AerospaceEurope Conference2021
WARSAW, POLAND | 23 -26 NOVEMBER 2021
CEAS Council of European Aerospace Societies

Simultaneous Localization and Mapping (SLAM) Problem – description of selected algorithm

Tomasz Małecki[1], Janusz Narkiewicz[2]

[1] Łukasiewicz – Institute of Aviation, Engineering Design Center
[2] Warsaw University of Technology, Faculty of Power and Aeronautical Engineering, Department of Automation and Aeronautical Systems

Abstract

A simultaneous localization and mapping (SLAM) algorithm was developed with a prospect of an on-line application for mobile platforms. The task of the algorithm is to search and map the operation area, avoiding contact with obstacles. The theoretical background of the problem, the model of the system and the operation scenario are described. The algorithm, which was implemented in MATLAB® software, is based on a linear discrete-time state transition model for determination of the? platform position and orientation, and a 'force' points method for collision avoidance and defining the next-step of platform motion. The uniform2D motion of a rigid platform operating in a confined space filled with landmarks of polygonal shape was considered. An ideal operation of a single on-board laser type sensor was assumed. The developed solution was tested via simulations to determine its applicability and to indicate directions for future work. It is expected that the proposed approach might be applicable in the cases of real-time applications with availability of limited computational resources.

Keywords: algorithm, modelling, path planning, SLAM.

List of symbols

d	max. sensor range
$F_v(t)$	state transition matrix
i	discrete index
n_m	number of measurements
p	point
p_f	'force point'
R	rotation matrix
s	distance travelled
t	discrete time index
$u_v(t)$	vector of control inputs
v_0	velocity of the vehicle
$v_v(t)$	vector of temporally uncorrelated process noise errors with zero mean and covariance
x_0,y_0	initial point; starting point
x_G,y_G	global reference frame coordinates
$x_v(t)$	state of the vehicle at a discrete time t
X_v	history of states
x_l,y_l	landmark coordinates
x_L,y_L	local reference frame coordinates
x_w,y_w	2D space points coordinates
α	measurement angle
θ	steering angle

1. Introduction

Simultaneous localization and mapping (SLAM) is a process of estimating in real time the structure of the surrounding environment perceived by sensors placed on a moving platform, creating a map and simultaneously providing the platform position on it. A mobile platform moves inside an unknown space, acquires data, builds a model of the surrounding environment and simultaneously localizes itself within it. The SLAM operation is applied on mobile platforms on which usually at least one sensor is mounted to gather data about its surroundings and algorithms embedded in processing equipment [1]. The main sources of SLAM data are usually image sensors: visual light cameras, radars, lidars, sonars. To support SLAM process additional data may be acquired by other sensors, especially those providing the platform position (GNSS, wheel encoders, INS) in an absolute reference system [2].
At the heart of the SLAM challenge is the recognition that localization and mapping are coupled issues when two quantities (platform location and environment mapping) are deduced from the same sensors' measurements. Thus, a solution can be obtained only if the mapping and localization process are considered together. The SLAM problem is widely described in the literature [1–8].

The automated SLAM consists of three basic factors which are crucial for the success of the process [3]:

1. A platform motion (also known as 'motion model') – a mathematical model of a mobile platform (agent) motion, allowing determination of the actual position.
2. Landmark localization and mapping (sometimes called an 'inverse observation model') – discovering and recognizing distinct features (landmarks) in the environment which are to be incorporated into the map; in the scene the status and positions of the newly discovered landmarks are determined from the data acquired by the sensors.
3. Landmark observation – the positions of landmarks which have already been mapped are used to improve self-localization of the platform and localization of other landmarks in the environment; a landmark that has already been mapped is used to obtain a measured position (so called 'direct observation model'). As a result, uncertainties of landmarks localization decrease.

Usually, the process is performed in a discreet way, in sequential time steps. The results of these operations strongly depend on sensor measurement errors, which (if not eliminated or decreased) may lead to an unacceptable increase in uncertainties of the SLAM in time. Models of these operations combined with position estimators allow construction of an automated, effective solution to SLAM process [6]. The position estimator is responsible for diminishing propagation of uncertainties in time, so very often filtering (as Kalman or particle type) is used [3].

Uncertainties and disturbances of platform motion and in determination of the position of the vehicle and the landmarks resulting from non-ideal sensor observation decrease reliability of the system (accuracy and precision of the results of the algorithm) and may lead to spurious results. In fact, all the factors in the process should be evaluated taking into account an influence of disturbances which then have to be filtrated.
In some papers it was questioned whether the Kalman filter framework is a tool robust and rigorous enough to map the stochastic environment [9]. It was shown that linearization errors may produce inconsistency problems in Extended Kalman Filter (EKF) solution for SLAM [8]. On the other hand, the linearization errors of nonlinear system are considered to be an inherent and unavoidable issue which might be mitigated but cannot be eliminated in the long term [8].
Another aspect is the scaling properties of the stochastic map solution to the SLAM problem. The basic solution would have $O(n^2)$ complexity, where n is the number of features in the map [4]. Some techniques were developed to reduce the required computational load and to support scalability of the solution, for instance the sparse extended information filters [7], the decoupled stochastic mapping [10] or the sequential map joining [11]. However, the linearization problem and the data association problem ambiguities are not completely solved [4].

Having in mind these challenges, the objective of this study is to examine an alternative and simplified solution to the SLAM problem. This approach is expected to reduce the complex nonlinear problem to a discretized and linearized one which may be solved efficiently. This approach may be applicable in real-time applications when memory and computational resources are of the primary concern.

2. SLAM process

The SLAM model in this research incorporates a vehicle motion model, which travels within an environment starting from an unknown location, and which is a confined 2D space p containing a population of features (landmarks). The vehicle is equipped with a generic sensor that can perform measurements of the relative location between an individual landmark and the vehicle itself. The absolute locations of the landmarks are not available.

The main task of the system is to perform mapping of an unknown environment with simultaneous localization of a vehicle within the constructed map of a defined space. A linear (synchronous) discrete-time model of the evolution of the vehicle and the observations of landmarks is adopted.

In this study a 2D platform motion is considered. The environment to be mapped is a closed polyline (in the global coordinate system) which defines borders for platform motion. The non-moving landmarks / obstacles are placed inside the environment area and modelled as convex polygons bounded by a closed polyline.

The motion of a selected point of a vehicle (the center of the vehicle) is considered. A starting point of its motion is chosen arbitrarily inside the space map.

A laser-type sensor located in the center of the vehicle with a limited range of observation is used. Inertial navigation is used additionally to obtain the platform position coordinates and velocity components at each time step. The laser sensor measurements are performed instantaneously (with no time delay) in all directions once for each time step. No sensor errors are assumed at this stage of the system development. Thus, measurements are considered perfect and the landmarks positions may be uniquely defined. A distance between the vehicle and previously identified space borders or landmark is used to modify (correct) the direction of the vehicle movement.

The final result of the system operation is a set of identified points of the space and landmark borders, which could be used to estimate the shape of the evaluated features. The complete description of the state of the system would consist of the position and orientation of the vehicle together with the position of all the landmarks.

In a real navigation problem, a vehicle motion and observations of landmarks are better described by nonlinear and asynchronous models. Yet, the use of linear synchronous models seems to be a reasonable choice here, assuming small increments of the state variation between the two steps.

The global coordinate frame is denoted by G, and L denotes a local Cartesian frame fixed to the moving platform, defined with respect to the global frame by a translation vector x_v and a steering angle θ representing the direction of the vehicle velocity vector v. The space borders are defined in a global reference frame. An origin of the local reference frame is the actual position of the vehicle.

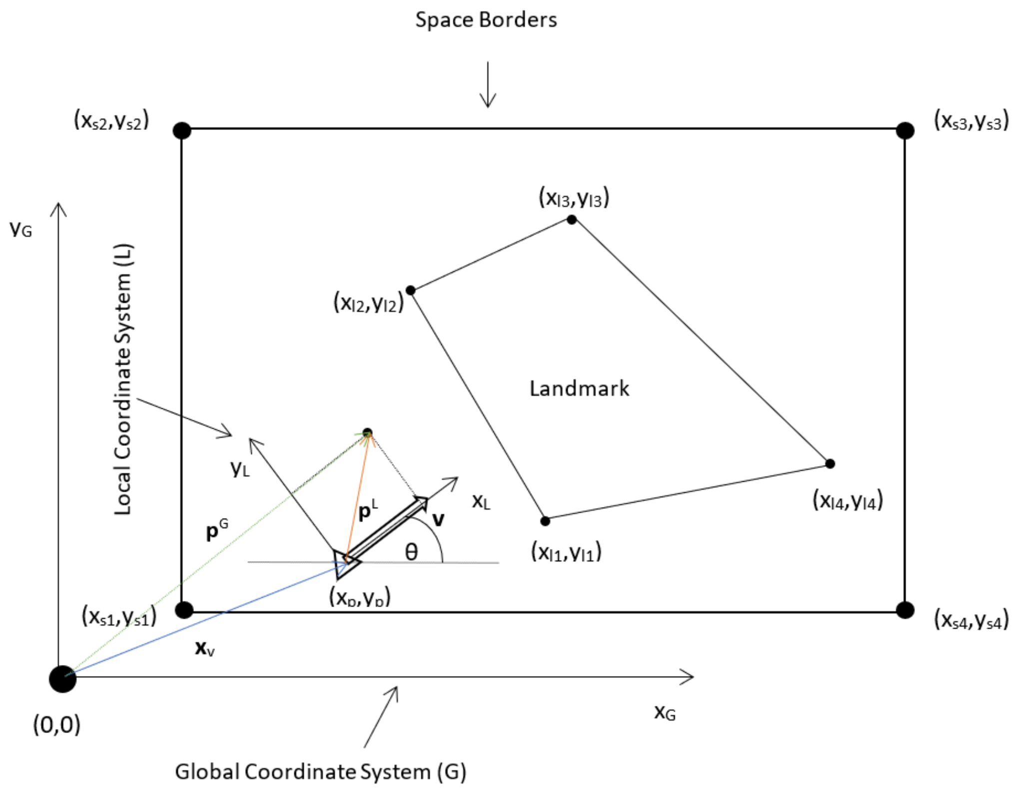

Figure 1 – Space definition and frames transformation.

A position of a vehicle in space p can be expressed in the global frame G or in the local frame L, related by transformation equations:

$$\mathbf{p}^{G} = \mathbf{R}\mathbf{p}^{L} + \mathbf{x}_{v} \qquad \text{from frame L to G} \tag{1}$$

$$\mathbf{p}^{L} = \mathbf{R}^{T}\left(\mathbf{p}^{G} - \mathbf{x}_{v}\right) \qquad \text{from frame G to L} \tag{2}$$

where **R** is the rotation matrix associated with the steering angle θ (2D case):

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{3}$$

Then position coordinates x and y in the i^th measurement step in the global reference frame are calculated according to the following formulas:

$$x_i^G = x_0 + x_i^L \cdot \cos(\theta) - y_i^L \cdot \sin(\theta) \tag{4}$$

$$y_i^G = y_0 + x_i^L \cdot \sin(\theta) + y_i^L \cdot \cos(\theta) \tag{5}$$

The motion of the vehicle through the environment is described by a linear discrete-time state transition equation of the form:

$$x_v(t+1) = F_v(t)x_v(t) + u_v(t+1) + v_v(t+1) \tag{6}$$

The state vector of the vehicle at a time step t is denoted by $x_v(t)$. A platform state transition matrix is denoted by $F_v(t)$ (which for the purpose of this study simplifies to the identity matrix) and $u_v(t)$ is a vector of control inputs at a given time t. The vector of temporally uncorrelated process noise (each observation is

independent of the previous observations) with the zero mean and constant covariance (white noise) would be $v_v(t) = 0$.
The platform velocity vector is assumed to be collinear with the vehicle local coordinate system x axis (Fig.2). The vehicle is controlled by changing a steering angle θ, at each time step according to the collision avoidance algorithm. The position of the vehicle in the time step t+1 is obtained using the following formulas:

$$x_{t+1}^{G} = x_{t}^{G} + v_0 \cdot dt \cdot \cos(\theta) \quad (7)$$
$$y_{t+1}^{G} = y_{t}^{G} + v_0 \cdot dt \cdot \sin(\theta) \quad (8)$$

The criterion for the end of vehicle operation is the distance traveled, which imitates the limited power available on board the vehicle. The total travelled distance in time step t+1 is:

$$s_{t+1} = s_t + vt \cdot dt \quad (9)$$

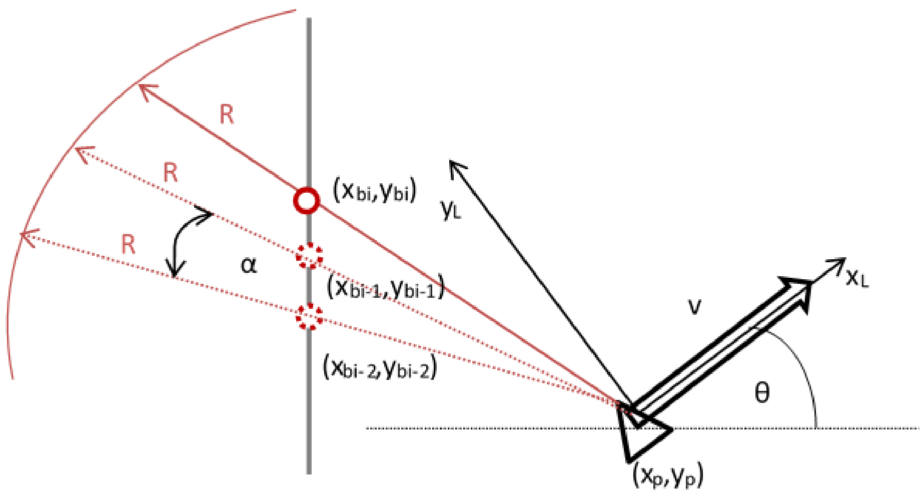

Figure 2 – Scheme of obstacle points determination.

The observation is modelled as a laser – type sensor, located in the center of the vehicle, which transmits uniformly distributed beams in a circular manner. The frequency of the beam generation is related to the beam angle α measured in the local vehicle system L. The frequency of measurements performed at each time step in real applications should be adjusted by decreasing α angle (Fig.2). For the total number of sensor beams denoted by n_m:

$$\alpha = \frac{360°}{n_m} = \frac{2 \cdot \pi}{n_m} \quad (10)$$

The coordinates of the points on the boundary of the sensor range corresponding to the i^{th} sensor beam could be evaluated as:

$$x_i = R \cdot \cos\left(\frac{2 \cdot \pi \cdot (i-1)}{n_m}\right) \quad (11)$$
$$y_i = R \cdot \sin\left(\frac{2 \cdot \pi \cdot (i-1)}{n_m}\right) \quad (12)$$

where: R is max. sensor range and i = 1, …,n_m.

When the laser beam encounters an obstacle in its path i.e., the laser beam crosses a line defining a landmark or a space border (which means that the intersection condition is fulfilled), consecutive points coordinates are added to the local measurement vector, which then could be transferred into the global reference frame and added to the global map. For this purpose, the dedicated function 'polyxpoly' from Matlab Mapping Toolbox was used.
A crucial functionality of the system is to 'sense' the proximity of a landmark or space border and to change the steering angle in a manner that assures collision avoidance. To fulfil this requirement some 'dynamic equivalent' was used to calculate the 'reaction force' that would be the measure of external body vicinity (it is worth noticing that dynamic similarity is not exact, because the vehicle is weightless and dimensionless and no 'true' force acts on it). The approach is similar to [12] but the proposed concept is simpler and inherently less computational resources demanding.
The first step is to extract significant points from the point of view of the vehicle. The radius r_f is a distance between the specified point of the already known position of an obstacle (x_s,y_s) and the actual position (x_a,y_a) of the vehicle:

$$r_f = \sqrt{(x_s - x_a)^2 + (y_s - y_a)^2} \tag{13}$$

It is calculated for all known points on the vehicle map. Then appropriate points (so called 'force points') are chosen for which r_f is smaller than or equal to a predefined 'force range'. Subsequently, a direct application of the next step prediction has to be done. If there is only one 'force point' on the measurement line at a given time step, the direction of the 'reaction force' vector is aligned with the measurement radius but has the opposite sign to the laser beam propagation vector. In the case of multiple points on different measurement radiuses, the resultant 'reaction force' is the sum of all the particular vectors at the given time step (Fig.3).

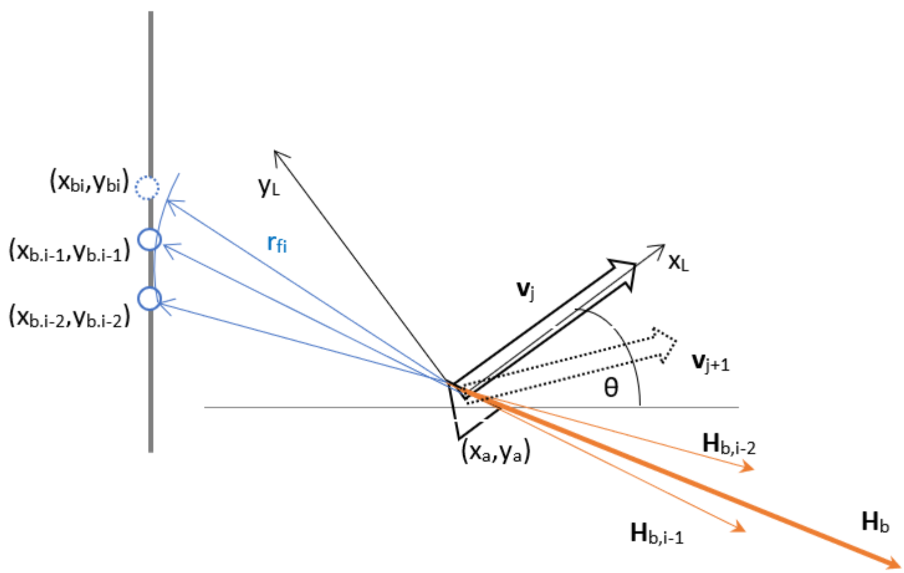

Figure 3 – Scheme of 'reaction force' idea.

Then, consolidated 'force point' coordinates p_x and p_y are calculated recursively taking into account all the intersection points indexed by 'i':

$$p_{x(i+1)} = p_{xi} - \frac{x_{bi}-x_a}{|x_{bi}-x_a|} \cdot \frac{1}{r_{fi}{}^2} = p_{xi} - \frac{x_{bi}-x_a}{|x_{bi}-x_a|} \cdot \frac{1}{(x_{bi}-x_a)^2+(y_{bi}-y_a)^2} \tag{14}$$

$$p_{y(i+1)} = p_{yi} - \frac{y_{bi}-y_a}{|y_{bi}-y_a|} \cdot \frac{1}{{r_{fi}}^2} = p_{yi} - \frac{y_{bi}-y_a}{|y_{bi}-y_a|} \cdot \frac{1}{(x_{bi}-x_a)^2+(y_{bi}-y_a)^2} \quad (15)$$

The resultant length of the 'repulsive' vector |***H**$_b$*| is calculated as follows:

$$|\mathbf{H}_\mathrm{b}| = \sqrt{(p_x)^2 + (p_y)^2} \quad (16)$$

If calculated |***H**$_b$*| = *0*, then |***H**$_b$*| is set to 1 in order to avoid division by 0 in the subsequent step. The predicted coordinates are calculated as:

$$p_{x\_pred} = \frac{p_x}{|\mathbf{H}_\mathrm{b}|} + \cos\theta \quad (17)$$

$$p_{y\_pred} = \frac{p_y}{|\mathbf{H}_\mathrm{b}|} + sin\,\theta \quad (18)$$

Then, the steering angle is evaluated based on the following equation:

$$\theta = \tan^{-1}\left(\frac{p_{y\_pred}}{p_{x\_pred}}\right) \quad (19)$$

Finally, the updated vehicle position is calculated based on equations (6) and (7) with the steering angle calculated as per eq. (19).

3. Implementation

An algorithm resulting from the considerations above is presented as a block diagram in Figure 4. For the purpose of a computer simulation the user defines the space (boundaries) for vehicle operation and the starting point within this space. All relevant system variables are initialized to allow computing of the vehicle position and orientation. The sensor range points are calculated using eq. (10), (11) and (12) in the local reference frame and then transformed into the global reference frame using eq. (4) and (5), and eventually added to the vehicle map. The? intersection between the sensor line and space/landmark borders is determined and the 'force points' radiuses are calculated using eq. (13), if at least one intersection point is present. Subsequently, the next step prediction is performed by evaluating the resultant 'force point' position (eq. (14) and (15)), calculating the length of the 'repulsive' vector (eq. (16)), computing the predicted position coordinates using eq. (17), (18) and determining the steering angle for the next time step (19). Finally, the vehicle position is calculated using eq. (7) and (8) based on the previously calculated values. The entire sequence of operations is repeated until the maximum range of the vehicle is reached (an equivalent to running out of fuel/battery).

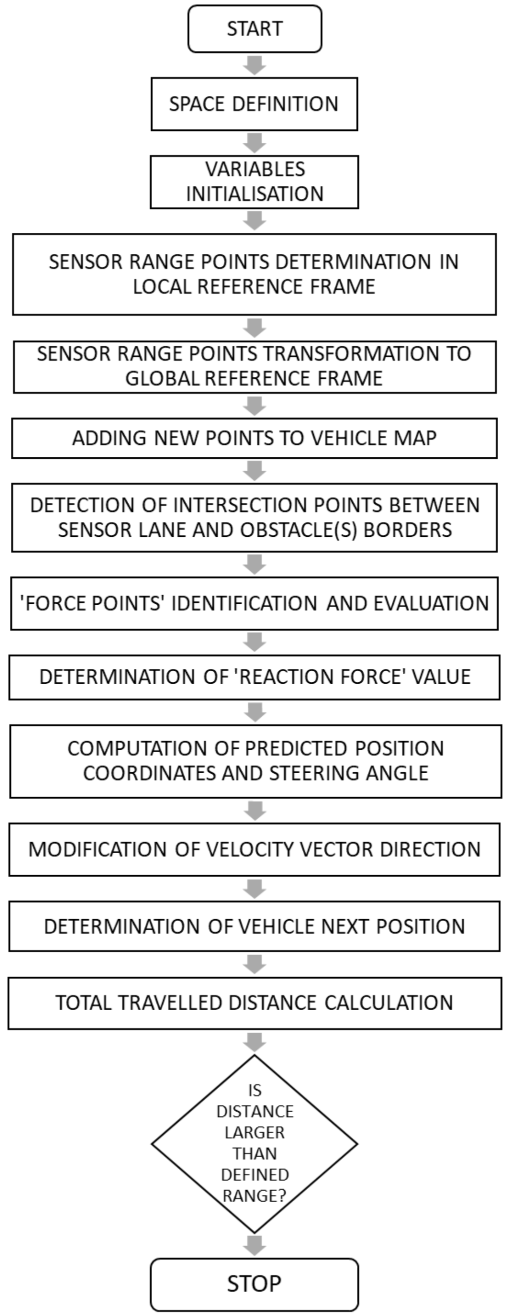

Figure 4 – Algorithm process diagram.

4. System simulation

There are two main group of simulation cases presented here, illustrating efficiency of the method. The input data in Table 1 was used for the purpose of simulation.

Table 1 – Variables used throughout simulation.

Variable name	Value	Comments
space_borders	[0 0; 0 5; 10 5; 10 0; 0 0]	coordinates of space borders

land_def	[5 2; 3 3; 7 2; 6 1; 5 2]	coordinates of landmark borders
spx	0.5	starting point x coordinate
spy	0.5	starting point y coordinate
v0	1	vehicle velocity [m/s] (constant)
theta	pi/2	initial steering angle in global coordinate system
range	1	sensor measurement range [m]
fs	5*v0/range	sampling frequency [1/s]
dt	1/fs	sampling time – how often measurements are performed [s]
s	0	actual length of travelled distance
smax	150	max. distance – depends e.g., on the battery endurance [m]
n_measure	36	number of measurements on azimuth; here performed every 10 degrees

4.1 A program launch for normal conditions

The initial conditions for the simulation were chosen to facilitate algorithm performance:

1. The starting point was within the operational space.
2. The distance between the space borders and obstacle were relatively large (no narrow passages).
3. The landmark was thick.

The output from the program is:

1.) A global measurements array – a two-column matrix of the identified space and landmark border points in the global coordinate system obtained during the operation of the vehicle.
2.) A vehicle trajectory – a two-column matrix of the registered vehicle positions in the global coordinate system (a movement history array).

The results of the example program execution are shown in Fig.5.

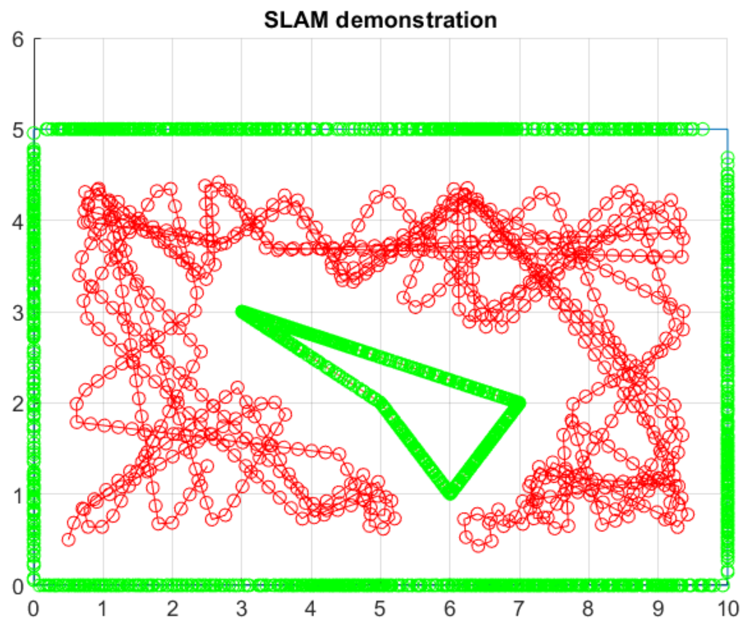

Figure 5 – Result of program launching under normal operation conditions.

4.2 A program launch for special cases.

To confirm applicability and limitations of the program some specific cases were considered:
a) A dummy starting point – the vehicle starts either outside/on the border of the defined space or inside the landmark.

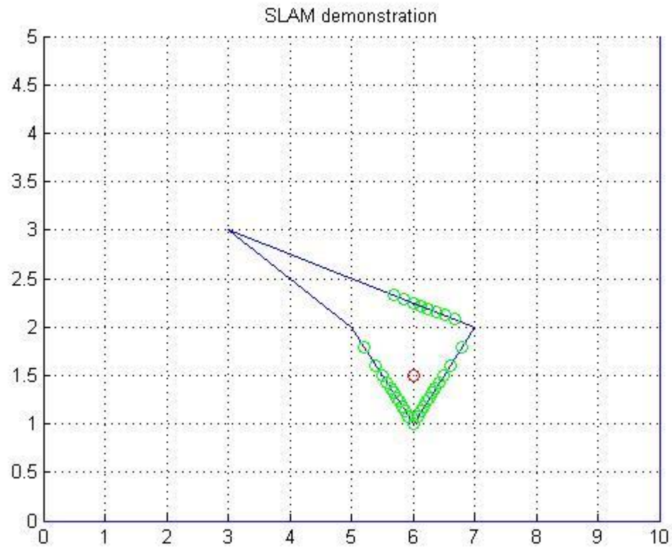

Figure 6 – Starting point inside the landmark.

This version of the algorithm is prone to a bad choice of the starting point. A solution to this problem may be an additional part of the code, which would prevent poor conditions occurrence (the appropriate function sp_def included into the code, which in this version of software is empty).

b) A narrow pass – only a tight pass between the space border and the landmark exists.

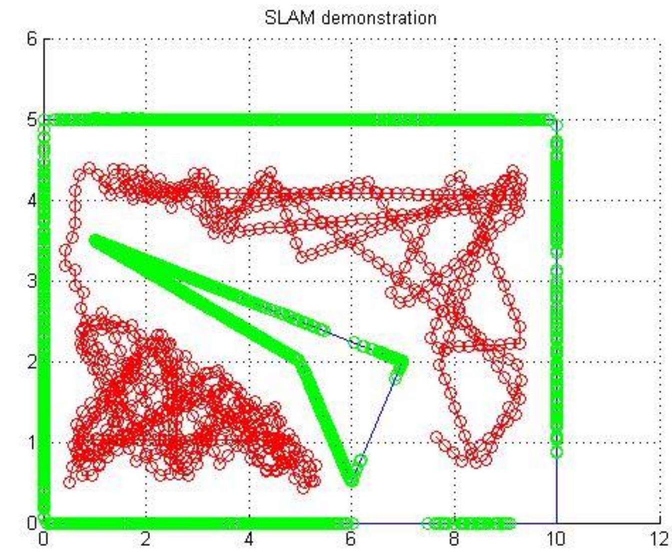

Figure 7 – Narrow pass between space border and the landmark.

Initially, the algorithm might have problems in a particular situation. To resolve this, settings of the force_range variable should be changed, which would decrease the significance of the obstacles vicinity and allow the vehicle to pass even through a narrow passage (it should be done carefully because too low a value of that variable could cause a movement outside the established space border).

c) A thin landmark – landmark of a small size.

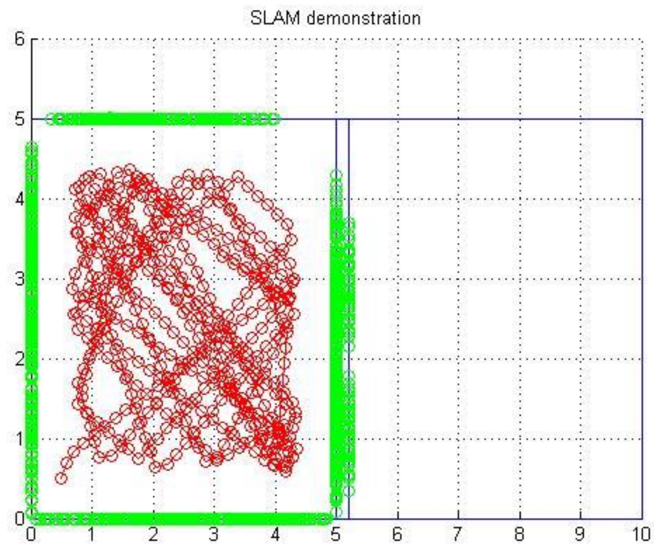

Figure 8 – Thin landmark case.

In this case the vehicle seems to look through the walls, which is non-physical behavior (a laser beam could not penetrate an obstacle except some special cases). The source of this problem lies in a mathematical function checking for intersections and the space map which was initially incorporated to simulate the reality. The actual version of the software does not contain a solution to the problem.

5. Conclusions

The problem of simultaneous mobile robot localization and automatic construction of a global map of the environment was addressed in this paper. A simple algorithm simulating solving selected aspects of SLAM was developed in MATLAB® software environment. The emphasis was placed on positioning, guidance (with a random factor), a collision avoidance method and mapping problem. Some limitations of the program were encountered via simulations, indicating directions for further development of the project. The main areas for improvement are introducing process noise and uncertainties in mapping and localization to better simulate the reality.

6. Contact Author Email Address

First Author – Tomasz Małecki, mailto: tomasz.malecki@ge.com.
Second Author – Janusz Narkiewicz, mailto: janusz.narkiewicz@pw.edu.pl.

7. Copyright Statement

References

[1] Durrant-Whyte H and Bailey T. Simultaneous localization and mapping (SLAM): Part I. *IEEE Robot. Autom. Mag.*, 2006.

[2] Bailey T and Durrant-Whyte H. Simultaneous localization and mapping (SLAM): Part II. *IEEE Robot. Autom. Mag.*, 2006.

[3] Sola J. Simulataneous localization and mapping with the extended Kalman filter. *Unpublished*, Available: http//www. joansola. eu/JoanSola/eng/JoanSola. html, 2013.

[4] Moreno L, Garrido S, Blanco D and Muñoz M L. Differential evolution solution to the SLAM problem. *Rob. Auton. Syst.*, 2009.

[5] Siemiątkowska B, Szklarski J and Gnatowski M. Mobile robot navigation with the use of semantic map constructed from 3D laser range scans. *Control Cybern.* (2011).

[6] Gamini Dissanayake M W M, Newman P, Clark S, Durrant-Whyte H F and Csorba M. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. Robot. Autom.*, 2001.

[7] Thrun S, Koller D, Ghahramani Z, Durrant-Whyte H, Ng A Y. Simultaneous mapping and localization with sparse extended information filters: Theory and initial results. *Springer Tracts in Advanced Robotics*, 2004.

[8] Castellanos J A, Neira and Tardós J D. Limits to the consistency of EKF-based SLAM. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2004.

[9] Julier S J and Uhlmann J K. A counter example to the theory of simultaneous localization and map building. *Proceedings - IEEE International Conference on Robotics and Automation*, 2001.

[10] Leonard J J, Feder H J S. Decoupled stochastic mapping. *IEEE J. Ocean. Eng.*, 2001.

[11] Tardós J D, Neira J, Newman P M and Leonard J J. Robust mapping and localization in indoor environments using sonar data. *Int. J. Rob. Res.*, 2002.

[12] Fan X, Guo Y, Liu H, Wei B and Lyu W. Improved Artificial Potential Field Method Applied for AUV Path Planning. *Math. Probl. Eng.*, 2020.