Animation of Refitted 3D Garment Models for Reshaped Bodies

Yifan Yan[1] Juntao Ye[1] Xiaoyang Zhu[1] Jituo Li[2]
[1]*National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences*
[2]*Institute of Engineering and Computer Graphics, Mechanical Engineering Department, Zhejiang University*
E-mail: *juntao.ye@ia.ac.cn*

Abstract

We present a framework for a virtual try-on system. Starting with a template human body mesh, we reshape it to acquire meshes of various body dimensions according to the user specified parameters. These reshaped bodies are further processed in two directions: refitting a garment model to the outside of the body and embedding a skeleton to the inside. The refitted garment mesh is then bound to the body mesh via our coat-to-mesh algorithm, which is the major contribution of this paper. The body mesh is skinned to the embedded skeleton with an implicit rigging process. This way, any deformation in the body mesh will lead to corresponding deformation in the garment mesh. During the deformation, both the mesh topology and spatial relationship between the body and the garment are maintained. At the end, we apply some third-party motion data to drive the skeleton, the body mesh, as well as the garment mesh, and create real-time animations of dressed human character.

Keywords: garment animation, virtual try-on, character modeling, reshaping, binding

1. Introduction

There's a pressing need to provide a more personalized experience for online shopping than just browsing through catalogs. This is even more critical for clothes shopping because different people have different physical characteristics and preferences. Researchers have created technologies such as virtual mirrors and video fitting using VR for "trying on" outfits online. However, these technologies haven't provided comfortable user interaction or achieved an emotional response in which users can visualize themselves wearing or using the products in a natural environment.

On the other hand, in fashion industry, garments that are tailored out of exquisite materials and artful designed patterns can be very sophisticated. They are not only envelopes for the human body, but also artworks that visualize cultural aspects, tendencies and trends. However, the current status of mass-production of garments is less artistic. Manufactures take limited advantages of consumer's body shapes and produce garments based on pre-defined pattern sizes (e.g. S, M, L, XL, XXL). Advanced garment customization that serves the masses is still uncommon. In recent years, the Internet has emerged as a compelling channel for garments sale, and initiated the concept of virtual try-on. Yet such Web applications nowadays have supported only basic functions and are far from a practically useful level.

Clothing animation is not only useful for designing and prototyping garments before the manufacturing process, but also in great demand in special effects industry. The most appealing technique is physically based simulation, which will give highly realistic result. However, simulation usually computationally costly such that it is seldom used in real-time applications.

In this paper we describe a framework for a virtual try-on system. Several key technical issues, including human body reshaping, garment refitting and garment animation, will be investigated.

2. Related work

Some earlier work mainly used image processing techniques to construct virtual try-on system. Hilsmann and Eisert [1] described a dynamic texture overlay method from monocular images for visualizing garments in a virtual mirror environment. In their follow-up work, a method to segment the user's clothes and retexture them using extracted shading and shape deformation information [2] was proposed. Zhang *et. al.* used fiducial markers to change the texture of a user's shirt [3]. In Spanlang *et al*'s work [4], a pre-generated 3D human model in target clothes was superimposed on a user's 2D picture. Their follow-up work [5] presented a virtual clothing system, in which a user is scanned and registered to the system once, and then clothes can be simulated on the reconstructed model. Shilkrot et al. [6] created a system that offers a virtual experience akin to trying on clothing. It clones the user's photographic image into a

978-1-4799-6854-1/14 $31.00 © 2014 IEEE
DOI 10.1109/ICVRV.2014.23

catalog of images of models wearing the desired garments. The process takes into account the user's skin color and body dimensions.

Different from image based systems, 3-dimensional systems are geometrically more accurate. Thanh and Gagalowicz [7] enables a user to load his/her own 3D model, select 3D clothes from a catalogue and superimpose them on the model by interactive positioning. Frederic *et al.* [8] presented a framework for a web-based solution using a generic database for dressing a user look-alike clothed avatar and simulating the clothes on the avatar. Meng *et al.* [9] and Wacker *et al.* [10] proposed solutions for sewing of clothing patterns on a virtual 3D human character and viewing the simulation results of the clothes. Zou *et al.* [11] implemented a web-based platform that allows interactive viewing of clothes simulation and selection of different hairstyles and other accessories on an avatar.

In recent years, some interactive virtual try-on solutions using augmented reality technique have been reported. A major challenging issue in AR based systems is the requirement of accurate pose recovery for fitting virtual clothes (or other accessories) on a user's image [3,13]. Today with the release of a new generation of sensing technologies capable of providing high quality videos of both color and depth (e.g., Microsoft Kinect [14]), it provides an opportunity to dramatically increase the capabilities of virtual try-on solutions. Hauswiesner [15] described an impressive virtual try-on system where a user's human model and cloth were reconstructed by deforming a SCAPE [12] model using a multiple-camera setup. Next, the skeleton obtained from a Kinect sensor is mapped to the SCAPE model so the model can follow the user's movements captured by the Kinect sensor. The advantage of this method is that the system enables users to enjoy a private virtual try-on experience at their own homes.

3. Pipeline of our virtual try-on system

The pipeline of our virtual try-on system is illustrated in Fig. 1. Starting with a template human body mesh, we derive meshes of various body dimensions by reshaping. We then do two things for these reshaped bodies: refitting a garment model to the outside of the body and embedding a skeleton to the inside. The refitted garment mesh is then bound to the body mesh via our coat-to-mesh algorithm. The body mesh is skinned to the embedded skeleton. At the end, we apply some third-party motion data to derive the skeleton, the body mesh, as well as the garment mesh.

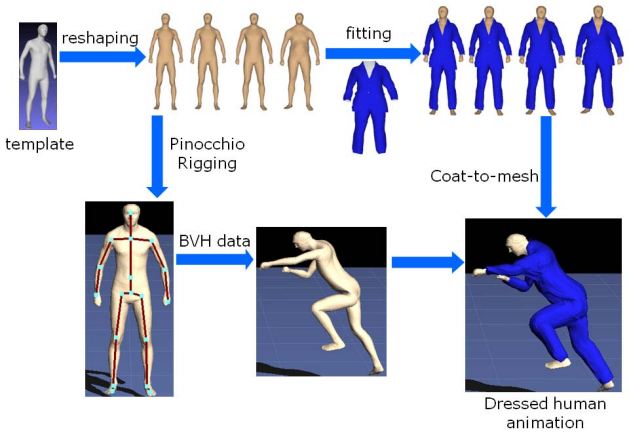


Figure 1. Pipeline of our virtual try-on system

3.1. Body reshaping

A natural way to acquire a person's body model is by using expensive acquisition device - 3D human scanner. This is, however, not only financially costly, but also requires tedious post-processing. Therefore, there is a great request from the market to have a parametric design tool for human bodies so that the shape of a 3D human body can be generated from a set of semantic input (e.g., height, chest-girth, waist-girth, hip-girth, inseam-length, etc.). We follow the work by Chu et. al [14] for reshaping human bodies.

Their method provides an intuitive way for the user to generate body shape by appointing a set of semantic values. Human models and the semantic parameters are correlated as a single linear system of equations. This approach involves simpler computation compared to non-linear methods while maintaining quality outputs. By this, a semantic parametric design in interactive speed can be implemented easily. A new method is developed to quickly predict whether parameter values are reasonable or not, with a set of training models as given in the public domain human model database [15] with 550 full body scans.

The body models are created by varying seven semantic parameters, namely *height*, *shoulder width*, *chest girth*, *waist girth*, *hip girth*, *arm length* and *leg length*.

3.2. Skeleton embedding and rigging

Nowadays a popular method for animating an articulated 3D character is a technique called *skinning* – vertices of the garment mesh are attached to bones and transformed along with the bone's coordinate system. Due to its low computational cost, this technique is still widely used in low-end movies and most 3D video games. However, setting up the skin weights needs to be done manually, which is non-trivial. Researchers have been

working hard to find solutions for automatic rigging. Baran and Popovic [18] put forward a method that works well under certain circumstances. We took advantage of their prototype implementation (called *Pinocchio*) and integrated their code into our system.

Pinocchio consists of two main steps: *skeleton embedding* and *skin attachment*. Skeleton embedding computes the joint positions of the skeleton inside the character by minimizing a penalty function. The skin attachment is computed by assigning bone weights based on the proximity of the embedded bones smoothed by a diffusion equilibrium equation over the character's surface. Once the skeleton is embedded into the character and attached to the surface, skeletal motion data can be used to animate the character.

We adopt Pinocchio's built-in skeleton for embedding, which consists of fixed number of bones. However, when driving this skeleton using BVH motion data, which may be motion-captured based on a different skeleton topology, a mapping between two skeletons needs to be built. This work will be detailed in Section 5. Pinocchio also demands the static character mesh to be a closed two-manifold, and any meshes with holes do not work. Fortunately, the body meshes we use are perfectly closed meshes.

3.3. Garment refitting

It is possible to reuse the models directly created in 3D space for different sized bodies. While retaining the style and the topology of the model, its shape will be shrunk or stretched, non-uniformly, to fit the target body. For refitting, we need to deal with either one of the two different cases. In the first case, the template human is dressed with a garment model, and this garment is to be transferred to another body which is actually reshaped from the template body. In the second case, a garment model is to be transferred to a human model that initially has no correspondence with the garment. Typically for the second case, the garment posture is largely different from the body posture.

We deal with the first case in our system, i.e. all garment models were created to fit the template body model at the beginning. As did in [17], the refitting has two stages: firstly the garment undergoes a global adjustment so that it roughly matches the reshaped body; secondly a fine-tune process to tackle the penetration issues at some regions and to adjust the garment-human ease allowance at the other regions. Once penetration is detected, it has to be resolved. There are two types of penetrations. In the first case, garment vertices are located inside the body mesh; in the second case, body vertices are inside the garment mesh. (Here we determine the inside and outside of a mesh according to the face normal.) These two cases are handled separately. The first case is relatively easy to tackle, and the second case is resolved with help of octree structure. More details can be found in [17].

3.4. Coating garment mesh to body mesh

Once a garment is fitted to the reshaped body, it is desirable that both the body mesh and the garment mesh can be animated. As a garment mesh is usually not closed, Pinocchio is unable to bind them to a skeleton. Instead of binding a mesh to a skeleton, is it possible to bind the garment to the body mesh? If yes, the garment mode will acquire its motion from the underling body motion. This motivates us to put forward a new technique called *coat-to-mesh* algorithm. Actually, this idea has potentials in other application context. With the emergence of new devices such as Kinect sensor, the motion and deformation of a human body can be captured directly, adding another option in addition to the skeleton driven animation. If a garment model is to move along with a "boneless" mesh, there has to be a technique to attach the garment mesh to the body mesh. This new technique is analogous to the idea of skin-to-bone. We have made it an automatic process, with little intervention from the user. This method is the main contribution of this paper, and will be detailed in Section 4.

3.5. Motion retargeting from BVH data

Following the previous steps, we should have bound the new body mesh to a skeleton, and bound a new garment mesh to the body mesh. Now the body mesh can be driven by the skeleton if proper motion data is applied, and the body mesh will in turn drive the garment mesh automatically.

To show the flexibility of our system, we intend to use popular motion data files, particularly those formats supported by motion capture devices. The BVH file format was originally developed by Biovision, a motion capture services company, as a way to provide motion capture data to their customers. A BVH file specifies a bone hierarchy, which may vary from file to file. It is also very likely that a BVH bone hierarchy is different from the Pinocchio's built-in structure. Therefore the motion needs to be retargeted from one skeleton to another. This will be detailed in Section 5.

In summary, through several stages of complex processing, we could refit a template garment model to a reshaped body and create animation for it.

4. The Coat-to-mesh algorithm

We want to bind the garment mesh to the body mesh, so that whenever the body deforms the garment will deform accordingly. This deformation must be in a

reasonable manner. It should not change the topology of the original mesh. It should maintain the spatial relationship of the two meshes. Particularly, if at the initial state the two meshes have no penetrations, so should be their deformed counterparts. We designed a binding approach, called the *coat-to-mesh* algorithm, which meets the above requirements.

4.1. Background: barycentric coordinates on triangles

Let us consider a triangle **T** defined as the convex hull constructed from the three vertices $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. Any point **x** located on the plane defined by these three vertices may be written as a weighted sum of them, i.e. $\mathbf{x}=\lambda_1*\mathbf{x}_1+\lambda_2*\mathbf{x}_2+\lambda_3*\mathbf{x}_3$, where λ_1, λ_2, and λ_3 are barycentric coordinates. They satisfy the constraint $\lambda_1+\lambda_2+\lambda_3=1$, thus there are only two independent components. If **x** is inside the convex hull, there is $\lambda_i>0$, i=1, 2, 3. Since barycentric coordinates are a linear transformation of Cartesian coordinates, it follows that they vary linearly along the edges and over the area of the triangle.

4.2. Binding out-of-plane point to triangle

Now we exploit the above in-plane barycentric coordinates to bind an out-of-plane point to a triangle. For a point **p** that is not coplanar with three vertices $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$, we use a quadruple (λ_1, λ_2, λ_3, d) for binding. First **p** is projected onto the plane along the plane normal $\mathbf{n}=(\mathbf{x}_2-\mathbf{x}_1)\times(\mathbf{x}_3-\mathbf{x}_2)/\|(\mathbf{x}_2-\mathbf{x}_1)\times(\mathbf{x}_3-\mathbf{x}_2)\|$. The projection point **p'** has in-plane barycentric coordinates (λ_1, λ_2, λ_3). Using the normal **n** to denote the positive side of the plane, we calculate the signed distance from **p** to the plane: $d=\langle\mathbf{p}-\mathbf{p'},\mathbf{n}\rangle$, where operator <,> stands for the inner product.

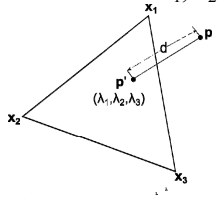
On the other hand for a new configuration of $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$, if the quadruple is known we can recover the Cartesian coordinate of point **p**:

$\mathbf{p}=\lambda_1*\mathbf{x}_1+\lambda_2*\mathbf{x}_2+\lambda_3*\mathbf{x}_3+d*\mathbf{n}$.

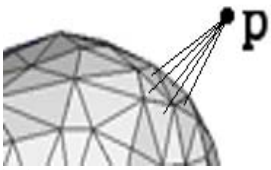
To be aware of the triangle area change caused by the new positions of $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$, we modify the above equation into

$$\mathbf{p}=\lambda_1*\mathbf{x}_1+\lambda_2*\mathbf{x}_2+\lambda_3*\mathbf{x}_3+\text{sqrt}(A/A_0)*d*\mathbf{n}, \quad (1)$$

where A_0 is the initial triangle area and A is the area of the new triangle. According to the above discussion, we compute λ_1, λ_2, λ_3, d and A_0 at the initialization step, and substitute them into Equ.1 to compute the new point position whenever $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ are re-located.



(a) bind **p** to one triangle



(b) bind **p** to multi-triangles

Figure 2 . Bind an out-of-plane point to triangles. point **p'** can be written as a weighted sum of triangles three vertexes.

4.3. Binding out-of-plane point to multiple triangles

So far we have introduced how to bind a point to one triangle. Usually a mesh consists of thousands of triangle, binding one point to only one triangle is prone to cause artifact. A better solution is binding one point to multiple triangles.

Suppose point **p** is to be bound to m triangles, we represent the final coordinate as a linear combination of m points: $\mathbf{p}=\sum_{i=0}^{m-1} w_i\mathbf{p}_i$, in which w_i is the weight of the i^{th} triangle.

We determine weight for each triangle only by to the point-to-triangle distance. The weight is the reciprocal of the distance. They are normalized to be within the range [0,1]. The formula for calculating weights is $w_i=\frac{1}{d_i\sum_{i=0}^{m-1}\frac{1}{d_i}}$. They satisfy the constraint $\sum_{i=0}^{m-1} w_i=1$.

4.4. The Coat-to-mesh algorithm

How do we decide the number of triangles from the body mesh that affect each garment vertex? The number should not either be too large or too small. If too large, the garment mesh will appear too rigid. If two small, the deformation will appear unsmooth. We let the user to determine this number and place a parameter in the configuration file to specify the maximum m triangles allowed to affect a point. In general, there are mountains of triangles in a character mesh. For each point, we find the nearest m triangles as the contributing triangles. This is done by the following steps:

Step 1: Create an array of m elements to store the m smallest point-to-triangle distances and a variable to save the index of maximum distances.

Step 2: Calculate distances one by one. Add the distance into the array until it's full.

Step 3: If the array is full, do comparison between the current distance and the largest value in array. Replace the largest value with the new one if the new one is bigger.

Step 4: Keep calculating until the last triangles.

Step 5: In order to guarantee the quality of binding, we filter the calculated m nearest triangles one more time. Set a distance range according to models' size and shape. Only triangles whose distances fall in the range can be preserved as affecting triangles. If no one satisfies this condition, just select the one whose distance is shortest as affecting triangle.

The pseudo-code for this process can be written as:

```
for (i←0 to n)  // n is total number of triangles
      do calculate the distance between p and the i^th
triangle.
      if i < m  // m is the max effecting triangles number
         put the distance in an m length array distance[]
         update maxIndex
      else
         if distance > distance[maxIndex]
            delete the maxIndex^th distance
            add the current distance in distance[]
            update max index
         end if
      end if
end for
for (i←0 to m)
     //reserve the shortest distance whatever
     if distance[i] is the shortest
       continue
     end if
     //the legal range of distance is [min, max]
     if distance[i]<min or distance[i]>max
       delete distance[i]
     end if
end for
```

The maximum number of affecting triangles *m* and the distance range is specified by user according to the size and the shape of the input model.

There are usually too many triangles in a mesh, therefore calculating all the point-to-triangle distances has quadratic complexity thus is time-consuming. We want to accelerate this process with some spatial data structure.

We determine an appropriate stride size, and divide the three dimensional space into small cubes of the same size. The point **p** is sure to fall in one and only one of them. Register a triangle to a cube if the cube either contains or overlaps with the triangle. To calculate point-to-triangle distance, first process triangles located within the same cube as **p**, then the neighboring cubes. With this method, the efficiency of the algorithm can be improved a lot.

5. Motion retargeting from BVH file

Once the binding is done, we could apply motion capture data to the character. However, we want to use BVH format file, which is one of the standard formats for the human skeleton animation, instead of the old specific animation data format Pinocchio use. So, we have to make a format conversion follow on the character rigging.

BVH file contains two pieces of information: the skeleton structure and the key frame data block. A human body skeleton structure described by a BVH file is shown in the figure below:

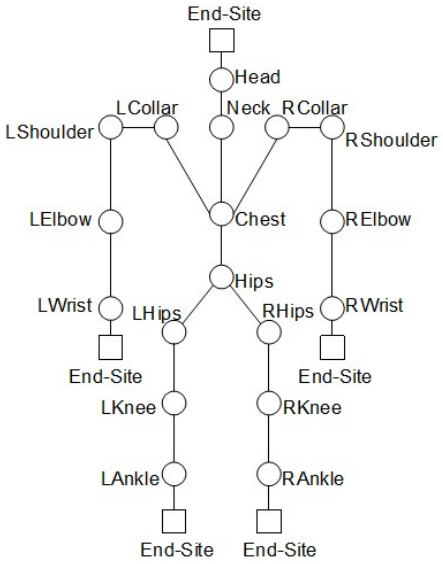


Figure 3. Virtual human skeleton structure described by BVH file

The rotation of the key points in BVH is Euler angles. So we need convert Euler angle to quaternion. How to transform Euler angle to the quaternion can be found in [19].

We then establish a mapping relationship between BVH skeleton and embedded skeleton. The animation data drive the BVH skeleton, and then indirectly drive the embedded skeleton, to produce the animation. An example of mapping diagram between BVH skeleton and embedded skeleton is shown below:

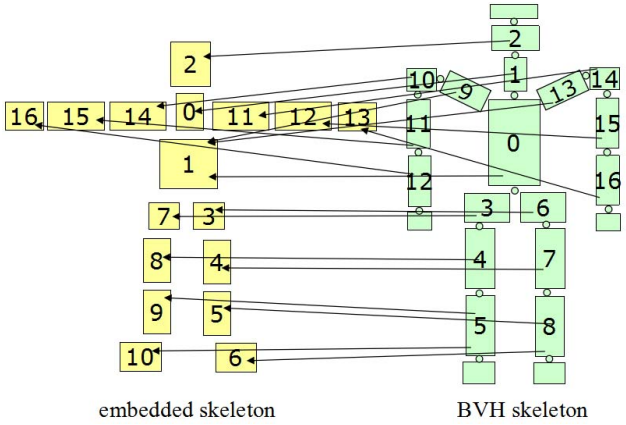


Figure 4. Mapping ralationship between embedded skeleton and BVH skeleton. Arrows indicate the mapping relationship. The animation data drive BVH skeleton, mapping to the embedded skeleton.

6. Experimental results

To test our dressed human animation system, we run various examples. Given a standard human body and several garments, we reshape the body model (as figure 5) and do garment fitting (as figure 6).

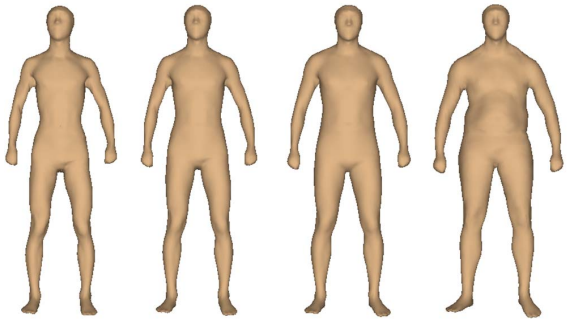


Figure 5. There are 6,449 vertices, 12,894 faces in this human model. We reshape it to different sizes without changing its topological.

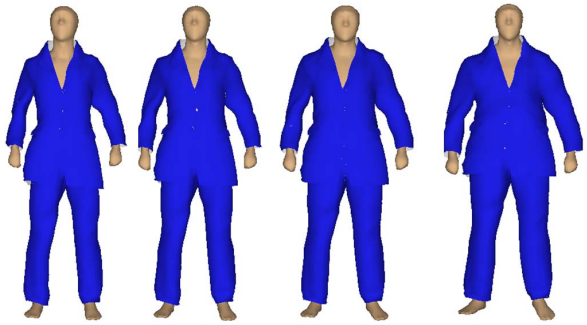


Figure 6. We dressed the human models with this suit. There are 3,101 vertices, 5,973 faces in it.

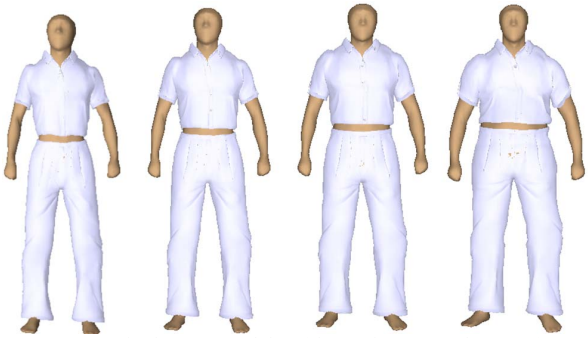
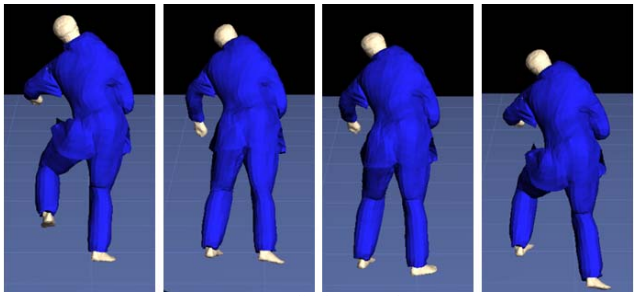
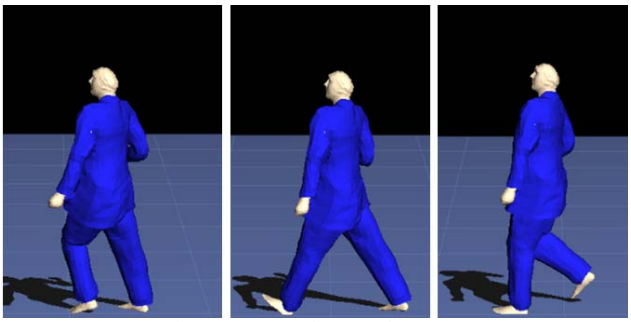


Figure 7. Dress the human models with another suit. This suit is much more precise. Its vertices number is 9,701 and faces number is 18,890.

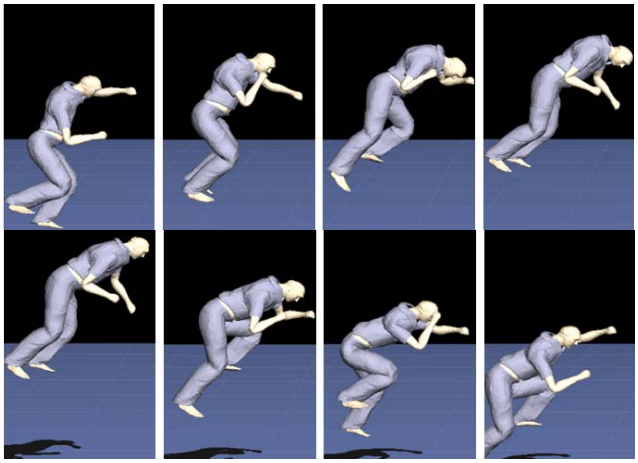
After body reshaping, garment fitting, skeleton embedding and skin attachment, we input the human body, the matched skeleton, the matched garment and the BVH animation file, vivid dressed human animation can be produced now. Figure 8 shows some screen-shots of two pieces of animation. One is a climbing male model; the other is the walking model. We dressed the model with two different suits. All these animations are generated in real-time.



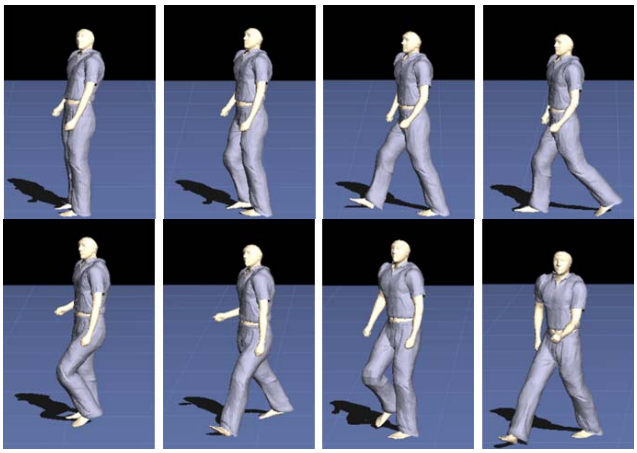
(a) Human climbing with suit #1



(b) Human walking with suit #1.



(c) Human climbing with suit #2.



(d) Human walking with suit #2.

Figure 8. Animation screen-shots

7. Discussions and future work

In summary, the virtual clothing framework provides a new tool, including body reshaping, garment refitting and garment animation, to create beautiful artworks. The implementation of the system relies on a series of sophistic techniques. We can envision the changes to happen to garment design and manufacturing, as well as animation production. Cloth animation effects for movies and games will be more realistic, and will be created in various ways. The garment industry will gradually change its traditional business mode, from design-craft-fit-purchase to design-fit-order-craft. With the new garment designing and modeling tool, the audience will go beyond the role of passive viewers and get more involved in the artistic process, either with their aesthetic appreciation, or with their body shapes and motions.

Acknowledgment

This work is partially supported by NSF China under the grant #61379096.

References

[1] A. Hilsmann and P. Eisert, "Tracking and retexturing cloth for realtime virtual clothing applications", in Proc. Mirage 2009—Comput.Vis./Comput. Graph. Collab. Technol. and App., Rocquencourt, France, May 2009, pp.94-105.

[2] P. Eisert and A. Hilsmann, "Realistic virtual try-on of clothes using real-time augmented reality methods", *IEEE COMSOCMMTCE-Lett.*, 2011, pp. 37–40.

[3] W. Zhang, T. Matsumoto, and J. Liu, "An intelligent fitting room using multi-camera perception", in Proc. Int. Conf. Intell. User Interfaces, 2008, pp. 60–69.

[4] B. Spanlang, T. Vassilev, and B. F. Buxton, "Compositing photographs with virtual clothes for design", in Proc. Int. Conf. Comput. Syst. And Technol., 2004, pp. 1–6.

[5] B. Spanlang, T. Vassilev, J. Walters, and B. F. Buxton, "A virtual clothing system for retail and design", Res. J. Textile and Apparel, 2005, pp. 74–87.

[6] R. Shilkrot, D. Cohen-Or, A. Shamir, L. Liu, "Garment Personalization via Identity Transfer", *IEEE Computer Graphics and Applications*, 2012, pp. 62-72.

[7] T. L. Thanh and A. Gagalowicz, "From interactive positioning to automatic try-on of garments", in Proc. Int. Conf. Comput. Vis./Comput. Graph. Collab. Technol., 2009, pp. 182–194.

[8] F. Cordier, W. Lee, H. Seo, and N. Magnenat-Thalmann, "Virtual try-on on the web", in Proc. Virtual Reality Int. Conf., Laval Virtual, 2001.

[9] Y. Meng, P. Y. Mok, and X. Jin, "Interactive virtual try-on clothing design systems", *Comput. Aid. Des.*, 2010, pp. 310–321.

[10] M. Wacker, M. Keckeisen, and S. Kimmerle, "Simulation and visualization of virtual textiles for virtual try-on", *Res. J. Textile and Apparel*, 2005, pp. 37–41.

[11] K. Zou, X. Xu, Y. Li, and Z. Li, "Research of interactive 3D virtual fitting room on web environment", in Proc. Int. Symp. Comput. Intell. and Des., 2011, pp. 32–35.

[12] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "SCAPE: Shape completion and animation of people", *ACM Trans. Graph.*, 2005, pp. 408–416.

[13] M. Fiala, "Magic mirror system with hand-held and wearable augmentations", in Proc. IEEE Int. Conf. Virtual Reality, 2007, pp. 251–254.

[14]C. Chu, Y. Tsai, C. Wang, "Exemplar-based statistical model for semantic parametric design of human body", TH Kwok - *Computers in Industry*, Elsevier, August, 2010, pp.541-549.

[15] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn and H.-P. SeidelA, "Statistical Model of Human Pose and Body Shape", *Computer Graphics Forum*, 2009, pp. 337–346, .

[16] S. Zhou, H. Fu, L. Liu, D. Cohen-Or, X. Han, "Parametric Reshaping of Human Bodies in Images", *ACM Transactions on Graphics* (Proceedings of SIGGRAPH) , 2010, vol. 29, no. 126.

[17] J. Li, J. Ye, Y. Wang, L. Bai, G. Lu. "Fitting 3D Garment Models onto Individual Human Models", *Computers and Graphics*, 2010, pp. 742-755.

[18] I. Baran,J. Popovic. "Automatic Rigging and Animation of 3D Characters", In Proc.SIGGRAPH, 2007, vol. 26(3), no.72.

[19] John Vince, Mathematics for Computer Graphics, Second Edition,Springer,2006.

[20] T. Akenine-Moller, E. Haines,Real-Time Rendering, Third Edition, India：AK Peters,2008.