Int. J. Appl. Math. Comput. Sci., 2010, Vol. 20, No. 2, 281–293
DOI: 10.2478/v10006-010-0021-7

VISUAL SIMULTANEOUS LOCALISATION AND MAP–BUILDING SUPPORTED BY STRUCTURED LANDMARKS

ROBERT BĄCZYK, ANDRZEJ KASIŃSKI

Institute of Control and Information Engineering
Poznań University of Technology, Pl. Marii Skłodowskiej-Curie 5, 60–965 Poznań, Poland
e-mail: robert.baczyk@put.poznan.pl

Visual simultaneous localisation and map-building systems which take advantage of some landmarks other than point-wise environment features are not frequently reported. In the following paper the method of using the operational map of robot surrounding, which is complemented with visible structured passive landmarks, is described. These landmarks are used to improve self-localisation accuracy of the robot camera and to reduce the size of the Kalman-filter state-vector with respect to the vector size involving point-wise environment features only. Structured landmarks reduce the drift of the camera pose estimate and improve the reliability of the map which is built on-line. Results of simulation experiments are described, proving advantages of such an approach.

Keywords: mobile robots, navigation, SLAM, artificial landmarks.

1. Introduction

The goal of the research is to investigate the tracking properties of the Simultaneous Localisation And Map-building (SLAM) algorithm, which reconstructs the trajectory of a mobile camera navigating in an unknown environment. By tracking properties we understand here the ability to properly extend the camera trajectory based on images of the surroundings; while continuing the course some landmarks escape the field of view but some new appear. Usually such a SLAM system works according to the sequential estimation scheme.

At the beginning of the mission, reference coordinates of the starting point have to be initialized. Then features of the environment are extracted from the current image, and characteristic objects are recognized. As such they are put on the dynamically constructed map, which has been empty so far.[1] At the next stage, as the camera moves, the following tasks have to be accomplished:

- determination of the updated camera position and orientation (shortly: pose) with respect to the global reference system set at the beginning of the mission,
- refinement of the location of features/objects on the map with respect to the global reference frame.

Moreover, there is usually the need to add new items on the map, which have been discovered from the new camera position. The next step is to move the camera forward along a trajectory and to establish a new viewer pose. Then such a sequence is repeated.

Self-localisation of the robot with a camera fixed on board is a problem tightly coupled with the issue of map building, while navigating in an unknown (new) neighbourhood. None can be solved on its own and they have to be worked out simultaneously.

Last years have witnessed substantial progress in mobile robots navigation systems (Durrant-Whyte and Bailey, 2006; Bailey and Durrant-Whyte, 2006; Neira *et al.*, 2008). Most approaches solve the SLAM task by using sequential Bayesian schemes. The Kalman filter is a good example of it (Bar-Shalom *et al.*, 2002). Working implementations of navigation systems based on such an approach emerge (Davison *et al.*, 2007), and it can be observed that the quality of solutions is correlated with the type and number of sensors used in a particular implementation (Thrun *et al.*, 2006).

The requested characteristics, fundamental to the working SLAM system, cover its ability to work on-line. A SLAM system having such a property can be applied

[1] In the case of bearing-only SLAM there is the need to bootstrap the system by injecting into the map some known 3D points or by recognising objects of known dimensions.

whenever the current camera pose estimate is used within the closed-loop navigational decision scheme of the robot controler.

SLAM can be used to guide autonomous agents (mobile robots) on the shop-floor, in hospitals or in offices, to support the mobility of humanoid robots or blind people, and, last but not least, to guide a military warfare. Some other applications areas can include: augmented reality, city-map building or extraterrestrial navigation systems, in which the communication latency prohibits the on-line remote control.

Most of the reported SLAM methods involve multiple sensors to observe the environment, while in this paper a system using only a single camera is described. With such an approach, hardware complexity is reduced, therefore system reliability grows and portable SLAM systems involving embedded PC solutions are feasible.

2. Discussion and original elements of the approach

Here, related earlier works will be presented as the background of the contribution.

Single camera SLAM systems reported in (Davison and Murray, 2002; Davison, 2003; Davison *et al*., 2007) allow exploring compact indoor spaces. A map generated by the SLAM process is sparse there and usually consists of a finite set of point-wise environment features (shortly: point-features) coordinates. The map is primary used to establish the camera pose with respect to the global reference system.

The problem is that the computational cost of the SLAM algorithm grows with the square of the number of map-items. So it is wise to reduce the system state vector dimension consisting of the camera state vector and the environment map items coordinates. Breaking the upper bound of the state-vector size can disable the on-line operation of the system, given the constrained on-board computing resources.

The second important issue while navigating is a viewer trajectory drift, which complicates the closing of the robot path loop (Clemente *et al*., 2007). Sometimes, it is necessary to explore the neighbourhood of some reference point along multiple exploratory trajectories. In such a case the viewer prefers to come back to the starting point following a new path in order to acquire some new map items. Loop closing is one of the most important properties required in SLAM since it allows reducing accumulation of errors (enables position resetting).

Looking for a solution which would keep the complexity of the map under control and compensate for the trajectory drift put on the agenda the issue of a sparse map management, which consists of not only point-features. To manage the map efficiently it is wise not to rely on the environment point-features only (natural landmarks), but to account also for purposefully located artificial simple landmarks (for example, hue-planar, passive patterns). A particular design of such a landmark,[2] aimed at providing efficient recognition results, was described in (Bączyk *et al*., 2003). Similar solutions are used in augmented reality systems. Such landmarks are introduced there because of the critical importance of the precise collocation of coordinates of virtual/artificial objects with the real-scene objects poses. In a SLAM system it is of general interest to establish a pose of the free-flying camera with respect to the current environment model, which is built on-line during the camera travel. By augmenting a SLAM system with the ability of structured landmarks recognition some well known limitations of the single camera as a bearing-only SLAM sensor can be overcome.

In known visual SLAM systems it is rare to account for map items other than natural point-wise image features. Line segments were accounted for in the system of Smith *et al.* (2006). There the goal was to improve point-features recognition reliability (important while solving the feature correspondence problem in the sequence of scene images). From the viewpoint of filtration process performance, such an extension was not really substantial, as finally the extracted segment endings were again treated simply as point-features.

Edgelet landmarks introduced in (Eade and Drummond, 2009) were not a substantial modification as in their algorithm they were treated as point-features as well.

Higher level structures in visual SLAM were also proposed in (Gee *et al*., 2008). Discovered lines and planes in space were fitted to points contained in the map, but images were not directly used for this purpose. Higher level objects allow reducing the state vector dimension and provide more meaning to the SLAM map as compared to collections of points only. On the other hand, this procedure is complex and there are a number of coefficients to be set. The authors mention that their approach can lead to estimates inconsistency and its ability to error reduction is controversial.

In (Castle *et al*., 2007a ; 2007b), some known planar patterns were used together with point-features. The planar pattern recognition sequence runs on the PC second core as a separate thread at around 1.5 Hz rate. In the recognition scheme the authors use SIFT descriptors and the RANSAC method, which are too computationally expensive for video frame-rate operation, and thus the insertion of planar structures into the map is delayed. The authors decompose homography (computed as the output of RANSAC) between the planar structure and the current image into camera calibration matrix, rotation and translation.

[2] It is important to distinguish point-features from landmarks (as in some papers they are called landmarks as well). By landmarks we mean here the structured patterns purposefully introduced to the explored space.

Extrinsic parameters are then determined using singular value decomposition. Differently enough here we show the advantages of embedding landmark measurement into the filtration process at frame-rate frequency. Landmarks' rectangular shape and unique code allows the use fast and robust recognition and a pose evaluation method (Bączyk *et al.*, 2003).

Another drawback of the mentioned method (Castle *et al.*, 2007a; 2007b) is that the planar object was represented on the created map merely by three boundary points. It is not clear if the use of three boundary points treated as distinct point-features in space can hold rigid nature of a planar object during the filtration process. It is not mentioned there how the comparison of prediction and measurement is applied in the case of planar structure recognition.

Here we propose to treat planar patterns as a new class of environment objects and, as such, to put them on the map. So the SLAM state vector containing a representation of two categories of features is maintained and in the single system a measurement in the image space for the first and in the 3D space for the second feature type is performed.

The next drawback of the Castle approach is that the covariance matrix (statistically characterising the uncertainty of the evaluated planar object position) is not formally obtained. Its diagonal elements were evaluated there only in a trial-and-error experiment, by testing the influence of their values on the convergence of the filtration process.

With our method, landmark position uncertainty with respect to the camera reference frame is formally obtained by propagating the uncertainty of image coordinates.

3. SLAM fundamentals

Assumptions. The following assumptions have been made at the beginning:

- The agent (a hand-held camera or mobile robot with a fixed camera) is moving with three degrees of freedom (planar motion case). Its references are x, y coordinates and orientation angle θ on the plane. However, the scheme presented in this paper can be easily generalised to the case of six degrees of freedom (free 3D camera motion).
- The only external sensor available in the system is a single camera.
- The guidance of the camera is not known *a priori* (*free motion*).
- Camera motion is smooth (continuous acceleration, no jerks).
- Intrinsic camera parameters are known *a priori* (the camera has been calibrated beforehand).

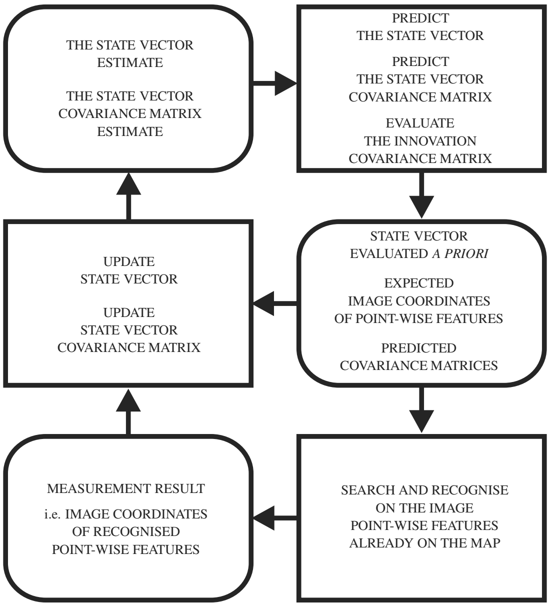

THE STATE VECTOR ESTIMATE
THE STATE VECTOR COVARIANCE MATRIX ESTIMATE
PREDICT THE STATE VECTOR
PREDICT THE STATE VECTOR COVARIANCE MATRIX
EVALUATE THE INNOVATION COVARIANCE MATRIX
UPDATE STATE VECTOR
UPDATE STATE VECTOR COVARIANCE MATRIX
STATE VECTOR EVALUATED *A PRIORI*
EXPECTED IMAGE COORDINATES OF POINT-WISE FEATURES
PREDICTED COVARIANCE MATRICES
MEASUREMENT RESULT
i.e. IMAGE COORDINATES OF RECOGNISED POINT-WISE FEATURES
SEARCH AND RECOGNISE ON THE IMAGE POINT-WISE FEATURES ALREADY ON THE MAP

Fig. 1. Cyclic tracking sequence using the EKF scheme in the case of point-wise environment features put on the map.

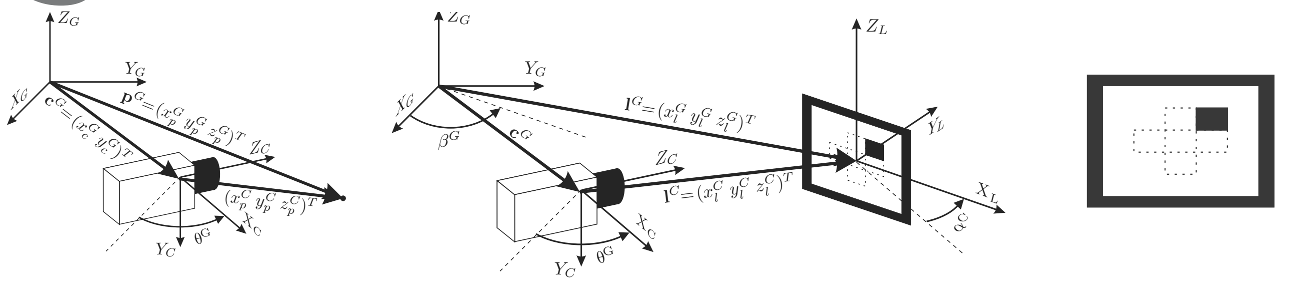
- The map consists of point-wise, stationary natural features fixed to the observation space and of freely distributed rectangular, planar landmarks.
- The position of landmarks is not known *a priori*, however, short edges of the landmark frame are kept perpendicular to the motion plane.
- The raw measurement data of the SLAM system are image-coordinates of the point-features and of the landmark corners.

System goals. The SLAM system uses an Extended Kalman Filter (EKF), working sequentially (Fig. 1) on the consecutive image frames. Its role is to track the observer trajectory, consisting of a position, camera optical axis orientation and its linear and angular velocity. Moreover the EKF tracks fixed features, puts them on the map and steadily refines their coordinates already on the map. The EKF calculates on-line the covariance matrix of measurements data to provide the uncertainty measure of tracked variables and of stationary coordinates.

It is a typical situation that camera pose uncertainty grows with the trajectory length; however, the uncertainty of the stationary data (such as features and landmarks coordinates) always decreases with the length of the observation sequence (Dissanayake *et al.*, 2001).

The EKF state-vector, which is critical data in the navigation system, can be defined as follows:

$$\mathbf{x} = \Big(\mathbf{c}^{GT}\ \theta^{G}\ \ \dot{\mathbf{c}}^{GT}\ \dot{\theta}^{G}\ \mathbf{p}_1^{GT} \cdots \mathbf{p}_n^{GT}\ \mathbf{l}_1^{GT}\ \beta_1^{G} \cdots \mathbf{l}_m^{GT}\ \beta_m^{G}\Big)^T. \quad (1)$$


Z_G
Y_G
X_G
$\mathbf{c}^G\!=\!(x_c^G\ y_c^G)^T$
$\mathbf{p}^G\!=\!(x_p^G\ y_p^G\ z_p^G)^T$
Z_C
$(x_p^C\ y_p^C\ z_p^C)^T$
X_C
Y_C
θ^G
Z_G
Y_G
X_G
β^G
$\mathbf{c}^G$
$\mathbf{l}^G\!=\!(x_l^G\ y_l^G\ z_l^G)^T$
Z_C
$\mathbf{l}^C\!=\!(x_l^C\ y_l^C\ z_l^C)^T$
X_C
Y_C
θ^G
Z_L
Y_L
X_L
α^C

(a) (b) (c)

Fig. 2. Geometrical relations in a global reference system between the camera and the point-wise feature (a) and the landmark (b); landmark template (c).

It consists of the observer trajectory subvector, whose coordinates are the observer position on the motion plane $\mathbf{c}^G = \left(x_c^G\ y_c^G\right)^T$, the orientation θ^G, the linear velocity subvector $\dot{\mathbf{c}}^G = \left(\dot{x}_c^G\ \dot{y}_c^G\right)^T$ and the angular velocity $\dot{\theta}^G$. The subsequent elements of the state vector are triples of absolute Cartesian coordinates of n observed point-features $\mathbf{p}_i^G = \left(x_{p_i}^G\ y_{p_i}^G\ z_{p_i}^G\right)^T$, $i = 1, \ldots, n$, fixed to the environment. Next elements of the state vector are poses of m landmarks also expressed in the global reference frame:

$$\left(\mathbf{l}_j^{G\,T}\ \beta_j^G\right)^T = \left(x_{l_j}^G\ y_{l_j}^G\ z_{l_j}^G\ \theta^G + \alpha_j^C\right)^T, \qquad (2)$$

$j = 1, \ldots, m$. In turn, the position of the j-th landmark expressed in the camera coordinates is denoted as $\mathbf{l}_j^C = (x_{l_j}^C\ y_{l_j}^C\ z_{l_j}^C)^T$ and its orientation as α_j^C. Geometrical relations are shown in the Fig. 2.

The state transition model is the following:

$$\hat{\mathbf{x}}(k+1|k) = f(\hat{\mathbf{x}}(k|k), \mathbf{n}_{\mathbf{x}}(k+1))$$

$$= \begin{pmatrix} \hat{x}_c^G(k|k) + \left(\hat{\dot{x}}_c^G(k|k) + \frac{1}{2}n_{\dot{x}}(k+1)\right) \triangle t \\ \hat{y}_c^G(k|k) + \left(\hat{\dot{y}}_c^G(k|k) + \frac{1}{2}n_{\dot{y}}(k+1)\right) \triangle t \\ \hat{\theta}^G(k|k) + \left(\hat{\dot{\theta}}^G(k|k) + \frac{1}{2}n_{\dot{\theta}}(k+1)\right) \triangle t \\ \hat{\dot{x}}_c^G(k|k) + n_{\dot{x}}(k+1) \\ \hat{\dot{y}}_c^G(k|k) + n_{\dot{y}}(k+1) \\ \hat{\dot{\theta}}^G(k|k) + n_{\dot{\theta}}(k+1) \\ \hat{\mathbf{p}}_1^G(k|k) \\ \vdots \\ \hat{\mathbf{p}}_n^G(k|k) \\ \hat{\mathbf{l}}_1^G(k|k) \\ \hat{\beta}_1^G(k|k) \\ \vdots \\ \hat{\mathbf{l}}_m^G(k|k) \\ \hat{\beta}_m^G(k|k) \end{pmatrix}. \qquad (3)$$

As velocity changes are not available from direct measurements, a simplified constant-velocity model is used. The model accounts for the random noise reflecting the velocity changes $\mathbf{n}_{\mathbf{x}}(k) = \left(n_{\dot{x}}(k)\ n_{\dot{y}}(k)\ n_{\dot{\theta}}(k)\right)^T$. In the state vector prediction step, a new camera position is calculated and the integration of noise is preformed with the trapezoid averaging rule instead of the rectangle method used in (Davison *et al.*, 2007). That approach provides better convergence of the computations.

It is worth noticing that in (3) only a subvector of the state-vector which represents the camera pose is modified. Predicted coordinates of the point-features and landmarks are left unmodified (as they are assumed to be stationary) and their estimates are modified only in the filtration *update* step.

4. SLAM scheme for point-features

4.1. Main algorithm.

The system covariance matrix P is predicted according to the following formula:

$$\mathbf{P}(k+1|k) = J_f(k) \left[\begin{array}{c:c} \mathbf{P}(k|k) & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{P}_{\mathbf{n}_{\mathbf{x}}}(k) \end{array}\right] J_f^T(k), \quad (4)$$

where $J_f(k)$ is the Jacobian matrix of the right side function (3) at the time k and $\mathbf{P}_{\mathbf{n}_{\mathbf{x}}}(k)$ is the covariance matrix of the noise $\mathbf{n}_{\mathbf{x}}(k)$ mentioned above.

Measurement prediction. Coordinates of point-features put on the map are measured on images. Thus based on the predicted state vector $\hat{\mathbf{x}}(k+1|k)$, the expected image-coordinates of features have to be evaluated. This is done by the use of the following pin-hole camera model:

$$\hat{\mathbf{h}}_i(k+1|k) = h_i\left(\hat{\mathbf{x}}(k+1|k), \mathbf{n}_{p_i}(k+1)\right)$$

$$= \begin{pmatrix} \lambda_u \frac{\hat{x}_{p_i}^C(k+1|k)}{\hat{z}_{p_i}^C(k+1|k)} + u_c + n_{u_{p_i}}(k+1) \\ \\ \lambda_v \frac{\hat{y}_{p_i}^C(k+1|k)}{\hat{z}_{p_i}^C(k+1|k)} + v_c + n_{v_{p_i}}(k+1) \end{pmatrix}, \qquad (5)$$

where λ_u, λ_v denote focal lengths for image u, v coordinates expressed in pixels, u_c, v_c denote image-coordinates of optical center, $\hat{\mathbf{h}}_i(k+1|k)$ denotes the predicted image coordinates of the i-th map-point, and the variables

$$\begin{pmatrix} \hat{x}^C_{p_i}(k+1|k) \\ \hat{y}^C_{p_i}(k+1|k) \\ \hat{z}^C_{p_i}(k+1|k) \end{pmatrix} = \hat{\mathbf{p}}^C_i(k+1|k) \tag{6}$$

denote its coordinates calculated at the $(k+1)$-th prediction step, expressed in the camera reference frame (see Fig. 2):

$$\begin{aligned} \hat{\mathbf{p}}^C_i(k+1|k) &= {}_C\hat{\mathbf{R}}^G(k+1|k) \\ &\times \Big(\hat{\mathbf{p}}^G_i(k+1|k) - \big(\hat{x}^G_c(k+1|k)\ \hat{y}^G_c(k+1|k)\ 0\big)^T\Big). \end{aligned} \tag{7}$$

The rotation matrix ${}_C\hat{\mathbf{R}}^G(k+1|k)$ can be calculated knowing the angle $\hat{\theta}^G(k+1|k)$ being the element of the predicted state $\hat{\mathbf{x}}(k+1|k)$. The model (5) accounts for the fact that readouts of image coordinates are corrupted by measurement noise vector: $\mathbf{n}_{p_i}(k) = \big(n_{u_{p_i}}(k)\ n_{v_{p_i}}(k)\big)^T$.

The function $h_i\big(\hat{\mathbf{x}}(k+1|k), \mathbf{n}_{p_i}(k+1)\big)$ is defined with the formulas (5)–(7) and it models the camera in the case of i-th point-features being put on the map.

The innovation vector represents the difference between the measured and predicted features coordinates in the image domain:

$$\mathbf{v}(k+1) = \mathbf{h}(k+1) - \hat{\mathbf{h}}(k+1|k), \tag{8}$$

where $\mathbf{h}(k+1)$ denotes the measurement result at the $k+1$ step and $\hat{\mathbf{h}}(k+1|k)$ denotes a result calculated with the state vector estimate predicted at the $(k+1)$-th step, based on the data from the k-th step, according to (5).

The innovation vector covariance matrix represents measurement and prediction uncertainty. It is obtained as a combination of the state-vector uncertainty $\mathbf{P}(k+1|k)$ and the measurement uncertainty $\mathbf{P}_{\mathbf{n}_p}(k+1)$:

$$\begin{aligned} &\mathbf{S}(k+1) \\ &= J_h(k+1)\left[\begin{array}{c:c} \mathbf{P}(k+1|k) & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{P}_{\mathbf{n}_p}(k+1) \end{array}\right] J_h^T(k+1), \end{aligned} \tag{9}$$

where $J_h(k+1)$ is the Jacobian matrix consisting of partial derivatives of the functions h_i (5) at the $(k+1)$-th step.

At each step, the measurement vector $\mathbf{h}(k+1)$ can vary its dimension as it is clear that at the current observation pose only some subsets of map elements are visible, while some of them can be undetected. Thus, at each sequential filtration step some rows from the vector $\hat{\mathbf{h}}(k+1|k)$ in (8) and the Jacobian matrix $J_h(k+1)$ have to be removed, namely, those having no corespondents in the measurements vector $\mathbf{h}(k+1)$.

EKF filtering scheme. In the general filtering scheme, the *update* for the state vector and its covariance matrix is defined with the following formulas:

$$\begin{aligned} &\hat{\mathbf{x}}(k+1|k+1) \\ &\quad = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}(k+1)\ \mathbf{v}(k+1), \end{aligned} \tag{10}$$

$$\begin{aligned} &\mathbf{P}(k+1|k+1) \\ &\quad = \mathbf{P}(k+1|k) - \mathbf{W}(k+1)\ \mathbf{S}(k+1)\mathbf{W}^T(k+1), \end{aligned} \tag{11}$$

while the weighting matrix $\mathbf{W}(k+1)$, appearing in (10) and (11), is established from

$$\begin{aligned} &\mathbf{W}(k+1) \\ &\quad = \mathbf{P}(k+1|k)\ \frac{\partial h\big(\hat{\mathbf{x}}(k+1|k), \mathbf{0}\big)}{\partial \hat{\mathbf{x}}}^T\ \mathbf{S}^{-1}(k+1). \end{aligned} \tag{12}$$

4.2. Point-feature initialization method. The procedure aims at calculating a rough position estimate of the new point-feature which appears on the image. Further, making use of the empirical knowledge on the size of the observed space, one can determine the probability distribution of the spatial position of the feature, which has been detected (as a new one) in the current image. A similar approach was used in (Gee and Mayol-Cuevas, 2006), but a detailed explanation was missing. A much more complex initialization method discussed in (Civera *et al.*, 2008) contrasts with the one presented below.

A point is put on the map with its uncertainty measure instantly, at the same step at which the point-feature is selected from the set of points detected in the current viewfield. Assuming that adding a point-feature to the map was undertaken at the instant k, just after the update stage (10), (11), the new, current system state vector has the following value:

$$\begin{aligned} &\hat{\mathbf{x}}_{\text{new}}(k|k) \\ &= f_2\big(\hat{\mathbf{x}}(k|k), u_{\text{new}}(k), v_{\text{new}}(k)\big) \\ &= \begin{pmatrix} \hat{\mathbf{x}}(k|k) \\ f_1\big(\hat{\mathbf{c}}^G(k|k), \hat{\theta}^G(k|k), u_{\text{new}}(k), v_{\text{new}}(k), \mathbf{0}, 0\big) \end{pmatrix}, \end{aligned} \tag{13}$$

where f_1 represents the inverse camera model. Thus it determines spatial coordinates of such a newly emerging point-feature:

$$\begin{aligned} &f_1\big(\hat{\mathbf{c}}^G(k|k), \hat{\theta}^G(k|k), u_{\text{new}}(k), v_{\text{new}}(k), \mathbf{n}_{p_{\text{new}}}, n_{\text{dst}}\big) \\ &= \big(\hat{x}^G_c(k|k)\ \hat{y}^G_c(k|k)\ 0\big)^T \end{aligned}$$

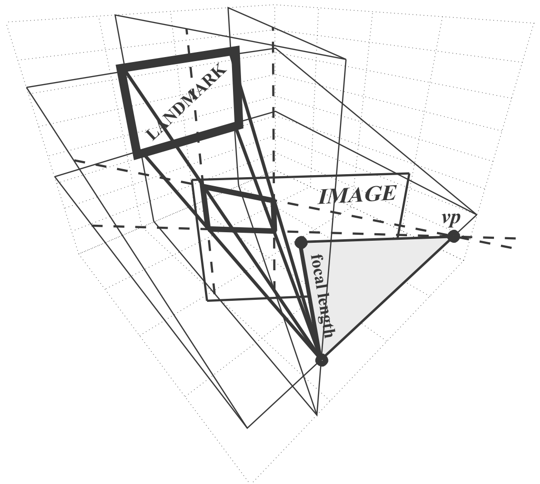


Fig. 3. Geometrical relations used for the evaluation of landmark rotation.

$$+{}_G\hat{\mathbf{R}}^C(k|k)\begin{pmatrix} \frac{\left(u_{\text{new}}(k)-u_c+n_{u_{p_{\text{new}}}}(k)\right)\left(dst_{\text{init}}+n_{\text{dst}}(k)\right)}{\lambda_u} \\ \frac{\left(v_{\text{new}}(k)-v_c+n_{v_{p_{\text{new}}}}(k)\right)\left(dst_{\text{init}}+n_{\text{dst}}(k)\right)}{\lambda_v} \\ dst_{\text{init}}+n_{\text{dst}}(k) \end{pmatrix}. \tag{14}$$

Similarly to (5), this model also accounts for the image measurement noise denoted here as $\mathbf{n}_{p_{\text{new}}} = \left(n_{u_{p_{\text{new}}}}\ n_{v_{p_{\text{new}}}}\right)^T$. The rotation matrix ${}_G\hat{\mathbf{R}}^C(k|k)$ is calculated with $\hat{\theta}^G(k|k)$ picked from the state vector after the *update* step. The constant dst_{init} is empirical. It represents the most likely value of the z-coordinate (in the Z_C direction) of the point-feature. The uncertainty of that coordinate is represented by some random variable $n_{\text{dst}}(k)$. The system covariance matrix augmented with the new points position uncertainty is calculated from

$$\begin{aligned} &\mathbf{P}_{\text{new}}(k|k) \\ &= J_{f_2}(k)\left[\begin{array}{c:c:c} \mathbf{P}(k|k) & \mathbf{0} & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{P}_{\mathbf{n}_{p_{\text{new}}}}(k) & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{0} & \sigma^2_{n_{\text{dst}}}(k) \end{array}\right] J^T_{f_2}(k), \end{aligned} \tag{15}$$

where $\sigma_{n_{dst}}$ is a standard deviation of n_{dst} which has to be evaluated empirically.

5. Introducing landmarks to SLAM

5.1. Determining the landmark pose. As to the landmarks, the measurement result is the pose of the landmark in the global reference frame. This pose is obtained from the image coordinates of the landmark corners. The proces of landmark pose determination is described below.

The landmark rotation angle $\boldsymbol{\alpha^C}$ (ref. to Fig. 2) with respect to the camera reference frame can be calculated from the following formula:

$$\begin{aligned} \alpha^C &= f_4(u_1,v_1,u_2,v_2,u_3,v_3,u_4,v_4) \\ &= \operatorname{atan}\left(\frac{\lambda_u}{f_3(u_1,v_1,u_2,v_2,u_3,v_3,u_4,v_4)}\right) \\ &= \operatorname{atan}\left(\frac{\lambda_u}{u_{vp}}\right), \end{aligned} \tag{16}$$

where the value of the function f_3 equals the image coordinate u_{vp} of the vanishing point. It is obtained by the crossing of lines fitted to the horizontal landmark sides, as shown in Fig. 3. The arguments of f_3 are image coordinates of the landmark corners $u_1\ v_1 \ldots u_4\ v_4$.

The landmark position $\mathbf{l}^C = \left(x_l^C\ y_l^C\ z_l^C\right)^T$ can be calculated by using the camera mathematical model:

$$\begin{aligned} &\begin{bmatrix} \lambda_u & 0 & 0 & 0 \\ 0 & \lambda_v & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} (\quad \mathbf{r}_1 \quad) & x_l^C \\ (\quad \mathbf{r}_2 \quad) & y_l^C \\ (\quad \mathbf{r}_3 \quad) & z_l^C \\ (0\ 0\ 0) & 1 \end{bmatrix}\begin{pmatrix} x_e^L \\ y_e^L \\ z_e^L \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} z_e^C u_e \\ z_e^C v_e \\ z_e^C \\ 1 \end{pmatrix}, \qquad \text{where} \qquad \begin{bmatrix} (\quad \mathbf{r}_1 \quad) \\ (\quad \mathbf{r}_2 \quad) \\ (\quad \mathbf{r}_3 \quad) \end{bmatrix} = {}_C\mathbf{R}^L, \end{aligned} \tag{19}$$

$e = 1,2,3,4$ stands for the appropriate edge index of the rectangular landmark. To simplify the formulas, it has been assumed that the origin of the image coordinate frame is in the optical centre of the image (compare (5) and (14) by taking $u_c = v_c = 0$). While having the value of α^C (16), the elements of the rotation matrix ${}_C\mathbf{R}^L$ can be calculated. One can shift all expressions to one side of Eqn. (19) and denote by d_{u_e} and d_{v_e} the obtained differences:

$$\begin{aligned} d_{u_e} &= u_e\Big(\mathbf{r}_3\big(x_e^L\ y_e^L\ z_e^L\big)^T + z_l^C\Big) \\ &\quad - \lambda_u\Big(\mathbf{r}_1\big(x_e^L\ y_e^L\ z_e^L\big)^T + x_l^C\Big), \\ d_{v_e} &= v_e\Big(\mathbf{r}_3\big(x_e^L\ y_e^L\ z_e^L\big)^T + z_l^C\Big) \\ &\quad - \lambda_v\Big(\mathbf{r}_2\big(x_e^L\ y_e^L\ z_e^L\big)^T + y_l^C\Big). \end{aligned} \tag{20}$$

Having the values of d_{u_e} and d_{v_e} one can determine a translation vector of the landmark $\mathbf{l}^C = \left(x_l^C\ y_l^C\ z_l^C\right)^T$ by minimizing the sum of the squares of d_{u_e} and d_{v_e}, denoted by f_5:

$$\begin{aligned} &f_5(u_1,v_1,\ldots,u_4,v_4,x_l^C,y_l^C,z_l^C) \\ &= f_5(\mathbf{X},\boldsymbol{\Theta}) = {d_{u_1}}^2 + {d_{v_1}}^2 + \cdots + {d_{u_4}}^2 + {d_{v_4}}^2, \end{aligned} \tag{21}$$

where

$$\mathbf{X} = (u_1\ v_1\ u_2\ v_2\ u_3\ v_3\ u_4\ v_4)$$

and

$$\boldsymbol{\Theta} = (x_l^C\ y_l^C\ z_l^C).$$

$$
\begin{aligned}
&f_7\big(\hat{\mathbf{c}}^G(k+1|k),\hat{\theta}^G(k+1|k),\hat{\mathbf{l}}^G(k+1|k),\hat{\beta}^G(k+1|k),\mathbf{l}^C(k+1),\alpha^C(k+1)\big)=f_7(k+1)\\
&=\begin{pmatrix}
\begin{pmatrix} x_l^C(k+1)\\ y_l^C(k+1)\\ z_l^C(k+1)\end{pmatrix} - {}_C\hat{\mathbf{R}}^G(k+1|k)\left(\begin{pmatrix}\hat{x}_l^G(k+1|k)\\ \hat{y}_l^G(k+1|k)\\ \hat{z}_l^G(k+1|k)\end{pmatrix}-\begin{pmatrix}\hat{x}_c^G(k+1|k)\\ \hat{y}_c^G(k+1|k)\\ 0\end{pmatrix}\right)\\
\arcsin\left(\sin\left(\tfrac{1}{2}\Big(\alpha^C(k+1)-\big(\hat{\beta}^G(k+1|k)-\hat{\theta}^G(k+1|k)\big)\Big)\right)\right)
\end{pmatrix},
\end{aligned}
\tag{17}
$$

$$
\mathbf{S}(k+1)=\left[\begin{array}{c:cc}\dfrac{\partial f_7(k+1)}{\partial\hat{\mathbf{x}}} & \dfrac{\partial f_7(k+1)}{\partial\mathbf{l}^C} & \dfrac{\partial f_7(k+1)}{\partial\alpha^C}\end{array}\right]
\left[\begin{array}{c:c}\mathbf{P}(k+1|k) & \mathbf{0}\\ \hdashline \mathbf{0} & \mathbf{P}_{(\mathbf{l}^C\ \alpha^C)}(k+1)\end{array}\right]
\left[\begin{array}{c}\dfrac{\partial f_7(k+1)}{\partial\hat{\mathbf{x}}}^T\\ \hdashline \dfrac{\partial f_7(k+1)}{\partial\mathbf{l}^C}^T\\ \dfrac{\partial f_7(k+1)}{\partial\alpha^C}^T\end{array}\right]
\tag{18}
$$

At the end, the pose vector of the landmark in the global coordinates frame is established with the function f_6:

$$
\begin{aligned}
\begin{pmatrix}\mathbf{l}^G\\ \beta^G\end{pmatrix}=\begin{pmatrix}x_l^G\\ y_l^G\\ z_l^G\\ \beta^G\end{pmatrix}&=f_6(x_c^G,y_c^G,\theta^G,x_l^C,y_l^C,z_l^C,\alpha^C)\\
&=\begin{pmatrix}\begin{pmatrix}x_c^G\\ y_c^G\\ 0\end{pmatrix}+{}_G\mathbf{R}^C\begin{pmatrix}x_l^C\\ y_l^C\\ z_l^C\end{pmatrix}\\ \theta^G+\alpha^C\end{pmatrix}.
\end{aligned}
\tag{22}
$$

5.2. Uncertainty of the evaluated landmark pose. The rotation angle α^C and the translation vector $\mathbf{l}^C = \begin{pmatrix}x_l^C & y_l^C & z_l^C\end{pmatrix}^T$ of the landmark with respect to the camera coordinates frame are both calculated based on the uncertainty of the same data set $(u_1\ v_1\ u_2\ v_2\ u_3\ v_3\ u_4\ v_4)$, which are image coordinates of the four landmark corners. Thus, there is a correlation between angle and position errors. A complete covariance matrix can be established in a single step:

$$
\begin{aligned}
\mathbf{P}_{(\mathbf{l}^C\ \alpha^C)}=&\left[\begin{array}{c}\dfrac{\partial g(\mathbf{X},\boldsymbol{\Theta})}{\partial\boldsymbol{\Theta}}^{-1}\dfrac{\partial g(\mathbf{X},\boldsymbol{\Theta})}{\partial\mathbf{X}}\\ \hdashline J_{f_4}\end{array}\right]\\
&\cdot\mathbf{P}_{\mathbf{n}_l}*\left[\begin{array}{c:c}\dfrac{\partial g(\mathbf{X},\boldsymbol{\Theta})}{\partial\mathbf{X}}^T\dfrac{\partial g(\mathbf{X},\boldsymbol{\Theta})}{\partial\boldsymbol{\Theta}}^{-1} & J_{f_4}^T\end{array}\right],
\end{aligned}
\tag{23}
$$

where

$$
g(\mathbf{X},\boldsymbol{\Theta})=\frac{\partial f_5(\mathbf{X},\boldsymbol{\Theta})}{\partial\boldsymbol{\Theta}}^T,
$$

and the matrix $\mathbf{P}_{\mathbf{n}_l}$ stands for uncertainty measure of the four corners' image coordinates (it is an 8×8 matrix in the case of a single landmark). As the translation vector $\mathbf{l}^C$ is not evaluated explicitly but as a result of iterative computations, translational uncertainty in (23) is calculated according to (Haralick, 2000).

5.3. Landmark innovation vector and its uncertainty. In the case of a landmark it is wise to compare its predicted Cartesian coordinates and rotation angle with the corresponding measurement. The most frequently encountered EKF implementations calculate the innovation vector as a difference of prediction and measurement results. For point-features put on the map, such a definition of the innovation vector is used (8). For landmarks, the innovation vector is calculated with (17) as the vector function f_7. Its covariance (18) is treated as the innovation covariance. Thus, once j landmarks are visible in the image, the matrix $\mathbf{S}$ is $4j\times 4j$.

5.4. Filtering. The updating of the state-vector covariance matrix $\mathbf{P}$ is performed according to (11). In the case of landmarks, the EKF weight matrix $\mathbf{W}$ is calculated in the following way:

$$
\mathbf{W}(k+1)=\mathbf{P}(k+1|k)\,\frac{\partial f_7(k+1)}{\partial\hat{\mathbf{x}}}^T\mathbf{S}^{-1}(k+1).\tag{24}
$$

The updating of the state-vector estimate is defined with (25):

$$
\hat{\mathbf{x}}(k+1|k+1)=\hat{\mathbf{x}}(k+1|k)-\mathbf{W}(k+1)\ f_7(k+1).\tag{25}
$$

Compared with (10), in this equation the sum has been replaced by the difference. This can be explained in the following way. The estimate $\hat{x}$ enhancement based on the measurement x can be expressed as

$$
\hat{x}_{\text{new}}=\hat{x}-k\frac{\partial f(x,\hat{x})}{\partial\hat{x}}f(x,\hat{x}),\tag{26}
$$

where $f(x,\hat{x})$ is an available monotonous comparation function. Such a function attains zero once x and $\hat{x}$ are equal. The factor k is some positive coefficient, determining the strength of the $\hat{x}$ estimate enhancement. The formula (26) is similar to (25). The definiteness of the weight matrix $\mathbf{W}$ depends on the definiteness of the gradient $\partial f_7(k+1)/\partial\hat{\mathbf{x}}$ since the matrices $\mathbf{P}$ and $\mathbf{S}$ are both positive definite.

5.5. Adding a landmark to the map. In much the same way as in the case of point-features, adding a landmark to the map is performed after the update stage ((25), (11)). The function adding the landmark to the map is denoted by f_8. If the operation takes place at the k-th step, it is defined by the following formula:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathrm{new}}(k|k) &= f_8\Big(\hat{\mathbf{x}}(k|k), x_l^C(k), y_l^C(k), z_l^C(k), \alpha^C(k)\Big)\\ &= \begin{pmatrix}\hat{\mathbf{x}}(k|k)\\ \mathbf{l}^G(k)\\ \beta^G(k)\end{pmatrix}\\ &= \begin{pmatrix}\hat{\mathbf{x}}(k|k)\\ f_6(\hat{\mathbf{c}}^G, \hat{\theta}^G, x_l^C, y_l^C, z_l^C, \alpha^C)\end{pmatrix},\end{aligned} \tag{27}$$

where f_6 is defined in (22). A new covariance matrix of the system can be defined as in (28):

$$\begin{aligned}&\mathbf{P}_{\mathrm{new}}(k|k)\\ &\quad = J_{f_8}(k)\left[\begin{array}{c:c}\mathbf{P}(k|k) & \mathbf{0}\\ \hdashline \mathbf{0} & \mathbf{P}_{(\mathbf{l}^C\ \alpha^C)}(k)\end{array}\right] J_{f_8}^T(k).\end{aligned} \tag{28}$$

The complete data processing scheme for landmarks and for point-features can be easily worked out based on the formulas presented so far.

6. Simulation experiment

The goal of the computational experiments is to evaluate the impact that the presence of landmarks has on the convergence of SLAM system operation. To this end, a simulator has been designed having the ability to generate runs for a number of variables characterising SLAM system performance. The simulator also generates a graphical output visualising the system state at any selected time instant.

6.1. Simulation tools. A typical in-door test environment has been used. The dimension of the building interior is 6 × 25 m. The orthographic projections of the environment are presented in Figs. 4 and 7. The ground truth camera pose and its estimate are visualised on those figures as pairs of line-segments forming V shapes, restricting their view-fields. The real trajectory is marked with the thick, solid line. The calculated camera trajectory is marked with the thin, dotted line. It is assumed that the navigation space is delimited with vertical walls on which point-features have been distributed randomly. They are marked in Figs. 4 and 7 with grey circles. The distribution density of point-features equals one feature per 0.5 m of the wall running length on average. The height of such features is random in the range $(0 \div 1.4)$ m. The camera moves horizontally at a 0.7 m height.

With a single camera only in an unknown environment, the initialization of the SLAM system is not possible. Thus, it is assumed that at the beginning of the mission some four point-features with precisely known coordinates are visible (they are marked as circles with gray filling).

Except for point-features, some artificial landmarks were placed in the environment. In the experiment under consideration, a single landmark was fixed to the right-most wall. In Fig. 7 it is marked as a thick black spot. It is assumed that the system is able to extract point-features from images acquired during the mission, as well as to recognise an encountered landmark. In Figs. 4 and 7, the main axes of the uncertainty ellipsoids of the objects location are represented with black line segments.

In Fig. 5, working step numbers of the system are marked along the real trajectory of the camera. The trial trajectory is the following: the camera moves to the right along the wall up to the distance of roughly 20 m and then (without turning) it is withdrawn approximately to the starting position.

In Figs. 6 and 8, some variables of the SLAM system are visualised as a function of the time of navigation. There are four groups of variables:

(a) the first one, representing the x-coordinates of the real position, real velocity vector of the camera and their estimates (which are visualized indirectly as trajectory envelopes $\pm 3\sigma$ wide[3]),

(b) the second one, comprising the y-coordinates of the position and velocity vector and their uncertainty bands,

(c) the third one, consisting of the real rotation angle and angular velocity of the camera as well as of their estimated values returned by SLAM (marked also as $\pm 3\sigma$ band),

(d) the fourth one, showing some additional parameters characterising the operation of the system, such as the number of point-features recognised from the current image, as well as the number of false negatives[4], time instants at which a new point-feature has been put on the map[5], the number of point-features kept in the map and the time of the maintaince of the earliest point-feature in the set of currently visible point-features.Moreover, in Fig. 8, the number

[3] It is worth noting that position uncertainty in the experiment under study is relatively small. In the case of the y-coordinate uncertainty the band is as narrow as it is invisible on the graph—it is covered by a thick run of a true y-coordinate.

[4] The number of unrecognised point-features in Figs. 6 and 8 is shown as a negative value of the Y-axis.

[5] The system is able to put a single point-feature on the map within a single iteration step. During on-line operation 25 iterations per second are performed to keep up with the TV frame frequency standard.

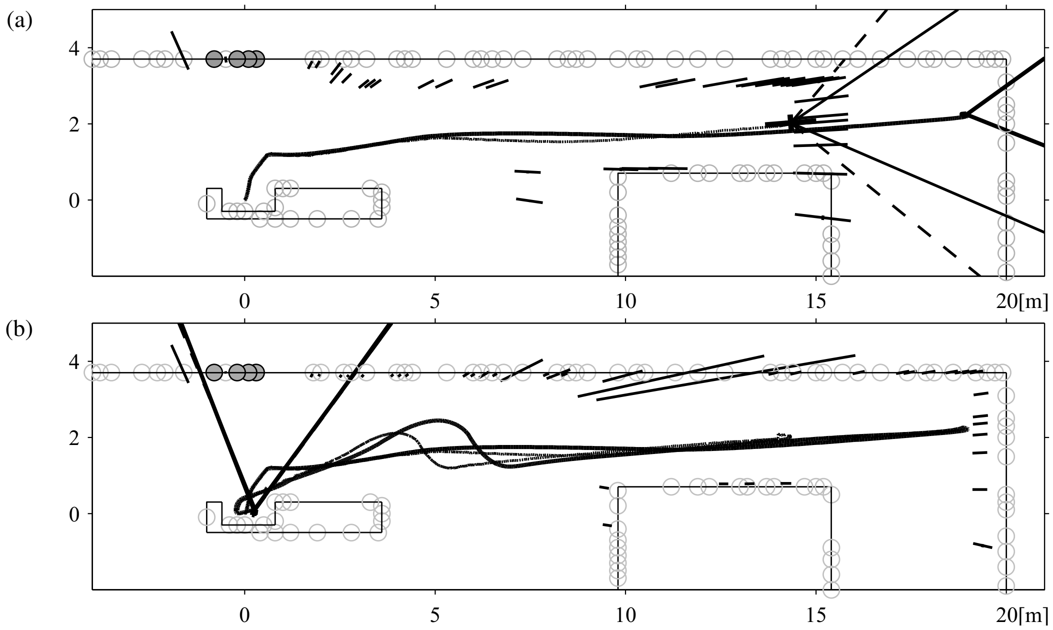


Fig. 4. Orthographic projections of the system state at the 920-th (a) and the 2000-th (b) simulation step (without landmarks). The navigation space is delimited with vertical walls (broken lines). Point-features randomly distributed on walls are marked with grey circles. The ground truth camera pose and its estimate are visualised as pairs of line-segments forming V shapes, restricting their view-fields. The real and calculated camera trajectories are marked with a thick solid line and a dotted line. The main axes of the objects' location uncertainty ellipsoids are marked with black line-segments.

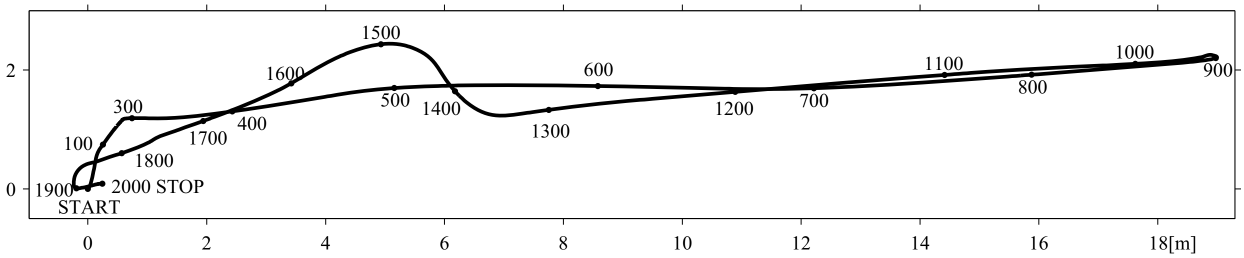


Fig. 5. Real camera trajectory with marked filtration steps numbers.

of recognised landmarks at a given time instant and the number of landmarks already put on the map is shown.

Some of the values have been rescalled to make the graph more readable. Scaling factors are described in Figs. 6 and 8 with comments accompanying the appropriate lines.

Having such a characteristic of SLAM behaviour, one can draw a number of conclusions. For example, while coming back and accounting for features that have been put on the map earlier, the live-span of the oldest extracted feature can be a good measure of the system ability to recognise the locations already visited before. A similar role can be played by a number of recognised/unrecognised point-features as well as the need of new point-feature initialisation.[6] The advantage of the simulation method is also that it can be used to judge the ability of trajectory reconstruction while comparing ground truth data with the estimated ones. This is frequently not possible with real systems if there is no real position of the camera data available. In particular, the obtained results demonstrate the discrepancies between the real trajectory of the camera and the uncertainty bounds around the SLAM-estimated trajectory.

The next two subsections present simulation results for two cases: one with only *natural* point-features accounted for and one with the landmark introduced to the environment. The simulations are carried out according to the same scenario, i.e., the trial trajectory to pass is identical, the distribution of point-features is the same and the same noise disturbs the system in both cases.

[6] The SLAM system attempts to add a new point-feature to the map whenever less than eight point-features have been recognised on the image from those already on the map.

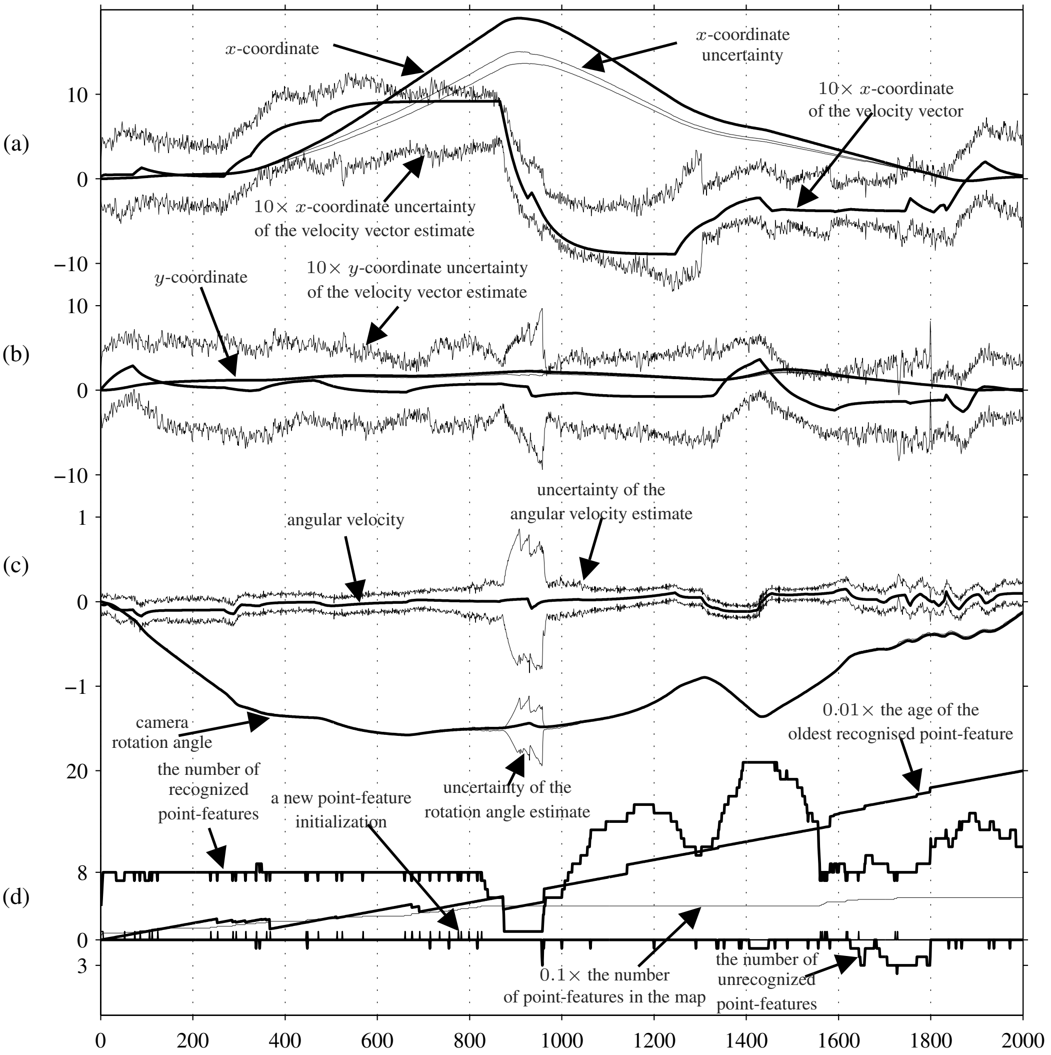


Fig. 6. Runs of real and estimated variables of the SLAM system in the absence of landmarks: x-coordinates (a) and y-coordinates (b) of the camera position/velocity vector; rotation angle and angular velocity of the camera (c). Estimates are visualized indirectly as trajectory envelopes $\pm 3\sigma$. Additional parameters characterising the operation of the system (d).

6.2. Testing the SLAM system without landmarks.

In Fig. 4(a) the system state at the 920-th simulation step is displayed, namely, the distribution of point-features put on the map so far and the camera pose. Two camera models are visible: the right-most one represents the real camera and the second one (closer to the starting point) represents the estimated pose of the camera. Dotted lines marked outside the nominal field of view of the *estimated camera* represent the view-field enlargement caused by the $\pm 3\sigma$ orientation uncertainty of the camera. At the 920-th simulation step, only a single point-feature appeared on the image, and thus the camera orientation uncertainty quickly grows. In turn, position uncertainty ellipsoids for the camera and for point-features are symbolically represented by their main axes representing the $\pm 3\sigma$ range. In the case of the camera, thick lines are visible in the apex of a V shape. Most frequently only a single axis is visible as the second one is too short.

In Fig. 4(b) the system state at the 2000-th simulation step is displayed (being a final step). From Fig. 6 it can be noticed that the final estimate of the camera pose equals the real pose and the earliest point-features put on the map have been recognized correctly. It is worth noting that, as a consequence of recognising the *old* point-features already present on the map, the system is able to correct the position estimates of all point-features put on the map. Thus the resulting map at this time step is better than the one available in the middle of the mission.

6.3. Testing the SLAM system in an environment with landmarks.

In Fig. 7(a) the SLAM system state at the 920-th step is visualized, while the system is able to re-

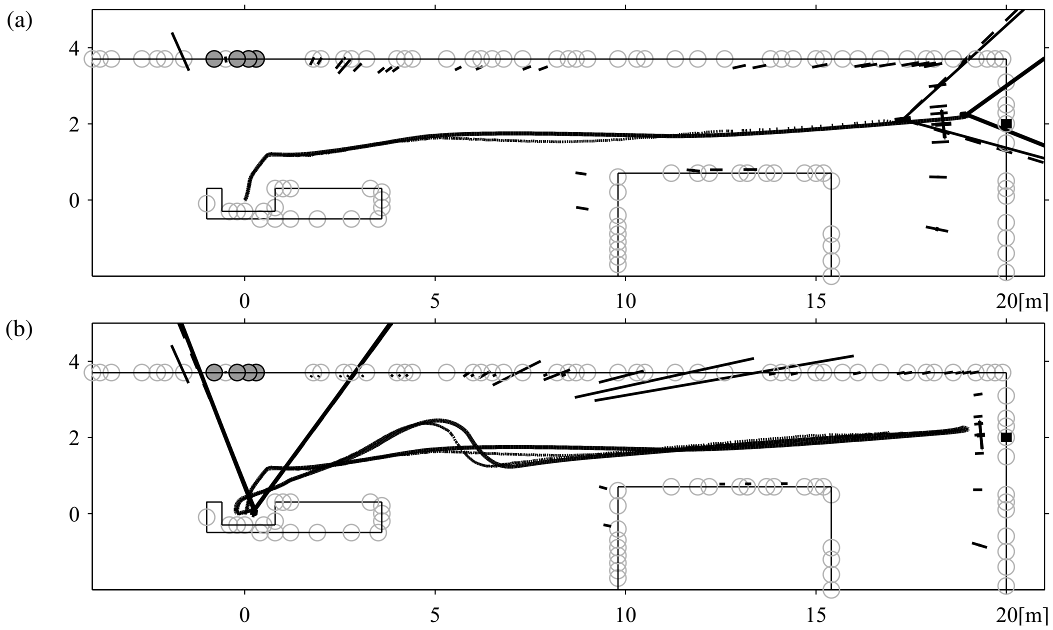


Fig. 7. Orthographic projections of the system state at the 920-th (a) and the 2000-th (b) simulation step (with a landmark). The navigation space is delimited with vertical walls (broken lines). A single landmark fixed to the rightmost wall is marked as a thick black spot. Point-features randomly distributed on walls are marked with grey circles. The ground truth camera pose and its estimate are visualised as pairs of line-segments forming V shapes, restricting their view-fields. The real and calculated camera trajectories are marked with a thick solid line and a dotted line. The main axes of the objects' location uncertainty ellipsoids are marked with black line-segments.

cognize the landmark and to put it on the map. The obtained map is much better than the one shown in Fig. 4(a). The estimated camera position and the estimated point-features coordinates are closer here to the ground truth data than in the previous simulation experiment. This is due to the appearance of the landmark within the camera view-field, which allows the more efficient estimation of camera velocity. Due to the visibility of the landmark, the navigation system implicitly receives the information on the underestimation of camera velocity (which was based on point-features data only so far). It is interpreted in the system as the calculated position of the camera is more advanced than the value of the position estimate available before the recognition of the landmark. As a consequence, the camera position as well as the position of all point-features already on the map are corrected. Such system operation is visible between 800-th and 900-th simulation steps in Fig. 8. Especially the correction of the camera x-coordinate estimate is remarkable. Surprisingly enough, the recognition of the landmark results in a camera position estimate uncertainty decrease despite the growing distance of the camera from the starting point of the trajectory. However, at the same time period, while the landmark was absent (Fig. 6), an excessive rise in the camera orientation and y-coordinate uncertainty were observed.

The next remark is that in Fig. 6 between the 1600-th and 1800-th simulation steps one can observe a number of false negatives, while in the case with landmarks (Fig. 8) during the same time period the system is able to recognise almost all the points that are in the field of view and which were previously placed on the map. Such an improvement is a consequence of better trajectory reconstruction, especially for the x-coordinate of the camera.

In Fig. 7(b) the system state is visualized at the 2000-th simulation step together with the real and estimated camera motion trajectory. It is evident that in this case, while the system is able to recognize the landmark and to put it on the map, the system-estimated trajectory is more close to the real one. Also the resulting map is of better quality than the one presented in Fig. 4(b), namely, some uncertainty elipsoids have the main axes shorter.

7. Conclusions

The reported experiments demonstrate the advantages of embedding landmark measurement into the filtration process at frame-rate frequency. By implementing the landmark-based navigation in the visual SLAM system one is able to get rid of the camera position estimate drift. Here we have aimed at treating planar patterns as a new class of environment objects and, as such, to include them in the map. So the SLAM state vector containing a representation of two categories of features is maintained and within the same system measurements in the image space for the first and in 3D space for the second feature type are performed. Also a scheme of image coordinates un-

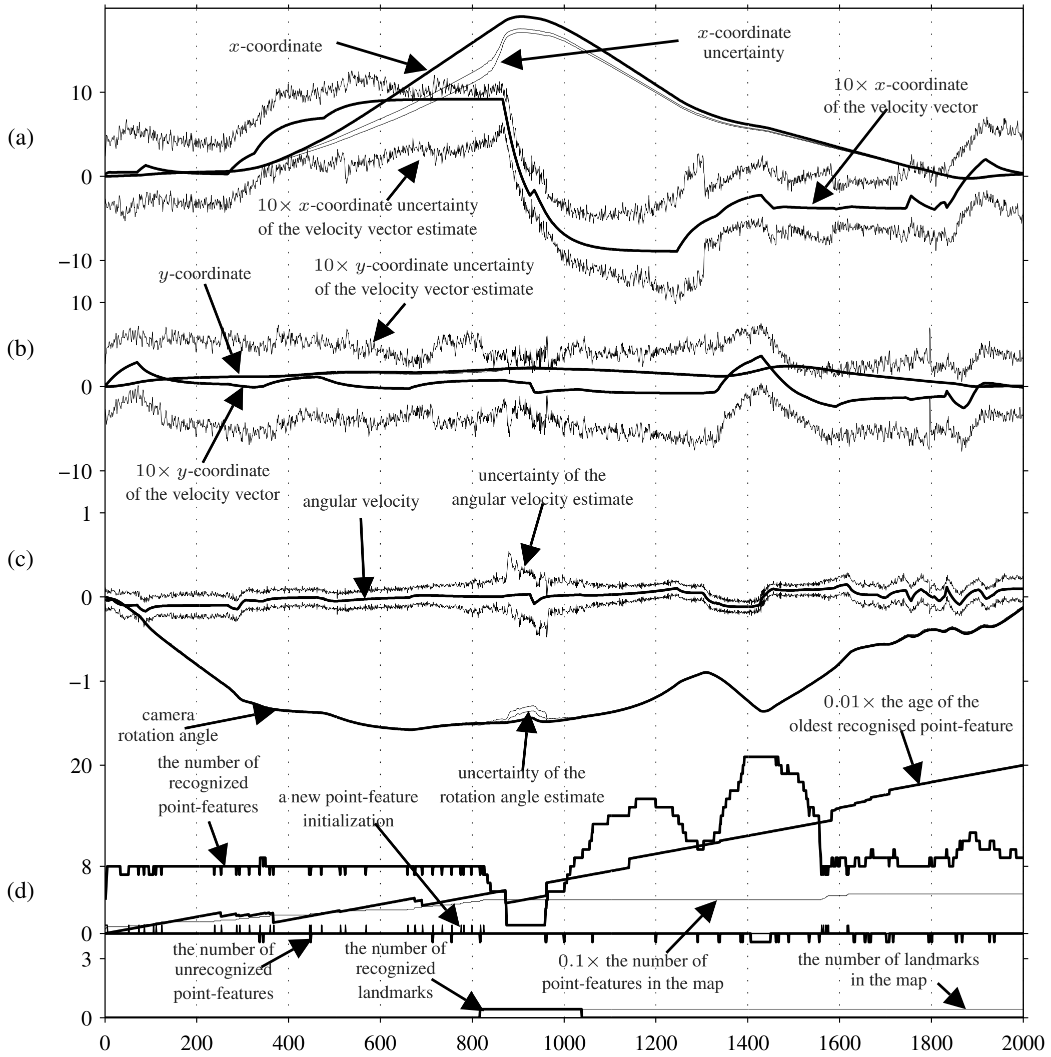


Fig. 8. Runs of real and estimated variables of the SLAM system in the case with a landmark: x-coordinates (a) and y-coordinates (b) of the camera position/velocity vector, rotation angle and angular velocity of the camera (c). Estimates are visualized indirectly as trajectory envelopes $\pm 3\sigma$. Additional parameters characterising the operation of the system (d).

certainty propagation while calculating landmark position uncertainty with respect to the camera reference frame has been proposed.

The only inconvenience of the method is that one has to place the landmarks in the environment before the mission. However, a possible solution is to provide a navigating robot with facility for distributing the landmarks.

One of important future research directions is to investigate a map management scheme in which all landmarks will be kept in the map as navigation beacons but old point-features will be removed in order to constrain the excessive expansion of the state-vector size. The map including more complex navigational objects (landmarks) can be sparser, being at the same time a more precise reference than the map generated with the SLAM system accounting for point-features only.

Acknowledgment

This work was founded by Grant No. N514 294635 of the Ministry of Science and Higher Education for the years 2008–2010.

References

Bailey, T. and Durrant-Whyte, H. (2006). Simultaneous localization and mapping (SLAM): Part II, *IEEE Robotics & Automation Magazine* **13**(3): 108–117.

Bar-Shalom, Y., Kirubarajan, T. and Li, X.-R. (2002). *Estimation with Applications to Tracking and Navigation*, John Wiley & Sons, Inc., New York, NY.

Bączyk, R., Kasiński, A. and Skrzypczyński, P. (2003). Vision-based mobile robot localization with simple artificial landmarks, *7th International IFAC Symposium on Robot Control (SYROCO), Wrocław, Poland*, pp. 217–222.

Castle, R.O., Gawley, D.J., Klein, G. and Murray, D.W. (2007a). Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras, *International Conference on Robotics and Automation (ICRA), Rome, Italy,* pp. 4102–4107.

Castle, R.O., Gawley, D.J., Klein, G. and Murray, D.W. (2007b). Video-rate recognition and localization for wearable cameras, *Proceedings of the 18th British Machine Vision Conference (BMVC), Warwick, UK,* pp. 1100–1109.

Civera, J., Davison, A. and Montiel, J. (2008). Inverse depth parametrization for monocular SLAM, *IEEE Transactions on Robotics* **24**(5): 932–945.

Clemente, L.A., Davison, A., Reid, I., Neira, J. and Tardos, J. (2007). Mapping large loops with a single hand-held camera, *Robotics Science and Systems (RSS)*, Georgia Institute of Technology, Atlanta, GA.

Davison, A. (2003). Real-time simultaneous localisation and mapping with a single camera, *Ninth IEEE International Conference on Computer Vision (ICCV), Nice, France*, Vol. 2, pp. 1403–1410.

Davison, A.J. and Murray, D.W. (2002). Simultaneous localisation and map-building using active vision, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(7): 865–880.

Davison, A.J., Reid, I.D., Molton, N.D. and Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **29**(6): 1052–1067.

Dissanayake, G., Newman, P., Clark, S., Durrant-Whyte, H.F. and Csorba, M. (2001). A solution to the simultaneous localization and map building (SLAM) problem, *IEEE Transactions on Robotics and Automation* **17**(2): 229–241.

Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localization and mapping: Part I, *IEEE Robotics & Automation Magazine* **13**(2): 99–110.

Eade, E. and Drummond, T. (2009). Edge landmarks in monocular SLAM, *Image and Vision Computing* **27**(5): 588–596.

Gee, A.P., Chekhlov, D., Calway, A. and Mayol-Cuevas, W. (2008). Discovering higher level structure in visual SLAM, *IEEE Transactions on Robotics* **24**(5): 980–990.

Gee, A.P. and Mayol-Cuevas, W. (2006). Real-time model-based SLAM using line segments, *2nd International Symposium on Visual Computing*, *Lake Tahoe, NV, USA,* pp. 354–363.

Haralick, R.M. (2000). Performance Characterization in Computer Vision, *in* R. Klette, H.S. Stiehl, M.A. Viergever, K.L. Vincken (Eds.) *Proceedings of the Theoretical Foundations of Computer Vision, TFCV on Performance Characterization in Computer Vision*, Kluwer, B.V., Deventer, pp. 95–114.

Neira, J., Davison, A.J. and Leonard, J.J. (2008). Guest editorial special issue on visual SLAM, *IEEE Transactions on Robotics* **24**(5): 929–931.

Smith, P., Reid, I. and Davison, A.J. (2006). Real-time monocular SLAM with straight lines, *British Machine Vision Conference (BMVC), Edinburgh, UK,* Vol. 1, pp. 17–26.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C.M., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G.R., Davies, B., Ettinger, S., Kaehler, A., Nefian, A.V. and Mahoney, P. (2006). Stanley: The robot that won the DARPA Grand Challenge, *Journal of Field Robotics* **23**(9): 661–692.

Andrzej Kasiński, Ph.D., D.Sc., graduated from the Poznań University of Technology in electrical engineering (1973) and Adam Mickiewicz University in mathematics (1974). He is a professor and the head of the Institute of Control and Information Engineering, Poznań University of Technology. He has authored over 130 technical papers and book chapters in the field of control engineering, computer simulation, digital systems design, robotics and computer vision. His current research focuses on image analysis, pattern recognition, multiagent systems and biocybernetics. He has held visiting positions at Cornell University (USA), the Delft University of Technology (Netherlands), and the University of Murcia (Spain).

Robert Bączyk received the M.Eng. degree in 1998 and the Ph.D. degree in 2008, both in robotics and automation, from the Poznań University of Technology, Poland. His research interests are practical applications of vision systems, camera calibration and optical distortions modelling, image processing and vision-based systems for autonomous robots.

Received: 7 July 2009
Revised: 30 November 2009