and third views, and of the positions of the observed 3D points, all relative to the first camera of the triple, and scaled so that the displacement of the second camera is unity.

### 2.3 Chaining the triples

The main problem with the procedure outlined in Sec.2.2 is that it only yields a local unscaled estimate of motion and structure. Indeed, if $x_{i(k)} = [u, v, w]^T$ are the (2D projective) calibrated image coordinates, in key frame $k$, of some point $i$, and $X_{i(k)} = [X, Y, Z]^T$ its 3D coordinates in the reference of view $k$, then

$$
\begin{align*}
x_{i(k)} & \propto X_{i(k)} \\
x_{i(k+1)} & \propto R_{1(k)}^{-1} X_{i(k)} + t_{1(k)} \\
x_{i(k+2)} & \propto R_{2(k)}^{-1} X_{i(k)} + t_{2(k)}
\end{align*}
$$

where $[R_{1(k)} | t_{1(k)}]$, $[R_{2(k)} | t_{2(k)}]$ are the rototranslations of views $k + 1$ and $k + 2$ relative to view $k$.

On the other hand, if $X_{W}$ are the 3D coordinates of point $i$ in some global reference (e.g. the camera frame at the start of the robot trajectory),

$$
X_{i(k)} = R_{W(k)} X_{W} + t_{(k)}
$$

where $[R_{W(k)} | t_{(k)}]$ represents the camera pose at view $k$, in the same global reference.

Now, the procedure of Sec.2.2 only provides estimates of $R_{1(k)}$ and $R_{2(k)}$ and $t_{1(k)}$, $t_{2(k)}$ of $X_{i(k)}$, $t_{1(k)}$, $t_{2(k)}$ up to a common scale factor $s(k)$, i.e. $X_{i(k)} = s(k)X_{i(k)}$ and similarly for translations. There remain the problem of determining the $s(k)$’s so that the local estimates can be merged together.

The problem of registering local 3D reconstructions from a sequence has been considered in [19], where a general projective solution was proposed. With a calibrated camera, and considering only adjacent overlapping triples, a simpler approach can be devised. With reference to Fig. 1 and Fig. 2, several relations between quantities pertaining to adjacent triples can be written down. Among those, we choose the following:

$$
\begin{align*}
s_{(k+1)} &= s_{(k)} \frac{R_{2(k)} t_{2(k)} - R_{1(k)} t_{1(k)}}{\| t_{1(k)} \|} \\
R_{(k+1)} &= R_{1(k)} R_{(k)} \\
t_{(k+1)} &= R_{1(k)} t_{(k)} + s_{(k)} t_{1(k)}
\end{align*}
$$

which provide a recursive updating procedure once some initial values $s^{(0)}$, $R^{(0)}$ and $t^{(0)}$ are given (for example, $s^{(0)} = 1$, $R^{(0)} = 1$ and $t^{(0)} = 0$).

![Figure 1: Relations between camera and world reference at time k.](image)

The knowledge of $s_{(k)}$, $R_{(k)}$ and $t_{(k)}$ allows then to transform the reconstructed points $X_{i(k)}$ into the global reference frame. There still remain the problems of combining the 3D estimates of points seen in different triples, and of devising a strategy for coping...
with the possibility of non-overlapping triples due to some accident. These issues are discussed in the next Section.

2.4 Integration with odometry

The above procedure yields a series of noisy measurements of the robot pose and of some landmarks in the environment. The standard technique for integrating such a series of measures is the Extended Kalman Filter (EKF), as e.g. in [10, 9, 13].

The measures so obtained, however, are only known up to a scale factor. This is an unavoidable ambiguity of monocular vision. The scale factor can be determined either by some knowledge of the environment (e.g. the size of some easily recognizable landmarks), or from some knowledge of the robot motion (odometry). In the first case, however, besides the need for information that may not be available, once the known landmarks are out of view, updating must rely on the estimated \(s^{(k)}\)'s which, in our experience, are rather unstable. For this reason, resorting to independent odometric information appears avoidable.

Moreover, a full-fledged EKF requires a detailed model of the robot motion in response to control inputs. Such information, even if available, makes the model heavily platform-dependent. We propose instead the following approach, which only requires minimal odometry information, i.e. the displacement of the robot between two key frames. In the absence of an independent odometer, the displacement predicted according to control inputs can be used as well.

Initially, the scale factor is set according to the odometric displacement between the first two key frames. Then, at each key frame time, the robot displacement estimated by the visual algorithm is compared to the odometric one. If the difference is below a predefined threshold, the odometry is used only for correcting the scale factor, and the 3D points are updated by a simple weighted average, using the covariances estimated by the bundle adjustment. Otherwise, the set of visual measurements is considered an outlier and discarded; the odometric displacement is used to update the robot position, while the 3D point estimates are not updated.

This use of odometry information makes the whole process more tolerant to sporadic failures. Indeed, if for some reason the visual estimates are not reliable at one or a few consecutive key frame times, the robot pose can still be updated blindly via odometry. However, this approach cannot cope with a catastrophic failure, e.g. a complete loss of tracked features due to an unwanted fast movement of the robot. In such a case, the procedure is restarted from the current position. When a sufficiently detailed local map is available, it is compared with the previously built global map, and the robot position estimate is then restored.

2.5 Map building

When the robot motion is constrained to be planar, as in a typical indoor environment, the 3D measures obtained from the vision algorithm can be used to build a 2D occupancy grid map [2, 3, 4]. The latter is a 2D metric map of the robot’s environment, where each grid cell contains a value representing the robot’s subjective belief whether or not it can move to the center of the cell. Grid maps are a convenient way of representing the global structure of the environment; matching a local map with the previously built global map allows an easy estimation of the robot location. Moreover, grid maps allow easy integration of measurements from different sensor types.

The use of 3D measures for building a 2D map requires collapsing the measures onto a reference “ground” plane, i.e. the plane containing the robot trajectory. The plane orientation relative to the camera axes can either be given once for all by a suitable calibration procedure, or estimated by the robot itself after it has wandered awhile. Moreover, assuming that the robot knows its size and the height of the camera above ground, the measures located below the plane or above a suitable height can be discarded.

There is extensive literature on map updating by probabilistic methods. However, the latter imply a model of the sensor statistics which is hardly applicable to the output of a vision system, as remarked e.g. in [8]. Therefore, we use a simple heuristic approach, similar to the one in the cited reference, which has proven adequate. The occupancy values are in the range \((0,1)\), with 0 indicating a surely free cell, and 1 a surely occupied one. Initially, every grid cell is set to a value of 0.5, representing complete lack of knowledge. With reference to Fig. 3, each new measure \(m\) (3D point) determines two regions on the grid: a region \(O(m)\) which is more likely occupied, given the measure, and a region \(C(m)\) which is likely to be clear. The size of \(O\) should be correlated to the measurement covariance, but for the sake of simplicity we adopted instead a fixed angular width \(\epsilon\) and a longitudinal size determined as a fraction \(2\alpha\) of the distance.
The map is then updated by increasing the values of cells in $\mathcal{O}$ by a fixed amount $\Delta$, and decreasing by the same amount the cells in $\mathcal{C}$. Typical values for the parameters are $\alpha = 0.05$, $\Delta = 0.08$ and $\varepsilon$ equal to the angle covered by one image pixel.

![Diagram](image.png)

Figure 3: Updating of the occupancy grid by a new measurement.

Using as measures $\mathbf{m}$ the estimated 3D points yields a map which can sometimes be too sparse. However, the estimated motion of the robot allows to compute a dense visual disparity map at each key frame using standard stereo algorithms. To this extent, the pair of images at two consecutive key frames can be rectified by polar mapping around the epipole [20]. The rectified images are then processed by a standard disparity detection algorithm (e.g. simple correlation), and the so obtained disparity is mapped back to the original image. The distances from the disparity map could then be used to update the occupancy grid as above.

## 2.6 Cyclic environments

The previously outlined procedure suffers from a drawback common to all recursive updating approaches, namely the drift of the estimated robot pose, whose error grows with time. This problem can be tackled in various ways, e.g. [4] exploits the (usual) orthogonality of walls in an indoor environment. A more general solution, however, can only be devised in the case of cyclic environments, where the robot comes back to some previously visited location, so that there is an opportunity to compare the estimated robot position in the environment to the actual one. This task is eased by the use of a visual sensor. In our implementation, a suitable module monitors the grabbed image searching for large well-identifiable image patches, which are stored as visual landmarks when found. When the estimated robot trajectory indicates that the robot’s field of view at some frame time $k$ may overlap the one at some previous frame time $i$ for which visual landmarks are available, this event is verified by comparing the stored landmarks against the current view. In case of success, features from frame $i$, and for which a 3D estimate is available, are found in frame $k$ and used to get an estimate of current robot pose. The error, i.e. the difference between the latter estimate and the one given by the recursive algorithm, is distributed over the trajectory from $i$ to $k$, and the 3D point estimates are corrected accordingly; a global bundle adjustment over the trajectory from $i$ to $k$ can then be performed both on camera poses and 3D world.

## 3 Experimental results

This section presents some results obtained by offline processing sequences of images acquired with our Pioneer 2 AT robot, equipped with a Sony XC55 progressive camera. This camera yields non-interlaced B/W images at a resolution of $640 \times 480$, 30 fps. However, we have found that lower resolutions, both spatial and temporal, are enough to give good localisation results.

For the sake of simplicity, the robot was programmed to follow a circular path in a library room, at a speed of about 0.1 m/s. Fig. 4 shows a frame near the end of the trajectory. This environment exhibits highly textured walls, which can be expected to give good results for what concerns corner detection, although reliable tracking is challenging due to repetitive texture.

![Image](image.jpg)

Figure 4: A robot view of the environment.

The initial robot pose in the environment has been estimated independently, by triangulating some points of known world coordinates seen in the initial frames of the sequence; as concerns odometry, we have preliminarily calibrated the robot response to control inputs, so we could use the latter instead of actual odometric readings, that were not available.

The results presented here were obtained after sub-sampling the images by a factor of two, yielding an effective $320 \times 240$ resolution; a decimation factor of
two was also applied in time (15 fps). The minimum key frame interval was set to 20 frames (about 1.3 s).

Fig. 5 shows four top views, in the world reference, of the results of the procedure outlined in the previous section with different sets of processing options. In the figure, the circles represent the initial and final robot positions along the estimated trajectory, while the dots are the final estimates of the observed 3D points. The CAD model of the room has been superimposed to facilitate a visual judgment of the quality of the results.

The results of Fig. 5(a) and (b) were obtained us-
ing the odometry only for setting the scale factor: (a) without taking into account the cyclic nature of the trajectory, (b) after applying the global adjustment mentioned in Sec. 2.6.

By contrast, the results in Fig. 5(c) and (d) were obtained with the outlier detection test, which in this case was triggered on about 25% of the key frames. The improvement, both in localisation and 3D reconstruction, provided by the outlier detection test is apparent. Indeed, in this latter case the effect of the global adjustment step is barely appreciable from the figure.

Finally, Fig. 6(a) shows an occupancy grid map obtained, from the 3D data corresponding to Fig. 5(a) by the method outlined in Sec. 2.5. For comparison, Fig. 6(b) shows the results obtained by processing full resolution $640 \times 480$ images at 15 fps. Note that the large central uncertainty area seen in both figures is a consequence of the chosen trajectory and of the limited camera field of view.

4 Conclusion
The above results indicate that monocular vision, supplemented with minimal odometric information, can be used profitably for SLAM. Localisation results appear adequate; on the other hand, grid maps are not as good as those obtained by other means. More work is therefore needed in this respect, e.g. by integrating dense disparity maps as mentioned at the end of Sec. 2.5.

References: