

## Hybrid robot control and SLAM for persistent navigation and mapping

Michael Milford<sup>a,b,c,\*</sup>, Gordon Wyeth<sup>a</sup>

<sup>a</sup> School of Engineering Systems, Queensland University of Technology, 2 George Street, Brisbane QLD 4001, Australia

<sup>b</sup> Queensland Brain Institute, The University of Queensland, Brisbane, Queensland 4072, Australia

<sup>c</sup> School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane, Queensland 4072, Australia

### ARTICLE INFO

#### Article history:

Available online 15 June 2010

#### Keywords:

Hybrid robot control  
SLAM  
Navigation  
Mapping

### ABSTRACT

For a mobile robot to operate autonomously in real-world environments, it must have an effective control system and a navigation system capable of providing robust localization, path planning and path execution. In this paper we describe work investigating synergies between mapping and control systems. We have integrated development of a control system for navigating mobile robots and a robot SLAM system. The control system is hybrid in nature and tightly coupled with the SLAM system; it uses a combination of high and low level deliberative and reactive control processes to perform obstacle avoidance, exploration, global navigation and recharging, and draws upon the map learning and localization capabilities of the SLAM system. The effectiveness of this hybrid, multi-level approach was evaluated in the context of a delivery robot scenario. Over a period of two weeks the robot performed 1143 delivery tasks to 11 different locations with only one delivery failure (from which it recovered), travelled a total distance of more than 40 km, and recharged autonomously a total of 23 times. In this paper we describe the combined control and SLAM system and discuss insights gained from its successful application in a real-world context.

© 2010 Elsevier B.V. All rights reserved.

### 1. Introduction

If mobile robots are to become a ubiquitous and functional part of our homes and workplaces, they will need to possess competent navigation capabilities and run control systems that manage many control processes of varying complexity. For many years, two of the robotics research fields addressing these challenges, namely Simultaneous Localization And Mapping (SLAM) and robot control, have continued in parallel with little overlap. While progress has been made in isolation in each field, this largely independent approach has only partially addressed the overarching goal of having mobile robots in everyday life.

While hybrid control architectures can be elegant and offer great promise, their practical integration with persistent SLAM systems operating in challenging real-world environments is an open area of investigation. When a roboticist brings together a SLAM system and a robot control system which were developed independently of each other, they face the challenge of adapting or abstracting the world representations created by the SLAM system

into a format that the chosen control system can use. An example of this process is the abstraction of occupancy grid maps into topological maps for global path planning. The corollary is that much of the information stored in the SLAM-generated maps can be redundant for the purposes of control. A millimeter resolution occupancy grid map is mostly unnecessary for a robot which must perform delivery tasks in an office environment—navigation from place to place can just as easily be achieved by combining a topological map with a robust local obstacle avoidance system. If the robot must place the delivery at a precise location, then the fine grained occupancy grid becomes useful as the robot approaches the delivery location—but at that point a reactive control procedure based on sensor readings will also do the job effectively. If instead, the robot is allowed to autonomously build its map of the world with the same control system that *uses* the map, problems of map abstraction and information redundancy are largely removed. The resulting representations are grounded in the control algorithms used to create them, simplifying their use in later control.

The caveat to this approach is that the robot must be equipped with a control system that enables it to robustly acquire and maintain a world representation. In this paper we present and demonstrate an operational hybrid robot control system which has this ability. We extend the notion of creating the map with the same control system that uses it, by performing SLAM and robot control simultaneously and continually whenever the robot is operating. Consequently, robot control is based on control of

\* Corresponding author at: School of Engineering Systems, Queensland University of Technology, 2 George Street, Brisbane, QLD 4001, Australia. Tel.: +61 7 3138 9969; fax: +61 7 3138 1469.

E-mail addresses: [michael.milford@qut.edu.au](mailto:michael.milford@qut.edu.au) (M. Milford), [gordon.wyeth@qut.edu.au](mailto:gordon.wyeth@qut.edu.au) (G. Wyeth).

many continuous processes rather than a set of discrete actions. By deploying the SLAM and control systems in a real-world, challenging application scenario, we hope to provide some unique insights and perspective on the benefit of hybrid control systems, to complement the extensive theoretical and simulation work that has already been performed in this field.

The paper proceeds as follows. In Section 2 we briefly review state-of-the-art techniques in SLAM and hybrid robot control. Section 3 provides an overview of the RatSLAM SLAM system, focusing on the characteristics of the maps it produces. The hybrid control system is presented in Section 4, along with details of the specific control processes. Section 5 describes an experiment in which the robot acted as a delivery robot in two in-use office buildings over a period of two weeks. The mapping and delivery performance results from this experiment are briefly presented in Section 6, before a more detailed examination of how the control system performed in a number of different situations that occurred during the two week period. Finally, in Section 7 we discuss some of the benefits of and insights gained from integrating a SLAM system with a hybrid control architecture, and detail some profitable areas of future work.

## 2. SLAM and hybrid control systems

To successfully combine a control system and SLAM algorithm on an autonomous mobile robot, consideration must be given to the various types and levels of control processes and the map resources provided by the SLAM algorithm. If map building and map use is to occur using the same control system, it must contain processes that will enable the SLAM process to occur even when the SLAM algorithm is first learning a map of the world, and later, when the SLAM algorithm is maintaining and updating the map. Hybrid control architectures offer a solution to this challenge by providing a mix of reactive stateless control processes and behaviors and higher level deliberative processes. In this section we summarize the key research outcomes in the SLAM and hybrid control topic areas that lay the groundwork for the combined control and SLAM system.

### 2.1. SLAM

The SLAM problem has been one of the major problems addressed in robotics research over the past two or three decades. Most SLAM algorithms are probabilistic to some degree and incorporate Kalman Filter, particle filter, or graphical approaches. There are several SLAM algorithms that can form accurate maps of large and complex environments such as office buildings [1] and mine shafts [2]. Researchers in the SLAM field typically test and refine their algorithms by running them on sensor datasets on a computer, rather than in real time on a robot. Typically these datasets are obtained by manually driving a robot or sensor platform (such as a car [3]) through an environment. Roboticists ‘play back’ the sensor data to their SLAM algorithms. To facilitate this approach, there are now many large, publicly available benchmark datasets available to SLAM researchers [4]. As evidenced by the wealth of SLAM algorithms available today [5], such as FastSLAM [6], GraphSLAM [7] and GridSLAM [8], to name just three, in some respects this dataset-driven approach has been quite productive. Although not strictly speaking SLAM techniques, a number of topological or hybrid metric-topological approaches have also been developed [9–12].

Unfortunately, the dataset-driven methodology only partially address one of the overarching goals of solving the SLAM problem, namely, enabling mobile robots to enter everyday life in domestic homes and the workplace. Work towards this goal involves incorporating SLAM algorithms into complete

autonomous robot control systems, and in this area research is much less progressed [13]. Of the few systems that combine SLAM algorithms and autonomous control systems on robots operating in real environments, most typically use a SLAM algorithm in a preliminary user-defined mapping phase. The map is then finalized and used for autonomous goal navigation. Examples of this approach include the museum tour guide robot MINERVA [14] and the MobileRobots MAPPER software package [15].

### 2.2. Hybrid control systems

In parallel with SLAM research, there has been extensive work on robot control systems and, of particular relevance to autonomous mobile robots, hybrid control architectures that combine deliberative, reactive and behavioral control. Deliberative control generally signifies reasoning about what to do based on state information, whereas reactive control refers to a control process where perception and action are directly coupled. In-between lies behavior-based control processes, which incorporate some form of planning into the perceive-act process but little state information. Examples of hybrid control architectures include the Autonomous Robot Architecture (AuRA) [16], Atlantis [17], the Playmate/Explorer CoSy Architecture Sub Schema (PECAS) [18] and others [19–21].

The Autonomous Robot Architecture (AuRA) combined a high level deliberative (hierarchical) planner based on classical AI techniques with a reactive controller derived from schema theory [17], implemented in a modular fashion. Once the high level planner handed over control to a reactive process, it ceased management until a failure event at the reactive level. Failures were handled by sequentially moving up a hierarchical tree of high level control processes. The Atlantis architecture followed the concept of managing a number of continual control processes through a hierarchical approach, rather than performing discrete sequential actions [17]. *Activities* consisted of specific physical and computational processes performed by a robot, while *decisions* were the results of computational processes which initiated or terminated activities. The hierarchy resulted from identifying low level activities as involving little computation, making decisions with short term effects, not using much state information and processing relatively ‘raw’, non-abstracted sensory data.

Common to many of these systems are the key concepts of hybrid control architectures that combine deliberative, behavioral and reactive control processes that run continually and at multiple levels in a hierarchical structure. A hybrid architecture lends itself naturally to application in the field of autonomous mobile robots, which are faced with dynamic changing environments and which need to perform tasks of widely varying complexity over a wide range of time and spatial scales. Certain hybrid control systems have been successfully demonstrated in simulation [21] or in specific real robot experiments [18]. In some of these experiments, the robot’s low level reactive behaviors enabled it to explore and learn simple spatial or perceptual information about its environment, which was then used by the control system to perform useful tasks. Hybrid control architectures have been coupled with SLAM techniques in specific task domains such as RoboCup rescue. However, in the context of developing persistent, highly mobile robots for workplaces and homes, there has been little investigation into the potential mutual benefits of pairing modern SLAM algorithms and hybrid control architectures, especially with respect to having the same control system create and use the world representations. In the rest of this paper, we describe a study investigating the use of an integrated control-SLAM system on an autonomous robot.

### 3. RatSLAM

RatSLAM is a robotic SLAM system inspired by models of the neural processes underlying navigation in the rodent brain, and specifically the rodent hippocampus. For details of the SLAM algorithm and its biological foundations please refer to [22–25]. Unlike many previous biologically inspired navigation systems which have emphasized biological fidelity at the cost of navigation performance [26–30], RatSLAM is a capable navigation system which has mapped areas as large as a city suburb using only a web camera [24]. In this section, we provide a brief overview of the map resource that RatSLAM provides, which is relevant to the design and operation of a complete autonomous robot control system.

RatSLAM produces an *experience map*, a semi-metric topological graph map, containing nodes called *experiences*, and links between these experiences (Fig. 1). Experiences exist in *experience space*, which is similar to real-world Cartesian space. The map layout is continually updated and corrected through a process similar to graph relaxation. The appearance of an experience map is similar to that of maps created by other techniques such as GraphSLAM [7], although with a very different backend generating the map.

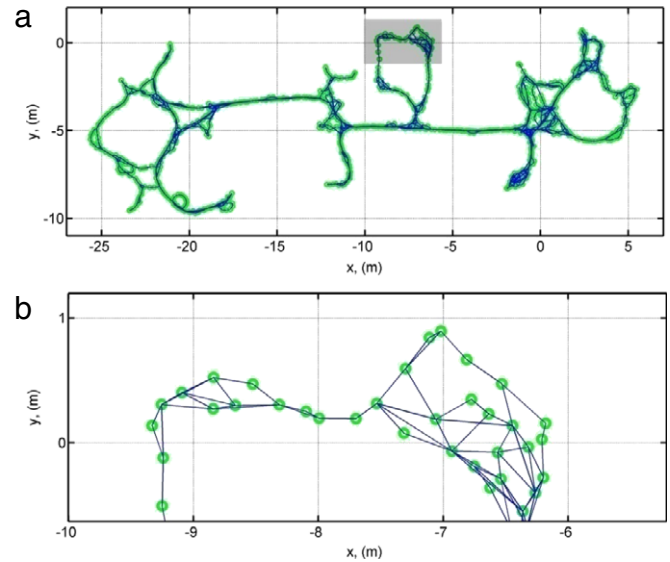
Experience links store several pieces of information obtained directly from low level control processes. The links store transition times indicating the typical time taken to move between connected experiences in the map. Low level behavioral information is also stored in the links, such as whether the robot was performing left-wall-following, right-wall-following or center-line following as it navigated between the two linked experiences.

The experience map is locally semi-metric in that its layout on a local scale is usually quite faithful to the geometry of the environment. Consequently, the Cartesian properties of the map on a local scale can be exploited and used to enhance the performance of reactive-only control processes, such as described in Section 4. Conversely, on a global scale the map is more topological than metrically accurate, and hence is suited to more abstract control processes such as higher level global navigation. The division of navigation into global and local processes follows the classical division first proposed by Latombe [31].

The experience map is also inherently dynamic in nature, which affects how it can be used by a control system. The mapping algorithm continues to add experiences as the robot explores more of the environment and as the environment changes perceptually and geometrically. The problem of unbounded map growth is addressed by an experience pruning algorithm [23], which consolidates parts of the experience map that exceed a spatial density threshold. Transitions between experiences are also added and deleted as part of this pruning process. This map pruning ensures long-term stability both in terms of the map and computation.

### 4. Control architecture

The overall control architecture is shown in Fig. 2. Control is split by function into a number of modules, each of which can contain several control processes. The control system also draws upon two different map resources (shown in the irregularly shaped diagram blocks); a local obstacle map in robot-centric coordinates derived in this implementation from laser and sonar sensors, and a global semi-metric topological map produced by RatSLAM known as the experience map. Three high level function modules (surrounded by a dashed line in Fig. 2) provide the robot with the capability to explore the environment, perform deliveries, and recharge. The internal workings of these modules are laid out in a deliberative sequential sequence but make use of various reactive or behavioral processes. The global navigation, local navigation and docking maneuvers modules interface between the high level



**Fig. 1.** Experience map for the environment shown in Fig. 5b. (a) Overall map and (b) closeup of the shaded area in (a) showing individual experiences and experience links. On a local scale the experience map is close to Cartesian in its representation of the environment, but on a global scale the representation becomes more topological.

function modules and low level reactive and behavioral control processes such as obstacle avoidance.

Most control modules and their associated processes run continually, but not all are actively controlling the robot at any one time. Active modules are managed on a *priority* basis—the robot will perform deliveries unless a low battery voltage activates the recharge module, which will then provide its own global goal location that subsumes the goal provided by the deliver module. If neither deliver nor recharge modules are active, the explore module will control the robot. The local navigation module is always active, although its motor commands may be subsumed by the docking maneuver module when the robot is homing in on the charging station.

#### 4.1. High level functions

The control system contains three high level function or task modules— explore, deliver, and recharge. Each module uses lower level control processes such as obstacle avoidance (local navigation module) and charging station homing (docking maneuvers module). The exploration module uses global map information and obstacle avoidance movement trajectories from the local navigation module to specify a local robot path trajectory that will take the robot into unexplored areas of the environment. The deliver module carries out mock deliveries to user-specified locations. The recharge module first navigates the robot to a location near the recharging station with an appropriate approach angle, and then uses a local homing procedure to physically dock with the station.

##### 4.1.1. Explore

The exploration module uses the global experience map and robot's localized position to extract a list of the routes that the robot has previously traversed from its current location. Segments from these routes centered on the robot's current location are converted into robot-centric coordinates. These previously traversed route segments are then compared with the obstacle-safe movement trajectories generated by the local navigation obstacle avoidance process. The explore module chooses the

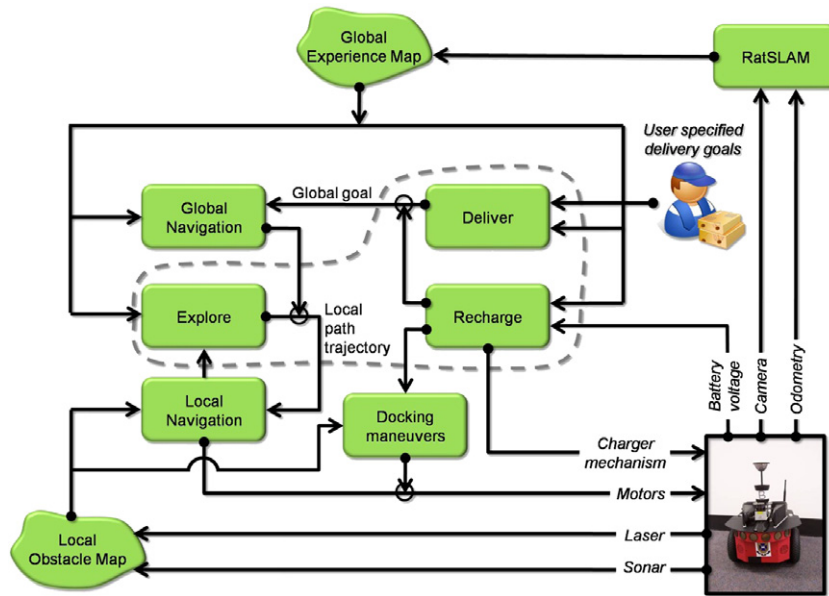


Fig. 2. The overall control architecture consists of a number of function modules which run continually and in parallel, which can each contain several processes. Only a subset of the modules are active at any one time, and higher priority modules such as Recharge can subsume lower priority modules such as Deliver.

trajectory that is least similar to any previously traversed, which improves the robot’s map coverage of the environment. For example, if the robot has previously turned left (using left-wall-following behavior) at an intersection, it will attempt to turn right (using right-wall-following behavior) to increase its map coverage of the environment. Exploration is the default behavior and is activated when the robot is initially placed in the environment, between deliveries, and during periods of bad localization.

4.1.2. Deliver

The deliver module uses a scheduler to pick a goal at random from a list of user-specified delivery locations. A global goal location is then provided to the global navigation module, which navigates the robot to that location. As in the AuRA control architecture, the lower level global navigation process then takes over control, returning control to the delivery module only after navigation has succeeded or failed.

The global navigation module itself runs in parallel with local navigation processes such as obstacle avoidance and trajectory generation.

4.1.3. Recharge

The recharge module first navigates the robot to the vicinity of the charging station, and then docks the robot and engages the charging mechanism. The global goal is not set by a user but rather is a location known to provide the robot with a suitable position from which to home onto the docking station plate. This location is tagged in the experience map during the robot’s exploration of the environment, using the robot’s laser and prior information about the cross-sectional geometry of the charging station. The recharge process then hands over primary control to the global navigation module.

The recharge module regains control when navigation succeeds or fails. If navigation is successful, the recharging system hands over control to the docking maneuvers module. First, a simple spin on the spot behavior rotates the robot until it is able to observe the docking station with its laser. Then a local vector field-based homing routine maneuvers the robot over the charging plate (Fig. 3). The homing process returns control to the recharge module which then engages the charging mechanism and charges the robot. Failure at any stage of this process hands back control to the recharge module.

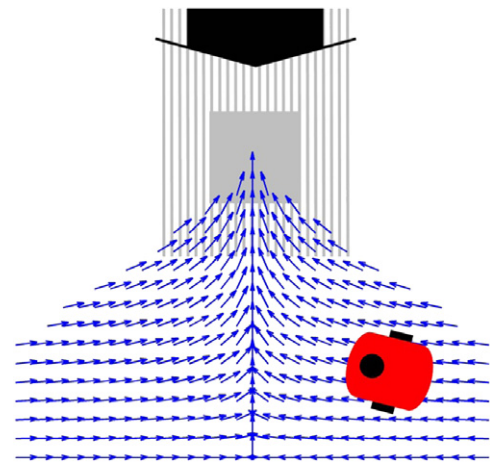
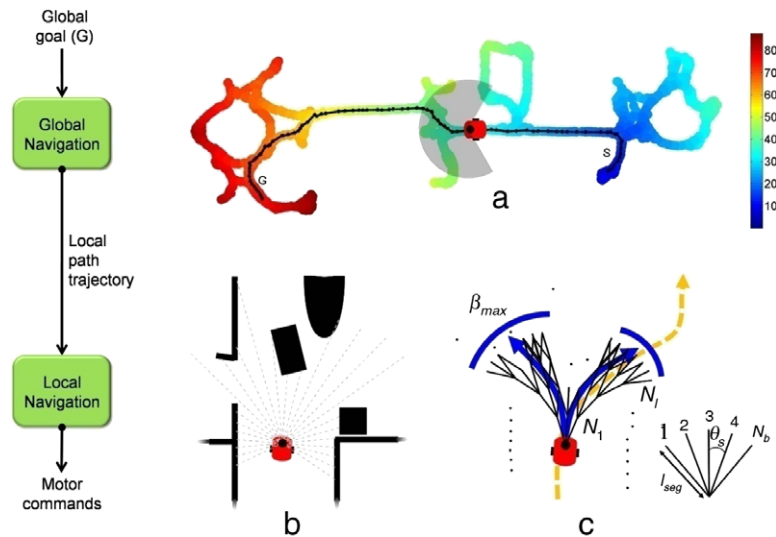


Fig. 3. A schematic of the reactive “move over dock” process which uses potential vector fields to home in on the charge plate. Because the recharge and global navigation modules ensure that the robot starts this process in an appropriate position in front of the docking station, the reactive homing procedure can be simple, robust, and precise.

4.2. Global navigation

The global navigation module provides the robot with the ability to plan and execute optimal journeys through the environment over distances that would prove challenging to a purely reactive system. In this particular implementation, journeys are optimized in terms of duration rather than distance covered, to ensure a maximum delivery rate, albeit at the cost of a slightly lower battery life. The information contained in the experience map is used to create a temporal map with a value of zero at the robot’s current location. Using the transition information between experiences, the temporal map is iteratively updated to find the minimum journey times to every other part of the map (a temporal map is shown in Fig. 4a). The optimal journey path is calculated by performing gradient descent from the goal location back to the robot’s current location (also shown in Fig. 4a). The navigation control layer then passes the immediate section of the path (located around the robot’s current position along the path)



**Fig. 4.** The global and local navigation modules combine to navigate the robot between a start (S) and goal (G) location. (a) The global temporal map and planned path, with the robot localized partway along the path. (b) The local obstacle map obtained from laser range scans (approximately corresponding to the shaded scan area in (a)). (c) Local obstacle avoidance and safe trajectory generation. The robot follows the local safe movement trajectory (thick solid arrow lines) most closely matching the desired movement trajectory (thick dashed arrow line) provided by the explore or global navigation routines.

to the local navigation module. The interplay between global and local navigation is shown in Fig. 4.

The global navigation module continually assesses the quality of the localization estimate being provided by the SLAM system. A quality score is derived directly from the time duration since the SLAM system last successfully matched its current location to an experience stored in the experience map. This time duration represents how long the robot has had to rely on dead reckoning (rather than visual place recognition) to update its global estimate of location. The localization quality is used by the delivery module as part of its recovery method.

### 4.3. Local navigation

The local navigation module shown in Fig. 2 calculates a set of safe and distinct movement trajectory options around obstacles, extracting these from a tree search of safe movement space in the robot's immediate environment. The module uses the local obstacle map but not the global experience map. In the implementation described in this paper, the local obstacle map is created using sensory input provided by a laser range finder device and sonar sensors. Any sensory input could be used as long as it provided a means of detecting safe paths through the environment and identifying the charging station bearing and distance. For example, a stereo camera rig would also be a suitable sensory driver for low level reactive control.

A tree search algorithm is performed within a local Cartesian obstacle map in order to calculate safe movement trajectories through space. The tree search is performed to a depth of  $N_1$  links. Each fork contains  $N_b$  branches, with each branch of length  $l_{seg}$ . Branch propagation ceases when the clearance distance decreases below an obstacle clearance threshold  $r_{min}$ . The path tree is clustered into groups where every group member's endpoint bearing relative to the robot falls within an angle  $\beta_{max}$ . The midpoint of each of these clusters becomes a candidate path point. The number of candidate path points found dictates what options the robot has for safe obstacle-free movement through the local environment. For example, two candidate path points indicate that the robot can perform either left-wall-following or right-wall-following. To generate actual motor commands, the candidate path points are converted into local trajectory arcs using the motion

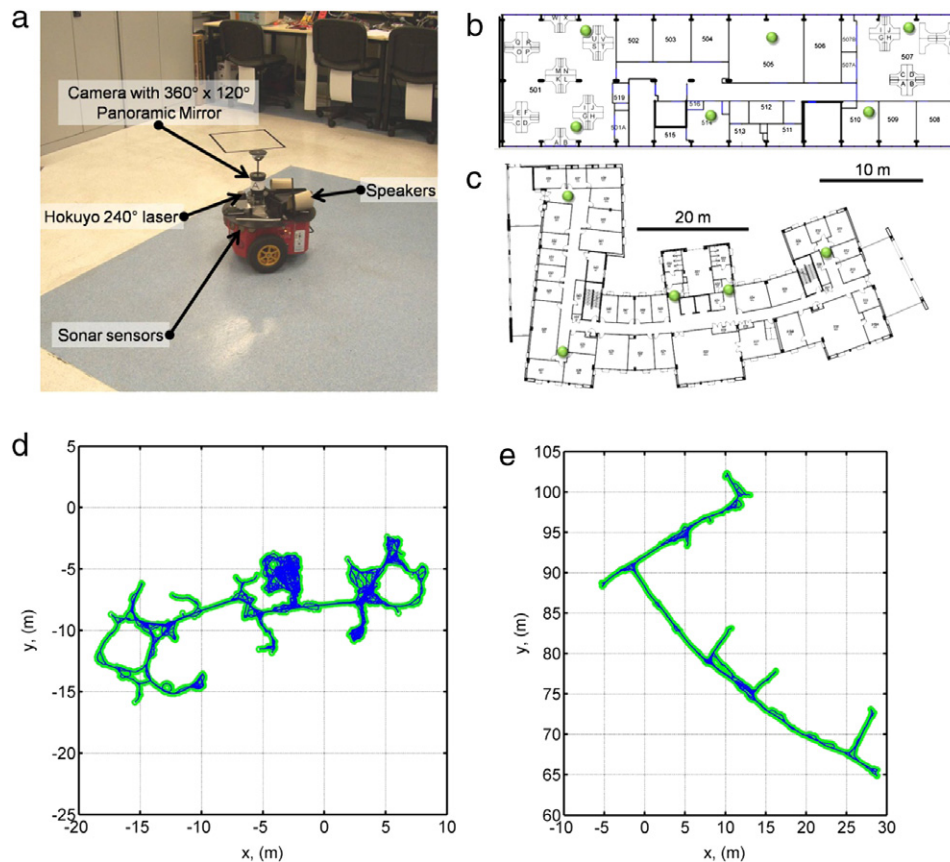
model for the robot. The tree search is similar to those used in outdoor robot experiments [32].

The local navigation module takes the desired local movement trajectory provided by the explore or global navigation modules as primary input. The local navigation module picks and executes the safe movement trajectory most closely aligned with the desired movement trajectory (Fig. 4c). The alignment of two trajectories is assessed using a mean sum of minimum distances between a sampling of points along each trajectory. The safe movement trajectories generated by the local navigation module are also provided as a resource to the explore module. In comparison to other techniques such as VFH\* [33], the local navigation module is only responsible for following a locally specified trajectory. The global navigation module ensures that the globally planned path is optimal (in terms of duration) and traversable, based on previous experience of motion along the route.

## 5. Experimental setup

The robot experiment was conducted in two different office building floors of Axon building (Fig. 5b) and GP South building (Fig. 5c) over a period of two weeks at the University of Queensland, at all times of day and night. The robot platform was a Pioneer 3 DX robot equipped with a panoramic imaging system, Hokuyo laser range finder, wheel encoders and an array of forward facing sonar sensors (Fig. 5a). The robot operated for a period of two to three hours before returning to the docking station to recharge, with a typical day consisting of one to four recharge cycles. The office buildings were standard, in-use environments, with human traffic during the day and cleaning staff at night. The robot was exposed to a range of room types including kitchens, offices, open plan floor space and corridors. As well as moving people, other items moved around in the environment including furniture and equipment such as trolleys. Since the experiment was run during the day and at night, lighting conditions varied significantly in some areas of the environment with large window areas.

To start the experiment, the experimenter positioned the robot in the Axon building floor and started the robot's control software. The robot started exploring the environment. After two hours, the experimenter provided the robot with six mock delivery locations by clicking on these locations in the experience map. Upon detection of the existence of delivery location candidates, the



**Fig. 5.** A summary of the mapping and delivery results. (a) The robot in operation in a laboratory area in the Axon building. The sensor suite included sonar sensors, a Hokuyo laser range finder with 240° field of view, a panoramic imaging rig and standard wheel encoders for odometry. (b) Floorplan of the Axon building and scale bar, with dots marking the six delivery locations. (c) Floorplan of the GP South building. (d) Experience maps at the end of the experiment shown for Axon building and (e) GP South building.

robot control system switched to delivery mode, with the robot starting navigating to delivery locations randomly selected from the list of candidates. When the battery level dropped below a threshold, the robot control activated the recharge module, and the robot returned to the charger and docked to recharge.

After more than a week in the Axon building, the robot was moved during a power down cycle to the GP South environment and turned back on. After a period of exploration, the experimenter provided the robot with five new delivery locations, and the delivery and recharge process was repeated in this second environment. After 54 delivery trials, the robot was finally placed back into the original Axon environment and a further 72 delivery trials were conducted.

## 6. Results

The robot created an experience map representation of the two environments which stabilized in number of experiences after the first few hours in each environment. The robot successfully performed 1143 deliveries out of 1144 attempts, to 11 different delivery locations spread over the two building environments. A delivery was counted as successful if the robot was in the vicinity of the correct delivery location when it reported it had made a delivery. The mean delivery accuracy was 1.0 m in the Axon building and 1.2 m in the GP South building (achieved without any local reactive homing procedure). Deliveries typically varied between one to three minutes in duration depending on the length of the path. The mapping results are summarized in Fig. 5d–e. For more quantitative details including computational stability please see [23]. In this section we focus on how the various control

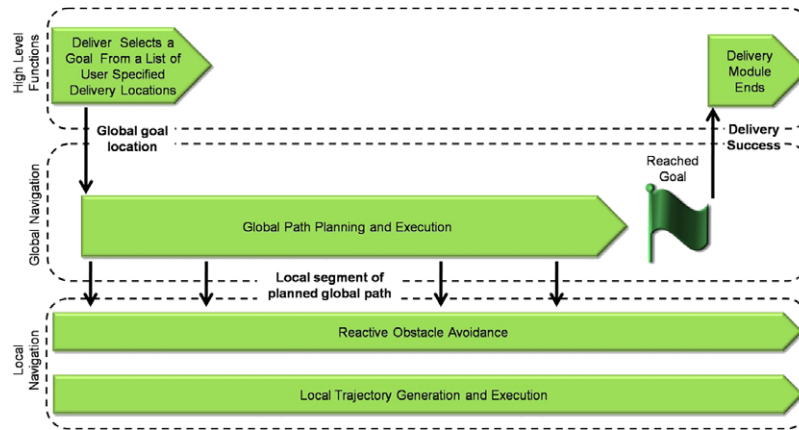
processes at all levels functioned in different situations which occurred during the experiment.

### 6.1. Delivery task performance

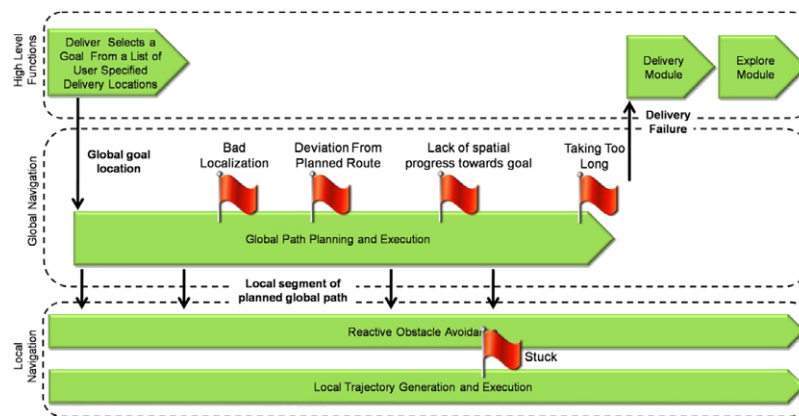
Fig. 6 shows a timeline of the various control processes for a successfully executed delivery during the experiment. After a high level decision to perform a delivery was made, a delivery location was chosen by the delivery module and passed as a global goal location to the global navigation module. The global navigation module ran continually, calculating the optimal route to the goal location and providing the local navigation module with a segment of this optimal route corresponding to the desired local trajectory from the robot's current location. The local navigation module also ran continually, calculating the safe movement trajectory that most closely matched the movement trajectory provided by the global navigation module. After the robot successfully reached the delivery goal location, control was passed back to the deliver module with a delivery success signal.

One delivery was not completed successfully due to a period of degraded localization performance in an open laboratory area (the area seen in Fig. 5a). During delivery trial number 579, the quality of the robot's localization estimate degraded temporarily during a sojourn into the laboratory area. Fig. 7 shows the non-critical and critical flags raised at various times by different control processes.

Due to the localization error, the robot started to deviate significantly from the route it had planned to the delivery location. A poor localization flag was raised within the global navigation module. Once the robot had deviated a certain distance from the planned route, a route deviation flag was raised by the path



**Fig. 6.** A successful delivery. With low level reactive processes running continually, the global navigation module guided the robot along a continuously calculated optimal path to the delivery location.



**Fig. 7.** The sole delivery failure, as processed by the control layers. The failure started as a period of bad localization, flagged in the global navigation module. Soon the robot deviated from the intended route and due to bad localization was not able to plan another route on the fly. A little later the path planner flagged that the robot was not getting closer to the global goal, while the local navigation system reported that the robot was 'stuck' in the same area of the environment. The final, critical flag was a time-out in the global navigation module, which caused control to cede back to the high level delivery module, which in turn ceded control to the explore module which re-localized the robot.

planning process, indicating that the robot was not correctly following the optimal planned route. Because the robot was not localized, the path planner was unable to plan a new route to the goal, so it continued to provide a local desired trajectory to the local navigation module, based on the last planned route and dead reckoning. After a further period of time, the global navigation module flagged that the robot had not appeared to get closer to the goal location, and the local navigation module detected that the robot was having movement difficulties. The robot was in reality, being kept in a corner of the laboratory by the interplay between the local and badly localized global navigation systems. Finally, after a period of five minutes (hard coded), the path planner reported that the delivery was taking too long to achieve and ceded active control back to the delivery module. The delivery module ended that particular delivery task, and control was ceded back to the default exploration module. While normally the exploration module would only run for 10 s between deliveries, a new delivery task only started when the robot had explored sufficiently to become well localized again.

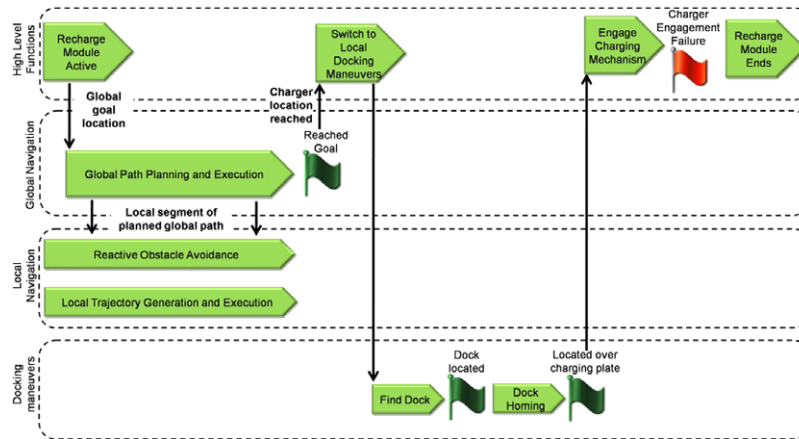
## 6.2. Autonomous recharging

In order to operate over the two week experimentation period, the robot needed to autonomously recharge itself. In the spirit of being capable of "out of the box" operation, the scenario was designed such that the user was required to place the robot charger

somewhere in the environment, but not notify the robot of its location. Consequently, the robot had to find and learn the location of the charger, as well as navigate to and dock with it when the robot's battery level dropped below a low charge threshold. The act of autonomously recharging used several different control modules and both deliberative and reactive control processes.

The robot activated the recharge module a total of 35 times over the two week period. Each time the robot attempted to navigate back to the charger's location, approach and move over the charger and successfully engage with the charging mechanism. Because of dust and the relative fragility of the docking mechanism, of the 35 times the robot placed itself over the charging dock, the charging mechanism only engaged properly 23 times (66% success rate).

Fig. 8 shows a complete recharge cycle which did not initially succeed due to an unsuccessful physical contact between the robot's recharging components and the charging station plate. After a battery level monitoring process detected a battery voltage below a minimum threshold, it activated the recharge module. The recharge module provided a suitable charging station approach location to the global navigation module as a goal. After the global navigation system successfully navigated the robot to this location, the recharge module then passed control to a sequence of two docking maneuver processes which first located the charging station and then maneuvered the robot onto the charging plate. Once located on the dock, a high level process deployed the docking mechanism.



**Fig. 8.** A charger engagement failure. After reacting to a low voltage warning, the recharge module became active and provided the global navigation module with a charging station goal location. After the global navigation module successfully navigated the robot to the vicinity of the docking station, control returned temporarily to the recharge module, which then handed control over to the dock maneuver processes, which located the charging station and maneuvered the robot onto the charging plate. Once positioned, the recharge module attempted to engage the robot charging mechanism.

In this particular example, the robot's docking mechanism did not successfully engage the charging plate. The failed engagement caused the control system to suspend delivery module activation, and temporarily suspend recharge module activation, causing the robot to revert to exploration in an attempt to 'go around' for another try. After a short period of exploration, the robot again activated the recharge module and successfully docked with and charged from the charging station.

## 7. Discussion and future work

In this paper we have described an integrated robot control and SLAM system and demonstrated its effectiveness in a long-term real-world experiment. In this section we discuss some key characteristics of the system that facilitate its deployment on real robots and highlight some areas for future work.

The key characteristic of the system is the tight coupling between the robot's control system and the SLAM algorithm's spatial representations. Unlike previous autonomous SLAM system approaches, map acquisition, maintenance and usage are all run continuously and use the same control system. Consequently, the resultant map is grounded in the robot's control processes. For example, the constrained paths the robot follows as it moves down a corridor due to the reactive obstacle avoidance and trajectory, planning processes are reflected in the experience map of that area. The experience layout and links reflect the reactive control behavior that navigated the robot down the corridor. The SLAM algorithm does not need to store detailed spatial proximity information because the path it followed during map acquisition will be the same path it uses later when performing deliveries, thanks to the low level reactive behaviors. A corridor becomes a pathway rather than a high resolution occupancy grid.

There have been recent attempts to provide a widely applicable evaluation framework for robot motion methods and systems [34]. Based on the results presented in this paper, we suggest that the quantitative metrics used in evaluation frameworks, such as navigation duration, should be complemented by more general metrics such as how flexible and robust a robot control and navigation system is. For example, while the average delivery time could be reduced, this would likely come at the expense of delivery reliability, which was very high for this long-term experiment (>99.9%). Furthermore, a robot control and navigation system that has minimal and flexible sensory requirements and hence can be deployed in a wide range of robot applications may be more valuable than one that is 10% faster in performing its task.

Performing this research has highlighted several areas for future work. As suggested by Power and Balch [21], the performance of the high level deliberative control layer could be improved by incorporating feedback from the reactive layer control processes. In the context of the delivery robot scenario presented in this paper, the recharging task provides one example of how this technique could improve overall performance. Based on the feedback from the dock locating and dock homing behaviors, the recharge module could adjust the battery threshold voltage. If the low level reactive processes were having a low success rate, the threshold might be adjusted upwards to allow the robot more time to dock. Conversely if the reactive processes achieved success rapidly, the voltage threshold could be adjusted downwards to allow the robot to complete more deliveries between recharge cycles. More sophisticated local navigation methods could also enrich the feedback from the reactive layer to the deliberative layer [32,35,36]. The issue of generalization is also an important one—the system development cycle was, relative to much of the work in hybrid robot control, specific to the delivery robot application. However, we also suspect that many of the generalized hybrid control architectures would require extensive tweaking and tailoring to be successfully deployed in an application-specific scenario such as a delivery robot task. Deploying more of the hybrid control architectures in real-world autonomous robot applications that require the robot maintain some form of world representation will reveal more key issues that must be solved on the way to a robot-rich society.

## References

- [1] G. Grisetti, C. Stachniss, W. Burgard, Improved techniques for grid mapping with rao-blackwellized particle filters, *IEEE Transactions on Robotics* 23 (2007) 34–46.
- [2] S. Thrun, D. Hahnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, W. Whittaker, A system for volumetric robotic mapping of abandoned mines, Presented at International Conference on Robotics and Automation, Taipei, Taiwan, 2003.
- [3] M. Cummins, P. Newman, Highly scalable appearance-only SLAM—FAB-MAP 2.0, Presented at Robotics: Science and Systems, Seattle, United States, 2009.
- [4] Radish: The Robotics Data Set Repository, Andrew Howard and Nick Roy.
- [5] [OpenSLAM.org](http://OpenSLAM.org), 2009.
- [6] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges, Presented at International Joint Conference on Artificial Intelligence, Acapulco, Mexico, 2003.
- [7] S. Thrun, M. Montemerlo, The GraphSLAM algorithm with applications to large-scale mapping of urban structures, *The International Journal of Robotics Research* 25 (2006) 403–429.
- [8] G. Grisetti, C. Stachniss, W. Burgard, Improving grid based SLAM with Rao blackwellized particle filters by adaptive proposals and selective resampling, Presented at International Conference on Robotics and Automation, Barcelona, Spain, 2005.
- [9] M.O. Franz, P.G. Scholkopf, H.A. Mallot, H.H. Bulthoff, Learning view graphs for robot navigation, *Autonomous Robots* 5 (1998) 111–125.

- [10] S. Thrun, A. Bucken, Integrating grid-based and topological maps for mobile robot, Presented at Thirteenth National Conference on Artificial Intelligence, 1996.
- [11] I. Ulrich, I. Nourbakhsh, Appearance-based place recognition for topological localization, Presented at IEEE International Conference on Robotics and Automation, San Francisco, CA, USA, 2000.
- [12] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, F. Savelli, Local metrical and global topological maps in the hybrid spatial semantic hierarchy, Presented at International Conference on Robotics and Automation, New Orleans, USA, 2004.
- [13] Springer Handbook of Robotics, Springer, 2008.
- [14] S. Thrun, Probabilistic algorithms and the interactive museum tour-guide robot Minerva, *International Journal of Robotics Research* 19 (2000) 972–999.
- [15] MobileRobots, Research & University Software.
- [16] R. Arkin, T. Balch, AuRA: principles and practice in review, *Journal of Experimental & Theoretical Artificial Intelligence* 9 (1997) 175–189.
- [17] E. Gat, Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling realworld mobile robots, Presented at 10th National Conference on Artificial Intelligence, San Jose, California, 1992.
- [18] N. Hawes, H. Zender, K. Sjo, M. Brenner, G. Kruijff, P. Jensfelt, Planning and acting with an integrated sense of space, Presented at 1st International Workshop on Hybrid Control of Autonomous Systems, Pasadena, United States, 2009.
- [19] J. Albus, 4d/rcs: a reference model architecture for intelligent unmanned ground vehicles, Presented at SPIE Aerosense Conference, 2002.
- [20] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J.v. Niekirk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, P. Mahoney, Stanley, the robot that won the DARPA grand challenge, *Journal of Robotic Systems* 23 (2006) 661–692.
- [21] T. Balch, M. Powers, An incremental approach to adaptive integration of layers of a hybrid control architecture, Presented at 1st International Workshop on Hybrid Control of Autonomous Systems, Pasadena, United States, 2009.
- [22] G. Wyeth, M. Milford, Spatial cognition for robots: robot navigation from biological inspiration, *IEEE Robotics and Automation Magazine* 16 (2009) 24–32.
- [23] M. Milford, G. Wyeth, Persistent navigation and mapping using a biologically inspired SLAM system, *The International Journal of Robotics Research* (2009).
- [24] M. Milford, G. Wyeth, Mapping a suburb with a single camera using a biologically inspired SLAM system, *IEEE Transactions on Robotics* 24 (2008) 1038–1053.
- [25] M.J. Milford, Robot Navigation from Nature: Simultaneous Localisation, Mapping, and Path Planning Based on Hippocampal Models, vol. 41, Springer-Verlag, Berlin, Heidelberg, 2008.
- [26] A. Arleo, W. Gerstner, Spatial cognition and neuro-mimetic navigation: a model of hippocampal place cell activity, *Biological Cybernetics* 83 (2000) 287–299.
- [27] N. Burgess, J.G. Donnett, K.J. Jeffery, J. O'Keefe, Robotic and neuronal simulation of the hippocampus and rat navigation, *Philosophical Transactions of the Royal Society of London B Biological Sciences* 352 (1997) 1535–1543.
- [28] J.-A. Meyer, A. Guillot, B. Girard, M. Khamassi, P. Pirim, A. Berthoz, The Psikharpax project: towards building an artificial rat, *Robotics and Autonomous Systems* 50 (2005) 211–223.
- [29] C. Giovannangeli, P. Gaussier, Autonomous vision-based navigation: goal-orientated planning by transient states prediction, cognitive map building, and sensory-motor learning, Presented at International Conference on Intelligent Robots and Systems, Nice, France, 2008.
- [30] A. Barrera, A. Weitzenfeld, Biologically-inspired robot spatial cognition based on rat neurophysiological studies, *Autonomous Robots* 25 (2008) 147–169.
- [31] J.-C. Latombe, *Robot Motion Planning*, 1st ed., Kluwer Academic Publishers, Boston, 1991.
- [32] D. Coombs, K. Murphy, A. Lacaze, A. Legowik, Driving autonomously offroad up to 35 km/h, Presented at Intelligent Vehicles Conference, Dearborn, United States, 2000.
- [33] I. Ulrich, J. Borenstein, VFH\*: local obstacle avoidance with look-ahead verification, Presented at International Conference on Robotics and Automation, San Francisco, United States, 2000.
- [34] D. Calisi, D. Nardi, Performance evaluation of pure-motion tasks for mobile robots with respect to world models, *Autonomous Robots* 27 (2009) 465–481.
- [35] D. Ferguson, T. Howard, M. Likhachev, Motion planning in urban environments: part I, Presented at International Conference on Intelligent Robots and Systems, Nice, France, 2008.
- [36] D. Ferguson, T. Howard, M. Likhachev, Motion planning in urban environments: part II, Presented at International Conference on Intelligent Robots and Systems, Nice, France, 2008.



**Michael Milford** received his B.E. degree in Mechanical and Space engineering in 2002 and Ph.D. degree in Electrical Engineering in 2006, both from the University of Queensland. He is currently a postdoctoral research fellow in the School of Engineering Systems at the Queensland University of Technology, having formerly worked as a Research Fellow at the Queensland Brain Institute. His research interests include biologically inspired robot mapping and navigation and the neural mechanisms underpinning navigation in animals.



**Gordon Wyeth** is a Professor of Robotics at the Queensland University of Technology. He holds a Ph.D. and a Bachelor of Engineering degree (with honours) in Computer Systems Engineering. He is the President of IEEE Control Systems, Robotics and Automation Queensland chapter, former president of the Australian Robotics and Automation Association and has served in various leadership positions in the RoboCup International Federation. He serves in various editorial positions for leading international robotics journals and conferences. Professor Wyeth's research receives funding through the Australian Research Council and other government and industry bodies. His team has designed and constructed more than 20 types of robots, including flying robots, wall-climbing robots, high performance wheeled robots, legged robots, manipulators and a humanoid robot. His robot soccer team, the RoboRoos, have been runners-up three times in the RoboCup World Cup of robot soccer.