Chapter 1

Cloth Modelling and Re-targeting

This chapter presents an automatic pattern based cloth modelling method that generates fit cloth for characters. Based on character measurements acquired by the method presented in chapter.**??**, the shape of cloth patterns are automatically adjusted. This method is able to create fit cloth efficiently for different characters with different body shapes and proportions without tedious manual operation that is required in traditional pattern based cloth modelling methods.

1.1 Introduction

In computer animation, creating assets such as models or scenes for a film is time consuming and costly. Therefore, the reusability of assets is curial to the film production budget. For rigid model such as cars or buildings, once created, they are stored into a asset warehouse and can be used in other films with little tweaking required. Such assets can also be traded among animation studios or animation artists. However, for clothes, their shapes follow the profile of a character, in most cases, they are bespoken for a particular character. Usually, constructing a complete set of outfits for an animation

character takes a well trained animation artist several days to finish. Due to the large difference between character designs among different films, reusing cloth models requires a large amount of manual operation to fit a cloth to another character with different body shapes and proportions.

Cloth is a soft object whose shape is determined by its design, body shape and posture of the wearer. Same cloth dressed by different character will results different cloth shapes. Moreover, in computer animation, characters are usually modelled in different postures, it is difficult to define a unified 3D form for cloth model to be stored, distributed and reused because for every different character, 3D cloth model needs to be heavily modified in order to cope with the new character body shapes and proportions.

In fashion industry, after generations of development, cloth pattern has become an important medium that transfers cloth design into wearable object. Cloth patterns are the basic items that constitutes a functional cloth. In order to implement a cloth design, firstly, the design needs to be broken down into a set of textile pieces, then through the assembling techniques such as stitch and adhesive, pieces are assembled together in a predetermined order to form a complete cloth. The process that breaks down a cloth design into patterns is called "patternmaking" (Armstrong 2000). Through this process, cloth design is transferred from design concept into a set of patterns that can be cut from the raw textile material. Therefore, the same cloth design can be reproduced easily(Hannah 1919). Modern cloth pattern is designed to follow the measurements of standard body template to maximize the generality. Then, by providing the measurements of a specific customer, the shape of each pattern can be adjusted respectively to fit the customer. In massive cloth production, a complex pattern resizing procedure named is developed to adapt the needs for dressing general public. Cloth pattern grading aims at resizing the cloth into several size categories based on the human anthropometric statistic data to fit the most individuals in the general public(Moore et al. 2001).

Cloth pattern in fashion industry has provided a standard to preserve a cloth design, when utilising made-to-measure tailing technique, it can be resized to fit any customer. The cloth modelling method presented in this chapter adapts this custom tailoring techniques from fashion industry into computation animation. The basic idea is that, clothes are stored in the form of cloth patterns, when modelling cloth for a character, the measurements are acquired for pattern shape adjustment. Each pattern is adjusted based on the corresponding measurements from the character. Finally, cloth pattern are assembled together following a predefined sewing rule to form a complete 3D cloth. .

However, in virtual world, the body shapes and proportions of a animation character are only limited by the imagination of artists, therefore, using size chart developed from real human body to perform proportional scaling to fit cloth to a animation character with exaggerated body shape longer applies. Therefore, for each unique character, their cloth needs to be bespoken from measurements. However, resizing cloth pattern based on measurements requires deep tailoring knowledge which few animation artist process, the difficulty of properly resizing a set of cloth pattern has become bottle neck for the application of pattern based cloth modelling method in computer animation. In order to improve the reusability of clothes in computer animation production, this chapter presents an automatic cloth pattern resizing method to adjust cloth pattern based on the measurements of a character. By automating the cloth pattern resizing process, deep knowledges in tailoring are no longer required in this method. Clothes model can be stored in an unified form (cloth pattern) into asset warehouse, with the presented automatic pattern resizing method, by providing measurements of a new character, cloth pattern can be reused to dress a different character with different body shapes and proportions. Figure 1.1 demonstrates the general workflow of the pattern based cloth modelling method presented in this chapter

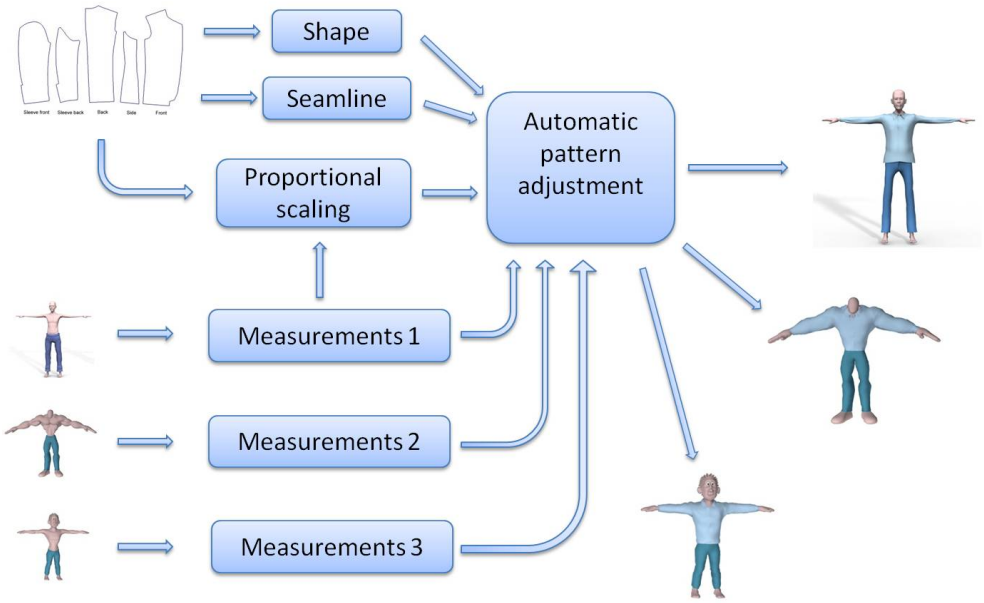

Figure 1.1: *Workflow of pattern based cloth modelling method presented in this thesis*

1.1.1 Patternmaking

Armstrong (2000) defines cloth patterns as the templates from which parts of a garment are traced onto the fabric before cutting out and assembled. When creating patterns, blocks(also as known as slopers) are created first. Blocks are two-dimensional templates that consists of basic design of the a cloth type(Howland 2008), Figure 1.2 demonstrates a block for suit. Blocks are constructed based on the measurements taken from standard body template or the wearer(Armstrong 2000). When measurements are taken from an individual, it provides a good indication of the body dimension that a cloth design is intend to fit. In made-to-measure tailoring, measurements acquisition is particularly important since it directly determines the fit of cloth. Therefore, this process is usually performed by an experienced tailor. In massive cloth production, blocks are usually created by using measurements from a size chart which is based on a particular ethnic or group of the target customers.

For example, US standard clothing size chart(ISO/TR-10652 1991) contains body proportions of the general public of the Americans and European standard clothing size chart(EN:13402 2001) contains the size data of the people who were born in Europe. Based on the blocks, cloth patterns are created by introducing pockets, style line, drapes and other adjustments. Figure 1.3 demonstrate the blocks for a woman's shirt.

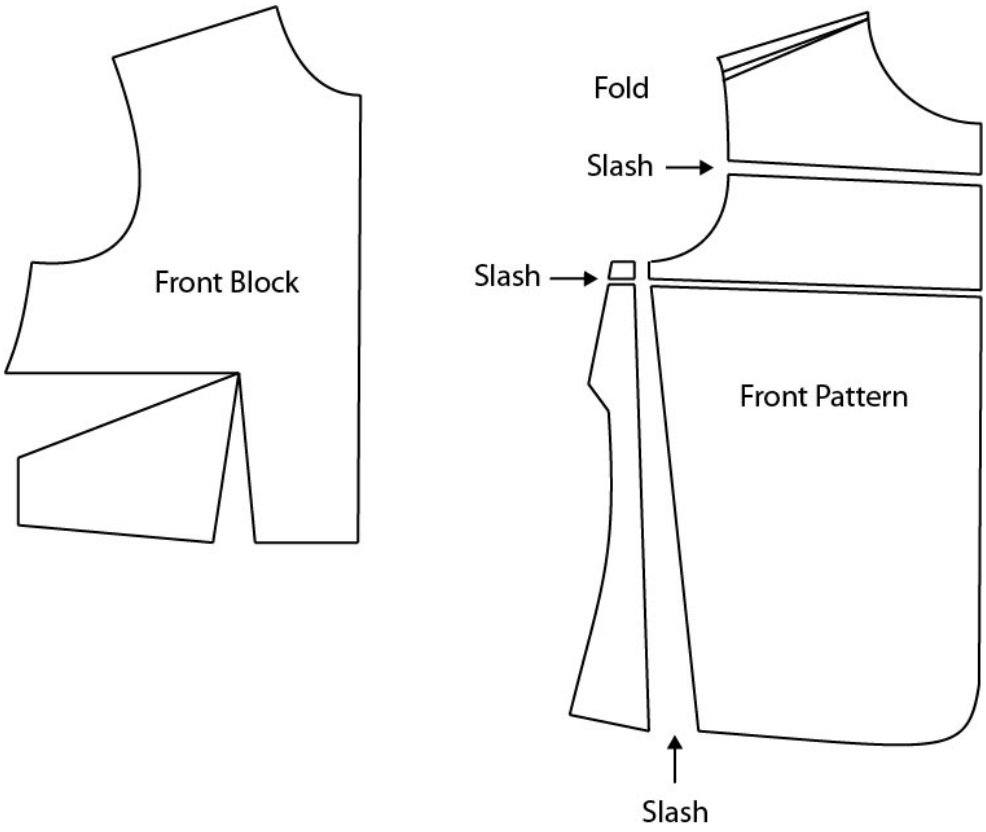

Figure 1.2: *Block(left) and a pattern(right) made from it by adding details(Howland 2008).*

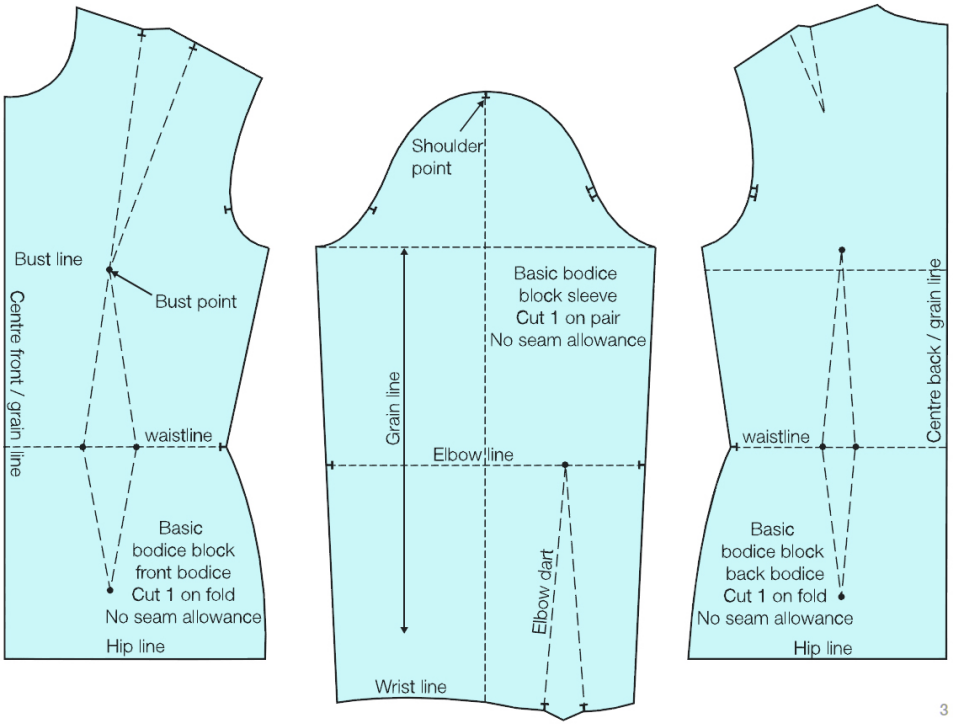

Figure 1.3: *Garment blocks for woman shirt(Rosen 2004).*

Figure 1.3 demonstrates landmarks that are associated with datum points on a character on a set of blocks. Pattern grading is the process of systematically increasing and decreasing dimensions of patterns in to a range of sizes for production. During pattern grading, patterns are scaled proportionally into different sizes with predefined intervals introduced by a size chart. This process not only retains the original design of the cloth during distribution but is also very cost effective during the process of manufacturing due to the people who fit a predefined size are normally distributed into an interval introduced by a size chart (Schofield & LaBat 2005; Moore et al. 2001).

However, in computer animation the situation is much more complexer than in reality. In reality, for each ethic or certain group of people, the similarity of body proportion exists among individuals. In virtual world, the body proportion is only limited by the imagination of artists, therefore using size

chart to perform proportional scaling to patterns to create fit cloth no longer applies to the virtual character. In most cases, clothes for animation character are bespoken to a particular character. Reusing a cloth to another character with different body shapes and proportions requires large amount of modification.

1.1.2 Pattern Resizing Criteria

During pattern grading process, all patterns are scaled proportionally to preserve the shape of pattern respectively (Moore et al. 2001). However, as aforementioned, the body proportion of the virtual character is significantly different from humans. Therefore, performing proportional scaling to patterns might leads to an undesired result.

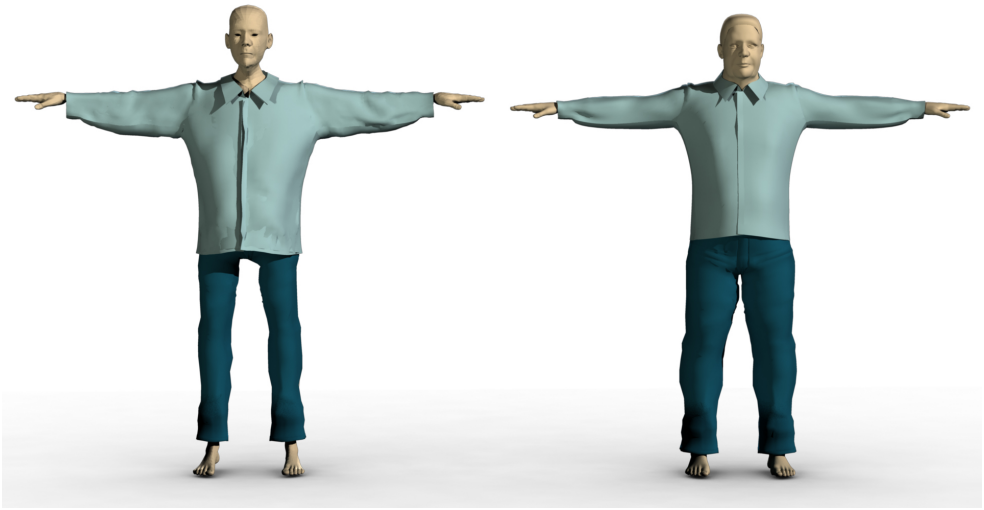

Figure 1.4: *Same shirt proportional scaled to different characters with same arm span and height using method presented by Moore et al. (2001)*

Figure 1.4 demonstrates the result of using same shirt to dress two characters with same height and arm span using proportional scaling method presented by Moore et al. (2001). Because these two characters share same

height with is the major measurements used to define the interval of a size chart, both of the character falls into the same grading intervals. Therefore, clothes are adjusted into same size. However, the character on the right is much stronger than the one on the left, the body of right side character supports cloth much better than the other. On the contrary, the skinny character has much thinner limbs which results a baggy shirt. In order to achieve better fit, during made-to-measure cloth tailoring, extra measurements such as "Arm Hole Length" or "Cuff Girth" are taken from customer in order to fine tune the cloth for better fit. With those extra measurements, different parts of a pattern are scaled in different manners. In order to generate patterns based on the measurements, a few geometric criteria need to be evaluated in order to ensure the fit of cloth as well as maintaining the design of cloth.

Character Measurements Cloth pattern represents different parts of a garment, which responsible for covering certain parts of body, therefore, all the patterns need to be deformed in a manner that each pattern matches to the measurements of its corresponding body part. Because all the patterns are derived from blocks, during pattern adjustment, for each block which a pattern is developed from, the distance between landmarks that are associated with the datum points on the character body needs to satisfy the requirement set by the measurements taken from the character. Especially when the character has an unusual body proportion, different parts of a block might be scaled in different fashions. For instance, in Figure 1.4, the arm of both two characters have the same length, however, the circumference of arm are mush different, Moreover, despite the difference between the circumference of arms, the wrist girth are the same for both characters. Therefore, in order to satisfy the all the measurement requirements, different parts of sleeve pattern need to be adjusted in different manners.

Pattern shape Cloth pattern is the most basic component of a complete gar-

ment and the shape of each pattern defines the shape of cloth. Pattern grading process is considered as the most difficult process during the tailoring which only the professionals is able to master. The overall shape of a cloth pattern can be altered significantly to accommodate the difference of body proportions between different customers. However, details of the pattern such as slops, darts and their relative location are kept (Moore et al. 2001). In computer animation, the goal of transferring cloth from one character to another is to fit a cloth onto a different character without altering the design of the cloth. However, each cloth pattern needs to be adjusted independently in order to match the measurements from character. Therefore a shape evaluation process is needed after adjusting cloth pattern in order to maintain the design of the cloth.

Seam Line In reality, cloth patterns need to be stitched together to form a complete garment. The adjacent edges between two patterns is the seam-line. A seam-line consists of many pairs of stitching points which are located on the boundary of two adjacent patterns. During the adjustment of pattern, each pattern is adjusted individually to meet the measurements. Therefore there is no guarantee that after the adjustment of the patterns, the seam-line remains consistent. Because seam-line determines the location of the patterns after it is assembled into a complete garment, without the preservation of the seam-line, original design of the cloth cannot be preserved.

1.2 Cloth Resizing Algorithm

In this section, an automatic cloth pattern adjusting method is presented. This method operates cloth size adjustment directly on 2D patterns and optimizes each pattern to ensure all the patterns satisfy the criteria introduced in previ-

ous section.

Two inputs are required for this method. The first input is a group of patterns representing a cloth design. Within each pattern, the landmarks that associated with the datum point on character body are defined. Pattern landmarks are key points on a pattern inherited from block. They are defined on the cloth block by patternmaker to associate body measurements with the length on a block (Rosen 2004). In real world, pattern landmarks are used as a guide line for pattern resizing where tailor adjusts distance between pattern landmarks to meet body measurements. Moreover, seam-lines are presented along with the pattern. Figure 1.5 demonstrates an example of input patterns.

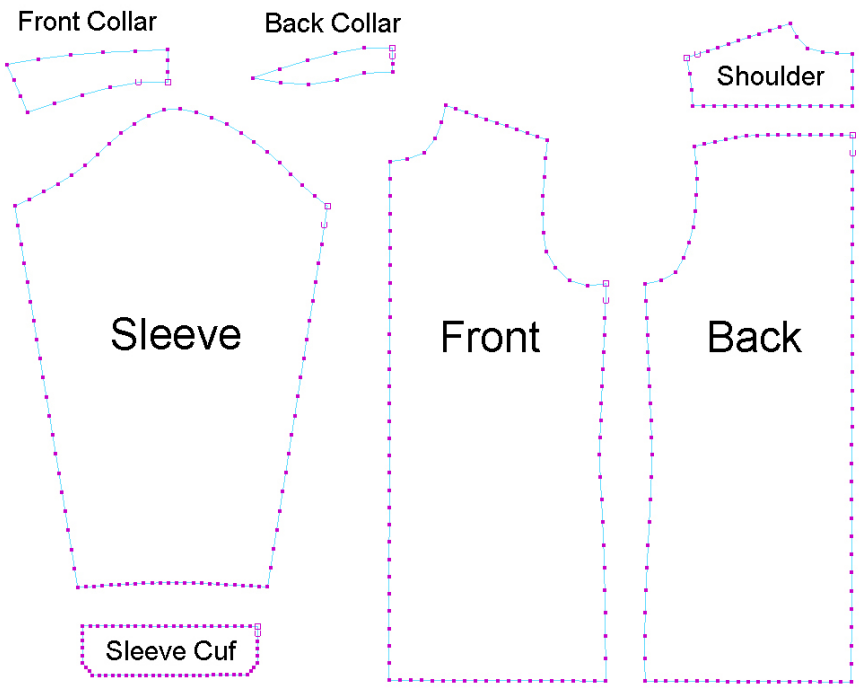

Figure 1.5: *An example of a cloth pattern*

The second input is the 3D character model which is modelled in a "T-pose". Because this method involves physical simulation for pattern assembling process, the contact between limbs will leads to the penetration between cloth mesh and character skin. Moreover, the measurements of the character

are also provided by using the method from previous chapter.

In order to properly determine the size of patterns, block for each pattern is generated at first. In fashion industry, cloth pattern is created from an unique block by adding details. Therefore, in this method, a block is refers to a 2D bounding box of each pattern and all the landmarks that are associated with datum points on character body are defined on the blocks. Moreover, because a block and a pattern have a strictly one to one association which means for any pattern, it is generated from only one unique block. Also, because a block contains much less details than the pattern, performing pattern adjusting algorithm on blocks can saves lots of computational resources.

When constructing a block from a pattern, firstly, the "critical points" on a pattern are selected. The "critical points" are defined by points which are the most capable of representing the geometrical features of a pattern. For example, the sharp turning points on the contour of the pattern or extreme points are considered as "critical points". Figure 1.6 illustrates "critical points" on the sleeve pattern. Finally, a bounding box is created for each pattern, within the bounding box, for every "critical point", two orthogonal line are created to form a grid. This is demonstrated in Figure 1.6.

Next, based on the gird, a nurbs plane is constructed and control points of the nurbs plane are intersection points on the grid(this is demonstrated by the green cross in Figure 1.6). All points on each pattern can be represented by a parametric coordinate on the nurbs plane. For every pattern, a unique nurbs plane is created and it is considered as the block of the given pattern.

Then, the proportional scaling used in traditional cloth grading method is performed onto each pattern. In general, the measurements that are associated with the length of limbs are used as the reference for scaling such as "Back Length" for torso patterns and "Arm Length" for sleeve patterns. Figure 1.7 demonstrates the result of performing proportional scaling to sleeve block based on the "Arm Length" measurement.

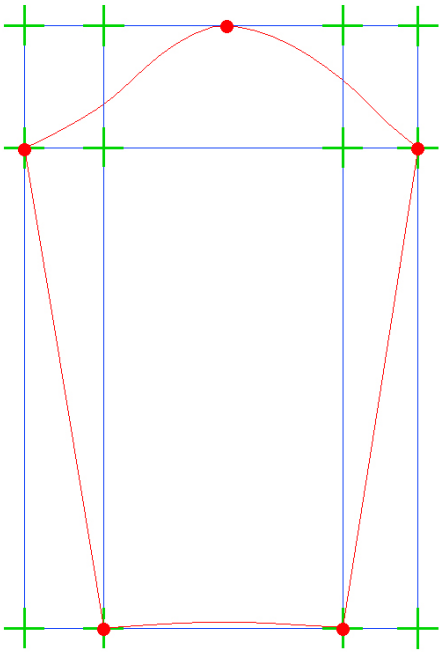

Figure 1.6: *Block generated for sleeve pattern(red closed curve). The red points are "critical points" on sleeve pattern.*

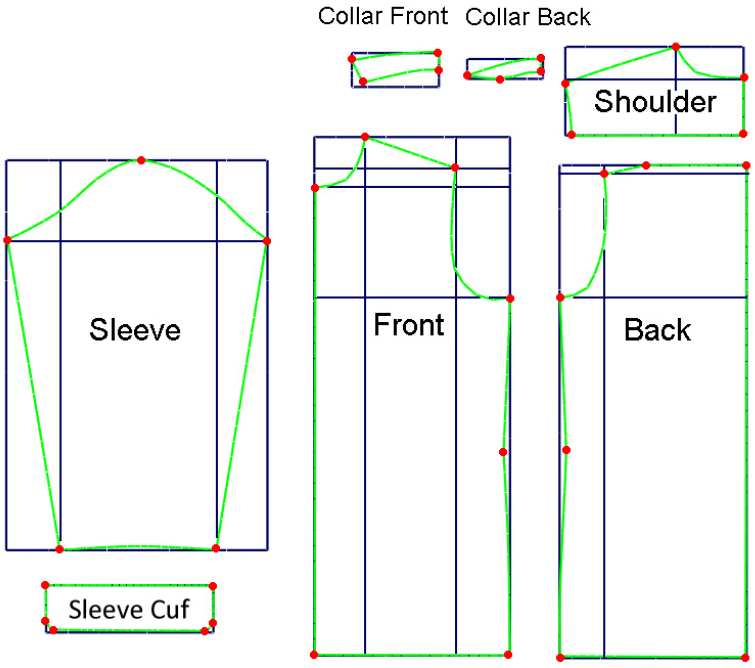

Figure 1.8: *Blocks after proportional scaling is performed.*

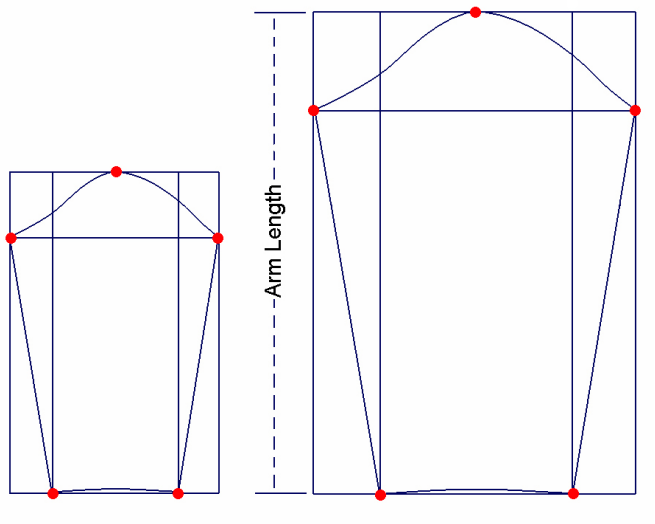

Figure 1.7: *Block before(left) and after(right) the proportional scaling. The height of the sleeve pattern after scaling has reached the measurement "Arm Length"*

Performing proportional scaling to cloth patterns has a significant drawback that needs to be improved. This drawback is demonstrated in Figure 1.9.

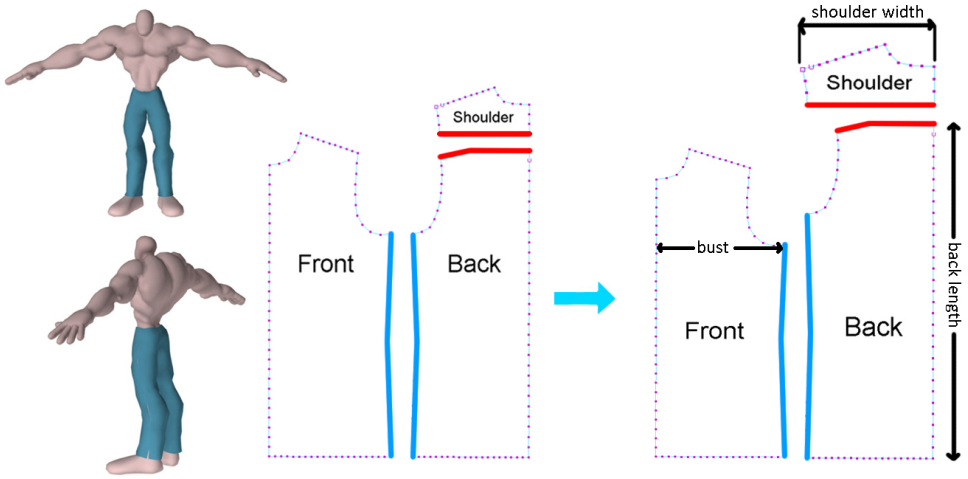

Figure 1.9: *Inconsistency of seam-line after scaled patterns proportionally*

Where 'Front", "Back" and "Shoulder" are three patterns of a cloth for Character A. Patterns are scaled according to their major measurement, For example, pattern "Front" is proportionally scaled based on measurement "bust", pattern "Back" is proportionally scaled based on measurement "Back

length". However, when pattern was initially designed, each pair of seam-lines between two adjacent patterns usually have same length and interval for each stitch, this is shown in Figure 1.9 as a pair of blue curves indicates a pare of seam-lines between pattern "Front" and "Back", a pair of red curves indicates seam-lines between pattern "Shoulder" and "Back". However, after proportional scaling is performed, the length of seam-line is no longer the same which further leads to undesired wrinkles around seam-line during pattern assembling process. At this stage, each pattern has good fit on measurements criteria but worsen on the seam-line criteria. If patterns are adjusted to meet the seam-line criteria, the fit on measurements will be break. Moreover, if patterns are firstly proportional resized based on measurements and then adjusted each pair of seam-line to satisfy seam-line criteria, the shape of patter will be changed. Actually, in many cases, these conditions may not concord with each other and in some cases even conflict to each other. Therefore, it is very difficult to adjust cloth patterns to fit a character based on one criteria.

1.2.1 Genetic Algorithm

Modelling a cloth from cloth patterns for a character requires each pattern to be adjusted individually. During the pattern adjustment, three criteria need to be satisfied in order to fit the cloth to the character as well as preserve the design and integrity of cloth. The goal of cloth fitting process is to generate a set of cloth patterns that constitute a complete garment which fits character while having the least distortion compare to original design of this garment. In this thesis, this process is considered as a multiple objective optimization process.

Optimization problem is a very hot research area that has been studied intensively for decades and many techniques have been developed to tackle

such a problem. In the case of cloth pattern adjustment, since a complete cloth is comprised by a set of patterns, each pattern is formed by a set of points (sampling points on the contour of a pattern). The parameters of this optimization problem are the coordination of these sampling points.

For current gradient based optimization methods, handling large amount of parameters requires a lot of computational resources. Among many optimization methods, genetic algorithm dwarfs others by its simplicity and efficiency. Genetic algorithm works on chromosomes, which are encoded parameters of solutions. Because this encoding mechanism largely reduces the number of parameters used for optimization, much less computation is required by genetic algorithm. Moreover, unlike traditional optimization method, which searches cost surface from a single point, genetic algorithm searches cost surface in a parallel manner, it is not only able to scan a large number of potential solutions very quickly, but also able to avoid local optimal solution effectively due to its parallel searching method. In general, the advantages of the genetic algorithm over other optimization method are,

1. Genetic algorithm searches through a wide range of the cost surface simultaneously. Therefore it is able to deal with very complex cost surfaces and avoid local minimum.

2. Genetic algorithm works on the chromosome instead of real parameters, therefore it is able to handle large number of parameters.

3. The initial proposals do not effect the final solutions as bad solutions are discarded by selection at every evolution. Therefore, the genetic algorithm is not sensitive to the initial seeds.

In the thesis, three criteria are used for evaluating fit and style preservation during pattern adjustment process. However, these criteria are often conflict to each other. The optimization algorithm needs to be able to handle multiple objectives. Moreover, cloth patterns are represented by a set of

nurbs curves which are controlled by a set of "Critical Points". Each "Critical Point" is a parameter to the optimization, large amount of parameters need to be handled at same time. Furthermore, the initial state of pattern is proportionally scaled based on each pattern's major measurements, to some non-exaggerated human like character, it will provides a good approximation that is close to the final fit cloth. However, for some largely exaggerated character such as character A in Figure 1.9, proportional scaling will worsen other criteria. Therefore, cloth pattern optimization should not be sensitive to its initial seeds. Genetic Algorithm has advantages these three aspects, therefore, this thesis utilises genetic algorithm for cloth adjusting process. Because cloth patterns are adjusted based on the measurements of the character, adjusting cloth patterns using genetic algorithm has three extra benefits. Firstly, once the parameters are set, genetic algorithm evolve the initial seeds towards best solution automatically, therefore, given measurements of any character, this method can generates cloth patterns that fit the character automatically. By automating the pattern adjustment process using genetic algorithm, traditional tailoring knowledge is no longer needed for this process, the duplication of effort required by traditional pattern based cloth modelling method can be eliminated and the efficiency for modelling cloth for different characters with different body shapes and proportion can be largely improved. Secondly, this method empowers the creativity of animation artists and amplifies their productivity by allowing them to use a large amount of existing cloth patterns in the fashion industry to create various clothes that fit different characters. Thirdly, because the 3D cloth is generated based on the adjusted patterns, the work flow of modelling cloth for the character is one direction. There is no turning back for extracting patterns from 3D cloth which is required by current cloth modelling method. Therefore, the shape distortion that is introduced by 3D surface flattening process can be avoid.

Genetic algorithm(GA) is an optimization method which is based on the principles of natural selection (Haupt & Haupt 2004; Deb 2001). During

natural selection, effected by environment, the biological traits of organism become more or less common in a population through generations of reproduction. A genetic algorithm allows a population that consists of many individuals to evolve under certain rules and to a state that minimizes the cost function (Holland 1992).

A genetic algorithm starts from a group of randomly generated solutions called "initial population". Within the initial population, each individual is a set of variables that represents a solution of the problem. By evaluating every individual using cost function, a ranking is assigned to the individual in terms of performance of cost function. Then, a selection method is applied to select a group of individuals to perform crossover and mutation operation. Finally, the convergence of current generation is evaluated. If current generation does not reach the minimum on the cost surface, this process loops back to step one, the algorithm is executed iteratively till the minimum cost is reached, this procedure is demonstrated in Figure 1.10

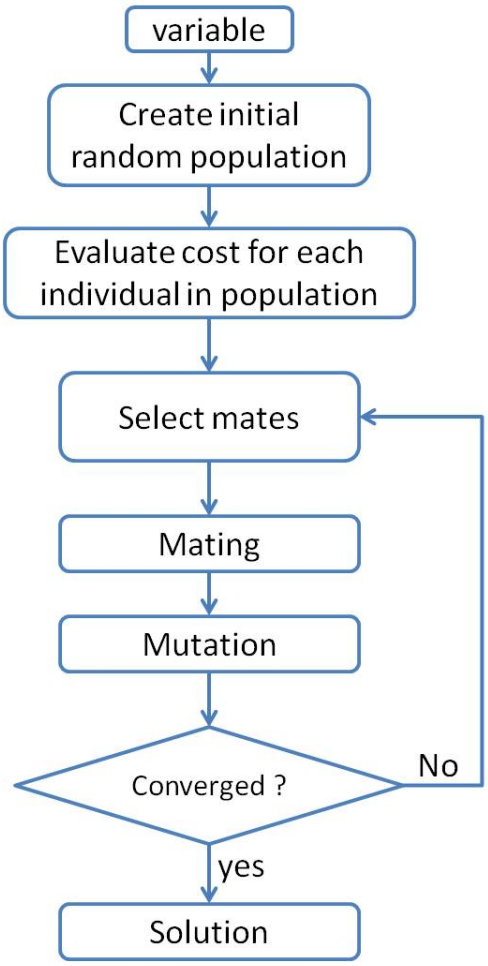

Figure 1.10: *Workflow for single objective genetic algorithm*

In genetic algorithm, "fitness" refers to the cost of an individual in terms of its performance to the cost function, higher the fitness, lower the cost. In single objective optimization, only one individual who has the highest fitness is selected as the final solution of the problem. However, in may applications, more than one objective are needed for describing the problem. Moreover, among these objectives, conflicts often occurs. For example, adjusting a pattern for a particular measurement usually leads to worsen the consistency of seam-lines. In this case, it is impossible to locate a single best solution that is optimum with respect to all objectives. This kind of optimization is called multi-objective optimization, the purpose of multi-objective optimization is to find as many good solutions as possible. The solution set results from a

multi-objective optimization is called Pareto-optimal solutions (Pareto 1906). However in may applications, only one solution is needed, therefore, usually, after Pareto-optimal solutions has been formed, a higher-level objective is used to select one solution out of the Pareto-optimal solutions as the final solution for the problem.

In the following section, a multi-objective genetic algorithm for pattern adjustment is presented in detail. This algorithm uses multi-objective optimization to find a combination of patterns that has best fits on measurement criteria, seam-line criteria and pattern shape criteria.

1.2.2 Definition of Population

Within a population, individual is the basic element represents one solution to the problem (Haupt & Haupt 2004). In our case, each pattern is described by a set of points located on the contour of a pattern $Pattern_i = [p_0, p_1, ..., p_n]$. A block is a nurbs plane that the pattern is inscribed to and it is defined by a set of control points, noted as $b_i = [cp_{i1}, ...cp_{im}]$. After a block is created based on a pattern, Points on the pattern can be interpolated by parametric coordinate on the block plane. Each pattern has its unique block, therefore, for each pattern, it can be represented uniquely by control points of the corresponding block plane. Note that $m << n$, which means the number of control points is much less than the actual number of points on the pattern, particularly, in experiments, m ranging from 4 to 30 and n ranging from hundreds to a thousand. Consequently, a pattern can be represented by control points of the block effectively.

$$\begin{cases} b_1 = & cp_{1_1} & cp_{1_2} & \cdots & cp_{1_m} \\ b_i = & cp_{i_1} & cp_{i_2} & \cdots & cp_{i_{m'}} \\ & & \vdots & & \\ b_n = & cp_{n_1} & cp_{n_2} & \cdots & cp_{n_{m''}} \end{cases} \tag{1.1}$$

Table 1.1: *The structure of gene*

Table 1.1 illustrates the structure of chromosomes. Where cp_i denotes a control point on block b_i, an individual is comprised by a group of blocks $ind = b_1, ..., b_n$ which representing a complete cloth. In general, for each individual, a gene represents a control point, a chromosome represents a block, and an individual is constituted by a group of chromosomes (blocks).

The initial population is the first sample over the cost surface. Because in the case of pattern adjustment, the measurements of character can be largely different form character to character, the boundary of the cost surface can be difficult to determine. Although for GA, the initial population will not affects the formation of Pareto-front, setting the initial population close to the final objective can largely reduce the number of generations required for generating the Pareto-front. Therefore, before performing genetic algorithm, traditional proportional scaling method is used to initialize the first generation of population. For every input pattern, the measurements taken from the subject that associated with this pattern is stored into an array, denoted as $M_i = [m_{i_1}, ..., m_{i_n}]$. For each stored measurement, it contains datum points that this measurement is taken from and every datum point on the subject has an unique corresponding landmark on the pattern. This is illustrated in Figure 1.11

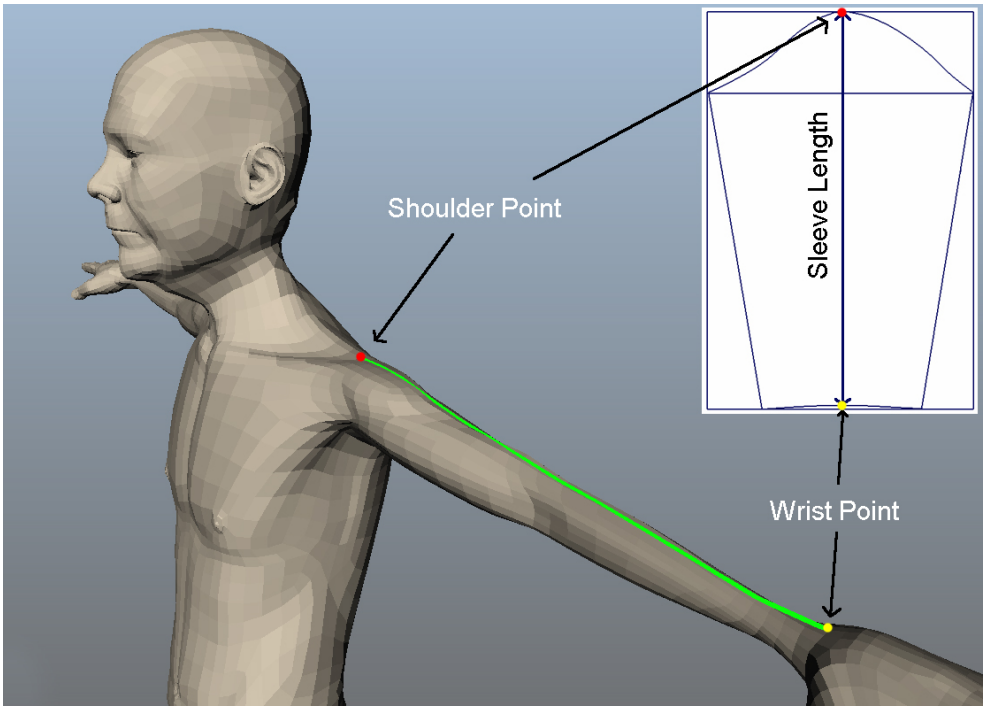

Figure 1.11: *Association between body datum point and patter landmarks. The green line is the geodesic from shoulder point(red point) to wrist point(yellow point). The red and yellow point on the sleeve pattern is the pattern landmarks that associated with the datum points on the body of character. When sleeve is well fitted, the "Sleeve length" should match the length of this geodesic.*

The scaling factor can be calculated by Algorithm 1.

Algorithm 1 Initial Pattern

procedure PROPORTIONALLY SCALE PATTERN(Pattern P, Measurements M)
  for measurement $m_i \in M$ **do**
    p_s and p_e is two datum points of m_i
    l_s and l_e is two landmarks on a pattern that $l_s \sim p_s$ and $l_e \sim p_e$
    $d \leftarrow$ distance between l_s and l_e
    $Sr \leftarrow m_i/d$(Sr denotes scaling factor)
  end for
  Select largest Sc from S as the scaling factor for P
end procedure

By using Sc as the scaling factor, the input patterns are proportionally scaled. Although the proportional scaling method cannot produce a well fit

cloth for the character, it is able to resize the patterns to a certain degree so that the cloth roughly fits the character. Then, the proportional scaled patterns are used as the seed for generating the initial population. In the next step, in order to create individuals that spread over the cost surface, a mutation operation are carried out on each pattern to vary its shape. This operation will be explained in detail in "Crossover and Mutation" section.

In order to improve the efficiency of the evolution, the individual can age during the evolution. In nature, the size of the population changes over generations because individuals who carries a fit gene has better crossover opportunity than those who carries less fit gene. Therefore, naturally, a fit gene is able to live longer during evolution than a less fit genes. When a gene dies, the individual who carries this gene also dies. Consequently, the number of the individuals in a generation is determined by the death of the less fit gene.

$$life_i = \begin{cases} life_{min} + \eta \frac{cost_{max} - cost_i}{cost_{max} - cost_e} & if \quad cost_i \geq cost_e \\ \\ \frac{1}{2}(life_{min} + life_{max}) + \eta \frac{cost_e - cost_i}{cost_e - cost_{min}} & if \quad cost_i < cost_e \end{cases} \tag{1.2}$$

Equation 1.2 is a bilinear method that introduced by Michalewicz (1996) for calculating life span of a gene. Where, $life_{min}$ and $life_{max}$ denote the shortest and longest life span for a gene, $cost_{max}$ and $cost_{min}$ denotes the highest and lowest cost of the current individual that carries this gene. $cost_e$ denotes the expected value of an individual and $cost_i$ is the actual cost value for current individual. When a gene reached its life span, it dies before it is evaluated by the cost function. During the evaluation, if lots of individuals died in one generation, inadequate number of individuals may results the limited searching range across the cost surface which further leads to premature convergence to local minima. Therefore, dead individuals are replaced by

newly mutated individuals to maintain the coverage over the cost surface.

In the presented algorithm, the lifespan of a gene is assigned every time when evaluation is performed to the newly generated individual. For every generation, the $cost_{max}$ and $cost_{min}$ of a gene are determined by the worst and the best gene of a pattern within the previous generation in terms of three evaluation functions accordingly. The $cost_e$ is determined by the average cost of all the contour points of a pattern that are derived from the block in the previous generation. This ensures a gene in the current generation can not lives longer than the previous better gene. When selection is performed, any individual who carries a gene that has reached its lifespan is discarded.

1.2.3 Crossover and Mutation

In nature, crossover is the source power for evolution. Because crossover is the only way for creating new individual that carries genes from their parents. The chromosome of offspring is the recombinations of the genes from their parents. Mutation is another method for alternating genes, it is able to introduce the new gene into the chromosomes of an individual without the need of pairing. Both method enables the genetic algorithm to explore new area on the cost surface. In general, mutation are often used to provide exploration and crossover are mostly used to lead the population to converge into current good solutions.

Crossover is an exploitations to a certain area on the cost surface. It requires two individuals to be involved to the process and normally generates two offspring. In order to perform crossover, two parents need to be selected from the population first, the difference between two parents needs be large enough to create an effective offspring, because two similar parent will result a very similar offspring and further leads to an over dense sampling in a small region near the parents, which will cause premature convergence. In

each generation, the individuals are sorted based on domination. Therefore, when selecting parents from population for crossover, an index interval is used for maintaining the difference between two selected parents. After both parents are selected, then recombination of the chromosomes takes place, this is demonstrated below.

$$Father = [cf_0, cf_1, cf_2, cf_3, ...cf_n]$$

$$Mother = [cm_0, cm_1, cm_2, cm_3, ...cm_n]$$

where cf_i is the chromosomes from father side and cm_i is the chromosomes from mother side. Normally, crossover generates two offspring, in which both child carries both part of chromosomes from their parents.

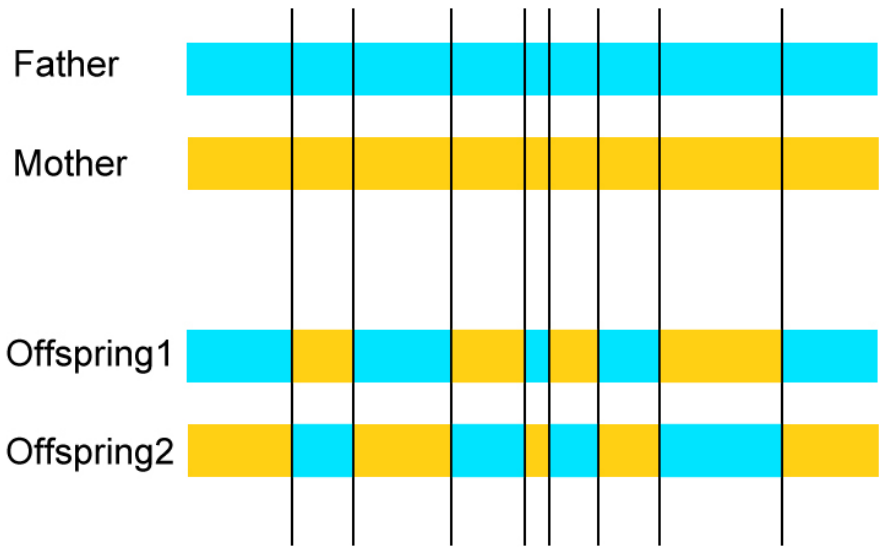

Figure 1.12: *Crossover of parents*

Figure 1.12 demonstrates the uniform crossover method introduced by Spears & Anand (1991); Gwiazda (2006); Ghosh & Tsutsui (2003). This method uses a fixed ratio exchange rate between two parents to generate off-

spring. The black line indicate the exchange point in chromosome that are randomly selected by a random number generator.

For pattern adjustment, each individual is constituted by a group of different patterns, when two individual mates, crossover operation only performs on same pattern in both side of parents. For example, crossover operation can only be performed on two parent "Front" pattern to generate two offspring "Front" patterns. The crossover operation is summarized in Algorithm 2.

Algorithm 2 Crossover

```
1: procedure CROSSOVER(Two individuals ind_1, ind_2, crossover rate cxpb)
2:     if cxpb > random() then
3:         for same block(b_1,b_2) in ind_1 and ind_2 do
4:             N ← number of control point in current block
5:             for i = 0; i < N; i++ do
6:                 if rand() < 0.5 then
7:                     offspring_{1_i} ← b_{1_i}
8:                     offspring_{2_i} ← b_{2_i}
9:                 else
10:                    offspring_{1_i} ← b_{2_i}
11:                    offspring_{2_i} ← b_{1_i}
12:                end if
13:            end for
14:        end for
15:    end if
16: end procedure
```

Because crossover only operates on the existing gene pool that is comprised by genes from two parents. It samples the area near their parents much more denser than mutation does. A higher crossover rate will increase the speed of convergence. In real world, a species will extinct without introducing new gene because the limited gene pool will be exhausted during the new born of the individuals. In genetic algorithm, this phenomenon appears as the over dense sampling into a local area around the initial population. This often results premature termination of the evolution.

Mutation is a process that new genes can be introduced into the gene pool so that the diversity of the gene pool is enriched and gene degradation can be avoided. In genetic algorithm, mutation can explore new area of the cost surface much more efficient than crossover. Haupt & Haupt (2004) indicates that two issues should be taken into the consideration when performing mutation operation to an individual, the type of the mutation and the rate of the mutation. Grefenstette (1986); Srinivas & Patnaik (1994) point out that the choice of the mutation rate is heavily problem specified. For different problems, best mutation rate varies significantly, it needs to be set empirically for each particular problem.

When performing genetic algorithm, a high mutation rate enlarges the searching range on the cost surface, whilst prevents the population to converge to a specific point. In the meantime, a very low rate of mutation will leads to a premature convergence very easily. According to the work of Yaman & Yilmaz (2010), in different stages of the the evolution, the genetic algorithm usually requires different exploration-exploitation ability. In this thesis, the non-uniform mutation, which presented by Michalewicz (1996), is applied, in which the possible impact of mutation to an individual decreased when generation evolves.

When initializing the initial population from seeds, a very wide distribution over the sampling cost surface is preferred, therefore, large mutation rate is set. Before evolution comes to the end, exploitations is much preferred than exploration so that the convergence can be ensured. At this stage, a high mutation rate will disturb the convergence of the algorithm. In the implementation of this algorithm, the mutation rate starts from 0.6 and decreased linearly by 0.002 for every generation that has evolved. Assume that Gen_{max} is the predefined maximum number of generations of evolution. Then, for each individual, the randomly chosen chromosome cp_i is replaces by one of

the two values demonstrated in Equation 1.3,

$$cp_i = \begin{cases} cp_i + \Delta(gen, t) & if \quad \gamma \geq 0.5 \\ \\ cp_i - \Delta(gen, t) & if \quad \gamma < 0.5 \end{cases} \tag{1.3}$$

where, gen denotes the index of the current generation, γ is a normally distributed random number from 0 to 1, $\Delta(gen, t)$ is a random variable that mutates cp_i in range $[0, t]$. The value of $\Delta(gen, t)$ is determined by the index of current generation gen by Equation 1.4, introduced by Michalewicz (1996),

$$\Delta(gen, t) = t * \left(1 - \lambda^{\left(1 - \frac{gen}{Gen_{max}}\right)^r}\right) \tag{1.4}$$

where, λ is an normally distributed random value from 0 to 1, Gen_{max} is the maximum number of generations of the evolution. The exponential factor r controls the influence of gen on the distribution of $\Delta(gen, t)$ within its range. In which this operation becomes an uniform mutation if $r = 0$. The mutation process is introduced in more detail by Algorithm 3

For each mutation, only one orthogonal direction is modified in either row wise or column wise. For example, this mutation operator performs on U direction only for a selected row of the control points or on V direction only for a selected column of the control points. The choice between two directions are randomly selected for each block. For each row of the block, The mutated U value of each control point is stored into an array denoted as U_{row}. The original U value of every control point is also stored into an array denoted as U'_{row}. For each pattern, its block is created as a rectangle nurbs plane that the pattern is inscribed to. Therefore the U value of each row or column of a block is in the same incremental or decremental order. this is demonstrated in Figure 1.13.

Algorithm 3 Mutation

procedure MUTATION(Individuals ind, Generation index gen, Mutation rate $mutpb$)
  if $mutpb > random()$ **then**
    for $block_i \in ind$ **do**
      $dpb \leftarrow random()$
      $U \leftarrow$ the number of control point on U direction
      $V \leftarrow$ the number of control point on V direction
      if $dpb < 0.5$ **then** ▷ Variate cp_i in U direction
        for Each row of cp_j on $block_i$ **do**
          Append u value of cp_j into array $Variables$
        end for
        Apply Equation 1.3 to each element in $Variables$ and evaluate the newly formed $Variables'$
        if $Variables'$ is invalid **then**
          Reapply Equation 1.3 to each element in $Variables$ and evaluate the newly formed $Variables'$ till $Variables'$ is valid
        end if
        for Each row of cp_j on $block_i$ **do**
          Assign back u value of cp_j from corresponding element in $Variables$
        end for
      else ▷ Variate cp_j in V direction
        for Each column of cp_j on $block_i$ **do**
          Append v value of cp_j into array $Variables$
        end for
        Apply Equation 1.3 to each element in $Variables$ and evaluate the newly formed $Variables'$
        if $Variables'$ is invalid **then**
          Reapply Equation 1.3 to each element in $Variables$ and evaluate the newly formed $Variables'$ till $Variables'$ is valid
        end if
        for Each column of cp on $block_i$ **do**
          Assign v value of cp_j with corresponding element in $Variables$
        end for
      end if
    end for
  end if
end procedure

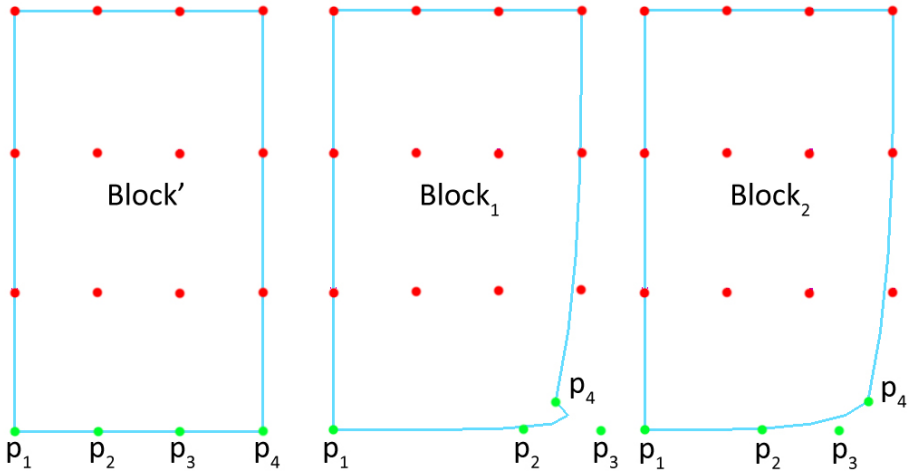

Figure 1.13: *$Block'$ is at its initial status, $Block_1$ and $Block_2$ are the results from mutation operation applied on the bottom row of control points(green points).*

When a block is at its initial status, such as $Block'$ indicated in Figure 1.13, the order of the control points in the selected row is denoted by $R' = [p_1, p_2, p_3, p_4]$. $Block_1$ and $Block_2$ are two possible results generated by mutation operation on $Block'$. Sorted by the U coordinate, the order of the control points in the corresponding row can be written as $R_1 = [p_1, p_2, p_4, p_3]$ and $R_2 = [p_1, p_2, p_3, p_4]$. In order to avoid face overlapping on a cloth pattern during mutation, the order of control points in any row or column need to be maintained. Since $R' \neq R_1$ and $R' = R_2$, $Block_1$ is marked as an invalid gene for current individual. In order to model cloth, each pattern need to be triangulated first to from the polygon from the outline of the pattern. In the case of $Block_1$, because the order in R_1 is changed, it will cause faces overlap to each other during the triangulation of the pattern. When a pattern is invalided, mutation needs to be reapplied till $R' = R_1$. In this case, $Block_2$ is a valid gene of the current individual.

1.2.4 Evaluation and Selection

In genetic algorithm, fitness function(cost function) describes the objective of the problem. Given a solution, the fitness function is used to measure how far the current solution is to the desired goal of convergence. When a population evolves into a new generation, the selection operator deletes the n worst individuals and breeds n new individuals from the best individuals. In order to perform such a process, each individual need to be awarded a rank to indicate how close it comes to the ideal solution, ranking is generated by applying fitness function to each solution. In this algorithm, objectives for evolution is to adjust size and shape of the each pattern so that each pattern fulfil the requirement of the measurement criteria . Also, among patterns, the topology of the seam-line where two patterns are joint together remains. Most importantly the consistency of cloth design must be retained after resizing. Therefore, three fitness functions are developed for evaluating the fitness for each individual respectively.

Measurements evaluation

Given a block, landmarks are associated with datum points on the body of character. To create a block which follows the correct measurements, the distance between two landmarks of the block and their associated measurements should be equal. Therefore, the fitness function for the measurement objective can be described as Equation 1.5.

$$Error_m = \frac{\sum_i \|M_n - Dist(l_{i_s}, l_{i_e})\|}{n} \tag{1.5}$$

Where M_i denotes the ith measurements associated with current block. l_{i_s} and l_{i_e} are two landmarks on the block that are associated with M_i. n denotes the number of the measurements that determines the size of the current

block b_m. Because cloth pattern is a 2D surface, euclidean distance between two landmarks is used to compare with the measurement resulted by measurement method presented in previous chapter.

Seam-lines evaluation

A cloth cannot be made into the right design without correct sewing. It is the most important method that joints two piece of textile together. Digest (2010) defines a stitch as a single loop of thread that on the textile and sewing is the craft of fastening objects using stitches.

In this thesis, a stitch refers to a constrain that attaches vertices on two patterns together and a seam-line is comprised by a group of stitches that from one end of the seam-line to another. Unless the design requires, normally, for a pair of seam-lines, the structure of both seam-lines are identical in order to form a flat and smooth pattern to pattern transition. Therefore, in this algorithm, two criteria is evaluated for seam-line evaluation, which are angle difference for each pair of point and length of corresponding edge on either side of a seam-line.

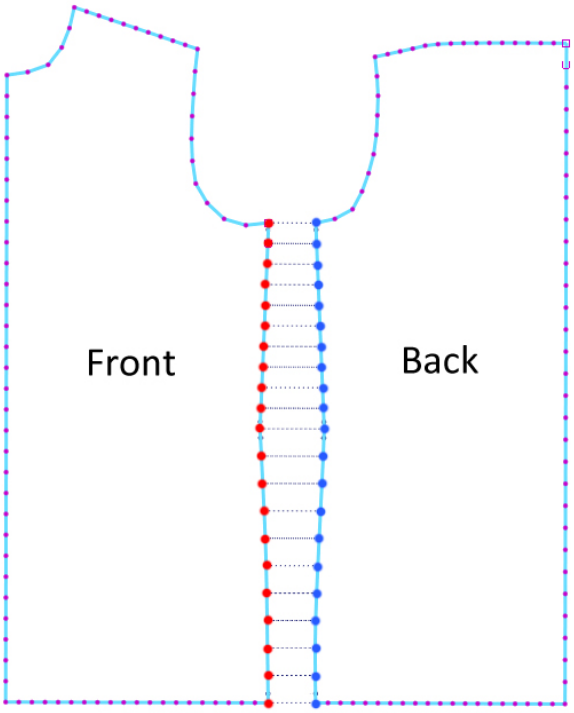

Figure 1.14: *Seam-lines between "Front" pattern and "Back" pattern. The red points indicate the seam-line on the front pattern and blue points are the seam-line on the back patter.*

Let $sline_f$ denotes a seam-line on "Front" pattern that depicted by the red points and $sline_b$ denotes a seam-line on "Back" pattern that depicted by the blue points in Figure 1.14. For each pair of points that connected by a constrain have the same index in its corresponding seam-line. Given a point pf_i on $sline_f$, the greatest included angle θ_f between $\overrightarrow{pf_{i-1}pf_i}$ and $\overrightarrow{pf_{i+1}pf_i}$ is recorded. Next, the point at the same location on $sline_b$ is also recorded as θ_b. The total angle difference of a seam-lines on a pattern comparing to the standard input pattern is used to evaluate the angle criteria and it can be calculated by Equation 1.6.

$$Error_{angle} = \sum_{i=0}^{n} (\left\| \theta_{i_f} - \theta_{i_b} \right\|) \tag{1.6}$$

Where n denotes the number of point pairs in a seam-line. The other

criteria is the edge length in seam-line. For a pair of seam-lines with same arc length, different point distribution will results an undesired tension along the seam-line in which causes wrinkle to occur around the seam-line. Within every seam-line, all points are stored in an predefined order, thus in order to ensure the consistency of the distribution of points in a seam-line pairs, edge that connects two successive points in a seam-line needs to be identical to the one in the other seam-line. Therefore, the edge length criteria can be written as Equation 1.7.

$$Error_{edge} = \sum_{i=0}^{n-1} (\left\| l_{i_f} - l_{i_b} \right\|) \tag{1.7}$$

Where l_{i_f} denotes the length of edge $\overline{p_i p_{i+1}}$ in a seam-line and l_{i_b} denotes the length of edge $\overline{p'_i p'_{i+1}}$ in another seam-line. n denotes the number of points in a seam-line. Therefore, the seam-line evaluation outputs the cost as the sum of angle criteria and the edge length criteria as Equation 1.8.

$$Error_s = \frac{Error_{angle} + Error_{edge}}{n} \tag{1.8}$$

where n denote the number of point in current seam-line.

Shape evaluation

Armstrong (2000) points out a cloth design can be translated into a set of pattern with predefined shape and sewing sequence. The shape of each pattern is the critical factor that determines the shape of the cloth after patterns are assembled. The consistency of the shape for each pattern needs to be kept through out the evolution in order to ensure the cloth that fits to a character remains the same. The shape of a geometry can be described via many methods. In this thesis, the definition introduced by Kendall (1977) is used to define a pattern shape. Kendall (1977) defines shape as “all the geomet-

rical information that remains when location, scale and rotational effects are filtered out from an object". In this algorithm, if the included angle on each corresponding crucial point is identical between two patterns, the shape of these two pattern is considered same.

In this algorithm, during the evolution, the changes of all the inner angle on every point of the contour of a block is measured as the cost of shape evaluation. Figure 1.15 demonstrates variations for each pattern, because when crossover and mutation is performed on a pattern, the location of each point on pattern contour is changed, which the shape of a pattern is changed consequently.

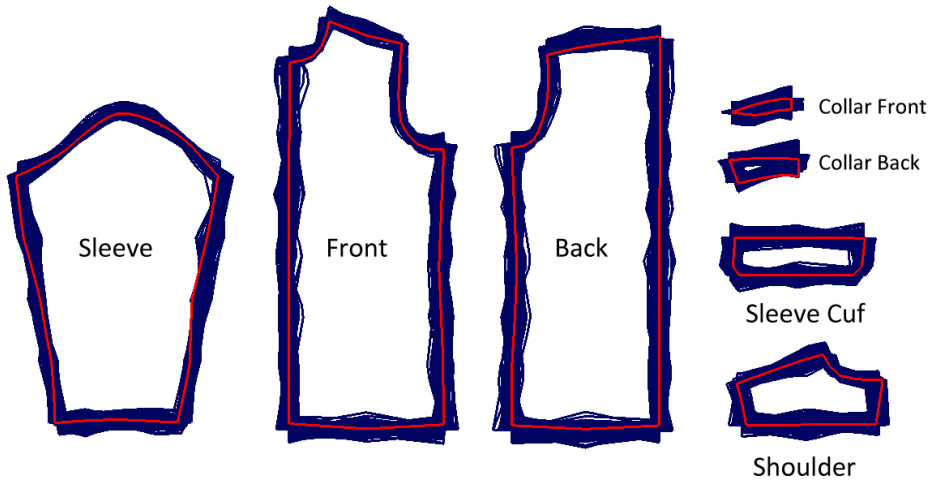

Figure 1.15: *Shape evaluation for the first generation patterns, Blue lines indicate the variations of the patterns, and the patterns that are indicated by the red line has the best shape preservation.*

The pattern adjustment algorithm presented in this chapter solves a multi-objective optimization problem in which objectives conflict to each other. This optimization can only results a set of solutions called "Pareto front" instead of a single best solution. Within "Pareto front", each solution is not dominated (explained in following section) by the rest of the solutions within this set (Michalewicz 1996). However, lots of real problems in the

real world are multi-objective problems that require a single solution. The common method for finding a single solution for a multi-objective optimization problem is that, multiple fitness function are used for the construction of "Pareto front", then a higher-level objective is used to find the "best" solution among "Pareto front".

When adjusting a cloth, given a character, only one size of the cloth is needed for dressing the character. In order to select one solution from "Pareto front", only one or several non-conflict criteria can be used to select the "best" solution from "Pareto front". In this thesis, Preserving cloth style whilst fitting it to a different character is the main goal. therefore, the shape similarity between original pattern and resized pattern is used as the higher-level objective to select one solution from "Pareto front". When evolution limit is reached, all solutions in "Pareto front" are considered as "good" solutions to the character dressing problem. In other words, all patterns in "Pareto front" has achieved good fit and all seam-lines are consistent. Therefore, at this stage, the solution which has the least shape difference comparing to the original cloth patterns is selected as the final solution of dressing character.

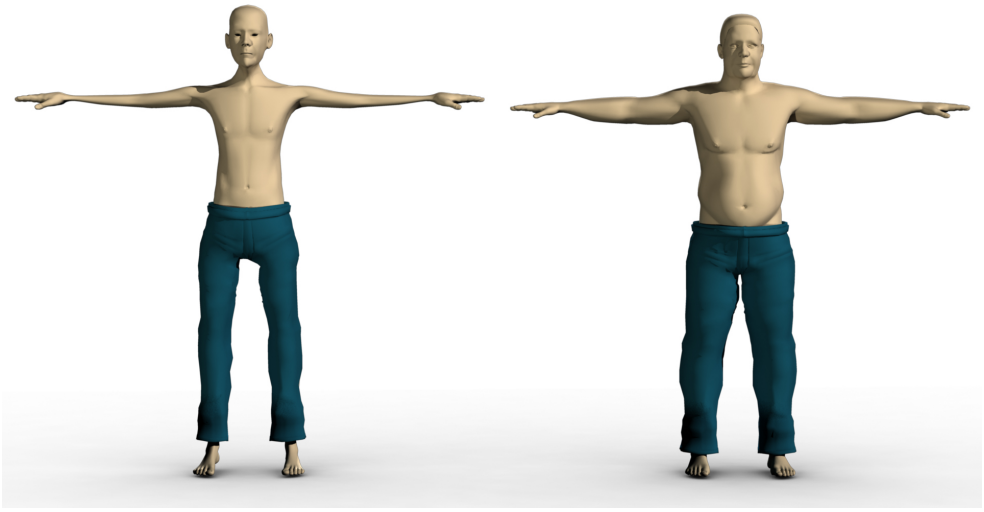

Figure 1.16: *Character on the left has much thinner arms than character on the right.*

For the character on the left side, because the arm is much thinner than the one on the right, the arc length of the "Arm hole" is much shorter than the one on the right. However, two character have identical height, therefore, their shirt have the same back length. Figure 1.17 demonstrates the "Front" pattern resized for the characters in Figure 1.16. The pattern is selected based on the minimum changes of inner angle of the "critical point" of the solutions. As demonstrated in Figure 1.18, the design of the shirt is preserved.

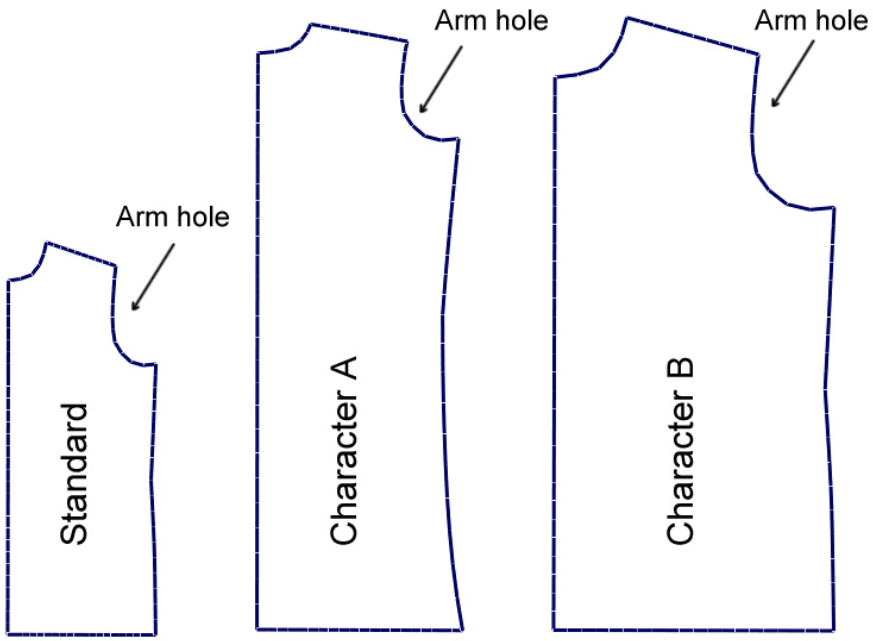

Figure 1.17: *The standard pattern(left), the "Front" pattern for the character on the left in Figure 1.16(middle), the "Front" pattern for the character on the right in Figure 1.16(right)*

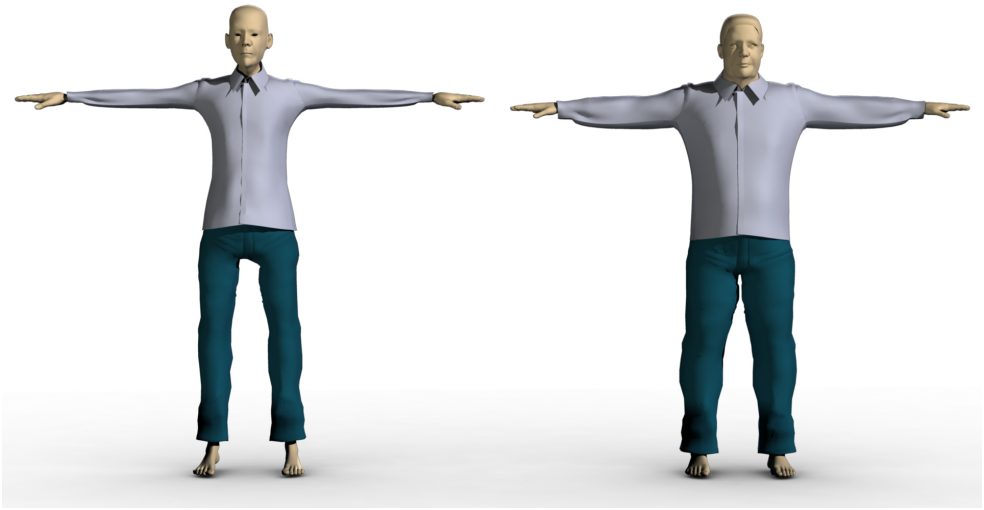

Figure 1.18: *Fit shirt modelled for characters*

Selection

Selection decides individuals who are fit enough to survive the natural selection and reproduce child in next generation. In single objective optimization, solutions can be sorted according to the fitness to objective. However in multi-objective optimization, because objectives are usually conflict to each other, that is, when comparing two solution A and B, A may be better than B in one objective but worse in another objectives at the same time. Therefore, the concept of domination is introduced into this area and two solution are compared by whether one is dominated by another. Deb (2001) defines the concept of domination, that is, a solution A is said to dominate solution B , if following conditions are both true:

Condition 1: The solution A is not worse than solution B in all objectives.

Condition 2: The solution A is strictly better then solution B in at least one objective.

To demonstrate the concept of domination, Figure 1.19 illustrates an optimization problem with five solutions $s_1 ... s_5$ and two objectives f_1 and f_2.

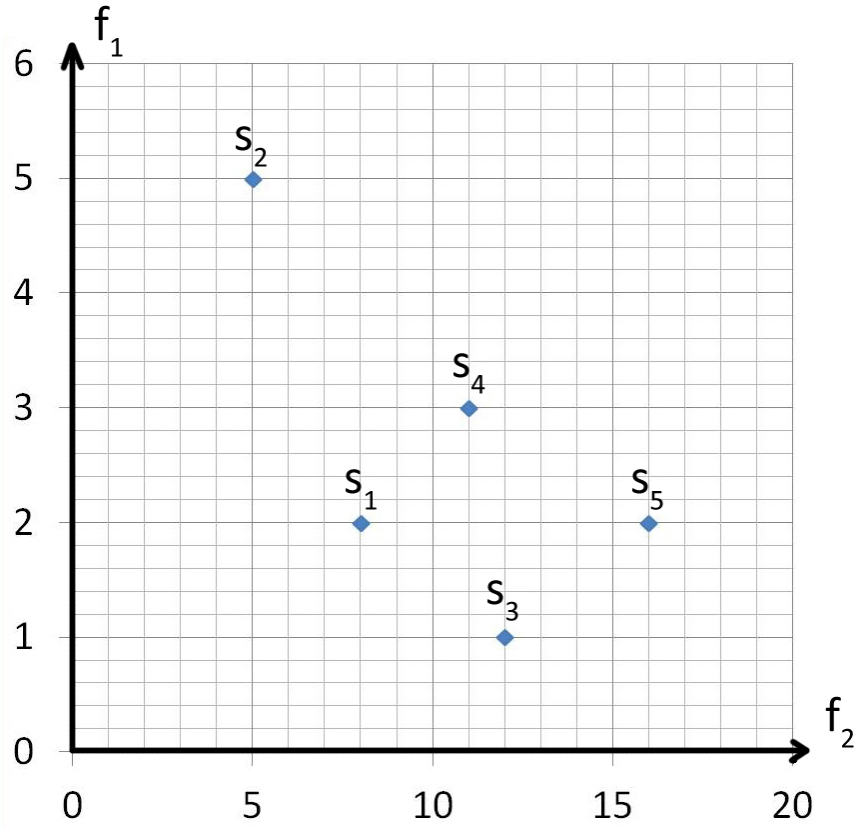

Figure 1.19: *Domination between five solutions*

f_1 and f_2 are two fitness functions for two objectives respectively where the f_1 needs to be minimized while f_2 needs to be maximized. In this case, two objectives f_1 and f_2 are equality important at all time, therefore it is impossible to select one best solution from this population according to both objectives. However, the definition of domination is able to decide the better solution among two solutions. As illustrated in Figure 1.19, when comparing s_1 and s_2, s_1 is better than s_2 in both f_1 and f_2. Thus, s_1 satisfy both conditions of the definition of domination and it can be said that s_1 dominates s_2. When comparing s_1 and s_5, s_5 is better than s_1 in f_2 but equals in f_1, this situation also satisfies both two conditions of the definition of domination, therefore, s_1 is dominated by s_5. When comparing s_4 to s_1, s_4 performs better on f_2 than s_1 but worse on f_1, therefore, s_1 and s_4 does not dominates each other.

Deb et al. (2002) introduced non-dominating sorting algorithm that utilises this method to select all solutions in a population that does not dominated by the rest of the population to construct "Pareto front". To be more specific, if there are two solutions that does not dominates each other, but each of them dominates the rest of the population, it can be said that, these two solutions are "equally the best" and the "Pareto front" is constituted by these two solutions.

In this case, after comparing all pairs of the solutions, it can be concluded that s_5 dominates s_1, s_2, s_4 as well as s_3 dominates s_1, s_2, s_4. Among s_3 and s_5, s_3 exceed s_5 in f_1 but dwarfed by s_5 in f_2. Therefore, s_3 and s_5 are non-dominated by each other but dominate the rest of the population. Consequently, "Pareto front" is constituted by s_3 and s_5.

In this chapter, all solutions are compared according to the definition of domination in terms of three fitness functions, that is, measurements, seamline and pattern shape to from "Pareto front". In the final step, the shape objective is used for selecting one solution with less shape distortion as the final solution for constructing cloth for characters. The general process for pattern resize genetic algorithm is summarized in Algorithm 4.

Algorithm 4 Pattern Resizing

procedure EVOLVE(pop denotes a population contains multiple pattern variations, $Npop$ denotes the number of individuals in pop , $NGen$ denotes the max number of evolve generation)
    for gen in $NGen$ **do**
        $parent$ = Selection(pop, len(pop))
        $offspring$ = Crossover($parent$)
        $mutated$ = Mutation(pop, $MutPb$)($MutPb$ denotes mutation probability)
        for $individual$ in $offspring$ **do**
            $individual.fitness$ = Evaluation($individual$)
        end for
        for $individual$ in $mutated$ **do**
            $individual.fitness$ = Evaluation($individual$)
        end for
        pop = $parent$ + $offspring$ + $mutated$
    end for
end procedure

1.3 Pattern Assembling

After every patten is adjusted into the correct shape and size based on the measurements from the character, they are ready to be placed on character and stitched together to form a complete cloth. In order to achieve such a task, all patterns are positioned onto 3D character simultaneously before sewing takes place. Three steps are involved for the construction of 3D cloth. Firstly, the bounding surface of each body part of character are created, then, patterns are triangulated and then based on the parametric coordinate of the bounding surface, all vertices on a 2D pattern are transferred onto bounding

surface to form a 3D mesh. Finally, according to the seam-line information, constrains are created to every point pair in seam-lines and physical simulation is performed to to stitch patterns together to give the final shape of cloth.

Bounding surfaces are used as a guideline for transferring 2D cloth patterns into 3D mesh. Each pattern is responsible for covering certain part of character body, the bounding surfaces are used to position patterns on their corresponding body location. For human body, the general shape of each body section such as torso, legs or arms, is a cylindrical shape. For this reason, the cylindrical surface is chosen as the bounding surface. Moreover, cloth patterns are developable surface as they are cut out from a flat piece of textile materials. Therefore, construct a developable surface around the body of the character first and then transfer patterns that are also developable onto to it minimize the distortion during the pattern positioning process.

In order to create bounding surface, character model is divided into six parts such as head, torso and four limbs. In this thesis, if the input character has skeleton attached to it, the body parts can be determined by the skin weight corresponding to each bone of the skeleton. If the character does not have skeleton, the body can be segmented interactively by user.

After body segmentation is completed, PCA is performed on each part of the body to calculate the principle vector of each body part. Then, by using the principle vector as the direction, central line of the each body is created as the central axis of the cylindrical bounding surface. The radius of cylindrical bounding surface is then determined by the number of pattern associated with this body part. At this step, pattern are arranged on the flattened bounding surface side by side according to position of the sewing relationship as showed in Figure 1.20. The width on the U direction of all the patterns is the circumference of the bounding surface. Figure 1.21 demonstrates all the bounding surfaces for a character.

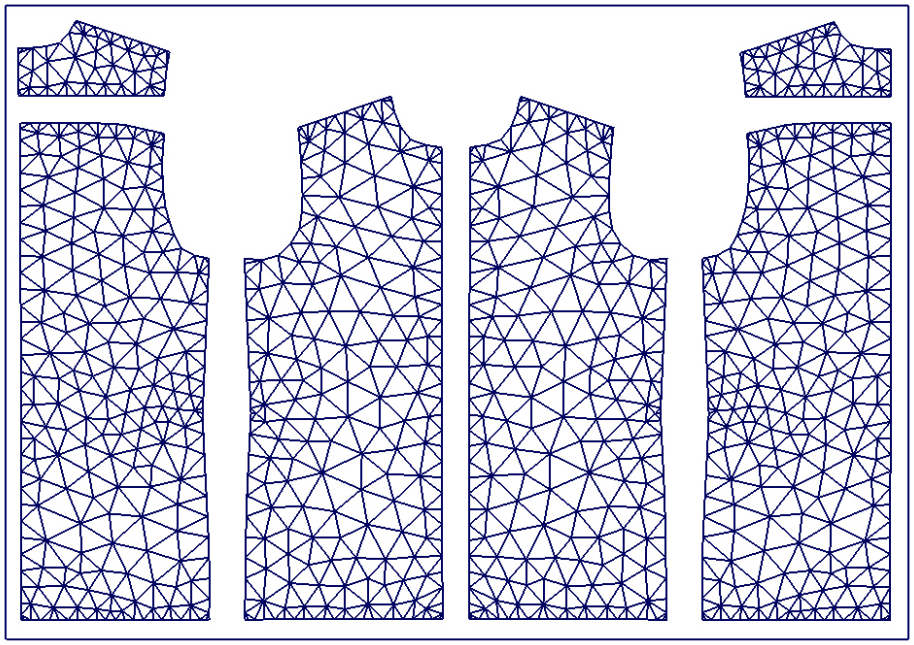

Figure 1.20: *Patterns arranged side by side within the range of a box, this box is the flattened bounding surface.*

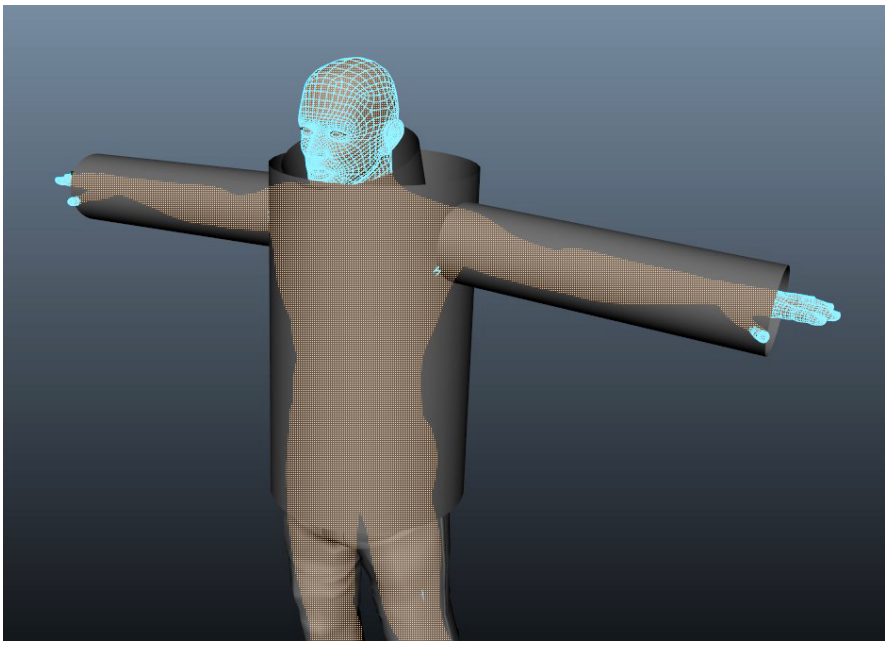

Figure 1.21: *Bounding surfaces on character.*

In the next step, the vertices of the 2D pattern are transferred onto bounding surface. Based on the parametric coordinate of each point of 2D

pattern on the flattened bounding surface, each point can be located by the same parametric coordinate on the cylindrical bounding surface. Finally, by applying the same topology structure to the points on bounding surface, 3D mesh be constructed from 2D cloth pattern. Figure 1.22 demonstrate the fully positioned shirt patterns of a character.

Figure 1.22: *3D patterns*

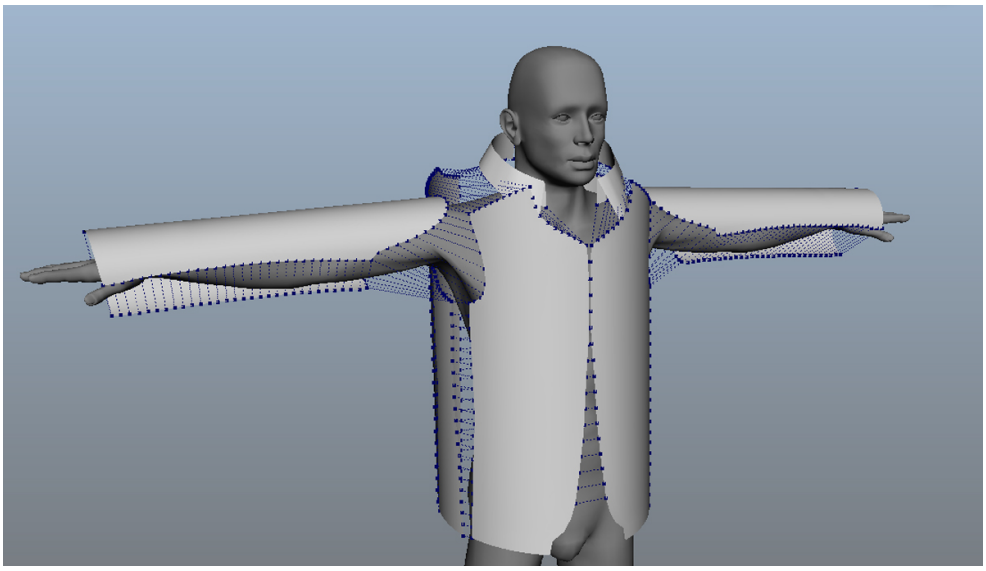

Figure 1.23: *3D pattern sewing*

In the final step, based on the seam-lines between each pattern, constrains are created between two points in a point pair on seam-line. Then physical simulation is performed to pull all the patterns together. In this thesis, the nCloth module in Maya2012 (Autodesk 2012) is used to perform the physical simulation. Figure 1.23 illustrates all the sewing constrains.

In order to validate the algorithm presented in this chapter, a shirt design and a trousers design presented in Xiong (2008) are used for dressing four different characters with largely different body shapes and proportions. This experiment is performed on an Intel Xeon 3.33GHz PC with 8GB RAM running Windows 7 (64-bit) operating system and Maya2012. For cloth pattern adjusting genetic algorithm, 400 initial solutions are generated at the beginning of the algorithm. For each character, 200 evolutions are performed. Figure 1.26, 1.27, 1.28 and 1.29 demonstrate the final results for each character respectively. Figure 1.30 and 1.31 demonstrate the final cloth fit onto each character. Table 1.2 lists out the fitness value of the final solution of all characters. Notices that for character A and D, because the body proportion is largely different from the standard human body proportion that the shirt pattern and trousers pattern are based on, cloth pattern requires larger deformation to fit the character A and D than character B and C. Therefore, the result of the shape evaluation for the patterns for character A and D is worse than the patterns for character B and C.

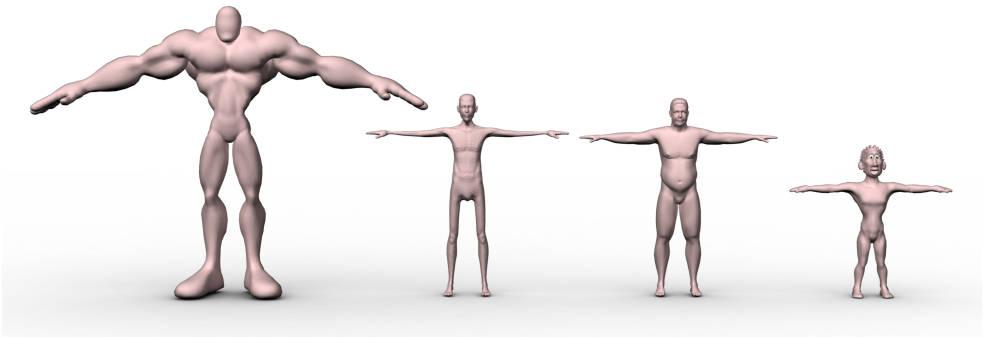

Figure 1.24: *Four different characters used for the cloth modelling experiments, from left, Character A, B, C and D.*

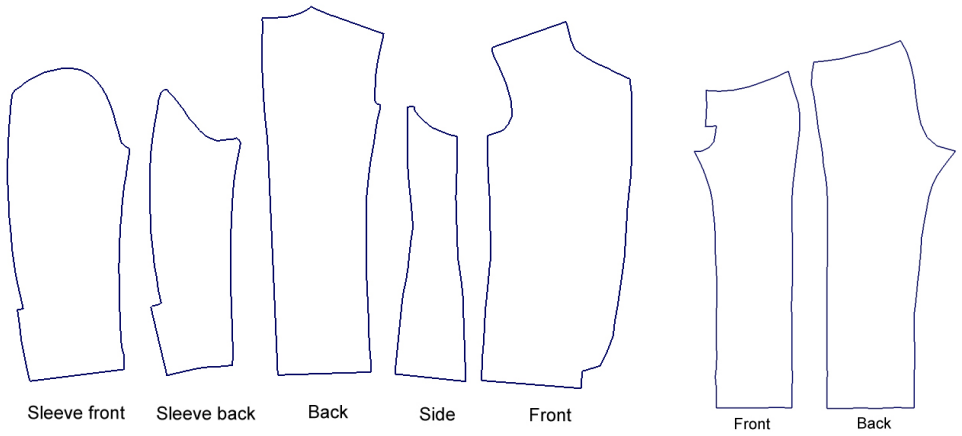

Figure 1.25: *Cloth patterns(Xiong 2008), standard shirt patterns(left); standard trousers patterns(right)*

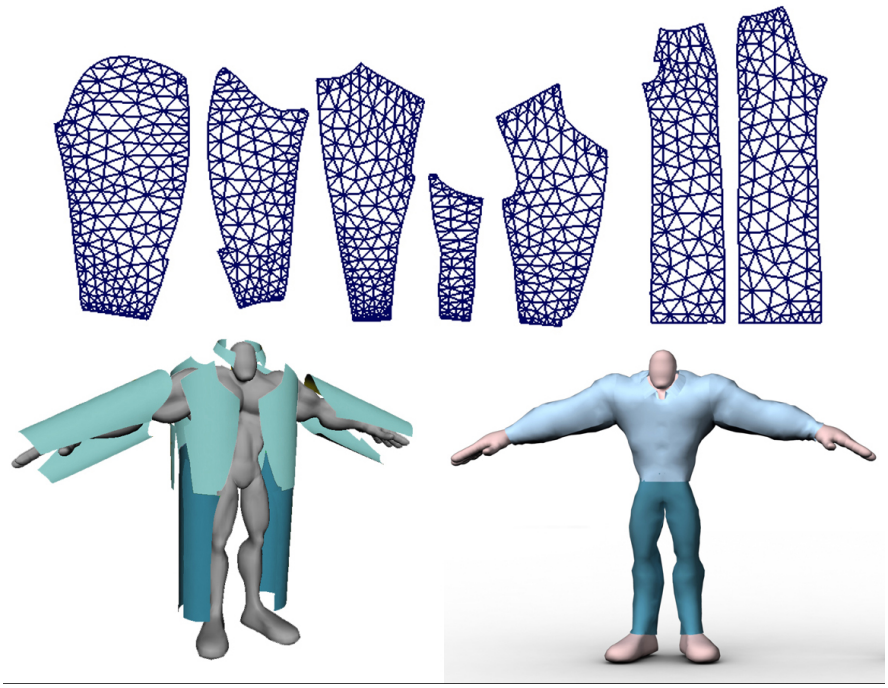

Figure 1.26: *Dressing result for character A*

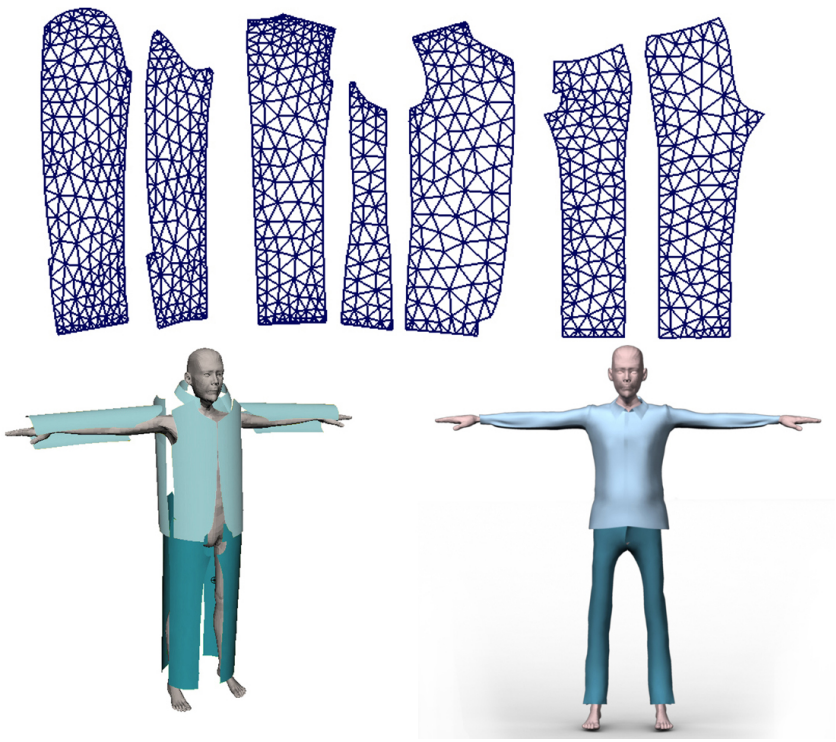

Figure 1.27: *Dressing result for character B*

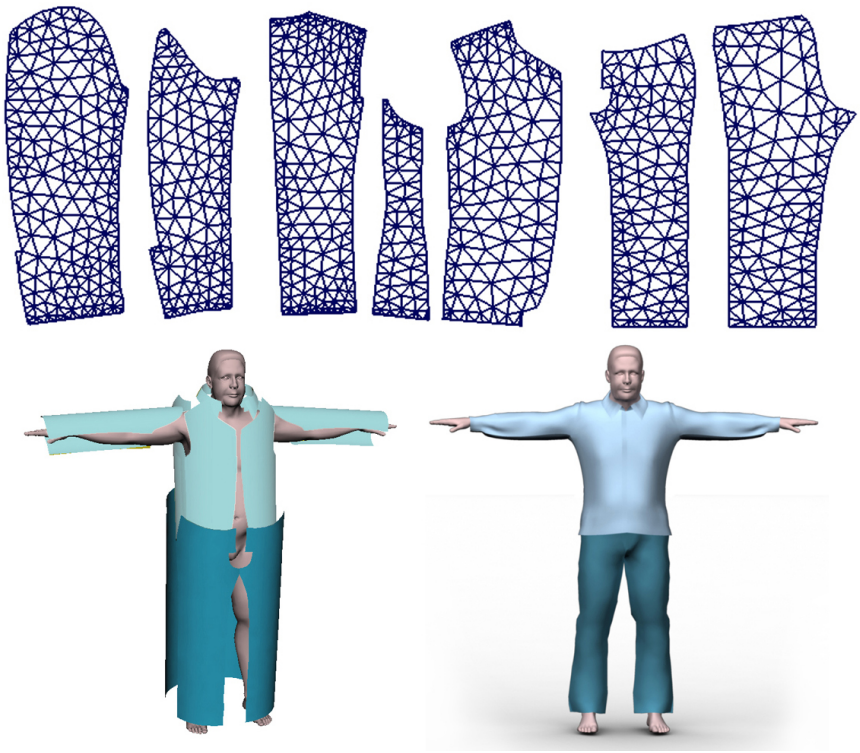

Figure 1.28: *Dressing result for character C*

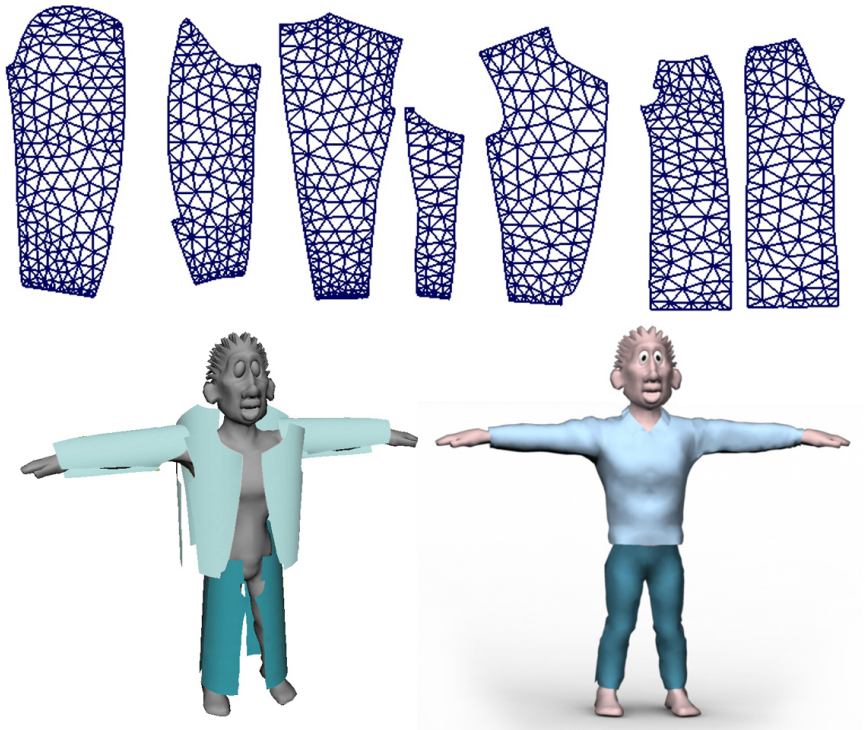

Figure 1.29: *Dressing result for character D*

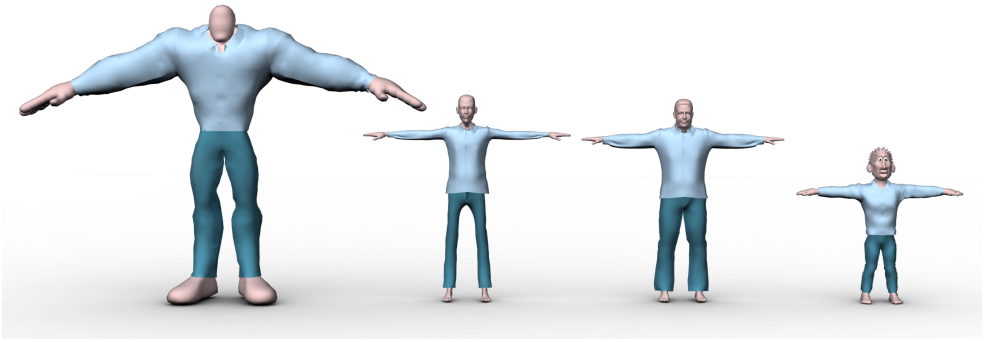

Figure 1.30: *Front view of four different characters in the same cloth design.*

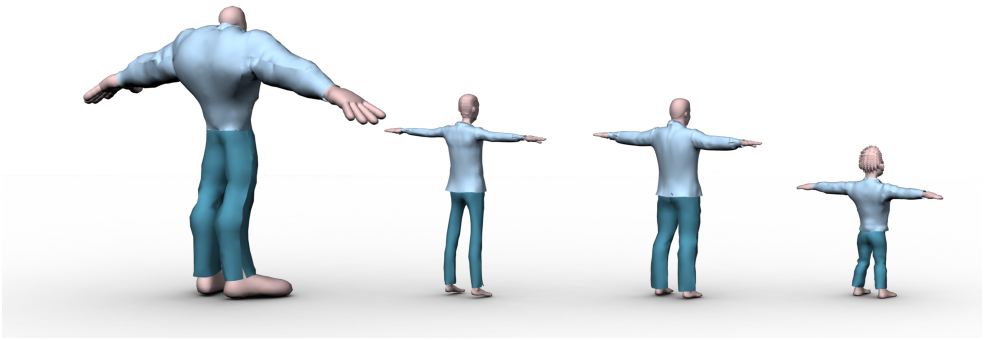

Figure 1.31: *Rear view of four different characters in the same cloth design.*

Evaluation function	Character A	Character B	Character C	Character D
Measurements	2.927	1.803	2.125	2.842
Seamline	1.142	1.248	1.440	1.384
Shape	0.446	0.112	0.092	0.389
Running time	784 sec	791 sec	752 sec	743 sec

Table 1.2: *The fitness values of three evaluation function of the final solution for all characters and the running time for each character, unit of time is second, node that smaller value indicates better fit to the criteria.*

In order to further demonstrate the modelling ability of proposed cloth modelling method, three new character is introduced and several types of

cloth ranging from female dress to t-shirt are modelled for these characters, the results are demonstrated in Figure 1.32 to Figure 1.35.

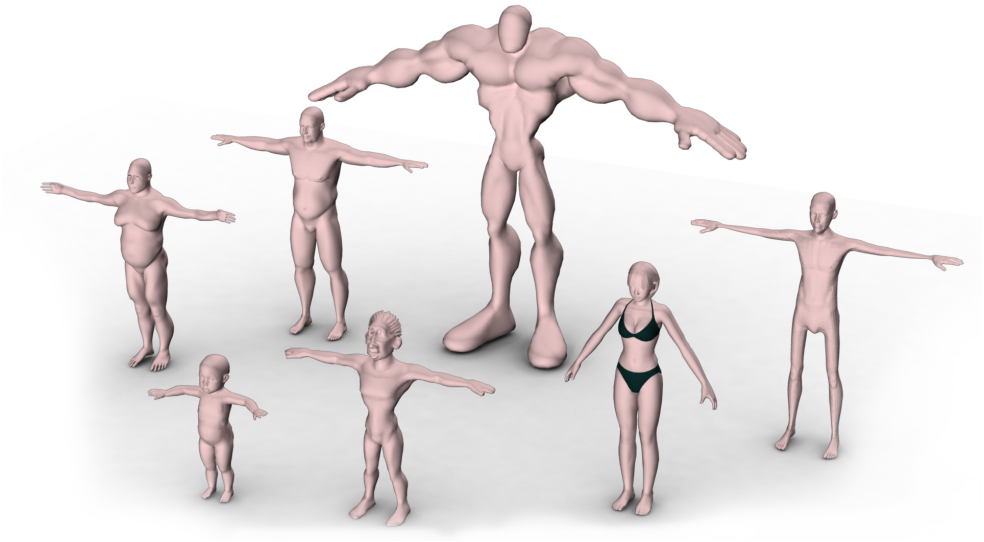

Figure 1.32: *Characters used for dressing various clothes*

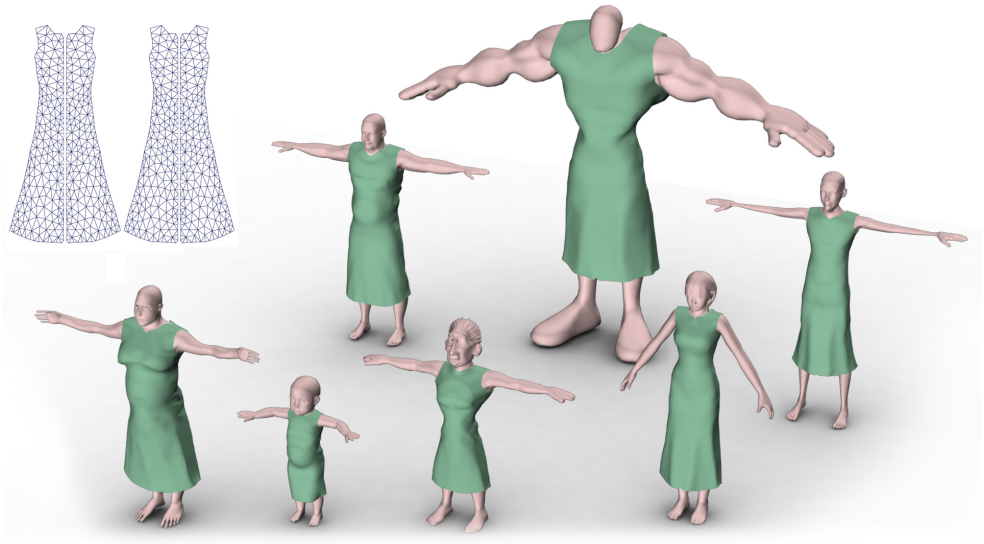

Figure 1.33: *Tight dress on different characters*

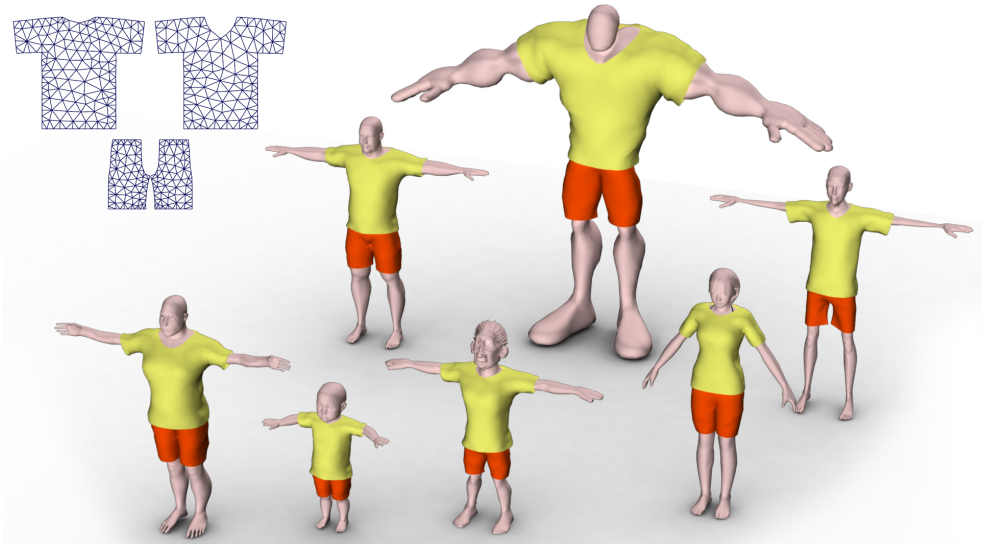

Figure 1.34: *T-Shirt and short on different characters*

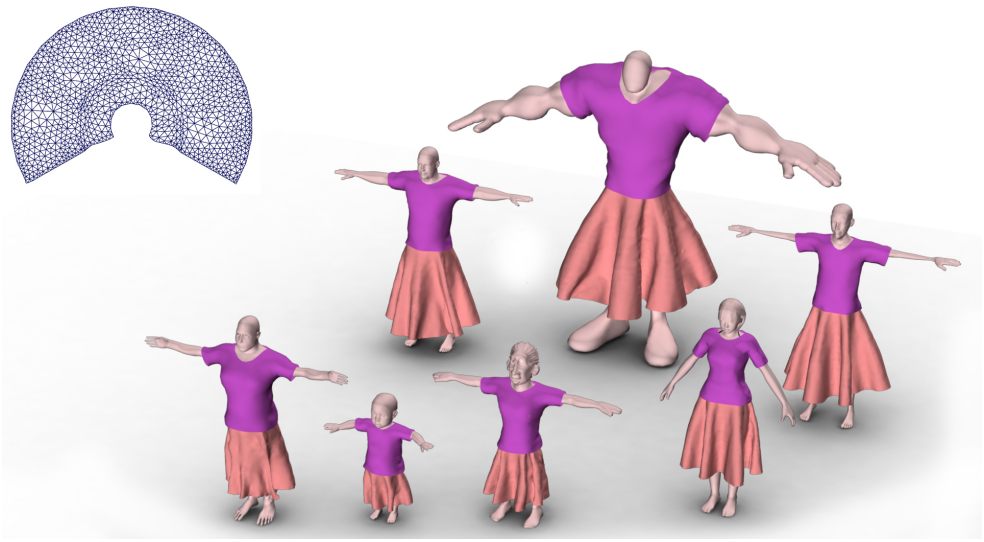

Figure 1.35: *Skirt on different characters*

1.4 Conclusions

This chapter presented an automatic pattern based cloth modelling method for dressing cloth onto characters with different body shapes and proportions. This method takes 2D cloth patterns as inputs. By extracting the measure-

ments from a character model, the shape and size of the cloth patterns are adjusted automatically to fit the cloth onto a character. During pattern adjustment, by considering the measurements, seam-line among the patterns as well as the shape of each pattern, the original design of the cloth is preserved throughout the fitting process. Finally, all patterns are transformed into mesh and positioned around the body, physical simulation is used to stitch all the patterns together based on seam-lines on each pattern to construct the final 3D cloth.

This method utilises genetic algorithm to adjust cloth patterns for cloth fitting. It has four major advantages.

Firstly, given the measurements of any character, this method can fits the cloth onto the character automatically. By automating the pattern adjustment process using genetic algorithm, the manual pattern adjustments that required by the current pattern based cloth modelling methods can be avoid. The duplication of effort required by traditional cloth modelling method can be eliminated and the efficiency for modelling cloth for different characters with different body shapes and proportion can be largely improved. This advantages shines itself even more when multiple different characters are required in the virtual environment.

Secondly, this method models cloth based on cloth design patterns without requiring tailoring expertise. By automating pattern adjustment process, deep tailoring knowledge involved in patternmaking process can be avoided. This method provides an efficient and easy-to-use solution for the animation artists. It empowers their creativity and improves their productivity by allowing them to use the large amount of existing cloth patterns in the fashion industry to create various clothes that fit different characters.

Thirdly, this method generates 3D cloth based on the adjusted 2D cloth patterns. The cloth modelling workflow is proceed in one direction. There is no turning back for extracting patterns from 3D cloth which is required

by current cloth modelling method. Therefore, the shape distortion that is introduced by 3D surface flattening process can be avoid and the extra computation for 2D pattern extracting can also be eliminated.

Most importantly, cloth are represented by 2D pattern initially, which is the unified form for all clothes. A modelled cloth can be stored in asset warehouse in the form of cloth pattern. When dressing a new character, only measurements of that character need to be recalculated and existing cloth pattern can be reused to fit a cloth onto the new character automatically. The reusability of modelled cloth is improved significantly by modelling cloth from patterns.

Bibliography

H. J. Armstrong (2000). *Patternmaking: for fashion design*. Pearson Prentice Hall.

Autodesk (2012). 'Maya2012'. Software. Autodesk Inc.

K. Deb (2001). *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley Interscience Series in Systems and Optimization. Wiley.

K. Deb, et al. (2002). 'A fast and elitist multiobjective genetic algorithm: NSGA-II'. *Evolutionary Computation, IEEE Transactions on* **6**(2):182–197.

R. Digest (2010). *New Complete Guide to Sewing: Step-By-Step Techniquest for Making Clothes and Home Accessories, Simplicity Patterns*. Reader's Digest. Reader's Digest Association, Incorporated.

EN:13402 (2001). *Size designation of clothes*.

A. Ghosh & S. Tsutsui (2003). *Advances in Evolutionary Computing: Theory and Applications*. Natural Computing Series. Springer.

J. Grefenstette (1986). 'Optimization of Control Parameters for Genetic Algorithms'. *Systems, Man and Cybernetics, IEEE Transactions on* **16**(1):122–128.

T. Gwiazda (2006). *Crossover for single-objective numerical optimization problems*. Genetic algorithms reference. TOMASZGWIAZDA E-BOOKS.

G. M. Hannah (1919). 'Dressmaker's Pattern Outfit'. US Patent No.1313496.

R. Haupt & S. Haupt (2004). *Practical Genetic Algorithms*. Wiley.

J. H. Holland (1992). *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA.

K. Howland (2008). 'The Merits of a Basic Fitting Pattern'. *Threads* **79**:48 – 52.

ISO/TR-10652 (1991). *Standard sizing systems for clothes*.

D. G. Kendall (1977). 'The Diffusion of Shape' .

Z. Michalewicz (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Artificial intelligence. Springer.

C. Moore, et al. (2001). *Concepts of pattern grading: techniques for manual and computer grading*. Fairchild Pub.

V. Pareto (1906). *Manual of political economy (manuale di economia politica)*. Kelley, New York. Translated by Ann S. Schwier and Alfred N. Page.

S. Rosen (2004). *Patternmaking: a comprehensive reference for fashion design*. Pearson Custom Library: Fashion Series. Pearson Prentice Hall.

N. A. Schofield & K. L. LaBat (2005). 'Exploring the Relationships of Grading, Sizing, and Anthropometric Data'. *Clothing and Textiles Research Journal* **23**(1):13–27.

W. Spears & V. Anand (1991). 'A study of crossover operators in genetic programming'. In Z. Ras & M. Zemankova (eds.), *Methodologies for Intelligent Systems*, vol. 542 of *Lecture Notes in Computer Science*, pp. 409–418. Springer Berlin Heidelberg.

M. Srinivas & L. Patnaik (1994). 'Adaptive probabilities of crossover and

mutation in genetic algorithms'. *Systems, Man and Cybernetics, IEEE Transactions on* **24**(4):656–667.

N. Xiong (2008). *World Classical fashion Design and Pattern*. Jiangxi Art Press.

F. Yaman & A. Yilmaz (2010). 'Investigation of fixed and variable mutation rate performances in real coded Genetic Algorithm for uniform circular antenna array pattern synthesis problem'. In *Signal Processing and Communications Applications Conference (SIU), 2010 IEEE 18th*, pp. 594–597.