

# Human Figure Synthesis and Animation for Virtual Space Teleconferencing

Karansher SINGH<sup>\*,†</sup>

Jun OHYA<sup>\*</sup>

Richard PARENT<sup>†</sup>

\* ATR Communication Systems Research Laboratories  
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-02, Japan

† Dept. of Computer & Information Science, The Ohio State University  
2036 Neil Avenue, Columbus, OH 43210, USA

## Abstract

*Human figure animation is a widely researched area with many applications. This paper addresses specific issues that deal with the synthesis, animation and environmental interaction of human figures within a virtual space teleconferencing system. A layered representation of the human figure is adopted. Skeletal posture is determined from magnetic sensors on the body, using heuristics and inverse kinematics. This paper describes the use of implicit function techniques in the synthesis and animation of a polymesh geometric skin over the skeletal structure. Implicit functions perform detection and handling of collisions with an optimal worst case time complexity that is linear in the number of polymesh vertices. Body deformations resulting from auto-collisions are handled elegantly and homogeneously as part of the environment. Further, implicit functions generate precise collision contact surfaces and have the capability to model the physical characteristics of muscles in systems that employ force feedback. The real time implementation within a virtual space teleconferencing system, illustrates this new approach, coupling polymesh and implicit surface based modeling and animation techniques.*

## 1 Introduction

An important application of human figure animation is the evolving area of virtual space teleconferencing [8]. At ATR CSRL, the authors are building a **VI**rtual **S**pace **TE**leconferencing (*VISTEL*) system [8], aimed at an environment where teleconference participants at different sites can feel as if they are all at one site, allowing them to hold meetings and work cooperatively. In *VISTEL*, 3D models of the participants at different sites are combined into 3D virtual space images, giving participants the sensation of meeting each other

in a common space (See Figure 7). Real time, realistic reproduction of motion of participants on their 3D models is important for good communication. Additionally, cooperative work requires efficient and robust handling of interaction of the human models with the virtual environment.

A layered approach [2] to the modeling and animation of articulated figures is a widely adopted methodology. With respect to human figures, the layers may be broadly classified into skeletal, muscle and skin, and clothing layers. Layers are often omitted, collapsed or further subdivided, depending on the sophistication of the model.

The **skeletal layer** is usually approximated as an articulated rigid body. Despite the problems arising from the approximations [5], realistic results may be obtained with such a model. In *VISTEL*, an articulated rigid skeleton is used. The skeleton is animated based on the posture of the teleconference participant obtained from magnetic sensors on the body.

Various techniques have been proposed for the modeling and animation of the **muscle, skin layer** in real time [2], [6],[16]. Deformations during animation to a geometric (typically polymesh) skin are specified empirically or based on an underlying muscle model.

The deformable nature of human muscle, fatty tissue and skin is described in [16], with respect to facial animation. A spring and damper mesh muscle, tissue model, attaches skin to the underlying skeleton and iteratively applied forces shape the skin. In *VISTEL*, real time facial animation is realized by visually tracking tape marks on the faces of the participants [8]. The tape mark positions drive deformations of the facial polymesh.

Free form deformations are used to empirically deform the skin layer in [2] based on joint angle values. Position of the skin polymesh around joints in terms

of a specific function local to the skeletal joint area is presented in [6]. There is no explicit muscle model in either case.

The muscle and skin layer is very important for visual realism and its effect on subsequent layers. Further, tight fitting apparel may be modeled as textures applied to the geometric skin. This is the approach adopted in *VISTEL*. The specific requirements virtual space teleconferencing imposes on this layer are the following:

1. Realistic appearance and efficient display of geometric skin for realistic skeletal postures.
2. Efficient collision detection and deformation of the skin with the environment (and rest of the body), the importance of which is increased by tolerable inaccuracy in posture computation.
3. Efficient modeling and computation of forces and contact areas for systems with force feedback.

The paper thus focuses on addressing the above requirements for this layer by combining implicit function and polymesh based techniques.

**Implicit functions** are a popular approach to the modeling and animation of physically deformable objects [18],[1], [10],[7],[3]. An implicit surface is defined as the set of points  $P$  satisfying an equation  $F(P) = 0$ . Complex shapes may be intuitively built and animated using simple implicit primitive shapes [18],[14]. Superquadric primitives [10] and blobs [7], can be fitted to real 3D data. Implicit functions are used to model exact collision contact surfaces in [3]. Implicit functions can replace discrete spring models with continuous stiffness fields [3]. Efficient implicit function evaluation facilitates efficient collision detection between objects [11].

An implicit function based model for a geometric skin with a physical interpretation for a muscle model [12], shows the effective use of implicit surfaces in modeling and animating human muscle and skin. Implicit surface representations, unfortunately, have inefficient display characteristics. To realize a teleconferencing system of the complexity and realism desired, with current computing resources, it is thus imperative that we make use of polygon based representations, which have extensive hardware support.

This paper presents an implicit function based muscle model that deforms a polymesh geometric skin. The model possesses the advantages of implicit functions [12] as well as the display efficiency of a polygon based structure. Its modular nature makes it simple

to integrate into existing systems and unify with other polymesh based techniques [2],[6].

The physical characteristics of objects are separated into **rigid** and **deformable** components. The implicit function based deformable component performs all physically based deformations including efficient collision detection and handling. For human figures, a geometric polymesh skin is first synthesized using digitized limbs blended together using implicit functions. The skin is then embedded in a hierarchy of implicit functions that model muscles. These implicit functions interact with one another and with other implicit functions in the environment, to model blends and collision deformations. Human figures (and other objects) are animated by rigid body transformation of the polymesh model on a skeletal structure and subsequent deformation based on contributing implicit functions (See Figure 4).

Section 2 describes the synthesis of the polymesh skin model and its animation as an articulated rigid body. A muscle model embedding the polymesh skin in a hierarchy of implicit functions is proposed in Section 3. Section 4 provides the working details of the deformable component. The interaction of the implicit functions for blending and to efficiently detect and model collision deformations on the underlying polymesh model is presented. Section 5 describes the implementation of the model within *VISTEL*. Section 6 presents conclusions and scope for future work.

## 2 Human Polymesh Model : Synthesis and Animation

**Synthesis** of polymesh models of real humans is a nontrivial problem. Various reconstruction methods, using sculpted models, range data, photographic images have been proposed [9]. In our approach a number of polymesh parts corresponding to various limbs are obtained, conveniently sought using a Cyberware digitizer [8]. These parts are then fitted together, which may be done by lofting between the end contours of segments. We choose, however, to blend the parts, juxtaposed in space [12]. This better preserves the overall length and shape of the limbs. Further, control over the region of the blend can help automatically attenuate glitches and noise in the scanned data that often results at the fringes.

Results are shown on a polygonized elbow in Figure 3. The upper and lower digitized limbs are treated as implicit surface primitives and blended together as described in [12]. A user controllable region is then defined within which the blended data is polygonized and seamlessly connected to the existing polymesh struc-

ture of the limbs. Color textures (skin and clothing) for the polygonized region maybe obtained by blending the two limb textures as shown in Figure 3. This provides a polymesh prototype representing the human in some prespecified skeletal posture.

**Animation** of the polymesh model based on motion of the underlying skeletal structure must then be specified. Posture computation for teleconferencing is in itself a nontrivial problem [8]. For this paper we assume skeletal posture computation to some reasonable level of accuracy.

Rigid bodies are well represented by polymesh models and can be animated efficiently and robustly. Various techniques, such as springs and dampers [15], free form deformations [2], and implicit primitives [10], [3] have been used to model deformable objects. This paper separates the physical characteristics of objects into **rigid** and **deformable** components that may be applied successively [3]. Specifically, the rigid component is determined by a skeletal structure; the deformable component by the interaction with other objects in the environment. This model is particularly well suited to human figure animation. We wish to model the deformable component using implicit functions, which facilitate an efficient computation of interaction of the deformable body with the environment.

For the rigid component it suffices to subject the polymesh model to the translations and rotations specified by the underlying skeleton. Care is taken to preserve connectivity and continuity around joints. This may be done using free form deformations [2]. Alternatively the quaternion based rotation around a joint is interpolated for vertices in the polygonized region (See Figure 3) around a joint [13]. This maintains smooth connectivity around joints but the animated model resembles a flexible pipe (See Figure 8).

The next two sections of the paper develop a model for the deformable component, that is applied to the polymesh model after the rigid component transformation described above. Figure 4 shows the overall perspective of such a system.

### 3 Deformable Model : Synthesis

The deformable model immerses the polymesh model in a hierarchy of functions, specified by a number of implicit primitives [18]. The vertices of the polymesh prototype are then calibrated based on their implicit function values, so that they all lie on some convenient implicit surface. During animation the vertices are transformed appropriately to stay on that implicit surface. The next subsection describes implicit function concepts crucial to the understanding of the proposed

model.

#### 3.1 Implicit Function primitives

A useful set of implicit surfaces can be generated as an algebraic combination of polynomial functions each of which is defined over a finite volume. For a summation (smooth blend)  $F(P) = \sum F_i(P) - T$ , where  $i$  runs over the primitive polynomial functions  $F_i$  and  $T$  is a threshold value  $\in [0, 1]$ . A subset of these surfaces are **distance surfaces** and **convolution surfaces** [1]. Each primitive is defined by a finite volume  $V$  (typically spherical), a skeleton within the volume  $S$  (typically the center of the sphere) and a function  $f$  (typically a polynomial as in Figure 1). The primitive only contributes within  $V$ . An example of  $f : [0, 1] \rightarrow [0, 1]$ , also called a density function, with the desired properties [18] is shown in Figure 1. For a point  $P$  within  $V$  the function value is determined by first computing a value  $\in [0, 1]$  called a distance ratio of  $P$ .  $F(P) = f(\text{distanceratio}(P))$  is the function value for the primitive. The distance ratio is computed by taking the ratio of the shortest euclidean distance from  $P$  to a point  $Q$  on  $S$ , and a value determined by the shape of  $V$  (usually a constant radius value  $R$ ). Such a primitive (See Figure 1) defines an **offset surface** [1].

Different objects have their own implicit functions that determine the object surface. These objects may interact by imparting additional implicit functions to one another [3]. The functions imparted can model interactions such as collision deformations as described in Section 4.1 (See Figure 6,8,9), fusion and fission [12].

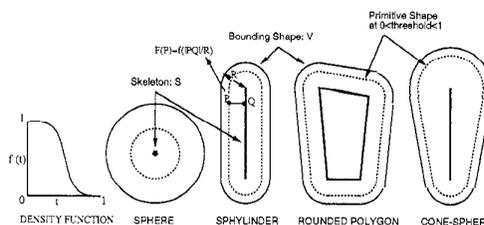


Figure 1: Implicit Primitive Shapes

#### 3.2 Implicit Primitive Model

An implicit primitive is attached to every skeletal limb and embeds the corresponding polymesh model around it. A convenient threshold value  $T$  (See Section 3.1) is chosen at which the implicit surface deforming the polymesh model will be sought. The choice of  $T$  is influenced by the deformable characteristics of the

object being modeled.

The shape of the implicit primitive [18] provides a bounded volume for the realm of influence of the associated function, as well as a mapping from a point within the volume to a value within the domain of the density function (to calculate the function at that point) [13].

The choice of the implicit primitive shape for an object (limb) is influenced by the following requirements :

1. Implicit function computation should be efficient.
2. The primitive shape at some threshold value should fit the embedded region of the polymesh model well [7]. The behavior of the polymesh region then closely follows that of the surface defined by the implicit primitive at that threshold.
3. The primitive bounding volume around the embedded polymesh region should be reasonably tight, so as to avoid wasted polymesh deformation computation on interaction with the environment.
4. Bounding volume intersection computation between the primitives used should be efficient, so as to determine primitives in the environment with which a given primitive interacts.

Primitive volumes satisfying these requirements well for different human parts are offset surfaces (spheres, sphynders, offset polygons) and conespheres [1],[14] (See Figure 1). Efficient computation methods for these primitives are described in detail in [14]. The choice of primitive shapes for various parts of the body is fairly intuitive and shown in Figure 2.

Figure 2 also shows the hierarchical implicit primitive structure used for the human figure. The implicit primitives are combined selectively as specified in [12] using a combination graph. Primitives that do not blend mutually, are treated as different implicit objects in the environment, that interact with each other for collision detection and deformation. Thus auto-collisions between different parts of the body are homogeneously handled along with any other implicit objects in the environment. Joint regions such as the elbow where both blending and collision deformations occur, are handled by the introduction of a dummy primitive at the joint that blends the two arm primitives together (See Figure 8).

Human bones on animation, often come close to the skin surface and define the shape of the skin [5]. This is a major shortcoming of virtual point-linked skeletons

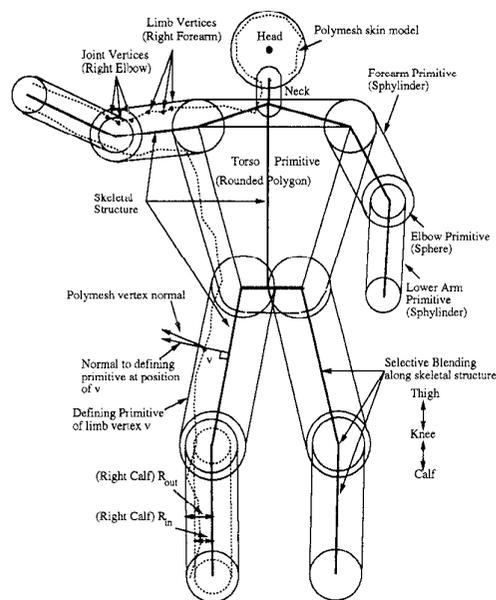


Figure 2: Implicit Primitive Hierarchy

and is handled elegantly by the implicit model [12]. Skeletal implicit primitives are specified and blended with the geometric skin primitives. On animation, the skeletal primitives contribute to the shape of the skin automatically, only when the primitive is close to the geometric surface primitive (See Figure 8). Other features like veins and wrinkles may be similarly incorporated [12].

### 3.3 Implicit Primitive Synthesis

Once an appropriate primitive shape for a limb or joint is selected, it must be fitted to the underlying polymesh data. The polymesh model synthesis, described in Section 2, partitions vertices into those belonging to different limbs and polygonized regions corresponding to joints. Vertices are henceforth referred to as **limb vertices** or **joint vertices**. Limb vertices contribute to the fitting of the corresponding limb primitive. Joint vertices contribute to the joint primitive and the primitives of the limbs joined. The primitives which a vertex contributes to are also called the **defining primitives** for the vertex and are largely responsible for its deformation on animation.

For offset surface primitives, skeletal joint centers are used to position the primitive skeleton  $S$ , as shown in Figure 2. Two radii  $R_{out}$  and  $R_{in}$  are then calculated for each offset primitive. These are based on the distances of contributing polymesh ver-

tices from the primitive skeleton  $S$ .  $R_{out}$  is the maximum and  $R_{in}$  the average of distances of contributing vertices from  $S$ .  $R_{out}$  is the bounding radius of the primitive and  $R_{in}$  the primitive of best fit.  $W = T/f(R_{in}/R_{out})$  is the shape weight for the primitive, where  $f$  is the associated density function. The function value for primitive  $i$  at a point  $P$  is then  $F_i(P) = W_i * f_i(distanceratio(P))$ . Thus the shape at threshold  $T$  of the primitive is the offset surface with radius  $R_{in}$ . Figure 5 shows primitives fitted to a human polymesh model. Bounding volumes for the right half of the body and primitives of best fit for the left half are shown. Complex primitives such as conespheres or superquadrics may be fitted as described in [7],[10],[13].

Once the implicit primitives have been synthesized, the polymesh model needs to be calibrated. Here a weight attribute for each vertex is calculated, ensuring that the implicit function model produces no deformation to the polymesh prototype for the prespecified posture, in the absence of any environmental interaction.

### 3.4 Polymesh Model Calibration

For a vertex  $v$  on the prototype polymesh, let  $D_v(P)$  be the implicit function computed based on its defining primitive(s) at a point  $P$  in space.  $v$  is calibrated by assigning a weight  $w_v = T/D_v(P)$ , where  $P$  is the spatial position of the vertex on the prototype polymesh. In the absence of environmental interaction the implicit function for the vertex at  $P$  is,  $F(P) = w_v * D_v(P) = T$ , ensuring that the polymesh prototype for the prespecified posture lies on the implicit surface determined by the implicit model at threshold  $T$ .

This completes the construction of the implicit model and polymesh calibration for the human figure model. A similar approach may be taken for any polymesh object, articulated or otherwise, as long as implicit primitive(s), with the required attributes, can be sought.

## 4 Deformable Model: Animation

The result of the rigid component transformations to the polymesh prototype and implicit primitives based on the skeletal posture, described in Section 2, forms the input to the deformable component model (See Figure 4).

The individually manipulated implicit primitives automatically maintain a smoothly blended body as well as collision deformations [12]. These functions then appropriately shape the embedded polymesh

model. Deformation of the polymesh models is carried out by deforming the position of each vertex of the model from its current position  $P$  to a point  $P'$ , such that  $F(P') = T$ .

### 4.1 Implicit Function Computation at Polymesh Vertices

The implicit function for a vertex  $v$  at point  $P$ , based on its defining primitive(s),  $D_v(P)$  is computed as follows :

**Limb Vertex:**  $D_v(P) = F_i(P)$ , where  $i$  is the defining limb primitive.

**Joint Vertex:**  $D_v(P) = F_i(P) + DIFF(F_j(P), F_k(P))$ , where  $i$  is the joint primitive and  $j, k$  the limb primitives. The  $DIFF$  function models the formation of creases at joints (See Figure 8). An example of  $DIFF$  is  $DIFF(a, b) = |a^n - b^n|^{1/n}$ .

The implicit function for  $v$  at  $P$  is then  $F(P) = w_v * D_v(P) + \sum_i COLL_{i,v}(P)$ , where  $i$  runs over all interacting primitives (where  $F_i(P) > 0$ ).

$COLL_{i,v}(P)$  is the collision deformation function [3] imparted by primitive  $i$  with a minor difference:

**Penetration zone:**  $COLL_{i,v}(P) = T - w_v * F_i(P)$ .

**Propagation zone:**  $COLL_{i,v}(P) = w_v * Prop_i(P)$ , where  $Prop_i(P)$  is the propagation function in [3].

Both  $DIFF$  and  $COLL$  are derived from collision deformation concepts presented in [3]. Precise mathematical examples and details on the behavior of these functions may be found in [13]. A modification to the function handles degrees of relative deformability of colliding objects [13]. Temporal modification of the functions handle various elasticity characteristics [13] (See Figure 9). Multiple objects mutually colliding are handled homogeneously [3].

The vertex  $v$  is constrained to a point  $P$ , where  $F(P) = T$ , or for example in the presence of a colliding primitive  $i$ , the common collision contact surface where  $D_v(P) = F_i(P)$ . Proof of correctness of the generated collision contact surfaces may be found in [3],[13].

### 4.2 Polymesh Deformation

Having laid the theoretical foundation that deforms the polymesh model, we address the algorithmic aspects for a practical implementation. The algorithm for deformable component transformation of objects is carried out in 3 steps as follows:

1. A list of interacting objects is constructed for each object in the environment.
2. The vertices of the polymesh model of each object are then deformed based on function values

computed using its defining primitive(s) and all objects the defining primitive(s) interact with.

3. Further processing using the deformed polymeshes.

*Step 1* benefits from efficient intersection computation of the bounding volumes of primitive shapes. It is an optimization step based on spatial coherence that does not affect the polymesh deformation in *Step 2*. A simple analytic solution [14] exists for intersection testing between simple shapes like offset surfaces, which may be exhaustively intersected with each other. Spatial subdivision techniques such as octrees may also be used.

*Step 2* is the crux of the algorithm. Every vertex of a polymesh object must now be deformed based on an implicit function  $F$ . This function is specific to the vertex and is calculated as described in Section 4.1. The position of this vertex must be deformed from its position  $P$  computed by the rigid component transformation, to a point  $P'$ , such that  $F(P') = T$ . As the implicit function defines a continuous implicit surface in the neighbourhood of  $P$ , the deformation mapping of  $P$  to  $P'$  for the vertex is ill-defined. This is an artifact of using a discrete polymesh representation to follow the deformations of a continuous implicit representation. The solution proposed is to deform  $P$  along its vertex normal. Alternatively,  $P$  maybe deformed along  $\nabla F(P)$  or the normal to the implicit function at that point (See Figure 2).

For offset primitives, the distance ratio function for any point along a ray can be represented in terms of its parametric distance along the ray [14]. Thus  $P'$  may be obtained analytically along any ray, by solving a sequence of quartic equations [14]. Alternatively, a hybrid Newton-Raphson, Regula Falsi iterative technique [14] may be used which is efficient in this case, as the deformations, being incremental are small.

*Step 3* deals with the computation of surface normals or other spatial attributes of the polymesh or any desired parameters based on the deformed polymesh. Additionally, for systems with force feedback, integration of forces at individual vertices of the object and collision contact area computation may be done in this step.

The worst case complexity of the above algorithm is  $O(m^2 + mn)$ , where  $m$  is the number of implicit primitives and  $n$  the number of polymesh vertices in the environment ( $n \gg m$ ). Despite fast octree based techniques for coarse collision detection, precise detection and handling using conventional polygon methods is  $O(n^2)$  in the worst case [11],[4], making our approach superior as the complexity of virtual worlds increase.

### 4.3 Dynamics

A physical muscle interpretation, which in conjunction with the skeleton controls the animation of the geometric model, is presented in [12]. Variations in tissue characteristics [16] are modeled by piecewise smooth density polynomials whose gradient reflects the change in stiffness, making reaction force computation for a vertex simple [13]. Area computation of polymesh faces whose vertices are deformed to lie on a collision contact surface [3] may be done in *Step 3* of the algorithm in Section 4.2. These areas can then be used in the computation of area dependent reaction and friction forces completing a force feedback loop.

## 5 Implementation

Figure 7 shows the *VISTEL* system in operation. Posture computation employs 4 Fastrak magnetic sensors (head, chest and wrists) and cyber gloves on each participant. SGI (Onyx, Reality Engine) machines perform graphics processing and display at each site. Real time 3D display of the synthesized virtual space is then projected on a 70-inch stereoscopic display.

Human figure data is obtained as a number of digitized parts using a Cyberware Color 3D digitizer. The polymesh parts are registered with color textures and fitted together on a virtual skeleton, using implicit function blending (See Figure 3) [12] or other techniques.

The implicit primitives are then constructed on the figure as shown in Figure 5. The human figure polymesh ( $\approx 7500$  vertices) is embedded in 23 implicit primitives. The implicit model is constructed hierarchically, in an object oriented fashion, making the fitting of other polymesh objects as well as the introduction of new implicit primitive shapes, a simple task.

Figure 8 shows the elbow region after rigid component transformation and its subsequent deformable component transformation. Spherical primitives modeling skeletal elbows cause the elbow to protrude in the bent arm and precise crease formation may be seen. The deformable component computation using a Regula Falsi-Newton Raphson approach typically takes 2-3 iterations per vertex.

Figure 6 shows deformations of an arm and a ball on collision. Figure 9 shows a ball bouncing off a head. The head is given a plastic attribute and the ball a viscoelastic one so as to clearly see the deformations that result.

Existing polymesh based muscle and skin modeling techniques may easily be integrated with the rigid component in our implementation. As an example

FFDs [2] on the spine, animate the torso (See Figure 5). Functional calculations for torso vertices, with respect to the torso implicit primitive, employ the undeformed spatial position in the calculations.

## 6 Conclusion

To summarize, this paper describes a model for the synthesis and animation of objects with a polymesh representation. The physical characteristics of the object are separated into rigid and deformable components. The implicit function based deformable component performs optimal collision detection and handling. The model is catered to address issues involved in the modeling and animation of the human muscle and skin layer within a virtual space teleconferencing system.

The implementation of the presented concepts shows their effectiveness both in terms of computation speed and the degree of realism obtained. The separation of the physical characteristics of objects into rigid and deformable components, works particularly well for human figures. The model handles auto-collisions of the body and skeletally based deformations (bone, crease in Figure 8) elegantly. Attribution of physical characteristics can be overlaid on the model for dynamic simulations [13].

The ability to apply implicit function techniques in general to existing polygon based data is an important advantage of our approach. It can unify and be integrated with existing polygon based or implicit surface based modeling and animation systems.

The approach achieves linear time complexity in terms of number of object vertices for collision detection and handling, which is important when dealing with complex virtual worlds [11].

The underconstrained nature of the deformable component mapping of polymesh vertices may cause surface consistency problems. A dense human figure polymesh and the radial nature of the limbs and primitives, causes simple displacements along vertex normals to give good results without vertices bunching together or diverging abnormally. Incorporation of techniques such as [17], that adaptively subdivide and coalesce the polymesh in real time are subject to current research.

There is scope for future work on construction and fitting of primitives to the polymesh. Poor fitting primitives may result in very close objects being deformed to abut at the implicit model contact surface. For *VISTEL* and other applications where visual realism dominates over spatial accuracy (we assume a tolerable inaccuracy in the transition from real to vir-

tual space), the above artifact does not pose a problem. Using a greater number and more complex primitives improves the fit but degrades implicit function and bounding volume intersection computation efficiency. An empirical tradeoff between a better fit and computation efficiency should therefore, be taken into consideration.

## References

- [1] J. Bloomenthal and K. Shoemake. Convolution surfaces. *Computer Graphics*, 25(4):251–256, 1991.
- [2] J. Chadwick, D. Haumann and R. Parent. Layered construction for deformable animated characters. *Computer Graphics*, 23(3):234–243, 1989.
- [3] M. Gascuel. An implicit formulation for precise contact modeling between flexible solids. *Proc. of SIGGRAPH*, 313–320, 1993.
- [4] J. Hahn. Realistic animation of rigid bodies. *Computer Graphics*, 22(4):299–308, 1988.
- [5] N. Magnetat-Thalmann and D. Thalmann. Complex models for visualising humans, *SIGGRAPH Course Notes C20*, 3–10, 1991.
- [6] N. Magnetat-Thalmann and D. Thalmann. Human body deformations using Joint Dependent Local Operators and Finite Element Theory. *Making Them Move*, Morgan Kaufmann, 243–262.
- [7] S. Muraki. Volumetric shape description of range data using the blobby model. *Computer Graphics*, 23(3):234–243, 1991.
- [8] J. Ohya, Y. Kitamura, H. Takemura, F. Kishino and N. Terashima. Real-time reproduction of 3D human images in Virtual Space Teleconferencing. *VRAIS*, 408–414, 1993.
- [9] A. Paouri, N. Magnetat-Thalmann and D. Thalmann. Creating realistic 3D human shape characters for computer generated films. *SIGGRAPH Course Notes C20*, 18–29, 1991.
- [10] S. Sclaroff and A. Pentland. Generalized implicit functions for computer graphics. *Computer Graphics*, 25(4):247–250, 1991.
- [11] A. Pentland. Computational complexity versus virtual worlds. *Computer Graphics*, 24(2):185–192, 1990.

- [12] K. Singh, J. Ohya and F. Kishino. Realistic modeling and animation of a muscle and skin layer for human figures using implicit function techniques. *IPS Graphics and CAD Conference*, 49–56, 1994.
- [13] K. Singh. Human Figure Synthesis and Animation for Virtual Space Teleconferencing. *ATR Technical Report*, 1994.
- [14] K. Singh and R. Parent. Fast scanline processing of useful implicitly-defined shapes: Sphylinders, cone-spheres and rounded polygons. *Ohio State Univ. Tech. Rep. OSU-CISRC-5/94-TR26*, 1994.
- [15] D. Terzopoulos and K. Fleischer. Deformable Models. *Visual Computer*, 4:306–331, 1988.
- [16] D. Terzopoulos and K. Waters. Physically based facial modeling, analysis and animation. *Journal of Visualization and Computer Animation*, Vol.1, No.2, 73–79, 1990
- [17] A. Witkin and P. Heckbert. Using particles to sample and control implicit surfaces. *Proc. of SIGGRAPH*, 269–278, 1994.
- [18] G. Wyvill C. McPheeters and B. Wyvill. Data structures for soft objects. *Visual Computer*, 2:227–234, 1986.

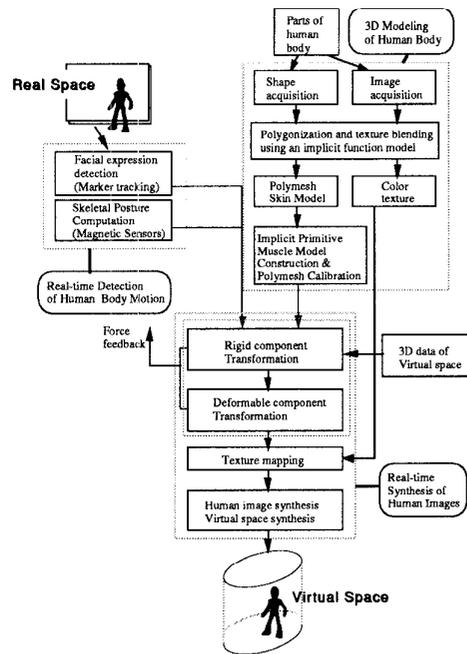


Figure 4: VISTEL Framework

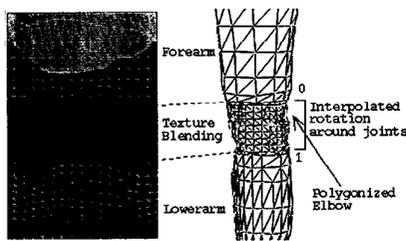


Figure 3: Human Figure Polymesh Synthesis

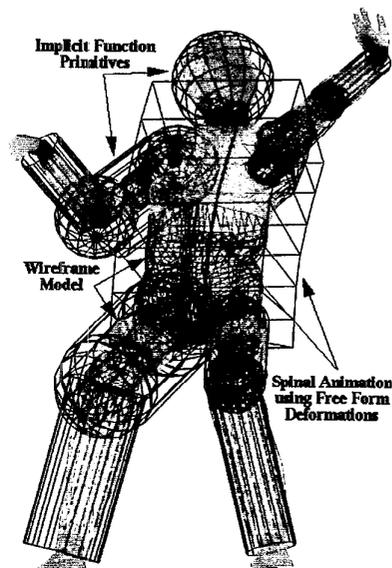


Figure 5: Implicit Primitive Synthesis

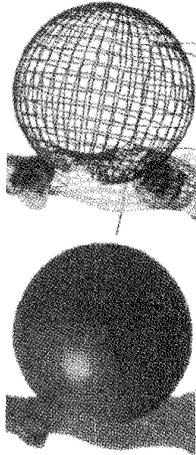


Figure 6: Collision Deformation

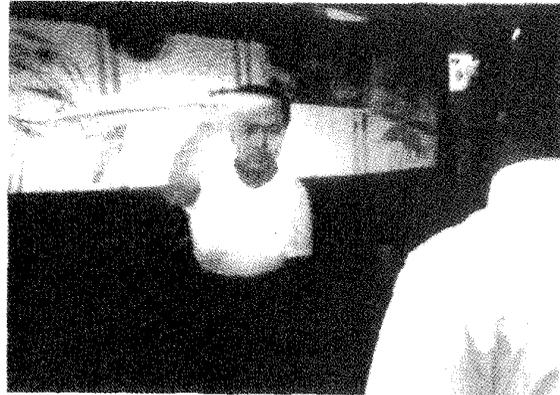


Figure 7: VISTEL System

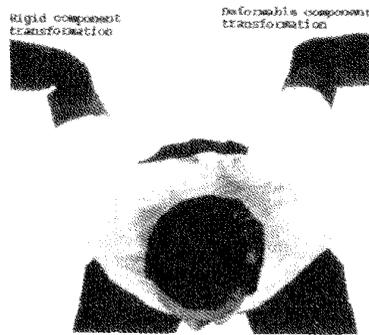


Figure 8: Rigid, Deformable Component Transformation



Figure 9: Temporal Collision Deformations (Head: plastic, Ball: viscoelastic)