Spatio-temporal model that aggregates information from sensors to estimate and predict states of obstacles for control of moving robots

Yuichi OHSITA[†], Shinya YASUDA[††], Taichi KUMAGAI[††], Hiroshi YOSHIDA[††], Dai KANETOMO[††], *Members*, *and* Masayuki MURATA[†], *Fellow*

SUMMARY Mobile robots need to be controlled to perform the required tasks without collisions with obstacles. However, there are moving obstacles such as other robots and people in the work area of the mobile robot. To control the robot efficiently without collisions with such obstacles, it is important to predict the risk of collisions at each location and each time. In this paper, we propose a framework that aggregates the information from sensors and predicts future states at each location to calculate the risks. This framework is based on the spatio-temporal model. In this framework, we introduce a manager that maintains the spatio-temporal model. We also deploy sensors in the environments. Then sensors send the monitored information to the manager. The manager maps the monitored information to the spatio-temporal model and updates the current and predicted future states. In this paper, we also demonstrate that the spatio-temporal model can predict the existence of obstacles at future time slots.

keywords: *Cloud, Robot, Control, Prediction, Conditional Random Field*

1. Introduction

Moving robots has become widely used in many areas such as factories and warehouses and are becoming used even in the areas with the interactions with people [1]. In such areas, robots are required to complete their tasks as immediately as possible while they must avoid accidents such as collisions to people or other robots. One approach to avoiding collisions is to stop the robot when a person or another robot close to the robot is detected [2]. But this approach degrades the performance of the robot.

By knowing the risk, we can avoid such risks without significant degrading the performance of the robots. For example, the robots can move fast if the risk is low. In addition, the robots can select the paths to avoid the risk.

The risk changes in time. For example, the area near people or robots has a high risk of collision. But people and robots move. As a result, the areas with high risks of collision changes. Therefore, the information of the risks should be updated to follow the changes of the environment.

Many robots have the sensors such as camera, ToF (Time-of-Flight) camera, and LiDAR (light detection and ranging) and can detect obstacles near them [3]. But the sensors of each robot are insufficient to calculate the risks because they cannot detect people coming from a blind spot. Therefore, aggregating the information from many sensors is required to know the risk.

In addition, not only the current states but also the future states are required to know the risk of collisions; even if a robot exists near another robot, the risk of collisions is low if they are going away from each other.

In this paper, we propose a framework that aggregates the information from sensors and predicts future states of the environment. This framework is based on the spatio-temporal model. In this framework, we introduce a manager that maintains the spatio-temporal model. We also deploy sensors in the environments. Then sensors send the monitored information to the manager. The manger maps the monitored information to the spatio-temporal model and updates the current and predicted future states.

2. Framework to Aggregate Information from Sensors based on Spatio-Temporal Model

Figure 1 shows the framework. In this framework, we introduce a server called manager. The manager has a spatio-temporal model of the target area. In this model, the target area is divided into multiple subareas, and the time is also divided into time slots. Then, the manager estimates or predicts the state for each subarea at each time slot. The spatio-temporal model includes the relation between the state and observation related to each subarea, and the relation between the states of near subareas. By using such relations, we can estimate and predict the probabilities of states of subareas whose information is not obtained yet from the monitored information.

In this framework, sensers are deployed in the area. The sensors extract features for subareas based on their observations and send the extracted features to the manager. The manager maps the information from the sensors to the spatio-temporal model and updates the spatio-temporal model based on the information.

[†]The authors are with Graduate School of Information Science and Technology, Osaka University.

[††]The authors are with Data Science Research Laboratories, NEC Corporation.

[a)] E-mail: y-ohsita@ist.osaka-u.ac.jp

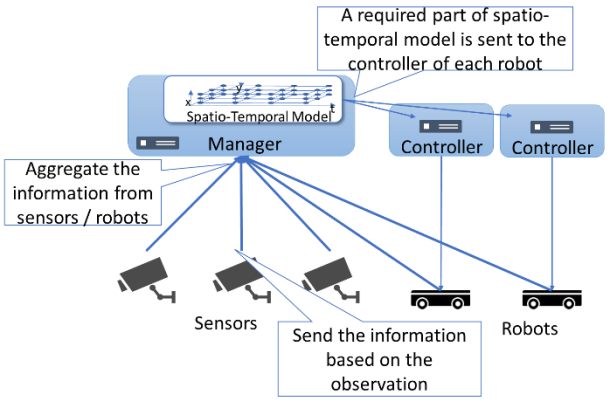


Fig. 1 Overview.

When controlling a robot, the controller of the robot requires only the information near the robot. In our framework, such required information can be extracted by extracting the subset of the spatio-temporal model. Then, the controller calculates the command based on the extracted information.

3. Spatio-Temporal Model to Estimate and Predict the States of Obstacles

3.1 Overview

In this section, we introduce a model that captures the correlation between subareas. The model should capture both of the spatial and temporal correlations. For example, if an obstacle that spans multiple spatial subareas exists, the states of all the subareas should be the same. Similarly, if an obstacle exists in a certain subarea, the obstacle may exist in the same subarea or adjacent subarea in the next time slot.

In this paper, we construct the spatio-temporal model including the above relation between the subareas based on the conditional random field [4]. In the conditional random field, the random variables are represented as vertices and the correlation between the variables are represented as edges between nodes. Figure 1 shows the spatio-temporal model based on the conditional random field.

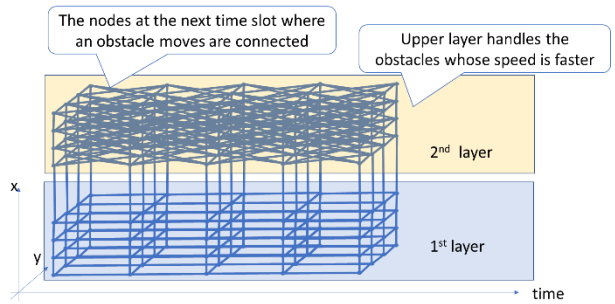


Fig. 2 Spatio-temporal model.

In this model, we construct multiple layers based on the speed of the moving obstacle because the related subareas at the next time slot, which are required to be connected, depend on the speed of the moving obstacle. In each layer, we define a random variable to each subarea at each time slot. Each random variable is shown as a vertex in the figure. Edges are added between the nodes that are adjacent at the same time slot. Edges between the nodes at the different layers for the same subarea at the same time slot are also added. Further, edges are also added between the nodes in the adjacent time slots where the obstacle whose speed is within the range handled by the layer may move. Hereafter, we denote the states of the subarea x, y at the time slot t at the layer l by $o_{l,t,x,y}$. We also denote the observation of x, y at the time slot t by $d_{t,x,y}$. O is a matrix including $o_{l,t,x,y}$ for all subareas, time slots and layers. D includes all observations.

In this spatio-temporal model, the probability distribution $P(O|D)$ of the variable O when the observation D is obtained is defined as follows.

$$p(O|D) = \frac{1}{Z(D)} \exp\big(E(O;D)\big)$$

where $Z(D)$ is a value defined so that $\sum_O p(O|D) = 1$, and

$$E(O;D) = \sum_{(t,x,y)\in N} f_{l,x,y}\left(o_{l,t,x,y}; d_{t,x,y}\right) + \sum_{((l1,t1,x1,y1),(l2,t1+t2,x2,y2))\in E} f_{(l1,x1,y1),(l2,t2,x2,y2)}\left(o_{l1,t1,x1,y1} o_{l2,t1+t2,x2,y2}; d_{t1,x1,y1}, d_{t1+t2,x2,y2}\right)$$

where N is a set of vertices, and $f_{l,x,y}\left(o_{l,t,x,y}; d_{t,x,y}\right)$ is a function of random variable $o_{l,t,x,y}$ defined by the observed value $d_{t,x,y}$, E is a set of edges, $f_{(l1,x1,y1),(l2,t2,x2,y2)}\left(o_{l1,t1,x1,y1} o_{l2,t1+t2,x2,y2}; d_{t1,x1,y1}, d_{t1+t2,x2,y2}\right)$ is a function of the random variables $o_{l1,t1,x1,y1}$ and $o_{l2,t2,x2,y2}$ which correspond to the both ends of the edge $(l1, t1, x1, y1), (l2, t1 + t2, x2, y2)$ and is defined by the observed values $d_{t1,x1,y1}$ and $d_{t1+t2,x2,y2}$.

In this model, the marginal probability $p\left(o_{l,t,x,y}|D\right)$ is obtained by approximate calculation using Loopy BP [5]. In this paper, the state of each subarea is defined as a label that is set based on the movement of the obstacle. Then, after obtaining the marginal probability $p\left(o_{l,t,x,y}|D\right)$, we can predict the probability of the existence of the obstacles in the subarea x, y at time slot t as $1 - \min_l p\left(o_{l,t,x,y} = NoObstacle|D\right)$ where $NoObstacle$ is the label corresponding to the states without any obstacles.

3.2 Estimation based on Spatio-Temporal Model

The manager obtains the information from sensors. Then the manager maps the obtained information to the model by applying the functions $f_{l,x,y}\left(o_{l,t,x,y}; d_{t,x,y}\right)$ and $f_{(l1,x1,y1),(l2,t2,x2,y2)}\left(o_{l1,t1,x1,y1} o_{l2,t1+t2,x2,y2}; d_{t1,x1,y1}, d_{t1+t2,x2,y2}\right)$ based on the obtained information. After mapping the obtained information, the manager updates the model by removing the nodes corresponding to the time slot more than the predefined time ago and adding the nodes for the time slots within the prediction target.

Then, the manager calculates marginal probability $p\left(o_{l,t,x,y}|D\right)$ by using the Loopy BP algorithm [5]. Finally, the state of each subarea is obtained by $p\left(o_{l,t,x,y}|D\right)$.

3.3 Training of Spatio-Temporal Model

In this paper, we set the functions $f_{l,x,y}(o_{l,t,x,y}; d_{t,x,y})$ and $f_{(l1,x1,y1),(l2,t2,x2,y2)}(o_{l1,t1,x1,y1}o_{l2,t1+t2,x2,y2}; d_{t1,x1,y1}, d_{t1+t2,x2,y2})$ by using the previous observations as a training data. To set the functions, we first set the labels to each subarea by grouping obstacles having similar trajectories and setting the label to each group. Then, we set the functions $f_{l,x,y}(o_{l,t,x,y}; d_{t,x,y})$ and $f_{(l1,x1,y1),(l2,t2,x2,y2)}(o_{l1,t1,x1,y1}o_{l2,t1+t2,x2,y2}; d_{t1,x1,y1}, d_{t1+t2,x2,y2})$ based on the label. The tendency of obstacle movement depends on the location as well as the type of obstacle. Therefore, in this paper, we define these functions based on the location.

$f_{l,x,y}(o;\ d)$ is

$$f_{l,x,y}(o;\ d) = \frac{Number[D_{t,x,y} = d, O_{l,t,x,y} = o]}{Number[D_{t,x,y} = d]}$$

where $D_{t,x,y}$ is the observed value of subarea x, y at time slot t in the training data, and $O_{l,t,x,y}$ is the label of subarea x, y at time slot t at layer l, and $Number[\ ]$ indicates the number of elements of training data that satisfy the conditions in $[\ ]$. If the observed value cannot be obtained, it is determined as follows.

$$f_{l,x,y}(o;\ None) = \begin{cases} \alpha & o \text{ indicates no obstacles} \\ (1-\alpha)\dfrac{Number[O_{l,t,x,y} = o]}{Number[D_{t,x,y}]} & \text{otherwise} \end{cases}$$

Similarly,

$$f_{(l1,x1,y1),(l2,t2,x2,y2)}(o_1o_2; d_1, d_2) = \frac{[D_{l1,t1,x1,y1} = d_1, D_{l2,t1+t2,x2,y2} = d_2, O_{t1,x1,y1} = o_1, O_{t1+t2,x2,y2} = o_2]}{[D_{t1,x1,y1} = d_1, D_{t1+t2,x2,y2} = d_2]}$$

If the observed values cannot be obtained,

$$f_{(x1,y1),(t2,x2,y2)}(o_1o_2; None, None) = \begin{cases} \beta & o1 \text{ and } o2 \text{ indicates no obstacles} \\ \gamma & \text{only } o1 \text{ indicates no obstacles} \\ \theta & \text{only } o2 \text{ indicates no obstacles} \\ \dfrac{Number[D_{t1,x1,y1} = d_1, D_{t1+t2,x2,y2} = d_2]}{Number[D_{t1,x1,y1}, D_{t1+t2,x2,y2}]} & \text{otherwise} \end{cases}$$

3.4 Reduction of computational time

The target area may include many subareas and prediction for such large area takes long time. But the number of the subareas including obstacles is small. Therefore, we can reduce the calculation time by focusing only on the subareas that may include obstacles. In this paper, we exclude the subareas whose observation indicates that they have no obstacles. In addition, we add only the nodes that may have obstacles by recursively adding the nodes that is connected to the nodes that have already been added after adding the nodes whose corresponding observations indicate they have obstacles.

4. Experiment

In this paper, we demonstrate that our framework predicts the existence of obstacles by simulation. In this simulation, we generate a moving obstacle and predict the existence of the obstacle. We implement the spatio-temporal model by using Direct Graphical Models C++ Library [6].

4.1. Environment

4.1.1. Moving obstacle

In this experiment, we use the area shown in Figure 3. The size of this area is 100×100 and divided into subareas whose size is 1×1. This area has two roads that intersect in one point. The widths of the roads are 10.

We generate one moving obstacle that enters the area from randomly selected point. Then, the obstacle randomly changes the direction at the intersection. The speed of the moving obstacle is set to the values randomly selected from 1 to 3 subareas per time slot. After entering the area, the moving obstacle does not change the speed.

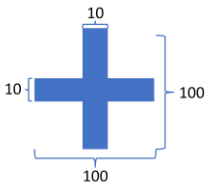


Fig. 3 Target area.

4.1.2. Definition of observations

In this experiment, we assume that we have sensors that can detect existence of the obstacles in each subarea and information of the existence of obstacles is sent to the manager. By using the information on the detected obstacles, the manager extracts the following information

$$d_{t,x,y} = [d^{EXIST}{}_{t,x,y}, d^{MOVE}{}_{t,x,y}, d^{LOCATION}{}_{t,x,y}]$$

$d^{EXIST}{}_{t,x,y}$ is a flag indicating the existence of the obstacles. $d^{EXIST}{}_{t,x,y} = 1$ if obstacles exist in the subarea x, y at time slot t, and $d^{EXIST}{}_{t,x,y} = 0$ otherwise. $d^{MOVE}{}_{t,x,y}$ is the information of the direction of the moving obstacles in the subarea x, y at time slot t. $d^{MOVE}{}_{t,x,y}$ is constructed of 8 bits. Each bit corresponds to the direction and is set to 1 if the obstacle exists in the subarea in the corresponding direction at the previous time slot. $d^{LOCATION}{}_{t,x,y}$ is the information of the location within the obstacles. $d^{LOCATION}{}_{t,x,y}$ is also constructed of 8 bits. Each bit is set to 1 if there does not exist any obstacles in the near subarea in the corresponding direction. By extracting the above information, we can extract the information to identify the direction of the moving objects.

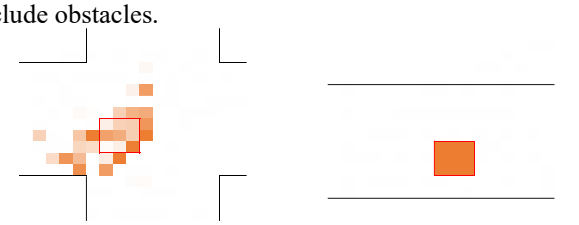
4.1.3. Training

In this experiment, we generate 20 moving obstacles and train the spatio-temporal model with 3 layers each of which handles the obstacles whose speeds are 1, 2, and 3

subareas per time slot. When training the model, we set the label of the obstacles based on the speed and direction of the obstacles; the obstacles going to the same direction with the same speed are given the same label. When the obstacle changes the direction, we set the different label before and after changing the direction.
In this experiment, we construct the model including past 10 time slots to predict the states of the future 4 time slots. In this paper, we set α to 0.9, β to 1, γ to 0.9 and θ to 0.0001.
After training the model, we generate the moving obstacles included in the trained model and demonstrate the prediction based on the trained model.

4.2. Results

Figure 4 shows examples of the prediction. Figure 4 (a) shows the prediction in the subareas near the intersection, and Figure 4 (b) shows the prediction in the subareas without the intersection. Both figures indicate the prediction results as of 4 time slots ago. In this figure, the subarea whose predicted probability of existence of obstacles is large is shown as the dark orange area. We also plot the subareas that actually include obstacles as red square.
As shown in this figure, when the obstacle goes straight, our model can accurately predict the future position of the obstacle. When the obstacle changes the direction, our model cannot accurately predict the obstacle's position but can predict the subareas that may include the obstacles. As a results, the actual position of the obstacle is within the subareas that are predicted as the subareas that may include obstacles.



(a)Near the intersection (b) On the straight road

Fig. 4 Example of prediction.

We also evaluate the accuracy of the prediction. To evaluate the accuracy, we set the threshold to the predicted probability and regard the subareas exceeding the threshold as the subareas with obstacles. We count the subareas that include the obstacles but are not predicted as the subareas with obstacles (False Negatives; FNs). We also count the subareas that are predicted as the subareas with obstacles but include no obstacles (False Positives; FPs). By counting above subareas, we define the following metrics.

$$False\ negative\ rate = \frac{\#\ of\ FNs}{\#\ of\ subareas\ with\ obstacles}$$

$$False\ discovery\ rate = \frac{\#\ of\ FPs}{\#\ of\ subareas\ predicted\ as\ subareas\ with\ obstacles}$$

We change the threshold to the predicted probability of existence of the obstacles from 0 to 1 with an interval of 0.01 and obtain the false negative rates and false discovery rates. Figure 5 shows the results. This figure demonstrates that our model accurately predicts the future position of the obstacles, while the accuracy slightly decreases as the prediction target becomes future.

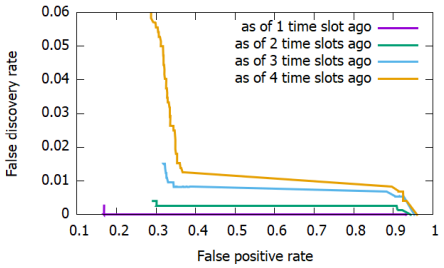


Fig. 5 Accuracy of the prediction.

5. Conclusion and Future Work

In this paper, we proposed a framework that aggregates the information from sensors and predicts future states of the environment based on the spatio-temporal model. In this framework, we introduce a manager that maintains the spatio-temporal model. We also deploy sensors in the environments. Then sensors send the monitored information to the manager. The manager maps the monitored information to the spatio-temporal model and updates the current and predicted future states.
We evaluated the spatio-temporal model included in our framework by simulation. The results demonstrate that the spatio-temporal model can predict the existence of the obstacles.
Our future work includes the evaluation of the control of moving robots based on the risk calculated by our framework.

Acknowledgments

This work was partly supported by NICT.

References

[1] Rey, Rafael, et al. "Human-robot co-working system for warehouse automation." 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). IEEE, 2019.
[2] Lai, Y. L., et al. "A novel automated guided vehicle for guidance applications." Journal of Physics: Conference Series. Vol. 2020. No. 1. IOP Publishing, 2021.
[3] Buck, Sebastian, et al. "Generic 3D obstacle detection for AGVs using time-of-flight cameras." 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2016.
[4] Y. Bengong and F. Zhaodi, "A comprehensive review of conditional random fields: variants, hybrids and applications," Artificial Intelligence Review, vol. 53, pp. 4289– 4333, Aug. 2020.
[5] Murphy, Kevin, Yair Weiss, and Michael I. Jordan. "Loopy belief propagation for approximate inference: An empirical study." arXiv preprint arXiv:1301.6725 (2013).
[6] S. Kosov, "Direct graphical models C++ library," 2013.