Fast Collision Detection Between Cloth and a Deformable Human Body

Nuria Pelechano, Lee Bull, Mel Slater

Department of Computer Science, University College London, UK

Abstract

This paper addresses the problem of performing collision detection between two flexible objects, the skin and the cloth in real-time animation of virtual humans,. In order to achieve fast collision detection we propose a suitable oriented bounding boxes (OBB) hierarchy for deforming bodies that can be constructed in a pre-processing phase, and then updated efficiently during the animation. These OBBs give a coarse approximation of the human body that can be used for a preliminary test and therefore reduce the number of triangles against which collision detection needs to be tested.

1. Introduction

An important aspect of computer animation is the interaction between objects in the virtual world. During simulation of clothing on a human body we need to test not only whether a collision with the skin has occurred, but also find out and report the exact collision location of the intersection and the response that needs to be applied. Such response will modify the next position and velocity of the colliding vertices by applying new forces.

Since the computational requirement increases as the square of the number of polygons in the modelled scene, collision handling can cause substantial difficulties for maintaining the animation in real-time.

Both the skin and the cloth models are typically represented by polygonal meshes that can each have several thousands of polygons. It is therefore very important to develop a robust and efficient algorithm to achieve real-time performance for large models.

There has been substantial research on collision detection for simulation of rigid bodies. Such methods are based on data structures that can be pre-computed and then used during run- time. In our case we are dealing with deformable bodies, so we need to either recompute these data structures at each time step or update them depending on the movements of the human body.

There are some principles that can be followed for an efficient collision detection algorithm:

- Reduce the number of potential colliders by applying a fast simple test.
- Use as much information as possible about the geometry.
- Exploit locality to reduce the number of potential collisions.
- Exploit frame-to-frame coherence between successive tests.

In this paper we present a new algorithm for fast and efficient collision detection between cloth and a deformable human body. The algorithm is based on the approach of spatially partitioned oriented bounding box (OBB) volumes. Our main contributions are:

- A coarse representation of the human body given by OBB, where for each bone there is an OBB with a spatial subdivision so that the number of triangles to test against is considerably reduced.
- A fast collision detection test between vertices of the cloth and the oriented bounding volumes.
- A fast collision detection vertex-triangle test using a projection method.

The remainder of this paper is organised as follows. In next section we briefly summarise the previous relevant work in collision detection. Section 3 presents the Avatar Toolkit used as the basis for this work. Section 4 discusses the details of our algorithm and its implementation. Section 5 demonstrates the results obtained. Finally we conclude and describe future work.

2 Previous Work

Collision detection can be a bottleneck in cloth simulation. In order to reduce its complexity there have been many approaches based on the idea of using some kind of bounding volume hierarchy. These bounding volumes (BV) are used to speed up the intersection test.

BVs that have been used to build coarse representations of the objects are for example spheres [1], ellipsoids [2], Axis Aligned Bounding Boxes (AABB) [3], Oriented Bounding Boxes (OBB) [4], Binary Space Partition Trees (BSP-Trees) [5], Quantized Orientation Slabs with Primary Orientations (QuOSPOs) [6]. Bandi and Thalmann proposed an adaptive spatial subdivision of the object space based on Oc-tree structure [7].

Another popular approach to collision detection is based on Voronoi diagrams [3], where spatial and temporal coherence has also been exploited.

Collision detection is performed between the cloth and the skin. Both elements are represented by a 3D polygonal mesh.

There are several cloth models in the literature, but the most generalised is composed of a network of masses and springs [8, 9], where movement is given by the fundamental law of dynamics $F=m\cdot a$. In the model described by Provot a regular quadrilateral mesh of n×m virtual masses is used. In this model, each mass is linked to its neighbours by massless springs of length non zero at rest. It also contains three different kinds of links in order to give different properties to the cloth.

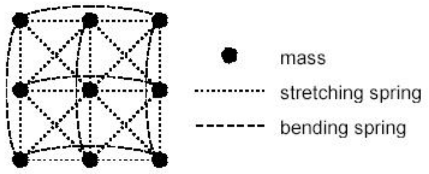

Figure 1. Mass-spring cloth model

Cloth movement is affected by two different kinds of forces, internal and external. The former results from the tension of the springs linking the masses, and the later can have several natures such as gravity, wind, viscous damping, etc [8, 10].

The model used in this paper is based on the method employed by Vollino, Courchesne and M. Thalmann [11], reformulated by Anderson [12] which consists in a physical model based on Newtonian mechanics acting on irregular triangle meshes.

The skin is usually modelled as an elastic triangle-mesh surface generated typically with a modelling program. This skin is then attached to the underlying bone hierarchy of the avatar and in some more complicated avatars is also attached to the underlying muscle model [13]. Changes in the joints of the skeleton result in changes to the position of the bones and therefore in the skin attached to those bones. Thalmann [14] introduced joint-dependent local deformation operators named JLDs to be able to control deformations in the vicinity of joints.

It is also possible to group the vertices into *contours* [15] and deform all of them together in order to speed up the deformation process.

Some other techniques with higher computational cost are based in the use of Free Form Deformations (FFDs) [16], and also Rational FFD or Dirichlet FFD in order to allow a more accurate control of the deformations.

Other approaches define the skin by metaballs, which provide realistic-looking virtual humans at a reduced storage cost [17].

3 Prometheus Avatar Toolkit

The PROMETHEUS project is a consortium which is involved in developing a studio broadcast platform for virtual 3D TV. Its aim is to allow television production using virtual sets and characters driven by real actors using a motion capture system. The system supports skin deformation and also contains facial simulation using motion capture [18].

Avatars are represented using a proprietary skinning system developed by the University of Surrey [19].

The articulated body is represented by a tree structure that represents the parent-child relationship between successive bones of the skeleton. Each bone has a default rotation and translation that gives its position in relation with the position of its parent, initially in a standard neutral pose.

The complete avatar is represented by a layered model given by a skeleton and the skin.

The avatars used in this project are seamless models. Each 3D model is defined as a single surface instead of a number of separate segments, so that it gives the illusion of a single surface. Each bone has an associated subset of all the vertices representing the whole model. While animation is being performed each bone affects the position of its particular subset of vertices in order to modify the complete shape of the body.

The Cloth Simulation Server within the Prometheus project contains:

- A rendering engine based on VRML97/H-Anim
- Cloth simulation system using a method based on Anderson [12] and Thalmann [20]
- Voxel collision detection (cloth/avatar) for segmented avatars without skin deformations, using Distance Fields.
- An animation system based on event sequencing of motion capture and other events from files or network streams.
- External buffer servers to handle time stamped data entering and leaving the cloth server system.

More information about the Prometheus Avatar Toolkit can be found in Prometheus project web sites [21, 22].

4 Our Algorithm

With respect to real-time systems it is impracticable to test every vertex or polygon of the cloth against every polygon of the skin. It is desirable to apply an efficient detection method whose complexity is independent of the complexity of the objects. In this paper we present a collision detection method based on OBBs.

This work has been implemented in C++ with OpenGL and is included in an already existing environment, the Prometheus Avatar Toolkit.

4.1 Oriented Bounding Boxes

Our main motivation to choose OBB against other kinds of bounding volumes is that not only can they bound geometry quite tightly but also they offer variable orientation and can easily be updated.

The skeleton of the avatar is already subdivided into segments corresponding to the bones. Each bone has associated a subset of vertices from the whole set of vertices describing the skin. This subset of vertices will be used to compute the OBB.

An OBB is defined by a centre and three axes defining the base of its coordinate system. The centre is computed by the mean of the coordinates of all the vertices associated with a particular bone. The three axes are easily computed by obtaining the covariance matrix of the subset of vertices and then obtaining its eigenvectors.

The covariance matrix is given by:

$$S = \frac{1}{n}\sum_{i=0}^{n}\left(X_i - \overline{X}\right)^t\left(X_i - \overline{X}\right)$$

Where $X_i = (x_i,y_i,z_i)$ are the 3D coordinates of the skin vertices.

The eigenvectors are obtained by the *Principal Components Analysis* method (PCA).

Finally we need to compute the size of the OBB, which is given by the maximum and minimum extents of the original set of vertices along each axis.

4.2. Subdivision of OBBs

In order to substantially reduce the number of vertices falling in each subspace, we subdivide the OBB into several parts. This is done by partitioning the main axis of the OBB and then placing orthogonal planes at each of the partitions.

During simulation, once a vertex is found to intersect an OBB, we only need to project it onto the main axis of that OBB to find out to which subdivision it belongs. Then we will perform collision detection tests only against the triangles falling in that subspace of the OBB.

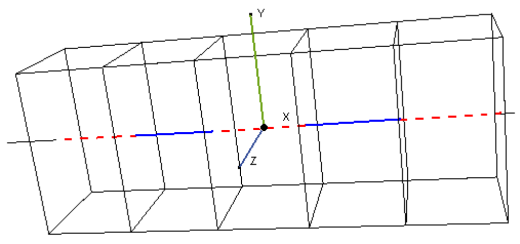

Figure 2. OBB with five subdivisions along the X-axis

In the example above there are five subdivisions along the principal axis of the OBB. Each subdivision will have associated the list of triangles which vertices are projected into it. The more subdivisions we have, the fewer triangles we will have to test against. Searching for the list of vertices is done using a look-up-table, so the computational cost will not be increased by the fact of scaling the number of subdivisions.

4.3. OBBs update during simulation

The method to compute the OBB has linear cost with the number of vertices in the object. Since the system must run in real-time, we cannot afford to compute the OBB for each step of the animation. Therefore we need to find a method to modify the OBB with a computational cost independent of the complexity of the model.

The main idea of our algorithm is to avoid recomputing the OBB with high cost but also without losing robustness.

The main axis of the OBB can be easily updated by applying the matrix transformation of the bone to which it is associated. The only problem now would be to recompute the size of the OBB. Since skin deformations can be approximately predicted within a certain area around the bone, our method simply allows some tolerance in the initial values of the size, so that we obtain a slightly bigger OBB. Therefore this initial value for each OBB size can be used during simulation time without any need to be updated.

Although this decision may lead to some additional collision detection tests afterwards, we save a lot of computational time updating the OBBs.

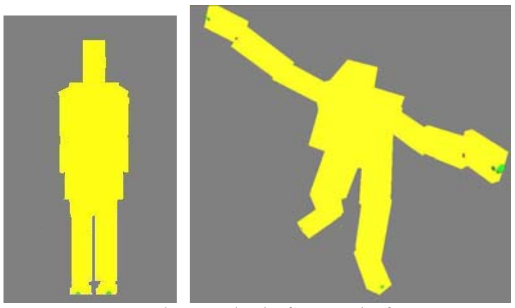
Figure 3. OBB hierarchy before and after animation

4.4 Collision Detection

Collision detection is tested between a vertex of the cloth and a triangle of the skin. This approximates true polygon-to-polygon collision detection for increased efficiency. Before this collision detection is performed we apply some tests to reduce the number of triangles against which we need to test each vertex of the cloth.

The first test is to determine whether the vertex is inside any of the OBBs representing the whole body.

This could be done by testing the cloth vertex against the six planes that define the OBB. A more efficient way to solve this problem consists of transforming the world coordinates of the vertex into local coordinates of the base that define the OBB [23]. Once this mapping has been applied, we only need to compare the coordinates of the vertex against the top right corner of the OBB.

The next step consists in determining which subdivision within the OBB the vertex lies. This is done by simple projection onto the main axis and then

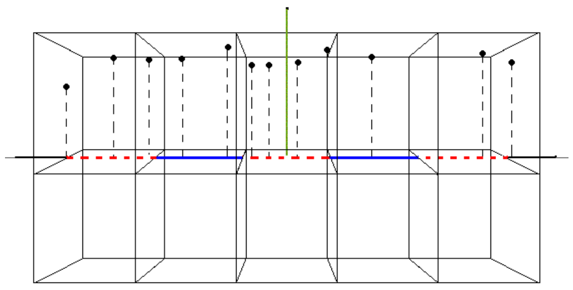
Figure 4. Cloth vertices projected into the main axis.

Since all the vertices have been mapped to local coordinates of the OBB, the projection into the X-axis is reduced to the X coordinate of each vertex.

Once the segment has been determined, a look-up table can be used to find out the list of vertices against which we need to finally apply the collision detection test. To give robustness to our algorithm we allow that vertices of the skin lying in the border between to subdivisions to belong to both lists of vertices. This implies more storage cost but saving computational time since there is no need of updating the lists of vertices during simulation.

Geometrical collision detection can consist of detecting whether the objects interpenetrate (interference detection), or the objects come closer than a given threshold distance (proximity detection). In our algorithm we perform proximity detection by testing collision against an offset surface of the skinned avatar to alleviate polygon intersections.

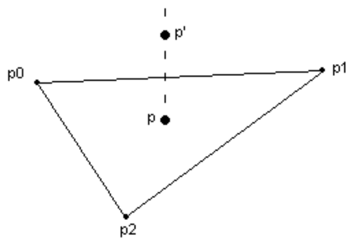

Figure 5. Cloth vertex p' projected onto the triangle plane in the location p.

To find out if a vertex intersects a triangle, we first compute the distance between the vertex and the plane given by the triangle. If that distance is less than a certain threshold we consider that there may be an intersection in the location p.

Once we have the point p, which obviously lies on the plane, we have to find out whether p is inside the polygon itself or outside. The problem now is therefore to determine if the point *p* given in 3D space is inside a polygon given also in 3D space. Since this is not an easy problem to be solved in 3D we change the problem to work in 2D space using a projection.

4.5 Projection

This method consists in projecting both the point *p* and the vertices of the polygon onto one of the principal planes (XY, XZ or YZ) [23].

The main idea of this method is that the inside/outside relationship between the projection point p and the projected polygon would be the same as the original point did to the original polygon. The principal plane to chose for the projection will be the one that is somehow most parallel to the plane defined by the triangle. This can be obtained by choosing the plane that minimises the angle between the normal to the plane of the triangle and the normal to the principal plane of projection.

4.6 Response

For every frame of the simulation, the velocity and position of every mass point in the cloth mesh needs to be computed. These new positions and velocities are given by the solving of a system of ordinary differential equations (ODEs). Several forces such as internal cloth, gravity, friction, wind and damping are included. Vertices are made to move with the character using a point modification method, by passing the solver. When a geometric intersection occurs, the mass point is set to the point of intersection. The vertex is tagged as being on the surface and until it leaves, will have all forces and velocity acting towards the surface removed.

Intersection with the outer threshold offers the potential of hard impact responses. If further penetration into the surface beyond the surface threshold occurs, a penalty force is applied that acts along the direction of the surface normal. This is a method commonly used to move mass points away from a surface.

The penalty force is given by Newton's Law: $F = m \cdot a$. Where m is the mass and a the acceleration we want to apply to the movement of the vertex. The magnitude of the acceleration is proportional to the negated distance between the vertex and the surface, and the direction $\vec{N}$ will be normal to the surface.

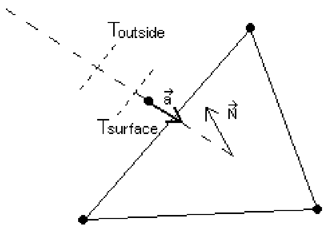

Figure 6. 3D Intersection vertex-triangle

The penalty force returned is added to the total force applied to the vertex in order to modify the direction of the movement for the following frames of the simulation and thus avoid intersection with the skin.

5 Results

In order to demonstrate the capability of our collision detection algorithm, several pieces of cloth were designed. The example in figure 7 shows an avatar dressed with a skirt, and figures 8 and 9 show a square piece of cloth falling onto the avatar's shoulder.

Our algorithm has been tested using an avatar described by 1290 vertices (2576 polygons to test against) and a piece of cloth model containing 264 vertices.

Even though the OBB are updated during the animation, so far we have conducted collision detection experiments only on a static human body.

The first step of our algorithm is to represent the avatar using a hierarchy of OBBs [1]. There is one OBB for each bone of the avatar, and each OBB is regularly subdivided in 15 subspaces. This coarse representation of the original model dramatically reduces the number of triangles against which collision detection needs to be tested. Our optimisation allows an overall reduction of about 98% in the number of triangles to test against each cloth vertex.

The algorithm we have implemented not only drastically reduces the number of collision detection tests to be done but also is also

[1] The skeleton is represented by a hierarchy of bones. Each bone has as a child node a OBB. These OBBs are therefore the leaves of the tree.

scalable since the percentage of reduction is independent of the complexity of the model.

Our experiments have been done using an 800MHz machine running Windows 2000 with an nVidia GeForce2 card. If we apply collision detection directly testing every vertex of the cloth against every polygon of the skin, the frame rate becomes 0.2% of what it was without collision detection. Applying our algorithm, the frame rate becomes 35% when collision detection and response are being performed. If we only consider collision detection, then the frame rate becomes 63% of what it was without collision detection

6. Conclusions and future work

Collision detection for Real-time graphics simulations, where the shapes of the bodies deform continuously over time, constitute a particular challenge since the possibilities of using pre-calculated data and data structures are dramatically decreased.

The method proposed in this paper is based on an OBB bounding volume hierarchy. OBBs can be constructed during a pre-process and can be updated quickly.

The result of this work is an efficient collision detection algorithm that works well for real-time simulations of deformable human bodies represented by seamless polygonal meshes.

Further work that could be undertaken includes:

- Using a space subdivision also for the cloth model or using adjacency graphs to narrow the number of collision detection tests.
- Using frame-to-frame coherence to estimate the most likely vertices to collide in the following frames of the animation and discard those that are far away from a potential collider.
- Using information about the viewpoint to perform collision detection with different levels of accuracy depending on how visible a vertex is from the current viewpoint.
- Improving the quality of the physics simulation and also avoiding self collision detection among vertices of the cloth.

Acknowledgments

I would like to thank Jonathan Starck of the University of Surrey for the PROMETHEUS Avatar Toolkit. Lee Bull's work is funded partially by the BBC and partially by the EPSRC under research grant GR/M88068/01.

References

1. P.M. Hubbard. Approximating Polyhedra with Spheres for Time-Critical Collision Detection. *ACM Transaction on Graphics*, 15(3): 179-210, July 1996. 2
2. I. Rudomín, J.L. Castillo. Real-Time Clothing: Geometry and Physics. WSCG 2002 Conference Program. 2
3. J.D. Cohen, M.C. Lin, D. Manocha, M.K. Ponamgi. I-COLLIDE: An Interactive and Exact Collision Detection System for Large-Scale Environments. *Proc. Of ACM Interactive 3D Graphics Conference*, pp. 189-196, 1995. 2
4. M.C. Lin, D. Manocha, J. Cohen and S. Gottschalk. Collision Detection: Algorithms and Applications. In JP. Laumond and M. Overmars, editors, *Algorithms for robotic motion and manipulation,* pages 129-142. A.K. Peters, Wellesley, MA, 1997. 2
5. B. Naylor, J.A. Amatodes, W. Thibault. Merging BSP Trees Yields Polyhedral Set Operations. *ACM Computer Graphics* (Proc. of SIGGRAPH'90), 24(4) pp. 115-124, 1990. 2
6. T. He, Fast Collision Detection Using QuOSPO Trees. Symposium on Interactive 3D Techniques, Proceedings of the 1999. pp. 55-62, Atlanta, GA USA, 1999. 2
7. Bandi, S., Thalmann, D., An Adaptive Spatial Subdivision of the Object Space for Fast Collision Detection of Animating Rigid Bodies. *Eurographics'95* (Aug. 1995), Maastricht, pp. 259-270. 2
8. Provot X. Deformation constraints in a mass-spring model to describe rigid cloth behaviour. *Proceedings of Graphics Interface* 1995; pp 147-155. 2
9. Louchet, J., Provot, X., Crochemore, D., Evolutionary identification of cloth animation models. *Eurographics Workshop on Animation and Simulation*, Maastricht. 1995. 2

10. T.I. Vassilev, B. Spanlang, Y. Chrysanthou. Efficient Cloth Model and Collisions Detection for Dressing Virtual People. Presented at the *ACM/EG Games Technology Conference*, January 2001. 2

11. P. Vollino, M. Courchesne, M. Thalmann. Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects, *Computer Graphics Proceedings*, Annual Conference Series, pp 137-144, 1995. 2

12. James N. Anderson. Fast Physical Simulation of Virtual Clothing based on Multilevel Approximation Strategies. PhD the sis, the University of Edinburgh, 1998. 2, 3

13. P.J. Schneider, J. Wilhelms, Hybrid Anatomically Based Modelling Of Animals. In *Proceedings of Computer Animation 1998,* pp. 161-169. 2

14. A. Aubel, D. Thalmann, Realistic Deformation of Human Body Shapes. Computer. *Proc. Computer Animation and Simulation 2000 (11th Eurographics Workshop*), Interlaken, pp. 125-135. August 2000. 2

15. P. Kalra, N.M. Thalmann, L. Moccozet, G. Sannier, A. Aubel, D. Thalmann, Real-time Animation of Realistic Virtual Humans. *IEEE Computer Graphics and Applications*, pp. 42–56, Sept. 1998. 2

16. M. Henne, A Constraint-Based Skin Model For Human Figure Animation. Master's thesis, University of California, Santa Cruz, Santa Cruz, CA 95064, June 1990. 2

17. S. Yoshimito, Ballerinas Generated by a Personal Computer. Visualization and Computer Animation, Vol. 3, No. 1, pp. 85-90 (1992). 2

18. M. Price, J. Chandaria, O. Grau, G.A. Thomas, D. Chatting, J. Thorne, G. Milnthorpe, P. Woodward, L. Bull, E-J. Ong, A. Hilton, J. Mitchelson, J. Starck. Real-time production and delivery of 3D media *Proc. International Broadcasting Convention*, Amsterdam, Netherlands, September 2002. 2

19. Smith, R., Hilton, A., and Sun, W. Seamless VRML Humans, UK, 24 May, 2000. 2

20. Magnenat Thalmann N., Carion S., Courchesne M., Volino P., Wu Y., Virtual Clothes, Hair and Skin for Beautiful Top Models, *Computer Graphics International '96*, Pohang, Korea, 1996, pp.132-141. 3

21. http://www.cs.ucl.ac.uk/research/vr/Projects/Prometheus/index.html 3

22. http://www.bbc.co.uk/rd/projects/prometheus/ 3

23. M. Slater, A. Steed, Y. Chrysanthou, Computer Graphics and Virtual Environments, From Realism to Real-Time. *Addison Wesley.* 4, 5

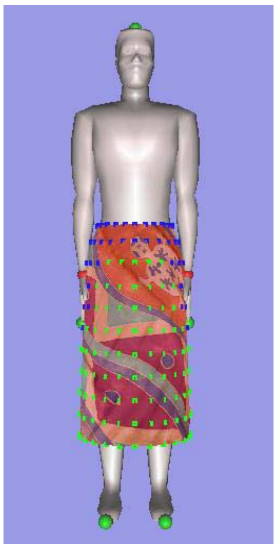

Figure 7. Skirt on an avatar

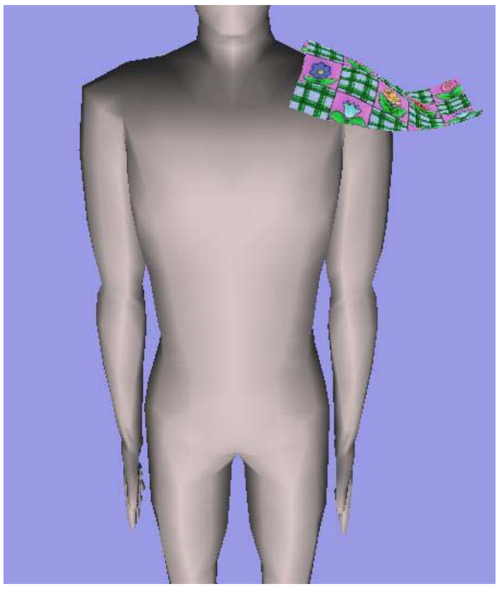

Figure 8. Piece of cloth falling onto the avatar's shoulder

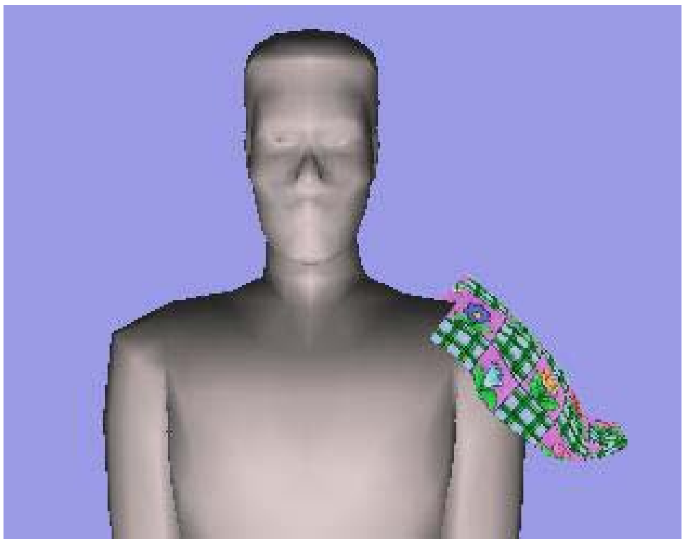

Figure 9. Piece of cloth falling onto the avatar's shoulder