Res. Lett. Inf. Math. Sci., 2011, Vol. **15**, pp. 21–34
Available online at http://iims.massey.ac.nz/research/letters/

[1]

A Reconfigurable Hybrid Intelligent System for Robot Navigation

N.H. Reyes, A.L.C. Barczak, Fatahillah and T. Susnjak
IIMS, Massey University, Auckland, New Zealand

Soft computing has come of age to offer us a wide array of powerful and efficient algorithms that independently matured and influenced our approach to solving problems in robotics, search and optimisation. The steady progress of technology, however, induced a flux of new real-world applications that demand for more robust and adaptive computational paradigms, tailored specifically for the problem domain. This gave rise to hybrid intelligent systems, and to name a few of the successful ones, we have the integration of fuzzy logic, genetic algorithms and neural networks. As noted in the literature, they are significantly more powerful than individual algorithms, and therefore have been the subject of research activities in the past decades. There are problems, however, that have not succumbed to traditional hybridisation approaches, pushing the limits of current intelligent systems design, questioning their solutions of a guarantee of optimality, real-time execution and self-calibration. This work presents an improved hybrid solution to the problem of integrated dynamic target pursuit and obstacle avoidance, comprising of a cascade of fuzzy logic systems, genetic algorithm, the A* search algorithm and the Voronoi diagram generation algorithm.

Keywords: robot navigation, cascade of fuzzy logic systems, hybrid fuzzy A*, genetic algorithm

1 Introduction

Getting robots immersed with our daily lives symbiotically is the grand challenge of robotics since its early beginnings. However, in order to achieve this feat, robust autonomous robot motion is among one of the many necessary precursors. This area of research alone spans multi-disciplinary fields, such as artificial intelligence, computer vision and machine learning, to name a few. Therefore, in order to decompose the challenge into one that is practicably feasible in scope, interesting robotics research platforms were introduced to the scientific community. One of them is the robot soccer game, a popular robotics platform that fosters artificial intelligence research in computer vision [1, 2], robot navigation with target pursuit, obstacle avoidance, opponent evasion, game strategy and cooperative multi-agents. This is the benchmarking testbed of choice for the algorithms presented herewith, and we present navigation simulations that adhere to the platform specifications defined by the Federation of Robot Soccer Association (FIRA) [3].

There are many design issues that arise in developing an autonomous robot navigation system. Firstly, depending on the size and complexity of the exploratory space, the time constraints imposed by the environment, and the requirement of path optimality, robot navigation may be implemented using a fast reactive control algorithm [4], or a path-planning algorithm [5]. Secondly, these algorithms may rely on the availability of a global map of the exploratory space, or simply on local information available from the neighbourhood by direct interaction with the objects in the vicinity.

Reactive methods are usually extremely fast, requiring relatively low memory space cost. They require access only to information within the neighbourhood, or some short history of sensor

[1] Author for Correspondence: n.h.reyes@massey.ac.nz

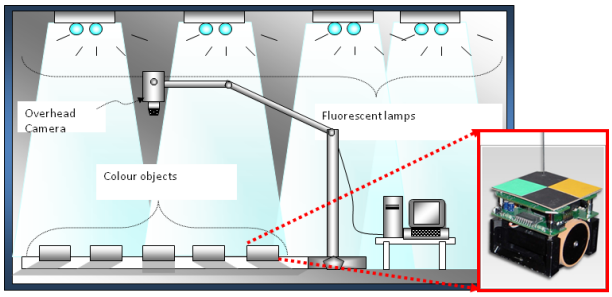

Figure 1: Robot Soccer System.

readings. Furthermore, they do not plan for a continuous path from start to the goal, allowing for real-time execution. However, in general, the drawback for these algorithms is that they could easily fall prey to an entrapment landscape, such as a U-shaped formation of barricades, or even closely-spaced obstacles. It has been reported that they also produce oscillatory behaviours in narrow corridors [6]. In Section 3, a fast reactive control algorithm is presented, comprising of a cascade of fuzzy logic systems for target pursuit and obstacle avoidance behaviours. Two fuzzy systems work together per robot behaviour. The target pursuit behaviour relies on a fuzzy system that calculates the precise steering angle for pursuit at any given neighbourhood configuration, and another fuzzy system that calculates the exact speed adjustment to hit the target accurately. The obstacle avoidance behaviour requires two fuzzy systems for similar reasons.

Path-planning algorithms may guarantee returning an optimal path from start to the goal, provided that it is given access to a complete global map of the scene of traversal. If the size of the exploratory space is not exceedingly large to be modeled (to fit into memory for searching), then path-planning algorithms will succeed in finding an optimal path if one exists. However, it still may be too slow for real-time execution. In Section 5.1, we propose using the A* algorithm as it guarantees returning the optimal path, and is the fastest optimal search algorithm. A* is adapted to the robot soccer domain of path-planning by either discretising the exploratory space into fixed-sized grids (Section 5.2), or utilising the Voronoi diagram (Section 5.3) for filtering the candidate set of points for path-planning. Lastly, this work proposes an extension to the works presented in [7, 8]. The previous works, as well as the extensions are all discussed in Sections 5.2 and 5.3 respectively, integrating a reactionary method with path-planning to get the best of both worlds. The hybrid system is comprised of a cascade of fuzzy systems, the A* algorithm and a Voronoi diagram generator algorithm. A* serves as the high-level path-planning algorithm that extracts a rough guide for the fuzzy systems, in terms of waypoints towards the goal. In turn, the cascade of fuzzy systems work out the exact steering angle and speed for a smooth, near-optimal path towards the destination, in real-time. These proposed algorithms are presented with accompanying simulation results to give more insights onto the individual system components, and as a complete hybrid system. System calibration techniques are also proposed and discussed.

2 The Proposed Hybrid Intelligent System Architecture

The architecture of the proposed hybrid intelligent system is highly modular and flexible. As depicted in Figure 2, there are line connections that appear dotted, indicating that they are optional, and therefore may be removed without crashing the system. In addition, some modules may actually be disengaged, depending on the desired robot behaviours, whether or not optimal path-planning is desired, or safety is taken as the highest priority. These options for system configuration are discussed further next.

The hybrid system combines together an optimal and deterministic search algorithm, called the

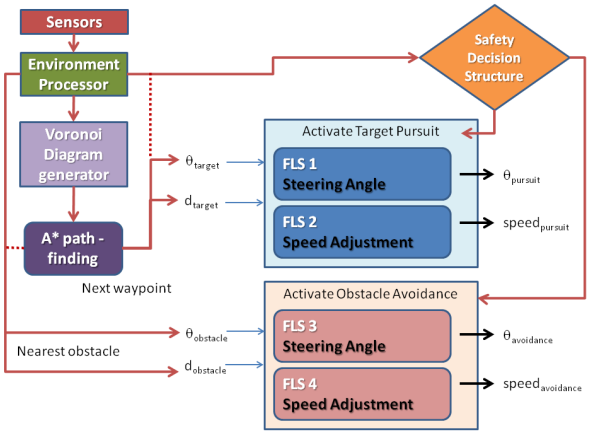

Figure 2: Proposed Hybrid System Architecture.

A* algorithm, a reactionary algorithm, comprising of a cascade of fuzzy systems, and a Voronoi diagram generator. The A* algorithm calculates a rough estimate of the path leading towards the destination in terms of waypoints. These waypoints are then fed to the fuzzy logic systems, which work out the exact steering angle and speed of movement of the robot for each step of the way (approximately every 30 msec.). As a result, the robot traverses a smooth and near-optimal path towards the destination, accelerating whenever the path is deemed safe and decelerating and avoiding obstacles whenever necessary, in real-time.

A more detailed understanding of the complete system could be obtained by referring to Figure 2. As depicted, (from the top) sensors take on information from the outside world and feeds them to an environment processor to find the position of the target destination and the position and orientation of the obstacles and the robot itself. In the FIRA robot soccer domain, Mirosot league [3], the sensor is an overhead camera (or two cameras) that captures an image of the bird's eye-view of the exploratory environment. On the other hand, a machine vision system serves as the environment processor that calculates the exact coordinates and orientation of all objects in the scene. This vision system is actually complex, as it needs to run in real-time a combination of several image processing algorithms, such as colour correction and classification algorithms [9], lens distortion correction, as well as labelling and robot recognition algorithms. All outputs coming from the environment processor go to the navigation algorithms which are described next.

The navigation algorithms in the system feed on the processed environment information and calculate whether it is safe to pursue the target, or if it is dangerously close to an obstacle already. If optimal path-planning is desired, the A* algorithm is engaged to calculate the next waypoint towards the target destination. A* is adapted to the problem of searching for the shortest path by discretising the exploratory space into a grid of fixed size cells for path searching. Only the next waypoint is returned by the A* algorithm, which is then received by the fuzzy system responsible for target pursuit. On the other hand, if safety is of top priority, the Voronoi diagram generator is executed first before A*. The Voronoi diagram generator calculates the safest traversable set of points that will be fed to the A* algorithm for path searching. Subsequently, the A* algorithm keeps the same role of calculating for the next waypoint and passing it to the fuzzy system for pursuit calculations.

If path optimality is not a requirement, the fuzzy systems are fed straight from the environment processor. One fuzzy system is for target pursuit calculation, while another is for obstacle avoidance. The fuzzy systems return the exact steering angle and robot speed which are eventually translated into left and right wheel velocities, or motor actions.

3 Cascade of Fuzzy Systems

The cascade of fuzzy systems (Figures 7 and 8) make up the reactionary part of the system, implementing the target pursuit and obstacle avoidance behaviours. This component of the system is able to deliver a quick and smooth response, without having the need of the complete mapping of all the objects in the scene of traversal. The only inputs required are the position and heading angle of the robot and the coordinates of the target and the nearest obstacle. Subsequently, all necessary angle and distance measurements are carried out from these inputs (Figure 3).

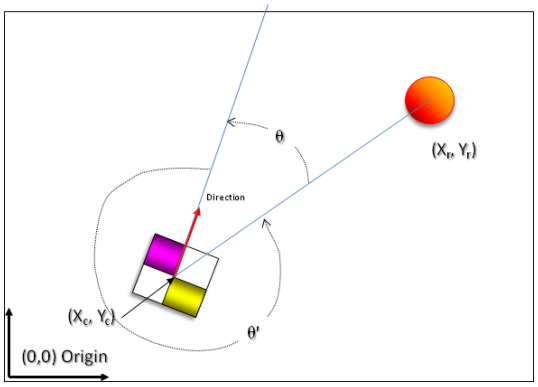

Figure 3: Angle and distance calculations.

Each robot behaviour is modeled as a combination of 2 fuzzy systems, one for steering angle computations, and another for speed adjustments. Therefore, for the 2 robot behaviours, there are 4 fuzzy systems that implement them. This section zeroes in on the details of the fuzzy system architecture.

3.1 Taking Advantage of the Presence of Symmetry

Traditional fuzzy system designs usually subdivide the entire range of input space, per parameter, into multiple overlapping subranges. For instance, for the input angle, the entire range of possible values for angle is subdivided into an odd number of overlapping subranges (usually 5 or 7 subranges). As can be seen in Figure 4(a), these subranges have their corresponding fuzzy sets (i.e. NL, NM, NS, ZE, PS, PM and PL).

On the contrary, the proposed fuzzy system design takes advantage of the presence of symmetry in the problem domain. For the input angle space, instead of covering the entire range of possible values, it considers only half of the space. As depicted in Figure 4(b), only 3 fuzzy sets are defined (i.e. small, medium, large), covering only the right-hand side of the robot. The other remaining half is calculated by simply negating the values of the same fuzzy sets assigned to the right-hand side. The negation operation is applied depending on the position of the target, relative to the robot's heading angle. If the target is on the left, feed the input angle to the same fuzzy sets defined for the right-hand side, but simply negate the result.

A negative value for the angle in this problem domain would mean turning the robot clock-wise; otherwise, if it's positive, it only means turning the robot counter clock-wise. A complete example of how this technique is utilised can be viewed in the fuzzy rules.

3.2 Target Pursuit

Figure 7 depicts the 2 fuzzy associative memory matrices detailing the rules for target pursuit, for calculating the steering angle and the speed adjustments. It can be seen that the A* algorithm

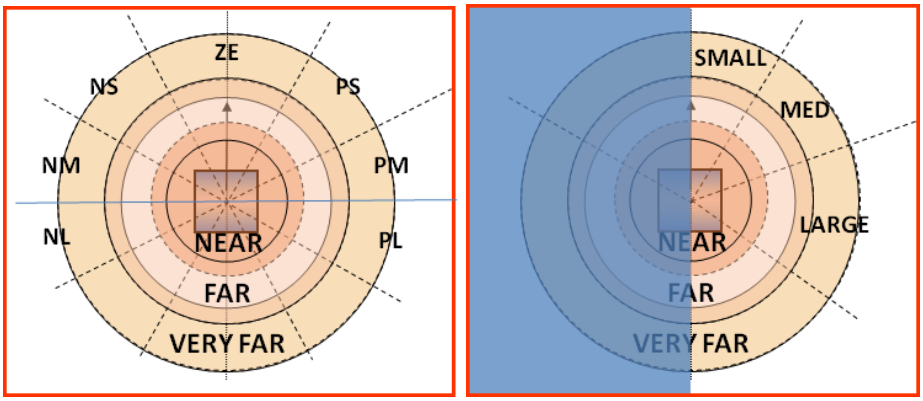

Figure 4: a) Traditional Fuzzy System Design. b) Proposed Fuzzy System Design. Legend: NL-Negatively Large, NM-Negatively Medium, NS-Negatively Small, ZE-Zero, PS-Positively Small, PM-Positively Medium and PL-Positively Large).

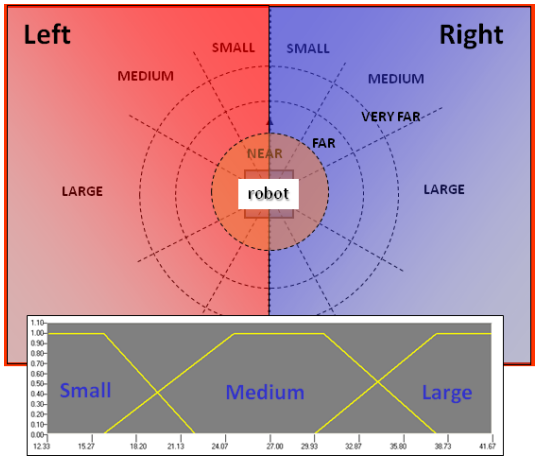

Figure 5: Fuzzy sets for the angles.

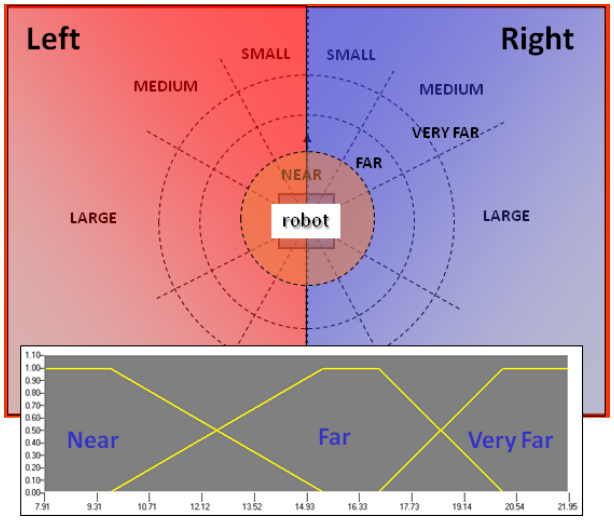

Figure 6: Fuzzy sets for the distances.

feeds the fuzzy systems with the coordinates of the next waypoint, along the optimal path towards the target destination. If A* is removed, the fuzzy systems would still function, provided that it feeds from the environment processor. However, the path calculated is no longer near-optimal and the robot may get trapped when obstacles form a U-shape.

To illustrate the idea behind the systems, a fuzzy rule for target pursuit reacts to the environment as follows:

```
if(Target is NEAR and angle from the Target is SMALL) then Make a Mild turn.
```

The rationale behind the fuzzy rule is that if the robot is about to hit the target, then it should make mild turns so as not to overshoot the target.

3.3 Obstacle Avoidance

The fuzzy systems for obstacle avoidance (Figure 8) only take into account the nearest obstacle during the course of traversal. Similar to target pursuit, 2 fuzzy systems are employed to deliver the obstacle avoidance behaviour, for angle and speed calculations. On the contrary, the systems are not dependent on the A* algorithm, but feeds directly from the environment processor.

To illustrate the idea behind the systems, a fuzzy rule for obstacle avoidance reacts to the environment as follows:

```
if(Obstacle is NEAR and angle from the Obstacle is SMALL) then Make a Very Sharp
turn.
```

The rationale behind the fuzzy rule is that if the robot is about to collide with an obstacle, then it should make a very sharp turn as quickly as possible to avoid it. A brief introduction to Voronoi diagrams follows next.

4 Voronoi Diagram

A Voronoi diagram is a representation of the results of a decomposition of a given space. The decomposition is called a Voronoi tessellation (also called a Dirichlet tessellation). The objective

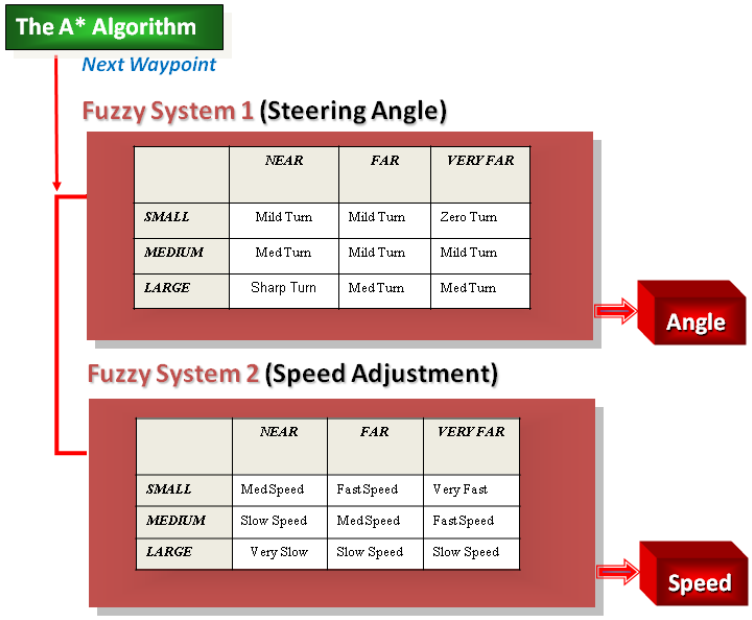

Figure 7: Fuzzy Logic System for Target Pursuit.

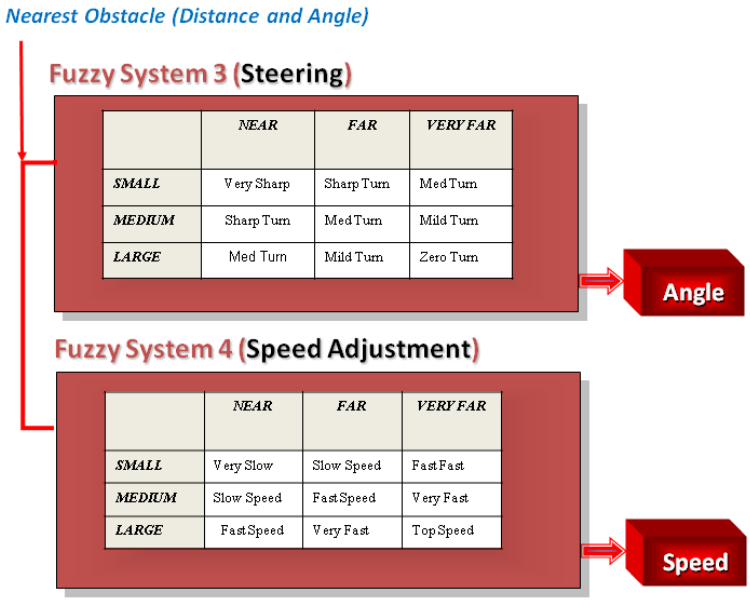

Figure 8: Fuzzy Logic System for Obstacle Avoidance.

of the decomposition operation is to represent specific geometric properties of the space, which lend themselves suitable for path determination (see for example [5, 10]).

The most common Voronoi diagram is the one where given a number of target points, the diagram, determines which points in the space are the closest to a certain given target point. In other words, which is the (infinite) set of points that are the closest to that target point, and are further away from all the other target points. The infinite set of points closest to a single point can be defined by a polygon.

There is a relationship between this type of Voronoi diagram and the Delaunay triangulation of the same set of points.

This type of Voronoi diagram can be used either to represent areas or to represent line segments that are equidistant between two target points. The diagram can help on path determination if the target points represent obstacles that should be avoided.

Consider that a Voronoi diagram is to be built in 2D and that there are 7 target points. Figure 9 shows the resulting Voronoi diagram, where the infinite lines where cut on the edges of the sub-area defined by the image.

There are several methods to build the Voronoi diagram. In this work, a modified implementation of Fortune's algorithm ([11]) was used. The algorithm has complexity $O(Nlog(N))$, which is usually fast enough for path determination purposes.

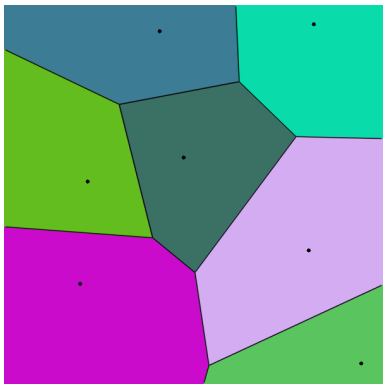

Figure 9: Sample Voronoi Diagram.

5 Variations of the Proposed Hybrid Fuzzy A* Algorithm

Depending on the preferred robot behaviour, the following configurations define whether or not optimal path planning will be sought by the robot, or safety will be prioritised on top of everything else.

5.1 Utilising Fixed Grid Cells: Pure A*

The A* algorithm alone could be used in lieu of the fuzzy systems. A* search could be employed for path-planning by discretising the exploratory space into a grid of fixed sized cells, then searching for the optimal path out of the grid cells (Figure 10(a)). A* guarantees returning an optimal path, but the smoothness of the path is highly affected by the granularity of the cells. However, the higher the granularity is, the longer path-planning takes, making it not suitable for real-time use for very large exploratory domains.

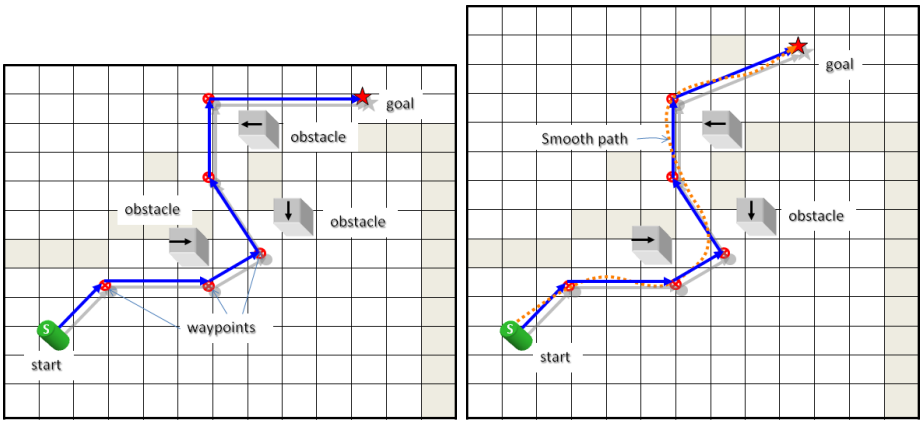

Figure 10: a) Waypoints generated by the A* algorithm. b) Smooth path generated by the Hybrid Fuzzy A* algorithm. Legend: red dots=waypoints; orange curve=smooth path.

5.2 Utilising Fixed Grid Cells: Hybrid Fuzzy A*

By combining fuzzy logic and the A* algorithm, the granularity of the grid cells could be made lower (i.e. cell size set approximately equal to the width of the robot); thus, making the search problem more significantly tolerable for computations. As illustrated in Figure 10(b), the combination allows the system to return a smooth, near-optimal path towards the destination, in real-time.

5.3 Utilising the Voronoi Diagram: Hybrid Fuzzy A*

Safety could be set as the top priority of the path-planning algorithms by utilising a Voronoi diagram generator in conjunction with the A* algorithm. The Voronoi diagram generator returns a set of traversable points that serve as inputs to the A* algorithm for searching a path. Due to the nature of the distribution of points in the Voronoi diagram, the path generated always guarantee that it is always as far as possible from any obstacles.

6 Experiments

Calibration experiments and the ultimate test on optimal path planning are discussed in this section, detailing how the fuzzy systems were fine-tuned to exhibit the target pursuit and obstacle avoidance behaviours.

6.1 Target Pursuit Calibration

Fuzzy logic systems are known to have relatively sensible control response, even without undergoing intensive calibration, provided that the fuzzy rules reflect a good approximation of an expert's knowledge about controlling the system. Nevertheless, for a cascade of fuzzy systems that are tied up together to form a more complex system, systems calibration could be a daunting task. We address this problem by presenting a new methodology utilising calibration maps. Firsly, 2 fuzzy systems (Section 3.2) are calibrated for target pursuit behaviour. Once this is done, the remaining 2 fuzzy systems for obstacle avoidance (Section 3.3) are calibrated while the target pursuit fuzzy systems simultaneously operate.

Figure 11 shows the calibration map used for fine-tuning the fuzzy system parameters (fuzzy rules and fuzzy membership functions) for target pursuit. At the centre of the figure is the target destination, a yellow object marked with an "X". The map is populated with a multitude of robot

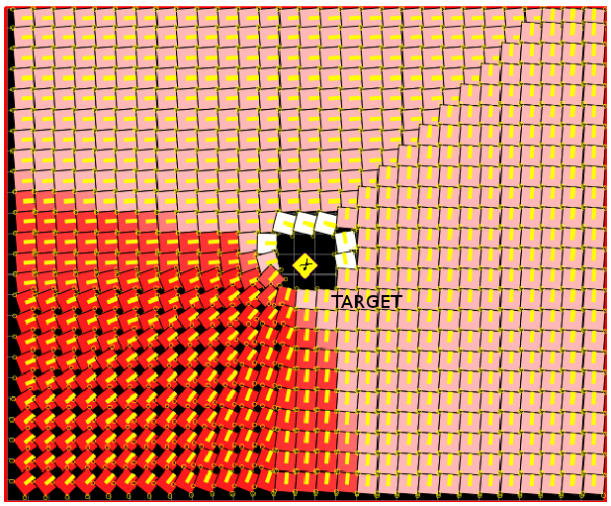

Figure 11: Target pursuit calibration initial set-up. Legend: yellow object with "X" mark = target; red, pink and white objects = robots employing target pursuit

instances surrounding the target. The colour of the robots intensify (turns more red) as its speed increases. It can be observed that robots pointing directly to the target have faster speeds than those oriented away from it. The white coloured robots indicate that they are moving very slowly, and from the figure, it can be viewed that they are almost oriented perpendicular to the target. Figures 12(a-to-e) depict the sequence of robot movements pursuing the stationary target object at the centre, from time=0 to t=36. It is evident from Figures 12(a-to-c) which robots accelerate, just by inspecting the change of their colours.

6.2 Target Pursuit with Obstacle Avoidance Calibration

It is a necessary precursor that the fuzzy systems implementing the obstacle avoidance behaviour be only calibrated after the fuzzy systems for target pursuit are completely fine-tuned. In addition, it is adamant that obstacle avoidance be calibrated while the finely-tuned fuzzy target pursuit systems are engaged. During the calibration process, it is important to check if the behaviours are switching properly according to the environment conditions.

Figures 13(a-to-g) show the calibration maps used for the combined target pursuit and obstacle avoidance behaviours. From the figures, it can be observed that there are 7 green obstacles that form barricades around the violet target object at the centre of the space. At time t=0, the robots are all initialised with the same heading angle of 45 degrees. At time t=2, robots start engaging their fuzzy target pursuit and obstacle avoidance systems. Therefore, all robots pointing towards the target, but within some safe distance from any of the obstacles have significantly increased their speed (indicated by their reddish colour). On the other hand, robots close to the obstacles, or heading towards any of the obstacles slow down their speed very cautiously (as indicated by their whitish colour).

From the progression of target pursuit and obstacle avoidance movements, it can also be observed from the figures that almost all the robots eventually reach their target destination, except for two robots that got stuck at Figure 13(g). This problem is addressed by using the A* algorithm to provide the waypoints for the fuzzy systems. A more challenging scenario is provided to illustrate the efficacy of the combined algorithms in Section 6.3.

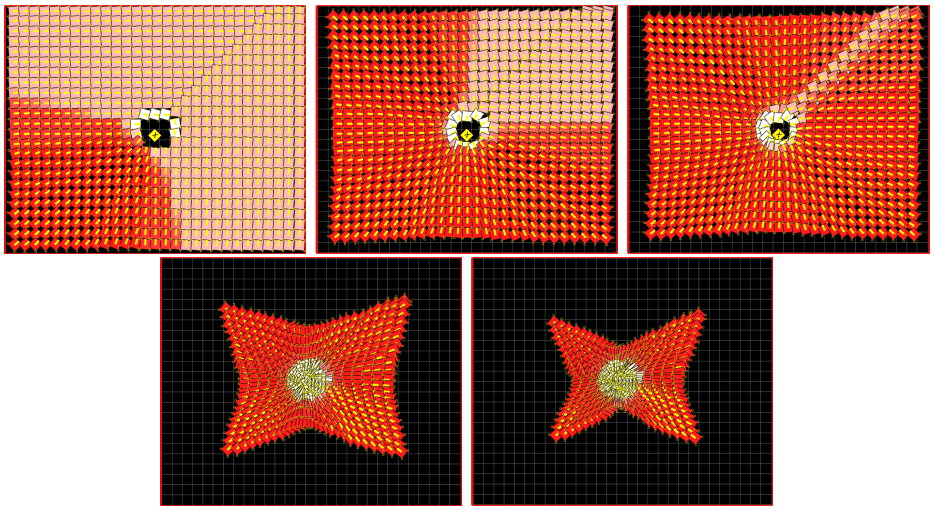

Figure 12: Target pursuit calibration map at different time sequences: a) At t=0. b)At t=5. c) At t=7. d) At t=28. e) At t=36. Legend: yellow object with "X" mark = target; red, pink and white objects = robots employing target pursuit

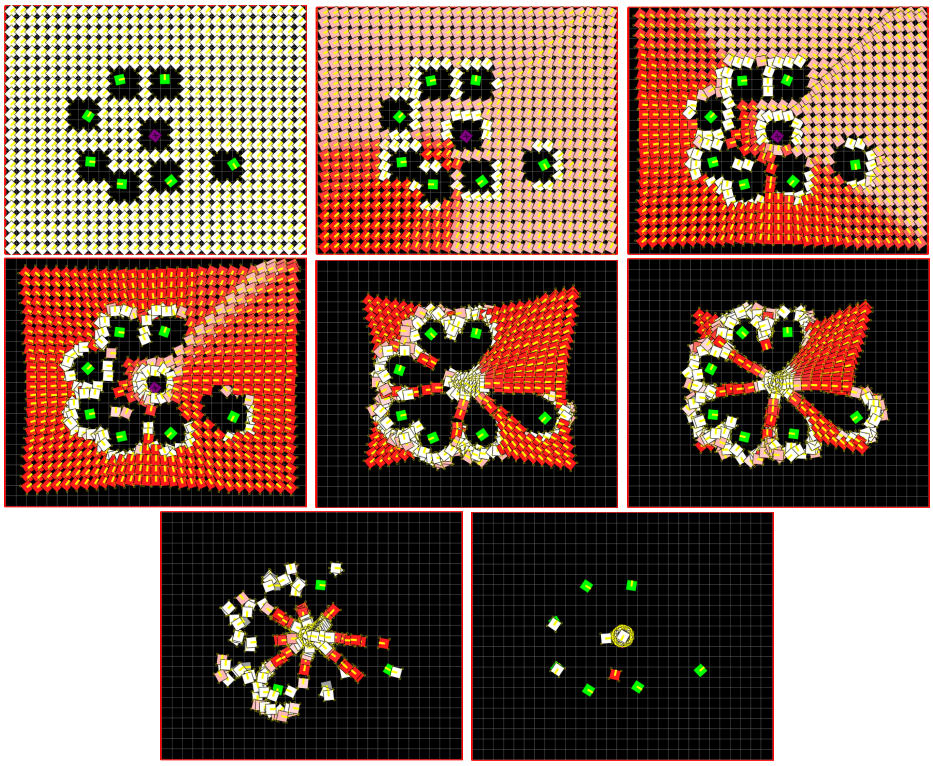

Figure 13: Target pursuit with obstacle avoidance calibration maps at different time sequences: a) At t=0. b)At t=2. b)At t=1. c)At t=5. d) At t=9. e) At t=24. f) At t=33. f) At t=72. g) At t=283. Legend: green objects=obstacles; violet object marked with an 'X'=target destination; white, pink and red objects = robots; Speed colour coding: white=slow speed; pink=medium speed; red=fast speed.

6.3 Ultimate Navigation Test: Escape from Local Minima

The complete robot navigation system is tested on a U-shaped enclosure to see if it can escape from the trap with near-optimal path-finding results. The test involves a moving target, with static obstacles, but only for brevity of discussions. The system actually works adaptively even on dynamic obstacles, but this is not shown in the figures. Figure 14 shows the important objects taken into account by the navigation system: the starting position of the robot (red object), the target destination (red dot, connected to the robot by a green line), and the closest obstacle (connected to the robot by a yellow line). The hybrid system works out the angles and distance values from these objects (for use by the fuzzy systems), as well as a discretised grid map of the world (for A* path-finding).

The actual sequence of robot movements are demonstrated by Figures 15(a-to-d). Firstly, the hybrid system engages the A* algorithm to calculate the optimal path. This optimal path is marked by yellow-coloured cells. Next, the cascade of fuzzy systems takes the next waypoint from the optimal path, then calculates the exact robot steering angle and speed, cautiously avoiding any intervening obstacles. The generated smooth path is marked by white dots for analysis (trail marks of the robot), as can be viewed from Figures 15(b-to-d). As evident from the results in the last few figures, the robot is able to escape from the trap with a smooth path closely matching the optimal path.

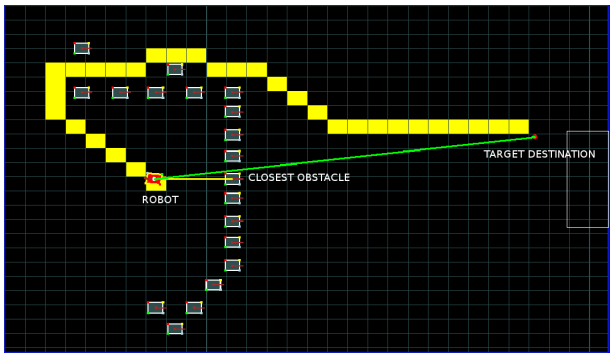

Figure 14: Escape from local minima, starting position. Legend: red object=robot; dark gray objects = obstacles; yellow blocks = waypoints generated by A*

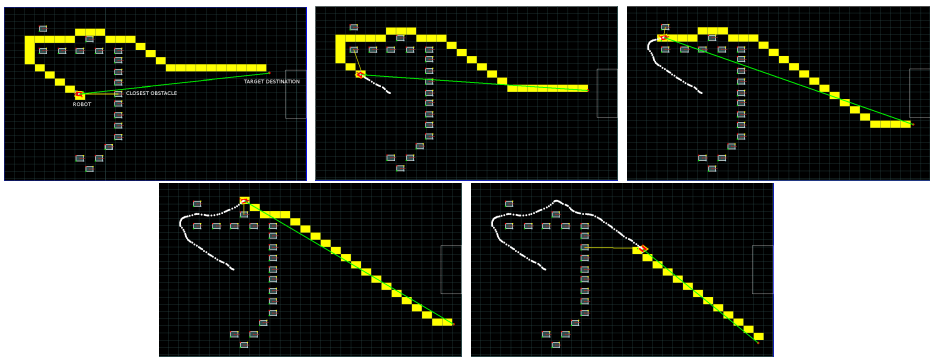

Figure 15: Escape from local minima test, at different time sequences: a) At t=0. b)At t=40. b)At t=117. c)At t=170. d) At t=270. Legend: red object=robot; dark gray objects = obstacles; yellow blocks = waypoints generated by A*

7 Safety first: Hybrid Fuzzy A* using the Voronoi Diagram

A variation of the hybrid fuzzy A* algorithm involves the utilisation of a Voronoi diagram. As can be seen from Figure 16(a), the diagram is calculated first using the obstacles as target points, thereby returning line segments that are equidistant between two obstacles.

Figure 16(b) depicts the utilisation of the Voronoi diagram by A*. As indicated by the yellow circles, the vertices in the diagram are extracted, then fed to the A* algorithm for path-planning. Subsequently, A* returns the safest path to traverse, and the cascade of fuzzy systems calculates the actual steering angle and speed of the robot as before.

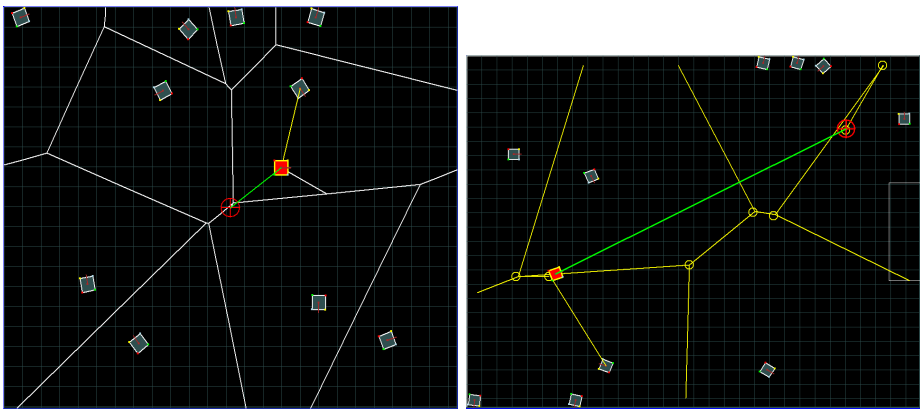

Figure 16: a) The Voronoi diagram calculated based on the obstacles in the simulated robot soccer scene. b) Robot navigation using the hybrid fuzzy A* with the Voronoi diagram. Legend: robot = red object; target destination = red cross hair, obstacles = dark gray objects

8 Conclusions

This work proposes a reconfigurable hybrid intelligent system, consisting of 4 fuzzy systems, the A* algorithm and a Voronoi diagram generator, for real-time, near-optimal robot navigation with target pursuit and obstacle avoidance behaviours. The integration architecture follows a modular approach, allowing parts of the system to be engaged or disengaged while still allowing the system to work.

The cascade of fuzzy logic systems for target pursuit and obstacle avoidance comprise the quick reactionary part of the hybrid architecture, calculating the exact steering angle and speed adjustment of the robot for a smooth navigation. On the other hand, the A* algorithm guarantees returning an optimal result when it is allowed to operate on a fixed sized grid and returns the safest path when allowed to operate on the vertices of a Voronoi diagram.

References

[1] Y.-C. Chang and B.-S. Chen, "Intelligent robust tracking controls for holonomic and nonholonomic mechanical systems using only position measurements," *Fuzzy Systems, IEEE Transactions on*, vol. 13, no. 4, pp. 491 – 507, aug. 2005.

[2] Y.-C. Chang and B.-S. Chen, "Robust tracking designs for both holonomic and nonholonomic constrained mechanical systems: adaptive fuzzy approach," *Fuzzy Systems, IEEE Transactions on*, vol. 8, no. 1, pp. 46 –66, feb 2000.

[3] J.-H. Kim. Federation of Robot-soccer Association. 2348 Undergraduate Building 2 Korea Advanced Institute of Science and Technology (KAIST) 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Korea. [Online]. Available: http://www.fira.net/

[4] A. Jayasiri, G. Mann, and R. Gosine, "Behavior coordination of mobile robotics using supervisory control of fuzzy discrete event systems," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 41, no. 5, pp. 1224 –1238, oct. 2011.

[5] S. Garrido, L. Moreno, D. Blanco, and P. Jurewicz, "Path planning for mobile robot navigation using voronoi diagram and fast marching," *International Journal of Robotics and Automation*, vol. 2, pp. 154–176, 2011.

[6] D. Calisi, "Mobile robots and vehicles motion systems: a unifying framework," Ph.D. dissertation, Sapienza Universita di Roma, 2009.

[7] A. Gerdelan and N. Reyes, "A novel hybrid fuzzy a star robot navigation system for target pursuit and obstacle avoidance," in *1st Korean-New Zealand Joint Workshop on Advance of Computational Intelligence Methods and Applications*, February 2006, pp. 75–79.

[8] A. Gerdelan and N. Reyes, "Synthesizing adaptive navigational robot behaviours using a hybrid fuzzy a star approach," in *Computational Intelligence, Theory and Applications*, B. Reusch, Ed. Springer Berlin Heidelberg, 2006, pp. 699–710.

[9] N. Reyes and E. Dadios, "Dynamic colour object recognition," *Journal of Advanced Computational Intelligence*, vol. 8, pp. 29–38, 2004.

[10] L. Vachhani, A. Mahindrakar, and K. Sridharan, "Mobile robot navigation through a hardware-efficient implementation for control-law-based construction of generalized voronoi diagram," *Mechatronics, IEEE/ASME Transactions on*, vol. 16, no. 6, pp. 1083 –1095, dec. 2011.

[11] S. Fortune, *Euclidean Geometry and Computers*. World Scientific Publishing Co., 1992, ch. Voronoi diagrams and Delaunay triangulations, pp. 193–233.