Applications of an Adaptive Hierarchical Mobile Robot Navigation System

Christopher P. Lee-Johnson
Christopher.Lee-Johnson@vuw.ac.nz
Praneel Chand
Praneel.Chand@vuw.ac.nz
Dale A. Carnegie
Dale.Carnegie@vuw.ac.nz
Victoria University of Wellington
Wellington, New Zealand

Abstract

A hierarchical robot navigation system has been developed and implemented on a range of real and simulated mobile robots for two independent research applications. One application is a study on emotion-modulated navigation. The other is an investigation of heterogeneous multi-robot navigation. The navigation framework shared by the various implementations is described. A multi-layered architecture is employed that incorporates reactive control, deliberative path planning and exploration capabilities. The reactive controller utilises a two-stage optimisation in directional and velocity space. Path planning employs a probabilistic A* algorithm applied to a dynamically-updated occupancy grid. Implementation differences between applications include the path planning cost function, and certain aspects of the reactive control system that are affected by different robot shapes and drive systems. Some key results from both applications are presented, including experiments performed in simulation and in the real world.

1 Introduction

A diverse assortment of large-scale mobile robots have been developed by our research group in recent years, including MARVIN, a security and public relations robot [Carnegie et al., 2004], Itchy and Scratchy, a pair of identical tricycle robots [Carnegie et al., 2005], and Tank and Rubble-Bot, members of a fleet of robots intended for urban search-and-rescue applications [Williamson and Carnegie, 2006]. These robots are currently employed in two parallel studies.

The first is an investigation of the effects of artificial emotions on mobile robot navigation [Lee-Johnson and Carnegie, 2006; 2007]. A number of authors have argued that artificial emotions could improve the adaptive capabilities of robots and autonomous agents [Arkin, 2004]. Nevertheless, the majority of research in this area focuses on the social domain [Breazeal, 2004; Moshkina and Arkin, 2005; Broekens, 2007]. Most examples employ emotions as simple logic states that drive behaviour selection [Velásquez and Maes, 1997; Gadanho and Hallam, 2001; Tingley and Browne, 2006]. In contrast, we model emotions as modulations of an independent navigation system, influencing rather than driving the robot's decisions and actions. Thus, we require a robust navigation system that can function independently of external influences, but can also be adapted in real time to meet the requirements of the robot's changing emotional states.

The second study focuses on task allocation and coordination in heterogeneous multi-robot systems [Chand and Carnegie, 2007a; 2007b]. Task allocation mechanisms distribute tasks between different robots [Gerkey and Mataric, 2004]. Coordination mechanisms allow individual robots within a group to take each others' actions into consideration [Farinelli et al., 2004]. Successful task allocation and coordination strategies lead to increased efficiency in performing tasks and improved robustness to the failure of individual robots. This study requires a generic navigation system to be employed on robots with different physical capabilities.

A single navigation system has been developed and adapted to suit each of these applications. Although they are conducted on different types of robots, the requirements for both applications are similar. The multiple implementations and adaptive requirements necessitate a high degree of flexibility in our design. Mapping, path planning and reactive control capabilities are required in order for the robots to cope with initially-unknown dynamic environments. This paper describes our navigation system and its implementations for emotion-modulated navigation and heterogeneous multi-robot systems.

2 Navigation Architecture Overview

Robot architectures can be divided into three main categories: reactive, deliberative and hybrid systems. Reactive architectures (e.g. [Brooks, 1986]) tightly couple sensing and actuation, employing a minimalist approach to world representation. Brooks [1986] argues that the

This work was supported by the N.Z. Tertiary Education Commission under a Top Achiever Doctoral Scholarship.

world is its own best model, and complex behaviours can emerge from the interactions of simple components. Robots with reactive architectures tend to be fast and responsive to dynamic environments. However, there are many tasks that cannot be easily accomplished with a purely reactive system.

Deliberative architectures (e.g. [Kosaka and Kak, 1992]) attempt to plan a robot's actions based on predictions of their outcomes. The robot's world representation is continually updated to incorporate new data, and its actions are replanned accordingly. This allows a robot to solve certain problems more easily than the trial-and-error approach of reactive systems. Computational overheads are unbound, increasing exponentially as the quantity of world information increases. Thus, the design of a deliberative system always involves tradeoffs between speed and optimality. Purely deliberative robot architectures are rarely implemented, as they are generally too slow to cope with real-world dynamic environments.

Hybrid architectures (e.g. [Murphy and Arkin, 1992; Gat, 1992; Simmons, 1994]) combine reactive and deliberative approaches. These architectures have been increasingly popular in recent years, as they can combine the strengths and mitigate many of the weaknesses of single-level systems. The key issue of hybrid architecture design is managing the interactions between reactive and deliberative components.

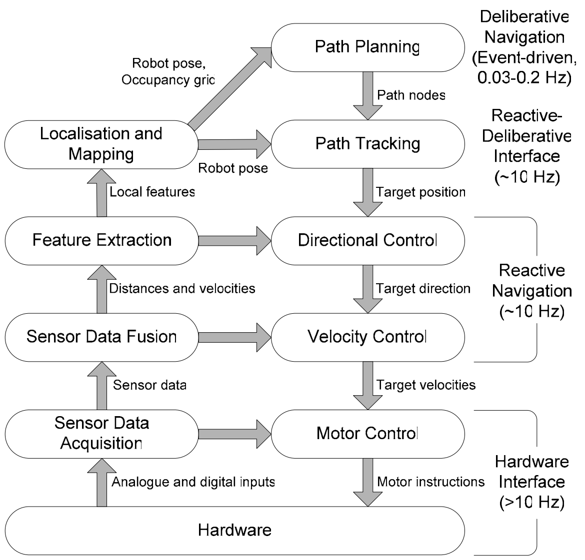

Figure 1: Block diagram of hierarchical navigation architecture. Components are sorted by abstraction level and update frequency along the vertical dimension. The horizontal dimension divides components by role: perception/representation (left) or action/ planning (right).

We employ a hierarchical hybrid approach, where the navigation task is decomposed into modules that incrementally add functionality (Figure 1). Perceptual information flows upwards through the hierarchy, becoming increasingly abstract and global. High level components are assigned a supervisory role, providing instructions that can be obeyed or ignored by lower level components as the situation dictates. This allows the architecture to benefit from the guidance of deliberative planning, while maintaining the real-time responsiveness of reactive control.

3 Reactive Navigation

Point-to-point reactive obstacle avoidance algorithms can generally be divided into two categories: directional approaches and velocity space methods.

Directional approaches such as the artificial potential field [Khatib, 1986] and vector field histogram methods [Borenstein and Koren, 1991] derive a target direction and speed based on external factors such as simulated forces or obstacle densities. Few of these techniques take a robot's kinematic or dynamic constraints into account, so the path followed only approximates the intended path. As a result, these approaches are generally unsuitable for navigation at high speeds or in dynamic environments.

Velocity space methods such as curvature velocity [Simmons, 1996] and dynamic window algorithms [Fox et al., 1997] select the robot's velocities directly from a set of those achievable given its kinematic and dynamic constraints, and safety considerations regarding obstacles. These approaches are better suited to high-speed obstacle avoidance in dynamic environments. However, they are often overly conservative, rejecting some velocities that would not necessarily result in a collision.

We attempt to mitigate the individual weaknesses of these approaches by combining elements of each. Our reactive controller employs a two-stage optimisation. First, an objective function $f_1(\theta)$ is applied to a polar histogram of candidate directions θ:

$$f_1(\theta) = W_1.a_1(\theta) + W_2.d_1(\theta) + W_3.i(\theta) \tag{1}$$

The candidate direction with the highest objective value is selected. This value is a linear weighted combination of angular error function $a_1(\theta)$, obstacle distance function $d_1(\theta)$ and angular inertia function $i(\theta)$. W_1, W_2 and W_3 are directional control weights.

Angular error function $a_1(\theta)$ gives preference to directions that are closer to the target direction θ_G. If the robot is following a planned path, θ_G is the direction of a point on the planned path that is distance d_L ahead of the closest point. Changing d_L influences how closely the robot adheres to the planned path. If planning is disabled, θ_G is instead the direction of the goal position.

Obstacle distance function $d_1(\theta)$ favours directions with more distant obstacles. For simplicity, the robot is represented as a point object, and each obstacle is represented by a circle of radius r_o. Typically, r_o is equal to the robot's radius plus some safety margin, but it can be altered in order to increase or decrease the strength of the robot's obstacle aversion drive. The points of intersection between each circular obstacle and a line extending out from the robot's position in direction θ are calculated. $d_1(\theta)$ is a function of the distance between the robot's position and the closest point of intersection, normalised by the maximum sensing range.

Angular inertia function $i(\theta)$ prefers smaller changes in direction, preventing the robot from oscillating between multiple directions that are otherwise equally favourable.

Next, a velocity couplet (v,ω) is selected that moves the robot in the intended direction at an appropriate speed. An optimization is performed within a discrete rectangular space bounded by the minimum and maximum linear and angular velocities achievable given the robot's current velocities, acceleration constraints and global limits. The selection process utilises the objective function $f_2(v,\omega)$, a linear weighted combination of angular error function $a_2(v,\omega)$, obstacle distance function $d_2(v,w)$ and speed function $s(v)$:

$$f_2(v,\omega) = W_4.a_2(v,\omega) + W_5.d_2(v,w) + W_6.s(v) \quad (2)$$

W_4, W_5 and W_6 are velocity control weights.

Angular error function $a_2(v,\omega)$ favours angular velocities that turns the robot towards the target heading θ_T, based on its predicted heading after one control period. $a_2(v,\omega)$ also favours smaller linear velocities if the angular error is large, since larger circular turns would result in increased deviation from the expected path.

Obstacle distance function $d_2(v,\omega)$ biases the controller towards smaller linear velocities if there are obstacles near the front of the robot, preventing potential collisions. It can also favour angular velocities that turn the robot away from detected obstacles.

Speed function $s(v)$ normally favours high linear velocities, but reduces the robot's speed as it approaches the target position.

4 Deliberative Navigation

All local obstacle avoidance techniques are to some degree susceptible to local minima. This problem can be alleviated by detecting and responding appropriately to local minima when they are encountered. However, convergence to the goal can only be guaranteed if they are combined with some form of global path planning.

Path planning typically involves the application of a search algorithm or heuristic to a graph structure representing a set of valid paths obtained from a map of the robot's environment. In cell-decomposition techniques, obstacles and free space are divided into discrete cells [Hwang and Ahuja, 1992]. Valid paths are obtained from the adjacency relationships between free cells. Selection of cell sizes is generally a trade-off between path quality and computational efficiency. Large environments can result in unacceptably large search spaces unless they are partitioned into manageable segments.

Roadmap methods such as visibility graphs [Nilsson, 1969], Voronoi diagrams [Aurenhammer, 1991], freeway nets [Latombe, 1991] and silhouette graphs [Canny, 1988] generate a set of curves in free space that connect nodes between obstacles. These techniques can greatly reduce the search space without impeding path quality, but the overhead of generating the roadmaps can negate any computational efficiency gains, and they are often suited to specific types of environments. For example, Voronoi diagrams perform well in confined indoor environments, but poorly in open outdoor environments.

Our map is represented by a rectangular occupancy grid [Thrun, 2003]. This was selected over more complex decompositions or roadmap methods because we require a simple representation that is functional in a wide range of environments.

Each node is assigned an occupancy probability p_o, a unit interval value that represents the estimated probability that the node is occupied by an obstacle. Occupancy probabilities are updated in real time based on proximities to sensor beams and obstacles. Each sensor is represented by a line that extends from its origin to the obstacle detected, bounded by a predefined maximum sensor range. Nodes that a sensor beam passes through are flagged as unoccupied, as are nodes currently occupied by the robot. Nodes containing detected obstacles are flagged as occupied, superseding any unoccupied status resulting from proximity to the robot or its sensor beams. Bayes' rule is employed to obtain p_o from these discrete occupancy measurements over time.

We employ the A* algorithm [Judea, 1984] to extract optimal paths from the occupancy information. A* is a best-first graph search algorithm that prioritises nodes by the estimated quality of their associated paths. The priority $f(x)$ of a node x is dependant on a measured cost $g(x)$ of the best path from the initial node to x, as well as a heuristic cost $h(x)$ of travel from x to the goal node:

$$f(x) = g(x) + h(x) \quad (3)$$

In standard A* path planning methods, nodes are assigned either occupied or unoccupied status, and all unoccupied nodes are weighted equally. We modify $g(x)$ to be continuous and dependant on $p_o(x)$. The exact form of this relationship varies between implementations.

The heuristic cost $h(x)$ is the Euclidian distance between node x and the goal node. This is an optimistic estimate; the actual cost is generally larger than the straight-line distance, especially once the additional costs of non-zero p_o values are taken into consideration. As a result, this approach is heavily biased towards optimality rather than computational speed. Nevertheless, search spaces are kept small enough to allow adequate real-time performance.

5 Emotion-Modulated Navigation

In recent years, many authors have argued that models of mind should incorporate not only "cold" cognition, but also "warm" affective states and processes such as emotions [Minsky, 1986]. The first application of the navigation system is an investigation of the effects of artificial emotions on MARVIN's navigation capabilities. Artificial emotions are implemented in our system as modulations of the robot's decisions and actions in response to certain situations that arise during its interaction with a dynamic environment.

Two levels of emotional interaction are modelled: reactive and deliberative. Reactive emotions modulate certain control parameters in response to appraisals based on short-term sensor data and local features. Emotion intensity values are damped, allowing emotions to persist for some time after their stimuli have abated. They are also subject to localised biases from deliberative emotions. Reactive fear modulates the global velocity limit v_L, limiting the robot's speed in constrained environments where collisions are likely, but allowing higher speeds in sparser environments. Reactive anger modulates the obstacle radius r_o, reducing the robot's obstacle aversion when its progress towards a goal becomes obstructed.

Deliberative emotions are associated with specific locations in the environment, biasing the robot's path planning. The measured cost $g(x)$ is a continuous variable dependant on the cost of the lowest-cost parent node x_{par}, the Euclidian distance $d_n(x,x_{par})$ between the two nodes, and the node's cost exponent $c_e(x)$:

$$g(x) = g(x_{par}) + d_n(x, x_{par}).B^{c_e(x)} \tag{4}$$

Base constant B is large enough that the robot is unlikely to plan a path through a node with a high cost when low-cost alternatives exist. However if no unobstructed paths exist, the planning algorithm fails gracefully by choosing the best of the unfavourable options available. The reactive controller prevents the robot from colliding with obstacles even if it is instructed to pass through them, and it can sometimes reactively navigate through nodes that the planner regards as occupied.

The cost exponent c_e is a unit interval variable subject to location-specific emotional biases:

$$c_e = p_o + \left(W_7 E_F + W_8 (1 - E_S) - W_9 E_H\right)(1 - E_A) \tag{5}$$

Deliberative fear E_F increases the cost of nodes in proximity to a collision, reducing the probability that the robot will subsequently plan a path through them. Surprise E_S reduces cost, compelling the robot to explore areas where its sensor data does not match its internal map data. Positive values of happiness E_H reduce cost, increasing the probability that the robot will plan a path that has resulted in previous success, whereas negative values (representing sadness) have the reverse effect. Emotional biases are suppressed by deliberative anger E_A, allowing the robot to escape from obstructed states resulting from these modulations. Planning weights W_7, W_8 and W_9 control the degree of influence that each emotion has over path planning.

This system is tested in a simulated office block environment (Figures 2-3). The environment is initially unmapped, and it contains fast-moving obstacles. Collisions are inevitable if the robot passes within an area occupied by dynamic obstacles, but the robot has no prior knowledge of them. Four passes are made through the environment; two in each direction.

When emotions are disabled (Figure 2), the robot lingers within dangerous areas occupied by dynamic obstacles, sustaining a significant number of collisions. It generally adheres to the first satisfactory path it finds, without searching for potential superior paths. If emotions are enabled (Figure 3), the robot quickly becomes “afraid” of dangerous locations and avoids them. Because the environment is initially unmapped, early paths yield a high level of surprise. This drives it to heavily favour exploration of the environment over exploitation of existing world knowledge. Only after it has explored the majority of the map does the robot converge on an optimal path. The large path times resulting from the exploration initially cause sadness, which increases the drive to find alternative paths even further. However, once the robot begins to travel more directly, the resulting lower path times yield happiness, which somewhat counteracts the effects of surprise by favourably biasing successful paths. The effects of reactive emotions are subtle, so they are largely eclipsed by deliberative emotions in this experiment. Nevertheless, the larger variance in speeds resulting from reactive fear does increase the robot’s tendency to converge on paths that pass through sparsely-occupied rooms rather than corridors.

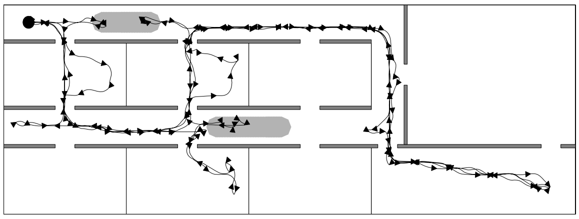

Figure 2: Example path with emotions disabled (v_L = 0.8 m/s, r_o = 0.35 m, W_7 = 0, W_8 = 0, W_9 = 0). Start and end position is marked by the black circle. Dynamic obstacles travel within the light grey areas.

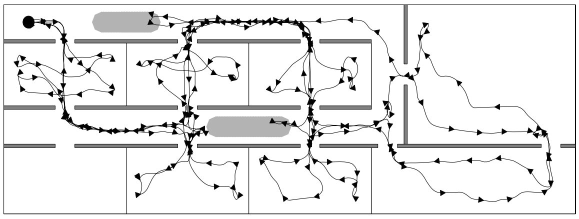

Figure 3: Example path with emotions enabled (v_L = 0.6→1.0 m/s, r_o = 0.35→0.45 m, W_7 = 0.8, W_8 = 0.35, W_9 = 0.2).

Table 1:
Performance comparison for emotion-modulated navigation

Performance parameter	Emotions disabled	Emotions enabled
Collision count	31 ± 7	14 ± 6
Exploration coverage	0.53 ± 0.04	0.76 ± 0.04
Path time (s)	730 ± 40	910 ± 70
Average velocity (m/s)	0.34 ± 0.02	0.33 ± 0.03

Results are averaged from 30 samples per condition. The presented uncertainties represent one standard deviation.

The quantitative results in Table 1 show that emotions reduce the average collision count, largely as a result of the fear response. The exploration coverage is greatly increased, mostly due to the actions of surprise and sadness. This increase comes at a cost to path time because the small number of passes through the environment per sample does not provide sufficient opportunity to recoup the initial time investment. Values and ranges for v_L were selected such that the average velocity would be approximately equal under both conditions, in order to minimize the bias to other performance measurements. However, upon performing the experiment, the average velocity is slightly higher when emotions are disabled, further widening the gap between path times.

Results from this and other experiments indicate that our implementation of artificial emotions can significantly improve a robot’s navigation performance in certain situations. Other results and further implementation details can be found in [Lee-Johnson and Carnegie, 2006; 2007]. Following its implementation on MARVIN, the navigation system has been applied to a heterogeneous multi-robot system.

6 Heterogeneous Multi-Robot Navigation

A key advantage in a multi-robot system is that a global task can be distributed across the team of robots. The second application of the navigation system is in a hierarchical and heterogeneous multi-robot system that comprises task manager and task executer robots. Computationally powerful task manger robots reside at the highest level of the hierarchy. They are responsible for maintaining global information and controlling an adaptive hierarchy of task executer robots. At the lower levels of the hierarchy, the task executer robots achieve the objectives of the global task.

The team of robots is required to perform a map-building and exploration task. Many of the task executer robots have limited processing and sensing capabilities. Thus, the map-building task is expressed as a set of multi-task robot single-robot task allocations, where the task executer robots can be planners and/or explorers. The global environment is divided into local environments based on the robots processing and sensing capabilities. Planners negotiate and allocate local environments for the explorers to explore. A two-tiered A* global path planning algorithm is employed by the planners to direct the explorers to their allocated local environments [Chand and Carnegie, 2007a].

Path planning within a local environment employs a slightly different approach from that utilised in the other application. Nodes whose occupancy probabilities exceed a threshold T_o are eliminated from the cost calculation. The cost $g(x)$ of all other nodes is linearly dependant on a cost multiplier c_m:

$$g(x) = \begin{cases} g(x_{par}) + d_n(x, x_{par}).c_m(x) & \text{if } p_o(x) < T_o \\ \infty & \text{otherwise} \end{cases} \tag{6}$$

The unit interval cost multiplier c_m is a function of the occupancy probability $p_o(x)$ of node x and the average of $p_o(y)$ for nodes y within a predefined robot clearance radius:

$$c_m(x) = 1 + W_{10}.p_o(x) + W_{11}.\overline{p_o(y)} \tag{7}$$

Weights W_{10} and W_{11} control the balance between the two inputs.

The multi-robot map-building and exploration task has been tested in a simulated environment with randomly distributed obstacles (Figures 4-5). Three heterogeneous robots are modelled: Itchy/Scratchy, a pentagon-shaped tricycle-drive robot, MARVIN, a circular-shaped differential-drive robot, and Tank, a rectangular-shaped differential-drive robot. The global environment is divided into local environments as shown by the dashed green lines. All robots are explorers and the rectangular robot is a planner. Collisions between robots are possible as the robots navigate towards their allocated local environments (Figure 4). The robots use reactive collision avoidance to successfully reach their allocated environments (Figure 5).

Once the robots are inside their allocated local environments they begin constructing a map of the local environment (Figure 5). A line scanning method that accounts for the sensing range of the robot is employed to direct the map-building and exploration. It allows the robot to plan a path from one end of the local environment to the other that favours unexplored space. A resource utilisation feedback mechanism that resides on the task manager robots can optimise the planning, sensing and actuation rates of the task executer robots as the multi-robot task progresses. Further implementation details and results of the multi-robot map-building and exploration task can be found in [Chand and Carnegie, 2007b].

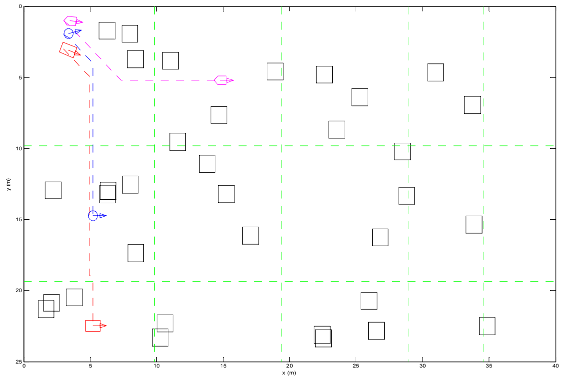

Figure 4: Three heterogeneous robots in the top left corner of the environment have been allocated local environments to explore.

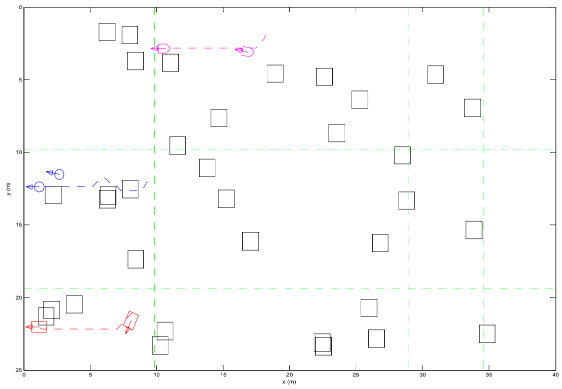

Figure 5: The robots are exploring and building a map of their local environment after successfully navigating to it.

At the directional control level, all robots are assumed to have a circular shape. However, this approximation is invalid for non-circular robots at the velocity control level. Representing such a robot as a circle would result in an imbalance in the level of clearance on different sides of the robot.

Integrating directional control and velocity control for reactive navigation requires modifications to a non-circular robot's reference frame, R (Figure 6). The physical centre of the robot is selected as the new reference point. The linear velocity v_1 and curvature radius r_1 are adjusted to this new reference before velocity control is employed. The y axis offset in R is y_{off} and the minimum curvature radius is $r_{1\min}$.

$$r = \begin{cases} \sqrt{r_1^2 + |y_{off}|^2}, r_1 \geq r_{1\min} \\ -\sqrt{r_1^2 + |y_{off}|^2}, r_1 \geq r_{1\min} \end{cases} \tag{8}$$

The linear velocity v is given by:

$$v = \frac{r}{r_1} v_1 = r\omega \tag{9}$$

The angle θ_v between the velocity vector v and the robot frame y axis is determined:

$$\theta_v = \sin^{-1}\left(\frac{|y_{off}|}{r}\right) \quad (10)$$

Angle θ_v varies for each (v,ω) pair. An axes rotation of the robot reference frame and obstacles is performed using θ_v so that the obstacle distance function is correctly determined.

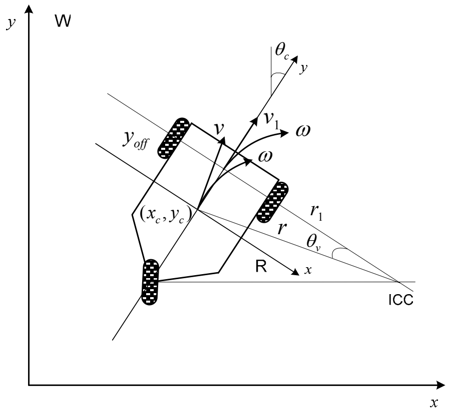

Figure 6: Modification of a non-circular robot's reference frame to allow compatibility between the directional and velocity controls.

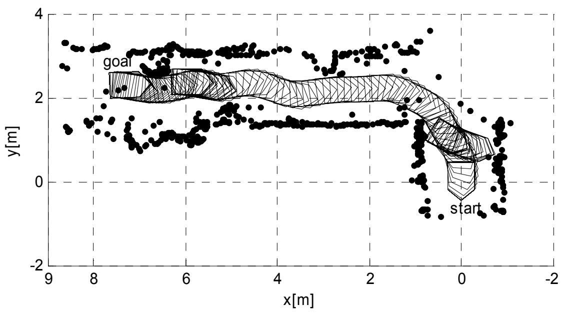

Figure 7: Tricycle robot reactive navigation test in office corridor at 0.3 m/sec.

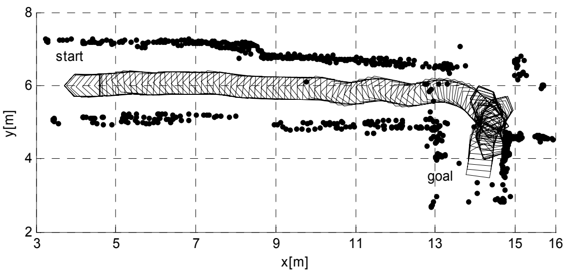

Figure 8: Tricycle robot deliberative and reactive navigation test in office corridor at 0.3 m/sec.

The hierarchical navigation system has been tested on Itchy and Scratchy, a pair of real-world tricycle robots (Figures 7-8). In the first test, the robot reactively performs a left turn in a corridor and moves past three rubbish bins to reach its goal (Figure 7). The second test combines both reactive and deliberative control levels (Figure 8). Despite the loss of localisation accuracy and actuator noise, the robot is able to reach its goal and avoid colliding with the corridor wall. Further implementation details and results can be found in [Chand and Carnegie, 2005].

7 Conclusion

The navigation system has been successfully employed for both applications. Simulation and real-world results have validated our approach. Nevertheless, there are a number of areas in which this research can be improved and extended.

Our real world results are currently limited to a single type of robot tested under constrained conditions. Further studies will investigate the system's real-world performance on other robots under less constrained conditions.

Localisation issues have not been fully considered in many of the simulation-based experiments. While sensor noise is currently modelled in simulation, localisation uncertainty is not. The system will therefore be extended to incorporate more robust localisation capabilities.

The navigation system has proven to be robust and flexible, as demonstrated by its implementation on multiple robots for different applications. Thus, it is anticipated that variations of the system will be employed in future research projects.

References

[Arkin, 2004] Ronald C. Arkin, Moving up the food chain: motivation and emotion in behavior-based robots. J. Fellous and M. Arbib, editors. *Who Needs Emotions: The Brain Meets the Robot*, pages 425–270. Oxford University Press, New York, 2005.

[Aurenhammer, 1991] Franz Aurenhammer. Voronoi diagrams: A survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, September 1991.

[Borenstein and Koren, 1991] Johann Borenstein and Yoram Koren. The vector field histogram – fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, June 1991.

[Breazeal, 2004] Cynthea Breazeal. Function meets style: Insights from emotion theory applied to HRI. *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*, 34(2):187–194, May 2004.

[Broekens, 2007] Joost Broekens. Emotion and reinforcement: Affective facial expressions facilitate robot learning. T. Huang, A. Nijholt, M. Pantic, A. Pentland, editors. *Artificial Intelligence for Human Computing*, *Lecture Notes in Artificial Intelligence, Volume 4451*, pages 113–132. Springer Berlin, Heidelberg, 2007.

[Brooks, 1986] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, March 1986.

[Canny, 1988] John F. Canny. *The Complexity of Robot Motion Planning*. MIT Press, Cambridge, Massachusetts, 1988.

[Carnegie et al., 2004] Dale A. Carnegie, Ashil Prakash, Chris Chitty and Bernard Guy. A human-like semi autonomous mobile security robot. *Proceedings of the Second International Conference on Autonomous Robots and Agents*, pages 64–69, Palmerston North, New Zealand, December 2004. Massey University.

[Carnegie et al., 2005] Dale A. Carnegie, Andrew D. Payne and Praneel Chand. The design of a pair of identical mobile robots to investigate cooperative behaviours. *Cutting Edge Robotics – Section V: Multi-Robot Systems*, pages 377–396. International Journal of Advanced Robotic Systems, Austria, 2005.

[Chand and Carnegie, 2005] Praneel Chand and Dale A. Carnegie. Reactive control of a tricycle mobile robot. *Proceedings of the Twelfth Electronics New Zealand Conference*, pages 129–134, Manukau, New Zealand, November 2005. Manukau Institute of Technology.

[Chand and Carnegie, 2007a] Praneel Chand and Dale A. Carnegie. Memory-time tradeoffs in a path planning approach utilising limited memory robots. *Proceedings of the International Conference on Computational Intelligence, Robotics and Autonomous Systems*, Palmerston North, New Zealand, November 2007. Massey University (to be published).

[Chand and Carnegie, 2007b] Praneel Chand and Dale A. Carnegie. Task allocation and coordination for limited capability mobile robots. *Proceedings of the Australasian Conference on Robotics and Automation*, Brisbane, Australia, December 2007. Australian Robotics and Automation Association (submitted for publication).

[Farinelli et al., 2004] Alessandro Farinelli, Luca Iocchi and Daniele Nardi. Multirobot systems: a classification focused on coordination. *IEEE Transactions on Systems, Man and Cybernetics*, 34(5):2015–2028, October 2004.

[Fox et al., 1997] Dieter Fox, Wolfram Burgard and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4(1):23–33, March 1997.

[Gadanho and Hallam, 2001] Sandra C. Gadanho and John Hallam. Robot learning driven by emotions. *Adaptive Behavior*, 9(1):42–64, March 2001.

[Gat, 1992] Erann Gat. Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 809–815, July 1992. Amercian Association for Artificial Intelligence Press.

[Gerkey and Mataric, 2004] Brian P. Gerkey and Maja J. Mataric. A formal analysis and taxonomy of task allocation in multi-robot systems. *International Journal of Robotics Research*, 23(9):939–954, September 2004.

[Hwang and Ahuja, 1992] Yong K. Hwang and Narendra Ahuja. Gross motion planning: a survey. *ACM Computing Surveys*, 24(3):219–291, September 1992.

[Khatib, 1986] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, March 1986.

[Kosaka and Kak, 1992] Akio Kosaka and Avi C. Kak. Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties. *Computer Vision, Graphics, and Image Processing – Image Understanding*, 56(3):271–329, November 1992.

[Latombe, 1991] Jean-Claude Latombe. *Robot Motion Planning*, Kluwer Academic Publishers, Boston, Massachusetts, 1991.

[Lee-Johnson and Carnegie, 2006] Christopher P. Lee-Johnson and Dale A. Carnegie. Towards a computational model of affect for the modulation of mobile robot control parameters. *Proceedings of the Third International Conference on Autonomous Robots and Agents*, pages 367–372, Palmerston North, New Zealand, December 2006. Massey University.

[Lee-Johnson and Carnegie, 2007] Christopher P. Lee-Johnson and Dale A. Carnegie. Emotion-based parameter modulation for a hierarchical mobile robot planning and control architecture. *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, California, October–November 2007. Institute of Electrical and Electronics Engineers (to be published).

[Moshkina and Arkin, 2005] Lilia Moshkina and Ronald C. Arkin. Human perspective on affective robotic behavior: A longitudinal study. *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1444–1451, Edmonton, Canada, August 2005. Institute of Electrical and Electronics Engineers.

[Murphy and Arkin, 1992] Robin R. Murphy and Ronald C. Arkin. SFX: An architecture for action-oriented sensor fusion. *Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robotics and Systems*, volume 2, pages 1079–1086, Raleigh, North Carolina, July 1992. Institute of Electrical and Electronics Engineers.

[Nilsson, 1969] Nils J. Nilsson. A mobile automaton: An application of artificial intelligence techniques. *Proceedings of the First International Conference on Artificial Intelligence*, pages 509–520, Washington D.C., 1969.

[Simmons, 1994] Reid G. Simmons. Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, 10(1):34–43, February 1994.

[Simmons, 1996] Reid Simmons. The curvature-velocity method for local obstacle avoidance. *IEEE International Conference on Robotics and Automation*, pages 3375–3382, Minneapolis, Minnesota, April 1996. Institute of Electrical and Electronics Engineers.

[Thrun, 2003] Sebastian Thrun, Robotic mapping: A survey. G. Lakemeyer and B. Nebel, editors. *Exploring Artificial Intelligence in the New Millennium*, pages 1–35. San Francisco, Morgan Kaufmann, California, 2003.

[Tingley and Browne, 2006] Chris S. Tingley and Will N. Browne. Developing an emotion-based architecture for autonomous agents. *Proceedings of the Third International Conference on Autonomous Robots and Agents*, pages 225–230, Palmerston North, New Zealand, December 2006. Massey University.

[Velásquez and Maes, 1997] Juan D. Velásquez and Pattie Maes. Cathexis: A computational model of emotions. *Proceedings of the First International Conference on Autonomous Agents*, pages 518–519, California, 1997. Marina del Rey.

[Williamson and Carnegie, 2006] David A. Williamson and Dale A. Carnegie. Embedded platform for search and rescue applications. *Proceedings of the Third International Conference on Autonomous Robots and Agents*, pages 373–378, Palmerston North, New Zealand, December 2006. Massey University.