2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Long-Term Localization With Time Series Map Prediction for Mobile Robots in Dynamic Environments

Lisai Wang, Weidong Chen*, and Jingchuan Wang

Abstract— **In many applications of mobile robot, the environment is constantly changing. How to use historical information to analysis environmental changes and generate a map corresponding with current environment is important to achieve high-precision localization. Inspired by predictive mechanism of brain, this paper presents a long-term localization approach named ArmMPU (ARMA-based Map Prediction and Update) based on time series modeling and prediction. Autoregressive moving average model (ARMA), a kind of time series modeling method, is employed for environmental map modeling and prediction, then predicted map and filtered observation are fused to fix the prediction error. The simulation and experiment results show that the proposed method improves long-term localization performance in dynamic environments.**

I. INTRODUCTION

Long-term localization is a significant topic of mobile robots. Typical localization methods [1] build a static map and localize on that, while in reality, the environment for long-term work is constantly changing. The existing long-term localization methods cannot generate maps well corresponding with current environment, or can only perform well in periodic environments. Moreover, the error between generated map and real environment is not well used.

The human brain utilizes historical information by generating external models and predicting external environment state [2], thus compressing information and finishing recognition of environment efficiently. Imitating predictive mechanism of brain, robots can use long-term information by modeling and predicting the environment changes.

Inspired by predictive mechanism of brain, we present a time series based map prediction and update method, namely ArmMPU. In our previous work [3], time series model was used in a long-term monocular VI SLAM system to predict map points. In this paper, we extend previous work by importing prediction errors in map update for fusing prediction with observation efficiently and accurately, and using a variable threshold data filter based on localizability [4] and matching degree to reduce the observation uncertainty, moreover extending the sensor to 2D LiDAR. Environmental maps are modeled and predicted by ARMA [5] to generate predicted maps, and the observation is filtered by the variable threshold data filter. Then prediction and filtered observation are fused by a Bayesian filter based on prediction errors to update predicted map and get more accurate maps, so as to get an improved localization performance.

The authors are with Institute of Medical Robotics and Department of Automation, Shanghai Jiao Tong University, and Key Laboratory of System Control and Information Processing, Ministry of Education, Shanghai 200240, China. (E-mail: lisaiw@sjtu.edu.cn; *corresponding author: wdchen@sjtu.edu.cn; jchwang@sjtu.edu.cn)

II. RELATED WORK

Related long-term researches can be divided into the following three categories.

The first class is map update and localization without spatio-temporal modeling. Saarinen et al. [6] defined each map grid as an independent Markov chain with two states, and updated grid state based on a probabilistic sensor model. Tipaldi et al. [7] used Hidden Markov Model to represent the dynamic properties of 2D map grids. Sun et al. [8] applied scan-matching [9] updated the map when a matching score exceeded the threshold. This kind of method focuses on spatial modeling and online learning of maps, while lacking modeling in time dimension. When the environment changes greatly, the initial matching degree between observation and the map will be too low to update the map.

The second type of method models the map in time dimension by analysing historical information. Biber et al. [10] proposed a sampling-based environment representation method. It described environment with the combination of scenes at different time. Dayoub et al. [11], [12] used attention mechanism to transfer the interested features to short-term memory maps, then selected stable features update to long-term memory maps. These methods make use of the historical information, but they are mainly based on combinations of information and do not have prediction, so they cannot predict accurately in the future time.

Krajnik et al. [13]–[15] transformed the environment changes from time domain into the frequency domain to modeling and prediction. This method (FreMEn) provided an accurate predicted map for localization, while it assumed the environmental changes were periodic, could not predict the aperiodic changes, and is easy interfered by random changes.

The third kind of methods is to deal long-term problems through deep learning. Schreiber et al. [16] achieved long-term grid prediction using recurrent neural networks. Chen et al. [17] achieved localization through visual place recognition, matching the appearance of current scene to a previously visited place based on two CNN architectures. The grid prediction methods by deep learning usually focus on dynamic object movement, and the prediction is only in several seconds. The visual place recognition methods can achieve a long time span localization, while the appearance match cannot perform well when environmental structure changes greatly or there are many dynamic obstacles. Besides, deep learning based methods are usually with significant computational overhead for mobility solutions.

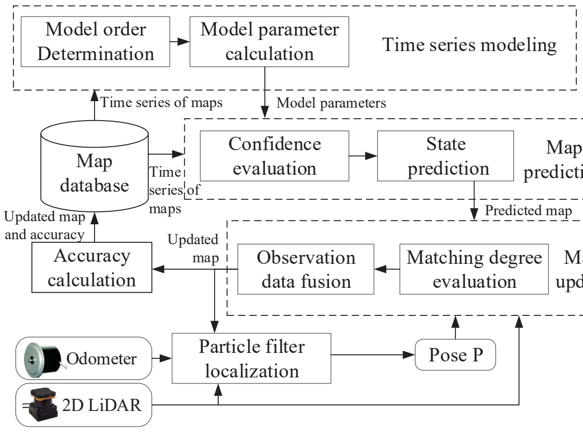


Fig. 1. Overall system architecture.

III. SYSTEM OVERVIEW

The proposed ArmMPU achieves environmental map modeling and prediction, as well as predicted map and observation fusion. Its architecture is shown in Fig. 1. It mainly includes three modules: time series modeling module, map prediction module, and map update module. And a particle filter is employed for localization.

To achieve long-term localization, a map database is built for organizing all maps. The map database consists of two kinds of maps: the original maps and the updated maps. The original maps are constructed by the SLAM program [18] at first for the initial modeling. Each mapping route is same for the consistency of maps. The model orders and model parameters for time series of map grids are calculated and regular updated in the time series modeling module. Then the map prediction module outputs predicted maps on time dimension based on received model parameters and map database data. The map update module filters the observation and fuses it with the predicted map, and outputs real-time updated maps. Finally, the updated maps are used for robots localization and sent back to the database.

The map database is composed of L, the time series of maps. $L = \{L_1, L_2, \cdots L_t \cdots, L_T\}$, each L_t represents a occupancy grid map at time t, and T is the duration of time series. $L_1, L_2, \cdots, L_M$ are the original maps, where T_M is time duration for model building. The maps are aligned to get the time series of each grid. The time interval of maps t_{mr} is uniform, and when a map at a certain moment is absent, it is supplemented by copying the map at adjacent moment.

$$L_t = \begin{bmatrix} l(1,1)_t & l(1,2)_t & \cdots & l(1,J)_t \\ l(2,1)_t & l(2,2)_t & \cdots & l(2,J)_t \\ \vdots & \vdots & \vdots & \vdots \\ l(I,1)_t & l(I,2)_t & \cdots & l(I,J)_t \end{bmatrix}, \tag{1}$$

where I and J are rows and columns of the map, and $l(i,j)_t$ is the state of a single grid in raw i and column j with a value in $[0,1]$, indicating the occupancy probability of each grid. The occupancy is implemented in log odds form [1], and the greater the value, the greater the occupancy probability.

IV. ALGORITHMS

The principle, formula and algorithm in ArmMPU are introduced in this section.

A. Time series modeling and map prediction

Some objects in the environment change with certain regularity. Map prediction can predict those changes and generate maps matching with current environment, and reduce the failure of map update caused by incompatibility between observation and the original map.

In this paper, 2D grid maps are used to describe the environment, and autoregressive moving average model (ARMA) [5] is built for each grid. ARMA is a statistical model for time series, providing description of stationary stochastic process in terms of polynomials. ARMA can not only model the periodic changes, but also aperiodic changes with regularity, and has good adaptability to random changes. ARMA model is defined as equation (2).

$$\hat{x}(i,j)_t = \sum_{i=1}^{p} \hat{\alpha}_i \hat{x}(i,j)_{t-i} + \hat{\varepsilon}(i,j)_t - \sum_{j=1}^{q} \hat{\beta}_j \hat{\varepsilon}(i,j)_{t-j}, \tag{2}$$

where $t = p+1, p+2, \cdots T$, p and q are the model order, $\hat{\alpha}$ and $\hat{\beta}$ represent the model parameters, which are independent for each grid. $\hat{\varepsilon}(i,j)_t$ represents white noise and $\hat{x}(i,j)_t$ represents the state of grid in (i,j) at t time, and $\hat{x}(i,j)_{t-i}$ is same as $l(i,j)_{t-i}$, which is got from the map database.

The map database should be established before time series modeling of grids, then the state of each grid is predicted in time domain. The process includes model order determination, parameter calculation, state prediction, confidence evaluation and model update.

1) Model order determination: The model order p and q denote the temporal correlation of the grid states and the prediction error. They are determined at first according to AIC [19]. The AIC criterion is used:

$$AIC_{i,j}(k,h) = \ln \hat{\sigma}^2(k,h) + \frac{2(k+h)}{n}, \tag{3}$$

where n is the length of the time series.

$$\begin{aligned} \hat{\sigma}^2(k,h) &= \frac{1}{n} \min \sum_{t=1}^{n} \left(\sum_{j=1}^{h} \hat{\beta}_j \hat{\varepsilon}(i,j)_{t-j} + \hat{x}(i,j)_t - \sum_{i=1}^{k} \hat{\alpha}_i \hat{x}(i,j)_{t-i} \right) \\ &= \frac{1}{n} \min \sum_{t=1}^{n} \hat{\varepsilon}(i,j)_t^2 \end{aligned}, \tag{4}$$

The (k,h) that minimizes the AIC value is the the required model order p and q.

2) Model parameter calculation: Then the model parameters $\alpha \in R^p$ and $\beta \in R^q$ are determined according to autoregressive approximation [20] as follow steps, which characterize the grid state relationship in time domain.

Firstly, the Autoregressive Models AR(p) is fitted using historical grid data. Then, recursively calculate the residual sequence $\{\hat{\varepsilon}(i,j)_t\}$ based on the above AR(p).

$$\hat{\varepsilon}(i,j)_t = \hat{x}(i,j)_t - \hat{\alpha}_1 \hat{x}(i,j)_{t-1} - \hat{\alpha}_2 \hat{x}(i,j)_{t-2} \cdots - \hat{\alpha}_p \hat{x}(i,j)_{t-p}, \tag{5}$$

Finally, regard the residual sequence $\{\hat{\varepsilon}(i,j)_t\}$ as an independent sequence. According the linear regression model shown as equation (2), the matrix form of linear regression model is:

$$\hat{x}(i,j) = \left(Z\hat{E}\right)\begin{pmatrix}\alpha\\ \beta\end{pmatrix} + \hat{\varepsilon} \tag{6}$$

where Z and $\hat{E}$ are matrix form of $\hat{x}(i,j)_{t-i}$ and $\hat{\varepsilon}(i,j)_{t-j}$ in equation (2). Get the estimated parameters as equation (7).

$$\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \left[\begin{pmatrix}Z'\\ \hat{E}'\end{pmatrix}\begin{pmatrix}Z & \hat{E}\end{pmatrix}\right]^{-1}\begin{pmatrix}Z'\\ \hat{E}'\end{pmatrix}\hat{x} = \begin{bmatrix}Z'Z & Z'\hat{E}'\\ \hat{E}'Z & \hat{E}'\hat{E}\end{bmatrix}\begin{pmatrix}Z'\hat{x}\\ \hat{E}'\hat{x}\end{pmatrix}, \tag{7}$$

3) State prediction: Predicted state of each grid $\hat{x}(i,j)_t$ at time t is got as equation (2), then it is binarized to 0 and 1 to get $x(i,j)_t$ for subsequent map update and localization. The whole predicted map is obtained by combining all the grids as equation (1).

4) Confidence evaluation: To evaluate the predicted credibility, the prediction confidence $G(i,j)$ is calculated inspired by [21] after model building as euqation (8). The maps used in modeling divided into two parts: one for generating predicted states, and another used as ground truth.

$$\begin{gathered} G(i,j) = G_0 + \frac{\sum_{t=1}^{T_p} d_t(i,j)}{T_p}, \\ d_t(i,j) = \begin{cases} d, & \hat{x}(i,j)_t = round\left(l(i,j)_t\right) \\ -d, & \hat{x}(i,j)_t \neq round\left(l(i,j)_t\right) \end{cases}, \end{gathered} \tag{8}$$

where, (i,j) is raw and columns of grid, $\hat{x}(i,j)_t$ is predicted states, $l(i,j)_t$ is true states, d is step, G_0 is initial belief, T_p is prediction length, $G(i,j)$ is prediction confidence.

The confidence is evaluated after building model. If $G(i,j)$ is lower than confidence threshold G_{th}, the model order is reselected, and model parameters are recalculated.

5) Model update: To maintain the accuracy of model, a model update mechanism is necessary. The average prediction accuracy $s(i,j)$ of each grid is calculated as equation (9) at regular interval T_{up}. When the prediction accuracy is less than a certain threshold, the model parameters α and β are recalculated using the data of latest a period of time T_M, which is same as the time duration for initial model building.

$$s(i,j) = \frac{\sum_{i=1}^{T_{up}} F_t}{T_{up}}, \quad F_t = \begin{cases} 1, & S_{pre} = S_{truth} \\ 0, & S_{pre} \neq S_{truth} \end{cases}. \tag{9}$$

S_{pre} and S_{truth} are the binarization of predicted state and updated state of grid. S_{th} are the model update threshold.

B. Map Update

After prediction, it is necessary to fuse current observation with predicted map to fix the prediction error and get more accurate map.

To reduce the observation uncertainty, and avoid introducing inaccurate information due to localization error, it is essential to establish a data filtering mechanism. According

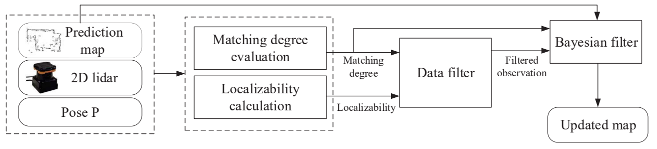


Fig. 2. Framework of map update.

to our previous work [22], a variable threshold data filter mechanism is employed here.

$$M = \lambda m_1 + (1-\lambda)m_2 > M_{th}, \tag{10}$$

where λ is the matching degree weight, m_1 is the matching degree between predicted map and LiDAR date, which reflects the prediction error; m_2 is the normalized value of localizability [4], which reflects the richness of environmental features at specific pose. M_{th} is the update threshold. Transform equation (10) to the follow:

$$\begin{cases} m_1 > \dfrac{M_{th} - (1-\lambda)m_2}{\lambda}, & trigger\ update \\ \qquad else \quad , & not\ update \end{cases} \tag{11}$$

Right of the inequality is the variable threshold of data filter. There is usually a relatively high matching degree in feature-poor pose. From equation (11), it can be seen that in the feature-rich pose, the lower matching degree can trigger the update; while in the feature-poor pose, a higher matching degree is required.

The m_1 and m_2 can be calculated as follows:

$$\begin{gathered} m_1 = \frac{\sum_{i=1}^{N} w_i}{N}, \\ w_i = \exp^{-d_i}, \end{gathered} \tag{12}$$

where w_i is the normalized value of matching degree between the i-th laser beam end with an obstacle in the map. d_i is the distance between the end of laser beam and corresponding grid. N is the number of laser beams.

$$m_2 = \det(\hat{\Im}(P)) = \det(\sum_{i=1}^{N_0}\begin{bmatrix} \frac{\Delta\hat{r}_{iE}^2}{\Delta x^2} & \frac{\Delta\hat{r}_{iE}^2}{\Delta x\Delta y} & \frac{\Delta\hat{r}_{iE}^2}{\Delta x\Delta\theta} \\ \frac{\Delta\hat{r}_{iE}^2}{\Delta x\Delta y} & \frac{\Delta\hat{r}_{iE}^2}{\Delta y^2} & \frac{\Delta\hat{r}_{iE}^2}{\Delta y\Delta\theta} \\ \frac{\Delta\hat{r}_{iE}^2}{\Delta x\Delta\theta} & \frac{\Delta\hat{r}_{iE}^2}{\Delta y\Delta\theta} & \frac{\Delta\hat{r}_{iE}^2}{\Delta\theta^2} \end{bmatrix}) \tag{13}$$

where $P = (x,y,\theta)$ is the pose of robot. $\hat{r}_{iE}^2$ is the desired distance if the i_{th} laser beam in the LRF model. N_0 is the laser beams' number.

After filtering the observation, a Bayesian filter is used for data fusion. The predicted map is combined with observation to obtain an accurate map. As the robot running, the map is constantly updated. Dynamic occupancy grid map [23] and HMMs [24] are used here. High matching degree indicates accurate localization and low prediction error. The higher the matching degree, the greater weight of the predicted map. This allows the map to be updated relatively quickly as well as avoiding introducing inaccurate information. The fusion is performed as equation (14).

$$Q_{t,k+1} = \left[Q_{t,k}A_c(1-m_1)+X_t m_1\right]B_z\eta,$$
$$A_c = \begin{bmatrix} 1-p_c^{f|o} & p_c^{f|o} \\ p_c^{o|f} & 1-p_c^{o|f} \end{bmatrix}, \qquad (14)$$
$$B_z = \begin{bmatrix} p(z \mid c=occ) & 0 \\ 0 & p(z \mid c=free) \end{bmatrix},$$

where $Q_{t,k} = \left[p_c^{o|f} \;\; p_c^{f|o}\right]$ is the k-th updated grid state at time t. A_c is the state transition matrix, which is identified through Expectation-Maximization algorithm in our experiments [23]. B_z is the observation, which is calibrated in advance and is the same for each grid. m_1 is the matching degree, $X_t = [x_t \;\; 1-x_t]$ and x_t is the predicted state, and η is the normalization factor.

In the fusion process, updated maps matching the environment can be got. The particle filter localization algorithm is used [1] for obtaining robot pose P based on updated maps. When the robot stops running or one time interval t_{mr} ends, the updated map is sent back to the map database.

V. SIMULATION

Simulations are done in MATLAB using the parking area scene. Periodical, gradually incremental and random changes are set.

A. Simulation platform and parameters setting

The parking area is 50 m × 50 m, with 32 parking spaces and 8 pillars. Parameters for model building and update are shown in Table I, which are same in experiment except T_M.

TABLE I
PARAMETERS IN SIMULATION AND EXPERIMENT

d	G_0	G_{th}	S_{th}	t_{mr}	T_{up}	T_M
0.5	0.5	0.75	0.7	1 hour	24 hours	168 hours

TABLE II
PARAMETERS FOR CHANGES

T_w (h)	T_d (h)	D	S_{th}	(μ_1,σ_1)	δ (h)	(μ_2,σ_2)
168	8, 12, 24	1/2, 1/3	0.5, 1, 1.5	(0.1∼0.45, 0.1∼0.5)	24, 72, 96	$(0.3, 0.35)$

Two scenes are set in simulation. In scene 1, the change is periodic superimposed randomness; in scene 2, the change is gradually incremental superimposed randomness, so that the effects of different changes can be respectively evaluated.

The form of periodicity, gradually increment and randomness are shown in Fig. 4. Periodic changes are set with daily period T_d, weekly period T_w and daily parking ratio D. For gradual increment, the D is gradual extension at rate k. For randomness, Gaussian noise $N\left(\mu_1, \sigma_1^2\right)$ is introduced in the grid occupation probability, and some parking spaces are selected to disappear in a period of time δ based on Gaussian distribution $N\left(\mu_2, \sigma_2^2\right)$. Parameters are shown in Table II, which are unequal for different parking spaces, and detail correspondence can be found in the attached video.

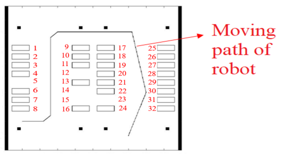


Fig. 3. Maps in simulation. The figues are serial numbers of parking spaces.

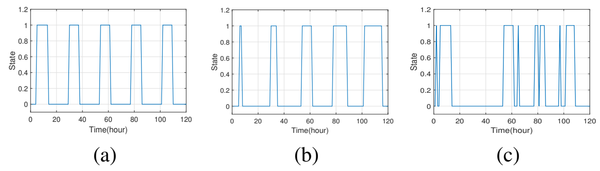


Fig. 4. Forms of changes. (a) Periodic. (b) Gradually incremental. (c) Randomness in periodicity.

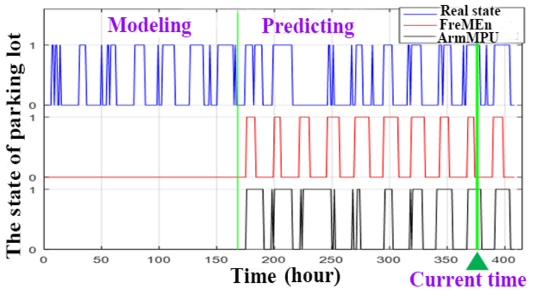


Fig. 5. State of the 16th parking space.

Time series of maps of 17 days are generated, with a resolution of 0.1 m. The states of parking spaces are modeled by ArmMPU and FreMEn [15] in two scenes. The prediction accuracy and localization performance are compared.

B. Map prediction

The models are built using previous 7-day data to predict maps in the next 10 days. The FreMEn code is implemented based on the author's open source code.

The 16th parking space with the gradually incremental change in scene 2 is shown in Fig. 5, and the car disappeared in the 10th day. ArmMPU can predict the incremental trend through state regression, and can correct the prediction by error slip for randomness. Although the prediction error was large on the 11th day, it had readjusted from the 12th day, while FreMEn can only extract the periodicity of changes.

TABLE III
AVERAGE PREDICTION ACCURACY IN SIMULATION

Algorithm	Average prediction accuracy	
	Scene 1	Scene 2
FreMEn	85.03%	65.44%
ArmMPU	86.82%	83.88%

The average prediction accuracy pre_ave is obtained by calculating the ratio of the correctly predicted parking spaces to the total parking spaces. The average prediction accuracy of 10 days is shown as Table III.

C. Localization performance

The localization performance of our method, FreMEn, the dynamic map method [7] and the static map method [1] are compared, and they all locate based on the particle filter.

Odometer and 2D LiDAR data used for localization are generated based on maps and moving path, and the noises respectively are Gaussian noise $N(0.03, 0.0004^2)$ and $N(0, 0.02^2)$. Several moments in different days are selected to test. In each run, the robot moves along a fixed path shown in Fig. 3, with a total of $n = 1801$ points. Parameters for localization and map update are shown in Table IV.

TABLE IV
PARAMETERS FOR SIMULATION

Particle number	Update threshold	Update weights	Loss threshold
100	$M_{th} = 0.6$	λ=0.7	1 m or 0.5 rad

The matching degree m_1 along the path in a certain run and corresponding maps are shown in Fig. 6. To show the effect of m_1, a larger update threshold $M_{th} = 0.7$ is set. In region A and B, the prediction is accurate, m_1 is relatively large, and the parking spaces nearby are updated.

The distance and angle error d_{eer} and θ_{eer} in a run are calculated as equations (15) and (16). (x_i, y_i, θ_i) and $(\hat{x}_i, \hat{y}_i, \hat{\theta}_i)$ are localization result and true value at the point i. Multiple test are averaged to obtain the average localization error.

$$d_{eer} = \frac{\sum_{i=1}^{n} \sqrt{(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2}}{n}, \tag{15}$$

$$\theta_{eer} = \frac{\sum_{i=1}^{n} \left|\theta_i - \hat{\theta}_i\right|}{n}. \tag{16}$$

When the localization error of a point exceeds a certain threshold, localization in this point is defined lost, and is not counted in the the average localization error. The localization loss rate is computed as the ratio of the lost points to the total test points. After 10 tests in each scene. The localization performance are shown as Fig. 6. The localization error of static map is not calculated since its loss rate is too high.

In both scenes, the localization performance of our method and FreMEn is better than dynamic map method and static map method, since ArmMPU and FreMEn have prior predicted maps corresponding to the environment.

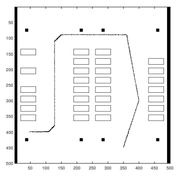
The prediction accuracy and localization performance of ArmMPU are slightly better than FreMEn in scene 1. In scene 2 where the changes are gradually incremental and random, performance of ArmMPU are much better than FreMEn. This is still because ArmMPU can better deal with aperiodic changes with regularity and changes.

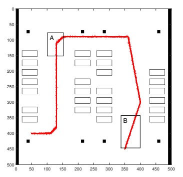
VI. EXPERIMENTS

Experiments are performed in indoor parking area and laboratory. Parameters in experiment are shown in Table I and V except T_M, which is 192 hours. The ground truth for map prediction are reference maps built by GMapping [18]. The ground truth for localization are measured by ruler.

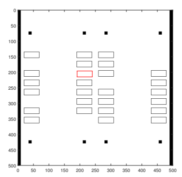
A. Experiment platform

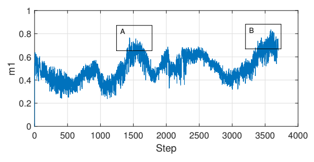
The JiaoLong intelligent wheelchair is used as experiment platform. Odometer and two 2D LiDARs (SICK TIM571) are employed here. An industrial personal computer with i7-6820EQ CPU is employed, installed with Ubuntu and ROS.


(a) Reference map

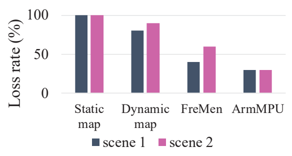


(b) Predicted map

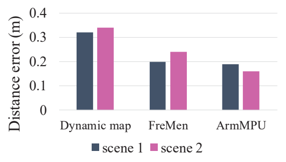

(c) Updated map



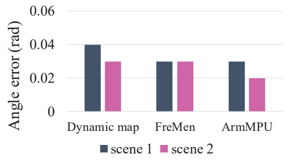
(d) m_1



(e) Loss rate



(f) Distance error

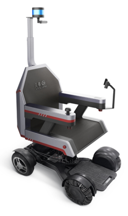


(g) Angle error

Fig. 6. Map update and localization performance. The black broken line in (a) is actual path, and the red one in (b) is localization result.

TABLE V
PARAMETERS FOR EXPERIMENT

Particle number	Update threshold	Update weights	Loss threshold
50	M_{th}=0.65	λ=0.7	3 m or 1 rad


(a) Wheelchair

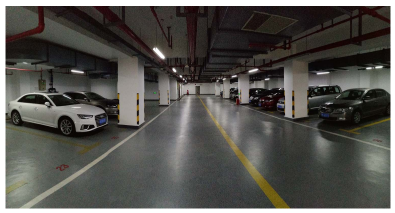

(b) Scene of indoor parking area

Fig. 7. Experimental platform and environment of indoor parking area.

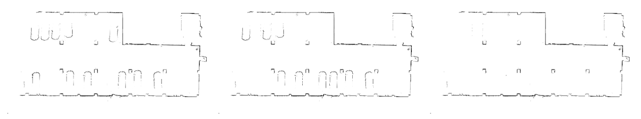
B. Experiment scene of parking area

The parking area is shown in Fig. 7, which is 51.7 m × 27.8 m, with 23 parking spaces.

In parking area experiment, the parking space instead of the single grid is modeled. The time series of parking space states are obtained before model building, and models for every parking space are built, then the entire predicted map is generated based on the predicted states. By this way, the modeling time can be significantly reduced. Since the shape and pose of the vehicle do not change much, setting them to fixed values in prediction is feasible in practical applications.

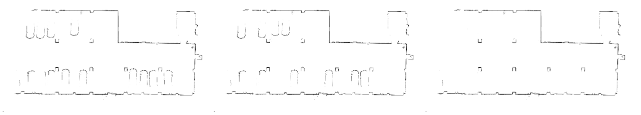
C. Map prediction of parking area experiment

The GMapping [18] is used for the original map and reference map build, which can generate maps with sufficient accuracy for our experiments. The grid map resolution is 0.1



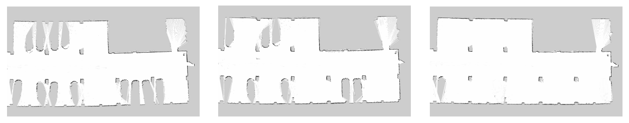
(a) 10:00 (b) 16:00 (c) 22:00

Fig. 8. FreMEn predicted maps.



(a) 10:00 (b) 16:00 (c) 22:00

Fig. 9. ArmMPU predicted maps.



(a) 10:00 (b) 16:00 (c) 22:00

Fig. 10. Reference maps.

TABLE VI
AVERAGE PREDICTION ACCURACY FOR 5 DAYS

	FreMEn	ArmMPU
Average prediction accurancy	75.51%	84.81%

m. 13 days data are collected at interval t_{mr}: previous 8 days data for model building and last 5 days for test.

Some reference and predicted maps in the 12th day are as Fig. 8-10. The average prediction accuracies for every day and whole 5 days are shown as Fig. 11 and Table VI.

The prediction accuracy of ArmMPU is higher than FreMEn. The actual parking environment has not only periodic changes, but also aperiodic and randomness changes. For example, on the beginning of weekly work, people may have longer parking hours because of overtime work. Our method can model aperiodic changes with regularity, and better deal with randomness through a certain error slip correction.

D. Localization performance of parking area experiment

The localization performance can reflect the final effectiveness of methods.

Four test points ABCD are chosen. In each test, push the wheelchair to move along the route for consecutive 3 circles. To highlight the role of parking space prediction, laser maximum scan range is set to 8 m. The wall is basically not scanned when robot moving on the path.

The matching degree m_1 in a certain test and corresponding maps are shown in Fig. 12. Between points A and B, the parking space prediction is basically correct, m_1 is high, and parking spaces nearby B are updated. After that, there is no car in BCD segment to help with localization, and some parking spaces between DA are wrongly predicted, so localization performance and m_1 gradually goes worse.

The average localization performance in 12 localization tests are shown as Fig. 13. The localization performance of ArmMPU are better than FreMEn. In case of not

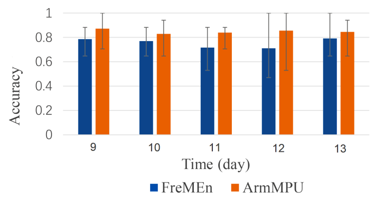
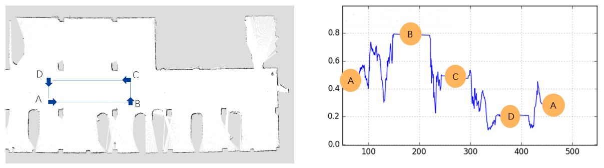
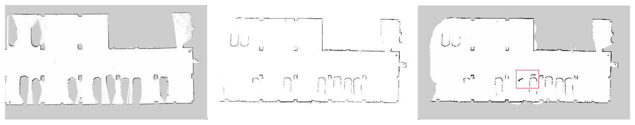


Fig. 11. Average prediction accuracy for every day.

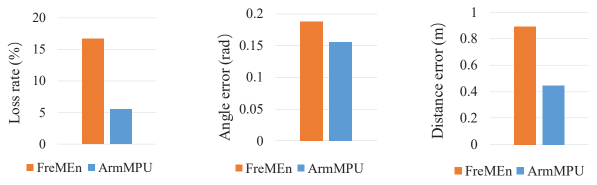


(a) Moving path (b) m_1



(c) Reference map (d) Predicted map (e) Updated map

Fig. 12. Map update performance.



(a) Loss rate (b) Angle error (c) Distance error

Fig. 13. Localization performance.

using the wall information, the accuracy of parking space prediction has a large impact on localization performance. In general, in the parking area experiment, ArmMPU has better performance.

E. Time and memory consumption

Time and memory consumption in parking area experiment of ArmMPU are evaluated, which are shown in Table VII and VIII. NumPy is used for model building with matrix operation vectorized. In memory consumption evaluation, two sizes of map, 51.7 m × 27.8 m (parking area) and 123.0 m × 81.4 m (several offices with hallways), are tested.

TABLE VII
TIME CONSUMPTION OF MAP MODELING AND PREDICTION

	Modeling in single grid		Modeling in parking space	
	Modeling	Prediction	Modeling	Prediction
Consumption of time (s)	676.69	6.42	29.7	0.78

TABLE VIII
MEMORY CONSUMPTION

	Parking area	Offices with hallways
Memory consumption (MB)	319	750

Time and memory consumption will increase with the map size extending, which will result in decrease of real-time performance. Due to the independently modeling of each grid, time consumption basically increases linearly, fully able to meet the requirements for model update once a day.

Meanwhile, we record the output frequency of updated maps and localization, which are 2 Hz and 10 Hz. In the parking environment where the changes are not severe, the real-time requirements can be met.

F. Experiments at laboratory

To enrich experimental scenes, experiments in a 18.1 m $\times$ 15.0 m indoor laboratory are supplemented. Changes of laboratory is more random due to human influence. Models for each grids are built, and the prediction and localization performance are shown in Table IX and X.

TABLE IX
AVERAGE PREDICTION ACCURACY

	FreMEn	ArmMPU
Average prediction accurancy	77.51%	78.16%

TABLE X
LOCALIZATION PERFORMANCE

Methods	Distance error/m	Angle error/rad	Loss rate/%
FreMEn	0.45	0.13	50
ArmMPU	0.42	0.10	17

The prediction accuracy of ArmMPU is a bit higher. That is because static object is in majority, which is easy to be correctly predicted. However, the dynamic object has great impact on localization, and the loss rate of ArmMPU are much lower than FreMEn. ArmMPU has better performance in the more randomized laboratory environment.

VII. CONCLUSION

In this paper, we propose a long-term localization method based on time series map prediction. The environmental map is modeled and predicted based on ARMA as the long-term information. The Bayesian filter combines prediction and observation with variable weight, providing accurate prior map for localization. Simulatiosn and experiments in different scenes show that our method can model and predict the periodic changes and aperiodic changes with regularity of the environment, and has certain adaptability to the random changes, and achieving better localization performance in the long-term changing environments. In the future work, we will consider making use of the correlation of grids, improving the real-time performance on the mobile robot to increase the frequency of map update, and extending the duration of the experiment.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (Grant U1813206 and 61773261).

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[2] M. C. Potter, B. Wyble, C. E. Hagmann, and E. S. McCourt, “Detecting meaning in rsvp at 13 ms per picture,” *Attention, Perception, & Psychophysics*, vol. 76, no. 2, pp. 270–279, 2014.

[3] B. Song, W. Chen, J. Wang, and H. Wang, “Long-term visual inertial slam based on time series map prediction,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Macao, Macau, China, 2019, pp. 5364–5369.

[4] Y. Wang, W. Chen, and J. Wang, “Map-based localization for mobile robots in high-occluded and dynamic environments,” *Industrial Robot: An International Journal*, vol. 41, no. 3, pp. 241–252, 2014.

[5] U. Hassler, “Autoregressive moving average processes (arma),” in *Stochastic Processes and Calculus*. Springer, 2016, pp. 45–75.

[6] J. P. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, “3d normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments,” *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1627–1644, 2013.

[7] G. D. Tipaldi, D. Meyer-Delius, and W. Burgard, “Lifelong localization in changing environments,” *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1662–1678, 2013.

[8] D. Sun, F. Geißer, and B. Nebel, “Towards effective localization in dynamic environments,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Daejeon, Korea, 2016, pp. 4517–4523.

[9] E. Olson, “M3rsm: Many-to-many multi-resolution scan matching,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Seattle, Washington, 2015, pp. 5815–5821.

[10] P. Biber, T. Duckett, *et al.*, “Dynamic maps for long-term operation of mobile service robots,” in *Robotics: science and systems*, 2005, pp. 17–24.

[11] F. Dayoub, G. Cielniak, and T. Duckett, “Long-term experiments with an adaptive spherical view representation for navigation in changing environments,” *Robotics and Autonomous Systems*, vol. 59, no. 5, pp. 285–295, 2011.

[12] T. Morris, F. Dayoub, P. Corke, G. Wyeth, and B. Upcroft, “Multiple map hypotheses for planning and navigating in non-stationary environments,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Chicago, Illinois, USA, 2014, pp. 2765–2770.

[13] T. Krajnik, J. P. Fentanes, G. Cielniak, C. Dondrup, and T. Duckett, “Spectral analysis for long-term robotic mapping,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Chicago, Illinois, USA, 2014, pp. 3706–3711.

[14] T. Krajník, J. P. Fentanes, M. Hanheide, and T. Duckett, “Persistent localization and life-long mapping in changing environments using the frequency map enhancement,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Daejeon, Korea, 2016, pp. 4558–4563.

[15] T. Krajník, J. P. Fentanes, J. M. Santos, and T. Duckett, “Fremen: Frequency map enhancement for long-term mobile robot autonomy in changing environments,” *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 964–977, 2017.

[16] M. Schreiber, S. Hoermann, and K. Dietmayer, “Long-term occupancy grid prediction using recurrent neural networks,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Montreal, QC, Canada, 2019, pp. 9299–9305.

[17] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, “Deep learning features at scale for visual place recognition,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Vancouver, Canada, 2017, pp. 3223–3230.

[18] G. Grisetti, C. Stachniss, and W. Burgard, “Improved techniques for grid mapping with rao-blackwellized particle filters,” *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[19] H. Akaike, “A new look at the statistical model identification,” in *Selected Papers of Hirotugu Akaike*. Springer, 1974, pp. 215–222.

[20] H. Akaike, “Fitting autoregressive models for prediction,” *Annals of the institute of Statistical Mathematics*, vol. 21, no. 1, pp. 243–247, 1969.

[21] Y. Dodge, *The concise encyclopedia of statistics*. Springer Science & Business Media, 2008.

[22] X. Hu, J. Wang, and W. Chen, “Long-term localization of mobile robots in dynamic changing environments,” in *Chinese Automation Congress (CAC)*, Xi’an, China, 2018, pp. 384–389.

[23] D. Meyer-Delius, J. Hess, G. Grisetti, and W. Burgard, “Temporary maps for robust localization in semi-static environments,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, 2010, pp. 5750–5755.

[24] L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.