Sketch-based Simulated Draping for Indian Garments

Sanjeev Muralikrishnan  
Department of Computer Science & Engineering  
IIT Bombay  
Powai, Mumbai - 400076, India.  
smk@cse.iitb.ac.in

Parag Chaudhuri  
Department of Computer Science & Engineering  
IIT Bombay  
Powai, Mumbai - 400076, India.  
paragc@cse.iitb.ac.in

ABSTRACT
Virtual garments like shirts and trousers are created from 2D patterns stitched over 3D models. However, Indian garments, like dhotis and saris, pose a unique draping challenge for physically-simulated garment systems, as they are not stitched garments. We present a method to intuitively specify the parameters governing the drape of an Indian garment using a sketch-based interface. We then interpret the sketch strokes to procedural, physically-simulated draping routines to wrap, pin and tuck the garments around the body mesh as needed. After draping, the garments are ready to be simulated and used during animation as required. We present several examples of our draping technique.

CCS Concepts
• Computing methodologies → Computer graphics;

Keywords
Automated garment draping; Indian garment draping; sketch-based

1. INTRODUCTION
Virtual garment simulation has been the topic of a lot of research in computer graphics. This is because the problem is of interest not only to character animation in animated productions, but also for the garment retail business and industry. It is because of this interest that virtual garments have traditionally been stitched from 2D patterns placed around the body mesh of the character. The stitching is simulated by adding additional forces or seams that hold the 2D patterns together. An example of this can be seen in Figure 2.

In contrast, many Indian garments, like dhotis and saris, typically consist of a single piece of cloth of a certain length draped over the body. During draping, this single piece of cloth may wrapped around the body multiple times, pleats may be added to a part of the cloth, some parts may be tucked, while other may be pinned. These drapes require skill, practice and dexterity from real users who put on these garments. An example of a complex simulated drape, created with our system can be seen in Figure 1.

Simulating these drapes virtually poses unique challenges. Firstly, the authoring of the drape is non-trivial. An intuitive interface is required to drape these garments over a virtual character. Next, while draping the garments is to be pleated and pinned at various places. Finally, this draping process must be implemented on a robust physically-based cloth simulation system and collision system so that the garment can be animated with the character, after it has been draped.

We provide a generic, intuitive sketching interface to enable users to drape Indian garments on their characters. The drape itself is automated and a total of 6 different types of garment drapes have been implemented. The user’s sketch inputs are interpreted and mapped to one of these six types of drapes. The draping procedure is implemented on top of the cloth simulator in Blender 3. The sketch interface and the draping procedures have been implemented as a Blender Add-On. The draping methods, however, are generic and can be implemented on top of any cloth simulator that pro-
provides the ability of assigning mesh vertices as anchoring points.

We start with a look at the background literature in the area of virtual garment simulation and draping in Section 2. We then present the details of our sketch-based interface and detail the challenges involved in the draping system and our solutions to them in Section 3. Finally we present example drapes created with our system in Section 4 and conclude with a discussion of future work in Section 5.

2. BACKGROUND

Virtual cloth simulation was introduced to computer graphics as elastic deformable models [14] and mass-spring models [11]. Baraff and Witkin [1] proposed a cloth model based on with an fully implicit solver. Over time, cloth models that use semi-implicit solvers [6] or central time differencing [4] methods have also been proposed. During cloth simulation, since bending energy computations are expensive, some authors have proposed alternate models to compute bend energy [8] [18]. Unstructured triangular meshes admit more accurate Lagrangian finite element solutions as presented by [10]. Some authors have also simulated in-extensible cloth using the idea of strain limiting [7]. Perceptual control for cloth simulation has also been recently looked at by Sigal et al. [13].

The particular problem of garment draping has also been investigated. Earlier interactive computer aided design of virtual garments bring together the editing of 2D patterns and the 3D garments themselves in one interface, for creating designer friendly environments [17] [16]. Recent work by Berthouzoz et al. [2] has also focused on automatically parsing 2D patterns to assemble the 3D garment. There have also been other attempts to provide more accurate collision handling while stitching the 2D patterns [12].

Sketch-based garment draping to simple, stitched garments was presented by [15]. Authors have also looked at self-dressing virtual humans. Ho and Komura [9] used topology constraints to animate a character stretch out its hands out of a piece of clothing. Wang et al. [20] use an electrostatic parameterization of the empty space around the character’s body to make them pull their limbs out of the garment in the direction of electric flux. Clegg et al. [5] animate the process of human dressing by identifying a set of primitive actions performed during dressing and then using a controller to select and perform each action. The controller plans the path the limbs must follow based on current garment configuration at each step. This method however, uses a simulator that considers the cloth mass-less in comparison to the body mass of the character. This prevents the system from handling the draping of tight and well fitting clothes.

Our system is complementary to these. We do not attempt to have the virtual character animated during draping, however, the drape itself is animated. We also have identified a set of primitive draping actions that are combined to perform a full drape. Our techniques however, can be added to any cloth simulator that can robustly handle multiple cloth-cloth and cloth-object collisions. To the best of our knowledge our method offers the only solution to the problem of Indian garment draping.

3. THE DRAPING SYSTEM

A garment drape consists of two steps. First, the cloth is placed on a target location on body, leaving one or more loose ends. Next, the cloth is wrapped around the body by moving the free ends. Different Indian garment drapings vary in either of these two steps. For example, a dupatta or scarf is placed around the neck while a dhoti is placed around the waist. Also, the different styles of dhotis and saris vary in the way they are wrapped around the body.

We mimic this same overall process while draping the virtual garments. We use a tessellated plane mesh to represent a cloth. We first place it at a canonical position as identified by some sketch strokes. These sketch inputs are further interpreted to identify the draping procedure to be applied on the cloth mesh. Once identified, the procedure is applied on the mesh and the draping happens automatically. The draping thus depends on the underlying cloth simulator to provide the feature of specifying mesh vertices as anchor points. Once defined as anchor points, these mesh vertices can be used to tug the free end of the garment and thus drape it around the 3D model. The sketch inputs that are provided by the user are described in the next section.

3.1 The sketch-based interface

The sketch stroke inputs provided by the user are used to determine the canonical position, the size of the garment and the draping procedure to apply. To further aid the user in specifying this drape, we allow the user to sketch on a reference image of the drape of their choosing. The image of the garment being draped is placed in front of the character model, such that the waistline of the image is aligned to the waist of the 3D model. The limbs in the image are also roughly aligned to the target character’s body. This placement is done manually. An example can be seen in Figure 5 where an image of a dhoti has been placed in front of the 3D character.

Next the user sketches the boundaries of the garment portrayed in the image using a sketching tool. In our particular implementation, we do this using the grease pencil tool in Blender. The sketch strokes that the user must mark on the garment are as follows.

- **Width Stroke:** The first stroke is marked along the waist of the garment in the image. This stroke is used to determine the waist width of the garment, and
Figure 3: Position the garment image in front of the character model.

height of the initial garment with respect to the character’s body. This stroke is shown below in Figure 4(a).

• Depth Stroke: The second stroke is marked along the depth of the character, on the waist of the 3D model. This stroke is used to determine the depth of the waist. This stroke is shown in Figure 4(b).

• Height Stroke: The third stroke is marked along the left or right boundary of the garment in the image. This stroke helps determine the height of the garment. An example of this stroke has been shown in Figure 4(c).

• First Bottom Stroke: This stroke is drawn along the bottom side of the garment in the image. It could be along the inner side of the feet or along floor of the image depending on the garment. This stroke is used to determine the leg spread of the character. It also helps identify if the bottom part of the garment is simply dropped down or tugged upwards. This latter decision is made by checking if this particular stroke fits a straight line better than a higher order curve. If this stroke is a closer fit to a higher order curve, it indicates the bottom side of the garment has been picked up while draping. If instead, it is closer fit to a straight line, it indicates the bottom side has been left as is. This is used to identify the type of drape to perform. An example of this stroke is shown Figure 5(a).

Figure 5: Examples of the two bottom strokes are shown.

• Second Bottom Stroke: This stroke is of the same category as the previous stroke, and is optional. It may or may not be required depending on the garment. This stroke is used for the same purposes as the fourth stroke, and augments the information provided by the fourth stroke. Certain types of garments like a lungi may not require this stroke as the front view of these garments, when draped do not show 2 distinguishable boundaries along the bottom. An example of this stroke is shown Figure 5(b).

3.2 The canonical drape start position

All the strokes described above, in addition to helping identify the required drape, also help determine the dimensions of the garment and it’s position in the 3D workspace. This known are the canonical drape start position. Since the drape is started from this position it is very important to achieve correct placement of the garment around the character. The canonical position leaves one or two free ends which are used to perform the actual drape. An example canonical drape is shown Figure 6. In the example shown, there are two free ends protruding from the front. The waist is approximated by an ellipse based on the first two strokes as
described in the earlier. More precisely, A part of the ellipse is C-shaped curve that matches the perimeter of the ellipse is extruded along the direction of the legs of the character. We then scale the bottom of the extruded mesh outward, along the "floor", to obtain a cone-like canonical position for the garment. The bottom strokes are also used to determine the leg spread of the humanoid. This is required to get the correct canonical position that encompasses the legs of the character inside the cloth. The legs should not penetrate the cloth and protrude outwards before the draping starts.

![Figure 6: Canonical position with free ends, poised for draping.](image)

4. RESULTS

We show the various drapes that we have created using our system. We also show the sketch inputs given for each of the 6 drape styles. For each drape result shown in this section, there is a depth stroke which is not visible in the front view. The side view has been omitted for brevity, but it is important to note that the depth stroke is present for each.

The reader should refer to the supplementary video submitted with the paper for animation results of the various drapes presented here.

4.1 Simple, pleated dhoti drape

In this drape the cloth mesh is started from the canonical position and then the extra cloth at on the free ends is folded into pleats and tucked at the front of the waist. Figure 7 shows the sketch input and the corresponding final drape produced.

![Figure 7: The sketch input strokes and the resulting simple, pleated dhoti drape are shown.](image)

4.2 Lungi drape

A common style of wearing the lungi is to drape the cloth around the waist and pick up the bottom, fold it up and tuck at the waist. An example of this style is shown in Figure 8.

![Figure 8: The sketch input strokes and the resulting household lungi drape are shown.](image)

4.3 Simple wrap around drape

Another common style is to wrap the cloth round the waist multiple times and then tuck it at some point on the waist. An example of this style is shown in Figure 9.

4.4 Pleated and tucked back drape

In this drape, after the initial wrap around, the cloth is left with two free ends. One end is taken from the beneath and tucked at the back. The other end is folded into pleats and tucked at the front. An example of this drape can be
4.5 Kshatriya style drape

This dhoti drape style is similar to the previous one. There are two free ends, which are both taken from beneath to be tucked behind. In a more accurate version, one of the free ends after being tucked behind at one point, would also be taken around the waist to tighten, and then again tucked behind. An example of this drape can be seen in Figure 10.

Figure 10: The input sketch and the resulting Kshatriya style drape are shown.

4.6 Monk style sari

In this drape the cloth is taken around the waist multiple times, and then the remaining cloth is hung over the shoulder to fall over the back, as worn by monks in monasteries. Figure 11 shows an example of this drape with the sketch strokes we use to generate it.

Figure 11: The input sketch and the resulting Monk style sari drape are shown.

Consecutive actions of a draping procedure must be separated by a sufficient number of frames. We found that if an action of the draping procedure is done too fast, it results in the garment being pulled to one side, giving it an asymmetric drape. Each action should be given enough time, to allow for the cloth to settle and to prevent generation of sudden high intensity forces.

4.7 Discussion

We faced multiple challenges in creating our draping system. There are some more features that can be implemented to allow for even more complicated drapes. We discuss these desired features and some implementation specific details in this section.

4.7.1 Key frame separation

Consecutive actions of a draping procedure must be separated by a sufficient number of frames. We found that if an action of the draping procedure is done too fast, it results in the garment being pulled to one side, giving it an asymmetric drape. Each action should be given enough time, to allow for the cloth to settle and to prevent generation of sudden high intensity forces.

4.7.2 Dynamic vertex anchoring

Vertex sets can be marked as anchored usually prior to a cloth simulation is started. Dynamic anchoring the ability to pin and unpin vertices during the simulation. Moreover, pinning and unpinning of vertices can also then be keyframed. This feature is currently absent in Blender. For example, while draping a sari, the part of the sari that should be dropped over the shoulder has a free end that should fall under gravity. However, since the free end vertices are pinned at the start to allow this part to be moved over the shoulder, they cannot freely fall under gravity. The ability to dynamically unpin these vertices would solve this problem.

4.7.3 Adaptive local re-meshing of cloth mesh and adaptive time-stepping

The draping process typically involves a lot of cloth-cloth interaction. The amount these interaction vary over the course of a drape. It would be beneficial to be able dynamically reduce the time-step of the cloth simulator when fine collisions need to be resolved, and then revert back to larger time-steps. This would allow accurate yet fast simulation. Adaptive re-meshing is the ability to locally re-mesh the cloth during the simulation to smaller or larger elements as required, to resolve finer geometric details on the garment. This would also help in creating finer pleats and wrinkles in the cloth without increasing the simulation substantially, as would happen if the entire cloth is re-meshed to smaller
sized elements everywhere. A possible method of doing this kind of re-meshing is presented by Narain et al. [10].

4.7.4 Vertex management in Blender

Blender provides a raw vertex access API which allows access to the cloth mesh vertices and their global and local coordinates. However, for identifying the anchor points and for manipulating them, different kinds of multi-vertex selection queries are required. Typical vertex selection queries include:

- Select a chain of vertices starting from current vertex along a direction given by a vector.
- Select every $n^{th}$ vertex from the current vertex along a direction given by a vector.
- Select the vertices with the top $k$ Y coordinates.
- Select $m$ vertices from the current vertex along a particular direction vector, skipping $d$ number of vertices after every selection.

For these and similar queries, it was necessary to add a layer of API above Blender’s for just the vertex management. This was done by us in our implementation of the Blender add-on.

5. CONCLUSIONS

We present a sketch-based method for draping Indian garments. We map sketch stroke inputs to various parameters necessary to drape Indian garments. We then animate the draping procedure using a robust cloth simulator. We also show many example drapes that demonstrate the versatility of our proposed system.

We have also discussed in detail the limitations and challenges faced by us during this work in Section 4.7 and we would like work toward overcoming these in the future, and be able to generate even more complex drapes.

6. REFERENCES


