A Sweep Surface Based on Bivariate B-spline Motion

Seung-Hyun Yoon*
Department of Multimedia Engineering, Dongguk University
Seoul 100-715, Republic of Korea
[e-mail: shyun@dongguk.edu]
*Corresponding author: Seung-Hyun Yoon

Received December 5, 2013; revised February 3, 2014; accepted February 24, 2014; published March 31, 2014

Abstract

We present a new method for generating sweep surfaces using bivariate B-spline motion. The sweep surface is defined as the trace of a single point under bivariate B-spline motion. Direct manipulation of the sweep surface is achieved by controlling its motion components while producing various editing effects. We demonstrate the effectiveness of our technique by modeling and deforming various three-dimensional shapes.

Keywords: Sweep surface modeling, B-spline motion, direct manipulation, NURBS, generalized cylinder.

This research was supported by the research program of Dongguk University and supported by the Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning(Grant No. 2012R1A1A1008908).

http://dx.doi.org/10.3837/tiis.2014.03.018

1. Introduction

Sweep surfaces and swept volumes are widely used in geometric modeling, computer-aided design and computer graphics [1, 2, 3]. The simplest sweeps produce surfaces of revolution and linear extrusions, and many engineering objects can be represented by Boolean combinations of these surfaces.

A general sweep can be considered as one-parameter family of motions $M(v) \in R^{4\times4}$ which describes the position, orientation and scaling of a local coordinate system in three-dimensional space. If each component $M(v)$ can be represented by a B-spline curve, then $M(v)$ is called a rational B-spline motion [4, 5]. A sweep surface generated by a rational B-spline motion $M(v)$ can be expressed as follows:

$$\mathbf{x}(u,v) = M(v)C(u), \tag{1}$$

where $C(u)$ is a cross-sectional curve defined in the local coordinate system.

A sweep surface $\mathbf{x}(u,v)$ is controlled by the motion components such as position, rotation and scaling, as well as by the curve $C(u)$ itself. This facility provides the designer with great conveniences in modeling and deforming a three-dimensional object intuitively. **Fig**. 1(a) shows a teapot model which consists of three sweep surfaces obtained from cylindrical sweep surfaces by simply controlling their motion components. This would be extremely tedious and time-consuming for freeform surfaces since lots of control points should carefully be positioned.

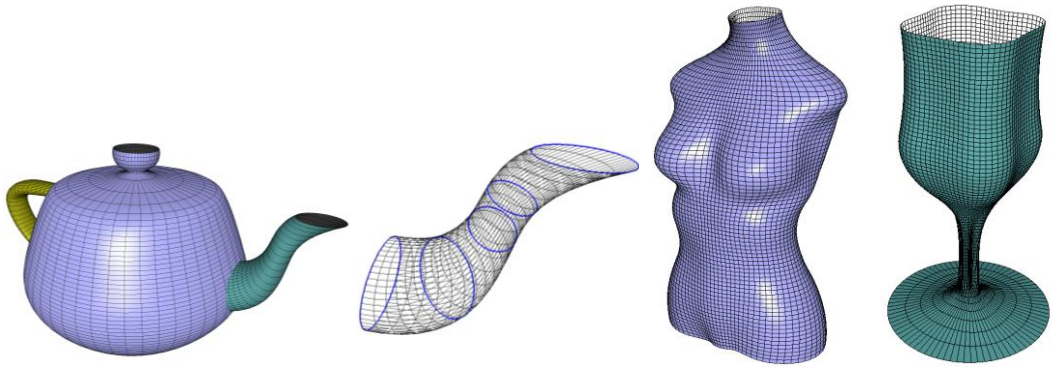
However, a sweep surface has some restrictions on cross-sectional shapes. In the simplest case, each iso-parametric curve $\mathbf{x}(u,v_0)$ in u direction will be a copy of $C(u)$ at different position, and potentially with a different scale and orientation (see **Fig**. 1(b)). It is definitely possible to use more flexible cross-sectional surfaces $C_v(u)$, which represent different cross-sectional shapes as the parameter v changes. This allows a wider range of shapes (see **Figs**. 1(c) and (d)), but it becomes more difficult to control the sweep surface because we should consider both of motion components and the cross-sectional object with increased parametric dimension [4, 6].

In this paper, we reformulate the traditional sweep surface by extending the rational B-spline motion $M(v)$ to a bivariate motion $M(u,v)$, while simplifying the cross-sectional curve $C(u)$ or surface $C_v(u)$ to a single point $\mathbf{p}$. A sweep surface $\mathbf{x}(u,v)$ is now the trace of $\mathbf{p}$ under the bivariate motion $M(u,v)$. The advantages of this formulation can be summarized as follows:

- The formulation provides the same modeling flexibility as a cross-sectional surface $C_v(u)$ under a one-parameter motion $M(v)$.
- Direct manipulation of an arbitrary surface point boils down to controlling a bivariate motion $M(u,v)$, and it is no longer necessary to consider a separate cross-sectional curve or surface, which greatly simplifies its mechanism compared to the existing methods [4, 6].

- A special type of editing such as twisting around an arbitrary surface point can easily be achieved by controlling the rotation surface of a bivariate motion $M(u,v)$.

The rest of this paper is organized as follows. In Section 2 we review some related recent work on sweep surfaces. In Section 3 we introduce the bivariate rational B-spline motion and show how to use it for generating sweep surfaces. We explain how to manipulate our sweep surfaces in Section 4, and some experimental results are presented in Section 5. Finally, we conclude this paper in Section 6.



(a) (b) (c) (d)

Fig. 1. Various sweep surfaces based on one-parameter motion $M(v)$: (a) a teapot model is constructed from three sweep surfaces generated by a circle $C(u)$ under $M(v)$, (b) the iso-parametric curves in the u direction (shown in blue) are simply instances of a circle with different positions, orientations and scales, (c) and (d) sweep surfaces generated by a variable cross-sectional surface $C_v(u)$ under $M(v)$.

2. Related Work

Johnstone and Williams [7] used motions to construct sweep surfaces, and Jüttler and Wagner [5] formulated motions from three B-spline curves, each of which determines trajectory, orientation and scaling of a local coordinate system. The resulting surfaces are compatible with commercial modeling systems based on the NURBS (nonuniform rational B-spline) representation. Chang et al. [4] further extended this technique to generalized cylinders, and proposed a technique for manipulating their shapes directly using a target-tracking method based on nonlinear inversion [8].

In sweep-based modeling, it is necessary to define a local coordinate system on a trajectory curve, and in general this can be supplied by the Frenet frame [1]; but Frenet frames flip at inflection points, where the second derivatives of the curve equations vanish. To solve this problem Jüttler and Wagner [5] used an additional B-spline curve to represent orientations on the trajectory curve and Wang et al. [9] introduced an efficient algorithm for computing a sequence of frames on a trajectory which minimize the rotations of the cross-section, and applied it to the construction of sweep surfaces. In general a sweep surface generated by a B-spline motion requires a relatively high degree. To reduce the degree of a B-spline motion while minimizing its distortion, Hyun et al. [10] presented an iterative optimization technique based on knot insertion and degree elevation. Rossignac and Vinacua [11] measured the quality of an affine motion by its steadiness, which is defined by the inverse of its ARA (average relative acceleration) and proposed the SAM (steady affine morph) algorithm for generating smooth affine motions.

Recently, sweeping has been reinterpreted as a control mechanism for the modification of existing shapes, rather than as a primary shape model. Lazarus et al. [12] introduced a technique for three-dimensional shape deformation using moving frames on a curve, and Singh and Fiume [13] extended this technique to asymmetric shape deformations. Hyun et al. [14] proposed a technique for creating and editing the shapes of human arms and legs using ellipsoidal sweeping; and they subsequently extended this technique to the modeling and deformation of a whole body using a sweep blending technique [15]. Lee et al. [16] showed how to change the shape of a human hand by combining a global sweep-based method with a local surface-based method. In this way they resolved the problems of controlling local details. Yoon and Kim [6] further extended this technique to the freeform deformation of three-dimensional objects and Park et al. [17] presented a real-time algorithm for realistically deforming 3D human blood vessels using sweep surfaces. These deformations [6, 17] are controlled by the interactions of the sweep surfaces while automatically detecting and avoiding interference among different sweep surfaces under deformation. The technique that we propose in this paper is similar, in as much as it can be used to deform existing objects, as well as to create new shapes.

3. Sweep Surface Based on Bivariate Motion

Let $\mathbf{t}(v) = (t_x(v), t_y(v), t_z(v), t_w(v))$ be a NURBS curve which describes a trajectory in three-dimensional space. The corresponding affine transformation can be represented by a 4×4 matrix $T(v)$ in a homogeneous coordinate system as follows:

$$T(v) = \begin{bmatrix} t_w(v) & 0 & 0 & t_x(v) \\ 0 & t_w(v) & 0 & t_y(v) \\ 0 & 0 & t_w(v) & t_z(v) \\ 0 & 0 & 0 & t_w(v) \end{bmatrix}. \tag{2}$$

In describing a one-parameter motion, a rotation curve $\mathbf{r}(v) = (r_w(v), r_x(v), r_y(v), r_z(v)) \in R^4$ can be used to represent unit quaternions $\mathbf{r}(v)/\|\mathbf{r}(v)\|$ in $S^3 = \{(w, x, y, z) \in R^4 \mid w^2 + x^2 + y^2 + z^2 = 1\}$ which specifies the orientations of a local coordinate system on the trajectory curve $\mathbf{t}(v)$. In this paper, we extend this rotation curve to become a rotation surface, as follows:

$$\mathbf{r}(u, v) = (r_w(u, v), r_x(u, v), r_y(u, v), r_z(u, v)), \tag{3}$$

where each component of $\mathbf{r}(u, v)$ is a B-spline function. The affine transformation corresponding to the rotation surface $\mathbf{r}(u, v)$ can be represented by the 4×4 matrix $R(u, v)$ in a homogeneous coordinate system as follows:

$$R(u,v)=\begin{bmatrix} r_{11}(u,v) & r_{12}(u,v) & r_{13}(u,v) & 0 \\ r_{21}(u,v) & r_{22}(u,v) & r_{23}(u,v) & 0 \\ r_{31}(u,v) & r_{32}(u,v) & r_{33}(u,v) & 0 \\ 0 & 0 & 0 & \delta(u,v) \end{bmatrix}, \tag{4}$$

where $\delta = r_w^2 + r_x^2 + r_y^2 + r_z^2, r_{11} = r_w^2 + r_x^2 - r_y^2 - r_z^2, r_{12} = 2(r_x r_y - r_w r_z), r_{13} = 2(r_x r_z + r_w r_y)$, $r_{21} = 2(r_x r_y + r_w r_z), r_{22} = r_w^2 - r_x^2 + r_y^2 - r_z^2, r_{23} = 2(r_y r_z - r_w r_x), r_{31} = 2(r_x r_z - r_w r_y), r_{32} = 2(r_y r_z + r_w r_x)$, and $r_{33} = r_w^2 - r_x^2 - r_y^2 + r_z^2$.

Similarly, a scaling surface with two parameters $\mathbf{s}(u,v) = (s_x(u,v), s_y(u,v), s_z(u,v), s_w(u,v))$ can be defined, and the corresponding 4×4 matrix $S(u,v)$ is

$$S(u,v)=\begin{bmatrix} s_x(u,v) & 0 & 0 & 0 \\ 0 & s_y(u,v) & 0 & 0 \\ 0 & 0 & s_z(u,v) & 0 \\ 0 & 0 & 0 & s_w(u,v) \end{bmatrix}. \tag{5}$$

Finally, the bivariate motion $M(u,v)$ can be defined as follows:

$$M(u,v) = T(v)R(u,v)S(u,v) = \begin{bmatrix} s_x t_w r_{11} & s_y t_w r_{12} & s_z t_w r_{13} & s_w t_x \delta \\ s_x t_w r_{21} & s_y t_w r_{22} & s_z t_w r_{23} & s_w t_y \delta \\ s_x t_w r_{31} & s_y t_w r_{32} & s_z t_w r_{33} & s_w t_z \delta \\ 0 & 0 & 0 & s_w t_w \delta \end{bmatrix}. \tag{6}$$

Since all the elements of $M(u,v)$ are products of B-spline functions, $M(u,v)$ can be represented as follows:

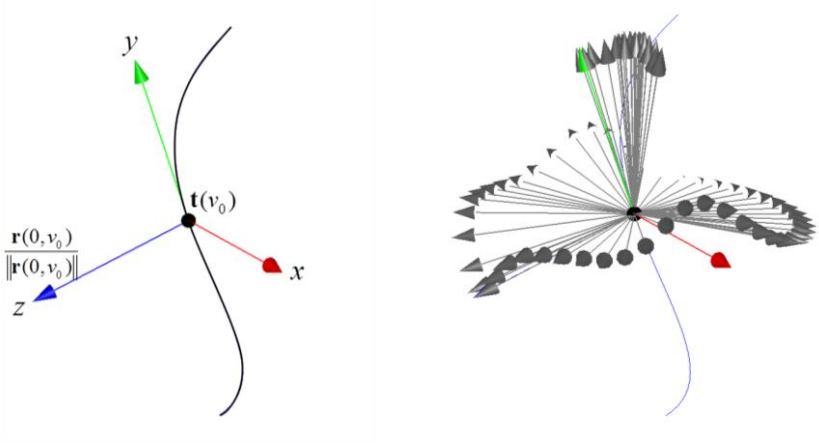
$$M(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} A_{ij} N_i^p(u) N_j^q(v), \tag{7}$$

where A_{ij} are control matrices and $N_i^p(u)$ and $N_j^q(v)$ are the B-spline basis functions of degrees p and q respectively.

Fig. 2 shows the rotation components of a bivariate motion $M(u,v)$. The position $\mathbf{t}(v_0)$ and orientation $\mathbf{r}(0,v_0)/\|\mathbf{r}(0,v_0)\|$ of a local coordinate system on the trajectory curve $\mathbf{t}(v_0)$ are shown in **Fig. 2(a)**, and the y and z-axes of the local coordinate system $M(u,v_0)$, which are continuously changing over the parametric interval $0 \le u \le 1$, are shown in **Fig. 2(b)**. A sweep surface can now be generated from the bivariate motion $M(u,v)$ as follows:

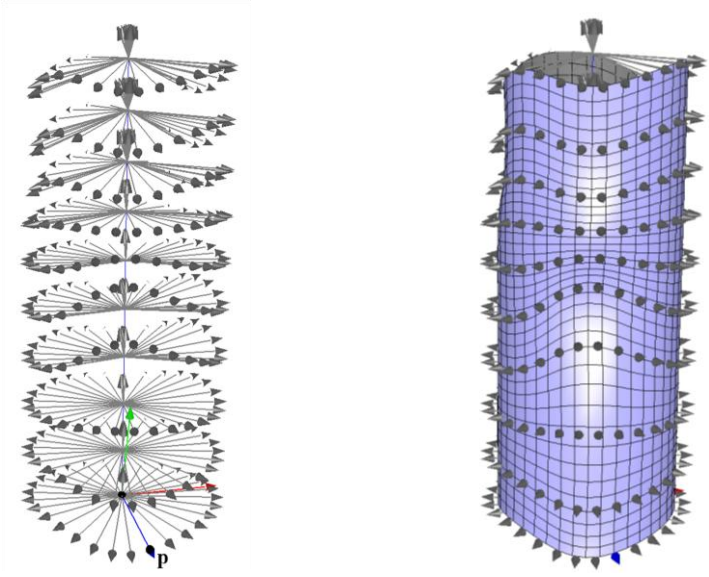
$$\mathbf{x}(u,v) = M(u,v)\mathbf{p}, \tag{8}$$

where $\mathbf{p} = \begin{bmatrix} p_x & p_y & p_z & 1 \end{bmatrix}^T$ is a point in a local coordinate system. For example, **Fig. 3(b)** shows the sweep surface generated by the point $\mathbf{p} = \begin{bmatrix} 0 & 0 & 1 & 1 \end{bmatrix}^T$ on z -axis under a bivariate motion $M(u,v)$ shown in **Fig. 3(a)**. Note that the sweep surface $\mathbf{x}(u,v)$ has different cross-sectional shapes depending on the parameter v , which is equivalent to the result of sweeping cross-sectional surfaces $C_v(u)$ under one-parameter motion $M(v)$.



(a) (b)

Fig. 2. Bivariate motion: (a) the local coordinate system $M(0,v_0)$ for the parameters $(0,v_0)$; and (b) the continuous local coordinate system $M(u,v_0)$, for $0 \le u \le 1$.



(a) (b)

Fig. 3. Creating a sweep surface: (a) a bivariate motion and a point $\mathbf{p}$ on the z -axis; (b) the resulting sweep surface $\mathbf{x}(u,v)$ with different iso-parametric curves in u direction.

4. Direct Manipulation

A curve, surface or solid can be used to generate a sweep surface using a one-parameter motion. This arrangement provides lots of ways of editing and manipulating sweeps: by editing the trajectory, rotation or scaling curve, or the object that is being swept. However, the existence of so many possibilities can complicate the provision of an appropriate mechanism of manipulating a sweep surface. In some previous approaches [4, 6], the edited displacements indicated by the user are interpreted as changes to the motion curves as well as to the object being swept.

Because we simplify the object being swept to a single point, the problem of editing and manipulating the sweep surfaces can be reformulated as one of controlling a bivariate motion. We will now show how this can be done simply and efficiently. We will start by introducing a method of editing each component of the bivariate motion, and then extend this method to produce various editing effects by specifying the way in which edited displacements provided by the user affect each component.

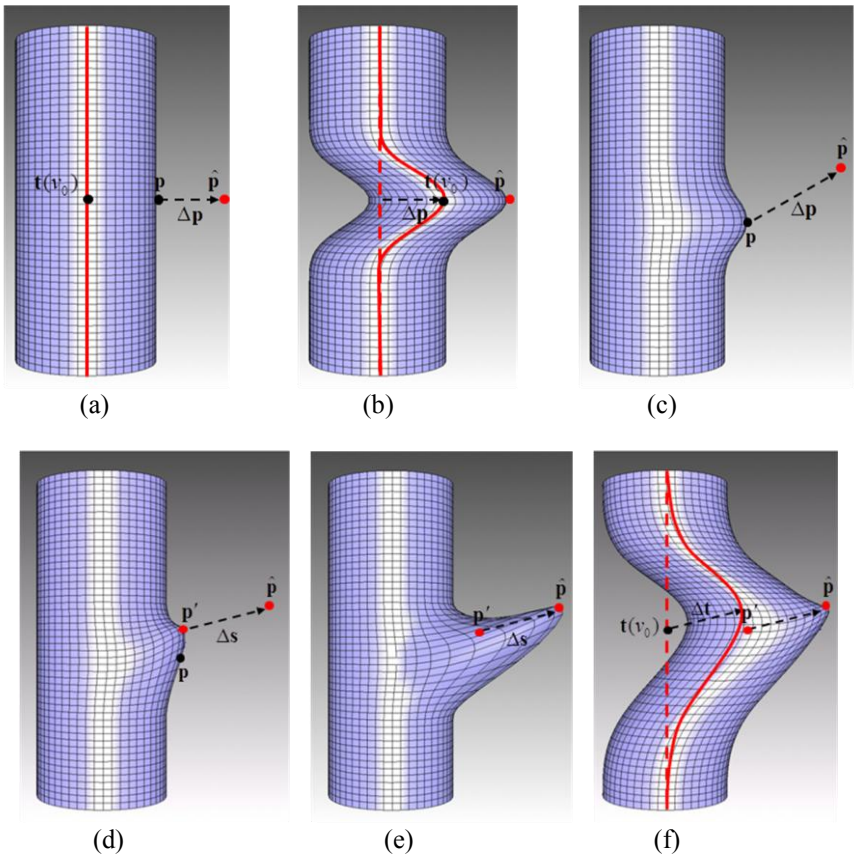


Fig. 4. Editing a sweep surface: (a), (c) a point $\mathbf{p}$ is selected and displaced by $\Delta\mathbf{p}$; (b), (d), (e) and (f) the edited sweep surfaces by controlling different motion components.

In **Fig. 4(a)**, the user selects an arbitrary point $\mathbf{p} = \mathbf{x}(u_0, v_0)$ on a sweep surface and moves it to a new position $\hat{\mathbf{p}}$. The specified displacement $\Delta\mathbf{p} = \hat{\mathbf{p}} - \mathbf{p}$ constitutes a translation $\Delta\mathbf{t}$ which moves a point $\mathbf{t}(v_0)$ on the trajectory curve to the position $\mathbf{t}(v_0) + \Delta\mathbf{t}$. This translation

can be achieved by direct manipulation [2, 18] of the NURBS curve. **Fig. 4(b)** shows the modified sweep surface by manipulating the trajectory curve.

Let $\mathbf{q} = \mathbf{r}(u_0, v_0) / \|\mathbf{r}(u_0, v_0)\| \in S^3$ be a selected orientation of local coordinate system on the trajectory curve $\mathbf{t}(v_0)$. If the sweep surface is generated by the point $\mathbf{p} = [0 \quad 0 \quad 1 \quad 1]^T$ on the z-axis, then $\mathbf{q}$ will correspond to the blue coordinate system in **Fig. 5**. To derive a rotation $\Delta\mathbf{r} \in R^4$ from $\Delta\mathbf{p} \in R^3$, we compute the rotational displacement $\Delta\mathbf{q} \in S^3$ as follows:

$$\Delta\mathbf{q} = \left(\cos\frac{\theta}{2}, \sin\frac{\theta}{2}\hat{\omega} \right), \tag{9}$$

where $\hat{\omega}$ (shown in green in **Fig. 5**) is the axis of rotation, which is the normalized cross-product of $\mathbf{p} - \mathbf{t}(v_0)$ and $\hat{\mathbf{p}} - \mathbf{t}(v_0)$ and θ is the angle between these vectors. **Fig. 5** shows how $\hat{\omega}$ and θ are derived from $\mathbf{p}$ and $\hat{\mathbf{p}}$.

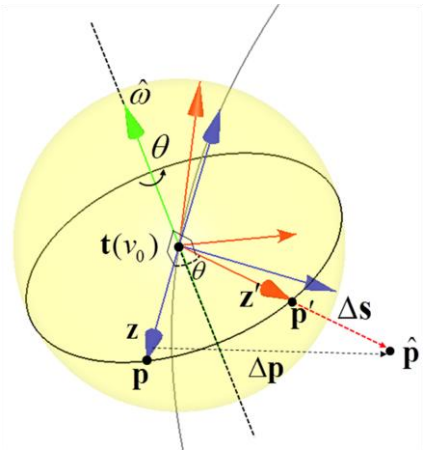


Fig. 5. Rotation and scaling components of an edited displacement.

Controlling rotation surface involves more computations. It does not make sense to simply move a point $\mathbf{r}(u_0, v_0)$ on rotation surface to the new position $\mathbf{r}(u_0, v_0) + \Delta\mathbf{q}$ in R^4 since $\Delta\mathbf{q}$ is in S^3. The new orientation $\hat{\mathbf{q}} \in S^3$ (shown in red in **Fig. 5**) of $\mathbf{q}$ is obtained by rotating the selected local coordinate $\mathbf{q}$ by $\Delta\mathbf{q}$ as follows:

$$\hat{\mathbf{q}} = \Delta\mathbf{q} \cdot \mathbf{q}, \tag{10}$$

where the dot $(\cdot)$ means the quaternion multiplication. Note that the $\Delta\mathbf{q}$ is computed in the global coordinate system rather than the local coordinate system $\mathbf{q}$, and thus it should be multiplied on the left side of $\mathbf{q}$. Then, the rotation $\Delta\mathbf{r} \in R^4$ can be computed as follows:

$$\Delta\mathbf{r} = \hat{\mathbf{q}} - \mathbf{q} = \Delta\mathbf{q}\frac{\mathbf{r}(u_0, v_0)}{\|\mathbf{r}(u_0, v_0)\|} - \mathbf{r}(u_0, v_0) = \left(\Delta\mathbf{q} - \|\mathbf{r}(u_0, v_0)\| e\right)\frac{\mathbf{r}(u_0, v_0)}{\|\mathbf{r}(u_0, v_0)\|}, \tag{11}$$

where $e = (1,0,0,0) \in S^3$ is the identity for quaternion multiplication. Finally, we apply NURBS manipulation [2, 18] to the rotation surface $\mathbf{r}(u_0, v_0)$ so that it passes through the new position $\mathbf{r}(u_0, v_0) + \Delta\mathbf{r} \in R^4$ and acquires desired orientation $\hat{\mathbf{q}}$. **Fig. 4(c)** shows the

specified displacement and **Fig. 4 (d)** shows the result of editing a sweep surface using this technique, where the iso-parametric curves in the u direction depend on v. In **Fig**. 4 (d), we can see that the surface point $\mathbf{p}$ moves to the new position $\mathbf{p}'$ rather than the target position $\hat{\mathbf{p}}$. For completeness, we compute a scaling $\Delta\mathbf{s} = \hat{\mathbf{p}} - \mathbf{p}'$ (see **Fig. 5**) and apply NURBS manipulation [2, 18] to the scaling surface $\mathbf{s}(u_0, v_0)$ so that it can pass through the point $\hat{\mathbf{p}}$, as shown in **Fig. 4(e)**. Alternatively, we can compute translation $\Delta\mathbf{t} = \hat{\mathbf{p}} - \mathbf{p}'$ and control the trajectory curve. **Fig. 4(f)** shows a different result of editing a sweep surface using translation $\Delta\mathbf{t}$. Note that editing the scaling surface influences local regions of the sweep surface, whereas editing the trajectory makes global changes.

In general, a specified displacement $\Delta\mathbf{p}$ can be decomposed into translation, scaling and rotation components, which can be used to edit a sweep surface. **Fig. 5** shows how a displacement $\Delta\mathbf{p}$ is decomposed into a rotation component $\Delta\mathbf{r}$ and a scaling component $\Delta\mathbf{s}$. As shown in **Fig. 4(f)**, the scaling component $\Delta\mathbf{s}$ can be replaced by a translation for producing a different editing effect.

5. Experimental Results

We implemented our technique in C++ on an Intel Core2 Duo 3.06GHz CPU with a 3GB main memory and an NVIDIA GeForce 8800 GS. **Fig. 6** shows a cylindrical sweep surface being edited by controlling the bivariate motion. **Fig. 6(a)** shows the initial sweep surface and the displacement of four points. **Fig. 6 (b)** shows a propeller shape created by these displacement, where each blade of the propeller is created by editing the scaling surface $\mathbf{s}(u, v)$. **Fig. 6(c)** shows the further displacement of two points on this propeller model. The rotation surface $\mathbf{r}(u, v)$ is changed by the rotations computed from these two displacements, and the new surface is shown in **Fig. 6(d)**. **Fig. 6(e)** shows a direct manipulation of the rotation surface $\mathbf{r}(u, v)$, with the result in **Fig. 6(f)**. Whereas it is quite difficult to twist part of a surface around a particular point using existing techniques [1, 2, 4, 6], our new technique readily provides this functionality by allowing changes to the rotation surface $\mathbf{r}(u, v)$ of the bivariate motion. For example, **Fig. 7** shows the advantage of our technique compared to the traditional NURBS representation commonly used in geometric modeling and computer graphics. **Fig. 7(b)** shows a NURBS surface represented by control points and **Fig. 7(c)** show its editing result which is obtained by manually selecting 35 control points and carefully repositioning them, whereas our technique can produce the similar effect by simply manipulating the selected orientation of the surface as shown in **Fig. 7(a)**.

Although our technique provides an intuitive control mechanism for editing sweep surfaces, the bivariate motion $M(u, v)$ in **Equation** (6) requires a relatively high degree since all the elements of $M(u, v)$ are products of B-spline functions. In general $M(u, v)$ has degree $d_t + 2 \times d_r + d_s$ for u and v directions, where d_t, d_r and d_s are degrees of a trajectory curve, a rotation surface and a scaling surface, respectively. For experimental results, we used cubic trajectory curves, linear rotation surfaces and quadratic scaling surfaces, which resulted in sweep surfaces of degree 7. Some sophisticated techniques [10, 11] can be applied for enhancing the generated sweep surfaces or reducing their degrees as necessary.

Fig. 8 shows the processes of modeling and deformations of a three-dimensional fish model. Similar to the propeller model, we start with a cylindrical sweep surface. **Figs. 8(a)-(f)** show

the intermediate stages of creating a fish model, where the direct manipulation techniques presented in Section 4 are used repeatedly. **Figs. 8(g)-(i)** show the results of various deformations of the fish model. These deformations are simply achieved by controlling rotation surface $\mathbf{r}(u,v)$ of the bivariate motion, and the textured fish models are shown in **Figs. 8(j)-(l)**.

Similarly, **Fig. 9** shows the result of modeling and deformations of a flower and flowerpot. For all models in experimental result, the whole design process took only a couple of minutes, starting from cylindrical sweep surface and manipulating surface points and motion components to the final three-dimensional shapes, and our technique shows a real-time performance for all models.

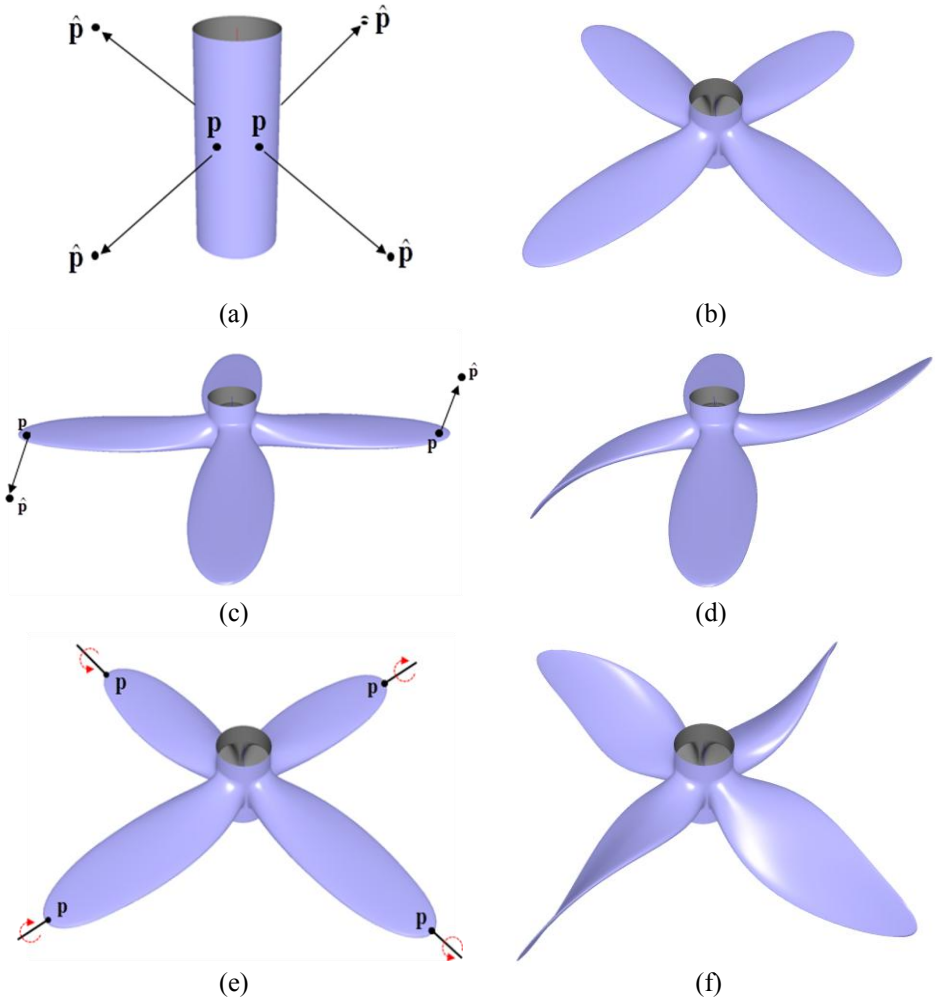
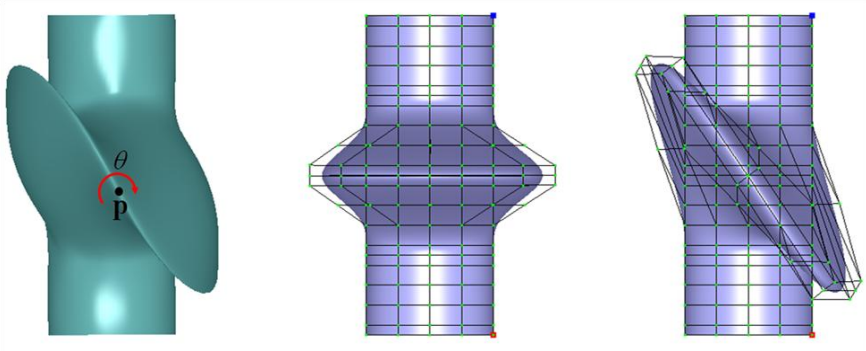
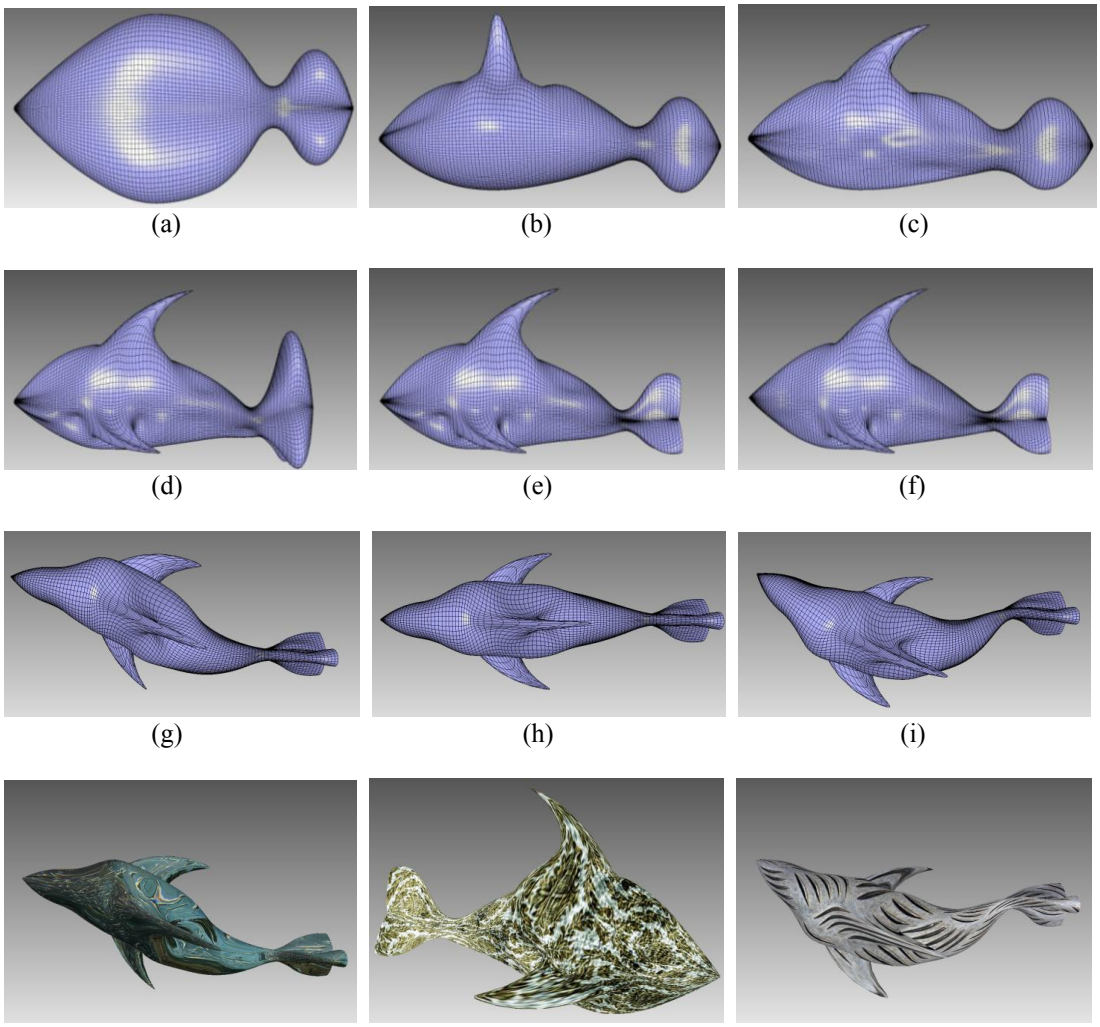


Fig. 6. Results of editing sweep surfaces based on two-parameter motions.



(a) (b) (c)

Fig. 7. Comparison to a NURBS model: (a) a sweep surface edited by the direct manipulation of rotation surface, (b) a NURBS surface model, (c) 35 control points are repositioned carefully for producing similar editing effect in (a).



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Fig. 8. Bivariate motion based modeling and deformations: (a)-(f) modeling processes of a fish model; (g)-(i) deformations by controlling rotation surface; (j)--(l) textured fish models.

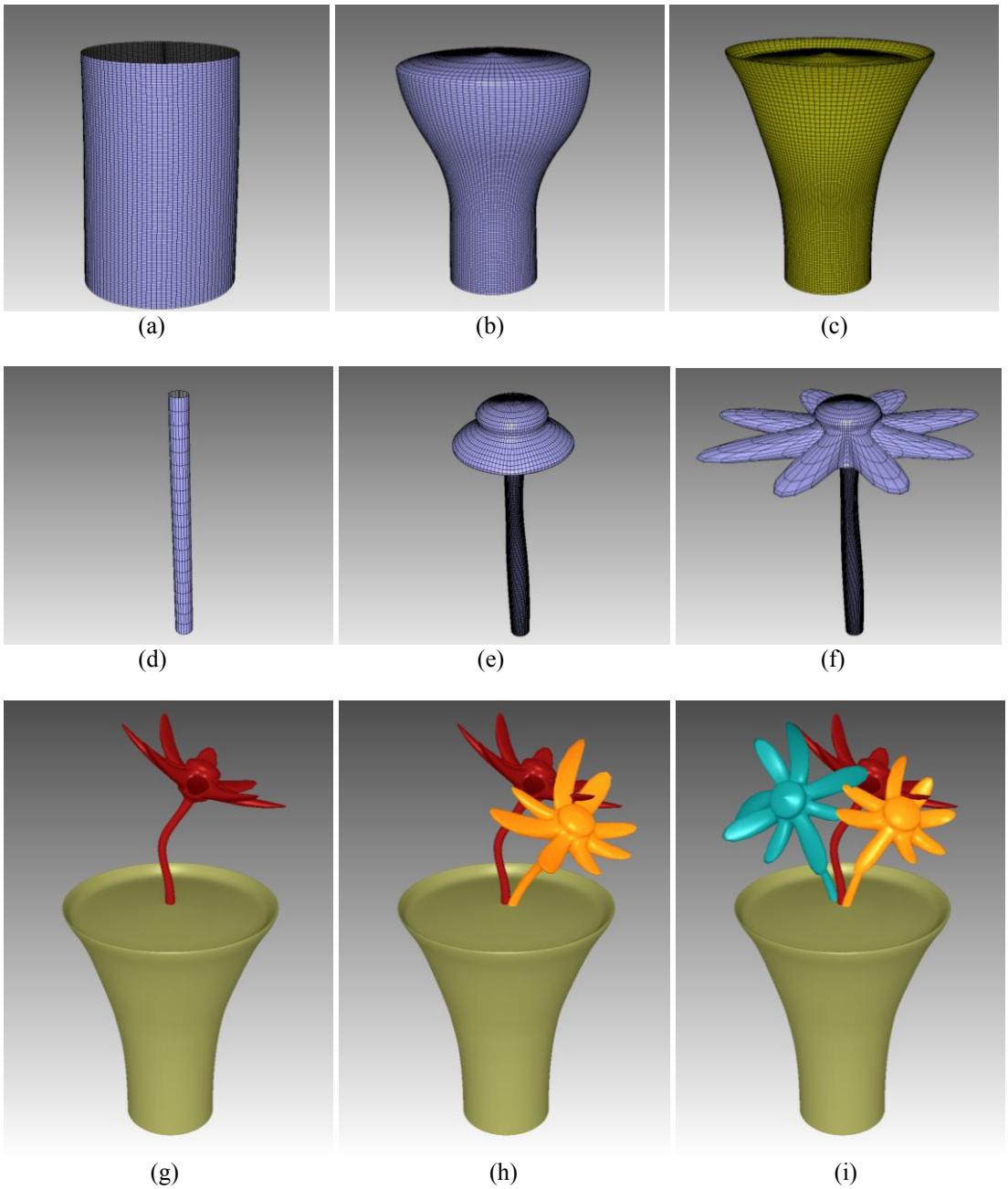


Fig. 9. Bivariate motion based modeling and deformations: (a)-(c) modeling processes of a flower; (d)-(f) modeling processes of a flowerpot; (g)-(i) potted flowers.

5. Conclusion

We have created a bivariate motion by extending the rotation and scaling curves used in standard sweeping techniques to surfaces, and used this technique to generate sweep surfaces. Since our sweep surface is the trace of a single point under a bivariate motion, editing and manipulating the shape of the surface becomes the problem of controlling the bivariate

motion.

To solve this problem, we have introduced a simple and intuitive technique in which a user-specified displacement is interpreted as changes to several components of the motion, and these linkages determine how the sweep surface is edited. We constructed various three-dimensional models and deforming them to demonstrate our technique, and showed that editing the rotation surface provided an easy way of controlling the sweep surface compared to the popular NURBS surface.

In future work, we plan to extend the motion components to multi-resolution representations and to investigate multi-resolution editing of sweep surfaces. We are also planning to reduce the degree of a sweep surface for obtaining its compact representation.

References

[1] Farin, G., *Curves and Surfaces for CAGD*, 5th Edition, Academic Press, 2002.

[2] Piegl, L., Tiller, W., *The NURBS Book*, 2nd Edition, Springer, 1997.

[3] Salomon, D., *Curves and Surfaces for Computer Graphics*, Springer, 2006.

[4] Chang, T.-I., Lee, J.-H., Kim, M.-S., Hong, S.-J., "Direct manipulation of generalized cylinders based on B-spline motion", *The Visual Computer*, vol. 4, no. 5, pp. 228–239, 1998. Article (CrossRef Link).

[5] Jüttler, B., Wagner, M.-G., "Computer aided design with spatial rational B-spline motions", *ASME Journal of Mechanical Design*, vol. 118, no. 2, pp. 193–201, 1996. Article (CrossRef Link).

[6] Yoon, S.-H., Kim, M.-S., "Sweep-based freeform deformations", *Computer Graphics Forum*, vol. 25, no. 3, pp. 487–496, 2006. Article (CrossRef Link).

[7] Johnstone, J.,Williams, J., "A rational model of the surface swept by a curve", *Computer Graphics Forum*, vol. 14, no. 3, pp. 77–88, 1995. Article (CrossRef Link).

[8] Kyung, M.-H., Kim, M.-S., Hong, S.-J., "A new approach to through-the-lens camera control", *Graphical Models and Image Processing*, vol. 58, no. 3, pp. 262–285, 1996. Article (CrossRef Link).

[9] Wang, W., Jüttler, B., Zheng, D., Liu, Y., "Computation of rotation minimizing frames", *ACM Transactions on Graphics*, vol. 27, no.1, article no. 2, 2008. Article (CrossRef Link).

[10] Hyun, D.-E., Jüttler, B., Kim, M.-S., "Minimizing the distortion of affine spline motions", *Graphical Models*, vol. 64, no. 2, pp. 128-144, 2002. Article (CrossRef Link).

[11] Rossignac, J., Vinacua, A., "Steady affine motions and morphs", *ACM Transactions on Graphics*, vol. 30, no. 5, article no. 116, 2011. Article (CrossRef Link).

[12] Lazarus, F., Coquillart, S., Jancène, P., "Axial deformations: an intuitive deformation technique", *Computer-Aided Design*, vol. 26, no. 8, pp. 607–613, 1994.

[13] Singh, K., Fiume, E., "Wires: a geometric deformation technique,"*in Proc. of ACM SIGGRAPH*, pp. 405–414, 1998. Article (CrossRef Link).

[14] Hyun, D.-E., Yoon, S.-H., Kim, M.-S., Jüttler, B., "Modeling and deformation of arms and legs based on ellipsoidal sweeping", *In Proc. of Pacific Graphics*, pp. 204–212, 2003.

[15] Hyun, D.-E., Yoon, S.-H., Chang, J.-W., Seong, J.-K., Kim, M.-S., Jüttler, B., "Sweep-based human deformation", *The Visual Computer*, vol. 21, no. 8–10, pp. 542–550, 2005. Article (CrossRef Link).

[16] Lee, J., Yoon, S.-H., Kim, M.-S., "Realistic human hand deformation", *Computer Animation and Virtual Worlds*, vol. 17, no. 3–4, pp. 479–489, 2006. Article (CrossRef Link).

[17] Park, J., Shim, M., Park, S.-Y., Kang, Y., Kim, M.-S., "Realistic deformation of 3D human blood vessels", *Computer Animation and Virtual Worlds*, vol. 24, no. 3-4, pp. 317-325, 2013. Article (CrossRef Link).

[18] Elber, G., "Multi-resolution curve editing with linear constraints", *The Journal of Computing & Information Science in Engineering*, vol. 1, no. 4, pp. 347–355, 2001. Article (CrossRef Link).

Authors' short biographies:

Seung-Hyun Yoon received the BS degree in mathematics from Hanyang University in 2001 and the PhD degree in computer science and engineering from Seoul National University in 2007. He is currently an associative professor of the Department of Multimedia Engineering, Dongguk University. His research interests are in computer graphics and geometric modeling.