A Genetic Algorithm Based Architecture for Evolving Type-2 Fuzzy Logic Controllers for Real World Autonomous Mobile Robots

Christian Wagner, Member, IEEE and Hani Hagras, Senior Member, IEEE

Abstract— The type-2 Fuzzy Logic Controller (FLC) has started to emerge as a promising control mechanism for autonomous mobile robots navigating in real world environments. This is because such robots need control mechanisms such as type-2 FLCs which can handle the large amounts of uncertainties present in real world environments. However, manually designing and tuning the type-2 Membership Functions (MFs) for an interval type-2 FLC to give a good response is a difficult task. This paper will present a Genetic Algorithm (GA) based architecture to evolve the type-2 MFs of interval type-2 FLCs for mobile robots that will navigate in real world environments. The GA based system converges after a small number of iterations to type-2 MFs which give a very good performance. We have performed a series of real world experiments in which the evolved type-2 FLCs controlled a real robot in an outdoor arena. The evolved type-2 FLCs dealt with the uncertainties present in the real world to give a very good performance that has outperformed their type-1 counterparts as well as the manually designed type-2 FLCs.

I. INTRODUCTION

Controlling a mobile robot to navigate autonomously in real world outdoor unstructured environments (i.e. environments that have not been specifically engineered for the robot) is a challenging and difficult task. One of the main challenges facing robot controllers in such environments is handling the large amounts of uncertainties and imprecision present in real world environments. Such uncertainties can have many sources, including:

• Uncertainties in inputs to the controller as the sensors’ measurements are typically noisy and affected by the conditions of observation (i.e. their characteristics are changed by the environmental conditions such as wind, sunshine, humidity, rain, etc.).

• Uncertainties in control actions. Such uncertainties can result from the change of the actuator’s characteristics which can be due to the inconsistency of the terrain or due to the environmental changes. For example, a “fast wheel speed” on a sunny day with dry ground may be different from “fast wheel speed” on a rainy day with muddy ground as wheels may slip, etc. Another source of uncertainty in the control actions is caused by changes of the mobile robot’s physical properties such as changing of the diameter of the wheels, loosening of drive belts which could be due to wear and tear, etc.

• Linguistic uncertainties as the meaning of words can be uncertain as words mean different things to different designers [12], [13].

• Uncertainties associated with the use of noisy training data that could be used to learn, tune, or optimise the controller.

The Fuzzy Logic Controller (FLC) is credited with being an adequate methodology for designing robust controllers that are able to deliver a satisfactory performance in face of uncertainty and imprecision [15]. As a result the FLC has become a popular approach to reactive mobile robot control in recent years [15], [16].

While traditionally type-1 FLCs have been employed widely in robot control, it has become apparent in recent years that the type-1 FLC cannot fully handle the high levels of uncertainties present as its Membership Functions (MFs) are in fact completely crisp [7], [13]. The linguistic and numerical uncertainties associated with dynamic unstructured environments cause problems in determining the exact and precise MFs during the robot FLC design [7]. Moreover, the designed type-1 fuzzy sets can be sub-optimal under specific environmental and operational conditions. However, the environmental changes and the associated uncertainties usually require the continuous tuning of the type-1 MFs as otherwise the type-1 FLC performance can deteriorate [7]. As a consequence, research has started to focus on the possibilities of higher order FLCs, such as type-2 FLCs that use type-2 fuzzy sets.

A type-2 fuzzy set is characterised by a fuzzy MF, i.e. the membership value (or membership grade) for each element of this set is a fuzzy set in [0,1], unlike a type-1 fuzzy set where the membership grade is a crisp number in [0,1] [12]. The MF of a type-2 fuzzy set is three dimensional and includes a footprint of uncertainty. It is the third-dimension of the type-2 fuzzy sets and the footprint of uncertainty that provide additional degrees of freedom making it possible to better model and handle uncertainties when compared to type-1 fuzzy sets.

It has been recently shown that in real world applications with large uncertainty levels such as mobile robot control, using interval or general type-2 FLCs (i.e. type-2 FLCs that use interval or general type-2 fuzzy sets) can handle the uncertainties better and outperform their type-1 counterparts [3], [5], [7]. However, manually designing and tuning a type-2 FLC to give a good response is a difficult task, particularly as the number of FLC MF parameters increase.
It has been shown that the selection of an optimal footprint of uncertainty for the individual type-2 MFs is highly complex and thus automatic optimisation processes are highly desirable [11], [12].

In the type-1 FLC domain, several researchers have explored the possibilities of using different learning and evolutionary techniques to specify FLC parameters [1], [2], [6], [8], [9]. However, the number of parameters and the complexity increase when dealing with type-2 FLCs. Recent work has proposed the use of neural network based systems to learn the type-2 FLC parameters [10], [11], [18]. However, such systems require existing data to optimise the type-2 FLC. Hence, they are not suitable for applications where there is no or insufficient data available to represent the various situations faced by the controller (as in the mobile robots domain).

Genetic Algorithms (GAs) do not require a priori knowledge such as a model or data but perform a search through the solution space based on natural selection using a specified fitness function. GAs have recently been used to optimise the type-2 FLCs parameters where in [17] two GAs were used to first optimise a type-1 FLC, then to blur the MFs of this type-1 FLC to create a footprint of uncertainty for the type-2 FLC.

In this paper, we will take a different approach by directly optimising the type-2 MFs rather than developing the type-1 MFs and then blurring them. We will also investigate the validity of the evolved type-2 FLCs generated in simulation for real world applications by employing the generated FLCs on a real mobile robot in an outdoor arena and comparing its performance to that of type-1 and manually designed type-2 FLCs.

The next section will briefly present the interval type-2 FLC and highlight its benefits. Section III will detail the GA based evolutionary system. Section IV will present the experiments and results. Finally, the conclusions will be presented in Section V.

II. INTERVAL TYPE-2 FUZZY LOGIC CONTROLLERS

The interval type-2 FLC depicted in Fig. 1(a) uses interval type-2 fuzzy sets (such as shown in Fig. 1(b)) to represent the inputs and/or outputs of the FLC. In the interval type-2 fuzzy sets all the third dimension values are equal to one. The use of interval type-2 FLC helps to simplify the computation (as opposed to the general type-2 FLC) which facilitates the design of a robot FLC that operates in real time [7].

The interval type-2 FLC works as follows: the crisp inputs from the input sensors are first fuzzified into input type-2 fuzzy sets; singleton fuzzification is usually used in interval type-2 FLC applications due to its simplicity and suitability for embedded processors and real time applications. The input type-2 fuzzy sets then activate the inference engine and the rule base to produce output type-2 fuzzy sets. The type-2 FLC rule base remains the same as for the type-1 FLC but its MFs are represented by interval type-2 fuzzy sets instead of type-1 fuzzy sets. The inference engine combines the fired rules and gives a mapping from input type-2 fuzzy sets to output type-2 fuzzy sets. The type-2 fuzzy output sets of the inference engine are then processed by the type-reducer which combines the output sets and performs a centroid calculation which leads to type-1 fuzzy sets called the type-reduced sets. There are different types of type-reduction methods. In this paper we will be using the Center of Sets type-reduction as it has reasonable computational complexity that lies between the computationally expensive centroid type-reduction and the simple height and modified height type-reductions which have problems when only one rule fires [12]. After the type-reduction process, the type-reduced sets are defuzzified (by taking the average of the type-reduced set) to obtain crisp outputs that are sent to the actuators. More information about the interval type-2 FLC and its benefits can be found in [7], [12].

![Fig. 1. (a) Structure of the type-2 FLC. (b) An interval type-2 fuzzy set.](image)

III. THE GA BASED EVOLUTIONARY SYSTEM

A Genetic Algorithm based system is used to evolve the parameters of the type-2 FLC MFs. As such, the GA chromosome includes the interval type-2 MFs parameters for both the inputs and outputs of the robot type-2 FLC. We have used interval type-2 fuzzy sets which are described by a Gaussian primary MF with uncertain standard deviation as shown in Fig. 1(b).

The GA based system uses real value encoding to encode each gene in the chromosome. Each GA population consists of 30 chromosomes. The GA uses an elitist selection strategy. The GA based system procedure can be summarised as follows:

Step 1: 30 chromosomes are generated randomly while taking into account the grammatical correctness of the chromosome (for example the inner standard deviation $\sigma_1$ is less than the outer standard deviation $\sigma_2$). The “Chromosome Counter” is set to 1 (the first chromosome). The “Generation Counter” is set to 1 (the first generation).
Step 2: A type-2 FLC is constructed using the chromosome identified by the “Chromosome Counter” and the robot is placed into the arena at a fixed position A. The FLC is executed on the robot for a sufficient distance until it reaches the end position after which the robot is repositioned at a different position B and the FLC is executed for a sufficient distance until it reaches another end position. It is during this stage that the fitness of the controller is being evaluated. The FLC is executed from two different starting positions A and B to minimize the risk of evolving controllers which only provide good performance under specific starting conditions.

During fitness evaluation a safety behaviour is in place to avoid the robot colliding with the wall. A controller that would have caused a collision is automatically assigned a disastrous fitness, making sure it is excluded through the selection process. After a controller’s fitness has been determined, a fixed controller takes over control and returns the robot to a correct position in respect to the wall (position A), enabling the test of the next controller.

Step 3: If “Chromosome Counter” < 30, increment “Chromosome Counter” by 1 and go to Step 2, otherwise proceed to Step 4.

Step 4: The best individual-so-far’s chromosome is preserved separately.

Step 5: If “Generation Counter” = 1 then store current population, copy it to a new population \( P \) and proceed to Step 6. Else, select 30 best chromosomes from population “Generation Counter” and population “Generation Counter” - 1 and create a new population \( P \).

Step 6: Use roulette wheel selection on population \( P \) to populate the breeding pool.

Step 7: Crossover is applied to chromosomes in the breeding pool and “chromosome consistency” is checked. (*)

Step 8: “Generation Counter” is incremented. If “Generation Counter” < the number of maximum generations or if the desired performance is not achieved, reset “Chromosome Counter” to 1 and go to Step 2, else go to Step 9.

Step 9: The chromosome which resulted in best fitness is kept and solution has been achieved; END.

(*) The crossover operator employed computes the arithmetic average between two genes [4]. It is used with a probability of 100% to force the GA to explore the solution space in between the previously discovered parental solutions [4]. With chromosome consistency we refer to the correctness of the chromosome’s genes in relation to their function in the FLC, (for example the inner standard deviation \( \sigma_1 \) is less than the outer standard deviation \( \sigma_2 \)). While in [17] the chromosome’s genes are re-arranged to achieve this, we either completely eliminate a chromosome from the population if it violates this criteria or, if the problem is restricted to the means of the MFs (for example the mean of the MF “Far” < mean of the MF “Near”), the means are adjusted to reflect the actual MF. If a chromosome is eliminated, it is replaced by a new, randomised but consistent chromosome, thus introducing new genetic material into the population.

IV. EXPERIMENTS AND RESULTS

A. Experimental Setup

In order to evaluate the proposed system, we have performed a series of experiments during which the controllers were evolved in our simulated arena and then tested in the real world. Below, we will briefly introduce our simulated and real world environments as well as the mobile robot that was used in the experiments.

We have used the Pioneer 2-wheeled mobile robot shown in Fig. 2(a) as the mobile robot platform in our experiments. The robot is equipped with 8 sonar sensors, a laser scanner and a camera.

The robot real world environment which is depicted in Fig. 2(b) consists of a circular outdoor arena. The arena presented the robot with a real-world outdoor environment with a large number of uncertainty sources not present in indoor environments. For example, the floor of the arena is road tarmac which provides an uneven surface which can also contain small debris such as leaves and small stones. In addition, the surrounding wall is constructed of dark grey bricks which due to their colour as well as their smooth and slightly glossy texture provided a challenge and led to high uncertainty levels in the input values. Furthermore, there were other sources of uncertainties ranging from wind, rain and significant differences in humidity.

The robot arena has been reproduced in simulation using the Webots simulator version 5.1.7. The simulated environment consists of a circular arena modelled using VRML as shown in Fig. 2(c). The wall is modelled using 16 flat panels, thus creating a roughly circular shape which tries to simulate the uncertainty levels present in the real environment.

![Fig. 2. (a) The mobile robot. (b) The outdoor robot arena. (c) The simulated environment.](image-url)
concept to evaluate the proposed system. The edge following behaviour used only the two side sonar sensors for navigation. The laser range finder was only used to give an accurate reading of the robots actual distance from the wall for performance evaluation purposes. All the computation was done using the on-board computer running Linux and Java version 1.5 while the results were logged on a server using a wireless link.

As shown in Fig. 3(a), for each controller run, the controller starts moving from a starting position till it reaches the end position at the end of the trial. As explained above, the characteristics of the arena wall caused unreflected sonar signals which led to missed readings. This could result in the robot drastically turning towards the wall as it “thinks” that this is a gap as shown in Fig. 3(b). The resulting “wobbling” of the robot affected the laser range finder which was unable to give accurate readings of the wall distance as shown in Fig. 3(b). We have identified this as a problem with the experimental setup and while the path plots clearly show any sensing problem, the Root Mean Square Error (RMSE) (based on the error in the desired distance kept to the wall which is sampled at every control step) calculations used correct readings only while eliminating any sensor-failure related “wobbles”.

B. Experiments and Results

The type-2 FLC used during the experiments has two sonar sensors as inputs which are represented by two Gaussian type-2 MFs (Near and Far) which have certain means and uncertain standard deviations. Therefore, each input type-2 MF is represented by three parameters (one mean and two standard deviations). Thus, 12 genes are used to represent the type-2 FLC inputs.

The type-2 FLC has 1 output governing the speed difference between the left and right wheels of the robot and thus, the turning angle. The FLC output is represented by 2 Gaussian type-2 MFs (Left and Right) which have certain means and uncertain standard deviations. Only the standard deviations of the output type-2 MFs are evolved and the means are fixed to guarantee that the FLC outputs are within the allowed domain of the possible outputs to avoid damaging the actuators. Thus, 4 genes are used to represent the type-2 FLC output. Hence as shown in Fig. 4, the GA chromosome for this type-2 FLC comprises 12(inputs) + 4(outputs) = 16 genes.

The fitness of each chromosome is generated by monitoring how the generated type-2 FLC has succeeded in following the edge at the desired distance over two consecutive runs when started from two different starting points.

Through repeated experiments, it was found that the GA based system evolves to good type-2 MFs after only about 14 generations. An example of the evolutionary progress is shown in Fig. 5 which shows the performance of the best individual found so far against the number of generations.

Due to the stochastic nature of the GA and to properly evaluate the performance of the proposed GA system, we have evolved 3 different type-2 FLCs. The RMSE results for the evolved type-2 FLCs are based on the average of the performance of the 3 evolved type-2 FLCs. In the real world environment due to the strong uncertainty levels, each robot controller run was repeated 3 times and the RMSE was based on the average of the 3 runs.

We have compared the performance of the evolved type-2 FLC against three type-1 FLCs with 4, 9 and 25 rules as well as against a manually designed type-2 FLC. Due to the space limitations, we are going to show only the control surfaces and robot paths for the evolved and manual type-2 FLCs as well as the 4 and 25 rules type-1 FLCs.

The control surfaces for all four controllers are shown in Fig. 6 while Fig. 7(a) shows the comparison of the robot
paths when executed in the simulated environment. As shown in Fig. 6, the type-1 FLC with 4 rules shows a non-smooth control surface which results in a very poor performance in simulation where the robot continuously “wobbles” from side to side. As the number of rules of the type-1 FLC is increased from 4 to 9 to 25, the control surface becomes more detailed and the robot path in Fig. 7(a) shows a better performance. The manually designed type-2 FLC shows a smooth control surface and produces equally a smooth motion around the arena. Finally, the evolved type-2 FLC control surface looks extremely smooth, which is reflected by a very smooth robot path in Fig. 7(a). As has been noticed in [7] and as can be seen in Fig. 6 and Fig. 7(a), as the number of rules increases for the type-1 FLC, both its control surface and its performance approaches that of the type-2 FLC. This is because the type-2 fuzzy sets contain a large number of embedded type-1 fuzzy sets which allow for a detailed description of the analytical control surface [7], [13].

Comparing the RMSEs of the 4 controllers shows that the 25 rule type-1 FLC produces a RMSE of 32.319 which vastly outperforms the 4-rule type-1 FLC which produced a RMSE of 147.843. The manually designed type-2 FLC with 4 rules produced a RMSE of 39.514. The evolved type-2 FLC with 4 rules, produced an average RMSE of 29.496 which outperforms the 4 and 25 rules based type-1 FLCs as well as the manually designed type-2 FLC.

In addition to the comparisons based on the simulated environment, we have performed experiments to compare the performance of the different controllers in the real world using the outdoor arena.

The performance of each controller in the outdoor arena is shown in Fig. 7(b). When comparing the robot paths, it can be seen that the type-1 FLC with 4 rules produces an unsatisfactory performance with a considerable amount of “wobbling”. The type-1 FLC with 25 rules on the other hand produces a smooth motion along the wall, reflecting its more detailed control surface. Both the manual and the evolved type-2 FLCs produce a very smooth motion along the wall.
Comparing the RMSEs, it can be seen that the evolved type-2 FLC with 4 rules produced an average RMSE of 88.200 which outperformed its manually designed type-2 counterpart (which produced a RMSE of 114.020) as well as both type-1 FLCS which gave a RMSE of 96.061 for the 25 rule type-1 FLC and 162.009 for the 4 rule type-1 FLC respectively).

As mentioned before it should be noted that the short wobbles in the robot paths in Fig. 7(b) where no wall is detected are due to sonar failure as explained in Fig. 3(b). These wobbles were nevertheless excluded from the RMSE calculation as explained above.

V. CONCLUSIONS

In this paper, we have investigated the possibility of using a GA based architecture to evolve the type-2 MFs parameters of interval type-2 FLCS aimed at mobile robot control in real world environments. We have shown that by employing our GA based architecture we were able to produce evolved type-2 FLCS that performed extremely well, both in simulation and in the real world. We have performed a series of evaluations in our simulated environment as well as on a real mobile robot in an outdoor arena. In both cases, the same FLCS were executed and it was found that the type-2 FLCS which were genetically evolved in simulation achieved a superior performance in comparison to the type-1 FLCS and the manually designed type-2 FLCS.

The results indicate that the genetic evolution of type-2 MFs can provide valid and high-performance type-2 FLCS without relying on any a priori knowledge such as logged data or a previously existing model. Thus, the proposed method is suitable for control problems where no such a priori knowledge is available such as in the mobile robots domain.

In the near future, we hope to extend our work on evolving type-2 FLCS to more complex controllers in terms of the challenge of the task the FLC is facing as well as the FLC complexity. As such, we are looking at evolving the parameters for general type-2 FLCS. Another significant aspect we are currently investigating is the possibility of creating a type-2 FLC for which the parameters are continually adjusted while the FLC is in operation, hopefully further increasing its potential to handle new environments and uncertainties.

ACKNOWLEDGMENT

We would like to acknowledge the support of the UK Engineering and Physical Sciences Research Council (EPSRC) and the University of Essex for supporting this work. We would like to thank all the members of the Intelligent Inhabited Environments Group at the University of Essex for their support.

REFERENCES


