Master's Thesis

Title

Real-time obstacle prediction utilizing edge-cloud cooperation in mobile robot control

Supervisor

Professor Masayuki Murata

Author

Kazuki Kimura

February 2th, 2023

Department of Information Networking

Graduate School of Information Science and Technology

Osaka University

Master's Thesis

Real-time obstacle prediction utilizing edge-cloud cooperation in mobile robot control

Kazuki Kimura

Abstract

Mobile robots have become widely used in many cases such as automated guided vehicles in warehouses. Robots are also becoming used in the area with other robots and people. In such cases, the robots are required to complete their tasks fast but to avoid collisions with other robots and people. The obstacle prediction is important to achieve such efficient control without collisions. Robots can avoid collisions by considering the risk of collision based on the prediction; robots can set their routes that avoids the area with high risk of collisions and can change their speed based on the risk. 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. We can also use the sensors fixed to the ceiling, wall and so on. By aggregating the information from the sensors, we can detect and predict the obstacles. To control mobile robots, the detection and prediction should be updated in real time. However, it takes a time to aggregate the information from sensors and predict future state of obstacles as the target area becomes large and the number of obstacles becomes large.

In this thesis, we propose a method to predict obstacles in real time by cooperation between cloud and edge computers. In this method, we divide the target area into multiple subareas and deploy an edge computer for each subarea. Each edge computer collects and aggregates the information from the sensors in the corresponding area, and detects and predicts obstacles in the corresponding area. In addition, we also deploy a cloud computing environment or high-performance computer. The cloud aggregates the information from the edge computers and detects and predicts the all obstacles in the target area. In addition, the cloud also shares its results of prediction with the edge computers. By using the information from the clouds, each edge can predict the obstacles from the outside of its corresponding subarea. In this thesis, we also propose a method to reduce the amount of information exchanged between the cloud and edge computers. In this method, each edge send only the information of the points whose states were not expected from the previously sent information. By sending only such information, the cloud can detect and predict obstacles accurately enough by exchanging a small amount of information. We evaluate our method by simulation. The results show that our method achieves the similar prediction accuracy to the method collecting all information, while our method updates the prediction in less than 100 ms at edges.

Keywords

Digital twin
Markov Random Field
Conditional Random Field
Spatio-temporal model
Edge-cloud cooperation
Loopy Belief Propagation

Contents

List of Figures

List of Tables

1 Introduction

Mobile robots have become widely used in many cases such as automated guided vehicles in warehouses [1]. Robots are also becoming used in the area with other robots and people [2]. In such cases, the robots are required to complete their tasks fast but to avoid collisions with other robots and people. The obstacle prediction is important to achieve such efficient control without collisions. Robots can avoid collisions by considering the risk of collision based on the prediction; robots can set their routes that avoids the area with high risk of collisions and can change their speed based on the risk [3]. 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. We can also use the sensors fixed to the ceiling, wall and so on. By aggregating the information from the sensors, we can detect and predict the obstacles. To control mobile robots, the detection and prediction should be updated in real time. However, it takes a time to aggregate the information from sensors and predict future state of obstacles as the target area becomes large and the number of obstacles becomes large.

In this thesis, we propose a method to predict obstacles in real time by cooperation between cloud and edge computers. In this method, we divide the target area into multiple subareas and deploy an edge computer for each subarea. Each edge computer collects and aggregates the information from the sensors in the corresponding area, and detects and predicts obstacles in the corresponding area. In addition, we also deploy a cloud computing environment or high-performance computer. The cloud aggregates the information from the edge computers and detects and predicts the all obstacles in the target area. In addition, the cloud also shares its results of prediction with the edge computers. By using the information from the clouds, each edge computer can predict the obstacles from the outside of its corresponding subarea. In this thesis, we also propose a method to reduce the amount of information exchanged between the cloud and edge computers. In this method, each edge send only the information of the points whose states were not expected from the previously sent information. By sending only such information, the cloud can detect and predict obstacles accurately enough by exchanging a small amount of information.

The rest of this thesis is organized as follows. Section 2 explains the related work.

In Section 3, we introduce the obstacle prediction based on spatio-temporal model and apply the edge-cloud cooperation to the obstacle prediction. In Section 4, we evaluate our method by simulation. Section 5 conclude this thesis.

2 Related work

2.1 Obstacle Prediction for controlling moving robots

Obstacle prediction is important for controlling mobile robots and autonomous vehicles. Therefore, many methods have been proposed.

Wang et al. proposed a method to predict the occupied areas in the case of the occlusion [4]. This method uses contextual information of environments and learns from prior knowledge to predict obstacle distributions in occluded space. Rozsa et al. proposed a method to identify the obstacles from the information obtained by a LIDAR Sensor [5]. This method categorize the obstacles from partial point clouds without shape modeling. They demonstrated the proposed method works on on-board computers, assuming that their method is implemented on an AGV. However, these methods do not consider the moving obstacles.

The methods considering moving obstacles have also been proposed. Ferguson et al. proposed a method to avoid dynamic obstacles for autonomous vehicles [6]. This method assumes that vehicles traveling on roads typically follow common rules of the road. This method predicts future position of the other vehicles based on this assumption, and avoids collisions based on the prediction. Foka [7] proposed a method to avoid collisions with human. This method assumes that there are some hot spots where people would have interest in visiting them. The long-term prediction of human movements are performed based on the hot spot. Then, robots are navigated to avoid collisions. Chiang et al. propose a method to navigates robots for dynamic environment with static and moving obstacles [8]. This method recognizes the velocity and moving direction of each obstacle and generates the corresponding Trajectory Prediction Table to predict the future trajectory. The possibility of the collision is calculated based on the table. If the collision is possible, this method searches a new path to avoid the obstacle. Liu et al. proposed a method for UAV to select the path considering the prediction of the obstacles [9]. In this paper, the Kalman filter is used to estimate the dynamic obstacle trajectory. Then based on the estimated trajectory, the UAV are controlled so as to avoid the obstacles. Ohsita et al. proposed a method to predict the areas with moving obstacles [10]. This method construct the spatio-temporal model of the moving obstacles considering the patterns of movements of obstacles. Then, each time new observations are obtained, this method updates the prediction by mapping the new observations to the spatio-temporal model.

To control mobile robots, the detection and prediction should be updated in real time. However, it takes a time to aggregate the information from sensors and predict future state of obstacles as the target area becomes large and the number of obstacles becomes large. In this thesis, we propose a method to predict obstacles in real time by cooperation between cloud and edge computers.

2.2 Edge cloud cooperation

Edge cloud cooperation is one of the promising approaches for real-time applications.

Liang et al. proposed a method for real-time object detection based on the edge-cloud cooperation [11]. In this method, edges carry out model reasoning and data uploading, while the cloud performs the timed training and weight updating of new data independently. Another example of the application of the edge-cloud cooperation is the control of the smart factory [12]. In this method, the edge has the ability of self-sensing, self-communication, self-analysis, and self-optimization, while a cloud platform is also set to make central decisions. As a result, the manufacturing system can make decisions following a cloud-edge cooperation mechanism.

In this thesis, we apply edge cloud cooperation to the prediction of the moving obstacles based on the censored information.

3 Real-time obstacle prediction utilizing edge-cloud cooperation in mobile robot control

3.1 Overview

In this thesis, we propose a method to detect and predict obstacles in a warehouse for controlling mobile robots. We assume that many other robots and people exist in a warehouse. To control the mobile robot efficiently without collisions, it is required to predict if an obstacle exists at a certain time and point.

We also assume that multiple sensors are deployed in the target area and we can detect the obstacles by using the information from the sensors.

In this thesis, we divide the target area into multiple subareas and deploy an edge computer for each subarea. Each edge computer collects and aggregates the information from the sensors in the corresponding area, and detects and predicts obstacles in the corresponding area. In addition, we also deploy a cloud computing environment or high-performance computer. The cloud aggregates the information from the edge computers and detects and predicts the all obstacles in the target area. In addition, the cloud also shares its results of prediction with the edge computers. By using the information from the clouds, each edge can predict the obstacles from the outside of its corresponding subarea.

In this thesis, we also propose a method to reduce the amount of information exchanged between the cloud and edge computers. In this method, each edge send only the information of the points whose states were not expected from the previously sent information. By sending only such information, the cloud can detect and predict obstacles accurately enough by exchanging a small amount of information.

3.2 Spatio-temporal model for obstacle prediction

3.2.1 Spatio-temporal model

In this section, we introduce a model that captures the correlation between points in the target area. The model should capture both of the spatial and temporal correlations. For example, 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 thesis, we use the spatio-temporal model including the above relation between the points based on the model proposed by Ohsita et al. [10]. This model is based on the Markov random field [13]. In the Markov random field, the random variables are represented as vertices and the correlation between the variables are represented as edges between nodes.

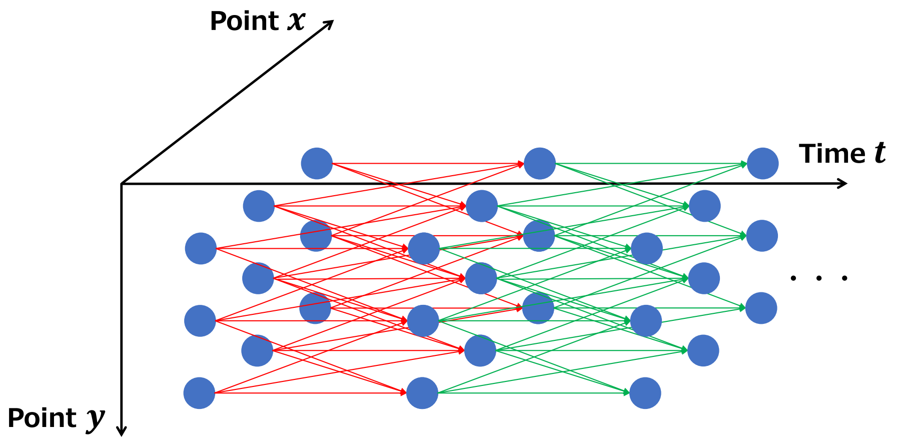


Figure 1: Spatio-temporal model

Figure 1 shows the spatio-temporal model for obstacle prediction. In this model, we define a random variable to each time and point. Each random variable is represented as a vertex. Edges are added between the nodes in the adjacent time slots where the obstacle may move. Hereafter, we denote the states of the point x, y at the time slot t by $o_(t, x, y)$. O is a matrix including $o_{t,x,y}$ for all points and times.

In addition to the vertices corresponding to each point and time, we also add the vertices corresponding to the observation. Edges between the vertices of the observation and the vertices of the corresponding times and points are also added. We denote the observation of x, y at the time slot t by $d_{t,x,y}$. D includes all observations.

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

$$p(O|D) = \frac{1}{Z(D)} F(O; D) \tag{1}$$

where

$$
\begin{aligned}
F(O;D) &= \prod_{d_{t,x,y}\in D} f_{t,x,y}\left(o^{\mathrm{ob}}_{t,x,y}; d_{t,x,y}\right) \\
&\times \prod_{o^{\mathrm{ob}}_{t1,x1,y1}, o_{t2,x2,y2})\in E} f^{\mathrm{ob}}_{t1,x1,y1,t2,x2,y2}\left(o^{\mathrm{ob}}_{t1,x1,y1} o_{t2,x2,y2}\right) \\
&\times \prod_{o_{t1,x1,y1}, o_{t2,x2,y2}\in E} f_{(t1,x1,y1),(t2,x2,y2)}\left(o_{t1,x1,y1} o_{t2,x2,y2}\right)
\end{aligned}
\tag{2}
$$

where N is the set of the random variables and E is the set of the edges. The function $f_{t,x,y}\left(o^{\mathrm{ob}}_{t,x,y}; d_{t,x,y}\right)$ maps the observation to the labels. The function $f^{\mathrm{ob}}_{(t1,x1,y1),(t2,x2,y2)}\left(o^{\mathrm{ob}}_{t1,x1,y1} o_{t2,x2,y2}\right)$ models the relation between the observation at $x1, y1$ at time $t1$ and the state at $x2, y2$ at time $t2$. The function $f_{(t1,x1,y1),(t2,x2,y2)}\left(o_{t1,x1,y1} o_{t2,x2,y2}\right)$ models the relation between the variables at $t1, x1, y1$ and $t2, x2, y2$.

In this model, the marginal probability $p\left(o_{t,x,y}|D\right)$ is obtained by approximate calculation using Loopy BP [14]. 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 $p\left(o_{t,x,y}|D\right)$, we can predict the probability of the existence of the obstacles in the point x, y at time slot t as $1 - p\left(o_{t,x,y} = NoObstacle|D\right)$ where NoObstacle is the label corresponding to the states without any obstacles.

3.2.2 Training

In the spatio-temporal model, the functions $f_{t,x,y}\left(o^{\mathrm{ob}}_{t,x,y}; d_{t,x,y}\right)$, and $f_{(t1,x1,y1),(t2,x2,y2)}\left(o_{t1,x1,y1} o_{t2,x2,y2}\right)$ are set 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. The tendency of obstacle movement depends on the location as well as the type of obstacle. Therefore, in the spatio-temporal model, we define these functions based on the location.

$f_{t,x,y}\left(o^{\mathrm{ob}}_{t,x,y}; d_{t,x,y}\right)$ is set by

$$
f_{t,x,y}\left(o^{\mathrm{ob}}_{t,x,y} = o; d_{t,x,y} = d\right) = \frac{\mathrm{Number}[o_{t,x,y} = o; d_{t,x,y} = d]}{\mathrm{Number}[d_{t,x,y} = d]} \tag{3}
$$

where Number[] indicates the number of elements of training data that satisfy the conditions in [].

Similarly, we set $f_{(t1,x1,y1),(t2,x2,y2)}\left(o_{t1,x1,y1}o_{t2,x2,y2}\right)$ by

$$f_{(t1,x1,y1),(t2,x2,y2)}\left(o_{t1,x1,y1}=o1,o_{t2,x2,y2}=o2\right)=\frac{\text{Number}[o_{t1,x1,y1}=o1,o_{t2,x2,y2}=o2]}{\text{Number}[o_{t1,x1,y1}=o1]} \tag{4}$$

$f^{\text{ob}}_{(t1,x1,y1),(t2,x2,y2)}\left(o^{\text{ob}}_{t1,x1,y1}o_{t2,x2,y2}\right)$ maps the observation to the state. In this thesis, we maps the observation to only the state of the same time and point. We use $f^{\text{ob}}_{(t1,x1,y1),(t2,x2,y2)}\left(o^{\text{ob}}_{t1,x1,y1}o_{t2,x2,y2}\right)$ defined by

$$f^{\text{ob}}_{(t1,x1,y1),(t1,x1,y1)}\left(o^{\text{ob}}_{t1,x1,y1}=o1o_{t1,x1,y1}=02\right)=\begin{cases}1 & (o1=o2)\\ \epsilon & (\text{otherwise})\end{cases} \tag{5}$$

3.2.3 Reduction of calculation time

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

3.3 Edge-cloud cooperation

The prediction in near future time slot is required to avoid collisions. By frequently updating the prediction based on new observations, we can predict obstacles in near future accurately. That is, real-time update is important. On the other hand, the long-term prediction is also useful. By considering the long-term prediction, robots can select their paths to complete their task fast without passing the points with high risk of collision. It is difficult to predict the obstacles in distant future. But, the new observations do not have a large impact on the results of the long-term prediction, compared with the prediction in near future.

Therefore, we propose a method to perform both of real-time prediction of near future and long-term prediction. In this method, we deploy multiple edge computers that update prediction in real-time. We also deploy a cloud or high-performance computer that perform long-term prediction. In this method, we divide the target area into multiple subareas. A edge computer is deployed for each subarea. Each edge computer collects and aggregates the information from the sensors in the corresponding area, and detects and predicts obstacles in the corresponding area. Focusing on the small subareas, edge computer can handle the information from the sensors and update the prediction in real time. The cloud aggregates the information from the edge computers and detects and predicts the all obstacles in the target area. The cloud has a large amount of computational resources to handle such a large area. In addition, the long-term prediction does not have to be updated frequently. Therefore, the interval to update the prediction by the cloud can be set to longer value. The cloud also supports the prediction of edge computers by sharing its results of prediction. By using the information from the clouds, each edge can predict the obstacles from the outside of its corresponding subarea.

We also propose a method to reduce the amount of information exchanged between the cloud and edge computers. In this method, each edge send only the information of the points whose states were not expected from the previously sent information. By sending only such information, the cloud can detect and predict obstacles accurately enough by exchanging a small amount of information.

Figure 2 shows the flow of the edge cloud cooperation. At time t, an edge computer e has the spatio-temporal model from time slot $t - N_e$ to time slot $t + H_e$ within the subarea A_e. The edge computer e collect the information from the sensor within the subarea A_e and update its spatio-temporal model at each time slot. On the other hand, the cloud c has the spatio-temporal model from time slot $t - N_c$ to time slot $t + H_c$ for all areas. The spatio-temporal model is updated by collecting the information from edges once in T time slots. After updating the model, the cloud sends the results of the prediction to $t + H_e + T$ time slot to edge computer e.

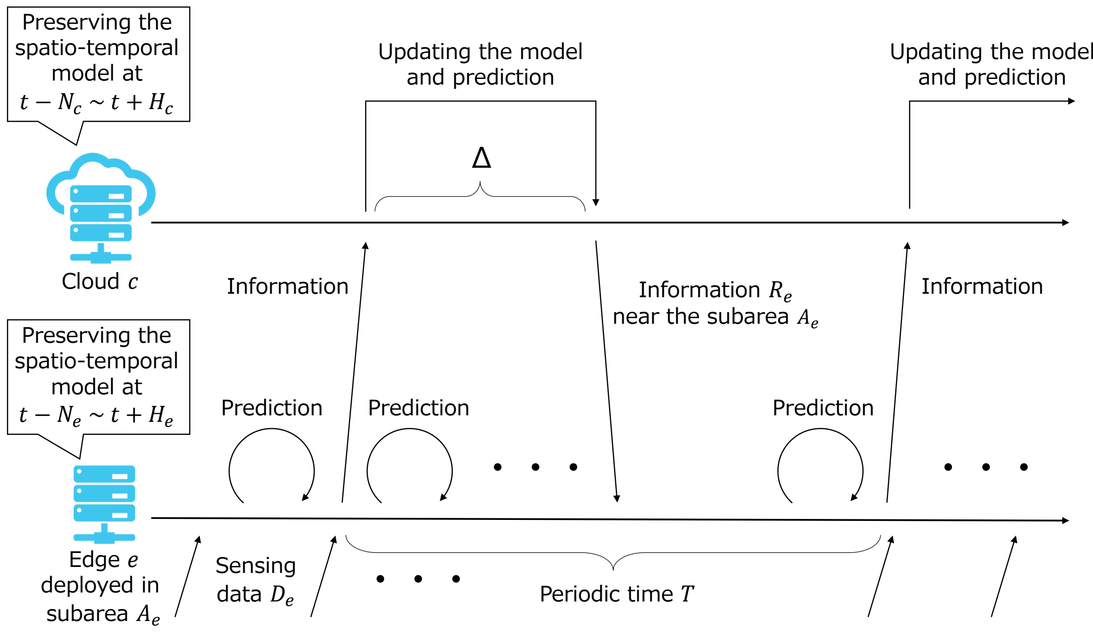

Preserving the spatio-temporal model at $t - N_c \sim t + H_c$
Cloud c
Updating the model and prediction
Δ
Updating the model and prediction
Information
Information R_e near the subarea A_e
Information
Preserving the spatio-temporal model at $t - N_e \sim t + H_e$
Prediction
Prediction
Prediction
Edge e deployed in subarea A_e
Sensing data D_e
Periodic time T

Figure 2: Flow of edge cloud cooperation

3.3.1 Process by edge

Updating the model and prediction Each edge computer e collects the information of the corresponding subarea A_e from the sensors. We denote the collected information by D_e. The edge computer e also has the information sent by the cloud R_e. R_e includes the predicted results near the corresponding subarea.

The edge computer updates the spatio-temporal model by using D_e and R_e. We describe the steps to update the spatio-temporal model as follows.

1. Add the vertices corresponding to observation D_e and the vertices corresponding to the states of the point that are included in the observation D_e.
2. Add the vertices corresponding to the information received from the cloud R_e.
3. Add the vertices corresponding to the point where the obstacle may exist.
4. Remove unnecessary vertices including the vertices before $t - N_e$ time slot and the vertices corresponding to the point where the obstacle cannot exist.
5. Obtain the prediction results of each time and point by Loopy BP algorithm.

Sending the information to the cloud The edge computer e send the information of the subarea A_e to the cloud so that the cloud can aggregate information of all subareas. The information sent to the cloud is how to update the spatio-temporal model. That is, the edge computer send the information on the added/update vertices and the deleted vertices. However, as the number of edge computers becomes large, the information sent to the cloud becomes large and nonnegligible.

Therefore, in our method, the edge computer send only the necessary information; each edge send only the information of the points whose states were not expected from the previously sent information. Each edge selects the information to be sent based on the prediction results. The information on the vertex $o_{t,x,y}$ is sent if the following condition is satisfied.

$$\max_{l} \left| o_{t,x,y}^{(e,t^{\text{current}})}(l) - o_{t,x,y}^{(e,t_{t.x,y}^{\text{send}})}(l) \right| > \lambda$$

where $o_{t,x,y}^{(e,t^{\text{pred}})}(l)$ indicates the prediction results by the edge computer e at t^{pred} corresponding to the point x, y, time slot t and label l. $t_{t,x,y}^{\text{send}}$ is the time slot when the previous information to the cloud was sent. $t^{\text{current}})$ is the current time slot. If the above condition is satisfied, the edge computer also send the vertex corresponding to the observation at t, x, y. In addition, if the vertex $o_{t,x,y}$ does not included in the current model but the following condition is satisfied, the edge computer notify that the vertex $o_{t,x,y}$ should be deleted.

$$\max_{l \neq NoObstacle} \left| o_{t,x,y}^{(e,t_{t.x,y}^{\text{send}})}(l) \right| > \lambda$$

By sending only the information satisfying the above condition, the edge computer sends only the information that cannot be predicted by the cloud. As a result, the cloud can detect and predict the obstacles accurately without collecting all information.

3.3.2 Process by cloud

Updating the model and prediction The cloud periodically collects the information from edge computers. Then by aggregating the collected information, the cloud updates the model and performs long-term prediction.

We describe the steps to update the spatio-temporal model at time slot t as follows.

1. Add the vertices included in the information obtained from the edge computers.

2. Add the vertices corresponding to the point where the obstacle may exist.

3. Remove unnecessary vertices including the vertices before $t - N_e$ time slot and the vertices corresponding to the point where the obstacle cannot exist.

4. Obtain the prediction results of each time and point by Loopy BP algorithm.

Sending the information to the edge computers After updating the prediction, the cloud sends the information to the edges. The edge computers requires the information that is related to the prediction within the corresponding subarea but cannot be detected from the sensors within the subarea. Therefore, the clouds sends the information of prediction near the subarea A_e from time slot t^{current} to time slot $t^{\text{current}} + H_e + T$. By sending the information, the edge computer can predict the obstacles from the outside of its corresponding subarea.

4 Evaluation

In this thesis, we simulate our method to demonstrate that our method based on edge cloud prediction works properly. 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 [15] and perform experiment using the computer show in Table 1.

4.1 Settings

Table 1: Execution environment

CPU	Intel(R) Xeon(R) W-2235 @ 3.80 GHz
OS	Ubuntu 18.04.6 LTS
RAM	31 GB

4.1.1 Area and subareas

In this experiment, we use the area shown in Figure 3. The size of this area is 100×100. This area has two roads that intersect in one point. The widths of the roads are 10. We divide this area into five subareas and deploy an edge computer for each area.

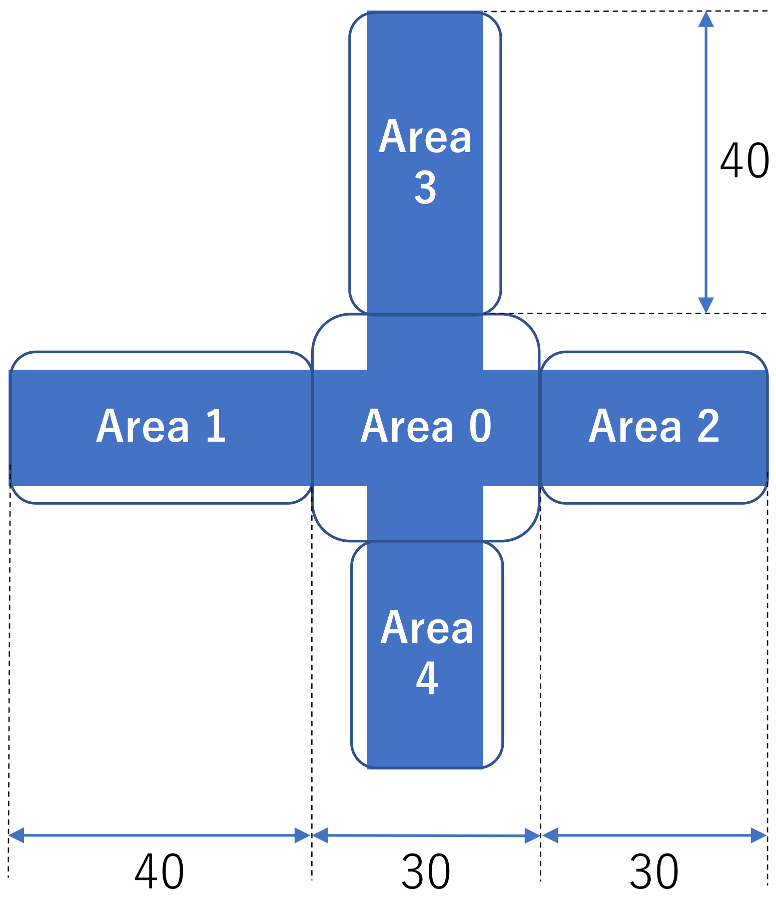


Figure 3: Target area

4.1.2 Obstacles

We generate one moving obstacle that enters the area from randomly selected point. At the intersection, the obstacle slows down. After stopping for 1 time slot, the obstacle randomly changes the direction and speeds up. We set the maximum speed of the moving obstacle is set to 3 per time slot. We set the size of the obstacle to 3×3.

4.1.3 Definition of observations

In this experiment, we assume that we have sensors that can detect existence of the obstacles in each point and information of the existence of obstacles is collected by the edge computer. By using the information on the detected obstacles, the edge computer extracts the information on the difference of the locations of the obstacles as the observations.

4.1.4 Training

In this experiment, we generate 20 moving obstacles and train the spatio-temporal model. 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.

4.2 Results

Figure 4 shows examples of the prediction by the edge computer for area 0. This figure indicate the prediction results as of the time slot before the obstacle enters the area 0. In this figure, the points whose predicted probability of existence of obstacles is large is shown as the pink area. As shown in this figure, several points were predicted as the points with obstacles though the edge did not observe any obstacles yet before the prediction. This is because the edge computer can use the information predicted by the cloud. As a result, if the cloud predicts obstacles entering the subarea, the edge computer corresponding to the subarea can also predict the obstacles.

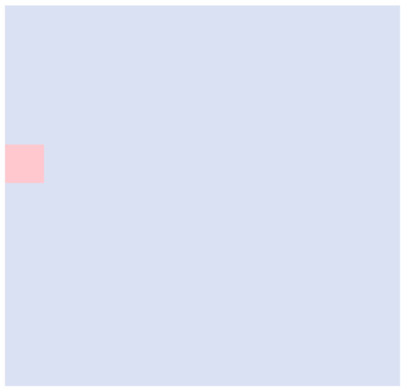


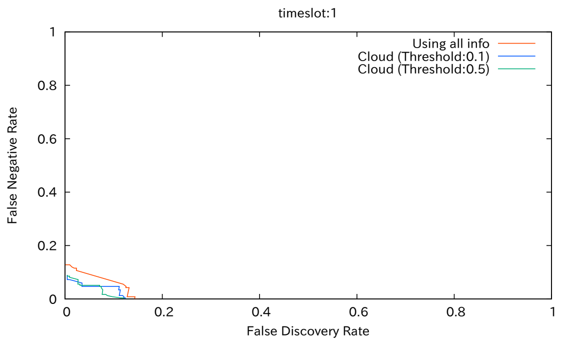
Figure 4: Prediction result of the time when the obstacle enters the area of edge 0 (before 1 time slot)

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

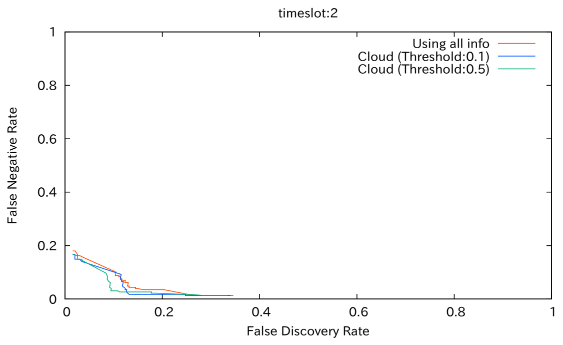
$$\text{False negative rate} = \frac{\text{Number of FNs}}{\text{Number of points with obstacles}} \tag{6}$$

$$\text{False discovery rate} = \frac{\text{Number of FPs}}{\text{Number of points predicted as points with obstacles}} \tag{7}$$

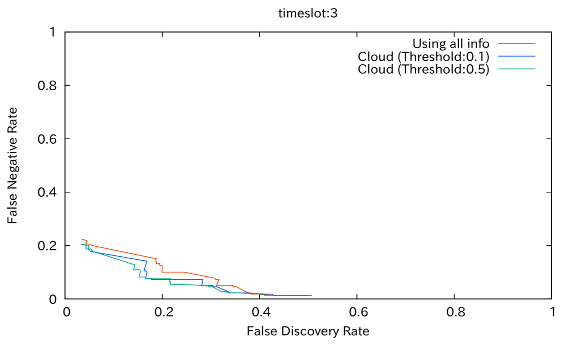
Figure 5 shows the results for the prediction by the cloud by setting the interval of updating the model by the cloud to 1, and the threshold λ to 0.1 and 0.5. We also plot the results by the predictor collecting all information. These figures demonstrate that our model accurately predicts the future position of the obstacles, while the accuracy slightly decreases as the prediction target becomes future. These figure also indicate that the cloud can achieve similar prediction accuracy to the predictor using all information. That is, the cloud can obtain the information required to predict.



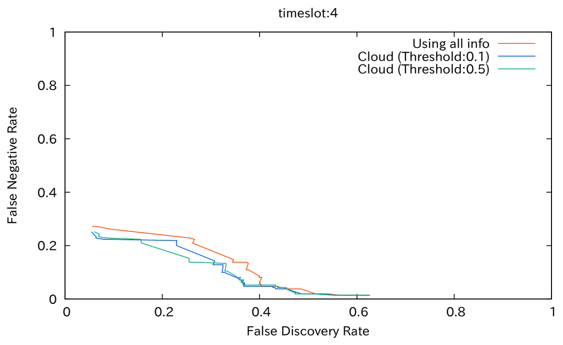
(a) As of 1 time slot ago



(b) As of 2 time slot ago



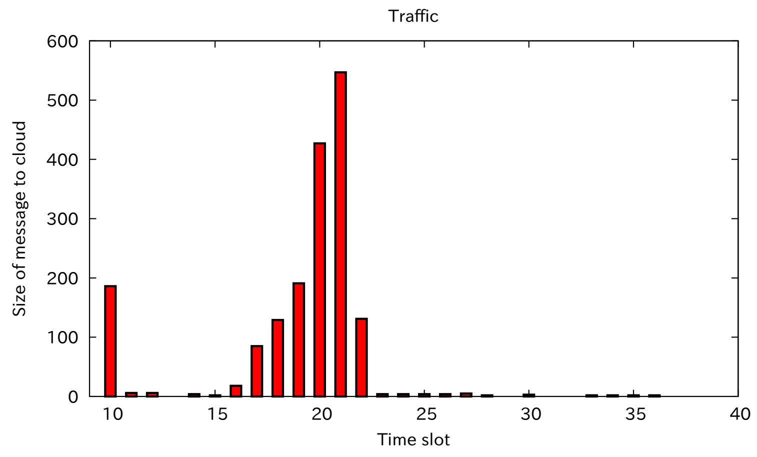
(c) As of 3 time slot ago



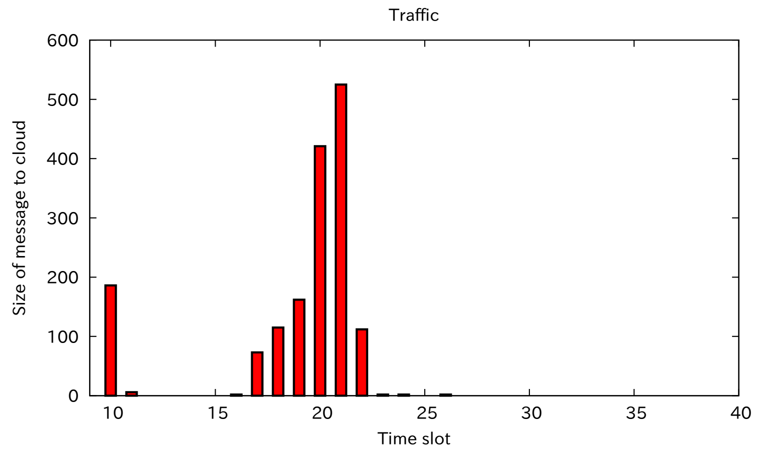
(d) As of 4 time slot ago

Figure 5: FDR and FNR (Cloud)

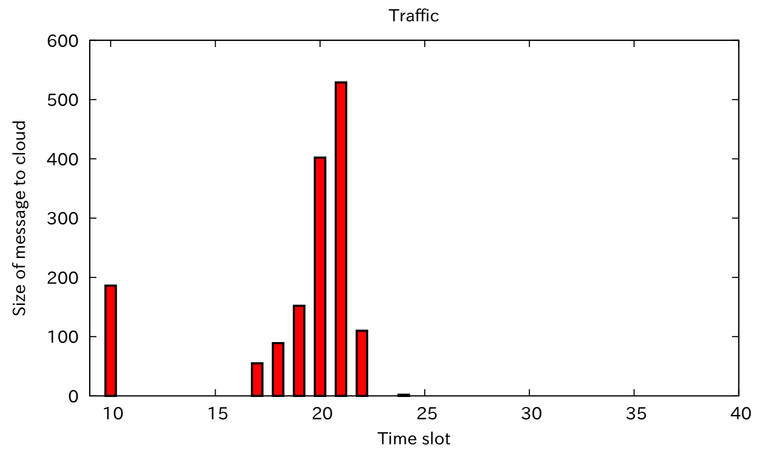
Figure 6 shows the number of points whose information was sent to the cloud. As shown in this figure, the number of points sent to the cloud changes in time. In some time slots, no information is sent. This is because edge computers send only the information that cannot be expected from the previously sent information.



(a) Threshold=0.01



(b) Threshold=0.1



(c) Threshold=0.5

Figure 6: Number of points whose information was sent to cloud

Figure 7 shows the calculation time of process in each step. In each step, each edge predict the state for 5 time slots by using the data with past 10 time slots. On the other hand, the cloud predict the state for 30 time slots. As shown in this figure, the calculation time of the edges is less than 100 ms. That is, we can update the model and prediction in real time.

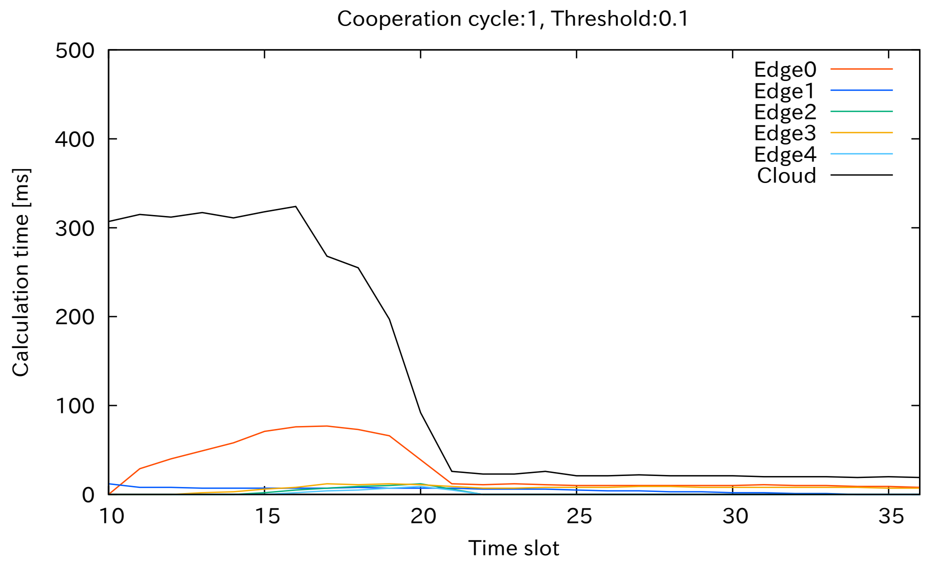
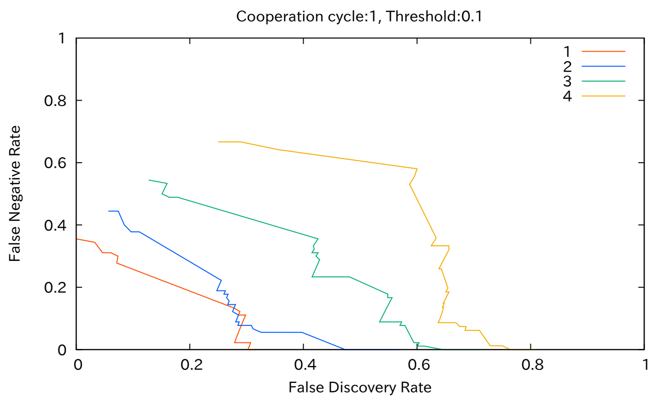


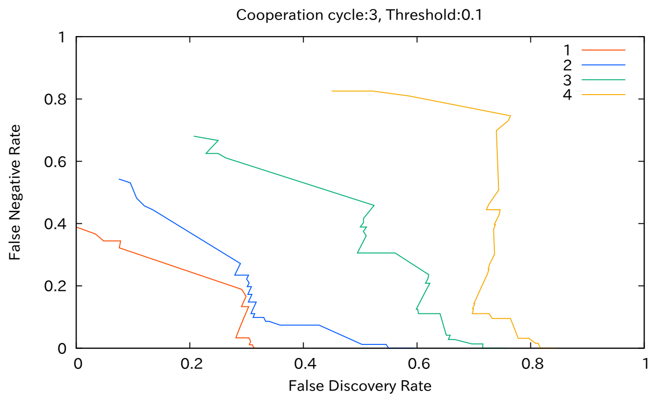
Figure 7: Calculation time in the cycle per 1 time slot and at the threshold of 0.1

Finally, we evaluate the accuracy of the prediction by edge computers. The interval to update the model of the cloud has an impact on the prediction by edge computers, because the edge computers use older information from servers as the interval becomes long. Therefore, we obtain the results for three cased on intervals, 1, 5, 10.

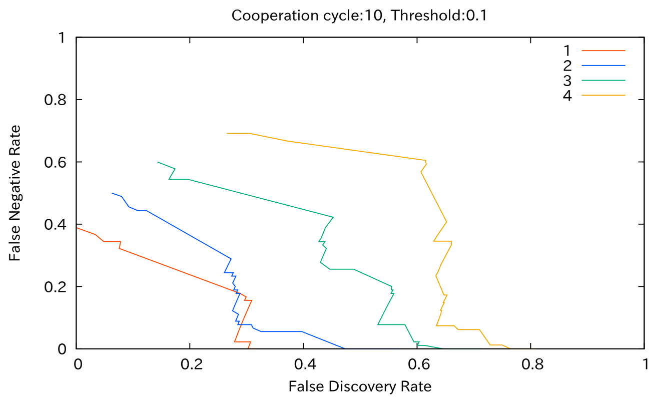
Figure 8 shows the results for the prediction by the edge computer of subarea 0 by setting the interval of updating the model by the cloud to 1, 3, and 10 and the threshold λ to 0.1. As shown in this figure, the edge computer of subarea 0 can achieve low FDR. This is because the edge computer can use the information predicted by the cloud. As a result, if the cloud predicts obstacles entering the subarea, the edge computer corresponding to the subarea can also predict the obstacles. Figure 8 also indicates that the edge computer achieves similar accuracy even in the case with the large interval to cooperate with the cloud. This is because the impact of the information from the cloud is limited only to the case that new obstacles may enter the corresponding subarea. That is, edge cloud cooperation works properly even when we set a large interval for the cloud to update their model.



(a) Interval of Cooperation=1



(b) Interval of Cooperation=3



(c) Interval of Cooperation=10

Figure 8: FDR and FNR (Edge 0)

5 Conclusion

To control mobile robots, the detection and prediction should be updated in real time. However, it takes a time to aggregate the information from sensors and predict future state of obstacles as the target area becomes large and the number of obstacles becomes large.

In this thesis, we proposed a method to predict obstacles in real time by cooperation between cloud and edge computers. In this method, we divide the target area into multiple subareas and deploy an edge computer for each subarea. Each edge computer collects and aggregates the information from the sensors in the corresponding area, and detects and predicts obstacles in the corresponding area. In addition, we also deploy a cloud computing environment or high-performance computer. The cloud aggregates the information from the edge computers and detects and predicts the all obstacles in the target area. In addition, the cloud also shares its results of prediction with the edge computers. By using the information from the clouds, each edge can predict the obstacles from the outside of its corresponding subarea.

In this thesis, we also proposed a method to reduce the amount of information exchanged between the cloud and edge computers. In this method, each edge send only the information of the points whose states were not expected from the previously sent information. By sending only such information, the cloud can detect and predict obstacles accurately enough by exchanging a small amount of information.

We evaluated our method by simulation. The results show that our method achieves the similar prediction accuracy to the method collecting all information, while our method updates the prediction in less than 100 ms at edges.

One of our future work is to improve the accuracy of the prediction. One approach is to use the more detailed information of the obstacles. For example, if we can obtain the attributes of the obstacles, more accurate model for obstacles can be built.

Acknowledgments

I am deeply grateful to Professor Masayuki Murata of the Graduate School of Information Science and Technology, Osaka University. He took time for me and gave me good advice and suggestions for my research in a regular meeting. It has been a great help to my research. I would like to express my sincere appreciation to Associate Professor Yuichi Ohsita of the Graduate School of Information Science and Technology, Osaka University. He kept in close contact with me and advised me on various aspects of this thesis even when my research was not progressing well. I could not have completed this thesis without his advice. I would also like to thank Associate Professor Shin'ichi Arakawa, Assistant Professor Daichi Kominami, Assistant Professor Tatsuya Otoshi, and Specially Appointed Assistant Professor Masaaki Yamauchi of the Graduate School of Information Science and Technology, Osaka University for their continued support. Finally, I would like to thank all the members of the Advanced Network Architecture Research Laboratory at the Graduate School of Information Science and Technology, Osaka University, for their discussion and advice.

References

[1] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674–691, 2021.

[2] A. C. Bavelos, N. Kousi, C. Gkournelos, K. Lotsaris, S. Aivaliotis, G. Michalos, and S. Makris, "Enabling Flexibility in Manufacturing by Integrating Shopfloor and Process Perception for Mobile Robot Workers," *Applied Sciences*, vol. 11, no. 9, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/9/3985

[3] S. Yasuda, T. Kumagai, and H. Yoshida, "Cooperative Transportation Robot System Using Risk-Sensitive Stochastic Control," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5981–5988.

[4] L. Wang, H. Ye, Q. Wang, Y. Gao, C. Xu, and F. Gao, "Learning-based 3D occupancy prediction for autonomous navigation in occluded environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 4509–4516.

[5] Z. Rozsa and T. Sziranyi, "Obstacle prediction for automated guided vehicles based on point clouds measured by a tilted lidar sensor," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2708–2720, 2018.

[6] D. Ferguson, M. Darms, C. Urmson, and S. Kolski, "Detection, prediction, and avoidance of dynamic obstacles in urban environments," in *2008 IEEE intelligent vehicles Symposium*. IEEE, 2008, pp. 1149–1154.

[7] A. F. Foka and P. E. Trahanias, "Probabilistic autonomous robot navigation in dynamic environments with human motion prediction," *International Journal of Social Robotics*, vol. 2, pp. 79–94, 2010.

[8] C.-h. Chinag and C. Ding, "Robot navigation in dynamic environments using fuzzy logic and trajectory prediction table," in *2014 international conference on fuzzy theory and its applications (iFUZZY2014)*. IEEE, 2014, pp. 99–104.

[9] B. Liu, X. Wang, W. Zhou, W. Zhou, Y. Chang, and Q. Cai, "Dynamic path planning based on variable step size rolling window derivation and obstacle prediction," in *IOP Conference Series: Materials Science and Engineering*, vol. 631, no. 3. IOP Publishing, 2019, p. 032041.

[10] Y. Ohsita, S. Yasuda, T. Kumagai, H. Yoshida, D. Kanetomo, and M. Murata, "Spatio-temporal model that aggregates information from sensors to estimate and predict states of obstacles for control of moving robots," in *Proceedings of International Conference on Emerging Technologies for Communications*, December 2022.

[11] S. Liang, H. Wu, L. Zhen, Q. Hua, S. Garg, G. Kaddoum, M. M. Hassan, and K. Yu, "Edge YOLO: Real-time intelligent object detection system based on edge-cloud cooperation in autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 25 345–25 360, 2022.

[12] W. Wang, T. Hu, and J. Gu, "Edge-cloud cooperation driven self-adaptive exception control method for the smart factory," *Advanced Engineering Informatics*, vol. 51, p. 101493, 2022.

[13] Y. Li, C. Li, X. Li, K. Wang, M. M. Rahaman, C. Sun, H. Chen, X. Wu, H. Zhang, and Q. Wang, "A comprehensive review of Markov random field and conditional random field approaches in pathology image analysis," *Archives of Computational Methods in Engineering*, vol. 29, no. 1, pp. 609–639, 2022.

[14] K. Murphy, Y. Weiss, and M. I. Jordan, "Loopy Belief Propagation for Approximate Inference: An Empirical Study," 2013. [Online]. Available: https://arxiv.org/abs/1301.6725

[15] S. Kosov, "Direct graphical models c++ library," http://research.project-10.de/dgm/, 2013.