Ortiz *et al*. *Intell Robot* 2021;1(2):131-50
DOI: 10.20517/ir.2021.09

Intelligence & Robotics

Research Article

Open Access

Check for updates

Autonomous navigation in unknown environment using sliding mode SLAM and genetic algorithm

Salvador Ortiz, Wen Yu

Departamento de Control Automatico, National Polytechnic Institute, Mexico City 07360, Mexico.

Correspondence to: Prof. Wen Yu, Departamento de Control Automatico, National Polytechnic Institute, Mexico City 07360, Mexico. E-mail: yuw@ctrl.cinvestav.mx

How to cite this article: Ortiz S, Yu W. Autonomous navigation in unknown environment using sliding mode SLAM and genetic algorithm. *Intell Robot* 2021;1(2):131-50. http://dx.doi.org/10.20517/ir.2021.09

Received: 9 Sep 2021 **First Decision:** 28 Oct 2021 **Revised:** 15 Nov 2021 **Accepted:** 2 Dec 2021 **Published:** 15 Dec 2021

Academic Editors: Simon X. Yang, Hao Zhang **Copy Editor:** Yue-Yue Zhang **Production Editor:** Yue-Yue Zhang

Abstract

In this paper, sliding mode control is combined with the classical simultaneous localization and mapping (SLAM) method. This combination can overcome the problem of bounded uncertainties in SLAM. With the help of genetic algorithm, our novel path planning method shows many advantages compared with other popular methods.

Keywords: Autonomous navigation, sliding mode, SLAM, genetic algorithm

1. INTRODUCTION

1.1. Autonomous navigation in unknown environment

Autonomous navigation (AN) has three jobs [1].

(1) Perception: Mapping from signal to information is the perception of AN [2]. Its algorithms can use human thought [3], intelligent methods [4], optimization [5], probability methods [6], and genetic algorithms [7].

(2) Motion planning: It has three classes, namely graph methods such as a roadmap [8], random sampling [9], and grid [10].

(3) Localization and mapping: In unknown environments, sensors, actuators, and maps may have big uncertainties.

Path planning (PP) can be performed under the following conditions:

(1) The environment is known. PP is an optimization problem[11–13].

(2) The environment is partially known. PP can find new objects during navigation[14,15].

(3) The environment is totally unknown. PP depends on the navigation and has a recursive solution[16–18].

Simultaneous localization and mapping (SLAM) can be used in unknown environments[19] or in partially unknown environments[20]. SLAM[21] uses the current position to construct a map, and it can be classified into feature-based[22], pose-based[23], appearance-based[24], and variants[25].

The most popular SLAM uses Kalman filter[21] for Gaussian noise. Nonlinear SLAM uses extended Kalman filter (EKF)[26], where the noise assumptions are not satisfied[27]. EKF-SLAM applies linearization[28].

1.2. Related work

Few AN uses SLAM. Visual SLAM uses several cameras[29]. AN can use both SLAM and GPS signals[30]. Robots can avoid moving obstacles using neural networks[31]. Swarm optimization helps robots follow an object[32]. Neural networks help robots construct the navigation path[33]. The optimal path is considered in the sense of trajectory length, execution time, or energy consumption.

Genetic algorithms (GA) have been developed recently[34,35]. They are easy to use for optimization in non-deterministic cases[36], uncertainty models[37], and robust cases[38]. GA can be in form of ant-based GA[39,40], cell decomposition GA[41], potential field GA[42], ant colony[43], and particle swarm optimization[44]. Finite Markov chain is a theory tool for GA[45,46].

1.3. Our work

In this paper, we try to design AN in an unknown environment in real time. The contributions are as follows:

(1) Sliding mode SLAM: The robustness of this SLAM is better than other SLAM models in bounded noise.

(2) GA SLAM: We use roadmap PP and GA to generate the local optimal map.

(3) Comparisons and simulations with other SLAM models were made by using a mobile robot[47].

2. SLIDING MODE SLAM

SLAM gives the robot position and environment map at the same time. At time k, the state is $\mathbf{x}_k^r = (x_k, y_k, \theta_k)$, where (x_k, y_k) is the position and θ_k is the orientation of the robot. $\mathbf{x}_k^m = (\mathbf{m}_k^1, \mathbf{m}_k^2, ..., \mathbf{m}_k^L)^T$ are landmarks, with $\mathbf{m}_k^i = (x_k^i, y_k^i)^T$ the ith landmark. We assume the true location is time-invariant.

$\mathbf{x}_k$ has two parts: the robot $\mathbf{x}_k^r$ and the landmarks $\mathbf{x}_k^m$. The state equation is

$$\mathbf{x}_{k+1} = \begin{pmatrix} \mathbf{x}_{k+1}^r \\ \mathbf{x}_{k+1}^m \end{pmatrix} = \begin{pmatrix} \mathbf{f}(\mathbf{x}_k^r, \mathbf{u}_k) + \mathbf{w}_k \\ \mathbf{x}_k^m \end{pmatrix} = \mathbf{F}(\mathbf{x}_k, \mathbf{u}_k) + [\mathbf{w}_k, 0]^T \quad (1)$$

where $\mathbf{f}()$ is the robot dynamics, $\mathbf{w}_k$ is the noise, and $\mathbf{u}_k$ is the robot control. Since $\mathbf{x}_k^m$ is not influenced by motion noise, the noise is $[\mathbf{w}_k, 0]^T$.

$\mathbf{z}_k$ is defined as the position between the robot and the landmark, whose model is

$$\mathbf{z}_k^i = \mathbf{h}(\mathbf{x}_k^r, \mathbf{m}_k^i) + \mathbf{v}_k^i \tag{2}$$

where $\mathbf{h}()$ is the geometry and $\mathbf{v}_k^i$ is the noise. Here, $\mathbf{w}_k$ and $\mathbf{v}_k^i$ are not Gaussian noises. We assume $\mathbf{w}_k$ and $\mathbf{v}_k^i$ are bounded.

To estimate $\mathbf{x}_k$ in Equations (1) and (2), EKF is needed. We linearize the state model in Equation (1) and the observation model in Equation (2) as

$$\begin{aligned} &\mathbf{x}_{k+1} = \mathbf{F}(\hat{\mathbf{x}}_k, \mathbf{u}_k) + \nabla\mathbf{F}_k \cdot \left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right) \\ &+O_1\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^2\right] + [\mathbf{w}_k, 0]^T \\ &\mathbf{z}_k^i = \mathbf{h}(\hat{\mathbf{x}}_k) + \nabla\mathbf{h}_k \cdot \left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right) + O_2\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^2\right] + \mathbf{v}_k^i \end{aligned} \tag{3}$$

where $\nabla\mathbf{F}_k = \frac{\partial\mathbf{F}}{\partial\mathbf{x}_k}|_{\mathbf{x}_k=\hat{\mathbf{x}}_k}$, $\nabla\mathbf{h}_k = \frac{\partial\mathbf{h}}{\partial\mathbf{x}_k}|_{\mathbf{x}_k=\hat{\mathbf{x}}_k}$, $O_1\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^2\right]$, $\hat{\mathbf{x}}_k$ is the estimation of $\mathbf{x}_k$.

Prediction. The estimation $\hat{\mathbf{x}}_{k+1}$ is based on past states, control, and landmarks:

$$\begin{aligned} &\hat{\mathbf{x}}_{k+1} = \mathbf{F}(\hat{\mathbf{x}}_k, \mathbf{u}_k) \\ &\mathbf{P}_{k+1} = \nabla\mathbf{F}_k\mathbf{P}_k\nabla\mathbf{F}_k^T + R_1 \end{aligned} \tag{4}$$

where R_1 is the covariance of $\mathbf{w}_k$, $R_1 = E\left\{[\mathbf{w}_k - E(\mathbf{w}_k)]\,[\mathbf{w}_k - E(\mathbf{w}_k)]^T\right\}$.

Correction. The new state is based on predicted states, landmarks, and current observations:

$$\begin{aligned} &\hat{\mathbf{x}}_{k+2} = \hat{\mathbf{x}}_{k+1} + \mathbf{K}_{k+1}\left[\mathbf{z}_{k+1}^i - \mathbf{h}(\hat{\mathbf{x}}_{k+1})\right] \\ &\mathbf{K}_{k+1} = \mathbf{P}_{k+1}\nabla\mathbf{h}_{k+1}\left[\nabla\mathbf{h}_{k+1}\mathbf{P}_{k+1}\nabla\mathbf{h}_{k+1}^T + R_2\right]^{-1} \\ &\mathbf{P}_{k+2} = \left[I - \mathbf{K}_{k+1}\nabla\mathbf{h}_{k+1}\right]\mathbf{P}_{k+1} \end{aligned} \tag{5}$$

The motivations of using sliding mode modification to the EKF bases SLAM based are the following:

(1) The noises $\mathbf{w}_k$ and $\mathbf{v}_k^i$ in Equations (1) and (2) are not Gaussian.

(2) There are linearization error terms, $O_1\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^2\right]$ and $O_2\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^2\right]$, in Equation (3), and the traditional EKF-based methods do not work well for these errors.

We use the sliding mode method to estimate the robot state $\mathbf{x}_k^r$ and the landmark $\mathbf{x}_k^m$.

Sliding modes have a number of attractive features, and thus have long been in use for solving various control problems. The basic idea behind design of system with sliding mode is the following two steps: (1) a sliding motion in a certain sense is obtained by an appropriate choice of discontinuity surfaces; and (2) a control is chosen so that the sliding modes on the intersection of those discontinuity surface would be stable. A general class of discontinuous control $u(x,t)$ is defined by the following relationships:

$$u(x,t) = \begin{cases} u^+(x,t) & \text{with } \mathbf{s}(\mathbf{x}) > 0 \\ u^-(x,t) & \text{with } \mathbf{s}(\mathbf{x}) < 0 \end{cases} \tag{6}$$

where the functions $u^+(x,t)$ and $u^-(x,t)$ are continuous.

The function $\mathbf{s}(\mathbf{x})$ is the discontinuity surface (subspace). The objective of the sliding mode control is to design some switching strategy of the continuous control $u^+(x,t)$ and $u^-(x,t)$, such that

$$\mathbf{s}(\mathbf{x}) = 0 \tag{7}$$

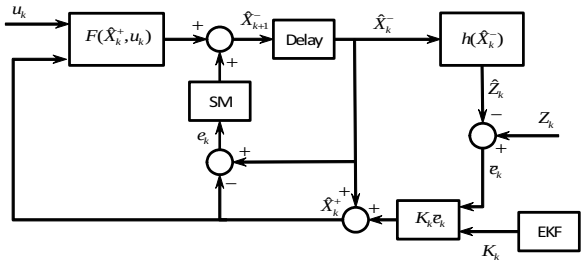

Figure 1. Sliding mode simultaneous localization and mapping.

In this paper, the sliding surface is defined by the SLAM estimation error as

$$e(k) = \mathbf{x}_k - \hat{\mathbf{x}}_k \tag{8}$$

Here, the discontinuity surface is $e(k) = [e_1 \cdots e_n]$. We consider the following positive definite function,

$$V = \frac{1}{2} e^T(k) P e(k) \tag{9}$$

where P is diagonal positive definite matrix, $P = P^T > 0$. The derivative of V is

$$\dot{V} = e^T(k) P \dot{e}(k) \tag{10}$$

The motion $e(k)$ satisfies

$$\dot{e}(k) = -\rho \times sgn[e(k)], \quad \rho > 0 \tag{11}$$

where $sgn[e(k)] = [sgn(e_1), \ldots, sign(e_n)]^T$, $sgn(e_i) = \begin{cases} 1 \text{ with } e_i(x) > 0 \\ -1 \text{ with } e_i(x) < 0 \end{cases}$, $sgn(0) = 0$, then (10) is

$$\dot{V} = e^T(k) P \{-\rho \times sgn[e(k)]\} = -\rho e^T(k) P sgn[e(k)]$$

because $P = diag\{p_i\}$, $p_i > 0$, and $e_i \times sgn(e_i) = |e_i|$

$$\dot{V} = -\rho \sum_{i=1}^{n} p_i |e_i| \tag{12}$$

Thus, $\dot{V} \leq 0$. By Barbalat's lemma[48], the estimation error is $e(k) \to 0$.

The classical SLAM in Equations (4) and (5) is modified by the sliding surface in Equation (11). The sliding mode control can be regarded as a compensator for Equation (4):

$$\hat{\mathbf{x}}_{k+1} = \mathbf{F}(\hat{\mathbf{x}}_k, \mathbf{u}_k) - \rho \times sgn[e(k)] \tag{13}$$

where ρ is a positive constant. The correction step is the same as EKF in Equation (5). The sliding mode SLAM is shown in Figure 1. Here, the estimation error, $e(k)$, is applied to the sliding surface to enhance the robustness in the prediction step with respect to the noise and disturbances.

It is the discrete-time version of Equation (6). We give the stability analysis of this discrete-time sliding mode SLAM at the end of this section.

For the mobile robot, the sliding mode SLAM can be specified as follows. We define a critical distance $d_{\min}$ to limit the maximal landmark density. It can reduce false positives in data association and avoid overload with

useless landmarks. If the new landmark is far from the other landmarks on the map, then the landmark is added; otherwise, it is ignored. If the distance between the new landmark $\mathbf{x}_{k+1} = [x_{m+1}, y_{m+1}]$ and the others is bigger than $d_{\min}$, it should be added into $\mathbf{x}_k$, *i.e.*,

$$\mathbf{x}_{k+1} = g(\mathbf{x}_k^r, \mathbf{z}_k) \tag{14}$$

It can be transformed into an absolute framework as

$$\mathbf{x}_{k+1} = \begin{pmatrix} \mathbf{x}_k \\ g(\mathbf{x}_k^r, \mathbf{z}_k) \end{pmatrix} = \mathbf{T}(\mathbf{x}_k, \mathbf{z}_k) \tag{15}$$

The nonlinear transformation function $\mathbf{T}$ also applies to the uncertainties. We approximate the transformation $\mathbf{T}$ by the linearization. $\mathbf{P}_k$ can be expressed as

$$\mathbf{P}_k = \begin{pmatrix} \mathbf{P}_k^r & \mathbf{P}_k^{rm} & \mathbf{0} \\ \left(\mathbf{P}_k^{rm}\right)^T & \mathbf{P}_k^m & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{V}_k \end{pmatrix} \tag{16}$$

where

$$\mathbf{P}_k = \nabla\mathbf{T}\mathbf{P}_k\nabla\mathbf{T}^T$$

with $\nabla\mathbf{T} = \begin{pmatrix} \mathbf{I}_r & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_m & \mathbf{0} \\ \nabla\mathbf{g}_x & \mathbf{0} & \nabla\mathbf{g}_z \end{pmatrix}$, $\nabla\mathbf{g}_x := \frac{\partial\mathbf{g}}{\partial\mathbf{x}_k^r}(\mathbf{x}_k, \mathbf{z}_k)$, $\nabla\mathbf{g}_z := \frac{\partial\mathbf{g}}{\partial\mathbf{z}}(\mathbf{x}_k, \mathbf{z}_k)$.

For the motion part, we use the Ackerman vehicle model[49]

$$\begin{pmatrix} x_k^r \\ y_k^r \\ \theta_k^r \end{pmatrix} = \begin{pmatrix} x_{k-1}^r + T_{k-1}v_{k-1}\cos\theta_{k-1}^r \\ y_{k-1}^r + T_{k-1}v_{k-1}\sin\theta_{k-1}^r \\ \theta_{k-1}^r + T_{k-1}\frac{v_{k-1}}{b_a}\tan\alpha_{k-1} \end{pmatrix} + \mathbf{w}_k \tag{17}$$

where $\mathbf{w}_k$ is the process noise, v_k is the linear velocity, α_k is the steering angle, T_k is the sample time, and b_a is the distance between the front and the rear wheels.

At the beginning of map building, the vector $\hat{\mathbf{x}}_k$ only contains the robot states without landmarks. As exploration increases, the robot detects landmarks and decides if it should add these new landmarks to the state.

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{T}(\mathbf{x}_k, \mathbf{z}_k)\begin{pmatrix} \mathbf{x}_{k,x}^r \\ \mathbf{x}_{k,y}^r \end{pmatrix} + r_k^j\begin{pmatrix} \cos(\theta_k^i + \mathbf{x}_{k,\phi}^r) \\ \sin(\theta_k^i + \mathbf{x}_{k,\phi}^r) \end{pmatrix} \\ \mathbf{z}_k &= \begin{pmatrix} \sqrt{(m_x^i - x_k)^2 + (m_y^i - y_k)^2} \\ \arctan\left(\frac{m_y^i - y_k}{m_x^i - x_k}\right) - \phi_k \end{pmatrix}_i + \mathbf{V}_k \end{aligned} \tag{18}$$

where $\mathbf{x}$, y, $\mathbf{z}$, and m are defined in Equations (1) and (2),

We exploit the same property in the sliding SLAM. The landmarks with fewer corrections are removed from the state vector.

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k + G_x^T\begin{bmatrix} u_{k,v}\delta_t\cos(\mathbf{x}_{k,\phi}^r) \\ u_{k,v}\delta_t\sin(\mathbf{x}_{k,\phi}^r) \\ u_{k,\gamma}\delta_t \end{bmatrix} + \sigma_k \tag{19}$$

where $G_x = \begin{pmatrix} 1 & 0 & 0 & \underbrace{0 \cdots 0}_{2N} \\ 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$, σ_k is the compensator, and

$$\sigma_k = -\rho \times sgn\left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right) \tag{20}$$

This sliding SLAM algorithm is given in the following algorithm.

Sliding mode SLAM. $\hat{x}_1 = 0, P_{1|1} = \alpha I, k = 1, \alpha \gg 1$ u_1 =get_controls, z_1 =get_observations; $k_z = 1$ $\left[\hat{\mathbf{x}}_1, \mathbf{P}_1\right]$ =add_features$\left(\hat{\mathbf{x}}_1, \mathbf{P}_1, \mathbf{z}_1\right)$ (1) While not_stop if controls_are_available $\left[\hat{\mathbf{x}}_{k+1}, \mathbf{P}_{k+1}\right]$ =prediction$\left(\hat{\mathbf{x}}_k, \mathbf{P}_k, \mathbf{u}_k\right)$ (2) u_k =get_controls end if if observations_are_available get_observations z_k data_association$\left(\mathbf{z}_k, \hat{\mathbf{x}}_{k+1}, \mathbf{P}_{k+1}\right)$ $\left[\hat{\mathbf{x}}_{k+2}, \mathbf{P}_{k+2}, \mathbf{c}_k\right] = \left(\hat{\mathbf{x}}_{k+1}, \mathbf{P}_{k+1}, \mathbf{z}_k\right)$ (5) $\left[\hat{\mathbf{x}}_{k+2}, \mathbf{P}_{k+2}\right] = \left(\hat{\mathbf{x}}_{k+2}, \mathbf{P}_{k+2}, \mathbf{z}_k\right)$ (1) $k_z = k_z + 1$ end if if $\text{mod}(k_z, K_z) = 0$ $\left[\hat{\mathbf{x}}_{k+2}, \mathbf{P}_{k+2}\right]$ =pruning$\left(\hat{\mathbf{x}}_{k+2}, \mathbf{P}_{k+2}, \mathbf{c}_k, \mathbf{a}_k\right)$ end if $k = k + 1$ end While

---

The discrete-time sliding mode SLAM in Equation (19) can be written as

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k + \hat{F}(\hat{\mathbf{x}}_k, \mathbf{u}_k) + \sigma_k$$

where $\hat{F} = G_x^T \begin{bmatrix} u_{k,v}\delta_t \cos(\mathbf{x}^r_{k,\phi}) \\ u_{k,v}\delta_t \sin(\mathbf{x}^r_{k,\phi}) \\ u_{k,\gamma}\delta_t \end{bmatrix}$, $e(k) = \mathbf{x}_k - \hat{\mathbf{x}}_k$, $\sigma_k = \rho \times sgn[e(k)]$

The correction step for $\hat{\mathbf{x}}_{k+2}$ is the same as EKF:

$$\begin{aligned} \hat{\mathbf{x}}_{k+2} &= \hat{\mathbf{x}}_{k+1} + \mathbf{K}_{k+1}\left[\mathbf{z}^i_{k+1} - \mathbf{h}(\hat{\mathbf{x}}_{k+1})\right] \\ \mathbf{K}_{k+1} &= \mathbf{P}_{k+2}C_{k+1}\left[C_{k+1}\mathbf{P}_{k+2}C^T_{k+1} + R_2\right]^{-1} \\ \mathbf{P}_{k+2} &= \left[I - \mathbf{K}_{k+1}C_{k+1}\right]\mathbf{P}_{k+2} \end{aligned} \tag{21}$$

where $C_k = \nabla \mathbf{h}_k = \frac{\partial \mathbf{h}}{\partial \mathbf{x}_k}|_{\mathbf{x}_k = \hat{\mathbf{x}}_k}$.

The error dynamic of this discrete-time sliding mode observer is

$$e(k+1) = A_k e(k) - A_k \mathbf{K}_k C_k e(k) + \sigma_k + d_k \tag{22}$$

where $d_k = \hat{F}(\hat{\mathbf{x}}_k, \mathbf{u}_k) + \xi_k$ is bounded uncertainty, $\|d_k\|^2 \le \bar{d}$, $A_k = \nabla \mathbf{F}_k = \frac{\partial \mathbf{F}}{\partial \mathbf{x}_k}|_{\mathbf{x}_k = \hat{\mathbf{x}}_k}$, and $\mathbf{K}_k$ is the gain of EKF in Equation (21).

The next theorem gives the stability of the discrete-time sliding mode SLAM.

Theorem 1 *If the gain of the sliding mode SLAM is positive, then the estimation error is stable, and the estimation error converges to*

$$\|e(k)\|^2 \le \frac{\lambda_{\max}\left[\mathbf{P}^{-1}_{k+1}\right](\bar{\rho} + \bar{s}) + \bar{\rho}}{\alpha \lambda_{\min}\left[\mathbf{P}^{-1}_{k+2}\right]} \tag{23}$$

where $\|\sigma_k\|^2 \le \bar{\rho}, \sigma_k \|d_k\|^2 \le \bar{s}$, $\mathbf{P}_{k+2}$ *is the gain of EKF in Equation (21),* $0 < \alpha = \frac{1}{(1+\bar{p}\bar{a}^2/\underline{q})(1+\bar{k}\bar{c}+\lambda)} < 1$, $\underline{p}I \le \mathbf{P}_{k+2} \le \bar{p}I$, *and* $\underline{q}I \le R_1$.

Proof 1 *Consider the Lyapunov function as*

$$V_k = e\left(k\right)\mathbf{P}_k^{-1}e\left(k\right) \tag{24}$$

where $\mathbf{P}_{k+2}$ *is the prior covariance matrix in Equation (21), and* $\mathbf{P}_{k+2} > 0$. *From Equation (22),*

$$\begin{aligned} &V_{k+1} = e\left(k+1\right)\mathbf{P}_{k+1}^{-1}e\left(k+1\right) \\ &= e\left(k\right)\left(I-\mathbf{K}_kC_k\right)^TA_k^T\left(\mathbf{P}_{k+1}\right)^{-1}A_k(I-\mathbf{K}_kC_k)e\left(k\right) \\ &+2(d_k+\sigma_k^T)\mathbf{P}_{k+1}^{-1}\left[A_k(I-\mathbf{K}_kC_k)e\left(k\right)\right] \\ &+(d_k+\sigma_k^T)\mathbf{P}_{k+1}^{-1}(d_k+\sigma_k) \end{aligned} \tag{25}$$

Because $\|\sigma_k\|^2 \le \bar{\rho}$, $\|d_k\|^2 \le \bar{s}$, *the last term on the right side of Equation (25) is*

$$(d_k+\sigma_k^T)\mathbf{P}_{k+1}^{-1}(d_k+\sigma_k) \le \lambda_{\max}\left[\mathbf{P}_{k+1}^{-1}\right](\bar{\rho}+\bar{s}) \tag{26}$$

where $\lambda_{\max}\left[\mathbf{P}_{k+1}^{-1}\right]$ *is the maximum eigenvalue of* $\mathbf{P}_{k+1}^{-1}$.

The second term of Equation (25) is

$$\begin{aligned} &2(d_k+\sigma_k^T)\mathbf{P}_{k+1}^{-1}\left[A_k(I-\mathbf{K}_kC_k)e\left(k\right)\right] \\ &= 2(d_k+)\mathbf{P}_{k+1}^{-1}\left[A_k(I-\mathbf{K}_kC_k)e\left(k\right)\right] \\ &+\sigma_k^T\mathbf{P}_{k+1}^{-1}\left[A_k(I-\mathbf{K}_kC_k)e\left(k\right)\right] \end{aligned} \tag{27}$$

where $\mathbf{K}_k$ *is the gain of EKF in Equation (5). In view of the matrix inequality*

$$X^TY + \left(X^TY\right)^T \le X^T\Lambda^{-1}X + Y^T\Lambda Y \tag{28}$$

which is valid for any $X, Y \in \Re^{n\times k}$ *and for any positive definite matrix* $0 < \Lambda = \Lambda^T \in \Re^{n\times n}$, *the first term of Equation (27) is*

$$\begin{aligned} &2d_k\mathbf{P}_{k+1}^{-1}\left[A_k(I-\mathbf{K}_kC_k)e\left(k\right)\right] \\ &\le d_k\Lambda(d_k+) + e\left(k\right)\mathbf{P}_{k+1}^{-1}\left[A_k(I-\mathbf{K}_kC_k)\right]\Lambda^{-1}e\left(k\right) \\ &\le \bar{s}\lambda_{\max}\left[\Lambda\right] + \left\|\left(\mathbf{P}_{k+1}\right)^{-1}\left[A_k(I-\mathbf{K}_kC_k)\right]\Lambda^{-1}\right\|\|e\left(k\right)\|^2 \\ &\le \bar{s}\lambda_{\max}\left[\Lambda\right] + e\left(k\right)\left[\mathbf{P}_{k+1}^{-1}\left[A_k(I-\mathbf{K}_kC_k)\right]\Lambda^{-1}\right]e\left(k\right) \end{aligned} \tag{29}$$

We apply the sliding mode compensation in Equation (20) to the second term of Equation (27):

$$\begin{aligned} &\rho \times sgn\left[e\left(k\right)\right]_k^T\Upsilon_ke\left(k\right) \\ &= -\rho\sum_{k=1}^{m}\left|e\left(k\right)\right|\left(l_{kk}+\sum_{i=1,i\ne k}^{m}l_{ki}sign\left(\left[e\left(k\right)\right]e_i\left(k\right)\right)\right) \end{aligned} \tag{30}$$

where l_{ij} *are the elements of the matrix* Υ, $\Upsilon_k = \mathbf{P}_{k+1}^{-1}\left[A_k(I-\mathbf{K}_kC_k)\right]$. *When the the orientation* θ_k *is not big,* $\sin\theta_k \approx 0$, $\cos\theta_k \approx 0$,

$$l_{kk} \gg \sum_{i=1,i\ne k}^{m}\left|l_{ki}\right|, \quad l_{kk} > 0, \quad k = 1,\ldots m, \tag{31}$$

Thus, the second term of Equation (27) is negative.

The first term on the right side of Equation (25) has the following properties:

$$\mathbf{P}_{k+2} \ge \left(I-\mathbf{K}_kC_k\right)\mathbf{P}_{k+2}\left(I-\mathbf{K}_kC_k\right)^T$$

$(I - \mathbf{K}_k C_k)$ *is invertible, and we have*

$$(\mathbf{P}_{k+2})^{-1} \leq (I - \mathbf{K}_k C_k)_k^{-T} (\mathbf{P}_{k+2})^{-1} (I - \mathbf{K}_k C_k)^{-1} \tag{32}$$

According to EKF,

$$\mathbf{P}_{k+1} = A_k \mathbf{P}_{k+2} A_k^T + R_1 = A_k (\mathbf{P}_{k+2} + A_k^{-1} R_1 A_k^{-T}) A_k^T$$

Thus,

$$(\mathbf{P}_{k+1})^{-1} = A_k^{-T} (\mathbf{P}_{k+2} + A_k^{-1} R_1 A_k^{-T})^{-1} A_k^{-1}$$

By the following matrix inversion lemma,

$$(\Gamma^{-1} + \Omega)^{-1} = \Gamma - \Gamma(\Gamma + \Omega^{-1})^{-1}\Gamma$$

where Γ *and* Ω *are two non-singular. matrices,*

$$\mathbf{P}_{k+1}^{-1} = A_k^{-T} [\mathbf{P}_{k+2}^{-1} - \mathbf{P}_{k+2}^{-1} (\mathbf{P}_{k+2}^{-1} + A_k^T Q^{-1} A_k)^{-1} \mathbf{P}_{k+2}^{-1}] A_k^{-1}$$

Using Equation (32) and defining $L = (I - \mathbf{K}_k C_k)$,

$$\begin{aligned} \mathbf{P}_{k+1}^{-1} &\leq A_k^{-T} L^{-T} [\mathbf{P}_{k+2}^{-1} \\ &- (\mathbf{P}_{k+2})^{-1} L^{-1} (\mathbf{P}_{k+2}^{-1} + A_k^T R_1^{-1} A_k)^{-1} L^{-T} \mathbf{P}_{k+2}^{-1}] L^{-1} A_k^{-1} \end{aligned} \tag{33}$$

Now,

$$\mathbf{P}_{k+2}^{-1} = \mathbf{P}_{k+2}^{-1} (I - \mathbf{K}_k C_k)^{-1} = \mathbf{P}_{k+2}^{-1} L^{-1}$$

Hence,

$$L^T A_k^T \mathbf{P}_{k+1}^{-1} A_k L \leq (I - (I + \mathbf{P}_{k+2}^{-1} A_k^T Q^{-1} A_k)^{-1} L^{-T}) \mathbf{P}_{k+2}^{-1}$$

Combining the last term of Equation (29) with the first term on the right side of Equation (25),

$$\begin{aligned} & e(k) (I - \mathbf{K}_k C_k)^T A_k^T \mathbf{P}_{k+1}^{-1} A_k (I - \mathbf{K}_k C_k) e(k) \\ & \leq e(k) (1 - (1 + \bar{p}\bar{a}^2 / \underline{q})^{-1} (1 + \bar{k}\bar{c} + \lambda)^{-1}) \mathbf{P}_{k+2}^{-1} e(k) \\ & \leq (1 - \alpha) \left\| \mathbf{P}_{k+2}^{-1} \right\| \|e(k)\|^2 \end{aligned} \tag{34}$$

where $\|A_k\| = \sqrt{tr(A_k A_k)} \leq \bar{a}$, $\|C_k\| = \sqrt{tr(C_k C_k)} \leq \bar{c}$, $\|\mathbf{K}_k\| = \sqrt{tr(\mathbf{K}_k \mathbf{K}_k)} \leq \bar{k}$, $\lambda = \left\| \Lambda^{-1} \right\|$, $\underline{p} I \leq \mathbf{P}_{k+2} \leq \bar{p} I$, $\underline{q} I \leq R_1$, *and*

$$\alpha = \frac{1}{(1 + \bar{p}\bar{a}^2 / \underline{q})(1 + \bar{k}\bar{c} + \lambda)} < 1$$

Combining Equation (26), the first term of Equation (29), and Equation (34),

$$\begin{aligned} V_{k+1} &= (1 - \alpha) \left\| \mathbf{P}_{k+2}^{-1} \right\| \|e(k)\|^2 \\ &+ \lambda_{\max} [\Lambda] \bar{\rho} + \lambda_{\max} \left[\mathbf{P}_{k+1}^{-1} \right] (\bar{\rho} + \bar{s}) \\ &\leq (1 - \alpha) e(k) \mathbf{P}_{k+2}^{-1} e(k) \\ &+ \lambda_{\max} [\Lambda] \bar{\rho} + \lambda_{\max} \left[\mathbf{P}_{k+1}^{-1} \right] (\bar{\rho} + \bar{s}) \end{aligned} \tag{35}$$

Thus,

$$V_{k+1} - V_k \leq -\alpha V_k + \kappa$$

where $\kappa = \lambda_{\max}\left[\mathbf{P}_{k+1}^{-1}\right](\bar{\rho} + \bar{s}) + \lambda_{\max}[\Lambda]\,\bar{\rho}$. *If*

$$\alpha\lambda_{\min}\left[\mathbf{P}_{k+2}^{-1}\right]\|e(k)\|^2 \geq \kappa$$

then $V_{k+1} - V_k \leq 0$, $\|e(k)\|$ *decreases. Thus,* $\|e(k)\|$ *converges to Equation (23).*

3. GENETIC ALGORITHM AND SLAM FOR PATH PLANNING

Path planning is one key problem of autonomous robots. Here, the map is built by the sliding mode SLAM:

- The obstacle set is defined by $B_{obs}(t)$.
- The position is $x_r(t)$, $B_{free}(t) = B \backslash B_{obs}(t)$.
- The path planning is $f(x(t), x_S, x_T)$, $x_S = x_r(t)$.

The previous map is $B_{free}(t)$, which requires the path $f(x(t), x_S, x_T)$.

We assume the previous map is obstacle-free, the initial point is x_S, the target point is $x_T \in B$,

$$B_{free} = \{z_r \in B \mid A(z_r) \cap B_{obs} = \emptyset\}$$

the obstacle is $B_{obs} = \frac{B}{B_f \ e}$, z_r is the shape of the robot, and $A(z_r)$ is the area of the robot. The objective of the path planning is to find a path $f(x, x_S, x_T) \in B_{free}$ that allows the robot to navigate.

D is defined as the search space. We use the GA to find an optimal trajectory $f(x, x_S, x_T)$, such that

$$\min_{x \in D} f(x, x_S, x_T), \text{ where } f : D \rightarrow R \tag{36}$$

Here, we use stochastic search for GA, and each iteration includes: reproduction or selection, crossing or combination, and mutation. The population is $P(k) = \{S_1^k, S_2^k, .., S_m^k\}$ with m being the size of the population that represents the possible solutions:

(1) Every chromosome S_i^t has a solution in D

$$S_i^k = [\varsigma_l, \varsigma_{l-1}, \ldots, \varsigma_2, \varsigma_1] \text{ with } \varsigma_i \in D \ \ \forall i = 1, 2, \ldots, l$$

(2) Crossing the chromosomes. An intersection in $S_a^k = [\varsigma_l^a, \varsigma_{l-1}^a, \ldots, \varsigma_2^a, \varsigma_1^a]$ and $S_b^k = [\varsigma_l^b, \varsigma_{l-1}^b, \ldots, \varsigma_2^b, \varsigma_1^b]$ belongs to D, such that $S_a^t \cap S_b^t \neq \emptyset$; then,

$$S_{a'}^k = [\varsigma_l^a, \varsigma_{l-1}^a, \ldots \varsigma_i^{ab}, \ldots, \varsigma_2^b, \varsigma_1^b]$$
$$S_{b'}^k = [\varsigma_l^b, \varsigma_{l-1}^b, \ldots \varsigma_j^{ab}, \ldots, \varsigma_2^a, \varsigma_1^a]$$

where $S_{a'}^t$ and $S_{b'}^t$ are the next generation from two compatible chromosomes by crossing.

(3) Mutation. It replace a number of chromosomes by chromosomes in D.

The mutation operation is calculated by the fitness of each chromosome,
$P(Mut) = [fit(S_1^{Mut}), fit(S_2^{Mut}), \ldots, fit(S_n^{Mut})]$, where n is the number of mutations and fit uses the

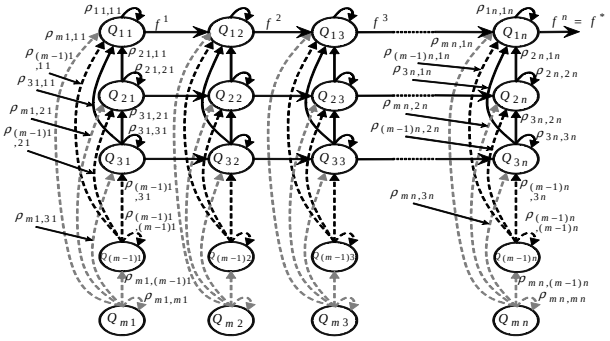

Figure 2. Roadmap genetic algorithm model as a finite Markov chain.

Euclidean distance. The total fitness is $Fit = \sum_{i=1}^{n} fit(S_i^{Mut})$. Therefore, the probability of selection p_i of a chromosome S_i for $i = 1, 2, \ldots, n$ is

$$p_i = \frac{fit(S_i^{Mut})}{Fit} \tag{37}$$

An optimal solution p^M in D is mutated by

$$p^M = \lim_{n\to\infty} (1 - p_i) = \lim_{n\to\infty} \left(1 - \frac{fit(S_i^M)}{\sum_{i=1}^{n} fit(S_i^M)} \right) = 1 \tag{38}$$

In the mutation operation, an optimal solution with $p^M = 1$ is a global solution if $n \to \infty$. To prove the convergence, we use a Markov chain, as shown in Figure 2. Each chromosome can move from Q_{ij} to the state $Q_{i(j+1)}$. The moving probability is $\rho_{ji,ik} > 0, i = 1, 2, \ldots, n, k, j = 1, 2, \ldots, m$.

The operators, selection, crossing, and mutation create $P(k)$ with p^k. It preserves the best chromosomes of $P(k-1)$. $P(k+1)$ in the population $P(k)$ can be regarded as the Markov transition:

$$H\{Q_{k+1} = p^{k+1} | Q_k = p^k\} = H(p^{k+1}, p^k) \tag{39}$$

Theorem 2 *If GA for the roadmap is an elitist process, then the probability of p^* in D is exponential.*

Proof 2 *The iteration Q_1 is changed with the chromosomes when genetic process is elitist,*

$$H(Q_1 = p^* \forall 0 < \tau \le n) = \sum_{i=2}^{n} \rho_{i1,11} = \frac{n-1}{n} \tag{40}$$

where n is the size of the population. If for all $\alpha, \beta \in D$, there is $0 < \tau \le m$ such that $H^\tau(\alpha, \beta) \ge \epsilon > 0$, then

$$\epsilon = \min \{H^\tau(\alpha, \beta) \forall 0 < \tau \le n\} \le 1 \tag{41}$$

This implies that, given certain state Q_t, the probability of transition in time t between t and $t + m$ is at least ϵ,

$$H(Q_t \ne p^* \forall t < \tau \le t + n) \ge 1 - \epsilon \tag{42}$$

Without loss of generality, the transition in the iteration $k + 1$ is

$$H(Q_{k+1}) = H(Q_t \ne p^* \forall 0 < t \le (k+1)n)$$
$$H(Q_t \ne p^* \forall 0 < t \le kmn) H(Q_t \ne p^* \forall kn < t \le (k+1)n)$$

Using Equation (42), we have

$$\begin{aligned} &H(Q_{k+1}) \le H(Q_t \ne f^* \forall 0 < t \le km)(1-\epsilon) \\ &\le H(Q_t \ne f^* \forall 0 < t \le (k-1)m)(1-\epsilon)^2 \\ &\le H(Q_t \ne f^* \forall 0 < t \le 0) H(Q_t \ne f^* \forall 0 < t \le m)(1-\epsilon)^k \\ &= \tfrac{1}{m}(1-\epsilon)^k \end{aligned}$$

where $H(P_t \ne p^* \forall 0 < t \le 0) = 1$, *then*

$$\begin{aligned} &\lim_{k\to\infty} H(Q_{k+1}) \le \lim_{k\to\infty} \tfrac{1}{n}(1-\epsilon)^k \\ &= \tfrac{1}{n} \lim_{k\to\infty} (1-\epsilon)^k = 0 \end{aligned}$$

Since $0 < \epsilon \le 1$, *the algorithm converges exponentially to* p^* *in: population size n and iteration number* k.

The algorithm of the SLAM-based roadmap GA for the path planning is as follows.

SLAM based roadmap GA. (1) Initiate population randomly $P(k)$ of size n that belong in set D. (2) The fitness value is fit with Euclidean distance for each chromosome S_i. (3) The population is from lower to higher fitness: $fit(S_1) \ge fit(S_2) \ge \cdots \ge fit(S_M)$. (4) Crossing set in the chromosomes, $S_i \cap S_j \to S_{ij}, S_{ji}$. (5) Next population $P(k+1)$ is replaced by the chromosomes with poor skills. (6) Random mutation with poor skills. (7) Go to Step (2)

---

4. AUTONOMOUS NAVIGATION

Our AN uses both sliding mode SLAM (Algorithm 1) and the roadmap GA method (Algorithm 2). The autonomous navigation algorithm is:

Autonomous navigation. The initial state S_I, the target S_T $P^- = covariance(R_1)$, $P^+ = covariance(R_2)$ ρ = gain sliding mode, r_O = search radius Map =search_obstacles(S_I, r_O) S_n = Path_Planning(Map, S_I, S_T) U_0 = Controller(S_I, S_n) while $S_n \ne S_T$

$$\left[Map_k, \hat{X}_k\right] = SM\_SLAM\left(S_k, S_n, U_k, P_k^-, P_k^+, \rho, z_k\right)$$

S_i = $\hat{X}_k$(end state) $Map\,(end : length(Map_k))$ = Map_k [S_n θ_i] =Path_Planning(Map, S_i, S_T) if $\theta_i > \pi$ $S_n = ComputeOutsideLocal(Map, S_i, S_T, \theta_i)$ end if; U_{k+1} = Controller(S_i, S_n) end while return $S_n, \hat{X}_k$

The PP needs the map, robot position, and target. This information is given by the sliding mode SLAM algorithm. When the algorithm falls into a local solution, we use the $ComputeOutsideLocal$ function to provide another S_n which is outside the local zone.

5. COMPARISONS

In this section, we use several examples to compare our method with the three other recent methods: the polar histogram method for path planning[50], the grid method for path planning[51], and SLAM with extended Kalman filter[52].

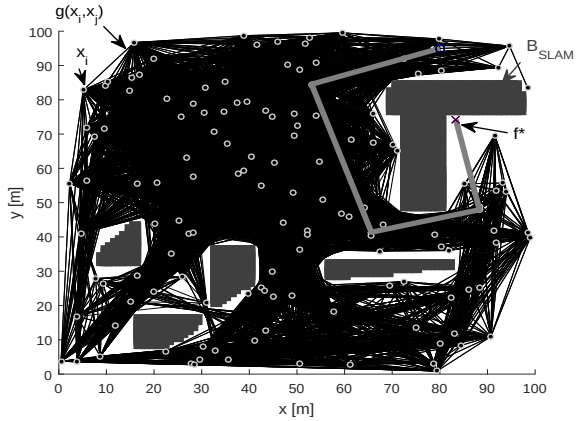

Figure 3. Sliding mode simultaneous localization and mapping.

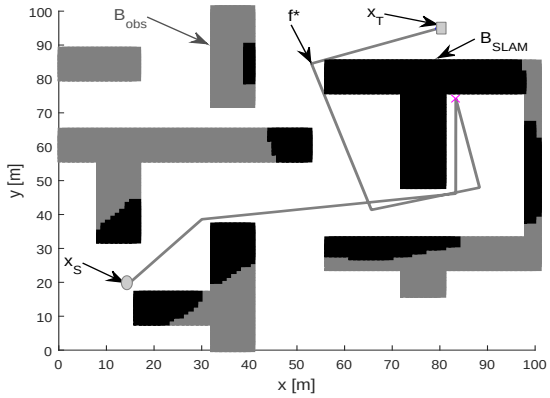

Figure 4. Autonomous navigation using sliding mode simultaneous localization and mapping and genetic algorithm method.

5.1. Simulations

The following simulations were implemented in partially unknown and completely unknown environments. The size of the environments was 100 m × 100 m, in which a solution was sought to find a trajectory from the initial point x_S to the target point x_T. The sliding mode gains were selected as $\rho = diag([0.1, \cdots, 0.1])$.

In the partially unknown environments, $B_{obs}(0) \neq \emptyset$. The path planning solution p^* was partial because the environment $B_{obs}(t)$ was variant in time. Figure 3 shows a partial solution p^* from an initial point x_S to the objective point x_T for the partially unknown environment. Figure 4 shows the overall result of the robot navigation from point x_S to point x_T with the robust SLAM algorithm combined with the GA.

Here, the SLAM algorithm was used to construct the environment and find the position of the robot. At the beginning of navigation in the partially unknown environment, there was a planned trajectory of navigation through the GA algorithm; however, if an obstacle was found in the planned trajectory, the GA algorithm needed to be used to search for a new trajectory within the built environment by the SLAM, B_{SLAM}. The planned trajectory belonged to the set of obstacles that prevent reaching the goal, $p^* \subset B_{obs}(t)$; therefore, it was necessary to look for a new trajectory using RGA that allowed reaching the goal.

For the completely unknown environments, $B_{obs}(0) = \emptyset$. In these environments, the SLAM algorithm was required to know the environment B_{SLAM} and the position of the robot; in this way, when an obstacle was found that contained the planned trajectory, a new trajectory with the GA algorithm was searched on the map

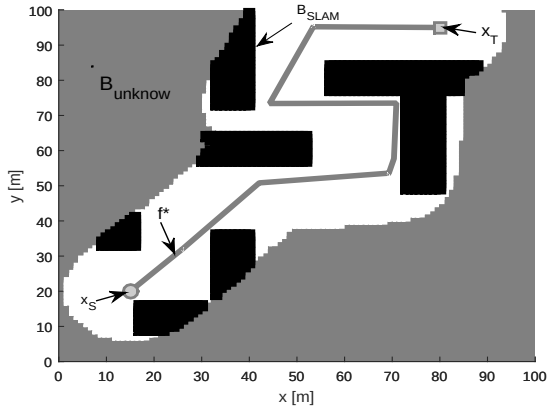

Figure 5. Sliding mode simultaneous localization and mapping and genetic algorithm in complete unknown environment.

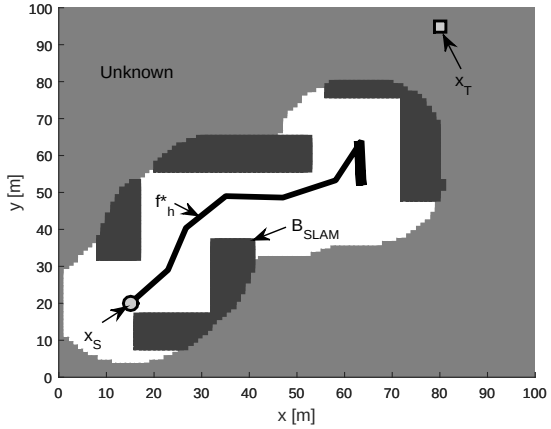

Figure 6. Polar histogram method in complete unknown environments.

B_{SLAM} until the target point was reached the results obtained are shown in Figure 5. When we used the polar histogram method for path planning[50], only the local solutions could be found [Figure 6].

Now, we compare the path lengths with the polar histogram method. The following density of the obstacles give the navigation complexity. The environment is free of obstacles when $d_{obs} = 0$. The whole environment is occupied by the obstacles when $d_{obs} = 1$. The index for the trajectory error is

$$E_{PP} = \frac{effective\ path - optimal\ path}{100} \tag{43}$$

We use the averages of the path length. The obstacles density is defined as

$$d_{obs} = \frac{\sum_{n \in G_O} S_{obs}(n)}{\|G_E\|} \tag{44}$$

The path length is defined as

$$l_{sub} = \frac{effective\ path}{optimal\ path} \tag{45}$$

The averages of the path lengths of our RA and the polar histogram are shown in Figure 7. When the density

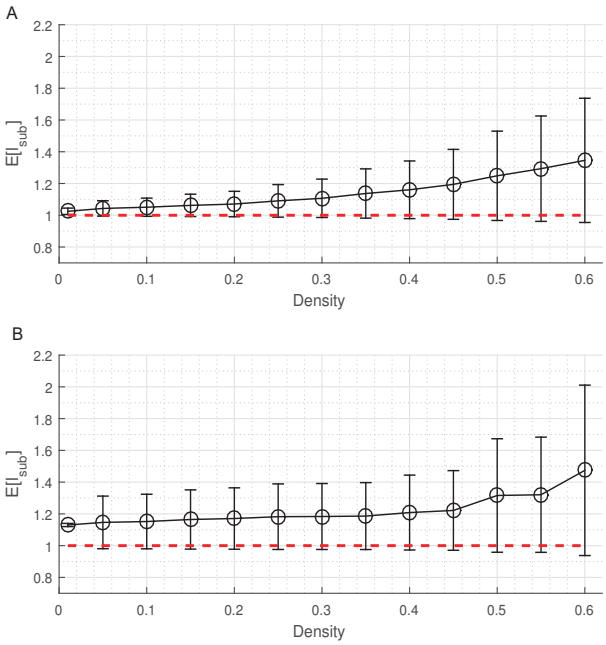

Figure 7. The average path length: (A) sliding mode simultaneous localization and mapping and genetic algorithm; and (B) polar histogram.

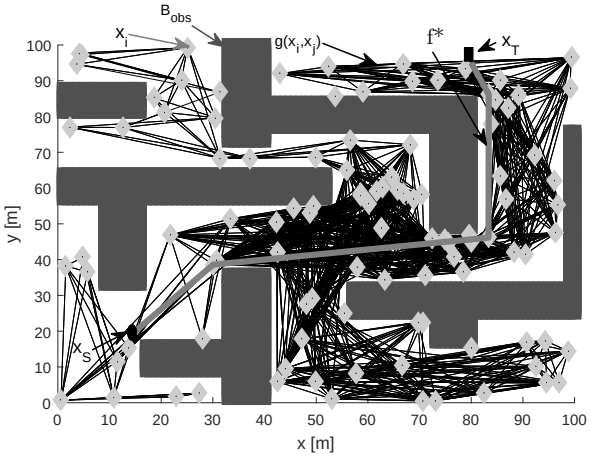

Figure 8. Sliding mode simultaneous localization and mapping (gray) and grid method (black)

of obstacles was bigger, the path length of the polar histogram grew more quickly than that of ours. When the obstacle density d_{obs} was 0.3, $E[l_{sub}, Navigation1] = 1.053$, $E[l_{sub}, Navigation2] = 1.152$.

Next, we compare our method with the grid method[51]. The comparison results are shown in Figure 8. For the task of navigating the robot or system in partially unknown or completely unknown environments, the SLAM algorithm was used to construct the environment and know the position of the robot. At the beginning of navigation in the partially unknown environment, there was a planned trajectory of navigation through the GA algorithm; however, if an obstacle were found in the planned trajectory, the GA algorithm needed to be used to search for a new trajectory within the built environment by the SLAM, B_{SLAM}.

The size of the environments was 100 m × 100 m, in which a solution was sought to find a trajectory from the initial point x_S to the target point x_T. Figure 9 shows a path planning based on the proposed methods to find

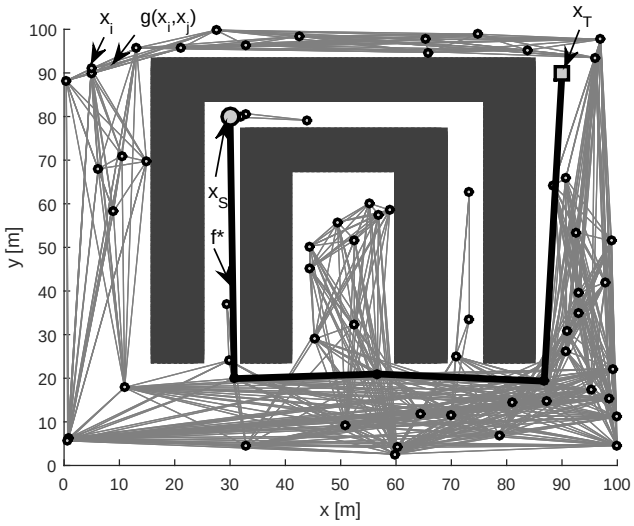

Figure 9. Sliding mode simultaneous localization and mapping based path planning (bold) and grid method (gray).

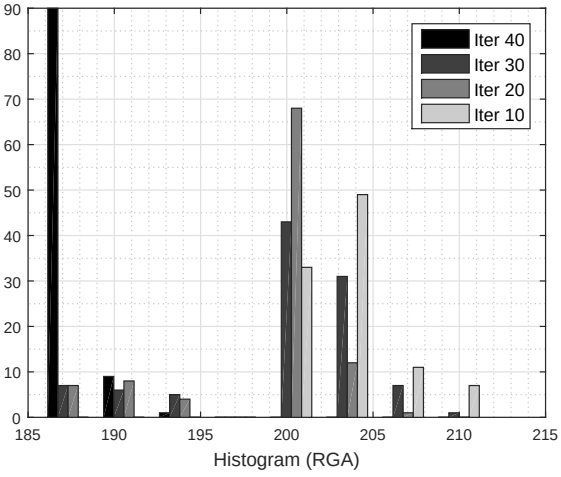

Figure 10. Performance of the genetic algorithm.

a solution f^*. Here, 60 local targets were generated x_i with the search space D generated by the trajectories of the local targets that do not intersect with the set of obstacles; thus, it became an optimization problem to find an optimal path.

In an environment with previously generated obstacles of 100 m × 100 m, 120 possible targets were randomly generated x_i; therefore, the search space D would be all trajectories $g(x_i,x_j)$ that do not intersect with the set of obstacles, where the roadmap genetic algorithm solved the problem of optimization to find a solution to the problem of path planning. For the problem presented above, we found that the proposed algorithm converged in 40 iterations. For these results, 100 tests were performed for each number of iterations and, as shown in Figure 10, the roadmap genetic algorithm converged with greater probability within 40 iterations.

5.2. Application

The Koala mobile robot by K-team Corporation 2013 was used to validate our sliding mode SLAM. This mobile robot has encoders and one laser range finder. The position precision is less than 0.1 m.

Figure 11. The environment of the autonomous navigation.

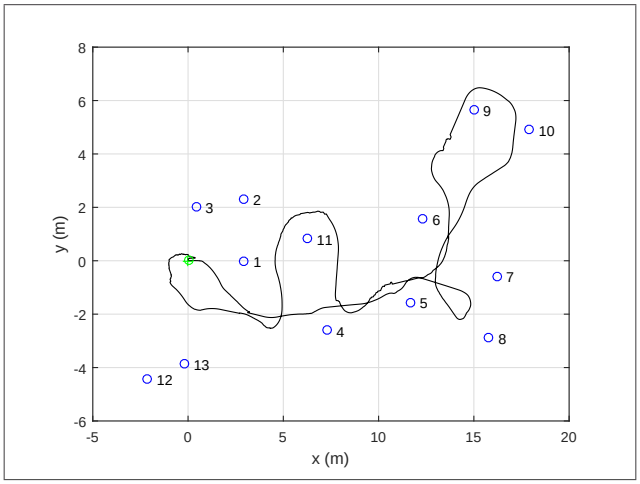

Figure 12. Results of extended Kalman filter simultaneous localization and mapping and sliding mode simultaneous localization and mapping with small noises.

The objective of this autonomous navigation is to force the robot to return to the starting point. The sliding mode SLAM was compared with SLAM with extended Kalman filter (EKF-SLAM)[52].

The initial covariance matrices are zero. The parameters of the algorithm are

$$\rho = diag([1e^{-3}, 1e^{-3}, 4e^{-3}, 2e^{-4}, \cdots, 2e^{-4}])$$
$$R_1 = diag([0.05, 0.05, 0.005]), R_2 = diag([6e^{-4}, 1e^{-5}])$$

Since the robot moves in the environment with bounded noise (see Figure 11), the noises are not Gaussian. Two different conditions are considered: (1) Koala robot pre-processes off-line the sensors data to reduce 90% noises; and (2) the computer uses sliding mode SLAM on-line.

For the first case, Figure 12 shows that sliding mode SLAM and EKF-SLAM are similar. Both sliding mode SLAM and EKF-SLAM work well for the case with less noise. The robot can return to the starting point, and the map is constructed correctly.

For the second case, Figure 13 gives the results of EKF-SLAM. As can be seen, the robot cannot return to the starting point and the map is not exactly the same as the real one with EKF-SLAM because EKF-SLAM is sensitive to non-Gaussian noises.

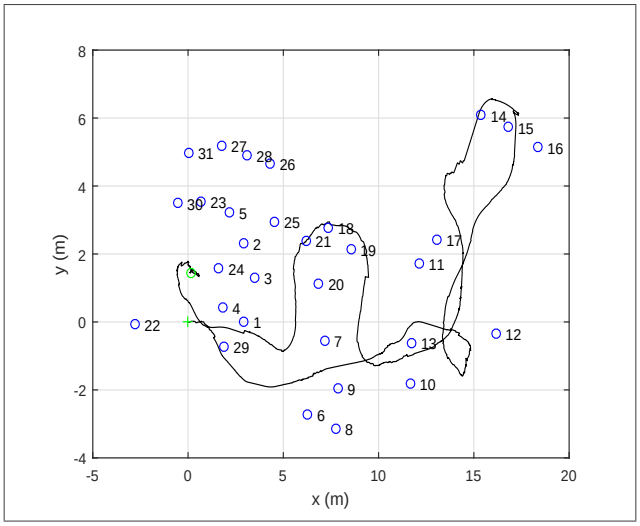

Figure 13. Results of extended Kalman filter simultaneous localization and mapping with noises.

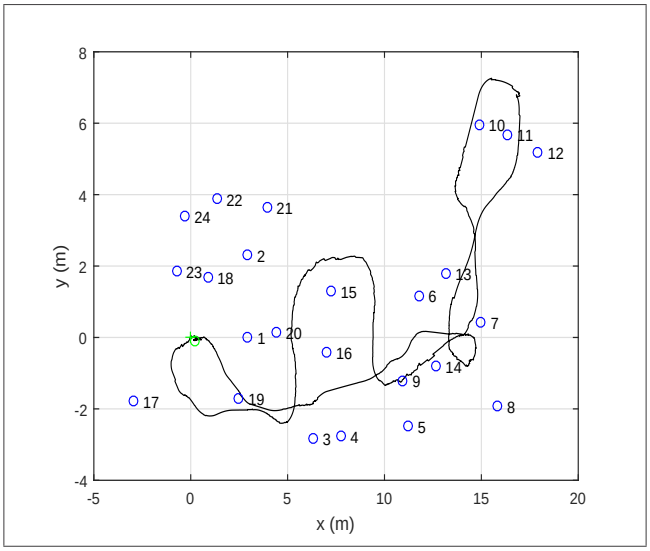

Figure 14. Results of sliding mode simultaneous localization and mapping with noise.

Figure 14 shows the results with SM-SLAM. Under the same bounded noises, SM-SLAM works very well, because of the sliding mode technique.

To compare the errors, we define the average of the Euclidean errors as

$$E_d = \frac{1}{N_T}\sum_{k=1}^{N_T}\sqrt{(x_k - x_k^*)^2 + (y_k - y_k^*)^2}, E_a = \frac{1}{N_T}\sum_{k=1}^{N_T}\left|\phi_k - \phi_k^*\right| \tag{46}$$

where N_T is the data number; x_k^*, y_k^*, and ϕ_k^* are real values for robot position and orientation; and x_k, y_k, and ϕ_k are estimations of them. Figure 15 shows the errors of EKF-SLAM and sliding mode SLAM. Obviously, the errors of EKF-SLAM increase quickly. Robots have better estimation in long distances with sliding mode SLAM.

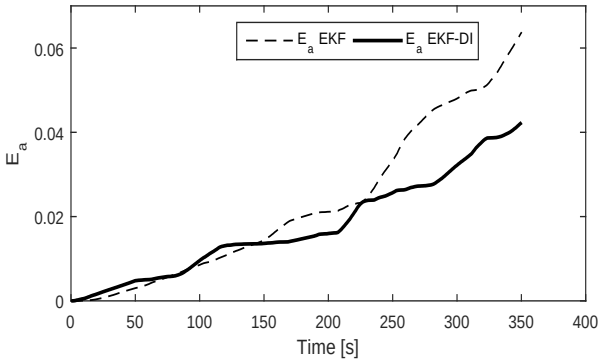

Figure 15. Direction estimation errors of sliding mode simultaneous localization and mapping (SLAM) and EKF-SLAM. EKF: Extended Kalman filter.

6. CONCLUSION

Navigation in unknown environments is a big challenge. In this paper, we propose sliding mode SLAM with genetic algorithm for path planning. Both sliding mode and GA can work in unknown environments. Convergence analysis is given. Two examples were applied to compare our model with other models, and the results show that our algorithm is much better in unknown environments.

DECLARATIONS

Authors' contributions

Revised the text and agreed to the published version of the manuscript: Ortiz S, Yu W

Availability of data and materials

Not applicable.

Financial support and sponsorship

None.

Conflicts of interest

Both authors declared that there are no conflicts of interest.

Ethical approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

REFERENCES

1. Luettel T, Himmelsbach M, Wuensche HJ. Autonomous ground vehicles—concepts and a path to the future. *Proceedings of the IEEE* 2012;100:1831-39.
2. Galceran E, Carreras M. A survey on coverage path planning for robotics. *Rob Auton Syst* 2013;61:1258-76.
3. Lowe DG. Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 2004;60:91-110.

4. Arel I, Rose DC, Karnowski TP. Deep machine learning-a new frontier in artificial intelligence research. *IEEE Comput Intell Mag* 2010;5:13-8.
5. Scheirer WJ, de Rezende Rocha A, Sapkota A, et al. Toward open set recognition. *IEEE Trans Pattern Anal Mach Intell* 2012;35:1757-72.
6. Ramos FT, Kumar S, Upcroft B, et al. A natural feature representation for unstructured environments. *IEEE Trans Robot* 2008;24:1329-40.
7. Lillywhite K, Lee DJ, Tippetts B, et al. A feature construction method for general object recognition. *Pattern Recognition* 2013;46:3300-14.
8. Chabini I, Lan S. Adaptation of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks. *IEEE trans Intell Transp Syst* 2002;3:60-74.
9. LaValle SM, Kuffner JJ Jr. Randomized kinodynamic planning. *Int J Rob Res* 2001;20:378–400.
10. Valencia R, Andrade-Cetto J. Mapping, planning and exploration with Pose SLAM. Berlin: Springer; 2018. p. 60-84.
11. Wang X, Shi Y, Ding D, et al. Double global optimum genetic algorithm–particle swarm optimization-based welding robot path planning. *Eng Optim* 2016;48:299-316.
12. Jones M, Peet MM. A generalization of Bellman's equation with application to path planning, obstacle avoidance and invariant set estimation. *Automatica* 2021;127:109510.
13. Rehman NU, Kumar K, Abro GeM. Implementation of an autonomous path plan-ning & obstacle avoidance UGV using SLAM. 2018 International Conference on Engineer-ing and Emerging Technologies (ICEET); 2018 Feb 22-23; Lahore, Pakistan. IEEE; 2018. p. 1-5.
14. de Moura Souza G, Toledo CFM. Genetic algorithm applied in UAV's path planning. 2020 IEEE Congress on Evolutionary Computation (CEC); 2020 Jul 19-24; Glasgow, UK. IEEE; 2020. p. 1-8.
15. Zhang X, Zhao Y, Deng N, et al. Dynamic path planning algorithm for a mobile robot based on visible space and an improved genetic algorithm. *Int J Adv Robot Syst* 2016;13:91.
16. Clemens J, Reineking T, Kluth T. An evidential approach to SLAM, path planning, and active exploration. *Int J Approx Reason* 2016;73:1-26.
17. Chen Y, Huang S, Fitch R. Active SLAM for mobile robots with area coverage and obstacle avoidance. *IEEE ASME Trans Mechatron* 2020;25:1182-92.
18. da Silva Arantes M, Toledo C F M, Williams B C, et al. Collision-free encoding for chance-constrained nonconvex path planning. *IEEE Trans Robot* 2019;35:433-48.
19. Yu W, Zamora E, Soria A. Ellipsoid SLAM: a novel set membership method for simultaneous localization and mapping. *Autonomous Robots* 2016;40:125-37.
20. Williams H, Browne WN, Carnegie DA. Learned action slam: sharing slam through learned path planning information between heterogeneous robotic platforms. *Appl Soft Comput* 2017;50:313-26.
21. Dissanayake MWMG, Newman P, Clark S, et al A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans Rob Autom* 2001;17:229-41.
22. Thrun S, Liu Y, Koller D, et al. Simultaneous localization and mapping with sparse extended information filters. *Int J Robot Res* 2004;23:693-716.
23. Folkesson J, Christensen HI. Closing the loop with graphical SLAM. *IEEE Trans Robot* 2007;23:731-41.
24. Ho KL, Newman P. Loop closure detection in SLAM by combining visual and spatial appearance. *Rob Auton Syst* 2006;54:740-9.
25. Nieto J, Guivant J, Nebot E. Denseslam: simultaneous localization and dense mapping. *Int J Robot Res* 2006;25:711-44
26. Chen SY. Kalman filter for robot vision: a survey. *IEEE Trans Ind Electron* 2011;59:4409-20.
27. Sibley G, Matthies L, Sukhatme G. Sliding window filter with application to planetary landing. *J Field Robot* 2010;27:587-608.
28. Kaess M, Johannsson H, Roberts R, et al. iSAM2: Incremental smoothing and mapping using the Bayes tree. *Int J Robot Res* 2012;31:216-35.
29. Yang S, Scherer SA, Yi X, et al. Multi-camera visual SLAM for autonomous navigation of micro aerial vehicles. *Rob Auton Syst* 2017;93:116-34
30. Weiss S, Scaramuzza D, Siegwart R. Monocular-SLAM–based navigation for autonomous micro helicopters in GPS-denied environments. *J Field Robot* 2011;28:854-874
31. Zhu A, Yang SX. Neurofuzzy-based approach to mobile robot navigation in unknown environments. *IEEE Trans Syst Man Cybern C Appl Rev* 2007;37:610-21.
32. Juang CF, Chang YC. Evolutionary-group-based particle-swarm-optimized fuzzy controller with application to mobile-robot navigation in unknown environments. *IEEE Trans Fuzzy Syst* 2011;19:379-92.
33. Villacorta-Atienza JA, Makarov VA. Neural network architecture for cognitive navigation in dynamic environments. *IEEE Trans Neural Netw Learn Syst* 2013;24:2075-87.
34. Song B, Wang Z, Sheng L. A new genetic algorithm approach to smooth path planning for mobile robots. *Assem Autom* 2016;36:138-45.
35. Zhang Y, Gong D, Zhang J. Robot path planning in uncertain environment using multi-objective particle swarm optimization. *Neurocomputing* 2013;103:172-85.
36. Karami AH, Hasanzadeh M. An adaptive genetic algorithm for robot motion planning in 2D complex environments. *Comput Electr Eng* 2015;43:317-29.
37. Pereira AGC, de Andrade BB. On the genetic algorithm with adaptive mutation rate and selected statistical applications. *Comput Stat* 2015;30:131-50.
38. Tsai CC, Huang HC, Chan CK. Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation. *IEEE Trans Ind Electron* 2011;58:4813-21.

39. Zhu Q, Hu J, Cai W, et al. A new robot navigation algorithm for dynamic unknown environments based ondynamic path re-computation and an improved scout ant algorithm. *Appl Soft Comput* 2011;11:4667-76.
40. Arantes MS, Arantes JS, Toledo CFM, et al. A hybrid multi-population genetic algorithm for uav path planning. Proceedings of the Genetic and Evolutionary Computation Conference 2016; 2016 Jul 20. New York, NY: Association for Computing Machinery; 2016. p. 853-60.
41. Tuncer A, Yildirim M. Dynamic path planning of mobile robots with improved genetic algorithm. *Comput Electr Eng* 2012;38:1564-72.
42. Raja R, Dutta A, Venkatesh KS. New potential field method for rough terrain path planning using genetic algorithm for a 6-wheel rover. *Rob Auton Syst* 2015;72:295-306.
43. Arora T, Gigras Y, Arora V. Robotic path planning using genetic algorithm in dynamic environment. *Int J Comput Appl* 2014;89:9-12.
44. Roberge V, Tarbouchi M, Labonté G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *IEEE Trans Industr Inform* 2012;9:132-41.
45. Bhandari D, Murthy CA, Pal SK. Genetic algorithm with elitist model and its convergence. *Intern J Pattern Recognit Artif Intell* 1996;10:731-47.
46. Hu ZB, Xiong SW, Su QH, et al. Finite Markov chain analysis of classical differential evolution algorithm. *J Comput Appl Math* 2014;268:121-34.
47. K-Team Corporation. Available from: http://www.k-team.com/ [Last accessed on 10 Dec 2021]
48. Souahi A, Naifar O, Makhlouf AB, et al. Discussion on Barbalat Lemma extensions for conformable fractional integrals. *Int J Control* 2019;92:234-41.
49. Erdman AG, Sandor GN, Kota S. Mechanism design: analysis and synthesis. Upper Saddle River, NJ: Prentice hall; 2001.
50. Wilde N, Kulić D, Smith SL. Learning user preferences in robot motion planning through interaction. 2018 IEEE International Conference on Robotics and Automation (ICRA); 2018 May 21-25; Brisbane, QLD, Australia. IEEE; 2018. p. 619-26.
51. Farzan S, DeSouza GN. Path planning in dynamic environments using time-warped grids and a parallel implementation. *arXiv:1903.07441* 2019.
52. Saeedi S, Nardi L, Johns E, et al. Application-oriented design space exploration for SLAM algorithms. 2017 IEEE International Conference on Robotics and Automation (ICRA); 2017 Jul 24; Singapore. IEEE; 2017. p. 5716-23.