Generating Chaos with Neural-Network-Differential-Equation for Intelligent Fish-Catching Robot

Haruhiro Inukai, Mamoru Minami, Akira Yanou
Graduate School of
Natural Science and Technology,
Okayama University
3-1-1 Tsushima-naka, Kita-ku, Okayama-shi, Japan
Email: en422807@s.okayama-u.ac.jp

***Abstract*—We have been investigating a method to compose some intelligent robot. Continuous catching and releasing experiment of several fishes induces the fishes to find some escaping strategies such as staying stationarily at corners of the pool. To make fish-catching robot intelligent more than fish's adapting and escaping abilities, we have proposed a chaos-generator comprising Neural-Network-Differential-Equation(NNDE) and an evolving mechanism to have the system generate chaotic trajectories as many as possible. We believe that the fish could not be adaptive enough to escape from chasing net with chaos motions since unpredictable chaotic motions of net may go beyond the fish's adapting abilities to the net motions. In this report we confirmed that the proposed system can generate plural chaos by examining chaotic characters of chaos trajectories generated by NNDE through Lyapunov number, Poincare return map, initial value sensitivity, fractal dimension and bifurcation map.**

I. INTRODUCTION

Animal world has been used conceptually by robotics as a source of inspiration for machine intelligence. For the purpose of studying animal behavior and intelligence, the model of interaction between animals and machines is proposed in researches like [1]. Crucial characteristic of machine intelligence is that the robot should be able to use input information from sensor to know how to behave in a changing environment and furthermore can learn from the environment for safety like avoiding obstacle. As known universally the robot intelligence has reached a relatively high level, still the word "intelligence" is an abstract term, so the measurement of the intelligence level of a robot has become necessary. A practical and systematic strategy for measuring machine intelligence quotient (MIQ) of human-machine cooperative systems is proposed in [2].
In our approach to pursue intelligent robot, we evaluate the intelligence degree between fish and the robot by Fish-Catching operation. We considered that the competitive relation can be very meaningful as one way to discuss robotic intelligence.
In recent years, visual tracking and servoing in which visual information is used to direct the end-effector of a manipulator toward a target object has been studied in some researches [3], [4]. By evolutionary algorithms [5], Visual Servoing and Object Recognizing based on the input image from a CCD camera mounted on the manipulator has been studied in our laboratory(Fig.1) [6], and we succeeded in catching a fish by a net attached at the hand of the manipulator based on the real-time visual tracking under the method of Gazing GA [7] to enhance the real-time pose-tracking ability.

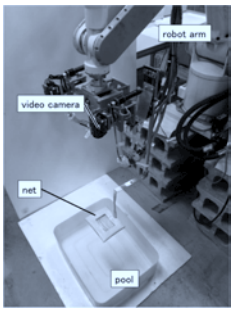


Fig. 1. Fish Catching system PA10

Through above experiments, we have learned that it is not effective for fish catching to simply pursue the current fish position by visual servoing with velocity feedback control. Actually, the consistent tracking is sometimes impossible because the fish can alter motion pattern suddenly maybe under some emotional reasons of fear. Those behaviors are thought to be caused by emotional factors and they can also be thought as a kind of innate fish intelligence, even though not in a high level.

While observing the fish's adapting behavior to escape in the competitive relations with the robot, that is continuous catching/releasing experiments, we found that we can define a "Fish Learning Speed"(FLS) representing decreasing velocity of fish number caught by the robot through continuous catching/releasing operation. Through this measure we can compare the innate intelligence of the fish and the intelligence of the robot.
It has been well known that many chaotic signals exist in our body, for example, in nerves, in motions of eye-balls and in heart-beating periods [8], [9]. Therefore we thought that imitating such animal's internal dynamics and putting chaos into robots have something meaningfulness to address fish's intelligence. We embed chaos into the Robot Dynamics in order to supplement the deficiency of our Fish-Catching system [10]. Therefore what we have to pay attention to the fish's nature that the fish does continue to conceive always escaping strategy against new stressing situation. This means that robot's intelligence to override the fish's thinking ability needs infinite source of idea of catching motions.

978-1-4799-7862-5/15/$31.00 © 2015 IEEE

To generate such catching motion, we have proposed Neural-Network-Differential-Equation(NNDE) that can generate plural chaos and inherently have a possibility to be able to generate infinite varieties of chaos, derived from the neural network's ability to approximate any nonlinear function as accurate as with desirable precision[11], [12]. In this paper, we report analyses of chaos generated by NNDE.

II. FISH TRACKING AND CATCHING

The problem of recognition of a fish and detection of its position/orientation is converted to a searching problem of $\boldsymbol{r}(t) = [x(t), y(t)]^T$ in order to maximize $F(\boldsymbol{r}(t))$, where $F(\boldsymbol{r}(t))$ represents correlation function of images and fish-shaped matching model. $F(\boldsymbol{r}(t))$ is used as a fitness function of GA [7]. To recognize a target in a dynamic image input by video rate, 33 [fps], the recognition system must have real-time nature, that is, the searching model must converge to the fish in the successively input raw images. An evolutionary recognition process for dynamic images have been realized by such method whose model-based matching by evolving process in GA is applied at least only one time to one raw image input successively by video rate. We named it as "1-Step GA" [6]. When the converging speed of the model to the target in the dynamic images should be faster than the swimming speed of the fish in the dynamic images, then the position indicated by the highest genes represent the fish's position in the successively input images in real-time. We have confirmed that the above time-variant optimization problem to solve $\boldsymbol{r}(t)$ maximizing $F(\boldsymbol{r}(t))$ could be solved by "1-Step GA". $\boldsymbol{r}(t) = [x(t), y(t)]^T$ represents the fish's position in Camera Frame whose center is set at the center of catching net, then $\boldsymbol{r}(t)$ means position deviation from net to Fish, means $\boldsymbol{r}(t) = \Delta\boldsymbol{r}(t)$ The desired hand velocity at the i-th control period $\dot{\boldsymbol{r}}_d^i$ is calculated as

$$\dot{\boldsymbol{r}}_d^i = \boldsymbol{K}_P \Delta\boldsymbol{r}^i + \boldsymbol{K}_V(\Delta\boldsymbol{r}^i - \Delta\boldsymbol{r}^{i-1}) \tag{1}$$

where $\Delta\boldsymbol{r}^i$ denotes the servoing position error detected by 1-Step GA [6]. $\boldsymbol{K}_P$ and $\boldsymbol{K}_V$ are positive definite matrix to determine PD gain.

Now we add chaos items to (1) above, and we also need to redefine the meaning of $\dot{\boldsymbol{r}}_d^i$. The simple PD servo control method given by (1) is modulated to combine a visual servoing and chaos net motion into the controller as follows,

$$\Delta\boldsymbol{r}^i = k_1 \cdot \Delta\boldsymbol{r}_{fish}^i + k_2 \cdot \Delta\boldsymbol{r}_{chaos}^i \tag{2}$$

Here $\Delta\boldsymbol{r}_{fish}^i = \begin{bmatrix} \Delta x_{fish}^i & \Delta y_{fish}^i \end{bmatrix}$ is the tracking error of fish from the center of camera frame, and $\Delta\boldsymbol{r}_{chaos}^i = \begin{bmatrix} \Delta x_{chaos}^i & \Delta y_{chaos}^i \end{bmatrix}$ denotes a chaotic oscillation in $x-y$ plane around the center camera frame. the hand motion pattern can be determined by the switch value k_1 and k_2. $k_1 = 1$ and $k_2 = 0$ indicate visual servoing, and $k_1 = 0$ and $k_2 = 1$ indicate the net will track chaotic trajectory made by NNDE being explained later in this report.

The robot used in this experimental system is a 7-Link manipulator, Mitsubishi Heavy Industries PA-10 robot.

III. PROBLEM OF FISH-CATCHING

In order to check the system reliability in tracking and catching process, we kept a procedure to catch a fish and release it immediately continuously for 30 minutes. We released 5 fishes (length is about 40[mm]) in the pool in advance, and once the fish got caught, it would be released to the same pool at once. The result of this experiment is shown in Fig.2, in which the vertical axis represents the number of fishes caught in successive 5 minutes and the horizontal axis represents the catching time. Based on the idea that the fish may get tired as time passing, we had expected that the capturing operation would become smoother.

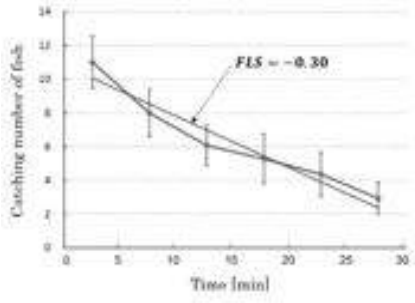
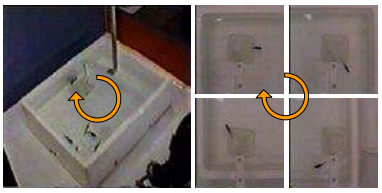


Fig. 2. Result of catching number



(a) Motion (1) of a fish

(b) Motion (2) of a fish

(c) Motion (3) of a fish

Fig. 3. Fish motion

But to our astonishment, the number of fishes caught decreased gradually. The reason of decreased catching number may lie in the fish learning ability. For example, the fish learn how to run away around the net (Fig.3(a)) by circular swimming motion with about constant velocity, having made a steady state position error that the net cannot reach to the chasing fish.

Or the fish stay in the opposite corner against the net in the pool (Fig.3(b)). And also, the fish keep staying within the clearance between the edge of the pool and the net (Fig.3(c)) where the net is inhibited to enter. To solve these problems, and to achieve more intelligent fish catching systems, we thought that the net's chaos behavior with many chaotic variey can be method to overcome those fish's escaping intelligence, since huge variety of chaos trajectories seems to be unpredictable for the fish to adapt them. This strategy to overcome fish's adaptive intelligence is based on a hypothesis that unpredictability of the motion of the chasing net produced by plural chaos can make the fish's learning logic confuse. Then we propose Neural-Network-Differential-Equation to generate chaos as many as possible.

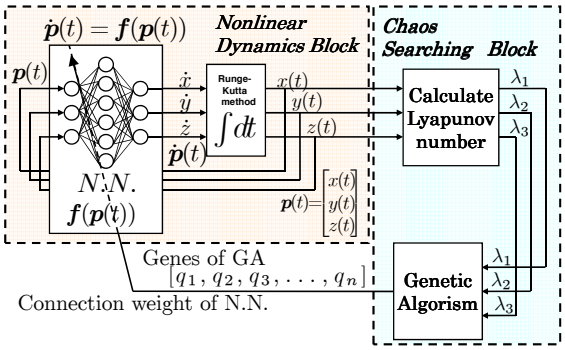


Fig. 4. Block diagram of Chaos Generation

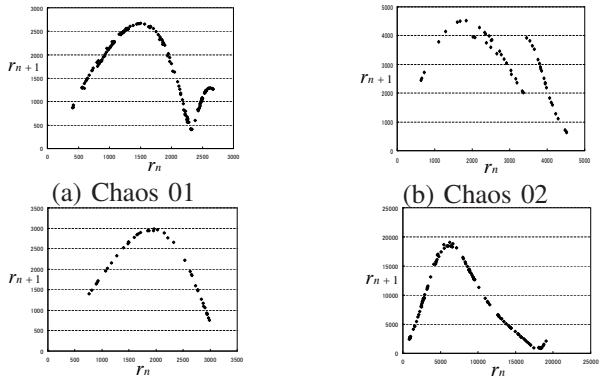


Fig. 5. Poincare return map of Chaos 01∼04

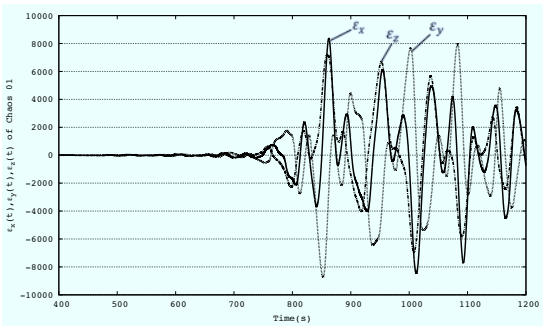


Fig. 6. Sensitivity of initial value (Chaos 01)

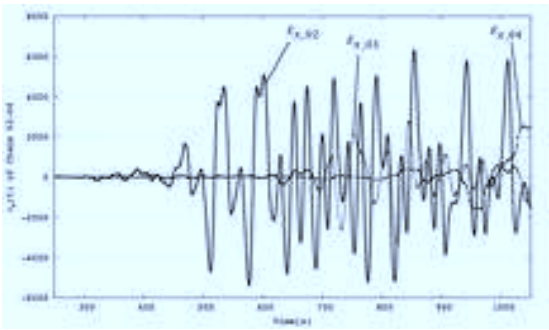


Fig. 7. Sensitivity of initial value (x-coordinate)

IV. FISH LERNING SPEED

To evaluate numerically how fast the fish can learn to escape the net, we adapted Linear Least-Square approximation to the fish-catching decreasing tendency, resulting in FLS=-0.30 as shown in Fig.2, which exhibit the number of fish caught by the robot in five minutes, on condition of the caught fish released into the same pool immediately. The decreasing coefficient -0.30 represents adapting or learning speed of the fish as a group when the fish's intelligence is compared with robot's catching ability. We named the coefficient as "*Fish Learning Speed*"(FLS), since the decreasing tendency that is the value of coefficient can represent the fish's learning speed to conceive a new escaping strategies—stay at corner or swim with constant speed on a circle trajectory.

V. CHAOS GENERATE SYSTEM

we proposed a new chaos generator using N.N. feedforward. In the chaos generator we proposed, nonlinear differential equation is expressed with N.N.. N.N. has been proven to have an ability to approximate any nonlinear functions with arbitrarily high accuracy[13][14][15]. Including the function expressed by N.N. in a differential equation, a nonlinear function part of the nonlinear differential equation can be changed variously. Considering N.N. which has input layer, hidden layer and output layer and has nonlinear mapping $\boldsymbol{p}(t) = [x(t), y(t), z(t)]^T$ to $\boldsymbol{f}(\boldsymbol{p}(t))$(3). The N.N. output is $\dot{\boldsymbol{p}}(t) = [\dot{x}(t), \dot{y}(t), \dot{z}(t)]^T$ and $\boldsymbol{p}(t)$ is obtained by integrating $\dot{\boldsymbol{p}}(t)$ with Runge-Kutta method. A closed loop system is composed by feedback $\boldsymbol{p}(t)$ to the N.N. input and this system represents Eq. 3. A block diagram which represents Eq .3 as N.N. is shown in Fig. 4. Here, a method that search the N.N. weight coefficients which generate chaos is introduced. Fig. 4 represents the block diagram that finds chaos by using Genetic Algorithm(GA). The GA evolves genes representing a vector $\boldsymbol{q_i} = [q_{1i}, q_{2i}, \ldots, q_{ni}]^T$ which means N.N. weight coefficients quantity to search N.N. weight coefficients maximizing g_i defined as follow,

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{f}(\boldsymbol{p}(t)). \tag{3}$$

$$g_i = k_1 \cdot \lambda_1 i - k_2 \cdot |\lambda_2 i| - k_3 \cdot \lambda_3 i. \tag{4}$$

where k_1, k_2 and k_3 are positive coefficients. To evaluate $\boldsymbol{q_i}$, $\boldsymbol{q_i}$ is set to N.N. as weight coefficients and Eq. 3 is solved by numerical integration and solved trajectory $p_i(t)$ appears. In addition, Lyapunov numbers $\boldsymbol{L} = [\lambda_1, \lambda_2, \lambda_3]^T, (\lambda_1 > \lambda_2 > \lambda_3)$ of the trajectory are calculated. Evolution of GA tries to maximize the fitness function g_i. The relationship between positive and negative Lyapunov spectrum is $(+, 0, -)$ and g_i is composed to get large value when $\lambda_1 i, \lambda_2 i, \lambda_3 i$ correspond with Lyapunov spectrum $(+, 0, -)$. g_i maximization by GA leads NNDE to have chaos trajectory. This procedure is repeated many times and chaos trajectory can be generated by searching a trajectory which satisfy Lyapunov number of chaos with GA.

VI. CHAOS VERIFICATION

In this section, Lyapunov number, Poincare return map, Sensitivity of initial value and Fractal dimension are introduced as indices of chaos.

A. Lyapunov number

As one of criteria to evaluate a chaos' character of time function $\boldsymbol{f}$(t) at discrete time t_i in time domain, Lyapunov

number expressed by the following equation is well known,

$$\lambda = \lim_{N\to\infty} \frac{1}{N} \sum_{i=0}^{N-1} \log |f'(t_i)|, \tag{5}$$

where positive value can represent that the irregular oscillation diverts from a standard trajectory, which expands like exponential function

B. Poincare return map

Poincare return map verifies whether trajectory has a structure of stretching and folding. Stretching means a trajectory goes away from a point of convergence and folding means a trajectory is brought back to a point of convergence. This structure is a basic chaos property.

C. Sensitivity of initial value

The small perturbation of the current trajectory may lead to significantly different future behavior. Sensitivity of initial value is popularly known as the "butterfly effect". This structure is a basic chaos property too.

D. Fractal dimension

In this subsection, Fractal dimension is introduced as an index of chaos. Fractal dimension represents self-similarity of chaos attractor and if Fractal dimension of a trajectory is not integer, it can be said that the trajectory has self-similarity.

VII. CHAOS VERIFICATION RESULT

So far we have found four chaos patterns with different weight coefficients explored by GA mentioned in the previous section. We named them with serial numbers as chaos 01∼04. The followings are verifications of these chaos with each individual characters.

A. chaos 01

In this subsection, chaos 01 is verified whether the trajectory has chaos properties or not from viewpoints of Lyapunov number, Sensitivity of initial value, Poincare return map and Fractal dimension.

1) Lyapunov number: Lyapunov numbers are $\lambda_1 = 0.014585$, $\lambda_2 = -0.003314$ and $\lambda_3 = -0.165381$. These corresponded to the Lyapunov spectrum of chaos,$(+, 0, -)$.

2) Poincare return map: Poincare section is put at $x - z$ plane($x < 0$) and a difference between the origin and trajectory intersection with poincare section is defined as r. Chaos 01's poincare return map appear in Fig. 5(a). One dimensional map can be seen in the figure, from which we can understand that the map represents streaching (left half of the Fig. 5(a)) and folding (right half) that are essential characters to generate chaos.

3) Sensitivity of initial value: Here, a difference between trajectories with minutely different initial value are shown as $\varepsilon_x, \varepsilon_y, \varepsilon_z$. Initial values of a trajectory $(x_1(t), y_1(t), z_1(t))$ are set as $x_1(0) = 1.00, y_1(0) = 1.00, z_1(0) = 1.00$ and initial value of another trajectory $(x_2(t), y_2(t), z_2(t))$ are set as $x_2(0) = 1.01, y_2(0) = 1.01, z_2(0) = 1.01$. $\varepsilon_x, \varepsilon_y, \varepsilon_z$ are defined as $\varepsilon_x = x_1(t) - x_2(t), \varepsilon_y = y_1(t) - y_2(t), \varepsilon_z = z_1(t) - z_2(t)$. Fig. 6 shows $\varepsilon_x, \varepsilon_y, \varepsilon_z$ of chaos 01. Because $\varepsilon_x, \varepsilon_y, \varepsilon_z$ are almost zero from 0s to 400s, they are not shown until 400[s]. We can see from Fig. 6 that the difference between the two trajectories with minute difference of initial values diverts suddenly about 800 seconds having passed, this means the slight different initial values make large separation with each other, indicating sensitivity of initial value, which is one of the character of chaos.

TABLE I. CHARACTER OF CHAOS 02∼04

	Chaos02	Chaos03	Chaos04
Lyapunov number	0.01919 −0.00733 −0.10379	0.015934 −0.002172 −0.123026	0.01208 −0.00143 −0.075448
Fractal dimension	1.78474	1.89099	1.8799
Poincare return map	Fig. 5(b)	Fig. 5(c)	Fig. 5(d)
Sensitivity of initial value	Fig. 7		

TABLE II. WEIGHTS OF THE CHAOS 03 AND CHAOS 04

	chaos 03	chaos 04	Fig. 9
q_{11}	0.829098955	-0.108415351	$-1.0 \sim +1.0$
q_{21}	-0.561699855		
q_{31}	-0.902555886	all values are identical to the left values from q_{21} to w_{63}	all values are identical to the left values from q_{21} to w_{63}
q_{12}	-0.640772107		
q_{22}	-0.017715724		
q_{32}	0.196276799		
q_{13}	-0.627588312		
q_{23}	0.862119478		
q_{33}	0.678095674		
q_{14}	0.815579461		
q_{24}	-0.79250782		
q_{34}	0.100541695		
q_{15}	-0.333791104		
q_{25}	-0.891996643		
q_{35}	-0.869291218		
q_{16}	-0.616632334		
q_{26}	-0.895109483		
q_{36}	0.547966735		
w_{11}	-0.145769436		
w_{21}	-0.479484245		
w_{31}	0.047714961		
w_{41}	-0.52028687		
w_{51}	-0.749568933		
w_{61}	0.153582055		
w_{12}	-0.659693294		
w_{22}	0.469321736		
w_{32}	0.830869001		
w_{42}	0.083115892		
w_{52}	0.758999008		
w_{62}	0.554406043		
w_{13}	0.858548867		
w_{23}	-0.639246204		
w_{33}	-0.589715419		
w_{43}	-0.660547799		
w_{53}	-0.480460822		
w_{63}	0.649317159		

4) Fractal dimension: Fractal dimension of chaos 01 is 1.36058, which is non integer and chaos 01 has self-similarity. Therefore, the chaos property of chaos 01 has been verified from the viewpoint of Lyapunov number, Sensitivity of initial value, Poincare return map and Fractal dimension.

B. chaos 02∼04

Verifications of chaos 02∼04 are summarized at TABLE I. Lyapunov number column represent $\lambda_1, \lambda_2, \lambda_3$ from the top and Lyapunov spectrums of chaos 02∼04 are confirmed as $(+, 0, -)$. And also the table shows Fractal dimensions of chaos 02∼04 are non integer and thus chaos 02∼04 have self-similarity. Poincare return maps are shown in Fig. 5(b),(c),(d) and one dimensional map can be seen in these figures. Time-profile of differences between trajectories with minutely different initial value are shown in Fig. 7. $\varepsilon_x\_i(i = 02, 03, 04)$ represent chaos 02∼04 x-coordinate differences between trajectories. Because $\varepsilon_x\_02, \varepsilon_x\_03, \varepsilon_x\_04$ are almost zero from 0s to 250s, they are not shown until 250s. Initial values are same as chaos 01's. As for y and z coordinates, they are similar to x, omitted to spare the space. We can see from Fig. 7 that the difference between the two trajectories with minute difference of initial values diverts suddenly. Each chaos properties are confirmed about chaos 02∼04 as well as chaos 01.

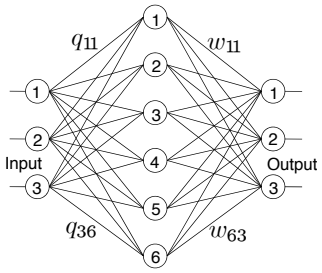


Fig. 8. Neural network structure for nonlinear function generation

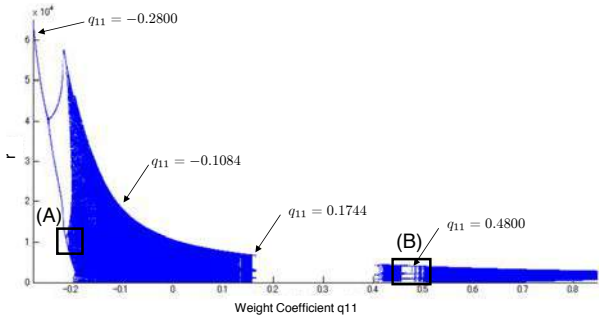


Fig. 9. Bifurcation diagram against q_{11}

VIII. SENSITIVITY OF NEURON'S WEIGHT

A. Bifurcation diagram

After a solved trajectory $\boldsymbol{p}(t) = [x(t), y(t), z(t)]$ has been obtained by numerical integration with weight coefficients found by GA procedure, $\boldsymbol{p}(t)$ crosses through $x - z$ plain. In case of the period of $\boldsymbol{p}(t)$ being 1 cycle, one fixed point appears on the $x - z$ plain. If it is 2 cycle, two fixed points appear. Further, with the $\boldsymbol{p}(t)$ having k cycle, the spots on $x - z$ plain number k.
Bifurcation diagram represents fluctuation in the number of the crossing points depending on changing a parameter of chaos. In this case, a changing parameter is one weight coefficient value of a neural network . When $\boldsymbol{p}(t)$ is chaos, it has infinite periods, then the crossing points appear on the $x - z$ plain infinitely.

B. In the case of chaos 03 and chaos 04

In TABLE II, N.N. weight coefficients of chaos 03 and chaos 04 are shown. From this table, we noticed weight coefficients of N.N. that generated chaos 03 are almost similar to chaos 04's. Only one weight coefficient is different, that is "q_{11}" in Fig. 8.(Weight coefficients of chaos 01 and chaos 02 are completely different from these of chaos 03 and chaos 04) From the fact, we think "q_{11}" is related to the generation of chaos. So we increased the weight q_{11} gradually from "-1.0" to "$+1.0$" at 0.0001 interval and compare their trajectories. Fig. 9 shows the bifurcation diagram and the vertical axis means a distance from the center of trajectory. Other weight coefficients were not changed. In the case of $-1.0 \leq q_{11} \leq -0.28$, $0.18 \leq q_{11} \leq 0.39$ and $0.85 \leq q_{11} \leq 1.0$, the trajectories diverged without cyclic motion.

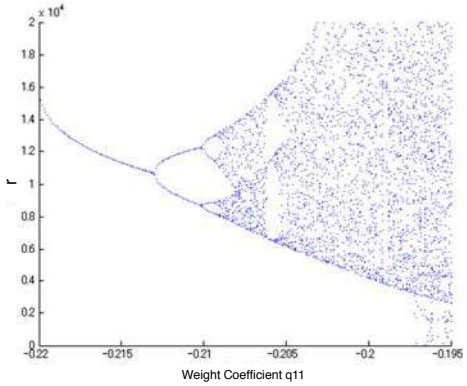


Fig. 10. Pitchfork bifurcation ((A) of Fig.9)

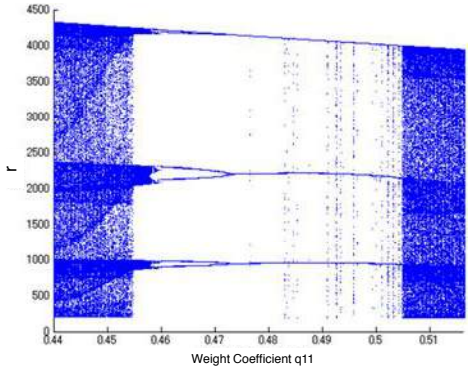
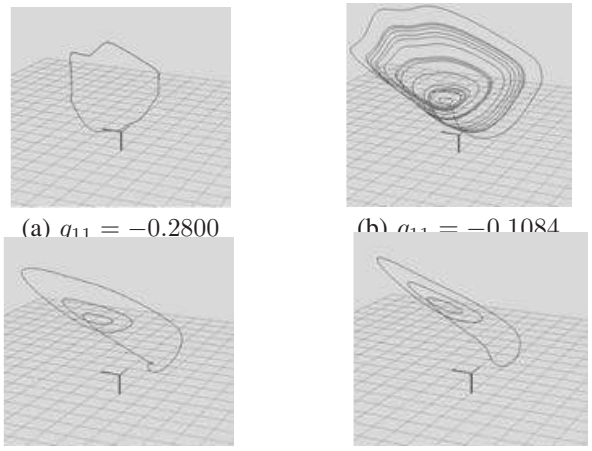


Fig. 11. Window with 3 period ((B) of Fig.9)



(a) $q_{11} = -0.2800$ (b) $q_{11} = -0.1084$

(c) $q_{11} = 0.1744$ (d) $q_{11} = 0.4800$

Fig. 12. Generated Trajectories

C. Confirmation of Pitchfork bifurcation

Fig. 10 shows a magnification of Fig. 9(A)($-0.220 \leq q_{11} \leq -0.195, 0.0 \leq r \leq 20000$). As increasing q_{11}, the trajectory $\boldsymbol{p}(t)$ is bifurcated into two, four, and falls into chaos behavior. This is said to be "period doubling", and typical phenomenon in chaos. Pitchfork bifurcation is confirmed in Fig. 10.

D. Confirmation of chaos window

Fig. 11 shows a magnification of Fig. 9(B)($0.440 \leq q_{11} \leq 0.5167, 0.0 \leq r \leq 4500$). Window with 3 period is confirmed

in Fig. 11, which is the typical phenomenon of chaos.

E. Generated trajectories in x-y-z space

Fig. 12(a)∼(d) show trajectories generated at $q_{11} = -0.2800, -0.1084, 0.1744, 0.4800$ respectively. (a) shows a period trajectory, (b) shows chaos trajectory and (c),(d) show three periods trajectory. When q_{11} is in a interval which has infinite periods, chaos trajectory is generated. q_{11} of chaos 03 and chaos 04 are respectively $0.829098955, -0.108415351$, which are in the interval. It is confirmed that by changing weight coefficients of N.N. trajectory period changes and chaos and non chaos trajectories are appeared.

IX. ADAPTENCE CHAOS TO ROBOT

The chaos made by NNDE is actually adapted to the robot dynamics. Fig.13 shows the results of visual servoing and chaos experiments. In comparison with visual servoing (FLS= $-$ 0.30), it is said that chaos trajectory (FLS= $-$ 0.08) reduced the learning speed of fish.

Fig.14 and 15 show fish positions recognized by the robot in Visual Servoing and chaos experiments. Fig.14 (a)(b) show fish escaped from the net in circular swimming and Fig.14(c) shows fish gradually stayed at corner to escape from the net. In the case of chaos, Fig.15 shows fish didn ' t tend to change the oneself movement like staying at corners as time goes by. It indicates that the robot tried to drive fish from the corners by chaos and the fish couldn ' t take escaping strategy that fish stay at corners. From these results, chaos has possibility of catching fish taking the escaping strategy.

X. CONCLUSION

In this paper, we propose a new chaos generating system using N.N., named as "NNDE" for Fish-Catching Robot. This method is confirmed that can generate plural chaos. Also, the characters of plural chaos generated by this method has been examined on the following points; Lyapunov number, Poincare return map, sensitivity to initial value, fractal dimension and bifurcation diagram.

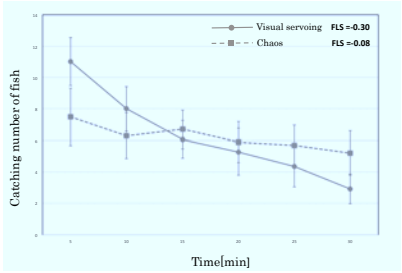


Fig. 13. Catching number of fish

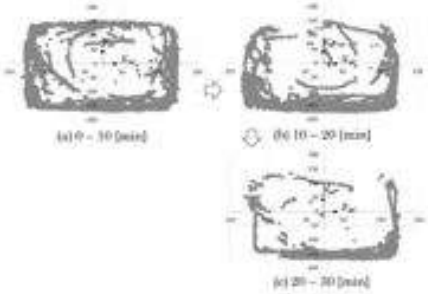


Fig. 14. Positions of fish which robot recognized using visual servoing

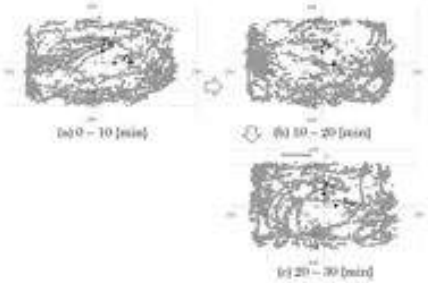


Fig. 15. Positions of fish which robot recognized using chaos

REFERENCES

[1] M. Bohlen: "A robot in a cage-exploring interactions between animals and robots", CIRA. 1999, pp.214-219.

[2] Hee-Jun Park, Byung Kook Kim, Kye Young Lim: "measuring the machine intelligence quotient (MIQ) of human-machine cooperative systems", IEEE Trans. vol.31, 2001, pp.89-96.

[3] R. Kelly: "Robust Asymptotically Stable Visual Servoing of Planar Robots", IEEE Trans. Robot. Automat., vol.12, no.5, 1996, pp.759-766.

[4] P.Y. Oh, and P.K. Allen: "Visual servoing by partitioning degrees of freedom", IEEE Trans. Robot. Automat., vol.17, pp.1-17, Feb.2001

[5] M. Minami, H. Suzuki, J. Agbanhan, T. Asakura: "Visual Servoing to Fish and Catching Using Global/Local GA Search", Int. Conf. on Advanced Intelligent Mechatronics, Proc., 2001, pp.183-188.

[6] M. Minami, J. Agubanhan, and T. Asakura: "Manipulator Visual Servoing and Tracking of Fish using Genetic Algorithm", Int. J. of Industrial Robot, Vol.29, No.4, 1999, pp.278-289.

[7] Visual Servoing to catch fish Using Global/local GA Search Hidekazu Suzuki, Mamoru Minami IEEE/ASME Transactions on Mechatronics, Vol.10, Issue 3, 352-357 (2005.6)

[8] K. Aihara: "Chaos in Neural System", pp.126-151, 1993 (in Japanese).

[9] R. FitzHugh: "Impulses and physiological states in theoretical models of nerve membrane", Biophy.J.1, pp.445-466 (1961).

[10] Jun Hirao and Mamoru Minami, "Intelligence Comparison between Fish and Robot using Chaos and Random", Proceedings of the 2008 IEEE/ASME international Conference on Advanced Intelligent Mechatronics July 2 - 5, 2008, Xi ' an, China, pp552-557.

[11] C. T. Lin and C. S. Lee, "Neural Fuzzy Systems", *Englewood Cliffs*, NJ:Prentice Hall PTR, 1996.

[12] Limin Peng and Peng-Yung Woo, "Neural-Fuzzy Control System for Robotic Manipulators", *IEEE Control Systems Magazine*, 2002, pp.53-63.

[13] K. Funahashi : On the Approximate Realization of Continuous Mappings by Neural Networks, Neural Networks, **2**, 183/191 (1989)

[14] G. Cybenko : Approximation by superpositions of a sigmoidal function, Math. Control, Signals, and Systems, **2**, 303/314 (1989)

[15] K. Hornik, M. Stinchcombe and H. White : Multilayer feedforward networks are universal approximators, Neural Networks, **2**, 359/366 (1989)