Example-based Deformation with Support Joints

Kentaro Yamanaka
Waseda University
3-4-1 Okubo
Shinjukuku
Tokyo, Japan, 1698555
kentaro.05-27@suou.waseda.jp

Akane Yano
Waseda University

Shigeo Morishima
Waseda University
shigeo@waseda.jp

ABSTRACT

In character animation field, many deformation techniques have been proposed. Example-based deformation methods are widely used especially for interactive applications. Example-based methods are mainly divided into two types. One is Interpolation. Methods in this type are designed to interpolate examples in a pose space. The advantage is that the deformed meshes can precisely correspond to the example meshes. On the other hand, the disadvantage is that larger number of examples is needed to generate arbitrary plausible interpolated meshes between each example. The other is Example-based Skinning which optimizes particular parameters referencing examples to represent example meshes as accurately as possible. These methods provide plausible deformations with fewer examples. However they cannot perfectly depict example meshes. In this paper, we present an idea that combines techniques belonging to the two types, taking advantages of both types. We propose an example-based skinning method to be combined with Pose Space Deformation (PSD). It optimizes transformation matrices in Skeleton Subspace deformation (SSD) introducing "support joints". Our method itself generates plausible intermediate meshes with a small set of examples as well as other example-based skinning methods. Then we explain the benefit of combining our method with PSD. We show that provided examples are precisely represented and plausible deformations at arbitrary poses are obtained by our integrated method.

Keywords

Example-based, skinning, deformation, pose space deformation, PSD, skeletal-subspace deformation, SSD, support joints

1. INTRODUCTION

Character deformation plays an important role in computer animations. Poor articulated character deformations are easily perceived when we see 3DCG animations. In order to generate desired character deformations, skilled animators spend much time working on tedious processes. Many character deformation methods have been proposed to resolve this problem, but it hasn't been completed yet.

In articulated character deformation, skeleton-based deformation is widely used to model articulated motion because they are intuitive to use. Among skeleton based deformation methods, Skeleton Subspace Deformation (SSD) [MLT88] is the most common technique, which is also called enveloping or smooth skinning, because it is fast to compute and easy to implement. Despite those advantages it brings, large deformations lead to undesirable effects such as very visible loss of volumes near joints. In order to compensate the defects of SSD, example-based methods are widely used especially in interactive applications.

Example-based deformations can be divided into two types – Interpolation and Example-based Skinning. Interpolation type focuses on how to interpolate examples smoothly in a pose space. Example meshes are always precisely represented, but interpolated ones are not always plausible and many examples are needed to generate plausible interpolated meshes at arbitrary poses. In contrast to interpolation, example-based skinning uses examples only to optimize particular deformation parameters such as vertex weights. The advantage is that plausible intermediate meshes at arbitrary poses between examples are obtained, but example meshes themselves are not fully represented.

In this paper, we present an idea that combines these two types and takes advantage of them. First we propose an example-based skinning method called support joint deformation. It optimizes joint transformation matrices introducing virtual joints called "support joints". Then we integrate our method with Pose Space Deformation (PSD) [LCF01], which is the most widely used in the interpolation methods, and demonstrate how effective our idea is.

The original support joint deformation method was proposed in our former work [YYM09]. We improve the support joint deformation method and combine the method with PSD in this paper in order to precisely represent examples when characters are deformed. To integrate with PSD effectively, we focused on artifacts occurring around a bending joint instead the previous work took all errors between examples into consideration.

This paper is organized as follows: After giving an overview of related work in character animation area in Chapter 2, we explain the theory and the disadvantage of PSD in Chapter 3 and introduce our core idea of example-based skinning with support joint optimization called support joint deformation in Chapter 4. The way how to set vertex weights suitable for support joint deformation is explained in Chapter 5, before we demonstrate the results of our idea and conclude in Chapter 6.

2. RELATED WORK

Skeleton-based deformation

SSD, which is based on a work published by Magnenat-Thalmann et al. [MLT88], has been very widely used for a long time in character animation field and has been adopted by most computer animation software because it is fast to compute, easy to implement and intuitive to use. The deformation of vertex with SSD is simply represented as follows:

$$\mathbf{v}'_i = \left(\sum_{j=1}^{N} w_{ij} \mathbf{T}_j \right) \mathbf{v}_i \qquad (1)$$

where $\mathbf{v}_i$ is the initial position of the i-th vertex of a character mesh and $\mathbf{v}_i'$ is a position after $\mathbf{v}_i$ is deformed. w_{ij} is a vertex weight which means the amount of influence of the j-th joint on the i-th vertex and it is normalized as $\sum_{j=1}^{N} w_{ij} = 1$. $\mathbf{T}_j$ is a transformation matrix of the j-th joint. N is the number of joints in a skeleton.

Though SSD has been very common, there exists an undesirable defect in this method. Artifacts such as "candy-wrapper" should be unavoidable. They are caused by the very visible loss of volumes when a character mesh is largely deformed.

Many works have been presented to overcome these artifacts. Rohmer et al. proposed a geometrically volume-preserving deformation method with interesting shape controls [RHC09]. Kavan et al. proposed spherical blend skinning [KZ05] and dual quaternion blend skinning [KCZO07] [KCZO08] to realize as rigid deformations as possible.

Example-based deformation

Example-based deformation is one of the attempts to get over the artifacts of SSD. It is often used in interactive application and is roughly divided into two types. The first is interpolation and the second is parameter-optimization. In this paper, we call the latter type example-based skinning. PSD [LCF01] is the most famous work in the interpolation group. It interpolates examples by Radial Basis Function (RBF) in a pose space using joint angles of SSD as deformation parameters. After PSD was proposed, many example-based interpolation methods have been published. "Shape by example" by Sloan et al. enabled extrapolation by adding low order linear polynomials to PSD [SRC01]. Kry et al. proposed a method suited to commercial graphics hardware reducing data of interpolation with PCA [KJP02]. Weighted Pose Space Deformation (WPSD) [KM05] [RLN06] generates arbitrary plausible intermediate poses with fewer examples. Some attempts to use various kinds of measured 3D data in example-based deformation have been presented [ACP02] [KM05]. Provided example meshes are always entirely represented. However, a large set of examples is needed to generate plausible meshes at arbitrary poses.

Example-based skinning needs smaller set of examples than interpolation, because given examples are used only to optimize some particular deformation parameters. Weber et al. proposed their original skeleton based deformation method and, in the paper, they additionally optimized vertex weights from examples in order to add "context" to the results [WSLG07]. Multi-weight enveloping (MWE) published by Wang et al. also optimizes vertex weights [WP02]. They introduced a novel vertex weight which had a value for each element of transformation matrix $\mathbf{T}_j$, instead of w_{ij} in equation (1), and described how to optimize the vertex weights. Mohr et al. proposed additional joints [MG03]. First they optimized values of vertex weights from examples and then they placed additional joints where some artifacts still occurred comparing to examples. It can be said that they optimized vertex weights w_{ij} and a number of joints N in equation (1). Shi et al. [SZTDVG08] optimized parameters for simulation using examples and characters were deformed according to the simulation. We propose an example-based skinning method which optimizes transformation matrices $\mathbf{T}_j$ in equation (1) using support joints.

Vertex weights

Vertex weights play an important role in our method as well as in the other skeleton based deformation methods. Baran et al. [BP07] solved heat equilibrium over character surface to determine vertex weights. They adopted heat equilibrium to satisfy three conditions below. First, vertex weights should

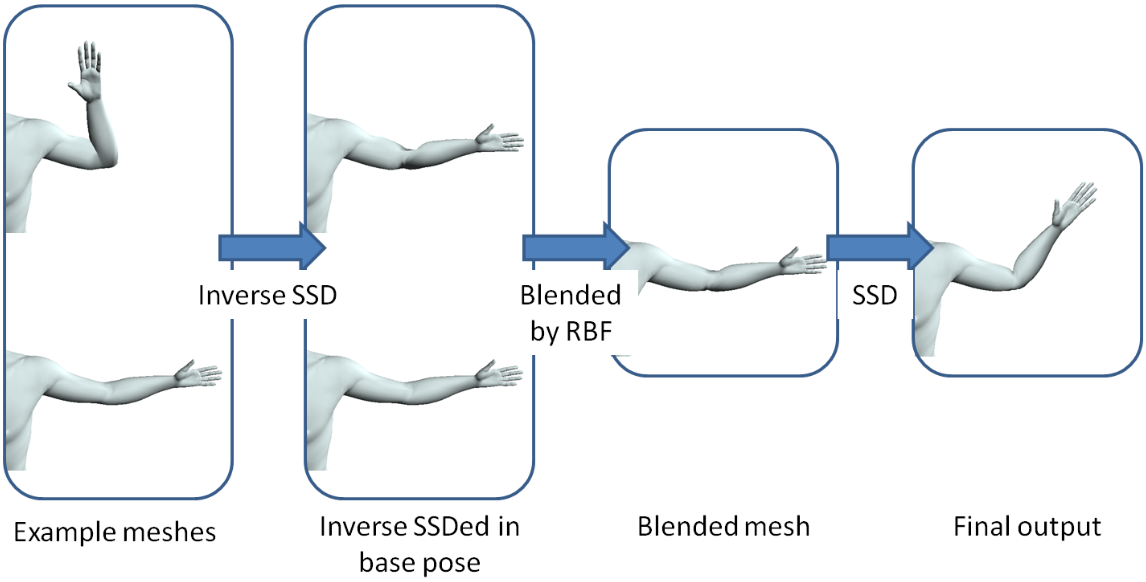

Figure1. The process of PSD

not depend on the mesh resolution. Second, the weights vary smoothly along the surface. Finally, the width of a transition between two bones meeting at a joint should be roughly proportional to the distance from the joint to the surface to avoid folding artifacts. They satisfied those conditions. However, the method needs an initialization of the weight before starting the diffusion smoothing. They suggested to initialize the values to $1/d^2$ when the segment linking the vertex to the bone lies entirely within the mesh volume, and zero otherwise. This automatical discontinuous definition works in most cases, but gives a bad initial value of the skinning weights if the mesh exhibits large non-convexities. Small variations in vertex positions may change dramatically the initial values when the line segment passes close to the mesh boundary. Rohmer et al. compensated this disadvantage and satisfied the conditions above by determining vertex weights according to geodesic volumetric distance [RHC09].

Our previous work [YYM09] defined a distance field μ on a surface mesh based on geodesic distance and determined the vertex weights according to the field. Computing μ is, however, costly because all-pair distances should be calculated and the vertex weights are recalculated at every step of transformation matrices optimization, which also needs much computing time.

3. POSE SPACE DEFORMATION

PSD is a hybrid approach that combines SSD and morphing. Various example meshes are deformed into the "base pose" with inverse SSD, and the resulting meshes are morphed and then deformed with SSD. The process of PSD is shown in Figure1.

Let $\mathbf{v}_{i,p}$ be the position of vertex i of the p-th example. The p-th example mesh is first transformed into its “base pose”:

$$\mathbf{v}_{i,p}^{b} = \left(\sum_{j=1}^{N} w_{ij} \mathbf{T}_{j,p} \right)^{-1} \mathbf{v}_{i,p} \tag{2}$$

where $\mathbf{v}_{i,p}^{b}$ is the position of vertex i of the p-th example in its base pose, w_{ij} be the weight value of joint j on a vertex i and $\mathbf{T}_{j,p}$ is the transformation matrix of joint j of the p-th example. Let $s_p(\mathbf{P})$ be the weight value for the interpolation at an arbitrary pose $\mathbf{P}$, satisfied with conditions as follows:

$$\begin{aligned} &\textstyle\sum_{p=1}^{n_{pose}} s_p(\mathbf{P}) = 1, \\ &s_p(\mathbf{P}_k) = 1 \qquad (p = k), \\ &s_p(\mathbf{P}_k) = 0 \qquad (p \neq k) \end{aligned} \tag{3}$$

where $\mathbf{P}_k$ is the k-th example pose and n_{pose} is the number of example poses. $s_p(\mathbf{P})$ is resolved using Radial Basis Function (RBF) as follows:

$$s_p(\mathbf{P}) = \sum_{p=1}^{n_{pose}} c_{p,k} \phi(d(\mathbf{P} - \mathbf{P}_k)) \tag{4}$$

where $c_{p,k}$ is a constant and $\phi(\mathbf{r}) = \exp(-\mathbf{r}^2/2D^2)$ is used in this paper. D is a parameter whose value is determined by users and D = 1.0 in this paper. If D becomes smaller, interpolation weight for the nearest example pose becomes bigger. $d(\mathbf{P} - \mathbf{P}_k)$ is a distance from an arbitrary pose $\mathbf{P}$ to an example pose $\mathbf{P}_k$ in an axis-angle manner. Then, each example surface in the base pose is interpolated by using a morphing method:

$$\mathbf{u}_i^b = \sum_{p=1}^{n_{pose}} s_p(\mathbf{P}) \mathbf{v}_{i,p}^b \tag{5}$$

where $\mathbf{u}_i^b$ is the interpolated vertex position in a base pose. Finally, the morphed surface is deformed with

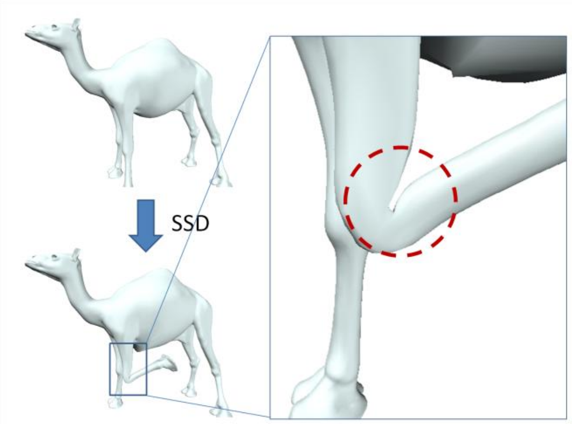

Figure2. An artifact of SSD

Large deformations frequently lead to very visible loss of volume around joints. That causes artifacts of PSD as well.

SSD:

$$\mathbf{u}_i = \left(\sum_{j=1}^{N} w_{ij} \widetilde{\mathbf{T}}_j \right) \mathbf{u}_i^b \qquad (6)$$

where $\mathbf{u}_i$ is the vertex position of the resulting deformed surface, $\widetilde{\mathbf{T}}_j$ is the transformation matrix that is calculated by interpolating the matrices of examples using $s_p(\mathbf{P})$.

As explained above, PSD is largely based on SSD. With SSD, however, large deformations lead to undesirable results such as very visible loss of volume near joints (See Figure2). That also causes artifacts in PSD. The artifacts of PSD are mainly caused by the steps of the inverse SSD and SSD in the process (See Figure1). Figure3 shows an example of artifacts in PSD due to SSD. When examples are deformed into the base pose with SSD (inverse SSD), artifacts frequently occur. Then the artifacts remain on the blended mesh. Besides, more artifacts may happen at the last SSD step. Figure1 shows mechanism how artifacts of PSD occur as well. Users have been forced much more tedious works preparing additional examples for the results to look better. In order to avoid this otiose task, a deformation method which has fewer artifacts should be adopted instead of SSD to improve the quality of PSD. Then we propose such a method in the next chapter.

4. EXAMPLE-BASED SKINNING WITH SUPPORT JOINTS

Motivation

In order to overcome artifacts of PSD derived from the defects of SSD such as apparent loss of volumes around a joint when the joint is largely bent, we considered that example-based skinning method is the most suitable and efficient alternative for SSD because there are already prepared examples and it usually takes much time for users to prepare additional ones. Therefore this chapter presents example-based skinning method which reduces artifacts around a bent joint.

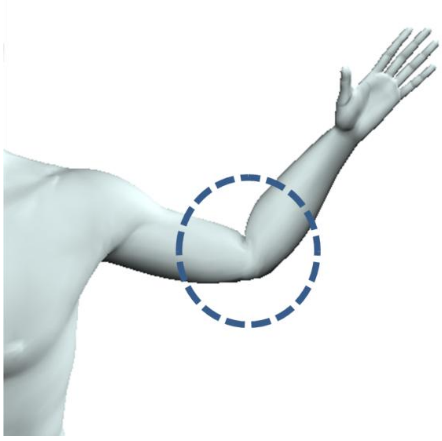

Figure3. An artifact of PSD because of SSD

Example-based skinning optimizes particular deformation parameters from examples for the results to look good. Wang et al. [WP02] introduced a novel vertex weight which has one value for each of twelve elements of transformation matrix in equation (1), and optimized them. Then it is called Multi Weight Enveloping (MWE). After optimizing vertex weights Mohr et al. optimized the number of joints each of which has a special function such as scaling in order to reduce remaining artifacts and represent examples as correct as possible. It hasn't been presented yet a method which optimizes transformation matrices, or improves behavior of joints from examples though it is often said that the poor structure of a traditional hierarchical skeleton is one of the problems that causes various artifacts of skeleton based deformation. A method which optimizes transformation matrices and reduces the loss of volumes around a bent joint with SSD is thought to be a solution to reduce the artifacts of PSD.

Support Joint Deformation

In order to realize optimization of transformation matrices, a set of support joints **S** is defined besides the traditional hierarchical skeleton **H**. **S** doesn't have hierarchical structure and each joint of **S** can move around the corresponding joint of **H** according to the rule explained later, in contrast to that **H** has a hierarchical structure and the distances between joints are constant.

In our proposing method, it is assumed that example meshes and corresponding joint angles of **H** are given. After a base mesh is deformed according to the given joint angles of **H** in an example pose with SSD, transformation matrices of **S** are calculated to minimize error function $\xi(\mathbf{S})$ described below:

$$\xi(\mathrm{S}) = \sum_i \left\| \mathbf{v}_i^T - \left(\sum_{j=1}^{N} w_{ij}^S \mathbf{T}_j^S \right) \mathbf{v}_i^{SSD} \right\| \qquad (7)$$

where i is the index of vertex. $\mathbf{v}_i^T$ is a position of vertex i of a given example pose. w_{ij}^S means a vertex weights of the j-th joint of **S** on vertex i, which is described in Chapter 5 in detail. $\mathbf{T}_j^s$ is the transformation matrix of the j-th joint of **S** that should be optimized, and $\mathbf{v}_i^{SSD}$ is a vertex deformed by SSD according to the given joint angles. Therefore $\xi(\mathbf{S})$ returns a difference between a target mesh and a deformed mesh.

In this paper, there are rules how joints of **S** moves around joints of **H** to obtain the optimal position and the optimal transformation matrix efficiently. Discrete values are set on x-axis linked coordinate of the parent joint of **S** and iterative search algorithm which minimize$\xi(\mathbf{S})$ is solved. Figure4 shows how **S** moves. The rules are described below.

- ✓ The j-th support joint moves along a line from the bent joint j of **H** to the parent joint j-1.
- ✓ First, the j-th support joint is translated along the axis, and then it is rotated to the direction which a virtual bone connects j-th support joint of **S** and j+1-th joint of **H**.

According to the manner above, transformation matrix $\mathbf{T}_j^S$ is calculated.

Using the calculated transformation matrix above, plausible intermediate meshes are obtained at arbitrary poses. According to the pose desired by uses, the intermediate transformation matrix is calculated by interpolating the transformation matrices $\mathbf{T}_j^s$ for example poses. $s_p(\mathbf{P})$ in equation (5) is used to blend transformation matrices $\mathbf{T}_j^s$ of example poses. The result of support joint deformation can be obtained when $\mathbf{v}_i$, w_{ij}, and $\mathbf{T}_j$ in equation (1) are replaced by $\mathbf{v}_i^{SSD}$, w_{ij}^S, and interpolated $\mathbf{T}_j^s$.

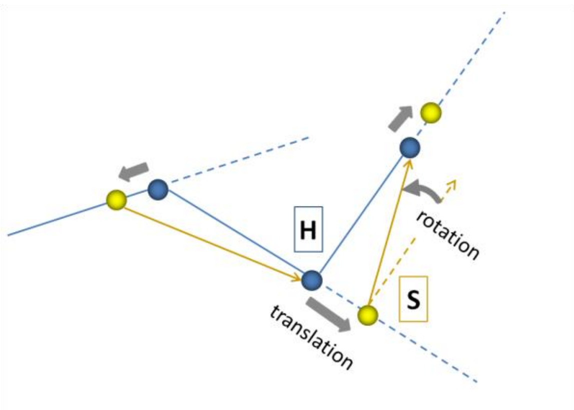

Figure4. The manner how support joints move

Blue: Hierarchical skeleton H, Yellow: A set of support Joints S. The j-th support joint moves on a line which connects j-th and j-1th hierarchical joint, and the direction of the j-th support joint are determined from the position of the j+1th hierarchical joint.

Figure5. The result of support joint deformation compared to SSD

Left: SSD, Right: support joint deformation

Both meshes are deformed with the same joint angles.

Figure5 shows how much our proposing example-based skinning reduces the artifacts. The artifacts of SSD could be sufficiently improved by our proposing support joint deformation.

5. VERTEX WEIGHTS

In skeleton based deformation vertex weights play a very important role and it's also true in our method. Especially for SSD, users often set vertex weights manually using painting tool in commercial animation software. However, it takes much time for users to set vertex weights, while there often exist puzzling cases in which no set of vertex weights can avoid an artifact. Then many techniques to decide vertex weights automatically have been proposed.

Example-based skinning methods decide optimal vertex weights to represent the examples the best referencing a set of the given example meshes and the corresponding skeletons. Mohr et al. optimized vertex weights before they introduced additional joints [MG03]. Weber et al. optionally optimized vertex weights from examples to add "context" to the results after they deformed characters with their original deformation method [WSLG07].

Though support joint deformation described in the previous chapter is also classified into example-based skinning, vertex weights in our method are determined not based on examples but geometrical information. The reason is that we need to determine two types of vertex weights. The first one is for SSD represented as w_{ij} in equation (1). The second one is for support joint deformation represented as w_{ij}^S in equation (7). w_{ij} can be determined from examples like other example-based skinning methods because values of the other variables are already known. On the other hand, w_{ij}^S cannot be optimized by examples because the transformation matrix $\mathbf{T}_j^S$ is unknown and it is required to be calculated from geometrical information of examples. Because of the efficiency, we adopt a technique which is able to calculate both different vertex weights from geometrical information. Both vertex weights are required to meet conditions below.

- Vertex weights vary smoothly and continuously along the surface.
- Vertex weights avoid artifacts caused by incorrect association between a joint and vertices.
- About w_{ij}, a transition of vertex weights around joints between two bones is roughly proportional to the distance from the joint to the surface.
- About w_{ij}^S, values of weights are high (close to 1) around the bent joint and converge to 0 smoothly according to the distance from the joint.

The simplest technique to determine vertex weights referencing only geometrical information of a mesh is to calculate based on Euclidean distance from a joint to vertices. It can satisfy the first condition although it cannot meet the second and often causes artifacts that attach weights from wrong joint to a vertex because topology of character mesh is unconsidered. In this paper we adopt the technique which Baran et al. poposed [BP07] to determine both two types of vertex weights.

Calculation of geodesic volumetric distance

Character mesh is required to be filled with voxels (Figure6) prior to the measurement of geodesic volumetric distance. We assume that a voxelized model is already pre-computed. In order to calculate the distance from a joint or a bone to vertices, a voxel which includes the joint (or bone) inside and a voxel which includes a vertex inside need to be specified. The geodesic volumetric distance from the joint (or bone) voxel to vertex voxels are calculated using dijkstra's algorithm applied to three dimensions.

STEP1: Initialize $g(v) = \infty$ about all vertex voxels.

STEP2: Select a base voxel b, set $g(b) = 0$, and insert b to VLIST.

STEP3: Take the vertex voxel v which has the smallest g(v) in VLIST and remove it from VLIST.

STEP4: For each vertex voxel v_a adjacent to v, if $g(v) + length(v, v_a) < g(v_a)$, update $g(v_a) = g(v) + length(v, v_a)$ and insert (or reinsert) v_a to VLIST, where length(v, v_a) is defined as the Euclidean distance from the center of v to the center of v_a. Notice that the adjacency of vertex voxels is determined whether a voxel is a member of a cube, which consists of 27 (= 3*3*3) voxels whose center is v, or not.

STEP5: Repeat Step3 and 4 until VLIST become empty.

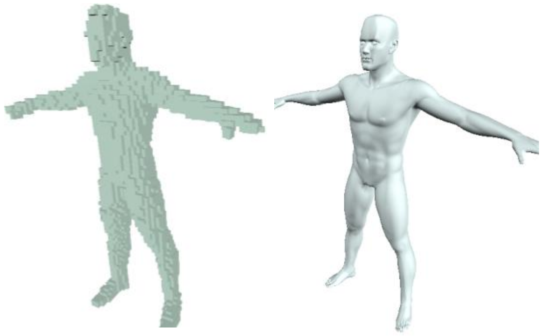

Figure6. A voxelized character mesh
Left: A voxelized character mesh, Right: The corresponding character mesh

Figure7. Geodesic volumetric distance from a forearm bone
Red means that the distance from a voxel including the bone to the vertex is small. Blue means the vertex is distant from the bone voxel.

Geodesic volumetric distance calculated with this algorithm is shown in Figure7.

Determination of vertex weights

In this paper, two types of vertex weights are determined using the calculated distance above. The first weights are designed for SSD. It is required to meet the first, the second and the third condition previously mentioned. To satisfy the third condition, the distances from bones are used to determine the vertex weights. Therefore, bone voxels are set as base voxels in the calculation process. Figure8 shows vertex weights from forearm bones. A vertex weight is determined by reciprocal ratio of squared distances from the nearest and the second nearest bone, and the result is normalized.

The second weights are for support joint deformation. It is required to meet the first, the second and the fourth condition and therefore distance from the bent joint is adopted. The vertex weights for support joint deformation are determined by multiplication of a gaussian function of a distance from a joint and the error of a vertex position on a base mesh deformed by SSD from a vertex position on an example mesh. Note that, when multiplied, the errors are normalized to let the maximum value be 1. Figure9 shows the resulting vertex weights for support joint deformation. Vertex weights for support joint deformation are determined referencing the distance from the joint and the error between a SSDed mesh and the corresponding example mesh.

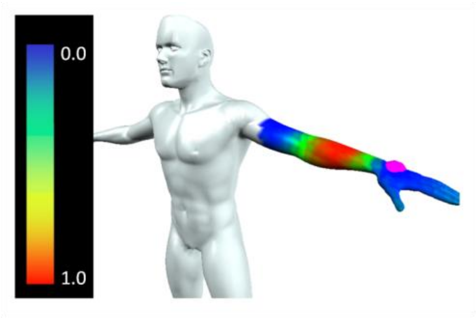

Figure8. Vertex Weights of the forearm bone for SSD

Red: a vertex where $\mathbf{w_{ij}} = \mathbf{1.0}$, Blue: the value of $\mathbf{w_{ij}}$ is close to 0.0, White: precisely $\mathbf{w_{ij}} = \mathbf{0.0}$

Figure9. Vertex Weights of the elbow joint for support joint deformation

Red: a vertex where $\mathbf{w_{ij}} = \mathbf{1.0}$, Blue: the value of $\mathbf{w_{ij}}$ is close to 0.0, White: precisely $\mathbf{w_{ij}} = \mathbf{0.0}$

6. RESULTS AND CONCLUSIONS

Integrated method

Finally we present a method which combines PSD as an interpolation and support joint deformation as an example-based skinning. It is simply explained as following. The inverse SSD step in the process of PSD is replaced by the inverse support joint deformation, and the last SSD step is replaced by support joint deformation. First, transformation matrices $\mathbf{T_j^S}$ of bent joints of examples are optimized. Second, examples are deformed into base pose by inverse support joint deformation. Note that inverse support joint deformation consists of two steps. First step is inverse SSD and the second step is multiple inverse of $\mathbf{T_j^S}$ to a deformed mesh. Then deformed examples in base pose are morphed. Lastly, The morphed mesh is deformed with support joint deformation. Though our method needs additional computational time to PSD in order to optimize the transformation matrices as pre-processing, the deformation is executed as fast as PSD because we only changed SSD to support joint deformation.

Result

Figure10 shows our method compared with PSD. It demonstrates that our integrated technique of PSD and support joint deformation entirely represents the example poses as PSD does and generates arbitrary plausible intermediate poses between each example. It means that plausible results at arbitrary intermediate poses can be obtained with fewer examples than PSD because PSD needs more examples to improve the results. Therefore, it can be said that our proposing method successfully takes advantages of interpo-

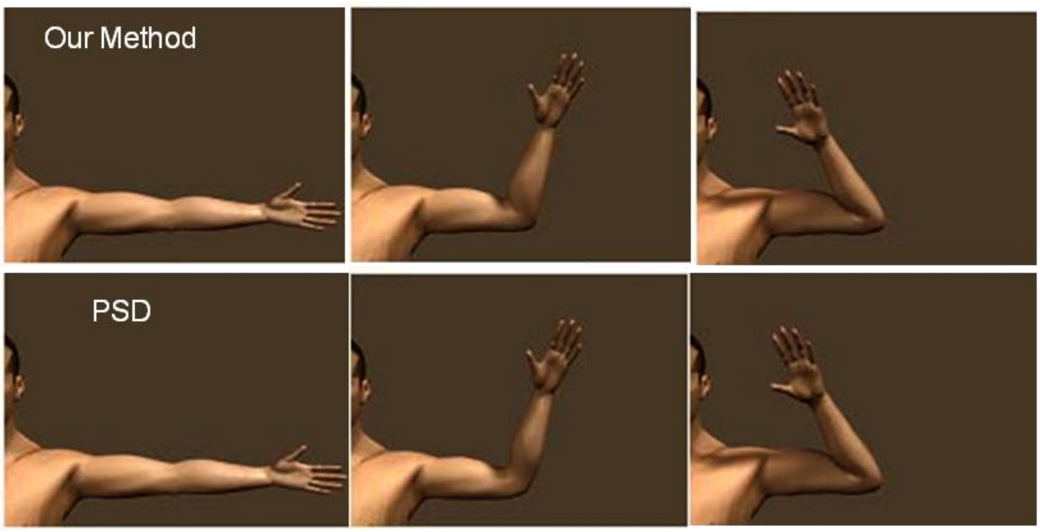

Figure10. The result of our method compared with PSD

Upper: Our proposing method that combines PSD and our original Support Joint Deformation.

Lower: PSD

Left and right images show that our method precisely represents the example poses as PSD does, and the middle images demonstrate our method is able to generate more plausible intermediate poses than PSD.

lation and example-based skinning.

Future work

We would like to integrate support joint deformation into Weighted Pose Space Deformation (WPSD) [KM05] [RLN06] for the results to look better with smaller set of examples than combining with PSD. This paper regulated motion of support joints as described in Chapter4. We would like to deregulate the manner and reduce all kinds of artifacts. Geodesic volumetric distance is adopted to measure distance between a joint (bone) and a vertex when vertex weights are determined. Instead, interior distance [RLF09] can be also employed to reduce the computing time of pre-processing.

7. REFERENCES

[ACP02] ALLEN, B., CURLESS, B., and POPOVIC, Z. Articulated body deformation from range scan data. Proceedings of the 29th annual conference on Computer graphics and interactive techniques, pp.612-619, 2002.

[BP07] BARAN, I., and POPOVIC, J. Automatic rigging and animation of 3d characters. ACM SIGGRAPH 2007 papers, p.72, 2007.

[KCZO07] KAVAN, L., COLLINS, S., ZARA, J., and O'SULLIVAN, C. Skinning with dual quaternions, Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. ACM Press, pp.39-46, 2007.

[KCZO08] KAVAN, L., COLLINS, S., ZARA, J., and O'SULLIVAN, C. Geometric skinning with approximate dual quaternion blending. ACM Transaction on Graphics, 27, 4, 2008.

[KJP02] KRY, P., JAMES, D. L., PAI, D. K.. EigenSkin: Real time large deformation character skinning in hardware. Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp.153-159. 2002.

[KM05] KURIHARA, T., and MIYATA, N. Modeling deformable human hands from medical images. Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation (Aire-la-Ville, Switzerland), Eurographics Association, pp.355-363, 2005.

[KZ05] KAVAN, L., and ZARA, J. Spherical blend skinning: a real-time deformation of articulated models. Symposium on Interactive 3D Graphics and Games, pp.9-16, 2005

[LCF01] LEWIS, J. P., CORDNER, M., and FONG, N. 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, pp.165-172, 2000.

[MG03] MOHR, A., and GLEICHER, M. Building efficient, accurate character skins from examples. ACM SIGGRAPH 2003 Papers, pp.562-568, 2003.

[MLT88] MAGNENAT-THALMANN, N., LAPERRIERE, R., and THALMANN, D. Joint-dependant local deformations for hand animation and object grasping. Proceedings on Graphics interface '88, pp.26-33, 1988.

[RLF09] RUSTAMOV, R., LIPMAN, Y., FUNKHOUSER, T. Interior Distance Using Barycentric Coordinates. Computer Graphics Forum (Symposium on Geometry Processing) 28(5), July, 2009

[RLN06] RHEE, T., LEWIS, J., and NEUMANN, U. Real-time weighted pose-space deformation on the GPU. Computer Graphics Forum 25, 3. 2006

[RHC09] ROHMER, D., HAHMANN, S., and CANI, M.-P. Exact volume preserving skinning with shape controls. Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp.83-92, 2009.

[SRC01] SLOAN, P.-P., ROSE, C., and COHEN, M. Shape by example. Proceedings of the 2001 symposium on Interactive 3D graphics, pp.135-143, 2001.

[SZTDVG08] SHI, X., ZHOU, K., TONG, Y., DESBRUN, M., BAO, H., and GUO, B. Example-based Dynamic Skinning in Real Time. ACM Trans. on Graphics, August 2008.

[YYM09] YAMANAKA, K., YANO, A., MORISHIMA, S. Example Based Skinning with Progressively Optimized Support Joints. ACM SIGGRAPH Asia 2009 poster, December 2009.

[WP02] WANG, X. C., and PHILLIPS, C. Multi-weight enveloping: Least-squares approximation techniques for skin animation. Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation, pp.129-138, 2002.

[WSLG07] WEBER, O., SORKINE, O., LIPMAN, Y., and GOTSMAN, C. Context-aware skeletal shape deformation. Comp. Graph. Forum 26, 2007.