

METHOD AND SYSTEM FOR RETURNING A DISPLACED AUTONOMOUS MOBILE ROBOT TO ITS NAVIGATIONAL PATH

TECHNICAL FIELD

- 5 The present disclosure relates to mobile robot localization and navigation, and more particularly to returning a displaced autonomous mobile robot to its navigational path.

BACKGROUND

- 10 For mobile robots, spatial localization is a fundamental competency that an autonomous robot should possess since information on the robot's location with respect to its environment is needed as input for making decisions on further actions.
- 15 Existing work introduced an adaptive Monte Carlo localization approach (AMCL) for mobile robot navigation and localization using a particle filter. Given an internal map of its environment, the algorithm estimates the position and orientation (collectively referred to as pose) of the robot in the internal map as it tracks its movement in the environment. As the robot moves, its sensors
- 20 provide updated information about its surrounding, and the algorithm assigns and updates the weight to each particle/point in the configuration space according to a likelihood of the robot occupying the space. The particle filter therefore represents a probability distribution of the robot's likely location in the configuration space. The AMCL is robust to noises collected by the sensors due

to noisy environments since it is able to compensate for accumulated sensor errors, e.g. errors in the raw odometry data during navigation. Traditional probabilistic mapping algorithms are therefore used to build the internal maps with input from perception sensors, such as a laser range finder. Coupled with
5 an advanced localization module such as the AMCL, robot navigation and localization can be achieved deliberately.

However, sometimes a robot may unexpectedly fail to track itself in its environment during navigation. This can happen as a result of abrupt wheel
10 slippage resulting in significant errors in the odometry input, or when the robot unexpectedly collides with an obstacle, or when the robot's sensors are blocked, or when the robot is physically moved to another place (or kidnapped). The well-known robot kidnapping problem is a special global localization issue in mobile robotics. Since a kidnapped robot will fail to track its current position due
15 to a gap in the information collected by its sensors, this will result in invalid path planning and therefore, the robot will be unable to perform unfinished navigation tasks. It is therefore important for mobile robots to have the ability to re-localize itself after it unexpectedly loses tracking its pose relative to its surroundings due to abrupt wheel slippage, unsteady movements on uneven floors, collision with
20 obstacles, sensor blocking or kidnapping.

Existing work on re-localization does not provide a satisfactory solution. For example, despite AMCL's ability for robot navigation and localization, it is not able to effectively re-localize or return to its original navigational path (self-

recover) within a pre-built global map because the algorithm depends on the pre-built global map which is not effective for a robot that is lost or displaced.

A further existing work on localization failure detection adopts a semi-globalization approach for failure recovery. However, the approach is only suitable for abrupt wheel slippages due to uneven flooring as the algorithm requires that the robot's pose does not change significantly. In addition, the approach also requires manual operation for the robot to recover its pose.

Machine learning techniques have been applied to match the current laser scan to an offline database in an attempt to re-establish a robot's position in the internal map. However, this only works if the lost robot is in a known environment i.e. within the pre-built global map. The work also does not address the system awareness of when the robot is kidnapped.

15

Existing works require that the robot is lost within a known environment and thus, this has limited applications. The lost recovery problem then becomes just a localization problem.

Therefore, it is desirable to provide ways for a displaced autonomous mobile robot to return to its planned navigational path in order to address the problems mentioned in existing prior art and/or to provide the public with a useful choice.

SUMMARY

According to a first aspect, there is provided a method of returning a displaced autonomous mobile robot to its planned navigational path, the mobile robot having a sensor and a camera the method comprising, ,before autonomous
5 deployment of the robot, (i) constructing an occupancy grid map based on data received from the sensor of the robot's vicinity along the planned navigational path, the collected data including robot poses along the planned navigational path; and (ii) constructing a cognitive map of the planned navigational path based on a set of visual cues captured by the camera of the robot's vicinity
10 along the planned navigational path, the visual cues being associated with the robot poses in the occupancy grid map; and, after autonomous deployment of the robot, (iii) upon detection that the robot is displaced, constructing a local occupancy grid map of a vicinity of the displaced robot; (iv) selecting a destination point in the local occupancy grid map; (v) navigating the mobile
15 robot to the destination point along a designated path; (vi) as the mobile robot travels along the designated path, searching for a location that corresponds to a visual cue of the visual cue set defined in the cognitive map; and (vii) upon finding the location, recalling the robot pose associated with the visual cue to return the robot to the planned navigational path.

20

Advantageously, as discussed in the described embodiment, the robot can explore its vicinity (whether within or outside a global grid map associated with the planned navigation path) so that the displaced robot has the ability to

rediscover the planned navigational path by identifying visual cues in its vicinity which is associated with locations on the navigational path.

Preferably, the method may further comprise, when the mobile robot is travelling along its planned navigation path, detecting if the robot is displaced by determining whether or not there is misalignment between the robot's actual location and the planned navigation path as defined by the occupancy grid map. In one specific example, the robot may be detected as being displaced only when the misalignment persists for at least 2 seconds.

10

Preferably, the local occupancy grid map may be constructed when the mobile robot is stationary. The method may also comprise rotating the robot 360° to construct the local occupancy grid map. In a specific exemplary implementation, the destination point may be randomly selected from the local occupancy grid map.

15

Preferably, the method may further comprise collecting further information of the robot's vicinity while the robot navigates to the destination point in step (iii); and updating the local occupancy grid map with the further information.

20

If no location is found which corresponds to the visual cue, the method may then repeat steps (iii) to (v) until such a location is identified. The method may also comprise, wherein recalling the robot pose in step (vi) comprises replacing

the local occupancy grid map with the occupancy grid map. In that case, the method may further comprise performing a 360° rotation of the robot to converge the robot's laser with the occupancy grid map.

- 5 According to a second aspect, there is provided a method of returning a displaced autonomous mobile robot to its planned navigational path based on a cognitive-occupancy grid map, the cognitive-occupancy grid map having a set of visual cues associated with locations along the planned navigational path, the method comprising (i) upon detection that the robot is displaced, constructing a
- 10 local occupancy grid map of a vicinity of the displaced robot; (ii) selecting a destination point in the local occupancy grid map; (iii) navigating the mobile robot to the destination point along a designated path; (iv) as the mobile robot travels along the designated path, searching for a location that corresponds to a visual cue of the visual cue set defined in the cognitive-occupancy grid map;
- 15 and (v) upon finding the location, recalling a pose of the robot associated with the visual cue to return the robot to the planned navigational path.

The methods of the first and second aspects may be performed by the mobile robot.

20

According to a third aspect, there is provided a system for returning a displaced autonomous mobile robot to its planned navigational path, the mobile robot having a sensor and a camera, the system comprising a global map builder configured to construct an occupancy grid map based on data received from the

sensor of the robot's vicinity along the planned navigational path, the collected data including robot poses along the planned navigational path, and construct a cognitive map of the planned navigational path based on a set of visual cues captured by the camera of the robot's vicinity along the planned navigational path, the visual cues being associated with the robot poses in the occupancy grid map, before autonomous deployment of the robot; a local map builder configured to construct a local occupancy grid map of a vicinity of the displaced robot, after autonomous deployment of the robot and upon detection that the robot is displaced; a path planner configured to select a destination point in the local occupancy grid map; a navigator configured to navigate the mobile robot to the destination point along a designated path; a place recognition module configured to identify a location along the designated path that corresponds to a visual cue of the visual cue set defined in the cognitive map; and a recovery module configured to recall the robot pose associated with the visual cue to return the robot to the planned navigational path.

Preferably, the system may include a lost detection module configured to detect if the robot is displaced by determining whether or not there is misalignment between the robot's actual location and the planned navigation path as defined by the occupancy grid map stored.

Preferably, the recovery module may further comprise a pose-reinitialization module configured to replace the local occupancy grid map with the occupancy grid map.

- 5 The system may be implemented in various ways and in one example, this may be by way of an autonomous mobile robot comprising the system as discussed above.

It should be apparent that features relating to one aspect may be relevant to the
10 other aspects.

BRIEF DESCRIPTION OF THE DRAWINGS

An exemplary embodiment will be described with reference to the accompanying drawings in which:

- 15 Figure 1a shows an exemplary mobile robot according to a preferred embodiment;

Figure 1b shows the mobile robot of Figure 1 with its casing removed to show its internal components;

- Figure 2 is a block diagram of a system architecture of the mobile robot
20 of Figure 1;

Figure 3 is a flow diagram illustrating actions performed by the mobile robot of Figure 1 to self-recover when the mobile robot is lost;

Figures 4a – 4d are grid images showing different laser alignments between the robot's laser scans and the occupancy grid map constructed during a map pre-building stage of the system architecture of Figure 2;

Figure 5 is a flow diagram of an exemplary method for returning a lost
5 robot to its navigational path performed by the robot of Figure 1;

Figures 6a – 6d are grid images of a local occupancy grid map with valid and invalid destination points selected during the exemplary method of Figure 5;

Figure 7 illustrates a cognitive-occupancy grid map constructed based on the flow diagram of Figure 3 and used during evaluation of the robot;

10 Figures 8a – 8c illustrate a scenario of how the robot of Figure 1 performs lost recovery after the robot is kidnapped;

DETAILED DESCRIPTION

The following description contains specific examples for illustrative purposes.

15 The person skilled in the art would appreciate that variations and alterations to the specific examples are possible and within the scope of the present disclosure. The figures and the following description of the particular embodiments should not take away from the generality of the preceding summary.

20

Figure 1a illustrates a mobile robot 100 which can self-recover when the mobile robot 100 is lost or displaced according to a preferred embodiment. The mobile robot 100 includes an outer casing 110 and Figure 1b shows the mobile robot

100 with the outer casing omitted to show a skeleton 120. As can be seen from the skeleton 120, the mobile robot 100 includes a robot base 121 having wheels 122 equipped with wheel encoders 123 (not shown in figure 1b, but see Figure 2) for providing raw odometry data from the wheels 122 when the mobile robot

5 100 travels. In this embodiment, the mobile robot 100 includes a camera in the form of an RGB-D sensor 124, and a laser sensor 125 which operates as a high resolution laser range finder. The skeleton 124 includes an elongate frame 126 supported by the robot base 121 and the elongate frame 126 extends upwards from the robot base 121. The RGB-D sensor 24 is mounted to and near the top

10 of the elongate frame 126 around 90cm from the ground, and the laser sensor 125 is mounted above the robot base 121 around 30cm from the ground. The wheel encoders 123, the RGB-D sensor 124 and the laser sensor 125 cooperate together to provide input of the robot's vicinity or surrounding to enable robot localization and navigation. The mobile robot 100 may also include

15 other perception sensors.

Figure 2 is a schematic block diagram of a system architecture of the mobile robot 100. The mobile robot 100 includes a processor (or controller) 127 arranged to be communicatively coupled to the wheels encoder 123, the RGB-D sensor 124, the laser sensor 125 and also to control movement of the wheels

20 122. The system architecture also includes a global map builder 201 for constructing a cognitive map 211 and an occupancy map 212 based on the output of the RGB-D sensor 124 and the laser sensor 125 respectively. For easy of reference, the term 'cognitive-occupancy grid map' or 'global map' is

used to refer to both the cognitive map 211 and the occupancy grid map 212 collectively. The cognitive-occupancy grid map 213 is stored in computer readable memory. The processor 127 is specifically configured to execute instructions of various modules stored in computer readable memory and in particular, a lost detection module 128 and a self-exploration module 150. The self-exploration module 150 includes sub-modules and these comprise a map builder 151, a path planner 152, a navigator 153, a place recognition module 154 and a recovery module 155. The operation of each of these modules will be explained next in relation to Figure 3, which illustrates a flow diagram having two stages to explain how the mobile robot 100 recovers itself when the mobile robot 100 is lost.

As illustrated in Figure 3, a method 200 of recovering the mobile robot 100, when it is lost or displaced, includes two stages according to this embodiment and these comprises an offline global map building stage 210, and an online navigation and lost recovery stage 220.

During the offline global map building stage 210, a remote controller is used to control the mobile robot 100 manually to explore the environment along a designated route or navigational path from a starting point to a destination or target so as to collect training data. The RGB-D sensor/camera 124 captures images of the environment or the robot's vicinity as the robot 100 moves along the navigational path. At step 221, the captured images are then used by the processor 127 to construct/build the cognitive map 211. Visual cues are

acquired, learned and stored in the cognitive map 211 during the map building process. The visual cues of the cognitive map are associated with actual poses in the occupancy grid map of the robot 100 using a continuous attractor network and in this way, the cognitive map 211 is thus associated with or correlated to
5 the occupancy grid map 212.

Simultaneously with step 221, during the movement, the laser sensor 122 also collects data of the environment which the robot 100 has explored along the navigational path and the collected data is used to construct an occupancy grid
10 map 212 at step 222. In this embodiment, the occupancy grid map 212 is built using ROS gmapping package. Alternatively, other map building algorithms known to the skilled person may be used to construct the occupancy grid map 212. At 223, the cognitive-occupancy grid map 213 is thus formed which comprises the cognitive map 211 and the occupancy grid map 212.

15

Upon generation of the cognitive-occupancy grid map 213, the robot 100 is ready to be deployed for fully autonomous operation along the designated route or navigation path.

20 At the navigation and lost recovery stage 220, the robot 100 is deployed to navigate autonomously at step 224 along the navigation path from a starting point to a target location 226. The occupancy grid map 212 is used for localization and obstacle avoidance during the navigation of the robot 100 to the target location 226. The processor 127 also executes the lost detection module

128 at steps 225 and 227, almost immediately after deployment of the mobile robot 100, and the lost detection module 128 is configured to run continuously as the mobile robot 100 makes its way from the starting point to the target location 226 in order to detect whether or not the robot 100 is lost while
5 navigating along the navigation path. As long as the robot 100 is not lost, the robot 100 would continue to move along the navigation path towards its target location 226. Details of how the lost detection module 128 detects whether or not the robot 100 is lost will be described later.

10 When the robot 100 is detected to be lost, the self-exploration module 150 is then executed by the processor 127 to execute self-exploration steps 230.

Broadly, the self-exploration module 150 starts by constructing a local vicinity occupancy grid map at step 231 of the displaced robot's surrounding. This
15 enables the robot 100 to then explore its vicinity or surrounding in search for an ever-visited-place at step 232. Once the robot 100 finds an ever-visited place corresponding to a visual cue defined in the cognitive map 211 of the global map 213, an accurate pose associated with the visual cue is recalled from the cognitive map 211 and the occupancy grid map 212 is reloaded at step 233.

20

Thereafter, at step 234, the processor 127 switches to the recovery module 155 and executes pose re-initialization. After the robot 100 has returned to its original pose prior to being displaced, the robot 100 resends the target location

and the navigation module 153 guides the robot 100 to continue to its target location 226.

The actions performed by lost detection module 128 and lost recovery by the
5 self-exploration module 150 are discussed in further details below.

Lost detection

Three scenarios are considered for lost detection: (1) the robot 100 is kidnapped, (2) the laser sensor 125 is blocked, and (3) the robot 100 is pushed.
10 Although only three scenarios are discussed, it is noted that the lost detection module 128 may also be used in other lost robot scenarios resulting from unstable movements, abrupt wheel slippage etc.

Figures 4(a)-(d) illustrate grid images 300 with different laser alignments
15 between the robot's laser scans 301, 302, 303, 304 recorded at time t_i and the map boundary points 350 in the occupancy grid map 212.

The occupancy grid map 212 may be accurately learned from high resolution and long range laser using the Rao-Blackwellized particle filter and thus, an
20 accurate occupancy grid map 212 may be obtained too. During the navigation, if the robot 100 is well localized, the laser scan 301 would be properly aligned with the map boundary 350 of the occupancy grid map 212 as shown in Figure 4a.

However, if the robot 100 is not well-localized, the laser scan 302,303,304 would not match the map boundary 350 of occupancy grid map 212 properly, as illustrated in Figures 4b to 4d. Figure 4b shows the scenario where the robot 100 is pushed which caused the laser scan 302 to be rotated. In Figure 4c, the robot 100 is kidnapped from a point A to a point B. However, in this instance, there is no input from the raw odometry data and therefore the robot 100 is unable to update its position in the map. As such, based on the laser scan 303 in Figure 4c, the robot's position appears to not have changed i.e. kept at point A. Finally, Figure 4d shows the scenario where the laser sensor 125 was blocked. Therefore, the laser scan 304 of Figure 4d is partially obscured.

A laser-map matching error function $d_{t_i}(L_{t_i}, B)$ is defined herein as

$$d_{t_i}(L_{t_i}, B) = \frac{\sum_{l_i \in L_{t_i}} \min \|l_i, B\|^2 + \sum_{b_j \in B} \min \|L_{t_i}, b_j\|^2}{|L_{t_i}| + |B|} \quad (1)$$

wherein $||$ and $\| \cdot \|^2$ denote the number of the point set and the Euclidean distance from a point to a set, respectively. L_{t_i} is the set of laser beam at current time t_i . B is the set of the boundary points of the occupancy grid map 212, and b_j is any boundary point. Eq. (1) intuitively determines whether the 2D scans from the laser ranger finder and the map are well aligned. A Boolean parameter t_i is defined to determine whether the laser beams from the laser sensor 125 at the current time t_i are well aligned with the occupancy grid map 212:

$$r_{t_i} = \begin{cases} 1 & \text{if } d_{t_i} > \delta \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

If the value of $\sum_{t_i} r_{t_i} \geq N_{lost}$ within a duration of 2~3 seconds, the robot 100 is
 5 identified to be lost. The thresholds δ and N_{lost} are required to be well identified
 depending on different environments. For example, in the present embodiment,
 δ is 1.5 m and N_{lost} is 99.

Lost recovery

10 The lost recovery problem formulation as addressed by the self-exploration
 module 150 is given below.

Given an initial configuration space, X , an initial obstacle space, X_{obs} , a
 destination map M_{goal} , a starting point q_{start} in a vicinity of the destination map,
 15 a kinematic constraint, K_{con} , and a possible cost function $c(\cdot)$. Find a (or an
 optimal) path from the starting point q_{start} to an ever-visited-place which is
 defined by the destination map.

The initial configuration and obstacle spaces are built up by the robot 100
 20 without any movement or with safety movement such as 360 degrees rotation
 i.e. the robot 100 is stationary. Subsequently, the local vicinity occupancy grid
 map can be built up at step 231 by the local map builder 151 of the self-
 exploration module 150 and such a map may also be called an on-line local

map.

The problem formulation on path planning addressed by the path planner 152 of the self-exploration module 150 is given as below.

5

Given an initial configuration space, X , an obstacle space, X_{obs} , a vicinity starting point q_{start} , a vicinity destination point g_{goal} (or a destination region), a kinematic constraint in the map, K_{con} , and a possible cost function $c(\cdot)$. Find a (or an optimal) path from the vicinity starting point q_{start} to the vicinity destination point g_{goal} .

10

There are two differences between the problem formulation on lost recovery and the problem formulation on path planning. They are as follows:

- (1) Both the vicinity starting and destination points are exactly known in the path planning. However, only the starting point is known in the lost recovery;
- (2) Both the full configuration and obstacle spaces are available in the path planning while only initial configuration and obstacle spaces are available in the lost recovery.

20

If a vicinity destination point is randomly selected in the local vicinity occupancy grid map, then both the starting and destination points are available in the lost recovery problem. A path is then set up using the initial configuration and

obstacle spaces for the robot 100 to explore the unknown environment. The local vicinity occupancy grid map is updated as the robot 100 navigates from the vicinity starting point to the vicinity destination point such that there are more valid candidates to be selected in the subsequent rounds. In addition, it is
5 unknown whether the randomly selected vicinity destination point is an ever-visited-place. It is thus also necessary to perform place recognition using the destination map M_{goal} during the navigation.

As such, self-exploration can be extended from path planning by adding three
10 new components:

- (1) a new component to randomly select the destination point in the on-line local map (i.e. the local vicinity occupancy map);
- (2) a new component to update the on-line local map;
- (3) a new component to verify whether there is any point in the path to be an
15 ever-visited-place which is defined by the destination map.

When the robot 100 is lost, the robot 100 loses track of its pose in the global map i.e. occupancy grid map 212, and the occupancy grid map would not be effective for path planning, obstacle avoidance, and localization anymore.
20 However, similar to human being's spatial cognition and navigation ability, if the robot 100 can find an ever-visited place by self-exploring the environment, the robot 100 may retrieve the related visual cues that have been stored in the cognitive map 211. In other words, the robot 100 needs to find a path from its

current (lost) location to an ever-visited-place that is defined by the cognitive-occupancy grid map.

Figure 5 is a flow diagram of an exemplary method 400 to elaborate the self-exploration steps 230 further and as performed by the mobile robot 100 to return itself to its navigational path.

Upon detection that the robot 100 is displaced or lost as illustrated in Figure 3, the map builder 151 is executed by the processor 127 at step 231 to construct the local vicinity occupancy grid map of the displaced robot's surrounding or vicinity.

At step 231a, the path planner 152 selects a vicinity destination point within the local vicinity occupancy grid map. The path planner 152 includes a path planning algorithm to determine a designated path from the vicinity starting point to the vicinity destination point.

At step 231b, the navigator 153 navigates the robot 100 to the vicinity destination point along the designated path. During the navigation, the robot 100 collects more information in its vicinity. The local vicinity grid-occupancy map is updated with the collected information as the robot 100 travels along the designated path 'looking' for ever-visited places.

It is noted that the vicinity destination point has to be valid before the path

planner 152 can determine the designated path to the vicinity destination point and the navigator 153 can navigate the robot 100 to the vicinity destination point.

5 Figures 6a and 6b are images of the local vicinity occupancy grid map with examples of valid random poses 510, 520 of the robot 100 while figures 6c and 6d are images of the local vicinity occupancy grid map with examples of invalid random poses 530, 540. If the random pose 510,520 is valid, the robot 100 navigates to the randomly generated pose and simultaneously updates the local
10 vicinity occupancy map. Otherwise, another pose is randomly selected until a valid pose is generated.

In this embodiment, the search area σ is restricted in a bounded area 500, i.e., $6 \times 6m^2$ in the present embodiment. However, the search area can be enlarged
15 at the cost of more possible searching time. The Rao-Blackwellized particle filter is adopted to build a new occupancy grid map which has incorporated the dynamic window approach (DWA) for path planning and obstacle avoidance during the self-exploration. The purpose is to guarantee that the robot 100 would navigate to the target location 226 efficiently and safely.

20

Referring again to Figure 5, at step 232, the place recognition module 154 checks the 'live' images obtained via the RGB-D sensor 124 to survey the vicinity as the mobile robot 100 moves along the designated path to identify a

marker that corresponds to a visual cue defined in the cognitive-occupancy grid map 213.

If the marker is identified, the robot 100 relocates itself using the cognitive map 211. Otherwise, if the mobile robot 100 reaches the vicinity destination point without identifying the marker, a new destination point is generated in the updated local vicinity occupancy grid map. The robot 100 then repeats steps 231a, 231b and 232 until the robot 100 finds an ever-visited-place. It is assumed that every point in the local vicinity occupancy grid map has the same probability to be an ever-visited place. If this is not true, a better or smarter strategy can be adopted for step 231a to select the valid destination point. For example, a cost function can be defined for the lost recovery so as to find an optimal lost recovery solution.

Once an ever-visited place is found, the occupancy grid map 212 is reloaded at step 233. However, the previous obstacles recorded around the robot 100 during self-exploration are still in the robot's system. If the local vicinity occupancy grid map is not cleared, this may result in invalid paths found by global and local planners of the robot 100 and the robot 100 might collide with obstacles. Therefore, related obstacle information of the local vicinity occupancy grid map should be cleared by resetting global and local cost maps in the system

The global and local cost maps are used for obstacle avoidance during robot navigation. The global and local cost maps maintain information about where the robot 100 should navigate in the form of a grid map. The cost maps use both laser sensor data and information from the local occupancy grid map to
5 store and update information about obstacles in the dynamic office environment. Each cell in the cost maps can be either free, occupied, or unknown status. Each status has a cost value assigned to it upon projection into the cost maps.

At step 234, the recovery module 155 recalls a pose of the robot 100 that is
10 associated with the visual cue. Next, the system re-starts the navigation module 153. For pose re-initialization, the robot 100 rotates itself around 360 degree in order to converge the laser beams with the occupancy grid map 212. After the pose re-initialization process, the target goal would be re-sent again and the robot 100 would have self-recovered is able to continue with its unfinished tasks
15 or continue to the target location 226. It should be noted that during the pose recovery process, the laser scans do not need to match the maps exactly. Since the robot 100 re-starts navigation, it will be able to localize itself in the rest of the way by itself.

20 The Pseudo-code for the proposed lost recovery algorithm is provided below:

1. Initialization: $i \leftarrow 0$;
2. Initialize a local occupancy grid mapping using Rao-Blackwellized particle filter [5, 31], as shown in Fig. 4(a).
3. while $i < N$ do
 4. $q_{goal} \leftarrow \text{Sample}()$ in the local occupancy grid map; $i \leftarrow i + 1$;
 5. $P(q_{start}, q_{goal}) \leftarrow \text{PathPlanning}(q_{start}, q_{goal})$;
 6. Update the local occupancy grid map;
 7. $S \leftarrow \text{Recognize}(P(q_{start}, q_{goal}), M_{goal})$;
 8. if $s = 1$ then
 9. Go to pose recovery module;
 10. break;
 11. else
 12. $q_{start} \leftarrow q_{goal}$;

Evaluation

- 5 The mobile robot 100 is evaluated to test its operational readiness. For the evaluation, the system architecture 200 is implemented and runs on Robot Operation System (ROS) under Ubuntu 12.04. Table 1 includes the related parameter settings used in the lost robot recovery method of Figure 3.

Table 1: parameter settings

δ	Alignment threshold	1.5 m
$t_{resting}$	System Resting time	3 s
σ	Search Area	$6 \times 6 \text{ m}^2$
N_{lost}	Lost detection threshold	99

10

For the evaluation, the mobile robot 100 is configured to navigate in an office environment as illustrated in Figure 7. During the offline global map building stage 210, the remote controller is used to control the robot 100 to explore the office environment in which the robot 100 is meant to operate. The laser sensor

125 is used to build the occupancy grid map 212 using ROS gmapping package. The images captured from the RGB-D sensor 124 are used to build the cognitive map 211 using an improved version of RatSLAM. The cognitive map 211 has a topological structure in which the nodes are associated with the corresponding visual cues and locations, denoted as visual experiences. The visual experiences serve as a space memorization mechanism to remember all the ever-visited places that the robot 100 has been to. The cognitive map 211 contains a set of spatial coordinates that the robot 100 has travelled which are actual robot poses. Therefore, the cognitive-occupancy grid map (or global map) 213 is constructed.

Figure 7 illustrates the constructed cognitive-occupancy grid map 213 although in Figure 7, the reference 600 is used instead. The global map 600 is constructed in an office environment with the area of around $21 \times 18 m^2$.

15

The resulting occupancy grid map 212 built by data collected from the laser sensor 125 represents the map of the office environment. This is readily apparent as compared to the raw odometry 602 received from the robot wheel encoders 123 which show significant drifts from the actual robot poses without any drift compensation mechanisms in place. The raw odometry 602 runs outside of the mapped office environment. Visited visual experiences 604 are represented as square boxes in the cognitive map 211. The robot pose trajectory (not shown in Figure 7) is estimated by the Rao-Blackwellized particle filter.

In this evaluation, the mapping process took 780.66 seconds and the robot 100 travelled 265.11 meters around the office environment with an average speed of 0.34 m/s. After completing the map building process, the mobile robot 100 is
5 ready to be deployed and the resulting cognitive-occupancy map 600, together with the self-exploration, lost detection and self-pose recovery modules 150, 128 are used to self-recover the robot 100 if the robot 100 is lost.

Furthermore, when the robot 100 switches from one module to another module,
10 a resting time $t_{resting}$ is provided to allow sufficient time for system updating. For example, when the pre-built grid map (i.e. the occupancy grid map 212) is reloaded, the system rests for 3 seconds in order that there is sufficient time for reconfiguring the system and path planning.

15 During navigation, the AMCL technique for path planning, obstacle avoidance and tracking the pose of a robot 100 against the occupancy grid map 212 is used, and the lost detection module 128 is executed to detect if the robot 100 is lost as explained in relation to Figure 3. When the robot 100 is detected to be lost, the obstacle information is cleared by resetting the global and local
20 costmaps and start the self-exploration module 150 to create the local vicinity occupancy grid map and to look for a marker which corresponds to a visual cue of the ever-visited place, or re-visited place. Thus, the incoming visual inputs captured from the RGB-D sensor 124 are compared with the historical visual

experiences 604 stored in the cognitive map 211. If the latest input matches with one of the visual experiences 604 stored in the cognitive map 211, it is considered as a scene which had been seen/visited previously by the robot 100.

- 5 Figures 8a – 8c depict scenario of the robot 100 being kidnapped and actions performed by the mobile robot 100 to self-recover.

In Figure 8a, the robot 100 is well-localized during navigation (see the occupancy grid map 212 in Figure 4a) to move to a target location 226 along a
10 planned or designated navigation path 810 and it should be appreciated that the lost detection module 128 is running. The robot 100 is then kidnapped to another location outside of the planned or designated navigation path 810. After around 2 – 3 seconds, the lost detection module 128 detects that the robot 100 is lost and the self-exploration module 150 is triggered by the processor 127. In
15 Figure 8b, the local map builder 151 creates a local vicinity occupancy grid map 820 of the lost/displaced robot's vicinity or surrounding, identifies a vicinity destination point 822 and maps out a designated path and the robot 100 searches for all possible ever-visited-places along the designated path. In
20 Figure 8c, the robot 100 finds an ever-visited-place 824 along the designated path and steps 233 and 234 of Figure 5 are executed to re-initialise the robot's pose re-initialization by rotating itself 360°. The occupancy grid map 212 is reloaded and the robot 100 has returned to its planned navigation path. Thereafter, the recovered robot 100 continues on the planned navigation path until the robot 100 reaches its goal i.e. target location 226.

Figures 8a-8c illustrate the effectiveness of recovering the lost robot by creating a local vicinity occupancy map of the vicinity in which the robot is lost and trying to identify an ever-visited place (or familiar place) based on the cognitive-
5 occupancy grid map to recover itself. It should be apparent that the described embodiment is also applicable for other scenario such as when the robot 100 is pushed or when the laser sensor 125 is blocked causing the robot 100 to be displaced.

10 It is noted that in some regions of the office environment such as a long corridor, the laser sensor 125 may not provide sufficient information for accurate laser-map matching. Even though an ever-visited-place with accurate robot pose is found during the self-exploration process, the pose might not be successfully recovered due to a lack of sufficient laser information for laser-map matching. In
15 this case, a subsequent trial is tried again once the current self-recovery fails until the robot 100 is finally properly recovered.

Referring to Figure 7, it is noted that in the circled region 608, the laser sensor 125 did not provide sufficient information for accurate laser map matching
20 during the pose recovery process. Therefore, it required several trials to finally recover the robot 100. Table 2 shows the number of recovery attempts attempted by the lost robot 100 in the circled region 608.

Table 2

No of Experiments	Number of trials to recover the lost robot
1	8
2	8
3	7
4	6
5	9

To improve the performance, the ever-visited-places which have resulted in pose recovery failures may be excluded for the subsequent self-exploration process. Furthermore, the performance of the pure random exploration can be significantly improved by constraining the searching behaviours with some conditions. For example, if the robot 100 cannot find an ever-visited-place along a trajectory from position A to position B, the randomly generated poses within a band of this trajectory can be excluded. Subsequently, the efficiency of the self-exploration can be improved.

It should be appreciated that the lost recovery method 400 of the described embodiment seamlessly integrates map creation and updating, dynamic path planning, place recognition and localization of the mobile robot 100 using a cognitive-occupancy grid map.

The cognitive map 211 integrates visual cue-based episodic memory thereby endowing humans or animals with the capacity to learn and recall experiences in the context of space and thus, the cognitive map 211 is suitable for path

planning and navigation. The cognitive map 211 in robotics acquires, stores, and maintains information about the environment as spatial knowledge which is represented by various topological relationships within the cognitive map 211. Thus, the cognitive map 211 which encodes visual cues of its environments
5 provides an intuitive way to assist the robot 100 to self-recover by itself when the robot 100 is lost.

Furthermore, the proposed method 400 may not require modeling of uncertainties of landmarks, compared to the traditional probabilistic SLAM
10 methods. It is noted that while the cognitive map 211 is able to perform localization with a high accuracy, its ability to map the physical environment is relatively poor when closing a large loop due to odometry drift when the robot 100 has to travel long distances. The cognitive map 211 also does not perform well when used for path planning and obstacle avoidance during robot
15 navigation. However, the limitations of the cognitive map 211 are complemented by the occupancy grid map 212. Therefore, the cognitive-occupancy grid map 213 which synergizes the cognitive map 211 and the occupancy grid map 212 is designed to be implemented with the proposed method to address the lost recovery problem.

20

The proposed method is implemented on a mobile robot that performs autonomous mapping, navigation and localization, lost detection, self-exploration and pose recovery.

The described embodiment is particularly advantageous since the proposed method 400 includes map creation and updating and dynamic path planning via the self-exploration process. The proposed algorithm can thus be applicable to more scenarios.

5

The described embodiment thus proposes a robot system that is capable of detecting if the robot 100 is displaced from its navigational path, and re-localizes to return to its navigational path. In other words, the robot 100 is able to recover from a deliberate displacement from its known position or location. A
10 cognitive-occupancy grid map technique is adopted, which synergizes the cognitive map technique and the occupancy grid map technique to solve the lost robot self-recovery problem.

In other embodiments, a robot system is proposed that comprises an offline
15 global map builder such as the offline pre-building stage 210. The global map builder is arranged to construct an occupancy grid map that represents the location of the robot in relation to its surrounding, based on point cloud data captured using a laser source. The map builder further constructs a cognitive map that captures visual cues acquired, classified, and stored, visual cues
20 associated with position (pose) or location of the robot and captured.

In such other embodiments, the robotic system further comprises an online navigator for path planning and obstacle avoidance. The robotic system further comprises a lost detection & recovery module for ascertaining if robot has been

displaced; the module further comprises a self-exploratory module, which is triggered if the robot is ascertained to be displaced or lost. The module searches for previously traversed or visited location that is classified by the cognitive-occupancy grid map. Upon triggering or flagging that a visited location
5 has been recognized, a visual cue associated with the pose of the robot is recalled from the cognitive map; and initializes a pose re-initialisation and recovery module; and resumes navigation. As a result, the robotic system can self-recover from a displaced or lost situation.

10 It should be clear that although the present disclosure has been described with reference to specific exemplary embodiments, various modifications may be made to the embodiments without departing from the scope of the invention as laid out in the claims. For example, although the robot 100 is described as being deployed in an office environment, it should be clear that the described
15 embodiments can be suitably applied to a wide variety of operating environments. Similarly, although the framework is implemented on mobile robot 100, the lost robot recovery framework can be implemented on a broad variety of autonomous robotic systems, including for example, autonomous driving vehicles.

20

Further, various embodiments as discussed above may be practiced with steps in a different order as disclosed in the description and illustrated in the Figures. Modifications and alternative constructions apparent to the skilled person are understood to be within the scope of the disclosure.

25