UNIVERSITÉ DE GENÈVE

	FACULTÉ DES SCIENCES
Département d'informatique	Professeur J. Rolim
	FACULTÉ DES SCIENCES ÉCONOMIEQUES ET SOCIALES
Département de systèmes d'information	Professeur N. Magnenat-Thalmann

Real-time Animation of Dressed Virtual Humans

THÈSE

présentée à la Faculté des sciences de l'Université de Genève
pour obtenir le grade de Docteur ès sciences, mention informatique

par

Frederic Cordier
de
Lyon (France)

Thèse N° 3501

GENÈVE
Atelier de reproduction de la Section de physique
2004

Acknowledgments

This thesis owes its existence to the help, support, and inspiration of many people. In the first place, I would like to express my sincere appreciation and gratitude to my thesis supervisor, Prof. Nadia Magnenat-Thalmann for her support and encouragement during the more than five years of this thesis's work. She has provided for an optimum working environment at MIRALab, University of Geneva, where a lack of resources is something unimaginable due to her managerial skills and foresight. Her uncompromising quest for excellence significantly shapes everyone at MIRALab. I would also like to address my thanks to the jury members of the thesis Prof. José Rolim (University of Geneva), Prof. Sabine Coquillart (groupe I3D, INRIA) and Prof. Wolfgang Straßer (University of Tübingen) for the final review.

The discussions and cooperation with all of my colleagues have contributed substantially to this work: Pascal Volino for his valuable advices on cloth deformation, Tom Molet who has facilitated the integration of my work to VHD++ as well as Georgios Papagiannakis for providing me some of his VHD++ libraries, Marlene Arevalo and Christinane Luible for the design of the cloth models and the bodies, Dimitri Protopsaltou for his collaboration in the frame of the European projects, Nedjma Cadi for providing the motion data, Gregoire L'hoste for his advices on making videos, Hyewon Seo for her fruitful discussions on example-based approaches and her endless support. I am as well very grateful to all members of MIRALab for the cooperative spirit and the excellent working atmosphere, creating a unique setting for intellectual explorations. I also extend my appreciation to all staff members of the CUI (Centre Universitaire d'Informatique) for their assistance and support.

Finally, I owe special gratitude to my family for continuous and unconditional encouragement during all the time I have spent for my PhD.

This research would not be possible without the financial support from the FNRS (Le Fonds National Suisse de la Recherche Scientifique or the Swiss National Research Foundation).

Abstract

Cloth simulation has a long history, starting almost three decades ago. Thanks to the continuous increase of computation speed of computers, real-time simulation of clothes has become very recently possible. The research on real-time clothes has really begun few years ago when researchers have started simulating simple garments such as flags or tablecloths. Nowadays, the research aims at proposing methods that can simulate cloth in real-time independently of its complexity.

Most of the existing approaches for real-time cloth simulation consider clothes as independent object and are able to simulate the dynamics of cloth either as clothing or other garments like in furnishing tablecloth. Such assumption requires implementing a mass spring system with collision detection. The main drawback of these approaches is that they are slow at computation, making the simulation of dressed humans hardly achievable. In this dissertation, we propose to consider clothes as being anchored to the underlying skin surface rather than as independent object. With such an assumption, many optimizations can be made. Collision detection can be restricted to the cloth vertices and their anchorage on the skin. Cloth dynamics can be simulated with a faster method than the usual particle system. Starting from this idea, two different approaches have been developed and experimented.

The first approach works in a hybrid manner exploiting the merits of both the physically based and geometric deformations. It makes use of predetermined conditions between the cloth and the body model, avoiding complex collision detection and physical deformations wherever possible. Garments are segmented into pieces that are simulated by various algorithms, depending on how they are laid on the body surface and whether they stick or flow on it. Tests show that the method is well suited to fully dressed virtual human models, achieving real-time performance compared to ordinary cloth-simulations.

The main idea of the second approach consists of developing a cloth simulator that can learn the cloth behaviour through a sequence of pre-computed cloth simulation. This approach enables us to simulate the garment features such as wrinkles, pockets… in real-time. We setup the problem as simulating the garments in two phases. The first phase, a rough mesh reproduces the dynamic behaviour of the garments. Its physical properties are defined by the pre-calculated sequence. In the second phase, the fine mesh simulates the details of the garments.

Finally, we introduce one application of the real-time clothes: the Virtual Try-On. This application aims to provide to customers a mannequin of their size wearing the cloth of their choice.

Résumé

De nos jours, la simulation d'avatars en temps réel occupe une place de plus en plus importante dans la recherche dédiée à l'informatique graphique. Depuis quelques années, de multiples applications apparaissent comme la réalité virtuelle et augmentée, les jeux vidéos, les films d'animation. La simulation d'humains virtuels en temps réel inclut plusieurs technologies comme l'animation faciale, la déformation de la peau, la modélisation du comportement (intelligence artificielle)… Parmi toutes ces technologies, celle qui est la moins aboutie est la simulation des vêtements. L'état actuel de l'art ne permet de simuler que des habits très simples composés d'un nombre réduit de polygones.

L'objectif de cette thèse est d'étudier et de proposer des solutions pour la simulation des habits temps réel qui reste encore un sujet de recherche à explorer. Plus précisément, nous voulons développer un ensemble des méthodes pour résoudre les différents problèmes que pose l'animation temps réel des habits, c'est à dire la détection de collisions qui doit être rapide et précise et la simulation des propriétés mécaniques des tissus qui doit aussi être rapide à calculer et reproduire de façon la plus fidèle le comportement des habits.

1. Etat de l'art de la simulation temps réel

L'animation des habits sans la contrainte du temps réel est l'un des domaines des plus explorés de l'informatique graphique depuis deux décennies [TERZ 87] [AONO 90] [LAFL 91] [CARI 92] [BREE 94] [EBER 96] [BARA 98] [BRID 02] [CHOI 02] [VOLI 00] [VOLI 01] [VOLI 02]. La recherche a commencé par le développement de systèmes très simple à base de masse et ressorts. De nos jours, les multiples aspects de l'animation des habits ont été étudiés, comme la détection de collisions, la modélisation du contact du tissu avec les autres objets, les propriétés mécaniques. Un effort particulier a été fait pour augmenter la précision de tous les composants de la simulation. De nos jours, ces technologies ont abouti à de multiples applications commerciales comme les films d'animation "Sreck" ou encore "Monster & Co.".

L'animation temps réel est un sujet beaucoup moins exploré. Le développement de ce domaine de recherche est en partie lié à l'évolution de la puissance des ordinateurs. Ce n'est seulement que récemment que la puissance des machines a permit d'envisager une application temps réel des habits. Créer un modèle de simulation d'habits en temps réel implique de travailler sur les deux aspects principaux qui sont la modélisation du comportement de l'habit et la détection de collisions. Pour ces deux aspects, nous dressons un état de l'art.

1.1 Simulation avec les méthodes géométriques

La première approche consiste à simuler les habits avec des méthodes géométriques [WEIL 86] [AGUI 90]. La forme au repos des habits peut être reproduite par exemple avec une surface caténaire. D'autres surfaces paramétriques peuvent être aussi utilisées comme les surfaces basées sur les équations trigonométries pour modéliser les plis des habits. Le principal problème de ces méthodes est qu'elles sont incapables de reproduire correctement le mouvement dynamique de l'habit. Ces approches peuvent être considérées comme des outils de modélisation.

1.2 Simulation avec les méthodes basées sur la physique

La deuxième approche est basée sur un modèle physique. Le modèle le plus utilisé est le système "masses ressorts". L'habit est décomposé en un ensemble de masses reliées par des ressorts. L'ensemble de ces masses et ressorts compose la forme du tissu. Les équations du mouvement sont définies pour chaque particule de la façon suivante:

$$\ddot{x} = M^{-1}\left(-\frac{\partial E}{\partial x} + F\right) \tag{1}$$

L'accélération des particules est définie comme fonction de la masse M , le gradient de l'énergie interne au système $\frac{\partial E}{\partial x}$, et la somme des forces externes F . La façon dont est calculé le gradient de l'énergie interne permet de modéliser les propriétés mécaniques de l'habit comme la résistance du tissu à la torsion, compression, étirement… Un modèle qui reproduit fidèlement ces propriétés mécaniques est coûteux à calculer et donc ne peut pas être utilisé pour le temps réel. La plupart des méthodes dédiées au temps réel utilisent la configuration la plus simple du "masses ressorts" dans lequel chacune des arrêtes est un ressort.

L'équation (1) est une équation aux dérivées partielles qui doit être résolu à chaque étape de la simulation. Il existe deux méthodes différentes pour les résoudre, l'intégration Euler explicite (2) et implicite(3).

$$v_i^{t+h} = v_i^t + F_i^t \frac{h}{m_i} \qquad x_i^{t+h} = x_i^t + v_i^{t+h} h \tag{2}$$

$$v_i^{t+h} = v_i^t + F_i^{t+h} \frac{h}{m_i} \qquad x_i^{t+h} = x_i^t + v_i^{t+h} h \tag{3}$$

La principale différence entre ces deux approches est le choix du calcul de la force. La méthode peut utiliser soit les forces générées à l'instant courant F_i^t (2), soit les forces au pas de temps suivant F_i^{t+h} (3). L'intégration Euler implicite est la plus stable; elle permet d'utiliser des pas de temps importants, cet avantage est essentiel pour la simulation temps réel [BARA 98]. D'un autre coté, cette méthode dégrade la qualité de la simulation en augmentant artificiellement l'amortissement du mouvement du tissu. Une troisième méthode d'intégration consiste à combiner les avantages les deux méthodes décrites ci-dessus [EBER 00]. Néanmoins, l'intégration implicite reste la plus efficace pour ce qui est du temps de calcul et elle est le fondement de l'animation de phénomènes physiques en temps réel.

1.3 Simulation avec les méthodes hybrides

L'idée d'utiliser des méthodes basées à la fois sur la physique et la géométrie vient du constat suivant: les détails de l'habit qui sont les plus coûteux à calculer sont par nature géométrique. Ils peuvent être donc modélisés avec une surface déformée géométriquement sans perdre la qualité de la simulation. En clair, le comportement global de l'habit est simulé avec un maillage simplifié et les détails sont générés avec une surface paramétrique dont les points de contrôles sont les sommets du maillage simplifié [KANG 02] [OSHI 01]. Cette méthode est rapide mais son principal inconvénient est la qualité de la forme des plis. Il existe une très grande variété de la forme de ces plis et il est très difficile de définir une fonction assez générale qui permette de modéliser n'importe quel type de plis.

1.4 La détection de collisions

L'autre aspect de la simulation des habits est la détection de collisions. La détection de collisions en temps réel est un problème compliqué, plus particulièrement pour le cas des habits qui sont des surfaces déformables. Plusieurs solutions ont été proposées dans la littérature scientifique.

L'approche classique consiste à construire une hiérarchie de boites englobantes à partir de la surface [VOLK 00]. Chaque boite englobante contient un morceau de l'objet. La détection de collisions se fait par niveau de détails. Si une collision est détectée entre deux objets, l'algorithme teste la collision dans un niveau de détail plus élevé en descendant dans la hiérarchie. La principale limitation de cette approche est la mise à jour de cette arborescence à chaque déformation de l'habit. Cette méthode est couramment utilisée dans le cadre de simulations d'habits ou le temps de calcul n'est pas une contrainte.

La seconde approche est basée sur une partition de l'espace en cube de façon récursive de façon à former une hiérarchie [MEYE 00]. Chaque cube contient un "bit" indiquant si l'espace est occupé par l'objet ou non. Cette méthode est assez similaire à la première. Comme elle, elle soufre des même inconvénients: la nécessite de mettre à jour la hiérarchie à chaque modification de la forme de l'habit.

La troisième voie consiste à utiliser le processeur graphique [VASS 01]. Grâce à ce processeur, un calcul de rendu de l'habit peut être fait très rapidement et être utilisé pour savoir si l'habit touche le corps. La principale limitation est qu'elle ne permet pas de calculer les auto-collisions de l'habit.

La quatrième solution est basée sur les surfaces implicites [RUDO 00]. Le principal avantage des surfaces implicites est la possibilité de calculer de façon instantanée si un point est à l'intérieur ou l'extérieur du volume défini par la surface. Le principal inconvénient est que cette méthode ne peut être utilisée que pour le calcul de détection sur des volumes, c'est à dire sur le corps seulement.

1.5 La simulation basée sur l'utilisation d'exemples

Cette classe de méthodes consiste à créer un interpolateur qui est capable d'apprendre la déformation d'un objet à partir d'exemples donnés par l'utilisateur. Cette approche est assez récente et a été utilisée pour la déformation de la peau. Un essai d'adaptation de cette méthode aux cas des habits a été publié cette année [JAME 03]. L'interpolateur est construit à partir d'exemples d'animations d'habits pré-calculés avec un logiciel commercial. La principale limitation de ce simulateur est le nombre de degrés de libertés offert à l'utilisateur/trice: L'habit est attaché à une porte qui ne peut être déplacement que suivant un axe fixe et selon trois différentes vitesses. Cette limitation s'explique par le fait que la forme des objets physiques comme les habits dépendent d'un nombre très important de paramètres comme la vitesse et la position de chaque point de l'habit. Ce nombre important de paramètres rend l'espace d'interpolation très grand et donc nécessite un nombre important d'exemples pour construire l'interpolateur.

1.6 Conclusion

Parmi ces différentes méthodes, nous pensons que la meilleure approche peut être obtenue en associant plusieurs méthodes afin de combiner les avantages des unes et des autres. Le deuxième constat que nous pouvons faire de cet état de l'art est que les méthodes les plus rapides sont celles qui sont basées sur des hypothèses fortes quant aux possibilités d'interactions sur l'habit. Le meilleur exemple est donné par la publication [JAME 03]. La simulation est d'une qualité qu'aucun autre simulateur ne peut atteindre. Par contre, les hypothèses sont que l'habit reste attachée à une porte qui ne peut se déplacer selon trois vitesses prédéfinies. Les actions que l'utilisateur peut produire sur l'habit sont très limitées. L'objectif de la recherche sur les habits temps réel est de trouver un juste milieu entre les hypothèses et la rapidité de calcul.

Notre première hypothèse est que l'habit soit toujours porté par un humain virtuel. Mous n'envisageons pas le cas d'une simulation de vêtements sans support. Le fait d'attacher l'habit au squelette permet de multiples optimisations, en particulier pour la détection de collisions. L'habit est considéré comme une seconde peau.

Cette hypothèse nous permet de développer des méthodes différentes pour chaque comportement de l'habit. Les habits peuvent être catégorisés selon trois groupes: les habits très serrés, les habits qui restent sur la même région du corps durant toute la durée de la simulation et enfin, les habits flottants dont les mouvements les mettent en contact avec différentes parties du corps. En utilisant une méthode dédiée pour chaque cas, le temps de calcul peut être grandement réduit. L'exemple le plus simple est celui des habits serrés. Ces habits suivent exactement la déformation de la peau; une déformation géométrique est suffisante pour simuler un tel comportement. La détection de collisions et la déformation avec un système "masses ressorts" ne sont pas nécessaires.

2. Déformation de la peau et création des habits

Afin de pouvoir simuler un habit en temps réel, il est nécessaire de développer d'abord les technologies pour la déformation de la peau et le dessin des habits. Cette section décrite notre modèle de déformation de la peau ainsi que les outils pour la création des habits.

La simulation de la peau est un problème qui a été étudié depuis de nombreuses années. La méthode la plus couramment utilisée est le "skinning". Cette méthode calcule la position d'un point de la peau comme une somme avec poids de la position du même sommet dans le repère local de chaque joint qui influence le sommet. Le principal inconvénient est que cette méthode écrase le volume de la peau quand les angles des joints du squelette sont importants.

$$P_v = P_{Dress} . \sum_i w_i . M_i \quad \text{avec} \quad M_i = M_{i,C} \cdot M_{i,Dress}^{-1} \qquad (4)$$

Très récemment, beaucoup de travaux ont été développés pour améliorer cette méthode. La plupart d'entre eux sont basée sur des interpolateurs construits avec des exemples. Le principe de fonctionnement de ces méthodes est assez simple. Un artiste dessine la forme du corps pour plusieurs postures appelées postures clés. La méthode calcule les formes intermédiaires du corps en interpolant les postures clés.

Notre objectif n'est pas de développer une méthode de déformation de la peau toute seule. En fait, nous utilisons la déformation de la peau dans un cadre bien particulier, celui de l'animation des habits. Il existe donc un certain nombre de contraintes qui sont:

- La méthode doit être capable de calculer le repère local de chaque sommet de la peau.
- Les structures de données doivent être les mêmes que ceux de la méthode usuelle de la déformation de la peau. Ceci facilitera son implémentation.
- Finalement, la méthode ne doit pas avoir l'inconvénient principal de la méthode "Skinning", c'est à dire, l'écrasement du volume de la peau quand le squelette est animé.

L'origine du principal défaut de la méthode "skinning" vient de la façon dont sont combinées les matrices de transformations. Afin d'obtenir un calcul correct de combinaisons de matrices de transformations, nous avons utilisé les résultats de recherche de Alexa [ALEX 02]. Un opérateur pour additionner plusieurs matrices est défini de la façon suivante:

$$\bigoplus_i w_i \cdot M_i = e^{\sum_i w_i \log(M_i)} \qquad (5)$$

En remplaçant $\sum_i w_i . M_i$ dans l'équation (4) par $\bigoplus_i w_i \cdot M_i$, nous obtenons une déformation qui est bien meilleure que la méthode usuelle. De plus, la matrice est calculée correctement pour chaque sommet de la peau.

Afin de pouvoir simuler des habits en temps réel, la première étape est de le dessiner et de les adapter à la forme du corps. Ceci est accompli grâce à un logiciel que nous avons développé dans le cadre de MIRALab.

3. Déformation hybride basée sur la géométrie et la physique

La première méthode que nous avons expérimentée utilise une forme statique de l'habit adapté à la forme du corps de l'humain virtuel. L'idée principale de cette approche est de découper l'habit en plusieurs segments, chaque segment étant simulé par une méthode qui lui est dédié. Chacun de ces segments correspond un comportement spécifique de l'habit que nous avons classifié en trois catégories.

La première catégorie englobe tous les habits serrés comme les sous-vêtements ou tee-shirts moulant. La forme des habits suit exactement les déformations de la peau. La simulation de ce type d'habits peut tout simplement être faite en gardant une distance constante entre les surfaces de l'habit et de la peau.

La deuxième catégorie couvre tous les habits qui se déplacent par rapport à la peau mais qui est en collision avec toujours la même région du corps. Le déplacement de ces habits se fait principalement verticalement par rapport à la surface la peau. Un modèle a été spécifiquement développé pour ce cas. La détection de collisions a été réduite; elle n'est calculée qu'entre l'habit et sa région associée du corps aussi appelée point d'ancrage.

La troisième catégorie regroupe tous les habits qui ne sont pas spécifiquement associés à une partie du corps. Un exemple typique est la robe large. La partie droite de la robe peut entrer en contact avec la jambe gauche et vice versa. Il est donc impossible de définir pour ces habits un point d'ancrage.

3.1 Préparation des habits

La préparation des habits est nécessaire pour deux raisons. Il faut d'abord définir la répartition des sommets de l'habit suivant la catégorisation définie dans le paragraphe précédent. Deuxièmement, les structures de données doivent être préparées par chaque sommet en fonction de sa catégorisation.

Un algorithme a été développé pour la segmentation semi-automatique des habits. Elle est basée sur le calcul de la distance minimale qui sépare la peau de l'habit étudié. Le principal problème rencontré est que la forme statique de l'habit ne fournit pas suffisamment d'information quant au comportement de l'habit. Un habit proche de la peau peut être classifié comme moulant alors qu'il peut s'agir d'une robe. Un travail de l'utilisateur est nécessaire afin de corriger cette segmentation.

La deuxième étape est le calcul des structures de données nécessaire pour chacune des trois méthodes de déformation. Les deux premiers types d'habits sont associés à une région de la peau. Les points d'ancrage de ces habits sont définis et enregistrés dans les structures de données.

3.2 Simulation temps réel

Dans cette section nous décrivons les différentes approches qui ont été développées pour chaque type de régions.

3.2.1 Simulation des habits serrés

La simulation des habits serrés s'effectue de la même façon que la peau, c'est-à-dire avec la méthode de "Skinning". Une étape dans la préparation des habits est nécessaire pour calculer les données d'attachement. Elle se fait en cherchant la surface de la peau la plus proche se trouvant sous les habits.

Bien que très simple, cette technique permet de simuler correctement la déformation des habits moulants. D'autre part, cette méthode est efficace en temps de calcul.

3.2.2 Simulation des habits larges

Les habits larges sont les habits qui se déplacent d'une certaine distance autour d'une même région du corps. Le pantalon est un exemple typique. Nous avons défini un certain nombre d'hypothèses quant aux déplacements possibles de ces habits. Premièrement, ces habits sont en contact avec toujours la même région de la peau. Des mouvements d'habits comme se retrousser les manches ne sont pas considérés. Deuxièmement, le mouvement de ces habits se fait dans un plan qui est perpendiculaire à l'axe du joint du squelette. Les mouvements le long des bras ou des jambes sont interdits. Finalement, l'habit ne se déplace pas au-delà d'une certaine distance de ce point d'ancrage. En nous basant sur ces hypothèses, nous avons développé une méthode ou chaque point de l'habit se déplace sur un demi disque qui est fixé au squelette. Les points sont animés par la gravité seule et une contrainte de collision les maintient sur chacun de leur demi disque. La méthode modifie la taille des demi disques de façon à ce que leur diamètre diminue quand l'habit s'approche de la surface de la peau.

Comme pour le cas de la simulation des habits serrés, cette méthode requiert peu de temps de calcul. La détection de collisions est réduite au seul maintien des sommets de l'habit sur leur demi disque respectif. D'autre part, les sommets ne sont pas liés entre eux par des ressorts. Il n'existe donc pas d'interdépendance des points entre eux; la simulation ne nécessite pas l'implémentation du système "masses ressorts".

De l'autre cote, le résultat remplie les objectifs que nous avions fixés au départ, c'est à dire reproduire correctement les mouvements dynamiques et la forme de l'habit.

3.2.3 Simulation des habits flottants

Les habits flottants sont les habits qui ne sont pas spécifiquement attachés à une région du corps. Dans la plupart des cas, ces habits sont des robes larges qui tombent sur les jambes. La meilleure façon de simuler le mouvement de tels habits est le system "masses ressorts".

La détection de collisions est limitée aux jambes seulement. Leur surface est interpolée par des cylindres. Détecter les collisions revient donc à calculer la distance entre les points du système "masses ressorts" et les axes principaux des os des jambes. La correction des positions s'effectue avec la méthode de dynamique contrainte; les corrections sont effectuées sur les positions, vitesses et forces à chaque étape de l'intégration Euler implicite.

3.2.4 Lissage des points aux frontières des régions

Une étape de lissage entre les régions de l'habit animée avec des méthodes différentes est nécessaire afin d'obtenir une déformation continue entre ces régions. Les positions des sommets se trouvant à la frontière de deux régions sont calculées par les deux méthodes. Ces deux positions sont ensuite combinées avec un poids qui correspond à l'influence de la région sur le sommet.

3.3 Conclusion

Cette approche remplit les objectifs qui ont été fixés au début de cette thèse, c'est à dire la simulation en temps réel d'humains virtuels complètement habillés. L'utilisation des trois méthodes permet simuler la plupart des habits. Les performances en temps de calcul sont très différentes entre ces trois méthodes de simulation. La déformation des habits serrés et large est très rapide. Par contre, l'animation des habits flottants est beaucoup plus lente, elle nécessite en moyenne 100 fois plus de temps de calcul.

Malgré tout, cette approche a un certain nombre de défauts qui sont décrits en détail ci-dessous.

Premièrement, l'étape de segmentation de l'habit ne peut pas se faire de façon entièrement automatique. La forme statique de l'habit ne contient pas suffisamment d'information quant au comportement de l'habit. L'utilisateur doit donc corriger à la main la segmentation calculée par le simulateur. Les inconvénients qui en découlent sont au nombre de deux. Premièrement, l'étape de préparation des habits est beaucoup plus lente puisqu'elle requiert l'action de l'utilisateur. Deuxièmement, il est très rare que l'utilisateur/trice puisse créer un habit du premier coup. En pratique, la création d'un habit nécessite un nombre important d'essais. L'action de l'utilisateur dans la création des habits introduit potentiellement des erreurs.

Deuxièmement, la méthode n'est pas capable de simuler de façon réaliste les plis des habits. Ceci est particulièrement vrai pour la première et deuxième méthode. Simuler ces plis nécessite la modélisation des interactions entre les sommets du maillage de l'habit. L'approche n'est pas aussi capable de simuler des habits spéciaux comportant des parties rigides. Certains types d'habits comme les capes ne peuvent pas aussi être modélisées correctement.

La deuxième approche présentée dans le chapitre suivant doit tenir compte de tous ces points.

4. Déformation basée sur l'interpolation d'exemples

La principale conclusion de la méthode précédente est que l'utilisation d'une forme fixe de l'habit n'est pas suffisant pour préparer les habits de façon totalement automatique. Dans ce chapitre, nous présentons une autre approche basée sur l'analyse d'une séquence pré-calculée de l'habit. Une telle analyse permet d'obtenir des informations beaucoup plus précises quant au comportement de l'habit.

La méthode présentée dans ce chapitre est basée sur une classe d'approches appelée l'animation basée sur l'interpolation d'exemples. Cette classe d'approches est apparue très récemment et a été utilisée avec succès pour l'interpolation de formes géométriques (déformation de la peau, création de corps de différentes tailles…) et l'animation du squelette. La principale idée de cette méthode est d'interpoler les données (animation du squelette ou forme géométrique) à partir d'exemples créés par l'utilisateur. Différents types de fonctions peuvent être utilisés pour l'interpolation, comme l'interpolation linéaire ou RBF…

La principale difficulté pour la conception d'un tel interpolateur est de définir la dimension de l'espace d'interpolation. Un espace d'interpolation dont la dimension est trop petite offre peut de possibilités d'interaction pour l'utilisateur. Au contraire, en définissant un espace d'interpolation trop grand, il devient nécessaire de fournir beaucoup plus d'exemples pour que le calcul d'interpolation se fasse correctement.

Une des contributions de notre travail est l'adaptation des déformations basées sur l'interpolation d'exemples au cas des habits. Contrairement aux objets de déformation géométriques, la forme des habits dépend d'un nombre important de paramètres qui sont les positions et les vitesses de tous les sommets du maillage de l'habit. Il semble donc qu'à première vue, la simulation des habits est difficilement adaptable aux déformations basées sur l'interpolation d'exemples.

Pour résoudre cette difficulté, nous allons faire une analyse détaillée du comportement de l'habit. Nous pouvons identifier deux différents niveaux de déformation.

- Le mouvement global: un vêtement animé d'une certaine vitesse va continuer sur sa lancée quand l'avatar qui le porte va s'arrêter de bouger. La simulation dynamique est la meilleure façon de modéliser ce type de mouvements. Dans notre cas, elle sera faite avec un maillage contenant un nombre réduit de points.
- Les détails de l'habit: ces détails sont principalement les plis. Ces plis apparaissent et disparaissent en fonction des contraintes locales. Plusieurs chercheurs ont noté que ces plis ont un comportement géométrique et ne nécessitent donc pas l'utilisation de méthodes "masses ressorts". Pour ces déformations, la déformation basée sur l'interpolation d'exemples est un bon choix.

La méthode est décrite en détaille dans la suite du chapitre. La description de la modélisation du comportement dynamique de l'habit est donnée dans la première section. La deuxième section décrit la méthode pour générer les détails de l'habit. Le chapitre est clos par une discussion sur les performances de cette approche.

4.1 Simuler le comportement dynamique

La modélisation du comportement dynamique est faite à l'aide d'un système "masses ressorts". Le nombre de point de ce système est limité puisque l'objectif est de pouvoir calculer les déformations en temps réel. Il est donc nécessaire de construire un modèle simplifier du maillage de l'habit.

4.1.1 Segmentation du maillage

La segmentation du maillage permet de construire une version simplifiée de ce maillage. La segmentation consiste à sub-diviser le maillage en patchs (ensemble de triangles interconnectés). Chaque patch forme un sommet du maillage simplifie. Cette segmentation est faite suivant les étapes décrites ci-dessous:

- Un sommet sélectionné sur le maillage. Ce sommet constitue le premier point du patch.
- Le patch est agrandi par l'ajout successif des sommets voisins au patch. La sélection du sommet à ajouter se fait à l'aide d'une fonction "coût" qui est calculée pour tous les sommets candidats. C'est le candidat dont le coût est le plus faible qui est choisi.
- Le grossissement du patch est termine quand le coût le plus faible calculé parmi tous les sommets candidats est supérieur à une certaine valeur.

Cette fonction "coût" permet d'attribuer une valeur à chaque sommet candidat en fonction de trois critères décrits ci-dessous.

- Ce maillage sera utilisé par le système "masses ressorts" pour simuler le comportement dynamique de l'habit. Selon les travaux de recherches de [VOLI 02], le maillage doit avoir une structure la plus régulière possible. Autrement dit, les arêtes qui composent ce maillage doivent être de longueur pratiquement égale. La fonction coût intègre ce critère. Elle pénalise les patchs dont la forme s'éloigne du disque.
- La fonction "coût" intègre aussi une composante qui permet de contrôler la taille des patchs. Les objectifs sont doubles. Cette composante permet d'obtenir des patchs de tailles égale, ceci est une

des conditions pour obtenir un maillage régulier. Deuxièmement, l'utilisateur peut contrôler facilement le nombre de sommet qui sont animés par le système "masses ressorts".

- Finalement, une troisième composante permet de favoriser le regroupement de sommets dont les déplacements des uns par rapport aux autres sont faibles. Cette contrainte permet de construire un maillage simplifié dont les déformations sont les plus représentatives de celles du maillage original.

4.1.2 Préparation de la déformation du maillage simplifié

La détection de collisions est une des raisons de la lenteur de calcul des systèmes "masses ressorts". Nous proposons une méthode qui permet de détecter les collisions de façon très rapide en effectuant une analyse de la séquence pré-calculée de l'habit à simuler.

L'observation de la déformation d'un habit sur un corps humain nous permet de décrire deux types déformations. Quand l'habit est serré, le tissu suit exactement la déformation de la peau; sa forme peut être générée de la même façon que la peau. A l'opposé, quand le vêtement est très large, son contact avec le corps est très réduit; son mouvement est principalement contrôlé par les propriétés mécaniques du tissu et la gravité. Dans la pratique, le mouvement de la plupart des vêtements est le résultat d'une combinaison de ces deux types de déformations. Notre analyse consiste à identifier l'importance de ces deux composantes. Si nous arrivons à détecter les parties moulantes de l'habit, les temps de calcul nécessaire pour la simulation peuvent être réduits; la déformation des habits moulants ne nécessite pas un système "masses ressorts".

Les deux composantes sont identifiées en essayant d'ajuste une surface déformée par la méthode de "skinning" à la surface de l'habit sur toute la durée de la séquence pré-calculée. Cette méthode a déjà été utilisée pour reconstruire les données de skinning d'un modèle dont plusieurs poses ont été dessinées par un artiste. La reconstruction est en fait un problème de minimisation qui peut être résolue par régression linéaire. Notre cas est plus compliqué que [MOHR 03] parce que nous utilisons une version modifiée du "skinning". La minimisation n'est pas linéaire, nous avons du remplacer la régression linéaire par la méthode "Powell" qui est plus lente. Cette lenteur n'est pas un problème majeur puisqu'elle ne concerne pas l'animation temps réel.

En plus des données de "skinning", cette étape nous permet aussi de connaître la qualité de l'ajustement en analysant la valeur résiduelle de la minimisation. Une faible valeur résiduelle indique que la déformation de "skinning" permet de simuler correctement la déformation de la peau. Au contraire, une valeur résiduelle importante permet de conclure que l'animation du vêtement est influencée par d'autres contraintes qui sont la gravité et les propriétés mécaniques du tissu.

Grâce à la méthode de "skinning" que nous utilisons, nous pouvons calculer le repère local associé à chaque sommet du maillage. Et dans ce repère local, nous pouvons calculer les déplacements locaux de l'habit qui sont due à l'imperfection de l'ajustement des données du "skinning". Ces déplacements locaux correspondent aux effets de la gravité et des propriétés mécaniques du tissu. Les positions locales sur toute la durée de la simulation pré-calculée forme un nuage de point. Nous considérons que ce nuage de points correspond à toutes les positions que peut occuper l'habit pendant la simulation. L'enveloppe convexe de ce nuage est calculée et utilisée pour la détection de collisions en temps réel.

4.1.3 Simulation en temps réel du maillage simplifié

La simulation en temps réel du maillage simplifié est faire à l'aide d'un system masses ressorts. Chaque sommet du maillage est associé à une coque convexe calculée lors de l'étape de préparation du modèle. Cette coque permet de garder les sommets au squelette.

4.2 Simuler les détails du vêtement

Les détails du vêtement sont générés avec une méthode basée sur l'interpolation d'exemples. Nous avons vu que l'utilisation d'une surface paramétrique nécessite de la part de l'utilisateur de définir les paramètres de déformation. L'utilisation d'exemples, c'est à dire la simulation pré-calculée, permet de définir tous les paramètres de façon entièrement automatique.

4.2.1 Préparation de la déformation du maillage détaillé

Le maillage détaillé, c'est à dire celui qui est affiché à l'écran, est calculé à partir du maillage simplifié dont chaque sommet forme un point de contrôle. La déformation est faite à l'aide de fonction d'interpolation qui, étant donné la position des points de contrôle, va calculer la position des points du maillage final. Cette fonction est une fonction polynomiale du premier degré qui est construite à l'aide de l'animation pré-calculée par régression linéaire. Nous aurions pu utiliser d'autres méthodes d'interpolation comme les RBF. Mais nous avons trouvé que les fonctions polynomiales, malgré leur simplicité, sont performantes pour la déformation du maillage final. De plus, elles sont extrêmement rapides à calculer.

4.2.2 Déformation en temps réel du maillage final

Apres avoir calculé la déformation du maillage simplifié, le maillage final est généré par l'évaluation des fonctions d'interpolation. Le temps de calcul requis pour cette étape est en moyenne dix fois plus faible que la déformation du maillage simplifié.

4.3 Conclusion

Comparé à l'approche basée sur l'analyse du maillage statique de l'habit, cette méthode a de nombreux avantages. Premièrement, l'étape de préparation est entièrement automatique. L'analyse simulation pré-calculée permet de connaître en détail le comportement de l'habit. Deuxièmement, son champ d'application est beaucoup plus large; elle peut simuler une plus grande gamme d'habits. La raison de cela est que le simulateur est capable d'apprendre le comportement de l'habit, quelles que soient ses propriétés mécaniques.

5. Etude de cas: Le "Virtual Try-On"

Le Virtual Try-On est une application dédiée à l'essayage virtuel d'habits. L'utilisateur/trice entre les mesures de son corps, choisit un habit dans la collection du Virtual Try-On et peut ensuite voir un mannequin de sa taille se déplacer en temps réel avec les habits. Cette application intègre plusieurs technologies: la génération de corps à partir de mesures, la déformation et le changement de dimensions des habits

5.1 Génération de corps à partir de mesures

Cette technologie a été développée par [SEOH 03]. Cette méthode est basée sur l'interpolation de corps réels scannée qui ont été préalablement préparée. Ces corps scannées sont arrangés suivant leurs tailles dans une base de donnée. Le corps généré peut être immédiatement animé. Les données de "skinning" sont définies dans la base de données.

5.2 Déformation en temps réel des habits

La déformation en temps réel des habits est calculée avec la première méthode (Déformation hybride basée sur la géométrie et la physique). Cette méthode est basée sur la segmentation de l'habit en trois régions: serrée, large et flottante. Chaque région est animée par une méthode qui lui est dédiée. Le principal avantage de cette méthode est l'implémentation assez facile du module de changement de dimensions des habits.

5.3 Adaptation des habits aux changements de la taille du corps

Les changements de dimensions des habits sont effectués différemment sur les trois régions de l'habit. Les parties flottantes qui sont les jupes et les robes, sont redimensionnée en fonction de la taille des jambes. Les points d'encrages des régions serrées et larges sont déplacés avec la déformation de la peau due aux changements de dimensions du corps.

5.4 Conclusion

Le Virtual Try-On est exemple d'application de la simulation des habits en temps. Elle donne quelques solutions pour l'intégration de plusieurs technologies afin de produire une application finale qui peut être utilisée par le grand public.

6. Conclusion

Dans cette dissertation, nous avons décrit deux méthodes pour la simulation en temps réel des habits. Ce travail est original sur plusieurs points.

Premièrement, nous avons proposé de simuler l'habit non pas de façon autonome, mais comme étant attachée au squelette de l'avatar qui le porte. Cette limitation n'est pas très contraignante puisque par définition, les habits sont souvent portés. D'un autre coté, cette hypothèse permet de simplifiée le modèle mécanique et la détection de collisions.

La deuxième contribution de ce travail découle directement de la première contribution. Comme l'habit est attaché au squelette, nous avons pu classer les sommets de l'habits par leur comportement, c'est à dire suivant que l'habit est serré, large ou flottant. Cette segmentation est permit de développer des méthodes spécialisées pour chaque comportement. Nous avons vu dans le chapitre consacré à l'état de l'art que les méthodes basées sur des hypothèses fortes sont beaucoup plus rapides que les méthodes polyvalentes.

Troisièmement, nous avons montré que la déformation basée sur l'interpolation d'exemples peut être adaptée au cas de la simulation des habits. Cette classe de déformation est normalement appliquée pour les animations de type géométrique. L'animation des habits dépend d'un plus grand nombre de paramètres, rendant le problème beaucoup plus complexe. La solution que nous avons proposée a été de simuler le comportement dynamique avec un système "masses ressorts". Quant aux détails, ils sont générés par l'interpolation d'exemples.

Enfin, nous avons proposé une approche qui permet de simuler une grande variété d'habits. Le simulateur basé sur l'interpolation d'exemples est capable d'apprendre le comportement de l'habit grâce à l'analyse d'une séquence pré-calculée de l'habit.

Ce travail de recherche a néanmoins plusieurs limitations qui sont décrites ci-dessous.

Le premier défaut de cette méthode est son impossibilité de calculer la déformation de l'habit sous l'action d'objets autres que le corps humains. Il est par exemple impossible de simuler une robe qui touche le sol. Un futur travail de recherche pourrait être consacré à ce problème.

Le deuxième défaut concerne la préparation des habits dans le cadre de la méthode basée sur l'interpolation d'exemples. Dans l'implémentation courante, la préparation des habits est basée en grande partie sur la régression linéaire. Or, la régression linéaire est particulièrement sensible aux bruits. Afin d'obtenir une bonne analyse, il est nécessaire de travailler sur une séquence pré calculée suffisamment longue. Dans la pratique, il faut au moins 300 images clés, c'est à dire dix secondes d'animation. Un futur travail de recherche pourrait être consacré à remplacer la régression linéaire par une méthode d'analyse statistique plus solide.

D'autres extensions sont possibles. Il serait par exemple intéressant d'implémenter une technique de niveaux de détails pour la déformation des habits. Les nivaux de détails sont couramment utilisés pour l'animation temps réel. Un autre travail pourrait être le développement d'un simulateur de déformations dynamique basées sur le langage machine des cartes graphiques. Plusieurs travaux ont été publiés dans ce domaine et ont démontré que le gain de temps de calcul peut être important. Un troisième travail possible pourrait être l'adaptation de ces approches au cas de l'animation d'autres objets de déformation dynamique comme les cheveux.

Table of contents

List of Figures

List of Tables

Chapter I. Introduction

Research on cloth simulation has been started since few decades [WEIL 86] [TERZ 87] [TERZ 88] [LAFL 91] [CARI 92]. A tremendous work has been done in this field, yielding to many publications and successful applications. Nowadays, many commercial packages propose efficient cloth simulators. These animation techniques have been successfully used in several animation films such "Monster & Co." from Pixar [PIXA02] or "Shrek" from DreamWorks [DREA02], to cite few of them. If we look back through the history of computer graphics, we can see that the simulation of clothes has been developed after the simulation of skin and skeleton. The evolution of research on real-time simulation goes almost the same way, apart that the history of real-time deformations is much shorter. Nowadays, research on real-time skin deformation has reached a mature stage [KRYP 02] [LEWI 00] [SLOA 01] [ALEN 02] [MOHR 03] and very recently, research works on real-time clothes are coming to publications [CORD 02] [CORD 03] [VASS 01] [KANG 01] [JAME 03].

1. Problematic

Clothes have a dynamic behaviour. A cloth moving at a certain velocity will continue on its trajectory unless it is stopped by external constraints such as collision or friction. The shape of garments cannot be defined as a function of the only underlying skeleton unlike, for instance, the skin. Like every natural phenomena such as smoke, hair, or water, the shape of cloth depends on its past trajectory. Indeed, the cloth simulation is a function of a high number of input parameters, leading to highly computation intensive methods. That is the main reason why most of cloth simulators can hardly achieve real-time performance.

The problematic of real-time simulation is to find a good compromise between accuracy and computation speed. In general, accurate methods tend to be slower than methods based on approximations. In other words, the question is how to define an approach that can produce real-time performance while maintaining satisfying results. There are at least two strategies to limit the computation cost.

The first strategy consists of reducing the complexity of the model. This can be achieved by reducing the number of polygons of the mesh representing the cloth or using a simplified mechanical model for the cloth simulation. Such simplifications naturally lead to degraded simulation results. The model will lose its accuracy and some of cloth features such as wrinkles will not be animated correctly.

The second way of reducing the computation time is to make certain assumptions on the garments. Typically, one can consider the clothes in particular situations, such as dresses worn on virtual humans. In this way, tight garments can be simulated with a geometric method rather than mass-spring system and full collision detection. Such simplification naturally limits the versatility of the simulation. If we considerer that tight clothes can be animated with geometric deformation, they are assumed to remain tight throughout the simulation. For instance, dressing or undressing will not be possible. Another example that belongs to this category is the work presented by James et al. [JAME 03] where the cloth simulator can learn the cloth behaviour through a pre-computed sequence of cloth movements. During the run-time, the simulator plays the animation by interpolating the data acquired during the pre-processing stage. The resulting simulation is highly realistic but the method can only be used for simulating piece of tissues; the animation of clothes on bodies is not possible using this approach. Another drawback of these approaches is that it requires very often a pre-processing stage where the garment is analyzed and put into a form that can be used for real-time performance. On the other hand, these approaches do not automatically imply the degradation of the quality of the simulation.

In our choice of cloth simulation method and optimization strategy, we have first to identify the reasons of the slow computation of cloth simulation. More and more sophisticated mechanical models have been developed over years to increase the realism of the simulation and calculation speed. Despite the immense amount of previous work, the computation speed of existing cloth simulators does not allow real-time animation. There are two reasons that can be brought out. The collision detection still remains a complex task due to the complexity of body geometry and the self-collision on highly deformable garment surfaces. The second reason of slow computation is the simulation of the physical properties of the garments.

The research on real-time clothes should address these two limitations. Collision detection has to be simplified and optimized; the mechanical model has to be rewritten with the real-time application in mind. We focus our study on the simulation of dressed clothes, which is a special case of cloth simulation. The simulation of garments on bodies has less freedom in its motion in comparison with a piece of garment taken alone. A very tight trouser has the shape of the underlying skin and therefore can be simulated as the skin surface is. If we consider the case of loose clothes, the tissue still follows the movements of the skeleton. We believe that optimization on cloth simulation can be done by considering the simulation of the garments as an object anchored to skin surface rather than as an independent object. With this approach in mind, many optimizations can be done. Collision detection can be limited to the garments and its underlying skin. The mechanical model can be simplified since the garment surface is linked to the skin surface. On the other hand, the versatility of cloth simulation is partially lost as the simulation is based on the underlying skin. The research presented in this thesis aims to develop methods by taking advantage of using the underlying skin.

2. Motivating applications

The real-time simulation of clothes has a great potential. In fact, real-time clothes can be used wherever virtual humans are needed. Nowadays, many applications make use of virtual humans such as computer games, tele-presence in virtual worlds, augmented reality, and training platforms to cite few of them. Currently, these virtual humans are animated with simple geometric deformation of their external envelope. Garments and skin surface are deformed in the same way by attaching them rigidly to the skeleton.

Even though the simulation is lacking of realism, this technique is used very often because of the wide demand of real-time visualization of virtual humans. The possibility to simulate real-time cloth will make all these application much more realistic. We can say that the real-time simulation of clothes is the continuation toward more realistic and sophisticated virtual humans.

MIRALab – University of Geneva has a long tradition in physically based simulation, especially cloth animation. This research work [LAFL 91] [CARI 92] [YANG 93] [THAL 94] [VOLI 95] [VOLJ 00] [VOLK 00] [VOLI 00] [VOLI 01] [VOLI 02] [VOLI 03] is one of the major areas in MIRALab since its foundation and it has been mainly sponsored by the FNRS (Swiss National Science Foundation). This research on clothes has naturally been extended to the case of real-time simulation. Cloth simulation has become an integral component of the platform dedicated to the animation of virtual humans in real-time.

3. Objectives of this research

The principal objective of this research is to develop methods for real-time cloth simulation. A perfect simulator would give highly realistic simulations of any kind of clothes in real-time. This is of course not possible with existing graphics hardware. Even now, researchers are continuously improving mechanical models to achieve quality simulation within less computation time. This applies to both non real-time (not limited by the real-time constraint) and real-time simulations. In the following, we list a set of criteria sorted by an order of importance. These criteria will help us to drive our research and to evaluate our results.

- *Low computational cost*. Developing a simulation method for real-time is obviously the first objective. Unlike other physically based simulations, real-time simulation requires searching for specific strategies to maintain low computation cost. A good deal of research efforts has been especially devoted to real-time graphics application, such as level of detail, scene culling, etc. Indeed, real-time physically based simulation can be considered as a research topic on its own. In our case, we look for approaches that will use the underlying skin, which brings us naturally to the second objective.
- *Application to the simulation of virtual humans*. Throughout this dissertation, we limit our research scope to the simulation of garments worn on virtual humans. Three main reasons can be noted. Firstly, most of clothes are used as garments dressed by virtual humans. The need of simulating garments taken alone such as flag or tablecloth is relatively rare in practice. Secondly, on the other hand, the simulation of dressed humans with existing methods is one of the most time consuming tasks because it involves intensive collision detection and response. The mechanical model should also be accurate enough to avoid unrealistic stretching when the virtual human starts to move. Finally, we believe that efficient methods can be developed by taking advantage of this presumed condition. It is felt that existing techniques have matured enough so that the computation time cannot be further reduced significantly. Still, exploring new strategies for dressed virtual humans remains a challenge, which is one our goals.
- *Stability*: The stability of the cloth simulation is one of the issues to be taken into consideration. Very often, the smoothness of the virtual human movements cannot be presumed. This is especially true when the movement of virtual human model is provided on-line by motion capture systems. In such cases, the simulator should still be able to maintain stable simulation although the row data from the motion capture system presumably contain noises.
- *Realistic simulation*. Many research works have been devoted to improving this aspect of the simulation. New mechanical models and numerical solvers have been developed for simulating accurately the stretching, bending, and other fabric properties. It is clear that real-time simulations will not reach the level of realism shown by these simulators due to the limited computation time. However, one of the objectives is to approximate the quality of these simulators. Later in this document, we discuss real-time methods that can simulate complex cloth features such as wrinkles, pockets…

4. Overview of our approach

This dissertation proposes two different approaches for the real-time simulation of clothes. One common point of these approaches is that they both make use of the underlying skeleton of the character. As mentioned in the previous section, we have focused our investigation on cloth deformation controlled by skeleton or skin. Rather than being a freely moving object independently from its support, the garment is anchored to the underlying skeleton or skin. In other words, the garment is considered as a second "skin" layer. By further analyzing the movements of the cloth vertices in relation to their anchorage, we can assign a grade to each of these vertices depending on whether the vertex behaves more like a skin or a cloth. Figure I-1 shows such analysis of the cloth vertices on a trouser model. This analysis has been made on a pre-computed sequence of cloth animation that has been calculated with our in-house software. The animated cloth could also be generated with Maya Cloth or any other commercial graphic packages. The analysis method will be described in detail in Chapter V. 'Tight' regions appear in blue while 'loose' regions are coloured in green and red. The collision detection and the mechanical model have been adapted to give to this second 'skin' layer the behaviour of clothes.

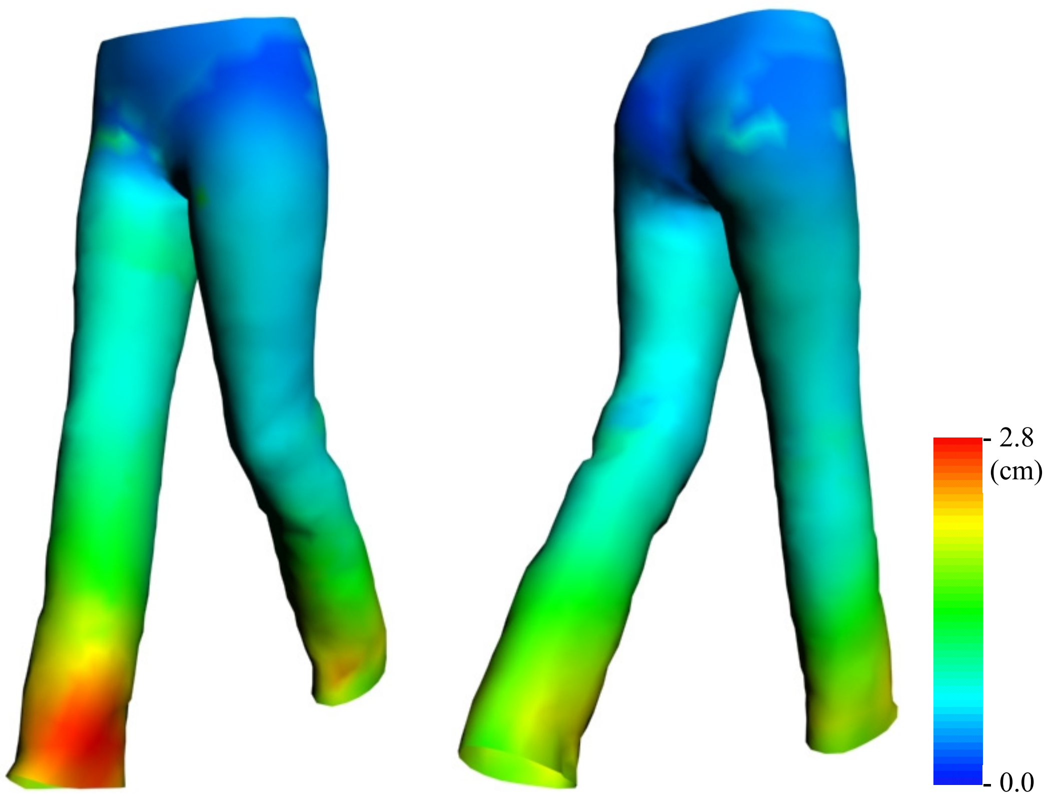


Figure I-1: Analysis of the vertex movements on a trouser

The reason of this choice is that many optimizations can be done. For instance, the collision detection problem returns to computing the intersection of the cloth vertices and their anchorage on the skin. The mechanical model can also be optimized by replacing the mass-spring system with a faster method. For instance, blue regions can be computed with a geometric method similar to the one used for skin deformation.

4.1. Hybrid approach using geometric and physically based simulation

The first approach works in a hybrid manner exploiting the merits of both the physically based and geometric deformations. It makes use of predetermined conditions between the cloth and the body model, avoiding complex collision detection and physical deformations wherever possible. Garments are segmented into pieces that are simulated by various algorithms, depending on how they are laid on the body surface and whether they stick or flow on it. Tests show that the method is well suited to fully dressed virtual human models, achieving real-time performance compared to ordinary cloth-simulations.

4.2. Example-based approach

The main idea of the second approach is to develop a cloth simulator that can learn the cloth behaviour through a sequence of pre-computed cloth simulation. Unlike the first approach where a good deal of simplification takes place, this approach enables us to simulate more complicated garment features such as wrinkles, pockets… at a real-time performance. We setup the problem as simulating the garments in two stages. At the first stage, a rough mesh reproduces the dynamic behaviour of the garments. Its physical properties are defined with the pre-calculated sequence. At the second step, the fine mesh simulates the details of the garments.

The main difference between these two approaches is the strategy for reaching real-time performance. The first approach resorts to a hybrid approach combining physically based and geometric deformation. The real-time performance has been made possible by reducing the complexity of the cloth mesh and the mechanical model. The second approach makes use of an example-based method. This approach delivers simulations of higher quality but requires a pre-computed simulation sequence of the clothes to be animated.

5. Organization of this document

In Chapter II, we examine previous works that have been leaded for real-time cloth deformation. We also provide a comparative analysis, pointing out advantages and drawback of each of these methods. We also detail the different strategies followed by researchers to make real-time cloth simulation possible.

Chapter III gives the description of the prerequisite technologies that have to be developed prior to cloth simulation. These include the development of an efficient skeleton driven deformation method and a platform for designing and computing the simulation of the garments. Simulating the garments is a necessary step as we use it in one of our method for real-time cloth to train the cloth simulator.

Chapter IV describes the first method of real-time cloth simulation. In the first section, we detail the pre-processing stage where the garments are put into a form that can be used for real-time simulation. A second section gives details of the core of the cloth simulation method. Finally, we close this chapter by giving the results of several tests done on this method.

Chapter V introduces our second approach for real-time cloth simulation. Unlike the first method, this method makes use of examples that have been pre-computed.

Chapter VI gives the description of an application of real-time cloth simulation, the Virtual Try-On. The Virtual Try-On is a Web application where the user can give his/her measurements and try-on clothes in real-time.

Finally, we close this dissertation with the Chapter VII where we describe possible directions for future research on real-time cloth.

Chapter II. Related work

Extensive research works have been carried out in cloth simulation since several decades. Many of them have focused on the quality of garment simulation, where the need of time-critical simulation was not a priority. Their aim was to develop physically based methods that allow simulating the dynamics of cloth independent of its use ─ whether as clothing or other situations like in furnishing tablecloth. They integrated complex collision detection in order to be able to simulate the physical behaviour of garments. MIRALab ─ University of Geneva has a long tradition in cloth simulation [LAFL 91] [CARI 92] [YANG 93] [THAL 94] [VOLI 95] [VOLJ 00] [VOLK 00] [VOLI 00] [VOLI 01] [VOLI 02] [VOLI 03]. The Chapter III.4 gives the details of the tools developed by MIRALab.

Figure II-1: Accurate animation of virtual garments [VOLJ 00]

Many other researchers have also worked in this area, resulting in plenty of publications [TERZ 87] [AONO 90] [BREE 94] [EBER 96] [BARA 98] [BRID 02] [CHOI 02]. The timeline of cloth research is shown on the Figure II-2.

Legend
BLUE: Geometric approaches
RED: Physically based approaches
GREEN: Hybrid approaches combining geometric and physically based approaches
ITALIC: Methods specifically created for real-time application

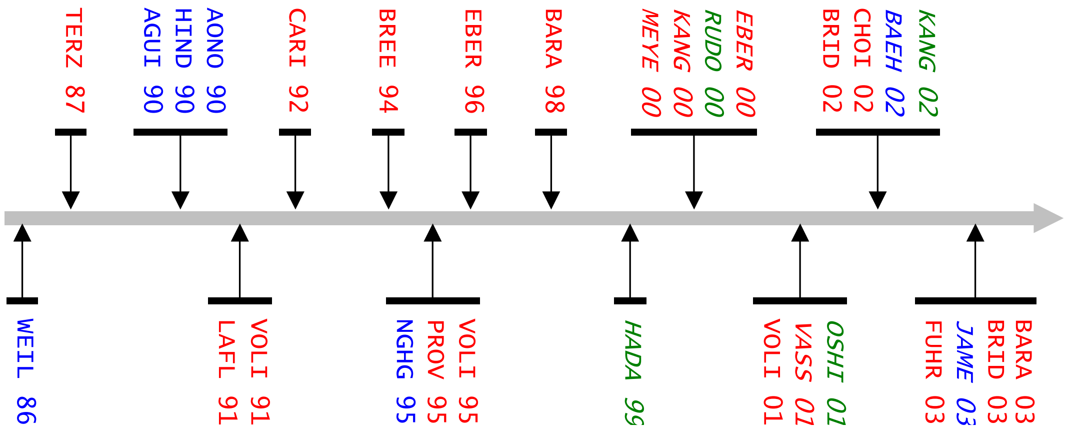


Figure II-2: Timeline of the research works for cloth simulation

Later, commercial packages have been developed, bringing this technology to market [MAYA 03] [HAVO 03] [STIT 03] [SIMC 03] [CREY 03] [SYFL 03] and several animation films have successfully used these proposed animation techniques — "Shrek" from DreamWorks [DREA 03], "Monster & Corporation" from Pixar [PIXA 03], to cite few of them.

Figure II-3: Shrek by DreamWorks [DREA 03]

However, it is our intention to restrict our survey to research for real-time clothes and the real-time cloth simulation remains the main subject of our work here. In the next sections, methods targeted to real-time clothes are reviewed in detail. Some other methods, not specifically developed for real-time clothes, are discussed as well because they have brought some ideas/methods that are useful for the case of real-time clothes.

With the recent advent of computation power of graphics hardware, the real-time simulation of simple clothes has become possible recently. The research on real-time cloth simulation in the true sense has started few years ago, with the simulation of simple garments such as flags or tablecloths. Nowadays, the research on clothes tends to provide higher quality and faster computation. We have classified these methods into four categories, each category corresponding to one section of this chapter.

The first section presents methods based on physics. The benefit of these approaches is the quality of simulation, allowing for the animation of complex cloth features such as wrinkles. Their main drawback is the computation speed. In this section, we also give a survey on collision detection.

Another category covers methods that are based on geometric deformation, which provide much faster result than the physically based ones. Unfortunately, they are not appropriate for simulating timely the movements of the clothes. They can be rather considered as advanced modelling tools to design garments.

The third category is the hybrid method, aiming at combining the benefits of physically based and geometric simulators. Most of the approaches belonging to this category make use of garments that are composed of two layers. Typically, a rough mesh is used to simulate the dynamic behaviour of the clothes with a physically based method while a fine mesh generates the details of the clothes with a geometric method.

The fourth category concerns the example-based approaches applied to the cloth simulation. Example-based approaches have been introduced in Computer Graphics very recently and show promising results in wide range of graphical objects.

Finally, we conclude this chapter by making a comparative analysis, comparing the advantages and drawbacks of each category. Other real-time physically based animators whose purpose is not simulating garments are also described. We introduce these techniques because we believe that they may inspire new ideas for the case of cloth application areas.

1. Physically based simulation

Cloth can be considered as passive objects. Unlike humans who exert forces to control their motions, the movements of clothes are due to external forces and internal reactions defined by the fabric properties. It means that the animation of a cloth is relatively well defined if we know the external forces acting on the clothes and if we have a good model of its mechanical reaction. The simulation of such behaviours is best accomplished with physically based methods.

The diversity of cloth behaviours results from its mechanical properties. Researchers have characterized several of them such as resistance to bending, stretching, shearing and compression. Mathematically, these properties can be described with the following partial derivative equation:

$$\ddot{x} = M^{-1}\left(-\frac{\partial E}{\partial x} + F\right) \tag{1}$$

The equation 1 is simple the rewriting of Newton's Second Law $\sum F = m\ddot{x}$ where the scalar *E* is the internal energy, *M* the inertia matrix, and *x* the state of the cloth. The internal forces are defined as a gradient of the energy. All external forces are contained in *F*.

Physically based simulations involve the computation of the internal forces and of solving the equation 1. The most commonly used technique to discretize equation 1 is to use particle system. The garment surface is decomposed into vertices connected together by edges, forming a spring mesh. The energy computation is defined for each vertex as a function of the vertex and its neighbours. The equation 1 then becomes a system of ordinary derivative equations. The animation is computed iteratively, using at every step the current state (position and velocity) of the vertices. The computation of one iteration is made by first evaluating the forces that act on the vertices and then calculating the new position of the vertices using a numerical solver. Almost all physically based methods follow this scheme. They only differ in ways how forces are calculated and the formulation of the numerical solver (i.e. integration method).

In the following sections, we use the above notations:

- The current time and time step are respectively denoted by *t* and *h*.
- x_i^t, v_i^t, a_i^t and m_i are respectively the position, velocity, acceleration and mass of the ith particle of the cloth mesh at time t. We also define $x^t = \left[x_1^t, x_2^t, \ldots, x_n^t\right]^{\mathrm{T}}$ and $v^t = \left[v_1^t, v_2^t, \ldots, v_n^t\right]^{\mathrm{T}}$.
- The state of the cloth mesh is the position and velocity of all the cloth particles: $Q^t \equiv \left(x^t, v^t\right)$.
- The sum of the forces exerted on the ith particle at time t is written as F_i^t. $F^t = \left[F_1^t, F_2^t, \ldots, F_n^t\right]^{\mathrm{T}}$.

1.1. Modelling of the internal forces

The modelling of the internal forces plays a crucial role as it directly influences the behaviour of the garments. A tremendous amount of work has been made to improve both stability and realism of garment animations. The simplest way of modelling internal forces is to consider every edge of the cloth mesh as a spring. Computing the internal forces on a vertex comes down to computing the contribution of all springs that are connected to it. The force exerted by a spring joining two masses *i* and *j* is

$$F_{(i,j)} = -k_{i,j}\left(\left\|x_i - x_j\right\| - l_{i,j}^0\right)\cdot\frac{\left(x_i - x_j\right)}{\left\|x_i - x_j\right\|} \qquad (2)$$

where $l_{i,j}^0$ and $k_{i,j}$ are respectively the rest length and the stiffness of the spring. The mass-spring system is far from delivering realistic cloth deformation because it often fails to simulate accurately high elongation and high compression [VOLK 00].

More sophisticated models [VOLI 01] [BARA 98] [CHOI 02] [BRID 02] have been developed for simulating other cloth properties such as bending and shearing. However, these mechanical models are hardly adaptable to real-time simulation because they are slow at computation. Despite its weaknesses, the simple spring-mass system is still preferably used for real-time application because of its high computation speed.

1.2. Numerical solvers

There are two main classes of numerical solvers. The first class is called explicit Euler method, which is also known as forward integration. The second class is the implicit integration method.

The explicit Euler method computes the next state of the clothes at time *t+h* by evaluating the forces at the time *t*. Formally, the Explicit Euler Integration method is described by the following equations:

$$v_i^{t+h} = v_i^t + F_i^t \frac{h}{m_i} \qquad (3)$$

$$x_i^{t+h} = x_i^t + v_i^{t+h} h \qquad (4)$$

The equation is explicit because v_i^{t+h} can be written explicitly as a function of v_i^t and x_i^t. Based on this explicit scheme, a family of methods has been developed, the well-known Runge-Kutta method [PRES 88] being one of them. The idea of Runge-Kutta methods is to approximate the function at t_{n+1} with a series of points from the interval $[t_n, t_{n+1}]$. The forward Euler method is the first order of Runge-Kutta methods. Higher order methods have also been used such as the midpoint (2nd order) and the 4th order method. Fourth order Runge-Kutta with a self-adjusting step size has proven to be accurate for cloth simulation. However, like other explicit methods, this method suffers from its instability when using large time steps. Other methods that belong to the explicit integration have also been developed, such as the Burlish-Stoer [VOLK 00] and the Euler Half-step methods [PRES 88]. The Explicit Euler approach has been widely implemented by researchers [PROV 95] [VOLI 95] [FUHR 03].

The second class called Implicit Integration or Backward Integration has been developed to overcome the weaknesses of the explicit integration. Unlike the explicit integration, this numerical solver makes use of the forces at time *t*+*h*. It performs the computation not by using the derivative at the current time *t*, but by using the predicted derivative for the next time step. Such assumption leads to a linear system to be solved. This will be explained in greater detail in the next section.

$$v_i^{t+h} = v_i^t + F_i^{t+h} \frac{h}{m_i} \tag{5}$$

$$x_i^{t+h} = x_i^t + v_i^{t+h} h \tag{6}$$

Hauth et al. [HAUT 01] have compared the efficiency of the explicit and implicit integration methods in terms of stability and computation time by using the Dahlquist's test equation

$$\begin{cases} x(t)' = \lambda x(t) \\ x(0) = x_0 \end{cases} \quad \text{with } \lambda \in \mathbb{C}. \tag{7}$$

An integration scheme is called stable if it delivers a bounded solution for the condition $\mathrm{Re}(\lambda) < 0$. The authors found that the explicit and implicit integration method give a bounded solution only if $|1 + h\lambda| < 1$ and $|1 - h\lambda|^{-1} < 1$, respectively, with *h* being the time step. For explicit methods, if $\mathrm{Re}(\lambda) \to -\infty$, *h* should as well tend to 0 in order to keep resulting in stable solution. Compared to implicit methods, the step size should be kept small in explicit methods, and thus make the computation speed of the numerical solution much slower owing to the need of a higher number of simulation steps. This implies that implicit integration methods are better approaches for real-time cloth simulation.

One of the most notable methods that belong to the implicit scheme is Adams-Moulton family method [BURD 93]. The Backward Euler [PRES 88] and Trapezoid [PRES 88] methods are respectively the first and the second order of this family. The Backward Euler method, which is also known as Implicit Euler Integration method, is perhaps the most widely used. The backward differentiation formula (BDF) [HAUT 01] has also been used to produce high quality cloth animations.

In the two following sections, several methods based on the explicit and implicit schemes are described in detail. In the last section, the description of a hybrid approach combining implicit and explicit Euler methods is given.

1.2.1. The explicit Euler integration method

Several researchers have worked on the explicit Euler method to resolve the problem of its instability with stiff systems. We will discuss in this section the two main methods: the Verlet integration and the Inverse Dynamics.

1.2.1.1. The Verlet integration

The Verlet integration [VERL 67] [MELI 02] [KACI 03] is a simple integration scheme known to be very stable. The update formula of the Verlet integration is based on the Taylor series approximation. $f(t)$ denotes the function giving the positions x^t of the cloth mesh. The approximation of $f(t)$ given by the Taylor expansion is:

$$f(t) \approx f(t_0) + (t-t_0)f'(t_0) + \frac{1}{2!}(t-t_0)^2 f''(t_0) + \frac{1}{3!}(t-t_0)^3 f'''(t_0)$$

$f(t)$, $f'(t)$ and $f''(t)$ are respectively x^t, v^t and a^t. The approximations of x^{t+h} is:

$$x^{t+h} \approx x^t + h \cdot v^t + \frac{1}{2}h^2 a^t + \frac{1}{6}h^3 f'''(t)$$

Similarly, the approximation of x^{t-h} is:

$$x^{t-h} \approx x^t - h \cdot v^t + \frac{1}{2}h^2 a^t - \frac{1}{6}h^3 f'''(t)$$

The summation of the two above equations gives the update formula of x^{t+h}.

$$x^{t+h} = 2x^t - x^{t-h} + h^2 a^t$$

The Verlet integration does not use explicitly the velocity of the particles. In the update formula, the velocity at time *t* is simply approximated by using the positions at time *t-h*. This method is found to be stable because of the approximation made on the velocity. However, the major problem of this approach is its lack of precision; the integration method performs well when using a small number of particles but makes the simulation over-damped when the system is composed of several thousands of particles.

1.2.1.2. Inverse dynamics

In order to increase the stability of the explicit Euler method, the stiffness of the system should be reduced. Stable simulation can be achieved either by reducing the strength coefficient of the springs or increasing the mass of the particles. The main drawback of this strategy is the resulting cloth that becomes "super-elastic". The cloth stretches too much under its own weight. Provot et al. [PROV 95] have proposed to modify the position of the particles after every simulation step to prevent the springs from stretching too much. Vassilev et al. [VASS 01] [VASS 00] have also implemented this approach.

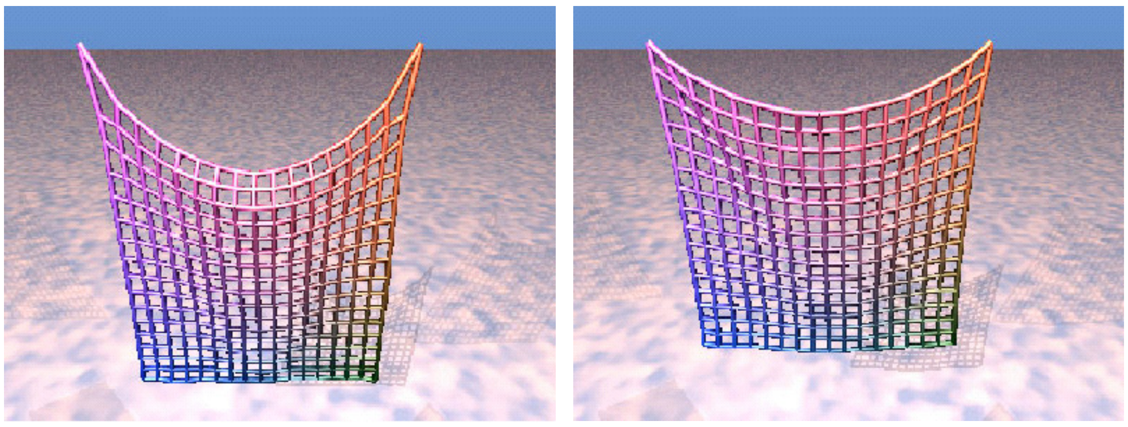


Figure II-4: Cloth simulation without (a) and with corrections using Inverse Dynamics (b) [PROV 95]

The position correction returns to a relaxation procedure during which the length of every spring is checked. If the spring length is above a certain threshold, the end points of the spring are moved along the spring line to make them closer. The velocity of the moved particles is also modified. The new velocity $\tilde{v}_i^{t+h}$ is obtained using the position of the particle at the previous step x_i^t and new corrected one $\tilde{x}_i^{t+h}$:

$$\tilde{v}_i^{t+h} = \frac{\tilde{x}_i^{t+h} - x_i^t}{h}$$

This method is not without disadvantages. First, the algorithm does not define any order in which the springs should be examined. In consequence, the correction of one spring may over stretch one of its neighbours and the algorithm has not been proven to work in all cases. In addition, the use of a geometric method to directly modify the position of the mass-spring system degrades the cloth dynamics. The method works in most cases because the over elongation of springs appears rather rarely.

1.2.2. The implicit Euler integration method

The implicit Euler integration method is described in detail in the first subsection. In the next two subsections are presented two improved versions of the implicit Euler method. In the last section is given the description of two approaches using the GPU for solving the linear system of the implicit Euler method.

1.2.2.1. Baraff et al.

Baraff et al. [BARA 98] are one of the first who have introduced the Implicit Euler Integration method to compute the cloth simulation. They stated that the bottleneck of real-time cloth simulation comes from the fact that the time-step must remain small in order to avoid instability.

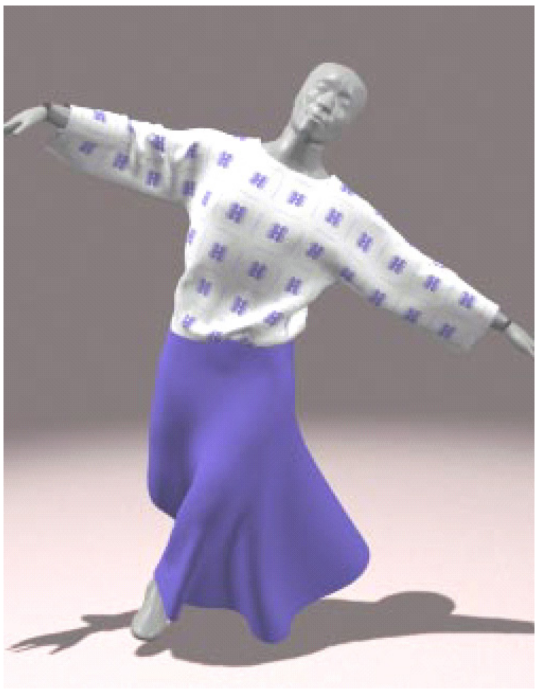


Figure II-5: Cloth simulation with the Implicit Euler integration, [BARA 98]

It was not within the scope of their work to develop a true real-time cloth simulator. They rather focused on a method that can stably take large time steps, suggesting the possibility of real-time simulation of simple objects. The Implicit Euler Integration method as introduced in the previous section takes the following forms:

$$v_i^{t+h} = v_i^t + F_i^{t+h} \frac{h}{m_i} \qquad (8)$$

$$x_i^{t+h} = x_i^t + v_i^{t+h} h \qquad (9)$$

To simplify the notation, we define:

$$\Delta x_i^{t+h} = x_i^{t+h} - x_i^t$$

$$\Delta v_i^{t+h} = v_i^{t+h} - v_i^t$$

$$\Delta x^{t+h} = \left[\Delta x_1^{t+h}, \Delta x_2^{t+h}, \ldots, \Delta x_n^{t+h}\right]^{\mathrm{T}}$$

$$\Delta v^{t+h} = \left[\Delta v_1^{t+h}, \Delta v_2^{t+h}, \ldots, \Delta v_n^{t+h}\right]^{\mathrm{T}}$$

The equation 6 involves the computation of F_i^{t+h} which can be calculated by:

$$F^{t+h} = F^t + \frac{\partial F}{\partial x} \Delta x^{t+h} \qquad (10)$$

where

$$F^t = \left[F_1^t, F_2^t, \ldots, F_n^t\right]^{\mathrm{T}}.$$

By substituting equation (10) into equation (8) and by writing $\Delta x^{t+h} = \left(v^t + \Delta v^{t+h}\right)h$, we can define:

$$\left(\mathrm{I} - \frac{h^2}{m}\frac{\partial F}{\partial x}\right) \cdot \Delta v^{t+h} = \left(F^t + h\frac{\partial F}{\partial x}v^t\right)\frac{h}{m} \qquad (11)$$

The equation 8 involves the following:

- h and m which are constant.
- F^t which can be easily calculated. For a simple mass-spring system, the force exerted by a spring joining two masses i and j is $F_{(i,j)} = -k_{i,j}\left(\left\|x_i - x_j\right\| - l^0_{i,j}\right) \cdot \frac{\left(x_i - x_j\right)}{\left\|x_i - x_j\right\|}$ where $l^0_{i,j}$ and $k_{i,j}$ are respectively the rest length and the stiffness of the spring.
- $\frac{\partial F}{\partial x}$ which is the derivative of the force. It is a large sparse matrix of size 3n by 3n, n being the number of particles. $\frac{\partial F}{\partial x}$ is also known as Hessian matrix in the research literature. For a spring between mass i and mass j, we have $\frac{\partial F_{(i,j)}}{\partial x_i} = -k_{i,j}\left[\frac{\left\|x_i - x_j\right\| - l^0_{i,j}}{\left\|x_i - x_j\right\|} \cdot \mathrm{I}_3 + l^0_{i,j}\frac{\left(x_i - x_j\right)^T \cdot \left(x_i - x_j\right)}{\left\|x_i - x_j\right\|^3}\right]$.
- $h\frac{\partial F}{\partial x}v^t$ which can be considered as the viscosity force; they are function of the velocity of the particles.
- $\left(\mathrm{I} - \frac{h^2}{m}\frac{\partial F}{\partial x}\right)$ which is a sparse matrix. The equation involves solving a large linear system that can be solved by, for instance, the conjugate gradient method [PRES 88].

Most of the research works on cloth simulation have been reusing this method. The main drawback of the Implicit Integration method is the computation of the large linear system, generating limitations for its use in real-time simulation. Most of the existing simulators use the conjugate gradient method to solve these large linear systems [PRES 88].

Several researchers have concentrated their efforts on the linear system of the Implicit Euler Integration method in order to reduce the computation time.

1.2.2.2. Desbrun et al.

Desbrun et al. [DESB 99] and Meyer et al. [MEYE 00] have rewritten the equation (11) as follows:

$$\Delta v^{t+h} = \left(\mathrm{I} - \frac{h^2}{m} \frac{\partial F}{\partial x} \right)^{-1} \cdot \widetilde{F}^t \frac{h}{m} \qquad (12)$$

with

$$\widetilde{F}^t = F^t + h \frac{\partial F}{\partial x} v^t .$$

F^t is the total internal forces of the system and $h \frac{\partial F}{\partial x} v^t$ the viscosity force F_i^{visco}. Viscosity forces are computed as follows:

$$F_i^{visco} = k \cdot h \sum_{(i,j) \in Edges} \left(v_j - v_i \right) \qquad (13)$$

The other part of the equation (12) involves the computation of $\frac{\partial F}{\partial x}$. They stated that the matrix $H = \frac{\partial F}{\partial x}$ is not constant in 3D due to the non-zero rest lengths of springs. The reason is that the spring forces have a non-linear component as shown below:

$$F_{(i,j)}^{linear} = -k_{i,j} \left(x_i - x_j \right) \qquad (14)$$

$$F_{(i,j)}^{non-linear} = k_{i,j} \cdot l_{i,j}^0 \cdot \frac{\left(x_i - x_j \right)}{\left\| x_i - x_j \right\|} \qquad (15)$$

Therefore, they propose to linearize the force field by assuming a zero rest length for all springs; $F_{(i,j)}^{non-linear}$ is not taken into account. As a result, the matrix H can be approximated by:

$$\left\{ \begin{array}{ll} H_{ij} = k_{ij} & \text{if i} \neq \text{j} \\ H_{ii} = -\sum_{j \neq i} k_{ij} & \end{array} \right\} \qquad (16)$$

where k_{ij} is the stiffness of the spring *(i,j)* and H_{ij} denotes the entry of H at the *i*th row and the *j*th column. Consequently, $\left(\mathrm{I} - \frac{h^2}{m} \frac{\partial F}{\partial x} \right)^{-1}$ remains constant during all the simulation as long as the time step and the masses remain unchanged. Their approximation introduces errors, as their integration scheme does not preserve the angular momentum of the springs. The non-linear component of the spring forces is not taken into account. They recover this error in a post-processing phase by correcting the angular velocity of the springs.

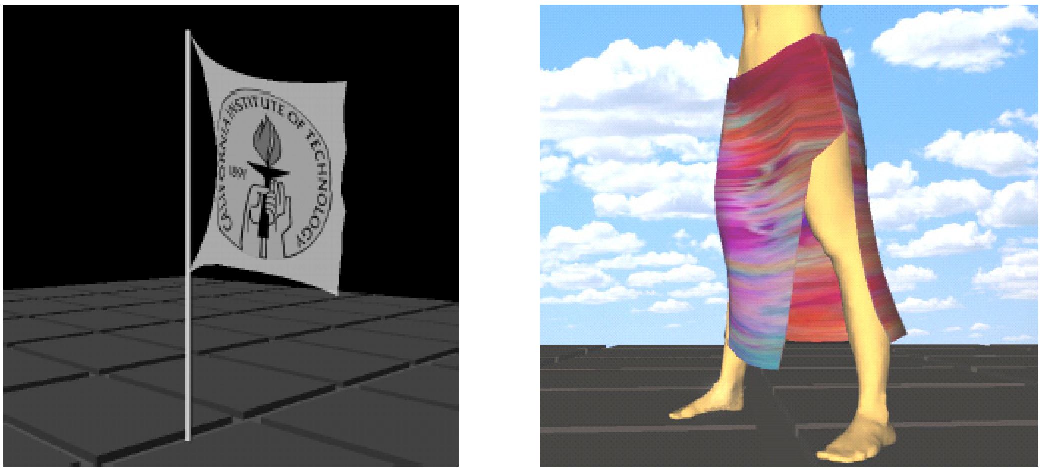


Figure II-6: Interactive Animation of Structured Deformable Objects [MEYE 00]

However, their work leaves several limitations. First, different cloth materials cannot be simulated accurately, because it is based on a simple mass-spring system. Second, it assumes that the physical parameters and the time step remain unchanged during the whole simulation. Finally, their collision detection scheme does not allow computing collisions with deformable objects, for instance a moving mannequin.

1.2.2.3. Kang et al.

Kang et al. [KANG 00] [KANH 00] have proposed yet another approximation of the linear system. They noted that the model proposed by Desbrun et al. has too many restrictions; the physical parameters cannot be modified during the simulation for instance. They rewrote the equation 12 as follows:

$$\left(1-\frac{h^2H_{ii}}{m_i}\right)\cdot\Delta v_i-\frac{h^2}{m_i}\sum_{(i,j)\in E}\left(H_{ij}\Delta v_j\right)=\frac{\tilde{F}^t h}{m_i}$$

E denotes the list of spring edges between the mass points. They adopted the approximation of the Hessian matrix proposed by Desbrun et al. and rewrote the equation above as follows:

$$\Delta v_i^{t+h}=\frac{\tilde{F}_i^t h+kh^2\sum_{(i,j)\in E}\Delta v_j^{t+h}}{m_i+kh^2n_i}$$

n_i is the number of neighbouring vertices to i. They approximate the unknown value of Δv_j^{t+h} by dropping the second term:

$$\Delta v_j^{t+h}\approx\frac{\tilde{F}_j^t h}{m_j+kh^2n_j}.$$

Finally, they wrote the equation that can be directly used to compute an approximation of Δv_i^{t+h}:

$$\Delta v_i^{t+h} = \frac{\widetilde{F}_i^t h + kh^2 \sum_{(i,j)\in E} \widetilde{F}_i^t h \big/ \left(m_j + h^2 k n_j\right)}{m_i + kh^2 n_i} \qquad (17)$$

In comparison with previous approaches, the simulator presented by Kang et al. requires less computation time because it does not need solving the linear system.

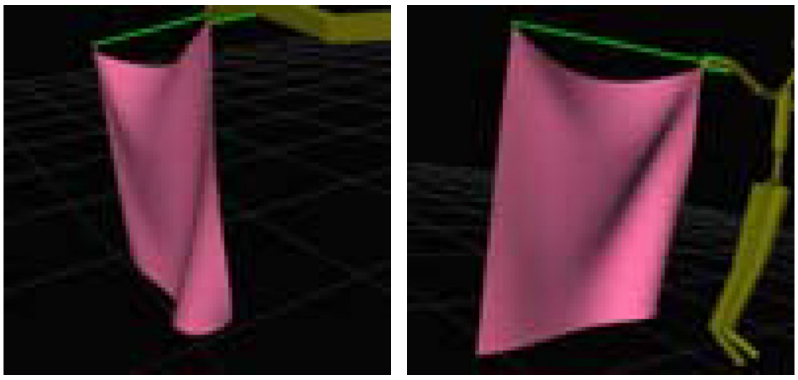


Figure II-7: Real-time animation of a flag [KANG 01]

According to the authors, its main limitation is the undesirable results when simulating garments with high number of polygons. This artefact is due to approximation in the equation 12 that determines the next position of a mass point by considering the positions of a limited number of linked neighbours.

1.2.2.4. The conjugate gradient method on the GPU

The implicit Euler integration method involves solving a large sparse linear system. In most of simulation engines, the solution of this linear system is computed by using the conjugate gradient method. Two original publications presented by Bolz et al. [BOLZ 03] and Krüger et al. [KRÜG 03] have introduced a method to compute the conjugate gradient with GPU (Graphics Processor Unit) programming [CGSH 03]. They propose an implementation of several basic matrix operators such as the matrix-vector multiply, the inner product, addition, subtraction, multiplication, and reduction of matrices, etc. Matrices and vectors are stored in the 2D texture memory. Instructions are encoded as a shader program. As noted by Krüger et al., the basic matrix operators are 12 to 15 times faster on GPU in average than their counterpart on CPU. The least-performance operator is the matrix reduction, which is only twice faster than the CPU. Their implementation has been tested with several physically based simulations such as fluid flow and water surface. To our knowledge, cloth simulation has not yet been implemented on GPU. Nevertheless, cloth simulation belongs to the same category of operation and therefore should work in the same way.

1.2.3. The hybrid method using implicit and explicit Euler methods

Hybrid approaches aim at taking advantage of the both implicit and explicit Euler methods. These two methods are complementary; stiff elements of the system are handled with the time-consuming implicit method while non-stiff elements are integrated explicitly and quickly. This scheme has been given the name of IMEX and several researchers [ASCH 95] [ASCH 97] have worked on this. Eberhardt et al. [EBER 00] have described one that was specifically developed for the simulation of the cloth dynamics. They started by the rewriting of the Ordinary Differential Equation of the simulation system:

$$\dot{Q}^t = f(t, Q^t) + g(t, Q^t)$$

Q^t denotes the state of cloth mesh: $Q^t \equiv (x^t, v^t)$. f and g are respectively the non-stiff and stiff parts of the system. The development at the first order gives:

$$Q^{t+h} = Q^t + hf(t, Q^t) + hg(t+h, Q^{t+h}) \qquad (18)$$

The IMEX method requires classifying the forces of the system according to their stiffness. As Desbrun et al. [DESB 99] already mentioned, the spring force has a linear and a non-linear components (see equation (14) and (15)). The linear component is stiff whereas the non-linear component is not. The reason is because the non-linear component has a constant absolute value. Eberhardt et al. [EBER 00] incorporated as well the bend and shear forces into the non-stiff part. In addition, the viscosity forces (see equation (13)) have been put into the stiff part. Finally, the equation of the stiff part is:

$$Fs(x, v) = k_{i,j}(x_j - x_i) + h_{i,j}(v_j - v_i)$$

$h_{i,j}$ denotes the damping coefficient. By using the convention given by Desbrun et al. (see equation (16)), the above equation becomes:

$$Fs = H \cdot x + D \cdot v \qquad (19)$$

with

$$\left\{ \begin{array}{ll} D_{ij} = h_{ij} & \text{if i} \neq \text{j} \\ D_{ii} = -\sum_{j \neq i} h_{ij} & \end{array} \right\}$$

By inserting equations (3) and (5) into (18), we get:

$$v^{t+h} = v^t + Fs(x^{t+h}, v^{t+h})\frac{h}{m} + Fn(x^t, v^t)\frac{h}{m}$$

$$x^{t+h} = x^t + v^{t+h}h$$

Non-stiff forces are denoted by Fn. The rewriting of the two above equations with the equation (19) gives the update formula of the velocity:

$$v^{t+h} = v^t + (H \cdot x^t + h \cdot H \cdot v^{t+h} + D \cdot v^{t+h})\frac{h}{m} + Fn(x^t, v^t)\frac{h}{m}$$

Finally, the update formula comes down to solving the linear system:

$$\left(I-\frac{h^2}{m}\cdot H-\frac{h}{m}\cdot D\right)v^{t+h}=v^t+\frac{h}{m}+Fn\left(x^t,v^t\right)\frac{h}{m}+H\cdot x^t\cdot\frac{h}{m}$$

The right side of the equation involves the computation of the explicit component of the forces with x^t and v^t. v^{t+h} is then calculated by solving the linear system with the conjugate gradient method [PRES 88].

According to Melis et al. [MELI 02], the IMEX method is slightly faster than the method by Desbrun but slower that the one proposed by Kang et al. This method offers good performance but does not allow simulating several thousands of particles in real-time.

1.3. Collision detection

Other researchers have recognized the collision detection as one of the bottlenecks to real-time animation. Collision detection is a fundamental problem in Computer Graphics and is a research topic by itself. Many approaches have been developed over years, some of them focusing on the precision of the collision detection, others giving priority to the computation speed. Several collision detection algorithms are based on bounding volumes; objects are enclosed into simple geometric primitives. Some others compute the collisions not on the objects themselves but on their projections along specific axes. Some other techniques organize the object regions in the form of a hierarchy according to their proximity. Volino et al. [VOLK 00] describes in detail these collision detection approaches. In this section, we will limit our survey to the techniques that have been specifically developed for the clothes with the real-time application in mind. In the next subsections, four different approaches are discussed.

1.3.1. Collision detection with object subdivision methods

The idea of using hierarchies of bounding volumes finds its origin from the observation that collisions can be detected solely among objects that are close to each other. Subdivision methods are based on a hierarchical structure created by successive subdivisions of the entire scene, according to the proximity rules. If the scene is composed of triangles, each node of the hierarchy is a polygon. For each of these nodes, a bounding volume containing the polygon is computed. The collision detection is performed by computing the intersection of the bounding volumes by starting from the root the hierarchy. If there is an intersection between two bounding volumes, the collision algorithm checks the intersection of the bounding volumes of the children nodes recursively.

This technique has been widely used for simulating clothes in non-real-time [VOLK 00] [BRID 02]. Many other algorithms have been developed recently for collision detection among rigid object [REDO 02] [KLOS 98]. Unfortunately, clothes are highly deformable objects; the bounding volume hierarchy needs to be updated at every frame of the simulation.

1.3.2. Image-based collision detection

Image-based collision detection makes use of the graphics rendering hardware to reduce the computation cost. This class of methods has become very popular recently [GOVI 03] [BACI 02]; high performance 3D graphic systems are part of almost every personal computer or game console.

Vassilev et al. [VASS 01] [VASS 00] have implemented this method for their real-time collision detection between the clothes and body. They have used the z-buffer to generate depth and normal maps. They first compute the rendering of the body without cloth from the front and back view. Each rendering provides two maps: one contains depth values on Z and the other contains the orientation of the surface normal vector encoded in RGB data structure. To check for a collision they convert the appropriate vertex coordinates of the garment to an index in the depth and normal buffer. A check for collision is accomplished by simply comparing the z value of the vertex coordinate with the corresponding value in the depth buffer. The direction of the collision is approximated with the normal map. Depth and normal maps are pre-computed prior to simulation.

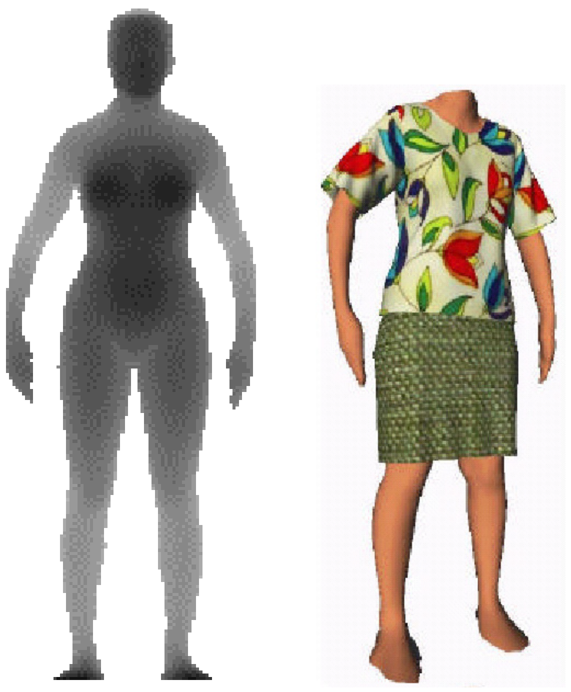


Figure II-8: Collision Detection using Z-buffer maps [VASS 01]

Computation time of their collision detection does not depend on the complexity of the body. However, the maps need to be pre-computed before simulation. Moreover, their method does not support self-collisions in cloth objects.

1.3.3. Voxel-based collision detection

This approach is based on a spatial subdivision. The algorithm starts by selecting the voxel size that meets the requirements of targeted accuracy and performance. It then partitions the space into regions of free space, object surface and object interior. All voxels are organized hierarchically to form an octree as shown on the figure below. Voxel-based collision detection has been implemented by Meyer et al. [MEYE 00] and McNeely et al. [MCNE 99].

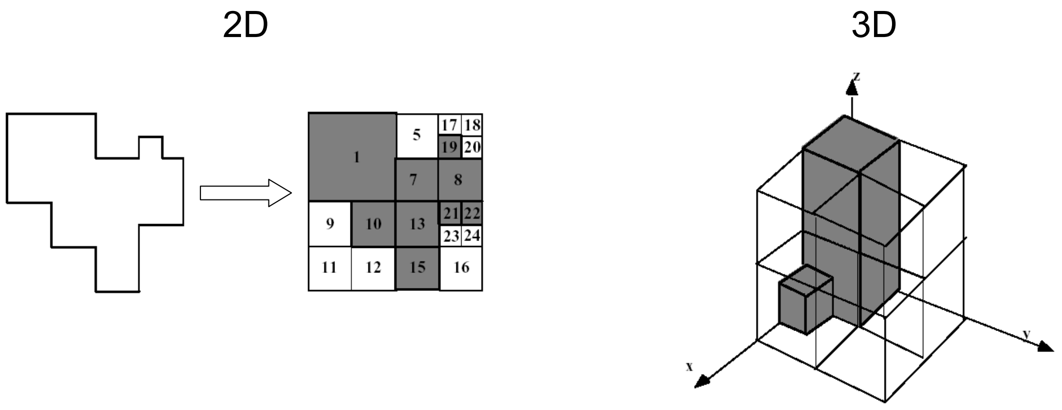


Figure II-9: Portioning the space with octrees (2D and 3D cases)

The collision detection remains as computing which voxel the cloth vertex belongs to and checking the flags of the voxel to know whether the cloth vertex is in collision or not. The main advantage of this algorithm is its rapid computation speed. On the other hand, the space partitioning needs to be calculated during the pre-processing, restricting the collision detection to rigid objects only.

1.3.4. Collision detection with implicit surfaces

This technique consists of approximating the surface of the colliding objects by implicit surfaces [BLOO 97]. Implicit surfaces are iso-valued surfaces created from blending primitives (points, lines, polygons, spheres, ellipsoids, etc.) represented by implicit equations of the form F (x, y, z) = 0. Each primitive is a procedure that returns a functional value for the field defined by the implicit equation. The animation of implicit surfaces is achieved by controlling the location of the primitives.

Thank to implicit surfaces, the collision check comes down to computing the scalar field at the location of the cloth vertex and checking if the computed value is above or below a threshold. One advantage is that implicit surfaces enable to know the location of the closest external surface and therefore the penetration depth. Moreover, implicit surfaces offer the possibility to manage easily different cloth layers by using different iso-surfaces. An example of the use of iso-surfaces for collision detection has been shown by Rudomin et al. [RUDO 00].

Unfortunately, the use of implicit surfaces is restricted to the modelling of volumes (human bodies for instance) and not surfaces such as clothes.

2. Geometric-based simulation

Some other researchers have used geometrical approaches [WEIL 86] [AGUI 90] [HIND 90] [NGHG 95] [BAEH 02] for the simulation of the cloth dynamics. Geometrical models do not consider the actual physical properties of the cloth; they rather provide techniques that produce plausible cloth shapes by using mathematic models such as catenary curves or surfaces. Obviously, these techniques do not reproduce the movements of clothes. Moreover, geometrical techniques require a considerable amount of user intervention to design the garments. They can be regarded as a form of advanced design tools for garments.

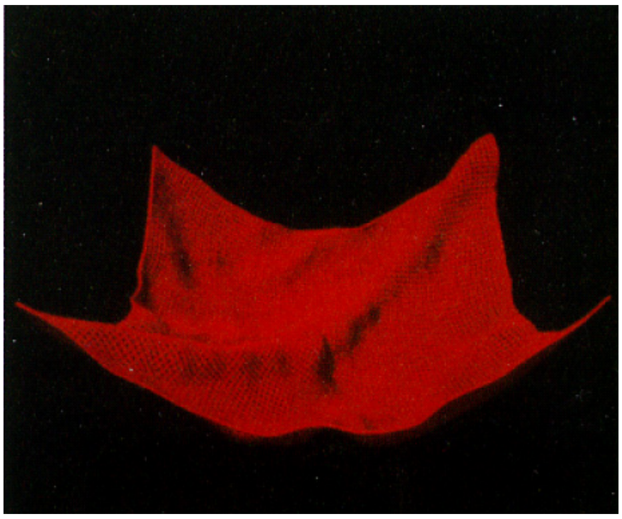


Figure II-10: Computation of the cloth shape using catenary surfaces [WEIL 86]

3. Hybrid approaches

Hybrid approaches aim to combine nicely physically based deformation and geometric deformation. The idea of hybrid approach stems from the following observation: physically based simulations are slow to compute but produce realistic results while simulations based on geometric method are much faster but not really suitable to animate full clothes. By combining advantages of the both approaches, one can expect to have acceptable results within moderate computation time. In most cases, physically based models are used to compute the global movements of garments and the details such as wrinkles are generated with geometric models.

Kang et al. [KANG 01] [KANG 02] improved the visual quality of the garments of small number of polygons by tessellating the triangles. With a cubic spline curve, their tessellation algorithm can simulate the wrinkles.

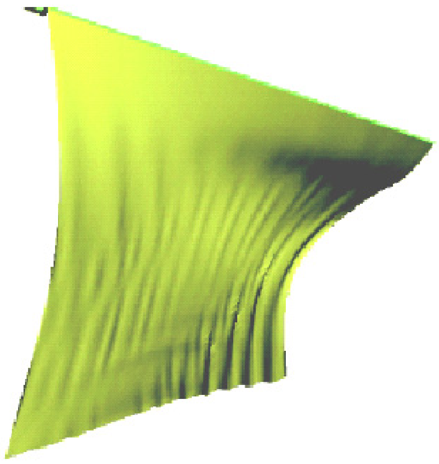


Figure II-11: Simulation of wrinkles using spline curves [KANG 02]

Oshita et al. [OSHI 01] use a similar approach to Kang et al.

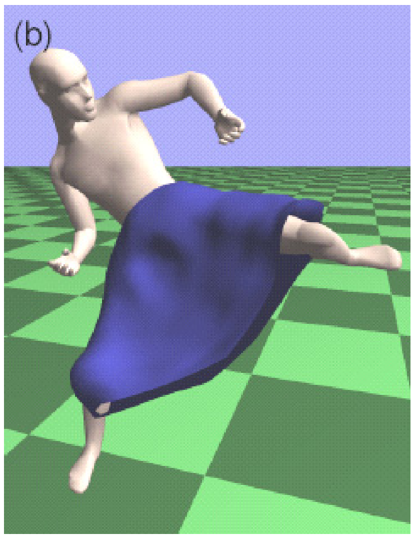


Figure II-12: Soothing of the cloth surface using tessellation [OSHI 01]

These both methods are mainly applicable to flat surfaces where physical simulation can be done with a relatively small number (<1,000) of polygons. However, highly curved surfaces, such as sleeves, need to be simulated with a higher number of polygons.

Recently, Hadap et al. [HADA 99] have proposed to simulate the cloth wrinkles with bump map. The global movements of the mesh are simulated with a particle system. The bump/displacement map is the product of a modulation map and a wrinkling pattern. The modulation map is generated by computing the local deformation of cloth triangles and the wrinkling pattern is designed by the CG artist.

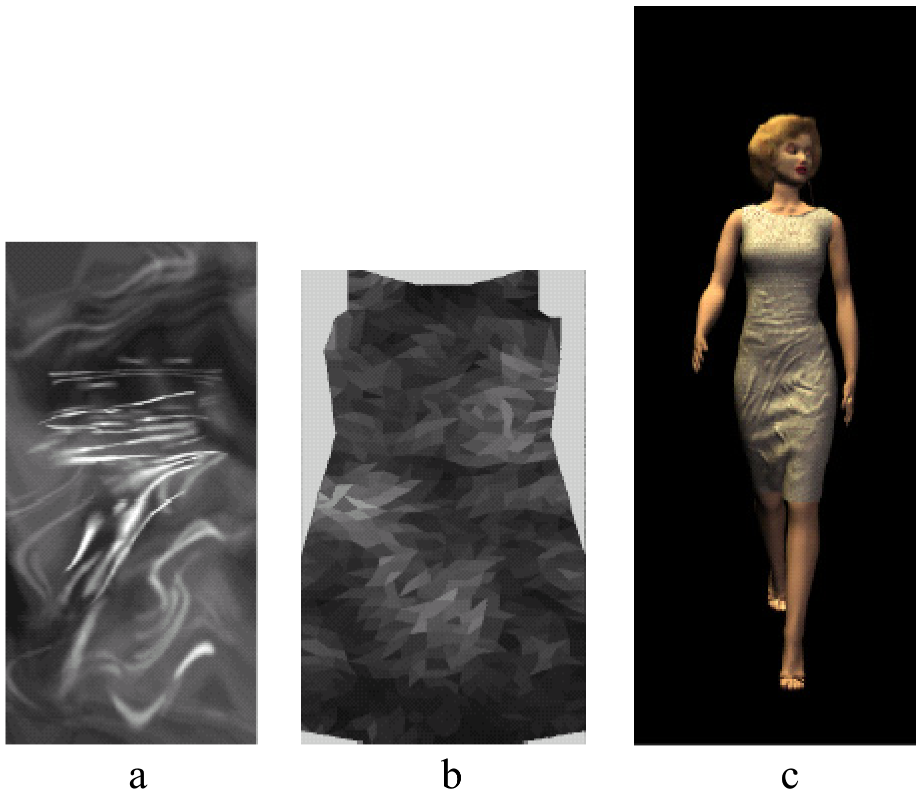


Figure II-13: The generation of wrinkles using bump map, the winkling pattern (a), the modulation map (b) and the generated wrinkles (c)

The computation time to generate wrinkles is negligible in comparison to the rendering time. However, their simulation method requires the design of the wrinkles pattern.

4. Example-based methods

Example-based approaches, also known as data-driven deformations have become popular since several years. They have been successfully used for motion data and other graphical objects such as deformable human face and body models. The main idea of example-based is to construct a simulator that can learn deformations through a set of examples. This simulator can be considered as a black box computing for each input parameter (user interaction) the corresponding output (the shape of the deformable object). During the training phase, a set of meaningful input/output data is given to the simulator, which learns. During the runtime execution, the simulator upon receiving an input computes the corresponding output. If the given input data corresponds exactly to one of the samples, the simulator will return the output of the sample. Otherwise, the returned output will be interpolated using one of the interpolation techniques such as RBF, k-nearest neighbours, neural network, or linear interpolation, etc.

Grzeszczuk et al. [GRZE 98] have used a neural network to animate dynamic objects. They have shown that physically based models can be replaced by a large neural network that automatically learns to simulate similar motions by observing the models in action. The neural network is trained with a set of samples that have been computed with the physically based method. Not surprisingly, their simulator works in real-time. However, it has not been proven that the same method can be applied for complex simulation such as clothes.

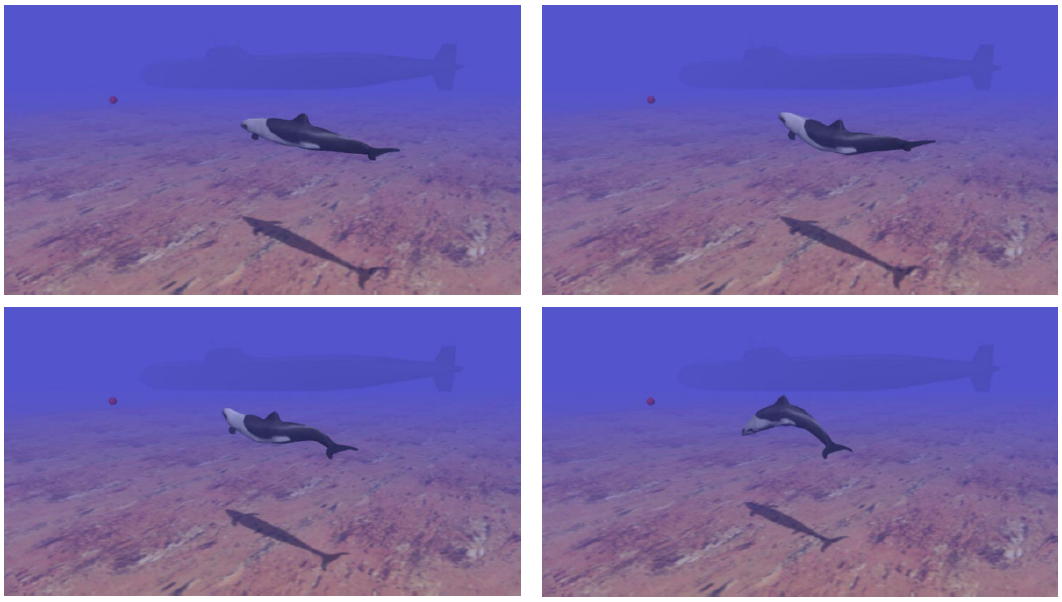


Figure II-14: Animation using neural network [GRZE 98]

Very recently, James et al. [JAME 03] have proven that the example-based approach can be successfully adapted to cloth simulation. The examples of cloth simulation they used had been calculated using commercial graphics package. Their approach makes use of Impulse Response Functions (IRF). Each IRF is indexed by the initial state *x* and two user-modelled parameter vectors, α^I and α^F, that describe the initial *Impulse* and persistent *Forcing* respectively. Each IRF contains a sequence of *T* states (positions and velocity) defining an orbit, *T* being the time step of the simulation.

IRF: $\xi\,(x, \alpha^I, \alpha^F; T)=(x^0= x, x^1, x^2,\ldots,x^T)$ with $\xi^t=x^t$, $t=0,\ldots,T$

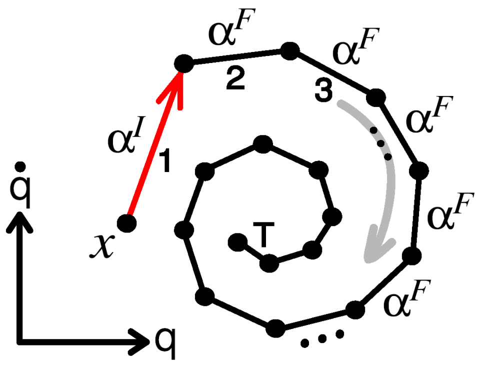


Figure II-15: Impulse Response Function and its orbit

To model a set of possible interactions (α^I, α^F), they construct a palette of IRF. Each IRF corresponds to a particular user interaction as shown in the figure below.

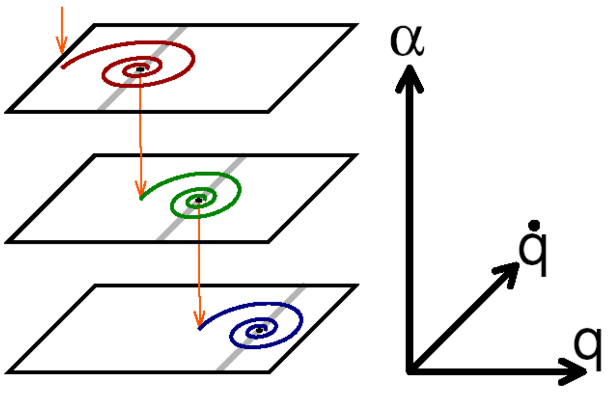


Figure II-16: Three IRF orbits corresponding to the three rotation speeds of the door maintaining the cloth

During the runtime simulation, the orbit corresponding to the user interaction is played. If the user interaction is changed, a new orbit corresponding to the new user interaction is found and replaces the current one. To avoid discontinuity while changing the Impulse Response Function, they blend the two orbits with the coefficient $e^{-\lambda t}$ *t* being the duration of the blending stage.

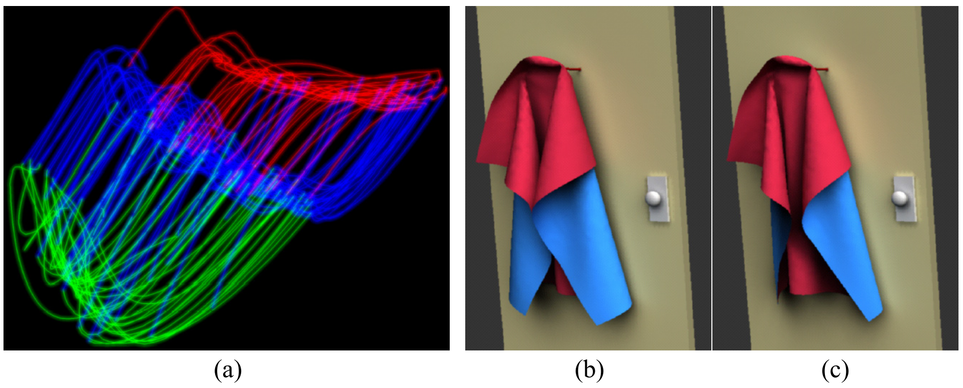


Figure II-17: Real-time simulation of piece of tissue by [JAME 03] with the orbits (a), and real-time simulation (b) and (c).

This work is interesting in the sense that it is the first method that aims to introduce example-based strategy in cloth simulation. Also, it is the only method that can truly simulate different cloth materials in real-time. However, this method is not adaptable to the simulation of clothes on virtual humans with its main drawback being the very limited user interaction. The authors have given the example of a piece of tissue (see Figure II-17) attached to a door that can be moved according to only three different speeds. The human skeleton is composed of 30 joints at minimum, which makes the simulation problem much more complex than just a door that has only one degree of freedom.

5. Comparative analysis

Table II-1 summarizes the existing cloth deformation approaches target to real-time application. We classified these methods according to the following four criteria:

- Computation speed: how fast the method can work. This criteria is most important as we want to obtain the simulation in real-time.
- Versatility: what are the limitations of the garment to be simulated. Does the model support multi-layer cloth? Is there any limitation in designing patterns?
- Quality of simulation: how realistic is the simulation, what are the capabilities of the method to simulate the physical properties of materials such as silk, wool, and cotton…
- Pre-processing: does the method require a pre-processing phase prior to simulation and if so, how long is the duration of pre-processing phase. Does it require user intervention or is it automatic?

Cloth Deformation Models	**Computation Speed**	**Versatility**	**Quality of simulation**	**Pre-processing**
Implicit Euler Integration method by [BARA 98]	Slow	Yes	Good depending on the underlying mechanical model	No
Modified Implicit Euler Integration method by [MEYE 00] and [KANG 01]	Fast	Limited by the maximum number of vertices that the method can animate	Not able to simulate different cloth materials because based on a simple mass-spring system	Negligible
Hybrid methods combining physically based and geometric deformations by [KANG 02] and [OSHI 01]	Fast	No. The algorithm fails in simulating highly curved surfaces such as sleeves	Not able to simulate different cloth materials because based on partially on geometric deformation	No
Explicit Euler Integration (Verlet or inverse dynamics) with collision detection based on Z-buffer by [VASS 01]	Average	Limited because the Z-buffer maps need to be rendered before simulation	Not able to simulate different cloth materials because they use a simple mechanical model.	Yes. The pre-processing is used to render the Z-buffer maps.

Cloth Deformation Models	Computation Speed	Versatility	Quality of simulation	Pre-processing
Textured-based wrinkles by [HADA 99]	Fast	No. Each cloth requires designing the wrinkle pattern	Depends on the drawing of the wrinkles	Yes, the wrinkle pattern needs to be designed
Geometric Deformation by [WEIL 86] [AGUI 90]	Fast	No	No. Their main drawback is their inability to simulate cloth movements	Requires a lot of user interventions to define the clothes
Pre-computed Interactive Dynamic Deformable Scenes by [JAME 03]	Fast	No. The user interaction is very limited. Cannot simulate clothes on bodies.	Good. It can simulate different fabric materials, i.e. those that have been used to compute the animation sample.	Yes. The pre-processing takes time but is fully automatic.

Table II-1: Comparative analysis of cloth simulation methods

Several observations can be made regarding this comparative analysis:

- Researchers have developed two strategies for reducing the computation time. The first strategy consists of reducing the complexity of the cloth mesh and/or the mechanical model. For instance, the original complex cloth mesh surface can be simplified into a smaller number of polygons. The simple mass spring system can then be chosen for simulating the fabric properties. Collision detection can also be simplified by using low-polygonal mesh or geometric primitives such as cylinders to represent other environmental objects like the human body. Note that this strategy does not necessarily restrict the simulation to particular cases. Methods that belong to this category can be used for simulating various types of clothes like flag, tablecloth, or garments. On the other hand, they cannot simulate accurately the details of clothes such as wrinkles. The second strategy, on the other hand, reduces the computation time by limiting the versatility of the clothes in simulation. This simplification comes from the observation that not all cloth properties have to be computed if the cloth is used in a particular context. In fact, it is often the case that the physically based deformation can be beneficially replaced by a geometric interpolation, as James et al. [JAME 03] have noted. In their simulation system, the cloth is attached to a door that can rotate along one axe according to three pre-defined velocities. This method effectively returns to playing back of pre-recorded cloth animations according to the movements of the door. In general, these methods produce simulations of higher quality. Of course, their main drawback is that their use is limited to particular cases. For instance, the work by James et al. cannot be used for the animation of clothes on bodies.

- Despite the immense amount of work that have been presented, none of the existing methods can simulate complex clothes in real-time. This underlines that further research can be conducted on real-time clothes. This is one aim of this dissertation and we have been particularly focusing on methods that can simulate dressed virtual humans in real-time.
- In general, versatile simulators tend to be slower than specialized simulators. This is because specialized models make certain assumptions on the properties of the garments and avoid unnecessary reasoning. One good example is the example-based simulator presented by James et al. [JAME 03]. This method gives a quality of simulation (in terms of both realism and speed) that no other method can obtain. However, this simulation cannot be used for the animation of clothes on bodies.
- We believe that hybrid approaches are promising because they try to decompose the simulation problem into several layers where the method (either geometric or physics-based) that best fits the needs of the layer can be applied.

6. Other physically based models for real-time animations

Other works have been made in real-time simulations for other kinds of objects such as volumetric meshes. We survey these research works since we believe that they are analogous to the simulation of garments in terms of new ideas.

James et al. [JAME 99] have described the boundary integral equation formulation of static linear elasticity as well as the related Boundary Element Method discretization technique. Their model does not use true dynamics, but rather a collection of static postures, limiting its potential applications.

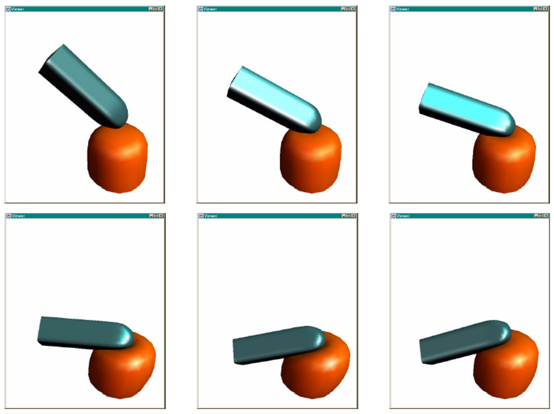


Figure II-18: Real-time animation of volumetric mesh by James et al.

Debunne et al. [DEBU 00] [DEBU 01] have recently introduced a technique for animating soft bodies in real time. Their approach is interesting in the sense that the level of detail of simulated mesh is adapted to local strains. Higher level of detail is used whenever the mesh is under higher deformation whereas the pieces of the mesh that are not deformed remain at a lower resolution, reducing the complexity of the overall computation. Unfortunately, their method works on volumetric meshes and therefore it is not directly applicable to thin objects such as cloth.

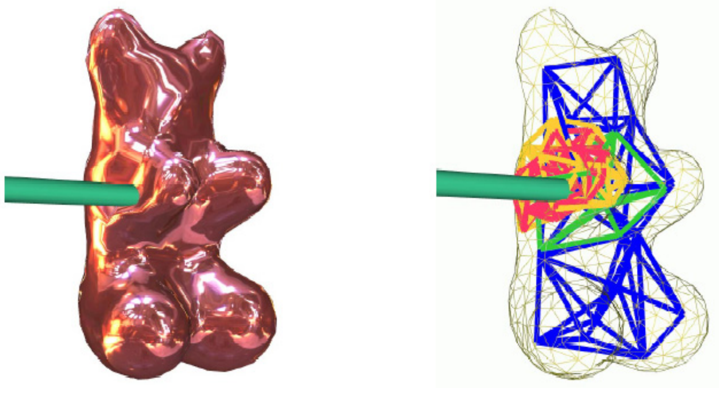


Figure II-19: Dynamic Real-Time Deformations using Space & Time Adaptive Sampling [DEBU 01]

Chapter III. Real-time deformation of skin and the making of garments

This chapter details the prerequisites needed for real-time cloth simulation. Any simulation of dressed virtual human requires choosing a body model as well as designing the garment model. This section describes the body model and the way it is deformed as the skin deformation is an important component of our cloth deformation model. A description of the garment design is also supplied.

1. Skeleton hierarchy

In most cases, skin deformation is controlled by animating the underlying skeleton. The skeleton consists of a set of joints connected together to form a hierarchy. A transformation applied to one joint results in the displacement of its entire descendent joints. The animation of the skeleton hierarchy is achieved by the modification of the rotation angles of the joints. The root joint can also be translated. In the literature, many different conventions have been defined for the virtual human skeleton. Here, we have chosen the H-Anim structure [HANI 03], as this standard is widely used in the research community and is supported by many commercial packages.

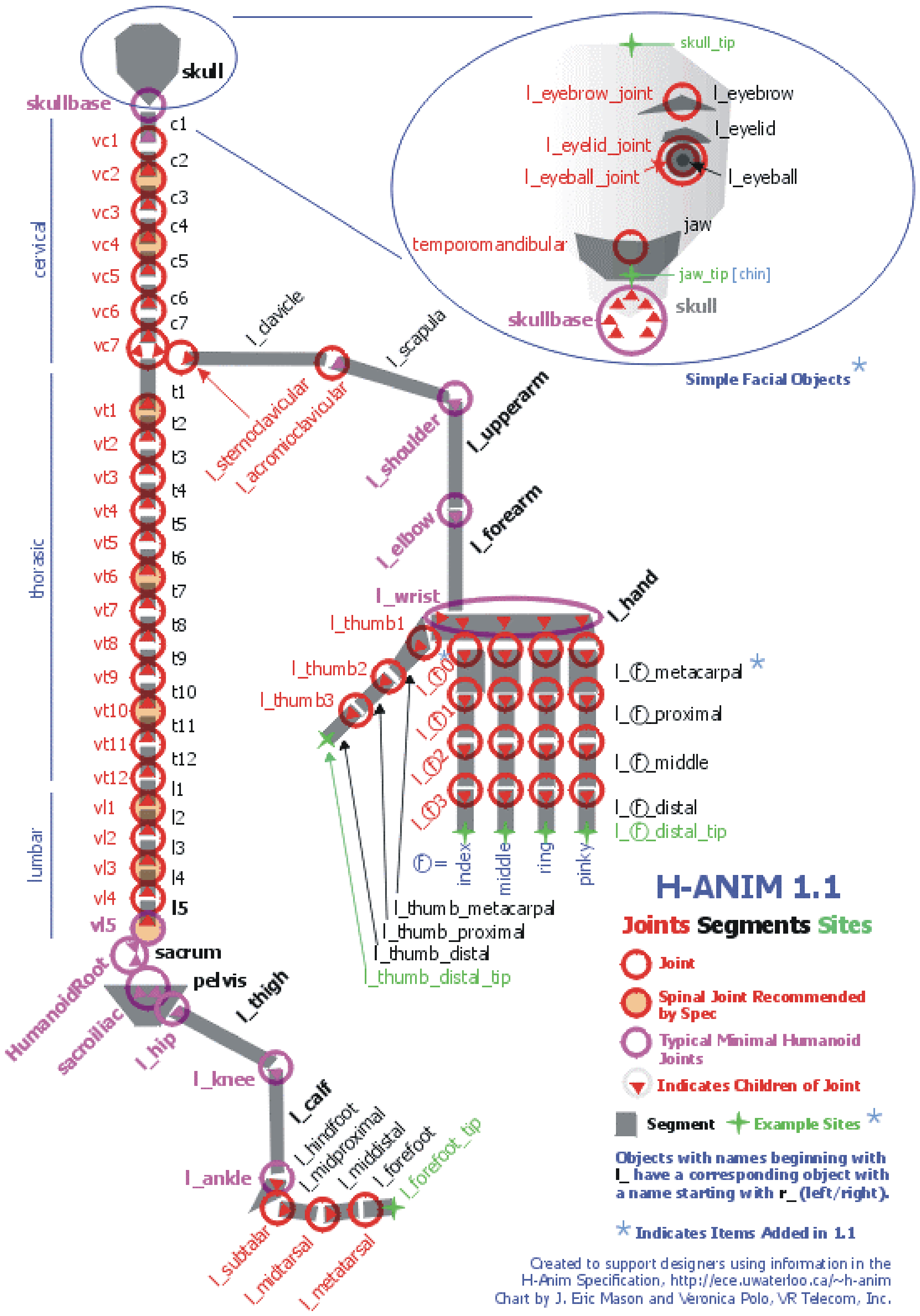


Figure III-1: The H-Anim standard

2. Skeleton Driven Deformation

The skeleton-driven deformation, a classical method for the basic skin deformation is perhaps the most widely used technique in 3D character animation. In research literature, an early version was presented by Magnenat-Thalmann et al. [MAGN 88], who introduced the concept of Joint-dependent Local Deformation (JLD) operators to smoothly deform the skin surface. This technique has been given with various names such as Sub-Space Deformation (SSD), linear blend skinning, or smooth skinning [WEBE 00] [LAND 98]. This method works first by assigning a set of joints with weights to each vertex in the character. The location of a vertex is then calculated by a weighted combination of the transformation of the influencing joints as shown on the equation 1.

$$P_v = \sum_i w_i (M_{i,C} \cdot M_{i,Dress}^{-1} \cdot P_{Dress}) \qquad (1)$$

The skeletal deformation makes use of an initial character pose, namely dress pose, where $M_{i,Dress}^{-1}$, the transformation matrix of i^{th} influencing joint, and P_{Dress}, the position of the vertex are defined. While this method provides fast results and is compact in memory; its drawbacks are the undesirable artifacts such as the "candy-wrapper" collapse effect on wrists and collapsing around bending joints (Figure III-2). The artefacts occur because vertices are transformed by linearly interpolated matrices. If the interpolated matrices are dissimilar, as in a rotation of nearly π radians, the interpolated transformation is degenerate, so the geometry must collapse.

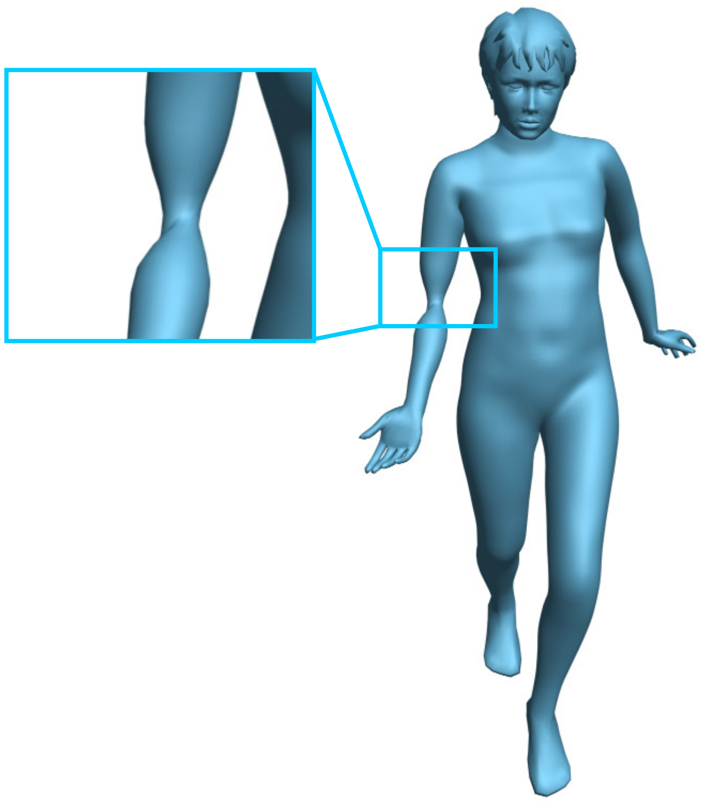


Figure III-2: The problem of the Sub-Space Deformation

In the following section 2.1 we give a survey of existing methods that have been proposed to overcome the problems of the SSD described above. We then present the approach we have implemented in Section 2.2.

2.1. Review of the exiting methods to improve the Skeleton Driven Deformation

Several attempts have been made to overcome the limitation of geometric skin deformation by using examples of varying postures and blending them during animation. Aimed mostly at real-time applications, these example-based methods essentially seek for solutions to efficiently leverage realistic shapes that come either from captured skin shape of real people, physically based simulation results, or sculpted by skilled designers.

Pose space deformation [LEWI 00] approaches the problem by using artistically sculpted skin surfaces of varying posture and blending them during animation. Each vertex on the skin surface is associated with a linear combination of radial basis functions that compute its position given the pose of the moving character. These functions are formed by using the example pairs – the poses of the character, and the vertex positions that comprise skin surface. More recently, Kry et al. [KRYP 02] proposed an extension of that technique by using principal component analysis (PCA), allowing for optimal reduction of the data and thus faster deformation.

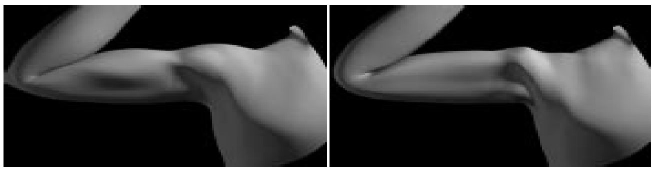


Figure III-3: Pose space deformation (left) result compared to the skeleton driven deformation (right)

Sloan et al. [SLOA 01] have shown similar results using RBF for blending the arm models. Their contribution lies in that they make use of equivalent of cardinal basis function. The blending functions are obtained by solving the linear system per example rather than per degree of freedom, which potentially is of a large number, thus resulting in an improved performance.

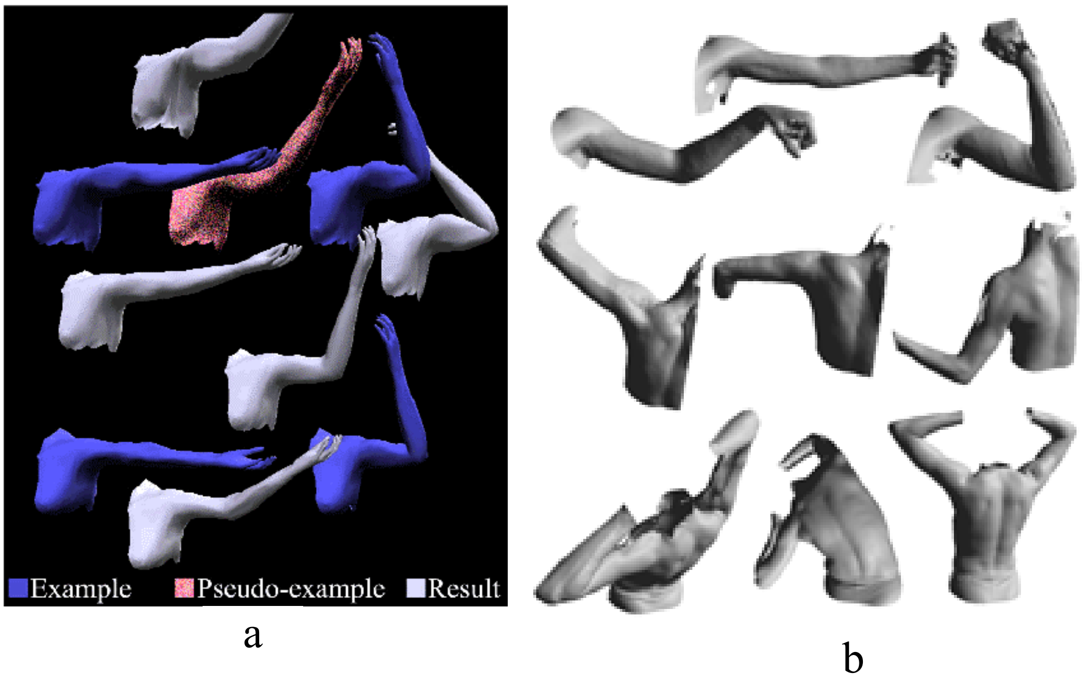


Figure III-4: Example based skin deformation by Sloan et al. [SLOA 01](a) and by Allen et al. [ALEN 02](b)

Allen et al. [ALEN 02] present yet another example-based method for creating realistic skeleton-driven deformation. Unlike previously published works, they start from an unorganized scan data instead of using existing models that have been sculpted by artists. After the correspondence is established among the dataset through a feature-based optimization on a template model followed by a refitting step, they build blending function based on the pose, using *k*-nearest-neighbours interpolation. Additionally, they build functions for combining subparts of the body, allowing for blending several datasets of different body parts like arms, shoulder and torso.

More recently, Mohr et al. [MOHR 03] have shown the extension of the SDD by introducing pseudo joints. The skeleton hierarchy is completed with extra joints inserted between existing ones to reduce the dissimilarity between two consecutive joints. These extra joints can also be used to simulate some nonlinear body deformation effects such as muscle bulges. Once all the extra joints have been defined, they use a fitting procedure to set the skinning parameters of these joints. The weights and the dress position of the vertices are defined by a linear regression so that the resulting skin surface fits to example body shape designed by artists. Having weights well defined, those examples could be discarded during the runtime.

2.2. Our approach for the Skeleton Driven Deformation

The development of a new SSD deformation method is not the main purpose of our research. However, the way the skin deforms is important in our framework since the skin shape forms the basis in our cloth deformation method (see Chapter IV and Chapter V). We have defined three requirements that the method should fulfil for this particular use:

- The method should overcome the undesirable effect of vertex collapsing as shown on Figure III-2. Unlike other skin features such as wrinkles or muscle bugles, which becomes less important to be simulated when they are hide by garments, this artefact remains particularly visible even if garments cover the skin.
- The method should provide an easy way to compute the local coordinate system for each skin vertex. The calculation of these local coordinate systems is necessary, as we want to compute the position of the cloth surface in relation to the skin surface (see Chapter IV and Chapter V).
- If possible, the data structure of this new method should be as close as possible to the one used by SSD. This will simplify the integration of this method to existing software infrastructure.

Many SSD methods have been developed and been proven to be efficient but none of them entirely fulfils the requirements we have defined for our particular use for cloth deformation. Thus, we have developed a modified version of the SSD, focusing on solving its main drawback (collapse effect). The SSD is based on matrix blending, as shown in equation (1), which can be rewritten as follows:

$$P_v = P_{Dress}.\sum_i w_i.M_i \text{ with } M_i = M_{i,C} \cdot M_{i,Dress}^{-1} \qquad (2)$$

The weighted summation of two or more matrices as defined in equation 2 leads to degenerated matrices, resulting in poor skin deformation. This is especially true when these matrices have a rotation close to π radians relative to each other (Figure III-5).

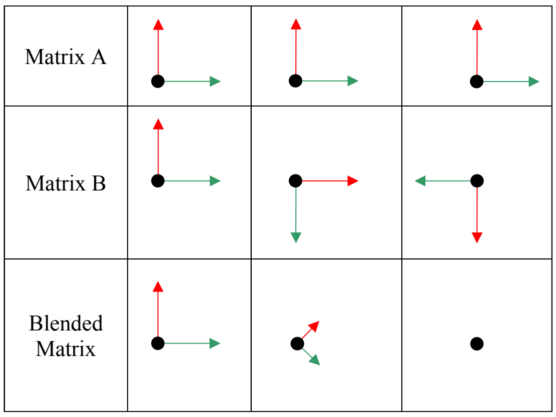


Figure III-5: Degenerated matrix with the usual matrix blending

The SSD deformation can be greatly improved if we rewrite the linear combination of the matrices in the equation 2. Here, we have adopted the matrix operator defined by Alexa [ALEX 02]. The combination of *i* matrices M_i with their weight w_i is given by:

$$\bigoplus_i w_i \cdot M_i = e^{\sum_i w_i \log(M_i)} \tag{3}$$

In equation (3), the matrix additions are made in the log-space where the matrices can be linearly combined. This technique has been originally demonstrated for key frame interpolation in animation. Each key frame is a transformation matrix and intermediate matrices are interpolated using the operator as defined in equation (3).

Unlike the other approach proposed by Mohr et al. [MOHR 03], the SSD deformation based on this matrix operator does not require to insert extra joints. Moreover, this method makes use of the same data structure as the common SSD, making its implementation straightforward.

The main drawback of this matrix operator is that it is slower at computation because it requires the evaluation of the logarithm and exponential of matrices. One possible solution to reduce the computation time is to combine the two matrix blending methods. When the matrices to be combined have similar orientation, the second matrix blending can be used. Matrix combination with higher rotation angles can be made using the second method. The rotation angles can be easily evaluated by scalar product between the two axes of the transformation matrices.

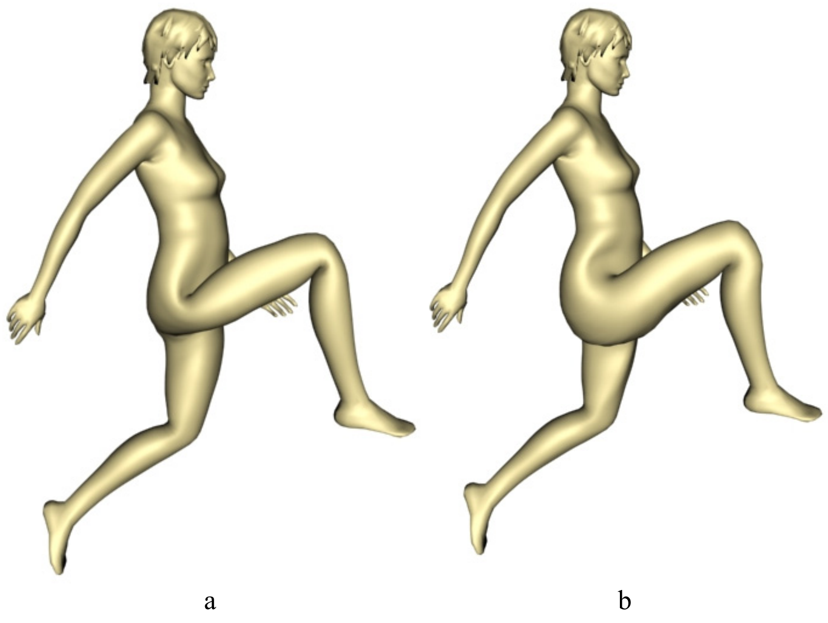


Figure III-6: Skeleton driven deformation without (a) and with matrix blending (b)

3. Pre-processing the body model

We start from a body model that is represented by a polygonal mesh, which can be animated using skeletal deformation. The first step in body model preparation is to define the attachment of the skin surface to the joints of a skeleton, i.e. which vertex is influenced by which joint. In our current implementation, this is done using an external application [DIGI 03]. This attachment information is later used for skeletal deformation. Next, the skin mesh is segmented using the weight values defined in the attachment information. Each triangle belongs to the joint corresponding to the highest weight. During the segmentation, vertices that are located along the segment boundaries are duplicated. Needless to say, those duplicated share the same attachment information. Finally, each segment is attached to the corresponding joint in the H-ANIM skeleton [HANI 03]. The result is an H-ANIM compliant human body model as shown in Figure III-7.

During the animation, the movement of vertices that belong to a single joint is not calculated but automatically moved as they are attached to their corresponding joint. In this way, the computation is reduced to only those vertices that have two or more influencing joints, making the deformation much faster. Duplicated vertices that are on the boundaries have the same position as well as vertex normal. Subsequently, the boundaries among segments are rendered smoothly and the segmented mesh appears as a seamless body model in the rendering view port. This method combines the speed of the deformation of segmented bodies and the visual quality of seamless bodies.

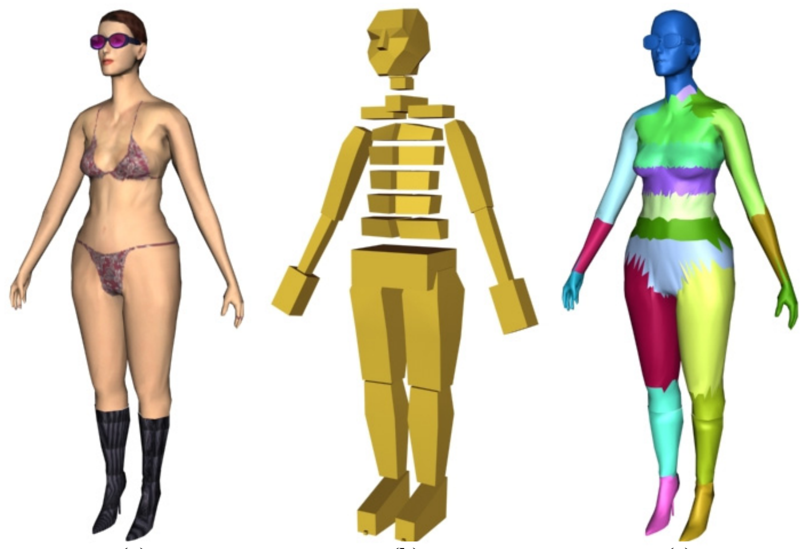


Figure III-7: Automatic segmentation of skin mesh, a: the original body, b: the skeleton, c: the segmented body

4. Garment design

The garment creation is made by using our in-house software [VOLI 02] [VOLI 03]. The garment designer is assisted in drawing 2D patterns and defining seaming lines on the borders of the garment patterns, referring to the polygon edges that are to be joined during the initial garment construction process. The patterns are then tessellated into a triangular mesh and are placed around the 3D virtual body (Figure III-8(a)). Next, the remaining garment shape is computed, as illustrated in Figure III-8 (b). The shape of the body model guides the surface of the cloth as a result of the collision response (Figure III-8 (c)).

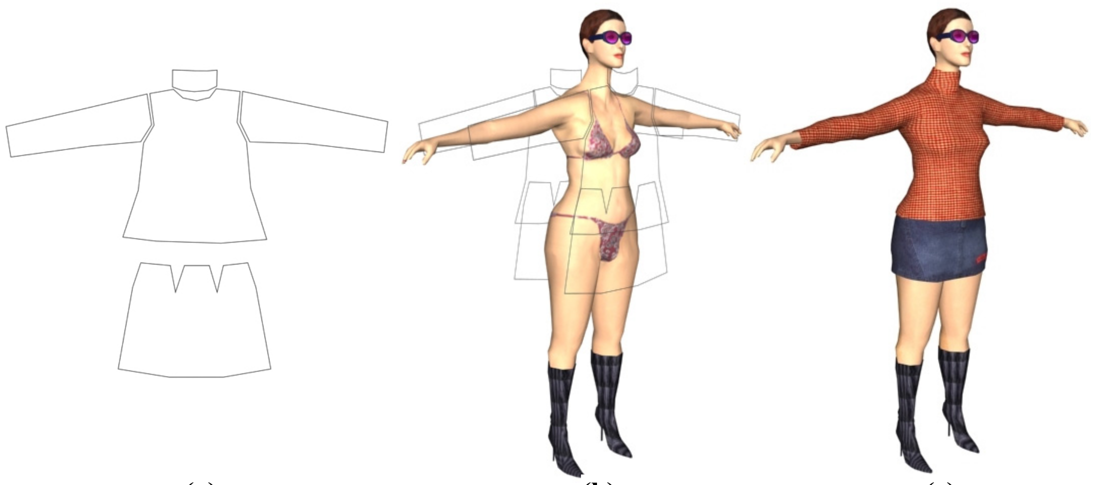


Figure III-8: The three steps of the making of garments, a: the drawing of the patterns, b: positioning of patterns around the body, c: the fitting of the clothes to the body

The cloth simulation system has been implemented on two different applications: Fashionizer (Figure III-9), which is a stand-alone program and MIRACloth (Figure III-10), a plug-in to 3D Studio Max. The main advantage of using Fashionizer is that it runs independently of the 3D Studio Max platform and therefore users do not need to know this platform. The use of MIRACloth requires mastering 3D Studio Max. In the other hand, this plug-in takes advantages of the rich features of this platform.

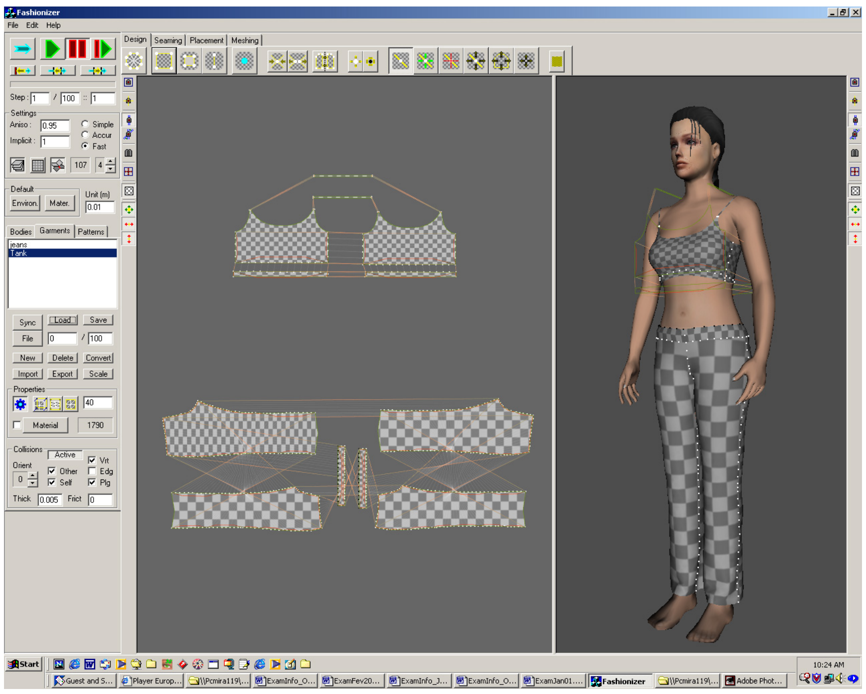


Figure III-9: Fashionizer, the garment design software [VOLI 02] [VOLI 03]

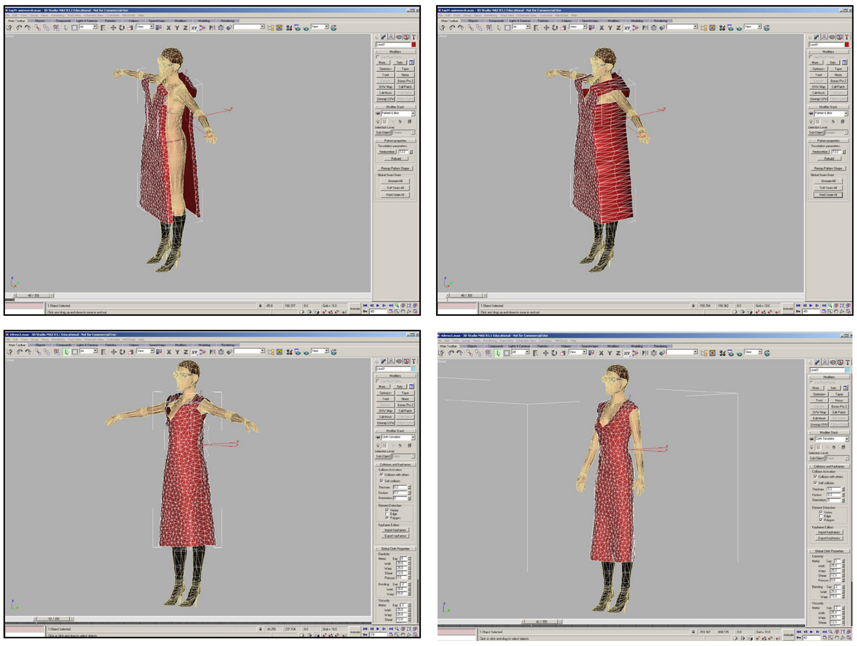


Figure III-10: MIRACloth, the cloth simulation plug-in to 3D Studio Max

Chapter IV. Hybrid approach using geometric and physically based simulation

In Chapter II, we have shown that most of the existing approaches are based on a general-purpose simulation method using collision detection and physically based simulation for the garment. Simulations that calculate all potentially colliding vertices may generate a highly realistic movement, but do not guarantee interactive frame rate. In this chapter, we investigated a new simulation model that avoids heavy calculation of the collision detection and particle system wherever possible. We do not intend to integrate yet another, more precise physical model of garment behaviour. Rather, our focus is on the real-time constraints for the simulation of the cloth features to which an observer is sensitive.

Our assumption is that the whole cloth worn on a body does not need to be simulated with a general-purpose simulation method. For instance, the computation load of the simulation of tight garments can be significantly reduced by considering that the garment surface follows the underlying skin with a constant distance. The simulation of trousers can be as well simplified by taking into account that the trouser will never collide with the arms. Collision detection can be optimized by restricting the computation of collision check only to potentially colliding areas.

We aim to develop a hybrid approach exploiting the merits of both the physically based and geometric deformations, by making use of predetermined conditions between the cloth and the body model. The method is based on the segmentation of the cloth where each segment is simulated by the most appropriate method to reduce the computation load while keeping realistic deformation. A pre-processing stage defines these groups or sub-regions over the garment depending on whether the region is tight, loose or floating. During the real-time simulation, each region will be deformed using different methods.

The chapter is composed of five sections. First, an overview of the approach is given. In the next section, the process of segmenting the cloth is described. Three other sections are dedicated to the description of the three cloth simulation methods. Finally, the last section gives the performance of the approach and describes its advantages and limitations.

1. Our strategy for reducing the computation time

The Chapter II has shown that real-time simulation of full clothes is difficult to achieve with current methods. It exists two main bottlenecks that we have to tackle. The first bottleneck is the collision detection which computation speed is a function of the complexity of the clothes and the body; the more detailed is the body and/or clothes, the slower is collision detection. The second reason of slow computation finds its root in the simulation of the mechanical properties of the fabrics. Simulating accurately these mechanical properties requires using a physically based deformation method.

Our strategy for reducing the computation time is based on the analysis of the behaviour of the clothes. When observing a garment worn on a moving character, we notice that the movement of the garment can be classified into several categories depending on how the garment is laid on and whether it sticks to, or flows on, the body surface. For instance, a tight pair of trousers will mainly follow the movement of the legs, whilst a skirt will flow around the legs. The first part of the study is to identify all the possible categories:

- Garment regions that stick to the body with a constant offset. In this case, the cloth follows closely the movement of the underlying skin surface.
- Garment regions that flow around the body. The movement of the cloth does not follow exactly the movement of the body. In case of a long skirt, the left side of the skirt can collide with the right legs.
- Garment regions that move within a certain distance to the body surface are placed in another category. The best examples are shirtsleeves. The assumption in this case is that the cloth surface always collides with the same skin surface and its movement is mainly perpendicular to the body surface.

Each of these three categories is animated with a dedicated method. The idea behind the proposed approach is to avoid the heavy calculation of physical deformation and of collision detection wherever possible, i.e. where collision detection is not necessary. The main interest of our approach is to pre-process the target cloth and body model so that they are efficiently computable during runtime. The garment is divided into a set of segments and the associated simulation method is defined for each of them. For each cloth category, we propose solutions and explain why they have been chosen.

2. Segmentation of the clothes

The approach makes use of three different simulation methods, each method being specialized to the simulation of a specific cloth category (tight, loose and floating clothes). Therefore, a pre-processing stage is necessary in order define which cloth regions will be animated with which method. The pre-processing stage consists of sorting all the cloth vertices into one of the three cloth categories (Figure IV-1(c)). This process is done manually by interactive selection.

When the garment is composed of several layers, our simulation method severely limits its target and only the outermost layer is animated. As soon as the garment segmentation is completed, hidden layers and the skin parts that are covered by the outer layer are automatically removed. Figure IV-1(d) shows the body model after such optimization on the initial model illustrated in Figure IV-1(b). The assumption made here is that we do not need to compute garments that remain invisible during the simulation.

Once the cloth pre-processing has been completed, the body model is segmented (Figure IV-1(e)) and both models are exported in VRML format together with the pre-processing data so that they can be loaded on the real-time simulation platform.

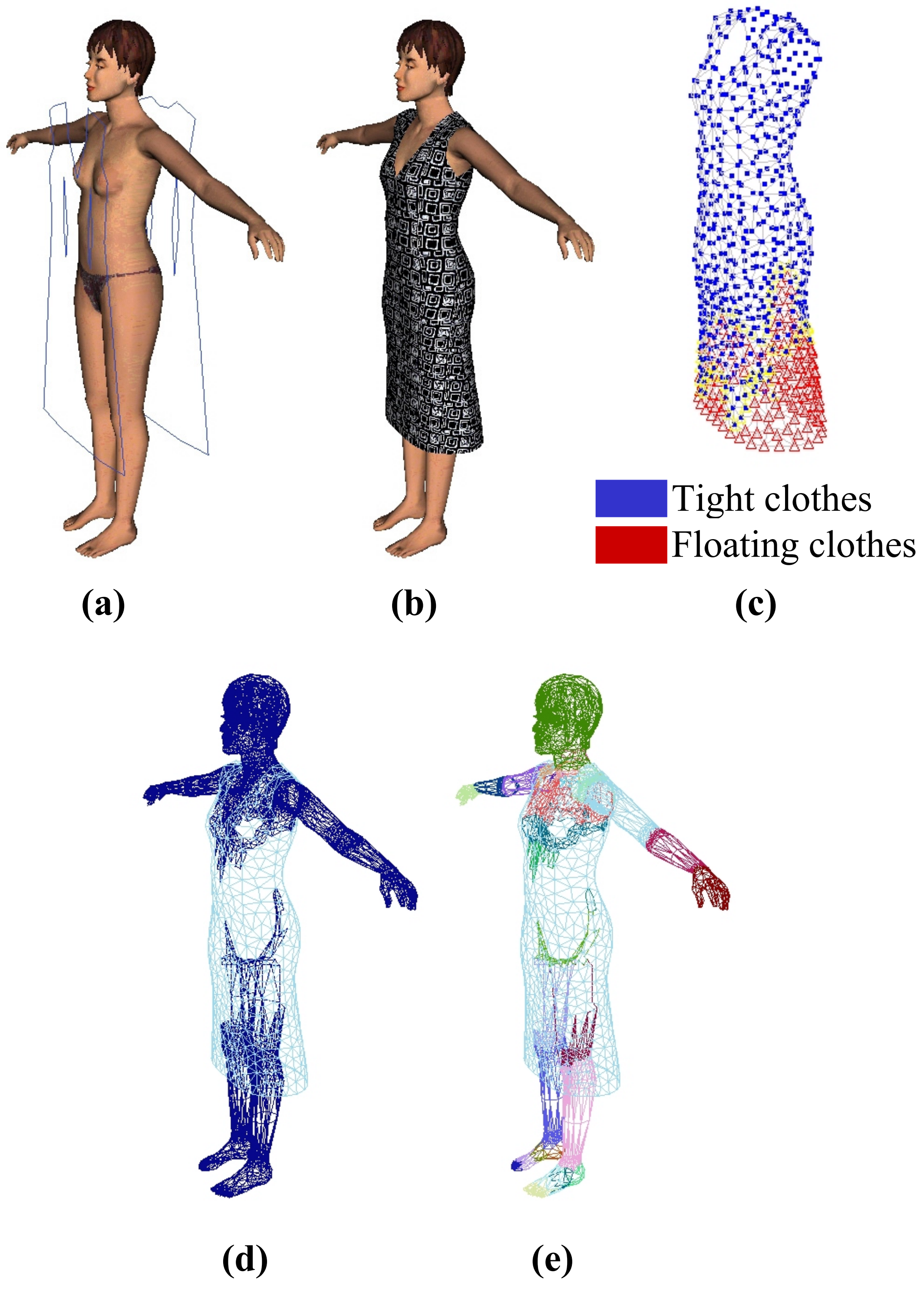


Figure IV-1: The making of the body and garments: the 2D patterns placed around the 3D body (a), the rest shape of the cloth (b), segmentation of the garment (c), skin mesh optimization (d) and skin segmentation (e)

3. “Tight clothes”

As shown in Figure IV-2, stretchy or tight clothes keep constant distance with the underlying skin surface, i.e., the deformation of these clothes follows that of the underlying skin. Therefore, we have chosen to use the skin deformation method described in Chapter III.2.2 to animate tight clothes. Since this method does not involve any collision detection or physical deformation, its computation load is negligible.

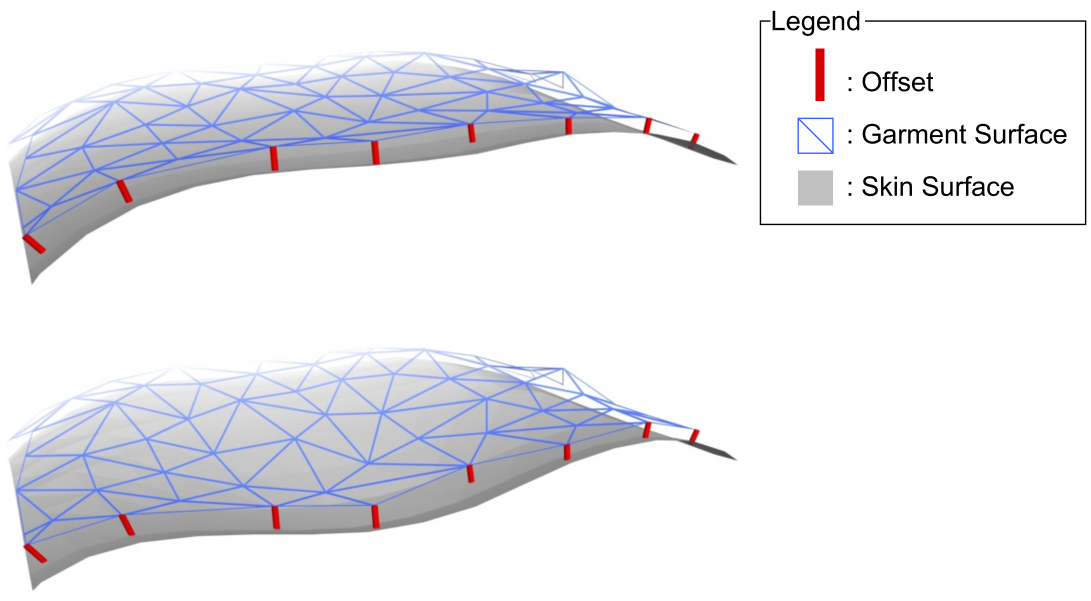


Figure IV-2: Deformation of the tight clothes

As the deformation of the tight clothes is done with the skeleton driven deformation, the skinning data of these cloth surfaces need to be computed. This information is defined by mapping the attachment information of the underlying skin to these cloth vertices. Each vertex of the garment mesh is associated to the closest triangle, edge or vertex of the skin mesh.

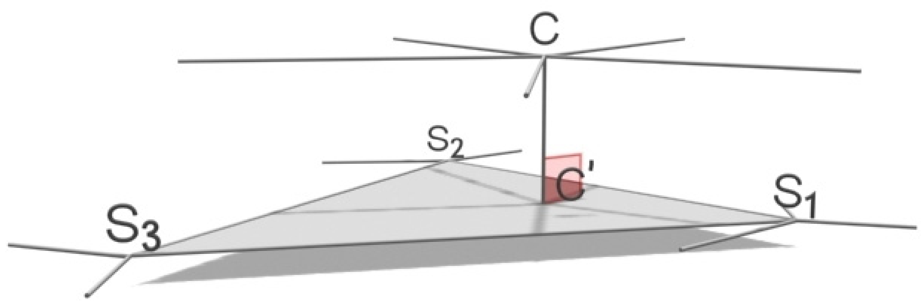


Figure IV-3: Mapping of attachment information

In the Figure IV-3, the garment vertex *C* collides with the skin triangle $S_1S_2S_3$. We define *C'* as the closest vertex to *C* located on the triangle $S_1S_2S_3$. We then define the barycentric coordinates of *C'* with S_1, S_2 and S_3. The attachment information of *C* is calculated by combining the attachment information of S_1, S_2 and S_3 weighted with the barycentric coordinates.

In conclusion, the computation cost of deforming tight clothes is very low. It is as cheap as deforming skin because both models are based on the same geometric deformation method: the skeleton driven deformation (see Chapter III.2).

4. "Loose clothes"

In case of loose clothes, the movements of clothes relative to the skin remain relatively small, keeping a certain distance from the skin surface. To have an intuitive understanding of such cases, consider the movement of sleeve in relation to the arm: for a certain region of the garment, the collision area falls within a fixed region of the skin surface during simulation. With this in mind, the scope of the collision detection can be significantly limited. A basic assumption made is that the movement of the garment largely depends on that of the underlying skin and yet it does not follow the skin surface rigidly. It is necessary to simulate the local displacement of the garment from the skin surface.

Two different methods have been developed, one for cloth deformation on the limbs (trousers and sleeves) and the other for the deformation of cloth around the torso. The reason of using two distinct techniques is due to the assumptions that we have defined differently for the limbs and torso. The assumptions that have been made for the cloth regions covering the limbs are:

- The limbs (legs and arms) are approximated with cylinders.
- The movements of the clothes on the limbs are constrained to a plan perpendicular to the main axis of the limb joint. In addition, the displacement of the cloth vertices is limited to a certain distance from the skin surface. Cloth movements such as rolling the sleeves up are not considered.
- Finally, the cross-section shape of sleeves or trousers under the gravity or acceleration is approximated with a catenary shape.

The assumptions of clothes located on the torso have been defined differently:

- The movement of these clothes is affected by the posture of the arms. The cloth model should correctly compute the cloth movement when the virtual human raises its arms. This implies that cloth vertices are interdependent; the movements of the cloth regions on the shoulders influence those located on the lower part of the shirt.

- The clothes are kept on the body. Cloth movements such as dressing or undressing are not taken into account.

4.1. Sleeves and trousers

Most of the existing simulation methods simulate the cloth behaviour by connecting the vertices to their neighbours with springs. With such interconnectivity, the movements of the vertices are highly interdependent. A special class of methods has been developed to carry this interdependence and they require high computation power (see Chapter II.1). By removing the interdependence among cloth vertices, the computation load can be greatly reduced. However, vertices need to be attached to a common base so that they have consistent movements to each other.

Our approach consists of keeping each vertex onto a transparent disc that is rigidly attached to the skin surface. The Figure IV-4 illustrates this idea. Vertices can move freely on their disc as long as they do not reach the border of the disc. These discs are perpendicular to the main axis of the joint and their size permits to control the maximum distance between the skin and cloth surfaces. The size of the discs is modified dynamically; if the cloth surface is coming closer (by the action of gravity for instance), the size of the discs is reduced and vice versa.

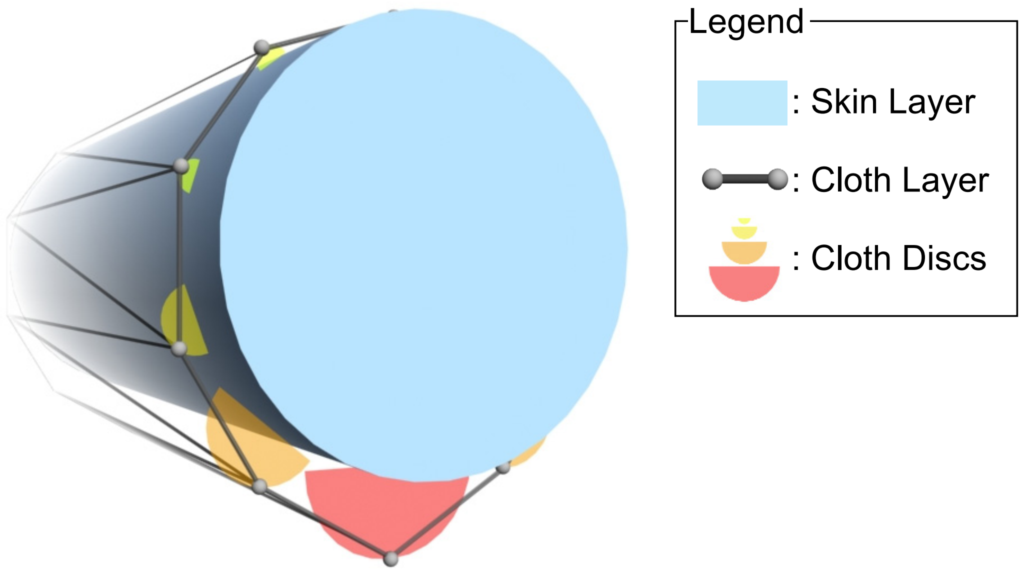


Figure IV-4: Cross section of a limb with a garment

Like the method dedicated to tight clothes (see Section 3), the simulation of loose clothes is made of two steps: the pre-processing and the runtime algorithm.

- Initially, the position of the disc centres are calculated using the attachment information defined in the pre-processing stage. They are obtained by mapping the attachment information of the underlying skin to the cloth vertices (see Section 3). The axe of the discs is parallel to the limb joint.

- During simulation, the movement of the vertices on their disc follows the equation of the rigid body motion. Vertices move independently from each other. Using the second Newton's law, velocity and position are easily calculated with the gravity force. In case a vertex leaves its disc, a kinematic correction [VOLI 00] is applied to the velocity and the position to put the vertex back on the disc surface. The Figure IV-4 shows a cross-section of a limb with a garment.

The size of the disc is a function of the angle between the cloth acceleration vector and the normal to the skin surface. The cloth acceleration vector is obtained by the subtraction of the gravity to the acceleration of the associated skin surface. The function to compute the cloth discs is defined in a way that the size of the disc follows roughly the catenary shape; i.e. the shape of a hanging wire. It can be proven that if a heavy flexible cable is suspended between two points, then it takes the shape of a curve with the equation:

$$y = c + a\cosh\left(\frac{x}{a}\right).$$

Figure IV-4 gives an example of the variation of the size of the disc along the body surface. We approximate the limbs (arms, legs…) to cylinders and then in order to determine the size of the discs for the cloth deformation we consider the equation of the circle (the cylinder that approximates the limb) and the equation of the catenary (the garment holt by the limb) as shown on the graph in Figure IV-5.

The equation of the half disc of diameter *2r* in Cartesian coordinate system (x, y) is:

$$y = \pm\sqrt{r^2 - x^2}\ .$$

The equation of the catenary in the same coordinate system (Figure IV-5):

$$y = a\left(\cosh\frac{x}{a} - \cosh\frac{r}{a}\right)$$

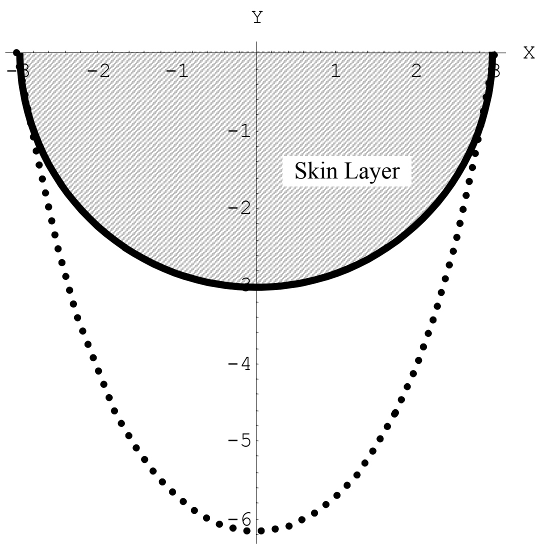


Figure IV-5: Equation of the catenary and the disc

where *2r* is the diameter of the limb and *a* is the scaling factor of the catenary curve. It is used to control the size (largeness) of the garment. We define the ideal size of the disc as the difference of the two equations above:

$$a\left(\cosh\frac{x}{a}-\cosh\frac{r}{a}\right)-\sqrt{r^2-x^2}$$

We define a quadratic equation to approximate the size of the disc. This equation should be fast to compute, as it will be used intensively to calculate the movement of every vertex at every frame. This approximation is done by fitting the polynomial function *P(x)* to the three points P_0(x=-r), P_1(x=0) and P_2(x=r).

$$P(x)=C\left(\left(\frac{x}{r}\right)^2-1\right)$$

where

$$C=a\left(1-\cosh\frac{r}{a}\right)+r\,.$$

In this equation, *C* is a constant that is related to the size (largeness) of the cloth. This parameter is defined by the user at the pre-processing stage. The term $\left(\frac{x}{r}\right)^2$ is calculated using

$$\sin\theta=\frac{x}{r}\,.$$

$\sin\theta$ is evaluated by projecting the acceleration component $\overline{OA}$ to the normal $\overline{ON}$ of the skin surface as shown in Figure IV-6. $\overline{OA}$ is the acceleration applied to the cloth. It is the subtraction of the acceleration of the skin vertex *O* to the gravity component.

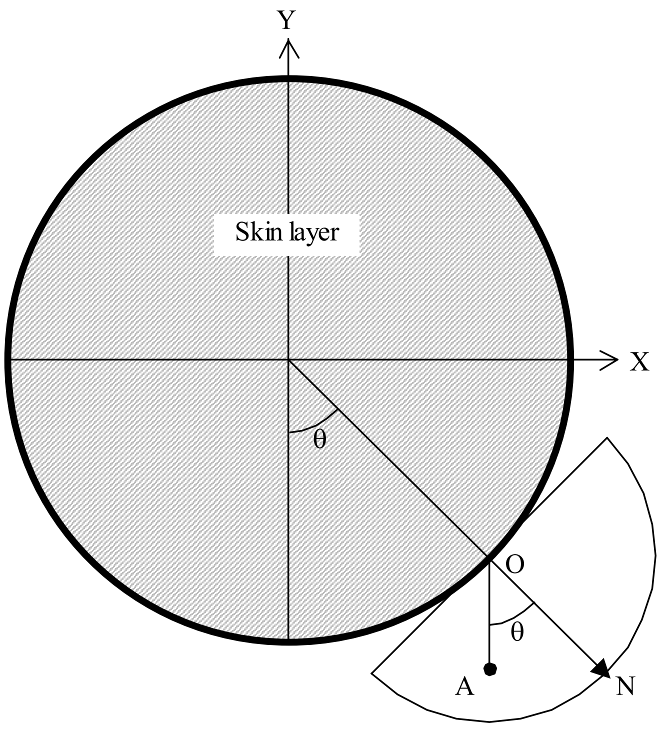


Figure IV-6: Computation of the disc diameter

This method ensures that the rest shape of the garment takes the shape of real garment hanged on a limb. All the discs containing the vertices are parallel to their corresponding limb joints. This ensures that every vertex of the same limb will fall down to the same side on the limb, even in case the direction of the limb is almost vertical. If the limb is perfectly vertical, the result is unpredictable. However, this case never happens in practice.

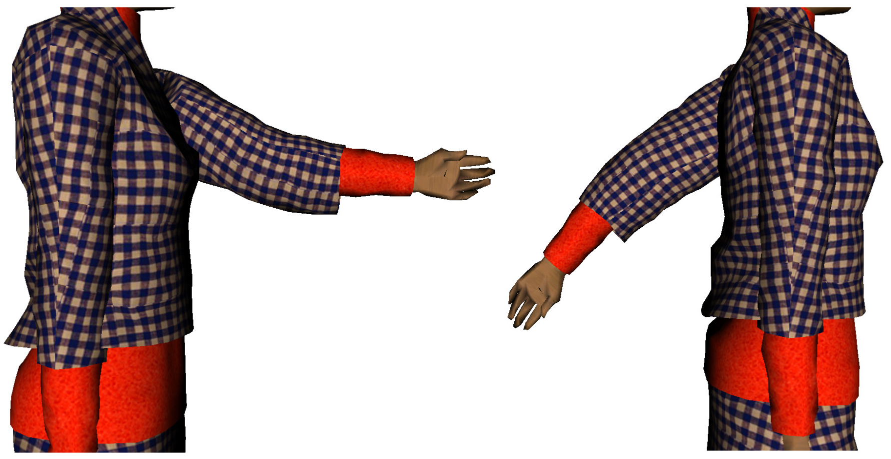


Figure IV-7: Deformation of sleeves

In this method, no collision detection on surfaces is necessary. Our particle system is very simple, each vertex moves independently. Therefore, the acceleration applied on vertices is constant. The stability of the system does not depend on the time step duration. The resulting motion of the garments provides the appearance of dynamic movements as well as the shape of real clothes. An example of a sleeve deformation is shown on Figure IV-7.

4.2. Clothes on the trunk

The hybrid technique described in Section 4.1 works well if the skin surface can be approximated by a moderately thin cylinder and the movement of the garment is more or less perpendicular to the skin surface. This assumption is true for the arms and legs; the friction prevents the garments to move along the limbs. In such cases, vertices are considered to move independently of each other. However, such an assumption cannot be made for garment regions around the trunk where, unlike the arm, the movements of the garment are driven not only by the torso but also by the potentially complex movement of the shoulder and the arm. To get an intuitive understanding of the problem, consider a shirt worn by a virtual human who is raising its arms. The whole garment around the torso part moves along the vertical direction as the arms are raised. This implies the necessity to simulate the elastic property of the tissue. Therefore, a dedicated method for cloth deformation on the trunk has been developed.

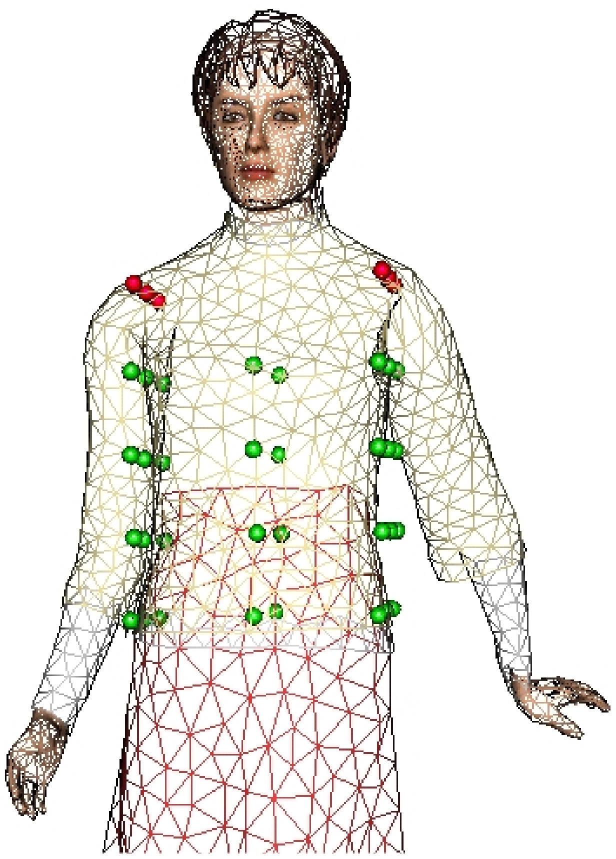


Figure IV-8: Control points in green on the trunk, red points for maintaining the mesh.

A simplified mesh composed of 42 nodes is animated using a simplified mass-spring system. The movement of these nodes is driven by two interactions: gravity and springs in connection with neighbouring nodes. Each of these nodes is a control point used to deform the cloth mesh on the trunk. Figure IV-8 on colour section shows an overview of the simplified mesh. These control points are uniformly placed on the trunk, they form a grid of 3×4×3 points. Six additional control points are attached to the shoulder. These control points maintain the grid mesh on the trunk and prevent the mesh from falling due to the gravity. Each control point is included in a half-sphere that is attached to the skeleton. The collision detection algorithm verifies if each control point is contained within its corresponding hemisphere. In case the control point is not in its hemisphere, a kinematical correction is applied on the position and the speed.

The cloth mesh is deformed with the Freeform Deformation method (FFD) [SEDE 86] using the position of the nodes of the simplified mesh. Our implementation uses a 3×4×3 control point lattice for the FFD and uses Bernstein polynomials as the basis functions. The example in Figure IV-8 shows the grid, composed of the 36 green nodes. The weight values associated to the vertices, deformed by the FFD, are pre-calculated.

5. "Floating clothes"

Cloth regions categorized as "floating clothes" are composed of vertices that freely flow around the body. This will take care of such cases as a large skirt floating around the legs. Unlike tight clothes, floating clothes are slightly affected by the body. Their movements are similar to those of the flags or tablecloths and cannot be defined as a function of the joint angles of the avatar skeleton. The simulation of these floating clothes uses the classical approach with particle system and collision avoidance.

5.1. Particle system

The mechanical behaviour of the clothes is reproduced with a mass-spring system. Two sorts of forces are considered: gravity and the forces exerted by the springs connecting the particles to each other. The simulation is performed using the Implicit Euler Integration proposed by Baraff et al. [BARA 98] and Volino et al. [VOLI 01].

5.2. Collision detection

Calculating collisions between the cloth and the skin mesh would not be feasible in real-time; instead, collision detection is performed on a simplified model of the body. Given the assumption that the floating clothes are mainly skirts, the collision detection is calculated for the legs only. The legs are modelled as cylinders; the collision detection comes down to computing the distance between the cloth surface and the axis of the leg joints. It is possible to further optimize by restricting the number of collision distances that we compute for each segment. During the pre-processing stage, a list of possible colliding vertices is defined for each segment. The criteria of selecting vertices for collision check are the distance between the vertices and the legs and the normal orientation for each vertex on the skirt on the initial model.

Note that only the vertices are taken into account in the collision detection. A more robust method would compute collisions on edges and triangles as well. However, the collision check on vertices is sufficient because the edge length of the cloth mesh is small. A collision on edges or triangles automatically implies collisions on the neighbouring vertices.

In addition, the self-collisions are not taken into account. Self-collisions play an important role for the modelling of the cloth-to-cloth interactions, especially when the clothes are composed of several layers of garments. However, our cloth simulation method deforms only the outermost layer (see section 2); the absence of self-collisions does not degrade the final results.

5.3. Collision response

Two classes of collision response methods have been developed over years, the penalty forces and constraint dynamics [WITK 01] [VOLK 00].

Collision response based on penalty forces consists of generating repulsive forces between the two objects being in collision to prevent them from further penetrating each other. This method requires generating forces of high strength to avoid the collisions, making the differential equations of the simulation highly stiff. These stiff differential equations may cause instability in the simulation when being processed by the numerical solver.

Constrained dynamics is preferable because accurate collision response can be achieved without using forces of high strength. Instead of modelling the collision response as repulsive forces, the constrained dynamics consists of modifying the existing forces of the clothes so that the cloth surface does not go further inside the body. Constrained dynamics has been proven to be accurate [WITK 01] [VOLK 00]; this method has been chosen for our collision response.

The collision response consists of cancelling the normal components of the velocity and acceleration (see Figure IV-9) to prevent the vertex from moving closer to the skin surface.

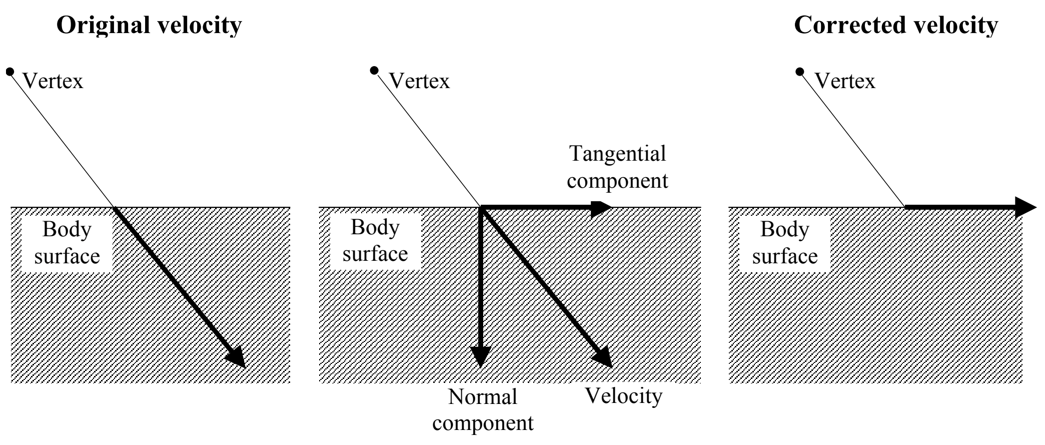


Figure IV-9: Correction of the velocity

Note that the collision response model does not integrate the friction force that acts against the sliding movements of the clothes on the skin surface. This force could be modelled by modifying the strength of the tangential component of the velocity and acceleration. The corrections of the velocity and acceleration are applied gradually as shown on Figure IV-10. If the vertex goes closer to the body surface, the position is modified as well.

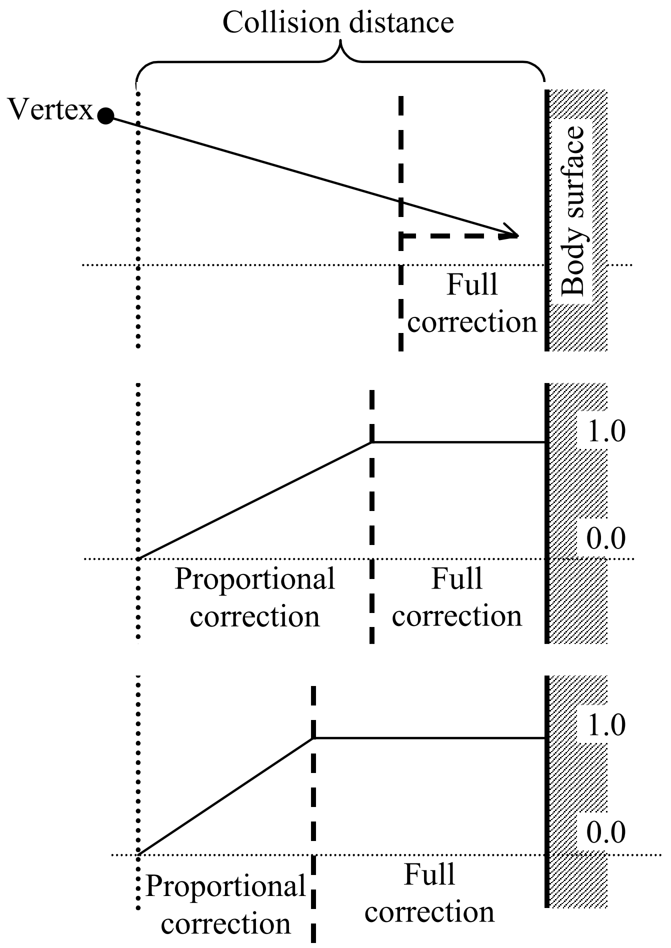


Figure IV-10: Gradual collision response

6. Deformation on segment boundaries

Extra computation is required in order to keep smooth deformations among the cloth segments. This smoothing is necessary because the segments are animated with different methods. The position of the vertices located on the boundary of two segments is computed twice, once by each method of the adjacent segments. The final position is obtained by a weighted summation of the two positions. The value of the weight is dependant on how much the vertex belongs to the segment.

7. Surface tessellation

Due to the time constraint, the maximum number of vertices simulated with the physically based simulation method used is limited. In order to improve the visual quality of clothes, we have implemented a tessellation algorithm that replaces flat triangles with curved patches and higher-order normal variation. Such technique offers an efficient solution to increase the number of polygons while keeping low computation cost.

After calculating the deformation of the three layers, each triangle is decomposed into three sub-triangles. The positions of the newly inserted vertices are calculated using the method described by Vlachos et al. [VLAC 01]. Surface tessellation is mainly used for the floating cloth regions where the computation load is high. During the pre-processing stage, the user can define triangles that will be tessellated during the real-time animation (Figure IV-11).

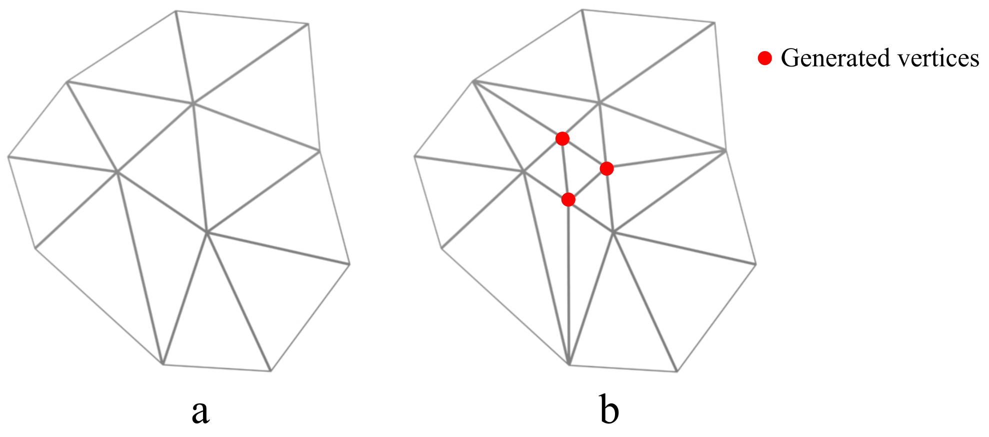


Figure IV-11: Tessellation of a triangle, before (a) and after (b)

During the simulation, the topology of the tessellated mesh is kept unchanged. However, the position and the normal of the generated vertices have to be updated according to the position of the neighbouring vertices.

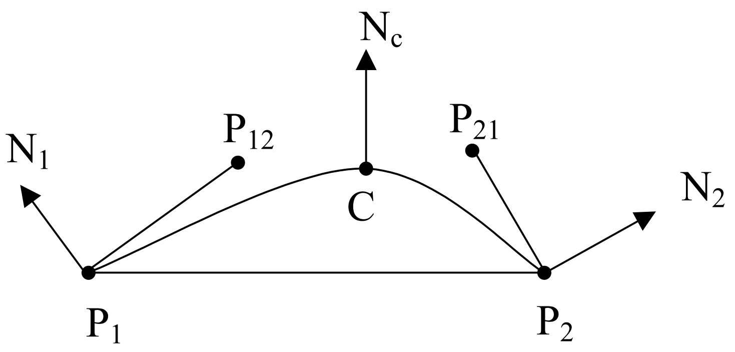


Figure IV-12: Quadratic Bezier interpolation

The position of the generated vertex C is computed using the quadratic Bezier spline. The algorithm starts by computing the two control points P_{21} and P_{12} (see Figure IV-12).

$$w_{ij} = (P_j - P_i) \cdot N_i$$

$$P_{12} = (2 \cdot P_1 - P_2 - w_{12} \cdot N_1)/3$$

$$P_{21} = (2 \cdot P_2 - P_1 - w_{21} \cdot N_2)/3$$

$$E = (P_{21} + P_{12})/2$$

$$V = (P_1 + P_2)/2$$

$$C = E + (E - V)/2$$

The normal is computed by linear interpolation of the normal vectors of the two end points P_1 and P_2.

$$N_c = (N_1 + N_2)/2$$

8. Results and discussion

Our method has been tested on several dressed virtual humans. Figure IV-13 illustrates the computation time with an increasing number of cloth triangles. The code was executed on a 1GHz PC with 512 Megabyte RAM and a GeForce2 graphics card. The computational time does not include the deformation of the avatar skin and the real-time rendering. Note that the time of computing a frame must be lower than 40 milliseconds for real-time applications.

The deformation of floating regions is in average thirteen times slower than that of the loose regions. This is because the deformation of loose clothes is based on an optimized model with simplified collision detection and without numerical integration. The deformation of tight regions is even faster because it makes use of the skeleton driven skin deformation. The floating region algorithm is able to simulate a maximum of 1,000 non-tessellated polygons in real-time. In most cases, this is sufficient to produce aesthetically pleasing results. Most of the clothes can be categorized as tight or loose clothes. Limited number of clothes falls to the third category.

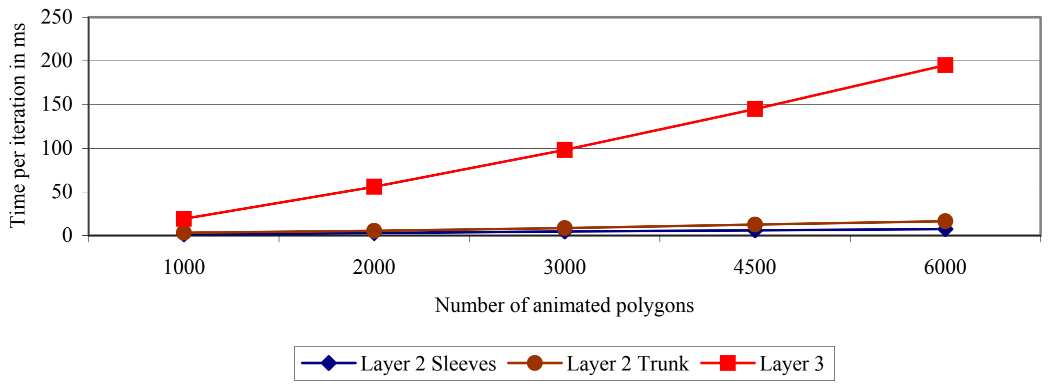


Figure IV-13: Computation time per iteration versus number of triangles

Figure IV-14 and Figure IV-15 demonstrate three examples of real-time animations. These simulations (rendering, cloth and skin deformations) ran at a speed of 25 to 30 frames per second (fps).

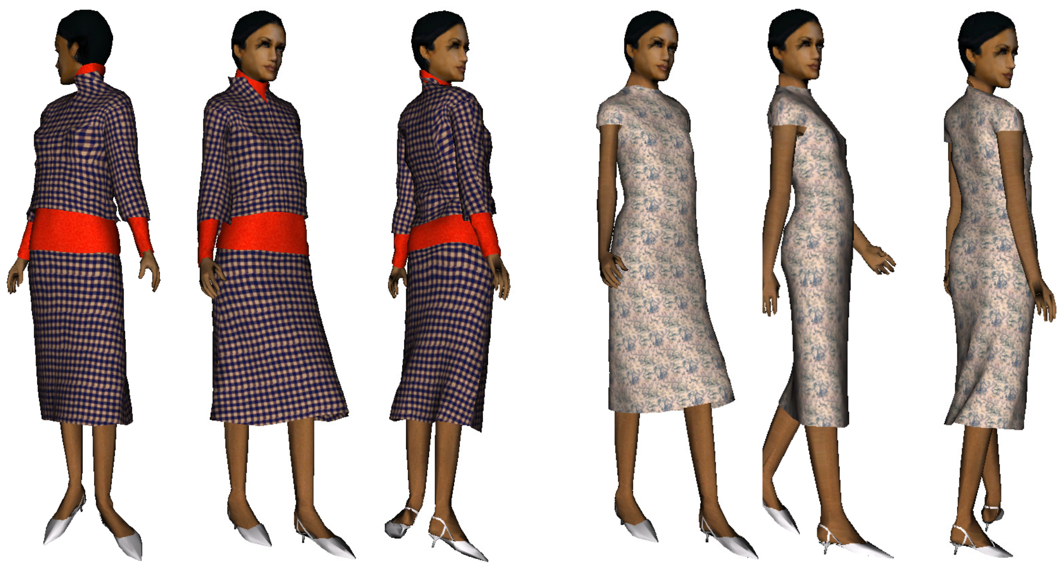


Figure IV-14: A virtual human with a skirt and jacket

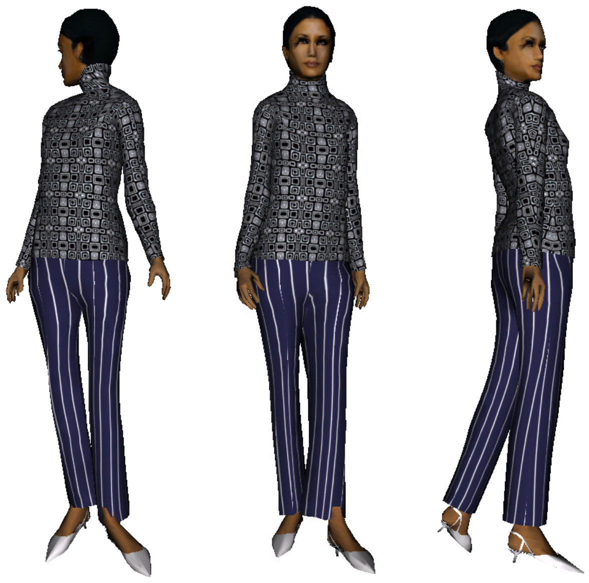


Figure IV-15: Virtual human with a trouser and tight pullover

The 4-12 summarizes the performances of these three animations.

Model Type / Nb. of Polygons	Skirt and Jacket	Dress	Trouser and Pullover
Original model Cloth & Body	8413	5887	7445
After optimization Cloth & Body	4874	4526	5187
On Layer 1	542	847	2697
On Layer 2	1505	0	0
On Layer 3	618	885	593
Performance (fps)	29	26	25

Table IV-1: Performance table

Our experiment has demonstrated that simulating fully dressed virtual humans with middle-range PC is feasible. Such an approach has certain number of limitations due to its optimization.

No self-collision is calculated. In addition, collision detection is restricted to garments and their associated body segment: right sleeve with right arm, left sleeve with left arm, etc… For instance, simulating the hand grasping a dress is not possible with such approach. However, this approach is applicable for any movement that does not involve body-cloth interactions such as grasping clothes.

Another limitation of the proposed approach is its physical model for the loose regions. The cloth deformation on these regions does not define any interaction among vertices. Vertices move independently to each other and their movements are restricted to the plan perpendicular to the limb. Moreover, it does not allow simulating the tensile behaviour of garments such as wrinkles.

9. Conclusion

In this chapter, we have proposed an approach that is able to animate in real-time fully dressed virtual humans. Instead of calculating all potentially colliding vertices with physical simulation, we used the fact that not all the cloth vertices need to be animated with a versatile physical method. The clothes are partitioned into regions, each of which is associated with the most appropriate method that fulfils the minimum requirement: visual realism.

Nevertheless, the use of this method has revealed several limitations. First of all, it has been found that the automatic segmentation of the clothes into the three regions does not work properly. Most of the time, it requires manual work from CG artists. The main reason is that the segmentation is made using the distance between the cloth and the skin surface on the initial cloth model. Cloth vertices located far from the skin surfaces are considered as floating region whereas cloth regions close to the skin surface are categorized as tight or loose regions. The static model of the clothes provides too few information; using a static shape of the cloth is not sufficient to know in detail its behaviour.

Secondly, it has been found that the automatic processing is preferable to the manual work done by CG artists. The user interaction is required at many steps of the pre-processing stage such as the identification of the three cloth regions (see Section 2) and the selection of the triangles to be tessellated (see Section 7). This manual work requires knowing how the clothes may behave in response to the body movements. Therefore, CG artists need a training period during which they get familiar with the user interface. Even after learning the software in detail, it is rare that CG artists succeed at processing clothes at the first shot; most of the time, the making of the clothes requires several trials.

Finally, the simulation method is not able to generate accurately some of the cloth details such as the wrinkles. This remark is especially true for the garments that are classified as tight or loose regions. The method also fails at simulating certain kinds of the clothes such as cloak. These clothes are categorized as floating region but they are located on the upper part of the body. The collision detection that is performed on the floating region takes into account the lower limbs only. Extending the collision detection to the full body would require so much computation power that the simulation would not run in real-time anymore.

The proposed method fulfils our principal goal – real-time performance. However, we believe that the approach can be improved by analysing carefully the reasons of its limitations.

This chapter has introduced some interesting ideas/concepts that are worth further investigating: the segmentation of the cloth surface into three regions and the attachment of the cloth to the underlying skin surface. In the other hand, the use of a static model of the cloth for the pre-processing has severely limited the efficiency of the approach. Using a pre-computed animation sequence of the clothes would make possible a finer analysis of the cloth properties. This is the purpose of the next chapter where we introduce the cloth simulation by examples.

Chapter V. Example-based approach

In the previous chapter, we have described the first method for real-time cloth deformation. Garments are segmented into pieces that are simulated by various algorithms, depending on how they are laid on the body surface and whether they stick or flow on it. Its two main limitations are 1) the inability to simulate accurately the wrinkles of the garments and 2) the need of the user input for some of the steps of the pre-processing stage. The main reason for these drawbacks is the use of a static clothes model, which provides little information about the cloth behaviour.

In this chapter, we propose a different approach based on the analysis of a pre-computed, physically based cloth simulation sequence rather than a static shape. The robustness of the segmentation stage (classification of the vertices to tight, loose and floating cloth regions) will be improved. In addition, the pre-computed cloth animation will make possible the generation of the cloth details in real-time.

This chapter is organized as follows: The first section overviews the example-based cloth simulation method and the chosen strategy for reducing the computation time. The second section gives a description of the first level of deformation. The pre-processing stage where the initial mesh is segmented and the rough mesh is constructed is detailed. This section describes as well how the deformations and collisions are handled on the rough mesh. The third section is dedicated to the description of the second level of cloth deformation. The process of building the interpolation function to generate the final cloth shape for real-time simulation is described in detail. Finally, we will close this chapter by demonstrating the application of this method on several clothes. We will also compare this approach with the previous one described in Chapter IV.

1. Our strategy for reducing the computation time

Prior to searching for an efficient cloth simulation method, it is necessary to make a further analysis of the cloth movements. This will help us to determine what can and what cannot be pre-computed for the real-time cloth simulation. The cloth movements can be classified according to two levels of detail:

- The global movement of clothes. These movements can be viewed as group behaviour of several neighbouring cloth vertices. If the garment has freedom from the body surface, the movements are determined by the law of dynamics. On the contrary, if the garment is tight, it follows the skin shape. In this case, its deformations can be approximated a geometric method similar to the one used for skin deformation.
- The details of the cloth surface. This movement mainly concerns the deformation of wrinkles. Wrinkles exist due to the local constraints exerted on the garments. Wrinkles appear mostly when the garment is shrink and disappear when it is stretched. Their location on the garments can be random or fixed whether the garment tissue has a memory shape or not. Wrinkles are best simulated with a highly discretized mesh (i.e. high number of polygons). It follows that the simulation of wrinkles requires high computation power. Fortunately, the cloth-wrinkling problem is geometric in nature [TOSI 90], [HADA 99] and [HING 96]. Generating realistic wrinkles can be achieved by geometric deformation that requires less computation time than physically based models.

Cloth details are rather difficult to generate in real-time due to the high number of triangles that is required. Different approaches have been proposed in the research literature, for simulating the cloth wrinkles while keeping low computation load. Some of the researchers have used geometric models such as cubic spline curve [KANG 01] [KANG 02] (Chapter II.3) or sinusoidal curve [TOSI 90]; some others have simulated the wrinkles with textures [HADA 99] (Chapter II.2). We propose to animate the wrinkles by deforming the mesh using an example-based approach. Unlike textured wrinkles, wrinkles simulated directly by the shape of the cloth mesh offer higher degree of realism. In addition, compared to geometrically deformed wrinkles, example-based approach makes it possible to simulate accurate wrinkles as they appear on the pre-computed cloth simulation.

The idea of using examples to build an interpolator has been widely used in CG, such as for modelling a variety of human body shapes [SEOH 03], for skeleton-driven character skin deformation [ALEN 02], and motion synthesis [ARIK 03]. The basic idea is to construct a database of pairs of input parameters and the targeted graphical objects. During the run-time simulation, given an input, the interpolator will produce the corresponding output by interpolating the examples available in the database. Unlike skin deformation or motion synthesis, the complexity of building a cloth interpolator from examples is much higher because of the dimension of the input parameters. Clothes are dynamic objects and their movements depend not only on the input parameters (i.e. posture of the 3D character that wears the clothes), but also on their velocity and position. Very recently, James et al. [JAME 03] have demonstrated that example-based approach can be adapted to the special case of physically based objects (see Chapter II.4). The main disadvantage of their approach is the limited interaction that the user can exert on the cloth. The approach cannot be used for the simulation of clothes on bodies.

The use of example-based approach requires carefully choosing the input parameters. Too few input parameters will lead to a simulator that offers few possibilities of user interaction. The simulator will play back a pre-recorded simulation with few possibilities of interaction. Too many parameters will make the construction of the interpolation not feasible. The dimension of space of examples will be too vast and the example distribution will be too sparse. The main difficulty of example-based approach is how to build a versatile simulator (i.e. with a high number of input parameters) while keeping the construction of the interpolator feasible. The problem is particularly acute for the case of clothes because these objects have dynamic behaviour. Their movements are function of the position and velocity of all the vertices, making the number of necessary input parameters higher. Our strategy is based on the classification we have defined for the cloth movements:

- Global cloth movements will be simulated with a physically based method using a mass spring system. The cloth mesh will be segmented into a set of patches. Each patch will be a vertex of the rough mesh. The topology of the rough mesh will be constructed by computing the connectivity among patches.
- Details of the clothes are generated with an example-based approach. Given the shape of the rough mesh computed at the first stage, the method will generate the corresponding fine mesh of the clothes.

The overview of the architecture is given on the Figure V-1. The data structures for both the rough mesh simulator and the wrinkle interpolator are computed during the pre-processing stage. The initial cloth mesh is first segmented into a set of patches; each of them is considered as a vertex of the rough mesh. The runtime simulator first starts by computing the deformation of the rough mesh. The final shape is then generated using the wrinkle interpolator.

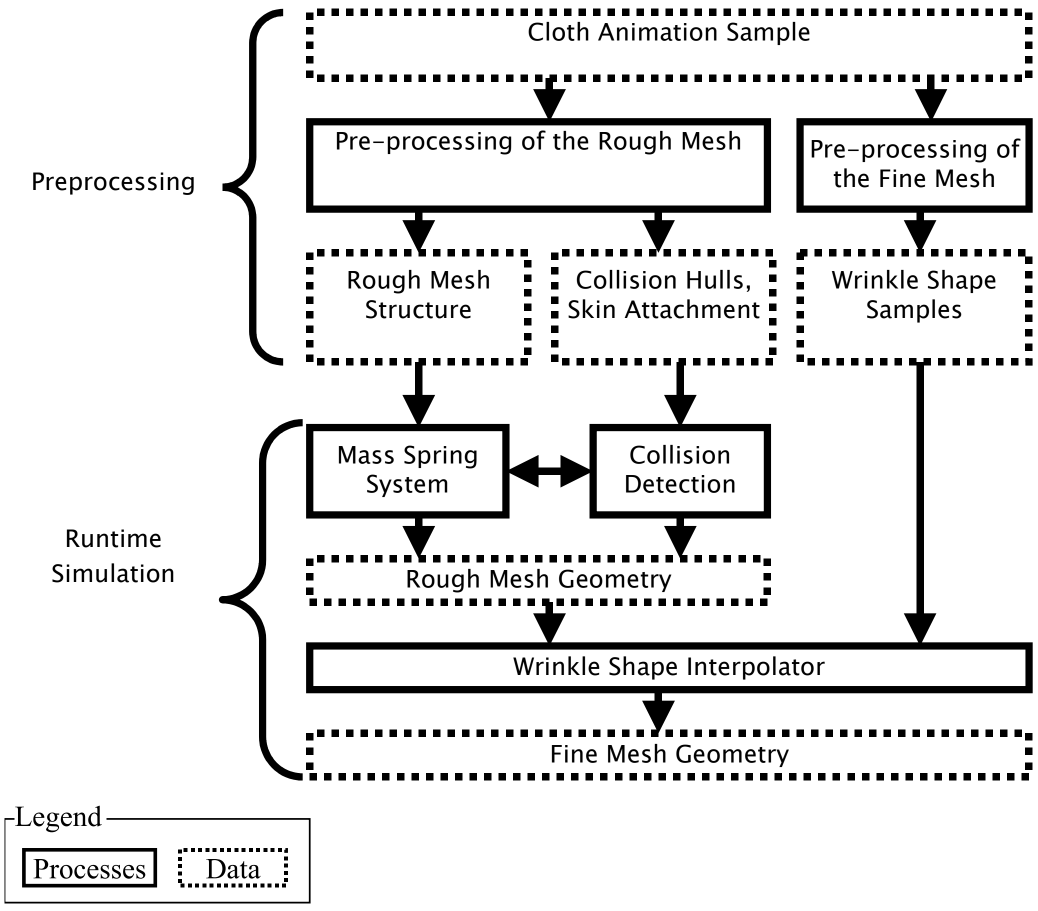


Figure V-1: Overview of the architecture

2. Simulating the gross behaviour of clothes

Due to the computational expenses of solving the full numerical system of the physics-based deformation, we seek simplifications by constructing a coarse mesh representation of the garment. The coarse mesh is used to deduce the gross behaviour of the cloth in a data-driven manner, based on the input pre-simulated sequence. A number of optimization strategies are adopted: The two following sections describe a pre-processing that constructs and segments a coarse mesh representation into different region types. We then describe in the next two sections the spring-mass system and collision handling of the coarse mesh at each frame of the simulation. Also described is the runtime process.

The dynamic behaviour of cloth can be best simulated with a physically based method. However, due to the computational expenses of solving the full numerical system of the physics-based deformation, we seek simplifications by constructing a coarse mesh representation of the garment. The coarse mesh is used to deduce the gross behaviour of the cloth in a data-driven manner, based on the input pre-simulated sequence. Figure V-2 shows an overview the pre-processing and the runtime simulation of the dynamic behaviour. We first segment the initial mesh into patches by analyzing the movements of the vertices on the animated cloth from the physically based pre-computation. A rough mesh is then constructed by computing the connectivity of these patches. In addition, data structures for collision detection – collisions hulls and skinning data – are also constructed using the pre-computed animation. The runtime simulation is based on the simulation of the generated rough mesh with a particle system (see Chapter IV.5.1).

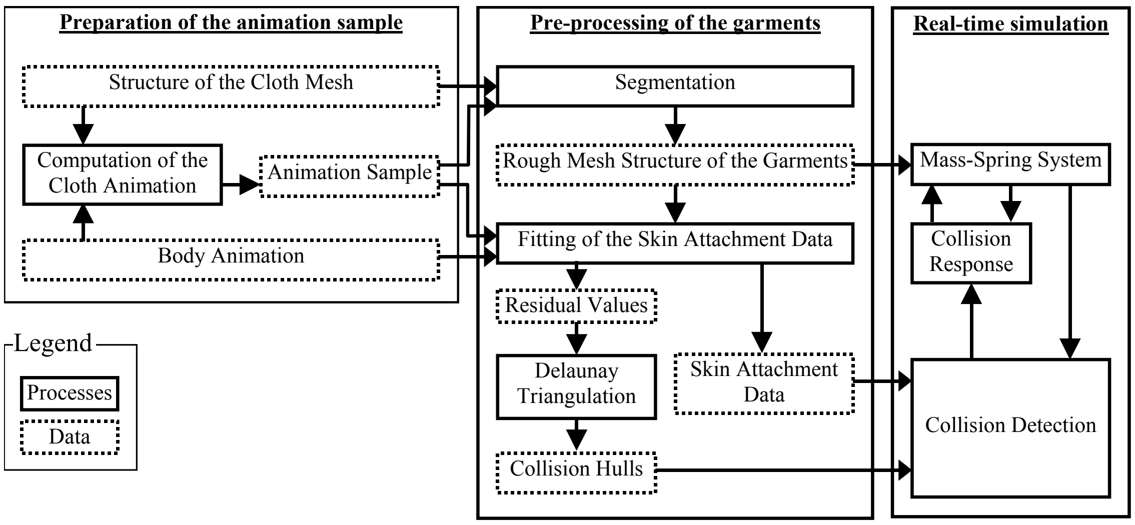


Figure V-2: Overview of the pre-processing and the runtime simulation of the rough mesh

2.1. Segmentation

The main purpose of the segmentation is to generate the rough mesh used for the simulation the dynamic behaviour of the clothes. Each patch on the initial mesh forms a vertex on the rough mesh.

The segmentation is performed by grouping vertices into patches using a cost function. The segmentation starts by searching a vertex that has not yet been attributed to a patch. This vertex is considered as the first vertex of a new patch. The patch is then grown by adding neighbouring vertices. The choice of the vertex to add to the patch is based on the value of a cost function computed for each neighbouring vertex. This cost function takes into account three criteria:

- Minimizing the displacement among vertices belonging to the same patch.
- Minimizing the "shape factor", sqrt(Surface Area)/Contour Length. The objective is to obtain "well-shaped patches", patches that have circular shape.
- Obtaining patches of equivalent surface area.

The construction of the patch is finished when the minimal cost among all the vertex candidates is above a threshold.

2.1.1. Rigidity

One of the criteria to attach a vertex to a patch is its relative movement in relation to the vertices that already belong to the patch. This is done by computing the distance variation between the candidate vertex v_{cand} and other vertices of the patch $v_{segment}$ for the whole pre-computed animation, as evaluated by:

$$F_{rig}(v_{cand}) = \sum_{i \in Animation} \left\| \left(v^{i}_{cand} - v^{i}_{segment}\right) - \left(v^{i+1}_{cand} - v^{i+1}_{segment}\right) \right\|$$

v^{i}_{cand} and $v^{i}_{segment}$ are respectively the position of the vertex candidate and one vertex of the patch at frame *i*. The function *F* computes the relative displacement between the candidate and the patch vertex between two consecutive frames of the pre-computed animation.

2.1.2. Shape factor

A semi-regular mesh is required in order to obtain accurate simulation of the cloth mass-spring system [VOLK 00]; in other word, the rough mesh should not have any degenerated triangle. One way to obtain regular patches set is to construct them in a way that they have circular shape. The implementation of such criterion is straightforward: the cost function evaluates, for the vertex candidate, the shape factor of the patch, i.e. the ratio of the surface of the patch to its circumference. We remind that the disk is the shape that has the maximum area for a given circumference. The vertex that has the smallest ratio is the best candidate.

$$F_{shp}(v_{cand}) = \frac{\mathrm{sqrt}(F_{surface})}{F_{circumference}}$$

$F_{surface}$ and $F_{circumference}$ denote the surface and the circumference of the patch including the vertex candidate, respectively.

The cost function is a weighted summation of the three functions described above, as given by:

$$F_{t}(v_{cand}) = \alpha_{rig} \cdot F_{rig}(v_{cand}) + \alpha_{surf} \cdot F_{surf}(v_{cand}) + \alpha_{shp} \cdot F_{shp}(v_{cand})$$

2.1.3. Patch surface

The second term of the cost function provides a simple way to control the size of patches by penalizing large and small surface area. The cost function is given as below:

$$F_{surf}(v_{cand}) = \left\| F_{Surface} - F_{reference} \right\|$$

$F_{Surface}$ denotes the surface area of the patch with the vertex candidate, which is simply computed by the summation of the surface area of each triangle that belong to the patch in consideration. $F_{reference}$ is the reference surface area, the area that we want the patches to have. The modification of this value allows controlling the size of the patches. Figure V-3 shows three examples of segmentation using different value of $F_{reference}$.

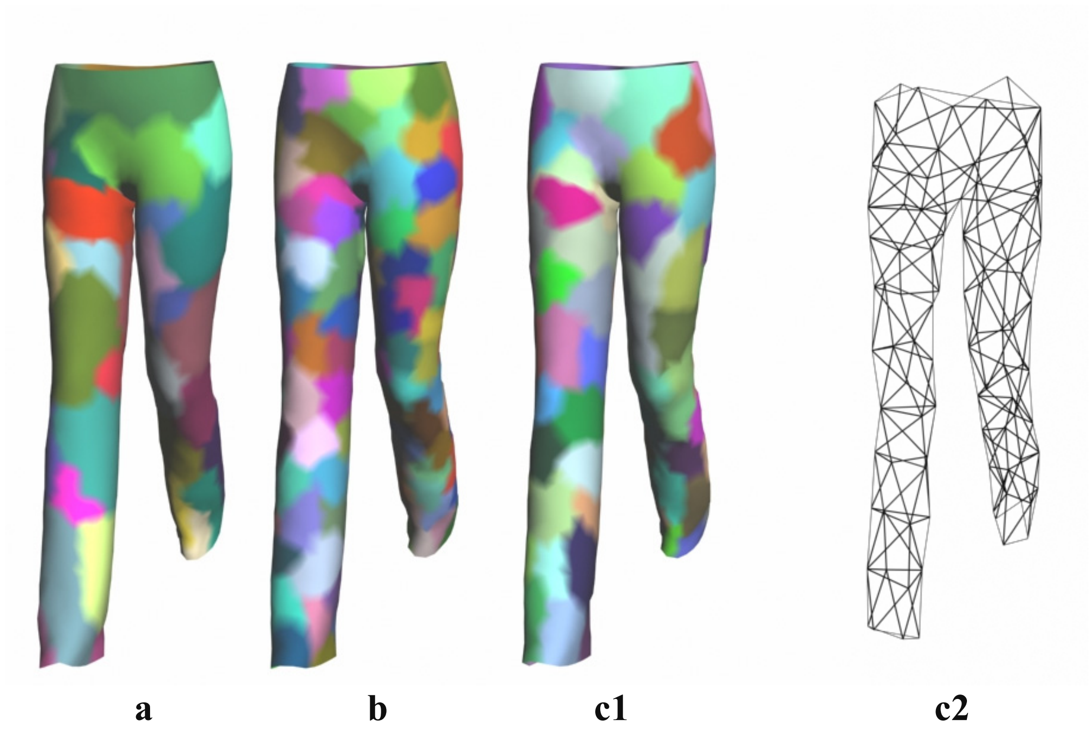


Figure V-3: Three examples of segmentation with patches of large (a), small (b) and medium surface area (c1). (c2) is the rough mesh corresponding to the segmentation (c1). The different colours are randomly assigned to the different patches.

2.2. Pre-processing of the rough mesh and collision detection

We identify two main causes that explain the garment movements. The first cause is the dynamic behaviour inherent to clothes. Such movements are best simulated by a mass spring system. For some of the large clothes such as long skirt, these movements are most prominent; unlike tight trousers, the collisions of the skirt with the legs occur rather rarely. The second cause of cloth movements is the interactions with the body surface, which can be well formulated by the skeleton driven deformation. For example, a very tight trouser can be considered as a second skin layer and therefore can be simulated using the same deformation as the one used for the skin surface. Our approach for deforming the rough cloth mesh makes use of a combination of these two types of deformations, skeleton-driven deformation for tight garments and mass-spring system for loose garments.

- The position of all the vertices of the rough cloth mesh is first calculated with the skeleton-driven deformation. These computed positions are considered as anchorage points of the cloth vertices. Their main role is to approximate the position of the underlying skin surface. The chosen method for skeleton-driven deformation has been presented in Chapter III.2.2. In addition to the position, this method computes as well the local transformation matrix of each anchorage point.

- The simulation of the cloth dynamics is made with a mass spring system and the collision detection of each vertex is computed in the local transformation matrix of its associated anchorage point.

The main advantage of using this two-stage approach is to lessen the computation time required for collision detection. Collision detection is known to be one of the bottlenecks of cloth simulation (see Chapter II.1.3). Most of the existing collision algorithms come down to checking collision among all the vertices belonging to the body and cloth meshes. By associating an anchorage point to each cloth vertex, we can restrict the collision check to the local coordinate of the anchorage point. Moreover, tight clothes do not require collision check because they follow the skin surface and therefore they remain immobile in the coordinate system of their associate anchorage point.

2.2.1. Cloth anchorage points

Each vertex of the mass spring system (i.e. rough cloth mesh) is associated with an anchorage point which movements follow those of the underlying skin. The main purpose of the cloth anchorage points is to provide a local coordinate system where the collisions can be efficiently resolved. If the cloth is tight, its movements in the coordinate system of the cloth anchorage point will be negligible. On the contrary, the movements of a loose cloth will occupy a volume centred at the origin of the coordinate system of its associated anchorage point. The collision check will therefore come down to keeping the cloth vertex in that volume. Later in this dissertation, we name this volume as collision hull.

The position of cloth anchorage points is computed with our skeleton driven deformation method (Chapter III.2.2). This method is based on the matrix blending computed with the matrix operator presented by [ALEX 02]. The blended matrix is obtained by a weighted combination of the transformation matrix of the influencing joints as shown in Figure V-4.

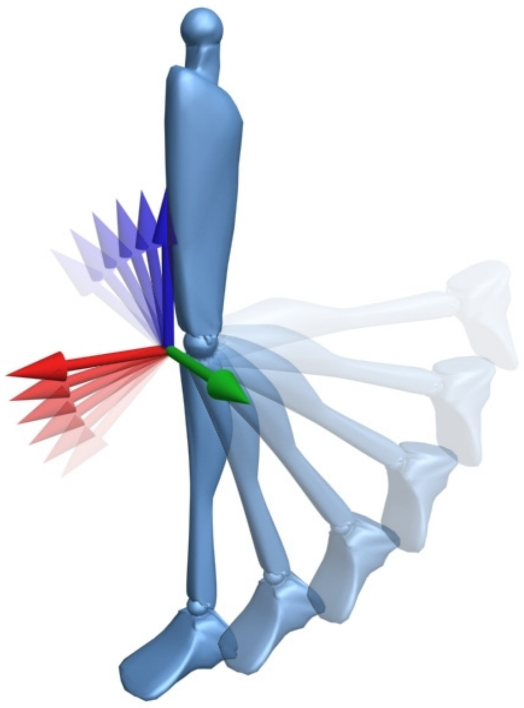


Figure V-4: The result of the blending of two transformation matrices with a weight of 0.5 for each of them; while the lower joint is rotated by π/2, the rotation of the blended matrix is π/4.

In the next sections, we will explain how are computed the skinning data of the anchorage points; skinning data are necessary of the skeleton driven deformation. We will as well describe how are generated the collision hulls.

2.2.2. Computation of the skinning data

The list of the influencing joints and the skinning data (the weight of each of the influencing joints) are computed using the pre-calculated cloth animation. The driving idea is to estimate these data so that the movement of the cloth patch computed by the skeleton driven deformation is as close as possible to that of pre-computed cloth animation. The process of defining the skinning data is composed of the following two steps:

- The process starts by first finding a number of candidate joints for each cloth patch. They are the joints in which the average displacement of the cloth patch is the smallest over the pre-computed animation sequence. Note that computing the displacement of the vertex in relation to a joint is equivalent to computing its position in the local coordinate system of that joint.
- The skinning data is then computed by linear regression for all the possible subsets of the list of joint candidates defined above.

The above algorithm involves the evaluation of the skinning data for a given set of joints. This is a minimization problem that can be summarized as follows: define the weights and dress position that best fit the sample data. A number of existing methods, such as the least square fitting by Mohr et al. [MOHR 03] could successfully be adopted. They successively calculated weights and dress positions in a loop. The result of the first minimization is taken by the second iteration, and so forth. In our case, however, the minimization problem is not linear due to the modified skeleton driven deformation we use (see Chapter III.2.2). Thus, we have adopted the Powell method (see [PRES 88]) to compute the skinning data. In comparison with other methods based on linear regression, it is significantly slower. However, this disadvantage is negligible as it concerns the pre-processing stage only.

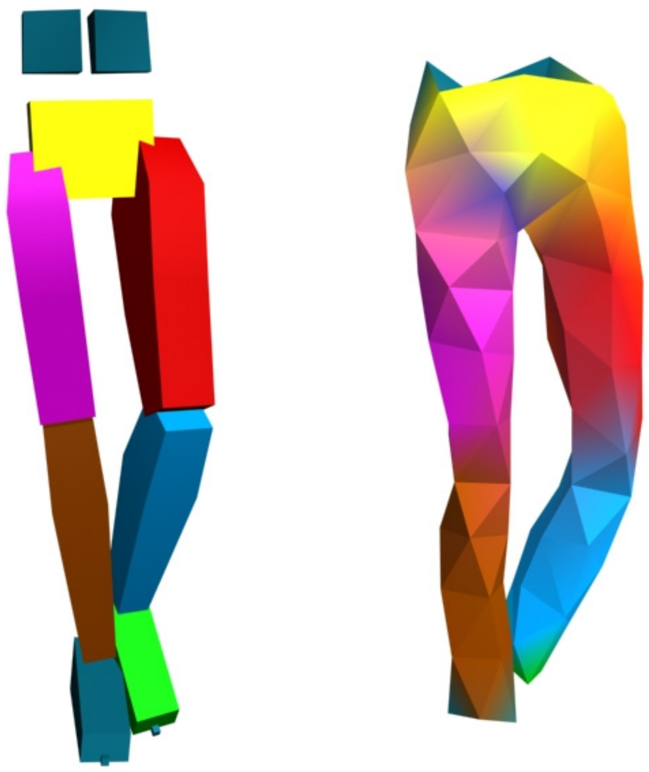


Figure V-5: The influence of the joints on the trouser deformation. The different colours are randomly assigned to the different joints.

The difference between the patch positions taken from the pre-computed animation and those estimated by the skin deformation can be considered as residual. Low residual values mean a good fitting and the movement of the patch can be fully described by the skeleton driven deformation. High residual values indicate that the movement of the patch is influenced by other constraints such as the gravity and/or the elasticity of the clothes. On Figure V-6, blue regions are tight garments (low residual values) whereas green and red regions are loose garments (high residual values).

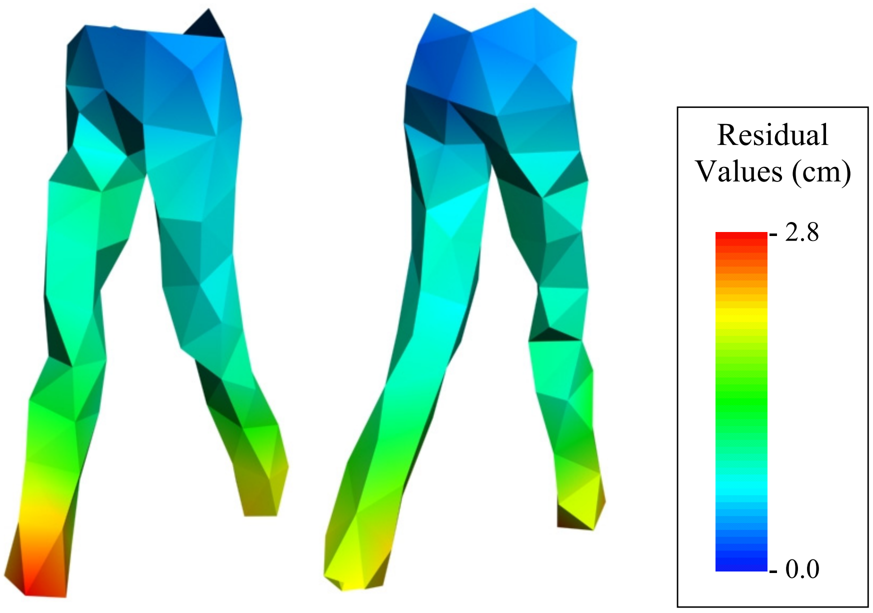


Figure V-6: Residuals of the fitting of the skinning data

2.2.3. Computation of the collision hulls

Our method for skeleton driven deformation computes the local transformation matrix of the vertex being deformed (see Chapter III.2.2). Therefore, it is possible to compute the movements of the cloth patch in the local coordinate system of its anchorage point for the pre-computed cloth animation. These local movements correspond to the second component, the one related to the dynamic behaviour of the clothes.

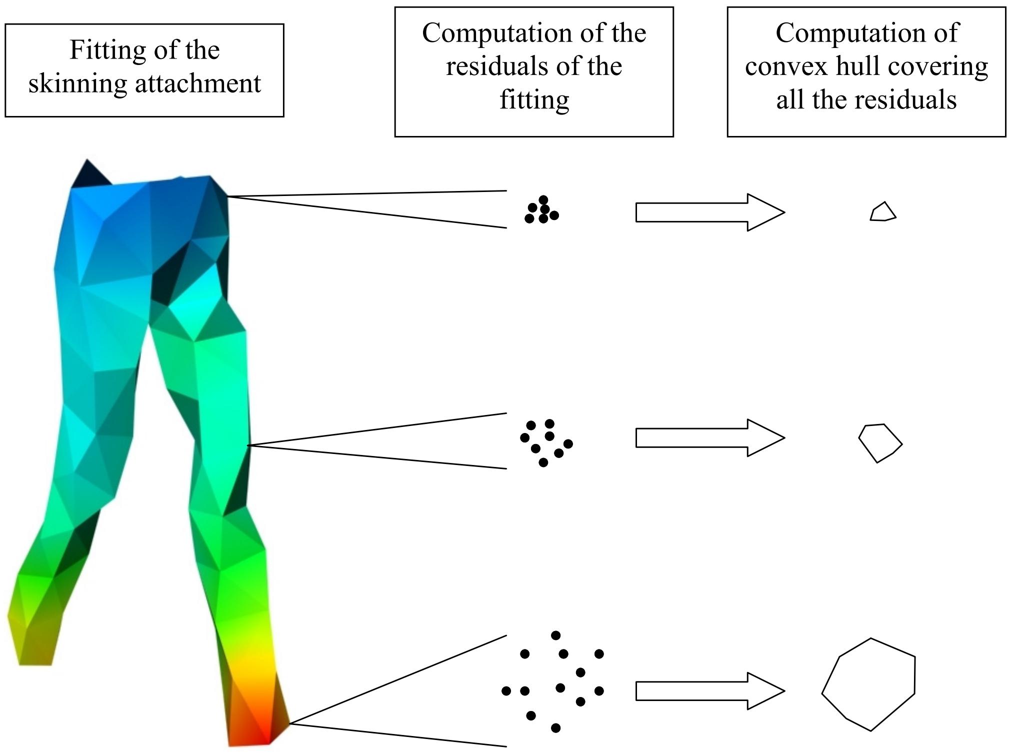

Fitting of the skinning attachment
Computation of the residuals of the fitting
Computation of convex hull covering all the residuals

Figure V-7: Computation of the collision hull for each cloth patch

We compute the convex hull covering these local movements. The convex hull of a set of points is the smallest convex set that contains the points. We use the method presented by Barber et al. [BARB 96] to accomplish this work. This hull covers the space of all possible positions that the cloth patch may occupy when animated by the mass-spring system. They are later used for the collision detection on the rough mesh. The Figure V-8 shows the convex hulls generated for a trouser model.

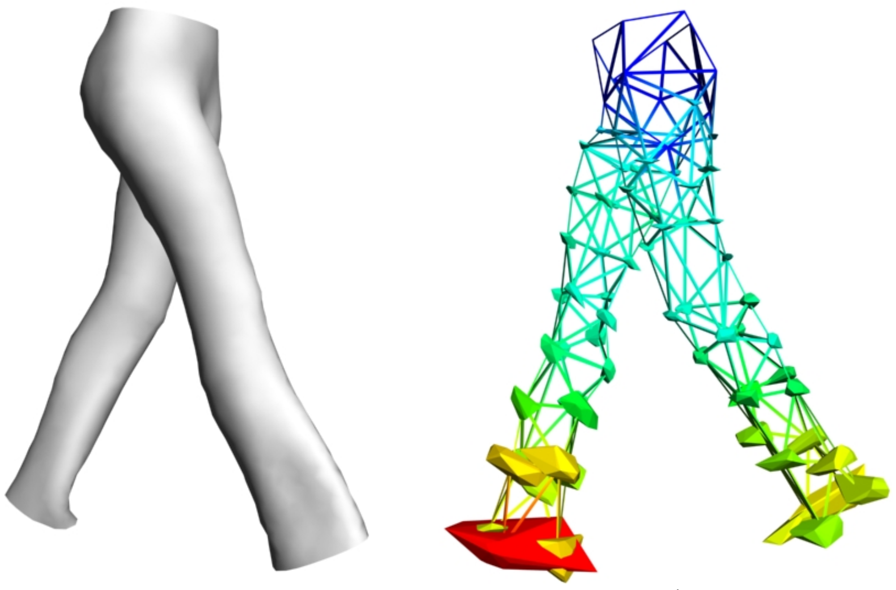


Figure V-8: Collision Hulls on the trouser (a: original trouser, b: the computed collision hulls)

2.2.4. Identification of the three regions (tight, loose, floating)

The main role of the convex hulls is to keep the cloth surface within a certain distance from the skeleton, as it would be attached to it. In case of floating garments such as skirt, cloth patches may collide with several joints. On the contrary, the local movements of some cloth patches (like tight shirts) are negligible so that these patches can be considered as being attached rigidly to the joint of the skeleton. The corresponding collision hull is so small that it can be approximated by a single point. A pre-processing stage is necessary to identify the three regions: the regions that potentially interact with several joints, those that are loosely attached to one joint and those that are rigidly attached to a joint. The identification of these regions is made thanks to the residual values of the skin attachment fitting (section 2.2.1).

The cloth regions with small residual values are attached rigidly to their anchorage points. By doing this, the computation load is significantly reduced because these cloth regions are not included in the simulation of the mass spring system.

High residual values indicate that the movements of the clothes are not dependent of a specific skeleton joint. Therefore, additional collision check is required to handle the interaction of the clothes with other skeleton joints. For these cloth regions, the closest distance to each joint of the skeleton hierarchy is calculated using the pre-computed cloth simulation sequence. This distance computation helps to define the list of the potentially colliding joints together with the collision distance.

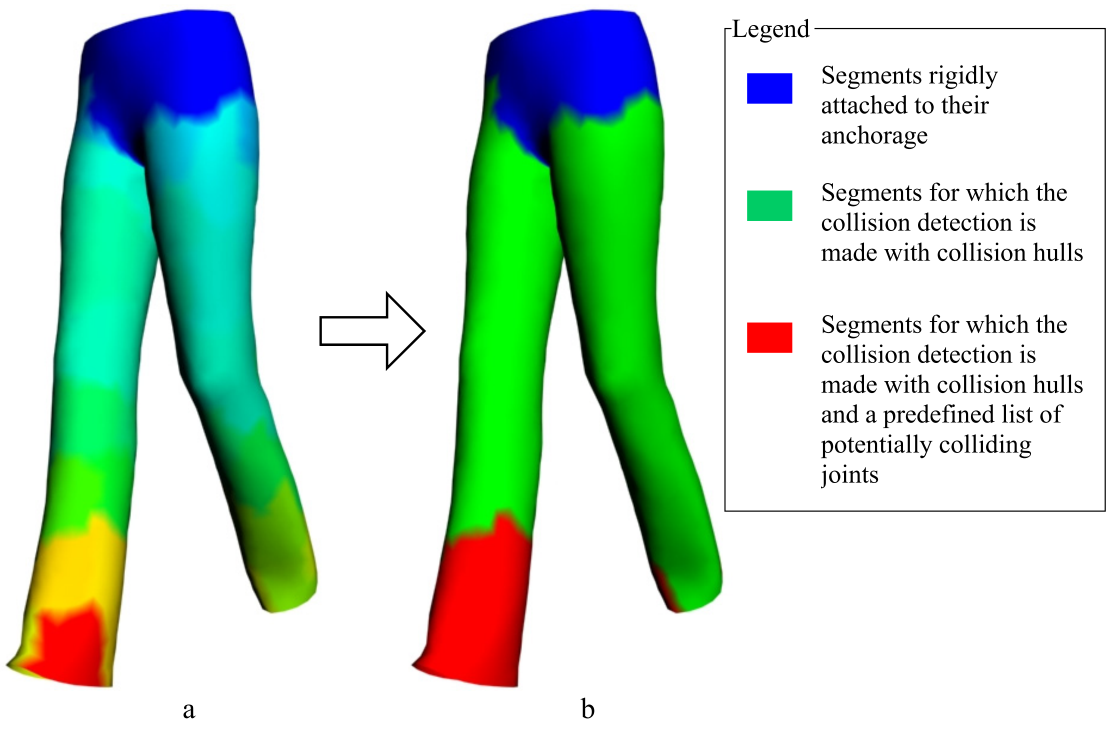

Legend
🟦 Segments rigidly attached to their anchorage
🟩 Segments for which the collision detection is made with collision hulls
🟥 Segments for which the collision detection is made with collision hulls and a predefined list of potentially colliding joints
a
b

Figure V-9: The three regions computed on the trouser shown on (b) based on the analysis of the residual values on (a)

2.2.5. Defining the mechanical properties of the rough mesh

Physically based deformations are commonly handled by a mass spring system, which is also used here to simulate the rough mesh. There are several parameters that need to be identified prior to the simulation. First of all, the edge length of the rough mesh must be computed. As most of clothes are not stretched by their own weight (i.e. the gravity does not affect the length of the clothes), we estimate the edge length by computing the average length of the pre-computed simulation. Other parameters such as bending coefficient and mass are the same as those used for the computation of the cloth animation sample.

2.3. Runtime deformation

In this section we describe what is really computed during the real-time simulation. Figure V-10 gives an overview of the computation process. The deformation of rough mesh is handled by a mass spring system and collision detection is made using the collision hulls described in Section 2.3.1.

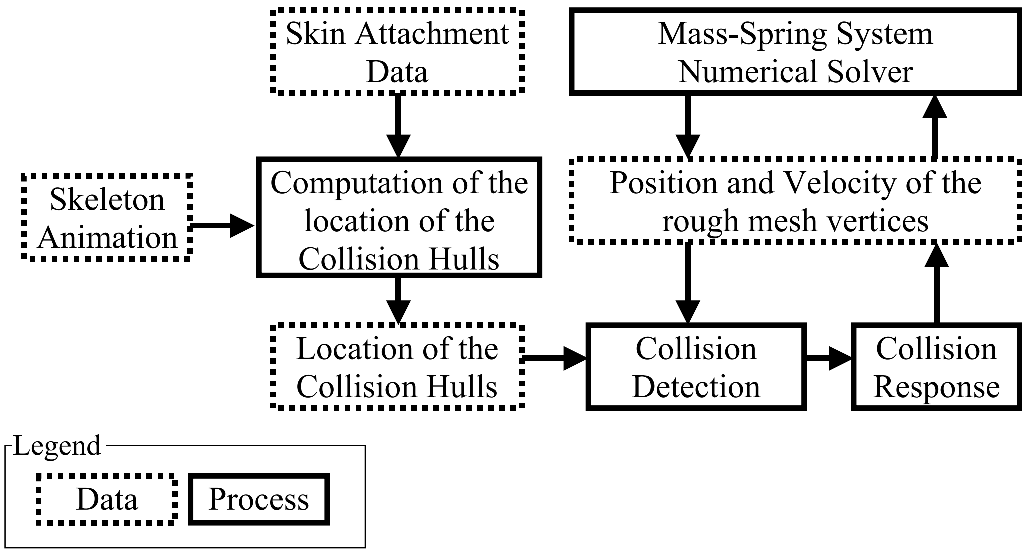

Skin Attachment Data
Mass-Spring System Numerical Solver
Skeleton Animation
Computation of the location of the Collision Hulls
Position and Velocity of the rough mesh vertices
Location of the Collision Hulls
Collision Detection
Collision Response
Legend
Data
Process

Figure V-10: Runtime deformation

The mass-spring system based on the backward Euler method (see Chapter II.1.2.2) is adopted. This method has been introduced in Chapter II.1.2.2 and also implemented in Chapter IV.5. The backward Euler method permits large time step while keeping stable cloth simulation (see Chapter II.1.2.2 for further details).

Collision detection can be efficiently handled by using the collision hulls that have been computed during the pre-processing phase. After every simulation step, the positions of the particles are corrected geometrically so that they remain inside their corresponding hulls. Thus, the collision detection returns to computing the inclusion the particle to the hull. The inclusion checking takes the following three stages:

- At first, the transformation matrix of the anchorage point is computed using the skeleton driven deformation method as defined in Chapter III.2.2. The position of a cloth particle is represented in the local coordinate system of its anchorage point.
- It then searches the vertex on the hull surface that is the closest to the cloth particle. The searching method is based on the Gilbert-Johnson-Keerthi algorithm [GILB 88]. A detail description of this algorithm is given in Section 2.3.1.
- Once the closest vertex on the convex hull has been found, the cloth vertex is checked on whether it is inside the hull or not. If it is outside, the position of the cloth vertex is corrected by projecting it onto the closest triangle, edge or vertex on the hull surface. The velocity and acceleration are adjusted as well using the same approach described in Chapter IV.5.3.

For cloth patches that belong to the floating cloth regions, additional collision checks need to be made by computing the distance between the patch and the list of potentially colliding skeleton joints. If the distance is smaller than a threshold, the position and the velocity of the patch are modified to move it away from the skeleton joint.

2.3.1. Computation of the distance between a vertex and convex polyhedron

The computation of the minimum distance between two convex polyhedra is a well-known problem in robotics for path planning and collision avoidance. Extensive research has been carried out in this area. Our case is slightly different that the usual problem encountered in robotics, as we need to compute the distance between a vertex and polyhedron and not between two polyhedra. Nevertheless, the same approach can be used in the both cases, for instance the Gilbert-Johnson-Keerthi algorithm (GJK). This method has been proven to be robust and fast [CAME 97], [GILB 88]. GJK is essentially a descent method for finding the point on the polyhedron surface closest to the origin. In the following explanation, we will use the above notation:

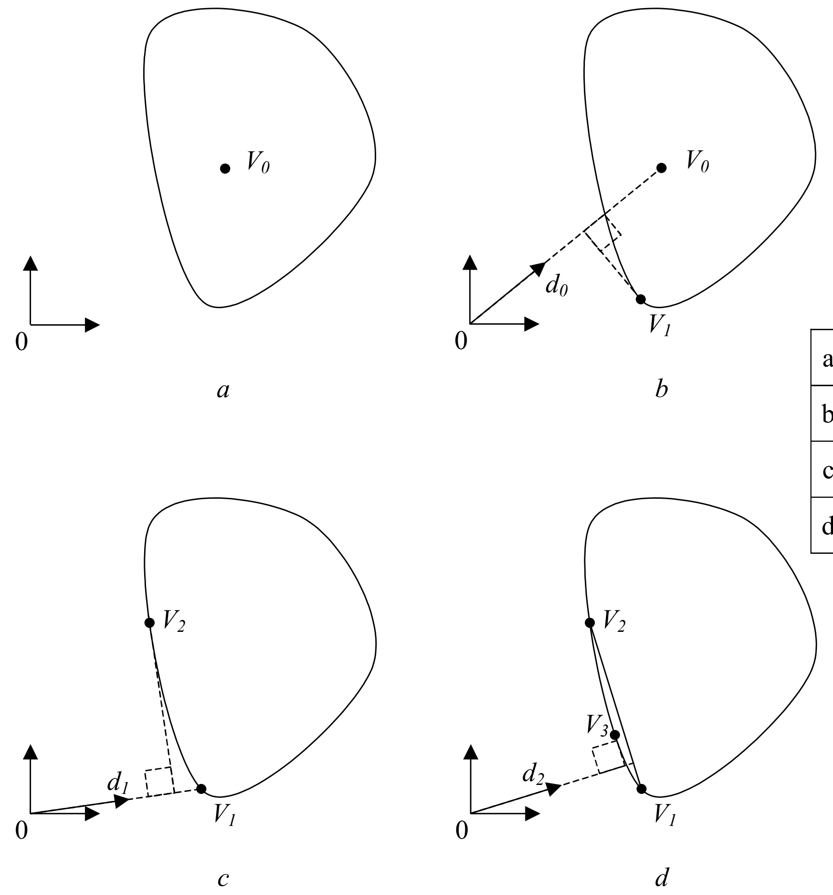
- d being the dimension of the space, we define C a set of vertices in the space such that $C \subset \mathbb{R}^d$. $v(C)$ denotes the nearest point to the origin: $v(C) \subset C$ and $\|v(C)\| = \min\{\|x\| \,|\, x \in C\}$.
- S_C is a support mapping function which maps $p \in C$ to $S_C(p) \in C$ and $p \cdot S_C(p) = \max\{p \cdot x | x \in C\}$. In other word, $S_C(p)$ finds the vertex x belonging to C which is the furthest to the origin along the line passing through p and the origin.
- The convex hull of a finite point set $X = \{x_1, \ldots, x_n\}$ is the set of convex combinations of points in X, denoted by $\mathrm{conv}(X) = \left\{ \sum_{i=1}^{n} \lambda_i x_i : \sum_{i=1}^{n} \lambda_i, \lambda_i \geq 0 \right\}$.
- A simplex is the convex hull of a set of affinely independent vertices. A set of points is called affinely independent if no point in the set is an affine combination of the other points. In the 3-dimensional space, a simplex can be an edge, triangle or tetrahedron.

The GJK algorithm makes use of two variables: W_k, which is the set of vertices that compose the simplex at the iteration k and v_k such that $v_k = v(W_k)$. The algorithm computes a sequence of simplices, each new generated simplex being closer to the origin than the simplex constructed at the previous step. At the initialisation, we take $W_k = \varnothing$ and v_k an arbitrary vertex in convex hull. At each iteration, the algorithm proceeds as follows:

- **Step 1:** define w_k such that $w_k = \mathrm{conv}(-v_k)$.

- **Step 2:** define $v_{k+1} = v(\text{conv}(\text{W}_\text{k} \cup \{w_k\}))$. This step is done using the projection of the hull along the line *d_k*.
- **Step 3:** W_{k+1} is defined as the smallest set $X \subseteq \text{W}_\text{k} \cup \{w_k\}$ such that v_{k+1} is included in $\text{conv}(X)$.

An example of the functioning of the GJK algorithm in 2 dimensions is shown on the figure below.



	Simplex	Direction
a	∅	
b	V_1	d_0
c	V_1, V_2	d_1
d	V_1, V_3	d_2

Figure V-11: A two dimensional example of GJK algorithm in action

The computation time can be greatly reduced by initialising the GJK algorithm with the same vertex as the closest one that was found at the previous frame. If the distance covered by the polyhedron during two consecutive frames remains small, the vertices found by the GJK algorithm on the two consecutive frames should be very close to each other.

2.4. Results on the simulation of the rough mesh

The use of a rough mesh significantly reduces the computation load. Any cloth can be efficiently simulated with a rough mesh of hundred vertices. However, the chosen approach for collision detection implies several limitations that are listed below:

- First of all, this method can only be applied for worn garments because the collision detection is based on skeleton driven deformation and the underlying skeleton. It follows that the garments have to be kept on the body for the whole simulation. For instance, dressing and undressing is not possible.
- The method requires a pre-computed animation of the clothes to be animated because the computation of the collision hulls is based on the analysis of an existing movement of the clothes. It is also required that the body animation that is used in the pre-computed cloth sequence is sufficiently varying. Experience has shown that a body animation that is composed of forward walk of 2 steps and turns on left and right is sufficient to be able to simulation most of body animations.
- The collision detection does not compute self-collisions. The purpose of the rough mesh is to simulate the global behaviour of the clothes. Self-collisions are partially handled in the second level of deformations.

3. Example-based approach for generating the cloth details

So far we have shown the first part of our simulation, that is, the coarse level simulation. We now describe the second part of the simulation, by which detailed cloth shape such as wrinkles or folds are depicted. Again, the main challenge here is obtaining the highest possible realism while maintaining acceptable computation load, in order to meet the real-time requirements.

As recognized in earlier works [HADA 99] [KANG 01], wrinkles are geometric in nature and can be efficiently animated with a geometric method, which is adopted here. Unlike previous methods, however, our wrinkling function is not hand-drawn, nor geometrically approximated, but rather trained from on the analysis of the pre-simulated sequence.

Geometric methods may not be suitable for simulating the dynamic behaviour of the clothes. However, the simulation of cloth dynamics is primarily handled at the coarse representation level, as described in the previous section, and it is deliberately omitted at the fine mesh level for the sake of lighter computation load.

The driving idea is to use examples from the pre-simulated sequences in order to generate the detail shape as accurately as possible. The shape of a patch is defined as a function of the vertex position and its neighbours on the rough mesh. The function is constructed by using an interpolation method that makes use of samples taken from the pre-computed cloth simulation. Several techniques exist for shape interpolation using examples, such as Radial Basis Function or parametric interpolation [GOME 98]. We have chosen to adopt the linear interpolation, as since it provides satisfactory results at an inexpensive computation cost compared to, for instance, RBF.

The linear interpolation is defined as a weighted summation of the position of the associated vertex and its neighbours. We will explain in the next section how the weights are determined from the pre-simulated sequences.

3.1. Pre-processing

Three interpolation functions are defined for each vertex in the patch, one for each dimension *X*, *Y* and *Z*. For every vertex x in a patch, the interpolator function takes the associated mass point in the coarse mesh, and its neighbours as input. In this work, we choose to represent the wrinkle displacement in the local coordinate system used for SDD. This makes our wrinkle parameterization invariant of all joints of higher hierarchy than the currently influencing joint.

To calculate the position of x from the input, the wrinkle interpolators interpolate the positions of the coarse mesh points, weighted by coefficients determined the regression model of the following forms:

$$F_x(x_{cp}, x_{cpn1}, x_{cpn2}, \ldots) = \alpha + \alpha_{cp}.x_{cp} + \alpha_{cpn1}.x_{cpn1} + \alpha_{cpn2}.x_{cpn2} + \ldots$$

$$F_y(y_{cp}, y_{cpn1}, y_{cpn2}, \ldots) = \beta + \beta_{cp}.y_{cp} + \beta_{cpn1}.y_{cpn1} + \beta_{cpn2}.y_{cpn2} + \ldots$$

$$F_z(z_{cp}, z_{cpn1}, z_{cpn2}, \ldots) = \gamma + \gamma_{cp}.z_{cp} + \gamma_{cpn1}.z_{cpn1} + \gamma_{cpn2}.z_{cpn2} + \ldots$$

where x_{cp} and $x_{cp1}, x_{cp2}, \ldots$ are respectively the position of the control point x and its neighbours; they are all expressed in the SDD coordinate system of x. The values β , α , γ , β_{cp} , α_{cp} , γ_{cp} , β_{cpn1} , α_{cpn1} , γ_{cpn1} … are the interpolation coefficients. They are defined by multi-linear regression on a set of pairs (positions of coarse mesh vertices, fine mesh vertices) extracted from the pre-simulated cloth sequence.

Figure V-12 illustrates the displacement of *x* (a vertex in the patch) in response to the movements of the control point x_{cp} and its two neighbours x_{cpn1} and x_{cpn2}. The aim is to express the position of *x* as a function of the positions of x_{cp} , x_{cpn1} and x_{cpn2} for the three dimensions separately.

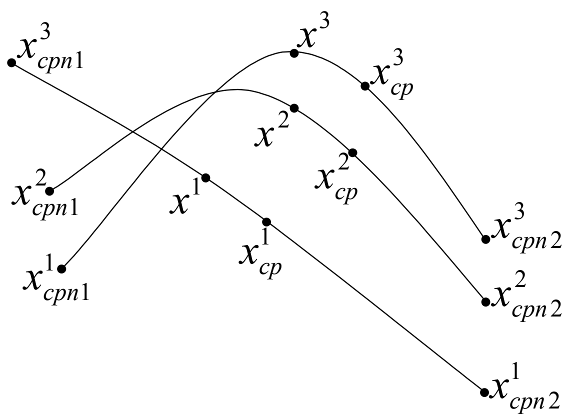


Figure V-12: Positions of the vertex *x* in respect to the positions of the control point x_{cp} and its neighbours x_{cpn1} and x_{cpn2}.

The coefficients of the interpolation function for one vertex and one dimension are calculated as the following. Given $X = \begin{pmatrix} 1 & x^1_{cp} & x^1_{cpn1} & \cdots & x^1_{cpni} \\ 1 & x^2_{cp} & x^2_{cpn2} & \cdots & x^2_{cpni} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x^j_{cp} & x^j_{cpn2} & \cdots & x^j_{cpni} \end{pmatrix}$ with ($x^1_{cp} \ldots x^j_{cp}$) the positions of the main control point and ($x^1_{cpni} \ldots x^j_{cpni}$) the positions of the *i* neighbouring control points in the *j* pre-computed cloth simulation frames, $Y = \begin{pmatrix} x^1 \\ x^2 \\ \vdots \\ x^j \end{pmatrix}$ the positions of the vertex, $A = \begin{pmatrix} \alpha \\ \alpha_{cp} \\ \alpha_{cpn1} \\ \vdots \\ \alpha_{cpni} \end{pmatrix}$ the coefficients, the expression of the linear regression is $Y = XA + \varepsilon$. ε is a vector of random errors. For further details on linear regression, the book edited by Montgomery et al. [MONT 02] may be consulted.

3.2. Runtime deformation

Figure V-13 on the below illustrates the deformation of a patch according to the displacement of a control point. The computation load of such an approach is light, as the interpolation function requires few multiplications and additions.

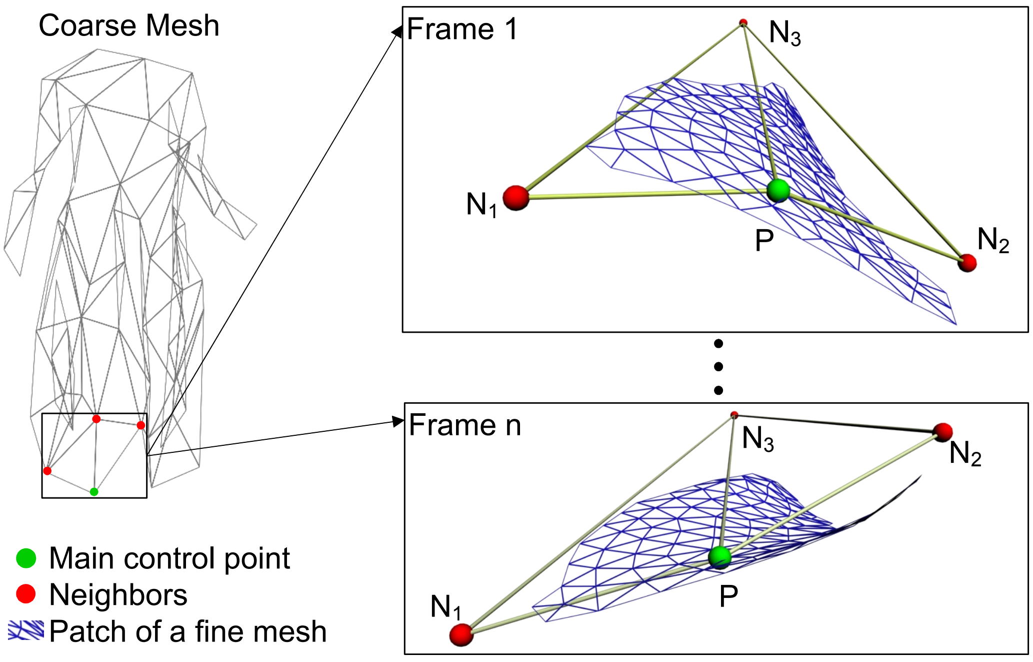


Figure V-13: Shape of the patch with respect to the positions of the control point P and its neighbours N1, N2, and N3.

Despite its simplicity, linear interpolation works fairly well provided a sufficient number of pre-simulated frames for the multi-linear regression. A condition of a good working interpolator is that the input (i.e. position of the coarse mesh vertices) should be within the range of the pre-simulated data. In other words, the wrinkle interpolator can only work for the input range for which it has been trained. This condition is maintained thanks to the collision detection (Section 2.2.3). This also keeps the smoothness of the boundaries between patches. Figure V-14 illustrates the deformation of the wrinkles.

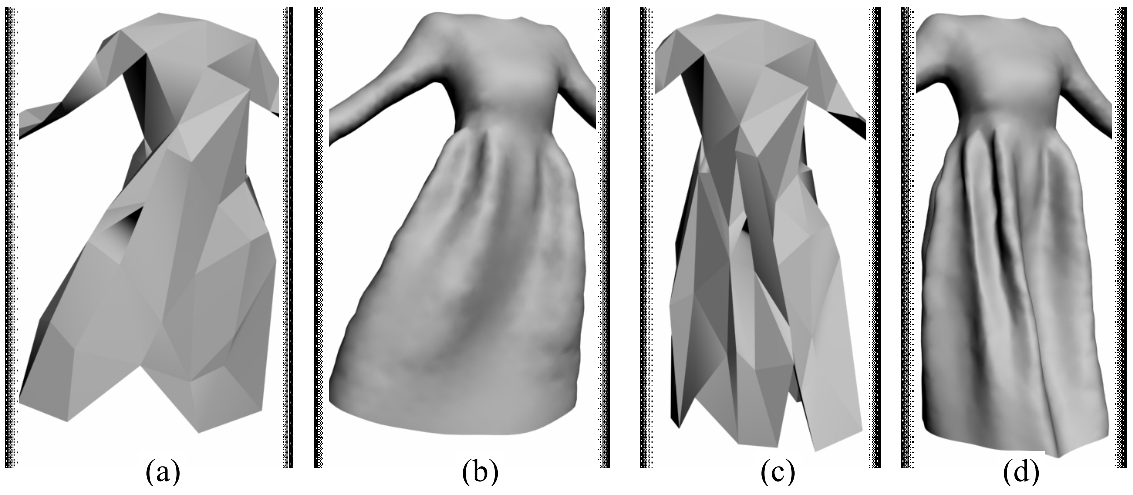


Figure V-14: The wrinkling interpolator in action: wrinkles in (b) and (d) are generated geometrically with (a) and (c) as input.

4. Results and discussion

We have measured and validated the proposed real-time cloth simulation method along three criteria: the variety of clothes to be simulated, the computation speed and the range of body motion in the pre-simulated cloth sequence. Pre-simulated sequences obtained by the cloth simulator of Volino et al [VOLK 00] were used in our pre-processing. Figure V-15, Figure V-16, and Figure V-17 show the pre-processing stages and simulation results of three different cloth models.

4.1. Variety of clothes

We used our framework to different types of clothes, as shown in Figure V-15 to 18.

- The "evening" dress (Figure V-15) is chosen to demonstrate our wrinkle interpolator on large garment regions.
- The "evening" dress (Figure V-16) is chosen to demonstrate our wrinkle interpolator on large garment regions.
- The "cocktail" dress (Figure V-17) is a relatively complex model; the bottom is composed of two layers of tissues and has folds made of large number of vertices, inducing many self collisions.
- The "Jeans" outfit (Figure V-18) is a good example of a model where the SDD based geometric approximation can reduce the number of mass points substantially by simulating only a few regions that contribute significantly to the dynamic behaviour.

Our simulator behaves fairly well on a wide variety of clothes, including those with highly stiff mechanical properties. Figure 1 and Figure 18 show the pre-processing and run-time simulation results for a long dress, a skirt and underwear. Moreover, performance will increase due to the fact that the smallest number of triangles will be processed for the real-time rendering.

However, the method may introduce flaws in simulation for some tight clothes, due to the approximate handling of collision detection. For some body movements, the skin surface may slightly intersect the cloth surface. Similarly, the same problem may arise for self-collisions on clothes. This effect is particularly visible on the "Cocktail" dress that is made of two layers of tissues (see videos). The deletion of the skin triangles covered by the garment surface can partially correct this drawback.

Note that the cloth simulation is also restricted to clothes worn on bodies. While offering high computation speed, the cloth simulator cannot handle some cloth movements such as those appearing during dressing or undressing. More generally, the clothes are unable to interact with objects other than those that have been taken into consideration during the pre-processing phase. The list of objects that can potentially interact with clothes and the way these objects interact are defined at the pre-processing stage and cannot be changed during the real-time simulation. Finding a method to update the list of possible interacting objects automatically could be a subject for future research.

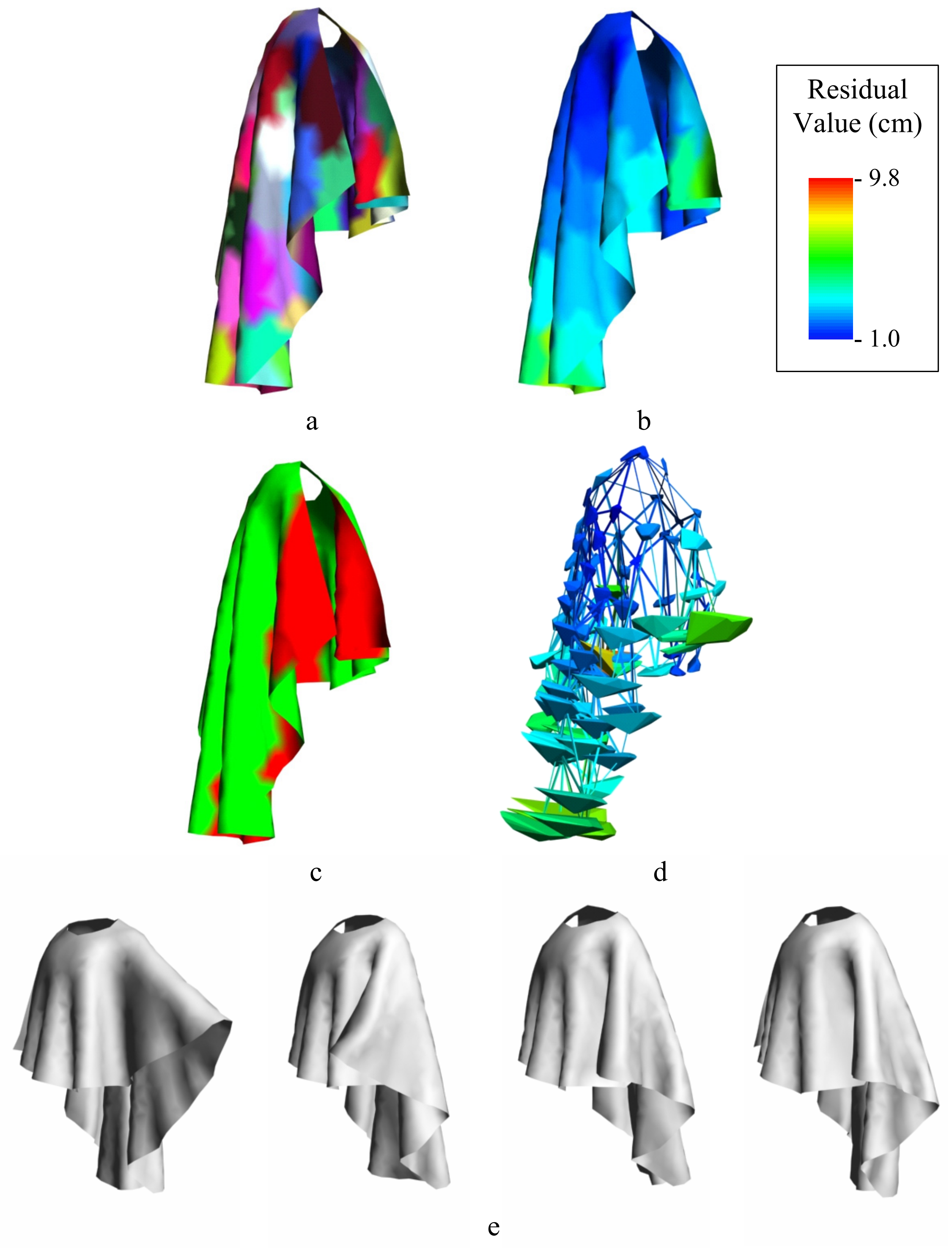


Figure V-15: Simulation of the cloak: the segmentation (a), analysis of the vertex movements (b), identification of the three regions (tight, loose, floating) (c), the resulting rough mesh with the collision hulls (d), samples of the real-time animation (e)

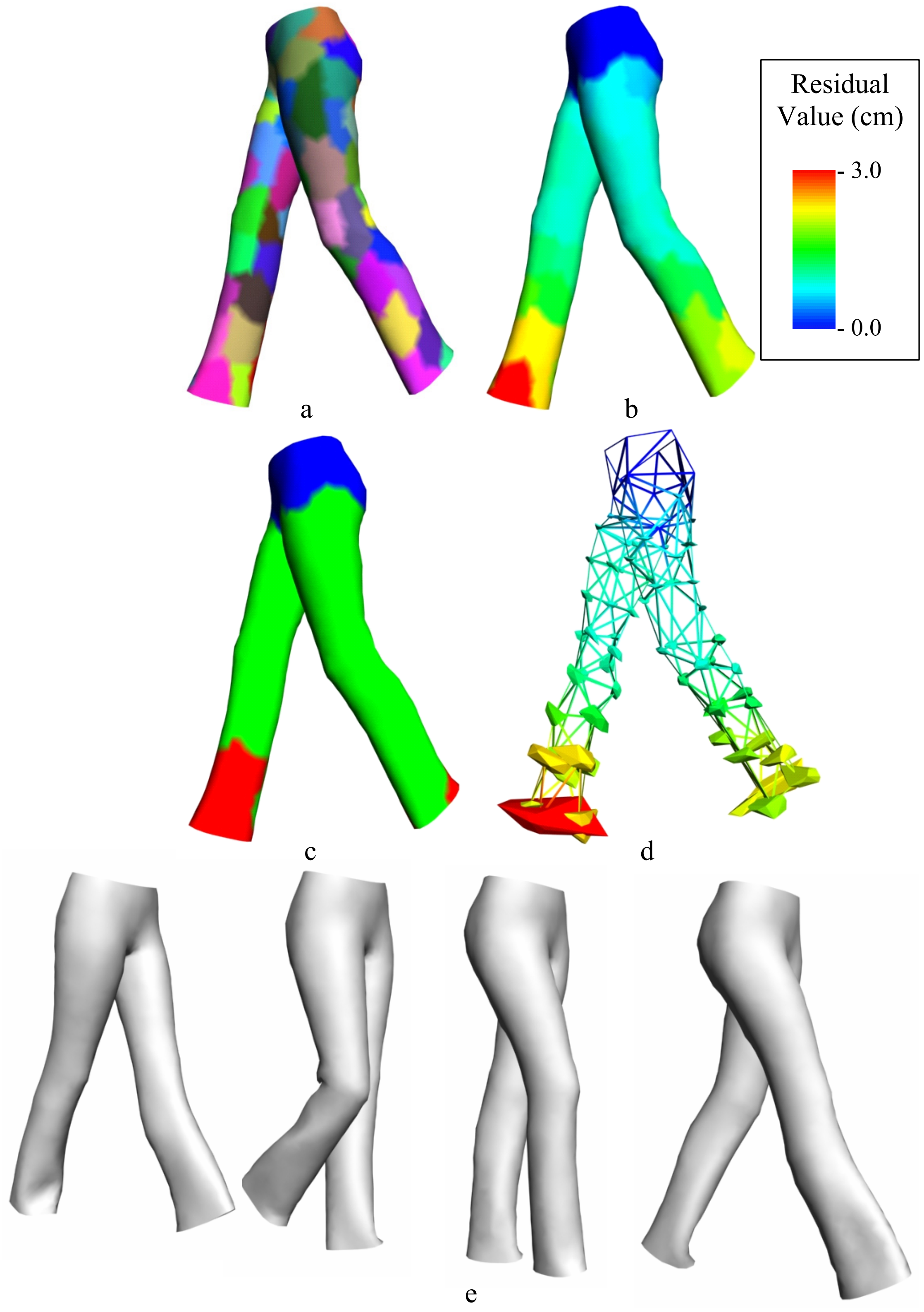


Figure V-16: Simulation of the trouser: the segmentation (a), analysis of the vertex movements (b), identification of the three regions (tight, loose, floating) (c), the resulting rough mesh with the collision hulls (d), samples of the real-time animation (e)

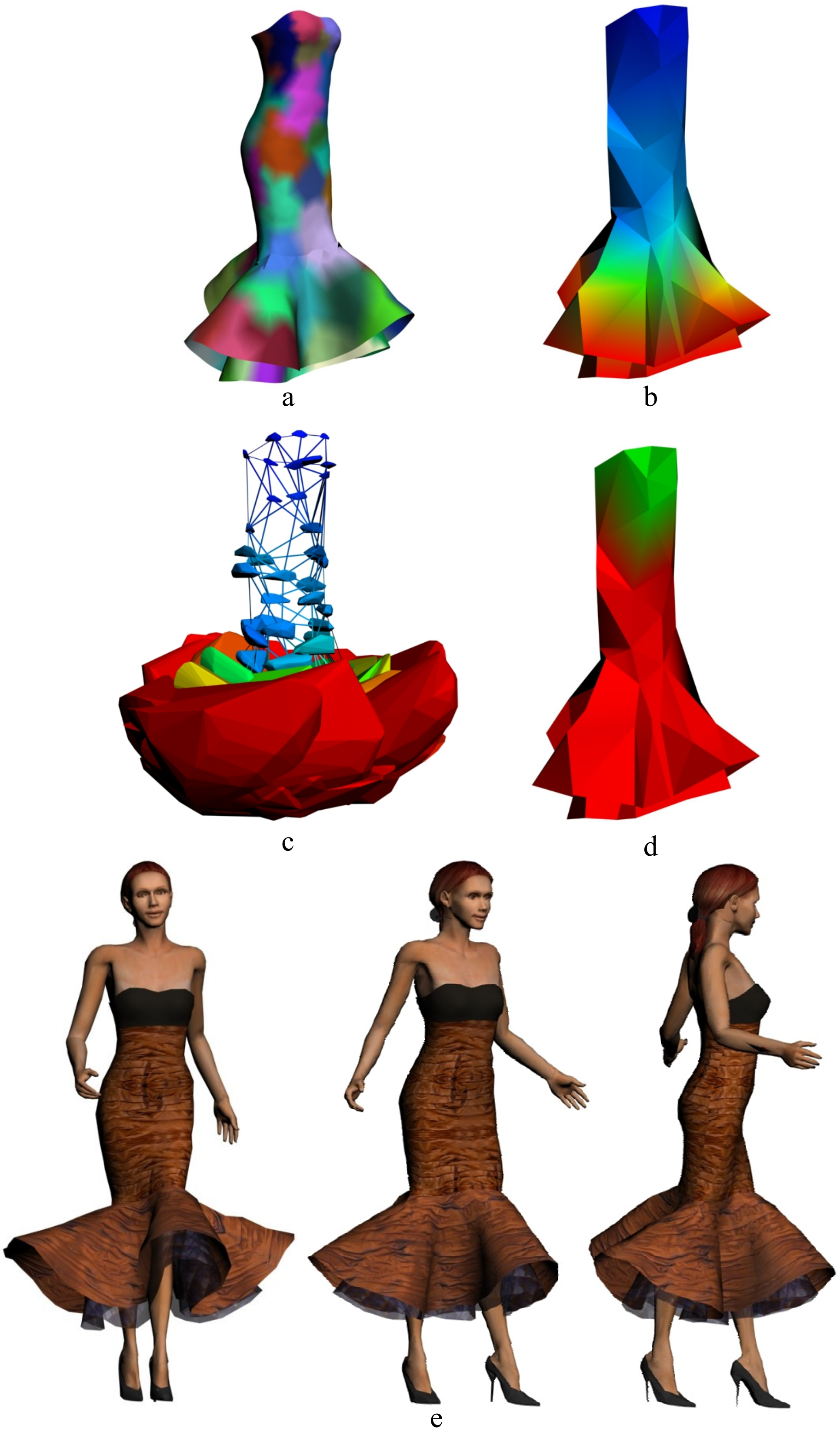


Figure V-17: Simulation of the cocktail dress: the segmentation (a), analysis of the vertex movements (b), identification of the three regions (tight, loose, floating) (c), the resulting rough mesh with the collision hulls (d), samples of the real-time animation (e).

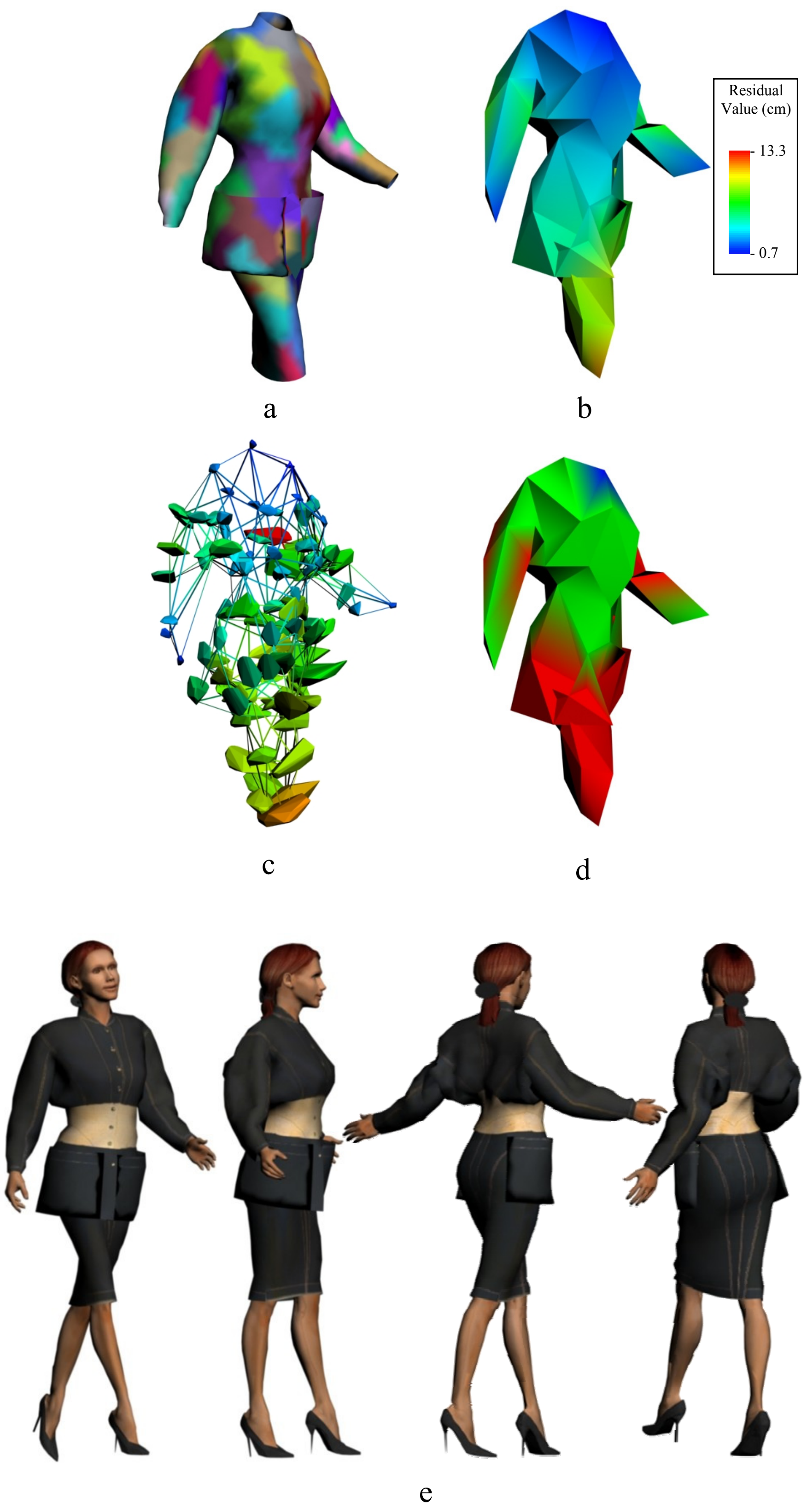


Figure V-18: Simulation of the jean outfit: the segmentation (a), analysis of the vertex movements (b), identification of the three regions (tight, loose, floating) (c), the resulting rough mesh with the collision hulls (d), samples of the real-time animation (e).

4.2. Computation speed

Table V-1 summarizes the performance of the cloth simulator running on a 1 GHz Windows PC. The pre-processing of all the cloth models took less than 10 minutes. All examples run in real-time at approximately 25 to 50 frames per second (fps), with the coarse mesh deformation process taking about 75% of the total CPU time. As expected, the duration of the pre-simulated sequence is not a factor of the runtime computation speed. In practice, the performance lowers down at a low rate as the complexity of the collision hulls increases, which tends to be governed by the number of pre-simulated frames.

	Cloak	Trouser	Cocktail Dress	Jeans Outfit
Number of faces	2602	4180	1331	2131
Pre-processing time (min.)	1.33	2.5	4.0	3.5
Number of vertices on the rough mesh	94	78	82	97
Performance rough mesh (fps)	102	80	74	63
Performance fine mesh (fps)	250	169	588	322
Overall Performance (fps)	72	54	63	51

Table V-1: Computation speed

4.3. Range of motion

As expected, the quality of the simulation depends on the number and variety of examples – the pre-simulated sequence in our case. To show that the simulator faithfully recreates the cloth movement used for training, we compared the real-time simulation with the pre-simulated one in the first video. The character walks at a normal pace without any fast movements.

In the second video, different body movements from those of the training were supplied as input to our real-time simulator and the results are compared with the ones generated with a high quality simulator.

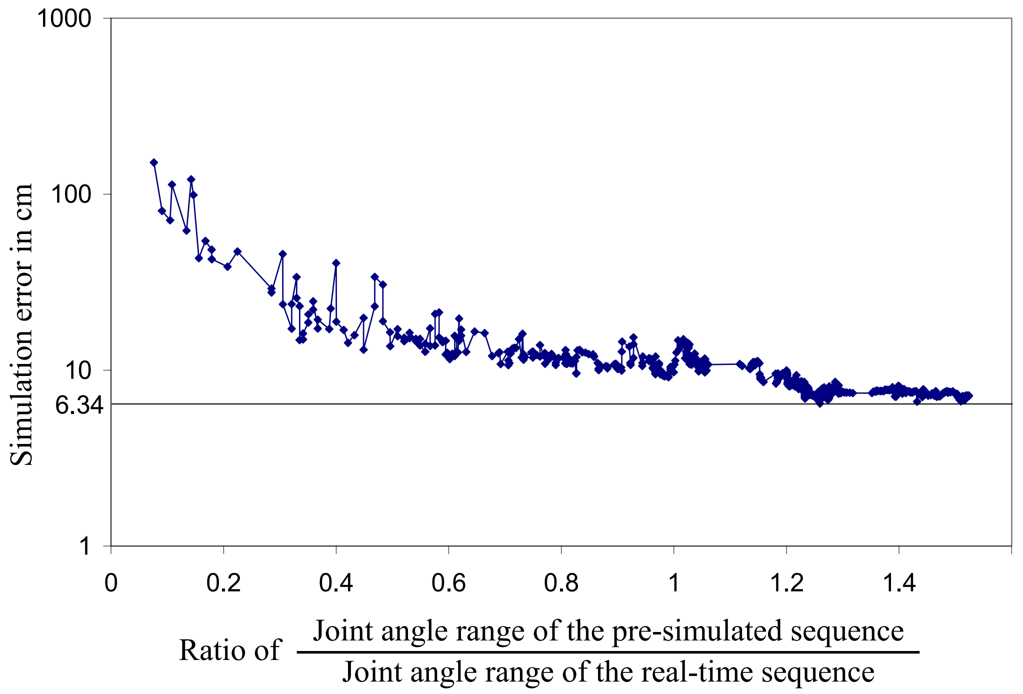


Figure V-19: Estimation of the error when reducing the range of body motion in the pre-simulated sequence

To measure the simulation quality, we compared our simulation results with the pre-simulated sequence, using a deformation metric. It measures the still shape and movement by the sum of edge length difference and the mass velocity difference over the cloth mesh. The Figure V-19 shows the importance of the variability of the body motion in the pre-simulated sequence. The best quality is achieved when the range of the body motion in the pre-simulated sequence is approximately 30 % larger than the one used in the real-time simulation.

Our simulator works well for interpolation (i.e. joint angles within the range of those of the pre-simulated sequence) but often fails for extrapolation. The main reason for this limitation is collision detection, which does not allow the clothes to have different locations on the body from those calculated in the pre-simulated sequence.

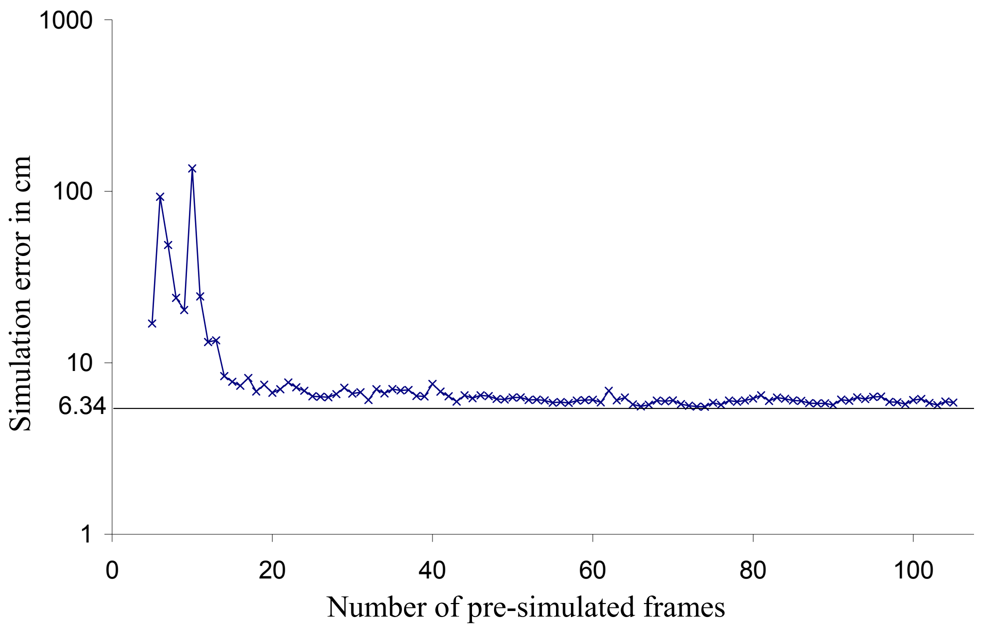


Figure V-20: Estimation of the error when reducing the number of pre-simulated frames.

Figure V-20 shows the effect of using motion of different durations (expressed in number of frames) with same joint angle ranges. With less than 70 pre-simulated frames, the real-time simulation loses its quality.

5. Conclusion

In this chapter, we have shown that the example-based approach can be successfully adapted to the case of physically based deformation models. Unlike geometric deformation models, physically based models depend on a high number of parameters such as the position and the velocity of the vertices of the model, making the real-time simulation difficult. The problem has been solved by using a hybrid approach where the dynamic behaviour of the garments is handled with a simplified mass-spring system and the mesh details are generated with an example-based approach. Our framework has been successfully used to produce visually pleasing motion of a wide range of clothes – It offers high computation speed while producing realistic deformations, since the bottleneck of the simulation, the collision detection and the computation of the mass-spring system has been significantly reduced thanks to the pre-processing of the pre-calculated sequence of the clothes.

One of the main advantages of the proposed cloth deformation method is its ability to handle cloth deformation as well as skin deformation. It can as well reproduce the mechanical properties of the fabrics that are close to original.

Another advantage is that the pre-processing stage does not require any user interaction. Both the mass-spring system and collision detection have been rewritten to take advantage of the pre-simulated sequence of the clothes to be animated. Unlike the method presented in

Chapter IV, the pre-processing is fully automatic, making the creation of real-time clothes as simple as one mouse click. Consequently, our cloth simulator is able to construct a model for real-time animation without user intervention and can deal with different types of clothes from tight to floating with low computation consumption.

On the other hand, this approach has some drawbacks inherent to example-based approaches and the approximations made for reducing the computation time.

- First, it requires a set of examples (pre-computed cloth simulation sequence in the case of this method). It has been found that using too few examples (less than 200 frames) makes the cloth simulation performing unrealistically. This limitation finds its origin in the statistical analysis involved in the pre-processing. With two few samples, the linear regression is too sensitive to the noise.
- Another limitation is its inability to extrapolate the cloth shape. This case arises when the skeleton animation used for real-time performance is completely different that the one used in the pre-processing stage.
- Its third limitation is the approximated collision detection. For some of the frames, it happens that the skin surface slightly crosses the cloth surface. This limitation is not a serious issue because the user has the possibility to delete the skin triangles hidden by the clothes (see Appendix B.2.3). The slight penetration of the skin into the clothes will not be visible anymore. Moreover, it will increase the performance thanks to the smallest number of triangles to be processed for the real-time rendering. The same problem was found in the hybrid approach (see Chapter IV) and treated in the same way.
- Finally, it is not suitable for the simulation of clothes other than those worn on characters. The use of the skeleton makes the cloth simulation faster but limits its versatility.

There are many interesting avenues for future work. First, the approach could be extended to simulating other physics-based models such as hair and fluid. We also believe that the work on collision hulls is promising. The current mesh model of collision hulls could be replaced by implicit surfaces or voxel maps. Therefore, for a cloth vertex, it could be possible to compute several collisions hulls in relation to different objects in the scene and to compute their intersection for real-time collision detection. By doing so, it may be possible to process collisions on a higher number of objects while maintaining low computation cost. We also believe that the precision of the collision detection could be improved by replacing the convex shape by a surface to follows more closely the trajectories of the vertices.

Chapter VI. Case study: The Virtual Try-On

The two previous chapters have described real-time cloth deformation techniques. In this chapter, we present one of the important applications of these techniques, in the frame of clothing and VR related projects — the Virtual Try-On. The Virtual Try-On is an application for an online clothing store, which provides easy access and manipulation of garment related items to facilitate garment size selection, pre-visualization, and sales. Because an online clothing store is a Web application running over the Internet, it not only requires real-time cloth simulation techniques but also needs specific research and development for the fine adaptation of clothes to the body sizes and the architecture design of the Web application.

Our system supports a number of efficient and interactive operations, such as automatic adjustment of the 3D mannequin according to the shopper's body sizes [SEOH 03], the selection of different garment items in the catalogue that can be instantly resized to fit well to the 3D mannequin, and real-time simulation of the garment as the mannequin moves under control of the user. A significant step in this process is the application of 3D graphics technology to help create and simulate the virtual store and we discuss various relevant research problems: the creation and fitting of garments, simulation of body/garment movement and the overall architecture of the application.

The use of the real-time cloth technologies in the framework of the targeted system has raised several issues, which are discussed in this chapter. The first issue concerns about how the proposed cloth simulation techniques can be integrated with other components of the system. The Virtual Try-On is based on several technologies such as body sizing, garment fitting, skin and cloth deformation, body motion control, etc… One of the main requirements of the Virtual Try-On is its ability to run over the Internet through an available web browser. Its architecture, which is based the client/server scheme should be carefully designed to offer the interactivity to the user while keeping the amount of required data transfer over Internet as low as possible.

The second issue is the benefits of using real-time clothes. The first advantage of these technologies is that it enables the user interactivity, meaning that the user can control the avatar movements with real-time response of clothes. Secondly, real-time cloth methods can advantageously replace pre-recorded cloth simulation. Clothes are highly deformable objects and the amount of data corresponding to a sequence of cloth animation is huge; the animation of every vertex of the cloth mesh should be sent from the server to the client. The use of real-time cloth simulation does not make necessary anymore the transfer of these data. The skeleton animation of the avatar together with the pre-processing data of clothes is sufficient to play a cloth simulation sequence on the user's computer.

Finally, the Virtual Try-On application gives an example of the application of real-time cloth technologies for the everyday life. Other uses include video games, virtual and augmented reality, etc…

This chapter is organized around nine sections. The first section gives an overview of the application, as well as the system architecture. The description of the body, garment and motion database is given in the three next sections. Three other sections are dedicated to the description of the client and its two modules for garment sizing and simulation. Finally, the results and the conclusion are discussed in the last two sections.

1. System overview

1.1. Background and enabling technologies

The most common problems of selling clothes online include poor fit, bad drape, or unpleasant feel while wearing the item, or surprise as to the colour of the garment (Figure VI-1). Customer dissatisfaction in any of these cases drives returns and a costly occurrence for e-tailors. In consequence, high product return rates persist, and most consumers are still either hesitant to purchase garments online or are unsatisfied with their online shopping experience [BECK 03]. Many online clothing stores have been developed to offer solutions to the garment sales through Internet, with an aim of allowing customers to "virtually" try-on clothes prior to purchases.

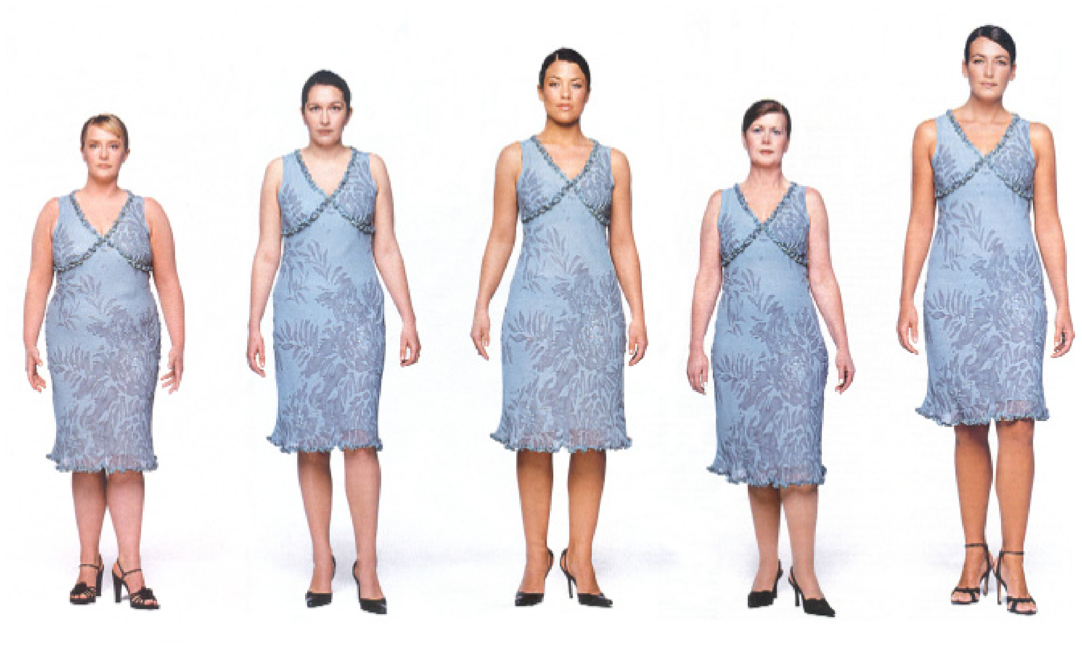


Figure VI-1: The same cloth worn by people of different measurements [SUTI 02]

Several works have been published, addressing different aspects of developing such online clothing stores. Ebert et al. [EBER 02] have proposed an approach for the automatic resizing of the clothes in response to the modification of body measurements. Groß et al. [GROB 02] have developed a Java application, which automatically positions the patterns around the human body and computes the fitting of the garments to the body using a mass spring system. Hinderer et al. [HIND 01] have stated that the Virtual Try-On can be greatly improved by the use of a virtual consultant giving advices to the customer.

A number of initiatives have arisen in the industry as well across the world [NORD 03] [MACY 03] [WELC 03], revolving around the concepts of Made-to-Measure manufacturing and shopping via the Internet. These initiatives are fuelled by the current Web technologies available, providing an exciting and aesthetically pleasing interface to the general public.

However, until now such Web applications have supported only basic functions, such as viewing apparel items in 2D or 3D, combining different items together, mix and match of colour and texture, sometimes with a mannequin adjusted to the shopper's proportions.

An online clothing store designed as a Web application requires flexible manipulation, fast transmission and efficient storage of the display content. In particular, dealing with a relatively huge database of garments and the simulation of complex graphical objects, such as skin and the cloth, the most critical limitation is perhaps the real-time performance constraint. When the simulation of garment movement is performed using physically based model, due to the impossibility to achieve the real-time performance, simulation results are pre-recorded in order to display the animated garment at an interactive rate [PROT 02].

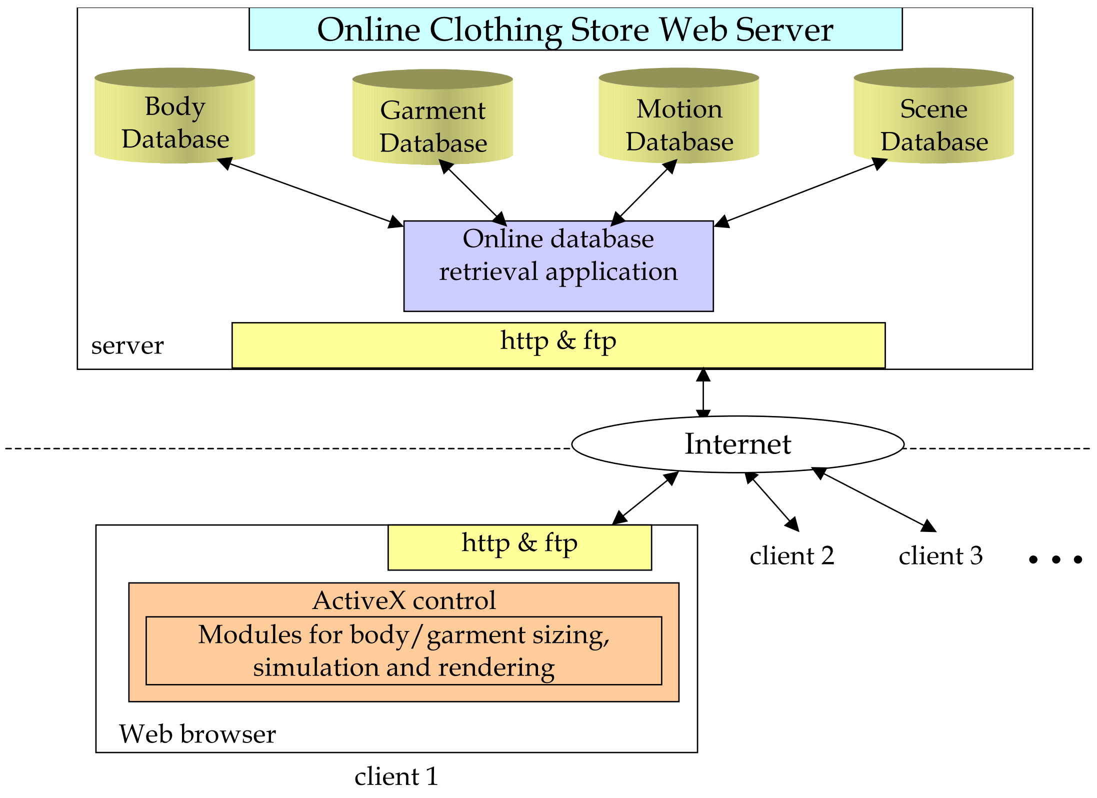


Figure VI-2: Web application architecture of online clothing store

Although it will simplify the online computation, it requires however to transfer for each frame the position data of each vertex constituting the visual representation of a garment, increasing the response time of the client application. In addition, expense to simulate each 3D garment item in the database is not negligible. Typically, it takes about four to twelve hours to simulate the garment movement of a one-minute sequence depending on the complexity of the 3D geometry. In our application, we choose a better alternative: the garment simulation is calculated on the fly while keeping the response time interactive.

Human body modelling and simulation is another key technology enabling the support of automated garment sizing and size selection. Since the advent of 3D-image-capture technology, there has been a great deal of interest in the application of this technology to the measurement of the human body. In the market, there are now also available some systems that are optimized either for extracting accurate measurements from parts of the body, or for realistic visualization for use in various fields including e-commerce applications. Cyberware Inc. [CYBE 03]'s DigiSize^{TM}, for instance, was developed in a joint government project to improve and automate fitting and issuing of military clothing. They are aiming at complete and state-of-the-art solutions to age-old tape measurements and trial-and-error fitting problems.

Despite their recent efforts devoted to the use of 3D body scanners, limiting factor lies in the inability to electronically and automatically integrates body scan data to application software. Such limitations led us to consider another body modelling scheme. One important feature of the online clothing store is its ability to build a 3D mannequin according to user input measurements on various parts of the body. We perform the online construction of the 3D mannequin that satisfies the given measurements, avoiding the complication of going through the scanner and/or the time consuming process of downloading large data models. Despite the seemingly difficulties in building the whole body geometry from the limited amount of information within the real-time constraint, we show how it is made feasible through our robust modelling technique for practical applications. Moreover, with such approach, the measurements and the body motion can be modified interactively.

Figure VI-2 shows the system architecture. Our approach provides a minimal response time to the user, since a major part of the content to be manipulated is generated on the client side rather on the server. Computing the body and cloth animations on the server and sending this data through Internet would generate too much traffic and would reduce the response time of the application. Thus, our solution is to move body/garment sizing as well as cloth/skin animation to the client side avoiding the download of large pre-calculated models.

1.2. The server

The online clothing store Web server consists of several databases and an online database retrieval application module for retrieving data.

- **Body database:** This database contains two 3D mannequins for each gender, which we refer as *generic* models, plus certain statistical information that is collected from existing models through 3D shape capture technologies. This is essentially the information necessary to derive new body geometries from measurement inputs by the body/garment sizing module in the client side.
- **Garment database**: A set of 3D garment models has been created for the generic models and categorized. They are available for the user to choose from the garment catalogue pages. Ideally, the online clothing shop proposes a great number of different garments. Moreover, it makes easy to update the database to keep the coherence between the clothes shown on the Web application and the clothes available for sale. Therefore, this garment database is located on the server side. Upon user selection, the chosen 3D garment model is downloaded to the client. These garments are saved into virtual reality modelling language (VRML) format, a widely used format for representing 3D on the Web.
- **Motion database**: The animation database contains samples of body motion data. Similarly to the garment data, selected motion data can be downloaded upon request. We obtain the motion data through the prerecording of a real person’s movement using an optical motion capture system.

- **Scene database**: Graphical elements that compose the background scene are stored in VRML files.

1.3. The client

Our Web client performs program execution after the executables are downloaded to the client as an ActiveX control. There are two main modules:

- The **body/garment sizing module** provides functionalities to deform the 3D mannequin from the customer's input body size and the resize the selected garment accordingly.
- Once the static shape is completed on the 3D mannequin and the garment, the animation of the dressed mannequin is taken care of by the real-time garment simulation module. This approach drastically reduces the amount of data transferred from the server to the client.

2. Body Database

Our measurement-driven body modelling approach largely consists of a pre-processing phase and a runtime phase. The body database is designed to store the pre-processing results that are fed into the runtime module. In this section we mainly discuss the pre-processing phase by describing what measurements were used, how the generic body models are prepared, and what data is required for the runtime body/garment sizing.

2.1. Use of measurements

The application uses 8 primary measurements (listed in Table 6-1), which have been defined in sizing surveys [HOHE 02] for apparel design. We assume that the users are aware of their own measurements or that measurements have been retrieved from 3D scanners or tapes.

Body measurement	Definition
Stature	Vertical distance between the crown of the head and the ground.
Crotch length	The vertical distance between the crotch level at center of body and the ground.
Arm length	The distance from the armscye shoulder line intersection (acromion) over the elbow to the far end of the prominent wrist bone (ulna) in line with small finger
Neck girth	The girth of the neck-base
Chest/Bust girth	Maximum circumference of the trunk measured at bust /chest height
Underbust girth	Horizontal girth of the body immediately below the breasts
Waist girth	Horizontal girth at waist height
Hip girth	Horizontal girth of the trunk measured at hip height

Table VI-1: Measurement definitions [SEOH 03]

2.2. Generic models

Instead of constructing a new geometry for each model, we assume that the topology of the model is known in prior and shared by all resulting models. The idea behind this is to exploit the common structure of the objects we are dealing with. Aside from making the problem simpler and the modelling faster, deforming existing model to obtain a new one makes it easier to immediately animate the newly generated model.

The generic model is composed of a standard [HANI 03] human skeleton structure and a textured skin surface of approximately 6,000 vertices and 10,000 triangles. There are two models, one for each gender. The preparation of the generic models consists of three successive procedures: (1) An HANIM [HANI 03] skeleton is created and adjusted properly in relation to the body mesh. (2) The skinning data is then calculated. (3) The body mesh is segmented and the whole body structure is exported into the HANIM format as described in Chapter III.3. The whole process is made in the 3DS Max ^{TM} environment [3DSM 03].

2.3. Body sizing by example-based approach

The body-sizing module is in charge of modifying the body according to the input measurements. The approach keeps constant topology of the body and only the geometry is modified. The approach is based on interpolation trough examples. A pre-processing stage is needed for the interpolator to learn the body shape variation through a set of scans of real bodies with measurements. The run-time application generates a new body shape by interpolating the shapes available in the database. Further explanations are given in the three papers [SEOH 03] [SEOI 03] [SEOJ 03].

3. Garment database

A set of garments is calculated for the generic model. These garments are grouped together in the garment database. For creating clothes usable in the online clothing store, designers must draw the patterns, pre-process the garments and export them into VRML format. Several authoring tools [VOLI 02] [CORD 02] have been developed to assist designers in their work.

3.1. Design and pre-processing of Garments

The garment creation is done using our in-house software [VOLI 03]. Since the garment sizing is handled online by the body/garment-sizing module, only one simulation is necessary for each garment item to be prepared on the generic model.

The pre-processing consists of preparing the garments so that they can be directly used for real-time animation. The cloth simulation is based on the approach given in Chapter IV. Each vertex of the cloth is assigned to one of three segments: tight, loose and floating garments. Like the design of garments, the pre-processing is done once. For change of body measurements during the runtime, the garment-sizing module adjusts automatically the size of the cloth without going trough the process of pattern design.

Defining skinning data is one step of the pre-processing stage to be done on the garments. As stated in the previous section, the cloth deformation method makes use of the shape of the underlying skin. Each vertex of the garment mesh is associated to the closest triangle, edge or vertex of the skin mesh (Figure VI-3).

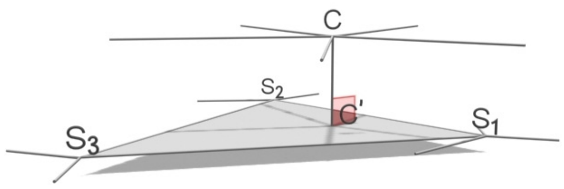


Figure VI-3: Mapping of attachment information

Once segmented, 3D cloth models and the associated data can be exported to VRML format for use in the online clothing store. These models are located on the server, allowing easy update and maintenance.

4. Motion database

What are probably the major criteria in garment animation is of much less concern in body animation. Commercially available motion capture systems offer a relatively easy way of recording a human performer's movement. Because the generic models we use are described in HANIM standard, the animation data has been converted such that it is immediately applicable to HANIM models. The converter first computes the correspondence between the two skeleton hierarchies (one from the motion capture system and the other from the HANIM) and performs the transformation on each joint angle for each frame of animation in order to resolve the difference in their stand-by posture.

The converted animation data is exported in VRML format using the *Interpolator* node and can be applied to the HANIM body model in a frame-by-frame basis. The database is made of several typical motions that people do when trying clothes – walking and turning to mention a few.

5. Integration of the client

The client application is not only involved in the visualization of garments, but also used for the calculation of the cloth and body deformation. As shown in Figure VI-4, the architecture of the client makes use of three different layers for the implementation: C++, JavaScript and HTML.

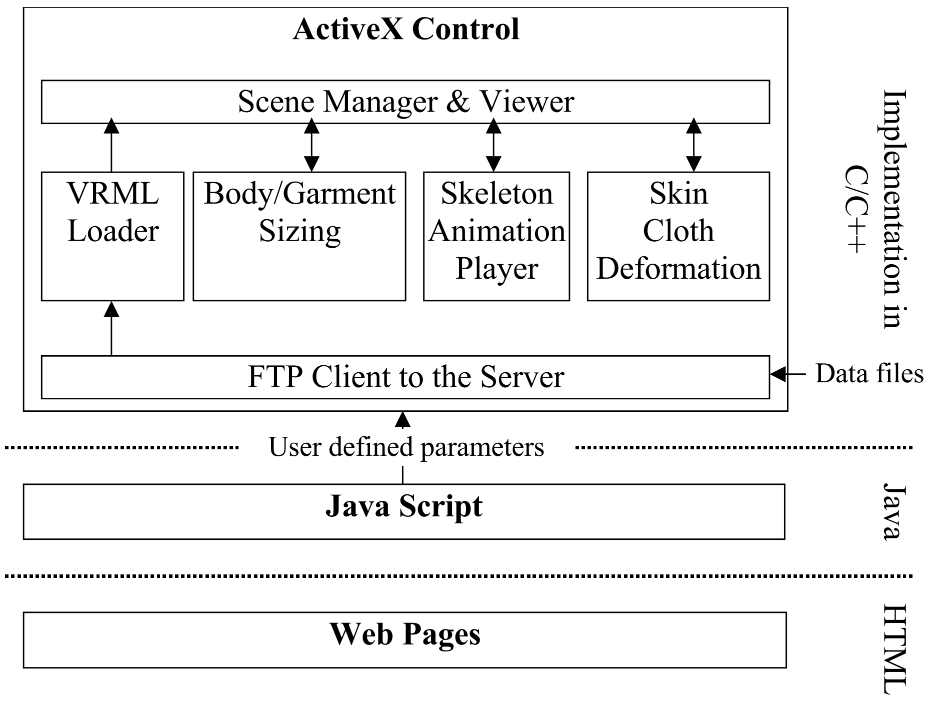


Figure VI-4: Overview of the Client Architecture

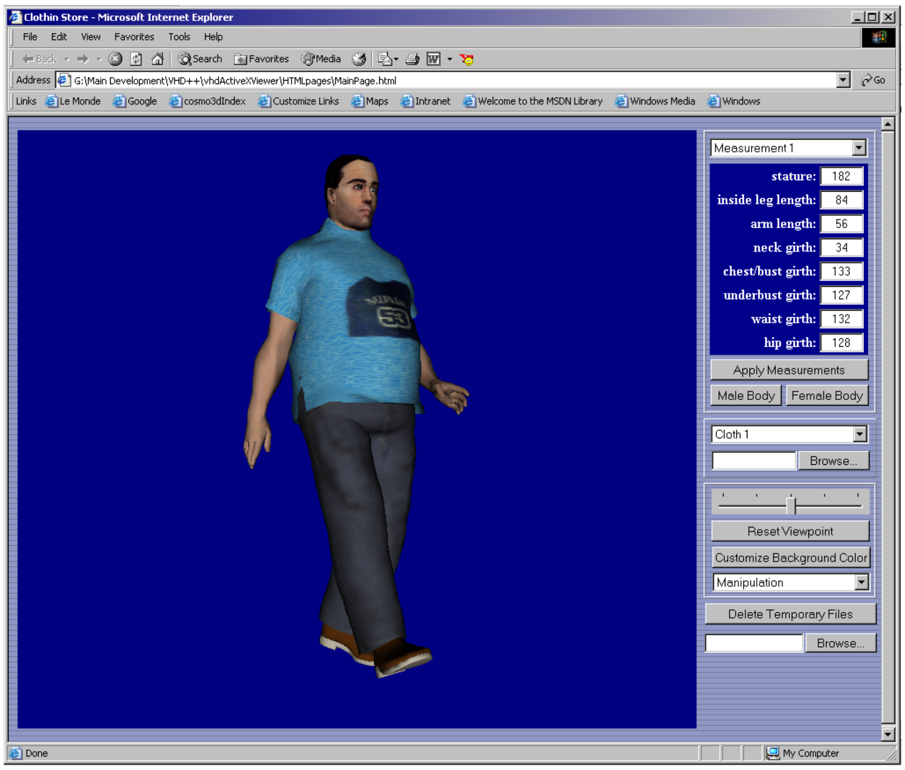
5.1. The ActiveX Control

Modules for real-time animation and visualization have been developed in C++ and integrated into an ActiveX control. ActiveX controls are components that can be embedded within Web browsers like Internet Explorer. Because they can be written in any language, they offer the best performance for time-critical applications. Another advantage of ActiveX controls is that their installation on the client machine is transparent to the user. They are automatically downloaded and installed at the fist access to the Web page.

Our ActiveX control is composed of several modules: the FTP client for downloading data from the server, the VRML loader that is in charge of loading the data into the scene manager, the skeleton animation player for animating the mannequin skeleton, the body/cloth sizing module for fitting the body and clothes to the user measurements, the skin and cloth deformation module for real-time animation, and the 3D viewer for visualization of the scene.

5.2. JavaScript and HTML Pages

JavaScript together with HTML are used for the implementation of the graphical user interface. They provide a good solution for user interaction. JavaScript allows complex functionalities such as keeping trace of the user choices, widget managements, and sending the user defined parameters to the ActiveX Control. Figure VI-5 shows the Web page where the user enters the body measurements and visualizes the animation of the 3D mannequin.



Clothin Store - Microsoft Internet Explorer

Address G:\Main Development\VHD++\vhdActiveXViewer\HTMLpages\MainPage.html

Measurement 1

stature:	182
inside leg length:	84
arm length:	56
neck girth:	34
chest/bust girth:	133
underbust girth:	127
waist girth:	132
hip girth:	128

File Edit View Favorites Tools Help
Back Search Favorites Media
Links Le Monde Google cosmo3dIndex Customize Links Maps Intranet Welcome to the MSDN Library Windows Media Windows
Go
Apply Measurements
Male Body
Female Body
Cloth 1
Browse...
Reset Viewpoint
Customize Background Color
Manipulation
Delete Temporary Files
Browse...
Done
My Computer

Figure VI-5: ActiveX viewer and HTML/JavaScript user interface

6. Body/garment sizing module

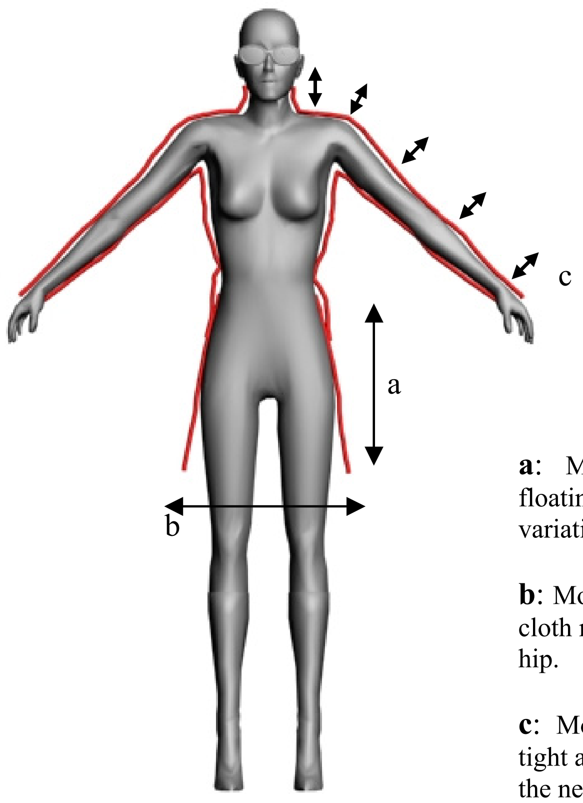
The main task of body/garment sizing module is to manage the proper sizing of the 3D mannequin in a VRML scene. As described above, it makes use of the generic model and the interpolators to evaluate the joint and shape parameters to deform the generic model. It first deforms the body model by applying the evaluated parameters from the interpolators as a function of the measurement input. The garments are then deformed accordingly such that they fit to the deformed body.

6.1. Body sizing

Upon the measurement values or a specific location in the measurement space chosen by the application at runtime, the interpolator evaluates the necessary deformation by efficiently blending the examples with known measurements to produce an interpolated shape [SEOH 03] [SEOI 03] [SEOJ 03].

6.2. Garment sizing

After the body model is properly deformed from the given measurements, the garment-sizing module modifies the garments in a way that they fit to the new body.



a: Modification of the length of the floating cloth region using the length variation of the legs.

b: Modification of the width of the floating cloth region using the width variation of the hip.

c: Modification of the rest shape of the tight and loose cloth regions so that it fits to the new body surface.

Figure VI-6: Sizing of the clothes according to the new body measurements

As explained earlier, the cloth mesh is segmented into three parts, two regions where the cloth vertices are attached to the body surface and one region where the garments move freely. Thanks to the cloth attachment data (Figure VI-3), the position of vertices belonging to the first two regions are easily computed from the position of the underlying skin vertices weighted with the barycentric coordinates (Figure VI-6(c)). Cloth vertices belonging to these two regions follow the skin surface, even in the case where the skin is deformed by the body-sizing module.

For the sizing of garments belonging to the floating cloth region such as skirts or dresses, the diameter and the length of legs are calculated in the body deformation module. The length and the width of the garments are modified accordingly to the new measurements of the legs (Figure VI-6(a) and (b)).

7. Skin/Garment simulation module

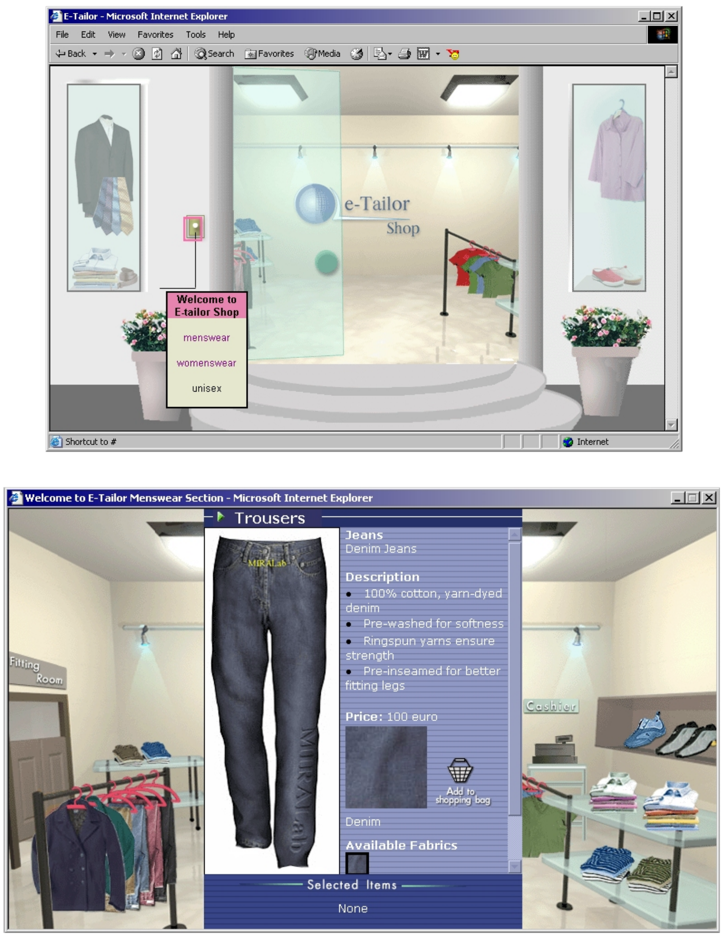
After generating the dressed body according to the user measurements, the motion data applied to the HANIM skeleton drives the skin and the cloth deformation. This section addresses the computation of the real-time simulation of bodies and clothes according to the animation of the underlying skeleton.

The skeleton driven deformation is based on the method described in Chapter III.2.2. At each frame of the animation, the position of vertices of the skin surface is calculated using the weight values and the transformation matrix of the joints. The result is a segmented body that appears as a seamless body in the rendering view port. This method combines the speed of the deformation of segmented bodies with the visual quality of seamless bodies.

The garment simulation makes use of the approach described in Chapter IV. Vertices are animated with different methods, depending on the segment they belong.

8. Results and discussion

Figure VI-7 shows the main graphical user interface (GUI).


E-Tailor - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
e-Tailor
Shop
Welcome to E-tailor Shop
menswear
womenswear
unisex
Shortcut to #
Internet
Welcome to E-Tailor Menswear Section - Microsoft Internet Explorer
Trousers
Fitting Room
Cashier
Add to shopping bag
Selected Items
None

Jeans
Denim Jeans

Description
- 100% cotton, yarn-dyed denim
- Pre-washed for softness
- Ringspun yarns ensure strength
- Pre-inseamed for better fitting legs

Price: 100 euro

Denim

Available Fabrics

Figure VI-7: The online clothing store front page [ATCA 03]

After user authentication, the communication is first established; the ActiveX control is downloaded if it has not already been installed on the machine. Initialization of the control is executed, including the download of the data for the body/garment sizing module. The visitor to the online virtual store follows several successive steps provided:

- After the selection of the garment item through the 2D garment catalogue, the user selection is passed to the ActiveX control through JavaScript functions.
- The requested garments and motion data are downloaded and are then loaded into the scene using the VRML loader.
- The user continues to the body measurements page containing a number of fields that are adjustable. After the user enters their measurements, the body and garments are resized accordingly.
- Finally, a trigger action starts the execution of real-time simulation. While the skeleton animation player updates the angle values of the mannequin's skeleton, the skin and clothes are deformed accordingly.

The Virtual Try-On clothes have been created from real models as shown on the figure below.

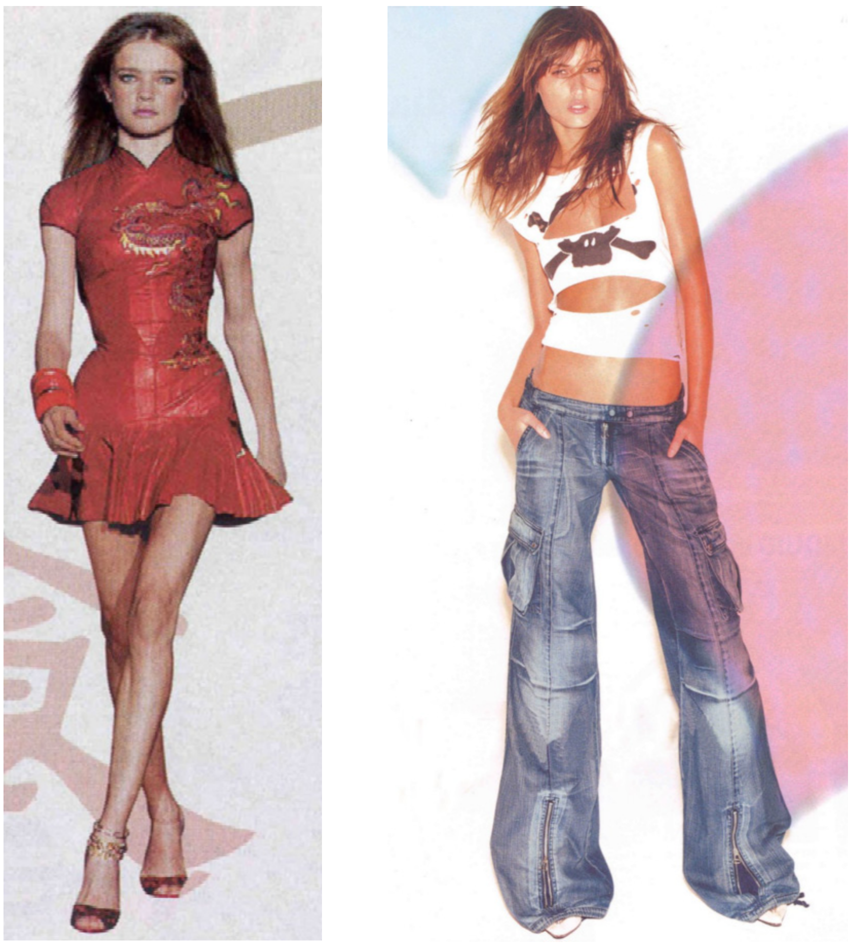


Figure VI-8: Real models that have been used for the making of the virtual clothes

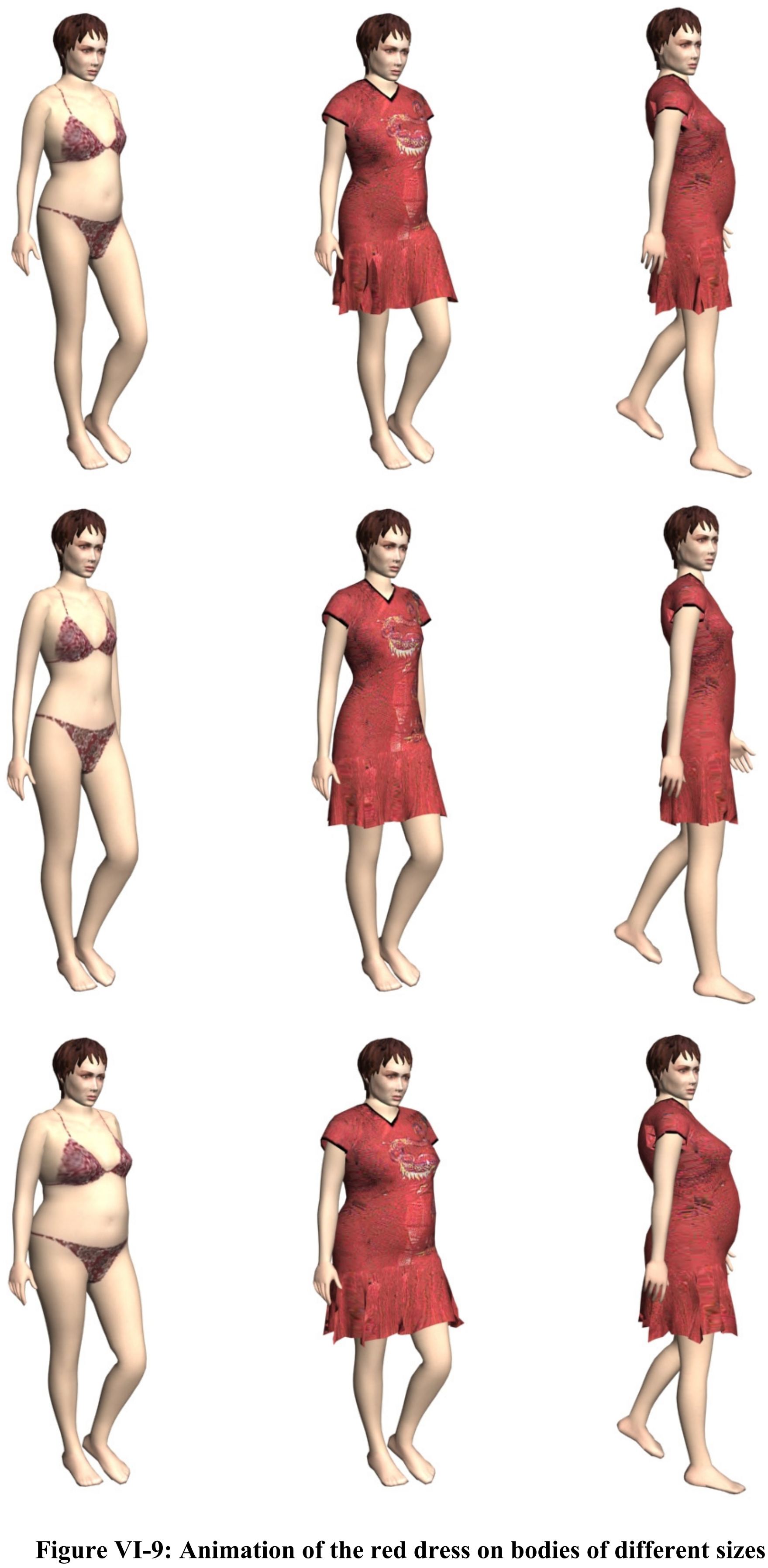


Figure VI-9: Animation of the red dress on bodies of different sizes

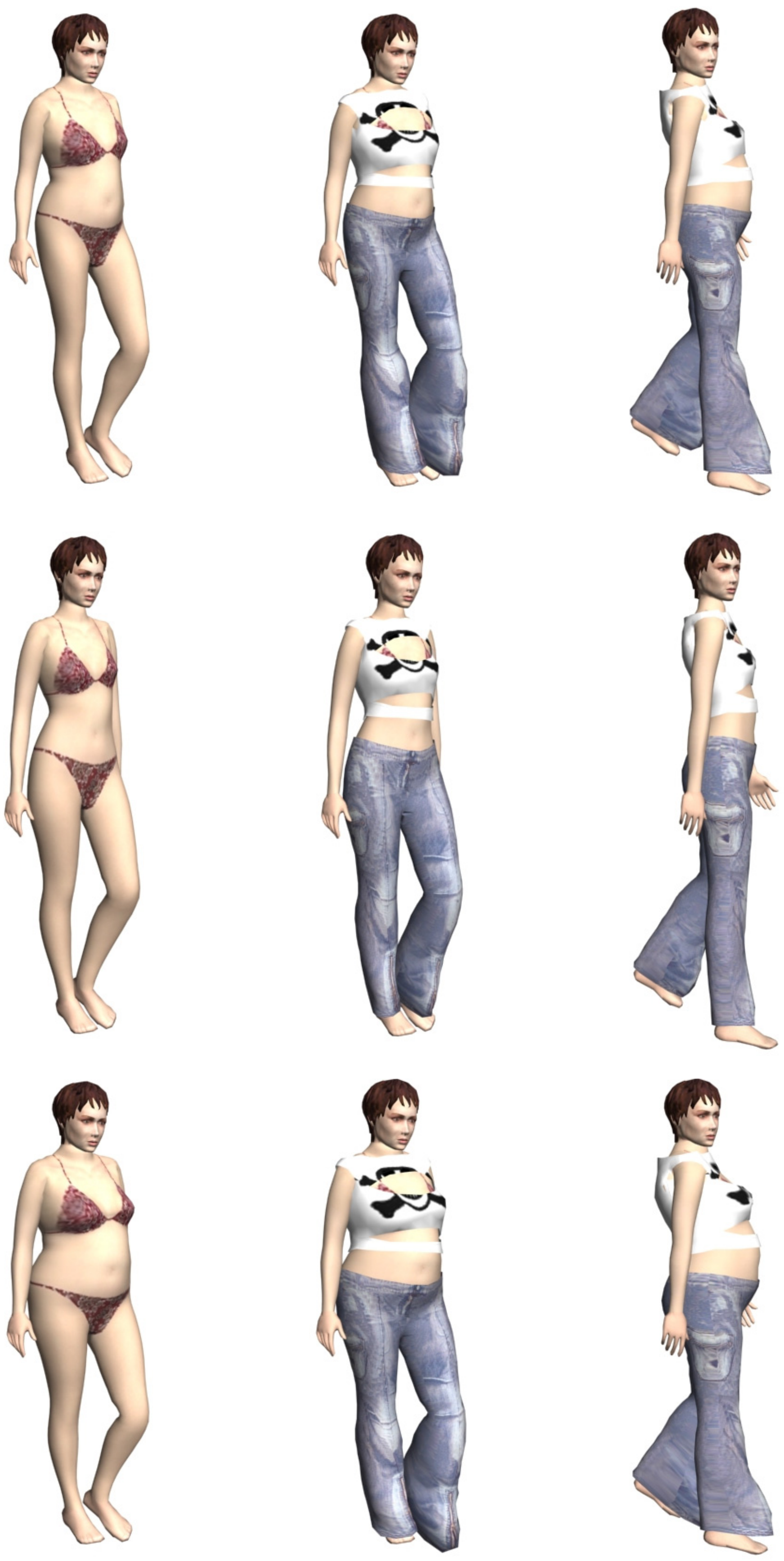


Figure VI-10: Animation of the trouser on bodies of different sizes

The time needed for the simulation is obviously function of the complexity of the body and the garment model. The table below shows the average size of the different data to be downloaded for a simulation sequence.

Data	Average size in Mbytes
Body data with generic textured model (8000 polygons)	2.0
Motion data (Duration 10 sec.)	0.2
Garments with textures (4000 polygons)	1
Scene with textures (1000 polygons)	1

Table VI-2: Average size of downloaded data

The biggest amount of data to be downloaded is the body database. Fortunately, this database needs to be downloaded only once at the initialization stage of the ActiveX control. Other data needs to be downloaded every time the user changes the garments and/or motion data. The frame rate of the real-time animation of the mannequin in the ActiveX control is about 20 to 30 frames per second. See Chapter IV.8 for a further discussion of the performance of the body/garment deformation module.

9. Conclusion

In our work, various research problems have been solved in order to build a framework for the online clothing store. Design decisions, as well as related issues on creating, simulating and fitting 3D garments and human body models have been discussed. The process of preparing various data components has matured, making it feasible to efficiently handle the creation of graphical contents, which are immediately usable for interactive visualization on the Internet. The bodies and the garments created with our approach correspond to real body measurements at the critical locations defined for the garment design. In particular, methods for efficient handling of the online body modelling, garment selection, fitting and simulation have been developed and integrated into Web components, optimized for the execution on the Internet. Finally, we set up an interactive and realistic virtual clothing store, where visitors can choose among many different types of garments and visualize them on animated mannequins based on their measurements.

Nevertheless, there are still rooms for improvements. The main limitation of the garment sizing is its inability to support wide changes of body measurements. Important changes of body shape cause the modification of cloth behaviour. For instance, loose cloth regions may become tight in case the body size has been largely increased. Our cloth deformation method relies on pre-determined tight and loose cloth regions; therefore it is necessary to update these regions every time the body size has been modified.

The sizing works only for both the garment and the body and the simulation of differently sized garments on the same body remains as future work. We also plan to enhance the input parameters to model the 3D mannequin such that it allows the best possible correspondence to a particular individual.

Chapter VII. Conclusion

In this chapter, we review the proposed methods for real-time cloth simulation. The methods will be compared with the state of the art and the contributions will be identified.

This chapter is organized around three sections. In the first section, we compare the advantages and drawbacks of the hybrid and example-based methods respectively described in Chapter IV and Chapter V. The second section identifies the contribution and the strengths of the approaches. In the third section, limitations and future research directions are listed.

1. Comparison of the hybrid and example-based approaches

In this dissertation, two techniques for animating clothes in real-time have been proposed. The first technique is based on the segmentation of a garment into pieces that are simulated by various algorithms, depending on how they are laid on the body surface and whether they stick to or flow around it. The second technique proposes a cloth simulator that can learn the cloth behaviour through a sequence of pre-computed, physically based simulation. It simulates the garment in two phases: In the first phase, a rough mesh reproduces the dynamic behaviour of the garments, where the physical properties of the mesh are defined by the pre-calculated sequence. In the second phase, the fine mesh simulates the detail shape of the garments by using an interpolator that has been constructed from the pre-calculated sequence.

1.1. Common points

Both techniques rely on the classification of the cloth vertices into the tight, loose and floating regions (see Chapter IV.2 and Chapter V.2.2.4). Such classification has permitted to utilise methods that are optimized for specific cloth behaviours. These specialized methods are faster than general, versatile ones because they are based on strong assumptions about the movements of the garments in relation to the skin surface and thus they avoid doing unnecessary computation.

As well, both techniques share the same idea: the cloth surface is attached to the skeleton and its deformation is computed in relation to it. In other words, garments are considered as a second skin layer and their movements closely related to the skeleton animation. With this idea in mind, the mechanical model and the collision detection have been adapted.

1.2. Differences

The main difference between the two techniques concerns the input data that is needed for the pre-processing. The hybrid approach requires only the static shape of the clothes fitted to the body whereas the example-based method additionally uses a pre-computed animation. Consequently, at least three differences can be identifiable between the two approaches. First, the static mesh of the garments does not provide enough information about the cloth behaviour. We remind the reader that the segmentation used in the hybrid method is only based on the distance between the cloth and the skin surface at a static shape. Cloth vertices that are located close to the skin surface are categorized into tight region and those that are far into floating region. Unfortunately, the behaviour of the clothes cannot be predicted accurately only by using the static shape. Therefore, the hybrid approach requires the user to interactively correct the segmentation afterwards. On the contrary, the example-based approach fully utilizes a pre-computed sequence of the clothes. The analysis of the movements of the garment vertices in relation to the skeleton joints makes it possible to automatically process the clothes without any user intervention. On the other hand, it requires much more time to generate a pre-computed sequence of the clothes than to just compute the rest shape of the clothes that are well fit to the body, as it is required by the hybrid method.

In addition to the automatic pre-processing, the example-based approach takes advantage of the statistical study on the pre-computed cloth animation. Whereas the hybrid approach makes use of a unique mechanical model for each of the three cloth regions (tight, loose, and floating), the example-based one builds an interpolator whose internal parameters is adapted to the cloth behaviour, so that it can simulate a wide range of fabric properties. In other words, the example-based approach is able to learn the mechanical properties of the fabrics through the pre-computed simulation.

Another advantage of the example-based approach is that the simulation of three cloth regions is based on a uniform set of simulation techniques and data structure, whereas the hybrid approach uses different methods for each cloth region. The use of the uniform methods significantly simplifies the integration of the three different regions. We note that extra computation is needed in the hybrid approach, in order to smoothly connect vertices of different regions (see Chapter IV.6).

Finally, the Chapter VI has shown one application of the hybrid approach that is not handled by the example-based approach at present time. One important feature that is needed in the frame of the Virtual Try-On application is the possibility of sizing clothes, as described in Chapter VI.6.2. Cloth sizing seems to be difficult to adapt to the example-based approach because the modification of the size of a garment necessarily changes its behaviour. For the time being, the example-based approach is limited to the simulation of cloth behaviours for which it has been trained; the dynamic change of cloth behaviour during the runtime simulation is not handled with the current technique.

2. Contributions

Through our research on real-time cloth simulation, we have introduced four main key ideas to improve the computation speed of clothes while keeping satisfactory deformations.

2.1. Cloth surface as being attached to the skeleton

In this thesis, we have introduced two novel approaches where the cloth surface is considered not as an independent deformable object but rather as being attached to the skeleton of the virtual human. The mechanical model and the collision detection algorithm have been modified accordingly. Such approaches permit many optimisations especially for the collision detection.

2.2. Alternative solutions to physically based deformation

Our second contribution has been to develop a set of alternative methods to simulate clothes. Based on the first point 2.1, two approaches have been developed; the first one based on a hybrid approach using geometric and physically based methods, the second approach exploiting a pre-computed simulation sequence of the cloth to animate. The deformation of physically based deformable models requires huge computation power and can only be simulated with a class of methods that has been especially developed for these objects. The main property of these objects is that their shape depends on the position and velocity of the vertices that compose them. The simulation requires taking into account a large number of degrees of freedom whereas geometric objects depend on very few parameters. Several researchers have tried to replace the physically based methods with a neural network [GRZE 98] or example-based method [JAME 03]. However, these approaches suffer from the limited number of degrees of freedom that they can handle. For instance, the approach by James et al. cannot be used for the animation of clothes on bodies. We did not propose to replace totally the physically based method but we rather limit its use to the simulation the global movements of the clothes. Details are generated with geometric methods that require less computation power.

2.3. Versatile deformation

The versatility is one of the main strength of the example-based deformation method presented in Chapter IV. This versatility has been achieved by the use of hybrid approaches using physically based and geometric deformation. We have demonstrated such versatility by applying the same method for simulating the skin as well as large skirts. It can be successfully applied to the simulation of a wide range of clothes, from tight to floating garments.

2.4. Fully automatic pre-processing

The hybrid deformation method detailed in Chapter IV has exposed the necessity that the pre-processing of the clothes should run without user interaction. Two main reasons can be highlighted: First, methods that require the user input may have higher failure rate because the user may give wrong information. For instance, an incorrect segmentation would result in an incorrect simulation. Second, manual input would require many trials and errors, making the garment creation process much slower. The second approach (Chapter V) has been developed with this criterion in mind. This approach is based on the analysis of a pre-computed cloth simulation sequence. Such analysis provides all the information needed for the pre-processing of the clothes for real-time simulation.

2.5. Application of the real-time cloth technology

In the last chapter, we have introduced a possible application of real-time cloth technologies and how it could be used for the everyday life. We have as well discussed how these technologies could be integrated with other components such as body sizing and animation. Various research problems have been solved in order to build a framework for the online clothing store. Design decisions, as well as related issues on creating, simulating and fitting 3D garments and human body models have been discussed.

3. Limitations and future research

Through the evaluations we have made on the proposed method, some limitations have been shown. In this section, we summarize all of them and we discuss what can be done to remedy these limitations. We propose as well possible directions for further research on real-time clothes.

3.1. Better interaction of the clothes with other objects

The research on real-time cloth simulation methods returns to a trade-off between the versatility of the simulator and the computation load. Versatile methods tend to be slower. As these methods make use of general assumptions, they offer more robust solution but in some cases do extra computation that is not necessary for the simulation of clothes. In the other hand, specialized methods are faster because they use stronger assumptions and avoid doing unnecessary computation. Our both approaches fall to that second category. While offering high computation speed, they cannot handle some of the cloth movements such as those appearing during dressing or undressing. More generally, the clothes are unable to interact with objects other than those that have been taken into consideration during the pre-processing phase. The list of objects that can potentially interact with clothes as well as the way these objects interact is defined at the pre-processing stage and cannot be change during the real-time simulation. One future research work could be to investigate a method to update automatically the list of possibly interacting objects.

3.2. Level of Detail

In the framework of real-time simulation and visualisation, the Level of Detail is one of the most common techniques to reduce the computation load. The main idea of the Level of Detail is to adapt the complexity of the model and/or quality of simulation according to the distance of the deformable object from the camera viewpoint. A model that is located far from the viewpoint takes small screen area and therefore does not to be highly detailed. The adaptation of the cloth simulation to the Level of Detail can be one of the future developments. In our case, the simplification can be typically achieved by replacing the simulation of the mass-spring system by the skeleton driven deformation. Another way of reducing the computation load could be to decrease the number of polygons dynamically on the cloth under simulation. Several techniques have been proposed in the literature and can be easily adapted to our case [GARL 97], [HUGE 97].

3.3. More accurate statistical analysis

In the example-based method proposed in Chapter V, some of the pre-processing computations are based on statistical analysis. The approximation of the parameters of the interpolation function to generate the shape fine detail of the mesh is made by linear regression. The linear regression is known to be sensitive to outliers if the sample size is too small. Outliers are the samples that are far from the interpolation function and they significantly degrade the accuracy of the regression coefficients. One future work would be to adopt statistical methods to automatically identify and remove the outliers from the linear regression.

3.4. More robust cloth deformation for body sizing

Deformation of clothes has two causes, one related to the body animation, and the other one to the body sizing. The cloth deformation caused by body animation has been extensively analysed and two approaches have been proposed in the frame of this thesis. The other way of deforming clothes, which is body sizing, has been quickly tackled in Chapter VI in the frame of Virtual Try-On. The method fits the rest shape of the clothes to new body measurements. One drawback of this method is that it does not take into account that the body sizing may modify the behaviour of clothes. For instance, loose clothes worn over a skinny body may become tight after the body has been resized to larger measurements. A more robust approach would be to re-compute the pre-processing data of the cloth simulation method and update the segmentation of the cloth surface.

3.5. Extension of the approach to other physically based deformable models

The example-based approach presented in Chapter V is worth being extended to the simulation of other physically based models such as hair. In this approach, the simulation is made up of two steps. First, the global cloth behaviour is generated with a physically based model. In the second step, the details of the cloth surface are simulated with a geometric method. The adaptation of this model to the simulation of hair would require the rewriting of the mechanical model to handle the simulation of hair strands. In addition, the geometric method would have to be adapted in order to handle the deformation of triangular stripes instead of polygonal surfaces.

3.6. Use of graphics hardware

Recent research works have shown that graphic hardware can be used to decrease the computation time of the physically based simulation by implementing the matrix solver of the implicit Euler method on the GPU. This has been made possible because matrices and textures have similar array data structure. According to recent studies reported in [BOLZ 03] [KRÜG 03], matrix operators on GPU are 15 to 20 times faster than their counterpart on CPU. These GPU matrix operators could be implemented in the two cloth-simulation methods presented in Chapter IV and Chapter V as they both make use of the conjugate gradient method.

Appendix A. Implementation and Software Architecture

Our research on real-time clothes has involved the development of a set of tools and libraries. Authoring tools aim at helping CG artists in their work for the creation/edition of the virtual human models. The libraries are modules that are used by our platform to simulate in real-time virtual humans and clothes.

1. Architecture of the authoring tools

Authoring tools are used by CG artists to provide the content of real-time simulations: virtual human models, objects in the scene, motion data… These tools can be classified into three categories. The first category comprises all the tools for the edition and conversion of motion data for animating virtual humans. The second category includes the tools dedicated to the modelling of the virtual human models and the skin deformation. The last category contains the tools for cloth editing and pre-processing.

The pipeline of creating virtual humans is partially based on commercial software. Commercial software offers a broad range of features and some of them have required several years of development. Integrating them into our pipeline has shortened the time of development. Five commercial applications have been used:

- 3D Studio Max [3DSM 03] is an animation and modelling tool. The most used features are mesh edition and animation by key frame. This application offers the possibility to develop plug-ins through a SDK.
- BonesPro [DIGI 03] is 3DS Max plug-in that is used to define the skinning data for the skeleton driven deformation. This plug-in offers the possibility to export the generated skinning data. We use this feature for creation of the real-time models.
- Vicon [VICO 03] is a motion-capture system using optical trackers. Using this tool, body motion of real actors cab be tracked and applied on our virtual human models.

- Character Studio [CHAR 03] is a "3D Studio Max" plug-in that is dedicated to character animation. One of its features is the computation of the skeleton animation from the marker positions provided by the Vicon system. Its offers as well a set of tools for body motion editing.

There are mainly two reasons for which the tools have been developed.

The first reason is to develop tools to fill or improve the missing features of the commercial tools. This development includes the creation of applications to avoid repetitive tasks, to integrate heterogeneous tools together to create a seamless pipeline, to develop plug-ins to facilitate the exchange of data among software. In other word, these tools aim at making easier the CG artist work.

The second reason of developing new applications is to make available to CG artists new functionalities. These tools aim at extending the ability of designers to work on new area such as real-time cloth simulation.

1.1. 3D Studio Max plug-in architecture

This section briefly describes the internal architecture of 3D Studio Max. Scenes and objects are represented through a scene graph. The scene graph defines the relation parent/child among the objects. This scene graph is acyclic; it means that every node has only one parent. The children inherit the transformation matrix of their parents. Such hierarchy can be used to model the H-Anim skeleton (Figure A-1). An in-depth description about the 3D Studio Max environment can be found in the official user manual [3DSM 03].

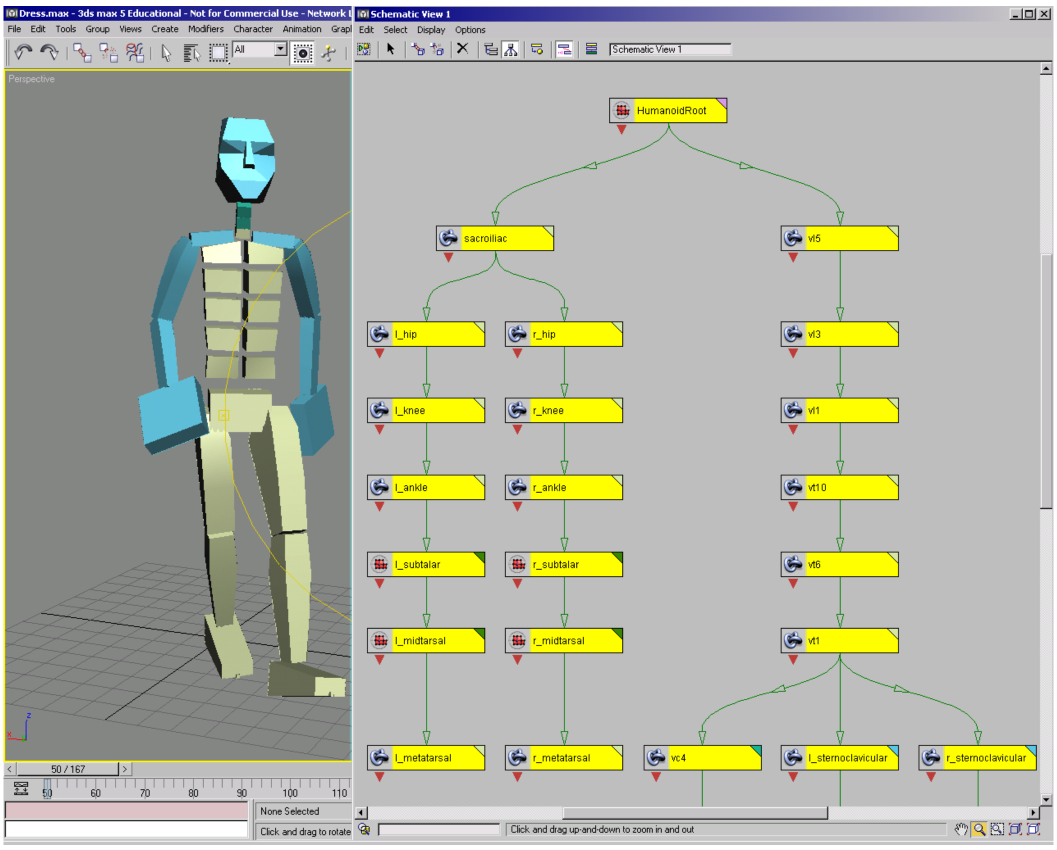


Figure A-1: H-Anim hierarchy in 3D Studio Max scene graph

Each object is modelled through a pipeline. The bottom of the pipeline is the base object. They can be a mesh, NURBS surface, patch or spline. The geometry and topology of this base object is modified through a certain class of plug-ins named as "Modifier". A modifier takes as input the results given by the modifier below in the pipeline, processes the data, and produces the output to the next modifier in the pipeline. Modifiers can modify the geometry, topology, and materials of the object. The output of the final modifier is rendered into the viewport window.

3D Studio Max offers a Software Development Kit to develop user-defined plug-ins. The SDK is a set of C++ classes and related routines. Writing a plug-in involves creating objects from the classes and implementing methods to allow communication between the plug-in and the system. There are several ways of developing plug-ins. In the frame of the research on real-time clothes, we have mainly used three of them: a) import/export plug-ins to exchange data between 3D Studio Max and other programs, b) modifier plug-in to apply user-defined modification of the scene-graph objects, c) utility plug-ins that can be used to implement modal procedures. For the in-depth discussion on writing plug-ins for 3D Studio Max, we refer to SDK documentation [3DSM 03] [BICA 00].

1.2. Architecture of the pipeline production

The pipeline production comprises all the tools used by CG artists to create the content for virtual reality scenarios. This section aims at giving an overview of the stage of creating the virtual human models. The creation of these models is done throughout four steps.

The creation of models starts by the construction of the skeleton hierarchy. The hierarchy defines the proportions of the virtual human. Character Studio [CHAR 03] is the main tool for the execution of this task. Several plug-in have been developed to convert automatically the Character Studio skeleton (namely Biped) to H-Anim skeleton. H-Anim is a worldwide-recognized format for the modelling of avatars for real-time simulation. CG artists have also the possibility to create these models from scratch by creating the skeleton hierarchy in 3D Studio Max using boxes. However, the construction of the skeleton by hand is a complex task. Character Studio is preferable because building the hierarchy is done in one mouse click.

The second step is the creation of the motion data. The motion data can be either created by hand using the 3D Studio Max environment or generated by motion capture. The second option offers higher quality because the movements can be generated by real actors being captured by the Vicon system [VICO 03]. The recorded movements are saved in the form of tracker trajectories (CSM files). These trajectories are converted into skeleton animation by Character studio. Finally, the skeleton animation is adapted to the H-Anim structure by our in-house plug-in.

The third step consists of attaching the skin surface to the skeleton, i.e. defining the data for the skeleton driven deformation. This step is done with BonesPro [DIGI 03]. This 3D Studio Max plug-in attaches the skin to the skeleton with default parameters. These parameters can be modified to fine tune the skin deformation. Once the skeleton driven deformation has been completed, the skinning data is converted to the H-Anim skeleton.

The final stage is the modelling and the pre-processing of the clothes. The design of the clothes is made with Fashionizer [VOLI 00]. The patterns are drawn and placed around the body. The cloth surface is discretized into triangles and the simulator computes the fitting of the clothes to the body shape. The resulting cloth shape is then processed to create the cloth model that can be used for real-time animation.

Additional works can be done to enhance the realism of the human model such as facial animation [KSHI 03] and hair simulation.

Note that Character Studio is the main commercial tool in the production pipeline. It is used for the creation of the skeleton and motion editing. In consequence, most of the in-house plug-ins has been develop to enable the conversion of the data (skeleton and motion data) between Character Studio and the H-Anim format.

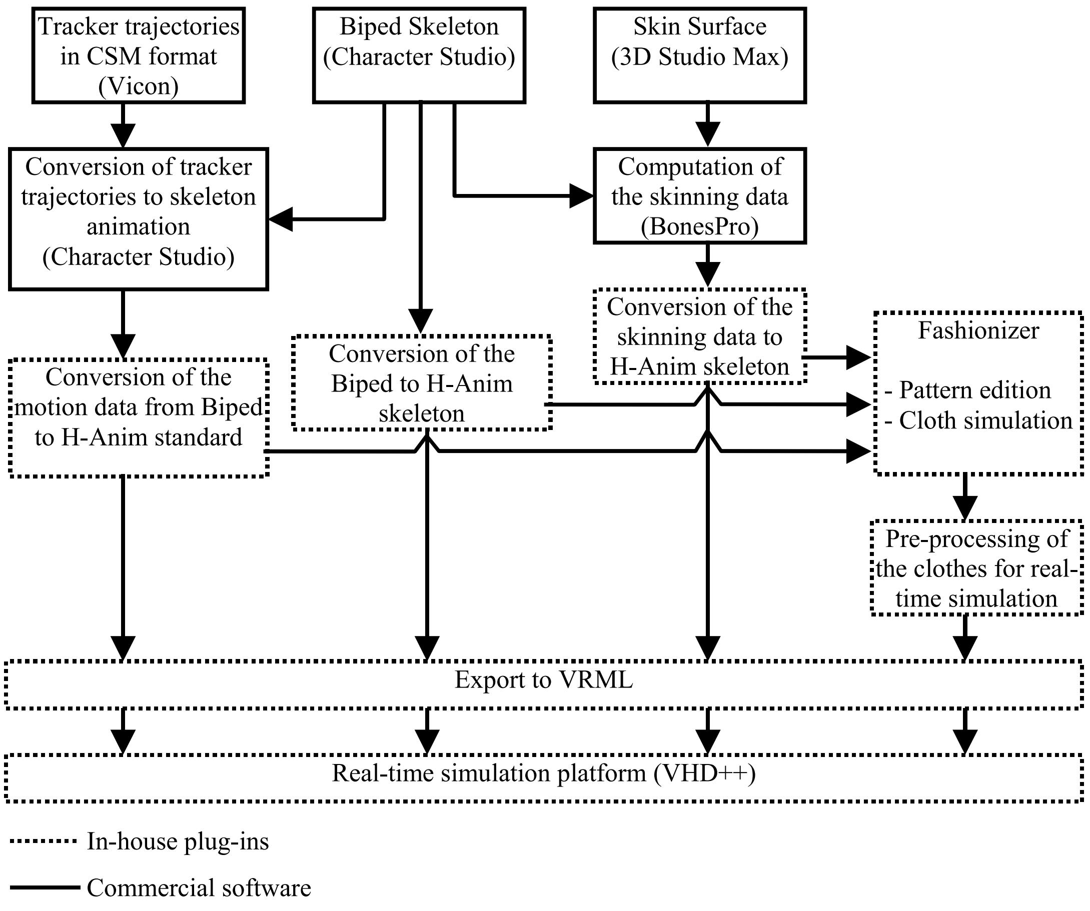

Tracker trajectories in CSM format (Vicon)
Biped Skeleton (Character Studio)
Skin Surface (3D Studio Max)
Conversion of tracker trajectories to skeleton animation (Character Studio)
Computation of the skinning data (BonesPro)
Conversion of the skinning data to H-Anim skeleton
Conversion of the Biped to H-Anim skeleton
Conversion of the motion data from Biped to H-Anim standard
Fashionizer
- Pattern edition
- Cloth simulation
Pre-processing of the clothes for real-time simulation
Export to VRML
Real-time simulation platform (VHD++)
In-house plug-ins
Commercial software

Figure A-2: Architecture of the pipeline production

1.3. Motion editing tools

Motion editing tools assist CG artists in their work for the creation of motion data to animate virtual humans. These tools have been implemented as an Utility plug-in [3DSM 03]. Prior to the creation of the H-Anim skeleton, the Biped (Character Studio skeleton) should be designed. The Biped skeleton can then be automatically converted to H-Anim using a one-to-one correspondence between the H-Anim and Biped joints. The animation data can be as well transferred from the Biped to the skeleton H-Anim. Some additional tools help the user to export/import H-Anim motion data and create postures on the H-Anim skeleton. See Appendix B for an in-depth description of these tools.

1.4. Skinning tools

The skinning tools help CG artists to attach the skin surface to the skeleton. The skeleton driven deformation makes use of skinning data (see Chapter III.2). These skinning data are defined thank to BonesPro. They can be exported to a text file and re-imported to 3D Studio Max using one of the in-house plug-ins. Other skinning tools are in charge of the preparation of the virtual human model for real-time simulation: segmentation of the skin surface (see Chapter III.3) and exportation of the body into VRML format.

1.5. Cloth editing tools

There are two platforms for cloth edition. The first platform is dedicated to the design of the patterns and the simulation of clothes based on realistic mechanical model of the tissue. Using this software, CG artists can pre-compute a sequence of the clothes they want to prepare for real-time simulation. This sequence is then used in our second cloth simulation platform.

2. Architecture of the VHD++ modules

Researchers have intensively participated to the virtual reality platform named VHD++. VHD++ is real-time development framework being a proprietary middleware solution of both MIRAlab and Vrlab [VRLA 03] laboratories. VHD++ is a highly flexible and extendible real-time framework supporting component based development of interactive audio-visual simulation applications in the domain of AR and VR with particular focus on virtual character simulation technologies. C++ has been chosen as the main implementation language. The most important features and functionalities of the VHD++ framework as seen from these different perspectives are:

- Support for real-time audio-visual applications
- Extendible spectrum of technologies
- Middleware portability, extendibility and scalability
- Runtime flexibility: XML based system and content configuration
- Complexity curbing: multiple design patterns improve clarity and abstraction levels simplify description of problems
- Fundamental components and pluggable components hide implementation level details allowing to work on required abstraction level simplifying implementation constructs
- Large-scale code reuse: fundamental components and readymade components encapsulating heterogeneous technologies.

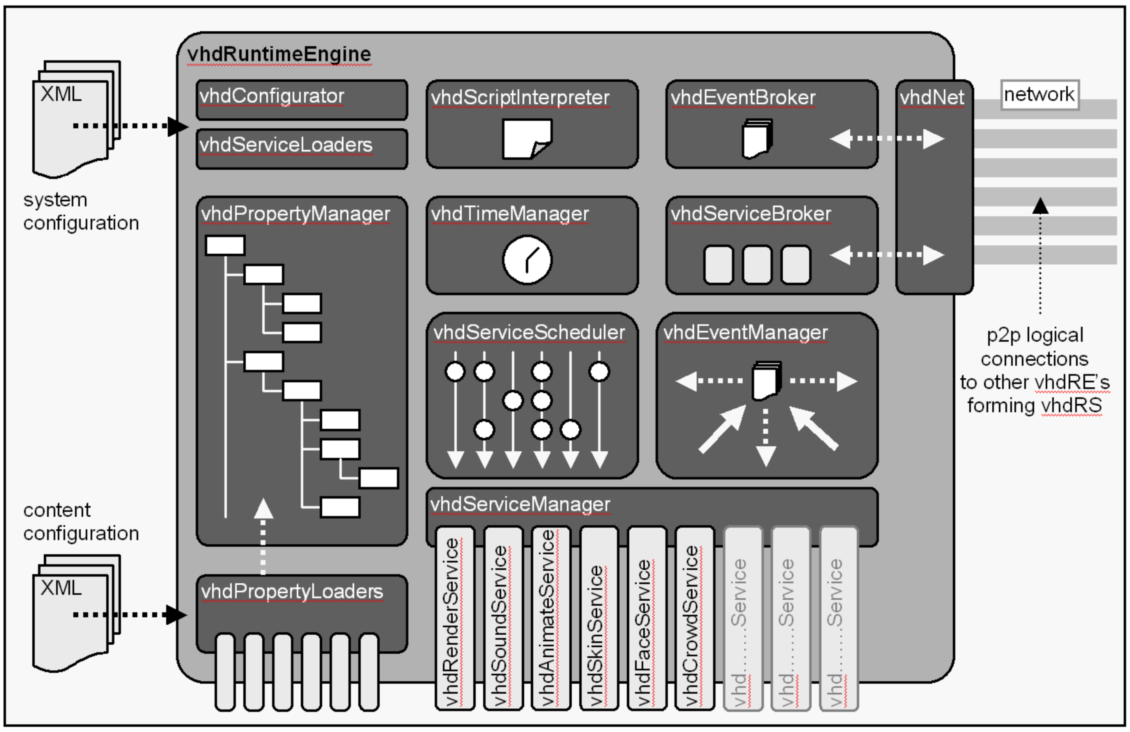


Figure A-3: Architecture of VHD++ [POND 03]

VHD++ is currently based on the Optimizer library. Optimizer is a toolkit that enables developers to draw large models interactively through the use of culling, multi-threading, level-of-detail (LOD) rendering, etc… OpenGL Optimizer uses Cosmo 3D scene graphs to organize its data. Cosmo 3D is a scene graph API which provides multi-thread processing of scene graph data, back face culling, engines, and sensors. For more information about Optimizer and Cosmo3D, see the Optimizer Programmer's Guide [OPTI 03].

The platform VHD++ has been widely used in European projects for all the real-time performances. Several papers [POND 03] [PAPA 03] dedicated to VHD++ have been published. They can be consulted for further details about this platform.

Our research work has been implemented as a VHD++ module, namely "vhdSkin". This module is composed of three different libraries, one dedicated to the skin deformation and the two others for the cloth deformation. It has been decided that the three libraries would be integrated into a single VHD++ module because they share most of the data structures.

2.1. Architecture of the VHD++ body animation pipeline

The module "vhdSkin" together with the other modules form the body animation pipeline. The simulation of virtual humans requires using different technologies such as skin deformation, skeleton and facial animation, etc. The pipeline defines the way these modules are arranged together and the order in which the different modules are executed. The overview of the pipeline is given on the figure below.

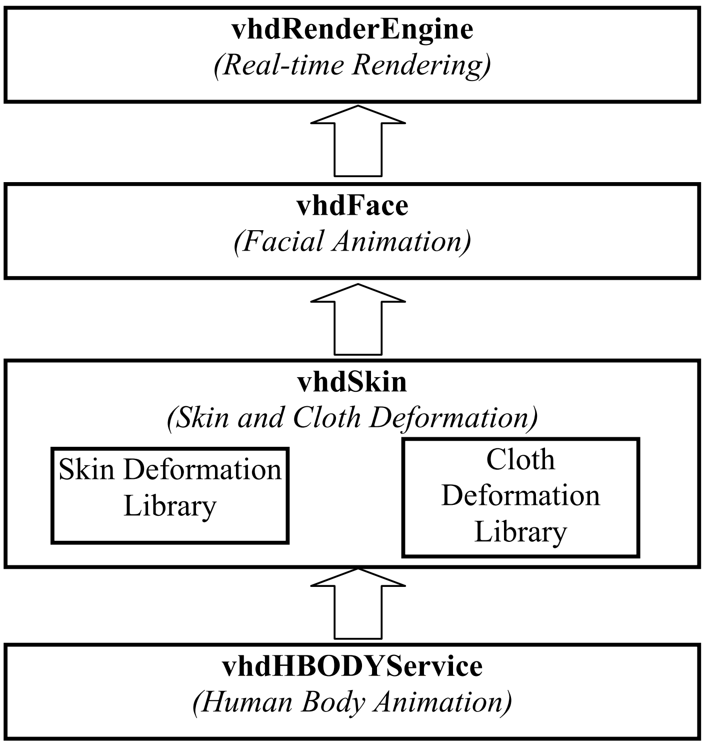


Figure A-4: Architecture of the body animation pipeline

2.2. Skin Deformation library

The skin deformation library is in charge of the deformation of the skin surface. Its implementation is based on the skeletal deformation method described in Chapter III.2.2. The library first starts by computing the transformation matrices of the joints by traversing the skeleton graph. A check is done to define whether the posture has been modified since last update; this check is necessary to avoid unnecessary computation. Finally, the new skin surface is computed using the new joint matrices. The skinning library is also able to handle rigid objects that are attached to the avatar skeleton such as hat, shoes, and bones… The interface the skinning library is composed of two methods, one for the initialisation of the data structure and the other to update the skin surface.

2.2.1. List of the header files to include

Three header files are required for using the skin deformation library:

- "CsClothSkinDeform.h" is the main header file. It contains the declarations of all the functions needed for the computation of the skin deformation. This is the only file that is required to be included for using the skin deformation library.
- "SkinTypes.h" includes the declaration of all the data structures used by the skin deformation library.
- "Vect3.h" includes a set of functions for the manipulation of 3D vectors: addition, subtraction, scalar and cross products, etc.

The two files "SkinTypes.h" and "Vect3.h" are includes in "CsSkinDeform.h".

2.2.2. Compilation flags

The file "CsClothSkinDeform.h" is the header file of four different libraries: the skin, cloth, level of detail, and body sizing libraries. A set of flags allows controlling which libraries to include in the compiled version.

- `CLOTH_DEFORM`: if defined, the cloth library is included.
- `HUMAN_LOD`: if defined, the LoD library is included.
- `BODY_GEN`: if defined, the body-sizing module is integrated [SEOH 03].
- `FLEXLM_DLL`: if defined, the library is licensed.

2.2.3. List of the functions

The functioning of the library is organized around three phases: initialisation, runtime and destruction.

2.2.3.1. Initialisation

```
void LoadSkin(csTransform *root,char *newPath);
```

- `root`: the node corresponding to the humanoid root in the Cosmo3D hierarchy the avatar.
- `newPath`: the file system path of the directory containing the segments of the avatar.

2.2.3.2. Runtime functions

This following function computes the deformation of the skin according to the current angles of the avatar joints.

```
float Deform(float timeStep=-1.0f);
```

- `timeStep`: this variable contains the elapsed time since previous frame. This time step is used for the cloth simulation (see section 2.3). If the time step is set to a negative value, it is then calculated using the internal clock of the computer.

The next function update the number of skin triangles according to the value given as a parameter. This function works only if the LoD has been activated at the initialisation of the library.

```
void DecimateSkinEdgesForLoD(float LoDRatio);
```

- `LoDRatio`: this parameter defines the level of detail; a value of 1 makes the avatar mesh at the lowest resolution, 0 the full resolution.

2.2.3.3. Destruction

The destruction of the library is processed with the function:

```
~CsClothSkinDeform();
```

2.3. Cloth deformation library (Hybrid approach)

The cloth deformation library has been developed on top of the skinning library, for the simple reason that they share most of the data structures. This library has been implemented based on the research work described in Chapter IV. The API of the library is composed of two methods. One method is used at the initialisation of the library to load the cloth model and initialises its internal data structures. The other method computes the cloth animation for the time step given as a parameter. If no time step is given, it then computes the time since last update using the internal clock of the computer.

2.3.1. List of the header files to include

The files to be included are the same as for the skin deformation library.

2.3.2. List of the functions

The same class is used for cloth and skin deformation. In this section is given the description of the function specific to the cloth deformation.

2.3.2.1. Initialisation

```
bool Init(char *newClothName,bool initLoD=true,int gender=0);
```

- `newClothName`: full name of the cloth file to be loaded with the body.
- `InitLoD`: define whether the LoD will be used with the body.
- `gender`: define the gender of the loaded body. The library integrates the body-sizing module [SEOH 03].

2.3.2.2. Runtime

The deformation of the clothes is computed in the same function as the skin deformation library (see section 2.2.3.2). The library contains two additional functions dedicated to the cloth deformation.

```
void SetSimulationSpeed(float simulationSpeed);
```

- `simulationSpeed`: set the simulation speed of the cloth. This function is rarely used because the default value is sufficient for most of the cases.

The following function reset the shape of the clothes. This function is usefull when the clothes is not any more on the body after a jump of the avatar body.

```
static bool resetCloths(void);
```

2.3.2.3. Destruction

The following function unloads the clothes.

```
void UnloadCloth();
```

2.4. Cloth deformation library (Example-based approach)

The second cloth deformation library is the implementation of the research work described in Chapter V. Like the two other libraries, the interface is very simple. It is composed of two methods, one for the initialisation of the library and the other one for the runtime execution.

2.4.1. List of the header files to include

The use of this library requires including two files: "RC2Cloth.h" which is the main header file and "Vect3.h". Unlike the first cloth library, the functions have been put to a separate file. This is because this cloth deformation method does not make use of the skin deformation.

2.4.2. List of the functions

This section gives the description of the functions.

2.4.2.1. Initialisation

```
bool Init(char *newClothName,CsClothSkinDeform* theSkin);
```

- `newClothName`: full name of the cloth file to be loaded with the body.
- `theSkin`: the instance of the class "CsClothSkinDeform".

2.4.2.2. Runtime functions

```
void Deform(float timeStep=-1.0f);
```

- `timeStep`: this variable contains the elapsed time since previous frame.

```
void ResetPosAndSpeed();
```

The following function reset the shape of the clothes.

2.4.2.3. Destruction

The destructor of the class is:

```
~RC2Cloth();
```

Appendix B. User Manual

In this appendix is given the content of the user manual. In the three first sections, the tools for motion editing, skinning and cloth pre-processing are described. The last section summarizes all steps to prepare virtual human models from motion capture to real-time animation.

All the tools have been compiled into one plug-in file "BodyManager.dll". This file has to be installed the plug-in folder of 3D Studio Max. Most of tools are available through the utility plug-in named "BodyManager".

1. Motion editing tools

Motion editing tools are the tools to create, modify and export body motion data for the real-time platform "VHD++" [PAPA 03].

1.1. Edition of skeleton poses

This module contains various functions for manipulating skeleton poses. The H-Anim skeleton has to be selected first.

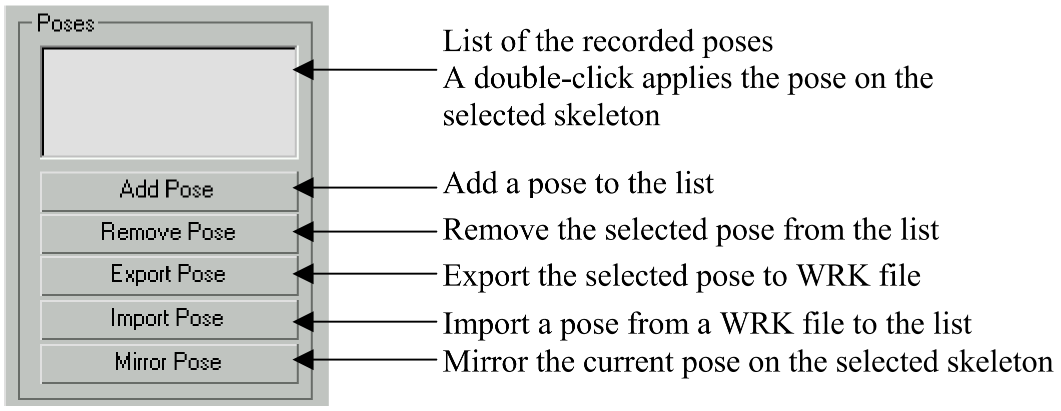
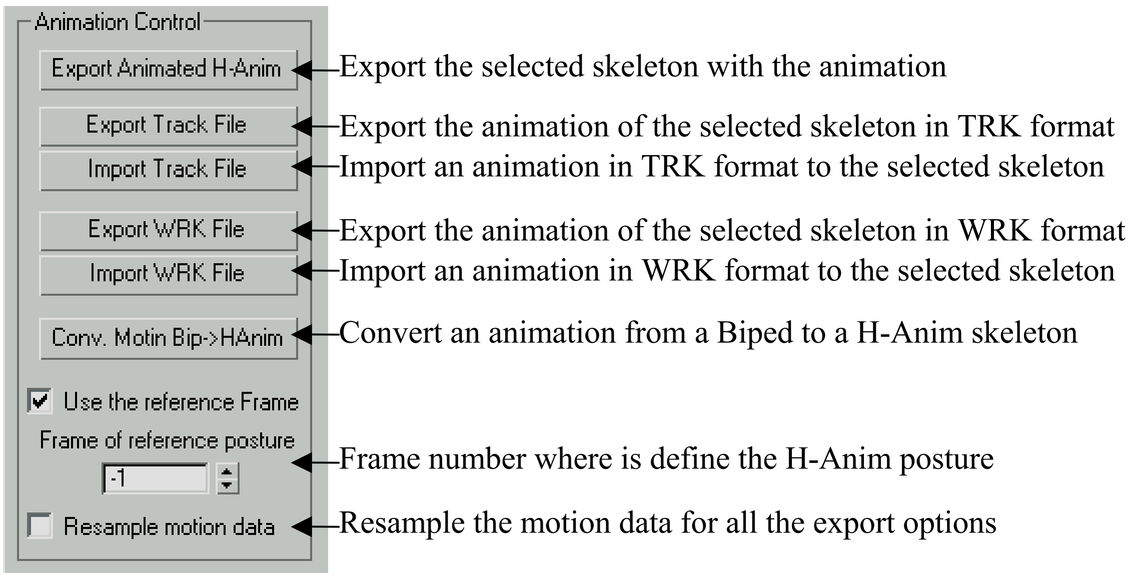

Poses
Add Pose
Remove Pose
Export Pose
Import Pose
Mirror Pose
List of the recorded poses
A double-click applies the pose on the selected skeleton
Add a pose to the list
Remove the selected pose from the list
Export the selected pose to WRK file
Import a pose from a WRK file to the list
Mirror the current pose on the selected skeleton

Figure B-1: User Interface for pose edition

1.2. Edition of motion data

This module contains all the functions to export/import motion data to/from TRK and WRL format. This module provides as well a function to transfer motion data from Biped to H-Anim skeleton.

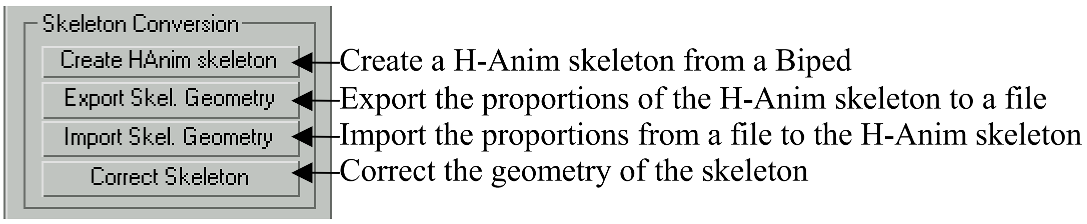


Export the selected skeleton with the animation
Export the animation of the selected skeleton in TRK format
Import an animation in TRK format to the selected skeleton
Export the animation of the selected skeleton in WRK format
Import an animation in WRK format to the selected skeleton
Convert an animation from a Biped to a H-Anim skeleton
Frame number where is define the H-Anim posture
Resample the motion data for all the export options

Figure B-2: User Interface for motion editing

1.3. Skeleton conversion

This module is dedicated to the conversion of the Biped skeleton to H-Anim skeleton.



Create a H-Anim skeleton from a Biped
Export the proportions of the H-Anim skeleton to a file
Import the proportions from a file to the H-Anim skeleton
Correct the geometry of the skeleton

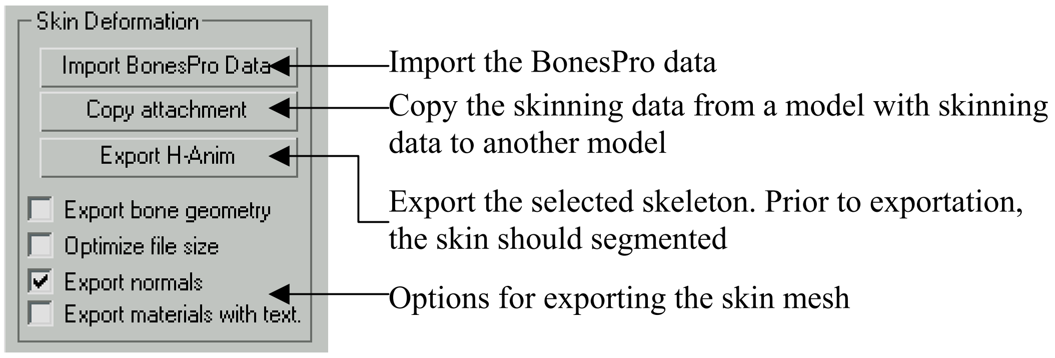
Figure B-3: User Interface of the skeleton conversion

2. Skinning

The skinning tools help CG artists to create and export virtual human models to the real-time platform "VHD++".

2.1. Import/Copy skinning data

The skinning data are first generated with BonesPro [DIGI 03]. These data are then exported and imported using the UI described below. Prior to any operation, the root of the H-Anim skeleton [HANI 03] needs to be selected.


Skin Deformation
Import BonesPro Data
Copy attachment
Export H-Anim
Import the BonesPro data
Copy the skinning data from a model with skinning data to another model
Export the selected skeleton. Prior to exportation, the skin should segmented
Options for exporting the skin mesh

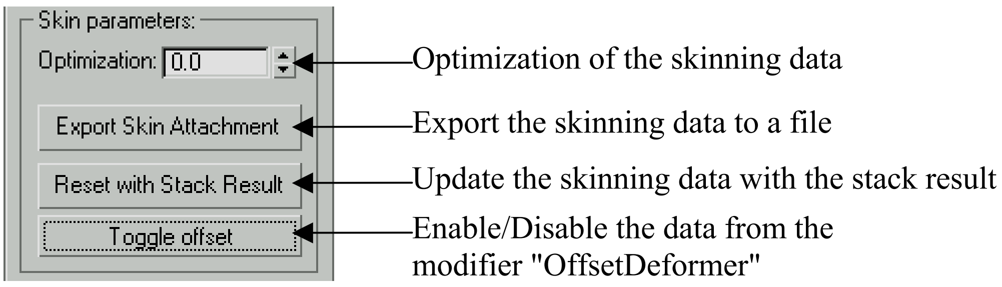
☐ Export bone geometry
☐ Optimize file size
☑ Export normals
☐ Export materials with text.

Figure B-4: User Interface for the manipulation of skinning data

After the importation of the BonesPro data, a modifier "FastSkinMod" is added on top of the skin node. This modifier provides additional features for the skinning data. They are described in the next section.

2.2. Processing of the skinning data

This user interface is part the modifier "FastSkinMod".


Skin parameters:
Export Skin Attachment
Reset with Stack Result
Toggle offset
Optimization of the skinning data
Export the skinning data to a file
Update the skinning data with the stack result
Enable/Disable the data from the modifier "OffsetDeformer"

Optimization: 0.0

Figure B-5: User Interface for processing of the skinning data

2.3. Segmentation of the body

This user interface is part of the modifier "FastSkinMod".

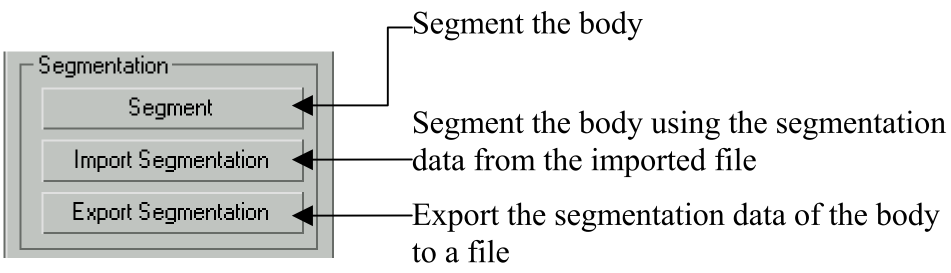

Segmentation
Segment
Import Segmentation
Export Segmentation
Segment the body
Segment the body using the segmentation data from the imported file
Export the segmentation data of the body to a file

Figure B-6: User Interface for body segmentation

The bodies cannot be exported before they have been segmented. The segmentation assigns each vertex to its most influencing joint.

The plug-in offers the possibility for the user to hide some of the triangles of the skin mesh. This feature is useful for reducing the number of polygons that are shown in the rendering window of VHD++. Note that the change of the number of rendered triangles does not modify the number of vertices of the model. Other VHD++ modules that work on the list of vertices such as facial animation can be used without any modification. The process of hiding the triangles is done using the triangle-editing mode. With this mode, CG artists can select triangles and specify if they will be visible or not in the rendering window (Figure B-7).

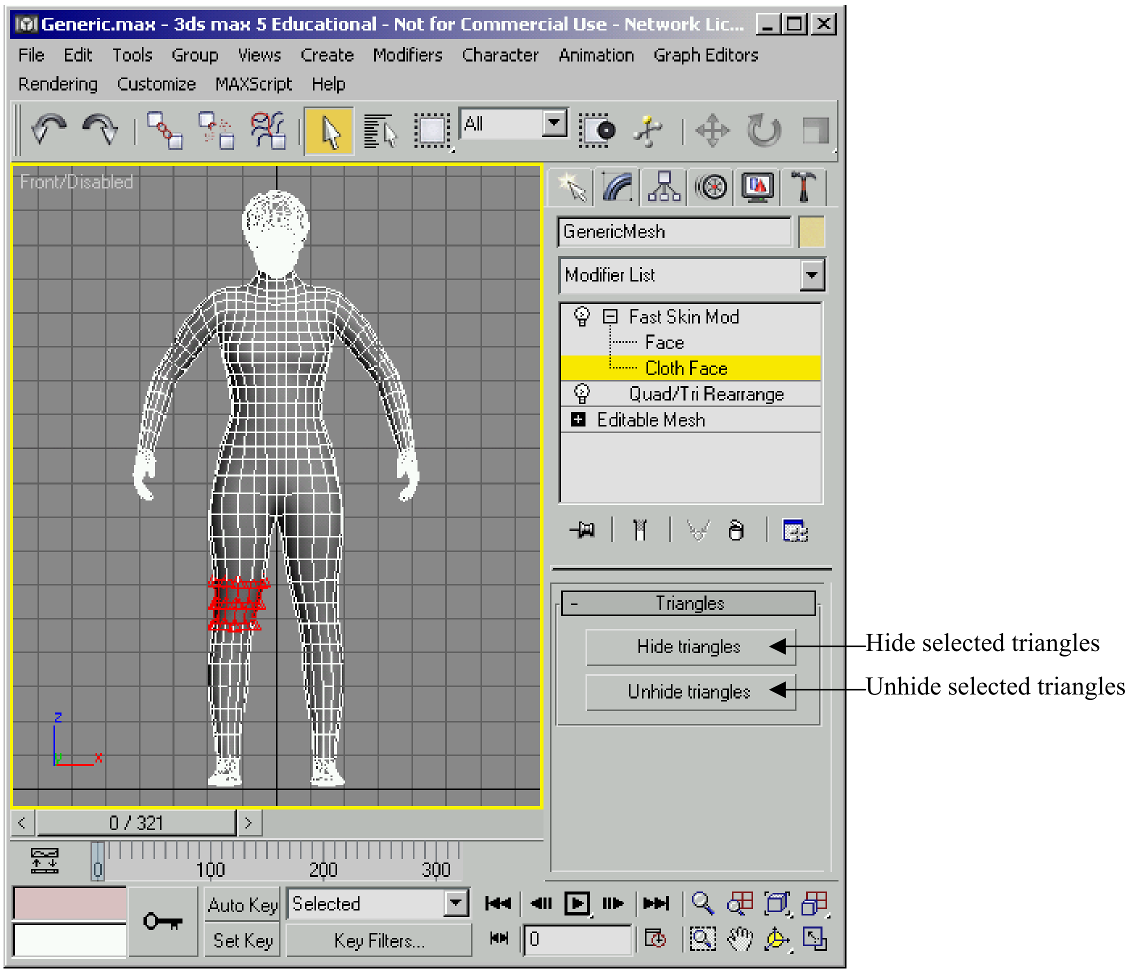

Generic.max - 3ds max 5 Educational - Not for Commercial Use - Network Lic...
File Edit Tools Group Views Create Modifiers Character Animation Graph Editors Rendering Customize MAXScript Help
Front/Disabled
GenericMesh
Modifier List
Fast Skin Mod
Face
Cloth Face
Quad/Tri Rearrange
Editable Mesh
Triangles
Hide triangles
Unhide triangles
Hide selected triangles
Unhide selected triangles
0 / 321
Auto Key
Selected
Set Key
Key Filters...

Figure B-7: Hide/unhide the triangles of the skin mesh

2.4. Skeleton driven deformation with matrix blending

This tool implements the skeleton driven deformation described in Chapter III.2.2. This user interface is part the modifier "FastSkinMod".

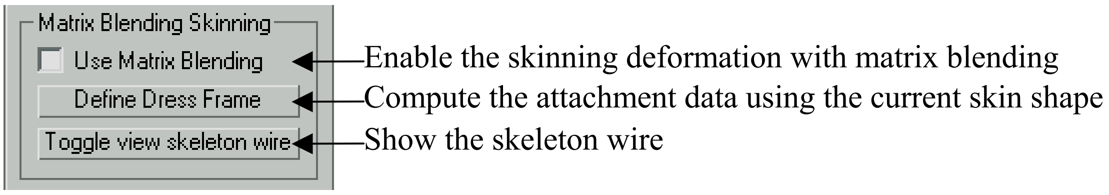

Matrix Blending Skinning
Use Matrix Blending
Define Dress Frame
Toggle view skeleton wire
Enable the skinning deformation with matrix blending
Compute the attachment data using the current skin shape
Show the skeleton wire

Figure B-8: Skeleton driven deformation with matrix blending

3. Pre-processing of the cloth

The creation of clothes for real-time requires having a model of the clothes already fitted to the body. If the clothes have to be animated with the example-based approach, the user has to provide in addition a pre-calculated animation of the clothes together with the skeleton animation. The pre-processing functions are accessible through the utility plug-in "BodyManager". After the completion of the cloth pre-processing, the resulting data are kept in a dedicated modifier that is put on top of the cloth node stack. By selecting this modifier, the user has access to the parameters of the real-time cloth simulation. These parameters can be modified to fine tune the cloth simulation.

3.1. Hybrid approach

This module is the implementation of the method presented in Chapter IV. It is composed of to user interface. The execution of the pre-processing is accessible through the utility plug-in. The modification of the simulation parameters has to be done through the modifier "Cloth Pre-processing M1".

3.1.1. Starting of the pre-processing

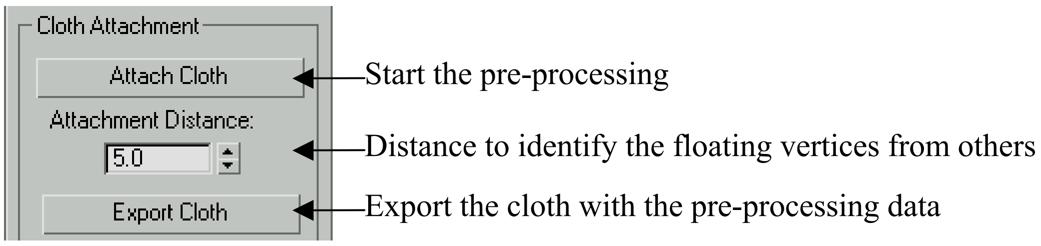

Cloth Attachment
Attach Cloth
Attachment Distance:
5.0
Export Cloth
Start the pre-processing
Distance to identify the floating vertices from others
Export the cloth with the pre-processing data

Figure B-9: User Interface for the cloth pre-processing (Hybrid approach)

3.1.2. Modification of the simulation parameters

The modification of the simulation parameters is made through the modifier "Cloth Pre-processing M1". The user interface of this modifier is composed of three panels. The first panel gives the number of triangles belonging to the floating cloth regions.

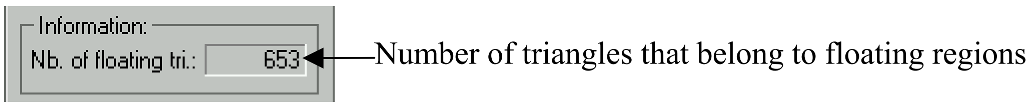
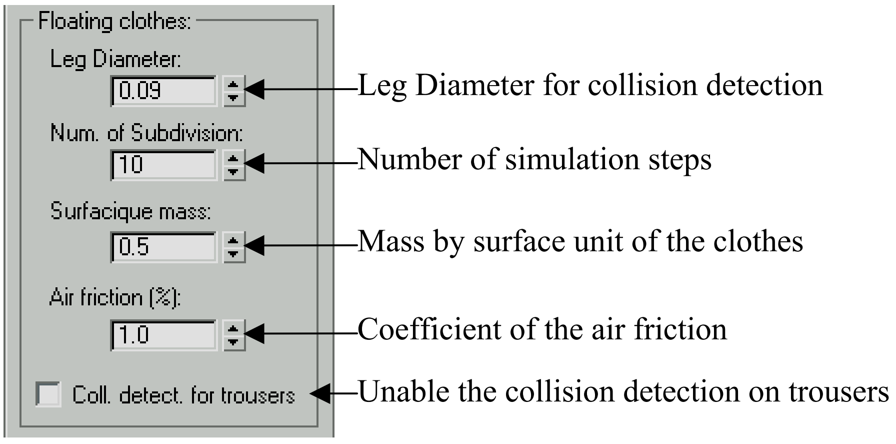

Information:
Nb. of floating tri.: 653
Number of triangles that belong to floating regions

Figure B-10: User Interface to display the number of triangles (Hybrid approach)

The next panel is dedicated to the modification of the simulation parameters of the floating cloth regions (see Chapter IV.5).


Floating clothes:

Leg Diameter: 0.09 — Leg Diameter for collision detection

Num. of Subdivision: 10 — Number of simulation steps

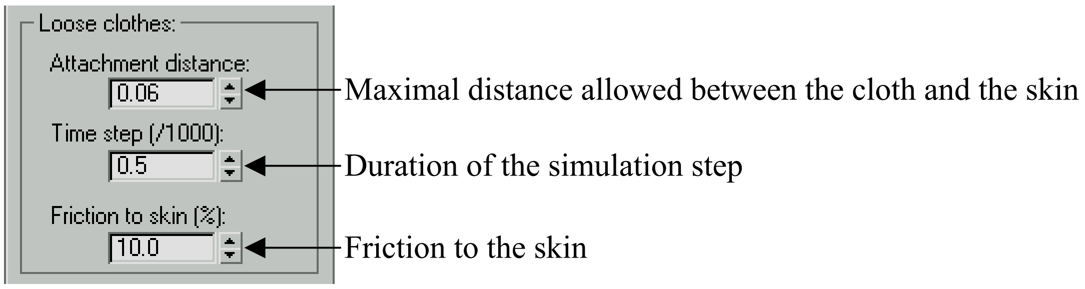
Surfacique mass: 0.5 — Mass by surface unit of the clothes

Air friction (%): 1.0 — Coefficient of the air friction

Coll. detect. for trousers — Unable the collision detection on trousers

Figure B-11: Simulation parameters of the floating clothes (Hybrid approach)

The following panel allows users to modify the simulation parameters of loose clothes (see Chapter IV.4).


Loose clothes:

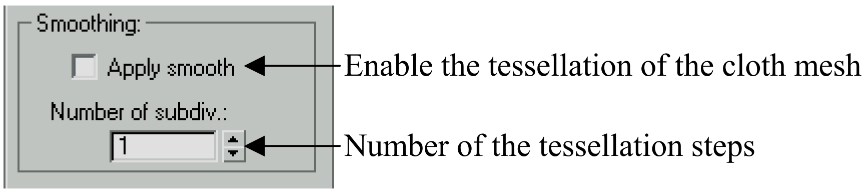
Attachment distance: 0.06 — Maximal distance allowed between the cloth and the skin

Time step (/1000): 0.5 — Duration of the simulation step

Friction to skin (%): 10.0 — Friction to the skin

Figure B-12: Simulation parameters of the loose clothes (Hybrid approach)

The last panel contains the parameters controlling the tessellation of the cloth surface (see Chapter IV.7).

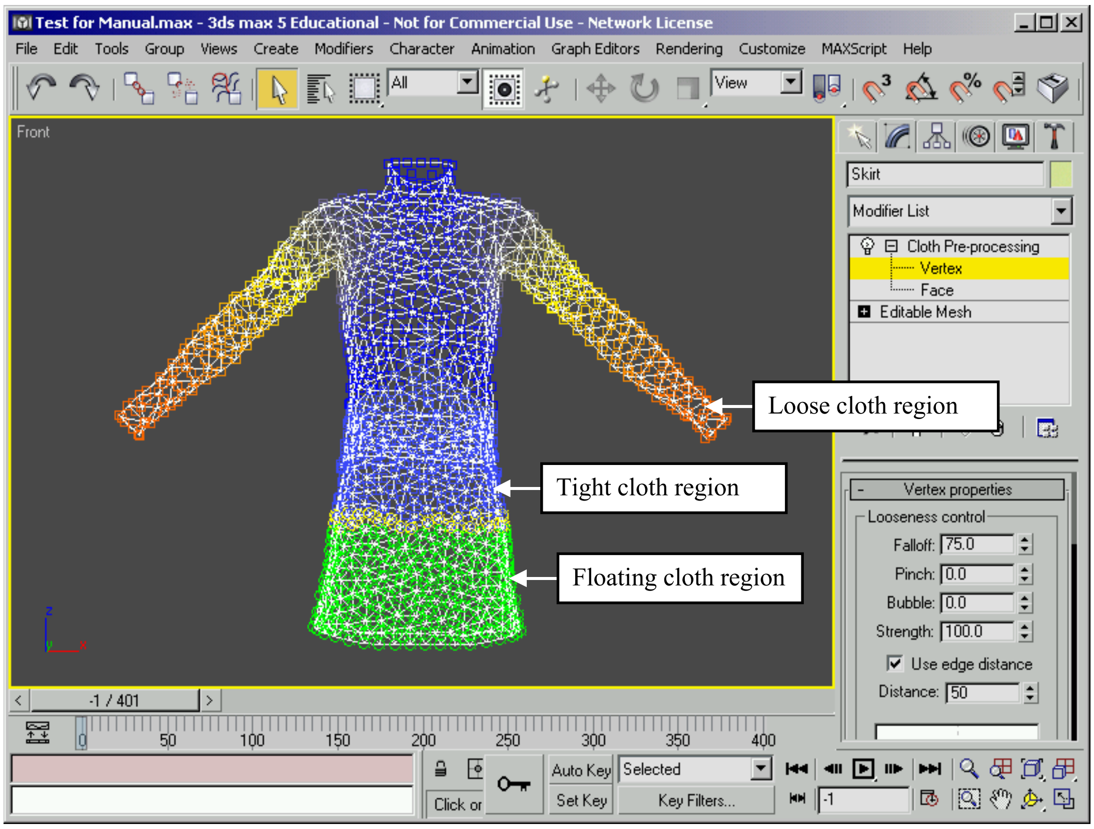

Smoothing:

Apply smooth — Enable the tessellation of the cloth mesh

Number of subdiv.: 1 — Number of the tessellation steps

Figure B-13: Tessellation of the cloth mesh (Hybrid approach)

In addition to the three panels to control the global simulation parameters, the modifier offers two special editing modes in which the user can select the vertices or faces of the cloth mesh and specify their properties.


Test for Manual.max - 3ds max 5 Educational - Not for Commercial Use - Network License
File Edit Tools Group Views Create Modifiers Character Animation Graph Editors Rendering Customize MAXScript Help
Front
Loose cloth region
Tight cloth region
Floating cloth region
-1 / 401
Auto Key
Set Key
Key Filters...
Click or

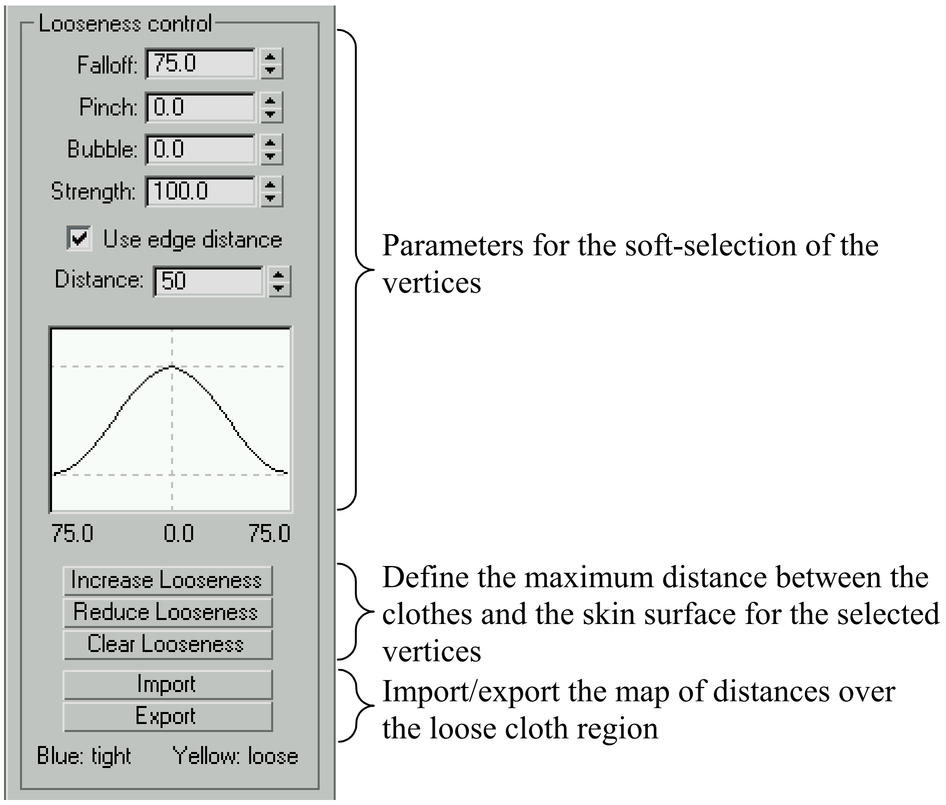
Skirt
Modifier List
Cloth Pre-processing
Vertex
Face
Editable Mesh

Vertex properties
Looseness control
Falloff: 75.0
Pinch: 0.0
Bubble: 0.0
Strength: 100.0
[✓] Use edge distance
Distance: 50

Selected

Figure B-14: User interface of the vertex selection mode (Hybrid approach)

With the vertex-selection mode, the user can modify the local simulation parameters of the loose cloth region. The first panel enables the user to control for each vertex the maximum distance possible between the clothes and the skin surface (Chapter IV.4).



Looseness control
Falloff: 75.0
Pinch: 0.0
Bubble: 0.0
Strength: 100.0
☑ Use edge distance
Distance: 50
75.0 0.0 75.0
Increase Looseness
Reduce Looseness
Clear Looseness
Import
Export
Blue: tight Yellow: loose

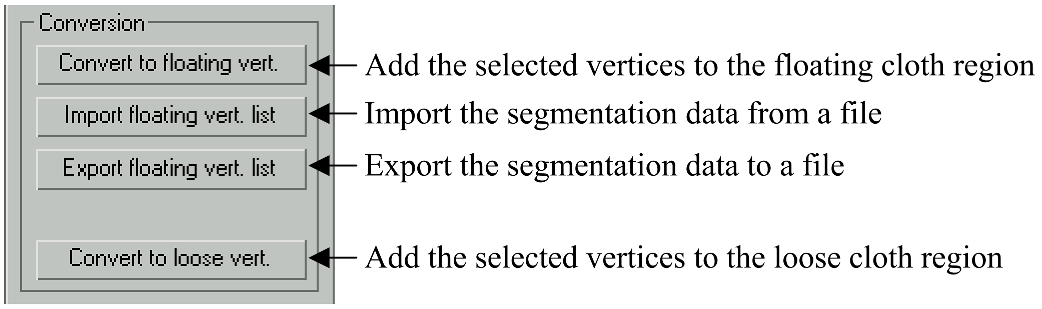
Parameters for the soft-selection of the vertices

Define the maximum distance between the clothes and the skin surface for the selected vertices

Import/export the map of distances over the loose cloth region

Figure B-15: Modification of the maximum distance between the loose cloth region and the skin surface (Hybrid approach)

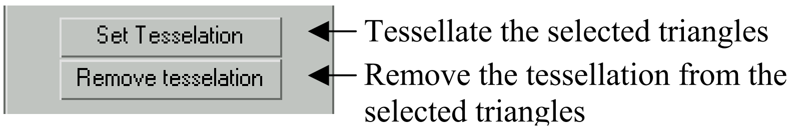
The other panel of the vertex selection mode enables the user to modify the segmentation of the clothes (Chapter IV.2).



Conversion
Convert to floating vert. ← Add the selected vertices to the floating cloth region
Import floating vert. list ← Import the segmentation data from a file
Export floating vert. list ← Export the segmentation data to a file
Convert to loose vert. ← Add the selected vertices to the loose cloth region

Figure B-16: User interface to modify the cloth segmentation (Hybrid approach)

Using the face selection mode, the user can select a group of triangles and specify if they should be tessellated or not (Chapter IV.7).



Set Tesselation ← Tessellate the selected triangles
Remove tesselation ← Remove the tessellation from the selected triangles

Figure B-17: The panel to specify the triangles to be tessellated (Hybrid approach)

3.2. Example-based approach

This module is the implementation of the example-based method described in Chapter V. Like the first cloth simulation method, this module is composed of two user interfaces. The execution of the pre-processing should be done from the utility plug-in. After the pre-processing has finished, the user can modify the simulation parameters through the modifier "Cloth Pre-processing M2".

3.2.1. Starting of the pre-processing

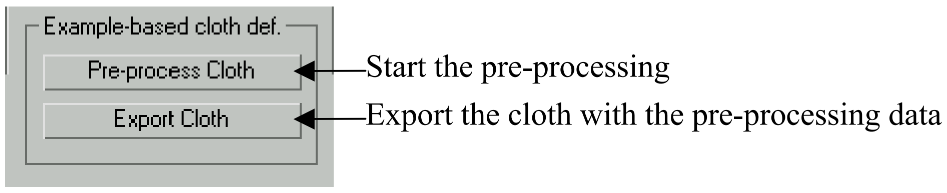


Figure B-18: User Interface for the cloth pre-processing (Example-based approach)

3.2.2. Modification of the simulation parameters

The following panel is part of the user interface of the modifier "Cloth Pre-processing M2". This panel enables the user to display the results of the pre-processing: the surface segmentation (see Chapter V.2.1) and the collision hulls (see Chapter V.2.2.3).

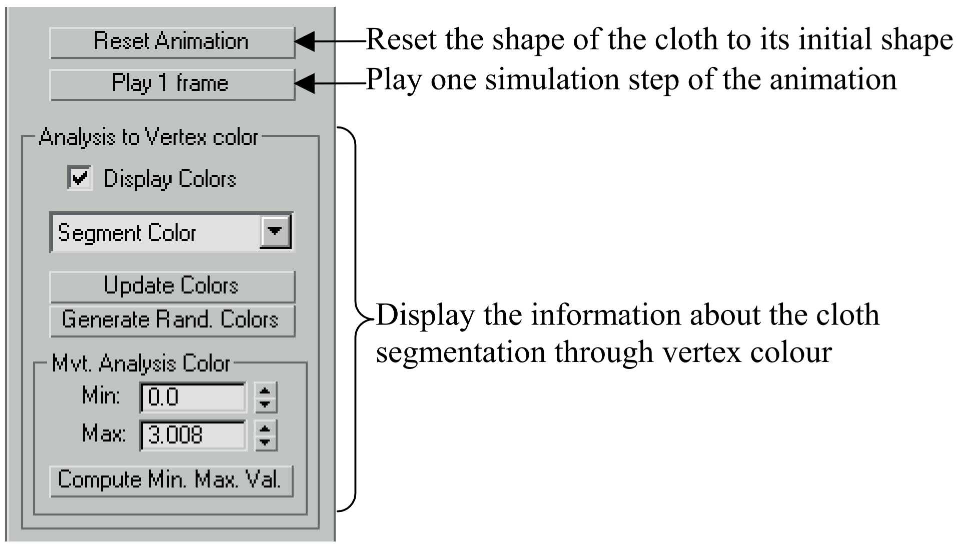


Figure B-19: User Interface to modify the simulation parameters (Example-based approach)

4. Description of the steps to create and animate virtual human models

This section gives an overview of the typical steps that CG artists have to follow for the creation and animation of virtual human models with clothes.

4.1. Preparation of the body model

The first stage of creating virtual human models is the design of the skin surface and the skeleton. Body models can be generated automatically from measurements [SEOK 03] or by hand. If created by hand, the CG artist has to:

- Create the skeleton hierarchy with Character Studio.
- Design the skin surface with the mesh editing tools in 3D Studio Max [3DSM 03]. The skin surface should match the skeleton hierarchy.
- Attach the skin surface to the skeleton using the BonesPro plug-in [DIGI 03]. At this step, the CG artist should have a completely functional human model inside 3D Studio Max environment. The next steps consist to prepare the model for exportation.
- Create the H-Anim skeleton from the Biped (see section 1.3).
- Clone and collapse the node containing the skin mesh.
- Export the skin assignment from BonesPro to a file. This file is then imported to attach the new skin node to the H-Anim skeleton [HANI 03].
- At this stage, the motion data and the clothes should be prepared (see section 4.2 and 4.3).
- The skeleton is segmented (section 2.3) and the body model is exported to a file (section 2.1).

4.2. Motion data

Optical motion capture systems are the best way to obtain realistic motion data of the human body. Vicon [VICO 03] is our main tool for the motion capture. The steps that CG artists have to follow are:

- Capture of the body movements of a real person. Markers are placed on the body of the person. The Vicon system tracks their movements and writes their trajectories into a CSM (Character Studio Motion) file [3DSM 03].
- The CSM file is then imported to 3D Studio Max using the plug-in Character Studio. This plug-in is able to retarget the motion data to the dimension of the Biped [3DSM 03] skeleton.

- The motion data is converted into the H-Anim format using the plug-in BodyManager (section 1.2).
- At this stage, CG artists may design the clothes and compute their animation for the converted body motion.
- The H-Anim body motion data can be exported to file "WRL" which can be used to animate the virtual humans on the platform "VHD++".

4.3. Preparation of the clothes

The design and the preparation of the clothes come at last. The drawing and the fitting of the patterns to the body shape are made with Fashionizer [VOLI 00]. The cloth animation is then imported to 3D Studio Max. The next steps are:

- The cloth shape is attached to the skin shape (section 3.1.1 and 3.2.1).
- The CG artist may modify the simulation parameters of the clothes (section 3.1.2 and 3.2.2). The artist can as well modify the triangles on the skin surface that are hidden when loading the cloth (section).
- The cloth model is exported to a file (section 3.1.1 and 3.2.1).

Appendix C. Publications

Following are the main publications that are closely related to the thesis work

- P. Volino, F. Cordier, N. Magnenat-Thalmann, "From Early Virtual Garment Simulation to Interactive Fashion Design", Computer-Aided Design, special issue on CAD Methods in Garment Design, published by Elsevier, in press, 2003.
- Cordier F., Seo. H., Magnenat-Thalmann N., (2003), "Made-to-Measure Technologies for Online Clothing Store", IEEE CG&A special issue on "Web Graphics", published by IEEE Computer Society, pp. 38-48.
- N. Magnenat-Thalmann, H. Seo, F. Cordier, "Automatic Modeling of Virtual Humans and Body clothing", Proc. 3-D Digital Imaging and Modeling, published by IEEE Computer Society Press, October, 2003.
- Papagiannakis G., Ponder M., Molet T., Kshirsagar S., Cordier F., Magnenat-Thalmann N., Thalmann D. "LIFEPLUS: Revival of life in ancient Pompeii", Virtual Systems and Multimedia, VSMM2002-invited paper, published by IOS Press and OHMSHA publishers, October, 2002.
- Magnenat-Thalmann N., Volino P., Cordier F., "Avenues of Research in Dynamic Clothing", Computer Animation 2002 proceedings, published by IEEE Computer Society, pp 193-202, 2002.
- Cordier F., Magnenat-Thalmann N., (2002), "Real-time Animation of Dressed Virtual Humans", Eurographics, published by Blackwell Publishers, September 2002.

Appendix D. Other Publications

Following are the other publications that have been published during the PhD thesis:

- Seo H., Cordier F., Magnenat-Thalmann N., "Synthesizing Animatable Body Models with Parameterized Shape Modifications", ACM SIGGRAPH/Eurographics Symposium on Computer Animation, published by ACM Siggraph, July 2003.
- Cordier F., Volino P., Magnenat-Thalmann N., "Integrating Deformations between Bodies and Clothes", Journal on Visualization and Computer Animation, published by John Wiley & Sons, volume 12, pp. 45-53, 2001.
- Cordier F., Magnenat-Thalmann N., "Integrated system for skin deformation", Proceedings of Computer Animation 2000, published by IEEE Computer Society, pp. 2–8, 2000.
- Seo H., Cordier F., Philippon L., Magnenat-Thalmann N. "Interactive Modelling of MPEG-4 Deformable Human Body Models", Deform 2000 proceedings, published by Kluwer Academic Publishers, pp. 120–131, 2000.
- Magnenat-Thalmann N., Cordier F. "Construction of a human topological model from medical data", Information Technology in Biomedicine, IEEE Transactions on Information Technology in Biomedicine, published by the IEEE Engineering in Medicine and Biology Society and the IEEE Computer Society, Volume: 4 Issue: 2, pp. 137 –143, June 2000.
- Kim J., Cordier F., Magnenat-Thalmann N., "Neural network-based violinist's hand animation", Computer Graphics International, published by the IEEE Computer Society, pp. 37 –41, 2000.
- Cordier F. and Magnenat-Thalmann N., "Comparison of two techniques for organ reconstruction using visible human dataset", in The Visible Human Project Conference Proceedings, edited by Richard A. Banvard, CD-ROM, 1998.

Appendix E. Bibliography

[3DSM 03] 3D Studio Max, http://www.discreet.com/products/3dsmax/

[AGUI 90] T. Agui, Y. Nagao, and M. Nakajma, "An Expression Method of Cylindrical Cloth Objects-An Expression of Folds of a sleeve using Computer Graphics", The Transaction of The Institute of Electronics, Information and Communication, published by Oxford University Press on behalf of IEICE, Col. J73-D-II, No. 7, pp. 1095-1097, 1990.

[ALEN 02] B. Allen, B. Curless, Z. Popovic, "Articulated body deformation from range scan data", Proceedings SIGGRAPH 2002, published by ACM SIGGRAPH Addison Wesley, pp. 612-619, 2002.

[ALEX 02] M. Alexa, "Linear Combination of Transformations", SIGGRAPH 2002 Conference Proceedings, Annual Conference Series, published by ACM SIGGRAPH Addison Wesley, 21(3), pp. 380-387, July 2002

[AONO 90] M. Aono, "A wrinkle propagation model for cloth", In Proceeding 8th International Conf. of the Computer Graphics Society on CG International '90, published by IEEE press, pp. 95-115, 1990.

[ARIK 03] O. Arikan, D. Forstyh, and J. O'Brien, "Motion synthesis from annotations", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, 22(3), pp. 402-408, July 2003.

[ASCH 95] U. M. Ascher, S. J. Ruuth, and B. T. Wetton, "Implicit-explicit methods for time dependent pde's, SIAM Journal on Numerical Analysis, published by Society for Industrial and Applied Mathematics, 32 (3), pp. 797-823, June 1995.

[ASCH 97] U. M. Ascher, S. J. Ruuth, and R. J. Spiteri, "Implicit-explicit runge-kutta methods for time-dependent partial differential equations", Applied Numerical Mathematics, published by Elsevier Science, 25, pp. 151-167, October 1997.

[ATCA 03] Athens Technology Center, http://www.atc.gr/e-tailor/

[BACI 02] G. Baciu and S.K. Wong, "Image-based techniques in a hybrid collision detector", IEEE Transaction on Visualization and Computer Graphics, published by IEEE Press, 2002.

[BAEH 02] H.J. Bae, K.W. Ryu, B.T. Jang, "Procedural Approach to generate Real Time Motions of Cloth", proceedings of Shape Modeling International 2002 (SMI'02), published by IEEE Computer Society, Banff, Canada, May 2002.

[BARA 03] D. Baraff, A. Witkin, "Untangling Cloth", ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 2003, published by ACM SIGGRAPH Addison Wesley, 22(3), pp. 862-870, 2003.

[BARA 98] D. Baraff and A. Witkin, "Large steps in cloth simulation", ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 1998, published by ACM SIGGRAPH Addison Wesley, pp. 43-54, 1998

[BARB 96] C. B. Barber, D.P. Dobkin, and H.T. Huhdanpaa, "The Quickhull Algorithm for Convex Hulls", ACM Transactions on Mathematical Software, published by ACM Press, 22(4), pp. 469-483, 1996.

[BECK 03] B. Beck, "Key Strategic Issues in Online Apparel Retailing -The Need For An Online Fitting Solution", http://www.techexchange.com/thelibrary/online_fit.html.

[BICA 00] A. Bicalho and S. Feltman, "Mastering MAXScript & the SDK for 3D Studio MAX", published by SYBEX, 2000.

[BLOO 97] J. Bloomenthal (Ed.), "Introduction to Implicit Surfaces", published by Morgan Kaufmann Publishers, ISBN: 155860233X, August 1997.

[BOLZ 03] J. Bolz, I. Farmer, E. Grinspun, P. Schröder, "Sparse Matrix Solvers on the GPU: Conjugate Gradients and Multigrid", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, 22(3), pp. 917-924, July 2003.

[BREE 94] D.E. Breen, D.H. House, M.J. Wozny, "Predicting the Drape of Woven Cloth Using Interacting Particles", SIGGRAPH Conference Proceedings, published by ACM SIGGRAPH Addison Wesley, pp 365-372, July 1994.

[BRID 02] R. Bridson, R. Fedkiw, J. Anderson, "Robust Treatment of Collisions, Contact, and Friction for Cloth Animation", ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 2002, published by ACM SIGGRAPH Addison Wesley, 21, pp. 594-603, 2002.

[BRID 03] R. Bridson, S. Marino, R. Fedkiw, " Simulation of Clothing with Folds and Wrinkles", ACM SIGGRAPH/Eurographics Symposium on Computer Animation, published by ACM Press, pp. 28-36, July 2003.

[BURD 93] R. L. Burden, J. D. Faires, "Numerical Analysis, Fifth Edition", published by PWS Publishing, ISBN 0-534-93219-3, 1993.

[CAME 97] S. Cameron. "Enhancing GJK: Computing minimum and penetration distances between convex polyhedra", International Conference on Robotics and Automation (ICRA '97), published by IEEE Computer Society Press, pp. 3112-3117, 1997.

[CARI 92] M. Carignan, Y. Yang, N. Magnenat-Thalmann, D. Thalmann, "Dressing Animated Synthetic Actors with Complex Deformable Clothes", Computer Graphics (SIGGRAPH'92 proceedings), published by Addison-Wesley, 26(2), pp 99-104, 1992.

[CGSH 03] CG Shaders, http://www.cgshaders.org/.

[CHAR 03] Character Studio, http://www.discreet.com/products/cs/.

[CHOI 02] K.-J. Choi, H.-S. Ko, "Stable but Responsive Cloth", ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 2002, published by ACM SIGGRAPH Addison Wesley, 21, pp. 165-172, 2002.

[CORD 02] F. Cordier, N. Magnenat-Thalmann, (2002), "Real-time Animation of Dressed Virtual Humans", Computer Graphics Forum, published by Blackwell Publishers, 21(3), September 2002.

[CORD 03] F. Cordier, H. Seo, N. Magnenat-Thalmann, (2003), "Made-to-Measure Technologies for Online Clothing Store", IEEE CG&A special issue on "Web Graphics", published by IEEE Press, pp. 38-48, Jan. 2003.

[CREY 03] Cloth Reyes from Reyes Inforgrafica, http://www.reyes-infografica.net/.

[CYBE 03] 3D scanning system Cyberware, http://www.cyberware.com/products/.

[DEBU 00] G. Debunne, M. Desbrun, M.-P. Cani, and A. Barr, "Adaptive simulation of soft bodies in real-time" proceedings of Computer Animation 2000, published by IEEE Press, pp. 133-144, May 2000.

[DEBU 01] G. Debunne, M. Desbrun, M.-P. Cani, A. Barr, "Dynamic Real-Time Deformations using Space and Time Adaptive Sampling", ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 2002, published by ACM SIGGRAPH Addison Wesley, 21, pp. 31-36, 2002.

[DESB 99] M. Desbrun, P. Schröder, and A. H. Barr, "Interactive Animation of Structured Deformable Objects", In Graphics Interface'99 proceedings, published by Morgan Kaufmann, pp. 1-8, June 1999.

[DIGI 03] Digimation, BonesPro, a plug-in to 3DS Max, http://www.digimation.com/.

[DREA 02] PDI/Dreamwroks, http://www.dreamworks.com/.

[EBER 00] B. Eberhardt, O. Etzmuss, M. Hauth, "Implicit-Explicit Schemes for Fast Animation with Particles Systems", Proceedings of the Eurographics workshop on Computer Animation and Simulation, published by Springer-Verlag, pp 137-151, 2000.

[EBER 02] A. Ebert, I. Ginkel, H. Barthel, A. Divivier, M. Bender, "Efficient Assistance Of Virtual Dress Fitting Using Intelligent Morphing", IASTED International Conference on Visualization, Imaging and Image Processing (VIIP 2002), published by ACTA Press, Malaga, Spain, 2002.

[EBER 96] B. Eberhardt, A. Weber, W. Strasser, "A Fast Flexible Particle-System Model for Cloth Draping", Computer Graphics in Textiles and Apparel (IEEE Computer Graphics and Applications), published by IEEE Press, pp 52-59, Sept.1996.

[EIDO 03] 'Tomb Raider' from Eidos, http://www.tombraider.com/.

[ETAI 99] E-Tailor project, http://www.atc.gr/e-tailor/, IST-1999-10549.

[FUHR 03] A. Fuhrmann, C. Gross, V. Luckas, "Interactive Animation of Cloth Including Self-Collision Detection", Journal of WSCG, ISSN 1213-6964, 11 (1) pp 141-148, 2003.

[GARL 97] Michael Garland, Paul S. Heckbert, "Surface simplification using quadric error metrics", SIGGRAPH 97, Conference Proceedings, Annual Conference Series, ACM SIGGRAPH, published by Addison Wesley, pp. 209-216, 1997.

[GILB 88] E. G. Gilbert, D.W. Johnson, and S. S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space", IEEE Journal of Robotics and Automation, published by IEEE Press, 4(2), pp. 193-203, 1988.

[GOME 98] J. Gomes, L. Darsa, B. Costa, L. Velho, "Warping and Morphing of Graphical Objects, published by Morgan Kaufmann Publishers", 1998.

[GOVI 03] N. Govindaraju, S. Redon, M. C. Lin, D. Manocha, "CULLIDE: interactive collision detection between complex models in large environments using graphics hardware", Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware, published by ACM Press, pp. 25-32, 2003.

[GROB 02] Groß C., Fuhrmann A., "Automatic Pre-positioning and Physically Based Simulation of Cloth", Computer Graphik, published by E.h. Hon and J. L. Encarnação, Vol. 14, pp. 15-16, 6/2002.

[GRZE 98] R. Grzeszczuk, D. Terzopoulos, and G. Hinton, "Neuroanimator: Fast neural network emulation and control of physics-based models", SIGGRAPH 98 Conference Proceedings, Annual Conference Series, published by ACM SIGGRAPH Addison Wesley, pp. 9-20, July 1998.

[HADA 99] S. Hadap, E. Bangarter, P. Volino, N. Magnenat-Thalmann, "Animating Wrinkles on Clothes", IEEE Visualization '99. San Francisco, USA, published by IEEE Press, pp. 175-182, October 1999.

[HANI 03] H-Anim specification, http://www.h-Anim.org

[HAUT 01] M. Hauth, O. Etzmuss, "A high Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods", Eurographics, published by Blackwell Publishers, Vol. 20(3), pp. 319—328, September 2001

[HAVO 03] Reactor plugin for Discreet developed by Havok, http://www.havok.com/

[HIND 01] H. Hinderer, T. Gurzki, U. Rotter, "A Platform for Fashion Shopping with Individualized Avatars and Personalized Customer Consulting", Proceedings of the World Conference on Mass Customization and Personalization MCPC 2001, published by TUM Research Center on Mass Customization and Customer Integration, 2001.

[HIND 90] B.K. Hinds and J. McCartney, Interactive Garment Design, Visual Computer, published by Springler-Verlag, Vol. 6, 1990, pp. 53-61

[HING 96] N. N. Hing, L. R. Grimsdale, "Computer Graphics Techniques for Modeling Cloth", Computer Graphics in Textile and Apparels, published by IEEE Press, Vol. 16(5), pp. 28-41, September 1996.

[HOHE 02] Hohenstein, Uniform Body Representation, Workpackage Input, E-Tailor project, 2002.

[HOUS 00] D. H. House, D. E. Breen, "Cloth Modeling and Animation", published by A.K. Peters Ltd., ISBN 1-56881-090-3, Jan. 2000.

[HUGE 97] Hugues Hoppe, "View-dependent refinement of progressive meshes", SIGGRAPH 97, Conference Proceedings, Annual Conference Series, ACM SIGGRAPH, published by Addison Wesley, pp. 189-198, 1997.

[JAKO 01] T. Jakobsen, "Advanced character physics - the fysix engine", presented at the Game Developer's Conference 2001, URL http://www.ioi.dk/Homepages/thomasj/publications/gdc2001.htm.

[JAME 03] D. L. James and K. Fatahalian, "Precomputing Interactive Dynamic Deformable Scenes", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, Vol. 22(3), pp. 165-172, July 2003.

[JAME 99] D. James and D. Pai, "Accurate real-time deformable objects". SIGGRAPH 99 Conference Proceedings, Annual Conference Series, published by ACM SIGGRAPH Addison Wesley, pp. 65-72, August 1999

[KACI 03] Z. Kacic-Alesic, M. Nordenstam, D. Bullock, "A practical dynamics system", ACM SIGGRAPH/Eurographics Symposium on Computer Animation, published by ACM Press, pp. 7—16, July 2003.

[KANG 00] Y. M. Kang, J. H. Choi, H. G. Cho, D. H. Lee, C. J. Park, "Real-time Animation Technique for Flexible and Thin Objects", WSCG proceedings, ISSN 1213-6964, pp 322-329, 2000

[KANG 01] Y.-M. Kang, J.-H. Choi, H.-G. Cho, D.-H. Lee, "An efficient animation of wrinkled cloth with approximate implicit integration", The Visual Computer Journal, published by Spinger-Verlag, 2001.

[KANG 02] Y.-M. Kang, H.-G. Cho, "Bilayered Approximate Integration for Rapid and Plausible Animation of Virtual Cloth with Realistic Wrinkles", proceedings of Computer Animation 2002, Geneva, Switzerland, published by IEEE Press, pp. 203, June 2002.

[KANH 00] Y.-M. Kang, J.-H. Choi, H.-G. Cho, C.-J. Park, "Fast and Stable Animation of Cloth with an Approximated Implicit Method", Chan-Jong Park, Proceedings of Computer Graphics International 2000, Geneva, Switzerland, published by IEEE Press, pp. 247-255, June 2000.

[KLOS 98] J. Klosowski, M. Held, J.S.B. Mitchell, H. Sowizral, and K. Zikan, "Efficient collision detection using bounding volume hierarchies of k-dops", IEEE Transaction on Visualization and Computer Graphics, published by IEEE Press, Vol. 4(1), pp. 21-37, 1998.

[KRÜG 03] J. Krüger and R. Westermann, "Linear Algebra Operators for GPU Implementation of Numerical Algorithms", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, Vol. 22(3), pp.908-916, 2003.

[KRYP 02] P. G. Kry, D. L. James, and D. K. Pai, "EigenSkin: Real Time Large Deformation Character Skinning in Graphics Hardware", ACM SIGGRAPH Symposium on Computer Animation, published by ACM Press, pp.153-159, 2002.

[KSHI 03] S. Kshirsagar, N. Magnenat-Thalmann, "Visyllable Based Speech Animation", Proceedings of Eurographics, Granada, Spain, published by Blackwell Publishers, vol. 22(3), 2003.

[LAFL 91] B. Lafleur, N. Magnenat-Thalmann, D. Thalmann, "Cloth Animation with Self-Collision Detection", IFIP conference on Modeling in Computer Graphics proceedings, published by Springer-Verlag, pp. 179-197, 1991.

[LAND 98] J. Lander, "Skin Them Bones: Game programming for the Web Generation", proceedings of Game Developer Magazine, published by CMP Media LLC, pp 11-16, May 1998.

[LEWI 00] J. P. Lewis, M. Cordner, N. Fong, "Pose Space Deformations: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, pp. 165-172, 2000.

[MACY 03] Macy's Passport 99 Fashion Show http://www.shoutinteractive.com/Fashion/index.html

[MAGN 88] N. Magnenat-Thalmann, R. Laperrière and Daniel Thalmann, "Joint-Dependent Local Deformations for Hand Animation and Object Grasping", Proceedings of Graphics Interface 1988, published by A K Peters, pp.26-33, 1988.

[MAYA 03] Maya Cloth from Alias, http://www.aliaswavefront.com/

[MCNE 99] W. A. McNeely, K. D. Puterbaugh, J. J. Troy, "Six degree-of-freedom haptic rendering using voxel sampling", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, pp. 401-408, August 1999.

[MELI 02] P. Melis, "Real-Time Cloth Simulation in a 3D Virtual Environment", http://wwwhome.cs.utwente.nl/~melis/clothsim.pdf, August 2002.

[MEYE 00] M. Meyer, G. Debunne, M. Desbrun, A. H. Barr. "Interactive Animation of Cloth-like Objects in Virtual Reality". Journal of Visualization and Computer Animation, published by John Wiley & Sons, 2000.

[MOHR 03] A. Mohr, M. Gleicher, "Building Efficient, Accurate Character Skins from Examples", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, 22(3), pp. 165-172, 2003.

[MONT 02] D. C. Montgomery, G. C. Runger, "Applied Statistics and Rpobability for Engineers", published by John Wiley & Sons, ISBN: 0-471-20454-4, 2002.

[NGHG 95] H. Ng and R.L. Grimsdale, "GEOFF-A Geometrical Editor for Fold Formation", Lecture Notes in Computer Science: Image Analysis Applications and Computer Graphic, published by Springer-Verlag, Vol. 1024, pp124-131, 1995.

[NORD 03] NORDSTROM, http://www.nordstrom.com.

[OPTI 03] OpenGL Optimizer and Cosmo3D libraries: http://www.sgi.com/software/optimizer/.

[OSHI 01] M. Oshita, A. Makinouchi, "Real-time Cloth Simulation with Sparse Particles and Curved Faces", Computer Animation, published by IEEE Press, pp. 220–227, 2001.

[PAPA 03] G. Papagiannakis, A. Foni, N. Magnenat-Thalmann, "Real time recreated ceremonies in VR restituted cultural heritage sites", CIPA XIXth International Symposium, to appear, 30 September.

[PIXA 02] Pixar, http ://www.pixar.com/

[POND 03] M. Ponder, B. Herbelin, T. Molet, S. Schertenlieb, B. Ulicny, G. Papagiannakis, N. Magnenat-Thalmann, D. Thalmann, "Immersive VR Decision Training: Telling Interactive Stories Featuring Advanced Virtual Human Simulation Technologies", 9th Eurographics Workshop on Virtual Environments (EGVE), ISBN 3-905673-00-2, 2003.

[POND 03] M. Ponder, G. Papagiannakis, T. Molet, N. Magnenat-Thalmann, D. Thalmann, VHD++ Development Framework: Towards Extendible, Component Based VR/AR Simulation Engine Featuring Advanced Virtual Character Technologies, Proceedings of Computer Graphics International (CGI), published by IEEE Press, 2003.

[PRES 88] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, "Numerical Recipes in C, The art of scientific computing", published by Cambridge University Press, 1988.

[PROT 02] D. Protopsaltou, C. Luible, M. Arevalo, N. Magnenat-Thalmann, "A body and garment creation method for an Internet based virtual fitting room", proceedings of CGI 2002 (Computer Graphics International), published by IEEE Press, July 2002.

[PROV 95] X. Provot, "Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior", Graphics Interface'95 proceedings, published by A K Peters, pp 147-154, 1995

[REDO 02] S. Redon, A. Kheddary, S. Coquillart, "Fast Continuous Collision Detection between Rigid Bodies", Eurographics, published by Blackwell Publishers, Vol. 21(3), pp. 279-288, 2002.

[RUDO 00] Rudomín G. Ma. Elena Melón J. Isaac, "Multi-Layer Garments using Hybrid Models", Visual 2000, published by Springer, pp. 118, 2000.

[SEDE 86] T.W. Sederberg, SR. Parry, "Free Form Deformation of Solid Geometric Models", SIGGRAPH 86 Conference Proceedings, Annual Conference Series, published by ACM SIGGRAPH Addison Wesley, Vol. 20, No. 4, pp. 151-160, July 1986.

[SEOH 03] H. Seo, N. Magnenat-Thalmann, An Automatic Modeling of Human Bodies from Sizing Parameters, ACM SIGGRAPH 2003 Symposium on Interactive 3D Graphics, published by ACM Press, pp. 19-26, 2003.

[SEOI 03] H. Seo, F. Cordier, N. Magnenat-Thalmann, "Synthesizing Animatable Body Models with Parameterized Shape Modifications", ACM SIGGRAPH/Eurographics Symposium on Computer Animation, published by ACM Press, pp. 120-125, July 2003.

[SEOJ 03] H. Seo, N. Magnenat-Thalmann, "An Example-Based Approach to Human Body Manipulation", in press, Graphical Models, published by Academic Press, 2003.

[SEOK 03] H. Seo, "Parameterized Human Body Modeling", PhD dissertation, MIRALab, University of Geneva, 2003.

[SIMC 03] SimCloth from Chaos Group, http://www.chaosgroup.com/

[SLOA 01] P. P. Sloan, C. Rose and M. Cohen, "Shape by Example", ACM SIGGRAPH 2003 Symposium on Interactive 3D Graphics, published by ACM Press, pp. 135—143, March 2001.

[STIT 03] Stitch from Digimation, http://www.digimation.com

[SUTI 02] Sunday Times Magazine, "Read my hips", published by Times Newspapers Ltd, pp. 31—32, March 2002.

[SYFL 03] Syflex from Syflex LLC, http://www.syflex.biz/.

[TERZ 87] D. Terzopoulos, J.C. Platt, H. Barr, "Elastically Deformable Models", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, Vol. 21, pp 205-214, 1987.

[TERZ 88] D. Terzopoulos, k. Fleischer, "Deformable Models", The Visual Computer, published by Springler-Verlag, Vol.4 (6), pp.306-331, 1988.

[THAL 94] N. Magnenat-Thalmann, "Tailoring Clothes for Virtual Actors", Interacting with Virtual Environments, edited by MacDonald, L., and Vince, J., John Wiley & Sons, 205-216, 1994.

[TOSI 90] T. L. Kunii and H. Gotoda, "Singularity Theoretical Modeling and Animation of Garment Wrinkle Formation Processes", The Visual Computer, published by Springler-Verlag, Vol. 6(6), pp. 326-336, 1990.

[UBIS 03] 'Ages Beyond Myst' from UbiSoft, http://www.ubi.com/US.

[VASS 00] T. Vassilev, B. Spanlang, "Efficient Cloth Model for Dressing Animated Virtual People", proceedings of Learning to Behave Workshop, Enschede the Netherlands, pp. 89-100, October 2000.

[VASS 01] T. Vassilev, B. Spanlang, "Fast Cloth Animation on Walking Avatars, Eurographics", published by Blackwell Publishers, September 2001.

[VERL 67] Verlet L., "Computer experiments on classical fluids, thermodynamical properties of lennard-jones molecules", Physical Review, published by Springer-Verlag, Vol. 159, pp. 98-103, 1967.

[VICO 03] Vicon, http://www.vicon.com/.

[VLAC 01] A. Vlachos, J. Peters, C. Boyd, J. L. Mitchell, "Curved PN Triangles", ACM SIGGRAPH 2001 Symposium on Interactive 3D Graphics, published by ACM Press, pp. 159—166, 2001.

[VOLI 00] P. Volino, Nadia Nadia Magnenat-Thalmann, "Implementing fast Cloth Simulation with Collision Response". Proceedings of CGI'00, published by IEEE Press, June 2000.

[VOLI 01] Pascal Volino, Nadia Magnenat-Thalmann, "Comparing Efficiency of Integration Methods for Cloth Animation", Proceedings of CGI'01, Hong-Kong, IEEE Press, July 2001

[VOLI 02] P. Volino, N. Magnenat-Thalmann, "The internet changing room", journal of Textileforum: New Technologies and Materials, published by Textilwerkstatt Verlag, pp. 22—23, November 2002.

[VOLI 03] P. Volino, F. Cordier, N. Magnenat-Thalmann, "From Early Virtual Garment Simulation to Interactive Fashion Design", Computer-Aided Design, special issue on CAD Methods in Garment Design, published by Elsevier Science, in press, 2003.

[VOLI 95] P. Volino, M. Courchesne, N. Magnenat-Thalmann, "Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects", Computer Graphics (SIGGRAPH'95 proceedings), published by Addison-Wesley, pp. 137-144, 1995.

[VOLJ 00] P. Volino, N. Magnenat-Thalmann, "Accurate Collision response on polygonal Meshes", proceedings of Computer Animation 2000, Annual Conference Series, published by IEEE Press, pp. 154, May 2000

[VOLK 00] P. Volino, N. Magnenat-Thalmann, "Virtual Clothing - Theory and practice", published by Springer Verlag, ISBN: 3-54067-600-7, December 2000.

[VRLA 03] Virtual Reality Lab, http://vrlab.epfl.ch/

[WAGN 00] M. Wagner, "3D E-Commerce: Fact or Fiction?", Proceedings of the 6th International Conference on Virtual Systems and MultiMedia (vsmm), published by Ohmsha Press, pp.634-642, 2000.

[WEBE 00] J. Weber, "Run-Time Skin Deformation", proceedings of Game Developers Conference 2000, http://developer.intel.com/ial/3dsoftware/mrm.htm.

[WEIL 86] J. Weil, "The synthesis of Cloth Objects, Computer Graphics", SIGGRAPH 86 Conference Proceedings, Annual Conference Series, ACM SIGGRAPH, published by Addison Wesley, Vol. 20, pp. 49-54, 1986

[WELC 03] Welcome to My Virtual Model™, http://www.landsend.com/

[WITK 01] A. Witkin, D. Baraff, "Physically based modeling", Siggraph 2001, Course Notes, http://www.pixar.com/companyinfo/research/pbm2001/index.html

[YANG 93] Y. Yang, N. Magnenat Thalmann, An Improved Algorithm for Collision Detection in Cloth Animation with Human Body, Proceedings of Pacific Graphics '93.