Deformable Mesh

Tai-Hei Wu and Cheng-Yuan Liou†
Department of Computer Science and Information Engineering, National Taiwan University

***Abstract*— This paper uses Neural Mesh to model deformable objects. The energy of Neural Mesh is minimized using a modified Hopfield neural network. In order to model deformable objects, the energy function of Neural Mesh is approximated by a second-order Taylor series expansion. Experiment results are given on flying flags.**

I. Introduction

Snake[21] is originally used to retrieve information about an image. Liou and Chang[7] extended the idea of snake into meshed snake and used Hopfield neural network to solve the minimization problem. The meshed snake is then called Neural Mesh and is used to solve other problems, such as Steiner Tree Problem[8], Minimial Surface Problem[6]. In this work, a special application of Neural Mesh, the deformable mesh, is developed. This deformable mesh can be used to model deformable objects, such as flying flag. To this goal, a modified Hopfield neural network that is more suitable for this problem is proposed.

II. Neural Mesh

Neural Mesh proposed by Liou and Chang[7] is a mesh which can capture information of image in the process of mesh evolution. The process of mesh evolution is driven by process of energy minimization. The energy function of Neural Mesh, which has a quadratic form, is defined by each point of the mesh:

$$E = \sum_{(i,j)\in M} (P_i - P_j)^2, \tag{1}$$

where M is a mesh having n points, P_i is the ith point in the mesh, and $L_{i,j}$ is an edge between P_i and P_j in the mesh (we ignore the bending term in the original work for simplicity.)

To evole the mesh, Neural Mesh finds a new configuration of points $P_i^{'} = P_i + D_i$ for the mesh which has smaller energy value than the original configuration:

$$\arg\min_D \big((P_i + D_i) - (P_j - D_j)\big)^2, \tag{2}$$

where $D = [D_1, D_2, ...D_n]$ is the displacement matrix for the mesh. Additional constraints may be imposed on D, according to different designs.

This research was supported by National Science Council under project number NSC 93-2213-E-002-008.

†Correspondent, cyliou@csie.ntu.edu.tw, Dept. of CSIE, NTU, Taipei, Taiwan, 106, ROC.

The quadratic property and the energy minimization property let it possible to minimize the energy function of Neural Mesh by Hopfield neural network.

The energy function of Hopfield neural network is defined as:

$$E_{Hopfield} = -\frac{1}{2}\sum_{i,j} W_{i,j}V_iV_j - \sum_i I_iV_i. \tag{3}$$

Let $E = E_{Hopfield}$ by appropriately setting W_i and I_i, the minimization of the Hopfield network will be equivalent to the minimization of the energy function of Neural Mesh.

III. Dynamics of the Mesh

To model deformable objects in Neural Mesh, the mesh is treated as the surface of an object. Analogous to mass-spring model in computer graphics, each edge of the mesh is treated as a spring. First, by extending Neural Mesh into three-dimensional, we have a 3D neural mesh. Consider a 3D neural mesh Q which consists of n points represented by $P_i, i = 1 \sim n$ and edges between points represented by $L_{i,j}$ if there is an edge between point i and point j. Next, every edge $L_{i,j}$ in Q is treated as a spring with natural length $\bar{L}_{i,j}$. The original Neural Mesh now becomes a special case of this 3D neural mesh with $\bar{L}_{i,j} = 0$ and two-dimension only.

Analogous to (1), an energy function of the mesh Q which represents the potential of springs is defined as

$$E_Q = \sum_{L_{i,j}\in Q} \alpha_{i,j}(|P_i - P_j| - \bar{L}_{i,j})^2, \tag{4}$$

where $\alpha_{i,j}$ is the weighting factor for $L_{i,j}$.

Then, similar to (2), the dynamics of the mesh is a minimization process of E_Q. Every spring minimizes its potential and the lost potential is transformed into kinetic energy. Now the deformation of the object, in theorically, can be simulated.

Since acceleration is involved in the deformation, the original Hopfield neural network used in Neural Mesh is not suitable to minimize E_Q, because the output of the Hopfield neural network is binary. Furthermore,a sophisticated energy function which involved acceleration should be considered.

Let $x_i(t) = [x_{i,1}(t), x_{i,2}(t), x_{i,3}(t)]$ be the coordinate vector, $v_i(t) = [v_{i,1}(t), v_{i,2}(t), v_{i,3}(t)]$ be the velocity vector,

$a_i(t) = [a_{i,1}(t), a_{i,2}(t), a_{i,3}(t)]$ be the acceleration vector of P_i at time t, respectively. Then for any time t_1, t_2 such that $t_2 = t_1 + \Delta t$, we can approximate $x_i(t2)$ by

$$x_i(t_2) = x_i(t_1) + v_i(t_1)\Delta t + \frac{1}{2}(a_i(t_1) + \Delta a_i(t_1))\Delta t^2, \quad (5)$$

where $\Delta a_i(t_1) = [a_{i,1}(t_1), a_{i,2}(t_1), a_{i,3}(t_1)]$ is the change of acceleration from time t_1 to t_2.

Rewrite (4) by substituting P_i with $x_i(t_2)$ and P_j with $x_j t_2$:

$$E_Q(t_2) = \sum_{L_{i,j}\in Q} \alpha_{i,j} \left(|x_i(t_2) - x_j(t_2)| - \bar{L}_{i,j}\right)^2$$

$$= \textstyle\sum_{L_{i,j}\in Q} \Bigg[\alpha_{i,j}\Big(\big|x_i(t_1) + v_i(t_1)\Delta t + \frac{1}{2}[a_i(t_1) + \Delta a_i(t_1)]\Delta t^2 - (x_j(t_1) + v_j(t_1)\Delta t + \frac{1}{2}[a_j(t_1) + \Delta a_j(t_1)]\Delta t^2)\big| - \bar{L}_{i,j}\Big)^2\Bigg]$$

$$= \sum_{L_{i,j}\in Q} \alpha_{i,j} \Bigg[\sqrt{\sum_{p=1\sim 3} \Big(C_{i,j,p}(t_1)) + \frac{1}{2}\Delta t^2 \big(\Delta a_{i,p}(t_1) - \Delta a_{j,p}(t_1)\big)\Big)^2} - \bar{L}_{i,j}\Bigg]^2,$$

where $E_Q(t)$ represents the potential energy of mesh Q at time t and $C_{i,p}(t)$ is defined as:

$$C_{i,p}(t) = x_{i,p}(t) + v_{i,p}(t)\Delta t + \frac{1}{2}(a_{i,p}(t))\Delta t^2,$$

and $C_{i,j,p}(t) = C_{i,p}(t) - C_{j,p}(t)$.

The problem of choosing the next configuration of the mesh Q which minimize $E_Q(t_2)$ is now becomes an minimization problem that, finding a parameter matrix

$$A(t_1) = \begin{bmatrix} \Delta a_1(t_1) \\ \Delta a_2(t_1) \\ \vdots \\ \Delta a_n(t_1) \end{bmatrix}$$

which minimize $E_Q(t_2)$.

In order to simulate the deformation more precisely and speed up the minimization process, $\Delta a_i(t)$ is used for probing only and is constrained into an unit vector, that is, $\Delta a_{i,1}^2(t) + \Delta a_{i,2}^2(t) + \Delta a_{i,3}^3(t) = 1$. After $\Delta a_i(t)$ is obtained, an energy change $\Delta E_i(t_1)$ from time t_1 to t_2 corresponding to every point P_i in Q is calculated:

$$\Delta E_i(t_1) = \sum_{L_{k,j}\in Q} \alpha_{k,j} \Bigg[\Big(|x_k(t_1) + v_k(t_1)\Delta t + \frac{1}{2}[a_k(t_1) + \delta(i,k)\Delta a_k(t_1)]\Delta t^2 - (x_j(t_1) + v_j(t_1)\Delta t + \frac{1}{2}[a_j(t_1) + \delta(i,j)\Delta a_j(t_1)]\Delta t^2)| - \bar{L}_{k,j}\Big)^2 - \Big(|x_k(t_1) - x_j(t_1)| - \bar{L}_{k,j}\Big)^2\Bigg], \quad (6)$$

where $\delta_{i,j}$ is the Kronecker delta function. $\Delta E_i(t_1)$ represents the energy change from t_1 to t_2 assuming that acceleration change only occurred in P_i.

From the idea of energy conservation, $\Delta E_i(t_1)$ is transformed into kinetic energy of P_i. Thus, the acceleration at time t_2 is obtained by

$$a_i(t_2) = a_i(t_1) + \Delta a_i(t_1)\sqrt{|\gamma \Delta E_i(t_1)|}, \quad (7)$$

where γ is a weighting parameter. The velocity $v_i(t_2)$ and position $x_i(t_2)$ can then be obtained from $a_i(t_2)$.

After velocity is obtained, energy lose due to various frictions is introduced by multipling v_i with a ratio $R_{friction}$. Additionally, all external forces are directly applied on velocity each iteration. More precisely, the followings are used to update $v_i(t_2)$ and $a_i(t_2)$:

$$\begin{aligned} v_i(t_2) &= v_i(t_2)R_{friction}, \\ a_i(t_2) &= a_i(t_2) + \text{external force}. \end{aligned}$$

The dynamics of the mesh is now well-described.

Because $\Delta a_{i,p}(t) \in R$, it is hard to minimize $E_Q(t)$ using a traditional Hopfield neural network. In the next section, a modified Hopfield neural network is proposed. The modified Hopfield neural network is more suitable for this situation.

IV. Modified Hopfield Neural Network

In this section, we extend Hopfield neural network with the capability to handle real number.

At first, we check the convergence and energy-minimization properties of Hopfield neural network[15]:

$$\begin{aligned} \Delta E &= -[\sum_{j\neq i} W_{i,j}V_j + I_i]\Delta V_i \\ &\leq 0. \end{aligned}$$

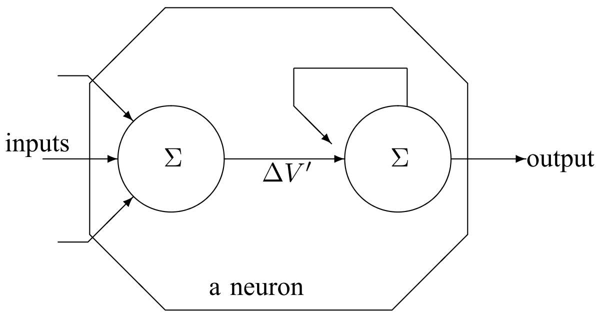

Fig. 1. **A neuron of the modified Hopfield Neural Network.**

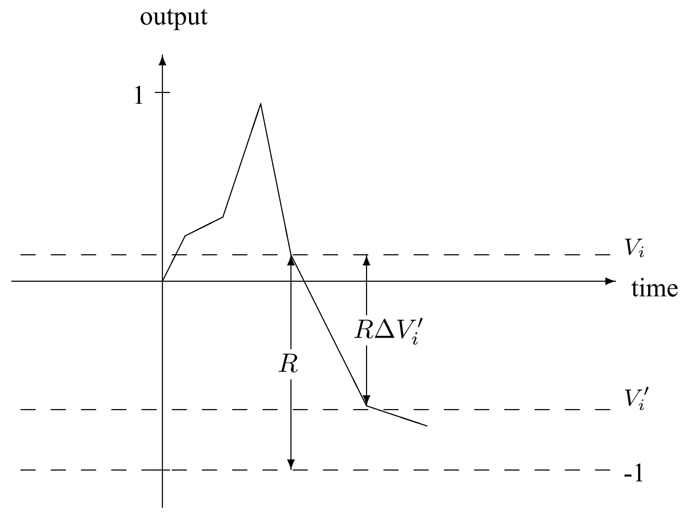

Fig. 2. **The Updating Rule.** One can think of the updating rule as the inputs influence the output in a biased way, but not directly decide the output. Previous output still has influence on the future output.

This result guarantees the convergence and energy-minimization properties of Hopfield neural network. But, in order to guarantee these properties, the outputs of this model are constrained to be binary. This constraint enforces following inequation to be true:

$$\Delta V_i\big(\sum_{j\neq i} W_{i,j}V_j + I_i\big) \geq 0. \tag{8}$$

In order to keep (8) true and make input/output of neurons be real, a neuron in Hopfield neural network is divided into two parts: input part and output part. The input part collects inputs from other neurons and sends them to the output part. The output part modifies and produces the output of this neuron according the inputs provided by the input part. The modified neuron can be viewed as a neuron with self-loop, although such neuron is not extactly the same with the modified neuron. Figure 1 depicts the modified neuron.

From this idea, we modify the updating rule of Hopfield neural network by

$$\Delta V_i^{'} = \frac{\sum_{j\neq i} W_{i,j}V_j + I_i}{\sum_{j\neq i} |W_{i,j}| + |I_i|} \tag{9}$$

and

$$\Delta V_i = R\Delta V_i^{'}, \tag{10}$$

where R is defined as

$$R = \begin{cases} 1 - V_i, & \text{if} \quad \Delta V_i^{'} \geq 0, \\ 1 + V_i, & \text{if} \quad \Delta V_i^{'} < 0. \end{cases} \tag{11}$$

The output is adjusted by

$$V_i' = V_i + \Delta V_i \tag{12}$$

where V_i' is the new output and V_i is the old output. Notice that for this updating rule to work, at least one of I_i and $W_{i,j}$ must be nonzero.

The divisor in (9) normalizes the dividend, keeping $-1 \leq \Delta V_i' \leq 1$. It is then served as the ratio of R. Obviously, R in (10) is always positive(from (11)), thus $\Delta V_i = R\Delta V_i$ always has the same sign with $\sum_{i\neq j} W_{i,j}V_iV_j$. Figure 2 shows how output is modified by this updating rule.

Now, check the energy function of the modified Hopfield neural network:

$$\begin{aligned} E &= -\frac{1}{2}\sum\sum_{i\neq j} W_{i,j}V_iV_j - \sum_i I_iV_i \\ \Delta E &= -[\sum_{j\neq i} W_{i,j}V_j + I_i]\Delta V_i \\ &= -[\sum_{j\neq i} W_{i,j}V_j + I_i] * R\Delta V_i^{'} \\ &= -R\frac{[\sum_{j\neq i} W_{i,j}V_j + I_i]^2}{\sum_{j\neq i} |W_{i,j}| + |I_i|} \\ &\leq 0. \end{aligned} \tag{13}$$

Thus the modified Hopfield neural network preverses the convergence and energy-minimization properties of the original Hopfield neural network. Inputs for neurons affect neurons in a less direct way.

To further examine the property of the modified Hopfield neural network, notice that the diagonal weight ($W_{i,i}$) of the original Hopfield neural network, in fact, needs not to be zero to satisfy the convergence property. It is necessary when $W_{i,i}$ is positive[13].

To verify the same property in the modified Hopfield neural network, redefine the updating rule and energy function to include diagonal terms, then

$$\Delta V_i = R\left(\frac{\sum_{j=1\sim n} W_{i,j}V_j + I_i}{\sum_{j=1\sim n} |W_{i,j}| + |I_i|}\right)$$

and

$$\begin{aligned} E &= -\frac{1}{2}\sum\sum_{i,j=1\sim n} W_{i,j}V_iV_j - \sum_i I_iV_i \\ \Delta E &= -[\sum_{i\neq j} W_{i,j}V_j + I_i]\Delta V_i - \frac{1}{2}(2W_{i,i}V_i\Delta V_i + W_{i,i}\Delta V_i^2) \\ &= -[\sum_{j=1\sim n} W_{i,j}V_j + I_i] * \Delta V_i - \frac{1}{2}W_{i,i}\Delta V_i^2 \\ &= -R\frac{[\sum_{j=1\sim n} W_{i,j}V_j + I_i]^2}{\sum_{j=1\sim n} |W_{i,j}| + |I_i|} - \frac{1}{2}W_{i,i}\Delta V_i^2 \\ &\leq 0. \end{aligned}$$

if $W_{i,i} \geq 0$.

Thus, the same property is also true in a modified Hopfield neural network.

Furthermore, to satisfy $\Delta E < 0$,

$$-R\frac{[\sum_{j=1\sim n} W_{i,j}V_j + I_i]^2}{\sum_{j=1\sim n} |W_{i,j}| + |I_i|} - \frac{1}{2}W_{i,i}\Delta V_i^2 \leq 0$$

$$\begin{aligned} \frac{1}{2}W_{i,i}\Delta V_i^2 &\geq -R\frac{[\sum_{j=1\sim n} W_{i,j}V_j + I_i]^2}{\sum_{j=1\sim n} |W_{i,j}| + |I_i|} \\ \frac{W_{i,i}R^2}{2}\frac{[\sum_{j=1\sim n} W_{i,j}V_j + I_i]^2}{(\sum_{j=1\sim n} |W_{i,j}| + |I_i|)^2} &\geq -R\frac{[\sum_{j=1\sim n} W_{i,j}V_j + I_i]^2}{\sum_{j=1\sim n} |W_{i,j}| + |I_i|} \\ W_{i,i} &\geq -\frac{2}{R}(\sum_{j=1\sim n} |W_{i,j}| + |I_i|) \end{aligned}$$

for $R \neq 0$, $(\sum_{j=1\sim n} |W_{i,j}| + |I_i|) \neq 0$, and $(\sum_{j=1\sim n} W_{i,j}V_j + I_i) \neq 0$. But this is always true for any $W_{i,i}$ by observing that $W_{i,i}$ appears in two sides of the inequation. Thus, in a modified Hopfield neural network, there is no constraint on $W_{i,i}$. The value of $W_{i,i}$ only affects the convergence speed of the modified Hopfield neural network.

V. The Deformable Mesh

The modified Hopfield neural network is used to minimize the energy function of 3D neural mesh, just as the original Hopfield neural network is used to minimize the energy function of the Neural Mesh. The modified Hopfield neural network consists of $n * 3$ mutual interconnected neurons, where n is the number of points of the mesh.

Let $V_{i,p}$ be the output of the (i,p)th neuron in the network and $W_{i,p;j,q}$ be the synaptic weight from the (j,q)th neuron to the (i,p)th neuron, the energy function of the modified Hopfield neual network , $E'_{Hopfield}$, can be written as

$$E'_{Hopfield} = -\frac{1}{2}\sum_{i=1}^{n}\sum_{p=1}^{3}\sum_{j=1}^{n}\sum_{q=1}^{3} W_{i,p;j,q}V_{i,p}V_{j,q} - \sum_{i=1}^{n}\sum_{p=1}^{3} I_{i,p}V_{i,p}. \tag{14}$$

Next, every neuron is used to represent the change of acceleration defined previous. Hence, replace $\Delta a_{i,p}$ in(6) by $V_{i,p}$ and plus it by a constrained energy which is minimized when $(V_{i,1}, V_{i,2}, V_{i,3})$ forms an unit vector: (Here we write E_Q instead of $E_Q(t)$ for simplicity)

$$\begin{aligned} &E_Q = \sum_{L_{i,j}\in Q} \alpha_{i,j}\Bigg[\\ &\sum_{p=1\sim 3}\left(C_{i,p} - C_{j,p} + \frac{1}{2}\Delta t^2(V_{i,p} - V_{j,p})\right)^2 \\ &-2\bar{L}_{i,j}\sqrt{\sum_{p=1\sim 3}\left(C_{i,p} - C_{j,p} + \frac{1}{2}\Delta t^2(V_{i,p} - V_{j,p})\right)^2} + \bar{L}_{i,j}^2\Bigg] \\ &+\sum_{i\in Q}\beta_i\left(1 - \sqrt{\sum_{p=1\sim 3} V_{i,p}^2}\right)^2. \end{aligned} \tag{15}$$

Replace the square root terms with the second-order taylor series expansion of them about these points $(V_{i,1}^0, V_{i,2}^0, V_{i,3}^0)$ and $(V_{j,1}^0, V_{j,2}^0, V_{j,3}^0)$:

$$\begin{aligned} &\sqrt{\sum_{p=1\sim 3}\left(C_{i,p} - C_{j,p} + \frac{1}{2}\Delta t^2(V_{i,p} - V_{j,p})\right)^2} \\ &= B_{i,j}^{-1} + \sum_{p=1\sim 3}(V_{i,p} - V_{i,p}^0)(\frac{1}{2}D_{i,j,p}B_{i,j}) \\ &+ \sum_{p=1\sim 3}(V_{j,p} - V_{j,p}^0)(-\frac{1}{2}D_{i,j,p}B_{i,j}) \\ &+ \sum_{p=1\sim 3}(V_{i,p} - V_{i,p}^0)^2\left(\frac{1}{8}\left(\Delta t^4 B_{i,j} - D_{i,j,p}^2 B^3\right)\right) \\ &+ \sum_{p=1\sim 3}(V_{j,p} - V_{j,p}^0)^2\left(\frac{1}{8}\left(\Delta t^4 B_{i,j} - D_{i,j,p}^2 B^3\right)\right) \end{aligned}$$

$$+ \sum_{p=1\sim 3}(V_{i,p}-V_{i,p}^0)(V_{j,p}-V_{j,p}^0)\Big(\frac{D_{i,j,p}^2B_{i,j}^3-\Delta t^4 B_{i,j}}{4}\Big)$$
$$+ \sum_{p\neq q}(V_{i,p}-V_{i,p}^0)(V_{j,q}-V_{j,q}^0)\Big(\frac{D_{i,j,p}D_{i,j,q}B_{i,j}^3}{4}\Big)$$
$$+ \sum_{p\neq q}(V_{i,p}-V_{i,p}^0)(V_{i,q}-V_{i,q}^0)\Big(\frac{-D_{i,j,p}D_{i,j,q}B_{i,j}^3}{4}\Big),$$

where

$$B_{i,j} = \sqrt{\sum_{p=1\sim 3}\Big(C_{i,p}-C_{j,p}+\frac{1}{2}\Delta t^2(V_{i,p}^0-V_{j,p}^0)\Big)^2}^{\,-1}$$

and

$$D_{i,j,p} = \Delta t^2(C_{i,p}-C_{j,p})+\frac{\Delta t^4}{2}(V_{i,p}^0-V_{j,p}^0).$$

The second square root is expanded as:

$$\sqrt{\sum_{p=1\sim 3}V_{i,p}^2} = F_i + \sum_{p=1\sim 3}(V_{i,p}-V_{i,p}^0)\frac{V_{i,p}^0}{F_i}$$
$$+\frac{1}{2}\sum_{p=1\sim 3}(V_{i,p}-V_{i,p}^0)^2\Big(\frac{F_i^2-(V_{i,p}^0)^2}{F_i^3}\Big)$$
$$-\sum_{p=1\sim 3}\sum_{q\neq p}(V_{i,p}-V_{i,p}^0)(V_{i,q}-V_{i,q}^0)\Big(\frac{V_{i,p}^0V_{i,q}^0}{F_i^3}\Big)$$

where

$$F_i = \sqrt{\sum_{p=1\sim 3}(V_{i,p}^0)^2}.$$

Finally, solve $E_Q = E'_{Hopfield}$, $W_{i,p;j,q}$ will be

$$W_{i,p;j,q} = T_1(i,p,j,q)+T_2(i,p,j,q)+T_3(i,p,j,q)+T_4(i,p,j,q)$$

where

$$T_1(i,p,j,q) = -2\beta_i\delta_{i,j}\delta_{p,q}+\delta_{i,j}\sum_{L_{i,k}\in Q}\Bigg(\frac{1}{2}(2-\delta_{p,q})\alpha_{i,k}\bar{L}_{i,k}\Big(\delta_{p,q}\Delta t^4B_{i,k}-D_{i,k,p}D_{i,k,q}B_{i,j}^3\Big)\Bigg)$$

$$T_2(i,p,j,q) = -(1-\delta_{i,j})\sum_{L_{i,j}\in Q}\Bigg(\alpha_{i,j}\bar{L}_{i,j}\Big(\delta_{p,q}\Delta t^4B_{i,j}-D_{i,j,p}D_{i,j,q}B_{i,j}^3)\Big)-2\alpha_{i,j}\Delta t^2\Bigg)$$

$$T_3(i,p,j,q) = \delta_{i,j}\delta_{p,q}\Big(-2\alpha_{i,j}\frac{\Delta t^4}{4}+2\beta_i\frac{F_i^2-(V_{i,p}^0)^2}{F_i^3}\Big)$$

$$T_4(i,p,j,q) = \delta_{i,j}4\beta_i\frac{V_{i,p}^0V_{i,q}^0}{F_i^3}$$

and

$$I_{i,p} = \sum_{L_{i,j}\in Q}\Bigg\{-\alpha_{i,j}\Delta t^2+2\bar{L}_{i,j}\Bigg[\frac{1}{2}D_{i,j,p}B_{i,j}-V_{i,p}^0\Big[\Delta t^4B_{i,j}-D_{i,j,p}^2B_{i,j}^3\Big]$$
$$+\frac{1}{2}V_{j,p}^0\Big[\Delta t^4B_{i,j}-D_{i,j,p}^2B_{i,j}^3\Big]$$
$$+\sum_{q\neq p}\frac{1}{2}(V_{j,q}^0-V_{i,q}^0)\Big[D_{i,j,p}D_{i,j,q}B_{i,j}^3\Big]\Bigg]\Bigg\}$$
$$+\beta_i\Bigg(-\frac{V_{i,p}^0}{F_i}+\frac{F_i^2-(V_{i,p}^0)^2}{F_i^3}-\sum_{q\neq p}V_{i,q}^0\frac{V_{i,p}^0V_{i,q}^0}{F_i^3}\Bigg).$$

After $W_{i,p;j,q}$ is obtained, the modified Hopfield neural network can be used to minimize the energy function of the deformable object to obtain $V_{i,p}$. From (7), acceleration a_i, velocity v_i and position x_i can also be obtained.

The whole algorithm is described as follow.

- Initial the mesh and network.
- Run modified Hopfield neural network until the network reaches one of stable states.
- Update the acceleration, velocity and position of points in the mesh.
- Update $W_{i,p;j,q}$.
- Repeatedly do above three operations.

VI. Experiment results

The experiment results are shown in Figure 3 and Figure 4. Table I shows the parameters used in these experiments.

TABLE I
EXPERIMENTS PARAMETERS

Figure	ΔT	$R_{friction}$	γ	elapsed time	node #
3	0.05	0.99	4000	93 mins	400
4	0.05	0.99	8000	54 mins	225

VII. Conclusion

This paper has implemented deformable objects on Neural Mesh. Similar to other mass-spring or energy solution, this method suffer from accuracy problem. Different γ and time step Δt lead to different outcomes. Another disadvantage is that the formula for $W_{i,p;j,q}$ is very complicated. It takes many calculations, both for computer and human. This approach also can only handle quadratic energy function; otherwise, Taylor series expansion must be introduced which brings more complicated calculations and the possibility of error.

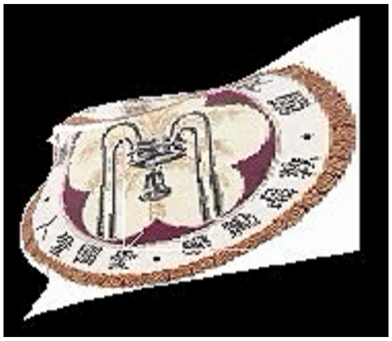

Fig. 3. **The badge of National Taiwan University on a flying flag.**

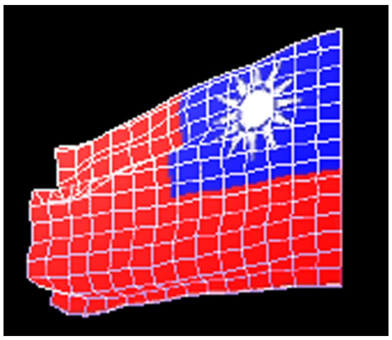

Fig. 4. **Another Experiment Result.**

On the other hand, it is a nice attempt to use neural network as a computational tool. Hardware supports and more sophisticated neural network model can speed up the computation and reduce the approximation error.

REFERENCES

[1] A. Radetzky, A. Nurnberger, D. P. Pretschner and R. Kruse, "The Simulation of Elastic Tissues in Virtual Laparoscopy Using Neural Networks," Proceedings of Neural Networks in Application (NN.98), pp. 167-174, University of Magdeburg, Magdeburg, Germany, 1998.

[2] A. Radetzky, A. Nurnberger, M. Teisler, D. P. Pretschner, "Elastodynamic Shape Modeling in Virtual Medicine," Proceedings of the International Conference on Shape Modeling and Applications, pp. 172, 1999.

[3] A. A. Amini, S. Tehrani, and T. E. Weymouth, "Using Dynamic Programming for Minimizing the Energy of Active Contours in the Presence of Hard Constraints," Proceedings of IEEE Conference of Computer Vision, pp. 95-99, 1988.

[4] B. Eberhardt, A. Weber, and W. Strasser, "A Fast, Flexible, Particle-System Model for Cloth Draping," IEEE Computer Graphics and Applications, vol. 16, issue 5, 1996.

[5] C. T. Tsai, "Minimizing the Energy of Active Contour Model Using a Hopfield Network," IEE Proceedings-E, vol. 140, no. 6, pp. 297-303, 1993.

[6] C. Y. Liou and Q. M. Chang, "Active mesh for minimal surface problems," Proceedings of International Conference on Nueral Information Processing, Dunedin, New Zealand, vol. 1, pp. 486-489, Nov. 1997.

[7] C. Y. Liou and Q. M. Chang, "Meshed Snakes," Proceedings of International Conference on Neural Network, Washington D. C., USA, vol. 3, pp. 1516-1521, Jun. 1996.

[8] C. Y. Liou and Q. M. Chang, "Numerical soap film for the steiner tree problem," Proceedings of International Conference on Neural Information Processing, Hong Kong, China, vol. 1, pp. 642-647, Sep. 1996.

[9] C. Y. Liou and Q. M. Chang, "Neural Mesh for the Steiner Tree Problem," Journal of Information Science and Engineering, vol. 13, no. 2, pp. 335-354, 1997.

[10] E. N. Gilbert and H. O. Pollak, "Steiner minimal trees." SIAM J. Allp. Math., vol. 16, pp. 1-29, 1968.

[11] F. Cordier and N. Magnenat-Thalmann, "Real-time Animationof Dressed Virtual Humans," EUROGRAPHICS 2002, vol. 21, no. 3, 2002.

[12] F. Leymarie and M. D. Levine, "Tracking Deformable Objects in the Plane Using an Active Contour Model," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15, no. 6, pp. 617-634, June 1993.

[13] Gene R. Gindi, Arthur F. Gmitro, and K. Parthasarathy "Hopfield model associative memory with nonzero-diagonal terms in memory matrix," Applied Optics, vol. 27, no. 1, 1998.

[14] J. Brown, S. Sorkin, C. Bruyns, J.C. Latombe, K. Montgomery, and M. Stephanides, "Real-Time Simulation of Deformable Objects: Tools and Application," Proceedings of Computer Animation, Seoul, Korea, November 2001.

[15] J. J. Hopfield and D. W. Tank, "Neural Computation of Decisions in Optimization Problems," Biological Cybernetics, vol. 52, pp. 141-152, 1985.

[16] J. Montagnat, H. Delingette and N. Ayache, "A Review of Deformable Surfaces: Topology, Geometry and Deformation," Image and Vision computing, vol. 19, issue 14, pp. 1023-1040, Dec. 2001.

[17] J. Mandziuk, "A Neural Network Designed to Solve the N-Queens Problem," Biological Cybernetrics vol. 66, no. 4, pp. 375-379, 1992.

[18] K. Tabb, S. George, N. Davey and R. Adams, "The Analysis of Animate Object Motion using Neural Networks and Snakes," Proceedings of the 6th International Conference on Engineering Applications of Neural Networks (EANN'2000) , D. Tsapsinos(ed), pp. 221-228, 2000.

[19] M. A. Greminger and B. J. Nelson, "Modeling Elastic Objects with Neural Networks for Vision-Based Force Measurement," Proceedings of International Conference on Intelligent Robots and Systems, Las Vegas, Nevada, Oct. 2003.

[20] M. Desbrun, P. Schroder and A. H. Barr, "Interactive animation of structured deformable objects," Proceedings of the 1999 conference on Graphics interface '99, pp. 1-8.

[21] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active Contour Models," Proceedings of the First International Conference on Computer Vision, IEEE Computer Society Press, 1987.

[22] M. Meyer, G. Debunne, M. Desbrun and A. H. Barr, " Interactive Animation of Cloth-like Objects in Virtual Reality," Journal of Vizualisation and Computer Animation vol. 12, issue 1, pp. 1-12, May 2001.

[23] O. Etzmub and B. Eberhardt, "Collision Adaptive Particle Systems," Proceedings of the 8th Pacific Conference on Computer Graphics and Application, pp. 338-453, Oct. 3-5, 2000.

[24] R. Grzeszczuk, D. Terzopoulos, and G. Hinton, "NeuroAnimator: Fast Neural Network Emulation and Control of Physics-Based Models," Proc. Int. Conf. on Computer Graphics and Interactive Techniques, pp. 9-20, 1998.

[25] Simon Haykin, "Neural Networks: A Comprehensive Foundation 2/e", Prentice Hall Press, 1999.

[26] S. W. Chen, G. C. Stockman and K. E. Chang, "SO Dynamics Deformation for Building of 3-D Models", IEEE Transaction on Neural Networks, vol. 7, no. 2, March 1996.

[27] S. Schein and G. Elber, "Placement of Deformable Objects," Computer Graphics Forum, vol. 23, no. 4, pp. 727-739, 2004.

[28] T. Kohonen, "Self Organized formation of topologically correct feature maps," Biological Cybernetics, vol. 43, pp. 59-69, 1982.

[29] T. H. Wu, Y. K. Tzeng, and C. Y. Liou, "Implement Neural Mesh on Cellular Neural Network," The 9th IEEE International Workshop on Cellular Neural Networks and their Applications, Taiwan, 2005.

[30] Vassilev, T., Spanlang, B, "A Mass-spring Model for Real Time Deformable Solids," East-West-Vision Sep. 2002.

[31] Wayne Sebastianelli, Elena Slobounov, Richard Tutwiler, "The Virtual Surgery Project," http://gears.aset.psu.edu/viz/services/projectlist/surgery/

[32] Y. J. Shen and M. S. Wang, "Apply Neural Schemes to Deformation Objects," International Journal on Graphics, Vision and Image Processing, vol. 4, 2005.