Cloth Simulation with Character Collisions

Patrick Tourniaire
École Polytechnique
patrick.tourniaire@polytechnique.edu

Abstract

Cloth simulation in computer graphics is a challenging task, especially when the cloth interacts with dynamic objects like characters. In this project, we present a simplified yet effective approach to managing cloth collisions with a moving character. Our method treats character joints as spherical objects and bone segments as cylindrical objects, allowing for efficient approximation of cloth-to-character collisions. While more accurate methods exist, our approach strikes a balance between computational complexity and realism.

1 Introduction

In this project we will be presenting a simple approach to managing cloth collisions which are attached to a moving character. We will present our approach which treats joints as spherical objects and bone segments as cylindrical objects. This simplistic view enables us to create an efficient approach to approximating the handling of cloth to character collisions. Indeed, more accurate approaches exist.

2 Notation

Throughout this report we will be referring to vectors in a 3D scene in bold (such as $\mathbf{x}$), and scalar values unbolded (such as r for radius).

3 Methodology

To be able to construct an effective approach to dealing with cloth collisions with a character whilst keeping the complexity at a reasonable level, we make a few simplifying assumptions. Dealing with collisions with primitive shapes such as spheres and cylinders is much more trivial than dealing with the endless edg-cases when using a high detail character mesh. Thus, for our character we use a skeleton and set joints to be spheres and have cylinders represent the connections between joints. Thus enabling us to take this simpler approach whilst still achieving the goal.

3.1 From Skeleton to a Character of Primitive Shapes

The character we will be dealing with is the Lola character from previous works. This character obviously have a much higher complexity than a skeleton built of primitive shapes. Thus, we optimise the radius of the spheres and cylinders such that they only slightly cover the mesh surface, an example of these primitive shapes on the skeleton can be observed in Figure 1. As we will observe in Section 4.

3.2 Collision with Primitive Shapes

In this section we will be presenting the different forms of constraints which our system deals with to avoid collisions between the cape and the character. As we aim to simulate a cape on the character, a natural question to ask is; how do we fix the cape to the character whilst it is moving?

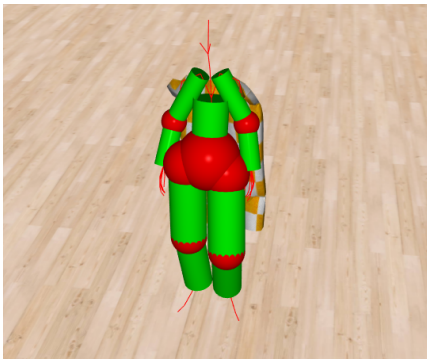

Figure 1: Showing how we construct a simple mesh for collision detection using primitive shapes such as spheres and cylinders exclusively Note how only a few essential bone segments have primitive collision shapes as most bone segments would be redundant to deal with collisions.

We can frame this as a constraint in our system, however, it is not a collision constraint in this case. It is simply enforced through 4 points on the cape which should always have the same position as some points/joints on the skeleton. We have decided to use 4 attach points due to the fact that 2 attach points can lead to the cape being elongated significantly whilst the character is moving. Thus, the first attach point is located on the left shoulder, the second on the midpoint between the left shoulder and the upper spine, the third is located on the midpoint between the upper spine and the right shoulder, and finally the fourth point is attached to the right shoulder. As we will observe in Section 4 this achieved much better results than only using shoulders as attach points.

3.2.1 Planes

Dealing with collisions with planes is trivial, however, it will serve as a good foundation for how we deal with sphere and cylindrical constraints. Assume we have some plane parametrised by a normal vector $\mathbf{n}_{\text{ground}}$. Then, we simply check that if the y-component of some cape particle is below the y-component of the normal. If that is the case then we perform the following update, where $\mathbf{p}_i$ represents some ith particle in the cape.

$$\mathbf{p}_i^{(y)} = \mathbf{n}_{\text{ground}}^{(y)} + \epsilon \tag{1}$$

Where ϵ represents some offset from the ground, which is set based on the physical thickness of the cape.

3.2.2 Spheres

To deal with sphere collisions, we take the following approach. First, every particle making up the mesh of the cloth will be compared with every single joint which has a spherical constraint. We note that not all joints have spherical constraints as this would be redundant for a multitude of smaller joints (such as finger joints). Let us denote some particle $\mathbf{p}$ to be a particle in the cape, and some sphere center by $\mathbf{c}$ with some radius r. We can detect a collision by evaluating the following distance.

$$||\mathbf{c} - \mathbf{p}|| \leq r + \epsilon \tag{2}$$

Here we introduce some ϵ such that we do not place the cape particle on the edge of the sphere. For our case, we set this to be a small enough value to reflect the thickness of the fabric, such that the cape does not slightly penetrate the surface of the joint sphere. The update step for the new position of the cape particle becomes.

$$\mathbf{p}' = (r + \epsilon)\mathbf{n} \tag{3}$$

Where $\mathbf{n}$ represents the unit normal, obtained by normalising $(\mathbf{p} - \mathbf{c})$.

3.2.3 Cylinders

The approach for dealing with cylindrical constraints is quite similar to the one presented for spheres. However, there is an additional layer of complexity added. We parametrise a cylinder by the following, it starts at some joint $\mathbf{p}_i$ and ends at some other joint $\mathbf{p}_j$, with some constant radius of r. Thus, we only want to define a collision when a cape particle is within this bone segment and is within the radius plus some additional ϵ. We determine this by projecting every cape particle on the bone segment spanning the cylinder and checking if the following two conditions hold. First we define the projection by the following.

$$\mathbf{p}_{\text{proj}} = \frac{\mathbf{p}(\mathbf{p}_i - \mathbf{p}_j)}{||\mathbf{p}_i - \mathbf{p}_j||} \tag{4}$$

Then evaluate the following conjunct to determine if we need to check that the particle is within the radius of the cylinder or not.

$$||p_{\text{proj}} - \mathbf{p}_i|| \leq ||\mathbf{p}_i - \mathbf{p}_j|| \quad \bigwedge \quad ||\mathbf{p}_{\text{proj}} - \mathbf{p}_j|| \leq ||\mathbf{p}_i - \mathbf{p}_j|| \tag{5}$$

If this conjunct returns true, then we will evaluate if the particle is within the bounds of the mesh surface or not. Which is achieved by evaluating the following inequality.

$$||\mathbf{p} - \mathbf{p}_{\text{proj}}|| \leq r + \epsilon \tag{6}$$

If this condition is true then we perform the following update step for the cape particle. Where $\mathbf{n}_{\text{proj}}$ represents the unit normal of $\mathbf{p} - \mathbf{p}_{\text{proj}}$.

$$\mathbf{p}' = (r + \epsilon)\mathbf{n}_{\text{proj}} \tag{7}$$

Through this approach we are able to use both spheres and cylinders to effectively create a rough framework for achieving, cloth-character collisions. However, we also apply external forces such as gravity and wind to the cape to acheive realistic results.

3.3 Physical Forces

The physical forces applied to the cape are wind, gravity and spring forces applied between cape particles. How we have applied this is similar to past works we have done and thus, we will not go into detail. However, we will present a brief overview for the sake of reproducibility.

3.3.1 Inter-Particle Spring Forces

3.3.2 Gravity

3.3.3 Wind

3.4 Complete Algorithm

3.5 Character Movement

The character movement is based on the previous works done, where we have a static animation for walking which is played in a loop. Then to achieve a seamless transition from animation start to animation end we perform a quaternion interpolation of each local joint frame of the skeleton from the end state to the start state. Thus, when the character stops moving a transition starts from the walking state to the idle state to also achieve this seamless transition between states.

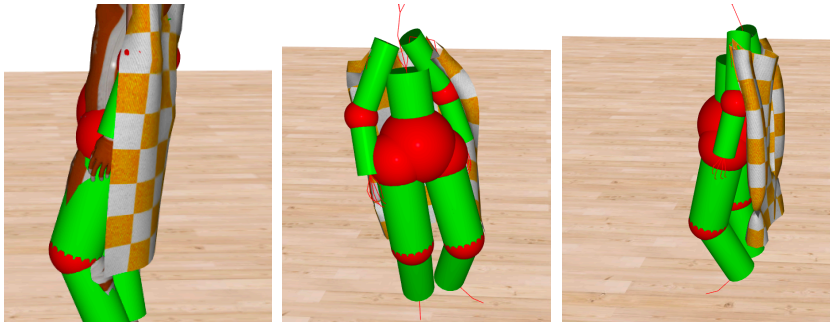

Figure 2: Qualitative results achieved for cloth collision with primitive shapes making up the character skeleton. Left image shows how these primitive shapes overlap slightly the complex mesh for the character.

4 Qualitative Results

Based on the results observed in Figure 2 we achieve decent approximations for the collisions with the complex mesh of the character. We note that the framerates do not drop after adding these constraints with the primitive shapes given that our approach is not computationally heavy. However, there are quite a few limitations with our approach.

If one looks more closely at smaller bone sgements such as fingers there are cases where they can poke through the cape, however, one would have to look very carefully to notice. We can go around this by simply adding constraints on the fingers using very small cylinders. However, that will not solve the issue by itself as the character can move quite fast wrt to the numerical approximation of the particle forces. Thus, in most cases the cape would be able to teleport through these shapes before our approach would be able to detect a collision and thus perform the iterative updates to the forces applied to the cape particles. One could in theory decrease the time delta for this approaximation but one would have to set an extremely low time delta which would result in unstable results. Another naive approach would be to slow down the movement of the character however this does not resolve the issue directly.

Therefore, this discussion highlights the downside of approximation methods for the computation of forces. To achieve better results one could apply more complex and exact techniques such as FastMassSpring presented by Tiantian Liu, et al. [**?**].

5 Conclusion

References