EURASIP Journal on Embedded Systems

RESEARCH Open Access

An intelligible implementation of FastSLAM2.0 on a low-power embedded architecture

Albert A. Jiménez Serrata[1], Shufan Yang[1,2*] and Renfa Li[2]

Abstract

The simultaneous localisation and mapping (SLAM) algorithm has drawn increasing interests in autonomous robotic systems. However, SLAM has not been widely explored in embedded system design spaces yet due to the limitation of processing recourses in embedded systems. Especially when landmarks are not identifiable, the amount of computer processing will dramatically increase due to unknown data association. In this work, we propose an intelligible SLAM solution for an embedded processing platform to reduce computer processing time using a low-variance resampling technique. Our prototype includes a low-cost pixy camera, a Robot kit with L298N motor board and Raspberry Pi V2.0. Our prototype is able to recognise artificial landmarks in a real environment with an average 75% of identified landmarks in corner detection and corridor detection with only average 1.14 W.

Keywords: Simultaneous localisation and mapping, Robotics, Embedded systems, Pixy camera

1 Introduction

The simultaneous localisation and mapping (SLAM) algorithm has been developed to meet the challenge of building and updating a map of the surroundings in moving mobile robots, which includes mapping robots poses without prior knowledge of their surroundings. It has the advantage of providing knowledge about the robots' own pose and representation of the environment using extensive computational resources to map robots' surroundings and simultaneously perform localisation [4, 5]. To enable a robot to navigate through an environment autonomously, an estimation of its position within a reference system or map could be obtained from sensors, such as dead-reckoning sensors, laser sensors, radar sensor or 3D camera sensors [6], GPS (Global Positioning System) [1, 3, 10, 19, 30]. However, those sensors are either expensive or unsuitable for small robots because of size, weight or power efficiency. For example, a GPS can be used to provide the solution for the global localisation. However, at some places, GPS is not possible to use it, such as in caves, underwater or places where no signal can be obtained [1].

Currently, researchers have moved into alternative affordable solutions, such as ultrasonic sensors or infrared sensors. However, those types of sensors do not provide accurate odometry information due to the relatively low signal-to-noise ratio in low-cost sensors compare with expensive sensors [17]. Furthermore, most of the low-cost sensors that were used to estimate the current position of the mobile robot tend to accumulate errors over time (known as statistically dependent) due to the noise generated intrinsically in the sensors. Recently, the SLAM community moves towards image sensors (refer to Visual SLAM), as they are affordable and able to provide a large amount of information compare with other sensors such as laser range finders. In addition, it can be applied to mobile robots that have smaller sizes, lower weight and less power consumption [8, 24] than industrial robots. At the same time, it presents great challenges due to the computational resource needed for the identification of landmarks in variable environments. Hence, SLAM presents an emerging challenge to be implemented with low-cost embedded systems, which are the most common platform for pervasive computing products.

* Correspondence: s.yang@wlv.ac.uk
[1]School of Mathematics and Computer Science, Faculty of Science and Engineering, University of Wolverhampton, City Campus, Wolverhampton, UK
[2]College of Computer Science and Electronic Engineering, Hunan University, Changsha, China

In this work, we combined visual SLAM and visual odometry algorithm on a low-cost embedded platform to demonstrate how a mobile robot navigating using artificial landmarks in an indoor environment. Our proposed method can lead to fast landmark detection with a reduced computational complexity using a low-variance resampling, which result that the sensor data measurement is kept outside of the main process loop during estimating the locations of the landmark.

2 Related work

In order to implement a cheap and small SLAM system, monocular visual SLAM solutions were invented to use only single image sensor [24]. However, the algorithms needed for monocular SLAM are much more complex since depth information cannot be directly inferred from a single frame captured from a single camera. Normally, each frame needs to be processed, even when no relevant information is provided by new frames. Furthermore, these solutions also present an accumulation error over time due to nonlinear Kalman filter related models.

The best-known implementation of monocular SLAM using a keyframe-based approach is PTAM [14]. The keyframe-based technique is used to select some of the frames with predefined features to save computational power. In PTAM system, a parallel processing with high-performance computing system was used for obtaining robot's pose and the location mapping, which is unsuitable for low-cost and low-power embedded systems due to power consumption issues.

Other implementations of monocular SLAM are also based on obtaining abstract regions of an image in each frame that are more useful for reconstruction of location than robots navigation [7, 21, 27]. Some of the algorithms for the detection of regions are SIFT, BRIEF, FREAK, SURF and ORB [11]. However, all these feature extraction approaches present the problem of data association in SLAM [21].

Data association problem is how to decide which noisy measurement corresponds to which feature of the map. Noise and partial observability can make the relationship between measurements and the model highly ambiguous. This problem is further complicated by considering the possible existence of previously unknown features in the map and the possibility of spurious measurements. For example, multiple measurements over time are obtained by a sensor, it is needed to associate which measurement belongs to a specific landmark. The methods for the solution of data association are based on statistical procedures such as Nearest Neighbor (NN) [2] or Joint Compatibility Branch and Bound (JCBB) [22, 25]. One way to solve associate measurements with low computational complexity is using artificial landmarks that consist the modification of environment features [26]. Several other studies tackle the problem by minimising the ambiguity of the measurements in a mapped environment [18, 25, 29, 32]. However, in these solutions, uncertainty in the measurements is taken before classifying a landmark, which results in the incorporation on specific characteristics of the sensor implemented.

Other studies present the imposition of physical artificial landmarks in an environment, such as QR codes or fiducial markers, where uncertainties of both measurements and data association reduced [15, 23, 31, 32]. Llofriu et al. demonstrated an embedded solution of SLAM using artificial landmarks [16], such as boxes with different colours. However, due to slow measurements presented by the image sensor, it might not have enough updates of the particle, adding uncertainty in the location. Another drawback is that the identification of artificial landmarks used in this study is based on a measurement likelihood that aggregates to the possibility of introducing wrong data association.

3 Methods and materials

Based on FastSLAM2.0 method, we develop a low-variance resampling method with a multimodal design style based on embedded system platforms [20]. The multimodal FastSLAM framework has been implemented in python, and its source code is released for public use in an online repository (see details in Appendix 3).

3.1 Adapted FastSLAM algorithm

FastSLAM2.0 used a particle filter to model the uncertainty of the robot pose [20]. It defines the posterior SLAM as the product between the posterior of the robot pose, and the posterior of the landmarks is conditioned by the robot path, presented in Fig. 1. The posterior is determined by a Dynamic Bayesian Networks (Eq. 1) and a Rao-Blackwell particle filter [9, 12, 13]. The posterior of robot path is obtained by the control vector (u_t), the data association (n_t) and landmarks (z_t) [20].

$$p(x_t, \theta_t | n_t, u_t, z_t) = p(x_t | n_t, u_t, z_t) p(\theta_t | n_t, u_t, z_t, x_t) \tag{1}$$

As we discussed in the last section about the data association issues in the SLAM algorithm, we adapted a simple data association approach into three main SLAM steps: the prediction step for calculating the current state of particles in motion model; the partial diversity updating step for recalculating the possibility of particles; the resampling step for deleting probable trajectories.

In prediction step, we replaced the probability of particle with a given id. The associations between particles and ids are made during the landmark detection. Then each particle stores its own belief of the landmarks, represented by a Gaussian distribution, and the distribution is updated using a Kalman filter on each particle independently. As

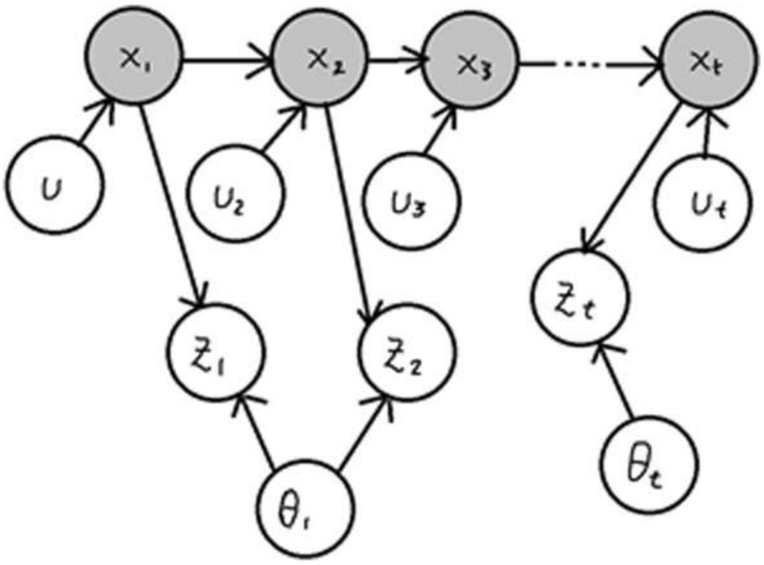
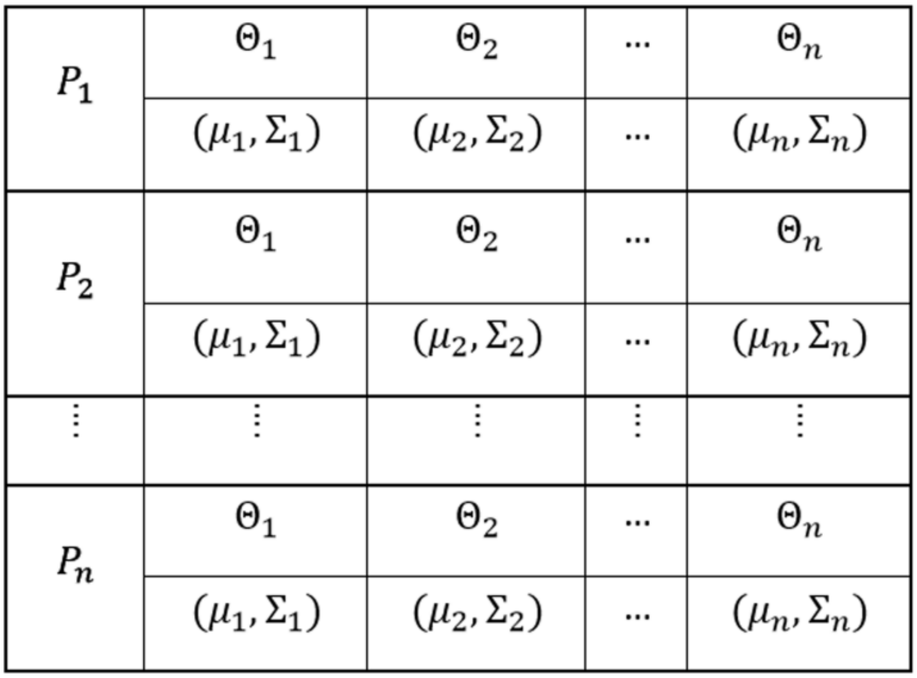


Fig. 1 Dynamic Bayesian Network for SLAM

illustrated in Fig. 2, each particle information is represented with each landmark by Θ_n with distribution (μ, Σ), characterised by a mean (μ_n) and covariance (Σ_n).

The new sample of the robot path is represented by a proposal distribution given the visual measurements and control vectors. Each particle obtained a new sample is based on a prediction using a motion model, which is described in Eq. (2). The control vector is composed of the linear velocity v and the angular velocity w. The pose of the robot is represented by the components of the position in x, y and the orientation (θ) of the robot after a movement.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v \sin\theta \\ v \cos\theta \\ \omega \end{bmatrix} \tag{2}$$



P_1	Θ_1	Θ_2	…	Θ_n
	(μ_1, Σ_1)	(μ_2, Σ_2)	…	(μ_n, Σ_n)
P_2	Θ_1	Θ_2	…	Θ_n
	(μ_1, Σ_1)	(μ_2, Σ_2)	…	(μ_n, Σ_n)
⋮	⋮	⋮	⋮	⋮
P_n	Θ_1	Θ_2	…	Θ_n
	(μ_1, Σ_1)	(μ_2, Σ_2)	…	(μ_n, Σ_n)

Fig. 2 Information of landmarks contains in particles

After obtaining the new sample of the position and a measurement, each particle must update their beliefs about the landmarks. This is performed using the observation model, defined in Eq. (3). In addition, the control vector frequency is higher than the frequency of measurement in order to increase the diversity of the particles.

$$\Theta = \begin{pmatrix} r\cos\varphi \\ r\sin\varphi \end{pmatrix} \tag{3}$$

In this work, we used an inverse model to estimate the Jacobian on the Kalman filter of each particle. The inverse matrix model is presented in Eq. (4). The variable $\boldsymbol{V}$ represents a vector to a landmark using the measured distance, represented by the variable r.

$$\boldsymbol{G} = \begin{pmatrix} \frac{Vx}{r} & \frac{Vy}{r} \\ \frac{-Vy}{r^2} & \frac{Vx}{r^2} \end{pmatrix}, \boldsymbol{V} = \begin{pmatrix} \boldsymbol{\Theta}_x - x \\ \boldsymbol{\Theta}_y - y \end{pmatrix} \tag{4}$$

In the resampling step, original FastSLAM2.0 is based on a Rao-Blackwell factorisation; therefore, it is necessary to do a resample of the particles in order to "keep alive" the particle with higher probabilities. To reduce computational costs, we adopted a low-variance resample method [28]. Based on the low-variance resample, we select the most likely particle after applying a minimisation of the sum of the weights of the particles with respect to a standard uniform distribution. The adapted SLAM process using the low-variance resample method can be found in Fig. 3.

3.2 Landmark detection

In order to represent the landmarks into a coordinate plane in the observation model, we just measure the distance and the angle from robot to landmark, represented by the variable r (distance) and φ (angle). In the monocular visual sensor scenario, we obtained a relative distance from the camera to an object by knowing the dimensions of the object and calculate the distance using the epipolar geometry.

In order to calculate the distance, we formulated a relationship between the focal length (f) and the distance to a landmark (r). As illustrated in Fig. 4, the height of the landmark is represented by h_r; the height of the landmark in the frame of the camera is represented by h_s. The relationship between the heights is shown in Eq. (5). This equation describes the relationship between the height of the object and the height of the sensor, using the total height of the image frame (I_h), and the height of the sensor (S_h), being these proportional to the relationship between the focal length and the desired distance.

```
Algorithm SLAM($u_t, x_{t-1}, z_t, ID, landmarks, particle\_set$):

    If $ID$ not in $landmarks$:
        Add $ID$ to $landmarks$            // add $ID$ to list of seen landmarks
     $Zt = < z_t, landmarks$.index($ID$) $>$   // get index of landmark, data association
    for $p = 1$ to length($particle\_set$) do:
        Sample($u_t, x_{t-1}$)             // Get new $x_t$ from motion model
    end
    for $z = 1$ to length($Zt$) do:
        $\epsilon$ = add_update($Zt, particle\_set$)  // Add or update feature to map, return state
        compute_weights($particle\_set, \epsilon$)   // assign sampling weight according to state
    end
    $particle\_set = \eta(particle\_set)$        //Normalize particle weights
    If $Zt$:
        low_variance_resample($particle\_set$) // Create a new particle set from resample
    return $particle\_set$
```

Fig. 3 Pseudocode for our adapted SLAM Algorithm with low-variance resample

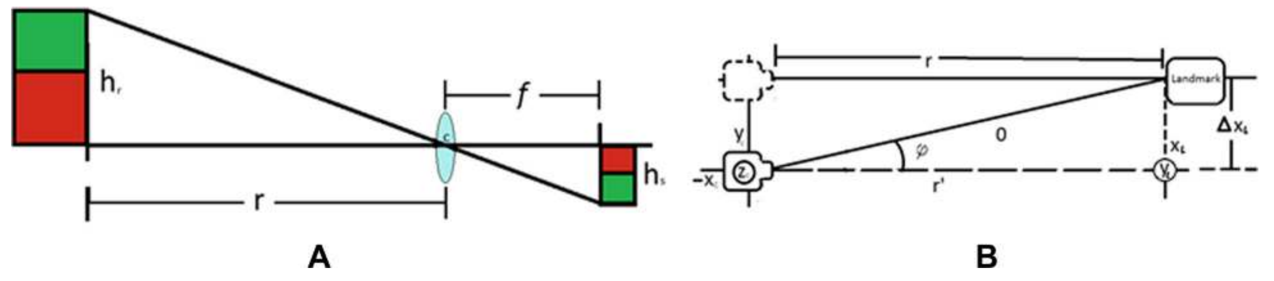


Fig. 4 Landmark detection. **a** Relation of distance towards a landmark. **b** Relation of change in position of a camera

$$r = f\frac{h_r}{S_h}\frac{I_h}{h_s} \tag{5}$$

Regarding the angle calculation, we assume that the landmark is at the front of the camera, projecting a horizontal line that forms an angle of π/2 with the object surface, as illustrated in Fig. 4b. The camera is represented by a three-dimensional plane (x_c, y_c, z_c) while the plane of the image is represented by the two-dimensional coordinate system (x_l, y_l). In a similar way, when the landmark is out of the view field of the camera, an angle between the distance and a delta x in the plane of the image is estimated. As a result, due to the plane is represented as a two-dimensional space, the change in z_c of the camera plane is not relevant, representing a point in Z plane as a projection in the plane x_c, y_c. Hence, no matter the height in the Z position at which this object is detected, the projected point keeps on the angle described.

Moreover, the changing angle can be calculated with a geometry relationship between the delta x and the projection of the distance. Equation (6) is presented as calculating of the desired angle towards a landmark. This angle could also be calculated using vectors and finding the angle between the vectors. The w_s variable, represents the delta x, while the I_w variable is the value of the total width of the image frame. The S_w corresponds to the width in millimetres on the CMOS sensor in the pixy camera. The upper bound of ration is $\frac{h_s}{w_s} = 2.2$; the lower bound ratio is $\frac{h_s}{w_s} = 2.0$.

$$\varphi' = atan\left(\frac{w_S}{f}\frac{S_w}{I_w}\right), ws = \Delta x_l \tag{6}$$

3.3 Hardware implementation

Our proposed multi-module SLAM implementation for mobile robotic systems in embedded systems includes three main modules: central computing node, odometry node and image processing node for landmark detection, as shown in Fig. 5 (Appendix 2 and 4). Our prototype has installed two wheel encoders for the detection of movement, a pixy camera to detect the landmarks and a central unit, Raspberry Pi, for the execution of the SLAM solution. The detailed schematic design can be found in Appendix 1. The central computing node is used to calculate the current state of the particle filter and estimate the location of landmarks. The landmark node generates a message to represent the measurement of the distance, the direction of the distance and an ID that associate each landmark. As a consequence, this node has the role of detecting landmarks in the environment. The odometry node sends a message to the central node every 100 ms to update the position of the robot. This message contains the linear velocity (v) and angular velocity (ω) with 10 samples per second. Both the linear velocity and the angular velocity are calculated by the wheel encoders installed on the prototype. In addition, the central node can use that information to reproduce a graphical representation of locations.

The prototype of this work uses the robot kit module with an incremental encoder attached to each wheel. The robot kit is controlled by an Arduino Uno, which has an output of 5 V and can be powered directly by the robot kit and shared with the odometry node. The DC motors have a separate battery bank to power the HL29N driver controlled by the Arduino. Meanwhile, pixy camera, as one of open source and low-cost image sensor, is used for colour detection. Pixy camera represents the Landmark Node of the system, as it has its own memory and processor, which uses the raw image data for the detection of coloured blocks. The resolution of the images in the pixy camera is 320×200 (I_w, I_h). The artificial landmarks are detected using poles of different colour combinations as the work presented by [16].

4 Evaluation methodology

In order to obtain empirical data from the prototype, corner detection and corridor detection were performed in a real-life scenario. Each experiment was repeated three times, with the objective to assess the detection of landmarks and obtain a location of the prototype, while storing a map using our intelligible SLAM solution.

Due to the lack of rotational camera in our prototype, we are unable to carry out loop-closing experiments using our prototype. Instead, a loop-closing experiment using our FastSLAM simulator can be viewed using the

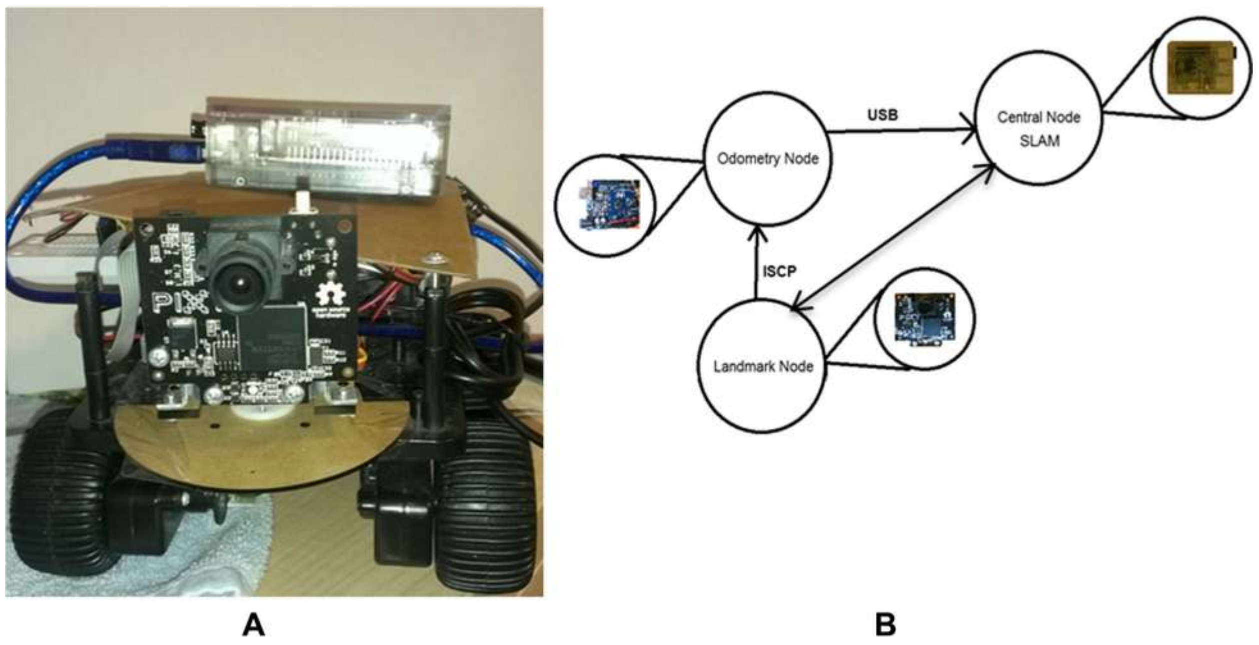


Fig. 5 (**a**) The project prototype using a pixy camera and a rasberry Pi 2 (connection can be found in Appendix). (**b**) The connections for modular design in the prototype

online link (https://youtu.be/cOghoHYNVuA). The corridor and corner detection experiments are used to demonstrate the robots' pose uncertainty and the ability to explore unknown terrain.

4.1 Corner detection

The first experiment involved the detection of a corner using six landmarks as shown in Fig. 6. As shown in Fig. 6a, we demonstrated the corner detection; the line in the figure represents the path performed by the mobile robot. Figure 6b shows the blue curve of the average trajectory of particles and the red curve of the covariance of landmarks detected. A pattern recognition was implemented to detect a combination of colours in a cylinder as a landmark. We choose a multiple colour floor in our experiments to produce noise readings for the pixy camera,

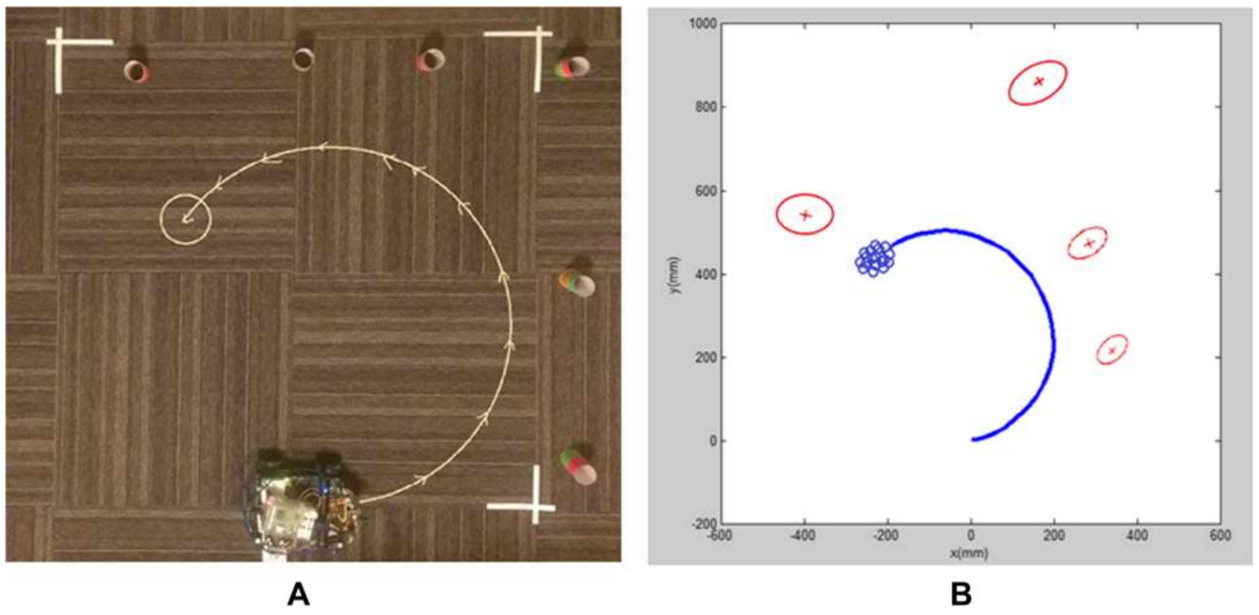


Fig. 6 Corner detection experiment results. **a** Semi-circular path implemented to map a corner. **b** Simulation results

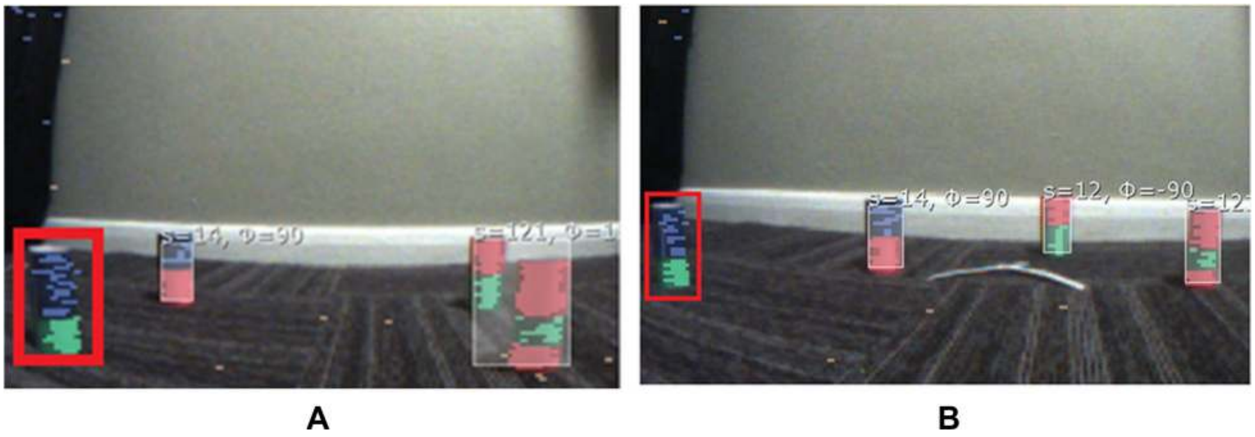


Fig. 7 Landmark detection. **a** Single detection of landmark. **b** Parallel detection of landmark

with the purpose of evaluating how well our data association scheme is performed. Landmarks are formed a semi-circular path with no loop closure. In this way, it produces an increase in the landmark's uncertainty. As a result, the noise produced by the floor plus the increment in uncertainty at the trajectory, it can be detected how well the SLAM solution discriminates the landmarks and associates the measurements. The average processing time for a single frame was 0.49 s. The landmarks are made of a cylindrical shape with a diameter of 40 ± 0.05 mm and a height of 80 ± 0.05 mm. In this first execution of experiment I, only 66.6% of the landmarks located in the test area were detected.

The performance of our prototype is limited by the resolution of pixy camera and the low precision of L298 motor driver. Two landmarks were not detected since the noise of the carpet had presented a greater influence by colour combination with blue colour. Figure 7 highlighted the undetected landmark mixed with blue colour and affected by the floor noise in a red rectangle. The detected landmarks are represented by a white rectangle with an associated ID, indicated by the variable 's' and an inclination angle represented by the variable 'Φ'.

4.2 Corridor detection

In the corridor detection experiment, we used the landmarks side by side in the trajectory of the prototype, simulating a corridor, which serves the purpose of revisiting the landmarks every step and having multiple loops for the solution of SLAM. In addition, this experiment produces closeness between the landmarks viewed from the frame of the camera. Due to a possible obstruction of detection, this experiment can evaluate how well our data association scheme is affected. As illustrated in

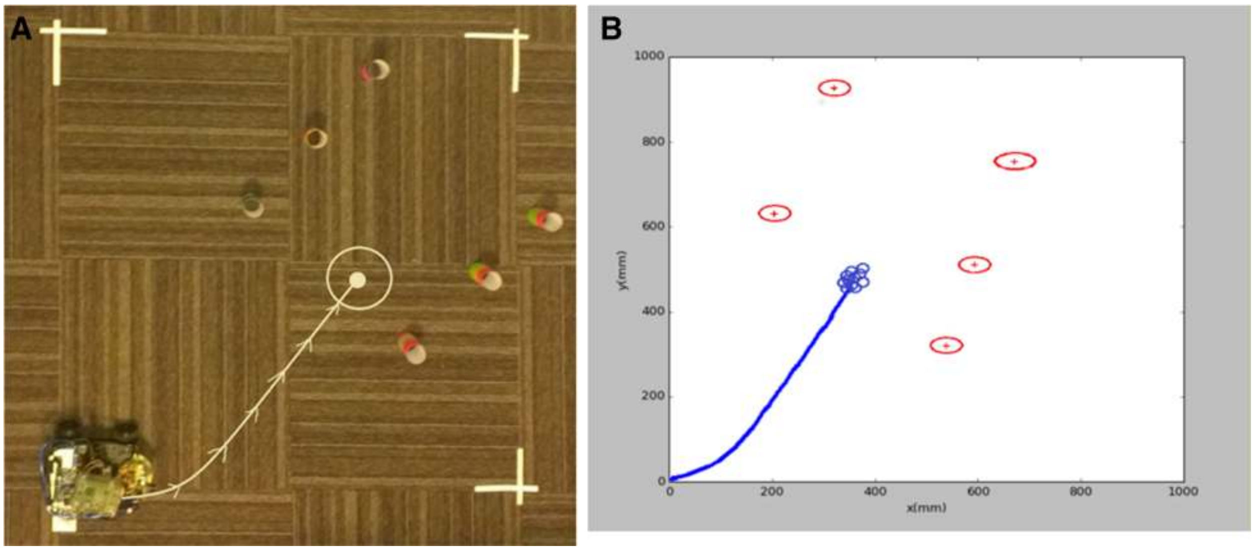


Fig. 8 Corridor detection experiment results. **a** Prototype results. **b** Simulation results

Table 1 The ground truth coordinates from pixy camera vs. estimate values from our SLAM

Coordinates	Landmark	Ground truth (cm)			Estimated (cm)		
		X	*Y*	Θ	*X*	*Y*	Θ
Corner detection	L01	123.01	10.01	4.7°	116.5	11.97	5.9°
	L02	118.9	55.19	24.9°	122.8	52.02	23°
	L03	121.4	104.5	40.7°	124.3	115	42.8°
	L05	57.88	106.5	61.5°	65.45	113.8	60.1°
	L06	18.8	103.8	79.7°	23.69	97.69	76.4°
Corridor detection	L01	64.63	19.95	17.2°	67.67	14.31	11.9°
	L02	84.8	32.32	20.9°	79.81	34.27	23.2°
	L03	93.91	47.06	26.6°	98.03	57.47	30.4°
	L04	58.56	79.6	53.7°	68.75	88.92	52.3°
	L05	44.68	65.06	55.5°	44.03	60.51	54°

Fig. 8a, this experiment has the same setup with exception of the relocation of the landmarks. The average processing time of each frame was 0.56 s. The separation between the top three landmarks on the right side and the second three landmarks on the left side is 48 ± 0.05 cm, as shown in Fig 8b.

The noise values for the solution of SLAM in the motion model of the experiments were 3 cm for the forward (v) motion and 2° for a turn in the movement (w) of the prototype. Similarly, for the observation model, the noise values were 70 mm for the distance measured and 0.027 rad for the angle in the camera frame. The distance measurements from the camera are an approximation of the object detected; as a consequence, these measurements were not incorporated into the proposal distribution. Furthermore, due to this consequence, the filter is overestimated in order to prevent the particle filter to diverge. Likewise, the number of particles used also affects the calculation of the weights assigned to the particles. The overestimation of the distance was 1.2 metres and a standard deviation of 2° for the angle measured. These values tell the filter its belief about how far could be the landmarks. In addition, in all the experiments, 100 particles were used due to the tests performed in a relatively short distance.

4.3 System performance evaluation

Followed the experiments, we conducted in Section 4.2 and 4.1, the artificial landmarks were placed randomly without prior knowledge. The ground truth 3D coordinates of the pixy camera had been recorded and were compared with estimated values from our SLAM solution, as shown in Table 1. Those figures show that on these experiments our SLAM solution gives localisation

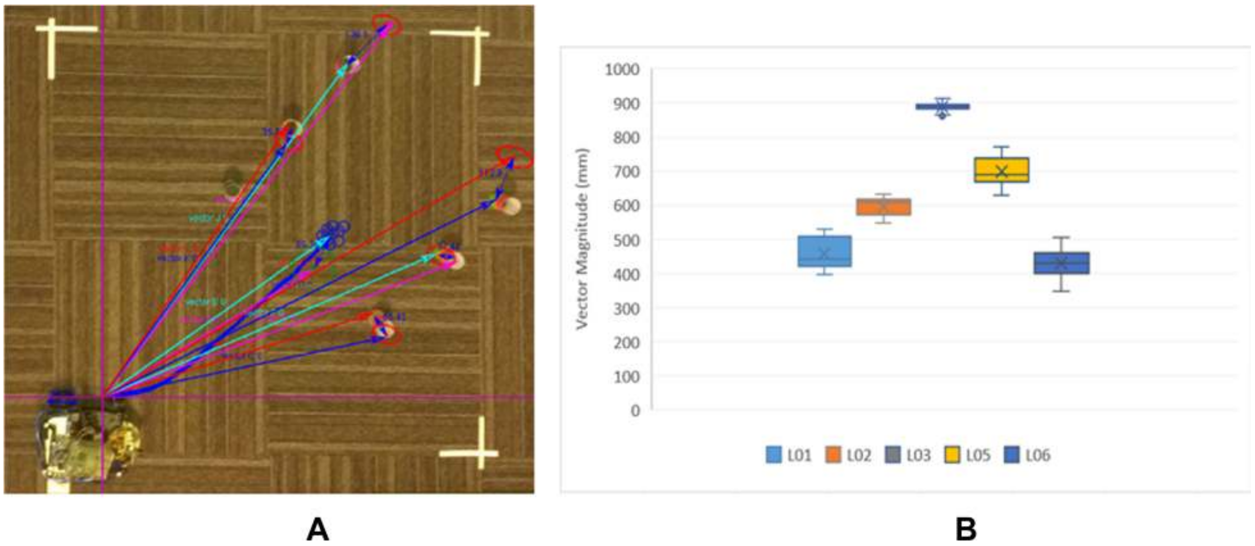


Fig. 9 System preformation evaluation in corner detection. **a** The experiment using prototype for corner detection. **b** Experiment for corner detection with distribution of vector magnitudes of the landmarks in the robot's frame

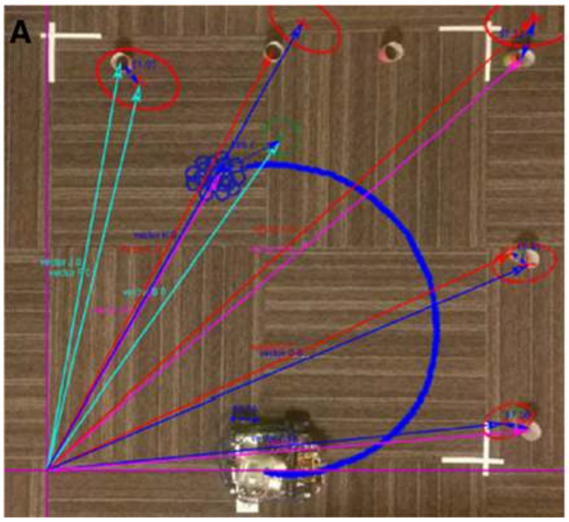
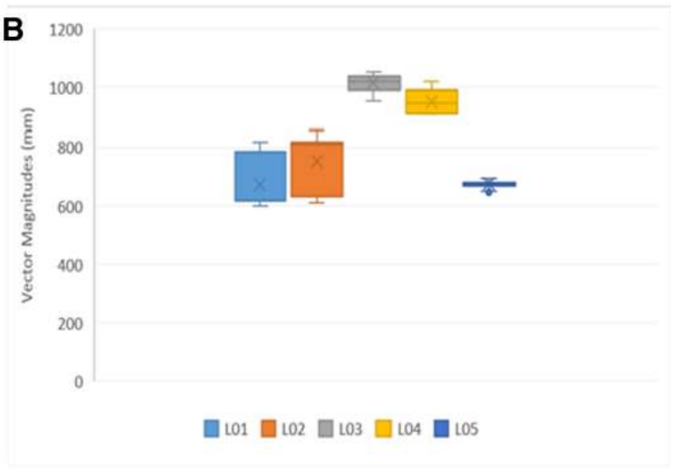



Fig. 10 System preformation evaluation in corridor detection. **a** The experiment using prototype for corner detection. **b** Experiment for corridor detection with distribution of vector magnitudes of the landmarks in the robot's frame

results accurate to a 5 cm. In the corner detection experiments, our central axis is moved due to the recorded path was from the path described by the left wheel in the mobile robots, thus resulting in a skewed trajectory. Figures 9 and 10 shows the error and variance of measurement of landmark's grand truth in both experiments.

Figures 9b and 10b presents the distribution of the landmarks for the three repetitions in experiments on corner detection and corridor detection using our SLAM solution. The graph is composed of the magnitude of the vectors to the landmarks. The greatest variation presented at the landmark is 10 cm, showing precision at mapping. As a result, 83% of the landmarks were detected, generating a map of the exposed landmarks. Being this percentage is sufficient to locate the prototype and differentiate the described environment. During this experiment, the average CPU usage was 26.72%. The second experiment consisting of the parallel detection of landmarks was executed three times. The first run is presented in Figs. 9a and 10a. The prototype detected 83.33% of the landmarks for the first run and a minimum of 66.6% of the landmark detected in the other runs.

4.4 Energy profile

The power consumed by the prototype was measured during above experiments using Tektronix oscilloscope with 1.14 W of total power usage in Raspberry Pi V2.0. Figure 11 illustrated our SLAM approach used an average of 45% ARM cortex-A7 processing time. A native implementation of FastSLAM2.0 requires O(M log(K)), where M is the number of particles in particle filter and K is the number of landmarks. We develop an integrated low-variance resample method to select the most likely particle applying a minimization of the sum of the weights of the particles with respect to a standard uniform distribution. Our approach makes it significantly faster than existing FastSLAM2.0 that reduces the running time of our intelligible SLAM approach to O(M log(K/N)), where N is sample intervals.

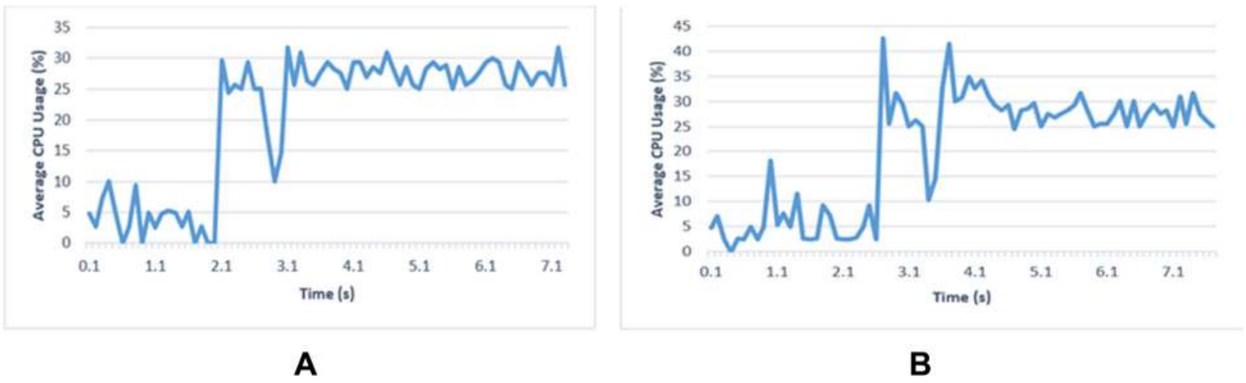


Fig. 11 History of CPU usage during execution of SLAM. **a** Experiment for corner detection. **b** Experiment for corridor detection

5 Conclusions

Monocular visual SLAM in low-power devices presents the challenge of obtaining location information just using a single camera. Using motion (visual odometry) as well as pinhole model or epipolar geometry presents a possibility of implementation; however, researchers have not yet explored a simplified data association solution from the algorithm development into low-cost and low-power embedded system design. This work demonstrates an intelligible implementation of FastSLAM algorithm using low-cost and low-power "on-shelf" devices. Our prototype is able to detect landmark with 75% of success rates and 0.53 s processing time using Raspberry Pi V2.0. Our experiment results demonstrate that our intelligible solution, based on low-cost image sensors to an adequate architecture and a simplified algorithm, is suitable to design embedded systems for SLAM applications in real time conditions.

In the future, with the selection of smart sensors for obtaining the position of the robot, such as a higher resolution encoder or multiple sensor fusion approaches in the prototype, we can improve the landmark detection accuracy and explore outdoor environments. Also, we plan to increase the accuracy of landmark detection using Deep Learning in embedded systems, where multiple objects in the environment can be used to train and evolved the neural network over time.

6 Appendix 1

Table 2 The specifications of pixy camera

Pixy camera specifications			
Field of view	46° vertical, 75° horizontal	Processor	NXP LPC4330, 204 MHz, dual core
Focal length (f)	2.8 mm	Image sensor	Omnivision OV9715, 1/4", 1280 × 800
Sensor height (S_h)	2.43 mm	RAM	264 K bytes
Sensor width (S_w)	2.43 mm		

7 Appendix 2

Table 3 The specification of Raspberry Pi 2

Raspberry Pi 2	
Processor	900 MHz 32-bit Quad-core ARM Cortex-A7
RAM	1 GB (shared with GPU)
USB ports	4 USB 2.0
GPU	Broadcom VideoCore IV @ 250 MHz
Model	B

8 Appendix 3

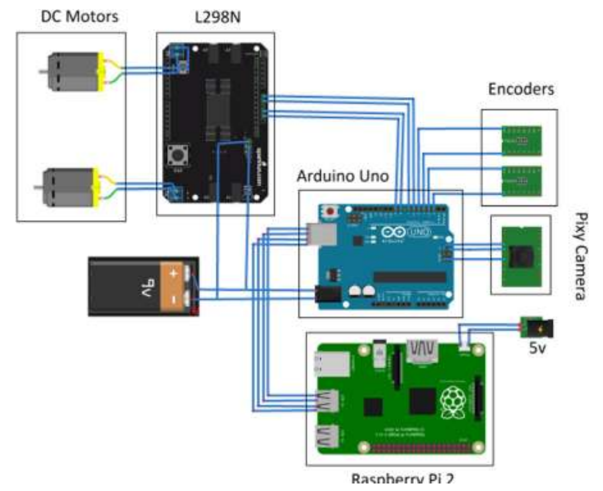


Fig. 12 Hardware schematic view. Raspberry Pi and motor control are powered separately

9 Appendix 4

A link to the most recent version of FastSLAM software as well as a link to the archived version referenced in the manuscript (https://github.com/WOLVS/VisualSLAM).

Funding
The project is partly supported by Research on Modelling and Algorithms of Functional Safety for New Generation Automotive Embedded Systems (61672217), National Natural Science Foundation of China. It is also supported through Quality Research funding, Faculty of Science and Engineering, University of Wolverhampton.

Authors' contributions
RL contributed on the conception and the design of study. AAJS drafted the manuscript and conducted an analysis of the experiment data. SY contributed on revising the manuscript critically for important intellectual content and also helped on the interpretation of experiment data. AAJS, RL, SY approved the final version of the manuscript to be published.

Competing interests
The authors declare that they have no competing interests.

Received: 30 November 2016 Accepted: 9 February 2017
Published online: 02 March 2017

References
1. DC Asmar, JS Zelek, SM Abdallah, Tree trunks as landmarks for outdoor vision SLAM. 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06). (2006), 196–196
2. Y Bar-Shalom, Tracking and data association Academic Press Professional, Inc. (1987)
3. T Botterill, *Visual navigation for mobile robots using the bag-of-words algorithm* (Doctoral dissertation). Retrieved from University of Canterbury database. 2011
4. E Brynjolfsson, A McAfee, The second machine age: work, progress, and prosperity in a time of brilliant technologies. WW Norton & Company. (2014)

5. M Calonder, V Lepetit, C Strecha, P Fua, Brief: binary robust independent elementary features. European Conference on Computer Vision. (2010), 778–792
6. T Chong, X Tang, C Leng, M Yogeswaran, O Ng, Y Chong, Sensor technologies and simultaneous localization and mapping (SLAM). Procedia Computer Science **76**, 174–9 (2015)
7. M Csorba, *Simultaneous localisation and map building*, 1997
8. AJ Davison, ID Reid, ND Molton, O Stasse, MonoSLAM: real-time single camera SLAM. IEEE Trans Pattern Anal Mach Intell **29**(6), 1052–67 (2007)
9. Doucet, A., De Freitas, N., Murphy, K., & Russell, S. Rao-blackwellised particle filtering for dynamic bayesian networks. Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence. (2000), 176–183.
10. M Fiala, A Ufkes, Visual odometry using 3-dimensional video input. Computer and Robot Vision (CRV), 2011 Canadian Conference on. (2011), 86–93
11. J Hartmann, JH Klüssendorff, A Maehle, A comparison of feature descriptors for visual SLAM. Mobile Robots (ECMR), 2013 European Conference on. (2013), 56–61.
12. G Hendeby, R Karlsson, F Gustafsson, A new formulation of the rao-blackwellized particle filter. 2007 IEEE/SP 14th Workshop on Statistical Signal Processing. (2007), 84–88
13. G Hendeby, R Karlsson, F Gustafsson, The rao-blackwellized particle filter: a filter bank implementation. EURASIP Journal on Advances in Signal Processing **2010**(1), 1 (2010)
14. G Klein, D Murray, Parallel tracking and mapping for small AR workspaces. Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on. (2007), 225–234.
15. H Lim, YS Lee, Real-time single camera SLAM using fiducial markers. Iccas-Sice **2009**, 177–82 (2009)
16. M Llofriu, F Andrade, F Benavides, A Weitzenfeld, G Tejera, An embedded particle filter SLAM implementation using an affordable platform, in *Advanced Robotics (ICAR), 2013 16th International Conference on*, 2013, pp. 1–6
17. S Magnenat, V Longchamp, M Bonani, P Rétornaz, P Germano, H Bleuler, F Mondada, Affordable slam through the co-design of hardware and methodology, in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 5395–401
18. D Meyer-Delius, M Beinhofer, A Kleiner, W Burgard, Using artificial landmarks to reduce the ambiguity in the environment of a mobile robot, in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 5173–8
19. M Milford, E Vig, W Scheirer, D Cox, Vision-based simultaneous localization and mapping in changing outdoor environments. Journal of Field Robotics **31**(5), 780–802 (2014)
20. M Montemerlo, S Thrun, D Koller, B Wegbreit, FastSLAM: a factored solution to the simultaneous localization and mapping problem. Aaai/iaai. (2002), 593–598
21. M Muja, DG Lowe, Fast approximate nearest neighbors with automatic algorithm configuration. Visapp (1), 2(331–340), 2 (2009)
22. J Nieto, J Guivant, E Nebot, S Thrun, Real time data association for FastSLAM. Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on, 1 412–418, (2003)
23. K Okuyama, T Kawasaki, V Kroumov, Localization and position correction for mobile robot using artificial visual landmarks, in *The 2011 International Conference on Advanced Mechatronic Systems*, 2011, pp. 414–8
24. T Pire, T Fischer, J Civera, P De Cristóforis, JJ Berlles, Stereo parallel tracking and mapping for robot localization, in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, 2015, pp. 1373–8
25. H Strasdat, C Stachniss, W Burgard, Which landmark is useful? Learning selection policies for navigation in unknown environments, in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, 2009, pp. 1410–5
26. KT Sutherland, WB Thompson, Inexact navigation, in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, 1993, pp. 1–7
27. T Suzuki, Y Amano, T Hashizume, Development of a SIFT based monocular EKF-SLAM algorithm for a small unmanned aerial vehicle. SICE Annual Conference (SICE), 2011 Proceedings of, 1656-1659 (2011)
28. S Thrun, Finding landmarks for mobile robot navigation. Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on, 2 958–963. (1998)
29. S Thrun, W Burgard, D Fox, Probabilistic robotics. MIT Press. (2005)
30. I Vallivaara, J Haverinen, A Kemppainen, J Röning, Magnetic field-based SLAM method for solving the localization problem in mobile robot floor-cleaning task, in *Advanced Robotics (ICAR), 2011 15th International Conference on*, 2011, pp. 198–203
31. J Wang, Y Takahashi, *Particle filter based landmark mapping for SLAM of mobile robot based on RFID system*, 2016
32. S Zhang, L Xie, MD Adams, Entropy based feature selection scheme for real time simultaneous localization and map building, in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 1175–80