行政院國家科學委員會專題研究計畫　成果報告

計算視覺模式 同形與形態(3/3)

計畫類別：個別型計畫
計畫編號：NSC93-2213-E-002-008-
執行期間：93 年 08 月 01 日至 94 年 07 月 31 日
執行單位：國立臺灣大學資訊工程學系暨研究所

計畫主持人：劉長遠

計畫參與人員：郭彥廷

報告類型：完整報告
報告附件：出席國際會議研究心得報告及發表論文
處理方式：本計畫可公開查詢

中　華　民　國 94 年 5 月 2 日

Conformal Self-organizing Map

Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan 106, Republic of China, Supported by NSC 93-2213-E-002-008

Key words: conformality, surface reconstruction, conformal mapping, deformation measure, self-organizing map

Abstract.
This project presents the implementation of a surface mesh on a genus zero manifold with 3D scattered data of sculpture surfaces using the conformal self-organizing map(CSM). It starts with a regular mesh on a sphere and gradually shapes the regular mesh to match its object's surface by using the CSM. It can drape a uniform mesh on an object with a high degree of conformality. It accomplishes the surface reconstruction and also defines a conformal mapping from a sphere to the object's manifold.

1. Introduction

Laser scanners can sample a 3D object's surface data quickly and accurately, and yield enormous amounts of scattered digitized point data useful for surface modeling [27][2]. Many 3D objects like sculptures are classified as genus zero manifold [22]. Mapping a smooth mesh onto a sculpture's surface is an important issue in surface parameterization [15]. The conformal self-organizing map (CSM) can mimic a given manifold by continuously and selectively tuning to the input point patterns [20][25][21][26]. That is, its neurons can span the manifold smoothly. Therefore, it is able to lay a smooth mesh on the manifold. The input pattern points for CSM are unorganized points, therefore, CSM is also capable of solving the surface reconstruction problem.

The topological space of a given manifold and the parameterization domain affect the mapping distortion [11]. A large amount of distortion is unavoidable when different topological spaces are parameterized, e.g. from a genus zero manifold to a flat $\mathcal{R}^2$ plane. In the texture mapping procedure, the range data must be segmented into an atlas [19]. But for applications such as morphing and remeshing, it is best to parameterize the mesh over a domain that is topologically equivalent to it [11].

In this project, we focus on the genus zero manifold. Many 3D manifolds belong to the genus zero class, such as creatures, sculptures of the human body, etc. It is natural to use spherical parameterization for genus zero manifolds. Thus, we extend the CSM to the conformal spherical self-organizing map (CSSM), which employs spherical network space.

We will introduce the CSSM in detail and then employ useful deformation indicators, conformality measures, to quantify the mapping quality. The result will be compared with results obtained using the self-organizing map (SOM)[20].

Related works

The CSSM method is capable of reconstructing a surface from unorganized points and defining a conformal mapping from a sphere to certain object's manifold. There are three modern methods to accomplish the surface reconstruction with varying degrees of success. They are neural network methods, interpolation methods and approximation methods. Yu [28] and Barhak *et al.* [2] employ SOM to reconstruct a closed surface of genus 0. Ivrissimtzis *et al.* [16] develops the growing cell structure, which is also derived from SOM, to generate fitting meshes for various objects. The interpolation methods include the α-shape by Edelsbrunner *et al.* [10] and the 'crust' by Amenta and Bern [1]. They work well for uniform and dense sampling, but the local topology may deviate and have holes due to undersampling. The approximation methods include algorithms developed by Hoppe *et al.* [14] and Curless *et al.* [8]. They calculate the normal vector from a data set and obtain its tangent plane. All of the three modern methods solve the surface reconstruction properly, but they do not seek a surface with the content of conformal mapping.

There are five approaches to achieve conformal parameterizations: harmonic energy minimization, Cauchy-Riemann equation approximation, Laplacian operator linearization, angle based method and circle packing [13]. Gu and Yau [12] introduce a method for modeling genus zero surfaces based on nonlinear optimization of harmonic energy. Their algorithm starts with a given mesh. That is, it is not designed to resolve unorganized points and noisy data. This project presents a novel flexible mesh that can resolve unorganized points and noisy data. This mesh is capable of reconstructing a surface from unorganized points.

2. Conformal spherical SOM (CSSM)

The conformal SOM, CSM [20][25], attempts to accomplish conformal transformations between forms. It uses a *Euclidean plane* as its network space, e.g., $\mathcal{R}^2$. But an $\mathcal{R}^2$ plane cannot wrap a genus zero manifold

(a) The network space. There are 11x11 neurons arranged uniformly in a rectangular plane $\in R^2$. (b) The input space. The location of each lattice node is represented by its corresponding neuron's weight vector $w_i \in R^3$.

Figure 1.

without producing seams. See an example in Figs. 1 and 2. Therefore, we extend the CSM to the CSSM, which uses a sphere as its network space. This is because a sphere is topologically equivalent to a genus zero manifold [23]. The details about arranging neurons and the CSSM algorithm will be given in successive subsections.

The SOM model [20] is made of n neurons. The neurons are usually placed regularly in one- or two-dimensional space that is named the network space. The neurons of the CSSM model are placed on the tessellation of a unit sphere. Each neuron has a weight vector (or synapse vector) w_i where w_i contains the location of the i^{th} neuron in the input space. Fig. 1 shows the positions of the neurons in the network space and the input space.

2.1. The spherical network space

The neurons of the SOM are usually arranged uniformly in Euclidean space lattices [18]. Adhering to this property, the neurons are arranged uniformly on a unit sphere. We use a geodesic dome to approximate this [17]. There are five tessellations (platonic bodies) of a sphere; terahedron, octahedron, cube, pentagondodecahedron, and icosahedron [23]. An icosahedron is preferred because each of its face is an equilateral

This figure shows that an $\mathcal{R}^2$ plane cannot wrap a surface with a genus zero manifold. The meshes in (c) and (d) are learned by SOM with $\mathcal{R}^2$ network space.

Figure 2.

triangle. The basic type of icosahedron has 12 vertices, 30 edges, and 20 equivalent equilateral triangular faces. It is varied by combining more icosahedrons into a single body. We use the term f(frequency) to denote its multiplicity. The formula of the icosahedron is

$$\begin{aligned} Faces &= 20f^2, \\ Vertices &= \frac{Faces}{2} + 2. \end{aligned} \tag{1}$$

See Fig. 3 for frequencies from 1 to 6.

Since the neurons are arranged on the surface of a unit sphere, its metric should no longer be *Euclidean.* Instead, we compute the distance along the sphere surface. The distance between two neurons is

$$d = \cos^{-1}(n_i \cdot n_j)\,|n_i|\,|n_j| = \cos^{-1}(n_i \cdot n_j)\,, \tag{2}$$

where n_i and n_j are 3 dimensional column vectors with a unit magnitude, $|n_i| = |n_j| = 1$, and contain the locations of neurons i and j on the sphere, respectively. The center of the sphere is at the origin, $(0, 0, 0)$.

Icosahedrons approximate spheres at different frequencies. From (a) to (f) at $f = 1$ to $f = 6$, respectively.

Figure 3.

2.2. Learning algorithm

The CSSM learning algorithm is very similar to the CSM. The only difference is in the distance metric, see Eq. 2. The CSSM model is a continuous version of the SOM with spherical network space. It uses conformal mapping to compute the precise location of a pattern mapped onto the network space. We will first introduce the CSSM model, some terminologies, and then the learning algorithm.

The CSSM model contains neurons that are arranged on a sphere surface (which is approximated by a multi-frequency icosahedron). Each vertex of the icosahedron is set as a neuron in CSSM, see Fig. 3. The evolution of these neurons' weights proceeds based on competitive learning with a conformal updating rule. Each neuron occupies a fixed location in the network space and represents a marker in the input space, a vertex point in a mesh. Here, let n_i be the i^{th} neuron's location in the network space and have a fixed value. Let $w_i(t)$, the neuron's weight vector, be the i^{th} neuron's location in the input space at learning time t. $w_i(t)$ is a 3D column vector. Let X denote all input patterns, the set of all scattered points sampled from the scanned model, and let $x \in X$ be an input pattern. The learning algorithm is as follows.

1. *Initialization.* Initialize the CSSM network. In all our simulations, we initialize neurons' weight vectors as their uniform locations on a sphere.

   $$w_i\,(t=0) = n_{i,} \text{ position of the } \; i^{th} \text{ neuron on a sphere as in Fig. 3.} \tag{3}$$

   The neurons' weight vectors denote the positions of the mesh vertices, see Fig. 1(b). Set the initial variance $\sigma_{t=0}$ and initial learning rate $\alpha_{t=0}$. The variance and learning rate decrease gradually with the annealing scheme, e.g., $\sigma_t = \sigma_0 \exp\left(-\frac{t}{\tau_1}\right)$ and $\alpha_t = \alpha_0 \exp\left(-\frac{t}{\tau_2}\right)$, where τ_1 and τ_2 are time constants and t denotes the learning time. We start the algorithm from $t=0$.

2. *Sampling.* Randomly choose an input pattern $x \in X$ with equal probability. X is the set of all scattered points of the model.

3. *Similarity Matching.* Determine the winning or best-matching neuron by using

   $$\|w_c - x\| = \min_i \|w_i\,(t) - x\|\,, \quad w_i\,(t) \in W(t), \tag{4}$$

   where w_c is the weight vector of the winning neuron for the corresponding input x in time t, and $W(t)$ is the set of all weight vectors. That is to find a nearest neuron to the sampled point x.

4. *Updating.* Update all weight vectors according to the following equation:

   $$\begin{aligned} \Delta w_i &= \alpha_t h\left(d\left(\mathcal{M}\left(x\right), n_i\right)\right)\left(x - w_i\left(t\right)\right) = \alpha_t h\left(d\left(r, n_i\right)\right)\left(x - w_i\left(t\right)\right), \\ w_i\left(t+1\right) &= w_i\left(t\right) + \Delta w_i \;, \end{aligned} \tag{5}$$

   where $\alpha_t \in [0,1)$ is the learning rate at time t, and h is the neighborhood function which decreases monotonously with the distance metric d in the network space. This step is to improve the similarity of the weight vectors toward the pattern x. Here, we use a Gaussian neighborhood function:

   $$h\left(d\right) = \exp\left(-\frac{d^2}{2\sigma_t^2}\right), \tag{6}$$

   where σ_t is the variance at time t. The distance metric d here is based on the spherical metric, $d = \cos^{-1}\left(n_i \cdot r\right)$. $r = \mathcal{M}\left(x\right)$ is the reference vector of input pattern x projected onto the network field. The function $\mathcal{M}$ first projects pattern x onto the simplex, s, formed by the winning neuron weight vector $w_c(t)$ and its adjacent neighboring neuron vectors and then maps it to the network space using conformal mapping. Fig. 4 illustrates the transformation of input x into the reference vector r.

If a pattern x does not project inside any simplex of the CSSM mesh, it will be tuned based on the updating equation:

$$\begin{aligned} \Delta w_i &= \alpha_t h\left(\|x - w_c(t)\|\right)\left(x - w_i(t)\right), \\ w_i(t+1) &= w_i(t) + \Delta w_i. \end{aligned} \tag{7}$$

5. *Continuation.* Continue with step 2 until a satisfactory result is obtained. One epoch means that all patterns $x \in X$ have been selected once. Successful learning requires many epochs.

In Step 4 of the learning algorithm, the function $\mathcal{M}$ requires the use of conformal mapping to map simplex s in the input space to s' in the network space, see Figs. 4 and 5. The conformal mapping from simplex s to equilateral simplex s' can be approximated by means of Schwarz-Christoffel mapping [20][7][9].

The mapping function from the v-plane to the z-plane is given by

$$z = f_1(v) = a_1 + B_1 \int_0^v \frac{1}{\zeta^2} \prod_{i=1}^{3} \left(1 - \frac{v_i}{\zeta}\right)^{-\beta_i} d\zeta. \tag{8}$$

The mapping function from the v-plane to the z'-plane is given by

$$z' = f_2(v) = a_2 + B_2 \int_0^v \frac{1}{\zeta^2} \prod_{i=1}^{3} \left(1 - \frac{v_i}{\zeta}\right)^{-\gamma_i} d\zeta. \tag{9}$$

Since β_i and γ_i are known, a, B and v_i have to be solved in the above equations. Therefore, the mapping from simplex s to simplex s' is $z' = f_2(f_1^{-1}(z))$, where z is any point on s, and z' is its corresponding point on s'. Then, the reference vector r is computed using $r = n_c + proj(z')$. Function $proj(z')$ projects z' in the complex plane onto the network space. In this project, r is always normalized with the same magnitude as that of n_i, $|r| = |n_i| = 1$.

3. Deformation measure

We now review the conformality measure [21]. It can be used to express both the distribution error and topology preservation for the self-organizing process. It achieves better performance than the MSE

The procedure for mapping input pattern x to the reference vector r in the network space.

Figure 4.

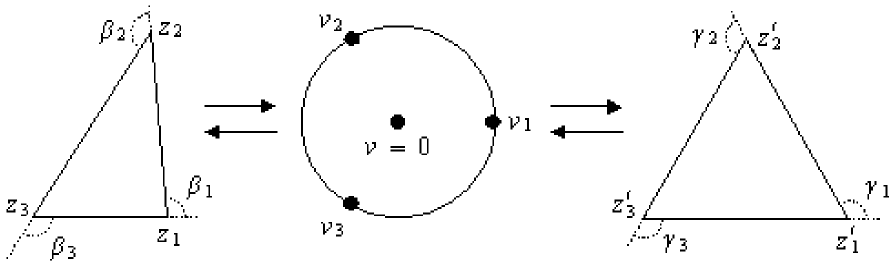

Figure 5.

Diagram of the relative input x' and the relative synapse v. x is the input pattern, w_c is the winning neuron weight, and w_i and w_j are the neighboring neuron weights. x', v_1, and v_2 form a 3D simplex.
Figure 6.

[4] or TPG [6] measure [21]. Although it is derived for the SOM, it is also applicable to the CSSM. To formulate it, we first define two vectors:

relative synapse vector, $v = w - w_c$;
relative input, $x' = x - w_c$.

Note that in this section v has a different meaning from those in Eqs. 8 and 9. Fig. 6 shows the topological representation of the synapse vectors. The topology formed through self-organization can be regarded as a collection of disjoint n-dimensional simplices. In this project, the pattern is in 3D and the network is intrinsic in 2D. Therefore, the simplex is 3D and is a 3-dimensional simplice.

¿From the CSSM synapse update equation, Eq. 5, we have

$$\begin{aligned} &w_c(t+1) = w_c(t) + \alpha h\left(d\left(\mathcal{M}(x), n_c\right)\right)\left(x - w_c(t)\right) \quad \text{for winning neuron and} \\ &w_i(t+1) = w_i(t) + \alpha h\left(d\left(\mathcal{M}(x), n_i\right)\right)\left(x - w_i(t)\right) \quad \text{for other neurons.} \end{aligned} \tag{10}$$

Then the update of relative synapse vector v, Δv, in the simplex is

$$\begin{aligned}\Delta v &= v\,(t+1) - v\,(t) = (w\,(t+1) - w_c\,(t+1)) - (w\,(t) - w_c\,(t)) \\ &= \alpha(h-1)x' - \alpha h v\,(t)\,, \end{aligned} \tag{11}$$

where h denotes $h(d(x', v)) = h(d(x, w))$. Note that the variables in the neighborhood function here are different from those in Eq. 5. To formulate the deformation measure in each adaptation step, the mapping function f is defined as

$$f(x', v\,(t)) = v\,(t+1) = v\,(t) + \Delta v = \alpha(h-1)x' + (1-\alpha h)v\,(t)\,. \tag{12}$$

Function f is the update equation for relative synapse v. We now introduce the Jacobian matrix J used to analyze function f. The Jacobian matrix can represent the derivative map of function f in a small neighborhood around certain point p [3]. The explicit definition of the derivative map is ignored here, but it can be thought of as a linear transformation that approximates function f near the point p, i.e., $f(p + \Delta p) = f(p) + J\Delta p$.

Let $f(x', v)$ be $(f_1, f_2, ..., f_i, ..., f_n)^T$, and let each component f_i be a function of $v = (v_1, ..., v_j, ..., v_n)^T$. Let us focus on each component of f, i.e.,

$$f_i(x', v) = \alpha(h-1)x'_i + (1-\alpha h)v_i, \quad i = 1, ..., n. \tag{13}$$

Here, we will use the Euclidean metric for d in the simplicial coordinate, that is $d(x', v) = \sum_{k=1}^{n}(x'_k - v_k)^2$. Note that the metric $d(x', v)$ should be normalized to work correctly. Hence, the partial derivatives of f, for $1 \le i, j \le n$, are

$$\begin{aligned}\frac{\partial f_i}{\partial v_i} &= -2\alpha\frac{\mathtt{d}h}{\mathtt{d}d}(x'_i - v_i)^2 + (1-\alpha h) = -2\alpha h'(x'_i - v_i)^2 + (1-\alpha h),\ \text{for } j = i\ ; \\ \frac{\partial f_i}{\partial v_j} &= -2\alpha\frac{\mathtt{d}h}{\mathtt{d}d}(x'_i - v_i)(x'_j - v_j) = -2\alpha h'(x'_i - v_i)(x'_j - v_j),\ \text{for } j \neq i. \end{aligned} \tag{14}$$

The derivative of h is

$$h' = \frac{\mathtt{d}h}{\mathtt{d}d} = \frac{\mathtt{d}\left(\exp\left(\frac{-d}{2\sigma^2}\right)\right)}{\mathtt{d}d} = -\frac{1}{2\sigma^2}\exp\left(\frac{-d}{2\sigma^2}\right). \tag{15}$$

Therefore, the Jacobian matrix, $\mathtt{d}f_v = A$, of function f is

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial v_1} & \frac{\partial f_1}{\partial v_2} & \cdots & \frac{\partial f_1}{\partial v_n} \\ \frac{\partial f_2}{\partial v_1} & \frac{\partial f_2}{\partial v_2} & \cdots & \frac{\partial f_2}{\partial v_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial v_1} & \frac{\partial f_n}{\partial v_2} & \cdots & \frac{\partial f_n}{\partial v_n} \end{bmatrix}$$

$$= \begin{bmatrix} -2\alpha h'(x_1 - v_1)^2 + (1-\alpha h) & \ldots & -2\alpha h'(x_1 - v_1)(x_n - v_n) \\ \vdots & \ddots & \vdots \\ -2\alpha h'(x_n - v_n)(x_1 - v_1) & \ldots & -2\alpha h'(x_n - v_n)^2 + (1-\alpha h) \end{bmatrix}. \tag{16}$$

Because matrix A is symmetric, every eigenvalue of A is real. Using the results in [21], the eigenvalues of matrix A are

$$\lambda_1 = 1 - \alpha h \quad \text{and} \quad \lambda_2 = -2\alpha h' \sum_{i=1}^{n} (x'_i - v_i)^2 + (1 - \alpha h) \tag{17}$$

with multiplicities $n-1$ and 1, respectively. If the Jacobian $J(v, f)$, the determinant of the Jacobian matrix $\mathtt{d}f_v = A$, is greater than zero, then the deformation of the mapping function f can be defined. Based on the above introduction, the three non-conformality measures are defined as follows:

1. The *deformation measure* : $Q(x', v) \equiv \sqrt{\frac{e_{\max}}{e_{\min}}}$,
2. The *non-conformality measure* : $K(x', v) \equiv \frac{\left(\sum_{i=1}^{n} \|\nabla f_i(v)\|^2\right)^{n/2}}{n^{n/2} J(v,f)}$,
3. The *deformation potential* : $E(x', v) \equiv \left(\sum_{i=1}^{n} \|\nabla f_i(v)\|^2\right)^{n/2} - n^{n/2} J(v, f)$.

In the deformation measure Q, $e_{\max}$ and $e_{\min}$ are the maximal and minimal eigenvalues of the Jacobian matrix A, respectively. The two distinct eigenvalues (λ_1, λ_2) of A, in Eq. 17, are all greater than zero. In addition, λ_2 is greater than or equal to λ_1:

$$\lambda_2 - \lambda_1 = -2\alpha h' \sum_{i=1}^{n} (x'_i - v_i)^2 \geqslant 0. \tag{18}$$

Above equation holds because $\alpha \in [0, 1)$ and $h' \leqslant 0$, where h is a monotonous decreasing function. Hence, the deformation measure for the CSSM is

$$Q(x', v) = \left(\frac{\lambda_2}{\lambda_1}\right)^{1/2} = \left(\frac{-2\alpha h'}{(1-\alpha h)} \sum_{i=1}^{n} (x'_i - v_i)^2 + 1\right)^{1/2}. \tag{19}$$

The value of Q is not less than 1. If $Q = 1$, the measure Q indicates that there is no deformation in the mapping function f.

In the non-conformality measure, a geometrical interpretation may give us a better sense of this criterion. The term $J(v, f)$ is the volume of the hyper-parallelpipe determined by the vectors $\nabla f_i(v)$, $i = 1, ..., n$. The term in the numerator, $\left(\sum_{i=1}^{n} \|\nabla f_i(v)\|^2\right)^{1/2}$, is the length of the diagonal in the hypercube formed by the n orthogonal vectors of length $\|\nabla f_i(v)\|$, $i = 1, ..., n$, and $\frac{\left(\sum_{i=1}^{n} \|\nabla f_i(v)\|^2\right)^{n/2}}{n^{n/2}}$ is the maximum volume of the hypercube inside a hypersphere with diameter $\left(\sum_{i=1}^{n} \|\nabla f_i(v)\|^2\right)^{1/2}$. K is always greater than 1 for any function f, where $J(v, f) > 0$. When $K = 1$, the mapping function f is conformal. After some derivations, the non-conformality measure in CSSM can be reduced to

$$K(x', v) \equiv \frac{\left[\left(-2\alpha h' \|x' - v\|^2 + 1 - \alpha h\right)^2 + (n-1)(1-\alpha h)^2\right]^{n/2}}{n^{n/2}(-2\alpha h' \|x' - v\|^2 + 1 - \alpha h)(1-\alpha h)^{n-1}}. \tag{20}$$

The non-conformality measure K in Eq. 20 may be infinity when its denominator is equal to or close to zero. This condition cannot be predicted at all in general.

The deformation potential E can measure the non-conformality without encountering this serious problem. After some derivations, the deformation potential in the CSSM is

$$\begin{aligned} E(x, v) = {} & \left[\left(-2\alpha h' \|x' - v\|^2 + 1 - \alpha h\right)^2 + (n-1)(1-\alpha h)^2\right]^{n/2} \\ & - n^{n/2}(-2\alpha h' \|x' - v\|^2 + 1 - \alpha h)(1-\alpha h)^{n-1}. \end{aligned} \tag{21}$$

The measures Q, K and E are all based on the individual sampled relative input, $x' = x - w_c$, and relative synapse $v = w - w_c$ of the neighboring neurons. To compute the network's overall performance, the individual deformation is averaged as follows:

$$\textit{Deformation measure of the whole network} : Q_{total} = \frac{1}{nP} \sum_{p=1}^{P} \sum_{i=1}^{n} Q(x_p, v_i),$$

$$\textit{Non-conformality measure of the whole network} : K_{total} = \frac{1}{nP} \sum_{p=1}^{P} \sum_{i=1}^{n} K(x_p, v_i),$$

$$\textit{Deformation potential of the whole network} : E_{total} = \frac{1}{nP} \sum_{p=1}^{P} \sum_{i=1}^{n} E(x_p, v_i),$$

where P is the total number of input data and n is the dimension of the simplex shown in Fig. 6. Furthermore, a *total non-conformality metric* [21] is introduced. It is the product of consecutive *non-conformality* measures

$$M_i = \prod_{t=1}^{T} K(x', v_i), \tag{22}$$

where T denotes the total number of the learning steps in the whole learning process. This metric indicates the accumulative deformation of the neuron i through the whole learning process.

4. Simulation

4.1. Process

In our simulation, 3D models were collected from the sample archive of the Cyberware company website [29], and the files were in the PLY format. The CSSM is capable of learning from scattered point data. Therefore, the source files were translated into point clouds to serve the input patterns in our simulation, see Fig. 7. The procedure for our simulation is described below.

1. 3D points were extracted out of the source file as raw input patterns, X. These points were scattered.

2. A CSSM network was initialized on a sphere by using an f frequency icosahedron, see Fig. 3.

3. The CSSM was trained to learn X until convergence was reached. The details of this step have been given in section 2.

4. Its conformality measures were computed.

The conformal mapping in function $\mathcal{M}$ was solved by using the MATLAB Schwarz-Christoffel toolbox [15]. We also applied a spherical network space to the conventional SOM model, which will be called SSOM in the following sections, for the purpose of comparison.

For the convenience of coding and debugging, we use MATLAB to implement our program. Solving Schwarz-Christoffel mapping, Eqs. 8 and 9, using the SC-map toolbox was a bottleneck in our program. About 40 minutes were required to complete one epoch with 3000 neurons and 20000 patterns on an Althon XP 2500+ with 768MB DDR RAM.

In our simulation, we used scattered data points as input patterns. (a) The original model after rendering, (b) the point cloud extracted from the original model.

Figure 7.

4.2. Results

In our simulation, we used two head models that came from the Cyberware company website [29]. Both models were extracted to obtain scattered data points and are shown in Fig. 8. The first model was a woman's head with a flaw beside her mouth, see Fig. 7(a). The second model was a female head scanned from a real person.

Figs. 9 and 10 show the CSSM and SSOM results obtained with different densities for surface reconstruction. Figs. 9(a,b) show the results obtained using the CSSM with 2562 neurons ($f = 16$). The number of learning epochs was set to 80, learning rate α was decreased from 0.01 to 0.001, and the variance σ decreased from 0.3 to 0.1. Figs. 9(c,d) show the results obtained using the SSOM with 2562 neurons ($f = 16$). Figs. 10(a,b) show the results obtained using the CSSM with 5762 neurons ($f = 24$). The number of learning epochs was 69, the learning rate α decreased from 0.01 to 0.001, and the variance σ decreased from 0.3 to 0.1. Figs. 10(c,d) show the results obtained using the SSOM with 5762 neurons ($f = 24$). All the learning criteria were set to be equal for the purpose of comparing these two methods. From the results obtained using these two methods, it shows that the CSSM learns smoother meshes than the SSOM does. The performance of the CSSM and SSOM is shown in Tables 1 and 2. The *deformation measure* Q_{total} and *non-conformality measure* K_{total} for both methods are close

The scattered data extracted from the PLY file. These point clouds are the input patterns in our simulations. (a) Venus model, 33587 data points, (b) Female model, 49463 data points.
Figure 8.

to 1, and the *deformation potential* E_{total} for both methods is close to zero. This shows that the results obtained using the CSSM and SSOM are close to conformal mapping.

Table. 1. The conformality measures of the CSSM and SSOM results with 2562 neurons. These data correspond to Fig. 9 (a) to (d).

The conformality measure	CSSM ($f = 16$)	SSOM ($f = 16$)
deformation measure Q_{total}	1.0188	1.0191
non-conformality measure K_{total}	1.001	1.0011
deformation potential E_{total}	0.006007	0.0061

Table. 2. The conformality measures of the CSSM and SSOM results with 5762 neurons. These data correspond to Fig. 10 (a) to (d).

The conformality measure	CSSM ($f = 24$)	SSOM ($f = 24$)
deformation measure Q_{total}	1.0088	1.0088
non-conformality measure K_{total}	1.0002	1.0002
deformation potential E_{total}	0.0012869	0.0012563

In Fig. 11, The CSSM and SSOM results for the second model are shown. In Figs. 11(a-c) show the results obtained using the CSSM with 2562 neurons ($f = 16$). The number of learning epochs was 88, the learning rate α decreased from 0.01 to 0.001, and the variance σ decreased from 0.3 to 0.1. Figs. 11(d-f) show the results obtained using the SSOM with 2562 neurons ($f = 16$). The number of learning epochs was 88, the learning rate α decreased from 0.01 to 0.001, and the variance σ decreased from 0.4 to 0.1. All the learning criteria were set to be equal for the purpose of comparison. Using simulation with this model, the SSOM failed to learn when the variance started at $\sigma = 0.3$. Hence, we started the variance at 0.4 ($\sigma = 0.4$). From the results shown in Fig. 11, the SSOM did not converge to smooth meshes and did not tighten the manifold. The performance of the CSSM and SSOM for the second model is shown in Table 3. The *non-conformality measure* K_{total} of the CSSM model was 1.175, which means that the map was a quasi-conformal mapping. The SSOM had worse performance than the CSSM for this model.

Table. 3. The conformality measures of the CSSM and SSOM results with 2562 neurons. These data correspond to Fig. 11 (a) to (f).

The conformality measure	CSSM ($f = 16$)	SSOM ($f = 16$)
deformation measure Q_{total}	1.1821	2.7646
non-conformality measure K_{total}	1.175	23.003
deformation potential E_{total}	1.0549	1949.3

The quality of the CSSM mesh is shown in Fig. 12. It shows the mesh angle distribution [24] of the Venus model in Figs. 9 and 10.

The adaptation procedure for the CSSM is applicable to the morphing problem. In this case we first use the CSSM to learn the first model

The results produced by the CSSM and SSOM model with 2562 neurons (vertices) using the CSSM (a,b) and using the SSOM (c,d). Comparing the forehead part of the CSSM and SSOM meshes, the mesh by the CSSM model is more regular than that by the SSOM model.

Figure 9.

and save the trained result. We then use this result as the initial mesh in a successive learning to learn the second model. We test this idea, and plot its result in Fig. 13. The number of learning epochs was 88, the learning rate α was 0.001 and the variance σ was decreased from 0.26 to 0.1. In Fig. 13, we show the shape changes from the first model toward the second model smoothly.

The results produced by the CSSM and SSOM model with 5762 neurons (vertices) using the CSSM (a,b) and using the SSOM (c,d). Comparing the forehead part of the CSSM and SSOM meshes, the mesh by the CSSM model is more regular than that by the SSOM model.
Figure 10.

To compare the shape difference between these two models, we calculate the total *non-conformality metric* M_i through the morphing process. The result is shown in Fig. 14.

5. Summary

This project presents a novel CSSM mesh. A conformal spherical self-organization method for parameterization of genus zero manifold mod-

The results produced by the CSSM and SSOM model for the second 3D model. All the figures are composed of the resulting meshes and rendered models. (a)-(c) CSSM results for 2562 neurons, (d)-(f) SSOM results for 2562 neurons. These results are obtained under the same parameters and show that CSSM gives a better mesh.

Figure 11.

The histogram of the mesh angle distribution. The Venus model with 2562 vertices (a) and with 5762 vertices (b) by the CSSM.

Figure 12.

The morphing results produced by the CSSM model. The CSSM starts to learn the model in Fig.8(a) toward the model in Fig.8(b). During learning, intermediate surface meshes are saved as in (b) to (k).

Figure 13.

els is presented. It differs from those by X. Gu *et al.* [13][12] and V. Surazhsky *et al.* [24]. X. Gu *et al.* proposed a method for finding global conformal parameterizations for surfaces which is derived from the gradient fields of conformal maps [13]. Their method needs mathematic techniques to process *zero points*. The proposed method is a deformable model and is fully automatic without *zero points*. V. Surazhsky *et al.* proposed the area-based smoothing method in vertex sampling for remeshing [24]. The proposed method has a similar function as remeshing, it finds the global conformal parameterization of the mesh. The neural network proposed by S.-W. Chen [5] utilizes the multilayer neural networks to learn the desired model. It does not necessarily have the conformal content.

The CSSM is intrinsically suitable for the morphing application, see Fig. 13, in its learning process. It is also suitable for studying morphological variability that is an important issue in many surface structure analyses, see Fig. 14. As for the long legs (sticking out the body) the proposed method needs extra-techniques. It is needed to include extra-nodes or links to accomplish such tasks. We did not develop such

The *total non-conformality metric* from the left model to the middle one. The metric values are plotted in the right column. The scale is normalized and double logged with different colors. The red area indicates a large difference while the blue area indicates a small difference.
Figure 14.

techniques. In CSSM the initial spherical mesh is extended toward the object surface without adding any node or link during self-organizing evolution. This CSSM mesh is capable of reconstructing a surface from unorganized points and defining a conformal mapping from a sphere to certain object's manifold. This mesh can resolve models with random noisy data. We are working on several applications shown below.

Hole recovery

The Venus model with 133446 sample points has a flaw near its right chin, see Fig. 15(a). We manually remove the data points of this flaw region, Fig. 15(b), and apply CSSM with 12962 vertices to fill this region, see Fig. 15(c). The learning rate α was set to 0.01 and the variance σ decreased from 0.2 to 0.01. CSSM can fill the missing region without any hole inside [10][1].

Mixed patterns

In this figure, the flaw region in the right chin of the Venus model is deleted and the CSSM can fill this hole. (a) The original Venus model using a mesh with 133446 vertices. (b) The input point cloud. The flaw region is removed. (c) The CSSM mesh with 12962 vertices.

Figure 15.

In this example, two models of male heads are mixed together. The total number of data points is 80507. The CSSM model has 12962 vertices. The results using CSSM are shown in Fig. 16. The CSSM mesh shows a new head that is similar to both of the two heads. The neighborhood variance σ is crucial in this example. The mesh in Fig. 16(d), σ decreased from 0.2 to 0.1, is smoother than that in Fig. 16(e), σ decreased from 0.2 to 0.01.

Model with random noise

One percent of uniform random noise is added in a male head model. The mesh by CSSM is in Fig. 17. Although the result has some imperfection, it recovers the head model that is not much affected by the noisy points.

References

1. Amenta N, Bern M, Kamvysselis M (1999) A new Voronoi-based surface reconstruction algorithm. Proceedings of SIGGRAPH, pp 415-421
2. Barhak J, Fischer A (2001) Parameterization and reconstruction from 3D scattered points based on neural network and PDE techniques. IEEE Transactions on Visualization and Computer Graphics, Vol 7, Issue 1, pp 1-16

Two male head models are mixed together. (a) The first male head model with 35091 vertices. (b) The second male head model with 30492 vertices. (c) The mixed point cloud. (d) The mesh by CSSM using $\sigma = 0.2\tilde{}0.1$ and $\alpha = 0.01$. (e) The mesh using $\sigma = 0.2\tilde{}0.01$ and $\alpha = 0.01$.

Figure 16.

3. Barrett O (1996) Elementary differential geometry, New York : Academic Press
4. Bauer HU, Pawelzik KR (1992) Quantifying the neighborhood preservation of self-organizing feature maps. IEEE Trans. Neural Networks, Vol 3, pp 570-578
5. Chen SW, Stockman GC, Chang KE (1996) SO dynamic deformation for building of 3-D models. IEEE Transactions on Neural Networks Vol 7, No 2, pp 374-387
6. Choi DI, Park SH (1994) Self-creating and organizing neural networks. IEEE Trans. Neural Networks, Vol 5, pp 561-575
7. Churchill RV, Brown JW (1984) Complex variables and applications, fourth edition, McGraw-Hill, New York
8. Curless B, Levoy M (1996) A volumetric method for building complex models from range images. Proceedings of SIGGRAPH, pp 303-312

A male head model with 1% random noise. CSSM can recover the model without topological error. (a) The head model with 1% noise. The model has 35091 points (green dots) and there are 351 uniform random noise points (red dots). (b) The mesh by CSSM using 12962 vertices. The rate α was set to 0.01 and the σ was decreased from 0.6 to 0.03. (c) The CSSM mesh same as (b).

Figure 17.

9. Driscoll TA (1996) A MATLAB toolbox for Schwarz–Christoffel mapping. ACM Trans. Math. Soft., Vol 22 , pp 168–186
10. Edelsbrunner H, Műcke DP (1994) Three-dimensional alpha shapes. ACM Transactions on Graphics, Vol 13, pp 43-72
11. Gotsman C, Gu X, Sheffer A (2003) Fundamentals of spherical parameterization for 3D meshes. ACM Transactions on Graphics, Vol 22
12. Gu X, Yau ST (2002) Computing conformal structures of surfaces. Communications in Information and Systems, Vol 2, No 2, pp 121-146, December
13. Gu X, Wang Y, Chan TF, Thompson PM, Yau ST (2004) Genus zero surface conformal mapping and its application to brain surface mapping. IEEE Transaction on Medical Imaging Vol 23, No 7, pp 1-8, July
14. Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W (1992) Surface reconstruction from unorganized points. Proceedings of SIGGRAPH, pp 71-78
15. Hoppe H (2002) Irregular to completely regular meshing in computer graphics. Invited Talk, 11th International Meshing Roundtable, Sandia National Laboratories, pp 141, September 15-18
16. Ivrissimtzis IP, Jeong WK, Seidel HP (2003) Using growing cell structures for surface reconstruction. International Conference on Shape Modeling and Applications, pp 78-88, Seoul, Korea
17. Kenner H (1976) Geodesic math and how to use It. Berkeley, University of California Press
18. Kohonen T (1982) Self-organized formation of topologically correct feature maps. Biological Cybernetics, Vol 43, pp 59

19. Levy B, Petitjean S, Ray N, Maillot J (2002) Least squares conformal maps for automatic texture atlas generation, In Computer Graphics, Proceedings of SIGGRAPH, Addison Wesley
20. Liou CY, Tai WP (1999) Conformal self-organization for continuity on a feature map. Neural Networks, Vol 12, pp 893-905
21. Liou CY, Tai WP (2000) Conformality in the self-organization network. Artificial Intelligence, Vol 116, pp 265-286
22. Ohmori K, Kunii TL (2001) Shape modeling using homotopy. Proceedings of International Conference on Shape Modeling and Applications, (SMI), pp 126-133, May 7-11, Genoa, Italy
23. Ritter H (1999) Self-organizing maps in non-euclidean spaces. Kohonen Maps, pp 97-108 , ed. Oja E, Kaski S
24. Surazhsky V, Gotsman C (2003) Explicit surface remeshing. Proceedings of Eurographics Symposium on Geometry Processing, pp 17-28, June
25. Tai WP (1997) Conformal self-organization. PhD. Thesis, Department of Computer Science and Information Engineering, National Taiwan University
26. Tai WP, Liou CY (2000) Image representation by self- organizing conformal network. The Visual Computer, Vol 16, issue 2, pp 91-105
27. Varady T, Martin RR, Cox J (1997) Reverse engineering of geometric models-an introduction. Computer-Aided Design, Vol 29, No 4, pp 255-268
28. Yu Y (1999) Surface reconstruction from unorganized points using self-organizing neural networks. Proceedings of IEEE Visualization, pp 61-64, San Francisco, CA
29. "http://www.cyberware.com/samples/index.html", Cyberware designs, manufactures, and sells standard and custom 3D scanning systems and software

Figure Legend

1. *figure 1:* (a) The network space. There are 11x11 neurons arranged uniformly in a rectangular plane$\in R^2$. (b) The input space. The location of each lattice node is represented by its corresponding neuron's weight vector $w_i \in R^3$.

2. *figure 2:* This figure shows that an $\mathcal{R}^2$ plane cannot wrap a surface with a genus zero manifold. The meshes in (c) and (d) are learned by SOM with $\mathcal{R}^2$ network space.

3. *figure 3:* Icosahedrons approximate spheres at different frequencies. From (a) to (f) at $f = 1$ to $f = 6$, respectively.

4. *figure 4:* The procedure for mapping input pattern x to the reference vector r in the network space.

5. *figure 5:* The conformal mapping from an arbitrary triangle to a unit disk and then to an equilateral triangle and vice versa.

6. *figure 6:* Diagram of the relative input x' and the relative synapse v. x is the input pattern, w_c is the winning neuron weight, and w_i and w_j are the neighboring neuron weights. x', v_1, and v_2 form a 3D simplex.

7. *figure 7:* In our simulation, we used scattered data points as input patterns. (a) The original model after rendering, (b) the point cloud extracted from the original model.

8. *figure 8:* The scattered data extracted from the PLY file. These point clouds are the input patterns in our simulations. (a) Venus model, 33587 data points, (b) Female model, 49463 data points.

9. *figure 9:* The results produced by the CSSM and SSOM model with 2562 neurons (vertices) using the CSSM (a,b) and using the SSOM (c,d). Comparing the forehead part of the CSSM and SSOM meshes, the mesh by the CSSM model is more regular than that by the SSOM model.

10. *figure 10:* The results produced by the CSSM and SSOM model with 5762 neurons (vertices) using the CSSM (a,b) and using the SSOM (c,d). Comparing the forehead part of the CSSM and SSOM meshes, the mesh by the CSSM model is more regular than that by the SSOM model.

11. *figure 11:* The results produced by the CSSM and SSOM model for the second 3D model. All the figures are composed of the resulting meshes and rendered models. (a)-(c) CSSM results for 2562 neurons, (d)-(f) SSOM results for 2562 neurons. These results are obtained under the same parameters and show that CSSM gives a better mesh.

12. *figure 12:* The histogram of the mesh angle distribution. The Venus model with 2562 vertices (a) and with 5762 vertices (b) by the CSSM.

13. *figure 13:* The morphing results produced by the CSSM model. The CSSM starts to learn the model in Fig.8(a) toward the model in Fig.8(b). During learning, intermediate surface meshes are saved as in (b) to (k).

14. *figure 14:* The *total non-conformality metric* from the left model to the middle one. The metric values are plotted in the right column. The scale is normalized and double logged with different colors. The red area indicates a large difference while the blue area indicates a small difference.

15. *figure 15:* In this figure, the flaw region in the right chin of the Venus model is deleted and the CSSM can fill this hole. (a) The original Venus model using a mesh with 133446 vertices. (b) The input point cloud. The flaw region is removed. (c) The CSSM mesh with 12962 vertices.

16. *figure 16:* Two male head models are mixed together. (a) The first male head model with 35091 vertices. (b) The second male head model with 30492 vertices. (c) The mixed point cloud. (d) The mesh by CSSM using $\sigma = 0.2\tilde{}0.1$ and $\alpha = 0.01$. (e) The mesh using $\sigma = 0.2\tilde{}0.01$ and $\alpha = 0.01$.

17. *figure 17:* A male head model with 1% random noise. CSSM can recover the model without topological error. (a) The head model with 1% noise. The model has 35091 points (green dots) and there are 351 uniform random noise points (red dots). (b) The mesh by CSSM using 12962 vertices. The rate α was set to 0.01 and the σ was decreased from 0.6 to 0.03. (c) The CSSM mesh same as (b).