

RESOLUTION INDEPENDENT CURVED SEAMS IN  
CLOTHING ANIMATION USING A REGULAR PARTICLE GRID

A Thesis

by

JACOB W. FOSHEE

Submitted to the Office of Graduate Studies of  
Texas A&M University  
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2004

Major Subject: Computer Science

RESOLUTION INDEPENDENT CURVED SEAMS IN  
CLOTHING ANIMATION USING A REGULAR PARTICLE GRID

A Thesis

by

JACOB W. FOSHEE

Submitted to Texas A&M University  
in partial fulfillment of the requirements  
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

---

John Keyser  
(Chair of Committee)

---

Donald H. House  
(Member)

---

Jianer Chen  
(Member)

---

Valerie Taylor  
(Head of Department)

August 2004

Major Subject: Computer Science

## ABSTRACT

Resolution Independent Curved Seams in  
Clothing Animation Using a Regular Particle Grid. (August 2004)

Jacob W. Foshee, B.S., Texas A&M University

Chair of Advisory Committee: Dr. John Keyser

We present a method for representing seams in clothing animation, and its application in simulation level of detail. Specifically we consider cloth represented as a regular grid of particles connected by spring-dampers, and a seam specified by a closed set of parametric trim curves in the cloth domain.

Conventional cloth animation requires the tessellation of seams so that they are handled uniformly by the dynamics process. Our goal is a seam definition which does not constrain the attached clothing panels to be of the same resolution, or even constant resolution, while not being a hindrance to the dynamics process. We also apply our seams to cloth defined on a regular grid, as opposed to the irregular meshes commonly used with seams.

The determination of particles interior to the cloth panel can be done using well-known graphics operations such as scan-conversion. Due to the particle-based nature of the simulation, the dynamics approach combines easily with existing implicit and explicit methods.

Finally, because the seams are resolution independent, the particle density per clothing panel can be adjusted as desired. This gives rise to a simple application of the given seams approach illustrating how it may be used for simulation level of detail.

To my family

## ACKNOWLEDGMENTS

“It is a good thing to give thanks”

To, you, the reader for your time, patience, and criticism

To Dr. John Keyser for four years of guidance, instruction, patience, counsel and liberty in research (and some things that weren't research)

To Dr. Don House for the in-depth introduction to the wonderful world of physically-based modeling and for many helpful and encouraging conversations

To Dr. Valerie Taylor for pushing the department in a positive direction and including students like myself in that process

To those who have provided helpful comments and service for my thesis including Dr. Jianer Chen, Dr. Salih Yurttas, Dr. Bruce McCormick and Dr. Fred Parke

To my illustrious officemates for a constant source of sanity-saving diversion (Nithin, Hao, 423B, Lei, Bin, Sarah, Mirko)

To the graphics research group for collaboration and useful discussions (Alex, Eric, Erin, Hong, Jacob, Kevin, Koji, Michael, Nathan, Ni, Ryan, Shreyas, Swapnil, and Zeki)

To TAGD (Texas Aggie Game Developers) for working so hard to invest in yourselves, each other, and Texas A&M

To my friends through these years (Ashley, Christian, Cody & Brandy, Aaron, Cody, Jayson, Mario, Jeff, Ramy, Stephanie, Gracie, June, Becca, Camp Boenig, Camp Shomaker, COFLO, and Fl!p)

To my grandfather, John W. Foshee, for constant support and funding for much of this research

To my grandmother, Jean Peterson, for deep and true concern, support, and availability

To Memaw and Pepaw, for helpful advice and steady affirmation

To Mom and Dad for a lifetime of wise shepherding, careful support, and boundless sacrifice

To Phillip for inspirational hard work on your own endeavors and a contagious excitement about mine

To Bethany for an example of graceful compassion and perfection, and the independence every brother wants in a sister

To Devanshi for more than I can enumerate— for challenging me, for loving me, for encouraging me

Thank you.

## TABLE OF CONTENTS

CHAPTER		Page
I	INTRODUCTION . . . . .	1
II	BACKGROUND . . . . .	3
	A. Particle-Based Cloth Dynamics . . . . .	3
	B. Seams and Trimming . . . . .	5
	C. SLOD . . . . .	5
	D. Multiresolution Cloth . . . . .	7
III	THEORY . . . . .	9
	A. Motivating Factors . . . . .	9
	B. Collinearity Requirement . . . . .	10
	C. Method Overview . . . . .	11
	1. Panel Specification . . . . .	11
	a. Panel Input . . . . .	11
	b. Internal Panel Representation . . . . .	12
	2. Seam Designation . . . . .	13
	3. Particle Classification . . . . .	14
	4. Simulation . . . . .	16
	a. Seam Forces . . . . .	16
	b. Seam Forces Algorithm . . . . .	18
	c. Seam Constraints . . . . .	18
	d. Seam Constraint Algorithm . . . . .	19
	e. Integration . . . . .	20
	5. Changing SLOD . . . . .	20
	a. Simplification . . . . .	21
	b. Refinement . . . . .	22
IV	IMPLEMENTATION . . . . .	24
	A. Details . . . . .	24
	1. Panel Scan-Conversion with a Graphics API . . . . .	24
	2. Rendering . . . . .	25
	3. Examples . . . . .	25
	B. Analysis . . . . .	29

CHAPTER	Page
1. Results . . . . .	29
2. Problems and Limitations . . . . .	30
V CONCLUSIONS AND FUTURE WORK . . . . .	32
A. Implications . . . . .	32
B. Future Work . . . . .	32
REFERENCES . . . . .	34
APPENDIX A . . . . .	39
APPENDIX B . . . . .	41
VITA . . . . .	43

## LIST OF TABLES

TABLE	Page
I Performance statistics over multiple resolutions for the T-shirt example. The headings refer to the panels: front, back, left sleeve, right sleeve and pocket. The values below each panel refer to the resolution along one axis of the trimmed cloth quad. FPS (the average number of frames per second) is given for animation with and without seam calculations. Finally, the estimated percentage of processing time devoted to seams is given. . . . .	28

## LIST OF FIGURES

FIGURE	Page
1	Given two curves there are two ways to force their correspondence. . . . . 13
2	Example intersections of a trim curve with the Voronoi diagram. Border particle p1 will have influence over region t1 to t2, and p2 from t2 to t3. . . . . 15
3	In some cases when reducing particle resolution, corresponding particles of the finer resolution may lie outside the trim curve. . . . . 21
4	T-shirt constructed from trimmed panels of regular mass particle grids. 26
5	The rasterized flat panels of the T-shirt at the beginning of simu- lation. These panels are trimmed from cloth quadrilaterals of 33 by 33 particle resolution. . . . . 27
6	Flat garment panels after one iteration of simplification. The panel resolution is 17 by 17. . . . . 27
7	In preparation for our second example we split the front panel to easily produce a jacket. . . . . 29
8	In our second experiment we show that by adding non-trimming seam curves we can produce an overlapping seam in the front of the jacket. . . . . 30

## CHAPTER I

### INTRODUCTION

This work lies in the broad context of computer graphics. It falls under the category of physically-based modeling, the science of generating realistic images and animations based on physical principles. Specifically, the subcategory of our work is clothing simulation.

The motivation of our work is to contribute to the broad challenge of interactive clothing animation. That is, we want an approximation of clothing behavior that is visually convincing and executes fast enough to generate motion in real-time. This goal is relevant to simulation and entertainment applications. It also has potential for computer-aided rapid prototyping of garments.

The current state of the art in interactive clothing animation exhibits only simple cloth shapes and clothing patterns (such as rags, flags and wrap-around skirts). One of our goals is to allow for more realistic and complex articles of clothing.

Another goal of interactive clothing animation is multiple clothed characters in the simulation. For this to be feasible we must use processor time economically, spending more time on simulations relevant to the viewer, and less time on those that are not. This implies using a Simulation Level of Detail (SLOD) approach.

Our contribution to both challenges is a resolution independent method for trimming and seaming cloth panels that are simulated via a regular grid of mass particles connected by dashpots.

In the next chapter we will describe previous work and relevant background information. Following that we will give the rationale behind and detailed description

---

The journal model is *IEEE Transactions on Automatic Control*.

of our approach to seams and SLOD. In Chapter III we discuss our implementation and experiments with the proposed method. In the final chapter we draw conclusions about our work and enumerate directions for future research.

## CHAPTER II

### BACKGROUND

#### A. Particle-Based Cloth Dynamics

Breen, et al. pioneered the particle-based approach to modeling cloth in their paper on predicting the drape of woven cloth [1]. Until recently their approach remained the only one using physically measured properties of cloth rather than manually tweaked coefficients or spring constants common in modern methods [2]. From their measurements they derived energy expressions that, when minimized, yielded the resting position of cloth draped over some object.

While their approach was far from interactive, the insight to use mass particles and quantify the relationship between them was an important one. Much work has followed in the animation of cloth using particles.

Provot provided a simple way to structure a spring mesh of a regular grid of particles that would approximate the three forces associated with cloth: stretch, shear and bend [3]. While this approach does not have the same accuracy as an arbitrary triangular mesh, it is appealing because it is easy to implement (consisting only of standard spring-dampers and point masses), and the bend calculation is faster. It has also been adopted in real-time applications [4].

The structure necessary for generating the internal forces is as follows. For a given particle  $P$ , we connect  $P$  to its vertical and horizontal neighbors to account for stretch. We connect  $P$  to its four diagonal neighbors to account for shear. Finally, we attach  $P$  to the particles just beyond its four nearest horizontal and vertical neighbors to account for bend.

Because of this layout particles must be collinear along rows and columns when

in their rest position. This restricts the shape of cloth panels. We shall refer to this restriction as the collinearity requirement.

Still, only using springs in this fashion requires very high damping coefficients to simulate cloth’s stiffness. This imposes a correspondingly very low upper limit on the timestep necessary to integrate the equations with acceptable error.

Such a small timestep would be unacceptable for interactive rates, so Provot uses less stiff coefficients. In order to avoid the super-elasticity associated with the under-damped springs there is a post integration correction step where particle positions are adjusted to impose stretch and stretch rate limits.

The appropriately titled “Large Steps in Cloth Simulation” by Baraff, et al. also takes a particle-based approach, though a regular structure is not necessary [5]. The major contribution is a purely implicit (backwards) integration scheme that greatly increases the stability of the integration of the “stiff” spring equations, allowing us to take larger time steps.

Desbrun, et al. introduced an approximation to the implicit integration solution that maintained stability at the expense of accuracy [6]. The major benefit of their approach is that the most expensive calculations (a large matrix inversion) can be precalculated.

Vassilev, et al. made two contributions to the field of interactive particle-based dynamics [7]. The first is an additional method to control super-elasticity by modifying particle velocities (as opposed to Provot’s approach of modifying positions). The second is a fast collision detection algorithm that uses graphics hardware to render depth, normal, and velocity maps of a figure for collision detection and response.

## B. Seams and Trimming

In their paper on dressing synthetic actors, Carignan, et al. derive a spring force expression to pull cloth panels together and a momentum-based constraint to keep panels together at a seam [8]. They also assert the necessity of applying dynamics to cloth as it is seamed so that it deforms properly around the wearer. Panels are specified as polygons.

Okabe, et al. describe user specification of trimming and seams as curved lines [9]. When the cloth is simulated the panels are tessellated such that seam curves are divided with equal divisions along each arc length.

In fact, the common approach in cloth simulation is to require vertex correspondence where panels meet along a seam [10][11][12][13]. While this does ease calculation, it does not allow for differing resolutions across seams. It also imposes an undue burden if the user is required to specify the panels as polygons.

## C. SLOD

Approximation and culling of numerical dynamics has always been used by engineers of virtual interactive environments due to the natural limits of real-time computing. Only recently has a formal approach been applied to generating Levels of Detail in Simulation. We will look at some of those, but unfortunately, no general or automatic method for generating SLODs has been published. Therefore, we will look to existing techniques in SLOD literature for what may be applicable or analogous.

The term Simulation Levels of Detail (SLOD) is originated by Carlson and Hodgins [14]. The example simulation is one-legged creatures bouncing around trying to avoid a giant puck. This paper is important because they begin to lay down criteria for SLOD: “the outcome of the [simulation] is essentially unchanged and the viewer’s

perception of the motion must be unchanged.” They give three examples of SLODs: dynamic with fewer degrees of freedom, a hybrid kinematic/dynamic approach, and purely kinematic.

Chenney, et al. complement this work by providing further criteria for SLOD: consistency and completeness [15]. Of particular importance is their introduction of simulation that is culled-out altogether when relevance to the viewer has reached a significantly low level.

While the culling of simulation alone is not new (it has been done for years in interactive applications like games) what is new is their approach to reintroducing a culled simulation in a state that is probable and meets aforementioned SLOD criteria.

Chenney, et al. expand upon their work on SLOD by building a VRML and Java architecture and libraries for constructing dynamic systems [16]. They further describe division of dynamic systems into periodic and aperiodic groups using a neural network and a statistical approach to cull and uncull these simulations respectively.

O’Brien, et al. contribute to the field by designing a system to automatically construct and dynamically adjust levels of detail for a particle simulation [17]. In their approach a given frame rate and fidelity are maintained by subdividing space into a hierarchy of Regions of Interest (ROI), where ROI are determined by the view frustum and critical events (i.e. collisions). The simplification occurs when a cluster of particles are treated as one.

In 2001 Chenney, et al. introduced the idea of Proxy Simulations [18]. Chenney and Adzima reinforce the idea that some simulations can be broken into discrete events, and by predicting when these events may occur we can avoid processing the simulation in the between time [19][20]. This leads to an event queue approach for handling off-screen simulation just enough that we can anticipate when it may need to be reintroduced into the dynamic simulation while being consistent with its previous

state.

While not directly addressing the general challenges of SLOD, Cordier and Magnenat-Thalmann’s recent work is very relevant [12]. They achieve a goal of real-time clothing animation by using a combination of geometric and physically-based approaches and by avoiding general collision detection. The key observation driving their selection of clothing animation methods is the localization of clothing relative to the body. Clothing sections are characterized as form-fitting, loose, and free-floating, then are animated accordingly.

Other important simulation level of detail work relevant to deformable bodies includes Perbet, et al.’s animation of prairies, and Ward, et al.’s animation of hair [21][22][23].

#### D. Multiresolution Cloth

Hutchinson, et al. describe a method for refining regular particle grids by increasing the particle frequency in areas where the bend angle exceeds some threshold [24]. They do not specify a simplification method. The treatment of introducing mass-particles is a little cumbersome since maintaining spring behavior requires the mass of all particles to be the same regardless of the local particle resolution. This violates intuition that high frequency particles should have less mass than low frequency particles.

Zhang and Yuen also contribute to the multiresolution cloth field by providing a refinement scheme [25]. One contribution is an alternative regular mass-spring structure based on triangles. Over the course of a draping simulation they refine the mesh by splitting triangles with a 1-to-4 split. They do not offer criteria for when to split triangles nor do they offer a simplification strategy.

Villard and Borouchaki offer another approach to subdividing cloth defined on a regular grid [26]. They do offer a more appropriate method for calculating particles masses, and an intuitive way to handle T-intersections. Still, they also provide no means for simplification.

Work by Debunne, et al. yields an evolution of approaches to real-time interaction with deformable objects simulated using a finite element approach [27][28][29]. While their work is not directly with cloth, many ideas may be extended to the cloth domain. Relevant contributions are self-consistency between resolution changes, guaranteed real-time performance and resolution-independent rendering of an external surface to avoid popping.

Volkov and Ling present a method for both cloth refinement and simplification based on the  $\sqrt{3}$  surface subdivision [30][31]. The given refinement requires a triangular mesh so is inapplicable for dynamics using a regular particle grid.

A general approach to refinement of deformable bodies simulated with a finite element approach is given by Grinspun, et al. [32]. CHARMS, or conforming, hierarchical adaptive refinement methods, are used to replace basis functions rather than elements. This very general idea is applicable to many domains, including cloth, but is not currently suited for real-time animation.

Capell, et al. also employ a basis adaptation approach to refining finite element solutions for dynamics deformations, though their approach is less general [33].

Finally, Bridson, et al. used collision-aware surface subdivision for the purpose of rendering [34]. They use Loop subdivision and caution that popping may occur for a dynamics mesh that is too coarse [35].

## CHAPTER III

### THEORY

#### A. Motivating Factors

In general we can classify prevalent particle-based cloth dynamics methods by two choices: explicit versus implicit integration and a (regular) structured versus unstructured particle mesh. In choosing which dynamics method to use in a system supporting cloth SLOD, we consider the advantages and disadvantages of each.

Implicit methods allow much larger time steps than explicit methods due to the assurance that (negative) derivatives at the next time step point back to the previous state. However, solving for the new state of the simulation cannot be carried out “forward” as in the explicit case. An implicit method requires solving a set of  $n$  linear equations, where  $n$  is the number of particles in the cloth mesh.

The primary challenge of using an implicit method in a SLOD scheme, where the resolution of the cloth mesh may change, is that the large sparse matrix representing interparticle connections must be recomputed every time a particle is added or removed. This detracts from any precomputation we could do given a static cloth mesh structure (for example in Desbrun, et al.’s approach [6]).

The second choice we must make in selecting a dynamics approach is whether the mesh should be structured or not. A structured mesh represents all forces in the cloth as standard dashpots. As such, it mandates that the particles in the cloth, and the spring-dampers between particles be arranged in such a way that they can approximate standard intercloth forces (which are stretch, shear and bend). So, it is quick and easy to compute the forces and their derivatives for this scheme.

An unstructured mesh, on the other hand derives forces differently for shear,

bend, and stretch respectively. For example, in a triangular mesh shear can be measured as some property of the angle between legs of the triangle. Bend must be calculated as the angle between triangles, and stretch is derived from the magnitude of triangle edges.

Therefore a structured approach imposes more of a challenge on cloth SLOD because when adding or removing particles to the system we must ensure that an appropriate structure is maintained. However calculating internal forces on a regular mesh is faster than on an irregular mesh.

So, the fastest dynamics combination of implicit integration with a regular particle grid also imposes the most constraints when developing cloth levels of detail.

To meet interactivity constraints we use the implicit integration approximation proposed by Desbrun, et al. and the particle structure proposed by Provot [6][3]. In an effort to allow garment resolution modification during simulation (for SLOD) and to meet the aforementioned dynamics constraints, we propose the use of resolution independent seams. In so doing, preprocessing for garment panels could be carried out at multiple resolutions. Then at simulation time a given cloth panel could be refined or simplified by substitution of the appropriate garment panel. The seam between panels would serve as an interface between panels allowing the simulation to proceed unencumbered.

## B. Collinearity Requirement

An important factor to consider when creating arbitrarily shaped garments from a structured particle grid is the collinearity requirement.

We desire the rest position of the cloth panel to be planar. The spring structure described above enforces this requirement as long as each row and column of particles

lies on a straight line. Rows (columns) need not be parallel to each other, nor perpendicular to columns (rows), but this does lead to a more uniform behavior across the cloth patch (as particles are more evenly distributed).

By setting rest lengths based on collinear particles the rest position of a particle must lie on a line between its neighbors. By allowing the rest position of a particle to be out of alignment with its neighbors, there may be more than one rest position for that particle, which need not be in the plane. Therefore we cannot move particles out of alignment and maintain a planar rest position.

### C. Method Overview

Our method consists of the following five steps, which we will describe in detail:

1. Panel Specification
2. Seam Designation
3. Particle Classification
4. Simulation
5. Changing SLOD

#### 1. Panel Specification

##### a. Panel Input

The most intuitive way for the user to define a cloth panel is by giving the set of trimming curves used to “cut” the panel out. An examination of existing clothing patterns shows that curved boundaries are common.

To ask the user to input a polygon instead of curved boundaries requires the user to make some guess as to the appropriate tessellation resolution of the cloth

panel. This is an unnecessary requirement and creates a coupling between the panel specification and its resolution.

Therefore panels are specified by providing an ordered list of control points which form a set of closed 2D trim curves bounding the panel. We use parametric curves because they give us a way to map constraints between panels without vertex correspondence.

#### b. Internal Panel Representation

Deforming the particles of the cloth mesh to fit the panel boundaries is out of the question because of the collinearity requirement. That is, we cannot move particle rest positions out of row (column) alignment to snap them to trim curves. Effectively, what we must do is snap trim curves to particles on the grid. Therefore we use a structured cloth quadrilateral as the atomic cloth element, and trim our panel from it.

Conceivably we could build up an arbitrarily shaped cloth panel by piecing together “cloth quads.” This represents a possible, yet complex solution. Representing cloth this way would likely require a higher composite resolution than would be needed for a single large quad. Also, it would be difficult to represent bend forces between quads, particularly since they would be different sizes and shapes. There is also the added difficulty of using a general implicit integration scheme that includes forces between quads. Finally, seaming panels together would either require a quad-to-quad correspondence at seams (something we are trying to avoid), or complex data structures to enforce seam constraints.

So, it is appealing to use a parametric curve to trim a simple cloth quadrilateral comprised of a regular grid of mass particles for the following reasons:

- The trim curve analogy corresponds exactly to the process used in actual cloth panel construction
- There is no constraint on the shape of the panel definition
- The only modification to the dynamics process occurs at seams
- Determining which particles on a regular grid belong to the panel is an instance of the well-known graphics operation scan-conversion
- Curves can be defined once and used for any desired resolution

## 2. Seam Designation

A seam is a constraint placed on two curves defined in the cloth domain such that the two curves are identical in world space. Therefore to designate a seam one must provide the two curves.

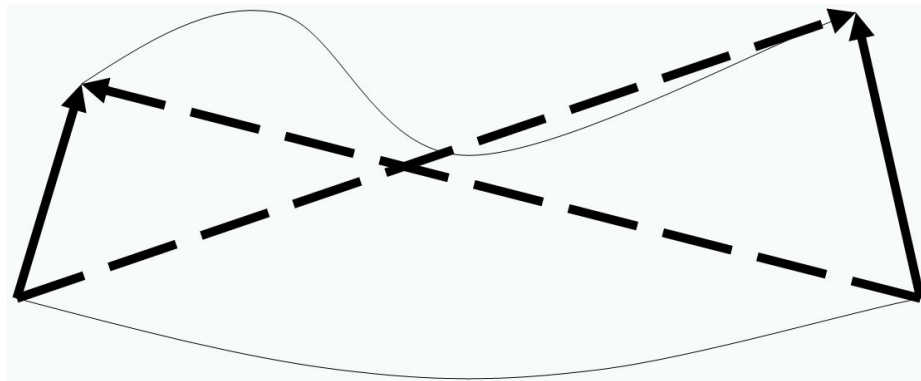


Fig. 1. Given two curves there are two ways to force their correspondence.

However that information is not enough. Given two curves there are two ways to force their correspondence (See Figure 1). Consider the question of which endpoints from opposite curves should be paired.

Because we are using parametric curves there is already an inherent direction associated with each curve. Therefore it is sufficient to supply a Boolean flag per seam for whether there is a direct or reverse parametric correspondence between seam curves.

### 3. Particle Classification

Once the trim curve is specified we must “transform” it into the cloth panel space and determine which particles will be used to represent the boundary, and how.

We characterize each active particle as inside the panel or on the trim curve. Particles that are “trimmed away,” that is lie outside and off of the trim curve, are referred to as outside or inactive. Those on a trim curve (on either the interior or exterior) are called border particles. Clearly particles may not lie identically on the trim curve, so we find the set of particles closest to the border.

We assume that each border particle influences the curve segment nearest to it. The question of what region of the curve is nearest a given particle is addressed directly by constructing a grid of lines equally spaced between particles (i.e. a Voronoi diagram, though the computation is much simpler for this case), and finding all intersections of the trim curve with those lines. The parametric curve will intersect only the cells corresponding to border particles. For every pair of intersections of the curve with one border particle’s Voronoi cell, we associate the corresponding parametric range,  $t_0 \dots t_1$ , with that border particle (See Figure 2). Note that mapping of the parametric range to particles can be performed regardless of the resolution of the grid, and could even be adapted to multiresolution grids.

To insure a parametric correlation between trim curves which are seamed together we use an arc length parameterization:  $s_0 \dots s_1$ . The arc length will be measured in panel space. In the case where we need a world space arc length, as we will when

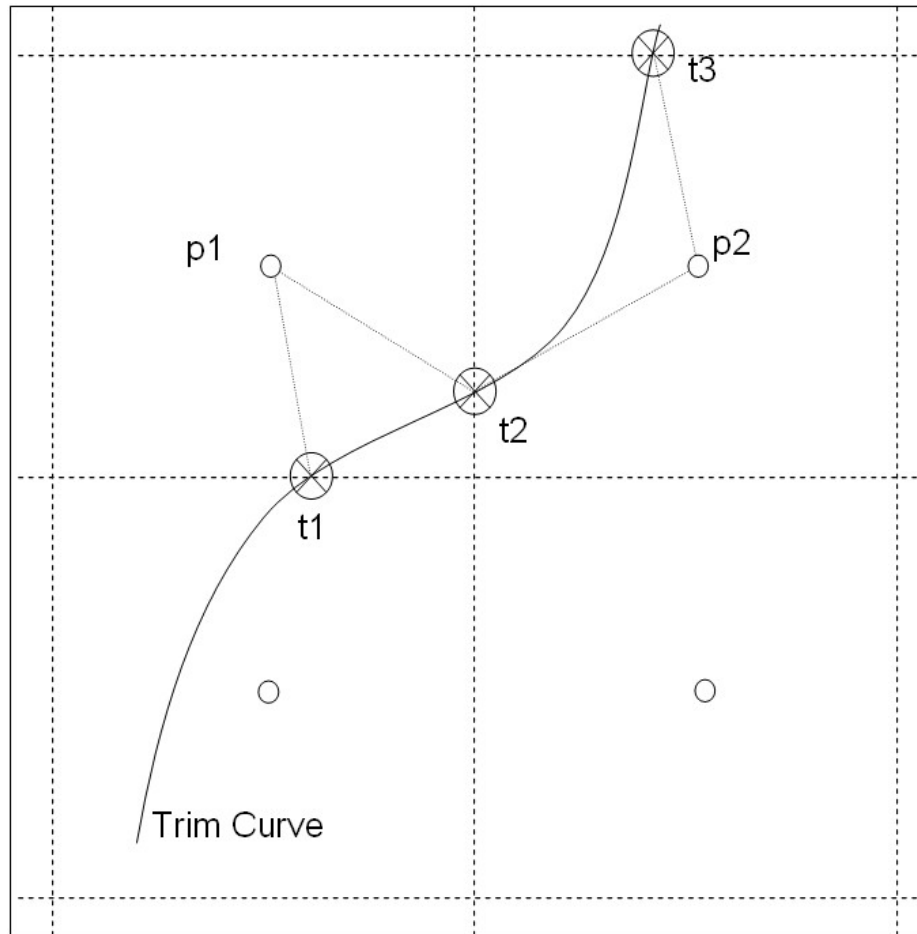


Fig. 2. Example intersections of a trim curve with the Voronoi diagram. Border particle p1 will have influence over region t1 to t2, and p2 from t2 to t3.

computing seam forces, we can simply scale up the panel space arc length by the same factor as the scaling of the panel.

To associate a single parameter value with each border particle we found it sufficient to simply average the parameters nearest a particle.

$$s_{mid} = \frac{s_0 + s_1}{2}$$

Finally, for each trim curve we store an ordered list of it's border particles sorted by parameter value.

## 4. Simulation

We structure the spring mesh according to the approach laid out by Provot [3]. For a given particle  $P$ , we connect  $P$  to its vertical and horizontal neighbors to account for stretch. We connect  $P$  to its four diagonal neighbors to account for shear. Finally, we attach  $P$  to the particles just beyond its four nearest horizontal and vertical neighbors to account for bend. In order to avoid the super-elasticity associated with springs we also have a post integration correction step where particle positions are adjusted to impose stretch and stretch rate limits. We add to the correction step the enforcement of seam constraints.

There are two possible ways of handling a seam: as an attractive force between curves and as an identity constraint between curves. We describe those below.

We also give listings for each algorithm in the appendices. In both cases we assume we have a seam which joins two curves,  $C_1$  and  $C_2$ , which may be from the same or different panels. We are also given an ordered list of border particles for each:  $BPL_1$ ,  $BPL_2$ .

### a. Seam Forces

In order to represent the attractive force between seamed panels, we imagine a spring force function acting over each curve to bring them together. (It cannot be considered a single force vector as trim curves may have arbitrary orientation with respect to each other.) That is, for any given parameter value  $t$ , and curves  $C_1$ ,  $C_2$

$$F(t) = K_s \|C_1(t) - C_2(t)\|$$

When the cloth is represented in world space, we no longer have a simple explicit representation of the location of the trim curve. Therefore, to create this seam force,

we instead apply forces at the border particles.

We now derive the force equation for seam particles. Because a particle must represent a curve region, we integrate over that region.

$$F_p = \int_{t=t_0}^{t_1} F(t)dt$$

Where  $F_p$  is the external seam force for border point  $p$ ,  $F(t)$  is as described above, and  $t_0$  and  $t_1$  are the range of curve parameter values corresponding to point  $p$ . However, in practice we choose to integrate over the world space arc length of the curve rather than the parameter value because force is proportional to mass, i.e. density.

$$F_p = \int_{s=s_0}^{s_1} F(s)ds$$

Because of the granularity of our solution we assume that force over the curve is constant for a given particle, so our force reduces to:

$$F_p = (s_1 - s_0)F(s_p)$$

Where  $F(s_p)$  is the spring force associated with particle  $p$ .

For the spring force calculation we must determine an appropriate rest length between particles on opposite trim curves. One may be tempted to store a signed distance to the curve with each particle. Then, for each pair of particles attracted by a seam spring force, one could approximate the rest length of the spring by summing the signed distances. Our early experiments using this approximation showed that it gave poor results and was too inaccurate.

Instead, we found that making the assumption that border particles lie identically on the trim curve gives more pleasing visuals. That is, we assume that the rest length between corresponding particles on opposite curves is zero, effectively snapping the

trim curve to the border particles.

$$F_p = (s_1 - s_0)K_s\|x_1 - x_2\|$$

Where  $p_1$  is a border particle on  $C_1$ ,  $p_2$  is the corresponding particle on  $C_2$ , and their respective positions are given by  $x$ .

#### b. Seam Forces Algorithm

Because we are not requiring a particle-to-particle correspondence we must iterate over each seam, tracking which particles from opposite curves represent overlapping regions.

Here we use the ordered list of border particles stored with each trim curve. We keep track of the minimum and maximum overlapping parameter values as we move down the curve. We then apply a spring force between particles scaled by the world space arc length of the overlap. When we reach the end of a particle's range of influence we increment to the next particle in the list.

#### c. Seam Constraints

It is insufficient to only represent the seam as a force between panels. In order to enforce that seams stay stitched together, it must be represented as a hard constraint. That is, we must constrain border particles from neighboring panels to lie on the same curve.

The key to enforcing seam constraints regardless of resolution is parametric correspondence. That is, we associate a parameter value with each border particle so that we may map it to its position on the opposite trim curve. In practice we may use a panel-space arc length parameterization in case of curves with different parameterizations.

#### d. Seam Constraint Algorithm

The idea behind the seam constraint algorithm is to move both trim curves of a seam half the distance towards each other. Doing this on a per particle basis, we must identify the destination position of each particle, and displace that particle halfway to that point.

For a given particle, we do this by identifying the particle on the opposite curve with the same parameter value, or the pair of particles surrounding that parameter value. In the latter case we simply take a linear combination of the opposite particle pair to find the desired location. We can safely assume that each particle is connected by a line segment because in the simulation neighboring particles are connected by a straight spring. This is a safe assumption because all nearest-neighbors of a particle are connected by straight springs, and neighboring particles on a trim curve must be nearest neighbors.

While the current parameter value is less than one we loop through each list of border particles. For the pair of current particles from each curve we compare parameter ( $s_{mid}$ ) values for three cases.

If the parameter values are equal, then there is an exact particle correspondence and we simply displace each particle half the distance to the other.

$$dx = \frac{(x_2 - x_1)}{2}$$

$$x_1 = x_1 + Cdx$$

$$x_2 = x_2 - Cdx$$

We include a coefficient,  $C$ , in order to control the convergence of the seam.

If  $s_1$  is greater than  $s_2$  then we must find the position of  $x_1$  by linearly interpolating between the positions of the corresponding particles on the opposite curve ( $s_2$

and  $next\_s_2$ ). The case where  $s_2$  is greater than  $s_1$  is analogous.

$$\tau = \frac{s_1 - s_2}{next\_s_2 - s_2}$$

$$x_1 = x_1 + C \frac{(1 - \tau)x_2 + \tau \cdot next\_x_2 - x_1}{2}$$

e. Integration

For our experiments we chose to use the implicit approximation described by Desbrun et al. This method was appealing because of its increased stability. We see no reason why our approach would not also work with explicit integration, given that a shorter time step may be necessary.

If the filtering matrix is calculated for a specific cloth panel, then no changes are necessary to the integration process. However, in our experiments we precalculated the force filtering matrix for the general (untrimmed) cloth quad because we found it convenient to store precalculations corresponding to different resolutions independent of trimming. Then, should the application call for a different resolution or cloth panel, the matrix would be ready at hand. The only necessary adjustment to the integration algorithm was to ignore forces due to inactive particles (those outside the trim curve).

## 5. Changing SLOD

To ease changing of resolutions, we use resolutions with  $2^n + 1$  rows and columns for some integer  $n$ . This yields a direct mapping from alternating particles of higher resolution to the next lower resolution.

As a preprocessing step we generate an array of multiple successive resolutions so that we can swap between them when appropriate. When switching resolutions we derive particle properties directly from the active resolution. In our experiment

we selected three particle attributes to preserve when changing resolution: position to minimize spatial popping, velocity to maintain behavior, and surface normal to maintain shading.

a. Simplification

For particles interior to the panel, simplification is a straightforward look-up into the active panel to retrieve particle states. We call this a direct mapping. However, due to the presence of trim curves, near the border the case is not always so easy.

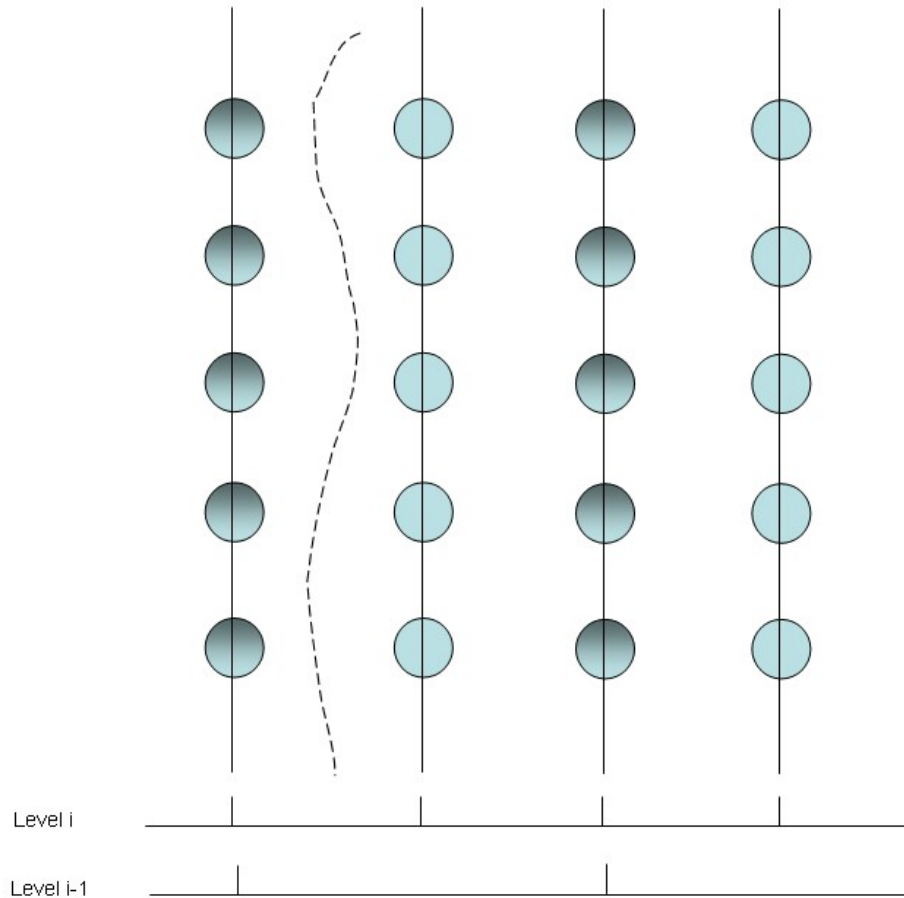


Fig. 3. In some cases when reducing particle resolution, corresponding particles of the finer resolution may lie outside the trim curve.

Consider a case that at resolution  $i$  column 0 falls outside of the trim curve, however at resolution  $i - 1$ , due to it being more coarse, column 0 lies on the trim curve (See Figure 3). Simplifying from  $i$  to  $i - 1$  causes a problem because column 0 of level  $i - 1$  cannot derive its position from the non-existent column 0 particles of level  $i$ .

We use the following heuristic in this case: when a directly mapped particle is marked to an inactive particle outside the panel, find its nearest neighbor that is inside the panel. We call this the nearest active neighbor, and we assign the coarser particle positions directly from it.

This is not unreasonable since the nearest active neighbor will be an immediate neighbor and therefore not already be mapped to a particle at the coarser level. Level  $i$  is twice the resolution of level  $i - 1$ , so at a finer resolution the curve must pass through at least one of the Voronoi cells surrounding the inactive particle in question. In this way we maintain the overall shape of the panel, and avoid bad mappings.

#### b. Refinement

Refinement introduces additional difficulties. Directly mapped interior particles are not a problem. However we must determine state values for the particles introduced by refinement.

For every introduced particle that has two or more opposite neighbors which are directly mapped to the coarser level we average the properties of those neighbors.

For the border case referred to above, where particles on the panel border do not have a simple correspondence to the coarse case, we introduce a mapping variable for each particle. During simplification when a direct mapping is made (either in the simple case or nearest active neighbor case) we assign the mapping variable of the finer resolution particle to the index of the corresponding coarse particle. This serves

to cement the mapping between the particles.

Afterwards when we perform refinement we do not need to perform complex tests or heuristics. We simply derive particle positions where mappings exist first. Then we interpolate where values are not known.

## CHAPTER IV

### IMPLEMENTATION

#### A. Details

Here we describe the details of our implementation of the described approach to multiresolution independent seams. Pseudo-code listings for the algorithms described in the previous chapter are available in the appendices.

##### 1. Panel Scan-Conversion with a Graphics API

The most challenging aspect of implementing this method can almost entirely be alleviated by using a graphics API. Finding line intersections with Bezier curves is a non-trivial task, so we describe an approach to use polygon and curve scan-conversion.

The first step is to set the video buffer resolution equal to our cloth resolution. Then render a filled polygon representing the cloth panel. Polygon vertices are merely samples along the trim curves taken frequently enough that polygon edges are no larger than a pixel. Then we read back the frame buffer and mark the particles corresponding to the colored pixels as being inside the panel.

Next we scan convert each trim curve independently, drawing the curve with a solid color in one channel (e.g. red), and using another channel (e.g. green) to store the parameter value. When reading back the border particles we store and sort on the parameter value.

In order to get the parameter range for a particle you may simply average the values of neighboring pixels to estimate parameters at pixel boundaries. It is difficult to offer a precise error for this estimate as the curve may take any path within the bounds of the pixels. If you wish to bound the error we suggest that you render the

curve at a finer resolution, or use a classic iterative solution to find intersections with the Voronoi diagram.

It is worth noting that some graphics hardware may have trouble interpolating colors for line segments shorter than a pixel, in which case you may use sufficient number of point primitives.

Any particle that lies on a seam curve (whether it be a trim curve or not) is put into an ordered list of border particles.

## 2. Rendering

We have not devoted much time to our rendering approach so it is straightforward. We simply iterate through the particles of the cloth mesh drawing quadrilaterals with neighboring particles. For interactive cloth the mesh must be fairly coarse resulting in a jagged outline for panels. This is an area for future work and is addressed in the Conclusion.

If necessary, it is easy to fill in gaps left by seams. We build a vertex list during the force calculations. At that time we consider particles from opposite trim curves a pair at a time, and we push that pair onto a data structure. To render we iterate through the data structure drawing quadrilaterals (specifically, a quad strip). The force calculation algorithm intrinsically selects particles that are nearby yielding an evenly distributed set of polygons. Even though some particles are considered twice this does not cause a problem because the degenerate quadrilaterals reduce to triangles.

## 3. Examples

We have implemented the proposed model in C++ using OpenGL for panel rasterization and garment rendering. Panels are specified using cubic Bezier trim curves.

Seams are specified as references to a pair of curves from one or two panels.



Fig. 4. T-shirt constructed from trimmed panels of regular mass particle grids.

In our first example we constructed a T-shirt as seen in Figure 4. The shirt consists of five panels: front, back, pocket, and two sleeves. The pocket is the most simple needing only two trim curves, one of which is seamed to a non-trimming seam curve on the front panel. The front and back panels each have eight trim curves. Each sleeve is comprised of five trim curves. Panels for the shirt are seen in Figures 5 and 6 at two different resolutions.

There are 11 total seams in this example. The front and back panels are seamed on the sides and at the shoulders. The sleeves are seamed to themselves and to the front and back arm holes. Finally, the pocket is seamed to the front of the shirt.

Panel shapes were determined by scanning paper garment patterns, then manually approximating their outlines with Bezier curves. The control points were then used to generate the panels in our software.

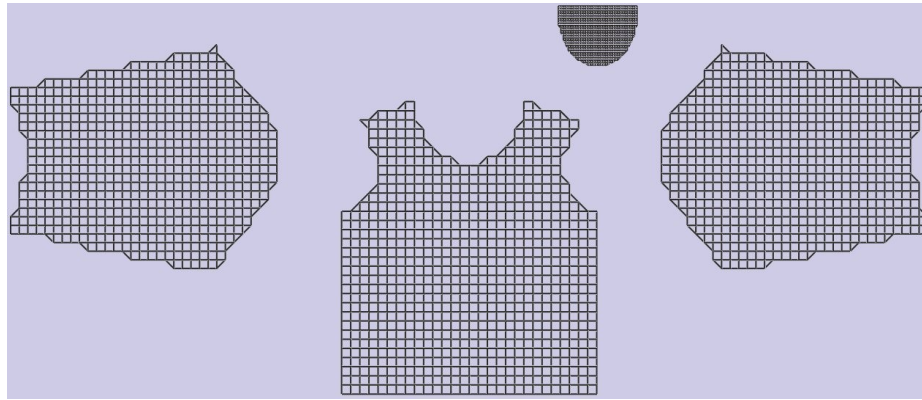


Fig. 5. The rasterized flat panels of the T-shirt at the beginning of simulation. These panels are trimmed from cloth quadrilaterals of 33 by 33 particle resolution.

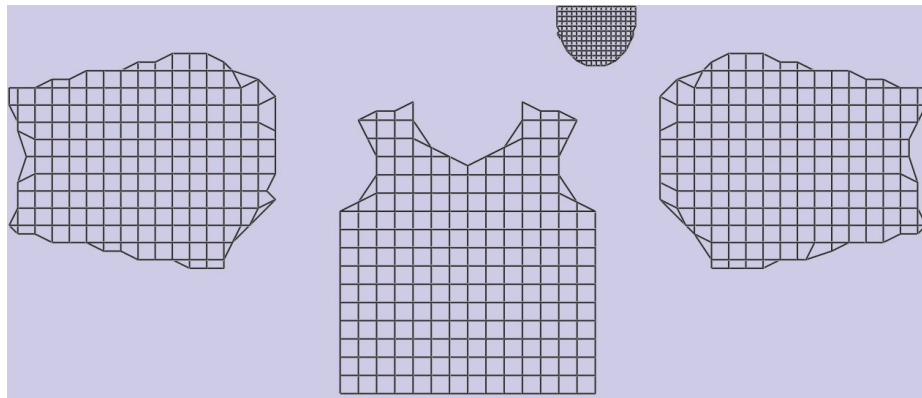


Fig. 6. Flat garment panels after one iteration of simplification. The panel resolution is 17 by 17.

We used a maximum resolution of 33 by 33 for the particle grid for each panel. Lower resolutions are automatically generated at dimensions of 17, 9, and 5. We observed resolutions could be independently changed for any panel with little effect on the panels seamed to it.

Because we have not yet implemented automatic SLOD selection criteria we changed panel resolutions manually. However, it was plain to see that reducing the resolution of back-facing, or distant panels produced little change in the perceived

Table I. Performance statistics over multiple resolutions for the T-shirt example. The headings refer to the panels: front, back, left sleeve, right sleeve and pocket. The values below each panel refer to the resolution along one axis of the trimmed cloth quad. FPS (the average number of frames per second) is given for animation with and without seam calculations. Finally, the estimated percentage of processing time devoted to seams is given.

Front	Back	Left	Right	Pocket	FPS	No Seams	Seam %
17	17	17	17	17	5.1	6.3	19%
17	9	17	17	17	6.1	7.4	18%
17	9	17	17	9	7.4	8.8	16%
17	9	9	9	9	12.6	14.8	15%
17	9	9	9	5	13.6	16.4	17%
9	9	9	9	5	30.5	33.3	8%

behavior.

Performance measurements were taken for a series of resolution simplifications such as those that might be seen in a SLOD application and are found in Table I. These measurements were taken on a modest AMD Athlon 850. In the experiment, 20 iterations of post-integration constraint enforcement were used per time step.

In our second experiment we modified the T-shirt panels by splitting the front panel to form a jacket shape (Figure 7). We then seamed together the front panels with two seam curves to show that our approach also supports overlapping seams.

In order to achieve the overlap two additional non-trimming seam curves are needed. The right edge of the left panel is seamed to a non-trimming curve on the right panel, and vice versa (Figure 8).



Fig. 7. In preparation for our second example we split the front panel to easily produce a jacket.

As expected, the cloth was more resistant to bending in this region than on a non-overlapping seam. Unfortunately, the amount of overlap is limited to the resolution of the cloth mesh. Still, when resolution is too coarse to represent the overlapped region, the seaming algorithms handle the simplification gracefully as the overlapped seams become a doubly enforced non-overlapping seam.

We derived the spring and damping coefficients used for cloth dynamics from estimates by Bhat, et al. [2].

## B. Analysis

### 1. Results

We have shown that we are able to use different resolutions per panel. We have also shown that our method supports other useful operations such as joining different materials, overlapping, topstitching and appliqué.

While we did not do a formal performance comparison, there was no noticeable latency after we added support for seams to our cloth engine. Cloth at low resolutions could still be animated at interactive rates on modest hardware. This makes sense considering seaming is linear in the order of the number of border particles, while the

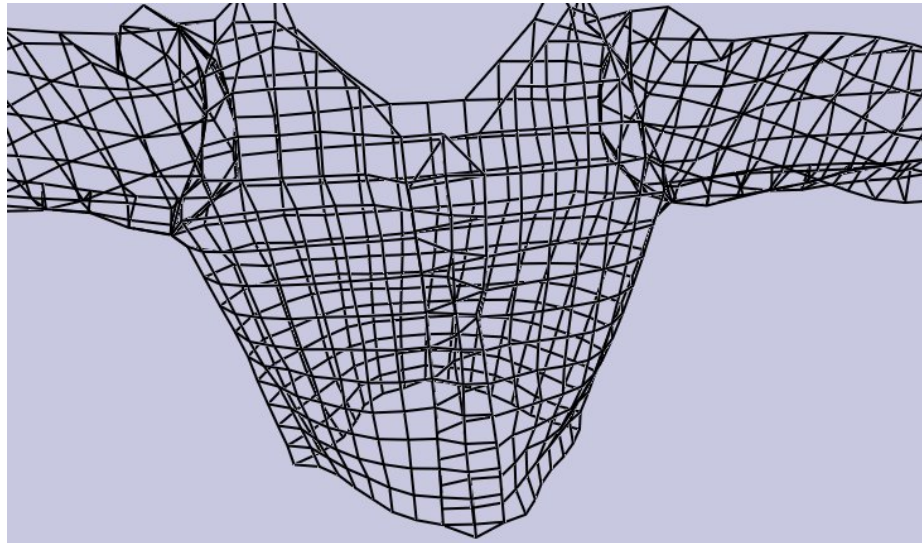


Fig. 8. In our second experiment we show that by adding non-trimming seam curves we can produce an overlapping seam in the front of the jacket.

dynamics process is quadratic over the total number of particles per panel, and thus dominates the running time.

## 2. Problems and Limitations

A noticeable issue when rendering clothing at low resolutions is aliasing. Trim curves which are not axis aligned may appear like angled stair steps in the resulting panel. This problem carries over directly from the field of rasterization, and is an intrinsic problem of using a structured mesh.

A possible resolution would be to use a different mesh for rendering. The vertices of that mesh would not be constrained to axis alignment, yet they could derive their position through a weighting of the underlying simulation particles.

Another issue associated with rasterization arises when using a graphics API for scan-conversion. Occasionally, this would lead to asymmetries in an otherwise symmetric panel (due to the bias in graphics hardware necessary for polygon fill-

ing). These asymmetries were usually only noticeable when the panel was flat, though sometimes they led to a slightly asymmetric garment.

A major limitation of the experiment described here is that there is no automatic transitioning between SLODs. That is, there is no automatic selection or blending between levels. Therefore some popping is visible. The main reason for visible popping is the rendering of the particle mesh. Necessarily, when reducing resolution the borders of panels change, some curvature is lost, and surface normals are altered slightly.

To compensate one could render a higher resolution mesh, as described in the next chapter. Also, SLOD blending could be employed to smooth the change in behavior between resolutions. Both options would represent a performance cost, and that would have to be weighed against the cost of visual popping. Still, we see no reason transitioning could not be added to the described SLOD operations.

Finally, the large number of constraint iterations necessary to represent the non-linear properties of cloth is a major bottleneck in performance. Often moving particles to meet spring constraints will invalidate other spring constraints. There is a need for a more systematic approach to constraining spring lengths and velocities.

## CHAPTER V

## CONCLUSIONS AND FUTURE WORK

## A. Implications

We have presented an approach for seaming together cloth panels of arbitrary shape when using a regular grid of mass particles. When compared with other approaches of garment construction we have successfully alleviated the need for vertex to vertex correspondence along seams. Also, we do not require the panel to be specified as a polygon. Our approach uses an arbitrary trim curve lending itself to any desired resolution.

Besides resolution, panels seamed together need not have the same material properties. Even panels of different densities are handled appropriately.

While working with our software we realized that by not requiring seam curves to be on panel boundaries the approach easily extended itself to topstitching and appliqué. The pocket on the shirt is an example of topstitching.

Furthermore, because the described seams operate only on ordered lists of particles (and their associated parametric properties), there is no reason this multiresolution seaming approach could not also be used on an irregular cloth mesh. In fact, it would even be possible to join an irregular mesh to a regular mesh, each with their own independent mechanics. Though typically slower, irregular meshes yield more accurate results and can better hide aliasing problems.

## B. Future Work

The resolution independent seams and resolution changing operations presented here serve as tools in cloth SLOD, but they do not represent a complete SLOD solution.

Research still needs to be done to determine error metrics and criteria useful for automatic changing of panel resolution. Also, the topic of transitioning between resolutions merits further study so that visible “popping” may be quantifiably reduced or eliminated.

A related area is automatic simulation time step determination and adjustment. A constraint in using the described implicit integration approximation is that the time step must be known during preprocessing. Therefore, research in automatically selecting an appropriate time step for given framerate requirements and material properties is warranted.

Clearly rendering the low resolution spring mesh used for interactive clothing animation is not the optimal choice. Modern programmable graphics hardware can handle many more polygons and can even facilitate vertex weighting operations. Therefore, it may be possible to render a high resolution polygon mesh representing clothing whose dynamics are generated by a low resolution mesh. In this case vertex programs running on the graphics card could be used to subdivide the low resolution mesh, or otherwise manipulate an existing high-polygon mesh.

A bottleneck referred to in the previous chapter was the high number of iterations necessary to limit the “super-elasticity” of spring models of cloth. A final area of future work is research in finding an optimal ordering of springs such that their lengths and stretch rates may be limited without invalidating previous relaxations.

## REFERENCES

- [1] D. E. Breen, D. H. House, and M. J. Wozny, “Predicting the drape of woven cloth using interacting particles,” in *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, Orlando, Florida, 1994, pp. 365–372, ACM Press.
- [2] K. S. Bhat, C. D. Twigg, J. K. Hodgins, P. K. Khosla, Z. Popović, and S. M. Seitz, “Estimating cloth simulation parameters from video,” in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, San Diego, California, 2003, pp. 37–51, Eurographics Association.
- [3] X. Provot, “Deformation constraints in a mass-spring model to describe rigid cloth behavior,” in *Proceedings of Graphics Interface (GI 1995)*. 1995, pp. 147–154, Canadian Computer-Human Communications Society.
- [4] J. Lander, “Devil in the blue faceted dress: Real-time cloth animation,” *Game Developer Magazine*, pp. 17–21, May 1999.
- [5] D. Baraff and A. Witkin, “Large steps in cloth simulation,” in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, Orlando, Florida, 1998, pp. 43–54, ACM Press.
- [6] M. Desbrun, P. Schröder, and A. Barr, “Interactive animation of structured deformable objects,” in *Proceedings of Graphics Interface (GI 1999)*. 1999, pp. 1–8, Canadian Computer-Human Communications Society.
- [7] T. Vassilev, B. Spanlang, and Y. Chrysanthou, “Fast cloth animation on walking avatars,” *Computer Graphics Forum (Eurographics 2001)*, vol. 20, no. 3, 2001.

- [8] Michel Carignan, Ying Yang, Nadia Magnenat-Thalmann, and Daniel Thalmann, “Dressing animated synthetic actors with complex deformable clothes,” in *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*, Chicago, Illinois, 1992, pp. 99–104, ACM Press.
- [9] Hidehiko Okabe, Haruki Imaoka, Takako Tomiha, and Haruo Niwaya, “Three dimensional apparel cad system,” in *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*, Chicago, Illinois, 1992, pp. 105–110, ACM Press.
- [10] Pascal Volino, Martin Courchesne, and Nadia Magnenat-Thalmann, “Versatile and efficient techniques for simulating cloth and other deformable objects,” in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, Los Angeles, California, 1995, pp. 137–144, ACM Press.
- [11] T. Vassilev, “Dressing virtual people,” in *Proceedings of Systemics, Cybernetics and Informatics 2000*, Orlando, Florida, July 2000, pp. 23–26.
- [12] F. Cordier and N. Magnenat-Thalmann, “Real-time animation of dressed virtual humans,” *Computer Graphics Forum (Eurographics 2002)*, vol. 21, no. 3, pp. 327–336, Sep 2002.
- [13] U. Cugini and C. Rizzi, “3d design and simulation of men garments,” in *Proceedings of Workshop WSCG 2002*, Praga, Feb 2002, pp. 4–8.
- [14] Deborah A. Carlson and Jessica K. Hodgins, “Simulation levels of detail for real-time animation,” in *Proceedings of the conference on Graphics interface '97*, Kelowna, British Columbia, Canada, 1997, pp. 1–8, Canadian Information Processing Society.

- [15] Stephen Chenney and David Forsyth, “View-dependent culling of dynamic systems in virtual environments,” in *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, Providence, Rhode Island, 1997, pp. 55–58, ACM Press.
- [16] Stephen Chenney, Jeffrey Ichnowski, and David Forsyth, “Dynamics modeling and culling,” *IEEE Comput. Graph. Appl.*, vol. 19, no. 2, pp. 79–87, 1999.
- [17] David O’Brien, Susan Fisher, and Ming C. Lin, “Automatic simplification of particle system dynamics,” in *Proceedings of IEEE International Conference on Computer Animation*, 2001, pp. 210–219.
- [18] S. Chenney, O. Arikan, and D. Forsyth, “Proxy simulations for efficient dynamics,” in *Proceedings of Eurographics 2001, Short Presentations*. Eurographics, 2001.
- [19] S. Chenney, “Simulation level-of-detail,” in *Game Developers Conference 2001*. March 2001, CMP Media.
- [20] J. Adzima, “Using AI to bring open city racing to life,” *Game Developer Magazine*, pp. 17–21, Dec 2000.
- [21] Frank Perbet and Marie-Paule Cani, “Animating prairies in real-time,” in *ACM Interactive 3D Graphics*, Mar 2001, pp. 103–110.
- [22] K. Ward and M. Lin, “Adaptive grouping and subdivision for simulating hair dynamics,” in *Proceedings of 11th Pacific Conference on Computer Graphics and Applications*, Canmore, Alberta, Canada, 2003.
- [23] K. Ward, M. Lin, J. Lee, S. Fisher, and D. Macri, “Modeling hair using level-of-detail representations,” in *Proceedings of Computer Animation and Social Agents (CASA 2003)*, New-Brunswick, New Jersey, USA, May 2003, p. 41.

- [24] Dave Hutchinson, Martin Preston, and Terry Hewitt, “Adaptive refinement for mass/spring simulations,” in *Proceedings of the Eurographics Workshop on Computer Animation and Simulation '96*, Poitiers, France, 1996, pp. 31–45, Springer-Verlag.
- [25] D. Zhang and M. Yuen, “Cloth simulation using multilevel meshes,” *Computers & Graphics*, vol. 25, no. 3, pp. 383–389, June 2001.
- [26] Julien Villard and Houman Borouchaki, “Adaptive meshing for cloth animation,” in *11th International Meshing Roundtable*, Ithaca, New York, 15–18 September 2002, Sandia National Laboratories, pp. 243–252.
- [27] Gilles Debunne, Mathieu Desbrun, Alan H. Barr, and Marie-Paule Cani, “Interactive multiresolution animation of deformable models,” in *Eurographics Workshop on Computer Animation and Simulation*, Sep 1999, pp. 133–144.
- [28] Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H. Barr, “Adaptive simulation of soft bodies in real-time,” in *Proceedings of the 2000 Conference on Computer Animation*, Philadelphia, May 2000, p. 15, IEEE Computer Society.
- [29] Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H. Barr, “Dynamic real-time deformations using space and time adaptive sampling,” in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, Los Angeles, California, Aug 2001, Annual Conference Series, pp. 31–36, ACM Press.
- [30] V. Volkov and L. Ling, “Adaptive local refinement and simplification of cloth meshes,” in *Proceedings of International Conference on Information Technology and Applications 2002*, Bathurst, Australia, November 2002.

- [31] Leif Kobbelt, “ $\sqrt{3}$ -subdivision,” in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, New Orleans, Louisiana, 2000, pp. 103–112, ACM Press/Addison-Wesley Publishing Co.
- [32] Eitan Grinspun, Petr Krysl, and Peter Schröder, “Charms: a simple framework for adaptive simulation,” in *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, San Antonio, Texas, 2002, pp. 281–290, ACM Press.
- [33] Steve Capell, Seth Green, Brian Curless, Tom Duchamp, and Zoran Popović, “A multiresolution framework for dynamic deformations,” in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, San Antonio, Texas, 2002, pp. 41–47, ACM Press.
- [34] Robert Bridson, Ronald Fedkiw, and John Anderson, “Robust treatment of collisions, contact and friction for cloth animation,” in *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, San Antonio, Texas, 2002, pp. 594–603, ACM Press.
- [35] C. Loop, “Triangle mesh subdivision with bounded curvature and the convex hull property,” Tech. Rep. MSR-TR-2001-24, Microsoft Research, Redmond, Washington, 2001.

## APPENDIX A

## SEAM FORCES ALGORITHM

```

applyForces()
  i, j, s = 0
  while (i<BPL1.size AND j<BPL2.size)
    // Retrieve the arc length ranges
    // for both particles
    sa1 = BPL1[i].s0
    sb1 = BPL1[i].s1
    sa2 = BPL2[j].s0
    sb2 = BPL2[j].s2
    // Find the range of overlap (sa...sb)
    // between the 2 particles
    sa = max( s, min(sa1, sa2))
    sb = min( sb1, sb2 )
    // Let v be the unit vector pointing
    // from Pj to Pi
    v = (BPL1[i].x - BPL2[j].x) /
        | BPL1[i].x - BPL2[j].x |
    // Approximate the spring force
    Fs = (sb - sa) * Ks *
        | BPL1[i].x - BPL2[j].x | * v
    // Accumulate the force for each particle
    F1 += Fs

```

```
F2 += Fs
// Let s be the arc length along the seam that
// we have already covered
s = sb
// If we have moved past the range of either
// of the particles move to the next one
if (s >= sb1) then i+=1
if (s >= sb2) then j+=1
```

## APPENDIX B

## SEAM CONSTRAINTS ALGORITHM

```
void applyConstraints ()
    t=0.0
    while (t < 1.0)
        // Retrieve parameters at particles
        t1 = BPL1[i].tmid
        t2 = BPL2[j].tmid
        // and next particles
        next_t1 = BPL1[i+1].tmid
        next_t2 = BPL2[j+1].tmid
        // Retrieve particle positions
        x1 = BPL1[i].x
        x2 = BPL2[j].x
        // If at same parameter
        // pull them together
        if (t1 == t2) then
            dx = (x2-x1) / 2
            x1 += C * dx
            x2 -= C * dx
        // If different parameters
        else if (t1 > t2) then
            // Get position of next particle
            // on opposite side
```

```

next_x2 = BPL2[j+1].x
// find linear combination
s = (t1 - t2) / (next_t2 - t2)
// Displace particle by halfway * C
x1 += C * ((1-s)*x2 +
           s*next_x2 - x1) / 2
else /* t2 > t1 */
  next_x1 = BPL1[i+1].x
  s = (t2 - t1) / (next_t1 - t1)
  x2 += C * ((1-s)*x1 +
            s*next_x1 - x2) / 2
// Move to the next pair of particles
if (next_t1 < next_t2) then
  i+=1
else if (next_t2 < next_t1) then
  j+=1
else /* next_t1 == next_t2 */
  i+=1
  j+=1
t=max(t1,t2)

```

## VITA

Jacob W. Foshee

21414 Hannover Pines Drive

Spring, Texas 77388

**Education**

Texas A&M University, College Station, Texas, 1996-2004

B.S. in Computer Science, May 2001

Overall GPA (4.0 scale): 3.589, Major GPA: 3.6

M.S. in Computer Science, August 2004

Overall GPA (4.0 scale): 3.558

**Honors and Awards**

Distinguished Student Award (Spring 97, Fall 99)

Nokia Scholarship Recipient

Graduate Student Leadership Award

**References**

Available upon request

The typist for this thesis was Jacob W. Foshee.