Hybrid Neural Network and Video Texture Techniques to Synthesize Realistic Fish Animation

Tong-Yee Lee[*] Hon-I Chen Ping-Hsien Lin
tonylee@mail.ncku.edu.tw *chichi@csie.ncku.edu.tw* *hsien@vision.csie.ncku.edu.tw*

Department of Computer Science and Information Engineering
National Cheng-Kung University, ROC.

* Corresponding author

Abstract

Computer Graphics technology has advanced dramatically in recent years. However, all these techniques do not obviously improve the reality of rendering in computer animation. Recently, a new type of medium called video textures is proposed. That is a new method to synthesize video-based animation and also raising many new methodologies to verify animation. We present several algorithms that not only enrich the original methods but also utilize sufficiently the information obtained from video.

In this paper, a novel method to implement a realistic fish animation is proposed. First, we capture a video clip and get the fish information by using several image-processing techniques. Then, in order to find the relation of the extracted properties, some specific parameters of all video frames are trained through neural network. Because of the lack of locomotion images in all orientation, we synthesize new fish images by rotating the images. So far, a simply, robust, and real-time animation playing system is available. On the other hand, we analyze the similarity of every frame and take video loops to generate synthetic animation. Moreover, we use a position map for video loops so as to select the proper video segment. With this map, we can create another type computer animation. Eventually, for eliminating some unavoidable constraints while playing fish animation, both two rendering method are combined.

Keywords: video texture, video-based animation, neural network.

1. Introduction

In early days computer animation appeared as successive 2D images such as cartoon. Presently, researchers in computer graphics and computer vision have proposed efficient methods to generate novel views by analyzing captured images. These techniques, called image-based rendering, demand somewhat user interaction [3] and permit photo-realistic synthesis of static scenes.

And yet, 3D animation industries have occupied the commercial entertainment market in recently years and offer more variation in visualization. However, no matter the computer animation is improved by researchers, it's still hard to reach an absolutely realistic object in all cases. In spite of texture mapping could fairly increase the reality in modeling or rendering, the object is still unnatural in most situation. Our vision is to create an interesting and natural animation.

We propose an alternate methodology to realize natural and realistic fish animation. Nevertheless, what on earth does nature mean? Most people will agree that the fish pictures show the immediate and most primitive appearance. If we display fish animation on 2D screen, the fish's texture in every moment is expected smoothly in visual. In fact, for general cameras, it is quite difficult to capture every detailed motion of fish. Intuitively, we can easily solve the problem through video camera. And further, video clips can be put down more information than static pictures, so instead of pictures we choose the video frames as our data source.

In detail, our main goal is to take advantage of video sufficiently and as much as possible grasping more information of specific objects within each frame. In our experiment, we put our video camera on the upper side of fish tank and capture a carp which will change its body obviously whilst turning. It's an interesting topic to study what video conveys through those consecutive video images. Because of images recorded in video successively, the varying motion of fish shows distinctly within every frame. All these images have closely relationship with each other, such as object positions, color similarity. So, when we observe a fish, it is easy to discover the fish almost swims in some similar trajectory, regular behavior, and resembling reflection. By exploiting these properties of video frames, we study the idiosyncrasies of a fish from video. After abstracting these video frames, it is easy to obtain many parameters about fish movement such as displacement, turn degree, head orientation etc. Further, in order to understand the variation of fish locomotion, all the parameters can be trained through the back-propagation neural network [4]. From extracted parameters we successfully explore a relation in mathematics that could be used in fish control system. That is, by giving reasonable control parameters, the control system will return the proper reaction of fish. On the other hand, because of the lack of locomotion images for all orientation, we synthesize new fish images by rotating the images. So far, a simply, robust, and real-time animation playing system is available. And more significant, the control system is derived by every frame of the video. In other words, this control system represents truly the fish behaviors that recorded in video.

In [1], Schödl et al. proposes a new type of medium named video textures, which provide a continuous, infinitely varying stream of video images. The basic concept of a video texture can be extended in several different ways to further increase its applicability. Among the techniques they proposed, the most important concept is video loops that can be extracted from video and played continuously without any visible discontinuities. However, this technique is not suitable anymore when the specific object moves arbitrarily on 2D screen. We solve this problem by introducing a table, which is called video loop position map. This table is recorded the orientation of each video loop and wherever the fish swimming, by this table, we can easily determine proper video loops which have identical orientation with fish. Eventually, in case the video is too short to be extracted enough video loops for rendering or the selected video loops exist some unavoidable limitation, we jump into another playing system that simulated by neural network to figure out the problem.

We successfully keep the characteristics of fish captured by video camera, and furthermore by embedding the original video loops the fish animation also would display realistically. However, creating the fish animation from video must overcome a lot of problems. The first difficulty is to decide the size of fish. If the fish is small, then the movement would be unclear and the fish body of most frames would become ambiguity. On the other hands, since the fish changes its body in a limit range, the extraction would be easy to observe and implement. If we employ other kind animal as our target, such as snake or eel, which has a complex body and change exceedingly while moves, it’s not easy to evaluate the variation from video. For the sake of convenience, we select fish to prove our methodology can work in some case. That is, our method is not suitable in generalization. Second, although carp swims close to the bottom of fish tank, it will tilt when swimming toward top. To capture an image, which contains a leaning fish, however, is hard to analyze motion parameters since it gives rise to ambiguity in motion characteristic. Third, object tracking is a tough problem in computer vision field. There is a great store of techniques for extracting moving objects from imagery. In order to simplify this problem, we adopt background subtraction method to extract the fish.

Our main contributions are listed as follows:

1 Through our method, the object within the video can be automatically analyzed and parameterized. Further, we can utilize neural network to learn the fish motion model.
2 Through video images, we can measure the fish motion by several parameters, such as displacement, change of body orientation, swimming state, etc.
3 The training of neural network could be done off-line. Animation from trained motion model, however, can generate real-time.
4 We can utilize the position map for video loops to select suitable video loop to synthesize new animation.
5 Combining neural network mode and video textures mode to generate a complete animation playing system.

1.1 Related Work

As Schödl et al. mention in [1], there has been little previous work on automatically generating motion by reusing captured video. Fortunately, they propose video sprites [2], which is another type of video textures, to make this field richer. In [1], they generalize image-based rendering to the temporal domain. Therefore, the core algorithms proposed in video textures are not suitable on specific moving objects. In order to overcome this problem, the video sprites that can be rendered anywhere on the screen to create a novel animation is presented in [2]. However, they labeled training data manually so as to train the classifier. In most cases, manual classification is difficult to evaluate. Evaluating a method, which is required manual operation, is not an easy work. For this reason, we devote to develop an automatically learning method that could objectively represent the characteristic of fish motion

Frankly, in addition to [1, 2], there is really a little work that has highly relation to our research. In this paragraph, we discuss some research that somewhat inspire our study. First, Tu et al. [5] propose a framework for animation with minimal input from the animator. They presented a method to make an artificial creature move by defining principal motion patterns of fishes and giving parameters of mental status, which are factors to cause the motion. They define a physics-based, virtual marine world, in which artificial fishes inhabit. These motion patterns are decided by using a few mental state variables. Each of which, however, remains some problems in ease to use and generality. In addition, a method to learn a fish's motion automatically is proposed [6]. Because these methods use complicated algorithm, however, it is difficult to generate motions in real time. Terzopoulos et al. [7] proposed a new method called NeuralAnimator that utilizes a neural network to simulate a physics-based model and they replaced the conventional animation simulators, which are made with numerical simulation.

Both Ziv Bar-Joseph [8] and X. Wen et al. [9] introduced wavelet techniques in analyzing and synthesizing video. Bar-Joseph retrieved the video by the decomposed parameters of video. Wen made a new synthetic video by wavelet coefficients. Finkelstein et al. [10] presented a new approach called multiresolution that provides a means of capturing time-varying image data produced at multiple scales, both spatially and temporally. The above studies of video are successfully analyzing the information of video and really encourage us to exploit the video information.

We also inspired by Takahashi et al. [11] and Lipton [12] that they both focus on the object of video and study the behavior and feature. Takahashi et al. proposed an estimation method of fish position and posture from video using object matching technique. Lipton presented a paradigm for data interaction called virtual video that can be interactively augmented in real-time.

1.2 System Overview

Our purpose is to generate a realistic fish animation by understanding and taking advantage of video clip. In our system, we separate our work into two major parts, one is neural network playing mode, the other is video loop playing mode (Figure 1).

The remainder of this paper describes, in Section 2, the representation used to obtain the parameters of a video image (frame). In Section 3, after observing the fish motion, we define several evident characteristics as motion parameters and train a motion model by neural network. In section 4, the method to detect video loops is proposed. And we discuss the some limitations when using video loops about moving 2D object. On the other hand, we also depict the solution of these limitations, that is, by switching into neural network playing mode. Our results are illustrated in Section 5 and conclusion or further extensions is put in Section 7.

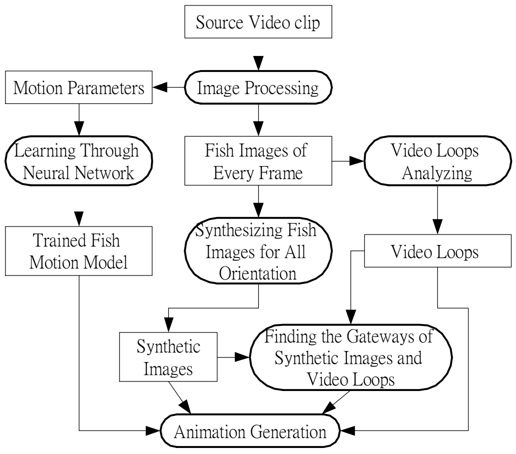

Source Video clip
Motion Parameters
Image Processing
Learning Through Neural Network
Fish Images of Every Frame
Video Loops Analyzing
Trained Fish Motion Model
Synthesizing Fish Images for All Orientation
Video Loops
Synthetic Images
Finding the Gateways of Synthetic Images and Video Loops
Animation Generation

Figure 1. *System overview diagram*. An input source video clip is decomposed into every single frame. We abstract fish information from these images and also, motion parameters could be obtained after image processing. Then, we utilize neural network to learn motion parameters in order to find a model for fish motion. On the other hand, through these frames, we not only synthesize fish images for all orientation but also analyze their similarity to find out video loops. Eventually, the gateways of synthetic images and source video clip, which are obtained by examining the resemblance, are used for the switching between the two kinds of playing modes.

2. Motion Parameters Extraction

The first step of our research is to examine the source video. After segmenting every fish image of video frames, we begin to analyze the motion information of fish so as to train through neural network. In this section, the method of segmentation and motion measurement will be discussed.

2.1 Displacement

As regards the image segmentation, there are a lot of methods for object extracting [13]. In our case, a fish in a simply fish tank, we just choose background subtraction to extract the fish image because it works well in general cases except when fish swims near to the walls of fish tank. Accordingly, we locate the fish global coordinate by computing the mass center of fish. Further, the displacement of fish could be derived by fish coordinate. That is an important characteristic of fish since it varies only when the fish changes its motion. In addition to above parameters, we can also locate head and tail of a fish by image-processing techniques. All these properties are very practical in learning by neural network while we can analyze them and take out their influence upon fish motion.

2.2 Motion Area

A carp is employed as our target fish for the reason of its body size and habit. The swimming fashion of carp belongs to the horse mackerel type. In this swimming fashion, the fish deforms extremely when it turns around. Therefore, we define a measurement to quantify the variation of fish motion. This measurement is called motion area.

First of all, we use image-thinning method [13] on the original fish images. After thinning processing, the fish images leave a thin line, which is like the backbone of fish. Second, we put four points on 1/6、1/3、1/2、5/6 of this thin line, one on the fish head and one on the fish tail respectively. (See Figure 2) Then, we produce a B-spline that is made up by these points, which are taken as control points. Eventually, the area circled by the B-spline is the motion area.

Why not put these points into six equal positions? In our observation, fish head is composed of bone and head its length is almost one-third of itself. Moreover, it has a backbone composed of a number of spines. Thus, thee head cannot deform, but the rear part of the body deforms smoothly. For this reason, it is sensible that the measurement of the points, which are distributed close to the fish head, could receive higher accuracy than that of the points are locating in the tail. Although we do not distinctly prove this hypothesis in mathematical form, the results determined by our method indeed represent the variation objectively.

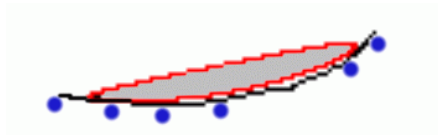
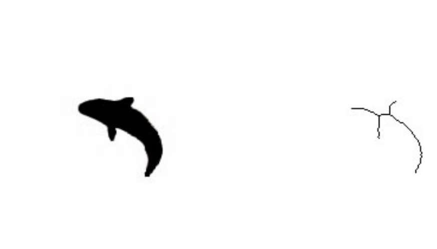


Figure 2. *Motion Area.* There are six points setting purposely into different place along the skeleton. Also, these points are the control points of a B-spline circle, which is the motion area and shows the magnitude of the deformation of fish objectively.

2.2.1 Branch Pruning

Any object of image will only leave the medial axes of itself after dealing with image-thinning procedure [13]. In our experiment, however, if the fish unfolds its pectoral fin, the final image will show several branches (Figure 3). This consequence will increase the difficulty while distinguish the backbone.



(a) (b)

Figure 3. (a) is the original fish image with stretching pectoral fin. After image-thinning process, some branches appears near the pectoral fin, which is showing in (b).

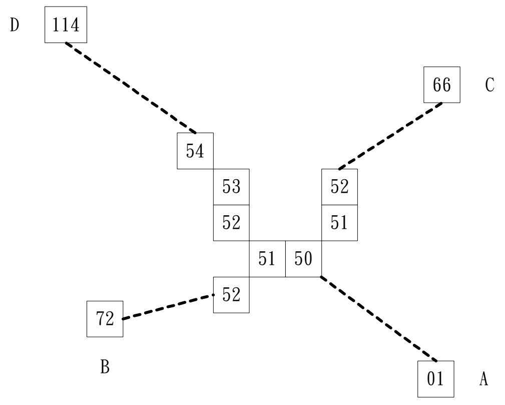


Figure 4. Among these branches, only the greatest number will be selected as backbone of fish. It is to say, in this figure, the line from A to D is the backbone.

As far as branch problem is concerned, we solve it by a simple method. First, we take down a number as the initial in the beginning of backbone and give another increasing number along this backbone. Once this process encounters the branch while going along the backbone, each branch of this joint would be put an increasing and identical number. The process will keep executing in respective branch until the end. Eventually, each branch will receive a number. Among these numbers, the greatest number will be picked out and the fish backbone is defined from beginning of branch till the end of this branch. For example, in Figure 4, the process starts at point A and give an initial number 0 and it is increasing along the backbone. When the number is 50, the process meets a branch and put 51 into each joint branch. Then, the process will execute respectively until the end of branch. Finally, the line from A to D will be taken as the backbone because the largest number is in D.

2.3 Included Angle of Orientation

In Figure 2, we show that the motion area can represent the deformation of fish body. Furthermore, if the fish is turning, that is, its shape will change from almost straight into twist, not only the posture but also the orientation of fish will vary gradually. In Figure 5, there are three images (5a, 5b, 5c) captured in different time. Through their motion area images (5a', 5b', 5c'), it is easy to find the variation of fish motion while the motion area and head orientation keeps changing. The included angle between current head orientation and previous one could also somewhat represent the magnitude of twist while fish is turning. In our case, we take 5 frames as one time interval and this measurement is desirable while the larger or less time interval will obtain unclear result. The relation of this angle and fish motion will be discussed in the next Section.

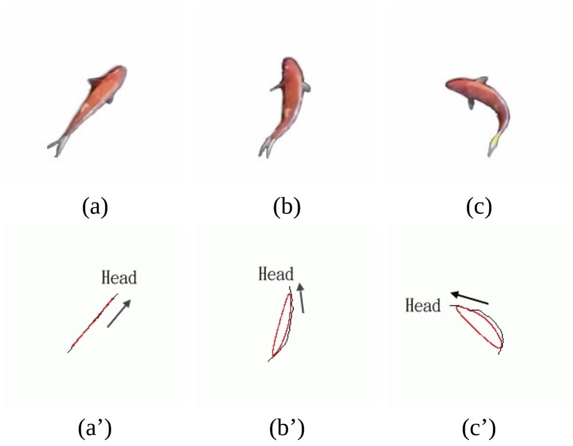


Figure 5. Here are 6 pictures. The above 3 pictures (a), (b), (c) are taken at different time t_0, t_0+k, t_0+2k and the below 3 (a'), (b'), (c') ones show the measurement of motion area, which is circled bye the red line, respectively. Obviously, if the fish is turning, then the corresponding motion area is larger then it is swimming straight.

3. Neural Network Playing Mode

So far, we have exploited several parameters about fish motion from the captured video. One of our major missions is to find out the motion model by employing the relation of these parameters. For this reason, we adopt neural network to analyze these parameters. In this section, the learning method and the method to apply the trained motion model to generate animation will be presented. Eventually, we regard the trained model as an "artificial fish". This trained model will return the proper fish state for given control parameters. Then, there are some unavoidable problems while rendering. We solve this problem by synthesizing the fish images. At last, the kind of animation generated by this trained model is called playing in neural network mode.

3.1 Learning through Neural Network

Originally, we suppose given the fish states and control value, then the feasible reaction could be obtained. Hence, the input parameters are separated as two classes: state inputs and control inputs. State inputs are made up by the current swimming type and motion area. As regard to control input, however, physically there is no any control force on the fish because it is almost impossible to control the fish swim fashion. Therefore, we use an alternate way to decide the control inputs. At first, the value, which is the difference of current state and previous state, is counted as the control vector. In Equation (1), for example, u and s stands for control vector and state vector respectively. As described above, δ is the time interval of sampling period.

$$u_t = s_{t+\delta} - s_t \qquad (1)$$

In Figure 6, the architecture of neural network is illustrated in detail. We choose the back-propagation algorithm in our experiment. And there are three layers in our network. Back-propagation refers to the practical, recursive method to calculate the component error derivatives of the gradient term. Applying the chain rule of differentiation, the back-propagation algorithm first computes the derivatives with respect to weights in the output layer and chains its way back to the input layer, computing the derivatives with respect to weights in each hidden layer as it proceeds. In this paper, the more detail mathematical principle of neural network, such as general delta rule or some other optimization methods etc., will not be mentioned. The more information of that can be obtained in [4].

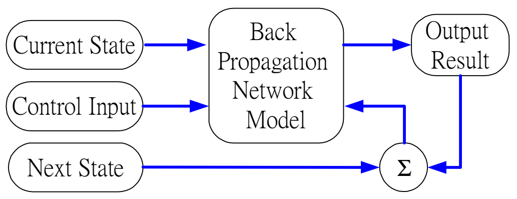


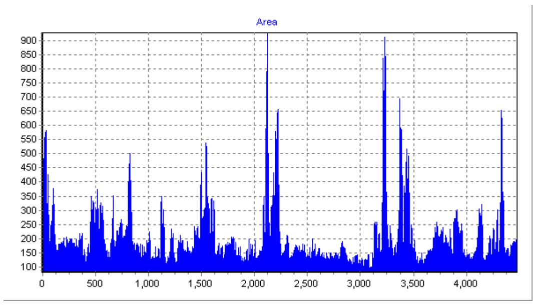
Figure 6. *Neural Network Architectur*e. Our network architecture is a Back-Propagation Network. It is a 3-layer with 4 input units, 20 hidden units, 2 output units.

3.2 Selecting Parameters of Motion Model

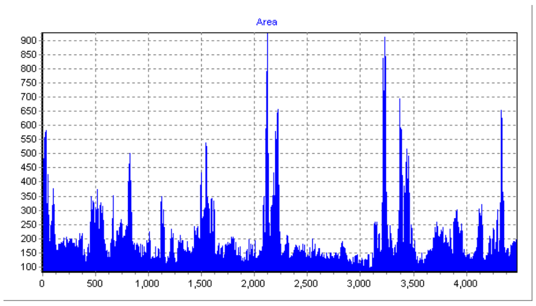
Training the motion model through the input video clip is one innovative approach in computer animation. However, how to make it?

In addition to the mentioned current states and control inputs that both could be obtained from video clip, the desired output is given from the next state of fish, which also can be received from video clip. In this way, the trained model will truly reflect the characteristic of fish motion that only appeared in the video.

Parameters selection is quite important for the training model of neural network. The training model will converge quickly if the selected parameters really represent the feature of all training data. In our experiment, the parameters we choose are described in Section 2. The variation of all these parameters can objectively correspond to the fish motion and we show that in Figure 7. The total time of our input clip is about 3 minutes. And there are 4480 frames in source video clip. Figure 7a shows the motion area of these frames. Figure 7b shows the swimming type of fish. The remains parameters also show similarity with motion area. Furthermore, in the period from 3150 to 3200, the fish turns right and the corresponding motion area is much higher than nearby ones that fish is swimming straight. On the contrary, in the period from 1600 to 2050, the fish swims straight and its mean motion area is small.



(a)



(d)

Figure 7. (a) The Motion Area of the input video clip. The y-axis is the magnitude of motion area and x-axis is the frame no. (b) The Swimming Type of Fish. The y-axis, which is the swimming type of fish, means straight swimming while y is 0; turning left while y is 1; turning right while y is –1.

Through these parameters, which are highly relevant to fish motion, it is sensible to get an accurate trained motion model. We choose the swimming type of fish and the motion area as the state parameters (See Equation (2)); the included angle of head orientation and the displacement as the control parameters (See Equation (3)). The training model is designed as Equation (4) with State Inputs and Control Inputs. In fact, the convergence of training process, which will be illustrated at Section 5, shows that not only this framework is practicable but also it can work well.

$$\mathrm{State}_t = \begin{bmatrix} \mathrm{Area}_t \\ \mathrm{Type}_t \end{bmatrix} \tag{2}$$

$$\mathrm{Control}_t = \begin{bmatrix} \mathrm{Displacement}_t \\ \mathrm{Angle}_t \end{bmatrix} = \begin{bmatrix} \left\| \mathrm{Position}_{t+\epsilon} - \mathrm{Position}_t \right\| \\ \mathrm{Included\,Angle}(\mathrm{Dir}_{t+\epsilon}, \mathrm{Dir}_t) \end{bmatrix} \tag{3}$$

$$\mathrm{State}_{t+\epsilon} = \mathrm{BPN\_Model}(\mathrm{State}_t, \mathrm{Control}_t) \tag{4}$$

3.3 Synthesizing Fish Images

We expect to obtain a proper state output by giving reasonable control parameters. And, since all the parameters are observable features in visual, this can make simpler on control function. Up to now, a simple motor system can be made by these definitely motion parameters. Afterwards, we have to choose a feasible fish images for rendering. Here comes a question: where are these feasible fish images?

In 2D plane, we define 24 orientations to coordinate fish images. By our observation, however, usually there are just a little fish images on some orientations although there are many fish images on the other orientations. It is to say, if we choose fish images from all the video images, then the fish images must be insufficient on some orientations because the fish indeed does not appear frequently on every orientation. In addition, the fish only swim straight or never turn right/left on some specific orientation. For these reason, if we would like to select fish images from the original video images for rendering, then it must be difficult to draw smoothly because of the lack of motion on every orientation (See Table 1). Of course, if we keep capturing the fish for more long time, the motion information of fish will be more enough on every orientation. However, this hypothesis is not practical, because expecting the fish showing various motions on every orientation does not make sense at all and more significant, the total time for capturing enough information of fish is unpredictable.

In order to solve this problem, we figure out an alternate way. First, we manually pick up successive fish images, which recorded the straight swim or it appears the most deformation on turning left/right. Then, we synthesize more images on every orientation by rotating these selected images so as to fulfill the fish motion. Until now, a new realistic fish animation can be generated since all the motion can be displayed by utilizing these synthetic fish images. Although we select successive fish images manually, it is quite efficient and worth to do because the amount of synthetic images is only 1.33% of the total video frames.

Table 1

Orient.	Straight (%)	Right (%)	Max Area of Right	Left (%)	Max Area of Left	Amount (%)
1	46.25	18.75	0.556	35	0.709	***1.78***
2	79.47	3.17	0.761	17.36	0.703	***4.24***
3	70.06	4.81	0.708	25.15	0.502	***3.72***
4	79.41	8.87	0.357	11.76	0.539	***3.79***
5	72.03	22.38	0.343	5.68	0.346	***4.7***
6	88	12.03	0.407	0	0	***7.25***
7	94.93	5.07	0.336	0	0	***6.6***
8	94.94	5.06	0.701	0	0	***12.81***
9	94.88	4.7	0.961	0.42	0.357	***10.46***
10	89.94	7.98	1	2.08	0.375	***11.76***
11	82.1	12.69	0.907	5.26	0.391	***4.24***
12	94.6	3.21	0.614	2.74	0.608	***4.88***
13	94.46	3.84	0.829	1.7	0.677	***5.24***
14	74.64	20	0.917	5.63	0.707	***1.58***
15	76.81	19.11	0.866	4.34	0.964	***1.54***
16	84.82	13.51	0.562	1.78	1	***2.49***
17	82.05	12.98	0.274	5.12	0.95	***1.74***
18	79.07	17.64	0.429	3.48	0.482	***1.91***
19	84.54	13.76	0.54	1.81	0.307	***2.45***
20	71.25	13.92	0.545	15	0.403	***1.78***
21	28	24.49	0.636	48	0.468	***1.11***
22	25	28.81	0.501	46.66	0.519	***1.33***
23	3.57	40.74	0.611	57.14	0.625	***0.62***
24	44.57	10.97	0.641	44.57	0.551	***1.85***

The first column is the orientation number. The Second, third, fifth column is the rate of swimming type, which is straight swimming, turning right, turning left respectively in that orientation. The fourth is the rate of the max turning-right motion area in that orientation corresponding to the max turning-right motion area of the video clip. The sixth column is similar to the fourth column except it records the turning-left information. And the last column is the rate of the fish image amount in that orientation relative to the total video clip.

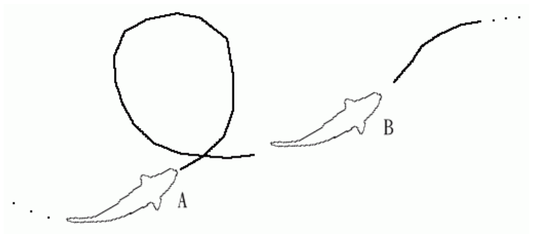


Figure 8. *Video Loop.* The fish image at A is quietly like that at B. That is, we can put a fish image, which is the fish image at A, after B and it will be seamlessly in visual.

4. Video Loop Playing Mode

Considering a loop with a single transition i →j, from source frame i to destination frame j, which we call a video loop if frame i is so similar to frame j that the fish motion in frame j can jump to the motion in frame i. And the more important, this progress will be look like seamlessly. Because the fish swims in a regular pattern, there is great possibility of extracting many video loops from video clip. In this section, the analysis method to video loops and how to make use of these video loops to synthesize a novel animation will be proposed. However, there still exist several problems while using these video loops, and also, we will describe the solutions.

4.1 Finding Video Loops

The first step to find video loops is to detect two fish images that are the most similar to each other and nonconsecutive in time domain. Basically, in order to select the video loops easily, we make a distance map (or difference map) of every two images of video clip can be mathematically defined as follows:

$$D_{ij} = \left\| F_i - F_j \right\|_2 \qquad (5)$$

where F_i, F_j is the frame number and D_{ij} is the magnitude of L_2 distance. (Schödl [1])

Because there are several special properties of fish, one can improve Equation (5) in order to accelerate the computing time through these properties, such as head orientation, swimming type. Actually, we do not calculate the distance of frame i and frame j if the fish properties are different to each other. We show this trick method in Equation (6) and its illustration in Equation (6-1) and Equation (6-2), respectively. Before counting Equation (6), we first calculate the value of Equation (6-1) and Equation (6-2). Once the return value is not equal to 1, than we abort this pair immediately. Because the computing time of D_{ij} is more than computing Equation (6-1) and Equation (6-2), building a distance map by using Equation (6) can save beyond 90% time than only using Equation (5). This result is shown in Figure 9.

$$\begin{aligned} D'_{ij} &= State(i,j) \bullet D_{ij} \\ &= (OH(i,j) \otimes ST(i,j) \bullet MA(i,j)) \bullet D_{ij} \qquad (6) \\ &\textit{where} \bullet \textit{is scalar } \text{multiplication} \end{aligned}$$

$$HD(i,j) \otimes ST(i,j) = \begin{cases} MAX & \text{if } OH(i) \neq OH(j) \text{ or } ST(i) \neq ST(j) \\ 1 & \text{if } OH(i) = OH(j) \text{ and } ST(i) = ST(j) \end{cases} \qquad (7)$$

$$MA(i,j) = \begin{cases} MAX & \left| MA(i) - MA(j) \right| > \text{Threshold} \\ 1 & \text{otherwise} \end{cases} \qquad (8)$$

In above equations, HD stands for "Orientation of Head", ST stands for "Swimming Type", MA(i) represents the motion area of fish in frame i, and MA stands for "Motion Area".

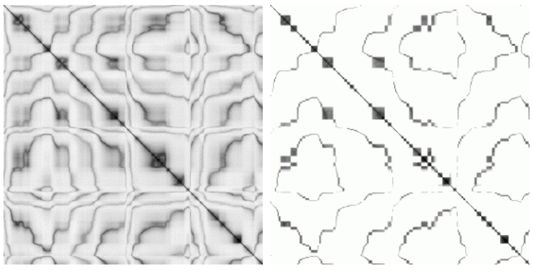
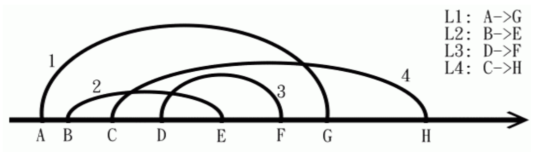

(a) (b)

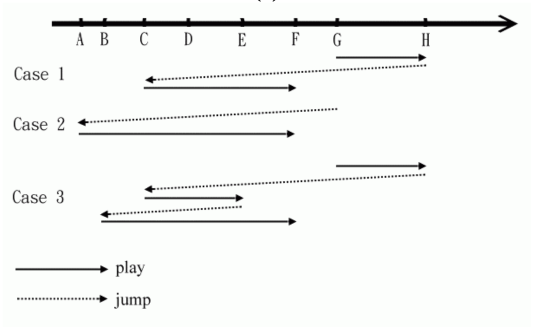
Figure 9. *Distance Map.* (a) is drawn by applying Equation (5), that is , the distance of every two images will be calculated whether the fish properties are similar or not. (b) is drawn by only applying Equation (6). Both of this two maps, the darker area means smaller distance or more similarity of the two fish images. On the contrary, the brighter means more dissimilar to each other.

4.2 Video Loop Position Table

After selecting feasible video loops, we can create a synthetic fish animation by reusing or compounding these video loops. Figure 10 shows three possible ways from G to F.



(a)

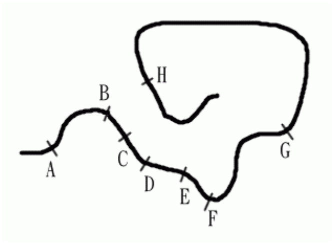


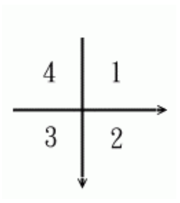
(b)

Figure 10. *The Relation of Video Loops in Time Domain.* In Figure (a), there are 4 video loops, L1, L2, L3, L4. If we are generating a synthetic animation and playing a transition from G to F, there are many ways to accomplish this operation. We show three possible ways to make it in Figure (b). The policy of case 1 is GHCDEF, and case 2 is GABCDEF, and case 3 is GHCDEBCDEF. The black line means play fish images successively and dotted line means jumping to other fish images.

However, the fish is moving on a limitary 2D plan instead of an infinite area. In Figure 11a, for example, it shows the fish trajectory and the beginning, end of each loop. Obviously, each case in Figure 10b will lead the fish swim to totally different place on 2D plane. If generating the animation only by understanding the relation of video loops in time domain, we never know where the fish will swim. In other words, making a relation table about the video loops in 2D space is necessary for the fish animation.

In order to manage more information of video loops, we design a table that is recorded the quadrant, which is measured by the later fish position of the video images relative to the former one. The quadrant is defined in Figure 11b and the table is Figure 11c. With this table, if the current fish image is in F, the end of Loop 3 (L3), and it should swim toward first quadrant, we know that it can keep playing the fish images from F to G because the fish position of G is in the first quadrant of F.





(a) (b)

	A	B	C	D	E	F	G	H
A		1	2	2	2	2	1	1
B			2	2	2	2	2	1
C				2	2	2	1	1
D					2	2	1	1
E						2	1	4
F		NA					1	4
G								4
H								

(c)

Figure 11. *Video Loop Position Table.* Figure (a) shows the trajectory of fish with these video loops of Figure 10a. Figure (b) is the defined 4 quadrants. Figure (c) is the *Video Loop Position Table* where the number of the cell represents the quadrant that is measured by the later fish position of the video images relative to the former one. For example, the value of (A, B) is 1, that is, B is at the first quadrant of A.

The Video Loop Position Table helps us to determine the proper video loops. However, there still exist some troublesome problem and we cannot solve it by this table. In Figure 11, if the current fish should swim toward third quadrant, then it can never find a video loop that bring the fish toward the goal because none of the video loops are at third quadrant to other loop.

Furthermore, In Figure 12, for example, there are 6 video loops in this video clip. The former 3 loops of this video clip overlap with each other and the later 3 loops do, too. From this figure, we know that these video loops can be separated into two groups that video loops of different group do not overlap to each other. This situation is highly possible because the characteristic of fish swimming fashion is regular and unpredictable. In short, once the source video clip exists such problem, it will result in a monotonous animation if we play video loops in the later area because the motion may not various in this region.

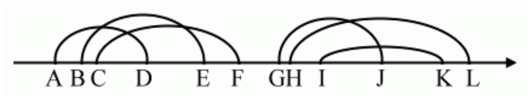


Figure 12. In time domain, the video loops is divided into 2 regions because the fish does not swim normally or the fish motion does not appear uniformly.

Moreover, if the video loop will lead the fish to some other orientation that is entirely opposite to current orientation (See Figure 13), then using this loop to generate the animation will be odd.

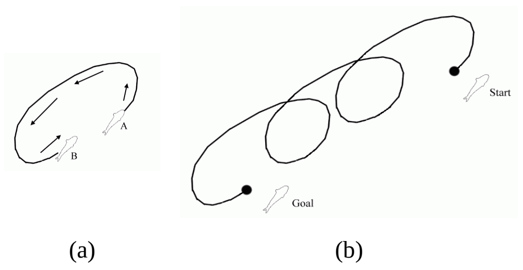


Figure 13. *Improper Video Loops.* If the fish head orientation is different from the orientation of the video loop, then the fish animation is quite unnatural by compounding this kind of video loop continuously.

4.3 Switching to Neural Network Playing Mode

All the problems mentioned at Section 4.2 could never be improved if we insist on generating the animation only by video loops. For this reason, we abandon playing in video loops mode once these problems occur. Instead, we introduce the neural network playing mode, which is defined in section 3, to improve this problem. We use “improve” instead of “solve” because it still possible that neural network playing model can not work. However, in our experiment, it can work well in most of time.

In order to switch the two different playing modes smoothly, the first step is to find the “gateways” of these two playing modes. When playing in neural network mode, it depends on the synthetic fish images. And playing in video loops requires video loops. That is, it requires source video images when playing in video loops. Intuitively, the gateways should be the fish images that are very similar to both data source. Although the synthetic fish image may not be identical to the video images, we can interpolate the both two images so as to play smoothly in visual when changing current playing mode to another mode.

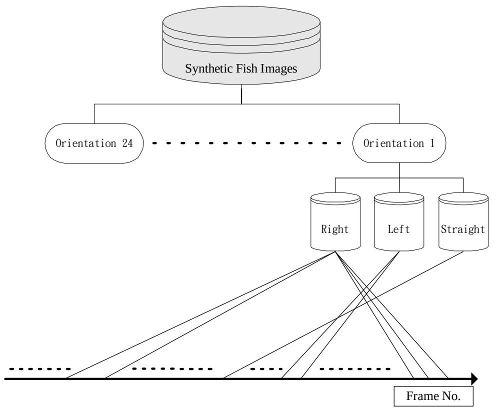


Figure 14. In order to find the gateway, the synthetic fish images of every orientation will be compared with the source video images to find the similar images.

5. Experiment Results

In this section, we will show several diagrams and discuss some important result of each major process.

5.1 Computing Time

As described before, there are 4480 frames in the source video clip. In Table 2, we will list the computing time of these major steps, which are mentioned in Figure 1. All the processed listed in Table 2 can be accomplished automatically without any user intervention.

Table 2

	Process	**Computing Time**
1	Extracting Fish Images from Video Clip	Several Hours
2	Build a Motion Model by Learning Through Neural Network	About 70 Seconds (200 epoch)
3	Video Loops Analyzing	About 30 minutes
4	Synthesizing Fish Images for All Orientation	About 15 minutes
5	Finding the Gateways of Synthetic Images and Video Loops	Several Hours
6	Animation Generation (Hybrid Playing Mode)	Real-Time

5.1.1 Learning through Neural Network

After extracting the fish motion information from video, we train the motion model through BPN algorithm. The network architecture is 3-layer network with 4 inputs, 20 hidden units, 2 outputs. In addition, we set the momentum value as 0.5, learning rate as 0.5. In Figure 15, the error rate of each epoch is showed in Figure 15.

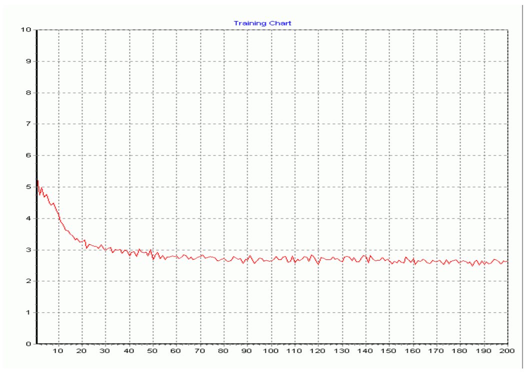


Figure 15. X-axis is the epoch number. Y-axis is the error amount of motion area prediction. In this experiment, (both learning rate and momentum are 0.5), after about 20 epoch, the convergence is prone to stationary gradually. We do not pay attention to study the criterion of terminating of training process. For more advanced knowledge about neural network could be obtained in [4].

5.2 Swimming through Specific Goal

We can select the playing mode not only the neural network playing mode (See Figure 16) or video loops mode but also hybrid this two modes (See Figure 17). For more experimental results and animated sequence, please visit our website at *http://couger.csie.ncku.edu.tw/vr/fish.html* .

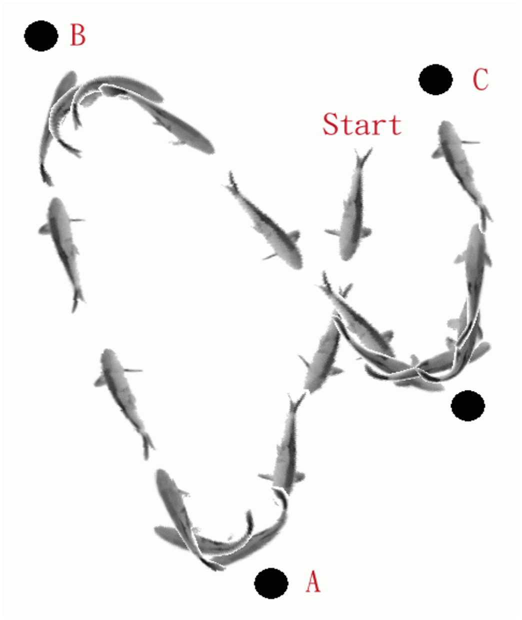


Figure 16. In this figure, we show that the fish can travel the specific goal (A, B, C) by only applying the trained motion model. As regards to the playing method will be illustrated at Appendix Section.

6. Conclusions

In this paper, a novel method to implement a realistic fish animation is realized. We propose two realistic fish playing modes, one is neural network playing mode, the other is video loop playing mode. And more significant, in order to eliminate some unavoidable constraints whilst playing fish animation, both two rendering method are combined. The motion model of neural network playing mode is trained by the fish motion parameters which are extracted from video. That is, this motion model is trained by itself. To our knowledge, there is no similar research on computer animation. On the other hands, we proposed video loop position table that greatly helps determine the proper video loop while playing in video loop mode.

Hopefully, we will pay our attention to find a more accurate extraction method of motion parameters. Furthermore, we will apply our methodologies on another targets such as human expression or something that exists regular idiosyncrasy.

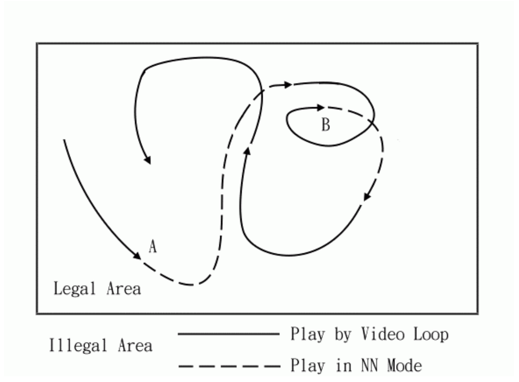


Figure 17 *Hybrid Playing mode.* When the fish is swimming until A in video loop playing mode, the program switch into another playing mode because of the lack of proper video loop, which can bring the fish to the legal area. In B, the orientation of current fish head is entirely different from the video loop and the worse is that no more video loop is available. For this reason, we have to abort video loop playing mode and switch into neural network mode.

Acknowledgements

This project is supported by NSC-90-2213-E-006-086

Reference

[1] Arno Schödl, Richard Szeliski, David H. Salesin and Irfan Essa. Video Textures. In Computer Graphics Proceedings, Annual Conference Series, pages 489 - 498, Proc. SIGGRAPH'2000 (New Orleans), July , 2000. ACM SIGGRAPH.

[2] A. Schödl, I. Essa, Machine Learning for Video-Based Rendering, Georgia Institute of Technology, GVU Center / College of Computing, 2000.

[3] P. Debevec et al., editors. *Image-Based Modeling, Rendering, and Lighting,* SIGGRAPH'99 Course 39, August 1999.

[4] S. Haykin, Neural Networks: A Comprehensive Foundation 2nd Edition, Prentice Hall, 1999.

[5] Demetri Terzopoulos, X. Tu, R. Grzeszczuk, Artificial Fishes: Autonomous Locomotion, Perception, Behavior, and Learning in a Simulated Physical World, Dept. of CS, U of Toronoto ,1994 ACM SIGGRAPH.

[6] Radek Grzeszczuk, Demetri Terzopoulos, Automated Learning of Muscle-Actuated Locomotion Through Control Abstraction, Dept. of CS, U of Toronto. 1995. ACM SIGGRAPH.

[7] Demetri Terzopoulos, R. Grzeszczuk, G. Hinton, NeuroAnimator: Fast Neural Network Emulation and Control of Physics-Based Models, Phd. Thesis, Dept. of CS, U of Toronoto ,May 1998.

[8] Ziv Bar-Joseph, Dani Lischinski, Statistical Learning of Multi-Dimensional Textures, Master's Thesis, Institute of Computer Science, The Hebrew University of Jurusalem, Israel, 1999.

[9] X. Wen, Theodore D. Huffmire, Helen H. Hu, and Adam Finkelstein, Wavelet-Based Video Indexing and Querying for a Smart VCR, Princepton University, 1997.

[10] Adam Finkelstein, Charles E. Jacobs, David H. Salesin. Multiresolution Video. Proceedings of SIGGRAPH 96, in *Computer Graphics* Proceedings, Annual Conference Series, 281-290, August 1996.

[11] H. Takahashi, J. Hatoya, N. Hashimoto, M. Nakajima, Animation synthesis for virtual fish from video, 2000.

[12] Lipton, Virtual Postman - Real-Time, Interactive *Virtual Video*, *IASTED CGIM*, Palm Springs, CA, October 1999.

[13] J. R. Parker, Algorithms for Image Processing and Computer Vision, Wiley Computer Publishing, 1997.

Appendix

In this section, we will illustrate two flow charts about the described playing mode. One is neural network playing mode (See Figure 18), and the other is the hybrid playing mode (See Figure 19).

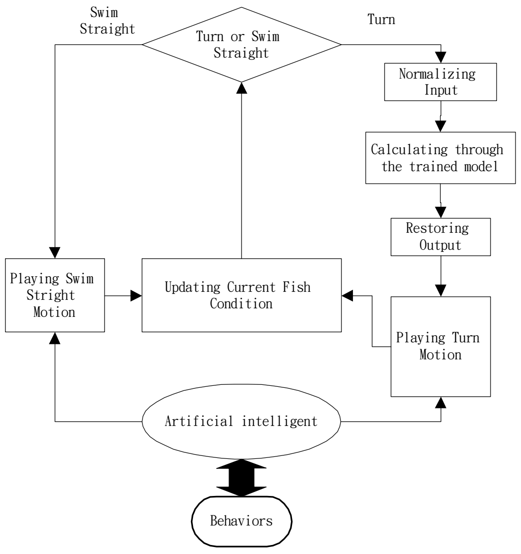
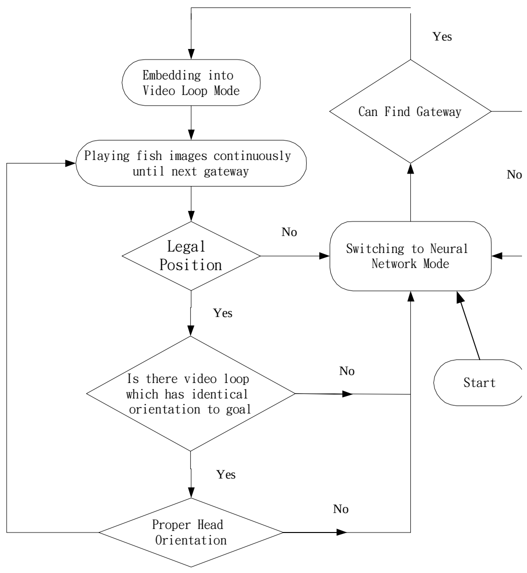


Figure 18. *Playing in Neural Network Mode.* We use different procedures for turning or straight swimming. The first step of this playing mode is to decide whether the fish will turn or not. If the fish suppose to turn, then the degree of turning, that is the ideal motion area, would be calculated through the trained motion model. If not, then the program will switch into another procedure that only plays straight swimming. Furthermore, we add several usual fish behaviors, such as sliding, accelerating, decelerating, so as to increase the variation of fish animation.



Yes
Embedding into Video Loop Mode
Can Find Gateway
No
Playing fish images continuously until next gateway
Legal Position
No
Switching to Neural Network Mode
Yes
No
Is there video loop which has identical orientation to goal
Start
Yes
No
Proper Head Orientation

Figure 19. *Playing in hybrid mode.* At the beginning of this playing mode, we generate the animation in neural network mode until find a proper gateway for current fish condition. Once switching into the video loop mode, whenever drawing until any gateway, we have to do several decisions or select proper video loop to ensure whether the current video is legal.