Optimization of Interval Type-2 Fuzzy Logic Controller Using Quantum Genetic Algorithms

Pintu Chandra Shill†, Md. Faijul Amin‡, M. A. H. Akhand ‡, and Kazuyuki Murase†
†Department of System Design Engineering, University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan
‡ Department of Computer Science and Engineering, KUET, Khulna 9203, Bangladesh

{pintu, murase}@u-fukui.ac.jp, mdfaijulamin@yahoo.com, akhand@cse.kuet.ac.bd

Abstract—A Type-2 Fuzzy logic controller adapted with quantum genetic algorithm, referred to as type-2 quantum fuzzy logic controller (T2QFLC), is presented in this article for robot manipulators with unstructured dynamical uncertainties. Quantum genetic algorithm is employed to tune type-2 fuzzy sets and rule sets simultaneously for effective design of interval type-2 FLCs. Traditional fuzzy logic controllers (FLCs), often termed as type-1 FLCs using type-1 fuzzy sets, have difficulty in modeling and minimizing the effect of uncertainties present in many real time applications. Therefore, manually designed type-2 FLCs have been utilized in many control process due to their ability to model uncertainty and it relies on heuristic knowledge of experienced operators. The type-2 FLC can be considered as a collection of different embedded type-1 FLCs. However, manually designing the rule set and interval type-2 fuzzy set for an interval type-2 FLC to give a good response is a difficult task. The purpose of our study is to make the design process automatic. The type-2 FLCs exhibit better performance for compensating the large amount of uncertainties with severe nonlinearities. Furthermore, the adaptive type-2 FLC is validated through a set of numerical experiments and compared with QGA evolved type-1 FLCs, traditional and neural type-1 FLCs.

Keywords- Interval Type-2 FLC; Interval Type-2 fuzzy sets; Quantum Genetic Algorithms; Mobile Robot; Optimization

1. INTRODUCTION

Fuzzy logic systems (FLS) are fundamental methodologies to represent and process linguistic information, with mechanisms to deal with uncertainty and imprecision. With such remarkable attributes, FLS have been widely and successfully applied to control [1], classification [2] and modeling problem and in a considerable number of applications [3]-[5]. A fuzzy model is a set of fuzzy rules and the associated membership functions (MFs) that maps inputs to outputs. Fuzzy rules and MFs are either provided by human experts or learned from sample data. Many decision-making and problem-solving tasks are too complex to be understood quantitatively. People however succeed by using knowledge that is imprecise rather than precise. The construction of fuzzy logic controllers (FLCs) based on the appropriate expert knowledge base (KB) can be quick and effective. On the other hand, without such an expert KB the design of FLCs can be frustrating as it relies on trial and error rather than a guided approach.

The optimal design and tuning of FLCs is one of the optimization problems, as they can cope with non-convex or non-differentiable objective functions because of controllers’ structures and nonlinearities [6]-[7]. Some optimization approaches based on evolutionary computation to tuning of fuzzy control systems include genetic algorithms (GA) [8]-[9], simulated annealing (SA) algorithms [10]-[11], particle swarm optimization (PSO) algorithms [12]-[13], gravitational search algorithms (GSAs) [14]-[15] in several fuzzy control system structures. Other optimization methods that have been used for tuning FLCs include embedding vector space approach from nonlinear network optimization [16], clonal selection algorithms [17] or genetic programming [18].

For most fuzzy logic control problems, the most important issue is to determine the parameters that define the type-2 MFs. Because of this, the type-2 MFs optimization problems can be converted to parameter optimization problems. These parameters are generally based on the expert KB that is derived from heuristic knowledge of experienced control engineers and/or generated automatically.

Mendel [19] and Hagras [20] have shown that the type-1 fuzzy logic systems (FLSs) may be unable to model and minimize the effect of uncertainties that prevails’ in the real world applications. One restriction is that a type-1 fuzzy set is certain in the logic where the membership grade for each input is a crisp value. On the other hand, interval type-2 FLCs (that use interval type-2 fuzzy sets, characterized by fuzzy MFs) can handle the uncertainties.

GA was used by Martinez [9] in optimization of type-2 FLC. He applied GA to design FLC for the control of the perturbed autonomous wheeled mobile robot. Melin and Castillo [21] proposed a method based on type-2 fuzzy sets and neural networks called neuro-fuzzy to learn the parameters of the fuzzy system for intelligent control of nonlinear dynamic plants. Tan [22] used GA to optimize the parameters of FLCs. His proposed approach used mixed (type-1 and type-2) fuzzy sets for real time control.

In this paper, quantum genetic algorithms (QGA) is employed to adaptively adjust all relevant design parameters of interval type-2 FLCs. QGA is employed as an optimizer to determine not only the
type-2 fuzzy sets but also rule sets for designing an interval type-2 FLCs. It is found that QGA is use to generate fuzzy rule set and tune the MFs of the generated fuzzy rules automatically. We simulated type-2 FLCs, QGA evolved type-1 FLCs, neural and traditional coded FLCs to perform a comparative analysis of the systems’ response, in the presence of uncertainty.

In this study, quantum optimization can be regarded as a new optimization tool that can be used to overcome the limitations of trial and error approach as it is a quantum computing technique and finds the optimal solutions effectively and efficiently by combinatorial searching the large and complex solution space. So, QGA can be employed as a powerful search method to perform tasks such as generation of fuzzy rule base (RB), optimization of fuzzy RB, generation of MFs, and tuning of MFs types.

The remaining of the paper is structured as follows: Section II presents an introductory explanation of type-2 fuzzy sets and interval type-2 FLCs. In Section III, the problem statement and the kinematic and dynamic models of the mobile robot is presented. In Section IV, we introduce the key ideas of our approach called the type-2 quantum fuzzy logic controller (T2QFLC), Section V provides a simulation study of the mobile robot using the controller described in Section IV. Finally, Section VI presents some concluding remarks and some future directions.

II. Type-2 FUZZY SETS AND FLC

The concept of a type-2 fuzzy set was introduced by Prof. Zadeh [23] in 1975. A type-2 fuzzy set is defined by MFs as shown in Fig.1. The fuzzy grade of that is a fuzzy set in the closed interval [0, 1] rather than a point in [0, 1]. A type-2 fuzzy set, denoted as $\tilde{A}$, is characterized by a type-2 MF $\mu_\tilde{A}(x,u)$ [24], where $x \in X$ and $u \in J_x \subseteq [0,1]$, i.e.,

$$\tilde{A} = \{(x,u), \mu_\tilde{A}(x,u)\} | \forall x \in X \quad \forall u \in J_x \subseteq [0,1]\}$$

in which $0 \leq \mu_\tilde{A}(x,u) \leq 1$. $A$ can also be expressed as follows [24]:

$$A = \big\{(x,u), \mu_A(x,u)\} \big| \forall x \in X \quad \forall u \in J_x \subseteq \big[0,1]\big\} \quad J_x$$

where $\big\|$ denotes union over all admissible $x$ and $u$. $J_x$ is called primary membership of $x$, where $J_x \subseteq [0,1]$ for $\forall x \in X$ [24]. The uncertainty in the primary memberships of a type-2 fuzzy set $\tilde{A}$, consists of a bounded region that is called the footprint of uncertainty (FOU)[24]. It is the union of all primary memberships [24].

A. Type-2 FLC

A type-2 FLC comprises five components, which are fuzzifier, knowledge base (KB) consisting of rule base (RB) and database (DB), fuzzy inference engine, type- reducer and defuzzifier as depicted in Fig.2.

![Figure 1. An Interval Gaussian type-2 fuzzy set where $\sigma_l$ and $\sigma_u$ are minimum and maximum resultant widths respectively](Figure)

B. Fuzzifier

Since the input is in crisp normalized values, a fuzzification operator $\tilde{f}$ is used to fuzzify it in fuzzy form. The fuzzifier maps a crisp input vector with

$$\rho = (x_1, \ldots, x_p) \in X_1 \times X_2 \times \cdots \times X_p = X$$

into input fuzzy sets, $A_x$ [25][26]. However, we will use most frequently used singleton fuzzification method as it is fast to compute and thus it is suitable for mobile real time operation. In the singleton fuzzifier, fuzzy set $A$ has only a single point of non-zero membership with support $x_j$, where $\mu_A(x,u) = 1$ for $x = x_j$ and $\mu_A(x,u) = 0$ for $x \neq x_j$ which input measurement $x$ is perfect crisp.

![Figure 2. A Interval Type-2 FLC](Figure)

C. Rule base

The rules will remain the same as in type-1 FLCs but the antecedents and consequents will be represented by interval type-2 fuzzy sets [26]. Like most FLCs [27], the FLC discussed here applies the concepts of fuzzy implication and the compositional rules of inference for approximate reasoning. Suppose that we need to design a multiple-input-multiple-output (MIMO) mobile robot type-2 FLC having $p$ inputs.
\( x_i \in X_i, \ldots, x_p \in X_p \) and \( c \) outputs \( y_i \in Y_i, \ldots, y_c \in Y_c \) with ith fuzzy rule of the form:

\[
R_i^{MISO} : \text{IF} \ x_i \text{ is } F^i_{x_i} \text{ and } x_p \text{ is } F^i_{x_p} \text{ THEN } \ y_i \text{ is } G^i_{y_i}, \ i = 1, \ldots, M
\]

Where \( F^i_{x_1}, \ldots, F^i_{x_p} \) and \( G^i_{y_1}, \ldots, G^i_{y_c} \) are the antecedent and consequent MFs associated with the linguistic \( p \) input variables and \( c \) output variables, respectively, and \( M \) is the number of rules in the rule base.

D. Fuzzy Inference Engine

The fuzzy inference engine combines rules and gives a mapping from type-2 fuzzy sets in the input universe of discourse \( U \in \mathbb{R}^p \) to type-2 fuzzy sets in the output universe of discourse \( V \in \mathbb{R}^c \) based on the fuzzy logic principle. The structure of ith type-2 rule is having one output \( y_k \in Y_k \):

\[
R^i : F^i_1 \times F^i_2 \times \cdots \times F^i_p \rightarrow G^i_k \text{ and the type-2 fuzzy relation can be expressed by membership function as:}
\]

\[
\mu_{R^i}(x, y) = \mu_{F^i_1}(x_{1i}) \prod_{k=2}^{p} \mu_{F^i_k}(x_{ki}) \prod_{l=1}^{c} \mu_{G^i_l}(y_l)
\]

Where \( \prod \) denotes meet operation. The membership grades in the input type-2 fuzzy sets are combined with those in the output type-2 fuzzy sets using the extended sup–star composition; multiple rules are combined using the Join operation. They are defined and explained in a greater detail in [28]-[29].

E. Type-reduction

Type-reduction is that when an interval type-2 fuzzy sets is reduced to an interval-valued type-1 fuzzy set and then these type reduced sets are defuzzified to obtain crisp outputs. In this paper, we use centroid type reduction due to its reasonable computational complexity. For the centroid type reduction process, firstly combines the output type-2 fuzzy sets using union [21] (minimum t-norm),

\[
\tilde{B} = \{\sum_{l=1}^{M} \tilde{B}^l\}, \text{ as: }
\]

\[
\tilde{\mu}_{B}(y) = \prod_{l=1}^{M} \tilde{\mu}(y) \quad \forall y \in Y
\]

where \( \tilde{\mu}_{B}(y) \) is the secondary membership function for the \( l \)th rule and it depends on join and meet operation. The centroid type reduction calculates the centroid of \( \tilde{B} \). The type reduced set using the centroid type reduction can be expressed as:

\[
y_c(x) = \sum_{i=1}^{N} \frac{\prod_{l=1}^{c} \tilde{\mu}_{G^i_l}(y_l)}{\sum_{i=1}^{N} \prod_{l=1}^{c} \tilde{\mu}_{G^i_l}(y_l)}
\]

where \( i = 1, \ldots, N \). To compute this process, at first \( y \) domain is discretized into \( N \) points and then \( J_{y_i} \) is discretized into \( T_i \) \((i=0,1,\ldots,N)\) points. Total number of computations is \( \prod_{i=1}^{N} T_i \).

F. Defuzzification

After the type-reduction stage, Defuzzify the type reduced interval set \( y_i(x) \), determined by its left most \( y_l \) and right most point \( y_r \) using the average of \( y_l \) and \( y_r \). Hence the defuzzified crisp output is

\[
Y(x) = \frac{y_l + y_r}{2}
\]

III. SYSTEM DEFINITION

A mobile robot has to move from an initial position to the target (dock) by avoiding collisions with a single stationary obstacle in optimal path. It may have to move along a straight path or take a turn depending on the current situations in order to generate a collision-free path. The problem is taken from [30]. Figure 3 depicts the simulated geometry for the robot and loading dock schematically, in which mobile robot is moving among single stationary obstacles, in the same workspace. The control system must find incrementally a path to the loading dock, independently of the initial position of the robot.

The path planning of the mobile robot is determined by the three input variables \( x, y \) and \( \phi \), (considered as a point mass), where \( x \) and \( y \) are the cartesian co-ordinates of the mobile robot and \( \phi \) is the robot direction angle relative to the horizontal axis \( x \). The output variable is the control steering signal \( \theta \). Thus controller is a function of state variables

\[
\theta = f(x, y, \phi)
\]

As a first investigation, let us assume that there exists enough clearance between the robot, the walls and the obstacle in the workspace so that we can ignore the \( y \)-position co-ordinate of the robot. The co-ordinate \( y \) will be re-introduced into the discussion shortly and simplifying the controller function to:

\[
\theta = f(x, \phi)
\]

The state spaces of two inputs are \(-115^0 \leq \phi \leq 295^0\) & \(0 \leq x \leq 100\), and one output \( \theta \) within \([-40^0, 40^0]\). At every stage, the simulated mobile robot only moved forward until it hits the border of the loading dock. The final states \((x_f, \phi_f)\) will be equal or close to \((10, 90^0)\). The robot kinematics model is described by the following dynamic equations.
\[ x_{r+1} = x_r + \cos(\phi_r + \theta_r) + \sin(\phi_r) \sin(\theta_r), \]
\[ y_{r+1} = y_r + \sin(\phi_r + \theta_r) - \sin(\phi_r) \sin(\theta_r), \]
\[ \phi_{r+1} = \phi_r - \sin^{-1} \left( \frac{2 \sin(\theta_r)}{l} \right) \]

Where \( l \) is the length of the robot, we assume \( l=4 \).

For obvious reasons such controller does not perform very well if the distance between the truck position and the loading dock is small.

IV. HYBRID Q-FUZZY CONTROLLER: DESIGN AND METHODOLOGY

QGA is a probability optimization algorithm based on the concept of the quantum computing. In quantum computing, the smallest unit of information is called Q-bit. A Q-bit can be represented as \( |\alpha\rangle + |\beta\rangle |1\rangle \),

Where \( |\alpha|^2 + |\beta|^2 = 1 \). \( |\alpha|^2 \) indicates the probability of finding the Q-bit in “0” state and \( |\beta|^2 \) indicates the probability of finding the Q-bit in “1” state. A Q-bit may be in “1” state, in “0” state or in a linear superposition of the two states.

A Q-bit individual as a string of \( m \) Q-bits is defined as

\[ \begin{bmatrix} \alpha_1 \\ \beta_1 \\ \vdots \\ \alpha_m \\ \beta_m \end{bmatrix} \]

Where \( |\alpha_i|^2 + |\beta_i|^2 = 1 \), \( i = 1, 2, 3, \ldots, m \). A Q-gate which is a quantum mutation gate is used to speed up the convergence. It is defined as:

\[ U(\Delta \theta_i) = \begin{bmatrix} \cos(\Delta \theta_i) & -\sin(\Delta \theta_i) \\ \sin(\Delta \theta_i) & \cos(\Delta \theta_i) \end{bmatrix} \]

where \( \Delta \theta_i \), \( i = 1, 2, 3, \ldots, m \), is the rotation angle of a Q-bit towards the “0” state or “1” state depending on its sign. After applying Q-gate, the Q-bit should satisfy the normalization condition \( |\alpha_i|^2 + |\beta_i|^2 = 1 \), where \( |\alpha_i|^2 \) and \( |\beta_i|^2 \) are the values of updated Q-bit.

A quantum-type-2 fuzzy system depicted in Fig. 4 is a fuzzy system that uses QGA to determine type-2 fuzzy sets and fuzzy control rules. In this paper, we employed QGA to optimize the parameters of the MFs of Type-2 FLC; we consider using Gaussian Interval Type-2 MFs to each one of our three variables. At the same time, we also employed QGA for the selection and definition of RB of type-2 FLC.

![Figure 3. Mobile Robot and loading dock illustration](image-url)

\[ \begin{align*}
    \phi &= x + \cos(\phi + \theta) + \sin(\phi) \sin(\theta), \\
    \phi &= \phi - \sin^{-1} \left( \frac{2 \sin(\theta)}{l} \right)
\end{align*} \]

\[ \text{Inputs and their ranges} \]
\[ \text{Outputs and their ranges} \]
\[ \text{Divide input ranges in different regions} \]
\[ \text{Divide input ranges in different regions} \]

![Figure 4. Integration of type-2 FLCs and QGA](image-url)

A. Encoding

Two input variable \( x, \phi \) and one output variable \( \theta \) are used to produce fuzzy rules and their corresponding MFs. For this reason these variables are encoded in a chromosome of QGA. The domains for \( x, \phi \) and \( \theta \) are divided into \( 5, 7 \) and \( 7 \) regions respectively. The linguistic terms (MFs) for each of the input and output variables are used to describe them. The rule base contains total 35 rules. Each rule includes the real value of \( x, \phi \) and \( \theta \). Each variable (input and output) is divided into three parts: center (\( \mu \)), lower MFs width (\( \sigma_l \)), and upper MFs width (\( \sigma_u \)). So the rule base looks like (Fig. 5):

![Figure 5. QGA coding schema of type-2 FLC](image-url)

An important characteristic of fuzzy models, FM, is the partitioning of the input and output space of system variables (input, output) into fuzzy regions using fuzzy sets. The range of \( x \) is divided into five non-uniform intervals \([0, 32.5], [32.5, 47.5], [47.5, 52.5], [52.5, 67.5], [67.5, 100] \) and they are represented by five linguistic terms LE, LC, CE, RC and RE respectively [31]. The range of \( \phi \) is divided into seven non-uniform regions \([-115, -27.5], [-27.5, 46], [46, 86.5], [86.5, 98.5], [98.5, 146], [146, 216], [216, 295] \) and then they are represented by seven linguistic terms NL, NM, NS,
ZE, PS, PM, and PL respectively. Similarly seven divided regions of the range of \( \theta \), \([-40, -28], [-28, -12.5], [-12.5, -2.5], [-2.5, 2.5], [2.5, 12.5], [12.5, 28], [28, 40]\) are represented by linguistic terms NB, NM, NS, ZE, PS, PM and PB [31].

In this study five and seven gaussian type-2 fuzzy sets were used to partition the input spaces \( x \) and \( \phi \) respectively and seven gaussian type-2 fuzzy sets for output spaces [31]. The rule base, then, contains thirty-five \((7\times5)\) rules to account for every possible combination of input fuzzy sets. The fuzzy control if then rules are of the form: If \( x \) is \((\{LE, LC, CE, RC, \text{and RE}\}) \) and \( \phi \) is \((\{NL, NM, NS, ZE, PS, PM, \text{and PL}\}) \) then \( \theta \) is \((\{NB, NM, NS, ZE, PS, PM, \text{and PB}\}) \), output is one of the type-2 fuzzy sets used to partition the output space. 315 genes are used to represent the rule set. Therefore, we need to encode a total of 315 parameters for each individual of our population. In order to make this encoding schema we design a chromosome of 315 consecutive real genes. Figure 5 show a schematic of the chromosome structure to our quantum-type-2 FLS optimization approach.

B. Real Coded Quantum Genetic Algorithm

In this paper, real coded quantum genetic algorithm (RCQGA) is used as an optimization process.

B.1. Representation

Each individual is represented by real coded triploid chromosome which can be defined as follows:

\[
(R_1 \cdots R_i \cdots R_n, \alpha_1 \cdots \alpha_i \cdots \alpha_n, \beta_1 \cdots \beta_i \cdots \beta_n)
\]

Where \((R_i, \alpha_i, \beta_i)^T, i = 1, 2, \ldots, n\) is the \(i^{th}\) allele of real coded triploid chromosome and \(R_i\) is the \(i^{th}\) rule in fuzzy rule base. \((\alpha_i, \beta_i)^T\) is a pair of probability amplitudes of one qubit and satisfies normalization condition \(|\alpha|^2 + |\beta|^2 = 1\), \(n\) is the length of real-coded triploid chromosome which is 35.

B.2. Mutation

Gaussian Mutation operator is applied to update population at each generation. The \(i^{th}\) allele is randomly selected from \(p_i\), and the centers (\(c\)) of the input-output variables in the rule of the selected allele are changed as follows:

\[
C^{t+1} = C^t + (\text{Max} - \text{Min}) N(0, (\sigma_{fi}^2)^\frac{1}{2})
\]

Where Max and Min are respectively upper and lower bound of the regions in which \(C^t\) lies. \(C^{t+1}\) may not be within the limit so it is clipped to keep it within the region of \(C^t\). The center of \(\theta\) is considered always the whole range of \(\theta\). \(N(0, (\sigma_{fi}^2)^\frac{1}{2})\) denotes the Gaussian distribution of mean 0 and variance \((\sigma_{fi}^2)^\frac{1}{2}\). The value of variance \((\sigma_{fi}^2)^\frac{1}{2}\) is either \(|\alpha_{fi}|^2 + |\beta_{fi}|^2\) based on “Fine Search” or “Coarse Search” to be implemented [32]. The width (\(W^t\)) of each center is updated as follows:

\[
W^t = \begin{cases} 
\max(r * (\text{Max} - C^{t+1}) | C^{t+1} > (\text{Max} + \text{Min})/2) \\
\max(r * (C^{t+1} - \text{Min}) | C^{t+1} \leq (\text{Max} + \text{Min})/2)
\end{cases}
\]

Where \(r\) is the uniformly distributed random number in the range \([0, 1]\). The pair probability amplitudes of the \(i^{th}\) allele is updated by the quantum rotation gate (QRG) as follows

\[
\begin{align*}
(\alpha_{fi}^{t+1}) &= \cos(\Delta \theta_{fi}) - \sin(\Delta \theta_{fi})(\alpha_{fi}^t) \\
(\beta_{fi}^{t+1}) &= \sin(\Delta \theta_{fi}) \cos(\Delta \theta_{fi})(\beta_{fi}^t)
\end{align*}
\]

Here \(\Delta \theta_{fi}\) is rotation angle of Q-bit and it is calculated as follows,

\[
\Delta \theta_{fi} = \text{sgn}(\alpha_{fi}^t, \beta_{fi}^t) \theta_0 \exp \left(-\frac{|\theta|^2}{|\theta|^2 + \gamma} \right)
\]

Where \(\theta_0\) is the initial rotation angle, \(\gamma\) is the scale parameter. These control the rotation angle and increase the speed of convergence, the sign \(\text{sgn} (...)\)

| TABLE I. OPTIMIZED MFs PARAMETERS (OPTIMAL SOLUTION) |
|---------------------------------|
| **MFs of x**                     |
| **LE**  | **LC**  | **RC**  | **RE**  |
| \(m\)  | \(\alpha_1\) | \(\alpha_i\) | \(\beta_l\) | \(\alpha_l\) | \(\alpha_i\) | \(\beta_{1l}\) | \(\alpha_{1l}\) | \(\alpha_{il}\) | \(\beta_{il}\) | \(\alpha_{il}\) | \(\beta_{il}\) |
| 0.01   | 0.01   | 0.01   | 0.01   |

| **MFs of \(\phi\)** |
|-------------------|
| **NL** | **NM** | **NS** | **ZE** | **PS** | **PM** | **PL** |
| \(m\)  | \(\alpha_1\) | \(\alpha_i\) | \(\alpha_l\) | \(\alpha_{1l}\) | \(\alpha_{il}\) | \(\alpha_{il}\) | \(\beta_{il}\) | \(\alpha_{il}\) | \(\beta_{il}\) | \(\alpha_{il}\) | \(\beta_{il}\) |
| 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   |

| **MFs of \(\theta\)** |
|-------------------|
| **NB** | **NS** | **ZE** | **PS** | **PM** | **PB** |
| \(m\)  | \(\alpha_1\) | \(\alpha_i\) | \(\alpha_l\) | \(\alpha_{1l}\) | \(\alpha_{il}\) | \(\alpha_{il}\) | \(\beta_{il}\) | \(\alpha_{il}\) | \(\beta_{il}\) | \(\alpha_{il}\) | \(\beta_{il}\) |
| 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   | 0.01   |
determines the direction of the rotation angle.

B.3. Discrete Crossover (DC) and Elitism

DC is performed repeatedly after a fixed number of generations and it expands the search space to find the suitable steering angle \( \theta \) with respect to input variables with minimized trajectory error (fitness value). The elitism technique is used to ensure that the rule base with best fitness value will not be lost.

B.4. RCQEA procedure

Step1: Initialization: A Population of \( N \) individuals \( P^t = \{ p_1, \ldots, p_j, \ldots, p_k \} \) is initialized by randomly chosen real numbers, where \( p_j^t \) is an individual.

Step2: Decode and Evaluation: At each generation \( t \) RCQGA maintains a population of real-coded triploid chromosome. A rule \( R_{ij}^t \) in \( p_j^t \) contains nine values, center (m), lower MFs and upper MFs for each of three variables \( (x, \varphi, \theta) \). Decode the every chromosome into RB and MFs for the construction of interval type-2 FLC and the constructed FLC is executed on the robot until it reaches the goal position or near to the goal position. Each potential solution (FLC) is evaluated and assigned a fitness value according to its performance to the problem. The fitness value for each chromosome is defined as the trajectory error which is defined as follows:

\[
\text{Trajectory / Fitness} = \frac{\text{Length of Robot Trajectory}}{\text{Distance(initial position, final position)}}
\]

Step3: Recombination: Apply mutation and discrete crossover operator to chromosomes and generate new chromosomes as well as new generation. Check the termination condition and go to step 2 if the termination condition is true otherwise go to step 4.

Step4: Stop: The best fitted chromosome is kept and solution has been achieved.

V. SIMULATION RESULTS AND EVALUATION

To evaluate the accuracy of the proposed system, we have carried out a series of experiments which the controller were evolved in our simulated arena. The average optimal means and standard deviations of MFs for \( x, \varphi, \) and \( \theta \) are shown in table I. The generated optimal control rule base (after the conversion from optimal parameter to linguistic form) also shown in table II.

We have obtained the fuzzy control rules from the best chromosomes of QGA after 100 generations.

Table III. Five initial positions \((x, y, \varphi)\) and their steps of result

| Case | 1 | 2 | 3 | 4 | 5 |
|------|---|---|---|---|---|
| \( x \) | 15 | 15 | 15 | 30 | 30 |
| \( y \) | 15 | 15 | 30 | 30 | 50 |
| \( \varphi \) | 60° | -30° | 15° | 25° | -45° |
| T1FLC | 40 | 56 | 34 | 31 | 45 |
| T2FLC | 28 | 40 | 27 | 22 | 38 |

Figures 6(a)-6(f) shows an example of tuned type-2 fuzzy sets for the efficient design of interval type-2 FLC. Figure 8 shows the times for the mobile robot to reach the goal position in 5 different initial conditions and their trajectories are plotted in Fig.7. Table III shows the five initial conditions for \((x, y, \varphi)\) with their steps.

![Figure 6. Tuned Gaussian Type-2 Fuzzy sets](image-url)
Figure 7. (a) Show robot trajectories avoiding stationary obstacle (cross-hatched circle) via T1QFLC, and (b) Show via interval T2QFLC, all from 5 different initial conditions.

Figure 8. Show the total steps to reach goal position by 5 different initial cases.

A. Evaluating the work

The mobile robot/truck is a classical control problem that is generally used as well-known benchmark problem for the evaluation of new control algorithms and as such it has been well analyzed [30][33]. The graph of in Figures 7, 8, 9 and 10 shows the performance comparison with QGA evolved type-1 FLC (T1QFLC), neural coded type-1 FLCs, traditional type-1 FLC and T2QFLC. From Figure 7, 8, 9, 10 and table III and table IV, it is obvious that the performances of T2GFLC are better than those in T1QFLC, neural type-1 and traditional type-1 FLCs. It not only takes less steps to arrive the goal position using interval T2QFLC, but also it shows the smoother trajectories (shown in Fig. 7 and Fig. 9).

Figure 9. Evaluation T2QFLC with other methods

Recently Li and Li in [33] have presented the fuzzy control system based on a hybrid clustering method and neural network. In their work, a two-stage technique, structure identification step and parameter identification step, are used for the constructing fuzzy logic control system. The clustering method is proposed to construct an initial fuzzy model to determine the number of fuzzy rules from the intuitionistic-desired trajectories. The clusters are automatically generated and the data are appropriately classified through clustering method. Then neural network is applied to obtain a more precise fuzzy model in the parameter identification for the truck/mobile robot control. The clustering method (off-line approach) is one of the most promising techniques when input/output samples are available for the system. It is not possible however to generate initial fuzzy model without such type of input output data.

The number of steps of trajectories depends on the physical orientation of the robot. We also analyse and compare the robot trajectories with other existing methods for different values of $\phi$. The number of steps of trajectories of the robot known as truck from given positions to back up to the loading zone controlled by the neural fuzzy system [33], traditional fuzzy system [33] and T2QFLC are given in IV.

| Case | 1  | 2   | 3   |
|------|----|-----|-----|
| $x$  | 15 | 15  | 15  |
| $y$  | 15 | 15  | 15  |
| $\phi$ | 0° | -30° | -90° |
| Traditional FLC | 68 | 84 | 91 |
| Neural FLC | 52 | 70 | 79 |
| T2QFLC | 29 | 32 | 34 |

Table IV. Trajectory steps for different values of $\phi$ with same $(x, y)$ coordinate.

Figure 10. Shows the results of Best fitness trajectory errors in T1QFLC and T2QFLC

It has been found that the QGA based system evolves to optimal type-2 MFs and RB after some generations. An example of the best fitness quantum genetic progress is presented in Fig. 10.
which demonstrations the performance of the best chromosome found so far against the number of generations.

VI. CONCLUSIONS

This paper has revealed the possibility of using QGA based adaptive architecture to evolve the type-2 MFs and their corresponding rule set parameters of interval type-2 FLCs. We have shown that T2QFLC is an adaptive model that is able to tune MFs parameters and establish a reliable fuzzy control rules aimed at mobile robot control in real world unstructured environments. The type-2 based control architecture could cope with the unstructured noisy environments and achieved a superior control performance that outperformed the QGA evolved type-1 FLCs, traditional type-1 FLCs and neural coded FLCs.

Suggestions for follow-up works that may come after this paper are as follows: This research work is to be extended for intelligent control of a robotic arm in the presence of moving obstacle, the path planning problem for multiple mobile robots with more than one obstacles either moving or fixed in the workspace.

REFERENCES

[1] Pierre Guillenm, “Fuzzy Logic Applied to Motor Control,” IEEE Trans. on Industry application, vol. 32, no.1, pp. 51-56, Jan/Feb 1996.
[2] Shang-Ming Zhou and John Q. Gan, “Constructing L2-SVM-Based Fuzzy Classifiers in High Dimensional Space with Automatic Model Selection and Fuzzy Rule Ranking,” IEEE Trans. Fuzzy Syst., vol. 15, no. 3, pp. 398-409, June 2007.
[3] Z. Liu and H.X. Li, “A probabilistic fuzzy logic system for modelling and control,” IEEE Trans. Fuzzy Syst., vol. 13, pp. 848-859, 2005.
[4] L.X. Wang, “Stable adaptive fuzzy control of nonlinear systems,” IEEE Trans. Fuzzy Syst., vol. 1, pp. 146-155, 1993.
[5] I-Isun Li and Lim-Wang Lee, “A hierarchical structure of observer-based adaptive fuzzy-neural controller for MIMO systems,” Fuzzy Sets and Systems, vol. 185, issue 1, pp. 52-82, 2011.
[6] M. López-Ibáñez and T. Stützle, “The impact of design choices of multi-objective ant colony optimization algorithms on performance: An experimental study on the bi-objective TSP,” in Proc. Genetic and Evolutionary Computation Conference, Portland, OR, USA, 2011, pp.71-78.
[7] J.C. Ferreira, C.M. Fonseca, and A. Gaspar-Cunha, “Assessing the quality of the relation between scalarizing function parameters and solutions in multi-objective optimization,” in Proc. IEEE Congress on Evolutionary Computation, Trondheim, Norway, 2009, pp.1131-1136.
[8] P.C. Shih, K. K. Pal, M.F. Amin and K. Murase, “Genetic algorithm based fully automated and adaptive fuzzy logic controller” in Proc. IEEE Int. Conf. Fuzzy Systems, Jun. 2011, pp.1572 -1579.
[9] R. Martinez-Soto, O. Castillo, and L.T. Aguilar, “Optimization of interval type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms,” Information Sciences, vol. 179, pp. 2158–2174, 2009.
[10] R. Jain, N. Sivakumar, and T.K. Radhakrishnan, “Design of self tuning fuzzy controllers for nonlinear systems,” Expert Systems with Applications, vol. 38, pp. 4466-4476, 2011.
[11] R.-E. Precup, R.-C. David, E.M. Petriu, S. Preitl, and M.-B. Rádac, “Fuzzy control systems with reduced parametric sensitivity based on simulated annealing.” IEEE Trans. Industrial Electronics, vol. 99, pp.1-15, 2011.
[12] G.-R. Yu, Y.-J. Huang, and L.-W. Huang, “T-S fuzzy control for magnetic levitation systems using quantum particle swarm optimization,” in Proc. SICE Annual Conference, 2010, Taipei, Taiwan, pp. 48–53.
[13] Z. Bingili and O. Karahan, “A Fuzzy Logic Controller tuned with PSO for 2 DOF robot trajectory control,” Expert Systems with Applications, vol. 38, pp. 1017–1031, 2011.
[14] R.-E. Precup, R.-C. David, E. M. Petriu, S. Preitl, and A. S. Paul, “Gravitational Search Algorithm-Based Tuning of Fuzzy Control Systems with a Reduced Parametric Sensitivity,” Soft Computing in Industrial Applications, A. Gaspar-Cunha, R. Takahashi, G. Schaefer, and L. Costa, eds, Springer-Verlag, pp.141–150, 2011.
[15] R.-E. Precup, R.-C. David, E. M. Petriu, M.-B. Rádac, S. Preitl, and J. Fodor, “Evolutionary optimization-based tuning of low-cost fuzzy controllers for servo systems,” Knowledge Based Systems, DOI: 10.1016/j.knosys.2011.07.006, 2011.
[16] E.G. Carrano, R.H.C. Takahashi, C.M. Fonseca, and O.M. Neto, “Non-linear network optimization - A embedding vector space approach,” IEEE Trans. Evolutionary Computation, vol. 14, pp. 206-226, 2010.
[17] X. Wang, X.Z. Gao, and S.J. Ovaska, “Fusion of clonal selection algorithm and harmony search method in optimization of fuzzy classification systems,” Int. J. Bio-Inspired Computation, vol. 1, pp. 80-88, 2009.
[18] T. Nakashima, Y. Yokota, Y. Shoji, and H. Ishibuchi, “A genetic approach to the design of autonomous agents for futures trading,” Int. J. Artificial Life and Robotics, vol. 11, pp.145–149, 2009.
[19] H. A. Hagras, “A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots,” IEEE Trans. Fuzzy Systems, vol.12, no.4, pp.524-539, 2004.
[20] J.M. Mendel, Rule-Based Fuzzy Logic Systems: Introduction and New Directions, Prentice-Hall, Englewood Cliffs, NJ, 2001.
[21] P. Melin and O. Castillo, “Intelligent control of nonlinear dynamic plants using type-2 fuzzy logic and neural networks,” in Proc. of AAAI/IAAI, 2002, pp. 22-27.
[22] D. W. W. Tan, “A simplified type-2 fuzzy logic controller for real-time control,” ISA Trans., vol. 45, pp.503-516, 2006.
[23] L. A. Zadeh, “The concept of a linguistic variable and its applications to approximate reasoning-1,” Inform. Sci., vol.8, pp.199-249, 1975.
[24] J. Mendel and R. John, “Type-2 fuzzy sets made simple,” IEEE Trans. Fuzzy Syst., vol. 10, pp. 117-127, Apr. 2002.
[25] Q. Liang, N. Karnik, and J. Mendel, “Connection admission control in ATM networks using survey-based type-2 fuzzy logic systems,” IEEE Trans. Syst., Man, Cybern. C, vol. 30, pp. 329-339, Aug 2000.
[26] J. Mendel, Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions, Upper Saddle River, NJ: Prentice-Hall, 2001.
[27] J. M. Joe, P. Y. Chen, and S. F. Yang, “An Adaptive Fuzzy Logic Controller: Its VLSI Architecture and Applications,” IEEE Trans. Very Large Scale Integration (VLSI) Systems, vol.8, no.1, pp. 52-60, Feb. 2000.
[28] N. N. Karnik and J. M. Mendel, “Operations on type-2 fuzzy sets,” Fuzzy Sets and Systems, vol. 122, pp. 327-348, 2001.
[29] M. Mizumoto and K. Tanaka, “Some Properties of fuzzy sets of type-2,” Infom.Control, vol. 31, pp.312-340, 1976.
[30] M. Mohammadian and R. J. Stonier, “Fuzzy logic and genetic algorithms for intelligent control and obstacle avoidance,” in Complex Systems: Mechanism of Adaptation, R. J. Stonier and X. H. Yu, Eds. Amsterdam: IOS Press, 1994, pp.149-156.
[31] P. C. Shih, M.A.H. Akhand, M. F. Amin, K. Murase “Optimization of “Fuzzy Logic Controller for Trajectory Tracking using Genetic Algorithm,” J. Advanced Computational Intelligence and Intelligent Informatics (JACIII), Vol. 15, No. 6, pp.639-651, 2011.
[32] R. Zhang and H. Gao, “Real-coded quantum evolutionary algorithm for complex function with high dimension,” in Proc. IEEE Int. Conf. Mechatronics and Automation, Aug. 2007, pp.2974-2979.
[33] Y. Li and Y. Li, “Neutral-fuzzy control of truck backer-upper system using a clustering method,” Neuro Computing, vol.70, pp. 680-688, 2007.
[34] R. I. John, “ Embedded interval valued type-2 fuzzy sets,” in Proc. FUZZIEEE Int. Conf., May 2002, pp. 1316-1319.
[35] R. I. John and C. Czarniecki, “A type-2 adaptive fuzzy inferencing system,” in Proc. IEEE Systems, Man and Cybernetics, pp.2068-2073, 1998.