creative commons
COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

Master's Thesis of Electrical and Computer Engineering

Pattern Mirroring Algorithm for Fast Simulation about Symmetric Cloth

대칭적인 의상의 시뮬레이션 가속을 위한 패턴 미러링 알고리즘

February 2019

Graduate School of Engineering
Seoul National University
Electrical and Computer Engineering Major

Inseo Jang

Pattern Mirroring Algorithm for Fast Simulation about Symmetric Cloth

Professor. Hyeong-Seok Ko

Submitting a master's thesis of Public Administration

February 2019

Graduate School of Engineering
Seoul National University
Electrical and Computer Engineering Major

Inseo Jang

Confirming the master's thesis written by

Inseo Jang

December 2018

Chair	Jin Young Choi	(Seal)
Vice Chair	Hyeong-Seok Ko	(Seal)
Examiner	Kyomin Jung	(Seal)

Abstract

Pattern Mirroring Algorithm for Fast Simulation about Symmetric Cloth

Inseo Jang
Department of Electrical and Computer Engineering
The Graduate School
Seoul National University

This paper describes the ‘Pattern mirroring algorithm’ to reduce simulation time for cloth body simulation. This method is applicable for symmetric panel and symmetric body meshes centered on YZ plane: typically, man's suit and ready-make cloth is target of this method.

As the ordinal simulation method, apply conjugate gradient method to every vertices on cloth mesh in order to solve system matrix. The problem is that the time for simulation is getting longer as the number of cloth vertices increases for high resolution. This is because the time complexity of conjugate gradient is exponential. Using pattern mirroring method, size of system matrix equation is half comparing ordinal method. So I can expect that the time for simulation reduces. The proposed method reduces simulation time up to 1.4 times (37%), by halving the matrix size of the linear equation.

At chapter 1 introduction, describe the process of simulation, method of solving system equation and collision handling. An iterative method 'conjugate gradient method' is used to determine velocity of vertices of clothes.

At chapter 2 relative work, explain about previous acceleration research for cloth simulation. At chapter 3, explain Pattern mirroring algorithm. But some problems could occur when using this method. At chapter 4, suggest solutions to handle these artifact as post-process step. At chapter 5, represent table to comparing the average time to simulate cloth in ordinal method and pattern mirroring method. Also represent image to difference of two result. Finally at chapter 6, describe conclusion and limitation of Pattern mirroring algorithm.

Keywords : pattern mirroring, cloth simulation
Student Number : 2016-25686

Contents

List of Figures

List of Tables

Chapter 1

Introduction

If real-time simulation becomes a reality, virtual fitting dressing will be possible at the clothing store. Therefore, Acceleration studies have been conducted to speed up simulation in computer graphics field. It is based on a physics-based simulation method to obtain quality similar to realistic clothes. This research also suggests an acceleration method in this context.

1.1 Time Integration Method

Physical-based cloth simulation has been problem in computer graphics. The two traditional methods to update new velocity of vertices are explicit forward and implicit backward Euler method. The difference of two method is this: the forward method's step is based solely on conditions at present time, while both condition of current and next step is used in the backward method.

In the explicit method, acceleration of vertices of cloth is obtained through velocity of current frame. The formula is

$$x(t+\Delta t) = x(t) + v(t)\cdot\Delta t \qquad [1]$$

Position x and velocity v of this time t determine position of particle at next step. The problem is that the system to determine velocity of vertices in cloth is too unstable to take a large time step, so the vertices can be diverse at next frame. In order for the time step to be small, high computational cost is needed,
because system in more frames need to be solved. In conclusion, explicit method is ill-suited to solving system equations because it requires small steps to stably advance the simulation forward, around 0.001s.

Implicit method① for cloth overcomes the performance limits in explicit method. This method allows the time larger using future velocity. The formula is:

$$x(t+\Delta t) = x(t) + v(t+\Delta t)\cdot\Delta t \qquad [2]$$

The difference is velocity v is next step $t+\Delta t$, not velocity of current step, which means current step doesn't determine position of vertices at next step $t+\Delta t$. This is quite wired that future velocity value is used, but this makes system stable even though large time step is used. Since the velocity of the next step is the derivative of the next position, the differential equation must be solved to get $x(t+\Delta t)$ in [2]. So, system matrix is consisted by implicit method which allows large time step around 0.01s.

---

① David Baraff(1998) [Large Steps in Cloth Simulation]

1.2 System Matrix

To locate vertices on the cloth mesh, amount of velocity change on each vertices $\ddot{x}$ is obtained from following Newton's second law equation which is linear equation form, $Ax = b$.

$$M\ddot{x} = f(x, \dot{x}) \qquad [3]$$

This system equation implies a force-acceleration relationship. As implicit method mentioned in [Baraff98], force from next state [position, velocity] $f(x_0 + \Delta x, v_0 + \Delta v)$ is used to get the amount of velocity and position change, where h is time step.

$$\Delta v = hM^{-1} \cdot f(x_0 + \Delta x, v_0 + \Delta v), \ \ \Delta x = h \cdot (v_0 + \Delta v) \qquad [4]$$

In this equation, the force is approximated from force Jacobian of velocity and position of current frame and the amount of change for position and velocity.

$$f(x_0 + \Delta x, v_0 + \Delta v) = f_0 + \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial v}\Delta v \qquad [5]$$

Eventually, system matrix is expressed by following equation that consists of the force Jacobian in mass matrix and force vector comparing equation[3].

$$(M - h\frac{\partial f}{\partial v} - h^2\frac{\partial f}{\partial x})\Delta v = h(f_0 + h\frac{\partial f}{\partial x}v_0) \qquad [6]$$

Assuming total number of the particle in a cloth mesh is n, size of matrix M and Jacobian $\frac{\partial f}{\partial v}$, $\frac{\partial f}{\partial x}$ is $(3n * 3n)$. Size of column vector Δv and f_0 is $3n$.

1.3 Conjugate Gradient Method

To solve system equation[5], directly inversing the matrix is intuitive, but it's commonly not easy to get inverse of matrix as size of matrix is getting large. The iterative method is suitable because normally the number of vertices in a cloth panel is over 1000.

The conjugate gradient method(CG) is an iterative method that solves $A\Delta v = b$ when matrix A is symmetric and positive-definite matrix. The iteration stops if $b - Ax < \varepsilon$ when ε is user -defined tolerance value and x is estimated value of this iteration. Pseudo code of CG is following above.

Algorithm 1 Pseudo code of Conjugate gradient

```
procedure CG(A,b,x,tol)
    Ax <- A * x
    r <- b-Ax
    r2  <- dot(r,r)
    iter <- 1
    while r2 > tol*tol do
        if iter > 1 then
            p <- r+ (r2/r2old)*p
        else
            p <- r
        end if
        Ap <- A * p
        dot <- dot(p, Ap)
        alpha <- r2/dot
        x <- x+  alpha * p
        r <- r − alpha *Ap
        r2old <- alpha*Ap
        r2 <- dot(r, r)
        iter <- iter + 1
    end while
end procedure
```

One of time-consuming part is solving the large linear equation. As n is larger (which means cloth has a larger number of vertices), the longer it takes to solve this linear equation, because time complexity to get the inverse of matrix is $O(n^3)$. Instead of getting A^{-1}, use iterative methods for solving linear equation such as conjugate gradient method whose time complexity is $O(n^{1.5})$ ②. With the proposed Pattern-mirroring algorithm, the expected speed up for solving the linear equation is $2^{1.5}$ as the number of vertices included in the calculation is halved.

1.4 Collision Handling Method

The other time-consuming process is collision detection and handling step. It can occur between cloth and body, between different clothes mesh or same clothes mesh (self-collision). To check collision, first, making axis aligned bounding box (AABB) and repeating for smaller region, composing a bounding volume hierarchy (BVH) for every object. For checking intersection between two objects, compare the largest bounding box to check whether two bounding boxes overlap. If it is, check the right smaller bounding box and iterate this process until checking region arrives leaf node (collision detection process). If there is intersection between two leaf nodes, apply repulsive force to each vertex to let separate object apart (collision response process).

I expect that time spent in collision reduce small in detection and large in response. In detection, halving the number of vertices leads to smaller size of searching tree. However, as BVH searching

---

② J. Shewchuk. An Introduction to the Conjugate Gradient Method without the Agonizing Pan. Technical Report CMU-CS-TR-94-125, Carnegie Mellon University, 1994 traverse is proportional to $\log n$, height of tree does not shrink significantly as the number of vertices decrease half. In response, assuming collision is reduces to about half, time for collision response decrease in half as well.

1.5 Overview of Pattern Mirroring Algorithm

Since real-time simulation is goal for researcher in computer graphics, simulation speed is important issue. Even though there are many effort and research, such as parallel GPU programming, data-driven method and machine learning, simulation acceleration is still unresolved issue. I suggest fast simulation algorithm that can improve the speed by about 1.4 times for limited target, symmetry cloth mesh. The pipeline of Pattern mirroring is following.

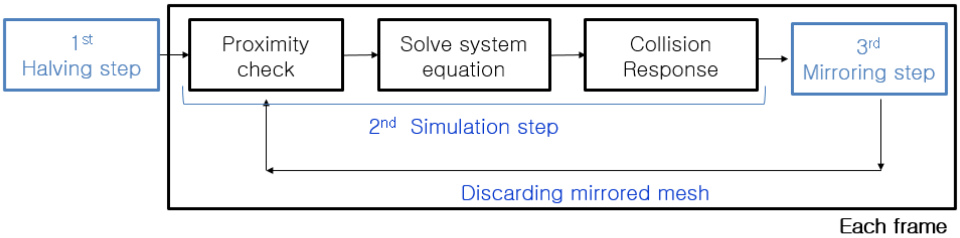


Figure 1.1: Pipeline for Pattern mirroring algorithm

Proximity check (collision detection), solving system equation and collision response in black boxes are traditional procedure in cloth simulation. In the other hand, halving step and mirroring step in blue box are procedures in Pattern mirroring algorithm, which will be described in chapter3.

Chapter 2

Previous Work

Stable and fast cloth simulation has been a problem for a decade. Prior researches have been made to overcome this limitation. [Baraff and Witkin 98] [1] proposed implicit method to stably simulate cloth in large time steps with energy function and it's derivative. Also [Volino and Magnenat Thalmann 00] [2] propose adding damping forces to simulation stability. But artificial damping force disturbs to generate wrinkles on the cloth. [Choi and Ko 02] [3] proposed a semi-implicit technique which makes simulation stable.

Collision handling methods and acceleration techniques have been introduced. [PROVOT 97] [4] proposed collision detection method. [Baraff and Witkin 03] [5] solved self-collision problem using history-free method. [Bridson and Anderson 2002] [6] proposed robust collision detection method and collision response in history-based approach to process contact and friction between cloth and obstacle. [Wicke and Gross 06] [7] proposed the collision method with intersection with boundaries based on history-free algorithm.

Chapter 3

Pattern Mirroring Method

Pattern mirroring algorithm based on the insight of the cloth garment's structure. Since human body model and basic cloth mesh are commonly symmetric, symmetricity used to speed up the simulation. Target of this algorithm is symmetric garment to the center plane, such as read-to-wear garments and man's suit and body mesh with symmetric motion. A simple post-process step is required for garments that have additional layers such as pocket, which harm the symmetricity: stitching the additional layer on basic mesh simulated by Pattern mirroring algorithm. The main idea of mirroring method is to solve the linear equation only for symmetric half mesh (M^{origin}) and duplicate the result to rest mesh(M^{mirror}).

The process of Pattern mirroring algorithm is as follows. The first step is preparation, set the constraint plane and halve original cloth mesh. The second step is simulation, solving the system equation and collision for half panel. The third is mirroring step, duplicate position of simulated mesh to rest half mesh. The last step is optional post-processing that can apply when there is a problem. The problem is that some vertices could penetrate other plane near the constraint plane and handling artifacts method is described in chapter 4.

3.1 1st step: Constraint Plane and Halving Mesh

At the first step, set the YZ-plane as ‘constraint plane’ and the vertices on this constraint plane as ‘constrained vertices’, before simulate cloth. In figure 3.1, constraint plane of symmetry jacket is in the middle of the original mesh, whose x position is 0. Constrained vertices can’t move x-direction, enforced to be on constraint plane, which means they are in 2-degree of freedom. This constraint prevents the half mesh slip down on body at [2nd step: simulation for half panel]. Halving mesh step is simply deleting one side mesh of the constraint plane. The criteria for cutting the mesh is as follow with reference to figure 3.3:

1. If even one of 3 vertices making up a triangle is on the side of original mesh direction $\boldsymbol{M}^{origin}$, save the triangle. Original mesh direction is direction of the mesh being simulated from constraint plane. Move the vertex in opposite direction onto the constraint plane(green arrow).
2. Otherwise, if all 3 vertices of a triangle are in opposite side, delete the triangle.(red X)

Result mesh of this step is the half mesh represented in Figure3.2: T-shirt and in Figure 3.4: square.

However, there can be erroneous triangles still remaining in the other directions, for un-symmetric mesh in figure 3.4 (b). This problem is solved by modifying the shape of the triangle, projecting crossed vertices to constraint plane. Even though by applying this step, shape of mesh would change, the outward appearance of the cloth is same.

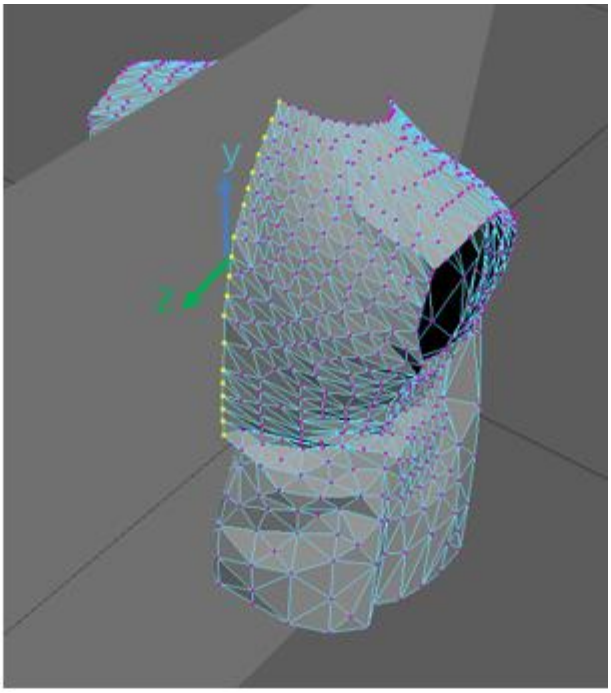


Figure 3.1: original jacket mesh and constrained vertices(yellow) on the contraint plane.

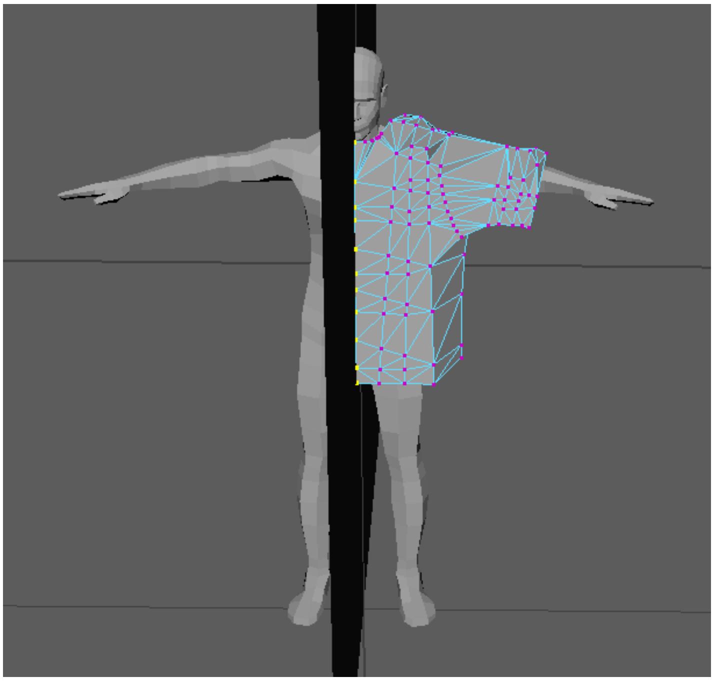


Figure 3.2: t-shirt mesh M and constrained vertices (yellow) on constraint plane(black).

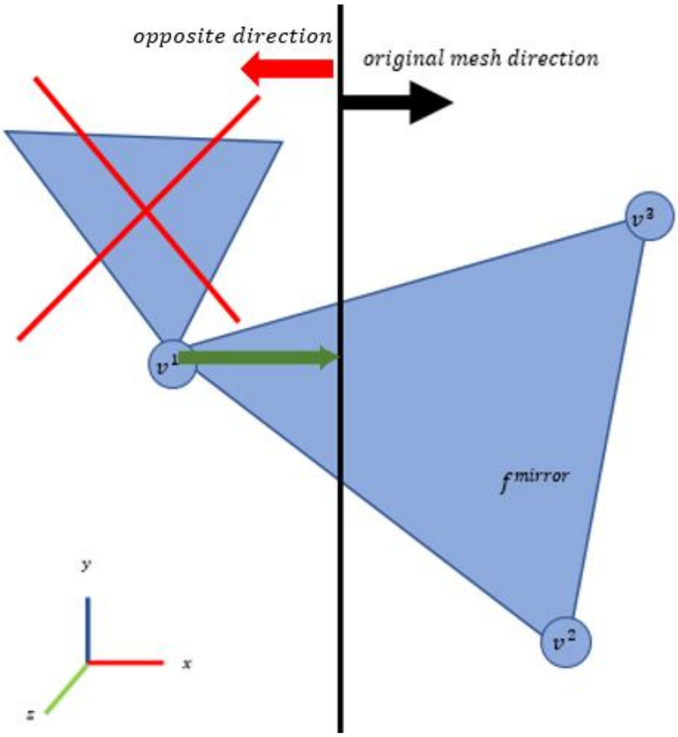
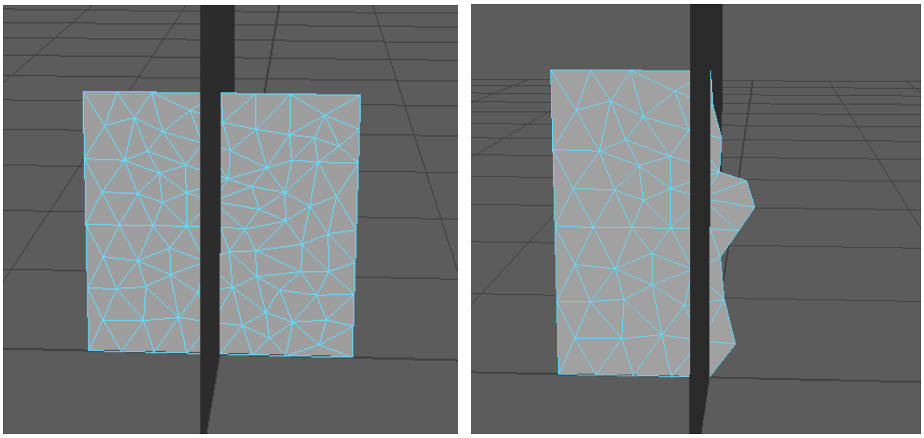


Figure 3.3: The criteria for cutting the mesh as half



(a) (b)

Figure 3.4: Un-symmetric square mesh
Half mesh after 2nd halving step (a)
Some triangles in other direction (b)

3.2 2nd step: Simulation for Half Panel

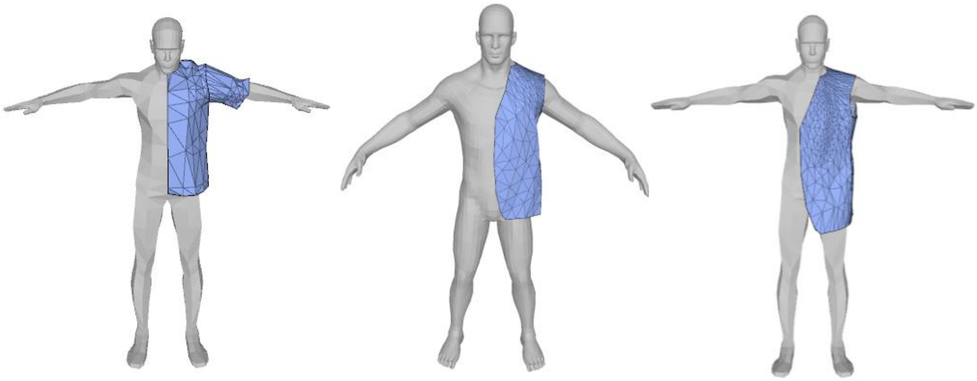
The second step of mirroring is solving linear equation [1] and collision handling for only M^{origin}. The result of simulation of T-shirt, jacket is in Figure 3.5.

Let's think about the time gain by Pattern-mirroring algorithm comparing with original method.

First in solving system equation, the simulation speed is up to $2^{1.5}$ times faster in solving system equation irrespective with the number of vertices, as the vertex number decrease as half. Because the time complexity of conjugate gradient method is $O(n^{1.5})$ as n is the vertex number.

Second in collision detection, it is hard to expect a big improvement in speed. The biggest part of the time for collision detection is BVH traverse search to find node that collision occur. So the complexity is prorportional to height of BVH, $\log n$. Even if total vertex number is reduced to half, it doesn't lose a lot of parts to be searched.

Third in collision response, the time for collision hanlding is the 2 times less, because region to apply repulsion force is reduced to half.



(a) T-shirt (b) Jacket (c) High resolution jacket

Figure 3.5: Result of simulation for half mesh at 2nd step

	solver	Collision	
		Detection	Response
Time complexity	Conjugate Gradient $O(n^{1.5})$	BVH traverse $O(\log n)$	Repulsion force $O(n)$
Expected Time Reduction $(\boldsymbol{n} \rightarrow \frac{\boldsymbol{n}}{\boldsymbol{2}})$	$\frac{n^{1.5}}{(\frac{n}{2})^{1.5}} = 2^{1.5} \approx 2.8$	$\frac{logn}{\log(\frac{n}{2})} = 1 + \frac{log\ 2}{log\ n - log2} \approx 1$	2

Table3.1: Expected time reduction by Pattern-mirroring algorithm

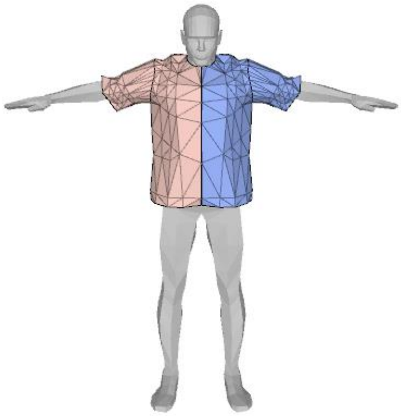
3.3 3rd step: Mirroring Half Mesh

The third step is reflecting simulated mesh to opposite side about constraint plane, symmetrically in x direction with respect to the constraint plane. Copy the position of vertices on original mesh M^{origin} to that of mirrored mesh M^{mirror}, multiplying x position by -1. As represented in figure 3.6, position of M^{origin} (blue) and M^{mirror} (red) are symmetric.

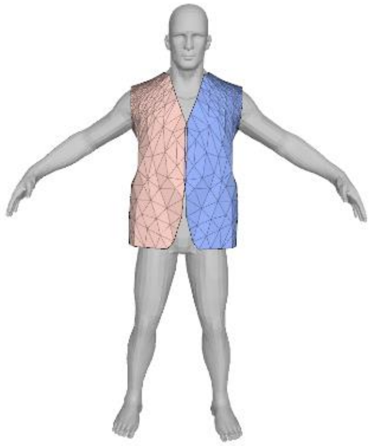
At the last, render M^{origin} and M^{mirror} after performing 3 steps. After rendering M^{origin} and M^{mirror}, this frame ends by discarding mirrored mesh M^{mirror}. Repeat these steps at every frame. In the next frame, simulation is performed using the position and velocity M^{origin} of this frame.

Unfortunately, jacket with high resolution example in Figure 3.7 (c), some unconstrained vertices in mirrored mesh cross the constraint plane and pass to already simulated region at middle part. Because the cloth is hanging down, passing through the center plane. This artifact is resolved in two ways that user can choose:

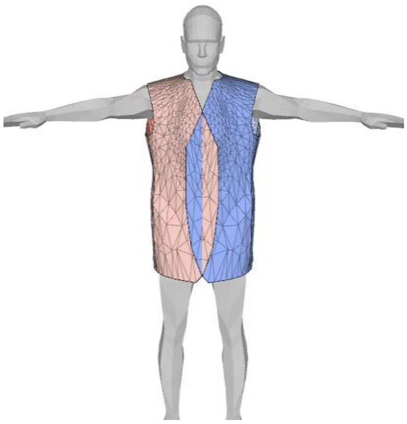
1. Mesh pulling: This method is useful if user doesn't want two mesh (M^{origin}, M^{mirror}) to invade other mesh, so that user can attach button to connect two meshes keeping the symmetry. The detail of two methods are explained at chapter 4.2, method 1.
2. Mesh Overlap: If user want the cloth to adjust his/her dress, move some vertices of M^{mirror} to overlap M^{origin}. Symmetry is not hold in appearance, even though target mesh is symmetric. In this case, modify position of M^{mirror} in order to for M^{mirror} to cover M^{origin} above. The details of two methods are explained at chapter 4.2, method2.



(a) T-shirt



(b) Jacket



(c) High resolution jacket

Figure 3.6: Mirrored result of Figure 3.5 at 3rd step

Chapter 4

Following Artifacts and Solution

This chapter discusses how to solve the artifacts that can happen in mirroring algorithm as mentioned in chapter 3. There are two problems, 1. Vertices lineup. 2. Penetration in 3rd mirroring step.

Described the criteria to cut whole mesh M into half mesh $M^{original}$ in chapter 3.1. However, the first problem is that vertices do not line up along the constraint plane, because some triangles are across the constraint plane. Explain how to change shape of crossed triangles in chapter 4.1. This phenomenon occurs because the composition of the mesh is not completely the same on the right and on the left, even if the composition of the clothes is symmetrical.

The second problem is penetration near constraint plane by reflection original mesh M^{origin} to make M^{mirror}, in chapter 3.3. User can choose how to align invading vertices in two ways: pulling mesh to sew zipper, overlapping mesh to put button.

4.1 Project crossed vertices at halving step

In this section, explain the solution of problem that can be happen at step1, halving step: some vertices can be in wrong place. The solution is to project the vertices closed to constraint plane to line up, as Figure4.1.

Whole mesh M changes to half M^{origin} (a) in [section3.1: halving step]. But the mesh illustrated in (a) is not suitable to simulate for some vertices invading opposite side of constraint plane. By projecting the vertices in wrong place to constraint plane, all triangles included M^{origin} should be in one side of constraint plane (b), for original mesh and mirrored mesh to not be overlapped. This method keep symmetry between M^{origin} and M^{mirror}.

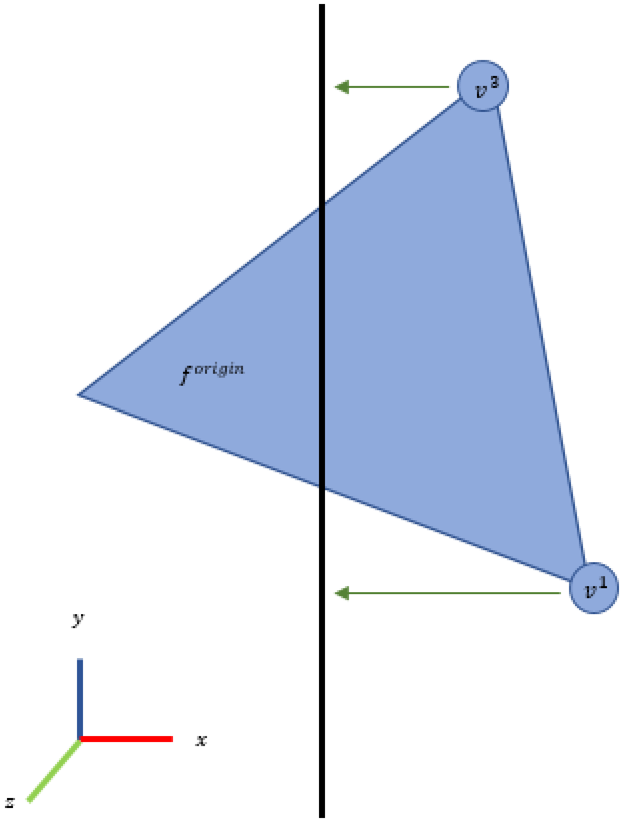
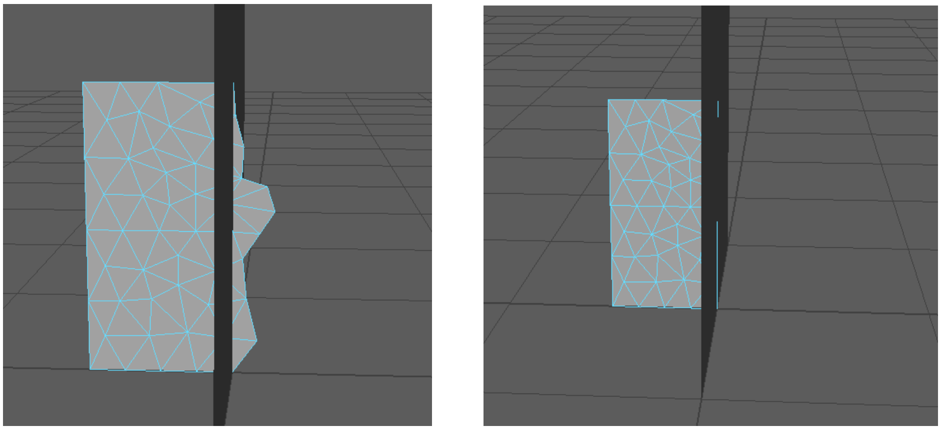


Figure 4.1: Projection to constraint plane.
v^1 and v^2 are invading vertices in M^{origin}.



(a) (b)

Figure 4.2: Half mesh after 2nd halving step. Some triangles in other direction (a), Project crossed vertices to constraint plane (b)

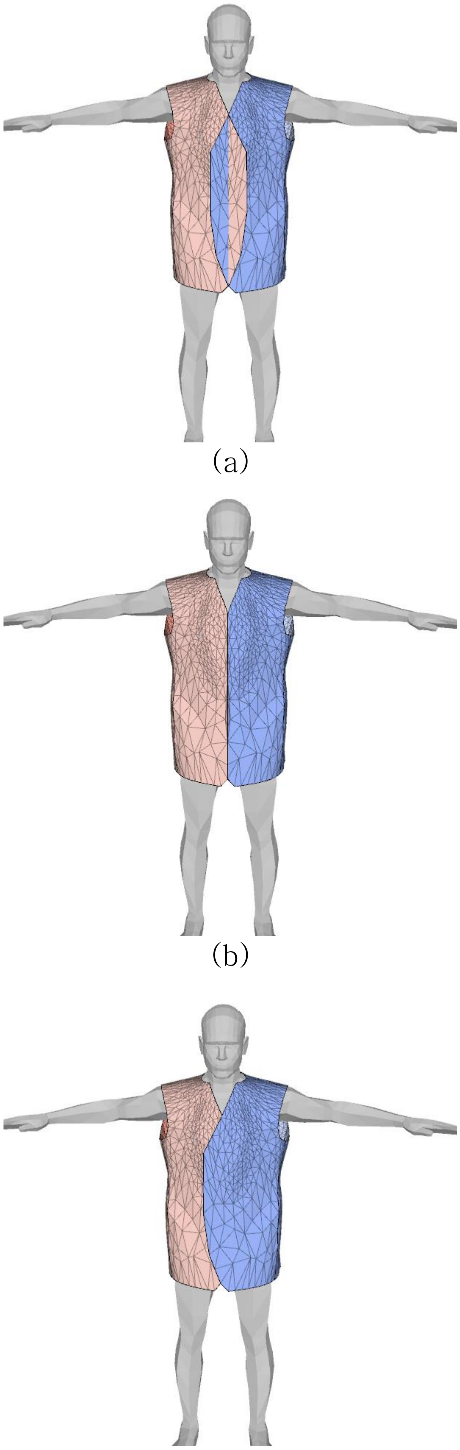
4.2 Penetration between original and mirrored mesh

In this section, explain the solution of problem that can be happen at step3: mirroring step. Though half mesh by halving step at section 3.1 gets constraint vertices by projecting invading vertex to constraint plane at section 4.1, there are some un-constraint vertices near constraint plane because user can want vertices in some region to move free. When the unconstrained vertices passed constraint plane, unconstraint vertices in original mesh M^{origin} can penetrate mirrored mesh M^{mirror}. Some edge lines belonging to blue panel penetrate red panel near the chest (Figure 4.2(a)). This problem is caused by the fact that some unconstrained vertices cross the constraint plane during simulation step.

This is optional step called as [post-process step] that if user considers the simulation result is in error, he/she can apply these methods. There are 2 choices to solve penetration: meshes pushing each other, overlap.

Method 1) meshes pulling case

This method prevents crossed vertices to invade other side so that user can sew zipper. In this case, apply manipulate position of end vertex in M^{origin} to attach constraint plane if it crosses the plane, to prevent it to invade other side. Although this is only case that shape of M^{origin} changes, this method keeps symmetry between meshes.



(a)

(b)

(c)

Figure 4.3: Jacket scene: mirroring method with penetration (a), mesh pulling case (zipper) (b) and overlap case (button) (c)

Method 2) overlap case

The goal of this method is overlapping simulated mesh M^{origin} and M^{mirror} by manipulate position of M^{mirror}, so that user can put a button on the cloth.

[Figure 4.2] represents how vertex (v^{mirror}) of M^{mirror} (red) moves, so that M^{origin} (blue) cover to M^{mirror} (red), completely seen from the front, camera location at positive z-direction. The black line is constraint plane and triangle f^1, f^2 is in M^{origin} (blue), and f^{mirror} is in M^{mirror} (red).

M^{mirror} is chosen to move because it is discarded every frame after rendering before solving system matrix. As moving to negative z direction, invading vertex (v^{mirror}) of M^{mirror} hides between M^{origin} (blue) and body.

First, finding a triangle of M^{origin} whose axis align bounding box (green box) includes a crossed vertex (v^{origin}) of M^{mirror}. Second, find the minimum z-position value among 3 vertices (v^1, v^2, v^3) of triangles f^2. Finally, move v^{origin} to minimum z-position value.

Because triangle f^1 included in M^{origin} is in front of the triangle f^{mirror}, it is well overlapping f^{mirror}. However, as f^{origin} penetrates between f^1 and f^2, f^2 is at the back of v^{origin}. v^{origin} should be moved to negative z-direction until its z position reach minimum z position value among 3 vertices (v^1, v^2, v^3) of triangles. Changed the z value of triangles is (z value of min (v^1, v^2, v^3) - ε), where ε is small value(≈0.005), for moved vertex to be back of f^{origin}.

Using this method, the covered surface of the garment (red) may look rugged but does not affect the simulation quality because it is not visible from the outside.

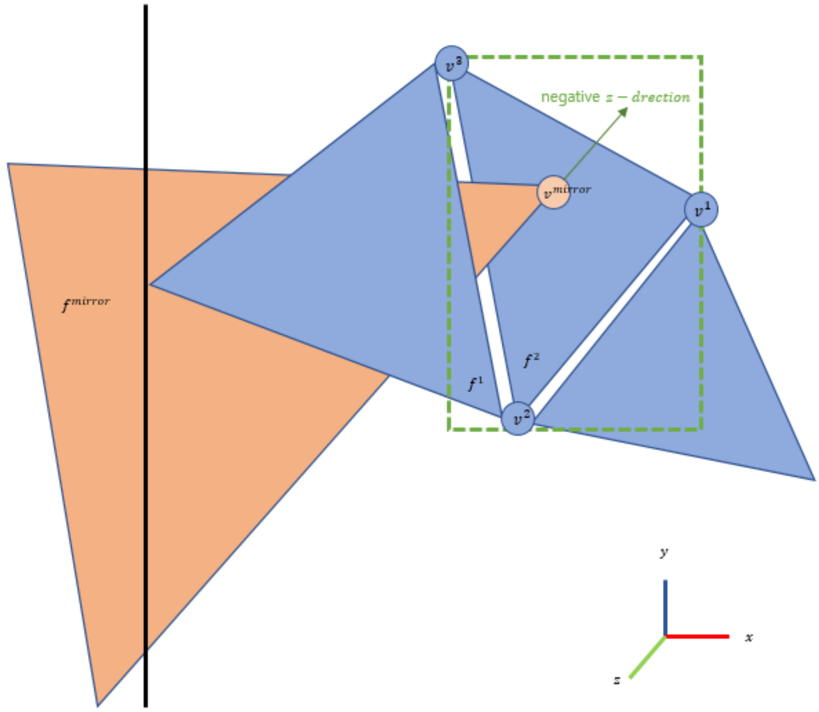


Figure 4.4: How to solve penetration. f^{origin} is fixed and f^2 moves to positive z direction.

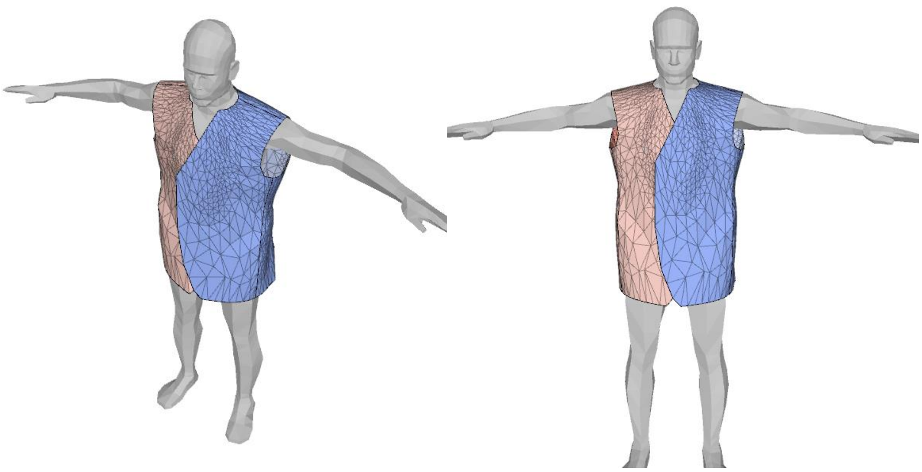


Figure 4.5: Jacket scene after penetration solved.
Side view (left) and front view (right)

Chapter 5

Experiment Result

We performed 2 experiments, T-shirt, jacket using ARCsim, open source simulator from University of California-Berkeley. It's good to compare with the results of other papers because ARCsim is commonly used simulator in cloth field.

First experiment is basic experiment, draping T-shirt on body in order to explain overall procedure. Draping jacket on body is to explain solution of penetration.

I compare original method and Pattern mirroring algorithm by comparing how much time spends in simulation. I represent averaging time using table and simulation time for each frame through graph. I use conjugate gradient solver to solve system matrix. The formula to evaluate how much mirroring method is faster than ordinary method in two ways.

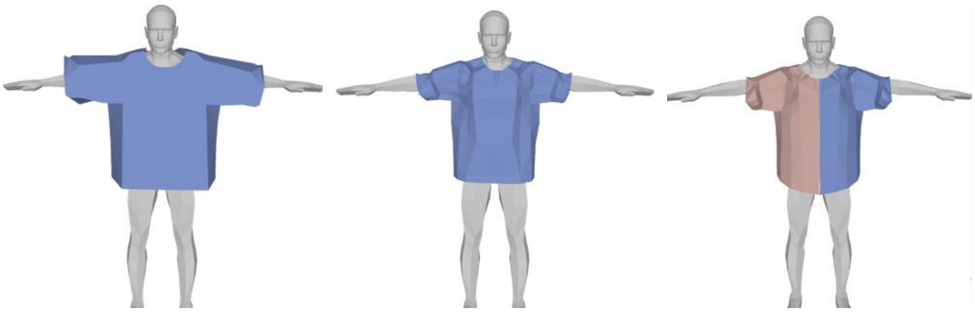
1. (original method – mirroring method)/original method*100 (%)
2. Original method / Mirroring method (x times)

The first one is difference of time spent between two methods in percent. It describes how much improved compared original method, so I choose this to estimate how much time improved. The second one is relative ratio of two method.

5.1 T-Shirt experiment

This is basic experiment to evaluate performance of mirroring algorith. The number of vertex in a full mesh is 329, and half mesh is 173. Figure 5.1 is comparing result of ordinary method and mirroring method with overlapping. Table5.1 is comparing time spent for simulation of ordinary method, and mirroring method in solver, collision and total time. Figure 5.2 show the time taken in each frame as graph. Collision detection is not represented in the graph because the two differences are not large.

Using mirroring method is x2.42 faster in solver time, x1.16 in collision time, and x1.49 in total simluation time than original method.



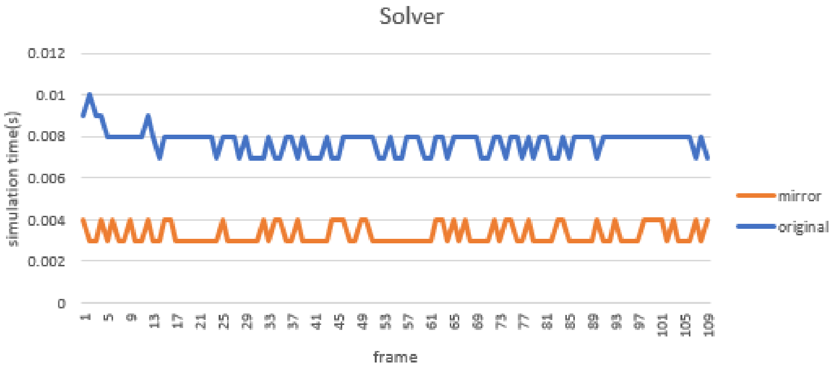
(a) (b) (c)

Figure 5.1: Comparing result of T-shirt example.
Panel of t-shirt before simulation (a),
simulation of original method (b) and mirroring method (c)

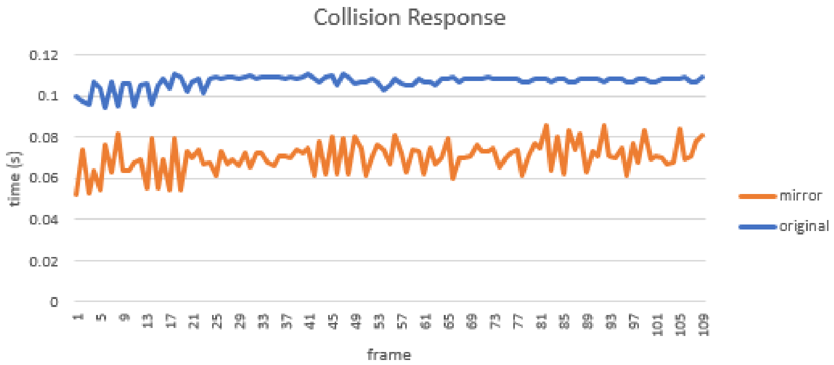
Average time (s)

	Solver (A)	Collision Detection(B)	Collision Response(C)	Total (A+B+C+alpha)
Original method	0.055149	0.750213	0.309153	1.674535
Mirroring method	0.01452	0.710525	0.14459	1.1455
Speed-up ($\frac{original\ mehtod}{mirroring\ method}$)	x 3.798159	x 1.055857	x 2.138079	x 1.461837

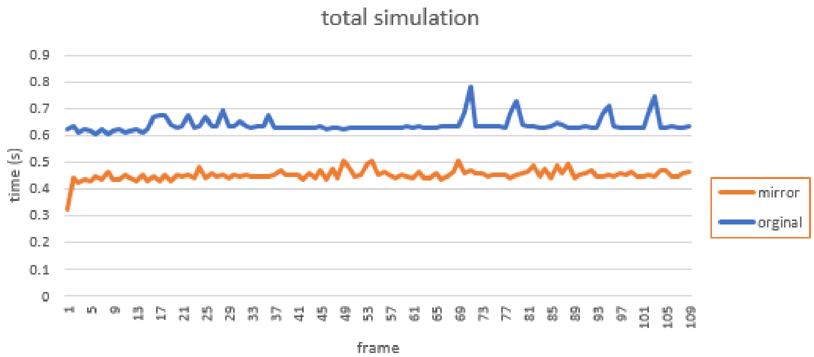
Table 5.1: Comparing time between original method and mirroring method



(a)



(b)



(c)

Figure 5.2: Time taken to simulate T-shirt in each frame as graph in solver (a), collision response (b) and total time (c)

5.2 Jacket Experiment

The first experiement is jacket simulation consisted of 1934 vertices which has the constrainted plane only on the back on human body, 807 vertices. The number of vertices of half mesh by 2nd halving step is 967.

5.2.1 original method vs mirroring method

Figure 5.1 is comparing result of ordinary method and mirroring method with overlapping. Table5.1 is comparing time spent for simulation of ordinary method, and mirroring method in solver, collision and total time. Using mirroring method is x2.2 faster in solver time, x1.72 in collision time, and x1.4 in total simluation time than original method.

In chapter 3.2, I guess that 'speed up in solver time is up to $2^{1.5} \approx 2.8$ in mesh in solver. The time is 2 times in collision step.' The guess was quite reasonable. Figure 5.2 and 5.3 is graph to compare time between ordinary method and mirroring method for each frame.

However time reduction in total time is smaller than in solver and collsion step. The reason that time reduction in total is smaller than time reduction in solver and in collision is there is other most time-consuming part, proxmity checking.

5.2.2 with / without overlapping

I compare mirroring method with and without overlapping, to know how much time spend to search bounding box of every triangle in M^{mirror} to find overlapped triangle with invading vertex of M^{origin}. Composing bounding box for every triangles in M^{mirror} could be quite high cost process. But the time cost for search aabb is less than 0.01second in system and collision, igornable.

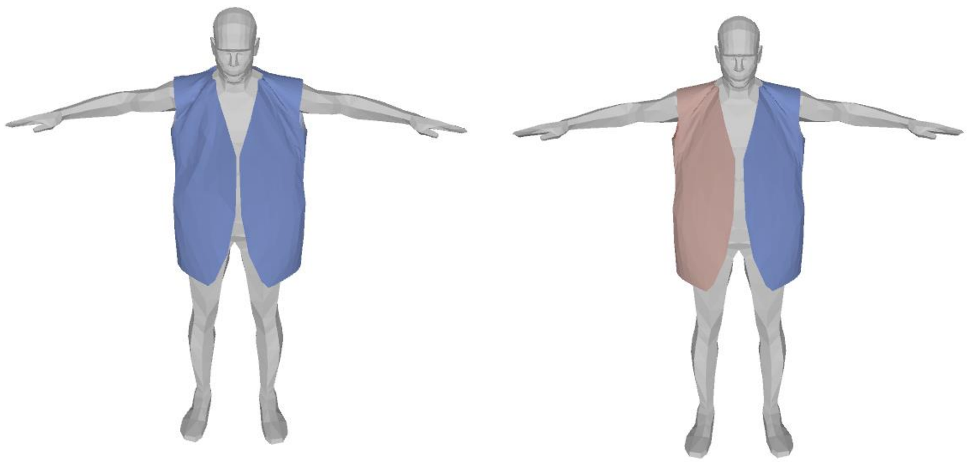
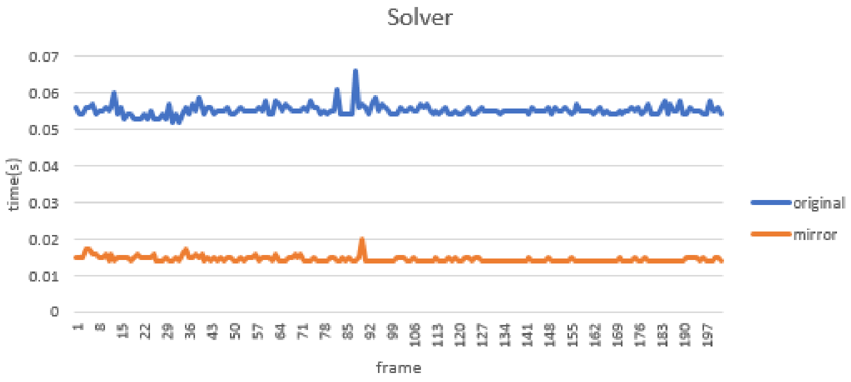


Figure 5.3: Comparing result of jacket. Original method (left) and mirroring method (right)

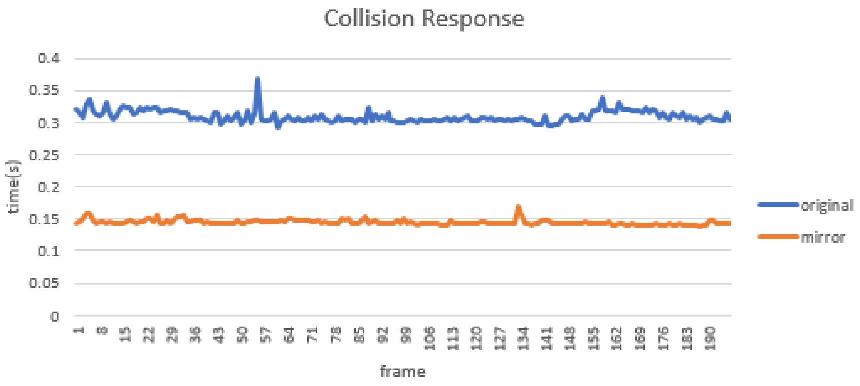
Average time (s)

	Solver (A)	Collision Detection(B)	Collision Response(C)	Total (A+B+C+alpha)
Original method	0.055149	0.750213	0.309153	1.674535
Mirroring method	0.01452	0.710525	0.14459	1.1455
Speed-up ($\frac{original\ mehtod}{mirroring\ method}$)	× 3.798159	× 1.055857	× 2.138079	× 1.461837

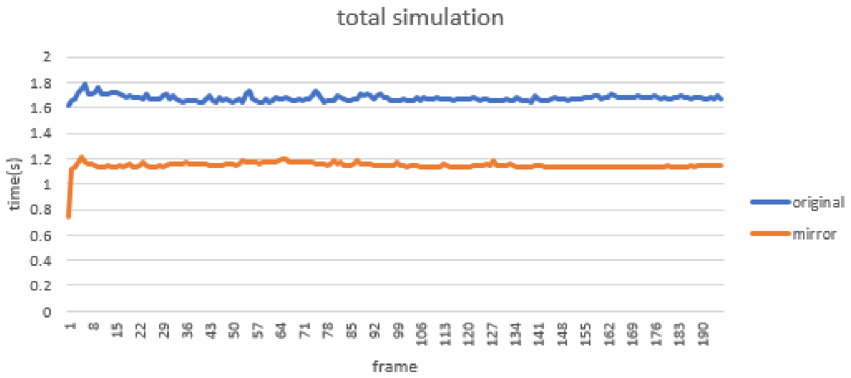
Table 5.2: Comparing time between original method and mirroring method



(a)



(b)



(c)

Figure 5.4: Time taken to simulate jacket in each frame as graph in solver (a), collision response (b) and total time (c)

5.2.3 Coarse vs dense mesh

In chapter 3.2, I guessed 'time reduction rate is consist irrespective with the number of vertices.' I check consistency by simulate different number of verices in mesh in solver and collision procedure. The number of vertices of full coarse mesh is 69, half mesh is 37 and full dense mesh is 1934 and half dense mesh is 967. There is some error, but it's the same as expected.

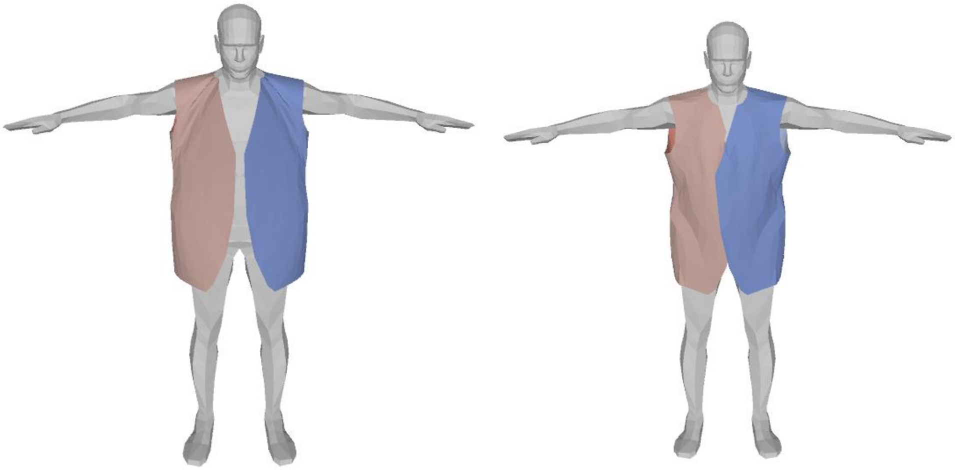


Figure 5.5: Mirrored coarse (left) and dense (right) jacket.

Average time (s)

		Solver (A)	Collision (B)	Total (A+B+alpha)
Coarse (# vert:37)	Original method	0.0008	0.0498	0.2575
	Mirroring method	0.0004	0.0586	0.2133
	Speed-up ($\frac{original\ mehtod}{mirroring\ method}$)	× 2	× 1.1767	× 1.2072
Dense (# vert:967)	Original method	0.0625	0.3880	2.8104
	Mirroring method	0.0282	0.2250	1.9570
	Speed-up ($\frac{original\ mehtod}{mirroring\ method}$)	× 2.2163	× 1.7244	× 1.4360

Table 5.3: Comparing speed-up between coarse mesh and dense mesh

Chapter 6

Conclusion

We propose a fast simulation method based on implicit method under a condition that cloth and body is both symmetric. Considering basic layer for complex garment is usually symmetric, this method can be widely used. The relative speed-up for cloth simulation using our method is x1.4 comparing original method, as the number of vertices of mesh increases the value is bigger.

However, the limitation is if cloth has fixed node hold. For example, hanging cloth with two fixed vertices swings forward and backward in Figure6.1. Vertices on the middle of the cloth (yellow) can be hold when resultant force from connected edges is zero. Once simulating for one half mesh, the vertices on the middle of cloth(yellow) takes on-zero force that makes it move left (orange arrow) or right (blue) side. The vertices will be fall from the lack of vertical force forward up, even though they are on constraint plane. In conclusion, this limitation come from problem of mirroring algorithm that only duplicate the simulated position, not force applied to vertex.

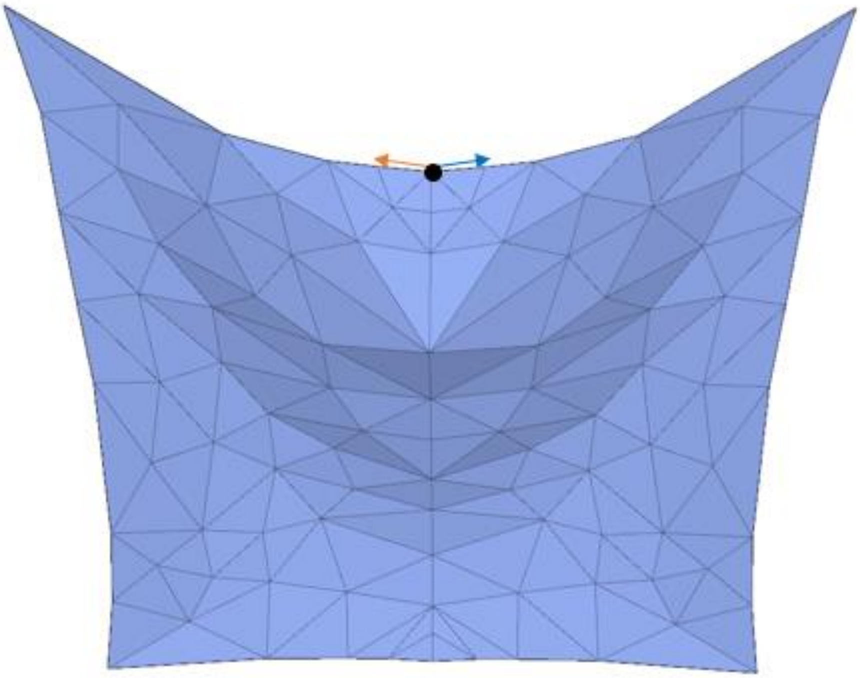


Figure 6.1: limitation: mirroring method can't apply to hanging cloth. If we simulate only half one side of mesh, vertices on the middle of cloth would fall even though they are on constraint plane, because a vertex (black) takes force of one side (orange or blue arrow).

Bibliography

① Gavin Hayler(2004) [Implicit and Explicit Methods of Cloth Simulation]

②[1] David Baraff, Andrew Witkin, Large Steps in cloth simulation, *SIGGRAPH,* 1998.

③ J. Shewchuk. An Introduction to the Conjugate Gradient Method Without the Agonizing Plan. Technical Report CMU-CS-TR-94-125, Carnegie Mellon University, 1994.

[2] Pascal VOLINO, Martin COURCHESNE, Nadia Magnenat THALMANN, Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects. Computer Graphics (Proc. SIGGRAH, 1995)

[3] Kwang-Jin Choi, Hyeong-Seok Ko, Stable but Responsive Cloth. ACM Trans. Graphics 2002.

[4] Xavier PROVOT, Collision and self-collision handling in cloth model dedicated to design garments, pp.177-189, Computer Animation and Simulation, 1997.

[5] David Baraff, Andrew Witkin, Michael Kaas, Untangling Cloth, *ACM SIGGRAPH 2003.*

[6] Robert Bridson, Ronald Fedkiw, John Anderson, Robust Treatment of Collisions, Contact and Friction for Cloth Animation,

ACM TOG SIGGRAPH pp.594-603, 2002.

[7] Martin Wicke, Hermes Lanker, Markus Gross. Untangling Cloth with Boundaries. Proceedings of Vision, Modeling and Visualization 2006

초 록

본 논문은 의상-바디 시뮬레이션의 속도 향상을 위한 패턴미러링 방법을 제시한다. 이 방법은 몸 매쉬와 옷의 패널이 위치한 Y-Z평면에 대해 대칭일 경우에 사용가능하다. 보통의 남성복이나 기성복과 같은 옷이 좌우가 대칭인 경우가 많다.

기존 시뮬레이션에서는 모든 옷의 정점들에 대해 conjuate gradient 방법을 이용해 시스템 행렬을 풀었다. 문제는 Conjugate Gradient 방법은 정점 수에 대해 지수적인 시간 복잡도를 가지므로, 고해상도를 위해 정점의 수가 증가할수록 시뮬레이션 시간이 지수적으로 증가한다는 것이다. Pattern Mirroring 방법을 이용하면 계산해야하는 시스템 방정식의 양이 반절로 줄어들기 때문에, 시뮬레이션에 필요한 시간도 줄어들거라고 기대할 수 있다. 결과적으로 패턴미러링 알고리즘을 이용하면 1.4배 (37%)의 속도 향상을 보였다.

1장 도입에서는 옷을 시뮬레이션하는 과정인 시스템 방정식을 푸는 방법, 충돌처리를 하는 방법에 대해 설명한다. iterative method인 conjuate gradient가 옷의 정점들의 속도를 결정하기 위해 사용되었다. 2장 관련 연구에서는 시뮬레이션 가속화를 위한 연구를 소개한다. 3장에서 pattern mirroring 알고리즘을 소개한다. 4장에서는 패턴 미러링 방법을 사용한다면 발생할 수 있는 문제가 몇가지 있는데, 이 문제를 해결하는 방법에 대해 설명한다. 5장에서는 패턴 미러링 방법을 기존의 시뮬레이션 방법과 비교해서 속도 향상을 도표로 제시하고, 결과 이미지를 비교한다. 마지막으로 6장에선 결론과 패턴미러링의 한계를 서술한다.

주요어 : 패턴 미러링, 결레 기울기법
학 번 : 2016-25686

감사의 글

2016년 여름에 입학함으로써 시작되었던 석사과정이 2019년 2월 졸업을 함으로써 2년 반의 시간이 지나갔습니다. 먼저 그래픽스에 대해 무지하던 저를 가르쳐주시고 이끌어주신 고형석 교수님께 덕분에 제가 석사학위를 마칠 수 있었습니다. 부족한 저를 제자로 받아주시고, 코딩의 세계로 이끌어주신 것에 대해 진심으로 감사의 인사를 올립니다.

바쁘신 와중에 심사위원이 되어주신 최진영 교수님, 정교민 교수님께도 감사의 인사를 드립니다.

논문을 작성할 수 있을 정도까지 성장할 수 있었던 이유는 뛰어난 선배, 동기들과 함께 연구 할 수 있었기 때문입니다. 학업의 과정 동안 어려울 때마다 든든한 힘이 되어준 연구실의 선배들에게 먼저 감사의 인사를 전합니다. 문제가 있을 때마다 새로운 관점과 아이디어를 제시해주시는 상빈이 형, 끝까지 논문을 검토해주시며 피드백을 주셨던 동훈이 형, 구현을 힘들어 할때마다 먼저 다가와서 조언해주시는 익훈이 형에게 특별히 감사드립니다. 다들 연구로 바쁜 와중에도, 친절하게 질문에 대답해주신 원섭이 형, 원영이 형, 임영이 형 모두 감사드립니다. 그리고 매번 밤을 새며 공부하고 연구한 동기와 후배들, 현승이 형, 영균이 형, 경원이, 경현이 형, 창진이 모두 연구에 성공해서 좋은 논문을 쓰시길 기도합니다. GML 가족들 모두 행복하시길 바랍니다.

사랑하는 가족들이 없었다면 저는 이자리에 없었을 것 입니다. 저를 위해 기도해주시는 어머니, 늘 현명한 조언을 해주시며 저를 공학도로 이끌어 주신 아버지, 그리고 엄마처럼 함께해준 누나, 모두 고맙고 사랑합니다.