International Journal of Automation and Computing 4 (2006) 336-347

Biologically Inspired Behaviour Design for Autonomous Robotic Fish

Jin-Dong Liu*, Huosheng Hu
Department of Computer Science, University of Essex, Colchester CO4 3SQ, UK

Abstract: Behaviour-based approach plays a key role for mobile robots to operate safely in unknown or dynamically changing environments. We have developed a hybrid control architecture for our autonomous robotic fish that consists of three layers: cognitive, behaviour and swim pattern. In this paper, we describe some main design issues of the *behaviour layer*, which is the centre of the layered control architecture of our robotic fish. Fuzzy logic control (FLC) is adopted here to design individual behaviours. Simulation and real experiments are presented to show the feasibility and the performance of the designed *behaviour layer*.

Keywords: Behaviour-based design, robotic fish, fuzzy logic.

1 Introduction

The biologically inspired or behaviour based approach has been widely adopted in the design of robotic systems, including design of mechanical structure, robot locomotion, navigation algorithm and multi-robot cooperation. A number of biologically inspired robot systems have been built over the last twenty years, such as the robot phonotaxis by Horchler[1] and biomimetic "robo-lobsters" by Grasso[2]. It has advantages of easy design, quick response, model-free, and robustness. It enables a mobile robot to operate in dynamic or unpredictable environments and can quickly response to the changes in the real world. An early successful example is a cleaning robot which can find and retrieve soda cans in an office environment[3].

Up to now, majority of robotic fish research work has been focused on the fish-like propulsion mechanism[4,5], the fin material[6], remote operation[7], multi-agent cooperation[8] and the mechanical structures[9]. There is no researcher who has developed behaviour-based architecture for autonomous robotic fish until now. Based on behaviour-based approach, our robotic fish is designed to autonomously swim and navigate in unknown and dynamical environments. Behaviour-based control is a key to deal with the dynamic factor in the real world.

The individual behaviour can be normally described as $\beta : s \rightarrow a$ which means that given stimulus s the behaviour yields response a according to mapping β. There are several kinds of mathematical methods to describe β, which can be classified into two groups: *Discrete Encoding* and *Continuous Encoding*. In the discrete encoding, β is a collection of IF-THEN rules. The general form of these rules is[10]: IF *antecedent* THEN *consequence*. The antecedent is a list of preconditions that are related with stimulus s and the consequence is one of possible responses. The antecedent and the consequence could be classical logic like *left bumper is ON*, and they also could be vaguely defined logic associated with a degree of truth, e.g. *the left obstacle is far in 0.6 degree of truth*. The latter formation is called *fuzzy logic*. There are many examples on both the classical logic[11,12] and the fuzzy logic[13,14]. The advantage of discrete encoding is that the relationship between antecedents and consequences is explicit and near to human natural language. So it is easy to combine human prior knowledge with the robot control. However, the disadvantage is that the limited number of antecedents is unable to represent all situations that the robot encounters, i.e. the generalization problem.

In contrast, the continuous encoding allows a robot to have an infinite space of potential stimulus and continuous response. *Potential Field Method* is one of the most common methods for implementing the mapping β. It was proposed for generating smooth trajectories for mobile robots by Khatib[15]. This method treats goals as attractors and obstacles as repulsers. The attractor generates an attraction potential force on the robot and the repulser generates a repulsion force. The magnitude of the force is inverse-proportional to the distance between the robot and the goal/obstacle and the force direction points from the obstacle to the robot

Manuscript received July 10, 2006; revised August 28, 2006.
This project is supported by London Aquarium.

*Corresponding author. *E-mail address*: jliua@essex.ac.uk

or from the robot to the goal. The vector summation of all potential forces from goals and obstacles consists of the moving direction and the moving speed for the mobile robot. Several researchers adopted the potential field method. For example, Borenstein[16] proposed a *Virtual Force Field* for obstacle avoidance, which lies in the integration of *Certainty Grids* for obstacle representation and *Potential Fields* for navigation. Slack[17] defined *Navigational Templates(NATs)* to describe continuous pathways in navigation space and Zapata[18] proposed *Deformation Zones* for collision avoidance for fast mobile robot (>5 m/s).

For the robotic fish situation, fuzzy logic, or discrete encoding, is better for the individual behaviour design because it has good reputation on dealing with uncertainty in sensory information and motor execution. Moreover, fuzzy logic provides a convenient way to incorporate human knowledge on the controller design. We will discuss it in details in Section 2.

The rest of this paper is organized as follows. Section 2 presents the structure of the hybrid control architecture and its content. *Behaviour layer*, one of layers in the architecture, is discussed in details according to its components and functions. Section 3 addresses the design procedure of each individual behaviour. Simulation experiments show their performance in the simulator. In Section 4, experimental results of real robotic fish show the feasibility and performance of the *behaviour layer*. Finally, conclusions and future work are given in Section 5.

2 Control architecture and behaviour layer

Our robotic fish is controlled under a hybrid control architecture[19] which is presented in Fig. 1. It consists of three layers: *Cognitive Layer, Behaviour Layer* and *Swim Pattern Layer*. The centre of the architecture is the behaviour layer where most instant decisions are made. The swim pattern layer is at the bottom of our three-layer architecture, which consists of all swim patterns for basic control purpose. Five swim patterns are designed here, namely *Cruise-Straight, Ascend/Descend, Coast, Sharp-Left/Right-Turn* and *Cruise-in-Left/Right-Turn*. Each *swim pattern* is a series of joint motion functions which control the tail joints to realize an associative movement in order to propel or maneuver the robotic fish. In the behaviour layer, states and actions are directly related to the robot physical sensors and swim patterns. In the cognitive layer, states related with high-level activities are abstracted from the behaviour layer and actions are correspond to behaviour coordination which directs the emergent result of several behaviours.

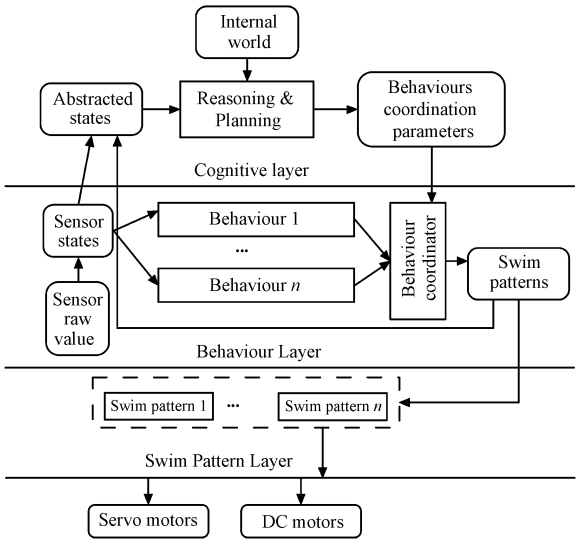


Fig. 1 The hybrid control architecture for our robotic fish

The configuration of the *behaviour layer* is presented in Fig. 2. It consists of three modules: 1) perception, 2) individual behaviours and 3) behaviour coordination. The perception module has two tasks: 1) to remove noise from raw sensor data and 2) to convert the filtered sensor data to physical meaningful values according to the control parameters set by the *cognitive layer*. These values are input signals of individual behaviours. For example, the perception module calculates the depth difference between the robotic fish depth and the desired depth for the *keep-level* behaviour controller.

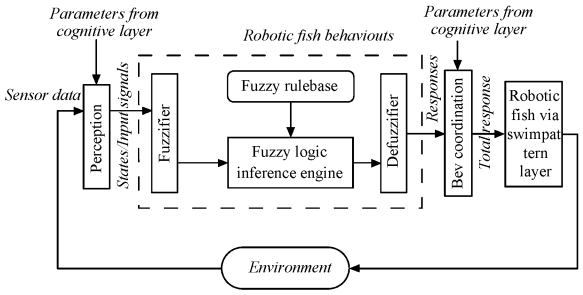


Fig. 2 Configuration of the behaviour layer

In terms of individual behaviours, all their prototypes come from the inspiration of real fish behaviours, particularly the carangiform fish, like carp, behaviour. On their design, continuous encoding methods do not fit here because of numerous special situations of the robotic fish in contrast to common mobile robots. For example, our robotic fish lacks either long-range (>1 m) sensors or computer vision to measure its relative position to goals/obstacles. To detect goal/obstacle, it is only equipped with four short-range

(<0.4 m) infrared sensors (IRs) due to the limited space in the fish head. They can roughly detect the obstacle in front/left/right/bottom of a robotic fish. The IR sensor information also contains uncertainty caused by the fish head swag during swimming. The unsteady water flow also affects infrared light emitted by IR sensors. So, discrete encoding methods, the fuzzy logic especially, is applied on the individual behaviour design. In the fuzzy logic, linguistic, not numerical, variables are used, which mimic human thinking. The prior knowledge is incorporated into the system in this way. The parameters to define the fuzzy logic are easy to be learnt online because there is no complex analytical equation to be solved. Also, many successful fuzzy logic applications[20,21] have shown its robustness.

The fuzzy logic controllers (FLCs) for individual behaviours are designed based on the Mamdani's principle which was proposed in 1975[22]. Each behaviour receives input signals from the perception module, processes them by a fuzzy controller and generates a response. A typical fuzzy controller is constructed by four parts: a *Fuzzifier*, a *Fuzzy Rulebase*, a *Fuzzy Logic Inference Engine* and a *Defuzzifier*. The *Fuzzifier* translates the input signals into linguistic values, i.e. fuzzy input variables, by membership functions. The *Fuzzy Logic Inference Engine* evaluates the set of fuzzy rules defined in the *Fuzzy Rulebase*, identifies the rules that apply to the current input variables and calculates the output linguistic values, i.e. the fuzzy output variables. The *Defuzzifier* translates the output variable to a crisp control signal, i.e. behaviour response.

During the design of a fuzzy controller, its parameters and membership function shape are selected according to the following two factors:

1) *Computation Time*: To realize the real-time control and quick response ability, it is necessary to reduce the complexity of algorithm. The factors affecting the level of complexity includes the number of the fuzzy input and output variables, the form of membership functions, the rule number in a rulebase, *etc.* So, using simply form membership functions and small fuzzy rule sets could save the computation time.

2) *Online Learning Issue*: One of the hardest problems in the online learning is related to the dimension of parameters to be learnt. The exponential growth of hypervolume as a function of dimensionality drags the learning speed and is a well-known problem of "Curse of dimensionality"[23]. So, the parameter number in an FLC should be limited as less as possible because these parameters will be learned or adjusted online.

After the construction of individual behaviours, the behavior coordination module merges the responses of all individual behaviours into an emergent response. Then it outputs the response to *swim pattern layer* which drives servo motors in a robotic fish. In this paper, *vector summation* coordination method is chosen, where each individual behaviour has assigned a gain value and the productions of the gains with behaviour outputs for all individual behaviours are summed up as the emergent response[10].

3 Design of robotic fish behaviours

The behaviours for the robotic fish are designed based on the real fish behaviours rather than man-made machines such as the submarine. Each behaviour we designed is inspired by real fish. The following behaviours are implemented:

1) *Keep-level*: It is most efficient for long distance swimming, i.e. from point a to point b at the same level.

2) *Avoid-obstacle*: This behaviour is to get away from other fishes, predators, rocks, *etc.* A significant feature of this behaviour is that a fish changes its swimming direction (pitch, roll and yaw angles) to avoid obstacles rather than slows down or stops. This is because changing direction is much easier than braking.

3) *Follow-wall*: Fish swims along with the surface of rocks to patrol its territory and find food. On the other hand, this behaviour helps a fish to explore an unknown environment in a safe way because it decreases the risk of predator attacking.

4) *Wandering*: Fish uses it for relaxation. The robotic fish applies this behaviour to swim around to find potential interesting things.

5) *Seek-goal*: Food, mate and home are the main goals for a fish. For the robotic fish, the charging station is one of goals and the tank wall is also a goal when it implements the tank exploration task. This behaviour enables a robotic fish to find a way to the charging station.

6) *Rescue*: In some dangerous situations, if a fish can not get itself out of trouble by other behaviours, *rescue* behaviour will be applied. Likewise, the robotic fish implements this behaviour if it faces uncontrollable situations such as servo motors malfunction, the extreme high internal temperature, *etc.* It will stop doing all other behaviours and only empty the water chamber to float up slowly.

7) *Feed*: The behaviour allows a fish to find food. For example, a shark chases a small fish and swallows it. For the robotic fish, *feed* behaviour enables it to adjust its gesture toward a charging station to charge its battery.

It should be noticed that we have only presented the design of three behaviours in this paper because of the limited space. A comprehensive introduction will

be presented in a forthcoming paper.

3.1 Design of follow-wall behaviour

This behaviour is to keep the robotic fish at a certain distance from the wall or obstacle edges based on the information gathered by IR sensors.

Our robotic fish has four IR sensors (GP2D12) to detect the objects at the directions of left, front, right and bottom. The arrangement of these IR sensors is shown in Fig. 3. Three IR sensors are arranged to be 45 degrees between each other to sense the reflected infrared light from front-left, front and front-right objects. One IR sensor points to the bottom direction. Their detection range is from 0.02 m to 0.4 m. Note that the IR sensor range has been normalized into [0,1]. If the distance of an object from the IR is more than 0.4 m or less than 0.02 m, its output is 0, otherwise its output is inversely proportional to the distant, i.e. from 1 to 0 according to from 0.02 m to 0.4 m. The normalized IR sensor output is called "IR Intensity", which is denoted as $\{d_l, d_f, d_r, d_b\}$ for left, front, right and bottom IR, respectively.

It should be noticed that IR sensors can not detect the obstacle in their dead-zone and the glasses wall. To deal with this problem, accelerometer sensor is applied to detect the collision between the robotic fish and obstacle, and the IR intensity $\{d_l, d_f, d_r\}$ is rectified to be a new value according to the collision direction and strength $|b|$. The IR intensity d_b is rectified by the depth z_f of the robotic fish when it is very close to the tank bottom. See (1) for the rectification procedure.

$$\begin{cases} d_l = 1, d_f = 1 & \text{if } |\,\mathrm{b}\,| > \Delta_{|\,\mathrm{b}\,|} \text{ and collision is left side} \\ d_r = 1, d_f = 1 & \text{if } |\,\mathrm{b}\,| > \Delta_{|\,\mathrm{b}\,|} \text{ and collision is right side} \\ d_b = 1 & \text{if } z_f < \Delta_{z_f} \end{cases} \tag{1}$$

where $\Delta_{|\,\mathrm{b}\,|}$ and Δ_{z_f} are collision thresholds. z_f is calculated from the pressure sensor data.

Fig. 4 shows a block diagram of *follow-wall* behaviour. The left, right and bottom IR sensor data are the input of the perception module. The front IR is ignored here because it provides less information on side walls. One more input W_d is added to define the relative position of the wall to the robotic fish. It is appointed by *cognitive layer* and defined in (2). For instance, if $W_d = 1$, the robotic fish should follow the wall on its left side. The perception module outputs an IR intensity d_s which is either d_l or d_r depending on W_d by (3). Through the fuzzy controller, the output ω'_d is tweaked by W_d to calculate $\omega_d = W_d\omega'_d$ which is one of inputs of the Bev coordination module. Swim pattern S_p is generated and finally executed on the robotic fish. Another output of the perception module is d_b which is the intensity of the bottom IR which affects another fuzzy controller output—P_t.

$$W_d = \begin{cases} 1 & \text{leftside} \\ -1 & \text{rightside} \end{cases} \tag{2}$$

$$d_s = \begin{cases} d_l, & \text{if } W_d = 1 \\ d_r, & \text{if } W_d = -1 \end{cases} \tag{3}$$

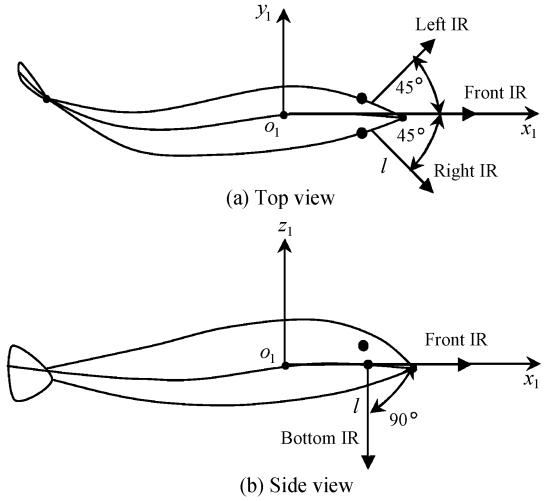


Fig. 3 The IR sensors arrangement on the robotic fish

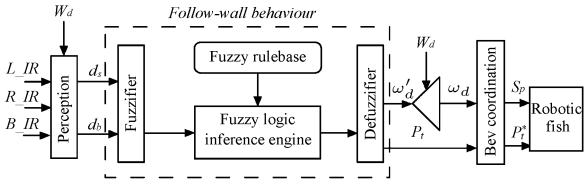


Fig. 4 The fuzzy logic representation of follow-wall behaviour

The form of the membership functions could be Gaussian, trapezoid, sharp peak, triangular, S-type, Z-type, *etc.* Taking into consideration of calculation costs and the learning procedure in future, Triangular, S-type and Z-type functions are chosen for the robotic fish due to their simpleness and many successful applications[20]. The mathematic expressions of the membership function in the fuzzifier are given as follows, for a triangular function:

$$\mu_{ij}(u_i) = \begin{cases} 1 - \dfrac{2\,|u_i - m_{ij}|}{\sigma_{ij}}, & m_{ij} - \dfrac{\sigma_{ij}}{2} < u_i < m_{ij} + \dfrac{\sigma_{ij}}{2} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

for a S-type membership function:

$$\mu_{ij}(u_i) = \begin{cases} 0, & u_i < m_{ij} - \dfrac{\sigma_{ij}}{2} \\ 1, & u_i > m_{ij} \\ 1 - \dfrac{2\,|u_i - m_{ij}|}{\sigma_{ij}}, & \text{otherwise} \end{cases} \quad (5)$$

for a Z-type membership function:

$$\mu_{ij}(u_i) = \begin{cases} 1 & u_i < m_{ij} \\ 0, & u_i > m_{ij} + \dfrac{\sigma_{ij}}{2} \\ 1 - \dfrac{2\,|u_i - m_{ij}|}{\sigma_{ij}} & \text{otherwise} \end{cases} \quad (6)$$

where $i = (1, 2)$ is the number of input signals and $j = (1, 2, 3)$ is the number of terms of the fuzzy input variable. $\mu_{ij}(u_i)$ is the membership function to calculate the degree of truth for the statement that the ith input is the jth fuzzy term. For example, μ_{13} is associated with the statement that d_s is *near*. u_i is the ith crisp input signal to the fuzzy controller, $\{u_1,\ u_2\} = \{d_s, d_b\}$. m_{ij} is the centre of the membership function according to μ_{ij}. σ_{ij} is the width of the membership function. Fig. 5 shows the membership function of *follow-wall*.

The rulebase for fuzzy inference engine is listed in Table 1 which shows that d_s and d_b independently affect ω'_d and P_t. The COG method is selected for defuzzification (7).

$$\omega'_d = \frac{\sum_{k=1}^{6} v_k^1 q_k}{\sum_{k=1}^{6} q_k},\ P_t = \frac{\sum_{k=1}^{6} v_k^2 q_k}{\sum_{k=1}^{6} q_k} \quad (7)$$

$$q_k = \min\{\mu_{1k_1}, \mu_{2k_2}\} \quad (8)$$

where v_k^1 and v_k^2 denote the estimated value of the kth rule output, which is related to the centre of membership functions of the fuzzy output variables ω_d and P_t. q_k is conjunction degree of the IF part of the kth rule, which is calculated by $\min(\cdot)$ operation because AND operator is applied. μ_{ik_i} is the degree of the membership for the ith input contributing to the kth rule, k_i is the number of terms of an input variable corresponding to the ith input in the kth rule. μ_{ik_i} is 1 if the corresponding input in Table 1 is "N/A". For example, for the 2nd rule, $k_1 = 2$, $\mu_{2k_2} = 1$.

To prove the feasibility of *follow-wall* behaviour, it is tested in a simulator proposed in [24]. All parameters of *follow-wall* behaviour are given as same as in this section. The experiments are done without the *cognitive layer*. Assume $W_d = -1$ for *follow-wall* behaviour, i.e. the robotic fish is appointed to follow the right side wall. The desired distance is 0.3 m to side walls and to the bottom.

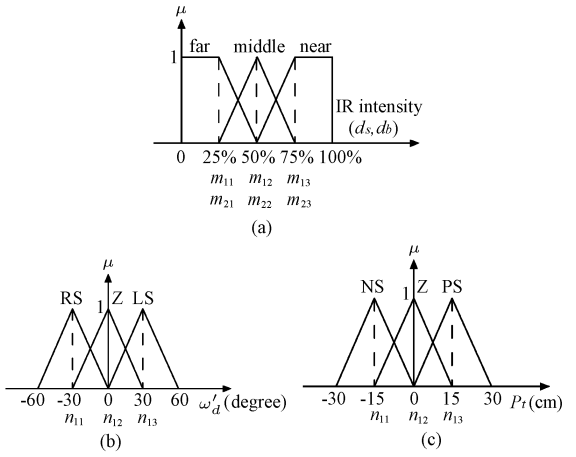


Fig. 5 Membership functions of the input and output variables for *follow-wall*. m_{ij}, n_{ls}: the centres of membership functions. μ: degree of truth. (a) The IR intensity d_s; (b) The yaw angle ω'_d relative to the current swimming direction (c) The position of pitch weights P_t

Table 1 Rulebase for *follow-wall* fuzzy controller

Rule No.	Input		Output	
	d_s	d_b	ω'_d	P_t
1	N	N/A	RS	N/A
2	M	N/A	Z	N/A
3	F	N/A	LS	N/A
4	N/A	N	N/A	PS
5	N/A	M	N/A	Z
6	N/A	F	N/A	NS

N: near, M: middle, F: far, RS: right small turn, LS: left small turn, Z: zero, PS: positive small(ascent), NS: negative small(descent), N/A: none such input or output for the rule

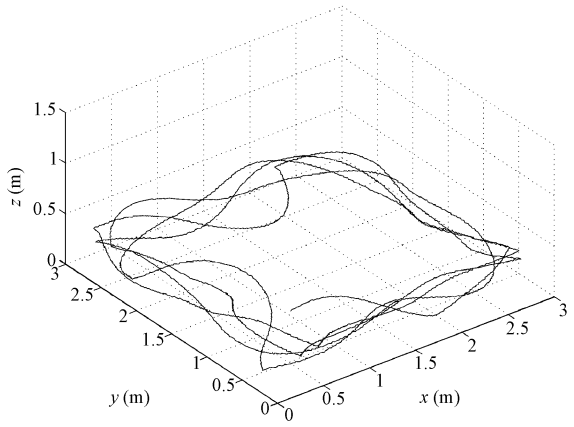
Fig. 6(a) shows the 3D trajectory of a simulated fish with follow-wall behaviour. Figs. 6(b) and (c) show the trajectory projection on the xy plane and the xz plane together with the desired trajectory and depth. Fig. 7 shows the mean square error between the experimental trajectory and the desired one. These illustrate that when the wall is on the right side, the robotic fish can follow it and keep a constant distance to the wall without bumping. However, the sharp changing of trajectory at the corner in Fig. 6(b) shows that the robotic fish can not avoid corners as it can not recognise the corner only with *follow-wall* behaviour.

3.2 Design of keep-level behaviour

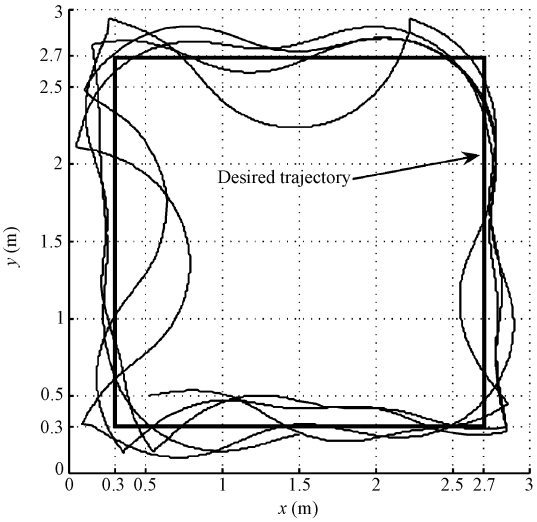
The aim of *keep-level* behaviour is to keep the robotic fish at a desired depth in water. Assume that z_f is the measured depth of the robotic fish and z_a is

the desired swimming depth. The robotic fish should satisfy

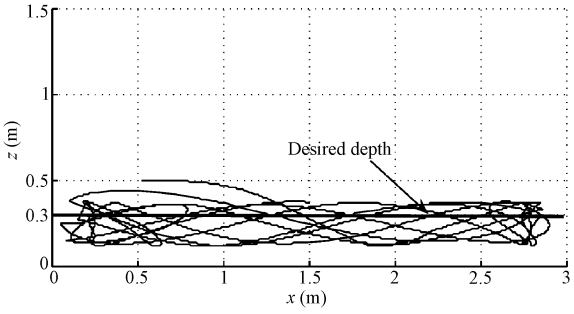
$$|z_f - z_a| = |z_e| < d_z \tag{9}$$



(a) 3D view

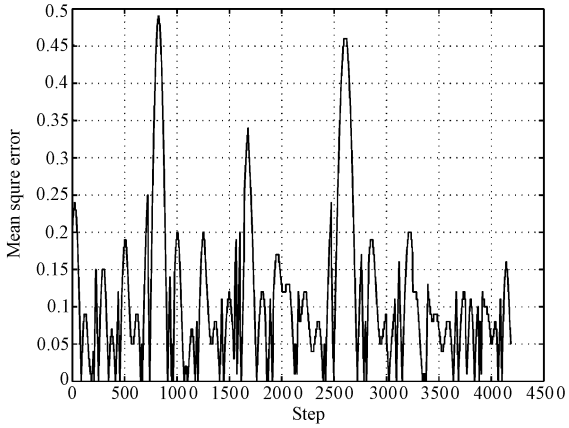


(b) xy plane view

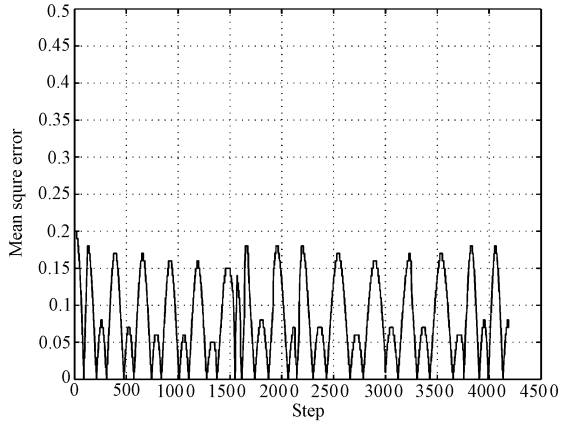


(c) xz plane view

Fig. 6 The trajectory of a simulated fish with the *follow-wall* behaviour



(a) xy plane error



(b) z coordination error

Fig. 7 The mean square error of the simulated fish trajectory

where z_e is the depth difference and d_z is a small positive number of the permitted error. The output of *keep-level* behaviour is used to control the position P_t of the pitch control weights inside the robotic fish head. So that the fish is able to ascent and descent.

Following the general fuzzy controller ruler, z_e and $\dot{z_e}$ are selected as input signals of *keep-level* behaviour. Considering the definition of z_e in (9), $\dot{z_e}$ can be replaced by $\dot{z_f}$. A block diagram of the *keep-level* fuzzy controller is shown in Fig. 8. Here, the pressure sensor data is proportional to the robotic fish depth from water surface.

Similar to the *follow-wall* fuzzy controller in Section 3.1, triangular functions (4), S-type (5) and Z-type (6) functions are chosen to represent fuzzy membership functions for fuzzy input variables $\{u_1,\ u_2\} = \{z_e, \dot{z_f}\}$ and output variables. The membership functions for inputs z_e, $\dot{z_e}$, and output P_t are shown in Fig. 9.

Nine rules are generated by prior knowledge given in Table 2. The logic operator "AND" connects each antecedent, i.e. "IF z_e is *low* AND $\dot{z_f}$ is *down*". Then the consequence is "THEN P_t is $PositiveBig$".

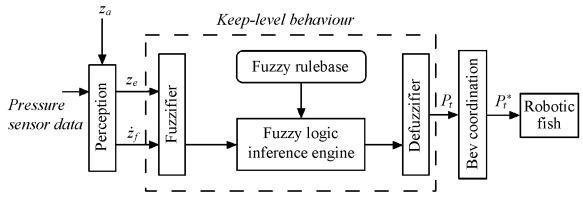


Fig. 8 The fuzzy logic representation of keep-level behaviour

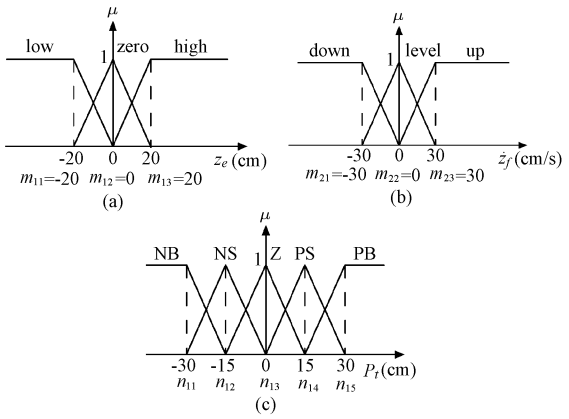


Fig. 9 Membership functions of the input and output variable. m_{ij}, n_{ls}: the centres of membership functions. μ: degree of truth. (a) Membership function of the depth difference z_e; (b) Membership function of the changing speed of depth $\dot{z_f}$ and (c) Membership function of the position of pitch weights P_t

Table 2 Rulebase for *keep-level* fuzzy controller

Rule No.	Input		Output
	z_e	$\dot{z_f}$	P_t
1	LO	D	PB
2	LO	L	PS
3	LO	U	Z
4	ZO	D	PS
5	ZO	L	Z
6	ZO	U	NS
7	H	D	Z
8	H	L	NS
9	H	U	NB

LO: low, ZO: zero, H: high, D: swim down, L: swim level, U: swim up, PB: positive big, PS: positive small, Z: zero, NS: negative small, NB: negative big

In the defuzzifier module, the "centre of gravity (COG)" method is used to combine the outputs represented by the implied fuzzy set from all rules to generate the gravity centroid of the possibility distribution for P_t. It is calculated by

$$P_t = \frac{\sum_{k=1}^{9} v_k q_k}{\sum_{k=1}^{9} q_k} \tag{10}$$

$$q_k = \min\{\mu_{1k_1}, \mu_{2k_2}\} \tag{11}$$

where all variables have the same definitions as in (7) and (8).

3.3 Design of avoid-obstacle behaviour

The objective of the *avoid obstacle* behaviour is to keep the robotic fish away from other static or moving objects during its swimming. The robotic fish, different from most of others mobile robots in the world, moves in a 3D space. It must deal with obstacles locating at all directions including its left, front, right, above and below. Since the robotic fish only swims forward, it is unnecessary to avoid obstacles at the back of the robot. Additionally, the obstacles above the robotic fish are not taken into account at the moment because our robotic fish has no sensor to monitor the top direction.

Fig. 10 shows a block diagram of the *avoid-obstacle* behaviour. The input signals come from four IR sensors, the pressure senor and the accelerometer. The output signals of the fuzzy controller are the yaw angle ω_d relative to the current fish heading and pitch weights position P_t.

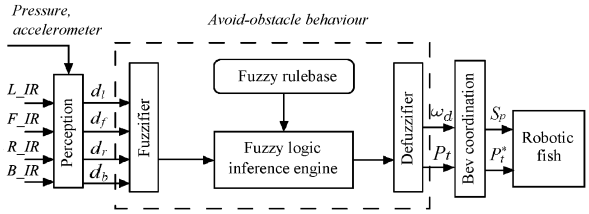


Fig. 10 The fuzzy logic representation of avoid-obstacle behaviour

Similar to the *keep-level* fuzzy controller in Section 3.2, triangular functions (4), S-type (5) and Z-type (6) functions are chosen to represent fuzzy membership functions for fuzzy input variables $\{u_1, u_2, u_3, u_4\} = \{d_l, d_f, d_r, d_b\}$ and output variables $\{\omega_d, P_t\}$. Fig. 11 shows the definitions of fuzzy sets. The rulebase for the fuzzy inference engine is listed in Table 3. The COG method is selected for defuzzification.

$$\omega_d = \frac{\sum_{k=1}^{16} v_k^1 q_k}{\sum_{k=1}^{16} q_k}, P_t = \frac{\sum_{k=1}^{16} v_k^2 q_k}{\sum_{k=1}^{16} q_k} \tag{12}$$

$$q_k = \min\{\mu_{1k_1}, \mu_{2k_2}, \mu_{3k_3}, \mu_{4k_4}\} \tag{13}$$

where all variables have the same definitions as in (7) and (8).

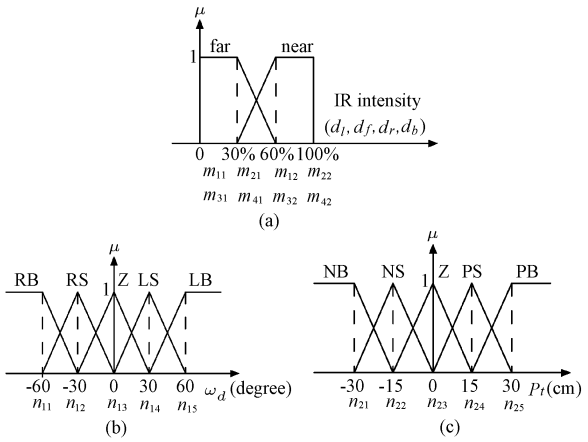


Fig. 11 Membership functions of the input and output variables. m_{ij}, n_{ls}: the centres of membership functions. μ: degree of truth. (a) IR intensities d_l, d_f, d_r, d_b; (b) The yaw angle ω_d relative to the current swimming direction;(c) The position of pitch weights P_t

Table 3 Rulebase for *avoid-obstacle* fuzzy controller

Rule	Input				Output	
No.	d_l	d_f	d_r	d_b	ω_d	P_t
1	N	N	N	N	LS	PB
2	N	N	N	F	LS	NB
3	N	N	F	N	RB	PS
4	N	N	F	F	RB	Z
5	N	F	N	N	LS	PB
6	N	F	N	F	LS	NB
7	N	F	F	N	RS	PS
8	N	F	F	F	RS	Z
9	F	N	N	N	LB	PS
10	F	N	N	F	LB	Z
11	F	N	F	N	LB	PS
12	F	N	F	F	LB	Z
13	F	F	N	N	LS	PS
14	F	F	N	F	LS	Z
15	F	F	F	N	Z	PS
16	F	F	F	F	Z	Z

N: near, F: far, LB: left big turn, LS: left small turn, Z: zero, RS: right small turn, RB: right big turn, PB: positive big (ascent), PS: positive small (ascent), NS: negative small (descent), NB: negative big(descent)

To prove the feasibility of both *keep-level* behaviour and *avoid-obstacle* behaviour, simulation is carried out. In the simulator, the water tank is designed in 3 m long, 3 m wide and 1.5 m deep. The robotic fish starts at the point (0.5,0.5,0.2) and heads to 10 degrees to x-axis. The pitch angle in the initial time is 0. Three column-niform obstacles with 0.2 m diameter are put in water at positions (1.5,1.5,1.0), (2.8,1.5,1.0) and (0.2,0.2,0.2). They are named as obstacles 1,2,3 in the order of appearance above. Fig. 12 shows the initial condition for all experiments.

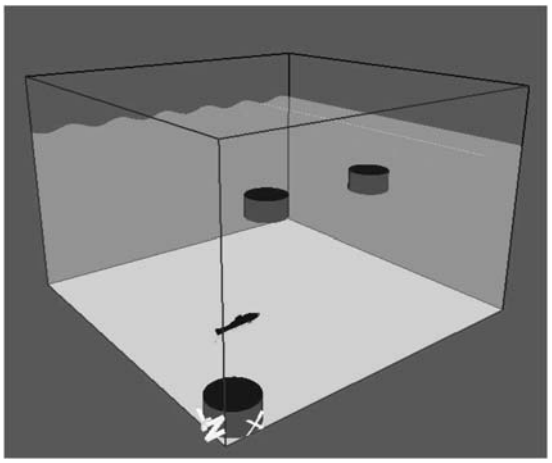


Fig. 12 The initial condition of simulation environment (3D view)

All parameters of both *keep-level* behaviour and *avoid-obstacle* behaviour are given as same as Section 3.2 and Section 3.3. The experiments are done without *cognitive layer* because the main purpose is to test the *behaviour layer*. Assume that the desired depth $z_a = 1$ for *keep-level* behaviour. The gain values in behaviour coordination are (1.0,0) for *avoid-obstacle* behaviour and (0.0,1.0) of *keep-level* behaviour. The *avoid-obstacle* behaviour horizontally controls ω_d and the *keep-level* behaviour controls P_t. Fig. 13 shows the 3D trajectory and its projection on the xy plane. These two figures illustrate that the robotic fish can deal with most of obstacles by using the *avoid-obstacle* behaviour. However, the robotic fish still can be stuck in a corner occasionally, i.e. the discontinuous trajectory at the bottom-left corner in Fig. 13.

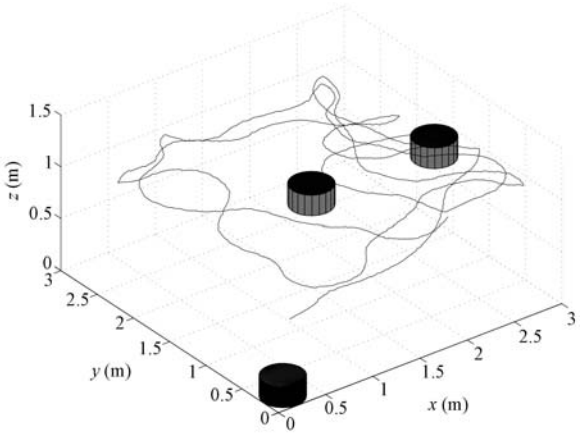
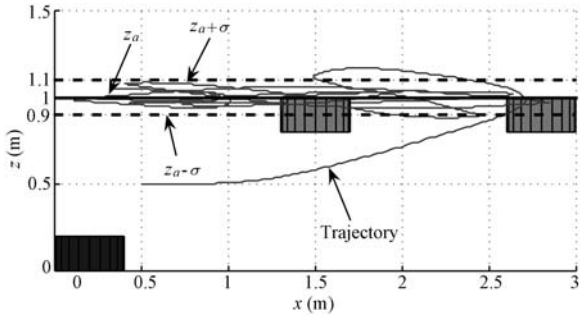


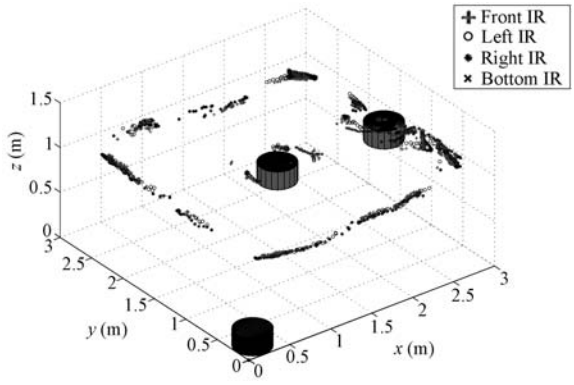
Fig. 13 The trajectory of a simulated robotic fish with the *avoid-obstacle* behaviour and the *keep-level* behaviour

Fig. 14(a) shows the trajectory projection on the xz plane. It displays the performance of *keep-level* be-

haviour. The robotic fish reaches the desired depth quickly and keeps in that level all the time. There is 0.2 m oscillation amplitude because the behaviour parameter is decided by prior knowledge at the moment. Fig. 14(b) shows the IR detection during swimming. The robotic fish detected the four walls of the water tank and the two obstacles at the same level with the desired depth. One exception is the up-right corner in Fig. 14(b), the robotic fish missed that corner because *avoid-obstacle* occurs before the fish reaches the corner.



(a) xz plane view



(b) 3D view of IR sensor data

Fig. 14 The trajectory and IR sensor data of a simulated robotic fish with the *avoid-obstacle* behaviour and the *keep-level* behaviour

4 Experiments and analyses on real robots

All of individual behaviours have been designed based on simulation. To further verify their feasibility, we have tested them on our G9-series robotic fishes. Fig. 15 shows its schematic structure. The robotic fish is equipped with several kinds of sensors to response to the dynamical changes in its environment, its position in the tank, the robot attitude and the internal status (e.g., the battery voltage). A standard configuration of the robotic fish includes four IR sensors, one dual-axis accelerometer/inclinometer, one piezoelectric vibrating gyroscope, one water pressure sensor, three electric current sensors and three servo turning angle sensors. It is able to sense obstacles around it within a range of 40 cm and its depth in the tank. It also can perceive the pitch/roll angle, the 1-order derivative of the yaw angle, the turning angle of the tail joints and the power consumption on them. However, the robotic fish has no ability to localize itself in the horizontal plane.

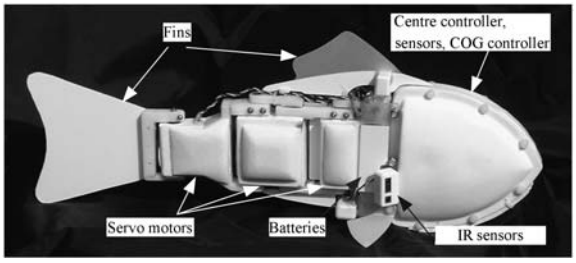


Fig. 15 G9 robotic fish profile

The experimental tank is 5.5 m in length, 1.7 m in width and 1.7 m in height, which has 1.5 m water in depth. The front wall is made of transparent glasses and all other walls are made of concrete. In each test, one robotic fish is put into the tank and its trajectory is recorded by a wide angle overhead camera. At the same time, the onboard pressure sensor data are logged into an MMC memory card in order to record the swimming depth history. Several stones are placed near the behind tank wall to be used as still obstacles. Each stone is about 0.2 m in diameter and 0.2 m in height. Two particular robotic fishes are used in experiments: G9-2 with blue skin and G9-3 with green skin. Both G9-2 and G9-3 have the same internal mechanical structure, but different skin size. G9-2 is about 50 cm in length, 20 cm in height and 12 cm in width, while G9-3 is a little longer which is about 60 cm in length, 22 cm in height and 12 cm in width.

To see the performance of the proposed individual behaviours, G9-2 or G9-3 is commanded to implement a specific task, e.g., to explore the tank border at a desired depth. In other words, the robotic fish is expected to keep its depth, follow the right side wall with a proper distance and avoid bumping with walls and obstacles. Five particular tests are selected here to demonstrate the swimming performance. Table 4 lists all the gain values and the parameters of individual behaviours in these 5 tests. Four factors are defined as follows to evaluate the performance of each test. Table 5 shows the performance of each test according to these evaluation factors.

1) TOC: Mean **T**ime cost of **O**ne **C**ircle swimming, unit is second

2) TSC: Mean **T**ime cost **S**pent in a tank **C**orner, unit is second

3) TBW: In one swimming circle, mean **T**imes of **B**umping with **W**all/obstacle except for corners

4) MSE: **M**ean **S**quare **E**rror to the desired trajectory, unit is m^2. The reference trajectory is 24 cm away from the wall.

Table 4 Parameter configuration of experiments

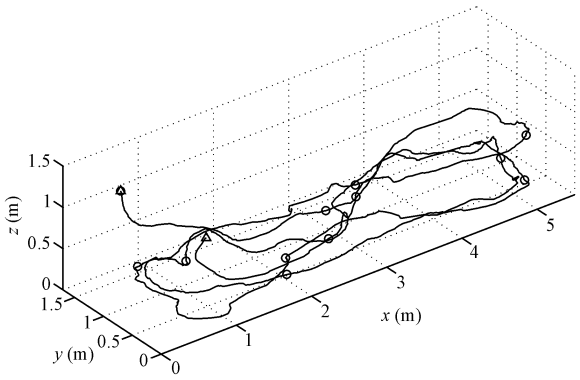
No.	G_{KL}	G_{WF}	G_{AO}	z_a	W_d	Fish
1	[0,1]	[0.5,0]	[1,0.3]	0.25	-1	G9-2
2	[0,1]	[1,0]	[1,0.3]	0.25	-1	G9-2
3	[0,1]	[1,0]	[1.5,0.3]	0.25	-1	G9-2
4	[0,1]	[1,0]	[0.5,0.3]	0.25	-1	G9-2
5	[0,1]	[1,0]	[1,0.3]	1.3	-1	G9-3

$G = [g_w, g_p]$ are the gain value for ω_d and P_t in behaviour coordination. KL denotes *keep-level*, WF denotes *follow-wall*, AO denotes *avoid-obstacle*. z_a is the desired depth setting for *keep-level*, W_d defined which side of wall to follow in *follow-wall*

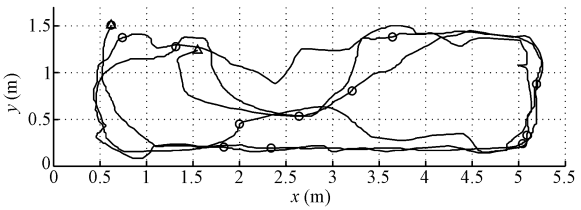
Table 5 Performance of each test in task implement

Test No.	TOC(s)	TSC(s)	TBW	MSE(m^2)
1	202	2.1	0.9	0.243 5
2	181	2.5	2.1	0.147 5
3	170	2.5	1.8	0.250 9
4	205	5.1	4.1	0.121 0
5	190	5.2	0.8	0.051 1

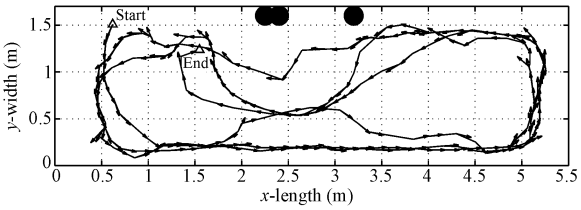
In test 1, G9-2 starts from water surface and its heading direction is pointing along the right side wall. Fig. 16(a) shows the 3D trajectory of G9-2. To make it clear, Fig. 16(b) shows the trajectory projection on the xy plane. Fig. 16(c) plots the heading direction of G9-2 during swimming and the position of obstacles which is indicated by black circle. Figs. 16(b) and (c) indicate that the robotic fish combined *follow-wall* and *avoid-obstacle* behaviours to follow the right side wall and avoid obstacles at the same time.



(a) 3D view



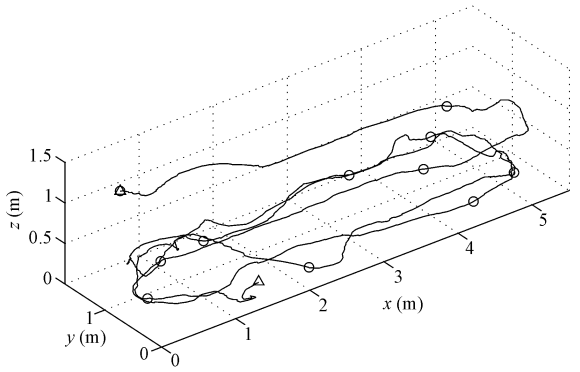
(b) xy plane view



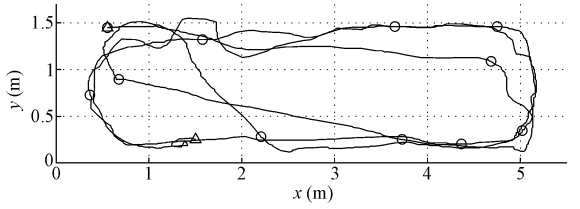
(c) xy plane view with heading direction

Fig. 16 The trajectory of tank border exploration in test 1

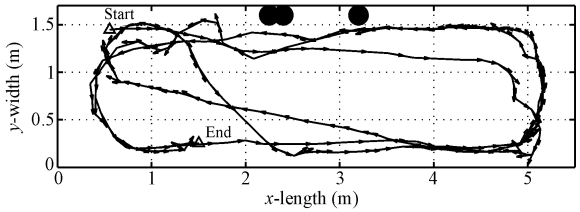
In test 2, G9-2 starts from water surface and its heading direction is pointing along the left side wall rather than the right side wall in test 1. Fig. 17(a) shows the 3D trajectory G9-2. Fig. 17(b) displays the trajectory projection on the xy plane. Fig. 17(c) shows the heading direction of G9-2 during swimming. G9-2 "seeks" the right side wall from "start" point until it reaches point (2.5,0.2) from where G9-2 followed the right side wall until end.



(a) 3D view



(b) xy plane view



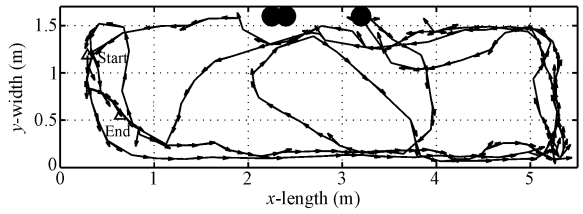
(c) xy plane view with heading direction

Fig. 17 The trajectory of tank border exploration in test 2

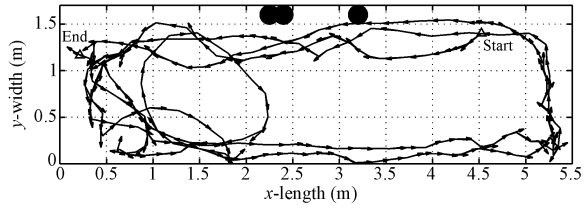
In test 3, G9-2 starts from the tank bottom and the gain value g_ω of *avoid-obstacle* is increased from 1.0 to 1.5 and g_ω of *follow-wall* is kept on 1.0. Fig. 18(a) shows that G9-2 spent less time in one circle swimming but it is more likely to lose the wall to follow because *avoid-obstacle* behaviour has more weights in behaviour coordination than *follow-wall* behaviour.

In test 4, G9-2 starts from the tank bottom and the gain value g_ω of *avoid-obstacle* is decreased from 1.5 to 0.5 and g_ω of *follow-wall* is kept on 1.0. Fig. 18(b) shows that G9-2 spent more time in one circle swimming and had smooth trajectory on following wall. However, lower weights on *avoid-obstacle* make G9-2 stumble on the bottom-left corner.

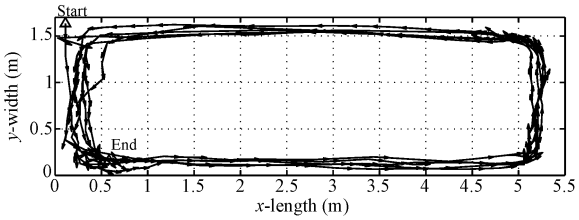
In test 5, to evaluate behaviours at different swimming levels with different length of robotic fish, G9-3 green fish is used to implement the tank border exploration at 1.3 m water level. Fig. 18(c) shows the performance. Because there is not outstanding obstacle at this level, the trajectory of G9-3 is quite smooth and close to the desired trajectory. However, due to longer body of G9-3, it didn't do very well to deal with tank corner and spent more time on avoiding corners.



(a) Test 3



(b) Test 4



(c) Test 5

Fig. 18 Swim direction and the xy plane trajectory projection of tank border exploration in test 3, 4 and 5

From these tests, it can be concluded that 1) Higher gain value of *avoid-obstacle* behaviour makes the robotic fish do better on avoiding obstacles and corners but it seems to be too sensitive to them, e.g., at the point (3.3, 1.5) in Fig .18(a) G9-2 did several left hard turnings to avoid the obstacle and made its trajectory deflect from the wall; 2) Higher gain value on *follow-wall* behaviour makes the robotic fish good at following wall and keeping trajectory smooth with smaller MSE. However, the time spent in a corner is significant much because the robotic fish relied more on *follow-wall* than on *avoid-obstacle*. 3) Fixed gain value in behaviour coordination can not satisfy all kinds of situations during swimming. If *cognitive layer* can adaptively change the gain values depending on the context swimming situation, the robotic fish would swim much better.

5 Conclusion

In this paper, we mainly discuss the design of the behaviour layer in the hybrid control architecture of our robotic fish, including the design of individual behaviours. All individual robotic fish behaviours are inspired by the real fish behaviours and implemented by FLCs because FLC has advantage on non-linear control and real-time implementation. Simulation results and real robotic fish experiments are given to prove their feasibility and performance. However, the constant gain value during entire task can not properly deal with all kinds of situations. An adaptive gain value is expected to do better, which will be investigated in the future.

Acknowledgments

The authous would like to thank Ian Dukes, George Francis and Rob Knight at Essex for their contribution toward the project.

References

[1] A. D. Horchler, R. E. Reeve, B. Webb, R. D. Quinn. Robot Phonotaxis in the Wild: A Biologically Inspired Approach to Outdoor Sound Localization. *Advanced Robotics*, vol. 18,

no. 8, pp. 801−816, 2004.

[2] F. W. Grasso, T. R. Consi, D. C. Mountain, J. Atema. Biomimetic Robot Lobster Performs Chemo-orientation in Turbulence Using a Pair of Spatially Separated Sensors: Progress and Challenges. *Robotics and Autonomous Systems*, vol. 30, no. 1-2, pp. 115−131, 2000.

[3] J. Connell. A Colony Architecture Foe and Artificial Creature. Techincal Report 1151, MIT Artificial Intelligence Lab, USA, 1989.

[4] J. M. Anderson, P. A. Kerrebrock. The Vorticity Control Unmanned Undersea Vehicle (VCUUV): An Autonomous Robot Tuna. In *Proceedings of 10th International Symposium on Unmanned Untethered Submersible Technology*. Durham, NH, pp. 63−70, 1997.

[5] K. Naom. Control Performance in the Horizontal Plane of a Fish Robot with Mechanical Pectoral Fins. *IEEE Journal of Oceanic Engineering*, vol. 25, no. 1, pp. 121−129, 2000.

[6] J. Witting, K. Safak, G. Adams. SMA Actuators Applied to Biomimetic Underwater Robots. *Neurotechnology for Biomimetic Robots*. J. Ayers, J. Davis, A. Rudolph, Eds., MIT Press, USA, pp. 117−136, 2001.

[7] J. Liang. Researchful Development of Underwater Robofish II—Development of a Small Experimental Robofish. *Robot*, vol. 24, no. 3, pp. 234−238, 2002.

[8] J. Yu, S. Wang, M. Tan. A Simplified Propulsive Model of Biomimetic Robot Fish and Its Realization. *Robotica*, vol. 23, no. 1, pp. 101−107, 2005.

[9] S. Guo, T. Fukuda, N. KATO, K. OGURO. Development of Underwater Microprobe Using IMPF Actuator. In *Proceedings of IEEE International Conference on Robotics and Automation*, Leuven, Belgium, pp. 1829−1834, May 1998.

[10] R. C. Arkin. *Behavior-based Robotics*, MIT Press, USA, chap. 3, p. 94, 1998.

[11] N. Nilsson. Teleo-reactive Programs for Agent Control. *Journal of Artificial Intelligent Research*, vol. 1, pp. 139−158, 1994.

[12] J. Connell, P. Viola. Cooperative Control of a Semi-autonomous Mobile Robot. In *Proceedings of IEEE International Conference on Robotics and Automation*, OH, USA, vol. 2, pp. 1118−1121, 1990.

[13] W. L. Xu, S. K. Tso. Sensor-based Fuzzy Reactive Navigation of a Mobile Robot Through Local Target Switching. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, vol. 29, no. 3, pp. 451−459, August 1999.

[14] P. Vadakkepat, O. C. Miin, X. Peng, T. H. Lee. Fuzzy Behavior-based Control of Mobile Robots. *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 4, pp. 559−565, 2004.

[15] O. Khatib. Real-time Obstacle Avoidance for Manipulators and Mobile Robots. In *Proceedings of IEEE International Conference on Robotics and Automation*, St. Louis, vol. 2, pp. 500−505, 1985.

[16] J. Borenstein, Y. Koren. Real-time Obstacle Avoidance for Fast Mobile Robots12. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 5, pp. 1179−1187, 1989.

[17] M. G. Slack. Situationally Driven Local Navigation for Mobile Robots. Technical Report JPL Publication 90-17, NASA Jet Propulsion Laboratory, Pasadena, CA, April 1990.

[18] R. Zapata, M. Perrier, P. Lepinary, P. Thompson, B. Jouvencel. Fast Mobile Robots in Ill-structured Environments. In *Proceedings of IEEE/RSJ International Workshop on Intelligence for Mechanical Systems*, Osaka, Japan, vol. 2. pp. 793−798, Nov. 1991.

[19] J. Liu, H. Hu, D. Gu. A Hybrid Control Architecture for Autonomous Robotic Fish. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, Aug 2006.

[20] P. Rusu, E. M. Petriu, T. E. Whalen, A. Cornell, H. J. W. Spoelder. Behavior-based Neuro-fuzzy Controller for Mobile Robot Navigation. *IEEE Transactions on Instrumentation and Measurement*, vol. 52, no. 4, pp. 1335−1340, Aug 2003.

[21] P. Angelov, D. Filev. An Approach to Online Identification of Takagi-Sugeno Fuzzy Models. *IEEE Transactions on Systems, Man and Cybernetics—Part B*, vol. 34, no. 1, pp. 484−498, 2004.

[22] E. Mamdani, S. Assilian. An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1−13, 1975.

[23] R. Bellman. *Adaptive Control Processes: A Guided Tour*, Princeton University Press, New Jersey, 1961.

[24] J. Liu, H. Hu. A 3D Simulator for Autonomous Robotic Fish. *International Journal of Automation and Computing*, vol. 1, no. 1, pp. 42−50, 2004.

Jin-Dong Liu received his B.Sc. degree in industrial automation from Shenyang Institute of Aeronautical Engineering, China in 1999, and the M.Sc degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences in 2002. He is currently pursuing his Ph.D. degree at the University of Essex in the U.K since 2003.

His current research interests include autonomous mobile robots, intelligent control and human Centred robotics. He has published a number of papers in these areas.

Mr. Liu is a student member of IEEE.

Huosheng Hu received his M.Sc. degree in industrial automation from the Central South University in China, and the Ph.D. degree in robotics from the University of Oxford in the United Kingdom. Currently, He is a professor in Department of Computer Science, University of Essex, leading the Human Centred Robotics Group.

He has published over 200 papers in journals, books and conferences, and received two best paper awards. His research interests include autonomous mobile robots, human-robot interaction, evolutionary robotics, multi-robot collaboration, embedded systems, pervasive computing, sensor integration, RoboCup, intelligent control and networked robotics.

Prof. Hu is a Chartered Engineer, a senior member of IEEE, and a member of IET, AAAI, ACM, IASTED and IAS.