Online Interval Type-2 Fuzzy Extreme Learning Machine Applied to 3D Path Following for Remotely Operated Underwater Vehicles

Adrian Rubio-Solis[a], Uriel Martinez-Hernandez[b], Luciano Nava-Balanzar[a], Luis G. Garcia-Valdovinos[a], Noe A. Rodriguez-Olivares[a], Juan P. Orozco-Muñiz[a], Tomas Salgado-Jimenez[a]

[a] *Center for Engineering and Industrial Development (CIDESI), Qro. México, (email: adrian.rubio@cidesi.edu.mx, Inava@cidesi.edu.mx, ggarcia@cidesi.edu.mx, noe.rodriguez@cidesi.edu.mx, juan.orozco@cidesi.edu.mx,t.salgado@cidesi.edu.mx)*

[b] *Department of Electronic and Electrical Engineering, University of Bath, UK, (email:u.martinez@bath.ac.uk)*

Abstract

In marine missions that involve 3D path following tasks, the overall goal of Underwater Vehicles (UVs) is the successful completion of a path previously specified by the operator. This implies that the path must be followed by the UV as closely as possible and arrive at a location for collection by a vessel. In this paper, an Online Interval Type-2 Fuzzy Extreme Learning Machine (OIT2-FELM) is suggested to achieve a robust following behaviour along a predefined 3D path using a Remotely Operated Underwater Vehicle (ROV). The proposed machine is a fast sequential learning scheme to the training of a more generalised model of TSK Interval Type-2 Fuzzy Inference Systems (TSK IT2 FISs) equivalent to Single Layer Feedforward Neural Networks (SLFNs). Learning new input data in the OIT2-FELM can be done one-by-one or chunk-by-chunk with a fixed or varying size. The OIT2-FELM is implemented in a hierarchical navigation strategy (HNS) as the main guidance mechanism to infer local control motions and to provide the ROV with the necessary autonomy to complete a predefined 3D path. For local path-planning, the OIT2-FELM performs signal classification for obstacle avoidance and target detection based on data collected by an on-board scan sonar. To evaluate the performance of the proposed OIT2-FELM, two different experiments are suggested. First, a number of benchmark problems in the field of non-linear system identification, regression and classification problems are used. Secondly, a number of experiments to the completion of a predefined 3D path using an ROV is implemented. Compared to other fuzzy strategies, the OIT2-FELM offered two significant capabilities. On the one hand, the OIT2-FELM provides a better treatment of uncertainty and noisy signals in underwater environments while improving the ROV's performance. Secondly, online learning in OIT2-FELM allows continuous knowledge discovery from survey data to infer the surroundings of the ROV. Experiment results to the completion of 3D paths show the effectiveness of the proposed approach to handle uncertainty and produce reasonable classification predictions ($\sim 90.5\%$ accuracy in testing data).

Keywords: Online interval type-2 Fuzzy learning, Extreme Learning Machine, hierarchical fuzzy behaviours, Neural Networks.

1. Introduction

Remotely Operated Underwater Vehicles (ROVs) are now being used for a variety of missions such as oceanic mapping, underwater structure inspection, environmental monitoring and exploration [1–10]. An ROV is an underwater vehicle generally guided by a human pilot through a link cord or tether providing its power and data communication [11]. In the sector of oil and gas industry, ROVs have become a well-established technology to complete offshore activities that usually involve regular maintenance, assessment and security of pipelines and ship hulls [11–13]. In missions where minimal human intervention is not possible, autonomy and navigational accuracy are often two capabilities demanded in ROVs, especially to localize targets within sub-meter accuracy for later data analysis [12, 14–22]. Without an operator in the loop, the ROV must able to determine its location, motion, orientation and interpret its surroundings such that it can autonomously maneuver in uncertain environments [23].

With continuous advances in control, artificial intelligence, computer and sensor technology, autonomous navigation applied to ROV is very attractive not only to provide human assistance in decision making, but also for various control scenarios such as trajectory/target tracking, path following/planning and formation producing. Particularly high accuracy positioning and path following have drawn compelling interest to the development of autonomous navigation strategies for the inspection of underwater structures in oil and gas industry as well as salvage and monitoring operations [11]. In underwater path following, navigational accuracy can be defined as the precision with which an underwater vehicle (UV) completes a mission [3], while autonomy is its ability to reduce overall human intervention to successfully complete a predefined path [14].

The literature contains numerous references to the development of control and navigation strategies for path following [1, 2, 8, 10–12, 17–22, 24–28]. In [10], a two-layer control framework that consists of a fuzzy PID control and its optimisation through an heuristic fuzzy approach, was developed to address the problem of 3D path following in underactuated UVs subject to uncertainties. In [11], authors addressed the problem of path following under model uncertainty, disturbance and measurement error by using redundant measurement and data fusion. An sliding mode controller was used to manage uncertainty and disturbance with a zero tracking error maintained by employing integral action in the control structure [12, 22]. The accurate accomplishment of marine missions, UVs usually faces three major challenges: (a) parametric uncertainties in the vehicle's model, (b) sensor measurement states suffer from errors due to the bias, drift and noises, and (3) vehicles are usually exposed to the high dynamics of currents and waves affecting significantly their missions [10, 27].

In an important branch of robotics and marine vehicle applications, fuzzy logic has been credited to be a suitable methodology for the design of robust guidance and control architectures able to provide a satisfactory performance in the face of nonlinearities, imprecision and uncertainty [12, 29, 30]. A number of path following and path planing strategies have employed hierarchical control structures in which fuzzy logic is the main guidance mechanism [2, 25, 26, 31]. Compared to traditional approaches based on sequential task decomposition for real-time response, hierarchical fuzzy control structures encode robot's behaviour as a fuzzy rule that maps each sensor's signal into control output according to a desired control policy [5]. These architectures are behaviour-based systems that facilitate real-time intelligence response by decomposing general robot's behaviour into a set of local-purpose routines that operate concurrently rather than sequentially [32, 33]. Such structures decompose complex motion control into a number of simple reactive fuzzy controllers which can be classified into the following categories: 1) conventional fuzzy control, 2) adaptive fuzzy control and 3) hybrid fuzzy control, including fuzzy PID control, fuzzy sliding mode control and neuro-fuzzy controllers [29]. Having a hierarchical architecture that divides fuzzy logic controllers and fuzzy learning for robot's control into smaller subsystems accounts to reducing the negative effect that a large rule-base may have on real-time applications [32]. Hierarchical structures also contribute to overcome the problem of insufficient knowledge for designing large fuzzy rule bases. For instance, in [31], a hierarchical/Lyapunov fuzzy control system for horizontal plane trajectory tracking in underactuated Autonomous Underwater Vehicles (AUV) was suggested. The fuzzy architecture involved two different hierarchies, in which at high level, guidance control laws were executed through a type-1 fuzzy inference system. At low level, vehicle's surge force and yaw control were generated by a kinetic control. In [25], a hierarchical type-2 fuzzy structure that consisted of two different levels was implemented in an mobile robot as the main path planner. An evolutionary approach based on genetic algorithms was used to optimise the parameters of each membership function (MF) in the fuzzy path planner. According to results presented in [25], in outdoor environments where uncertainty is inherent, Interval Type-2 fuzzy logic (IT2 FL) were employed to produce a better performance than traditional type-1 fuzzy logic systems (FLSs). IT2-FLSs have demonstrated to outperform their type-1 counterpart in a large number of applications [34]. This has been mainly attributed to the ability of IT2-FLSs to better treat uncertainty and imprecision, of which IT2 fuzzy sets can be seen to possess an uncountable number of embedded type-1 FSs [30, 35].

This paper describes an Online Interval Type-2 Fuzzy Extreme Learning Machine (OIT2-FELM) applied to the completion of predefined 3D paths in indoors water containers using a Remotely Operated Vehicle (ROV). The OIT2-FELM is integrated into a hierarchical navigation strategy (HNS) to achieve two goals. First, the OIT2-FELM is used to train an Interval Type-2 Fuzzy Inference System (IT2 FIS) of Takagi-Sugeno-Kang to classify online data collected by a micro data sonar. This information is used by an ROV to recognise the type of contour (or objects) around it. Secondly, the outcome of data classification is utilised by the HNS for collision-avoidance and obstacle inspection. The proposed machine is a fast sequential learning method to the training of IT2 Fuzzy Inference Systems (IT2-FISs) in which data may arrive one-by-one or chunk-by-chunk with a fixed or varying size. The OIT2-FELM is derived from the functional equivalence between SLFNs and IT2 FISs. The application of the OIT2-FELM follows the theoretical principles of ELM, where each antecedent is arbitrarily selected while the consequent weights are determined analytically. The OIT2-FELM integrates into its structure a fast type-reduction process based on the SC algorithm which is an improved version of the non-iterative Center-Of-Sets-Type-Reduction-Without-sorting-requirement method (COSWSR). The main contributions of the proposed OIT2-FELM are:

1. Compared to traditional IT2-FELM theory [36–38], the OIT2-FELM incorporates a non-iterative type-reduction process based on the SC algorithm [39]. By doing so, the proposed OIT2-FELM eliminates the need of sorting consequent weights that is usually performed in Karnik-Mendel algorithms and its variants. This makes the proposed OIT2-FELM more appropriate for cost-sensitive real-time applications, reducing not only the associated computational burden, but also the associated model complexity.
2. The proposed OIT2-FELM is a fast sequential learning approach for a generalised model of Interval Type-2 Fuzzy Inference Systems (IT2 FISs) to the solution of problems in the field of regression, classification and nonlinear system identification.
3. The final model of an OIT2-FELM is an IT2-FIS where capabilities and new efforts from the theory of neural networks and fuzzy logic can be applied under adequate design considerations.
4. Compared to conventional T1 fuzzy logic systems, the OIT2-FELM is a high order fuzzy system able to better handle uncertainties that mobile robots usually face in underwater environments where sensor measurements are typically noisy and affected by the conditions of the observations.

To investigate the performance of the OIT2-FELM under real conditions, two different types of experiments are carried out in this work. First, to compare the performance of the OIT2-FELM with other literature methodologies, a number of benchmark problems in the field of non-linear system identification, regression and classification are suggested. In the second experiment, a TSK OIT2-FELM is implemented in a Hierarchical Navigation Strategy (HNS) as the main guidance mechanism to successfully complete a predefined 3D path using an ROV. The path following task suggested in this work is a frequent application of UVs for the inspection of underwater structures. Such task consists of completing a number of predefined circuits at a different depth, where the ROV must follow a trajectory described by the geometric shape of a given underwater structure.

The proposed HNS, together with the OIT2-FELM is inspired by the behaviour of some nocturnal mammals for foraging and burrow in dark constrained spaces [40]. When foraging in unknown environments, some mammals such as desert rodents use the information gathered by their whiskers to distinguish objects by surface contact and texture. The central role of the OIT2-FELM is to classify information coming from the scanning sonar mounted in the ROV in order to infer relevant knowledge about its surroundings. This information is utilised by the HNS to facilitate the ROV with near-to-real time intelligence required to achieve the necessary level of autonomy in the completion of 3D paths. The proposed HNS is a two level strategy that follows the operation principle of fuzzy architectures with a bottom-up hierarchy of increased behavioural complexity. At low-level, the HNS provides autonomy via decomposing motion control capabilities into a number of fuzzy behaviours which are realised as IT2 fuzzy controllers. Such behaviours serve a single purpose by operating in a reactive manner. Each behaviour performs nonlinear mappings from different subsets of the ROV's sensor suite to common actuators. At high-level, fuzzy behaviours are organised by a number of IF-THEN rules as a set of building blocks for more intelligent composite behaviours. Therefore, overall ROV's behaviour is encoded as a rule-base hierarchy that maps relevant information in the sensor input domain into control outputs according to a desired control policy. Performance results of the OIT2-FELM and the HNS are presented on individual and composite fuzzy behaviours for the completion of 3D paths. Experiments highlight the relevance of the information provided by the OIT2-FELM to local planning as well as to target detection. The focus here is on indoor navigation, however the proposed methodology can be extended for outdoor maritime environments.

The structure of this paper is organised as follows: Section 2 reviews related background theory, while in Section 3, the proposed methods are detailed. The performance of the proposed OIT2-FELM and a discussion are provided in section 4 and 5 respectively. Finally, section 6 draws conclusion.

2. Background Material

This section briefly reviews the basic concepts of Extreme Learning Machine (ELM), the equivalence between Single Layered Feedforward Networks (SLFNs) and Fuzzy Inference Systems (FISs) as well as theory about Online Sequential Fuzzy ELM (OS-Fuzzy-ELM), Interval Type-2 Fuzzy Extreme Learning Machine (IT2-FELM) and SC type-reduction respectively.

2.1. Extreme Learning Machine (ELM)

For $'P'$ arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i = [x_{i1}, \ldots, x_{iP}]^T \in \mathbf{R}^N$ and $\mathbf{t}_i = [t_{i1}, \ldots, t_{i\tilde{N}}]^T \in \mathbf{R}^{\tilde{N}}$, SLFNs with M hidden nodes and activation function g(x) can be mathematically expressed by the following equation [41]:

$$\sum_{i=1}^{M} \beta_i g_i(x_j) = \sum_{i=1}^{M} \beta_i g(w_i \cdot x_j + b_i) = \mathbf{y}_j;\ j = 1, \ldots, N \tag{1}$$

in which, $\mathbf{w}_i = [w_{i1}, \ldots, w_{iN}]^T$ is the weight vector connecting the ith hidden node and the input nodes, and $\beta_i = [\beta_{i1}, \ldots, \beta_{i\tilde{N}}]^T$ is the weight vector connecting the ith hidden node to the lth output. From Eq. (1), a matrix notation can be written as $\mathbf{H}\beta = \mathbf{T}$, where $\mathbf{H} \in \mathbf{R}^{P\times M}$, $\beta \in \mathbf{R}^{M\times\tilde{N}}$ and

$\mathbf{T} \in \mathbf{R}^{P\times\tilde{N}}$, in which, $\mathbf{H}$ is the hidden layer matrix of an SLFN [42, 43]. According to ELM theory, SLFNs with M hidden nodes and activation function g(x) can approximate $'P'$ arbitrary distinct samples with zero error means $\sum_{i=1}^{P} \parallel \mathbf{y}_i - \mathbf{t}_i \parallel = 0$, if there exist parameters $\hat{\beta}_i$, $\hat{b}_i$ and $\hat{w}_i$ such that [43]:

$$||\mathbf{H}(\hat{w}_1, \dots, \hat{w}_M, \hat{b}_i, \dots, \hat{b}_M)\hat{\beta} - \mathbf{T}|| = \min_{\mathbf{w}_i, \mathbf{b}_i, \beta} ||\mathbf{H}(\mathbf{w}_1, \dots, \mathbf{w}_M, \mathbf{b}_1, \dots, \mathbf{b}_M)\beta - \mathbf{T}|| \tag{2}$$

Eq. (2) is equivalent to minimising a cost function based on squared error $E = \sum_{j=1}^{P}(\sum_{i=1}^{M} \beta_i g(w_i x_j + b_i) - \mathbf{t}_j)^2$. Therefore, from the linear system expressed in Eq. (1) whose minimum norm least-squares solution is unique, can be achieved by calculating the pseudo-inverse $H^\dagger$ as:

$$\hat{\beta} = \mathbf{H}^\dagger \mathbf{T} \tag{3}$$

where $H^\dagger$ is the Moore-Penrose (MP) generalised inverse of matrix $\mathbf{H}$. The projection method can be efficiently used to compute $\mathbf{H}^\dagger = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T$, if $\mathbf{H}^T\mathbf{H}$ is nonsingular or $\mathbf{H}^\dagger = \mathbf{H}^T(\mathbf{H}^T\mathbf{H})^{-1}$, if $\mathbf{H}\mathbf{H}^T$ is nonsingular. Based on ridge regression, the stability of ELM can be improved by adding a positive term $1/\lambda$ to the diagonal of $\mathbf{H}^T\mathbf{H}$ or $\mathbf{H}\mathbf{H}^T$. By adding $1/\lambda$, Eq. (3) can be written as [41]:

$$\hat{\beta} = \left(\frac{1}{\lambda} + \mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{T} \tag{4}$$

2.2. Functional equivalence between SLFN and FISs

As described in [44], ELM theory can be applied to the training of a class of FISs of type-1 that are functionally equivalent to Single Layered Feedforward Networks (SLFNs) [45]. An FIS of TSK (or Mamdani) type that is equivalent to a SLFN can be defined by a number of fuzzy rules R^i of the form [28, 36]

$$R^i : \text{IF } x_1 \text{ is } A_{1i} \text{ AND } x_2 \text{ is } A_{2i} \text{ AND } \dots \text{AND } x_N \text{ is } A_{Ni} \text{ THEN } (y_1 \text{ is } w_{k1}) \dots (y_{\tilde{N}} \text{ is } w_{i\tilde{N}}) \tag{5}$$

where, $A_{ki}(k = 1, \dots, N, i = 1, \dots, M)$ are the fuzzy sets that correspond to the kth input variable $\mathbf{x}$ ($\mathbf{x} = [x_1, \dots, x_N]^T$) in the ith rule, where $\tilde{N}$ is the dimension of the output vector $\mathbf{y} = [y_1, \dots, y_{\tilde{N}}]$. When an FIS employs a TSK inference engine, w_{ij} ($k = 1, \dots, M, j = 1, \dots, \tilde{N}$) is defined by a linear combination of input variables, i.e. $w_{ij} = q_{ij,0} + q_{ij,0}x_1 + \dots q_{ij,N}x_N$, otherwise if the FIS is of Mamdani type, w_{ij} is a crisp value. In Fuzzy Logic System theory (FLS), the degree to which any given input x_k that satisfies the quantifier A_{ki} in the ith rule is specified by its Membership Function (MF) $\mu_{A_{ki}}(c_{ki}, a_i)$, where usually a non-constant piece-wise continuous MF is used. The symbol $\otimes$ is used to the representation of any fuzzy logic AND operations, where the firing strength of the ith fuzzy rule can be computed as

$$R^i(\mathbf{x}; \mathbf{c}_i, a_i) = \mu_{A_{i1}}(x_1, c_{1i}, a_i) \otimes \mu_{A_{k2}}(x_2; c_{2i}, a_i) \otimes \dots \otimes \mu_{A_{Ni}}(x_N; c_{Ni}, a_i) \tag{6}$$

Each fuzzy rule R^i can be normalised as

$$G(\mathbf{x}; \mathbf{c}_i, a_i) = \frac{R^j(\mathbf{x}; \mathbf{c}_i, a_i)}{\sum_{i=1}^{M} R^i(\mathbf{x}; \mathbf{c}_i, a_i)} \tag{7}$$

Similar to [44], G is called fuzzy basis function. The system output $\hat{y}$ of the TSK fuzzy model can be defined as a weighted sum of each normalised rule [44].

$$\hat{\mathbf{y}} = \frac{\sum_{i=1}^{M} \mathbf{w}_i R^i(\mathbf{x}; \mathbf{c}_i, a_i)}{\sum_{i=1}^{M} R^i(\mathbf{x}; \mathbf{c}_i, a_i)} = \sum_{i=1}^{M} \mathbf{w}_i G(\mathbf{x}; \mathbf{c}_i, a_i) \tag{8}$$

From Equation (4), a functional equivalence between FISs and SLFNs can be established if:

- Each $G(\cdot)$ represents the output function of each hidden node.
- The vector $\mathbf{w}_i$ represents the output weight vector.

As indicated in [44], a SLFN with activation function $G(\cdot)$ can approximate any continuous target function as long as the parameters of the membership function $\mu_{A_{ki}}$ are randomly generated and e membership function $\mu_{A_{ki}}$ is bounded, nonconstant, and piecewise continuous.

2.3. Online Sequential-Fuzzy-ELM Algorithm (OS-Fuzzy-ELM)

Since a FIS can be viewed as a SLFN, ELM theory can be applied to the training of FISs either of Takagi-Sugeno-Kang or Mamdani. In such an equivalence, parameters of each MF (c and a) are randomly generated, while consequent of each fuzzy rule are determined analytically [44]. Given a number of $'P'$ distinct training samples $(\mathbf{x}_p,\mathbf{t}_p)$, where $\mathbf{x}_p = [x_{p1}, \ldots, x_{pN}]^T \in \mathbf{R}^N$ and $\mathbf{t}_j = [t_{j1}, \ldots, t_{j\tilde{N}}]^T \in \mathbf{R}^{\tilde{N}}$, a FIS of TSK type with L fuzzy rules can be expressed as [41]:

$$f_L(\mathbf{x}_p) = \sum_{i=1}^{M} w_i g_i(x_j) = \mathbf{t}_p;\ p = 1, \ldots, P \tag{9}$$

For a TSK fuzzy model, the consequent of each fuzzy rule is defined as a linear combination, in which, each weight $\mathbf{w}_i = \mathbf{x}_{p,e}^T \mathbf{q}_i$, and $\mathbf{x}_{p,e} = [1\ \mathbf{x}_p^T]^T$ is the extended version of the input vector $\mathbf{x}$.

$$\mathbf{q}_i = \begin{pmatrix} q_{i1,0} & \cdots & q_{i\tilde{N},0} \\ \vdots & & \vdots \\ q_{i1,N} & \cdots & q_{i\tilde{N},N} \end{pmatrix}_{(N+1)\times\tilde{N}} \tag{10}$$

Therefore, Equation (9) can be written as:

$$f_L(\mathbf{x}_p) = \sum_{i=1}^{M} \mathbf{x}_{p,e}^T \mathbf{q}_i G(\mathbf{x}_p, \mathbf{c}_i, a_i) = \mathbf{t}_p;\ p = 1, \ldots, P \tag{11}$$

A compact representation for Eq. (11) is given by

$$\mathbf{HQ} = \mathbf{T} \tag{12}$$

in which, $\mathbf{Q}$ is the matrix of coefficients $q_{ij,k}$. If a TSK implication is employed, hidden matrix $\mathbf{H}$ is defined by:

$$\mathbf{H}(\mathbf{c}_1, \ldots, \mathbf{c}_M, a_1 \ldots, a_M, \mathbf{x}_1, \ldots, \mathbf{x}_P) = [\mathbf{x}_{p,e}^T G(\mathbf{x}_p, \mathbf{c}_1, a_1), \ldots, \mathbf{x}_{p,e}^T G(\mathbf{x}_p; \mathbf{c}_M, a_M)] \tag{13}$$

and $\mathbf{Q}$ is the parameter matrix for the TSK model:

$$\mathbf{Q} = \begin{pmatrix} \mathbf{q}_1 \\ \vdots \\ \mathbf{q}_M \end{pmatrix} \tag{14}$$

According to [44], Online Sequential ELM (OS-ELM) for SLFNs with additive or hidden neurons can be linearly extended to the training of TSK FISs. To implement OS-Fuzzy-ELM, first all parameters $\mathbf{c}_i$ and $\mathbf{a}_i$ of each MF are randomly generated. Secondly, OS-Fuzzy-ELM is implemented in two main phases to the calculation of the consequent part of each fuzzy rule. Therefore, given a number of MFs $\mu_A ik$ and number of fuzzy rules L, training data $D = \{(\mathbf{x}_p, \mathbf{t}_p) | \mathbf{x}_p \in \mathbf{R}^N, \mathbf{t}_p \in \mathbf{R}^{\tilde{N}}, p = 1, \ldots\}, P_0 \geq L$ arrives sequentially one-by-one or chunk-by-chunk [44].

- **Step 1) Initialisation phase**. Initialise the learning using a small chunk of initial training data $D_0 = \{(\mathbf{x}_p, \mathbf{t}_p)_{p=1}^{P_0}$ from the given training data $D = \{(\mathbf{x}_p, \mathbf{t}_p) | \mathbf{x}_p \in \mathbf{R}^N, \mathbf{t}_p \in \mathbf{R}^{\tilde{N}}, p = 1, \ldots\}, P_0 \geq L$.

  a) Assign random MF parameters $(\mathbf{c}_i, a_i), i = 1, \ldots, L$.

  b) Calculate the initial matrix $\mathbf{H}_0$ for the TSK models

$$\mathbf{H}_0 = \mathbf{H}(\mathbf{c}_1, \ldots, c_L, a_1, \ldots, a_L; \mathbf{x}_1, \ldots, \mathbf{x}_{N_0}) \tag{15}$$

  c) Estimate the initial parameter matrix $\mathbf{Q}^{(0)} = P_0 \mathbf{H}_0^T \mathbf{T}_0$ where $\mathbf{P}_0 = (\mathbf{H}_0^T \mathbf{H}_0)^{-1}$ and $\mathbf{T}_0 = [\mathbf{t}_1, \ldots, \mathbf{t}_{\tilde{N}_0}]^T$

  d) Set $s = 0$

- **Step 2) Sequential learning phase**. Present the $(s+1)$th chunk of new observations $D_{l+1} = \{(\mathbf{x}_p, \mathbf{t}_p)\}_{p=(\sum_{l=0}^{s} P_l)+1}^{\sum_{l=0}^{s+1} P_l}$, where P_{s+1} is the number of observations in the $s+1$ chunk. In this step, compute:

  a) The partial matrix $\mathbf{H}_{s+1}$ for the $(s+1)$th chunk of data D_{s+1} for the TSK model, where $\mathbf{H}$ is obtained as:

$$\mathbf{H}_{s+1} = \mathbf{H}(\mathbf{c}_1, \ldots, c_L, a_1, \ldots, a_L; \mathbf{x}_{(\sum_{l=0}^{s} P_l)+1}, \ldots, \mathbf{x}_{\sum_{l=0}^{s+1} P_l}) \tag{16}$$

  where the matrix $\mathbf{H}$ is defined as (13). Set $\mathbf{T}_{s+1} = \mathbf{t}_{(\sum_{l=0}^{s} P_l)+1}, \ldots, \mathbf{t}_{\sum_{l=0}^{s+1} P_l})$

  c) Compute the parameter matrix $\mathbf{Q}^{(s+1)}$ as follows:

$$\mathbf{P}_{s+1} = \mathbf{P}_s - \mathbf{P}_s \mathbf{H}_{s+1} (\mathbf{I} + \mathbf{H}_{s+1} \mathbf{P}_s \mathbf{H}_{s+1})^{-1} \mathbf{H}_{s+1} \mathbf{P}_s$$

$$\mathbf{Q}^{(s+1)} = \mathbf{Q}^{(s)} + \mathbf{P}_{s+1} \mathbf{H}_{s+1}^T (\mathbf{T}_{s+1} - \mathbf{H}_{s+1} \mathbf{Q}^{(s)}) \tag{17}$$

In general, OS-Fuzzy-ELM involves two main phases, namely, a) a initialisation phase, where TSK FIS is trained using a batch data set that is discarded once the initialisation phase is finished. In the second phase,

2.4. Interval Type-2 Fuzzy Extreme Learning Machine (IT2-FELM)

IT2-FELM is a learning algorithm based on ELM theory for the fast training of IT2 FISs either of Takagi-Sugeno-Kang (TSK) or Mamdani type that are functionally equivalent to SLFNs. Similar to type-1 FISs, an IT2 FIS consists of a fuzzifier, a fuzzy rule base and an output processor. The main difference lies on the output processor of an IT2 FIS, where it includes a type-reduction + defuzzification process. The type-reducer produces a type-1 fuzzy set output and the defuzzifier transforms such set into a crisp number. A type-2 FLS is again characterised by IF–THEN rules, but its antecedent or consequent sets are now type-2. An interval type-2 fuzzy set $\tilde{A}$ (IT2 FS) is characterised by a three-dimensional MF, or a bivariate function on the Cartesian product, $\mu : X \times [0,1]$ into $[0,1]$, where X is the universe of the primary variable x. To illustrate the concept of an IT2 FS, in Fig. 1, an IT2 Gaussian MF is presented, in which the point value representation of $\tilde{A}$ is defined by:

$$\tilde{A} = \{(x,u), \mu_{\tilde{A}}(x,u) = 1 | \forall x \in X, \forall u \in [0,1]\} \tag{18}$$

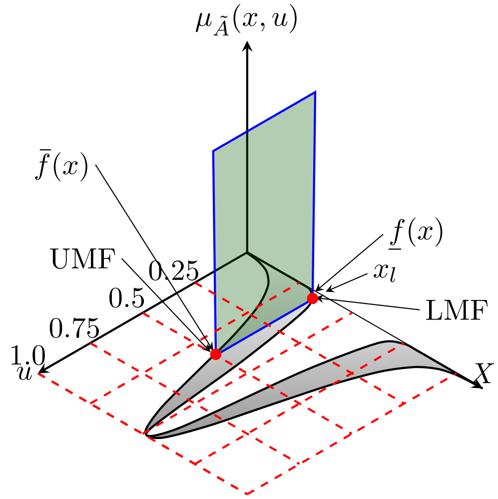


Fig. 1. Example of an Interval Type-2 Gaussian Membership Function, singleton fuzzification when $x = x_l$ (Taken from [36]).

As illustrated in Fig. 1, the 2-D support of $\mu_{\tilde{A}}$ is called Footprint of Uncertainty (FOU - shadowed area):

$$FOU(\tilde{A}) = \{(x,u) \in X \times [0,1] | \mu_{\tilde{A}}(x,u) > 0\} \tag{19}$$

where $FOU(\tilde{A})$ is bounded by its lower and upper membership functions $[\underline{f}(x), \bar{f}(x)]$ (LMF, UMF) respectively. For $'P'$ distinct training samples $(\mathbf{x}_p, \mathbf{t}_p)$, where $\mathbf{x}_p = \{x_{p1}, \ldots, x_{pN}\} \in \mathbf{R}^N$ is an input vector, and $\mathbf{t}_p = [t_{p1}, \ldots, t_{p\tilde{N}}]^T \in \mathbf{R}^{\tilde{N}}$ the corresponding target, a mathematical model for the jth output of a Multiple-Input-Multiple-Output (MIMO) IT2 FIS of TSK type with $'M'$ fuzzy rules and with a center-of-sets type-reduction is given by [36, 44, 45]:

$$y_p^j = \frac{1}{2}\left(y_l^j + y_r^j\right);\ j = 1, \ldots, \tilde{N} \tag{20}$$

where $\tilde{N}$ is the number of outputs of an IT2 FIS. In such a scheme, a product inference rule base for a TSK IT2 FIS is considered, where each rule is given by [37]:

$$R^i : \text{IF } x_{p1} \text{ is } \tilde{A}_{i1} \text{ AND } x_{p2} \text{ is } \tilde{A}_{i2} \text{ AND } \ldots \text{IF } x_N \text{ is } A_{iN} \text{ THEN } w_{ji} = q_{ji,1}x_1 + \ldots q_{ji,N}x_N \tag{21}$$

$k = 1 \ldots, N, i = 1, \ldots, M$ and each $\tilde{A}_{ik}$ is an IT2 FS of the kth input variable x_k [46]. If w_{ji} is a crisp value, the FIS is of Mamdani type. From Eq. (5), an IT2 FS $\tilde{A}_{ik}$ uses a primary Gaussian MF with a variable width $[\sigma_{ik}^1, \sigma_{ik}^2]$ and a fixed mean m_{ik}. The Footprint Of Uncertainty (FOU) of this MF is defined by its lower and upper MF $[\underline{\mu}_{\tilde{A}_{ik}}, \overline{\mu}_{\tilde{A}_{ik}}]$:

$$[\underline{\mu}_{\tilde{A}_{ik}}, \overline{\mu}_{\tilde{A}_{ik}}] := \begin{cases} \exp\left[-\frac{1}{2}\left(\frac{x_{pk}-m_{ik}}{\sigma_{ik}^1}\right)^2\right] \\ \exp\left[-\frac{1}{2}\left(\frac{x_{pk}-m_{ik}}{\sigma_{ik}^2}\right)^2\right] \end{cases} \tag{22}$$

The firing strength $\tilde{F}^i$ of each ith fuzzy rule is then obtained by performing fuzzy meet operation with the inputs using an algebraic product operation as follows [47]:

$$\tilde{F}^i = [\underline{f}_i(\vec{x}_p), \bar{f}_i(\vec{x}_p)] \tag{23}$$

$$\underline{f}_i(\vec{x}_p) = \prod_{i=1}^{M} \underline{\mu}_{ik}, \quad \bar{f}_i(\vec{x}_p) = \prod_{i=1}^{M} \overline{\mu}_{ik} \tag{24}$$

Each consequent of IT2-FELM is a linear model expressed as:

$$w_{ji} = q_{ji,0}x_0 + q_{ji,1}x_1 + \ldots q_{ji,N}x_N = \sum_{k=1}^{N} q_{ji,k}x_k \tag{25}$$

Therefore, the type-reduced set of the jth output (y_l^j, y_r^j) can be obtained using the Enhanced Karnik-Mendel algorithm (EKM) [48]. As indicated in [16], for $\tilde{N} > 1$, each output y_p^j of a TSK IT2-FELM can be associated to a submatrix representation as follows:

$$y_p^j = \frac{1}{2}\left(\mathbf{Y}_l^j + \mathbf{Y}_r^j\right)\mathbf{w}_i^T, \ j = 1, \ldots, \tilde{N} \tag{26}$$

in which $y_l^j = \mathbf{Y}_l^j \mathbf{w}_j^T$ and $y_r^j = \mathbf{Y}_r^j \mathbf{w}_j^T$ and

$$\mathbf{Y}_l^j = \frac{\bar{\mathbf{f}}^T Q_j^T E_{1j}^T E_{1j} Q_j + \underline{\mathbf{f}}^T Q_j^T E_{2i}^T E_{2j} Q_j}{r_l^T Q_j \bar{\mathbf{f}} + s_{lj}^T Q_j \underline{\mathbf{f}}} \tag{27}$$

$\mathbf{w}_j = [w_{j1}, \ldots, w_{jM}]^T$ is the set of original rule-ordered consequent weights, and $\mathbf{Y}_l^i = (\psi_{lj,1}, \ldots, \psi_{lj,M})$, and the terms E_{1j}, E_{2j}, r_{lj} and s_{lj} are defined as:

$$\begin{aligned}
E_{1j} &= (e_{1j}|e_{2j}|\ldots|e_{Lj}|\mathbf{0}|\ldots|\mathbf{0})^T \ \ L_i \times M \\
E_{2j} &= \left(\mathbf{0}|\ldots|\mathbf{0}|\xi_1^j|\xi_2^j|\ldots|\xi_{M-L_j}^j\right)^T \ \ (M - L_i) \times 1 \\
r_{lj} &\equiv (\underbrace{1, 1, \ldots, 1}_{L_j}, 0, \ldots, \ldots, 0)^T \ M \times 1 \\
s_{lj} &\equiv (0, \ldots, \ldots, 0 \overbrace{1, 1, \ldots, 1}^{M-L_j})^T \ M \times 1
\end{aligned}$$

in which L_j is the switching point that corresponds to the jth output, $e_m \in R_i^L$ $(m = 1, \ldots, L_i)$ and $\xi_m \in R^{M-L_i}$, $m = 1, \ldots, M - L_i$ as the elementary vectors where all the elements are zero except the mth one that is equal to 1.

$$\mathbf{Y}_r^j = \frac{\underline{\mathbf{f}}^T Q_i^T E_{3i}^T E_{3i} Q_i + \bar{\mathbf{f}}^T Q_i^T E_{4i}^T E_{4i} Q_i}{r_{ri}^T Q_i \underline{\mathbf{f}} + s_{li}^T Q_i \bar{\mathbf{f}}} \tag{28}$$

where $\mathbf{Y}_r^j = (\psi_{ri,1}, \ldots, \psi_{ri,M})$

$$\begin{aligned}
E_{3i} &= (e_{1i}|e_{2i}|\ldots|e_{Ri}|\mathbf{0}|\ldots|\mathbf{0})^T \ \ R_i \times M \\
E_{4i} &= (\mathbf{0}|\ldots|\mathbf{0}|\xi_{1i}|\xi_{2i}|\ldots|\xi_{M-R_i})^T \ \ (M - R_i) \times 1 \\
r_{ri} &\equiv (\underbrace{1, 1, \ldots, 1}_{R_i}, 0, \ldots, \ldots, 0)^T \ M \times 1 \\
s_{ri} &\equiv (0, \ldots, \ldots, 0 \overbrace{1, 1, \ldots, 1}^{M-R_i})^T \ M \times 1
\end{aligned}$$

with $e_m \in R^{Ri}$ $(m = 1, \ldots, R_i)$ and $\xi_m \in R^{M-R_i}$, $i = 1, \ldots, M - R_i$ as the elementary vectors where all the elements are zero except the mth one that is equal to 1 [49]. $\underline{\mathbf{f}} = (\underline{f}_1, \ldots, \underline{f}_M)^T$, $\bar{\mathbf{f}} = \left(\bar{f}_1, \ldots, \bar{f}_M,\right)^T$. Using Eq. (27) and (28), Eq. (26) can be expressed as:

$$\begin{aligned} y_p^j &= \frac{1}{2}\left(y_l^j + y_r^j\right) w = \frac{1}{2}\sum_{j=1}^{M}(\psi_{li,j} + \psi_{ri,j})w_{ij} \\ &= \frac{1}{2}\sum_{j=1}^{M} h_{pj}^i \left(\sum_{k=0}^{N} x_k q_{ij,k}\right), \ x_{ij,0} = 0, q_{ij,0} = 1; \end{aligned} \tag{29}$$

Where, $h_{pj}^i = (\psi_{li,j} + \psi_{ri,j})$. Therefore, Eq. (29) can be expressed as:

$$y_p^j = \boldsymbol{\phi}_i \mathbf{q} \tag{30}$$

where $\mathbf{q} = [q_{i1,k}, \ldots, q_{i1,k}, \ldots, q_{iM,k}, \ldots, q_{iM,k}]^T$ and $\boldsymbol{\phi}_i$ is

$$\begin{aligned} \boldsymbol{\phi}_p = \frac{1}{2}[(\psi_{li,1} + \psi_{ri,1})x_{p1}, \ldots, \psi_{li,1} + \psi_{ri,1})x_{pN}, \ldots \\ (\psi_{li,M} + \psi_{ri,M})x_{p1}, \ldots, \psi_{li,M} + \psi_{ri,M})x_{pN}]^T, \in \mathbf{R}^{M\times N} \end{aligned} \tag{31}$$

For P input vectors $\mathbf{x}_p$, a submatrix $\mathbf{H_A}$ can be written as

$$\mathbf{H_A}(\mathbf{x}) = (\boldsymbol{\phi}_1 \ \boldsymbol{\phi}_2 \ \ldots \ \boldsymbol{\phi}_P)^T \in \mathbf{R}^{P\times(M\times N)} \tag{32}$$

According to ELM theory and IT2 Fuzzy Logic, for a multidimensional output $\mathbf{T}$, a linear subsystem is required to determine each ith output in the OIT2-FELM. As indicated in [16, 50], at the heart of a TSK FIS, fuzzy modelling can be viewed as a process where the input data space is segmented into fuzzily defined regions which are parameterised and associated with a linear subsystem. In other words, a Multi-Input-Multi-Output FIS (MINO FIS) such as an IT2-FELM can be viewed as a linear combination of a joint block structured pattern that consists of a group of MISO fuzzy models [50]. Therefore, a linear subsystem can be defined for each ith output, where $\mathbf{H_A}$ can be now called $\mathbf{H}_\mathbf{A}^i$.

$$\mathbf{H}_\mathbf{A}^i(\mathbf{x})\mathbf{w}_i = \mathbf{t}_p, \ \mathbf{w}_i \in \mathbf{R}^{M\times\tilde{N}} \tag{33}$$

Thus, consequent parameters are estimated with a common block structure over all dimensions of the output variable as:

$$\mathbf{H}_\mathbf{B}^1\mathbf{Q}_1 + \ldots + \mathbf{H}_\mathbf{B}^i\mathbf{Q}_i + \ldots + \mathbf{H}_\mathbf{B}^{\tilde{N}}\mathbf{Q}_{\tilde{N}} = \mathbf{T} \tag{34}$$

$$\mathbf{H}_\mathbf{B}^i\mathbf{Q}_i = \begin{bmatrix} h_{11}^i & \ldots & h_{1M}^i \\ h_{21}^i & \ldots & h_{2M}^i \\ \vdots & \ldots & \vdots \\ h_{P1}^i & \ldots & h_{PM}^i \end{bmatrix} \begin{bmatrix} 0 & \ldots & w_{1i} & \ldots & 0 \\ 0 & \ldots & w_{2i} & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \\ 0 & \ldots & w_{Mi} & \ldots & 0 \end{bmatrix}$$

where $\mathbf{H}_\mathbf{B}^i \in \mathbf{R}^{P\times(M\times N)}$, $\mathbf{Q}_i \in \mathbf{R}^{(M\times N)\times\tilde{N}}$, and the target $\mathbf{T}$ is a matrix defined as follows:

$$\mathbf{T} = \begin{bmatrix} t_{11} & \ldots & t_{1\tilde{N}} \\ t_{21} & \ldots & t_{2\tilde{N}} \\ \vdots & \ldots & \vdots \\ t_{P1} & \ldots & t_{p\tilde{N}} \end{bmatrix}$$

in which, $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_{\tilde{N}}]$, is the desired output vector, and each $\mathbf{t}_i = [t_{1i}, \ldots, t_{Pi}]$. To determine the parameters of an IT2-FELM, a three-step process is implemented as follows:

- **Step 1. Random initialisation** of each MF's parameter m_{jk} and σ_{jk}
- **Step 2. Initialisation of each consequent** $q_{ij,k}$ using any of the two following methods:

  a) Calculate the initial value of each consequent $q_{ij,k}$ from the following linear system ($\tilde{N} \geq 1$):

$$\mathbf{Q}_A = \mathbf{H}_0^{\dagger}\mathbf{T} \tag{35}$$

  where $\mathbf{H}_0 = [\mathbf{h}_1, \ldots, \mathbf{h}_P]^T$, $p = 1, \ldots, P$ in which

$$\mathbf{h}_p = \frac{1}{2}[(y_l^1 + y_r^1)x_{p1}, \ldots, (y_l^1 + y_r^1)x_{pN}, \ldots (y_l^M + y_r^M)x_{p1}, \ldots, (y_l^M + y_r^M)x_{pN}]^T \tag{36}$$

  where $\mathbf{h}_p \in \mathbf{R}^{1\times(M\times N)}$. To calculate $\mathbf{Q}_A$, the value of y_l^j and y_r^j is obtained as:

$$y_l^j = \sum_{i=1}^{M} \underline{f}_i' w_{ji} \ , \underline{f}_i' = \frac{\underline{f}_i}{\sum_{i=1}^{M} \underline{f}_i} \tag{37}$$

$$y_r^j = \sum_{i=1}^{M} \bar{f}_i' w_{ji} \ , \bar{f}_i' = \frac{\bar{f}_i}{\sum_{i=1}^{M} \bar{f}_i} \tag{38}$$

  Using Eq. (37) and (38), $\mathbf{h}_p$ can be now computed as:

$$\mathbf{h}_p = \frac{1}{2}[(\underline{f}_1' + \bar{f}_1')x_{p1}, \ldots, (\underline{f}_1' + \bar{f}_1')x_{pN}, \ldots, (\underline{f}_M' + \bar{f}_M')x_{p1}, \ldots, (\underline{f}_M' + \bar{f}_M')x_{pN}] \tag{39}$$

  b) Calculate each entry of $\mathbf{H}_0$ using a close-form approach such as the Nie-Tan method:

$$\varphi_{pj} = \frac{\underline{f}_i + \bar{f}_i}{\sum_{i=1}^{M} \underline{f}_i + \sum_{i=1}^{M} \bar{f}_i} \tag{40}$$

  Therefore, Eq. (39) is defined by

$$\mathbf{h}_p = \frac{1}{2}[\varphi_{p1}x_{p1}, \ldots, \varphi_{p1}x_{pN}, \ldots \varphi_{pM}x_{pM}, \ldots, \varphi_{pM}x_{pN}] \tag{41}$$

- **Step 3. Refinement of each consequent** $q_{ji,k}$. Use the initial matrix $\mathbf{Q}_A$ to find the switching points L_j and R_j by applying the EKM to each jth sublinear system. Use this information to build each matrix $\mathbf{H}_{\mathbf{B}}^j$ and to refine each consequent matrix $\mathbf{Q}_j$ by:

$$\mathbf{Q}_j = (\mathbf{H}_{\mathbf{B}}^j)^{\dagger}\mathbf{t}_j \tag{42}$$

Finally, compute the target vector $\mathbf{T}$ using Eq. (33).

Table 1: SC algorithm for computing the end points y_l^j and y_r^j for each output of an EIT2-FELM.

Step	Computing y_l^j	Computing y_r^j
1	$\underline{f}_i = 0, \forall i \in [1, M]$, then	
	$y_l^j = \min(\underline{w}_{ji})$,	$y_r^j = \max(\overline{w}_{ji})$
	$\forall i \in [1, M]$ with $\bar{f}_i \neq 0$, Stop	
2	Initialise $z_i = 1$, $\Delta w_{ji} = \bar{f}_i - \underline{f}_i, \forall i \in [1, M]$	
3	Calculate: $\left\{ \delta_1 = \sum_{i=1}^{M} \bar{f}_i, \delta_2 = \sum_{i=1}^{M} \bar{f}_i w_{ji} \right\}$	
4	$flag = 0$	
5	For $i = 1$ to M, repeat the following operations of this step	
	$A_i = x_i \delta_1 - \delta_2$	
	If $A_i < 0$	If $A_i > 0$
	$z_i' = 1$, else $z_i' = 0$	
	If $z_i' \neq z_i$ then	
	If $z_i = 1$, $\begin{cases} flag = 1, \ z_1 = z_1 + \Delta w_i \\ z_i = z_i', \quad z_2 = z_2 + w_j \Delta w_i \end{cases}$	
	Else $\begin{cases} flag = 1, \ z_1 = z_1 - \Delta w_i \\ z_i = z_i', \ z_2 = z_2 - w_i \Delta w_i \end{cases}$	
6	If $flag \neq 0$ go to step **4**; else	
	$c_l = \delta_2/\delta_1$; $z_{li,j} = z_i$	$c_r = \delta_2/\delta_1$; $z_{ri,j} = z_i$

2.5. SC type-reduction Algorithm

Because of their iterative nature, the use of KM algorithms in type-reduction may hinder the deployment of IT2-FLSs to solve certain cost-sensitive real world problems [48]. This additional computational load in KM algorithms usually results from determining the switching points. Thus, a number of approaches for reducing the computational cost of IT2 FLSs has been suggested [48]. Such algorithms include three types of categories, namely 1) enhancements of the KM algorithm, 2) close-form type reduction methods and 3) the last category consists of simplified architectures of IT2 FLSs, which can be combined with any of the two categories mentioned above. In this section, a simplified version of the Center of Set Type Reducer Without Sorting Requirement Algorithm (COSTRWSR) [39] that is called SC algorithms is revisited. The SC algorithm is a center of set type reducer without sorting requirement that finds each $[y_l^j, y_r^j]$ based on a property of derivatives. Due to the functional equivalence between SLFNs and IT2 FISs either of Takagi-SUgeno-Kang (TSK) or Mandani type, the SC alforithm can be directly applied to determine the type-reduced set $[y_l^i, y_r^i]$. As pointed out in [39], finding the centroids y_l and y_r can be seen as a process of determining the max and min values of Y_{COS}. Thus, Eq. (37) and (38) can be reformulated as [39]:

$$y_l^j = \frac{\sum_{i=1}^{M_f} \bar{f}_i w_{ji} - \sum_{i=1}^{M_f} (1 - z_{li,j}) \Delta u_{ji} w_{ji}}{\sum_{i=1}^{M_f} \bar{f}_j - \sum_{i=1}^{M_f} (1 - z_{li,j}) \Delta u_{ji}} \tag{43}$$

And

$$y_r^j = \frac{\sum_{i=1}^{M_f} \bar{f}_i w_{ji} - \sum_{i=1}^{M_f} (1 - z_{ri,j}) \Delta u_{ji} w_{ji}}{\sum_{i=1}^{M_f} \bar{f}_i - \sum_{i=1}^{M_f} (1 - z_{ri,j}) \Delta u_{ji}} \tag{44}$$

in which, $\Delta u_{ji} = \bar{f}_i - \underline{f}_i, \forall i \in [1, M]$ is the difference of the Upper and Lower Membership Functions (UMF, LMF). Where terms $[z_{li,j}, z_{ri,j}]$ can take the values from the interval $[0, 1]$, and w_{ji} is the corresponding consequent part (weight). Moreover, if the values for $[z_{li,j}, z_{ri,j}]$ are taken either equal to *zero* or *one*, the resulting formula to determine each output y_p^j in an IT2 FIS (of TSK or Mamdani type) can be determined using the SC algorithm as shown in Table 1. Thus, Eq. (43) and (44) are two alternatives to KM algorithms where the need of a sorting process is eliminated. That means, the computing of terms L_j and R_j do not exist anymore. The SC type reducer is a simplified version of the Center of Set Type Reducer Without Sorting Requirement Algorithm (COSTRWSR) [39]. Hence, as detailed in Table 1, the SC algorithm finds each $[y_l^n, y_r^n]$ based on a property of derivatives. Compared to COSTRWSR, using the SC algorithm, $[y_l^n, y_r^n]$ can be obtained without adding the extra parameters $[z_{li,j}, z_{ri,j}]$ [51].

3. Methods

3.1. Proposed OIT2-FELM for Data Classification

As indicated in [37], the majority of applications of Interval Type-2 Fuzzy Extreme Learning Machine (IT2-FELM) has been concentrated on the solution of regression problems. Such systems are usually Multi-Input-Single-Output (MISO) systems with a Karnik-Mendel (KM) type-reduction where their training is in batch mode. In many real world applications sequential online learning in IT2 FISs may be preferred over batch learning as they do not require retraining whenever new data is received [52]. In this study, an Online sequential learning methodology for the training of a class of Interval Type-2 Fuzzy Inference Systems (IT2 FISss) of Takagi-Sugeno-Kang (TSK) is suggested. The proposed learning methodology called Online Interval Type-2 Fuzzy Extreme Learning Machine (OIT2-FELM) extends the concept of Online Sequential ELM (OS-ELM) for type-1 FISs to IT2 FISs where training data can be presented one-by-one or chunk-by-chunk with a fixed or varying size [44, 53].

To illustrate the proposed OIT2-FELM, a Multiple-Input-Multiple-Output (MIMO) IT2 FIS with a Gaussian MF $[\underline{\mu}_{\tilde{A}_{jk}}, \overline{\mu}_{\tilde{A}_{jk}}]$ having a fixed mean m_{jk}, variable standard deviation $[\sigma_{jk}^1, \sigma_{jk}^2]$, a predefined number of $'L'$ fuzzy rules is used. To eliminate the need of sorting each consequent part and creating a number of sublinear systems that implies using KM algorithms, a SC type-reduction is incorporated in the output layer of the IT2 FIS. As described in [54], a FIS is considered of interval type-2 if only one of its components uses Interval Type-2 Fuzzy Sets (IT2-FSs). Hence, in this study, each consequent part is defined as $\underline{w}_{ij} = \overline{w}_{ij} = w_{ij}$. Given a number of $'P'$ distinct different training samples $(\mathbf{x}_p, \mathbf{t}_p)$, the implementation of OIT2-FELM follows two main phases, i.e.: 1) initialisation and 2) sequential learning phase.

- **Step 1 - Initialisation phase:** Select an initial chunk of data $D_0 = \{x_p, t_p\}_{p=1}^{P_0}$ from a given training data set $D = \{(\mathbf{x}_p, \mathbf{t}_p) | \mathbf{x}_p \in \mathbf{R}^n, \mathbf{t}_p \in \mathbf{R}^{\tilde{N}}\}$ where $P_0 \geq M$ and its corresponding target vector $\mathbf{t}_p = [t_{p1}, \ldots, t_{p\tilde{N}}]$.

  $S1.a)$ Randomly generate the initial values to the MF parameters, for example, if a Gaussian MF of IT2 is selected, initial values of $[\sigma_1, \sigma_2]$ and m_k are randomly generated.

  $S1.b)$ For a TSK IT2-FIS, based on the procedure described in Eq. (35), initialise each consequent weight $q_{ij,k}$ by computing the initial consequent matrix $\mathbf{Q}_A^{(0)}$. A similar procedure can be implemented for a Mamdani fuzzy model. Note, this initialisation process is performed once.

  $S1.c)$ Use the SC algorithm to calculate each term $[z_{li,j}, z_{ri,j}]$. Build the matrix $\mathbf{H}_B^{(0)}$

$$\mathbf{H}_B^{(0)}(\boldsymbol{\sigma}_1^1, \ldots, \boldsymbol{\sigma}_M^1, \boldsymbol{\sigma}_1^2, \ldots, \boldsymbol{\sigma}_M^2, m_1, \ldots, m_M; \mathbf{x}_p) = [\mathbf{h}_1^{(0)}, \ldots, \mathbf{h}_P^{(0)}]^T \tag{45}$$

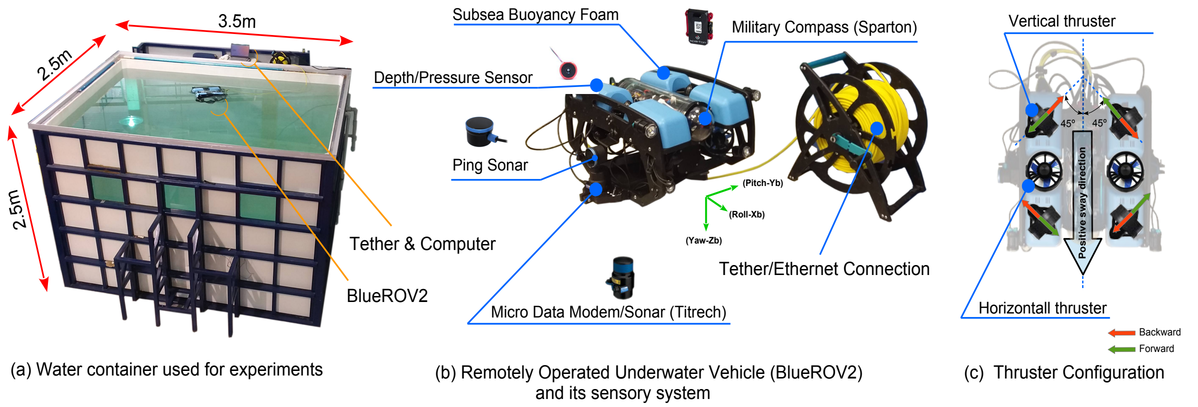


Fig. 2. (a) Water container (open-tank) and computer station, and an (b) ROV (BlueROV2) equipped with a sensory system with four sensors used for all experiments.

in which, $p = 1, \ldots, P$ and each vector $\mathbf{h}_p^{(0)}$ is defined as:

$$\mathbf{h}_p^{(0)} = [\phi_{p1}x_{p1}, \ldots, \phi_{p1}x_{pN}, \ldots, \phi_{pM}x_{pM}, \ldots, \phi_{pM}x_{pN}] \tag{46}$$

where each entry ϕ_{pi} is defined as follows:

$$\phi_{pi} = \frac{1}{2}\left(\frac{\bar{f}_i - (1 - z_{li,j})\Delta w_{ji}}{\sum_{i=1}^{M}\bar{f}_i - \sum_{i=1}^{M}(1 - z_{li,j})\Delta w_{ji}} + \frac{\bar{f}_i - (1 - z_{ri,j})\Delta w_{ji}}{\sum_{i=1}^{M}\bar{f}_i - \sum_{i=1}^{M}(1 - z_{ri,j})\Delta w_{ji}}\right) \tag{47}$$

in which, $\Delta w_{ji} = \bar{f}_i - \underline{f}_i, \forall i \in [1, M]$.

$S1.d)$ Refine each consequent part $q_{i,jk}$ using the Equation below:

$$\mathbf{Q}_B^{(0)} = \mathbf{P}_0\mathbf{H}_{B,0}T_0 \tag{48}$$

where $\mathbf{P}_0 = (\mathbf{H}_{B,0}^T\mathbf{H}_{B,0})^{-1}$ and $\mathbf{T}_0 = [\mathbf{t}_1, \ldots, \mathbf{t}_{P_0}]^T$

$S1.e)$ Set $s = 0$

- **Step 2 - Sequential learning phase:** Present the $(s+1)$ chunk of new observations $\mathbf{D}_{s+1x} = \{x_p, t_p\}_{p=(\sum_{l=0}^{s} N_l)+1}^{\sum_{l=0}^{s+1} N_l}$ where P_{s+1} is the number of observations in the $(s+1)th$ chunk. Thus, the process of finding each consequent vector $\mathbf{Q}_B^{(s+1)}$ is as follows:

$S2.b)$ By using Eq. (33), calculate the partial matrix $\mathbf{Q}_A^{(s+1)}$ for the $(s+1)th$ data chunk for the TSK model in order to initialise each consequent part $q_{ji,k}$. Then, apply the SC algorithm to calculate each term $[z_{li,j}, z_{ri,j}]$. Build the matrix $\mathbf{H}_B^{(s+1)}$ using Eq. (46-47).

$$\mathbf{H}_B^{(s+1)}\left(\boldsymbol{\sigma}_1^1, \ldots, \boldsymbol{\sigma}_M^1, \boldsymbol{\sigma}_1^2, \ldots, \boldsymbol{\sigma}_M^2, m_1, \ldots, m_M; \mathbf{x}_{(\sum_{l=0}^{k} N_l)+1}, \ldots, \mathbf{x}_{\sum_{l=0}^{s+1} N_l}\right) \tag{49}$$

Set $\mathbf{T}_{s+1} = [\mathbf{t}_{(\sum_{l=0}^{s} P_l)+1}, \ldots \mathbf{t}_{\sum_{l=0}^{s+1} P_l}]^T$

$S2.c)$ Calculate the parameter matrix $\mathbf{Q}_B^{(s+1)}$ using Eq. (50):

$$\begin{aligned}\mathbf{P}_B^{(s+1)} &= \mathbf{P}_B^{(s)} - \mathbf{P}_B^{(s)}(\mathbf{H}_B^{(s+1)})^T(\mathbf{I} + \mathbf{M}_{(s+1)})^{-1}\mathbf{H}_B^{(s+1)}\mathbf{P}_B^{(s)} \\ \mathbf{Q}_B^{(s+1)} &= \mathbf{Q}_B^{(s)} + \mathbf{P}_B^{(s)}(\mathbf{H}_B^{(s+1)})^T(\mathbf{T}_B^{(s+1)} - \mathbf{H}_B^{(s+1)}\mathbf{Q}_B^{(s)})\end{aligned} \tag{50}$$

$$\text{in which } \mathbf{M}_{(s+1)} = \mathbf{H}_B^{(s+1)} \mathbf{P}_B^{(s)} \left(\mathbf{H}_B^{(s+1)} \right)^T$$

$S2.d)$ Set $S = S + 1$. Then, go to **Step 2**.

OIT2-FELM is a sequential learning algorithm that consists of two main phases. First, an initial batch of data is used to train a TSK IT2-FIS, where initialisation data is discarded once the first phase is completed. Similar to its type-1 counterpart the OS-Fuzzy-ELM, it is recommended that the number of the initial training data is at least the same to the number of fuzzy rules of the TSK IT2-FIS. As detailed in the OIT2-FELM algorithm, in the first phase, a direct defuzzification method can be applied to approximate the value of each consequent weight w_{ij}. After the initialisation phase, the TSK IT2-FIS will learn new data one-by-one or chunk-by-chunk (with fixed or varying size). The second step is an iterative phase that involves $\tilde{N}$ calculations (where $\tilde{N} \geq 1$).

3.2. Robotic Platform

The ROV used for experiments is a BlueROV2 with a six-thruster vectored configuration as described in Fig. 2. All experiments were carried out in an indoors water container of $2.5 \times 2.5 \times 3.5$ metres size (Fig. 2(a)). To reproduce some undersea conditions, salty water with a density of about $1028 kg/m^3$ was added to the container.

Four out of the six thrusters are oriented in a vertical direction while the remaining two are oriented horizontally (Fig. 2(c)). This gives the ROV the ability to move itself up-down as well as to control its yaw orientation and moves forward and backward (See Fig. 2(b)). The ROV's dynamics vehicle are such that the vertical motion is largely decoupled from the lateral motion. The vehicle is also very stable in the roll and pitch axes due to the righting moment induced by four subsea buoyancy foams 2(b)). The ROV has an open-source electronics whose sensory system (See Fig. 3) was integrated at the Laboratory of Submarine Robotics, (LSR, CIDESI). As detailed in Fig. 3, such system consists of a **1)** pressure sensor that is able to measure up to 30 Bar (300m depth) with a depth resolution of 2mm (Bluerobotics), **2)** a ping sonar which is an open-source sensor able to measure distances up to 30 meters with a 30 degree beam width, a **3)** micro data sonar Titrech with a range resolution of 7.5mm, a beam width of 3°, and a variable scanned sector and **(4)** the Sparton compass that is a micro-sized and light weight attitude heading sensor with a static heading accuracy of 0.2° RMS and full 360° rollover capability.

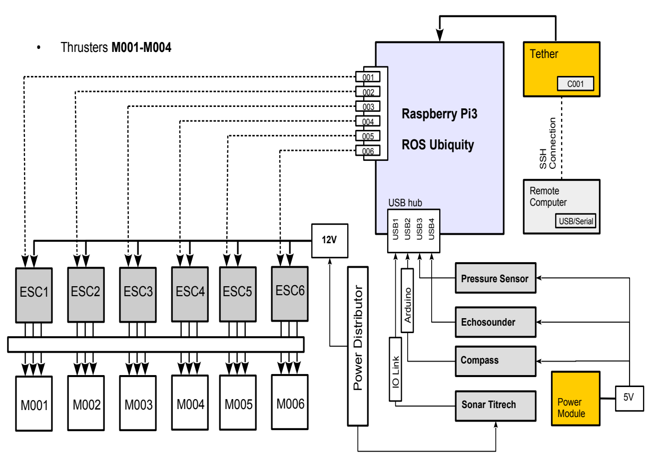


Figure 3: System's configuration used by the ROV (BlueROV2).

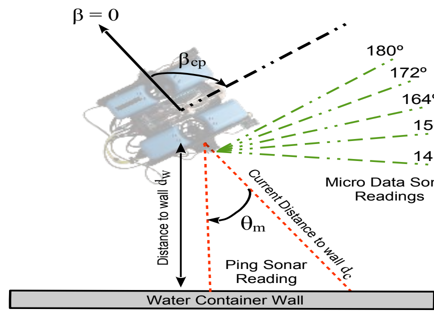


Figure 4: Scanning sector defined for the sonar.

The micron data sonar is used as a dynamic echo-sounder whose scanned sector is defined by a sample window of five beams separated at 8° one to the other as shown in Fig. 4. As indicated in Fig. 3, the main computer in the ROV is a Raspberri Pi3, in which the middleware Robotic Operating System (ROS ubiquity) was installed to implement all machine learning algorithms and controllers. In this work, Python, C++ and Matlab were the main coding languages used in ROS. A line SSH connection between the Raspberri Pi3 and an Ubuntu computer was used to monitor sensor values and define the parameters of each experiment (See Fig. 2).

3.3. Hierarchical Strategy Navigation (HNS) for path Following

In hierarchical fuzzy structures, the number of fuzzy rules increases linearly with the number of input variables rather than exponentially as in single fuzzy logic systems (FLSs) [34]. For example, an FLS with six input sensors and four fuzzy sets, its fuzzy rule base will contain $4^6 = 4096$ fuzzy rules. Such a big rule base is very difficult to design and very expensive in computational terms, leading to significant degradation in real-time response. Moreover, due to the monolithic structure of its fuzzy rule base, the implementation of single fuzzy logic structures that operate on multitask domains may become extremely complex. To cope with rule explosion, hierarchical architectures decompose general ROV's behaviour into a number of fuzzy local behaviours. In a hierarchical architecture (either fuzzy or crisp model), each ROV's behaviour is encoded as a mapping of each sensor's signal into a desired control policy.

The successful completion of a predefined path is directly related to the ability of the ROV to provide real-time planning with high autonomy. In this work, autonomy for the completion of predefined paths is achieved within a hierarchical navigation strategy (HNS) of fuzzy behaviours in which low-level navigation behaviours are realised using interval type-2 fuzzy logic control and the proposed OIT2-FELM, while high-level coordination behaviours are implemented as IF-THEN rules with singleton weights as detailed in Fig. 5(a). As depicted in Fig. 5(b, c), the proposed HNS follows a bottom-up hierarchy of increased behavioural complexity in which activity at a given level is the result of a function of behaviours given at the level below [5, 34]. In order to achieve multiple goals (or composite behaviours) whose priorities may change with time, the HNS decomposes overall ROV's motion into single fuzzy behaviours that serve single purpose by operating in a reactive or reflexive manner [5]. On the low-level navigation behaviours, single fuzzy behaviours are realised to perform nonlinear mappings from different subsets of the ROV's sensor suite to common actuators. These behaviours are building blocks for more intelligent behaviours where their coordination can be modulated on the high-level hierarchy suitable for complex goal-directed operations. In this paper, the completion of a 3D path-following task that consists of two circuits at different depth as illustrated in Fig. 6 is suggested. One circuit in the water container can be completed by following

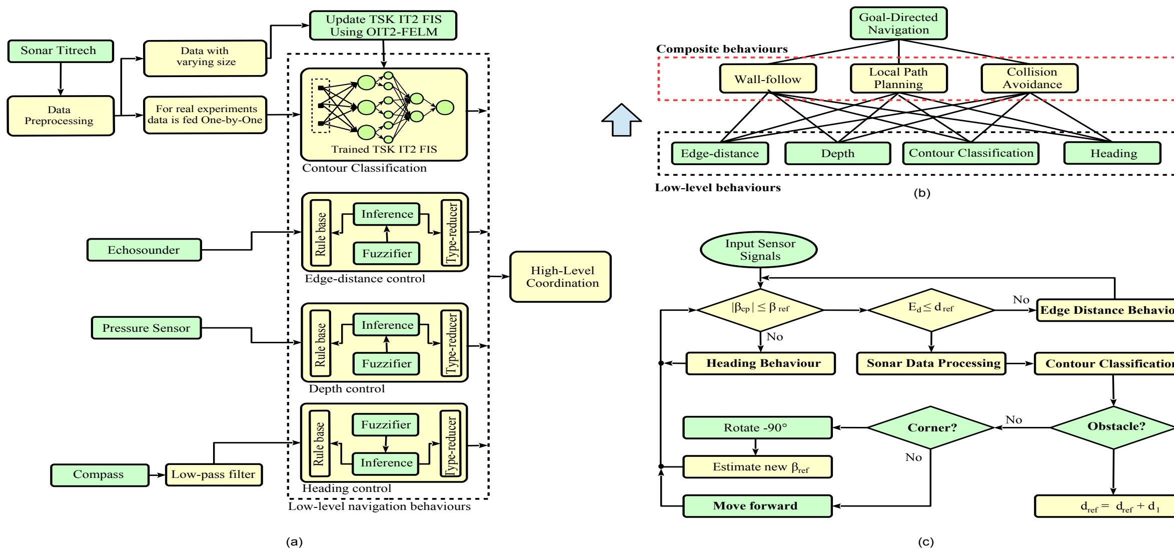


Fig. 5. (a) Hierarchical Navigation Strategy (HNS), (b) Hierarchical decomposition of the ROV's navigation, and (c) High-level coordination activated by IF-THEN rules in order to complete one circuit.

a clockwise/anticlockwise path navigating along walls 1-2-3-4/4-3-2-1 respectively. To complete one path, the proposed HNS decomposes goal-directed navigation as a behavioural function of wall-follow (or route-follow), local path planning (contour recognition when the ROV reaches a corner in the container) and collision avoidance (collision free-navigation to some location). These behaviours can be further decompose into local fuzzy behaviours, namely, 1) heading behaviour, 2) depth behaviour, 3) edge-distance control (minimum distance between a wall and the ROV) and 4) contour classification.

The first three low-level behaviors will be an interval type-2 Fuzzy Proportional Derivative Logic Controller (IT2 FPDFC) using interval type-2 fuzzy sets to represent the input and output variables of each behavior, while the last one will be the proposed OIT2-FELM. The first two behaviors provide the ROV with the necessary autonomy to navigate parallel to each wall at a predefined distance. The third behaviour consists of implementing an IT2 FPDC to control the vertical position of the ROV at a predefined depth. The fourth behaviour is related to the ability of the ROV to recognise the contour around it, i.e. what is it in front or next to? A corner, a wall or an object. In this behaviour, the proposed OIT2-FELM can learn incoming data that may arrive one-by-one or chunk by chunk (with fixed or varying size).

The idea behind the proposed HNS methodology is to integrate a number of control algorithms that guide the ROV to complete a circuit (as shown in Fig. 6). As illustrated in Fig. 5 (flow diagram), a compass and a pressure sensor are used to estimate the yaw angle (See Fig. 2(b)- axis Zb) and the ROV's vertical position (axis Zb) respectively. The ping sonar and the micro data sonar are utilised to estimate the distance between the ROV and the closest wall. While the ping sonar is a multipurpose single-beam echosounder used to estimate the distance between the ROV and obstacles on its right side (See Fig. 2), the micro data sonar provides five different distance readings (estimates) in a predefined scanned sector, where the value for reading 180° is aligned to the ROV's front as illustrated in Fig. 4. Next subsections describe the model of the Interval Type-2 Fuzzy Logic Controller used in each fuzzy behaviour and their implementation setup at low-level hierarchy as well as their coordination at high-level.

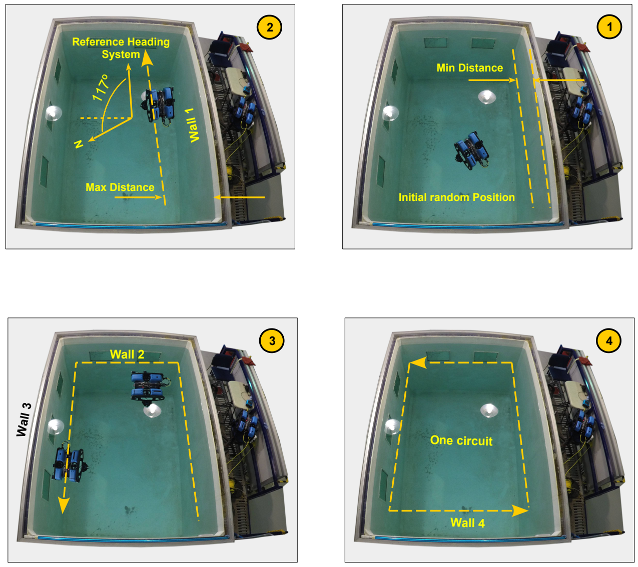


Figure 6: Trajectory followed by the ROV (BlueROV2) to complete one circuit.

3.4. IT2 FLC for heading, depth and edge-distance behaviour

Because of their model simplicity, set point tracking performance, easy to tune and high reliability at acceptable cost, the implementation of IT2 Fuzzy Logic PD and PI Controllers (IT2 FLCs) controllers have been employed for more than one decade not only in industrial control loops [55], but also in mobile robotics [34], submarine applications and medicine [56]. Compared to T1 Fuzzy Logic Controllers (T1-FLCs), the design and implementation of IT2-FLCs have demonstrated to achieve a superior performance while providing a smoother control surface around the origin in a large number of real world problems [57]. This is usually due to the extra dimension provided by the Footprint-Of-Uncertainty (FOU), and hence with the same number of MFs, an IT2-FLC offers more design freedom [57]. In this work, fuzzy behaviours for depth control, edge-distance control and heading control are implemented using an IT2 Fuzzy PD Controller (IT2 FPDC) whose structure follows the architecture illustrated in Fig. 7.

For simplicity, in our HNS, the structure of each fuzzy IT2 FPDC is configured with a predefined number of equidistant triangular MFs for inputs, and equidistant singletons for output. Each IT2 FPDC contains three triangular Membership Functions (MFs), in which the corresponding error (e) and derivative of error (Δe) are used as control inputs. The output of each IT2 FPDC is the Pulse Width Modulation (PWM) that controls the angular speed of each thruster in the ROV. Where the terms:

$$\begin{aligned} E(k) &= G_e e(k) = G_e(y_{ref} - y_f) \\ \Delta E(k) &= G_{\Delta e}\Delta e(k) = G_{\Delta e}(e(k) - e(k-1)) \end{aligned} \tag{51}$$

in which, y_f, y_{ref} is the output system and the reference signal respectively, and k is the sampling instance. The output variable $u(k)$ is calculated from the incremental output $\Delta u(k)$ and its previous value $u(k-1)$ as follows:

$$u(k) = u(k-1) + \Delta(k) \tag{52}$$

where $\Delta u(k)$ is obtained as:

$$\Delta u(k) = G_U U(k) \tag{53}$$

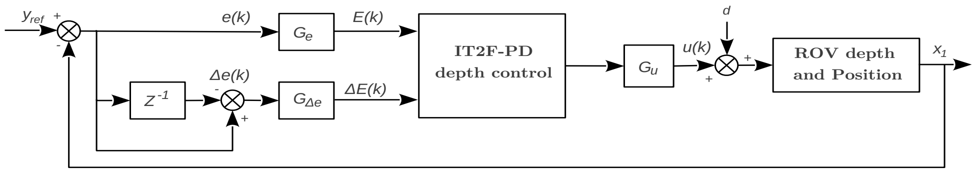


Fig. 7. (a) Structure of an IT2 Fuzzy PD Controller used to perform the heading, depth and edge distance behaviours.

where, G_U is an scaling factor. As detailed in Fig. 5, high-level decision-making in the HNS is performed by a number of IF-THEN rules that facilitates real-time intelligence by breaking down overall motion control into a set of IT2 fuzzy controllers with quasi-parallel execution.

3.5. Low-level Fuzzy Behaviours

3.5.1. Heading Behaviour

To navigate parallel to each wall in the water container, the ROV employs an IT2 FPDC to control its heading (yaw angle). The IT2 FPDC is a fuzzy architecture with three triangular MFs of IT2 and a Wu-Mendel type-reduction. As described in Fig. 5(c), heading behaviour is activated if the value of yaw angle is out of the limits $(\beta_{ref} - \beta_L) \leq \beta_{cp} \leq (\beta_{ref} + \beta_U)$, where β_{ref} is the reference yaw angle and $[\beta_L, \beta_U]$ are its lower and upper limit respectively. The current heading β_{cp} of the ROV is provided by a compass as an angle value which is low-pass filtered. To perform heading behaviour, an IT2 FPDC with three fuzzy rules, a Wu-Mendel (WM) type-reducer and one output to control each thruster's angular speed is suggested. To turn right/left, the ROV turn on all four vertical thrusters with a predefined configuration as shown in Fig. 2. As described in Fig. 6, at the beginning of each experiment (path), the ROV is randomly located in the water container. From there, by using the reference angle $\beta = 0$, the target heading β_{ref} that allows the ROV to keep a parallel alignment to its closest wall $\beta_{cp} = \beta_{ref}$ can calculated as shown in Fig. 4. Therefore, control inputs, $e = \beta_{ref} - \beta_{cp}$, and its change $\Delta e = e(k) - e(k-1)$. The angle β_{ref} can be computed by adding the current heading β_{cp} and the angle θ_m which are directly obtained from the compass and ping sonar readings correspondingly.

$$\beta_{ref} = \beta_{cp} + \theta_m \tag{54}$$

d_w is the target distance to the closest wall and d_c the current ping sonar reading measured in metres.

3.5.2. Depth behaviour

In underwater navigation, ROV's dynamics are highly nonlinear and for control purposes, the hydrodynamic coefficients of the vehicles are difficult to estimate under different underwater conditions. In this sense, IT2 fuzzy logic represents an ideal solution when a dynamic model of the ROV is not available. The depth behaviour is needed to control the vertical position of the ROV throughput the completion of one path. An IT2 FPDC with triangular MFs, a Wu-Mendel type-reducer and two inputs, i.e. e and Δe is considered.

$$e = d_{ref} - d_{ROV} \tag{55}$$

where d_{ref} is the desired depth, $d_{ROV}$is the current vertical position obtained by a depth sensor in cm. The change of e is calculated as $\Delta e = e(k) - e(k-1)$. Because of the dimensions of the water container, the largest value for d_{ref} is suggested as $d_{ref} = 2.3m$, while its smallest value is equal to $0.0m$. The output value of the proposed IT2 FPDC is the pulse width to regulate the angular speed of the two horizontal thrusters position at the center of the ROV chassis.

3.5.3. Edge Distance Behaviour

Edge distance behaviour is employed to follow a parallel trajectory along each wall at a desired distance. To execute edge distance behaviour, an echosounder was mounted on the right side of the ROV, where its reading is provided in *centimetres*. The signal obtained from the echosounder is the output of a low-pass filter used to discriminate signals with frequency lower than a selected cutoff frequency. To control lateral motion, an IT2F-PD controller with two inputs, i.e. error and change in error with respect to the distance measured by the echosounder is employed to regulate the angular speed of thrusters M001-M004 with a predefined configuration. The IT2F-PD controller has three MFs and a Wu-Mendel type reducer, in which the lower and upper boundaries of the FOU are determined experimentally.

3.5.4. Contour Classification

The proposed OIT2-FELM is integrated into the HNS structure to achieve two different goals. First, the OIT2FELM is utilised to train TSK IT2 FIS in order to classify the data coming from a micro data sonar and recognise the type of contour around the ROV. Three different types of contour are suggested, wall, corner and an irregular object (obstacle). Secondly, the information provided by the process of contour classification is used by the HNS for collision-avoidance and obstacle inspection. As shown in Fig. 5(a), for contour classification, the training of a TSK IT2 FIS follows a two-stage process. First, OIT2-FELM is applied to train TSK IT2 FIS using a static data set that is collected offline. Then, for real experiments, the trained TSK IT2 FIS is integrated into the structure of the proposed HNS, in which sonar data with varying size and collected during each experiment is fed to the TSK IT2 FIS. In the second stage, online data is preprocessed and used by the OIT2-FELM to update the parameters of TSK IT2 FIS. To input training data to TSK IT2 FIS, raw data coming from the sonar follows a two-stage preprocessing in which data is first low-pass filtered and then normalised to the interval $[0, 1]$. The initial training data is obtained by collecting sonar readings at different locations in the water container. As illustrated in Fig. 4, a scanning sector that consists of five angles, namely, $\mathbf{x}_k = [180°, 172°, 164°, 156°, 148°]$ is employed, in which k is the current sonar reading. The reading frequency in the compass at which data was collected was set equal to 0.02Hz. For contour classification in real experiments, a voting scheme based on a recursive accumulation on the average value of four consecutive outputs of the OIT2-FELM is implemented. This classification process is completed once a belief threshold β_e is exceeded, where $e = 1, \ldots, N_e$, and N_e is the total number of sonar readings fed into the TSK IT2 FIS. This action triggers a decision making process for the classification of each contour where the current predicted class is computed by:

$$\hat{c} = \arg\max_{c}(y_t(c|\mathbf{x}_k)) \tag{56}$$

in which c is used to denote the current class, and $\mathbf{x}_k$ is the sonar data collected at time k. Where y_t is calculated as:

$$y_t(c|\mathbf{x}_k) = \frac{1}{n_s}\sum_{k=k-n_s}^{k} y_i(\mathbf{x}_k) \tag{57}$$

where n_s is the number of consecutive samples used to compute the average value y_t, and $y_i(\mathbf{x}_k)$ is the output of the OIT2-FELM at time k. Finally, decision-making in the HNS is based on this information to provide the ROV with more autonomy and real time planning capabilities.

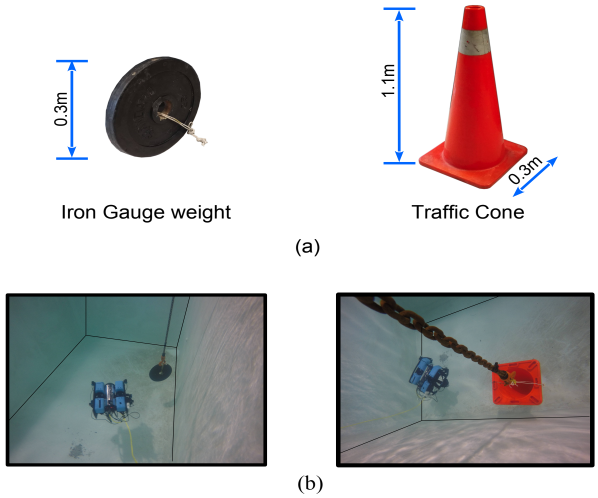


Figure 8: Objects used for the collection of sonar data located at either wall one or wall three.

3.6. High-Level Coordination

Coordination of low-level behaviours in the HNS is carried out by a high-level hierarchy that follows the principle operation of fuzzy architectures with singleton activation [5], in which the number of rules does not increase exponentially, but linearly with the number of input variables. Near-to-real-time response in the ROV is achieved by the concurrent operation of ROS and the rule-base structure of crisp IF-THEN rules that facilitates the HNS with quasi-parallel execution of two or more local behaviours. As described in Fig. 5(c), ROV's behaviour is encoded as a rule-base that maps relevant sensor inputs to control outputs according to a desired control policy or goal, while the HNS incorporates task-achieving behaviours whose priorities may change in time. Activation of fuzzy behaviours follow a scheme that employs a weighted decision-making based on each *qth* fuzzy rule as:

$$\begin{aligned} R^q : \text{Yaw angle is large } (\beta_{cp} \geq \beta_{ref} + 5^o) \text{ AND } \ldots \\ \text{ROV's depth is small } (d_{ROV} \leq d_{ref}) \ \ldots \\ \text{THEN depth and heading behaviour} \end{aligned} \tag{58}$$

3.7. Collection and Preprocessing of Training Data

The recognition of the type of contour was performed by the proposed OIT2-FELM. Initial chunks of data for training and testing were used for cross-validation purposes. A scanning data sonar installed in the ROV was used to collect data offline at different locations in the water container with a different depth, of which, a final data set of 1273 records was created. Each input vector in the data set consists of five attributes - each one corresponding to a distance measurement for the angles $\{180°, 172°, 164° \, 156°, 148°\}$ (See Fig. 4). For real experiments, online learning is continuously performed by the proposed OIT2-FELM, where new data may arrive one-by-one or chunk-by-chunk with a fixed or varying varying size.

In order to make online learning feasible, sonar data needed to be cleaned online. This cleaning process was necessary because sonar data - supplied as a set of five-dimensional points contained noisy and spurious information.

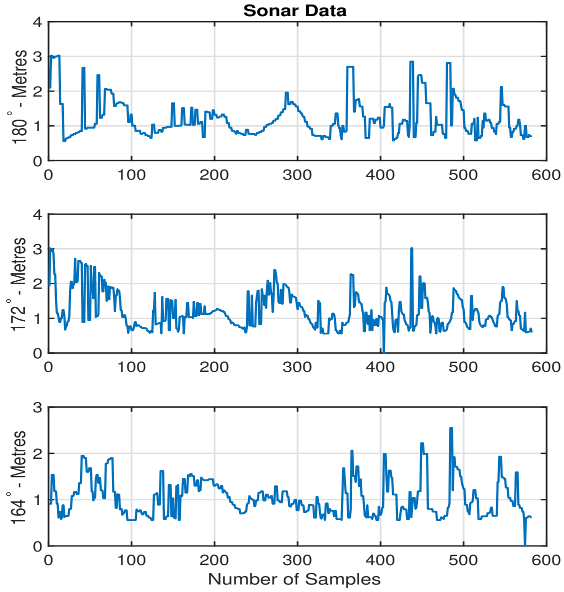


Figure 9: Example of sonar data used to cross-validate the proposed OIT2-FELM.

Especially, spurious values may result from multiple reflections due to the size of the water container and the materials used to build it. Other factors such as the pitching and the rolling of the ROV as well as inaccuracy and miscalibration of measurement devices may negatively affect data collection. Within this context, the purpose of the cleaning stage was not to remove noisy data but eliminate spurious readings and properly form input data to the OIT2-FELM. As shown in Fig. 8(a), to evaluate the performance of the proposed OIT2-FELM in the presence of irregular walls, two different objects are located next to either wall 1 or wall 3 during data collection. In this research work, a balanced data set for binary classification was collected as shown Fig. 9. 673 samples are used to denote a class that corresponds to sonar signals that describe a corner, while the remaining data is used to denote a wall or walls where an object has been placed at (See Fig. 8). Training was normalised to the range [0-1]. In Fig. 9, an example of 600 sonar records after cleaning for the angle $180^o, 172^o$ and 164^o is illustrated.

4. Performance Evaluation of the OIT2-FELM

To evaluate the proposed OIT2-FELM, three different types of experiments are suggested in this section. For the first two experiments, the OIT2-FELM is applied to solve popular benchmark problems in the areas of nonlinear system identification, regression and classification. Finally, the OIT2-FELM is integrated into a Hierarchical Navigation Strategy (HNS) of an ROV as its main navigation mechanism to complete a predefined 3D path. All simulations are carried out in MATLAB 16b environment running in a 2.7 GHz intel core i5 processor.

Table 2: PERFORMANCE FOR NONLINEAR SYSTEMS IDENTIFICATION.

	Model	Training (RMSE)		Testing (RMSE)		
		Mean	Time(s)	Mean	SD	# Rules
Gaussmf	OS-FELM	0.0781	2.290	0.0173	0.0119	15
	ANFIS	0.1311	145.0	0.0588	0.0078	27
	SAFIS	0.1493	22.47	0.0533	0.0103	30
	OIT2-FELM-SC	0.0662	2.990	0.0130	0.0100	15
	OIT2-FELM-NT	0.0701	2.410	0.0144	0.0911	15
	OIT2-FELM-EKM	0.0670	3.331	0.0139	0.0120	15
Cauchymf	OS-FELM	0.0773	2.197	0.0169	0.0124	15
	ANFIS	0.1264	150.8	0.0479	0.0096	27
	Simp1_eTS	0.3305	75.70	0.1169	0.0115	42
	OIT2-FELM-SC	0.0612	3.011	0.0117	0.0198	15
	OIT2-FELM-NT	0.0623	2.430	0.0128	0.0116	15
	OIT2-FELM-EKM	0.0609	3.761	0.0122	0.079	15

4.1. NonLinear System Identification

In this work, the nonlinear dynamic system to be identified is that suggested in [44], which is described by:

$$y(k) = \frac{y(k-1)y(k-2)(y(k-1)+2.5)}{1+y^2(k-1)+y^2(k-2)} \tag{59}$$

As indicated in [44], the equilibrium state of the unforced system described by Eq. 59 is $(0,0)$. The training input $u(k)$ is uniformly selected in the range $[-2,2]$ and the testing input data is generated by $u(k) = \sin(2\pi k/25)$. For cross-validation purposes, a data set of 5000 and 200 observations is created for training and testing the proposed OIT2-FELM respectively. A uniformly distributed noise in the range of $[-0.2, 0.2]$ is added to all training observations, while a noise-free signal is added to the testing data. For comparison reasons, in this work an input-output configuration $[y(k-1)\ y(k-2), u(k-1)]$ and $y(k)$ is selected correspondingly. Therefore, the identified model is given by:

$$\hat{y}(k) = \hat{f}(y(k-1)\ y(k-2), u(k-1))$$

To compare the performance of the proposed OIT2-FELM with respect to other existing fuzzy methodologies, an OIT2-FELM using three different output layers is suggested, namely: an output layer with an SC algorithm, with a NT approach and with an EKM algorithm respectively. For the nonlinear system identification, in Table 2, a comparison of the average performance of ten experiments between the proposed OIT2-FELM and other learning fuzzy methodologies such as ANFIS model [58], Sequential-Adaptive-Fuzzy-Inference-System (SAFIS) [59], and the classical version of sequential Online FELM (OS-FELM) [44]. For the implementation of OIT2-FELM, two different types of MFs are implemented, i.e. a) a Cauchy MF and b) a Gaussian MF. To evaluate the associated computational complexity in the OIT2-FELM, three different type-reduction approaches are implemented, namely, an Enhanced Karnik-Mendel algorithm (EKM), SC method, and a Nie-Tan direct deduzzification approach (NT).

Table 3: DETAILS OF REAL-WORLD BENCHMARK PROBLEMS.

Datasets	# Attributes	# Observations	
		Training	Testing
Regression Data Sets			
California Housing	8	10320	10320
Auto-MPG	6	196	196
2D Planes	10	10,000	30,768
Bank	8	4,500	3,692
Kinematics of Robot Arm	8	4,000	4,192
Classification Data Sets			
Page blocks	8	10320	10320
Spam Emails	6	196	196
2D Planes	10	10,000	30,768
Letter Recognition	8	4,500	3,692

From Table 2, it can be observed that the best performance is obtained by those fuzzy approaches with 15 rules. In particular, compared to other methodologies, the OIT2-FELM with an SC type-reduction offers a similar or better trade-off between accuracy, model simplicity and training time.

4.2. Regression and Classification Problems

In this section, eight data sets related to regression and classification problems are considered to evaluate the performance of the proposed OIT2-FELM. For the solution of regression problems an OIT2-FELM with a Multiple-Input-Single-Output structure is applied. For comparison reasons, a Multiple-Input-Multiple-Output (MIMO) structure for the OIT2-FELM is used to the solution of classification problems. Details of all data sets are included in Table 3, in which, for comparison reason, all algorithms use Cauchy MFs. Input and Output attributes for both types of problems are normalised in the range [0,1]. The optimum number of fuzzy rules in the OIT2-FELM is determined by trial and error until the best performance is achieved. To determine the best balance between training time and model accuracy, different experiments using data chunks with different size was implemented.

Table 4 presents the average performance of ten experiments, in which, the results obtained by other fuzzy methods are implemented, i.e. the full version of the ANFIS model [58], Simp1_eTS and the OS-FELM [44]. To compare the performance of the incorporation of the SC algorithm in the structure of an OIT2-FELM, three different versions of the proposed OIT2-FELM is implemented: a) an OIT2-FELM with a SC algorithm (or OIT2-FELM-SC for short), b) an OIT2-FELM having a NT direct-defuzzification method, and c) an OIT2-FELM with an Enhanced Karnik-Mendel algorithm (EKM). Based on our experiments (See Table 4), for both problems, it can be concluded that in general the proposed OIT2-FELM provides a similar or better testing performance with respect to other methodologies, especially when the number of fuzzy rules is small. Based on experimentation, it was determined that for all the data sets presented in Table 4, the smallest size of each training data chunk that produces the highest performance is in the interval $50 - 80$ samples. Generally speaking, the OIT2-FELM provides a high trade-off between model accuracy and model simplicity with a high efficiency in terms of computational complexity for the solution of regression and classification problems using benchmark data sets.

Table 4
AVERAGE MODEL PERFORMANCE FOR THE SOLUTION OF REGRESSION AND CLASSIFICATION PROBLEMS.

	Model	OS-FELM	ANFIS-Full	EIT2-RBFNN	OIT2-FELM-SC	OIT2-FELM-NT	OIT2-FELM-EKM	ELM
Data	Avg. Performance				Regression Problems			
California Housing	Training RMSE	0.1305	0.1123	0.1019	0.1307	0.1312	0.1302	0.1311
	Testing RMSE	0.1322	0.1445	0.1293	0.1313	0.1318	0.1306	0.1335
	Training Time	2.010	3002	1180	2.910	2.120	4.210	0.1310
	Number of Rules	5	256	5	5	5	5	25
Auto MPG	Training RMSE	0.0673	0.0301	0.0401	0.0583	0.0661	0.0679	0.0726
	Testing RMSE	0.0771	0.1012	0.0509	0.0680	0.0775	0.0642	0.0964
	Training Time	0.0510	5.2200	110.10	0.3100	0.1200	3.1130	0.0100
	Number of Rules	3	64	3	3	3	3	10
Census (House8L)	Training RMSE	0.0670	0.0589	0.0492	0.0636	0.0671	0.0601	0.0710
	Testing RMSE	0.0698	0.0911	0.0577	0.0649	0.0688	0.0612	0.0729
	Training Time	2.9010	3220.3	1321.1	3.5460	3.0110	7.1202	0.0641
	Number of Rules	8	256	8	8	8	8	20
Kinematics Robotic Arm	Training RMSE	0.0772	0.0211	0.0546	0.0701	0.0723	0.0669	0.0770
	Testing RMSE	0.0698	0.0901	0.0601	0.0811	0.0688	0.0612	0.0824
	Training Time	6.4010	1009.2	677.10	7.8013	6.8320	8.4772	0.9320
	Number of Rules	50	256	50	50	50	50	200
Bank	Training RMSE	0.0381	0.0199	0.0201	0.0357	0.0379	0.0372	0.0451
	Testing RMSE	0.0443	0.1126	0.0359	0.0372	0.0400	0.0382	0.0503
	Training Time	1.7800	1198.0	988.10	2.5910	1.9220	2.9030	4.2100
	Number of Rules	15	256	15	15	15	15	400
	Model	OS-FELM	Simp1_eTS	EIT2-RBFNN	OIT2-FELM-SC	OIT2-FELM-NT	OIT2-FELM-EKM	ELM
	Performance %				Classification Problems			
Page Blocks	Training RMSE	96.93	95.13	97.80	97.11	97.22	96.70	95.52
	Testing RMSE	95.92	94.55	96.89	96.39	95.97	96.21	95.89
	Testing Time	1.5300	17.780	601.30	3.440	2.401	5.109	0.156
	Number of Rules	10	14	10	10	10	10	45
Spam Emails	Training RMSE	92.64	87.48	94.01	92.70	93.01	92.64	94.77
	Testing RMSE	91.59	87.37	92.79	92.09	91.99	92.11	92.13
	Training Time	5.460	4348	1904	13.09	7.110	16.19	0.188
	Number of Rules	8	17	8	8	8	8	300
Letter Recognition	Training RMSE	96.27	81.71	97.19	96.85	96.24	96.72	96.46
	Testing RMSE	93.63	80.56	94.21	93.92	93.70	93.86	93.77
	Training Time	46.12	29790	9881	222.1	55.19	280.4	12.12
	Number of Rules	70	79	70	70	70	70	900

4.3. OIT2-FELM applied to Path following using an ROV

In this section, the performance of the proposed OIT2-FELM is evaluated in world-real conditions to the completion of a predefined 3D path following using an ROV in a water container of dimensions $2.5m \times 2.5m \times 3.5m$. As described in Fig. 10(a), in this work, each successful experiment is the completion of a predefined 3D path that consists of two circuits at different depths. Each circuit in the water container can be completed by following a clockwise/anticlockwise path navigating along walls 1-2-3-4/4-3-2-1 respectively. The trained TSK IT2 FIS is integrated into an HNS as the main classification mechanism to determine the type of contour that is in front and next to the ROV.

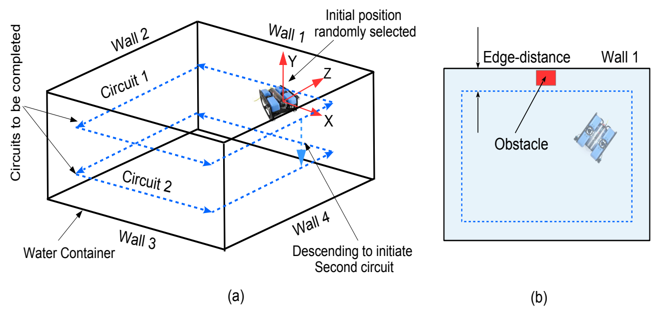


Figure 10: Predefined 3D path to be completed.

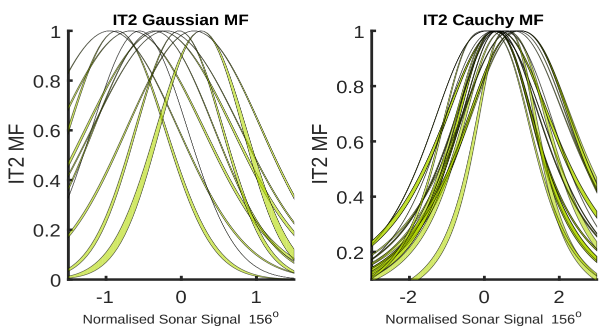


Figure 11: Final distribution of (a) Gaussian MFs and (b) Cauchy MFs.

This information is utilised by the HNS to decompose control motion into a number of four individual fuzzy behaviours. Three behaviours are applied to control the heading and vertical position (depth) of the ROV, as well as its distance to the closest wall or object. The fourth behaviour involves contour classification performed by the OIT2-FELM. At the low-level hierarchy, such behaviours are activated according to stimuli, while at the high-level hierarchy, they are coordinated by a number of IF-THEN rules with singleton activation. To investigate the average performance of the HNS and the proposed OIT2-FELM under real conditions, a number of five successful random experiments to the completion of the proposed 3D path using an ROV is implemented. At each experiment, the initial depth is defined as $d_{p1} = 0.0m$, while the depth d_{p2} for the second circuit is defined with the values $\{1.0m, 1.2m, 1.3m, 1.5m, 1.6m\}$ to each experiment correspondingly. A value of $d_{ref} = 0.55m$ (See Fig. 10(b)) is used for all experiments. To evaluate the performance of the OIT2-FELM in the presence of irregular shapes, an obstacle is located next to a random wall as illustrated in Fig. 10(b). The rest of this section presents results that correspond to the model accuracy achieved by the OIT2-FELM for off-line contour classification (off-line training), and the performance provided by the ROV to complete a 3D path under real world conditions (Online contour classification).

4.3.1. Off-line Contour Classification Results

For Contour classification, in this section two types of results are presented. First, the classification accuracy of the proposed OIT2-FELM is evaluated using a data set of 1273 sonar records of which, 1073 samples were used for training, and 200 samples for testing. The second type of results present the classification performance of the proposed OIT2-FELM in real experiments for the completion of a predefined 3D path. In Table 5, the average model accuracy of ten random experiments performance of the proposed OIT2-FELM is compared to other existing methodologies, in which two types of neural structures for the OIT2-FELM are implemented, i.e.: a) Multiple-Input-Single-Output (MISO) and b) Multiple-Output-Multiple-Output (MIMO). For contour classification using an MISO TSK IT2 FIS, the test sample is then classified by the following criterion:

$$y_p^i = \begin{cases} \text{Corner} & \text{if } y_p^i \ \leq -1 \\ \text{Wall} & \text{if } y_p^i \ > +1 \end{cases} \tag{60}$$

Table 5: PERFORMANCE FOR CONTOUR CLASSIFICATION.

	Model	Training (RMSE)		Testing (RMSE)		
		Mean(%)	Time(s)	Mean(%)	SD	# Rules
	MISO Models					
Cauchy MF	OS-FELM	90.81	2.41	80.06	0.019	16
Cauchy MF	OIT2-FELM-SC	92.27	3.42	83.53	0.009	16
Cauchy MF	OIT2-FELM-NT	89.29	2.80	81.24	0.150	16
Cauchy MF	OIT2-FELM-EKM	93.19	4.05	**83.81**	0.021	16
Gaussian MF	OS-FELM	91.25	2.45	82.28	0.009	16
Gaussian MF	OIT2-FELM-SC	93.92	3.44	84.10	0.010	16
Gaussian MF	OIT2-FELM-NT	89.88	2.60	82.01	0.005	16
Gaussian MF	OIT2-FELM-EKM	94.03	3.92	**85.10**	0.056	16
	ML-ELM	95.96	6.19	89.14	0.003	16
	BELM	90.21	4.94	78.77	0.002	30
	MIMO Models					
Cauchy MF	OS-FELM	89.96	2.53	82.97	0.003	16
Cauchy MF	OIT2-FELM-SC	94.05	3.02	**84.29**	0.019	16
Cauchy MF	OIT2-FELM-NT	89.90	2.62	81.79	0.095	16
Cauchy MF	OIT2-FELM-EKM	94.20	4.12	83.70	0.008	16
Gaussian MF	OS-FELM	95.21	2.59	82.69	0.05	16
Gaussian MF	OIT2-FELM-SC	95.01	3.00	83.00	0.017	16
Gaussian MF	OIT2-FELM-NT	90.19	2.55	81.10	0.023	16
Gaussian MF	OIT2-FELM-EKM	94.23	4.22	83.88	0.044	16
	ML-ELM	98.21	6.33	**87.41**	0.005	16
	BELM	92.02	5.01	80.01	0.090	30

For contour classification using an MIMO TSK IT2 FIS, the test sample is classified based on the criterion:

$$y_p^i = \begin{cases} \text{Corner} & \text{if } y_p^i = [-1, +1]^T \\ \text{Wall} & \text{if } y_p^i = [+1, -1]^T \end{cases} \tag{61}$$

From our experiments, it was found that an OIT2-FELM with sixteen fuzzy rules using a Gaussian MF of IT2 provided the best trade-off between model generalisation and model simplicity for both types of neural structures. For cross-validation, an initial chunk of 500 records was employed to calculate the initial values of each Online Model. Subsequent data chunks of size smaller or equal to 80 records were fed into the OIT2-FELM in order to update its parameters. According to our results, it was determined that chunks of data that are larger than 80 records do not improve significantly the generalisation capabilities of an OIT2-FELM in this experiment. Details of the average Root-Mean-Square-Error (RMSE) of five experiments for contour classification are presented in Table 5.

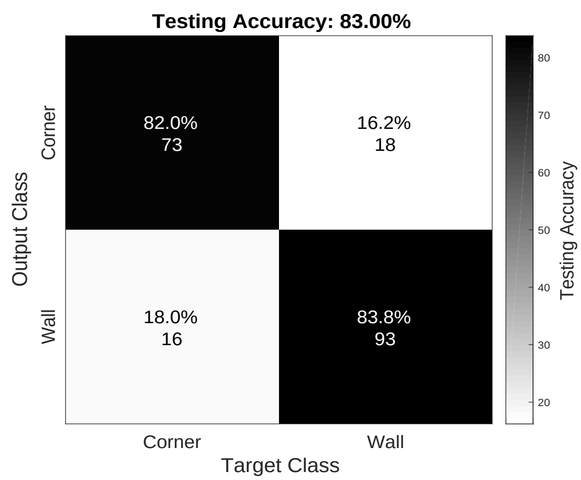


Figure 12: Confusion matrix of simultaneous classification provided by the OIT2-FELM.

As indicated in table 5, three versions of the OIT2-FELM are suggested, namely, the proposed OIT2-FELM with a type-reduction (called OIT2-FELM-SC for short), an OIT2-FELM with an NT algorithm and OIT2-FELM with an EKM algorithm respectively. To compare the efficiency of the proposed network, three neural models based on ELM theory are implemented, i.e. Online-Sequential ELM [60], Multilayer ELM (ML-ELM) [61] and Bayesian ELM [62]. Columns for training and testing were the percentage of correct classification over the validation data for each contour. Unlike networks with three layer, the neural structure of the ML-ELM includes two autoencoders and a classifier with a number of hidden units defined with $[45, 45, 30]$ correspondingly. Evidently, from the table, an average performance to recognise new data is about 80%, of which the highest accuracy is that provided by an ML-ELM. However, the best trade-off between low computational complexity, model simplicity and high generalisation properties is provided by an MISO OIT2-FELM-EKM and an MIMO OIT2-FELM-SC with sixteen fuzzy rules. In Fig. 11, an example of the final discourse of universe for the input Sonar Signal 156^o is illustrated. As described in Fig. 11, even though the MFs on individual variables are used to form input space partitioning, the distinguishability of these MFs produced by using traditional IT2-FELM is not guaranteed. In this study, the proposed OIT2-FELM is a data-driven neural fuzzy approach equivalent to a possibility fuzzy model in which sufficient overlapping is necessary for model accuracy but not for model interpretability [63].

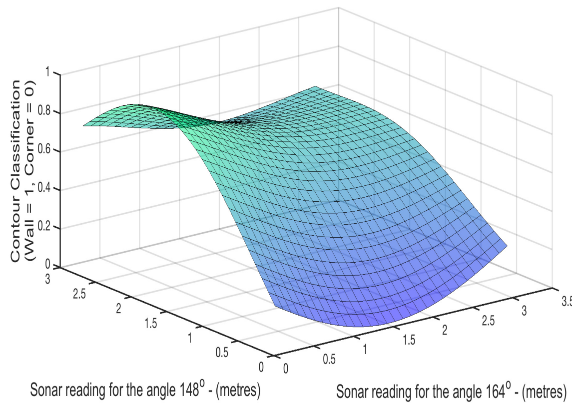


Figure 13: Contour classification regime that corresponds to sonar readings 148^o and 164^o.

In Fig. 12, the confusion matrix that corresponds to the average testing performance of an OIT2-FELM-SC with interval Gaussian MFs is presented. From the Figure, it can be observed that the largest confusion is produced to the recognition of corners.

The confusion between corners and walls (including objects) is consistent with visual data obtained by the optimum operation regime (surface) constructed from the prediction of a OIT2-FELM-SC as described in Fig. 13. From the figure, signal readings that correspond to the angle 148^o and 164^o are used as a representative classification regime, in which small values indicate the presence of corners or objects.

4.3.2. Performance of the ROV to complete 3D path-following

This section reports the average performance of the proposed HNS and the classification accuracy of an OIT2-FELM for the completion of 3D paths under real conditions. To discriminate between walls and corners, an MISO structure for the TSK IT2 FIS was suggested. Since the dynamic of the ROV is not available, the dynamic properties of the closed-loop structure and the parameters of each IT2-FS as well as the corresponding consequent of each IF-THEN fuzzy rule were derived intuitively and experimentally. Sonar data collected in real time was used by the proposed OIT2-FELM with a SC type-reduction to update the parameters of TSK IT2 FIS. To illustrate the performance of each fuzzy behaviour, in Fig. 14, the trajectory followed by the ROV using an OIT2-FELM and two different types of fuzzy controllers is presented. For example, in Fig. 14(a) and (b), the performance of the fuzzy behaviour for yaw angle control based on an IT2 FPD controller and a T1 FPD controller is illustrated respectively. From both figures, it can be observed the time necessary to navigate each wall in the water container is different. This is a natural response due to the different navigation transitions and uncertainties present in the container. In Fig. 14(c), the performance of edge distance behaviour using an IT2 FPD controller versus T1 FPD controller is presented. In Fig. 14(d) and (e), the performance of the fuzzy behaviour for depth control using an IT2 FPD controller and T1 FPD controller is illustrated correspondingly. As depicted in Fig. 14 [64], the implementation of IT2 FPD controllers as low-level fuzzy behaviours improves the overall performance of T1 FPD in the face of noisy signals and inherent uncertainties in the water container. As pointed out in [65], the incorporation of higher order fuzzy controllers not only allows more degree of freedom in the fuzzy sets, but also improves system stability with a better treatment of uncertainty. Recent advances the design of high order fuzzy logic controllers such as General Type-2 Fuzzy controllers (GT2 FLCs) [66] and interval Type-3 fuzzy control controllers (IT3 FLCs) have demonstrated to outperform conventional fuzzy controllers due to their capabilities to better

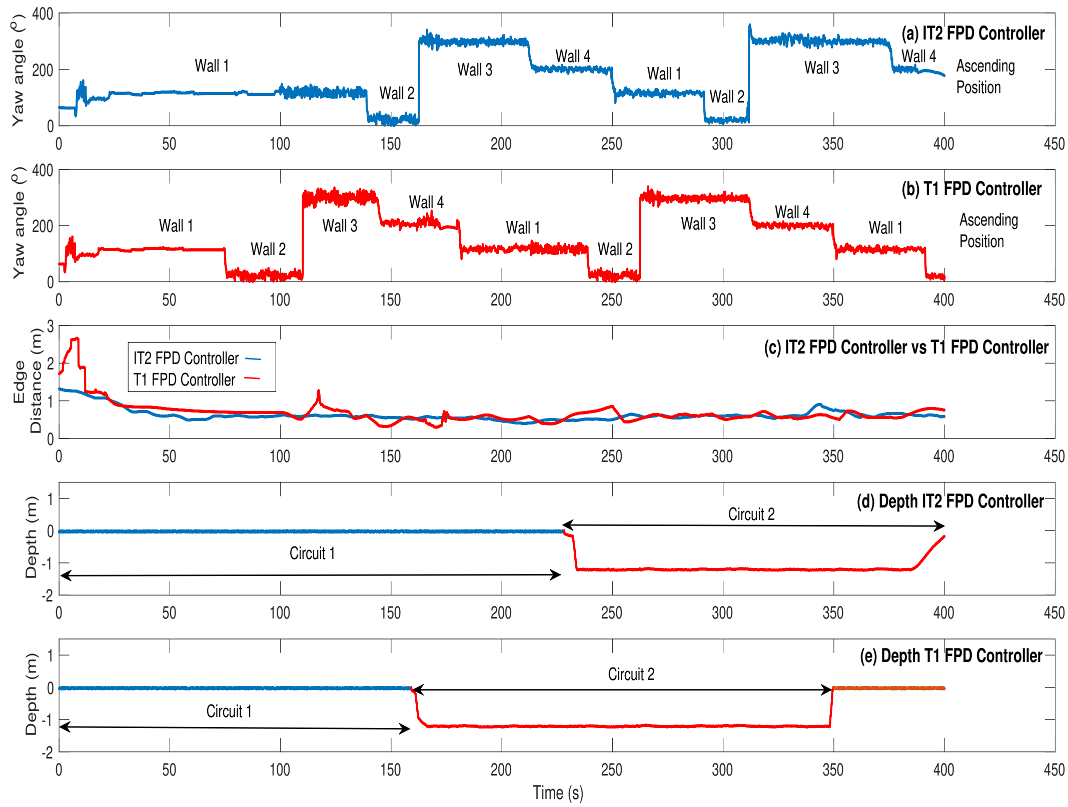


Fig. 14. Comparison performance of two different types of fuzzy controllers for 3D Path following using an ROV: Yaw angle control using an (a) IT2 FPD controller and (b) an T1 FPD controller. (c) Performance of edge distance control with IT2 FPD controller vs T1 FPD controller and Vertical position (depth) using an (c) IT2 FPD controller and (e) a T1 FPD controller.

handle dynamic perturbation and actuator nonlinearities [64]. While in IT3 FLCs, upper and the lower of the FOU are not constant and the secondary MF is of interval type-2, in GT2 and IT2 FLCs are T1 fuzzy sets and crisp numbers respectively.

In Fig. 15, the 3D trajectory followed by the ROV using an HNS whose low-level fuzzy behaviours are based on an (a) OIT2-FELM wtih a SC type-reduction and IT2 FPD control and (b) an OIT2-FELM wtih a SC type-reduction and T1 FPD control is presented. As described in 15, the initial navigation point in the first circuit begins at the position A, and ends in the point B. Once the first circuit is completed, the ROV descends 1.2m reaching the starting point of circuit two (location D). Finally, the successful completion of the 3D path is reached at point E. In this experiment, an obstacle (a traffic cone) was placed next to wall three at a depth of 1.3m in order to investigate the ability of the HNS and the proposed OIT2-FELM-SC to deal with irregular shapes. From Fig. 14(a), it can be observed that decision making in the ROV takes longer at the beginning of the first circuit (Fig. 14(c) and Fig. 15), and when it is navigating along the wall three during the completion of the second circuit.

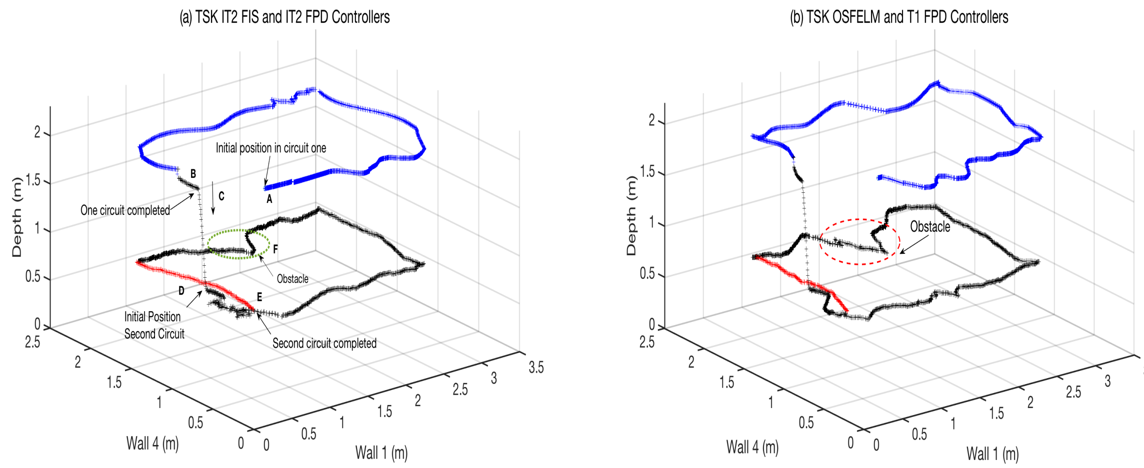


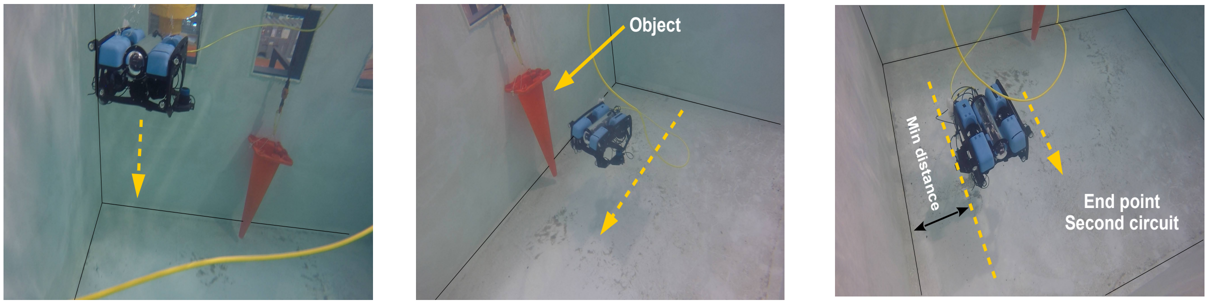
Figure 15: 3D path following task completed by the ROV using a water container of $2.5m \times 2.5m \times 3.5m$ and an obstacle situated in wall three using an (a) HNS based on IT2 FPD control and the proposed OIT2-FELM, and an (b) HNS based on T1 FPD control and OS-FELM (T1 fuzzy counterpart).

In the first circuit, this is mainly due to the number of navigation actions the ROV needs to complete before reaching its position to the closest wall.

First, the reference yaw angle β_{ref} is estimated using signal readings from the sonar, the echosounder and the compass.

This information is then used by the HNS to guide the ROV to maintain a position to the closest wall (or object, See Fig. 14(c)) with a parallel alignment and predefined depth (Fig. 14(d, e)). From this position, based on the information provided by each fuzzy behaviour, local path planning is executed in near-to-real-time. In the second circuit, in the presence of obstacles, a larger number of sonar samples is required to distinguish between walls, obstacles and corners. As shown in Fig. 14(a), the reference yaw angle β_{ref} is increased 90^o each time the ROV reaches a corner. At the beginning of each wall, the HNS estimates the new value for the reference angle β_{ref}. The heading behaviour is activated once the yaw angle is out of the range $[\beta_{ref} - 5^o, \beta_{ref} + 5^o]$. In general, from our experiments it can be concluded that the average time that takes the ROV to complete a 3D path that consists of two circuits is approximately $450sec$ $(7.5minutes)$.

In Fig. 16, a random example of the ROV's behaviour in three different locations during the completion of a predefined 3D path is illustrated. Fig. 16(a) shows the position at which the ROV has completed the first circuit and it is descending from the surface to the initial point of the second circuit. In Fig. 16(b), the ROV is next to an obstacle which creates an irregularity while navigating along the wall three. It is worth mentioning that in this position, the ROV is executing three actions concurrently, i.e. heading control, depth control and contour classification. Finally, in Fig. 16(c), the ROV reaches the final point of the 3D path, and from there it is ascending to the surface.



(a) ROV descending to start the second circuit

(b) recognition of irregular walls

(c) End point of the second circuit

Fig. 16. Path followed by the ROV for the (a) Yaw angle, (b) Edge distance and (c) Vertical position (depth).

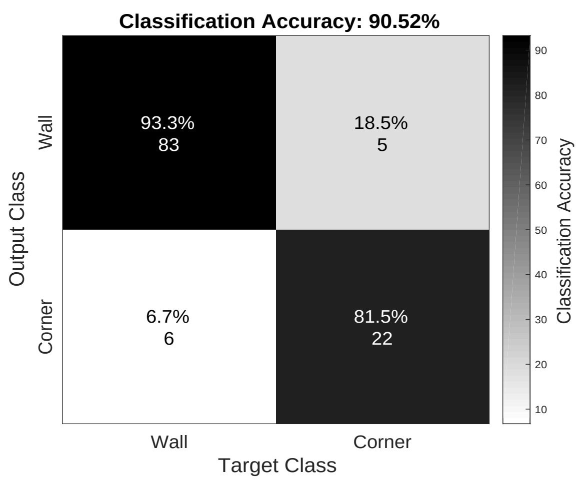


Figure 17: Confusion Matrix of real time contour classification.

From our experiments, it was found that a value $\hat{c} \geq 0.75$ and a max number of four consecutive sonar data signals favours an average classification performance of approximately 90% with an average computing time of $1.6s$. In Fig. 17, the average confusion matrix of five experiments is shown. In the matrix, it can be observed that the largest confusion is for the recognition of corners. We believe this is due to a dead zone that is created during the construction of the sonar data to discriminate corners from walls. Particularly to the correct definition at which values of $180^o, 172^o$ and 164^o it is close enough to consider a wall as a corner. From the confusion matrix, it can also be observed that an average number of 116 sonar samples can be collected at each experiment, of which, a larger number correspond to class walls.

5. Discussion

Path following is a navigation task frequently required in marine missions that involve the inspection of underwater structures such as dams, ship hulls, harbors and oil pipelines. Other applications for path following includes the mapping and monitoring of the marine environment, data collection in oceanic areas of difficult access to vessel-based instruments, e.g. underwater caves, under-ice missions in polar regions and tasks that involve scientific survey and sampling to the full depth of the ocean [67]. A crucial requirement for safe operation and recovery of UVs is their ability to autonomously complete predefined paths in uncertain environments [14, 15].

In this work, we implemented an Online-Interval Type-2 Fuzzy Extreme Learning Machine as the main guidance mechanism to the completion of a 3D path using a Remotely Operated Underwater Vehicle (ROV). The proposed OIT2-FELM is an online sequential learning scheme to the training of a more generalised model of Interval Type-2 Fuzzy Inference Systems (IT2-FISs) in which data may arrive one-by-one or chunk-by-chunk with a fixed or varying size. We use the term "more generalised" to refer to the design of neural structures that follows the general taxonomy of IT2-FISs. An OIT2-FELM is derived from the functional equivalence between SLFNs and IT2-FISs using a TSK inference, in which, capabilities from the theory of neural networks and fuzzy logic may be applied directly to its structure under some mild conditions. To extent the application of the OIT2-FELM onto FISs of Mamdani type, consequent weights become a crisp value. Similar to Fuzzy Logic Systems (FLSs), the OIT2-FELM is an IT2-FIS that contains four main components, i.e., a process for fuzzification, a rule base, an inference engine and an output processor. In the output processor, a type-reduction (TR) method projects IT2 fuzzy sets into an interval of numbers which are finally mapped by a defuzzifier to obtain their centroid (crisp value) [30, 34, 45]. Based on ELM theory, the parameters of each antecedent in the OIT2-FELM are arbitrarily chosen while the consequent weights are determined analytically. The resulting OIT2-FELM model can be viewed as a linear system in which each consequent is obtained through a generalised inverse operation of hidden layer output matrices. A common practice for the design of IT2-FELMs is to perform this process through the implementation of TR methods based on KM algorithms. This practice still represents a fast training process for the design of IT2-FELMs, however the iterative nature of KM algorithms may represent a bottleneck to certain cost-sensitive real world solutions. To reduce the associated computational load of KM algorithms, in this work we investigated three versions of the OIT2-FELM based on the type of TR. An enhanced KM algorithm (EKM), a simplified version of the COSTRWSR called SC algorithm, and the Nie-Tan closed-form method were implemented. In addition, two different types of bounded nonconstant piecewise continuous MFs are also suggested, namely, a Cauchy and Gaussian MF.

To further investigate the performance of the proposed OIT2-FELM with respect to other existing methodologies, an experiment that involves the solution of nonlinear system identification, regression and classification problems was implemented. From this experiment, data sets of different size and complexity were considered. For nonlinear system identification, the OIT2-FELM was evaluated in the presence of noise. Based on our simulations, it was observed an improved generalisation performance from the OIT2-FELM. This was mainly due to extra parameters that involve the design of each FS in the OIT2-FELM, and its ability to treat uncertainty as a deficiency that results from imprecise boundaries in the fuzzy sets. As suggested in [54], each FS in an IT2-FIS can be thought as a design degree of freedom. This accounts for a better treatment of noisy signals that may be translated into uncertainties in the antecedent MFs.

In terms of computational efficiency, the incorporation of IT2 FSs usually represents an increase of the final computational complexity. One solution, is the use of TR algorithms that avoid the iterative nature of KM algorithms. It is worth mentioning, in some cases reducing the size of each training data for online learning may contribute to an increase in the final training time [44]. In general, from the first experiment, the OIT2-FELM is a well-suited methodology for the modelling of complex data sets that provides a high trade-off between model simplicity, accuracy and computational burden. An OIT2-FELM inherits the capabilities of online learning fuzzy systems to naturally describe system behaviour as a series of linguistic rules. Such behaviour formulates an adequate optimisation of MFs in terms of semantic fuzzy criteria while properly dealing with the coverage, completeness and consistency of rules to get insights into the system being modelled.

The second experiment involved the implementation of the OIT2-FELM in a Hierarchical Navigation Strategy (HNS) to provide an ROV with the necessary autonomy to complete a predefined 3D path. The proposed OIT2-FELM proved to be a suitable methodology to extract in near-to-real time conditions useful knowledge about the surroundings of the ROV. In the face of uncertainty and imprecision, the OIT2-FELM demonstrated to deliver a better performance than the OS-FELM. External and internal uncertainties include the noisy signals coming from sensor measurement, in particular sonar data, and the disturbance and imprecision that results from the tether. As described in [30], IT2 fuzzy logic has been credited with being an adequate methodology to the robust design of FISs that are able to model and handle uncertainties [34]. The incorporation of Interval Type-2 fuzzy logic allows the HNS to handle the uncertainties in nonlinear control systems directly from the sensory system [68]. This is mainly due to the adaptiveness of each low-level fuzzy behaviour to define the bounds of each type-reduced interval FS with respect to input changes.

The information provided by the OIT2-FELM was crucial to the HNS to decompose the ROV's behaviour into a bottom-up hierarchy of increased behavioural complexity [5]. At the low level, fuzzy behaviours served a single purpose by operating in a reactive fashion. They performed nonlinear mappings from different sensors to motion actuators. Each behaviour knew nothing about other behaviours, and alone had been insufficient for autonomous navigation. These behaviours can be modulated through synergistic coordination to produce more composite behaviours allowing the ROV to achieve goal-directed operations (high-level hierarchy).

The proposed navigation methodology, together with the OIT2-FELM is also inspired by the decision making process used by some nocturnal mammals to distinguish objects and the shape of confined spaces in dark environments. In some desert rodents, whisking is a cognitive process that allows them to estimate where an object is located, how big it is, and what kind of surface texture it has [40]. This information can help the rodent to distinguish a stone from a seed or a threat. Similar to the whisking process used by some desert rodents, the sonar information is processed by the OIT2-FELM to provide the ROV with a more complete sensory picture of its surroundings [69]. This information also allows the ROV to distinguish a corner from a wall while local planning is achieved.

An HNS treats each fuzzy behaviour as building blocks for more intelligent composite fuzzy behaviours. This is achieved by the concurrent execution of two or more low-level behaviours. The completion of one circuit in the water container implies that goal-directed navigation is decomposed by the HNS as a behavioural function of wall-follow, local path planning and collision avoidance. These behaviours can be further decomposed into low-level fuzzy behaviours, of which, the outcome of contour classification influences the overall ROV's behaviour to a greater or lesser degree depending on the current situation.

The results presented in this work only apply to autonomous navigation in indoor environments. The proposed robotic platform and the OIT2-FELM can be applied in outdoor environments as well. Future research includes the application of this robotic platform and the proposed methodology for the inspection of oil structures in shallow-depth zones of the Gulf of Mexico.

6. Conclusion

In this paper, a new sequential Learning methodology for the training of TSK IT2 FISs and its application to 3D path following using remotely operated underwater vehicles (ROVs) is suggested. The proposed methodology called Online Interval Type-2 Fuzzy Extreme Learning Machine (OIT2-FELM) is a learning scheme applied to a more generalised model of IT2 FISs equivalent to SLFNs, in which data may arrive one-by-one or chunk-by-chunk with a fixed or varying size.

The OIT2-FELM was integrated into a Hierarchical Navigation Strategy (HNS) as the main mechanism for local path planning to the successful completion of a predefined 3D path in underwater remotely operated underwater vehicles (ROVs). To provide the ROV with the necessary autonomy to complete predefined 3D paths, the OIT2-FELM inferred the surroundings of the ROV by classifying the information collected by an on-board scanning sonar. This information was utilised by the HNS to break down the navigation strategy into a number of local fuzzy behaviours that facilitate an accurate following behaviour with near-to-real-time intelligence.

From our experiments, we showed that the OIT2-FELM is able to provide a well suited methodology for robust behaviour in autonomous navigation in the presence of noisy signals, of which, data can be learned one-by-one or chunk-by-chunk (with fixed or varying size). The ROV showed an accurate tracking performance under different types of uncertainties such as dynamic perturbation and intrinsic actuator nonlinearities common in underwater environments.

For a better treatment of uncertainty, future work includes the development and study of hierarchical navigation strategies where low-level behaviours are based Type-3 fuzzy logic control and adaptive Type-2 fuzzy controllers.

Acknowledgment

The authors would like to thank to the national program 'Catedras CONACYT-Mexico, project number 475' for its financial support, and to the Center for Engineering and Industrial Development, CIDESI Mexico, for providing the material and robotic platform necessary to complete the experiments.

References

[1] H. Wang, Y. Tian, H. Xu, Neural adaptive command filtered control for cooperative path following of multiple underactuated autonomous underwater vehicles along one path, IEEE Transactions on Systems, Man, and Cybernetics: Systems (2021).

[2] M. Panda, B. Das, B. Subudhi, B. B. Pati, A comprehensive review of path planning algorithms for autonomous underwater vehicles, International Journal of Automation and Computing (2020) 1–32.

[3] F. Jalal, F. Nasir, Underwater navigation, localization and path planning for autonomous vehicles: A review, in: 2021 International Bhurban Conference on Applied Sciences and Technologies (IBCAST), IEEE, 2021, pp. 817–828.

[4] O. Çatal, T. Verbelen, T. Van de Maele, B. Dhoedt, A. Safron, Robot navigation as hierarchical active inference, Neural Networks 142 (2021) 192–204.

[5] E. Tunstel Jr, T. Lippincott, M. Jamshidi, Behavior hierarchy for autonomous mobile robots: Fuzzy-behavior modulation and evolution, Intelligent Automation & Soft Computing 3 (1) (1997) 37–49.

[6] R. B. Wynn, V. A. Huvenne, T. P. Le Bas, B. J. Murton, D. P. Connelly, B. J. Bett, H. A. Ruhl, K. J. Morris, J. Peakall, D. R. Parsons, et al., Autonomous underwater vehicles (auvs): Their past, present and future contributions to the advancement of marine geoscience, Marine Geology 352 (2014) 451–468.

[7] M. Jacobi, Autonomous inspection of underwater structures, Robotics and Autonomous Systems 67 (2015) 80–86.

[8] D. Li, P. Wang, L. Du, Path planning technologies for autonomous underwater vehicles-a review, IEEE Access 7 (2018) 9745–9768.

[9] E. H. Binugroho, W. Ab, M. I. Mas' udi, B. Setyawan, R. S. Dewanto, D. Pramadihanto, erov: Depth and balance control for rov motion using fuzzy pid method, in: 2019 International Electronics Symposium (IES), IEEE, 2019, pp. 637–643.

[10] X. Xiang, C. Yu, Q. Zhang, Robust fuzzy 3d path following for autonomous underwater vehicle subject to uncertainties, Computers & Operations Research 84 (2017) 165–177.

[11] M. Hosseini, S. Seyedtabaii, Robust rov path following considering disturbance and measurement error using data fusion, Applied Ocean Research 54 (2016) 67–72.

[12] C. Yu, X. Xiang, L. Lapierre, Q. Zhang, Nonlinear guidance and fuzzy control for three-dimensional path following of an underactuated autonomous underwater vehicle, Ocean Engineering 146 (2017) 457–467.

[13] S. Krupiński, G. Allibert, M.-D. Hua, T. Hamel, Pipeline tracking for fully-actuated autonomous underwater vehicle using visual servo control, in: 2012 American control conference (ACC), IEEE, 2012, pp. 6196–6202.

[14] J. J. Leonard, A. Bahr, Autonomous underwater vehicle navigation, in: Springer Handbook of Ocean Engineering, Springer, 2016, pp. 341–358.

[15] A. Kim, R. M. Eustice, Real-time visual slam for autonomous underwater hull inspection using visual saliency, IEEE Transactions on Robotics 29 (3) (2013) 719–733.

[16] P. Angelov, C. Xydeas, D. Filev, On-line identification of mimo evolving takagi-sugeno fuzzy models, in: 2004 IEEE International Conference on Fuzzy Systems (IEEE Cat. No. 04CH37542), Vol. 1, IEEE, 2004, pp. 55–60.

[17] L. Lapierre, B. Jouvencel, Robust nonlinear path-following control of an auv, IEEE Journal of Oceanic Engineering 33 (2) (2008) 89–102.

[18] Y. Ma, M. Hu, X. Yan, Multi-objective path planning for unmanned surface vehicle with currents effects, ISA transactions 75 (2018) 137–156.

[19] H. C. Lamraoui, Z. Qidan, Path following control of fully-actuated autonomous underwater vehicle in presence of fast-varying disturbances, Applied Ocean Research 86 (2019) 40–46.

[20] X. Liang, X. Qu, Y. Hou, J. Zhang, Three-dimensional path following control of underactuated autonomous underwater vehicle based on damping backstepping, International Journal of Advanced Robotic Systems 14 (4) (2017) 1729881417724179.

[21] C. Petres, Y. Pailhas, P. Patron, Y. Petillot, J. Evans, D. Lane, Path planning for autonomous underwater vehicles, IEEE Transactions on Robotics 23 (2) (2007) 331–341.

[22] Z. Peng, J. Wang, Q.-L. Han, Path-following control of autonomous underwater vehicles subject to velocity and input constraints via neurodynamic optimization, IEEE Transactions on Industrial Electronics 66 (11) (2018) 8724–8732.

[23] W. Yang, S. Fan, S. Xu, P. King, B. Kang, E. Kim, Autonomous underwater vehicle navigation using sonar image matching based on convolutional neural network, IFAC-PapersOnLine 52 (21) (2019) 156–162.

[24] X. Liang, X. Qu, L. Wan, Q. Ma, Three-dimensional path following of an underactuated auv based on fuzzy backstepping sliding mode control, International Journal of Fuzzy Systems 20 (2) (2018) 640–649.

[25] T. Zhao, Y. Xiang, S. Dian, R. Guo, S. Li, Hierarchical interval type-2 fuzzy path planning based on genetic optimization, Journal of Intelligent & Fuzzy Systems (Preprint) (2020) 1–12.

[26] H. Mehrjerdi, M. Saad, J. Ghommam, Hierarchical fuzzy cooperative control and path following for a team of mobile robots, IEEE/ASME Transactions on Mechatronics 16 (5) (2010) 907–917.

[27] X. Liang, L. Wan, J. I. Blake, R. A. Shenoi, N. Townsend, Path following of an underactuated auv based on fuzzy backstepping sliding mode control, International Journal of Advanced Robotic Systems 13 (3) (2016) 122.

[28] A. R. Solis, G. Panoutsos, Granular computing neural-fuzzy modelling: A neutrosophic approach, Applied Soft Computing 13 (9) (2013) 4010–4021.

[29] X. Xiang, C. Yu, L. Lapierre, J. Zhang, Q. Zhang, Survey on fuzzy-logic-based guidance and control of marine surface vehicles and underwater vehicles, International Journal of Fuzzy Systems 20 (2) (2018) 572–586.

[30] C. Wagner, H. Hagras, Toward general type-2 fuzzy logic systems based on zslices, IEEE Transactions on Fuzzy Systems 18 (4) (2010) 637–660.

[31] F. M. Raimondi, M. Melluso, Hierarchical fuzzy/lyapunov control for horizontal plane trajectory tracking of underactuated auv, in: 2010 IEEE International Symposium on Industrial Electronics, IEEE, 2010, pp. 1875–1882.

[32] S. Panasuraman, V. M. Ganapathy, B. Shirinzadeh, Behavior based neuro-fuzzy controller for mobile robot navigation, in: International Conference on Mechatronics Technology 2002, Tokyo Institute of Technology, 2002, pp. 364–371.

[33] S. Nurmaini, B. Tutuko, Intelligent robotics navigation system: Problems, methods, and algorithm., International Journal of Electrical & Computer Engineering (2088-8708) 7 (6) (2017).

[34] H. A. Hagras, A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots, IEEE Transactions on Fuzzy systems 12 (4) (2004) 524–539.

[35] A. Castro-Lopez, J. Puente, R. Vazquez-Casielles, Fuzzy inference suitability to determine the utilitarian quality of b2c websites, Applied Soft Computing 57 (2017) 132–143.

[36] A. Rubio-Solis, U. Martinez-Hernandez, G. Panoutsos, Evolutionary extreme learning machine for the interval type-2 radial basis function neural network: A fuzzy modelling approach, in: 2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, 2018, pp. 1–8.

[37] A. Rubio-Solis, G. Panoutsos, C. Beltran-Perez, U. Martinez-Hernandez, A multilayer interval type-2 fuzzy extreme learning machine for the recognition of walking activities and gait events using wearable sensors, Neurocomputing (2020).

[38] A. Rubio-Solis, T. Salgado-Jimenez, L. G. Garcia-Valdovinos, L. Nava-Balanzar, R. A. Hernandez-Hernandez, U. Martinez-Hernandez, An evolutionary general type-2 fuzzy neural network applied to trajectory planning in remotely operated underwater vehicles, in: 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, 2020, pp. 1–8.

[39] M. A. Khanesar, A. J. Khakshour, O. Kaynak, H. Gao, Improving the speed of center of sets type reduction in interval type-2 fuzzy systems by eliminating the need for sorting, IEEE Transactions on Fuzzy Systems 25 (5) (2016) 1193–1206.

[40] S. A. Hires, L. Pammer, K. Svoboda, D. Golomb, Tapered whiskers are required for active tactile sensation, Elife 2 (2013) e01350.

[41] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1-3) (2006) 489–501.

[42] A. Rubio-Solis, G. Panoutsos, Iterative information granulation for novelty detection in complex datasets, in: 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, 2016, pp. 953–960.

[43] G.-B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (2) (2011) 513–529.

[44] H.-J. Rong, G.-B. Huang, N. Sundararajan, P. Saratchandran, Online sequential fuzzy extreme learning machine for function approximation and classification problems, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 (4) (2009) 1067–1072.

[45] A. Rubio-Solis, G. Panoutsos, Interval type-2 radial basis function neural network: a modeling framework, IEEE Transactions on Fuzzy Systems 23 (2) (2014) 457–473.

[46] Z. Deng, K.-S. Choi, L. Cao, S. Wang, T2fela: Type-2 fuzzy extreme learning algorithm for fast training of interval type-2 tsk fuzzy logic system, IEEE transactions on neural networks and learning systems 25 (4) (2013) 664–676.

[47] U. Martinez-Hernandez, A. Rubio-Solis, G. Panoutsos, A. A. Dehghani-Sanij, A combined adaptive neuro-fuzzy and bayesian strategy for recognition and prediction of gait events using wearable sensors, in: 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, 2017, pp. 1–6.

[48] D. Wu, Approaches for reducing the computational cost of interval type-2 fuzzy logic systems: overview and comparisons, IEEE Transactions on Fuzzy Systems 21 (1) (2012) 80–99.

[49] J. M. Mendel, Computing derivatives in interval type-2 fuzzy logic systems, IEEE Transactions on Fuzzy Systems 12 (1) (2004) 84–98.

[50] M. Luo, F. Sun, H. Liu, Joint block structure sparse representation for multi-input–multi-output (mimo) t–s fuzzy system identification, IEEE Transactions on Fuzzy Systems 22 (6) (2013) 1387–1400.

[51] C. Chen, D. Wu, J. M. Garibaldi, R. I. John, J. Twycross, J. M. Mendel, A comprehensive study of the efficiency of type-reduction algorithms, IEEE Transactions on Fuzzy Systems (2020).

[52] G.-B. Huang, N.-Y. Liang, H.-J. Rong, P. Saratchandran, N. Sundararajan, On-line sequential extreme learning machine., Computational Intelligence 2005 (2005) 232–237.

[53] J. Zhao, Z. Wang, D. S. Park, Online sequential extreme learning machine with forgetting mechanism, Neurocomputing 87 (2012) 79–89.

[54] J. M. Mendel, General type-2 fuzzy logic systems made simple: a tutorial, IEEE Transactions on Fuzzy Systems 22 (5) (2013) 1162–1182.

[55] G. Lakhekar, L. M. Waghmare, Robust maneuvering of autonomous underwater vehicle: an adaptive fuzzy pi sliding mode control, Intelligent Service Robotics 10 (3) (2017) 195–212.

[56] R. S. Yadav, Application of soft computing techniques to calculation of medicine dose during the treatment of patient: a fuzzy logic approach, in: Handbook of Computational Intelligence in Biomedical Engineering and Healthcare, Elsevier, 2021, pp. 151–178.

[57] A. J. Humaidi, H. T. Najem, A. Q. Al-Dujaili, D. A. Pereira, I. K. Ibraheem, A. T. Azar, Social spider optimization algorithm for tuning parameters in pd-like interval type-2 fuzzy logic controller applied to a parallel robot, Measurement and Control 54 (3-4) (2021) 303–323.

[58] A. Pandey, S. Kumar, K. K. Pandey, D. R. Parhi, Mobile robot navigation in unknown static environments using anfis controller, Perspectives in Science 8 (2016) 421–423.

[59] J. de Jesús Rubio, A. Bouchachia, Msafis: an evolving fuzzy inference system, Soft Computing 21 (9) (2017) 2357–2366.

[60] A. Rubio-Solis, A. Musah, W. P. Dos Santos, T. Massoni, G. Birjovanu, P. Kostkova, Zika virus: Prediction of aedes mosquito larvae occurrence in recife (brazil) using online extreme learning machine and neural networks, in: Proceedings of the 9th International Conference on Digital Public Health, 2019, pp. 101–110.

[61] J. Tang, C. Deng, G.-B. Huang, Extreme learning machine for multilayer perceptron, IEEE transactions on neural networks and learning systems 27 (4) (2015) 809–821.

[62] Z. Jin, G. Zhou, D. Gao, Y. Zhang, Eeg classification using sparse bayesian extreme learning machine for brain–computer interface, Neural Computing and Applications 32 (11) (2020) 6601–6609.

[63] S.-M. Zhou, J. Q. Gan, Low-level interpretability and high-level interpretability: a unified view of data-driven interpretable fuzzy system modelling, Fuzzy sets and systems 159 (23) (2008) 3091–3131.

[64] S. N. Qasem, A. Ahmadian, A. Mohammadzadeh, S. Rathinasamy, B. Pahlevanzadeh, A type-3 logic fuzzy system: Optimized by a correntropy based kalman filter with adaptive fuzzy kernel size, Information Sciences 572 (2021) 424–443.

[65] S. Dian, J. Han, R. Guo, S. Li, T. Zhao, Y. Hu, Q. Wu, Double closed-loop general type-2 fuzzy sliding model control for trajectory tracking of wheeled mobile robots, International Journal of Fuzzy Systems 21 (7) (2019) 2032–2042.

[66] T. Kumbasar, H. Hagras, A self-tuning zslices-based general type-2 fuzzy pi controller, IEEE Transactions on Fuzzy Systems 23 (4) (2014) 991–1013.

[67] A. D. Bowen, D. R. Yoerger, C. Taylor, R. McCabe, J. Howland, D. Gomez-Ibanez, J. C. Kinsey, M. Heintz, G. McDonald, D. B. Peters, et al., The nereus hybrid underwater robotic vehicle for global ocean science operations to 11,000 m depth, in: OCEANS 2008, IEEE, 2008, pp. 1–10.

[68] R.-E. Precup, R.-C. David, R.-C. Roman, A.-I. Szedlak-Stinean, E. M. Petriu, Optimal tuning of interval type-2 fuzzy controllers for nonlinear servo systems using slime mould algorithm, International Journal of Systems Science (2021) 1–16.

[69] N. F. Lepora, M. Evans, C. W. Fox, M. E. Diamond, K. Gurney, T. J. Prescott, Naive bayes texture classification applied to whisker data from a moving robot, in: The 2010 International Joint Conference on Neural Networks (IJCNN), IEEE, 2010, pp. 1–8.