VIRTUAL TRY ON SYSTEMS FOR CLOTHES: ISSUES AND SOLUTIONS

Mihai FRÂNCU[1], Florica MOLDOVEANU[2]

Virtual try on (VTO) systems for clothes are complex systems with many components. In this paper we introduce some 3D scanning and reconstruction techniques for fashion and then give a presentation of the available garment simulation techniques and possible simplifications. These methods were all implemented by us with a focus on constraint based simulation. In the end we refer to real-time rendering and simulation and its challenges together with possible solutions, e.g. GPGPU.

Keywords: virtual try on, cloth simulation, collision detection, 3D scanning

1. Introduction

The technologies needed for a virtual try on (VTO) system overlap a lot those required by CAD software for designing apparel [1]. Some of the companies that have developed such CAD systems include Optitex, Lectra (Modaris), Toray-Acs, Gerber (AccuMark), Investronics, Assyst-Bullmer, DressingSim. Another example is the software Marvelous Designer from CLO Virtual Fashion, which is used in both textile industry and entertainment.

The most well-known and documented experiments with VTO are those of MiraLab in Switzerland [2, 3]. They identify three most important modules of such a system: a body sizing module, a motion retargeting module and a cloth simulation module. The garments are always considered available in their studies from sewing together 2D patterns around a body, just like in CAD systems. Note that in this paper we are focusing mainly on real-time rendering and simulation solutions that make use of a virtual reality (VR) environment rather than an augmented reality (AR) one.

At the moment there are no working virtual try on systems available on the market but there has been ongoing work for more than a decade by companies and academia. For example, HumanSolutions participated in such a project with several German universities [4]. Other startups in the field include PhiSix, TriMirror, Fitiquette, 3D-a-porter and possibly others.

[1] PhD student, Faculty of Automatic Control and Computers, University POLITEHNICA of Bucharest, Romania, e-mail: mihai.francu@cs.pub.ro

[2] Professor, Faculty of Automatic Control and Computers, University POLITEHNICA of Bucharest, Romania

From our standpoint there are a few key challenges that are very similar but not always to those outlined by the references from MiraLab:

- *avatar creation* - can be done through parametric interpolation via a UI, but scanning and photos can be alternative sources;
- *avatar animation* - can be created by hand, from motion capture and there is also the option of real-time skeletal data from a Kinect like device ("virtual mirror");
- *garment acquisition* - we cannot always rely on patterns so we need to be able to reconstruct the model from existing physical clothes;
- *cloth simulation* - is the main performance bottleneck (especially on the collision detection side); ideally it has to be both realistic and run in real time: a middle ground has to be reached;
- *rendering* - this is preferably done in web browsers (WebGL) and on mobile (OpenGL ES), but desktop computers are the fastest;
- *cloud processing* - this is a possible solution for offloading some of the simulation computation; challenges include geometry or video streaming and interaction lag.

1.1. Avatar creation and animation

There exist 4 major ways of 3D reconstructing the body: from laser scans, from structured light, from depth cameras and from photos. The most accurate body reconstruction involves laser scanning and many times it is not fully automated, requiring some human input [1]. The person being scanned needs to wear tight clothes for accurate features determination and sizing. The alternatives are thermal/infra-red (IR), X-ray backscatter or millimeter wave scanners, but all these are expensive solutions that can only be installed in shops and can’t be used at home.

Body surface reconstruction methods from point clouds can be classified as three types [1]:

1. implicit (level sets, signed distance fields, moving least squares, radial basis function, Poisson reconstruction);
2. parametric (NURBS, T-spline) and
3. triangulation methods (Delaunay).

The laser scanning solution from Human Solutions [4] creates large point clouds (hundreds of thousands) which are then converted to NURBS surfaces and color information is also acquired as textures. All characteristic feature points are captured automatically and used to create the inner skeleton for animation and skinning. Another alternative is the TC2 body scanner using structured light (i.e. bands or other patterns of light that get deformed when projected on surfaces).

Depth cameras (e.g. Kinect) have been used in the recent years for scanning and reconstructing objects [5]. They can also be used for implementing

VTOs [6] mainly for capturing skeletal movement and many augmented reality (AR) based startups are using it (e.g. Fitnect). Their downside is that they are less accurate and noisy compared to laser scanners [7]. Commercial software for Kinect based reconstruction also exists, e.g. Skanect (Occipital) or MyBodee (Styku). Bodymetrics offers a reconstruction booth for in-shop use. More recently a portable IR sensor for the iPad (Structure Sensor from Occipital) has been launched that is similar to Kinect and has its own reconstruction mobile application.

Reconstruction from images is also possible [1], e.g. from 4 orthogonal directions photos. Such a solution is provided by the startup Fitle. This is similar to the general purpose method provided by Autodesk with 123D Catch. Other similar stereo computer vision based reconstruction software packages include Agisoft PhotoScan, Pix4D or VisualSFM; the latter two were used for scanning statues together with Poisson surface reconstruction and mesh post-processing [31]. A system which uses both webcams and Kinect cameras is presented in [8]: it only needs a front and a profile picture; they mention camera calibration issues and need to extract the background. Another problem may be that the pose is not symmetric (solutions are presented in [1]) or that users will be reluctant to stand in a regular pose (close to a T-pose). Wearing loose clothes is another obstacle but an image based solution has managed to overcome it [9].

A different option is to edit the avatar body parametrically using sliders for different body dimensions; this is available for example in Marvelous Designer. Such techniques are used by the MiraLab VTO and described in [2]; they can be categorized as follows: interactive, reconstructive, example based, anthropometric, and multi layered modeling. Basically a template body model is deformed in real time until the desired shape is attained. Using anthropometric standards (ISO-7250, ISO-8559, EN-13402) and dimensions of the main body regions they create a body that is then deformed using free form deformation (FFD) and radial basis functions (RBF) [10]. Other techniques relying on feature points, curves and patches or parametric design of human models (example based model synthesis) are presented in [1]. An existing solution is the open-source MakeHuman API.

Avatar animations can be created by hand for a standard size avatar using tools like Autodesk 3D Studio MAX or Maya. Motion capture is an alternative, i.e. capturing the motion of real people doing actions related to trying out clothes. When changing morphology of the avatar the animations have to be adapted to the new skeleton. This is not a straightforward task and is called motion retargeting. The MiraLab VTO uses spacetime constraints [2] as an alternative to inverse kinematics (IK). Other manual or automatic changes may have to be made in order to prevent limb inter-penetrations. Care must be taken to address footskate

[46] (feet move when they ought to remain planted) and balance, i.e. the zero momentum point (ZMP) should fall in the supporting area.

1.2. Garment acquisition

For reconstructing clothes again laser scanning is the best choice, but depth cameras can be used too. Recently methods using photogrammetry and light fields have given good results; structure from motion (videogrammetry) is also possible. One preferred method for reconstructing deformable surfaces from multi-view photography is the one in [42]. A popular example of software doing such 3D reconstruction from photos is Agisoft PhotoScan. An open-source alternative is VisualSFM but it behaves more poorly [11]. Usually these image scanning techniques require tens of DSLR cameras arranged in a cylindrical rig with lighting. The resulting mesh is cleaned up and texture is generated and projected onto it. Note that garments are photographed on a mannequin.

Capturing moving cloth techniques are described in [12] and [13]. The former needs no markers, uses 16 viewpoints, but needs to fill holes. The latter needs a color pattern on the cloth and uses multiple synchronized video cameras too. All these methods fall under the category of stereo computer vision, although there is some ongoing effort of using sequences of monocular images for reconstruction too [43]. More recently Zhou et al. [14] were able to reproduce garment from a single front image.

The type of the material is also important for the look of the simulation. One needs to make the difference between linen, fleece, satin, knit, silk, denim etc. The material parameters can be obtained through direct mechanical testing of the fabric, e.g. the Kawabata Evaluation System (KES) [3]. A more advanced method that uses both mechanical testing and 3D cloth surface reconstruction was developed recently [35]. Others have tried to deduce these parameters from video sequences (see [15] or more recently [16]).

2D patterns can be imported from a CAD format or scanned from paper or an image [17]. In the case of patterns not being available, another approach is to scan clothes using a mannequin of standard size. One ca use more mannequin sizes for different clothes sizes (e.g. S, M, L, XL) or use interpolation instead as in [4] or [51]. Care must be taken that the mannequin and background elements are subtracted from the garment mesh. If using photographic reconstruction techniques, the captured mesh might have holes that need to be filled (see [12]). Problems may also appear with transparent objects. Even if using laser scanning the resulting shape may have similar problems that need to be fixed.

2. Cloth simulation

Cloth simulation is a complex and time consuming step consisting of two parts: collision detection and the physics solver. We will start with the latter and

describe the methods that we used. Most cloth simulation methods treat the fabric as a network of mass particles and springs between them. These systems are solved either through implicit integration or constraint projection. Explicit integration is usually not an option as it can get very easily unstable, especially for time steps used in real-time simulation. Other simulation models treat the cloth as a continuum and use the finite element method (FEM) to solve the partial differential equations (PDE) of motion. Mass-spring systems on the other hand integrate directly ordinary differential equations (ODE). The FEM method is more accurate and better suited for simulating real fabrics; real-time simulation is possible but it is still slower. Therefore the mass-spring model is a faster solution for real-time but may not be realistic enough. A special case of mass-spring systems is constrained based dynamics (infinitely stiff springs), which is becoming more and more popular in games and even 3D authoring tools like Autodesk Maya (Nucleus [47]).

A favored method in games for simulating cloth is position based dynamics (PBD) as it is easy to implement; still it needs further improvements, e.g. on the self-collision side and optimization for real-time (parallelization). Open-source cloth engines include the one in Bullet (PBD), VegaFEM or OpenTissue (implicit). Commercial solutions are available from Havok and PhysX APEX (also available for free).

2.1. Mass-spring systems

Most often cloth is modeled as a particle system, i.e. masses connected by springs. The positions of the particles are denoted by vectors $\mathbf{x}_i$ and the forces in the springs have the form:

$$\mathbf{f}(\mathbf{x}_i, \mathbf{x}_j) = -k_{ij}(\|\mathbf{x}_{ij}\| - l_{ij})\hat{\mathbf{x}}_{ij}, \tag{1}$$

where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, $\hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij}/\|\mathbf{x}_{ij}\|$, l_{ij} is the rest length of the spring and k_{ij} its stiffness. These forces stem from an elastic potential energy function $U = \sum_{i,j} \frac{k_{ij}}{2}(\|\mathbf{x}_{ij}\| - l_{ij})^2$ and so the force acting on a particle is: $\mathbf{f}_i = -\frac{\partial U}{\partial \mathbf{x}_i}$. As you can see the resulting force is linear, but other models with higher order polynomials than quadratic potentials can be used too [36]. Using Newton's second law: $m_i\ddot{\mathbf{x}}_i = \mathbf{f}_i$, we can calculate the trajectories of the particles using a numerical integration scheme. The simplest methods are explicit and include Forward Euler, Symplectic Euler (SE), Verlet, Leapfrog, Runge-Kutta of order 2 or 4 (RK2 and RK4). Unfortunately they can blow up easily if the time step goes out of a stability domain: e.g. for SE and Verlet $h \le 2/\omega$, where ω is the highest angular frequency in the system. This forces us to use very small time steps which can incur a performance cost.

Implicit methods on the other hand are unconditionally stable at the price of artificially dissipating energy resulting in numerical damping. Examples of implicit schemes include Backward Euler, Implicit Midpoint, predictor-corrector, BDF-2 and Newmark [33, 34]. Backward Euler is the most used method in cloth simulation thanks to the seminal paper of Baraff and Witkin [18]:

$$m_i \mathbf{v}_i^{n+1} = m_i \mathbf{v}_i^n + h\mathbf{f}_i^{n+1} + h\mathbf{f}_{ext},$$
$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n + h\mathbf{v}_i^{n+1}.$$

Here the forces are evaluated at positions $\mathbf{x}_i^{n+1}$ which haven't been yet computed and this means we need to solve a nonlinear system in order to advance to the next frame. By using the Newton method we obtain a series of linear systems:

$$(\mathbf{M} - h^2\mathbf{K})\mathbf{v}^{n+1} = \mathbf{M}\mathbf{v}^n + h\mathbf{f}^n,$$

where $\mathbf{M} = \text{diag}(m_i)$ is the mass matrix and $\mathbf{K} = \dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}$ is the (tangential) stiffness matrix (see [20] for details). Usually most authors choose to run only one Newton iteration and solve only one linear system and obtain satisfactory results. This approach is called semi-implicit or linearly implicit. Other methods combine explicit integration with implicit integration of the stiffer components of the motion - these are known as IMEX methods [41].

We have implemented many of these integration schemes (Symplectic and Implicit Euler, Implicit Midpoint, Newmark) mainly as a reference. Also, we developed a new method for implicitly integrating mass-spring systems based on mathematical optimization and the Nonlinear Conjugate Gradient algorithm [27].

2.2. Position based dynamics

This method was originally used in molecular dynamics simulations and is a nonlinear form of constrained dynamics - most popular methods are Shake and Rattle [19]. It was pioneered in games by Jakobsen [21] and later refined by Müller [22] under the name position based dynamics (PBD). Since then the method has been used by many authors for many applications [23] and was better developed theoretically by Goldenthal [24] who called it *fast projection*. The most general equations of constrained dynamics are:

$$\mathbf{M}\ddot{\mathbf{x}} = \mathbf{J}^T\boldsymbol{\lambda} + \mathbf{f}_{ext}, \tag{2}$$
$$\mathbf{c}(\mathbf{x}) = \mathbf{0}, \tag{3}$$

where $\mathbf{c}(\mathbf{x})$ is the vector function of constraints, $\mathbf{J}^T = \nabla\mathbf{c}(\mathbf{x})$ is its gradient and $\boldsymbol{\lambda}$ is the Lagrange multipliers vector enforcing the constraints, i.e. the magnitudes of the internal forces. The equations of motion (2) can be integrated using any explicit method, preferably a symplectic method like Verlet for example:

$$\mathbf{x}^{n+1} = 2\mathbf{x}^n - \mathbf{x}^{n-1} + h\mathbf{M}^{-1}(\mathbf{J}^T\boldsymbol{\lambda} + \mathbf{f}_{ext}).$$

The projection method involves advancing the positions under external forces only (unconstrained step) and then projecting them onto the constraint manifold by solving the nonlinear equation (3). This is usually done by employing a relaxation method like nonlinear Gauss-Seidel, but we also used other methods, e.g. Newton method (fast projection) or optimization techniques [25].

Initially position correction methods were used to enhance spring based simulation and were called *inverse dynamics* [39] or *strain limiting* [40] techniques. We proved in our work that PBD is actually a full-fledged physical method and that it is equivalent to implicit integration of elastic systems [27].

2.3. Finite element method

The finite element method (FEM) for solving the partial differential equations of continuous elastic media can also be applied to thin surface materials like cloth. This involves defining a planar strain tensor based on the partial derivatives of a deformation function $\mathbf{w}(u,v)$ defined over a parametrization of the mesh (e.g. texture coordinates, warp and weft). We are only considering linear FEM here, so for a triangle (i,j,k) these are constant and expressed by [18]:

$$\begin{pmatrix} \mathbf{w}_u \\ \mathbf{w}_v \end{pmatrix} = \begin{pmatrix} u_j - u_i & u_k - u_i \\ v_j - v_i & v_k - v_i \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{x}_j - \mathbf{x}_i \\ \mathbf{x}_k - \mathbf{x}_i \end{pmatrix}$$

The above vector can be denoted by $\nabla \mathbf{w}^T$ and used to construct the Green-Lagrange strain tensor, i.e. a 2x2 symmetric matrix whose components are:

$$\begin{aligned} \boldsymbol{\varepsilon}_{uu} &= \tfrac{1}{2}(\mathbf{w}_u^T \mathbf{w}_u - 1), \\ \boldsymbol{\varepsilon}_{vv} &= \tfrac{1}{2}(\mathbf{w}_v^T \mathbf{w}_v - 1), \\ \boldsymbol{\varepsilon}_{uv} &= \boldsymbol{\varepsilon}_{vu} = \mathbf{w}_u^T \mathbf{w}_v . \end{aligned}$$

In the context of cloth it is actually more convenient to use this nonlinear strain instead of the linear Cauchy strain, resulting in a Saint Vennant-Kirchoff model of nonlinear elasticity [3]. The stress tensor $\boldsymbol{\sigma}(\boldsymbol{\varepsilon})$ is obtained from differentiating the energy density function: $\Psi(x) = \hat{\boldsymbol{\varepsilon}}(x)^T \mathbf{E} \hat{\boldsymbol{\varepsilon}}(x)$, where $\mathbf{E}$ is a matrix expressing the stress-strain relationship and the hat denotes conversion from symmetric tensor to 3-vector (for a derivation in 3D see [32] Part I). It usually has the form:

$$\mathbf{E} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{1-\nu}{2} \end{bmatrix},$$

where E is Young's modulus and ν is the Poisson ratio. Different Young's moduli can be used for each of the two main orthogonal fabric direction, i.e. warp and weft. Similarly we can use different values for the Poisson ratios to model full

anisotropy of the material, and we can also replace the last diagonal element in the matrix by G - the shearing modulus.

Forces are then computed from the stress tensor and applied to each particle [20] (masses are considered lumped in the nodes of the mesh). The integration method is again important for stability and so implicit schemes are preferred. You can find such a solution in [26] together with a spline-based approximation of the nonlinear stress-strain relationship. We chose to implement a different version of FEM cloth simulation based on constraints and PBD [27]. You can find another PBD variant of FEM cloth simulation in [37].

2.4. Damping and bending

There are two ways of adding damping to cloth simulation: along the spring directions and air drag. The former has a similar expression to (1):

$$\mathbf{f}_d(\mathbf{x}_i, \mathbf{x}_j) = -d_{ij}[\hat{\mathbf{x}}_{ij} \cdot (\mathbf{v}_i - \mathbf{v}_j)]\hat{\mathbf{x}}_{ij},$$

where d_{ij} is the spring damping coefficient, while the latter is a simple viscous drag force of the form $\mathbf{f}_v = -\eta\mathbf{v}$. The gradient of the damping force is the matrix $\mathbf{D}$ used for implicit integration. Many times it is approximated using Rayleigh damping: $\mathbf{D} = \eta\mathbf{M} + \gamma\mathbf{K}$ [34]. As you can see the first term corresponds to viscous drag and the second to dashpots having stiffness k_{ij} and damping factor $d_{ij} = \gamma k_{ij}$.

Bending is a more complex phenomenon as it takes place out of plane. The simplest approach is to add low stiffness springs between order 2 neighbors in the connectivity graph [30]. Another approach we used is a "nonlinear hinge" approach like in [18, 22]: this measures the dihedral angle between two triangles sharing an interior edge and tries to keep it constant. We also tried more accurate bending force models based on curvature energy and simplifications of it [29].

We added the above damping model to our direct force integration simulators and also to the PBD simulation by means of a transformation method that we devised [27]. Unfortunately we had to exclude the first term of Rayleigh damping, but we added back air drag through explicit integration.

5. Realism and performance

Cloth materials are usually very resistant to stretching and slightly less to shearing. The most freedom a fabric has is for out-of-plane movement, i.e. bending. Thus an accurate bending model is important for replicating real cloth. Other phenomena like anisotropy, hysteresis, plasticity, damping and viscosity are also important to cloth modeling [38], but each introduces further complexity and performance issues. Also, depending on the model chosen, it is not trivial to set the parameters of cloth so that they match a real material (e.g. silk, denim, linen,

cotton, polyester): some have to be chosen from measured parameters, some by hand and some from pre-stored presets that can be modified.

In conclusion, we had to make a trade-off between accuracy and speed, depending on the computing power available. Another factor that impacted both performance and realism was tessellation: coarser simulation meshes run faster but produce less folding and wrinkles. High frequency details like wrinkles can be added after the simulation in a plausible fashion [48]. There also exist adaptive tessellation techniques [28] that refine the simulation mesh depending on the current level of detail, visible parts etc.

Other methods involve "cheating": the simulation is pre-computed in all sorts of possible ways that the avatar could move and the new poses and animations are interpolated [53]. Also, if all the animations are known in advance, the whole simulation can be computed on a super-computer in the cloud. Still two problems arise: how to stream the data (e.g. geometry or video) and how to handle network lag (especially if using Kinect or the scene is non-deterministic).

6. Collision detection and handling

The physics solver also deals with the contacts between objects which are reported by the collision detection system. It is extremely important not to allow inter-penetrations and both solving and detecting such contacts can be computationally expensive. This is extremely problematic for the case of self-collisions of cloth, as the cloth is very thin, it gets tangled very easily [49] and it is very hard to tell which side is “the right side” for resolving contacts. In this regard *continuous collision detection* (CCD) algorithms and *bounding volume hierarchies* (BVH) were needed to solve and accelerate the (self-)collision problem [40]. We also had to account for friction, which can prove quite a daunting problem, and so we adapted a method from rigid body simulation [54].

One solution for reducing the overhead of collision with the body is to approximate it with primitives, e.g. spheres, ellipsoids, capsules. We did this manually but we had to use big tolerances to account for all possible movements.

We differentiate between cloth collisions with external objects and self-collisions (includings collisions with other pieces of cloth). External collisions can be with static objects or dynamic objects and both types can consist of a triangle mesh or a collection of volumetric primitives. Collision detection against static triangles or primitives was done in a discrete fashion for the case when the cloth is not moving too fast (or for small time steps): intersection was tested at the beginning of the frame with the current positions of the cloth vertices. When the relative velocity between the cloth and the possibly dynamic object was high, we resorted to CCD techniques, i.e. trajectories of objects are linearly parametrized in time as swept shapes (particles become segments, spheres become capsules, etc).

As you can imagine, CCD was more complex to implement and more computationally expensive. You can see some of our results in Fig. 1.

Fig. 1: Complex cloth simulation scenarios with self-collision simulated in real-time

In the mass-spring framework, contacts are solved using the penalty method: springs are added between colliding objects that push them apart until they do not intersect anymore. The elastic force can depend on the penetration depth or the volume of the overlapping region. These forces are the normal reaction forces $\mathbf{f}_n$ which can then be used to compute friction forces: these have magnitude equal to $\mu\|\mathbf{f}_n\|$ (where μ is the friction coefficient) and direction opposite to the sliding velocity vector.

In the context of rigid body simulation, penalty methods have not proven to be a robust method and so they were replaced by non-smooth constrained dynamics methods. On a mathematical level they can be expressed as a nonlinear complementarity problem [52]: $\mathbf{c}(\mathbf{x}) \geq \mathbf{0}, \boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\lambda}^T \mathbf{c}(\mathbf{x}) = \mathbf{0}$. These methods fit very well with the constraint projection framework of PBD and this is why we chose not to use penalty methods in our work [27, 54].

7. Rendering

When rendering cloth we had two options: either render the simulation mesh or render a different mesh that moves in sync with the simulation. The former is easier but usually the cloth mesh is quite coarse for rendering. The first solution was to further tessellate the cloth mesh at every frame. One way is to use subdivision surfaces [45] or use the points and their normals to do an implicit surface reconstruction [50]. We tested some tessellation methods ourselves: for rectangular grid meshes we used a bicubic interpolation method (e.g. Catmull-Rom) to subdivide the quads, while for triangle meshes we tried GPU enabled tessellation shaders, but the results proved unsatisfactory in both cases.

The tessellation can also be too expensive, so using a different render mesh can prove a better choice. An advantage of this second method is that you can use any type of mesh (e.g. non-manifold). The process we developed for moving the render mesh along with the cloth resembles FFD [44] and requires a registration step that maps each vertex on the render mesh to the simulation mesh. We stored the barycentric coordinates of the vertex with respect to the closest triangle along with the height of the resulting tetrahedron (with the vertex as the apex). This way when the reference mesh deforms the new vertex position can be linearly interpolated inside the triangle and then translated along the normal.

8. Acceleration

The most straightforward way of accelerating cloth simulation and collision detection as well as the other modules of the VTO is parallelization. This can be done on multi-core CPUs (e.g. ranging from Intel Core to Xeon Phi), clusters or on SIMD processors (SSE/AVX, GPU). The most promising of these was for us general purpose GPU computing (GPGPU) which can provide much more number crunching power than the traditional CPU even on consumer devices. Unfortunately not all programs can be ported to GPGPU and algorithms need to redesigned in order to exploit the very wide SIMD architecture of graphics processors. This is why we developed new constraint solvers based on the Jacobi and Conjugate Residuals methods and were implemented in parallel on the GPU using OpenCL kernels [27, 54].

8. Conclusions

In this paper we described the full pipeline needed to create a virtual fitting room for fashion and the challenges associated with it. Providing a real time simulation of realistic apparel on current generation computers or mobile devices is still a difficult task despite all the hardware advances. Of all the

modules that make up a VTO system we have dealt directly in our own work with cloth simulation, collision detection and rendering [25, 27, 54].

There are at least three methods for simulating cloth, each with advantages or computational costs. Mass-spring models are cheap, but are slowly being replaced by FEM for more accuracy and realism. PBD may seem less physically correct, but it can actually be used to simulate continuous elastic materials and it's becoming more and more popular for real-time and games. The collision detection phase is the other bottleneck besides the physical solver and it is also need of acceleration data structures and parallelization.

Acknowledgments

Part of the research presented in this paper was supported by the Sectoral Operational Program Human Resources Development, 2014-2015, of the Ministry of Labor, Family and Social Protection through a Financial Agreement between University POLITEHNICA of Bucharest and AM POS DRU Romania, project ID 132395.

R E F E R E N C E S

[1]. *Liu, Yong-Jin, Dong-Liang Zhang, and Matthew Ming-Fai Yuen*. "A survey on CAD methods in 3D garment design." Computers in Industry 61, no. 6 (2010): 576-593.

[2]. *Magnenat-Thalmann, Nadia, Bart Kevelham, Pascal Volino, Mustafa Kasap, and Etienne Lyard*. "3d web-based virtual try on of physically simulated clothes." Computer-Aided Design and Applications 8, no. 2 (2011): 163-174.

[3]. *Magnenat-Thalmann, Nadia, ed*. Modeling and simulating bodies and garments. Springer Science & Business Media, 2010.

[4]. *Divivier, A., R. Trieb, Aea Ebert, H. Hagen, C. Gross, A. Fuhrmann, and V. Luckas*. "Virtual try-on topics in realistic, individualized dressing in virtual reality." (2004).

[5]. *Keller, Matthias, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb*. "Real-time 3D reconstruction in dynamic scenes using point-based fusion." In 3D Vision-3DV 2013, 2013 International Conference on, pp. 1-8. IEEE, 2013.

[6]. *Philipp Presle*, A Virtual Dressing Room based on Depth Data, MS Thesis, 2012

[7]. *Stoyanov, Todor, Athanasia Louloudi, Henrik Andreasson, and Achim J. Lilienthal*. "Comparative evaluation of range sensor accuracy in indoor environments." In 5th European Conference on Mobile Robots, ECMR 2011, September 7-9, 2011, pp. 19-24.

[8]. *Lee, Won Sook, Jin Gu, and Nadia Magnenat-Thalmann*. "Generating animatable 3D virtual humans from photographs." In Computer Graphics Forum, **vol. 19**, no. 3, pp. 1-10, 2000.

[9]. *Bălan, Alexandru O., and Michael J. Black*. "The naked truth: Estimating body shape under clothing." In Computer Vision–ECCV 2008, pp. 15-29. Springer Berlin Heidelberg, 2008.

[10]. *Kasap, Mustafa, and Nadia Magnenat-Thalmann*. "Parameterized human body model for real-time applications." In Cyberworlds, 2007. CW'07. International Conference on, pp. 160-167. IEEE, 2007.

[11]. *Siegmund, Dirk, Timotheos Samartzidis, Naser Damer, Alexander Nouak, and Christoph Busch*. "Virtual Fitting Pipeline: Body Dimension Recognition, Cloth Modeling, and On-Body Simulation." (2014).

[12]. *Bradley, Derek, Tiberiu Popa, Alla Sheffer, Wolfgang Heidrich, and Tamy Boubekeur*. "Markerless garment capture." In ACM Transactions on Graphics (TOG) 27, no. 3, 2008.
[13]. *White, Ryan, Keenan Crane, and David A. Forsyth*. "Capturing and animating occluded cloth." In ACM Transactions on Graphics (TOG), **vol. 26**, no. 3, p. 34. ACM, 2007.
[14]. *Zhou, Bin, Xiaowu Chen, Qiang Fu, Kan Guo, and Ping Tan*. "Garment modeling from a single image." In Computer Graphics Forum, **vol. 32**, no. 7, pp. 85-91. 2013.
[15]. *Bhat, Kiran S., Christopher D. Twigg, Jessica K. Hodgins, Pradeep K. Khosla, Zoran Popović, and Steven M. Seitz*. "Estimating cloth simulation parameters from video." In Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 37-51. Eurographics Association, 2003.
[16]. *Kunitomo, Shoji, Shinsuke Nakamura, and Shigeo Morishima*. "Optimization of cloth simulation parameters by considering static and dynamic features." In ACM SIGGRAPH 2010 Posters, p. 15. ACM, 2010.
[17]. *Berthouzoz, Floraine, Akash Garg, Danny M. Kaufman, Eitan Grinspun, and Maneesh Agrawala*. "Parsing sewing patterns into 3D garments." ACM Transactions on Graphics (TOG) 32, no. 4 (2013): 85.
[18]. *Baraff, David, and Andrew Witkin*. "Large steps in cloth simulation." In Proceedings of the 25th annual conference on Computer graphics and interactive techniques, pp. 43-54, 1998.
[19]. *Barth, Eric, Krzysztof Kuczera, Benedict Leimkuhler, and Robert D. Skeel*. "Algorithms for constrained molecular dynamics." Journal of computational chemistry 16, no. 10 (1995)
[20]. *Müller, Matthias, Jos Stam, Doug James, and Nils Thürey*. "Real time physics: class notes." In ACM SIGGRAPH 2008 classes, p. 88. ACM, 2008.
[21]. *Jakobsen, Thomas*. "Advanced character physics." In Game Developers Conference, 2001.
[22]. *Müller, Matthias, Bruno Heidelberger, Marcus Hennix, and John Ratcliff*. "Position based dynamics." Journal of Visual Communication and Image Representation 18, no. 2 (2007)
[23]. *Bender, Jan, Matthias Müller, Miguel A. Otaduy, and Matthias Teschner*. "Position-based methods for the simulation of solid objects in computer graphics." EUROGRAPHICS 2013
[24]. *Goldenthal, Rony, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun*. "Efficient simulation of inextensible cloth." ACM Transactions on Graphics (TOG) 26, no. 3 (2007)
[25]. *Mihai Frâncu and Florica Moldoveanu*. "Minimum residual methods for cloth simulation." In System Theory, Control and Computing (ICSTCC), 2014 18th International Conference, pp. 550-555. IEEE, 2014.
[26]. *Volino, Pascal, Nadia Magnenat-Thalmann, and Francois Faure*. "A simple approach to nonlinear tensile stiffness for accurate cloth simulation." ACM Transactions on Graphics 28, no. 4 (2009)
[27]. *Mihai Frâncu and Florica Moldoveanu*, "Cloth Simulation Using Soft Constraints", Journal of WSCG, 2015
[28]. *Narain, Rahul, Armin Samii, and James F. O'Brien*. "Adaptive anisotropic remeshing for cloth simulation." ACM Transactions on Graphics (TOG) 31, no. 6 (2012): 152.
[29]. *Bergou, Miklos, Max Wardetzky, David Harmon, Denis Zorin, and Eitan Grinspun*. "A quadratic bending model for inextensible surfaces." In Symposium on Geometry Processing, pp. 227-230. 2006.
[30]. *Kenny Erleben, Sporring, Jon, Knud Henriksen, and Henrik Dohlmann*. Physics-based animation. Hingham: Charles River Media, 2005.
[31]. http://lgg.epfl.ch/statues_dataset.php
[32]. *Sifakis, Eftychios, and Jernej Barbic*. "FEM simulation of 3D deformable solids: a practitioner's guide to theory, discretization and model reduction." In ACM SIGGRAPH 2012 Courses, p. 20. ACM, 2012.
[33]. *Hauth, Michael*. "Numerical techniques for cloth simulation." 2005

[34]. *Sin, Fun Shing, Daniel Schroeder, and J. Barbič*. "Vega: Non-Linear FEM Deformable Object Simulator." In Computer Graphics Forum, vol. 32, no. 1, pp. 36-48, 2013.
[35]. *Miguel, Eder, Derek Bradley, Bernhard Thomaszewski, Bernd Bickel, Wojciech Matusik, Miguel A. Otaduy, and Steve Marschner*. "Data-Driven Estimation of Cloth Simulation Models." In Computer Graphics Forum, vol. 31, no. 2pt2, pp. 519-528, 2012.
[36]. *Breen, David E., Donald H. House, and Michael J. Wozny*. "Predicting the drape of woven cloth using interacting particles." In Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pp. 365-372. ACM, 1994.
[37]. *Bender, Jan, Dan Koschier, Patrick Charrier, and Daniel Weber*. "Position-based simulation of continuous materials." Computers & Graphics 44 (2014): 1-10.
[38]. *Pascal Volino, Nadia Magnenat-Thalmann*, Virtual Clothing: Theory and Practice, 2000
[39]. *Provot, Xavier*. "Deformation constraints in a mass-spring model to describe rigid cloth behaviour." In Graphics interface, pp. 147-147, 1995.
[40]. *Bridson, Robert, Ronald Fedkiw, and John Anderson*. "Robust treatment of collisions, contact and friction for cloth animation." In ACM Transactions on Graphics (ToG), **vol. 21**, no. 3, pp. 594-603. ACM, 2002.
[41]. *Eberhardt, Bernhard, Olaf Etzmuß, and Michael Hauth*. Implicit-explicit schemes for fast animation with particle systems. Springer Vienna, 2000.
[42]. *Bradley, Derek, Tamy Boubekeur, and Wolfgang Heidrich*. "Accurate multi-view reconstruction using robust binocular stereo and surface meshing." In Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, pp. 1-8. IEEE, 2008.
[43]. *Salzmann, Mathieu, and Pascal Fua*. "Deformable surface 3d reconstruction from monocular images." Synthesis Lectures on Computer Vision 2, no. 1 (2010): 1-113.
[44]. *Singh, Karan, and Evangelos Kokkevis*. "Skinning Characters using Surface Oriented Free-Form Deformations." In Graphics interface, vol. 2000, pp. 35-42. 2000.
[45]. *Schröder, Peter, Denis Zorin, T. DeRose, D. R. Forsey, L. Kobbelt, M. Lounsbery, and J. Peters*. "Subdivision for modeling and animation." ACM SIGGRAPH Course Notes (1998).
[46]. *Kovar, Lucas, John Schreiner, and Michael Gleicher*. "Footskate cleanup for motion capture editing." In Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 97-104. ACM, 2002.
[47]. *Stam, Jos*. "Nucleus: Towards a unified dynamics solver for computer graphics." In Computer-Aided Design and Computer Graphics, 2009. CAD/Graphics' 09. 11th IEEE International Conference on, pp. 1-11. IEEE, 2009.
[48]. *Kavan, Ladislav, Dan Gerszewski, Adam W. Bargteil, and Peter-Pike Sloan*. "Physics-inspired upsampling for cloth simulation in games." ACM Transactions on Graphics (TOG) 30, no. 4 (2011): 93.
[49]. *Baraff, David, Andrew Witkin, and Michael Kass*. "Untangling cloth." In ACM Transactions on Graphics (TOG), **vol. 22**, no. 3, pp. 862-870. ACM, 2003.
[50]. *Johan Huysmans*, Implicit Surface Reconstruction from Point Clouds, 2006
[51]. *Brouet, Remi, Alla Sheffer, Laurence Boissieux, and Marie-Paule Cani*. "Design preserving garment transfer." ACM Trans. Graph. 31, no. 4 (2012): 36.
[52]. *Otaduy, Miguel A., Rasmus Tamstorf, Denis Steinemann, and Markus Gross*. "Implicit contact handling for deformable objects", Computer Graphics Forum, **vol. 28**, no. 2, 2009.
[53]. *Kim, Doyub, Woojong Koh, Rahul Narain, Kayvon Fatahalian, Adrien Treuille, and James F. O'Brien*. "Near-exhaustive precomputation of secondary cloth effects." ACM Transactions on Graphics (TOG) 32, no. 4 (2013): 87.
[54]. *Mihai Frâncu and Florica Moldoveanu*, "An Improved Jacobi Solver for Particle Simulation", VRIPHYS 2014