Stereo Vision as a Sensor for EKF SLAM

Wikus Brink
Electronic Systems Lab
Electrical and Electronic Engineering
University of Stellenbosch
Email: 14839806@sun.ac.za

Corné E. van Daalen
Electronic Systems Lab
Electrical and Electronic Engineering
University of Stellenbosch
Email: cvdaalen@sun.ac.za

Willie Brink
Applied Mathematics
Department of Mathematical Sciences
University of Stellenbosch
Email: wbrink@sun.ac.za

***Abstract*—Simultaneous localisation and mapping (SLAM) is vital for autonomous robot navigation. The robot must build a map of its environment while tracking its own motion through that map. Although many solutions to this intricate problem have been proposed, one of the most prominent issues that still needs to be resolved is to accurately measure and track landmarks over time. In this paper we explain how stereo vision can be used for this purpose. We employ SIFT for the detection and tracking of image features, and triangulate matching features with stereo geometry, to obtain our set of landmark locations. We derive and linearise a measure of uncertainty in each of these locations, for use in our adaptation of the extended Kalman filter (EKF) approach for performing SLAM. Experimental results indicate that stereo vision is a viable option to be used as a sensor in a SLAM system, and worthy of further development.**

I. Introduction

In order to develop a truly autonomous robot we believe it is vital to first implement simultaneous localisation and mapping (SLAM). SLAM is a technique used by mobile robots to build a map of an unknown environment while simultaneously tracking its own motion. This presents a chicken-and-egg situation: an accurate map is necessary for localisation, and accurate localisation is necessary to build a map. The inter-dependency between the estimates of the robot location and the map of the environment makes SLAM an interesting and difficult research problem.

Most SLAM systems build a probabilistic map by filling it with landmarks as they are observed by the robot's sensors. In our context landmarks will be 2D points in an otherwise sparse map. The robot can then estimate its own movement as it observes landmarks over multiple time steps, as well as improve its belief in the location of said landmarks. There are many ways to approach the problem, mostly based on the extended Kalman filter (EKF) or the particle filter [1][2].

Although SLAM is considered to be solved at a theoretical and conceptual level, successful implementation has given rise to some issues that still need to be resolved, the most prominent of these being sensor related. If landmarks cannot be accurately identified and tracked over time, SLAM will be practically impossible.

Vision systems have increased in popularity as a sensor for mobile robotics in recent years. Cameras are not only generally much cheaper than alternative sensors such as laser range finders and radar systems, but they also contain more information per sample. It may, however, be difficult to extract the information and convert it into a usable form.

There has been some success using stereo vision for SLAM, most notably by Grisetti et al. [3] who use a particle filter based approach. For the sake of computational efficiency we decide to follow an EKF-based approach. The EKF executes SLAM by including the locations of landmarks in the state vector. Motion and measurement equations are linearised, and motion and measurement noise are approximated with Gaussians, enabling use of the normal Kalman filter equations on a non-linear system.

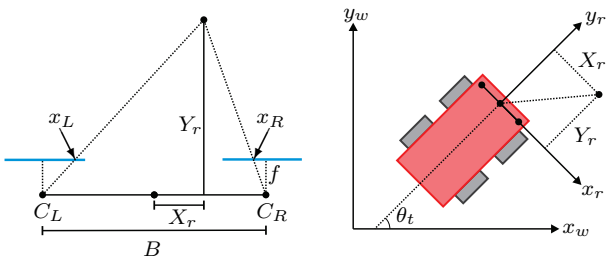
In this paper we describe our efforts to investigate the use of stereo vision as an effective sensor for EKF SLAM. We first discuss how we find landmarks in images taken by a stereo vision camera pair, and then explain how we track landmarks over time and how the map is managed. We provide the equations to transform feature data from the images to 2D landmarks and explain how we determine the noise associated with each measurement of a landmark. A brief overview of the motion model of our robot with a measure of uncertainty is given. We explain how the EKF is used for SLAM with the derivation of some of the equations. We then showcase some results from our implementation applied to recorded datasets.

II. Image features and stereo geometry

In order to execute SLAM we need to be able to track landmarks over time. We opt for a point landmark (or feature based) approach, where the map is a collection of 2D landmark locations. Although image features can be triangulated to 3D we project them onto the 2D plane in which the robot moves, for simplicity and speed of execution.

A. Feature detection and matching

There are many algorithms for identifying and matching points of interest in images. We investigate what might be considered the two most well-known algorithms: the scale-invariant feature transform (SIFT) [4] and speeded-up robust features (SURF) [5]. These algorithms perform the same task: find points in an image that have a unique descriptor in spite of scale, rotational and moderate affine transformation. The main difference between the two algorithms is that SURF sacrifices accuracy for speed, while SIFT is slow but typically much more accurate. In the case of SIFT, each descriptor is a vector containing 128 floating-point values, while the descriptors for



(a) camera geometry (b) robot geometry

Fig. 1. The geometry of our system.

SURF each contains only 64. Once we have points from two images, a nearest neighbour search can be performed on the descriptors to find matches.

For every new synchronised stereo image pair, we follow this detection and matching procedure to obtain a measurement of the features as a set of pairs of image coordinates. In order to use the EKF we model each pair as a measurement with Gaussian noise:

$$x = \begin{bmatrix} x_L \\ x_R \end{bmatrix} + \mathcal{N}(0, N_t), \tag{1}$$

where x_L and x_R are the horizontal image coordinates of the feature in the left and right images. By $\mathcal{N}(0, N_t)$ we mean a sample drawn from the normal distribution with mean 0 and covariance matrix N_t (the same notation is used throughout the rest of this paper). The vertical coordinates of the features are not used when calculating the 2D location of the landmark and are therefore omitted from the measurement. The noise covariance in Equation 1 is described by

$$N_t = \begin{bmatrix} \sigma_{x_L}^2 & 0 \\ 0 & \sigma_{x_R}^2 \end{bmatrix}, \tag{2}$$

with σ_{x_L} and σ_{x_R} the standard deviation in pixels of the match measurement.

We can then match the descriptors of a new measurement to the descriptors of the features in the map. We store both left and right image coordinates and descriptors in the map in order to match the descriptors from the new left image with the descriptors from previous left images, and similarly for the right images. Only the features that match consistently for left images and right images are used. Due to the increasing computational complexity of the EKF with an increasing number of features, we remove all the old features from the map that have not been found again at the current time step. All the new features that have not yet been observed are subsequently added to the map. If necessary, the removed features can be stored in a secondary map which is no longer updated by the EKF.

To remove faulty matches we use RANSAC [6] to find a fundamental matrix [7] that fits the largest subset of features from the current and previous left images. All the features that do not fit this fundamental matrix within a certain threshold are marked as matching errors and removed from the measurement. Depending on execution time constraints, this can also be done on the features from the right images.

B. Stereo geometry of calibrated images

Now that we have stereo image features that can be tracked over time, we convert them into 2D landmarks.

In order to calibrate the stereo camera pair we use the standard off-line calibration process included in the OpenCV library package [8]. We rectify a stereo image pair by projecting the epipoles to infinity so that all correspondences in these two images will have the same vertical coordinates. Some matching errors can also be removed by using this epipolar constraint.

Figure 1(a) depicts the stereo geometry of a pair of stereo cameras with camera centres at C_L and C_R, where the image planes have been rectified, and a landmark $\begin{bmatrix} X_r & Y_r \end{bmatrix}^T$ observed at x_L in the left image and x_R in the right image.

Using the geometry of the stereo camera pair, the landmark location in metres can be calculated in robot coordinates as

$$\begin{bmatrix} X_r \\ Y_r \end{bmatrix} = \begin{bmatrix} \frac{(x_L - p_x)B}{x_L - x_R} - \frac{B}{2} \\ \frac{fB}{x_L - x_R} \end{bmatrix} + \mathcal{N}(0, Q_t), \tag{3}$$

where B is the baseline, f the focal length and p_x the x-offset of the principal point, all obtained from the calibration process. Q_t is the covariance matrix of the measurement.

Note that we differentiate between robot coordinates (subscript r) and world coordinates (subscript w) as indicated in Figure 1(b). We calculate the world coordinates of the landmark as

$$\begin{bmatrix} X_w \\ Y_w \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \end{bmatrix} + \begin{bmatrix} \sqrt{X_r^2 + Y_r^2}\cos(\theta_t - \text{atan2}(X_r, Y_r)) \\ \sqrt{X_r^2 + Y_r^2}\sin(\theta_t - \text{atan2}(X_r, Y_r)) \end{bmatrix}, \tag{4}$$

where x_t, y_t and θ_t are the robot's position and orientation in world coordinates.

All that remains here is to find the covariance Q_t. We know that a transformation from N_t to Q_t is possible if we have a linear system and, since Equation 3 is not linear, we use a first order Taylor approximation to find the transformation matrix

$$W_t = \begin{bmatrix} \frac{\partial X}{\partial x_L} & \frac{\partial X}{\partial x_R} \\ \frac{\partial Y}{\partial x_L} & \frac{\partial Y}{\partial x_R} \end{bmatrix} = \begin{bmatrix} \frac{-B(x_R - p_x)}{(x_L - x_R)^2} & \frac{B(x_L - p_x)}{(x_L - x_R)^2} \\ \frac{-fB}{(x_L - x_R)^2} & \frac{fB}{(x_L - x_R)^2} \end{bmatrix}. \tag{5}$$

It then follows that

$$Q_t = W_t N_t W_t^T. \tag{6}$$

This approximation is performed in order to keep the noise model Gaussian, which is necessary for the EKF.

We can now include every measured feature pair i into the measurement z_t as a 2D landmark,

$$z_t^i = \begin{bmatrix} X_r^i \\ Y_r^i \end{bmatrix}, \tag{7}$$

with a corresponding noise covariance Q_t^i. Finally, for every measured landmark i that is found to correspond with a landmark j in the map, we define a correspondence $c_t^i = j$. All of these correspondences are stored in the vector c_t.

III. Motion model

We proceed to model the movement of the robot and the noise associated with it.

The velocity motion model as described by Thrun et al. [2] is used. We assume that the robot is controlled by two velocities: a translational velocity v_t and a rotational velocity ω_t. Hence at time t we have the control input

$$u_t = \begin{bmatrix} v_t \\ \omega_t \end{bmatrix} + \mathcal{N}(0, M_t), \tag{8}$$

with a Gaussian noise covariance of

$$M_t = \begin{bmatrix} \alpha_1 v_t^2 + \alpha_2 \omega_t^2 & 0 \\ 0 & \alpha_3 v_t^2 + \alpha_4 \omega_t^2 \end{bmatrix}. \tag{9}$$

The error parameters α_1 to α_4 are robot specific and have to be obtained through testing.

With this control input and the location of the robot at the previous time step we can estimate the robot's current location according to

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{bmatrix} + \begin{bmatrix} v_t T \sin(\omega_t T + \theta_{t-1}) \\ v_t T \cos(\omega_t T + \theta_{t-1}) \\ \omega_t T \end{bmatrix} + \mathcal{N}(0, R_t), \tag{10}$$

where T is the sample period. The motion noise covariance, R_t, has to be transformed in the same fashion as the measurement noise. Once again we use a first order Taylor approximation to obtain the transformation matrix as

$$\begin{aligned} V_t &= \begin{bmatrix} \frac{\partial x_t}{\partial v_t} & \frac{\partial x_t}{\partial \omega_t} \\ \frac{\partial y_t}{\partial v_t} & \frac{\partial y_t}{\partial \omega_t} \\ \frac{\partial \theta_t}{\partial v_t} & \frac{\partial \theta_t}{\partial \omega_t} \end{bmatrix} \\ &= \begin{bmatrix} T\cos(\theta_{t-1} + T\omega_t) & -T^2 v_t \sin(\theta_{t-1} + T\omega_t) \\ T\sin(\theta_{t-1} + T\omega_t) & T^2 v_t \cos(\theta_{t-1} + T\omega_t) \\ 0 & T \end{bmatrix}. \end{aligned} \tag{11}$$

The covariance of the noise associated with motion is then simply

$$R_t = V_t M_t V_t^T. \tag{12}$$

With this motion model and uncertainty measure we have all that is needed to continue to state estimation.

IV. SLAM with the extended Kalman filter

The EKF is a method to implement state estimation on non-linear systems by assuming belief distributions are Gaussian and linearising around the current mean and covariance. We base our approach to the EKF SLAM algorithm on the method explained by Thrun et al. [2]. Our approach is similar in concept but there are some key differences. Firstly, we change the way in which the measurements are handled. We found that it is easier to work in Cartesian coordinates when working with images features, while Thrun et al. [2] use polar coordinates. Secondly, we consider each individual landmark measurement to have its own noise covariance. Our motivation for doing so stems from the fact that, in stereo vision, triangulation uncertainty increases rather dramatically with distance. The third important difference is that we do not include the descriptors of the landmarks in the state vector. This is done to achieve faster execution time. Moreover, we believe that the inclusion of uncertainty in feature descriptors at this stage is slightly contrived.

Our EKF SLAM algorithm follows on the next page, with all the inputs needed to update the mean μ_t and covariance Σ_t of the location and orientation of the robot and the locations of the current landmarks at time step t.

It is important to note how we include landmarks in the state of our system. In Gaussian distributions, the mean is also the most likely point. The state of the system is thus the mean vector μ_t produced by the EKF. This vector is composed of the robot location and orientation and the locations of current landmarks in world coordinates. Hence we have

$$\mu_t = \begin{bmatrix} \mu_{t,x} & \mu_{t,y} & \mu_{t,\theta} & \mu_{1,x} & \mu_{1,y} & \cdots & \mu_{N,x} & \mu_{N,y} \end{bmatrix}^T, \tag{13}$$

where N is the number of current landmarks. The covariance matrix Σ_t also uses this ordering.

The EKF SLAM algorithm can be divided into two parts: a control update (lines 1 to 5 in the algorithm) and a measurement update (lines 6 to 21).

A. Control update

The first part of the algorithm updates the mean and covariance of the location and orientation of the robot using only the control u_t, and in doing so finds an estimate of the robot's new location.

The matrix F is used as a shaping matrix throughout the algorithm. Line 3 is the update of the location using Equation 10. To incorporate the uncertainty associated with the motion model, we use the derivative of Equation 10 evaluated at u_t and μ_{t-1} with respect to $\begin{bmatrix} x_{t-1} & y_{t-1} & \theta_{t-1} \end{bmatrix}^T$. The resulting Jacobian is shown in line 4. We calculate the new uncertainty with the Kalman filter equation in line 5 using the motion uncertainty from Equation 12. At this point we expect the uncertainty of the location of the robot to become much larger.

B. Measurement update

The second part of the EKF uses the measurement, z_t, and the values obtained during the control update to update the mean and covariance of previously observed landmarks in the map as well as the mean and covariance of the robot's location and orientation.

At line 7 the algorithm enters a loop that iterates over all the measured landmarks. If a landmark has not been observed before, we add its location, using Equation 4, to the mean vector. The uncertainty of a new landmark is set to infinity. Next we calculate δ, q and ϕ, purely to simplify further expressions. In line 15 we transform the mean of the landmark back to robot coordinates in order to be able to find the state prediction error used in line 19.

H_t^i is the Jacobian of the expression in Equation 4 with respect to $\bar{\mu}_t$, used to linearise the transformation of the covariance Σ_t. Note that in line 17 we have $s_\phi = \sin\phi$ and

Algorithm: 2D_EKF_SLAM $(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, c_t)$

1: $\theta = \mu_{t-1,\theta}$

2: $F_x = \begin{bmatrix} 1 & 0 & 0 & \underbrace{0 \cdots 0}_{2N} \\ 0 & 1 & 0 & \\ 0 & 0 & 1 & \end{bmatrix}$

3: $\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{bmatrix} v_t T \cos(\theta + \omega_t T) \\ v_t T \sin(\theta + \omega_t T) \\ \omega_t T \end{bmatrix}$

4: $G_t = I + F_x^T \begin{bmatrix} 0 & 0 & v_t T \sin(\theta + \omega_t T) \\ 0 & 0 & -v_t T \cos(\theta + \omega_t T) \\ 0 & 0 & 0 \end{bmatrix} F_x$

5: $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + F_x^T R_t F_x$

6: $j = c_t^i$

7: **for** all observed features $z_t^i = \begin{bmatrix} X_r & Y_r \end{bmatrix}^T$ **do**

8: **if** feature j never seen before **then**

9: $\phi = \bar{\mu}_{t,\theta} - \text{atan2}(X_r, Y_r)$

10: $\begin{bmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \end{bmatrix} = \begin{bmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{bmatrix} + \begin{bmatrix} \sqrt{X_r^2 + Y_r^2} \cos(\phi) \\ \sqrt{X_r^2 + Y_r^2} \sin(\phi) \end{bmatrix}$

11: **end if**

12: $\delta = \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} = \begin{bmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{bmatrix}$

13: $q = \delta^T \delta$

14: $\phi = \bar{\mu}_{t,\theta} - \text{atan2}(\delta_y, \delta_x)$

15: $\hat{z}_t^i = \begin{bmatrix} \sqrt{q} \sin(\phi) \\ \sqrt{q} \cos(\phi) \end{bmatrix}$

16: $F_{x,j} = \begin{bmatrix} 1 & 0 & 0 & \underbrace{0 \cdots 0}_{2j-2} & 0 & 0 & \underbrace{0 \cdots 0}_{2N-2j} \\ 0 & 1 & 0 & & 0 & 0 & \\ 0 & 0 & 1 & & 0 & 0 & \\ 0 & 0 & 0 & & 1 & 0 & \\ 0 & 0 & 0 & & 0 & 1 & \end{bmatrix}$

17: $H_t^i = \frac{1}{\sqrt{q}} \begin{bmatrix} -\delta_x s_\phi - \delta_y c_\phi & \delta_y s_\phi - \delta_x c_\phi \\ -\delta_y s_\phi + \delta_x c_\phi & -\delta_x s_\phi - \delta_y c_\phi \\ q c_\phi & -q s_\phi \\ \delta_x s_\phi + \delta_y c_\phi & -\delta_y s_\phi + \delta_x c_\phi \\ \delta_y s_\phi - \delta_x c_\phi & \delta_x s_\phi + \delta_y c_\phi \end{bmatrix}^T F_{x,j}$

18: $K_t^i = \bar{\Sigma}_t H_t^{iT} (H_t^i \bar{\Sigma}_t H_t^{iT} + Q_t^i)^{-1}$

19: $\bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^i)$

20: $\bar{\Sigma}_t = (I - K_t^i H_t^i) \bar{\Sigma}_t$

21: **end for**

22: $\mu_t = \bar{\mu}_t$

23: $\Sigma_t = \bar{\Sigma}_t$

24: **return** μ_t, Σ_t

$c_\phi = \cos\phi$. From the shaping matrix F, it is clear that each iteration of the loop has an effect on the robot location and the location of the landmark in question. Finally, in lines 18 to 20, we use the Kalman filter equations to compute a new mean vector and corresponding covariance matrix.

Once all the measurements have been included in the estimator, we expect the uncertainty of the observed landmarks and the uncertainty of the location and orientation of the robot to decrease.

V. Experimental results

In order to test our proposed EKF SLAM algorithm we captured datasets with two Point Grey Firefly MV cameras mounted on a Pioneer 3-AT from Mobile Robots, as seen in Figure 2. The cameras were synchronised to capture images at 2 Hz. The robot was controlled by human input and the control data was stored for the control update phase of the algorithm. Both indoor and outdoor datasets were captured. We implemented the algorithm in MATLAB and C++.

After some experimentation we found that the matching of image features had to be performed very carefully, because the EKF tends to be very sensitive to matching errors. As an example, Figure 3 depicts localisation errors resulting from only a few mismatched landmarks. We observed that these errors can become quite large, and this has some negative impacts on the performance of our implementation. Firstly, we found that the more accurate SIFT feature detector and matcher outperformed SURF quite significantly but, since SIFT executes much slower, it may prohibit real time implementation. Secondly, we had to impose strict thresholds on the epipolar constraint as well as in the RANSAC estimator, which may increase the likelihood of erroneously discarding correct matches. This in turn affects the completeness (or density) of our map.

Another problem was encountered when the robot would turn too quickly, causing consecutive images to change too drastically for enough features to be tracked over that time period. In such a case the EKF will essentially use the control update only, and the uncertainty of the robot's location and orientation will increase substantially. Unfortunately, the only way around this problem is either to turn the robot more slowly or to capture images at a faster sampling rate.

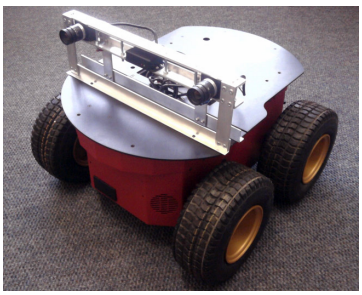


Fig. 2. Our Pioneer 3-AT robot, with two Point Grey Firefly MV cameras in a stereo configuration and an external synchronisation unit, used for capturing datasets.

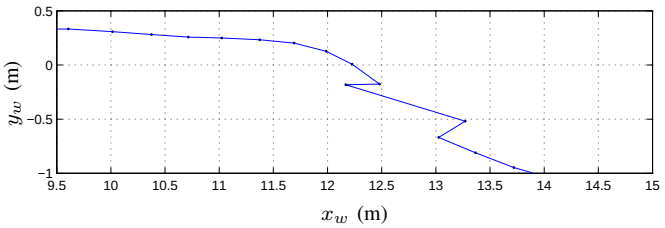
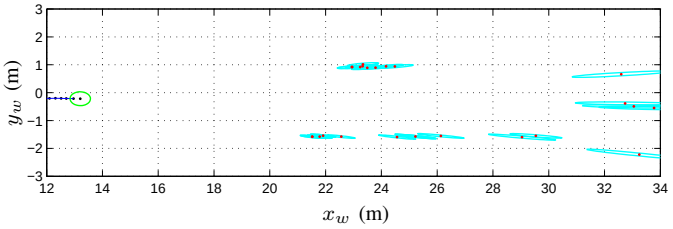
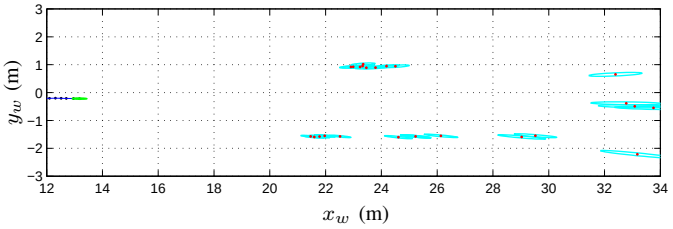


Fig. 3. Erratic behaviour in the robot's estimated location due to errors in the matching of image features.

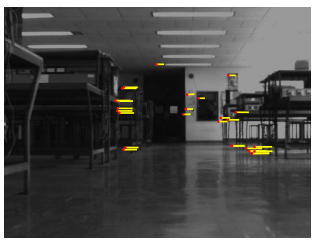
We first discuss results from testing our implementation on the indoor dataset. Figure 4 shows what happens at a typical time step. First the uncertainty of the robot's state increases as the control update is executed. When the measurements are then included the uncertainty of the location and orientation of the robot, as well as the uncertainties in the positions of the landmarks in the map, decrease. The positions of the robot and landmarks also shift as a result of the EKF algorithm updating the state mean values. This behaviour is exactly in line with what is to be expected.



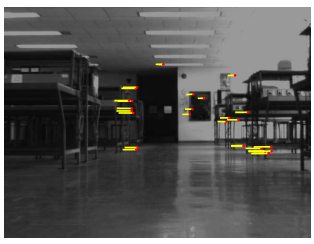
(a) estimated robot and landmark positions after the control update



(b) estimated robot and landmark positions after the measurement update



(c) left image



(d) right image

Fig. 4. A depiction of typical control and measurement updates. Confidence ellipses show the standard deviation from the mean of the positions of the landmarks and the robot. The left and right images show the location of features used (red dots) with the feature match disparities (yellow lines).

Figure 5 shows all the landmarks with their final confidence ellipses, the route according to the EKF SLAM algorithm, and the route obtained from using only the control data with Equation 10, for the indoor dataset. Some example images are also shown. We observe that for this dataset the route calculated by the EKF travels neatly between the two rows of benches, where most of the features were found.

Figure 5 also shows results from our algorithm applied to the outdoor dataset which contains 340 image pairs captured over a distance of about 75 m. Because we do not yet have a method to test accuracy, for this set and the previous one, it is difficult to tell exactly how well the EKF performs. We did, however, end the outdoor run more-or-less where we had begun it. From the figure we see that the drift introduced by the EKF SLAM is less than 2 m over this 75 m run, while the drift on the route calculated with only the control data is considerably more.

VI. Conclusions

We presented a method for implementing a practical SLAM system that uses stereo vision as a sensor. The method estimates the position and orientation of a robot simultaneously with the positions of landmarks observed by two synchronized cameras on the robot. These landmarks are identified and tracked with SIFT.

We found that in spite of a few problems — most notably the extreme sensitivity to mismatched features — our method performed well. We showed that great improvement in location estimates is possible by the inclusion of the stereo vision sensor, when compared to using only the robot's control data in an open-loop fashion. We therefore have a strong belief that stereo vision is a viable option to be used in a SLAM system, and definitely validates further research.

In future we would like to also implement a particle filter based approach to SLAM, which might be less vulnerable to matching errors. Ongoing work also includes the incorporation of differential GPS data, mainly for use as ground truth in evaluating the accuracy of our system.

References

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2006.

[3] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[4] D. Lowe, "Object recognition from local scale invariant features," *IEEE International Conference on Computer Vision*, pp. 1150–1157, 1999.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[6] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[7] R. Hartley and A. Zisserman, *Mutiple View Geometry*, 2nd ed. Cambridge University Press, 2003.

[8] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, Inc., 2008.

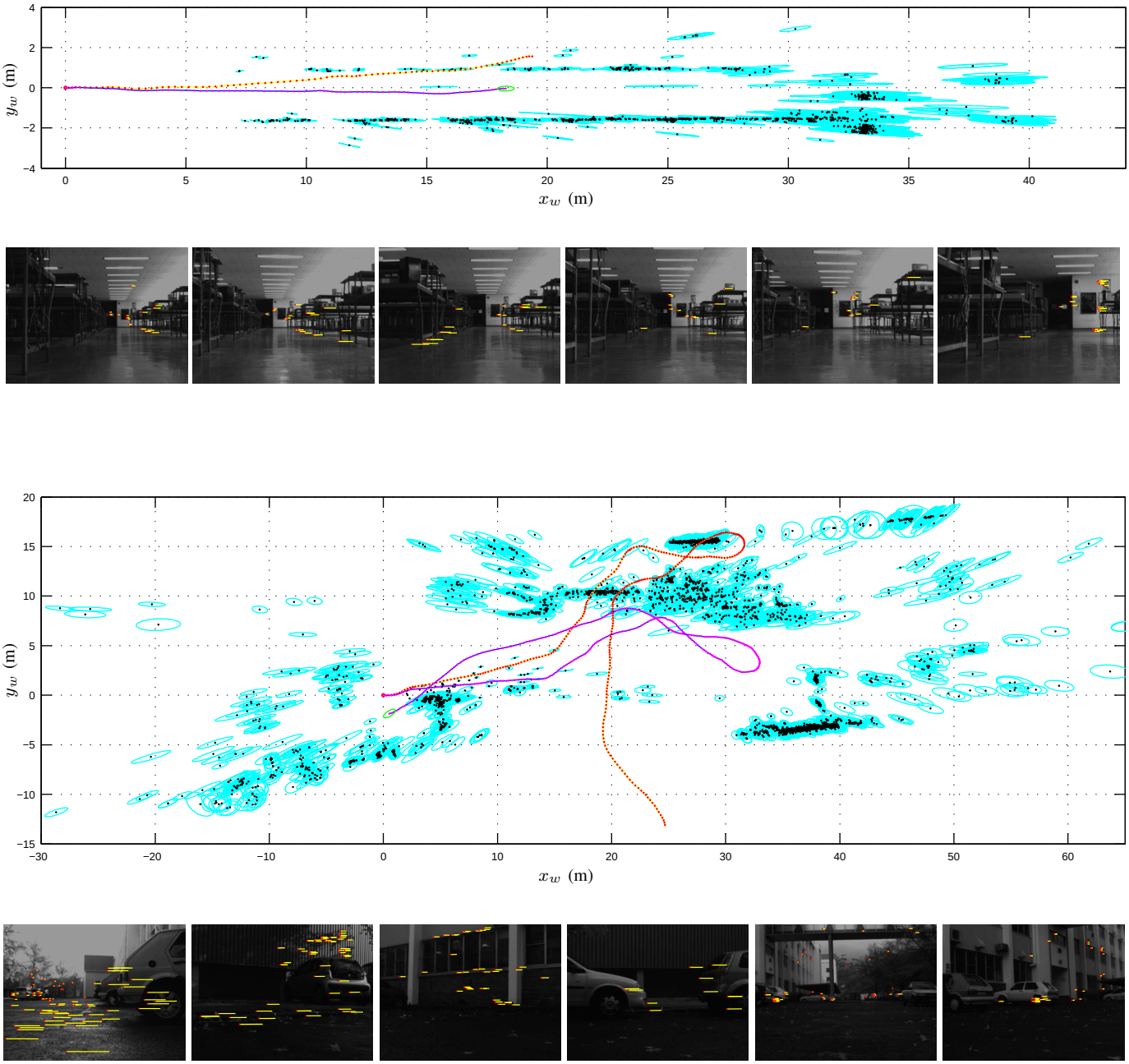


Fig. 5. Complete maps of the indoor (top) and outdoor (bottom) datasets, in metres, with typical images captured by the left camera and features used as landmarks. The landmarks on the maps are shown with corresponding confidence ellipses, while the features on the images are shown with their disparities. The routes shown in red were calculated by simply using Equation 10 (thus ignoring measurements from the stereo vision sensor), while the routes in magenta were produced by our complete EKF SLAM implementation.