Skeleton Subspace Deformation with Displacement Map

Feng Liu,[1] Ronghua Liang[1,2] and Dahai Ye[1]

[1] College of Computer Science, Zhejiang University, Hangzhou, P.R.China
[2] Institute of VR and Multimedia, Hangzhou Institute of Electronics Engineering, Hangzhou, P.R.China

Abstract

Skeleton Subspace Deformation (SSD) is a novel Free Form Deformation technology (FFD), and plays an important role in character animation, interactive game and other fields. In this paper, we present a new layered algorithm of SSD. After a preliminary conventional SSD, a displacement map is constructed based on the pose of the skeleton and the original surface. And then synthesize the displacement map with the defectively deformed surface and thus solve the inherent problems of conventional SSD, called "collapse" and "pinch". The experiment shows the effectiveness of this algorithm to create the realistic skin for an animated character.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Curve, surface, solid, and object representations I.3.7 [Computer Graphics]: Animation

1. Introduction

Free form deformation is an important technique in computer animation, interactive game and other fields. Shape interpolation[1] and Skeleton Subspace Deformation[2,3] are the most favorable FFD techniques in practice. Shape interpolation is particularly popular in facial animation[1,4]. The surface vertices are created through blending multiple spatial corresponding vertices on key shapes. The interpolated shapes, however, can not guarantee against collisions between skins (shapes) and skeletons, for most human motions are driven by the hierarchical skeleton motion data[5]. Aiming at creating skeleton based skins, researchers[2,3] proposed a simple but novel technique called Skeleton Subspace Deformation (SSD), in which the surface vertices are moved in response to several associated limbs. SSD has been widely used in commercial software packages[6,7] in part due to the hardware support.

However, problems called "collapse" and "pinch" occur and damage the realism of skins, when the limbs are rotated or twisted with large angles. Those defects are inherent in the process of SSD, which will be addressed in Section 2.1. In this paper, we propose a layered method to rectify these flaws: Create a deficient surface using SSD, construct a displacement map based on the pose and the reference surface and then synthesize it with the preliminary deformed surface.

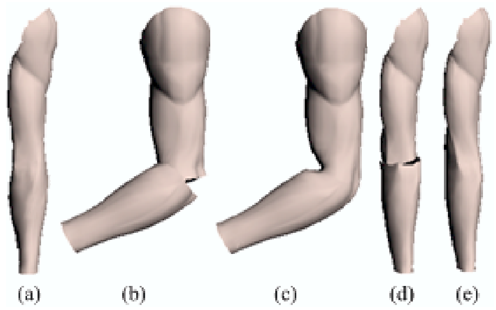

Figure 1: *Application of SSD. (a) is the initial arm,(b),(d) are the arms with the forearm rigidly rotating and twisting respectively, and (c), (e) are the corresponding SSD deformed arms.*

The remainder of this paper is organized as follows: in the next section, we give a brief review on recent work on SSD. In Section 3, we elaborate the modelling of the displacement map. We discuss the experimental result in Section 4 and conclude the paper in Section 5.

2. Previous Work

SSD suffers from its inherent problems, "collapse" and "pinch", though it has been the most widely used approach

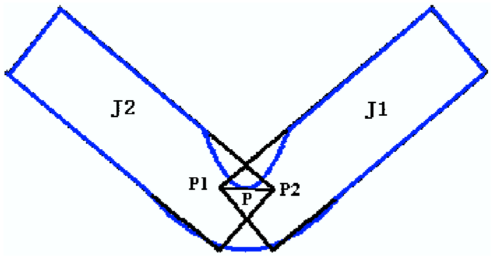

Figure 2: *SSD algorithm. P1 and P2 are the new positions of V according to the rigidly transformation of limb J1 and J2 respectively. P is the deformed position of V lying on the line P1P2.*

to skeleton-driven animation, and adopted in most commercial software. Recently, fruitful work has been conducted to relieve and solve these problems.

2.1. Skeleton subspace deformation

Skeleton-driven animation, in which character is driven by the hierarchical motion data according to its embedded skeleton, plays a dominant role in character animation and interactive game, especially recently when motion capture systems have been widely used. As shown in Figure 1(b) and (d), "crevasse" and "puncture" occur in the connections when skeleton is moved rigidly according to the hierarchical motion data. To eliminate such problems, a simple but novel method, called "Skeleton Subspace Deformation" (SSD) or "Vertex Blending"[2,3,8], is proposed, in which the vertices of an object's surface are moved interactively in response to some control mechanism. The number of vertices and their inter-connectivity do not change. The formulization of SSD is well described in Lewis[8] as follows.

Let V be a vertex on limb j, P^obe the position of V in the local coordinate system of limb j, L_j^o be the transform from the local coordinate system of limb j to the world coordinate system, L_k^δ be the transform moving limb k to the world coordinate system, Ω be the set of limbs which control the motion of V with relevant weights, the new position of V in the world coordinate system can be calculated as follows.

$$P = \sum_{k\in\Omega} \omega_k L_k^\delta L_k^{o^{-1}} L_j^o P^o \qquad (1)$$

$$\sum_{k\in\Omega} \omega_k = 1$$

where $L_k^{o-1} L_j^o P^o$ represents the original coordinate of V in the local coordinate system of limb k, $L_k^\delta L_k^{o-1} L_j^o P^o$ represents the new coordinate of V in the world coordinate system caused by the motion of limb k. The procedure of SSD for a two-limb joint is illustrated in Figure 2. As shown in Figure 1, "crevasse" and "puncture" can be well solved through SSD.

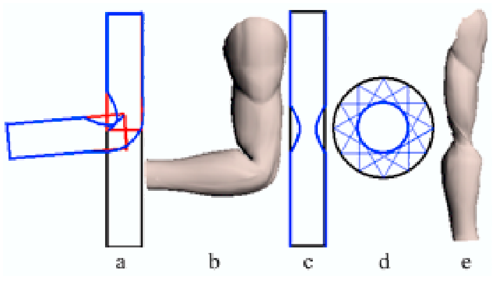

Figure 3: *Limitation of SSD. a illustrates the cause of "collapse" in SSD, and b shows the collapsed arm. c and d illustrate the cause of "pinch" and e shows the pinched arm.*

2.2. Recent work

As shown in Figure 3, SSD suffers from its inherent problem, "collapse" and "pinch". Bloomenthal[9] presented a medial based vertex blending technique, in which the weights associated with each vertex can be computed automatically by applying convolution to the medial axis/surface of the object, thus fewer undesired artifacts are evident in the animated surface. However, there are many unanswered questions concerning this technique, such as its robustness given a noisy medial, its dependence on the resolution of the medial and its ability to use hardware support. Lewis[8] proposed a unified approach to shape deformation and SSD, called pose space deformation. Starting from a simple SSD model, the vertex displacement offsets between the SSD surface and various character poses are stored. At the run time, the character may be simulated by mapping the interpolated displacement onto the underlying SSD character model, thereby providing a kinematic deformation model. Paul[10] proposed an "EigenSkin". In contrary to Lewis[8], in which the offsets between the SSD surface and those of the various pre-existing poses are stored and used for RBF based interpolation to create the offsets during the runtime, Paul[10] stores only the EigenSkins learned from the offsets mentioned above and synthesizes the offsets using these EigenSkins at the runtime. Thus the need for memory is reduced. Recently, Allen[11] presents a method to use range scan data to develop interactive character skins. However, both methods need to model the key poses in advance and are confined to the limited pose space.

Contrary to these above techniques, Mohr[12] proposed a method that lets users directly manipulate the vertex positions in a SSD deformed surface. The subspace of possible deformed vertex position is computed and displayed for users to place the vertex in this space, and then the correct weights for SSD are computed automatically. This method gives users as much direct control as possible and makes ex-

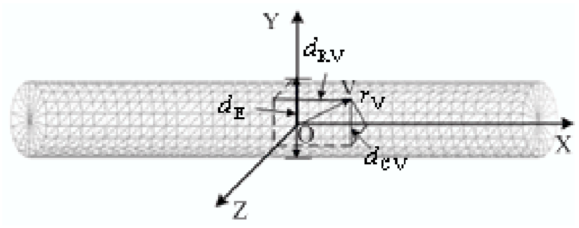

Figure 4: *Parameters of the displacement map.*

plicit what deformations are possible. However, as the positions of vertices are still confined to the subspace prescribed by SSD, some desired positions are unreachable. Moreover, it is tedious to correct the position of each vertex one at a time.

3. Pose based displacement map

As shown in Figure 3, "collapse" and "pinch" are the inherent problems of SSD, and can not be solved thoroughly through adjusting the weight distribution alone. We adopt a scheme similar to those of Lewis[8] and Paul[10], creating vertex offsets and mapping them onto the underlying SSD surface. In contrary to Lewis[8] and Paul[10], we model the vertex offsets as a displacement map and construct it according to the skeleton configuration, pose, as well as the original surface. No key pose surfaces need to be constructed beforehand and only a few displacement parameters are needed at the runtime. Thus the presented method is free of the pose space limitation and vast memory consumption. As the displacement map is controlled by only a few intuitive parameters, it also offers convenient manipulation for animators. We will introduce the modelling of the pose based displacement map in detail below with an example of a two-limb connection.

We establish a local coordinate system with the joint of a two-limb connection as the origin **O**, **X** axis along the right limb, **Z** axis perpendicular to the paper and out as illustrated in Figure 4. The displacement of vertex **V**,D_V, can be defined as follows.

$$D_V = A_E f_{CV} f_{RV} I_{WV} \tag{2}$$

where A_E is the amplitude of the displacement map, f_{CV} represents the distribution of displacement in the section of the limb parallel to the plane **YOZ**, f_{RV} represents the distribution of displacement along **X** axis, and I_{WV} is an identity vector representing the direction of the displacement. As the displacement map for a given limb varies along with poses only, this displacement map is called posed based displacement map.

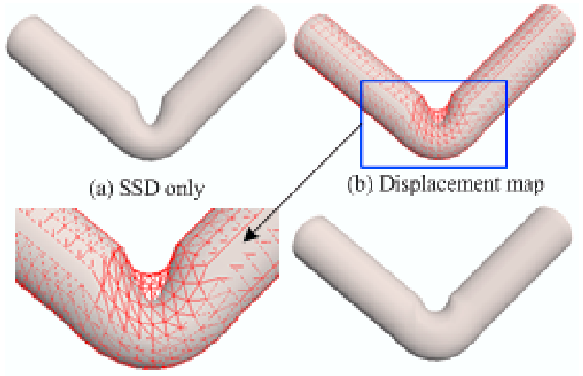

Figure 5: *Displacement map for rotation.*

3.1. Displacement map for rotation

When the right limb rotates around **Z** axis, the problem of "collapse" appears in the SSD deformed surface as shown in Figure 3(a) and (b). To solve this problem, a displacement map for rotation is constructed according to Equation 2 as follows.

A_E, the amplitude of the displacement map, varying along with the rotational angle and the size of the joint, can be defined as an increasing function of these two factors. For example, it can be defined as follow.

$$A_E = u d_E \tan(\theta/2) \quad (\theta \in [0,180))$$

where d_E is the thickness of the limb as shown in Figure 4, θ is the rotational angle of the right limb around **Z** axis, and u is a constant factor. Considering the possible postures of animated characters, it is reasonable to confine θ to [0,180). f_{CV}, the distribution of the displacement map on the section parallel to the plane **YOZ**, is relevant to the distance between the vertex V and the plane **ZOX**. A possible distribution is described as follows.

$$f_{CV} = d_{CV}/d_{Cmax}$$

where d_{CV} is the distance between V and the plane **ZOX**, and d_{Cmax} is the maximum among the d_{CV}s.

f_{RV}, the distribution of the displacement map along **X** axis, is related with the distance between the vertex V and the plane **YOZ**. In addition, the rotational angle θ determines the holistic distribution. Let d_{RV} be the distance between V and the plane **YOZ**, f_{RV} can be defined as follows.

$$f_{RV} = \exp(-d_{RV}^2/k(180-\theta)^2) \quad (\theta \in [0,180))$$

As SSD has no effect on the component of the position of V along **Z** axis, the direction of the displacement map lies on the rotational plane **XOY**. Let V' be the new vertex of V after SSD, $\overrightarrow{\mathbf{P}_{V'}}$ be the projection of the vector OV' on the

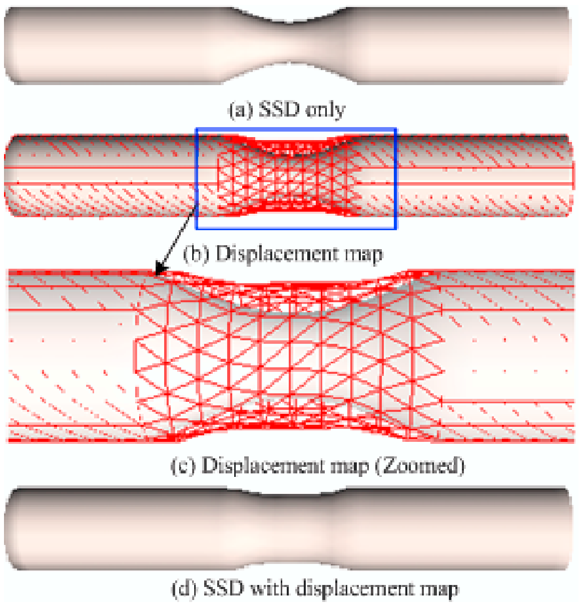

Figure 6: *Displacement map for twist.*

rotational plane, the identity vector I_{WV} can be defined as follows.

$$I_{WV} = \overrightarrow{\mathbf{P}_{V'}} / \|\overrightarrow{\mathbf{P}_{V'}}\|$$

The SSD deformed surface is rectified by mapping the above displacement map onto it, and the result is shown in Figure 5.

3.2. Displacement map for twist

When the right limb rotates around **X** axis, the problem of "pinch" occurs in the SSD deformed surface as shown in Figure 3(c), (d) and (e). To solve this problem, a displacement map for twist is constructed according to Equation 2 in the similar way to that for rotation.

A_E, the amplitude of the displacement map, varying along with the twisting angle and the size of the joint, can be defined as an increasing function of these two factors. For example, it can be defined as follows.

$$A_E = ud_E(1-\cos(\theta/2)) \quad (\theta \in [0,180))$$

where d_E is the thickness of the limb as shown in Figure 4, θ is the twisting angle of the right limb around **Z** axis, and u is a constant.

f_{CV}, the distribution of the displacement map on the section parallel to the plane **YOZ**, is relevant to the distance between the vertex V and **X** axis. A feasible distribution can be defined as

$$f_{CV} = r_V / r_{max}$$

where r_V is the distance between V and **X** axis, and r_{max} is the maximum among all the r_Vs.

f_{RV}, the distribution of the displacement map along **X** axis, can be defined in the same way as that in the rotational displacement map.

$$f_{RV} = \exp(-d_{RV}^2 / k(180-\theta)^2) \quad (\theta \in [0,180))$$

I_{WV}, the identity vector representing the direction of the displacement of V, can be defined as follows.

$$I_{WV} = \overrightarrow{\mathbf{P}_{V'}} / \|\overrightarrow{\mathbf{P}_{V'}}\|$$

where V'is the new vertex of V after SSD, and $\overrightarrow{\mathbf{P}_{V'}}$ is the projection of the vector OV' on the twisting plane **YOZ**.

The SSD deformed surface is rectified by mapping the displacement map above onto it, and the result is shown in Figure 6.

4. Experiment

We implement a skinning system based on the presented algorithm, in which users have options whether to create the displacement map for the SSD deformed surface. The system also provides an interface for users to manipulate the displacement. To create realistic skins, users firstly select the distribution for the displacement map. Our system provides several optional distributions similar to those addressed in Section 3. Next, users adjust the amplitude and parameters of the distribution through the user interface.

To verify the effectiveness of this algorithm, we have constructed an arm, to which SSD and the presented algorithm are applied respectively when its posture varies. The arms, when the forearm rotates with angles, 55 , 65 , 75 and 85 respectively, are shown in Figure 7. The SSD deformed arms are shown in Figure 7 (a), (b), (c), (d), and the arms deformed using the presented algorithm are shown in Figure 7(e), (f), (g) and (h) respectively. From the above comparison, we can see that our algorithm can solve the "collapse" effectively. The arms, when the forearm twists with angles, 100 , 110 , 120 , and 130 respectively, are shown in Figure 8. The SSD deformed arms are shown in Figure 8 (a), (b), (c), (d) and the arms deformed using the presented algorithm are shown in Figure 8(e), (f), (g) and (h). From the above comparison, we can also see that our algorithm can solve the "pinch" effectively.

5. Conclusion

SSD is a simple but novel Free Form Deformation technology in character animation and interactive game. However, it suffers from its inherent defects, called "collapse" and "pinch". Variation of weights can help to relieve these problems, but is unable to solve it thoroughly. A promising way is to rectify the SSD deformed surface with some offsets. The published algorithms in this way are mostly example

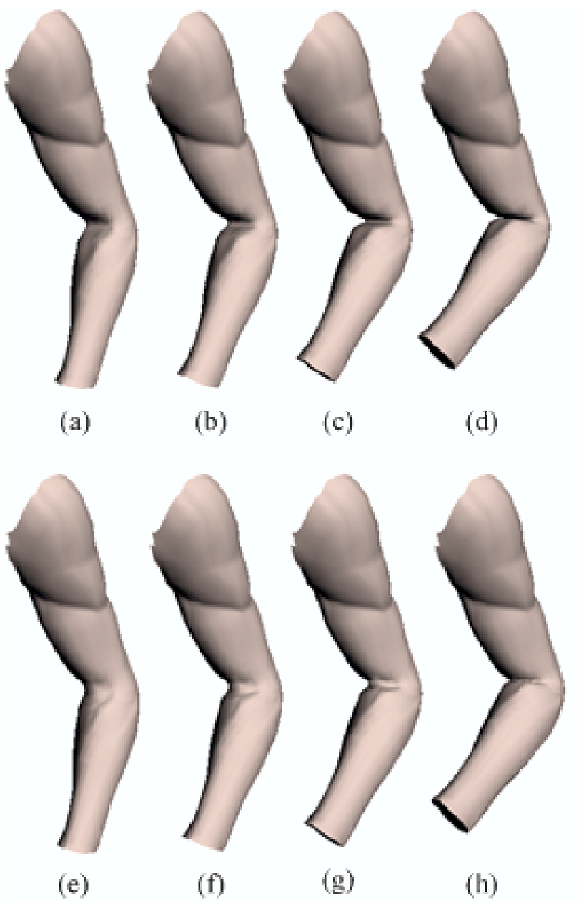

Figure 7: *Arms with forearm rotating.*

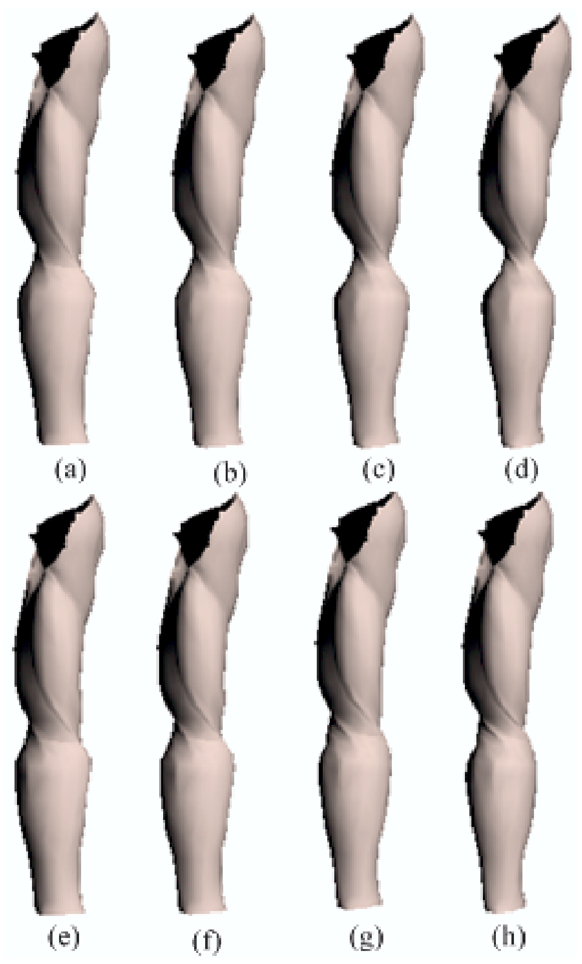

Figure 8: *Arms with forearm twisting.*

based methods, creating the offsets using some pre-existing surfaces constructed ahead. These methods suffer from the limitation of the existing pose space and need vast memory in the runtime. In this paper, we propose a displacement map modelling the offsets, which are created in the run-time according to the current pose and the original surface. No key surfaces need to be constructed in advance and no key offsets need to be stored in memory in the runtime. Thus the proposed displacement map is free of the limitation of the pose space and can also reduce the memory consumption in the runtime. Moreover, as the proposed displacement map is controlled by only a few intuitive parameters, this method gives users convenient and efficient control over the surface. The result shows the effectiveness of this algorithm.

In this ongoing research, we will focus on more powerful tools for users to manipulate the surface. Though the presented displacement map can be an effective amendment to SSD deformed surface, sometimes desired deformation still can not be achieved, for the distribution of the displacement map is confined to only a few control parameters. A promising way is to provide direct control over the displacement map. In contrary to Mohr[12], in which the position of the vertex is confined to the limited subspace, users can set the displacement at will. The constraint optimization method, in which an objective function is set up and solved while constraints such as the specifications of users and some necessary characteristics of the displacement map are met, can be employed to achieve the required displacement map, thus creating realistic surface.

6. Acknowledgements

This project was partly supported by NSFC grants 60203013, and Zhejiang Provincial Natural Science Foundation of China grants 602134.

References

1. P. Bergeron and P. Lachapelle, Controlling facial expression and body movements in the computer generated short 'Tony de Peltrie'. *SIGGRAPH 85 Tutorial Notes.* ACM, 1985.

2. Magnenat-Thalmann, N., and Thalmann, D. Human body deformations using joint-dependent local operators and finite element theory. *In making them move: mechanics, control, and animation of articulated figures*. Morgan Kaufmann,1991. pp.243-262.

3. F.Lazarus, S.Coquillart and P.Jancene. Axial deformations: an intuitive deformation. *Computer Aided Design*, **26**(8):607-613, 1994.

4. G. Maestri. *Digital character animation 2*, **1**,New Rider, Indianapolis, 1999.

5. F. Sebastian Grassia. Motion editing: Mathematical foundations. Motion editing: principles, practice, and promise. *Course of the 27th annual conference on Computer graphics and interactive techniques*, New Orleans, Louisiana, 2000.

6. J.Lander. A heaping pile of pirate booty. *Game Developer*, **8**(4):22-30, 2001.

7. T.Akenine-Möller and E.Haines. Full-range approximation of triangulated polyhedra. *Real-time rendering*, (2nd ed). Peters Ltd., 2002.

8. J.Lewis, M.Cordner and N.Fong. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* , New Orleans, Louisiana, 2000. pp. 165-172.

9. Jules Bloomenthal. Medial-based vertex deformation. *Proceeding of Symposium on Computer Animation 2002,* San Antonio,2002. pp. 147-151.

10. Paul G. Kry, Doug L. James, and Dinesh K. Pai. EigenSkin: Real time large deformation character skinning in hardware. *Proceeding of Symposium on Computer Animation 2002,* San Antonio,2002. pp.153-159.

11. Brett Allen, Brian Curless, and Zoran Popovid. Articulated body deformation from range scan data. *ACM Transactions on Graphics*, **21**(3):612-619, 2002.

12. Alex Mohr, Luke Tokheim, and Michael Gleicher. Direct manipulation of interactive character skins. *2003 Symposium on Interactive 3D Graphics*,April 2003 (to appear).