Modeling of Bodies and Clothes for Virtual Environments

Nadia Magnenat-Thalmann, Frederic Cordier, Hyewon Seo, George Papagianakis
MIRALab, University of Geneva
Centre Universitaire d'Informatique
24 rue du General-Dufour, CH-1211 Genève, Switzerland
E-mail: {thalmann, cordier, seo, papagianakis}@miralab.unige.ch

Abstract

Although graphical human modeling has been a long sought subject in computer graphics, when it comes to dealing with real-time applications, it raises a number of unique requirements for both CG artists and researchers. For example, for a real-time performance it is highly desirable to lighten the geometry and optimize the runtime animation of characters wherever possible. An important need in such systems is to assist the CG artists in seamlessly transforming their modeling work from CG packages to the real-time application, without limiting their expressivity.

In this paper we describe techniques by which dressed human characters are modeled, loaded into VR scenes, and simulated in real-time. In doing so, we point out challenging issues and solutions that such tasks imply at each phase of modeling and simulation. In particular, we turn our attention to what we consider as key technical components: body and clothes. Case studies show that our modelers successfully meet the need of real-time requirements over a wide variety of scenarios.

Keywords: character modeling, authoring tool, content creation, real-time simulation.

1. Introduction

Many applications of virtual environments (VE) require the modeling of human-like characters with high visual and physical accuracy. Many attempts have been made to model, animate, and interact with human-like characters in VE. Modeling of these characters, however, can become a non-trivial task when it comes to dealing with real-time constraints, raising a number of unique issues for both CG artists and researchers. The first is that, for better performance it is highly desirable to lighten the geometry and optimize the deformation of models, so that real-time performance can be maintained within available resources, at any time during the interactive simulation. While a considerable amount of geometric or simulation simplification techniques exist, the state of the art in articulated surprisingly remains an area of artist. In this paper we focus on the problem of optimizing deformation models for most heavy component of the virtual human models, such as bodies and clothes.

Another issue is the integration and compatibility with standards. Very often, VR systems adopt standards, aiming at compatibility and efficiency for data transfer. Given desired models to be loaded in the VR scene, it is required to structure or convert them into these standards.

In this paper we describe techniques by which dressed human characters are prepared and loaded in virtual scenes, in real-time. In doing so, we point out challenging issues and solutions that such tasks imply at each phase of modeling and simulation. In particular, we turn our attention to what we consider as key technical components: body and clothes.

- Bodies: Part of the difficulty in modeling bodies comes from the inherent complexity of skin and muscles, as well as variety of appearances among different individuals. We approach this problem by (1) offering a handful tools that allow to reuse previously worked skinning data on any surface model of a new character with recalculation of the skinning data and (2) overcoming the key limitation of linear blending by adopting matrix operator.
- Clothes: Many dressed 3D characters seen today come up with geometric deformation based on linear blending. While this approach gives satisfactory results for tight clothes, it is hard to achieve satisfactory results in simulating long, ample garments as shown in Figure 4 solely by using geometric deformation. In order to obtain higher realism, we developed a hybrid method that allocates resources to where it can be most efficiently exploited in reproducing visually pleasing cloth movements.

The remainder of this paper is organized as follows. After briefly reviewing existing techniques in Section 2, we turn our attention to key technical components, modeling of bodies, skin deformation, and clothes. We shall discuss them through Section 3, Section 4, and Section 5. We then consider the integration and implementation details in Section 6. In Section 7, we discuss the example application. Finally, we conclude our paper in Section 8.

2. Related work

Significant work has been done on modeling and simulating virtual humans, in areas such as training and education application, and interactive simulation. Despite the large amount of pioneering work on modeling and deformation techniques for virtual humans, there continue to be exiting new applications and improvements in enhancing realism, saving production time, and efficient simulation.

Numerous examples of human body modeling exist in the literature with particular areas of interest being deformations of skin and muscles [7][5][3], variety of appearances among different individuals [4][21], and face models. Significant support exist in commercial graphics packages such as Maya [14] and 3ds max[1], for designing and authoring the static and dynamic shape changes of realistic human-like characters.

There have also been significant recent developments for real-time simulation, especially for physically based simulation for dressed human-like characters.

3. Body modeling

Part of the difficulty in modeling bodies comes from the inherent complexity of skin and muscles, as well as variety of appearances among different individuals. While there are now available a number of ways to easily obtaining character surface models from a variety of sources (commercial graphics packages such as Poser©[19] or from range scanners), implanting the skeleton structure and defining appropriate skinning data, that too as of one model after another, remain as tedious manual task.

Given a surface model from any external source, our goal is to assist the designer in efficiently locating joint centers by reusing existing bone-to-skin data, that is to say to convert it into an animatable model.

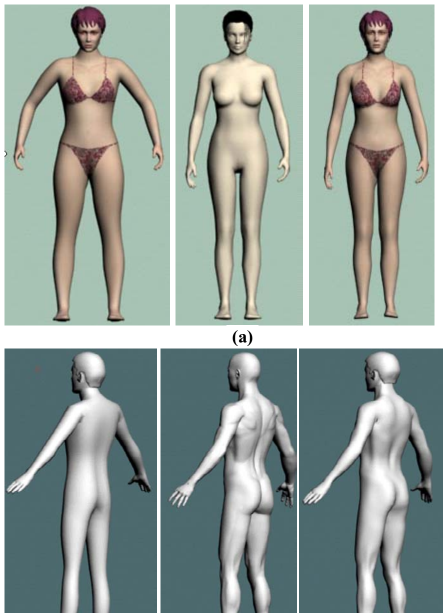


Figure 1: Our modeller converts topology as well as estimates joint centers, by conforming previously worked models (left) to the target surface (middle). Results after the conversion are shown on the right.

3.1. Locating the joint centers

We start by finding the relative position, proportion and the posture of scanned model by finding an appropriate mapping of the reference model to the target. We refer to the transformations applied to each joint of the reference skeleton as joint parameters. When applying different joint parameters during fitting, we use skeleton-driven deformation to calculate the new positions of the skin surface according to the transformation of the underlying skeleton. The most likely joint parameters, that is, scale, rotate, and translation of each bone in the hierarchy, are the ones that drive the template skin such that it best matches the *feature locations* that have been identified prior to mapping. Next, we improve the fitting accuracy by iteratively reducing the shape difference between the template and the target geometry. A curious reader may find details in [20]. Two of the results are shown in Figure 1.

3.2. Reusing the skin to bone data

Once the body shape has been modified through the displacement, the skin attachment data needs to be adapted accordingly so that the model retains smooth skin deformation capability. Generally, the deformed vertex location p is computed as

$$p = \sum_{i=1}^{n} w_i M_i D_i^{-1} p_d$$

where M_i and w_i are the transformation matrix and influence weight of the i-th influencing bone, D_i is the transformation matrix of i-th influencing bone at the time of skin attachment (in most cases D_i's are chosen to be so called dress-pose, with open arms and moderately open legs) and p_d is the location of p at the dress-pose, described in global coordinate system.

Recomputing the skin attachment data involves updating the location p_d for each of its influencing bone. Note that the model has to be back into the dress-pose when the recalculation takes place.

4. Fast skin deformation

4.1. 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 [13], 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 [12][24]. 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 the transformation matrix of *i*th influencing joint, and 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 around bending joints, as shown in Figure 3(a). The artifacts 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.

4.2. Matrix operator approach

We have defined three requirements that the method should fulfil for this particular use:

- The method should overcome the undesirable effect of vertex collapsing. Unlike other skin features such as wrinkles or muscle bugles, which becomes less important to be simulated when they are hidden by garments, this artifact 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.
- 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:

Error! Objects cannot be, created from editing field codes. (2)

with

Error! Objects cannot be created. from editing field codes.

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 2).

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 [2]. The combination of *i* matrices M_i with their weight w_i is given by:

Error! Objects cannot be created from editing field codes. (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 [15], 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.

One drawback of this matrix operator is that its computation takes longer time than the classical SSD, 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. In Figure 3, the result of our skinning method using matrix operator is shown in comparison with the classical SSD.

Error! Objects cannot be created from editing field codes.

Figure 2: Using classical matrix blending leas to degenerated matrix when the relative rotation of two blender matrices becomes close to π.

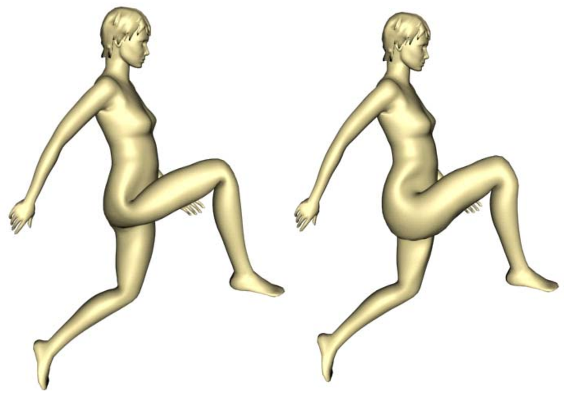


Figure 3: Skeleton driven deformation without (left) and with matrix blending (right).

. Dressing the characters

.1. Cloth modeling

For the garment creation, we have developed is made by using Fashionizer©, MIRALab's brand new virtual cloth design and simulation system [10][23].

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, with the shape of the body model guiding the surface of the cloth as a result of the collision response.

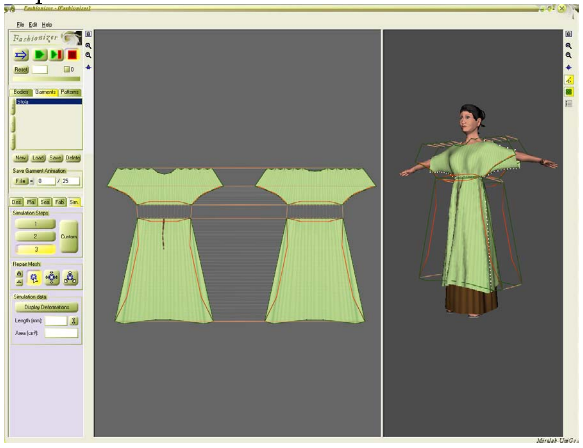


Figure 4: Fashionizer, the garment design software [23].

.2. Organizing the model in the H Anim structure

As mentioned earlier, VHD++, our run-time system, supports H-Anim standard for character representation and thus both the skeleton and the motion data have to be converted promptly. 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 [6]. 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 [10]. The result is an H-Anim compliant human body model as shown in Figure 5.

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.

Error! Objects cannot be created from editing field codes.

Figure : Automatic segmentation of skin mesh, (a) the original body, (b) the skeleton, (c) the segmented body, ready to be exported.

.3. Real time animation of cloth

The real-time animation has been integrated to the garment design tool to provide a preview of the clothes. Given an animated body and a garment whose rest shape has been computed with the garment design tool, this module provides the real-time visualization of the garments on the animated body. This module simulates in real-time the approximate behavior of garment movement at any time during the early design phase, drastically helping the user in prototyping the quality of their clothes under consideration.

.3.1. Garment Preprocessing. Simulating garments in real-time requires drastic simplifications of the simulation process to be carried out, possibly at the expense of mechanical and geometrical accuracy. Our approach, described by Cordier et al [9] is based on a hybrid method where the cloth is segmented into various sections where different algorithms are applied. 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, 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. Thus, we segment the cloth into three layers that we define as follows (Figure 7):

- *Layer 1* (Stretch cloth): Garment regions that stick to the body with a constant offset. In this case, the cloth follows exactly the movement of the underlying skin surface.
- *Layer 2* (Loose cloth): 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.
- *Layer 3* (Floating cloth): Garment regions that flow around the body. The movement of the cloth does not follow exactly the movement of the body. Collisions are not predictable; for a long skirt, for instance, the left side of the skirt may collide with the right leg during animation.

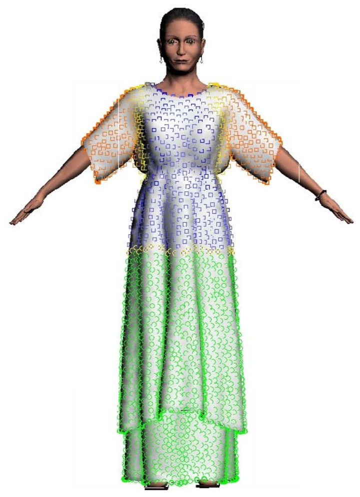


Figure 6: Segmentation of garments

These three categories are animated using three different cloth layers. The idea behind the proposed method 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 garments are divided into a set of segments and the associated simulation method is defined for each.

The segmentation process dispatches each garment region to its adequate layer. From the garment in its rest shape on the initial body, the distance between the garment and the skin surface is used to determine to which category the cloth triangles belong. Associated with each segment are distances from the skin surface that are used to determine the category. Each segment falls into one of three categories: tight, loose and floating clothes. Cloth vertices that are located closely to the skin surface belong to the first or the second layer. Cloth vertices that do not collide with any skin surface belong to the third layer.

.3.2. Garment Animation. In [9], techniques for real-time clothes simulation are proposed. Garment vertices are animated with three different methods, depending on which layer they belong that is defined during the pre-processing stage.

Tight clothes in *Layer 1* follow the deformation of the underlying skin. These deformations are calculated thanks to the mapping of the attachment data of the skin to the garment surface.

For *Layer 2* that is composed of loose clothes, the relative movements of clothes to the skin remain relatively small, keeping a certain distance from the skin surface. Consider the movement of sleeve in relation with 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 severely limited. A basic assumption made is that the movement of the garment largely depends on that of the underlying skin and yet it should 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), the other one for the deformation of cloth on the trunk. Cloth vertices on the limbs are enclosed in half spheres that are attached to the skin surface. Vertices inside these spheres are displaced with the equation of the rigid body motion. A function defines the diameter of the spheres depending on the relative position of the cloth vertex to the normal of the skin surface.

Cloth vertices located on the trunk are animated with a rough mesh. This rough mesh is animated with a physic-based method. The cloth mesh is deformed with the FFD method using the position of the vertices on the rough mesh.

Layer 3 is composed of vertices that freely float around the body. This will take care of cases, such as a large skirt floating around the legs. Any part on this skirt can collide with any part of the leg. The simulation of this layer uses a classical approach with particle system and collision avoidance.

6. Implementation detail

In this section we give an overall description of the modeling pipeline, featuring our tools developed for CG artists who create the contents for VR applications.

6.1. BodyManager

CG artists use authoring tools to provide the content of real-time simulations: virtual human models, objects in the scene, motion data, etc. In our work, this modeling for creating virtual humans is partially based on a commercial graphics package [1], taking advantage of a broad range of features offered while avoiding redundant developments. Instead, we concentrate on improving or filling the missing features of these packages. A handful of plug-ins have been developed (as a whole named as BodyManager package), which are listed in Table 1. It an implementation of all functionalities that are required to transform virtual human modeling to the contents of real-time applications. As a result, the user can avoid repetitive tasks and profit from the seamless integration of heterogeneous tools throughout the pipeline.

Table 1: List of plug ins comprising the BodyManager package.

Plug-in name	Plug-in type	Description
FastSkin	modifier	Performs skeleton-driven deformation using the imported attachment data from B*onesPro*©.

Feature Points	modifier	Assists users to identify feature points and contours on a geometric mesh object of *EditMesh* type.
ScanFitter	utility	Based on the feature points defined by the *FeaturePoints* modifier, and on the skeleton-driven deformation by the *FastSkin* modifier, *ScanFitter* finds the global proportion and posture of the template model that best fits the target (scan) body model.
Offset Deformer	modifier	When used after the *ScanFitter*, *OffsetDeformer* captures the fine detail of the target scan model and save it as a displacement map.
Quad/Tri Rearrange	modifier	Reorganizes the vertex coordinate of geometric object of *EditMesh* type such that the quadratic elements precede triangular ones. It is primarily designed to separate and/or modify the head, hands and the feet parts from the body.
Body Manager	utility	Offers various functionalities to convert and export body models into *H-Anim* standards. Also offered are the motion data import, export and conversion tools.
Spline Surface	modifier	Performs Bezier subdivision on the target geometric object of *QuatPatch* type.
Shape Interpolator	utility	Implements the *Modeling Synthesizer* and *Modifier Synthesizer* described in this dissertation.

6.2 Workflow

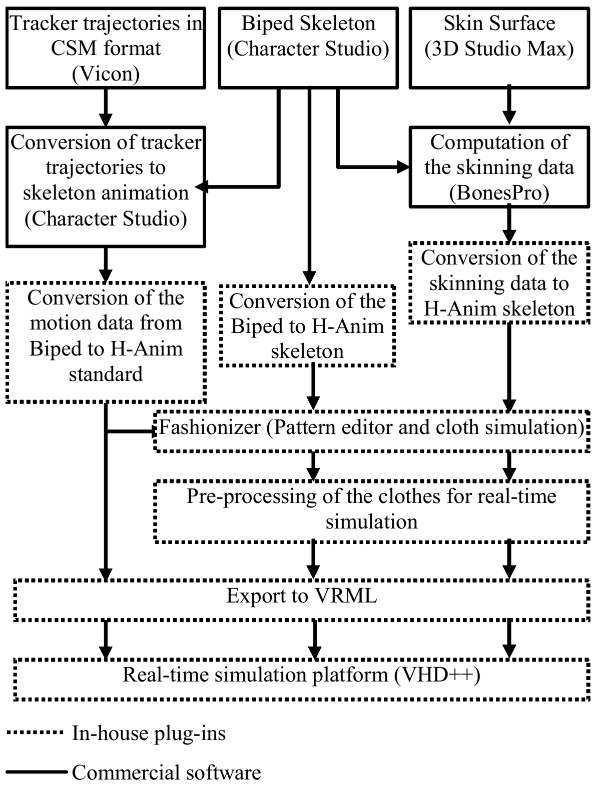
We briefly summarize the pipeline of the production, which largely comprises four steps. At the first step, the skeleton hierarchy is constructed. Character Studio [8] is the primary tool used in this task. Several plug-ins have been developed to convert the Character Studio skeleton (namely Biped) to the H-Anim standard skeleton.

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 [22]. 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 [6]. 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 modeling and the pre-processing of the clothes. The patterns are drawn and placed around the body. Prior to exportation into files, the cloth surface is discredited 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. This involves the segmentation, skin data generation by reusing the skin, and exportation into file. Additional works can be done to enhance the realism of the human model such as facial animation [16] and hair simulation.

Since the Character Studio is the main commercial tool in the production pipeline for the creation of the skeleton and motion editing, and since the real-time platform supports H-Anim standards, most of the in-house plug-ins have 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 motion data from Biped to H-Anim standard
Conversion of the Biped to H-Anim skeleton
Fashionizer (Pattern editor and 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 7: Overview of the production pipeline.

6.3 VHD++ system

Modern 3D VR systems relay heavily on the interplay of heterogeneous technologies. Because of that inherently interdisciplinary character VR domain can be viewed as a melting point of various technologies which although complementary are non-trivial to put together. Many of those technologies gain a lot of individual R&D interest but still it is not generally understood and there are barely any accepted guidelines and approaches in the matter of integration of those functional artifacts under the roof of *one consistent framework*. In other words we have nowadays many masterpieces of atomic technologies but still we miss a well understood and generally accepted strategy for putting them up so they would constitute *the whole bigger than the simple sum of its*

parts. The missing element is an open source *glue framework* which would curb the complexity and make the resulting system machinery a consistent and seamless unity, leaving at the same time open handles and hooks for replacements and extensions.
In order to proceed with the rapid development of such demanding high performance interactive immersive VR application, featuring advanced virtual human simulation technologies we adopted the VHD++ real-time framework as described by [17] (illustrated in the following figure).

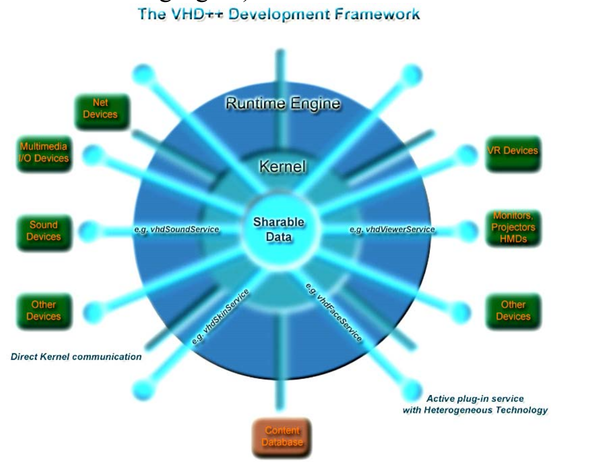


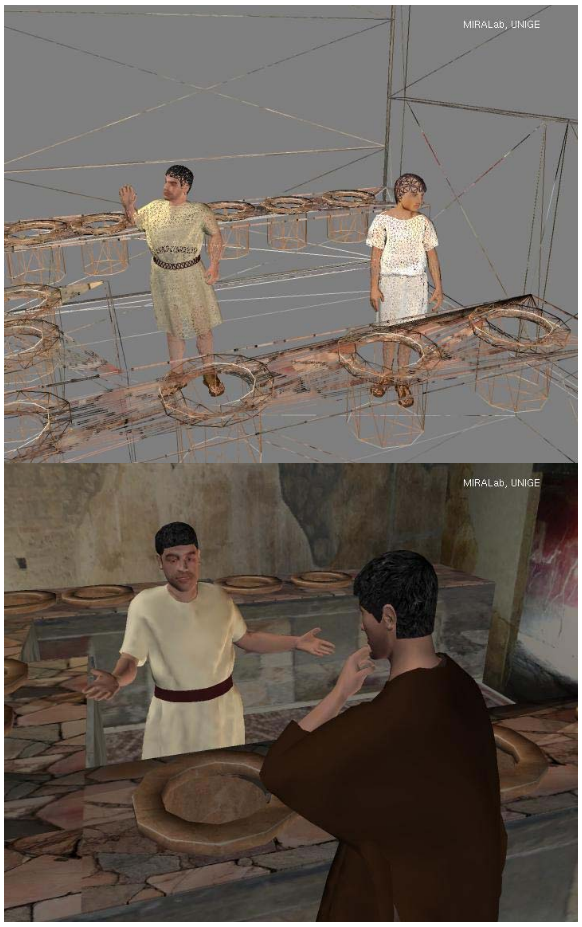
Figure 8: Concept of a generic VHD++ runtime engine with components (plug ins)

VHD++ addresses the most common problems related to the development, extension and integration of heterogeneous simulation technologies under a single system roof while combines both framework (complexity curbing) and component (complexity hiding) based software engineering methodologies. As a result a large scale architecture and code reuse is being achieved which radically increases development efficiency and robustness of the final VR and virtual character application. Figure 9 illustrates a graphical overview of the VHD++ framework. Without the adoption of such a framework it would have been extremely time consuming and technically not guaranteed that we would result in such a VR application featuring all these real-time heterogeneous technologies supported by the necessary processing tools: a) immersive 3D real-time graphics, b) immersive 3D sound, c) VR interaction, d) virtual human animation with skinning, e) real-time cloth simulation with body resizing, f) facial animation, emotion and speech.

7. Case Study: Revival of Life in Ancient Pompeii

With the help of the Superintendence of Pompeii [18], who provided us with all necessary archaeological and historical information, we have selected the 'thermopolium' (tavern) of Vetutius Placidus as the main site. We contacted our experiments reconstructing a typical scenario-based short theatrical play, involving various ancient Pompeian characters, based on a script provided by the historians and archaeologists [18]. For the integrated scenario, the python scripting mechanism of the [17] framework was utilized.

In Figure 9 we provide examples of the integrated virtual reality demonstration, featuring integrated real-time clothed virtual humans with motion captured based body animation, modeled as described in Section 3. The overall demonstration is running on a P4 Windows2000 workstation with NVIDIA uadro4 900 XGL graphics card, yielding performance of 25-30 frames per second, depending on the number of actors concurrently appearing in the scene (according to the scenario these vary from 1 till 3 humans and the user viewpoint (as view frustum culling is operating in the scene).



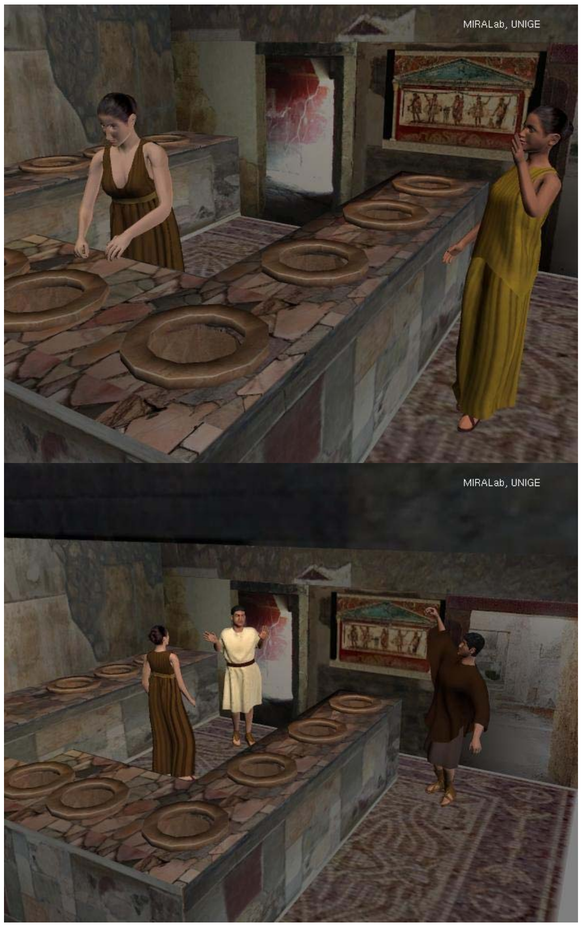


Figure 9: Final integrated results.

8. Conclusion

We have illustrated the process by which the bodies and clothes can be modeled and simulated with low CPU costs by exploiting fast skin deformation techniques. Given our results, we believe that the presented body modeling and cloth animation techniques are instrumental in increasing the level of realism in modern real time applications.

9. Acknowledgments

We would like to thank our designing team – Marlene Arevalo and Nedjma Cadi for their intensive tests and uses of the production tool towards beautiful outcomes and deliverable for the EU IST project LIFEPLUS, which has funded this work through the Swiss Federal Office of Science and Education (OFES).

10. References

[1] 3D Studio Max, http://www.discreet.com.
[2] 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.
[3] B.Allen, B. Curless, Z. Popovic, "Articulated body deformation from range scan data", *Proc. ACM SIGGRAPH*, Addison-Wesley, San Antonio, USA, 2002, pp.612–619.
[4] B. Allen, B. Curless, Z Popovic. "The space of human body shapes: reconstruction and parameterization from range scans", *Proc. SIGGRAPH '03*, pp.587–594, Addison-Wesley, 2003.
[5] A. Aubel, "Anatomically-Based Human Body Deformations", PhD dissertation, Computer Science Dept., EPFL.
[6] BonesPro, a plug-in to 3DS Max, Digimation, http://www.digimation.com/.
[7] J. E. Chadwick, D. R. Haumann and R. E. Parent, "Construction for Deformable Animated Characters", Proc. SIGGRAPH '89, Addison-Wesley, pp.243-252. 1989.
[8] Character Studio, http://www.discreet.com.
[9] F. Cordier, N. Magnenat-Thalmann, (2002), "Real-time Animation of Dressed Virtual Humans", Computer Graphics Forum, published by Blackwell Publishers, 21(3), September 2002.
[10] Fashionizer, MiraDreams, http://www.miradreams.com.
[11] H-Anim standard, http://www.h-anim.org.
[12] 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.
[13] 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.
[14] Maya Alias, http://www.alias.com.
[15] 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.
[16] S. Kshirsagar, N. Magnenat-Thalmann, "Visyllable Based Speech Animation", Proceedings of Eurographics, Granada, Spain, Blackwell Publishers, pp. 631-639, vol. 22(3), 2003.
[17] M. Ponder, G. Papagiannakis, T. Molet, N. Magnenat-Thalmann and D. Thalmann, VHD++ Development Framework: Towards Extendible, Component Based VR/AR Simulation Engine Featuring Advanced Virtual Character Technologies, IEEE Computer Society Press, in proceedings of CGI 2003, pp. 96-104, , 2003
[18] POMPEII: Archaeological Superintendence of Pompeii: http://www.pompeiisites.org.
[19] Poser, Curious Labs, http://www.curiouslabs.com/.
[20] H. Seo and N. Magnenat-Thalmann, "An Automatic Modeling of Human Bodies from Sizing Parameters", ACM SIGGRAPH 2003 Symposium on Interactive 3D Graphics, pp.19-26, Monterey, USA, 2003.
[21] H. Seo and N. Magnenat-Thalmann, "An Example-Based Approach to Human Body Manipulation", Graphical Models, Academic Press, Vol. 66, No. 1, pp.1-23, January, 2004.
[22] Vicon motion capture system, http://www.vicon.com.
[23] P. Volino, N. Magnenat-Thalmann, "Comparing Efficiency of Integration Methods for Cloth Animation", Proceedings of CGI'01, Hong-Kong, IEEE Press, July 2001.
[24] J. Weber, "Run-Time Skin Deformation", proceedings of Game Developers Conference 2000, http://developer.intel.com/ial/3dsoftware/mrm.htm.