applied sciences

MDPI

Article

On Sharing Spatial Data with Uncertainty Integration Amongst Multiple Robots Having Different Maps

Abhijeet Ravankar [1,*,†], Ankit A. Ravankar [2,†], Yohei Hoshino [1] and Yukinori Kobayashi [2]

[1] School of Regional Innovation and Social Design Engineering, Faculty of Engineering, Kitami Institute of Technology, Kitami, Hokkaido 090-8507, Japan
[2] Division of Human Mechanical Systems and Design, Faculty of Engineering, Hokkaido University, Sapporo, Hokkaido 060-8628, Japan
* Correspondence: aravankar@mail.kitami-it.ac.jp
† These authors contributed equally to this work.

Received: 30 May 2019; Accepted: 4 July 2019; Published: 8 July 2019

Abstract: Information sharing is a powerful feature of multi-robot systems. Sharing information precisely and accurately is important and has many benefits. Particularly, smart information sharing can improve robot path planning. If a robot finds a new obstacle or blocked path, it can share this information with other remote robots allowing them to plan better paths. However, there are two problems with such information sharing. First, the maps of the robots may be different in nature (e.g., 2D grid-map, 3D semantic map, feature map etc.) as the sensors used by the robots for mapping and localization may be different. Even the maps generated using the same sensor (e.g., Lidar) can vary in scale or rotation and the sensors used might have different specifications like resolution or range. In such scenarios, the 'correspondence problem' in different maps is a critical bottleneck in information sharing. Second, the transience of the obstacles has to be considered while also considering the positional uncertainty of the new obstacles while sharing information. In our previous work, we proposed a 'node-map' with a confidence decay mechanism to solve this problem. However, the previous work had many limitations due to the decoupling of new obstacle's positional uncertainty and confidence decay. Moreover, the previous work applied only to homogeneous maps. In addition, the previous model worked only with static obstacles in the environment. The current work extends our previous work in three main ways: (1) we extend the previous work by integrating positional uncertainty in the confidence decay mechanism and mathematically model the transience of newly added or removed obstacles and discuss its merits; (2) we extend the previous work by considering information sharing in heterogeneous maps build using different sensors; and (3) we consider dynamic obstacles like moving people in the environment and test the proposed method in complex scenarios. All the experiments are performed in real environments and with actual robots and results are discussed.

Keywords: information sharing; multi-robot systems; positional uncertainty; path planning; mapping

1. Introduction

Mobile robots are increasingly being used to automate many tasks; tasks which are mostly dull, dangerous, or demanding are a good fit for autonomous robots. The industrial sector has already benefited a lot from 'factory robots'. Recently, a new class of robots called 'service robots' have been increasing. These robots are used to provide several common services like cleaning and delivering, dispatching and moving items. These service robots are also used for specific tasks like patrolling and escorting people. Generally, multiple robots are used for such tasks in large service areas as there are several advantages. One of the major advantages of using multiple robots is wide area coverage.

Multiple robots can cover a large area and perform several tasks simultaneously. Task parallelism is possible as different robots can perform different tasks at the same time. Some robots may be cleaning, some patrolling, while others may be delivering items to specific locations. Fault tolerance is another advantage of multi-robot systems. Even if one of the robots goes out of service the entire service does not stop as other robots can finish the task. Moreover, with task coordination, multiple robots can perform the task efficiently and quickly.

However, with the introduction of multiple robots in a system, there are several challenges which need to be addressed. Among these problems, effective communication between the multiple robots is a major challenge. Communication forms the basis of other major modules like task coordination, task distribution and collective execution. Accurate and content rich information is important for the successful execution of many tasks.

Although there are many benefits of sharing spatial information in a multi-robot system, in this paper, we consider the case of sharing obstacle information in a multi-robot system. The environments at many service places, like hospitals and warehouses, are very dynamic with moving entities and new obstacles. To navigate autonomously in such environments, robots need a map of the environment and need to localize themselves within it. This is generally achieved through a SLAM (Simultaneous Localization and Mapping) [1] module. Generally, if one robot finds a new obstacle in the environment, it only updates its own map. The other robots do not benefit from this knowledge. However, if the robot shares this knowledge with other robots along with updating its map, other robots can update their maps and plan better paths with the real-time information. This is shown in Figure 1, in which, Robot R1 finds a new obstacle and blocked path at the center passage of the service area and shares the spatial coordinates of the obstacle with other robots R2, $\cdots$, R5 which can use this information in generating optimal trajectories. The extension of use-case scenarios other than obstacle information sharing is straightforward.

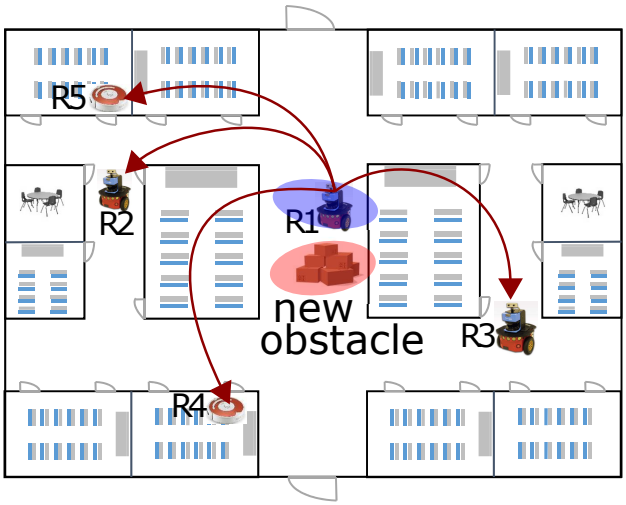


Figure 1. Robot R_1 finds a new obstacle blocking the path and shares this information with other robots ($R_1, \cdots, R_5$). The blue and red ellipses represents the robot's and obstacle's positional uncertainty, respectively.

Related Works

There is a plethora of previous works related to sharing information in multi-robot systems. Sharing corresponding matches of an object by two robots to calculate an accurate relative localization over time is proposed in Reference [2]. Work in Reference [3] proposes sharing visual information. In Reference [4], task negotiation between multiple robots by sharing information is proposed to decide the sequence in which the tasks should be performed by different robots. Work in Reference [5,6] proposes a protocol to share the region of interest between robots for efficient task cooperation. In-fact,

multi-robot sport activities like Robo-soccer [7,8] heavily relies on meaningful information sharing between robots to achieve a common goal. Virtual pheromones have been proposed to be used for coordinating master-slave robots in References [9,10]. Path planning of multiple robots using information from external security cameras is proposed in Reference [11]. In addition, a direct obstacle coordinate information sharing was proposed in our previous work [12] without considering the uncertainty. However, this is a limitation as in practical systems there is always some uncertainty associated with robot's localized information and mapped obstacle's position due to sensor errors. RoboEarth [13–15] is another platform which heavily uses information exchange through cloud.

Such information sharing has huge merits in robot path planning. Path planning is an active area of research and in the context of multi-robot systems path-planning has shown promising advantages through information sharing between robots. Multi-robot collision avoidance has been discussed in Reference [16]. Work in Reference [17] presents a mechanism in which robots share information about their remaining battery power and accordingly avoid collision by giving priority to a robot with less battery power over the shortest path. An interesting approach of collaborative navigation through visual-servoing is presented in Reference [18,19] which heavily relies on reliable and efficient inter-robot communication to share information. The proposed work focuses on multiple robots sharing information about the dynamic changes in the remote area of the environment. This enables the robots to use updated and timely information to efficiently plan their paths. Information sharing among multiple robots for efficient path planning usually involves a decentralized approach [20] in which each robot calculates its path individually and decisions to change paths or avoid obstacles is done later based on the received messages from other robots. This is unlike centralized path planners [21] in which all the paths of all the robots are calculated simultaneously. In Reference [22], a motion planner is proposed for multiple robots with limited ranges of sensing and communication to reach the goal in dynamic environments. In Reference [23], a navigational technique for multiple service robots in a robotic wireless network (RWN) is presented in which robots download map information from map servers for safe navigation. Semantic information is used among multiple robots for efficient task coorindation in Reference [24].

In Reference [25], a practical case of multi-robot navigation in warehouse has been discussed. The proposed work also deals with the positional uncertainty of robots and obstacles. In this context, a decentralized approach for collaboration between multiple robots in presence of uncertainty are considered for robot action in Reference [26]. A review of multi-robot navigation strategies can be found in References [27–29].

The proposed work is an extension of our previous work [12]. Our previous work proposed the idea of a 'Node-Map' and obstacle's confidence decay mechanism. However, there were many limitations which are addressed in this extended work. The new major contributions are:

1. **Uncertainty Integration in the Improved Confidence Decay Mechanism:** The previous work [12] did not consider the amount of estimated positional uncertainty of obstacles in the confidence decay. Both were decoupled entities. However, this was a serious drawback in the previous work because irrespective of the amount of positional uncertainty, confidence of all the obstacles decayed at the same rate. This caused several false map updates corresponding to dynamic obstacles which generally have large uncertainty associated. In the extended work, we have mathematically modeled the integration of positional uncertainty in the confidence decay mechanism. This is discussed in 'Section 4.1 Integrating Uncertainty in Confidence Decay Mechanism'.
2. **New Experiments with Heterogeneous Maps with Different Sensors:** Another shortcoming of the previous work was that it only worked with the same type of 2D grid-maps made with the same type of sensors. However, in the extended work, we include new experiments with heterogeneous maps (3 dimensional RGBD map and 2D grid-map) made from different sensors. In this regard, the merits of using the 'node-map' as a means of smoothly sharing information

coherently between heterogeneous maps are also discussed. This is discussed in 'Section 6.1 Experiments with Heterogeneous Maps'.

3. **New Experiments in Dynamic Environment with Moving People and Testing Under Pressure:** The previous work only worked with static obstacles. In the extended work, new experiments have been performed to test the method when people are randomly moving in the vicinity of the robot and obstructing its navigation. In this regard, the tight coupling of new obstacle's uncertainty in the confidence decay mechanism plays a vital role to avoid false map-updates corresponding to the dynamic obstacles. This is discussed in 'Section 6.2. Results with Dynamic Entities (Moving Obstacle)'.

The comparison of the previous work with the extended work is summarized in Table 1. In addition, the proposed work discusses the algorithm to generate the T-node map.

Table 1. Comparison of this extended work with the previous work [12].

Feature	Previous Work [12]	Extended Work
Sharing New Obstacle's Position Information	Yes	Yes
Consideration of Positional Uncertainty of Obstacles	No	Yes
Confidence Decay Mechanism	Yes	Yes
Uncertainty Influence Over Confidence Decay	No	Yes
Experiments in Very Dynamic Environment (e.g., Moving People)	No	Yes
Robots have Different Types of Sensors	No	Yes
Tests with Heterogeneous Maps	No	Yes

The paper starts by first explaining the correspondence problem in different maps in Section 2. The node-map representation is explained in Section 3. Section 4 briefly explains obstacle removal and update in the nodemap and Section 4.1 explains the integration of positional uncertainty in the confidence decay mechanism. Further, using this coupling with Extended Kalman Filter is explained in Section 5 with detailed algorithm. The experimental results are discussed in Section 6. Section 6.1 explains about the experiments with heterogeneous maps and Section 6.2 discusses the results with dynamic entities (moving people). Finally, Section 7 concludes the paper.

2. Correspondence Problem in Different Maps

In dynamic environments, the new objects in the environment could be the temporary or new permanent obstacles. Both needs to be estimated in the map for correct path planning. A robot estimates the absolute position (x_{obs}, y_{obs}) of an obstacle in its map through its SLAM module. This estimation also has an uncertainty (Σ_{obs}) associated with it which arises mainly from sensor errors. This information about the new obstacle $(x_{obs}, y_{obs}, \Sigma_{obs})$ is difficult to be directly shared with other robots.

A common problem occurring in multi-robot systems is information sharing in different types of maps (e.g., 2D grid-map, 3D semantic map, feature map etc.) made from different sensors used by the robots for mapping and localization. Even the maps generated using the same sensor (e.g., Lidar) can vary in scale or rotation and the sensors used might have different specifications like resolution or range. In such scenarios, the 'correspondence problem' in different maps is a critical bottleneck in information sharing. Moreover, the uncertainty of localization also adversely affects the information sharing. In other words, it is important to consider how to easily correspond local spatial information in one map to spatial information in a separate map of different type or scale while considering the uncertainty.

This is graphically explained in Figure 2. There is a scale difference between Map1 and Map2. Whereas, Map2 and Map3 differ by a rotation factor. A spatial obstacle information, for example, position (x^1, y^1) will correspond to different spatial coordinates in Map2 and Map3. In most real world scenarios, these scale and rotation differences are generally not known. Some previously proposed

techniques [30] to find the necessary translation and rotation can be applied to transform the spatial information in one map to another. However, the computation costs are expensive and could introduce undesired delays.

Although the example in Figure 2 is simplified for illustration, in actual scenarios different maps may have different levels of noise and even feature dimensions. Moreover, some robots may only have a partial map information. Similarly, Figure 3 discusses the problem of robots having different types of maps [31]. Figure 3a is a map in the form of a graph, Figure 3b is a dense 3D map, while Figure 3c is the gridmap of the same environment. It is difficult to for the robots to correlate spatial information in such different types of maps.

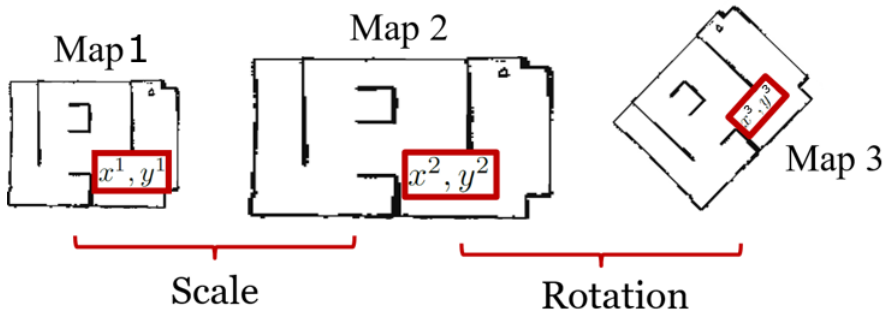
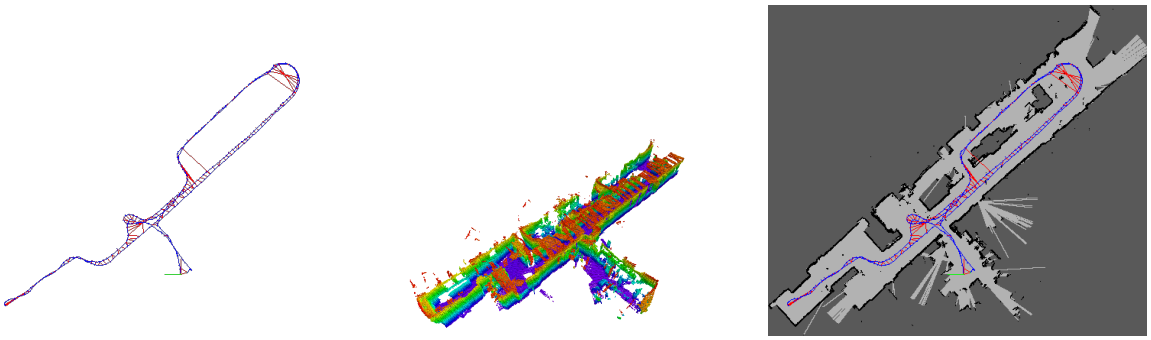


Figure 2. Correspondence problem due to the scale and rotational differences between maps.



(**a**) Graph Map. (**b**) Dense 3D Map. (**c**) Grid Map.

Figure 3. Correspondence problem due to the different types of maps [31] of the same environment. (**a**) Graph map. (**b**) Dense 3D map. (**c**) Grid map.

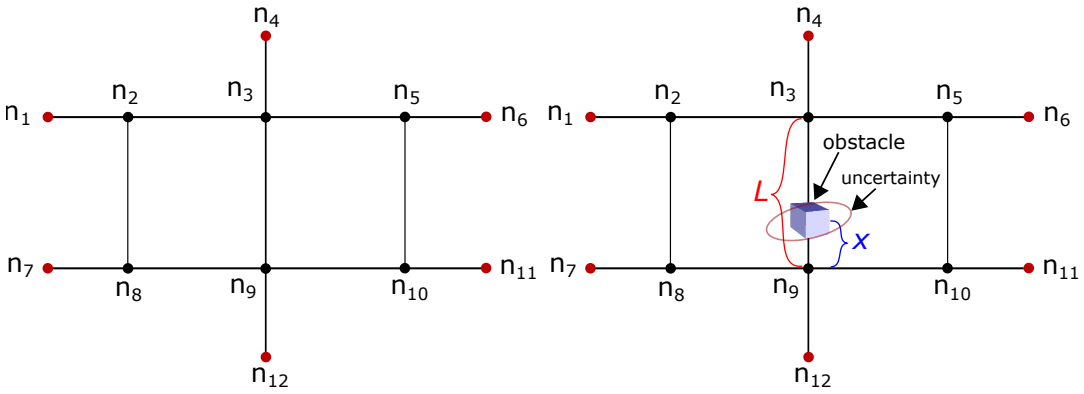
In the proposed work, it is assumed that the robots work in the common service area whose map is available to the robots. This map itself could be heterogenous, for example, grid-map, RGBD map and so forth, which is built using different sensors mounted on different robots. Moreover, the maps could be build from different anchor points. Thus, different robots could have heterogeneous maps.

3. 'T-Node' Map Representation

A T-node representation of the map has been proposed in our previous work [12]. We briefly explain the T-node map and how obstacles are represented in it. Later, we describe how path planning is done on the node map. It is assumed that each robot is also assigned a unique robot-id (R_{id}) and the robots are on the same network to exchange messages with each other.

A node is defined as a point of turn in a path of the map. The paths are represented as a network of these nodes in the map. Figure 4a shows the node representation of the environment shown in Figure 1. Notice that, the nodes $n_1, n_2, \cdots, n_{12}$ are the points of turns in the map. The terminal nodes are shown in red color in Figure 4a. Nodes are connected to each other through edges. Figure 4b shows the T-node map with an obstacle placed between the nodes n_9 and n_3. The distance between

the nodes n_9n_3 is L and the distance of the obstacle from node n_3 is x, which can easily be estimated using an on-board distance sensor.



(**a**) T-node map without obstacle. (**b**) T-node map with obstacle.

Figure 4. T-node representation of the environment shown in Figure 1. (**a**) T-node map without obstacle. (**b**) T-node map with obstacle between nodes n_9n_3.

A table stores T-node map's information viz. traversable/blocked edges (paths) and any changes at the edges. All the robots have access to this table. In the context of Figure 4b, the corresponding information is shown in Table 2. The table contains a set of four information about each path: (1) binary information of whether a new obstacle is found on an edge, (2) a binary information if the path is blocked and cannot be traversed, (3) details of the obstacle if the path is changed and (4) the timestamp (T_s) when the information was updated. The details of the information will vary according to the type of the sensor used. For example, in case of Lidar, the obstacle information will contain: the obstacle coordinates from the node (d_x, d_y), dimensions of the obstacle like width (w_{obs}) and height (h_{obs}) and the positional uncertainty associated in estimating the obstacle (σ_x, σ_y). The uncertainty information comes from the SLAM module used in the robot. As shown in T-node map of Figure 4b, only one of the edges n_3n_9 is obstructed. This information is reflected in Table 2. It is possible that a new obstacle is found on a path, however the path could be still be traversed. A blocked path cannot be traversed by the robot.

Table 2. T-node map information corresponding to Figure 4b.

Node Path	New Obstacle	Path Blocked	Meta-Data
n_1n_2	0	0	-
...	...	...	...
n_3n_9	1	1	{ d:(d_x, d_y),w:w_{obs},h:h_{obs},Σ:(σ_x, σ_y), T_s }
n_9n_{12}	0	0	-

Each robot has a copy of this table which has small memory requirement as the meta-data for only the changed paths are required. Moreover, information is communicated to other robots only when some path information is changed. A T-node representation makes it easier for a robot to share information with other robots. The local maps maintained by the two robots might differ by some rotation, translation or scale. As an example, Figure 5a shows the section of the map of Figure 4b with obstacles. Figure 5b shows a scaled version, Figure 5c a rotated version and Figure 5d a scaled and rotated version of Figure 5a. However, the nodes on the paths remains the same and information that there is an obstacle on one of the edges is still conveyed clearly from Table 2 which maintains the details of the obstacles. In addition, with a T-node representation a global map is not required.

Even for a large number of nodes, only those edges which are changed is communicated to the robots. The small data size ensures fast and reliable communication with small communication bandwidth.

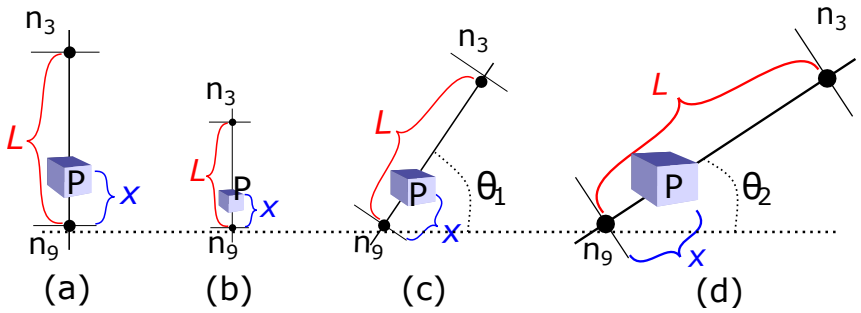


Figure 5. Scale and rotation effects on the T-node map. In all cases, meaningful information can still be shared between robots. (**a**) Original section n_9n_3 of Figure 4b. (**b**) Figure 5a scaled down. (**c**) Figure 5a rotated by angle θ_1. (**d**) Figure 5a scaled up and rotated by angle θ_2.

The T-node map can be generated by maneuvering the robot in the environment and setting points of turns (where the robot turns by around 90 degrees) as nodes. Automatic generation of T-node map is also possible if a map is available. For example, if there is a grid-map with obstacles (black), open (white) and unknown (grey) areas, the first step is to generate a binary image of the grid-map which is done by turning all unknown cells to blocked (black) value. This is shown in Figure 6a. Noise is removed by successively applying morphological erode and dilate operations [32,33]. The next step is to apply skeletonization algorithm [34,35]. Many skeletonization and thinning algorithms generate unnecessary tentacles which needs to be removed using pruning algorithm [36]. Result of showing skeletonization on binary map of Figure 6a is shown in Figure 6b. Line segments are then detected using techniques like SVD and Hough Transform [1]. The end-points of segments which are within a small threshold distance (δ) can be clustered [37] using k-means [38,39], fuzzy c-means [40] or density based clustering methods [41] into a single node as shown in Figure 6c,d. A graph 'N' of these nodes $\{n_1, n_2, \cdots, n_m\}$ form the T-node map of the environment. The pseudo-code is given in Algorithm 1.

Algorithm 1: T-node-map Generation

```
Data: m : Gridmap, m_height : map height, m_width : map width
Function node_mapping(m)
    for each row in m_height do
        for each col in m_width do
            if cell m[row][col] is unknown then
                m[row][col] ← l_occupied
Successively erode and dilate binary image multiple times [32,33]
Apply skeletonization algorithm [34,35]
Apply prunning algorithm [36]
Detect lines segments and their endpoints using algorithm [1]
Cluster nearby endpoints in range δ with k means algorithm. [37]
Mark clustered points as nodes N ← {n_1, n_2, ··· , n_m}
return(N)
```

It should be noted that a 'node' is merely a point of turn in the navigational graph. It does not include any feature information (e.g., corners, line, color, etc.) of the map. Hence, map-merger on T-Node map is not possible. However, traditional methods [42,43] can be used to first merge feature-rich maps and thereby T-node map. Moreover, since the characteristics of the navigational paths are different for unmanned ground vehicles (UGVs) and unmanned aerial vehicles (UAVs), this work does not consider the case of heterogeneous robots. Only ground robots are considered and the proposed method will work well for both differential drive robots and skid-steer drive robots.

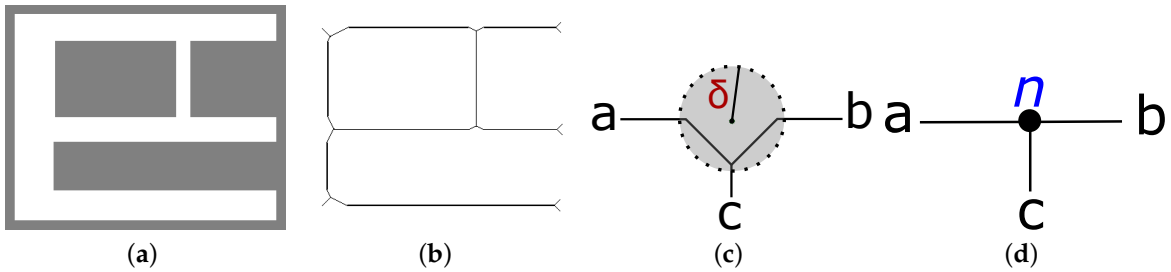


Figure 6. T-node map generation. (**a**) Binary grid-map. (**b**) Skeleton map. (**c**) Clustering within δ distance. (**d**) Clustered node n.

4. Obstacle Removal and Update in T-Node Map

This section briefly discusses about modeling the transience of obstacles first proposed in our previous work [12]. The new contribution of this extended work lies in integrating the positional uncertainty in the decay mechanism which is discussed in the next Section 4.1. The obstacles in the passages may be permanent or temporary and removed after some time. Regardless of the transience of the obstacles, all the robots update their respective T-node map. Newly added obstacles can only be re-confirmed if a robot actually visits the area near the obstacle. Hence, once an obstacle information on a particular node has been updated and communicated to other robots, robots update their map and if that obstacle is found again, the timestamp is updated. However, there is no upper time bound of when a robot would actually visit the particular location and update its map. The obstacle might already have been removed by that time. This problem needs to be modeled mathematically.

A timestamp (T_s) is maintained for each obstacle representing the time at which the obstacle was last seen. If an obstacle has recently been added to the map and a short time has elapsed since its addition, then the probability that it has not been removed is high. On the contrary, if a lot of time has elapsed since the addition of the obstacle, the probability that it still exists in the map is less. We model a confidence (c) measure which represents this probability $0 \leq c_{th} \leq 1$. The maximum value of confidence is 1 and its value decreases with time. The robots assumes that the obstacle still exists in the map until the corresponding confidence has not dropped to below a threshold confidence (C_{th}). Depending on the nature of the environment, a threshold time (t_{th}) is chosen in which the confidence decays to c_{th} value and the time in which the confidence decays to zero is (t_z). To model the confidence decay, the following family of curves are chosen.

$$c = 1 - \left(\frac{t_{th}}{t_z}\right)^n \tag{1}$$

The curves given by Equation (1) have the desired characteristic that for higher values of n, the curve flattens out more and delays confidence decay until the threshold time (t_{th}) and after that it decays quickly to zero in t_z time. For a given c_{th}, t_{th} and t_z, the value of the degree of the curve (n) can be found by solving Equation (1) as,

$$\begin{aligned} \left(\frac{t_{th}}{t_z}\right)^n &= 1 - c_{th}, \quad (0 \leq c_{th} \leq 1), \\ n \log\left(\frac{t_{th}}{t_z}\right) &= \log(1 - c_{th}), \\ \implies n &= \frac{\log(1 - c_{th})}{\log\left(\frac{t_{th}}{t_z}\right)}. \end{aligned} \tag{2}$$

Figure 7 shows the curves for the decay function given by Equation (1) for various values of n. The Ufactor in Figure 7 shows the uncertainty factor which is discussed in Section 4.1. The various curves have been generated for $c_{th} = 0.55$ and $t_z = 600$ s, for varying values of t_{th} between 300 s to

600 s. It can be seen that the corresponding values of n can be found for different t_{th} according to Equation (2). Moreover, as the value of n increases, the decay curves flatten out more taking more time to reach the threshold time and then quickly decrease to zero.

For a given instantaneous value of confidence c, the elapsed time t is calculated from Equation (1) as,

$$t = e^{\frac{1}{n}\log(1-c)+\log(t_z)}. \tag{3}$$

The time remaining (t_{rem}) to reach the threshold time (t_{th}) is,

$$t_{rem} = t_{th} - e^{\frac{1}{n}\log(1-c)+\log(t_z)}. \tag{4}$$

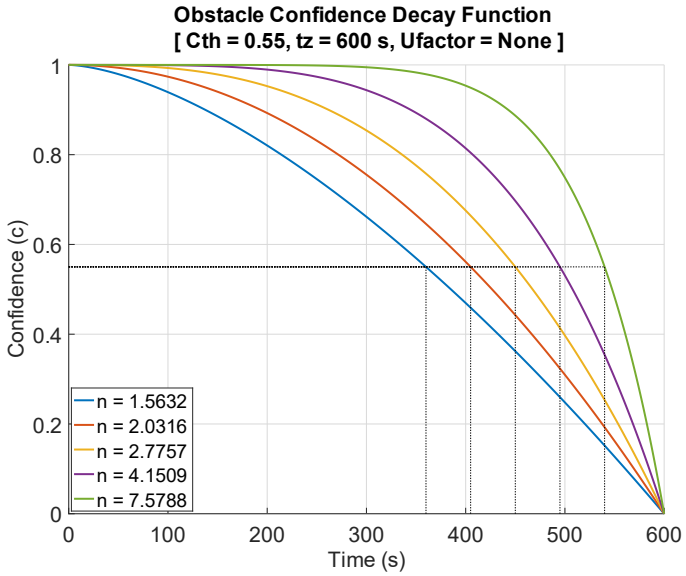


Figure 7. Obstacle confidence decay function. $c_{th} = 0.55$, $t_z = 600$ s. Effects of uncertainty are not considered and Ufactor = 0.

4.1. Integrating Uncertainty in Confidence Decay Mechanism

Uncertainty in obstacle's position affects the rate of confidence decay. If there is a large uncertainty in obstacle's spatial position, the threshold time t_{th} is reduced by an uncertainty factor. In case of no uncertainty, Equations (1)–(4) are used. In SLAM, the obstacle's uncertainty in state is generally represented by the covariance matrix (Σ_t),

$$\Sigma_t = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 \end{bmatrix} \tag{5}$$

If the uncertainty given by Σ_t is large, the confidence falls down faster and vice-versa. Hence, the confidence decay is modeled as,

$$\text{Confidence decay} \propto \frac{1}{\text{Uncertainty given by } \Sigma_t}. \tag{6}$$

The eigenvalues ($\lambda_1, \cdots \lambda_n$) and eigenvectors ($\vec{v_1}, \cdots \vec{v_n}$) of the matrix Σ_t denotes the magnitude of the variance. Two largest eigenvalues λ_1 and λ_2 control the decay of confidence. The threshold time after uncertainty integration $t_{th}^{'}$ is given as,

$$t_{th}^{'} = t_{th} - \frac{\Psi}{\sqrt{\lambda_1^2 + \lambda_2^2}}, \tag{7}$$

where, Ψ is a controlling factor. The new confidence $c^{'}$ is given as,

$$\begin{aligned} c^{'} &= 1 - \left(\frac{t^{'}_{th}}{t_z}\right)^n \\ &= 1 - \left(\frac{t_{th} - \frac{\Psi}{\sqrt{\lambda_1^2 + \lambda_2^2}}}{t_z}\right)^n \\ &= t_z^{-n}\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{-n}{2}}\left[t_z^n\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{n}{2}} - \left\{t_{th}\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{1}{2}} - \Psi\right\}^n\right]. \end{aligned} \tag{8}$$

The degree of the curve is given as,

$$\begin{aligned} n^{'} &= \frac{\log(1 - c^{'}_{th})}{\log\left(\frac{t^{'}_{th}}{t_z}\right)}, \\ \implies n^{'} &= \frac{\log\left(1 - t_z^{-n}\left(\lambda_1^2 + \lambda_2^2\right)^{-\frac{n}{2}}\left[t_z^n\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{n}{2}} - \left\{t_{th}\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{1}{2}} - \Psi\right\}^n\right]\right)}{\log\left(t_{th}\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{1}{2}} - \Psi\right) - \log\left(t_z\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{1}{2}}\right)}. \end{aligned} \tag{9}$$

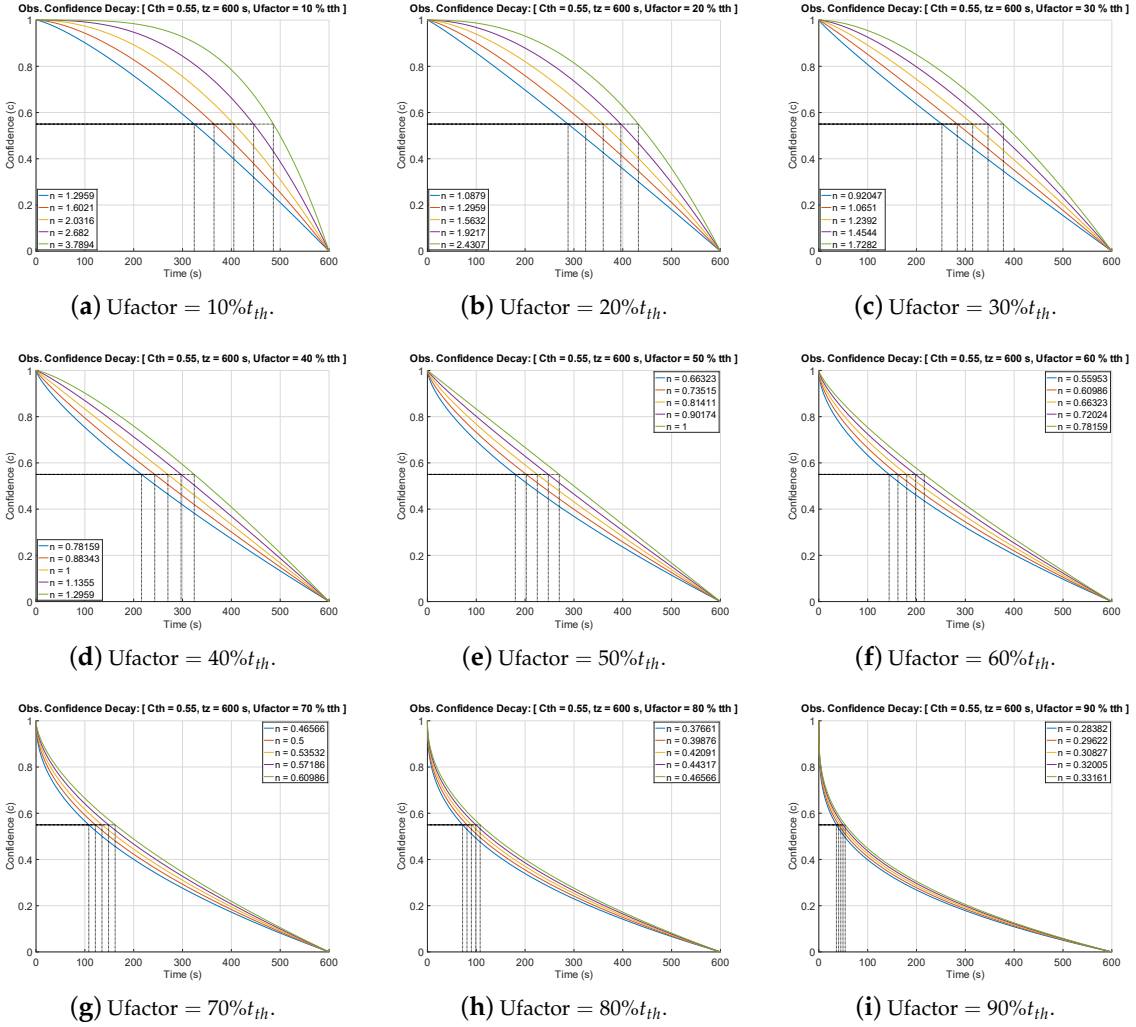
Figure 8 shows the results of integrating the spatial uncertainty in the obstacle confidence decay time for various values $c_{th} = 0.55$, t_z = 600 s and different values of the uncertainty factory (Ufactor). In Figure 8, Ufactor represents,

$$\text{Ufactor} = \frac{\Psi}{\sqrt{\lambda_1^2 + \lambda_2^2}}, \tag{10}$$

where, values λ_1 and λ_2 capture the amount of estimated uncertainty. In Figure 8, Ufactor is given as a factor of threshold time. It can be seen that for more uncertainty, the curve starts to fall faster to the threshold time. Appropriate values of Ufactor can be chosen depending on different scenarios. Moreover, this value can also be changed dynamically.

The obstacle confidence decay mechanism ensures a smooth robot operation in multi-robot system where multiple robots frequently inform each other about the new obstacle information. If a robot receives an obstacle information update from another robot while it is navigating towards its goal location, then it would have to stop and update its map information which consumes time and computation. To avoid this, a check is performed to see if the information received affects the current navigation towards the goal. This is easily achieved by checking the blocked flag of the corresponding edge. If the blocked flag is set to 1 and the current navigational path is affected, the timestamp and other meta-data for the blocked edge are checked. Based on the obstacle's confidence value, path re-planning or continuation on the same path can be decided according to the priority of the task at hand.

A major benefit of tightly coupling the obstacle's uncertainty with confidence decay mechanism is minimizing the false map updates corresponding to the dynamic obstacles in vicinity. Generally, the uncertainty of dynamic obstacles is larger than that of static obstacles estimated by the underlying SLAM module. In the absence of uncertainty integration, confidence all the obstacles irrespective of their positional uncertainty decays at the same rate. Therefore, if there is a false map update corresponding to a dynamic obstacle (like moving people), it decays at the same rate like other fixed obstacles. This increases the chances of false map updates due to dynamic obstacles. However, with the uncertainty integration, the confidence of obstacles with larger positional uncertainty decay faster than those with less uncertainty. In effect, this allows minimizing false map updates, as they decay out quickly. This also prevents false notifications to other robots.



(**a**) Ufactor = $10\%t_{th}$. (**b**) Ufactor = $20\%t_{th}$. (**c**) Ufactor = $30\%t_{th}$.

(**d**) Ufactor = $40\%t_{th}$. (**e**) Ufactor = $50\%t_{th}$. (**f**) Ufactor = $60\%t_{th}$.

(**g**) Ufactor = $70\%t_{th}$. (**h**) Ufactor = $80\%t_{th}$. (**i**) Ufactor = $90\%t_{th}$.

Figure 8. Obstacle confidence decay function with uncertainty integration. With more positional uncertainty, the confidence falls below the threshold confidence fast. In all the cases, $c_{th} = 0.55$ and $t_z = 600$ s.

5. Uncertainty Integrated Confidence Decay Mechanism with Extended Kalman Filter

The integration of confidence decay meachanism in EKF is given in Algorithm 2. The algorithm is straightforward and estimates the Kalman gain (K_t), robot's pose (μ_t) and the covariance (Σ_t) at time t until step 12. Later, Eigen values ($\lambda_1 \cdots \lambda_n$) are extracted from the covariance matrix (Σ_t) by applying Singular Value Decomposition. The degree of the confidence curve is then determined using the amount of uncertainty represented by the Eigen values in steps 14 and 15 of Algorithm 2 as explained in the previous section. Essentially, the degree of the curve is chosen to fasten the confidence decay inversely proportional to the positional uncertainty.

6. Experimental Results

This section presents the results of the experiments. The extended work discuss information sharing in heterogeneous maps made with different sensors and tests the proposed method under pressure with dynamic obstacles in the vicinity of robots.

We used Pioneer-P3DX [44] and Kobuki Turtlebot [45] robot shown in Figure 9a. Both the robots are wheeled differential drive robots and the motion model is explained in our previous work [12]. Both the robots used ROS [46] on Ubuntu computer and were on the same network to communicate with each other.

Algorithm 2: Uncertainty Integrated Confidence Decay with Extended Kalman Filter

1 # x_t: robot state, v_t, ω_t: translation and rotational velocity.
$x_t = [x \;\; y \;\; \theta]^T$

2 # EKF uses Jacobian to handle non-linearity. G_t: Jacobian of motion function w.r.t state
$$G_t \leftarrow \begin{bmatrix} 1 & 0 & -\frac{v_t}{\omega_t}cos\theta + \frac{v_t}{\omega_t}cos(\theta+\omega_t\Delta t) \\ 0 & 1 & -\frac{v_t}{\omega_t}sin\theta + \frac{v_t}{\omega_t}sin(\theta+\omega_t\Delta t) \\ 0 & 0 & 1 \end{bmatrix}$$

3 # V_t: Jacobian of motion w.r.t control
$$V_t = \begin{bmatrix} \frac{-sin\theta+sin(\theta+\omega_t\Delta t)}{\omega_t} & \frac{v_t(sin\theta-sin(\theta+\omega_t\Delta t))}{\omega_t^2} + \frac{v_t(cos(\theta+\omega_t\Delta t)\Delta t)}{\omega_t} \\ \frac{cos\theta-cos(\theta+\omega_t\Delta t)}{\omega_t} & -\frac{v_t(cos\theta-cos(\theta+\omega_t\Delta t))}{\omega_t^2} + \frac{v_t(sin(\theta+\omega_t\Delta t)\Delta t)}{\omega_t} \\ 0 & \Delta t \end{bmatrix}$$

4 # M_t: Covariance of noise in control space. $\alpha_1, \cdots, \alpha_4$: Error-specific parameters.
$$M_t = \begin{bmatrix} \alpha_1 v_t^2 + \alpha_2 \omega_t^2 & 0 \\ 0 & \alpha_3 v_t^2 + \alpha_4 \omega_t^2 \end{bmatrix}$$

5 # $\bar{\mu_t}$: Prediction updates in state.
$$\bar{\mu_t} = \mu_{t-1} + \begin{bmatrix} \frac{-v_t}{\omega_t}sin\theta + \frac{v_t}{\omega_t}sin(\theta+\omega_t\Delta t) \\ \frac{v_t}{\omega_t}cos\theta - \frac{v_t}{\omega_t}cos(\theta+\omega_t\Delta t) \\ \omega_t\Delta t \end{bmatrix}$$

6 # $\bar{\Sigma_t}$: Prediction updates in covariance.
$\bar{\Sigma_t} = G_t\Sigma_{t-1}G_t + V_tM_tV_t^T$

7 # $\bar{Q_t}$: Covariance of the sensor noise.
$$Q_t = \begin{bmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\phi^2 & 0 \\ 0 & 0 & \sigma_s^2 \end{bmatrix}$$

8 # $[m_{ix} \;\; m_{iy}]^T$: coordinates of the ith landmark. z_t^i: measurement. q: squared distance.
$q = (m_{k,x} - \bar{\mu}_{t,x})^2 + (m_{k,y} - \bar{\mu}_{t,y})^2$
$$\hat{z_t}^k = \begin{bmatrix} \sqrt{q} \\ atan2(m_{k,y} - \bar{\mu}_{t,y}, m_{k,x} - \bar{\mu}_{t,x}) - \bar{\mu}_{t,\theta} \\ m_{k,s} \end{bmatrix}$$

9 # H_t: Jacobian of measurement with respect to state.
$$H_t^k = \begin{bmatrix} -\frac{m_{k,x}-\bar{\mu}_{t,x}}{\sqrt{q}} & -\frac{m_{k,y}-\bar{\mu}_{t,y}}{\sqrt{q}} & 0 \\ \frac{m_{k,y}-\bar{\mu}_{t,y}}{q} & -\frac{m_{k,x}-\bar{\mu}_{t,x}}{q} & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

10 # S_t: Measurement covariance matrix.
$S_t^k = H_t^k\bar{\Sigma_t}[H_t^k]^T + Q_t$.

11 # $j(i)$: likely correspondence after applying maximum likelihood estimate.
$$j(i) = \text{argmax} \;\; \frac{1}{\sqrt{\det(2\pi S_t^k)}} \; e^{-\frac{1}{2}(z_t^i - \hat{z}_t^k)^T[S_t^k]^{-1}(z_t^i - \hat{z}_t^k)}$$

12 # K_t: Kalman gain, μ_t: state, Σ_t: covariance.
$K_t^i = \bar{\Sigma}_t[H_t^{j(i)}]^T[S_t^{j(i)}]^{-1}$
$\mu_t = \bar{\mu}_t + K_t^i(z_t^i - \hat{z}_t^{j(i)})$
$\Sigma_t = (I - K_t^iH_t^{j(i)})\bar{\Sigma}_t$

13 # Apply Singular Value Decomposition and get Eigen-values λ_i:
$$\lambda_1, \cdots \lambda_n = \text{svd}(\Sigma_t) = \text{svd}\left(\begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 \end{bmatrix}\right)$$

14 # n: degree of decay curve, t_{th}: threshold time, c_{th}: threshold confidence, t_z: time to decay to zero.
$$n = \frac{\log(1 - c_{th})}{\log\left(\frac{t_{th}}{t_z}\right)}.$$

15 # n': degree of decay curve with uncertainty integrated, Ψ: decay control factor.
$$n' = \frac{\log\left(1 - t_z^{-n}\left(\lambda_1^2 + \lambda_2^2\right)^{-\frac{n}{2}}\left[t_z^n\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{n}{2}} - \left\{t_{th}\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{1}{2}} - \Psi\right\}^n\right]\right)}{\log\left(t_{th}\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{1}{2}} - \Psi\right) - \log\left(t_z\left(\lambda_1^2 + \lambda_2^2\right)^{\frac{1}{2}}\right)}$$

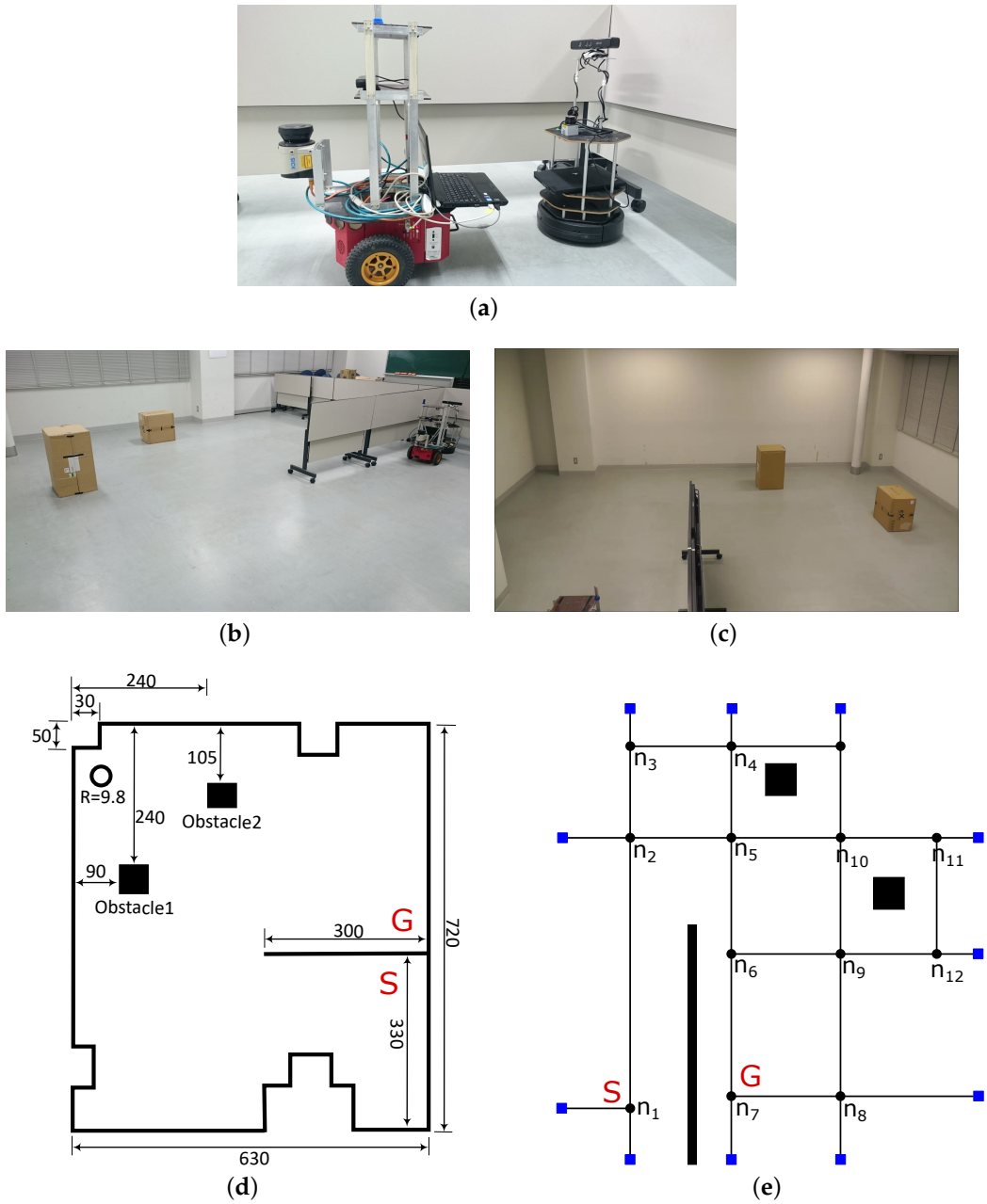


Figure 9. Experiment setup. (**a**) Differential drive robots Kobuki-Turtlebot2 and Pioneer-P3Dx. (**b**) Environment with initial position of robots. (**c**) Another view of the environment. (**d**) Environment dimensions. (**e**) Node-map of the environment where S and G are the start and goal points.

6.1. Experiments with Heterogeneous Maps

In this section, we describe the results of the proposed method with heterogeneous maps. The environment for experiments is shown in Figure 9b,c. The dimensions of the environment are shown in Figure 9d. The environment had two static obstacles 'Obstacle1' and 'Obstacle2' marked in Figure 9d. The start and goal positions are marked as 'S' and 'G', respectively, in Figure 9d. The T-node map is shown in Figure 9e.

The two robots used in the experiment were both equipped with 2D Lidar and RGBD sensors. As shown in Figure 9a, the Pioneer P3DX robot was equipped with a Sick-Lidar of 10 m range and ASUS Xtion-Pro RGBD camera. Turtlebot was equipped with a Hokuyo Lidar of 20 m range and a Kinect RGBD camera.

To test the proposed method with heterogeneous maps, Pioneer P3DX robot was programmed to use only the RGBD sensor to build a 3D map, and navigate in the environment. Pioneer P3DX first started navigation from location 'S' to the goal location 'G' as shown in Figure 10a. A* algorithm [47] and SHP algorithm [48,49] were used for path planning and path smoothing, respectively. As soon as the P3DX robot started moving, a long new obstacle was placed in the environment as shown in Figure 10b, well outside the range of the sensor. As shown in Figure 10c, the person moves in front of the robot and blocks its way purposefully. The details of dynamic obstacle are discussed in the next Section 6.2. P3DX perceives the new obstacle and alters its path towards the goal as shown in Figure 10d–f, while also updating the map with the newly added obstacle. P3DX was programmed to come back to its initial position 'S' and the navigation in shown in Figure 10g,h.

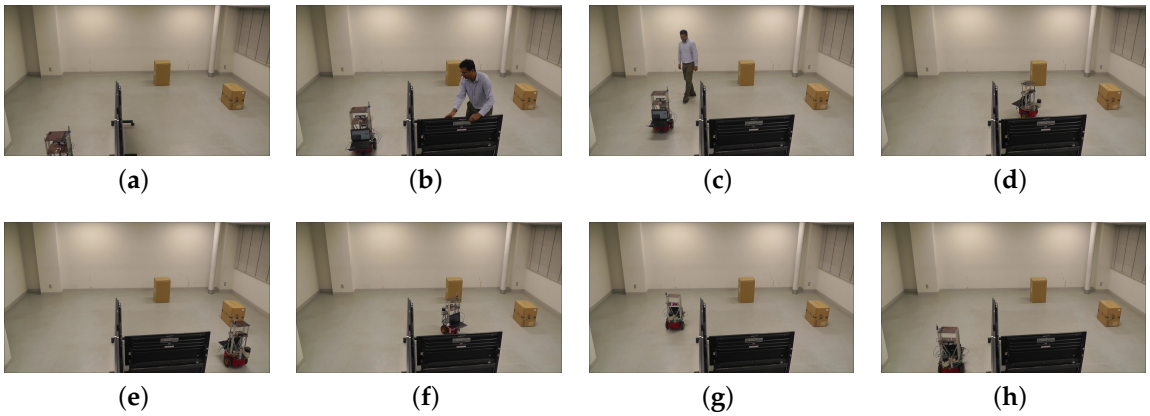


Figure 10. Timely snapshots of the experiment. (**a**) P3DX starts moving with old map. (**b**) Person adds a new obstacle. (**c**) Person moves in front of P3DX. (**d**,**e**) P3DX observes the new obstacle, changes trajectory and updates map. (**f**–**h**) P3DX return to the starting position. ***(Supplementary Materials)***

The updated 3D map build by P3DX robot is shown in Figure 11a. In this experiment, Turtlebot used only the 2D Lidar sensor with 2D gridmap. Hence, P3DX could not directly share the 3D point-cloud information due to the heterogeneous maps used by the two robots. By using the T-node map, P3DX blocked the path between nodes n_6 and n_7 and shared this information with the Turtlebot to plan appropriate path. More information regarding the dimensions of the new obstacle could also be shared for better path planning. Hence, the 3D information was converted to a 2D information to be shared with Turtlebot. Grid maps are the most commonly used 2D maps in which each grid-value represents whether the grid is occupied, free or unknown. The 3D point-cloud were projected to the ground which was detected using a RANSAC based plane detection [50]. This 2D information was shared by P3DX robot with Turtlebot and the updated T-node map is shown in Figure 11b. In the updated T-node map of Figure 11b, the obstacle is placed between the nodes n_6 and n_7 blocking it.

Turtlebot was programmed to navigate from the same start location 'S' to the goal location 'G'. In the absence of the proposed information sharing mechanism, the path planned by Turtlebot would be (Figure 11b),

$$S \rightarrow n_2 \rightarrow n_5 \rightarrow n_6 \rightarrow n_7$$

The Turtlebot would encounter a new obstacle between the nodes n_6 and n_7 and would have to re-plan a new path towards the goal. However, with the proposed information sharing mechanism, Turtlebot could directly plan a path considering the newly added obstacle and the planned path was (Figure 11b),

$$S \rightarrow n_2 \rightarrow n_5 \rightarrow n_6 \rightarrow n_9 \rightarrow n_8 \rightarrow n_7.$$

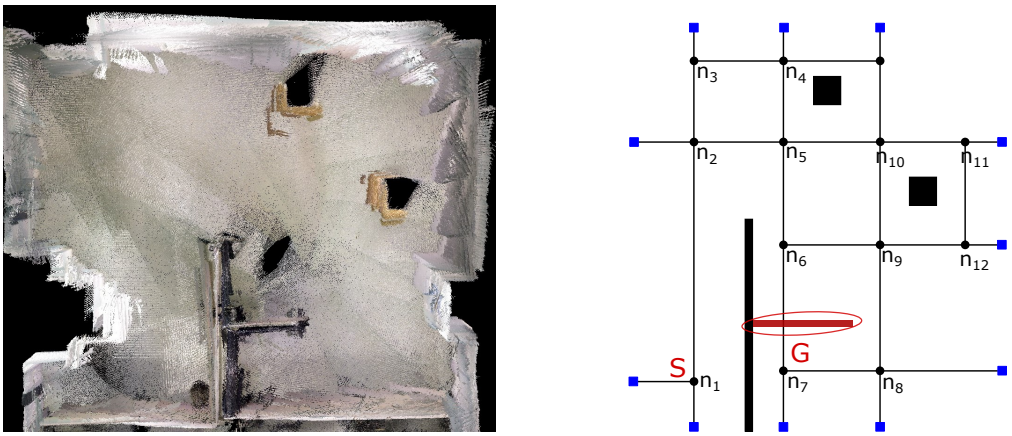


Figure 11. (**a**) RGBD map updated by P3DX. (**b**) Node-map. S and G are the start and goal points. The new obstacle is shown on nodes n_6n_7. Ellipse represents positional uncertainty.

Notice that, this appropriate path was generated by Turtlebot 'remotely' before actually encountering the new obstacle. Figure 12 shows the navigation of Turtlebot after considering the new obstacle. The entire navigation is illustrated between Figure 12a–h. In particular, it can be seen from Figure 12e–g, that Turtlebot maintains a safe threshold from the start itself. Turtlebot itself updated its map using the attached Lidar and the updated grid-map is shown in Figure 13.

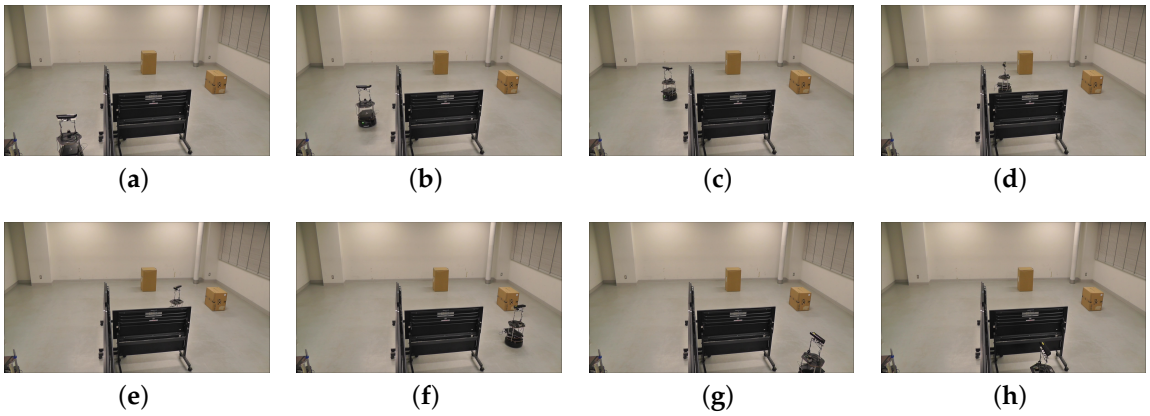


Figure 12. Timely snapshots of the experiment. (**a**–**h**) Turtlebot starts navigation with the updated information and plans a trajectory considering the new obstacle. (***Supplementary Materials***)

The dimensions of the obstacles in the experiment are given in Table 3.

The decay curve is shown in Figure 14. In the experiment, C_{th} was set to 0.45 and t_z to 20 min. Based on the uncertainty of the obstacle, the Ufactor was calculated as approximately 15% of t_z. Figure 14 shows the decay of confidence considering the uncertainty of the obstacles.

Figure 15 shows different decay curves for different amounts of estimated positional uncertainties of the new obstacle. Although Figure 14 shows the actual decay curve of the experiment, Figure 15 shows theoretical values for different values of uncertainty. Figure 15a–d shows the confidence decay with increasing uncertainty of 35%, 45%, 55% and 65%, respectively. It can be seen that, for increasing uncertainty, the curve decays much faster, as desired.

Thus, the T-node enables robots to share information across heterogeneous maps. Indeed, there is a need to transform the newly added obstacle's information to spatial coordinates but it can easily be achieved in real-time. Moreover, to avoid the overheads of such computation for time-critical

applications, only the blocked/un-blocked information could also be shared. Using the same approach, information among other type of maps could be shared effectively.

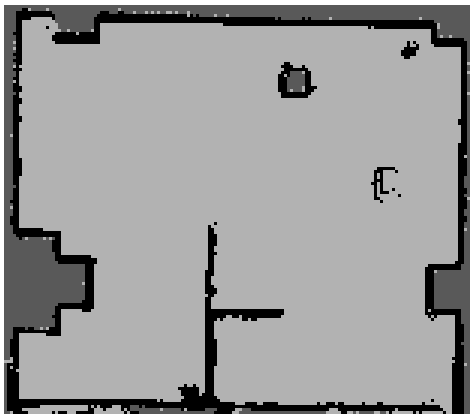


Figure 13. 2D gridmap updated by Turtlebot during navigation.

Table 3. Obstacle Dimensions in the Experiment.

Obstacle	Length × Width × Height
Obstacle1	40 cm × 40 cm × 68 cm
Obstacle2	50 cm × 35 cm × 50 cm
Newly Added Obstacle	300 cm × 5 cm × 100 cm

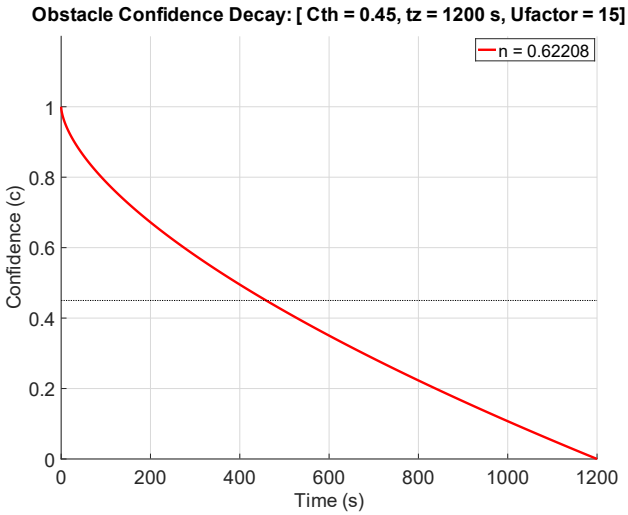


Figure 14. Confidence decay curve in the new experiment. C_{th} = 0.45, t_z = 1200 s, Ufactor = 15.

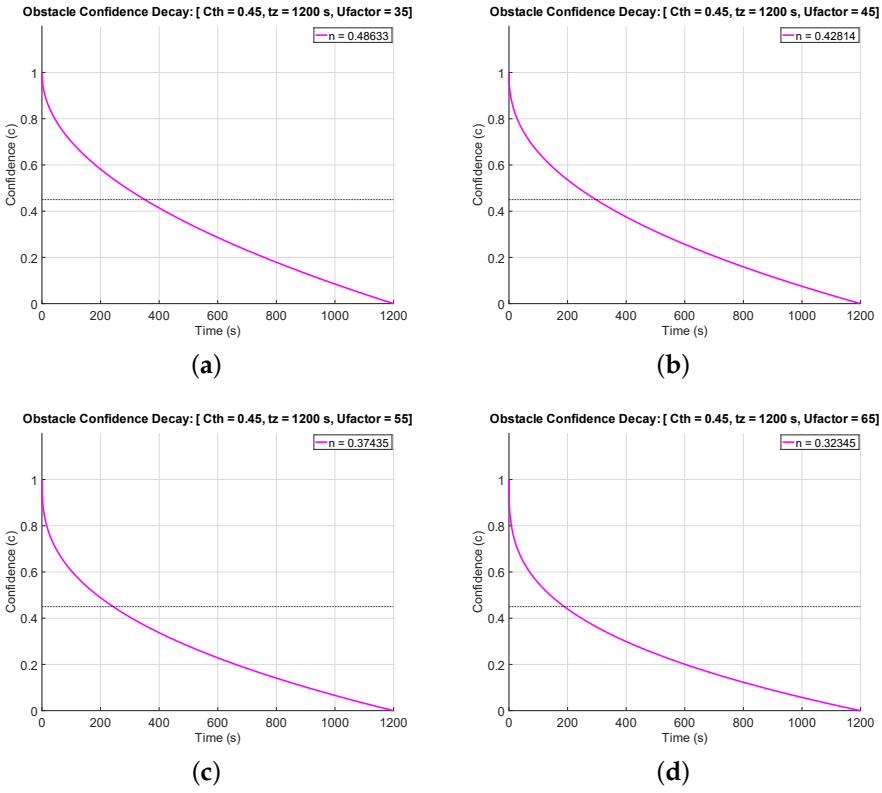


Figure 15. Estimation of confidence decay curve for different values of positional uncertainty with C_{th} = 0.45 and t_z = 1200 s. (**a**) Ufactor = 35%. (**b**) Ufactor = 45%. (**c**) Ufactor = 55%. (**d**) Ufactor = 65%. It can be seen that for higher uncertainties, the curve decays faster as desired.

6.2. Results with Dynamic Entities (Moving Obstacle)

The proposed method was tested under complex scenarios by purposefully moving a person in front of the robot and blocking its way. This is shown in Figure 16. As P3DX robot started navigation from start location 'S' to goal location 'G', a person blocked its way by randomly moving in front of the robot. This is shown in Figure 16a–j.

Similarly, the path of P3DX robot was blocked again while it was navigating back from the goal location 'G' to its start location 'S'. This is shown in Figure 17. The person randomly moved in front of the robot blocking its path as shown in Figure 17a–j.

In both the cases of Figures 16 and 17, the robot attempted to avoid collision and planned alternate trajectories or stopped if the person stands dangerously close to the robot. Moreover, in both the cases, the robot did not update the map corresponding to the person as a new obstacle in the map. This is because the positional uncertainty corresponding to the moving obstacle was large as calculated by Algorithm 2. Even if the person is falsely identified as a new obstacle and the map is updated, it has no adverse effects in the proposed method, as uncertainty is integrated in the confidence decay mechanism. Any wrong map update corresponding to dynamic obstacles has high probability of larger positional uncertainty corresponding to the dynamic obstacle and therefore a quicker decay given by Equation (6), (7) and (9). On the other hand, for static new obstacles in the map, the underlying SLAM (Algorithm 2) algorithm estimates smaller positional uncertainty and therefore a larger decay time, ensuring its permanence in the map.

Thus, uncertainty integration has two merits in the information sharing scheme. First, it acts a filter for wrong map updates corresponding to the dynamic obstacles in the environment through a quick confidence decay. Second, it ensures that only the correct information is shared with other robots corresponding to the new static obstacles. It should be noted that the dynamic detection of

moving people can be done using image processing for camera-based sensors [51], RGBD sensors [52] or leg detector for Lidar-based sensors [53] and integration of such approaches [54] will increase the robustness of the system.

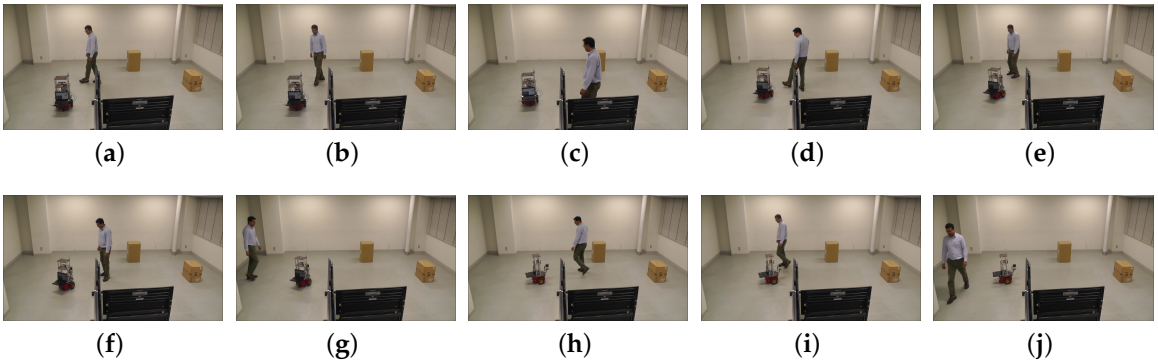


Figure 16. Dynamic obstacle experiment with P3DX navigation from position S to G in Figure 9. (**a**–**j**) Person moved randomly in front of P3DX for a long time moving in and out of the range of sensors. P3DX changed trajectories or stopped if the obstacle was dangerously close. (***Supplementary Materials***)

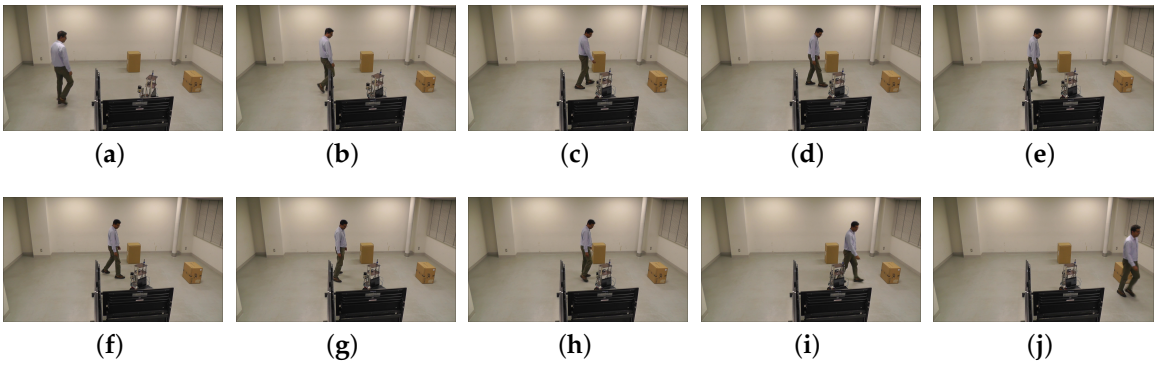


Figure 17. Dynamic obstacle experiment with P3DX navigation from position G to S in Figure 9. (**a**–**j**) To further test the method under pressure, a person moved randomly near P3DX. (***Supplementary Materials***)

7. Conclusions

Information sharing is a powerful technique which has many potential benefits in path planning of multi-robot systems. A node-map was proposed in our previous work to solve the problems of information sharing in different robots. We extended our previous work by integrating the positional uncertainty of the new obstacles in the confidence decay mechanism which models the transience of the obstacles. This minimizes false map updates and notifications in the system. New experiments were performed to share information about new obstacles in heterogeneous maps. The results shown that using the nodemap allows the robots to smoothly share the information. Moreover, since path planning is also done using the nodemap, efficient trajectories considering the position of new obstacles can be done in real-time. The information sharing mechanism allows the robots to obtain timely information about remote obstacles in the map without having to explicitly visit those areas. In addition, new experiments were performed to test the proposed mechanism in complex environments with moving people in the vicinity of the robots. Due to the tight coupling of uncertainty and decay mechanism, the dynamic obstacles could be filtered and avoided false update of the map. Even if there is some false update, the confidence corresponding to them decays fast due to larger uncertainty. Experiment results confirm that, in the long run in large environments employing multiple robots, the proposed method can improve the efficiency of the system in terms of shorter distance traveled by the robots and shorter planning time by eliminating path re-planning. In future, we will continue to test the

robustness of the proposed method in more complex and realistic environments such as cafeterias and offices.

Supplementary Materials: The supplementary materials are available online at http://www.mdpi.com/2076-3417/9/13/2753/s1.

Author Contributions: A.R. and A.A.R. conceived the idea, designed, performed experiments, and summarized the research; Y.K. made valuable suggestions to analyze the data and improve the manuscript. Y.H. provided important feedback to improve the manuscript. The manuscript was written by A.R.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ravankar, A.; Ravankar, A.A.; Hoshino, Y.; Emaru, T.; Kobayashi, Y. On a Hopping-points SVD and Hough Transform Based Line Detection Algorithm for Robot Localization and Mapping. *Int. J. Adv. Robot. Syst.* **2016**, *13*, 98. [CrossRef]
2. Wang, R.; Veloso, M.; Seshan, S. Multi-robot information sharing for complementing limited perception: A case study of moving ball interception. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 1884–1889. [CrossRef]
3. Riddle, D.R.; Murphy, R.R.; Burke, J.L. Robot-assisted medical reachback: using shared visual information. In Proceedings of the ROMAN 2005, IEEE International Workshop on Robot and Human Interactive Communication, Nashville, TN, USA, 13–15 Augest 2005; pp. 635–642. [CrossRef]
4. Cai, A.; Fukuda, T.; Arai, F. Cooperation of multiple robots in cellular robotic system based on information sharing. In Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Tokyo, Japan, 20–20 June 1997; p. 20. [CrossRef]
5. Rokunuzzaman, M.; Umeda, T.; Sekiyama, K.; Fukuda, T. A Region of Interest (ROI) Sharing Protocol for Multirobot Cooperation With Distributed Sensing Based on Semantic Stability. *IEEE Trans. Syst. Man Cybern. Syst.* **2014**, *44*, 457–467. [CrossRef]
6. Samejima, S.; Sekiyama, K. Multi-robot visual support system by adaptive ROI selection based on gestalt perception. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 3471–3476. [CrossRef]
7. Özkucur, N.E.; Kurt, B.; Akın, H.L. A Collaborative Multi-robot Localization Method without Robot Identification. In *RoboCup 2008: Robot Soccer World Cup XII*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 189–199. [CrossRef]
8. Sukop, M.; Hajduk, M.; Jánoš, R. Strategic behavior of the group of mobile robots for robosoccer (category Mirosot). In Proceedings of the 2014 23rd International Conference on Robotics in Alpe-Adria-Danube Region (RAAD), Smolenice, Slovakia, 3–5 September 2014; pp. 1–5. [CrossRef]
9. Ravankar, A.; Ravankar, A.A.; Kobayashi, Y.; Emaru, T. Avoiding blind leading the blind. *Int. J. Adv. Robot. Syst.* **2016**, *13*. [CrossRef]
10. Ravankar, A.; Ravankar, A.A.; Kobayashi, Y.; Emaru, T. On a bio-inspired hybrid pheromone signalling for efficient map exploration of multiple mobile service robots. *Artif. Life Robot.* **2016**, 221–231. [CrossRef]
11. Ravankar, A.; Ravankar, A.A.; Kobayashi, Y.; Emaru, T. Intelligent Robot Guidance in Fixed External Camera Network for Navigation in Crowded and Narrow Passages. *Proceedings* **2017**, *1*, 37. [CrossRef]
12. Ravankar, A.; Ravankar, A.; Kobayashi, Y.; Emaru, T. Symbiotic Navigation in Multi-Robot Systems with Remote Obstacle Knowledge Sharing. *Sensors* **2017**, *17*, 1581. [CrossRef] [PubMed]
13. Hunziker, D.; Gajamohan, M.; Waibel, M.; D'Andrea, R. Rapyuta: The RoboEarth Cloud Engine. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 438–444. [CrossRef]
14. Waibel, M.; Beetz, M.; Civera, J.; D'Andrea, R.; Elfring, J.; Gálvez-López, D.; Häussermann, K.; Janssen, R.; Montiel, J.M.M.; Perzylo, A.; et al. RoboEarth. *IEEE Robot. Autom. Mag.* **2011**, *18*, 69–82. [CrossRef]
15. Tenorth, M.; Perzylo, A.C.; Lafrenz, R.; Beetz, M. Representation and Exchange of Knowledge About Actions, Objects, and Environments in the RoboEarth Framework. *IEEE Trans. Autom. Sci. Eng.* **2013**, *10*, 643–651. [CrossRef]

16. Stenzel, J.; Luensch, D. Concept of decentralized cooperative path conflict resolution for heterogeneous mobile robots. In Proceedings of the 2016 IEEE International Conference on Automation Science and Engineering (CASE), Fort Worth, TX, USA, 21–25 Augest 2016; pp. 715–720. [CrossRef]
17. Ravankar, A.; Ravankar, A.A.; Kobayashi, Y.; Jixin, L.; Emaru, T.; Hoshino, Y. An intelligent docking station manager for multiple mobile service robots. In Proceedings of the Control, Automation and Systems (ICCAS), 2015 15th International Conference on, Busan, Korea, 13–16 October 2015; pp. 72–78. [CrossRef]
18. Ravankar, A.; Ravankar, A.; Kobayashi, Y.; Emaru, T. Hitchhiking Robots: A Collaborative Approach for Efficient Multi-Robot Navigation in Indoor Environments. *Sensors* **2017**, *17*, 1878. [CrossRef] [PubMed]
19. Ravankar, A.; Ravankar, A.; Kobayashi, Y.; Hoshino, Y.; Peng, C.C.; Watanabe, M. Hitchhiking Based Symbiotic Multi-Robot Navigation in Sensor Networks. *Robotics* **2018**, *7*, 37. [CrossRef]
20. Guo, Y.; Parker, L. A distributed and optimal motion planning approach for multiple mobile robots. In Proceedings of the 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292), Washington, DC, USA, 11–15 May 2002; Volume 3, pp. 2612–2619. [CrossRef]
21. Svestka, P.; Overmars, M.H. *Coordinated Path Planning for Multiple Robots*; Technical Report UU-CS-1996-43; Department of Information and Computing Sciences, Utrecht University: Utrecht, The Netherlands, 1996.
22. Clark, C.M.; Rock, S.M.; Latombe, J. Motion planning for multiple mobile robots using dynamic networks. In Proceedings of the 2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422), Taipei, Taiwan, 14–19 September 2003; Volume 3, pp. 4222–4227. [CrossRef]
23. Teng, R.; Yano, K.; Kumagai, T. Efficient Acquisition of Map Information Using Local Data Sharing over Hierarchical Wireless Network for Service Robots. In Proceedings of the 2018 Asia-Pacific Microwave Conference (APMC), Kyoto, Japan, 6–9 November 2018; pp. 896–898. [CrossRef]
24. Ravankar, A.A.; Ravankar, A.; Peng, C.; Kobayashi, Y.; Emaru, T. Task coordination for multiple mobile robots considering semantic and topological information. In Proceedings of the 2018 IEEE International Conference on Applied System Invention (ICASI), Chiba, Japan, 13–17 April 2018; pp. 1088–1091. [CrossRef]
25. Pinkam, N.; Bonnet, F.; Chong, N.Y. Robot collaboration in warehouse. In Proceedings of the 2016 16th International Conference on Control, Automation and Systems (ICCAS), Gyeongju, Korea, 16–19 October 2016; pp. 269–272. [CrossRef]
26. Regev, T.; Indelman, V. Multi-robot decentralized belief space planning in unknown environments via efficient re-evaluation of impacted paths. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 5591–5598. [CrossRef]
27. Gasparetto, A.; Boscariol, P.; Lanzutti, A.; Vidoni, R. Path Planning and Trajectory Planning Algorithms: A General Overview. In *Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*; Springer International Publishing: Cham, Switzeland, 2015; pp. 3–27. [CrossRef]
28. Tang, S.H.; Kamil, F.; Khaksar, W.; Zulkifli, N.; Ahmad, S.A. Robotic motion planning in unknown dynamic environments: Existing approaches and challenges. In Proceedings of the 2015 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS), Langkawi, Malaysia, 18–20 October 2015; pp. 288–294. [CrossRef]
29. Ravankar, A.; Ravankar, A.; Kobayashi, Y.; Hoshino, Y.; Peng, C.C. Path Smoothing Techniques in Robot Navigation: State-of-the-Art, Current and Future Challenges. *Sensors* **2018**, *18*, 3170. [CrossRef] [PubMed]
30. Montijano, E.; Aragues, R.; Sagüés, C. Distributed Data Association in Robotic Networks With Cameras and Limited Communications. *IEEE Trans. Robot.* **2013**, *29*, 1408–1423. [CrossRef]
31. Ravankar, A. Probabilistic Approaches and Algorithms for Indoor Robot Mapping in Structured Environments. Ph.D. Thesis, Hokkaido University, Sapporo, Japan, 2015.
32. Ravankar, A.; Kobayashi, Y.; Ravankar, A.; Emaru, T. A connected component labeling algorithm for sparse Lidar data segmentation. In Proceedings of the 2015 6th International Conference on Automation, Robotics and Applications (ICARA), Queenstown, New Zealand, 17–19 Feburary 2015; pp. 437–442. [CrossRef]
33. Ravankar, A.; Ravankar, A.A.; Kobayashi, Y.; Jixin, L.; Emaru, T.; Hoshino, Y. A novel vision based adaptive transmission power control algorithm for energy efficiency in wireless sensor networks employing mobile robots. In Proceedings of the 2015 Seventh International Conference on Ubiquitous and Future Networks, Sapporo, Japan, 7–10 July 2015; pp. 300–305. [CrossRef]
34. Zhang, T.Y.; Suen, C.Y. A Fast Parallel Algorithm for Thinning Digital Patterns. *Commun. ACM* **1984**, *27*, 236–239. [CrossRef]

35. Yang, D.H.; Hong, S.K. A roadmap construction algorithm for mobile robot path planning using skeleton maps. *Adv. Robot.* **2007**, *21*, 51–63. [CrossRef]
36. Bai, X.; Latecki, L.; Yu Liu, W. Skeleton Pruning by Contour Partitioning with Discrete Curve Evolution. *Pattern Anal. Mach. Intell. IEEE Trans.* **2007**, *29*, 449–462. [CrossRef] [PubMed]
37. Ravankar, A.A.; Hoshino, Y.; Ravankar, A.; Jixin, L.; Emaru, T.; Kobayashi, Y. Algorithms and a framework for indoor robot mapping in a noisy environment using clustering in spatial and Hough domains. *Int. J. Adv. Robot. Syst.* **2015**, *12*. [CrossRef]
38. MacQueen, J.B. Some Methods for Classification and Analysis of MultiVariate Observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*; Cam, L.M.L., Neyman, J., Eds.; University of California Press: Berkeley, CA, USA, 1967; Volume 1, pp. 281–297.
39. Lloyd, S.P. Least squares quantization in pcm. *IEEE Trans. Inf. Theory* **1982**, *28*, 129–137. [CrossRef]
40. Jain, A.K.; Dubes, R.C. *Algorithms for Clustering Data*; Prentice-Hall, Inc.: Upper Saddle River, NJ, USA, 1988.
41. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*; AAAI Press: Menlo Park, CA, USA, 1996; pp. 226–231.
42. Fox, D.; Ko, J.; Konolige, K.; Limketkai, B.; Schulz, D.; Stewart, B. Distributed Multirobot Exploration and Mapping. *Proc. IEEE* **2006**, *94*, 1325–1339. [CrossRef]
43. Ravankar, A.A.; Ravankar, A.; Emaru, T.; Kobayashi, Y. A hybrid topological mapping and navigation method for large area robot mapping. In Proceedings of the 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Kanazawa, Japan, 19–22 September 2017; pp. 1104–1107. [CrossRef]
44. Pioneer P3-DX. Pioneer P3-DX Robot. 2018. Available online: https://www.robotshop.com/community/robots/show/pioneer-d3-px (accessed on 11 Janurary 2019).
45. TurtleBot 2. TurtleBot 2 Robot. 2018. Available online: http://turtlebot.com/ (accessed on 11 Janurary 2019).
46. Quigley, M.; Conley, K.; Gerkey, B.P.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An Open-Source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 12–15 May 2009.
47. Hart, P.; Nilsson, N.; Raphael, B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *Syst. Sci. Cybern. IEEE Trans.* **1968**, *4*, 100–107. [CrossRef]
48. Ravankar, A.; Ravankar, A.A.; Kobayashi, Y.; Emaru, T. Path smoothing extension for various robot path planners. In Proceedings of the 2016 16th International Conference on Control, Automation and Systems (ICCAS), Gyeongju, Korea, 16–19 October 2016; pp. 263–268. [CrossRef]
49. Ravankar, A.; Ravankar, A.A.; Kobayashi, Y.; Emaru, T. SHP: Smooth Hypocycloidal Paths with Collision-Free and Decoupled Multi-Robot Path Planning. *Int. J. Adv. Robot. Syst.* **2016**, *13*, 133. [CrossRef]
50. Xu, B.; Jiang, W.; Shan, J.; Zhang, J.; Li, L. Investigation on the Weighted RANSAC Approaches for Building Roof Plane Segmentation from LiDAR Point Clouds. *Remote Sens.* **2015**, *8*, 5. [CrossRef]
51. Enzweiler, M.; Gavrila, D.M. Monocular Pedestrian Detection: Survey and Experiments. *IEEE Trans. Pattern Anal. Mach. Intell.* **2009**, *31*, 2179–2195. [CrossRef] [PubMed]
52. Spinello, L.; Arras, K.O. People detection in RGB-D data. In Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, USA, 25–30 September 2011; pp. 3838–3843. [CrossRef]
53. Pantofaru C. Leg Detector. 2019. Available online: http://wiki.ros.org/leg_detector (accessed on 3 May 2019).
54. Moeslund, T.B.; Granum, E. A Survey of Computer Vision-Based Human Motion Capture. *Comput. Vis. Image Understand.* **2001**, *81*, 231–268. [CrossRef]