JOURNAL OF THEORETICAL AND APPLIED MECHANICS
60, 2, pp. 317-328, Warsaw 2022
https://doi.org/10.15632/jtam-pl/149478

A COLLISION AVOIDANCE ALGORITHM IN THE SIMULTANEOUS LOCALIZATION AND MAPPING PROBLEM FOR MOBILE PLATFORMS

Tomasz Małecki

Łukasiewicz Research Network – Institute of Aviation, Engineering Design Center, Warsaw, Poland
e-mail: tomasz.malecki@ge.com

Janusz Narkiewicz

Warsaw University of Technology, Faculty of Power and Aeronautical Engineering, Warsaw, Poland
e-mail: janusz.narkiewicz@pw.edu.pl

A collision avoidance algorithm applicable in simultaneous localization and mapping (SLAM) has been developed with a prospect of an on-line application for mobile platforms to search and map the operation area and avoid contact with obstacles. 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 definition of the next-step of platform motion. The proposed approach may be incorporated into real-time applications with limited on-board computational resources.

Keywords: algorithm, modeling, path planning, SLAM

List of symbols

d	–	max. sensor range
$\mathbf{F}_v(t)$	–	state transition matrix
i	–	discrete index
n_m	–	number of measurements
p, p_f	–	point and 'force point'
$\mathbf{R}$	–	rotation matrix
s	–	distance travelled
t	–	discrete time index
$\mathbf{u}_v(t)$	–	vector of control inputs
v_0	–	velocity of vehicle
$\mathbf{v}_v(t)$	–	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
$\mathbf{x}_v(t)$	–	state of vehicle at 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, β	–	measurement and steering angle

1. Introduction

Simultaneous Localization and Mapping (SLAM) methodology is a process of simultaneously estimating in a real time a structure of the surrounding environment perceived by sensors placed on a moving platform, creating a map and providing the platform position. Typical applications are mobile platforms that move inside an unknown space, acquire data, build a model of the surrounding environment and simultaneously localize themselves within it. The SLAM operation is implemented on mobile platforms on which at least one sensor is mounted to gather data about the platform surroundings, and algorithms are embedded in the processing equipment (Durrant-Whyte and Bailey, 2006). The main sources of SLAM data are image sensors like visual light cameras, radars, lidars, sonars. To support the SLAM process additional data may be acquired by other sensors, especially by those providing the platform position (GNSS, wheel encoders, INS) in the absolute reference system (Bailey and Durrant-Whyte, 2006).

The SLAM challenge stems from the fact that localization and mapping are coupled tasks when the platform location and environment mapping are deduced from the measurements of the same sensors. Thus, a solution is obtained only if the mapping and localization processes are considered together. The SLAM problem is widely investigated in the literature (Dissanayake *et al.*, 2001; Castellanos *et al.*, 2004; Thrun *et al.*, 2004; Bailey and Durrant-Whyte, 2006; Durrant-Whyte and Bailey, 2006; Moreno *et al.*, 2009; Siemiątkowska *et al.*, 2011; Sola, 2013).

The automated SLAM takes into account three factors crucial for the success of the process (Sola, 2013):

1. Platform motion (also known as 'motion model') – a mathematical model of mobile platform (agent) motion, allowing determination of the actual position.
2. Landmark localization and mapping (sometimes called an 'inverse observation model') – a function of discovering and recognizing distinct features (landmarks) in the environment which would be incorporated into the map. In the observed scene, the status and positions of the 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 the measured position (so called a 'direct observation model'). As a result, the uncertainties of landmarks localization decrease.

Usually, the SLAM 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 the uncertainties of the SLAM in time. Models of these operations combined with position estimators allow one to design an automated, effective solution of the SLAM process (Dissanayake *et al.*, 2001). The position estimator is used for diminishing propagation of the uncertainties in time; very often filtering (as Kalman or particle type) is used (Sola, 2013) for this purpose.

The uncertainties and disturbances of platform motion and of determination of the positions of the vehicle and the landmarks resulting from non-ideal sensor observation decrease reliability of the system (accuracy and precision of localization and mapping) and may lead to spurious results. In fact, all the factors in the process should be evaluated taking into account the 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 a stochastic environment (Julier and Uhlmann, 2001). It was also shown that linearization errors may produce inconsistency problems in the Extended Kalman Filter (EKF) solution for SLAM (Castellanos *et al.*, 2004). On the other hand, the linearization errors of a nonlinear system are inherent and unavoidable issues in many processes which might be mitigated but cannot be eliminated (Castellanos *et al.*, 2004).

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 (Moreno *et al.*, 2009). Some techniques have been developed to reduce the required computational load and to support scalability of the solution, for instance sparse extended information filters (Thrun *et al.*, 2004), decoupled stochastic mapping (Leonard and Feder, 2001) or sequential map joining (Tardós *et al.*, 2002). However, the problems with linearization and data association ambiguities are not completely solved (Moreno *et al.*, 2009).

Having in mind these challenges, the objective of this study is to examine an alternative and simplified solution to the collision avoidance for SLAM methods. 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 explored 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. The environment is a confined space containing a population of features (landmarks). In the proposed model, the vehicle is equipped with a generic sensor (named here laser type) 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 task of the system is to perform mapping of the unknown environment with simultaneous localization of the vehicle within the constructed map and to propose a trajectory avoiding obstacles. A linear, synchronous discrete-time model of the evolution of the vehicle and the observations of landmarks is adopted.

In this study, 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 are modelled as convex polygons bounded by closed polylines.

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

A laser-type sensor located in the center of the vehicle with a limited range of observation is used. It is assumed that the inertial navigation is used additionally to obtain coordinates of the platform position and components of velocity 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, the measurements are considered perfect and the landmarks positions may be uniquely defined. The distance between the vehicle and previously identified space borders or landmarks 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, 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 time 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 $\mathbf{x}_v$ and

a steering angle θ representing the direction of the vehicle velocity vector $\mathbf{v}$. The space borders are defined in the global reference frame. The origin of the local reference frame is the actual position of the vehicle.

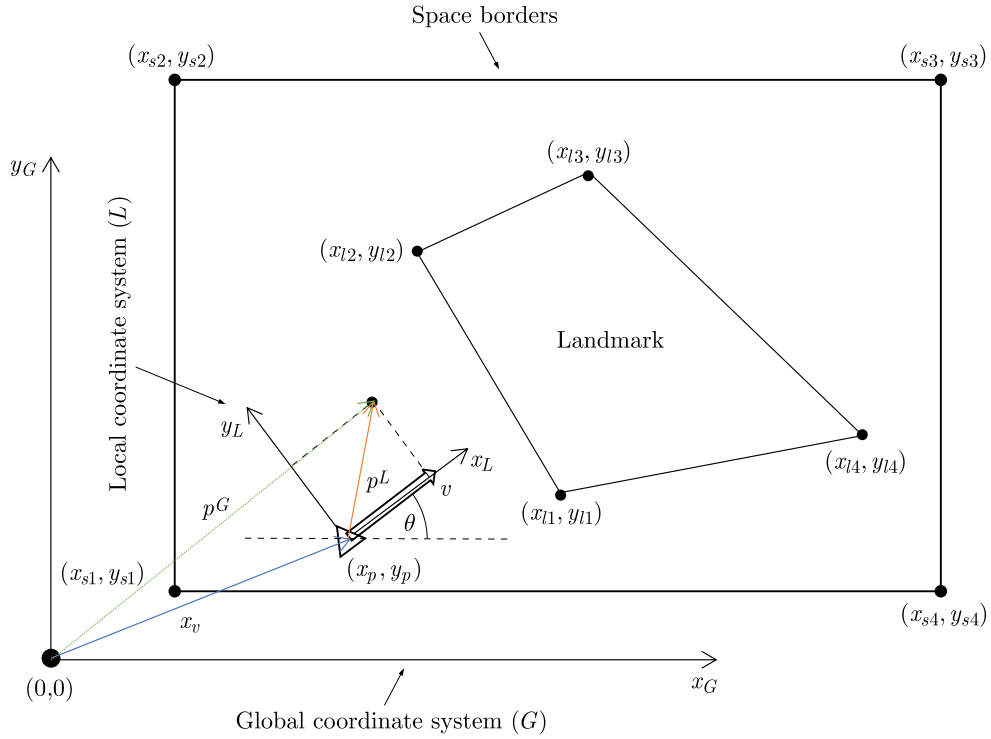

Fig. 1. Space definition and frames transformation

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

$$\begin{aligned} \mathbf{p}^G &= \mathbf{R}\mathbf{p}^L + \mathbf{x}_v \qquad \text{from frame } L \text{ to } G \\ \mathbf{p}^L &= \mathbf{R}^{\mathrm{T}}(\mathbf{p}^G - \mathbf{x}_v) \qquad \text{from frame } G \text{ to } L \end{aligned} \tag{2.1}$$

where $\mathbf{R}$ is the rotation matrix associated with the steering angle θ (2D case)

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{2.2}$$

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

$$\begin{aligned} x_i^G &= x_0 + x_i^L \cos\theta - y_i^L \sin\theta \\ y_i^G &= y_0 + x_i^L \sin\theta + y_i^L \cos\theta \end{aligned} \tag{2.3}$$

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

$$\mathbf{x}_v(t+1) = \mathbf{F}_v(t)\mathbf{x}_v(t) + \mathbf{u}_v(t+1) + \mathbf{v}_v(t+1) \tag{2.4}$$

The state vector of the vehicle at a time step t is denoted by $\mathbf{x}_v(t)$. The platform state transition matrix is denoted by $\mathbf{F}_v(t)$ (which for the purpose of this study simplifies to the identity matrix) and $\mathbf{u}_v(t)$ 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 $\mathbf{v}_v(t) = \mathbf{0}$.

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 the 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

$$\begin{aligned} x_{t+1}^G &= x_t^G + v_0\, dt \cos\theta \\ y_{t+1}^G &= y_t^G + v_0\, dt \sin\theta \end{aligned} \tag{2.5}$$

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

$$s_{t+1} = s_t + vt\, dt \tag{2.6}$$

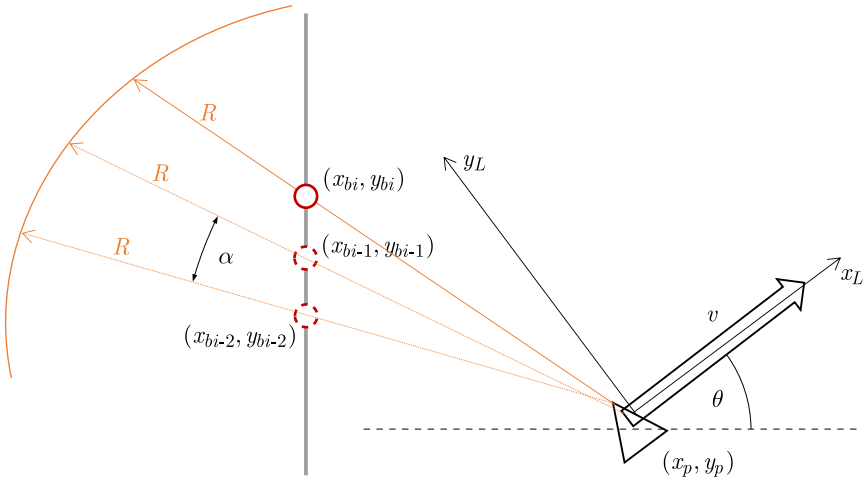

Fig. 2. Scheme of obstacle points determination

The observation sensor is modeled as 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 the α angle (Fig. 2). For the total number of sensor beams denoted by n_m

$$\alpha = \frac{360^\circ}{n_m} = \frac{2\pi}{n_m} \tag{2.7}$$

The coordinates of the points on the boundary of the sensor range corresponding to the i-th sensor beam could be evaluated as

$$x_i = R\cos\frac{2\pi(i-1)}{n_m} \qquad y_i = R\sin\frac{2\pi(i-1)}{n_m} \tag{2.8}$$

where R is the max. sensor range and $i = 1, \ldots, n_m$.

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.

The 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. Here a novel approach to the path planning problem, similar to (Fan *et al.*, 2020), has been proposed, but this concept is simpler and inherently less computational resources demanding. To meet this requirement some 'dynamic equivalent' was used to calculate the 'reaction force' that would be the measure of external body vicinity (it should be noted that dynamic similarity is not exact, because the vehicle is weightless and dimensionless, and no 'true' force acts on it).

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

$$r_f = \sqrt{(x_s - x_a)^2 + (y_s - y_a)^2} \tag{2.9}$$

It is calculated for all known/observed 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, 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 particular vectors at the given time step (Fig. 3).

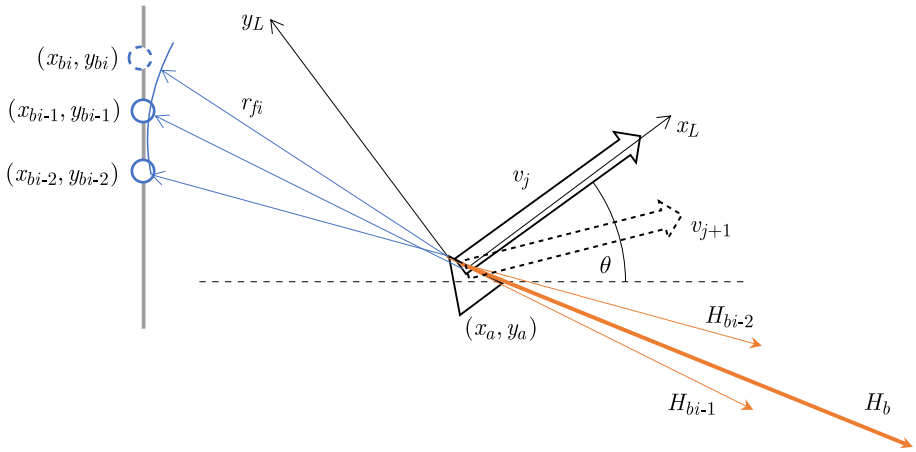

Fig. 3. Scheme of the 'reaction force' idea

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

$$\begin{aligned} p_{x(i+1)} &= p_{xi} - \frac{x_{bi} - x_a}{|x_{bi} - x_a|}\frac{1}{r_{fi}^2} = p_{xi} - \frac{x_{bi} - x_a}{|x_{bi} - x_a|}\frac{1}{(x_{bi} - x_a)^2 + (y_{bi} - y_a)^2} \\ p_{y(i+1)} &= p_{yi} - \frac{y_{bi} - y_a}{|y_{bi} - y_a|}\frac{1}{r_{fi}^2} = p_{yi} - \frac{y_{bi} - y_a}{|y_{bi} - y_a|}\frac{1}{(x_{bi} - x_a)^2 + (y_{bi} - y_a)^2} \end{aligned} \tag{2.10}$$

The resultant length of the 'repulsive' vector $|\mathbf{H}_b|$ is calculated as follows

$$|\mathbf{H}_b| = \sqrt{p_x^2 + p_y^2} \tag{2.11}$$

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

$$p_{x_{pred}} = \frac{p_x}{|H_b|} + \cos\theta \qquad p_{y_{pred}} = \frac{p_y}{|H_b|} + \sin\theta \tag{2.12}$$

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

$$\theta = \tan^{-1} \frac{p_{y_{pred}}}{p_{x_{pred}}} \tag{2.13}$$

Finally, the updated vehicle position is calculated based on equations (2.4) and $(2.5)_1$ with the steering angle calculated using Eq. (2.13). To describe platform motion, in the code, inverse tangent was implemented by MATLAB function 'atan2' which returns the four-quadrant inverse tangent of y and x. The atan2 function allows one to calculate one unambiguous arc tangent value from two variables y and x, where the signs of both arguments are used to determine the quadrant of the result. In addition, the atan2 function follows the convention that atan2(x, x) returns 0 when x is mathematically zero.

3. Implementation

The algorithm resulting from the approach given above is presented as a block diagram in Fig. 4. For the purpose of 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 the position and orientation of the vehicle. The sensor range points are calculated using Eqs. (2.7) and (2.8) in the local reference frame and then are transformed into the global reference frame using Eq. (2.3) 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. (2.9) if at least one intersection point is present. Subsequently, the next step prediction is performed by evaluating the resultant 'force point' position, Eqs. (2.10), calculating the length of the 'repulsive' vector, Eq. (2.11), computing the predicted position coordinates using Eqs. (2.12) and determining the steering angle for the next time step (2.13). Finally, the vehicle position is calculated using Eqs. (2.5) based on the previously calculated values. The entire sequence of operations is repeated until the maximum range of the vehicle is reached (equivalent to running out of the fuel/battery).

4. System simulation

The input data in Table 1 was used for the purpose of simulation.

Two groups of simulation cases are presented here illustrating efficiency of the method with special cases (narrow passages) included.

4.1. Simulations of normal conditions

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

1. The starting point is within the operational space.
2. The distance between the space borders and obstacle are relatively large (no narrow passages).
3. The landmark is of large width.

The output from the program is:

1) The 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) The 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.

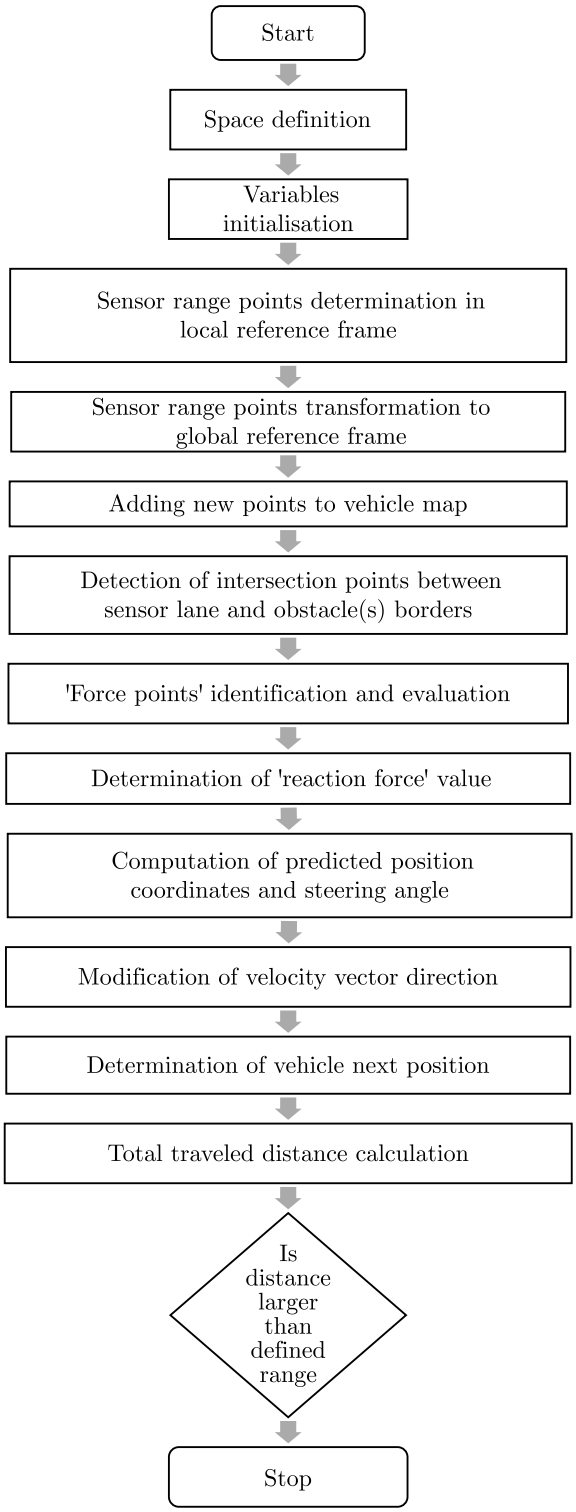

Fig. 4. Algorithm process diagram

4.2. Simulation of 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 (Fig. 6).

Table 1. Simulation data

Variable name	Value	Comments
space_borders	[0 0; 0 5; 10 5; 10 0; 0 0]	coordinates of space borders vertices
land_def	[5 2; 3 3; 7 2; 6 1; 5 2]	coordinates of landmarks borders vertices
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 battery endurance [m]
n_measure	36	number of measurements on azimuth; here performed every 10 degrees

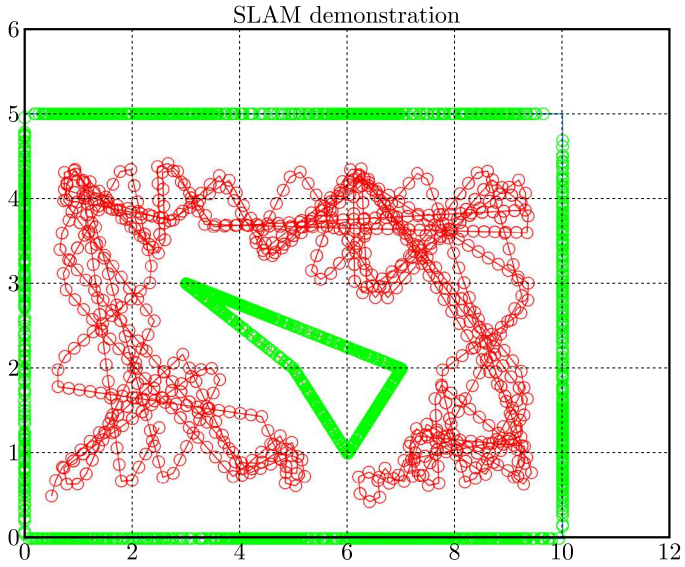

Fig. 5. Result of the algorithm launching under normal operating conditions

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

b) A narrow pass – only a tight pass between the space border and the landmark exists (Fig. 7). If the algorithm encounters problems, the settings of the force_range variable should be changed, which would decrease the significance of obstacles vicinity and allow the vehicle to pass through a narrow passage. A selection of this parameter should be done having in mind that a too low value of that variable could cause a movement outside the established space border.

c) A landmark of a small size (see Fig. 8) –

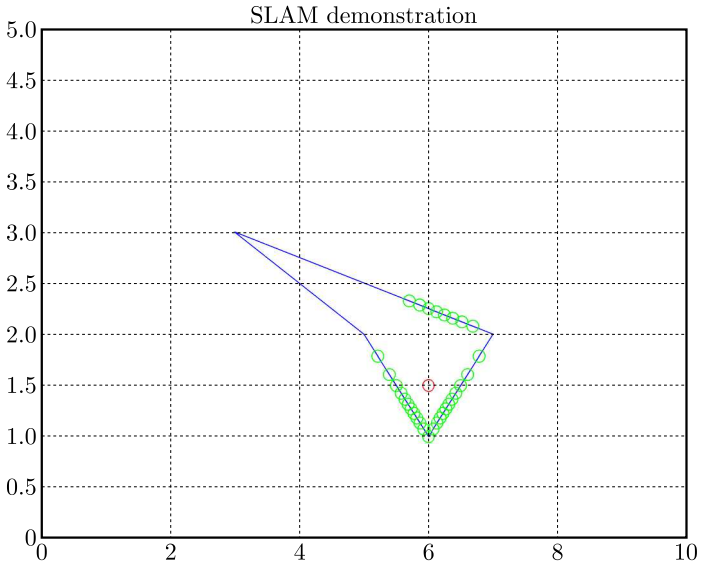

Fig. 6. Starting point inside the landmark

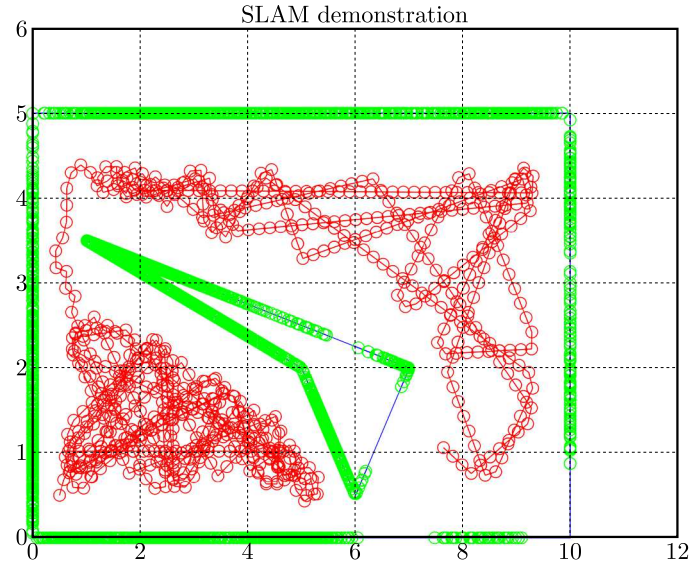

Fig. 7. Narrow pass between space border and the landmark

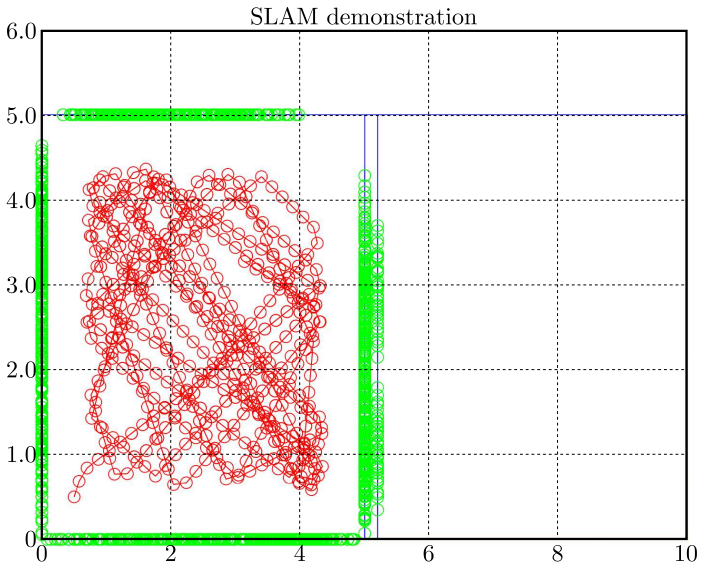

Fig. 8. Thin landmark case

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

5. Conclusions

The problem of simultaneous mobile robot localization and automatic construction of a global map of the environment are addressed in this paper. A simple algorithm simulating solving selected aspects of SLAM has been developed in MATLAB software environment. The emphasis is placed on path planning ('reaction force' approach), a collision avoidance method and the mapping problem. Some limitations of the program have been encountered via simulations indicating directions for improvement and further development of the method. The main areas for the improvement will introduce the process noise and uncertainties in mapping and localization to better simulate the reality. The presented algorithm has a potential to operate in a real-time application with limited on-board computational resources. The future work may focus on the in-depth control algorithm performance assessment and algorithm speed comparison with other methods.

References

1. Bailey T., Durrant-Whyte H., 2006, Simultaneous localization and mapping (SLAM): Part II, *IEEE Robotics and Automation Magazine*, **13**, 3, 108-117
2. Castellanos J.A., Neira J., Tardós J.D., 2004, Limits to the consistency of EKF-based SLAM, *IFAC Proceedings Volumes (IFAC-PapersOnline)*, **37**, 8
3. Dissanayake M.W.M.G., Newman P., Clark S., Durrant-Whyte H.F., Csorba M., 2001, A solution to the simultaneous localization and map building (SLAM) problem, *IEEE Transactions on Robotics and Automation*, **17**, 3, 229-241
4. Durrant-Whyte H., Bailey T., 2006, Simultaneous localization and mapping: Part I, *IEEE Robotics and Automation Magazine*, **13**, 2, 99-110
5. Fan X., Guo Y., Liu H., Wei B., Lyu W., 2020, Improved artificial potential field method applied for AUV path planning, *Mathematical Problems in Engineering*, **2020**, 1, 1-21
6. Julier S.J., Uhlmann J.K., 2001, A counter example to the theory of simultaneous localization and map building, *Proceedings of the IEEE International Conference on Robotics and Automation*, **4**, 4238-4243
7. Leonard J.J., Feder H.J.S., 2001, Decoupled stochastic mapping (for mobile robot and AUV navigation), *IEEE Journal of Oceanic Engineering*, **26**, 4, 561-571
8. Moreno L., Garrido S., Blanco D., Muñoz M.L., 2009, Differential evolution solution to the SLAM problem, *Robotics and Autonomous Systems*, **57**, 4, 441-450
9. Siemiątkowska B., Szklarski J., Gnatowski M., 2011, Mobile robot navigation with the use of semantic map constructed from 3D laser range scans, *Control and Cybernetics*, **40**, 1
10. Sola J., 2013, Simulataneous localization and mapping with the extended Kalman filter, unpublished, Available: http://www.joansola.eu/JoanSola/eng/JoanSola.html
11. Tardós J.D., Neira J., Newman P.M., Leonard J.J., 2002, Robust mapping and localization in indoor environments using sonar data, *International Journal of Robotics Research*, **21**, 4, 311-330

12. Thrun S., Koller D., Ghahramani Z., Durrant-Whyte H., Ng A.Y., 2004, Simultaneous mapping and localization with sparse extended information filters: Theory and initial results, [In:] *Algorithmic Foundations of Robotics V. Springer Tracts in Advanced Robotics*, J.D. Boissonnat, J. Burdick, K. Goldberg, S. Hutchinson (Edit.), **7**, 363-380

Manuscript received January 17, 2022; accepted for print April 7, 2022