An Implicit Surface Prototype for Evolving Human Figure Geometry

Technical Report OSU-ACCAD-11/00-TR2

Matthew Lewis*
Advanced Computing Center for the Arts and Design
The Ohio State University

Richard Parent†
Department of Computer and Information Science
The Ohio State University

Abstract

This paper describes one method of constructing a parametric human model for use in evolving humanoid character geometry. Interactive aesthetic selection is employed as a fitness function for producing successive populations of human models from an initial parametric human model constructed from a hierarchy of implicit surfaces. The parameters controlling the size, position, and orientation of groups of implicit surfaces are discussed, as well as the usage of a genetic algorithm within a 3D animation authoring package.

1 INTRODUCTION

1.1 Overview

Given the rising popularity of online games and avatar-based 3D virtual environments, the need is increasing for software that enables individuals with little or no experience in 3D modeling to generate custom content such as buildings, characters, materials, and objects for personal use in these spaces. In contrast to the usual constructive methods of design, aesthetic evolutionary design (AED) is a process in which a designer is presented with a population of dissimilar computer-generated objects and asked to judge them. The computer then combines members of the current population, taking the user's preferences into account, and generates a new population of design solutions. Evolution occurs as traits selected as desirable by the designer are passed on from ancestors to descendants in further rounds of selection. Images, human figures, architecture, consumer products, and motion have all been evolved by aesthetic evolutionary design (see section 1.2.2).

Constructing a parametric description of an object as a basis for evolution implies the building of a "prototype" model with adjustable parameters. A simple desk prototype, for example, might have parameters such as color, material, height, shape, number of drawers, drawer positions, and so forth. The parameters defined for this prototype would form an N-dimensional space of possible desks, i.e., each point in this space maps to a specific desk model. Instead of searching for a desired model by making small adjustments, slowly tweaking each attribute in turn, AED software gives the designer the ability to browse several regions of the implicitly defined "desk space" simultaneously. This is implemented by displaying a population of possible desk designs, and allowing the user to judge them by specifying which are the most (and/or perhaps the least) interesting. Given the user's critique, the system can then generate a new population of desk design proposals. The designer of the desk prototype does not so much attempt to craft a well made desk, as to select parameter ranges and weights that define a space containing all desks that the designer would like users of the prototype to be able to discover.

For this work, Side Effects Software Inc.'s *Houdini* [35] was used as a 3D software development environment to produce a high-level interactive evolution interface [26]. Houdini's embedded scripting languages and data flow networks provide a development environment flexible enough to enable the rapid construction of arbitrarily complex geometry manipulation interfaces and tools, such as the one presented in this paper.

1.2 Background

1.2.1 Human Figure Modeling

Human figures for animation and computer games are typically modeled by expert users of animation software packages such as *Maya* [2] or *3D Studio MAX* [4]. Ergonomic simulation products such as *Jack* [41], *BMD-HMS* [13], and the *Safework VirtualMan* [34] have all provided the ability to represent humans with highly realistic, percentile-based proportions. These ergonomics packages are generally focused on functional representation and animation and do not usually provide a great deal of aesthetic flexibility.

Avatar modeling software has specifically been created to allow users to create custom, low polygon human figures for use in online multi-user environments such as *Blaxxun* [9] and *Active Worlds* [1]. Many avatar creation systems allow for the replacement of individual body part segments with arbitrary geometric objects, individual part scaling, part coloring, and selection from a discrete set of textures [30] [38].

Poser provides a collection of human bodies with high polygon counts as well as an interface for smoothly scaling individual body parts [18]. *ANTHROPOS* is a NURBS based *3D Studio MAX* plug-in which allows body segment scaling and includes some high level parameters for adjusting attributes like race and weight [16]. Metaballs [11] have been used for modeling humans for quite some time [10]. Boulic's system, like the work described in this paper, also used metaballs as modeling primitives and provided several parameters for controlling the relative proportions of the body [14].

While most of the above at minimum allow for the individual sizing of body parts (e.g., make the left forearm longer or the right thigh thicker), games such as the many Quake [25] and Unreal [21] variants typically only allow players to select from a set of models, with little support for avatar customization beyond texture selection.

*ACCAD, 1224 Kinnear Road, Columbus, OH 43212 USA, mlewis@cgrg.ohio-state.edu

†CIS, 395 Dreese Lab, 2015 Neal Avenue, Columbus, OH 43210 USA, parent@cis.ohio-state.edu

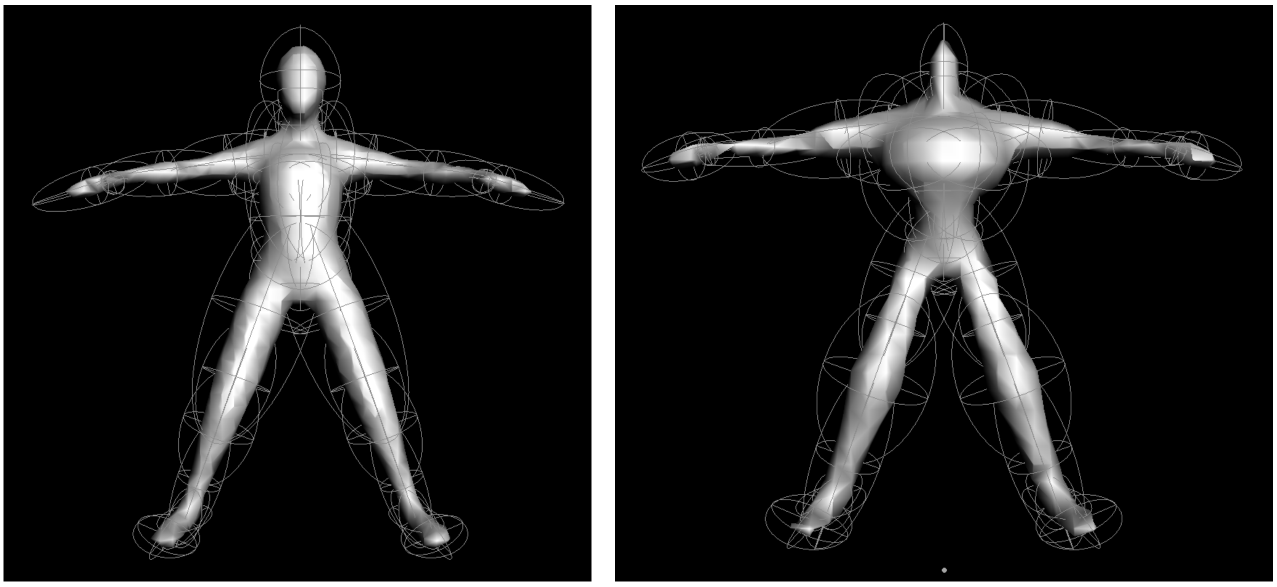

(a) Default Prototype Body (b) Transformed Segments

Figure 1: Metaballs

1.2.2 Aesthetic Evolutionary Design

For many formal visual design problem spaces, it is very difficult to produce a fitness function to allow the computer to automatically judge the quality of procedurally generated designs. In domains where a human designer must make aesthetic judgments, this is particularly difficult. One solution is to introduce interactive human evaluation into the assignment of fitness. This has been referred to as *aesthetic evolutionary design* (AED) [7]. AED can provide a very worthwhile design exploration tool for many design domains, as can be seen by the numerous systems which have made use of the technique.

Following Dawkins' biologically inspired Biomorphs program [19] which evolved 2D insect-like drawings, Todd, Latham [40], and Sims [36] were the first to evolve computer graphics objects via aesthetic selection. Images [22] [43] [24], sculpture [39], architecture [12] [17] [32], image processing [31], consumer products [8] [20] [37] [23], human bodies [27] and faces [15] [5], and character motion [3] [42] [29] have all been evolved by aesthetic selection.

Many others have made systems for evolving images and form using variations on the techniques employed by Sims, Todd, and Latham. Links to a number of these can be found at [28]. Rowbottom provides a detailed overview of the functionality of many of these as well [33].

2 Parametric Human Figures

Modifying the parameters of a generic prototype human model with a large number of continuous numeric attributes can produce a nearly infinite number of human figure designs. The model's parameters control the relative size and orientation of body parts, as well as localized deformation of the parts. For this implementation, the body part segments are constructed from a hierarchy of metaballs [11] as described below.

2.1 Prototype Geometry

The prototype body shown (Figure 1) consists of 35 metaballs. Body segments (mostly corresponding to bones between joints) are represented by one to three metaballs and are placed in a hierarchy rooted at the pelvis. The hierarchy used for this example corresponds roughly to LOA1 of the H-Anim standard with a few minor deviations [6].

Lateral symmetry is maintained when resizing segments. Segment placement is adjusted as segments are scaled in order to maintain relative joint positions. For example, if the thighs are uniformly scaled down, they are translated toward the pelvis to keep the hip joints at the top of the thigh segments. The calf segments are also translated upward to maintain knee joint placement. (Figure 3(a)). Each segment in the hierarchy is transformed[1] as follows:

$$T_G = \prod_{k=0}^{n} (T_{P_k} \cdot R_k \cdot T_{O_k}) \quad (1)$$

T_G is a given segment's global transform, n is the number of ancestors the segment has in the figure hierarchy, and k identifies an ancestor, with $k = 0$ referring to the segment itself. T_P is a segment's pivot translation matrix, which positions the segment relative to its parent joint:

$$T_P = T_{P_d} \cdot S \quad (2)$$

A segment's pivot translation T_P is always the initial default value of the pivot translation T_{P_d} multiplied by the segment's current scale S. So as a segment's size changes, so too does the distance between the segment and its parent, as

[1]The vertices represented as column vectors are pre-multiplied by the tranform matrices.

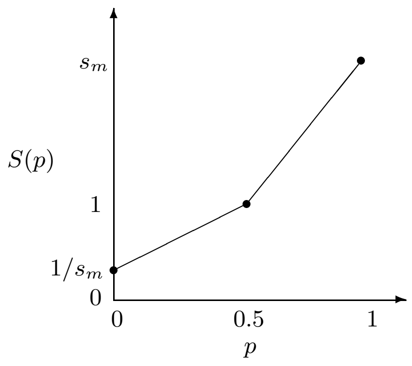


Figure 2: $S(p)$

well as the distance between each of the segment's children and the segment. R is the rotation of the parent joint of the segment. T_O is the offset translation which positions the segment relative to its parent segment.

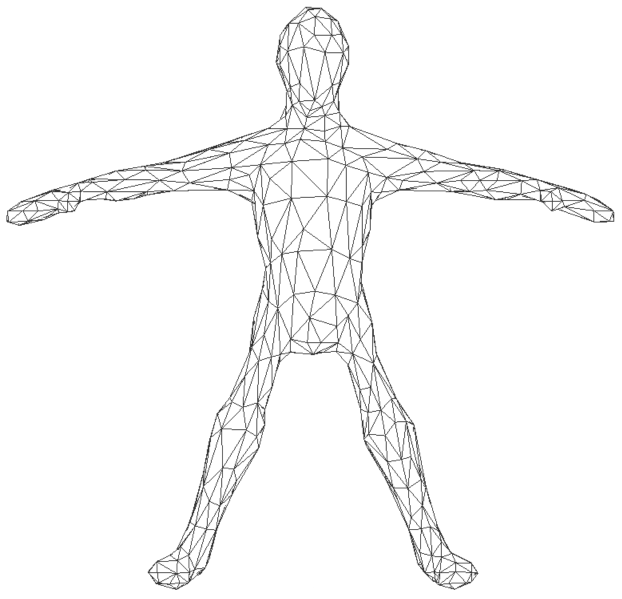
Note that while the representation used for this implementation of a parametric resizeable humanoid is a hierarchy of metaballs, any number of other representations could also have been used for the geometry. The first implementation of this approach used a single fixed polyhedral "skin" deformed by an internal hierarchical skeleton [27]. Either a simpler body consisting of separate polyhedral segments for each body part or a more complicated body built from many NURBS patches or subdivision surfaces could be used, so long as modifications to the body can be controlled by a set of parameters. The implicit surface technique was used this time to improve the smoothness of the skin under extreme deformation. A further improvement would be to use subdivision surfaces with localized crease control to allow for hard creases where necessary.

2.2 Segment Sizing Parameters

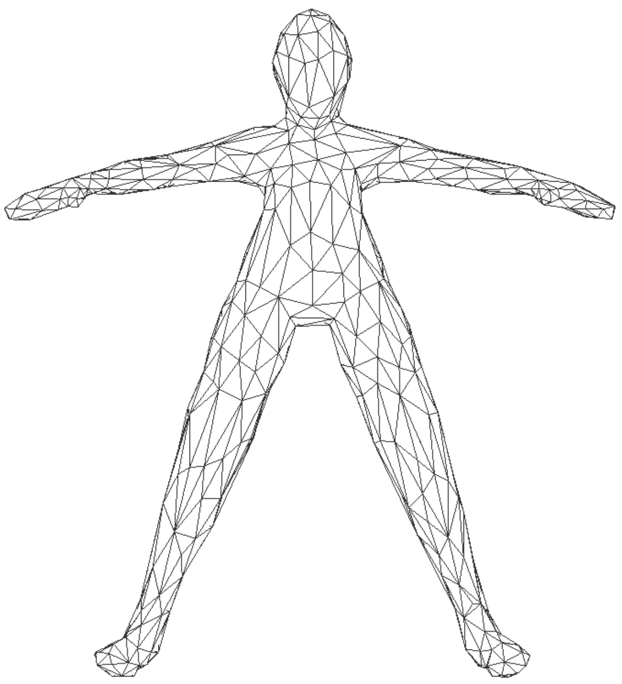
Each body segment has five sizing parameters controlling its scale. These are *length*, *width*, *depth*, *thickness*, and *size*. The *length* parameter controls scaling for most segments along the global Y-axis. The exceptions are the arm and hand segments, whose *length* aligns with the X-axis, and the feet, whose *length* aligns with the Z-axis. *Depth* controls scaling along the global Z-axis for all segments except the hands and feet, whose depth aligns with the Y-axis (hands are out at the sides, palm down). *Width* controls scaling orthogonally to *depth* and *length*. The *thickness* parameter is provided to permit simultaneous scaling in both the depth and width directions, while the *size* parameter allows simultaneous scaling of depth, width, and length (i.e., uniform scale).

All of the parameters controlling the adjustable properties of the body (and which will be referred to as *genes* later in the paper) have values in the [0..1] range, with their default set to 0.5. Most of these parameters need to be mapped from their normalized value domain into an appropriate parameter-specific range. Given a normalized parameter p controlling size and a maximum scale value s_m, the appropriate scaling value is determined by:

$$S(p) = \begin{cases} \frac{1}{s_m} + 2p(1 - \frac{1}{s_m}) & \text{if } p < 0.5 \\ 1 + (2p-1)(s_m - 1) & \text{otherwise} \end{cases} \quad (3)$$



(a) Scaling the Thighs



(b) Scaling the Torso

Figure 3: Part Sizing

Therefore, when a parameter controlling the scale of some body part is set to its default value of 0.5, the body part's scale is set to one (i.e., no change in size). If the parameter controlling the length of the thigh were to increase from its default value of 0.5 to its maximum normalized value of 1, then the left thigh would be scaled to s_m times its default length. If the parameter is decreased to zero, the thigh length is scaled by $1/s_m$ (figure 2).

The five parameters (i.e., *width*, *length*, *depth*, *thickness*, and *size*) discussed above which directly affect a segment's scale will by referred to as: $\langle p_w, p_l, p_d, p_t, p_s \rangle$. These are used to find the amount to scale along the *width*, *length*, and *depth* axes using equation 3 as follows:

$$scale = \begin{bmatrix} s_w \\ s_l \\ s_d \end{bmatrix} = \begin{bmatrix} S((p_w + p_t + p_s)/3) \\ S((p_l + p_s)/2) \\ S((p_d + p_t + p_s)/3) \end{bmatrix} \tag{4}$$

2.3 Hierarchical Sizing Parameters

In the previous section, parameters were described which control the size of individual segments in various dimensions. Similar parameters are also used to allow segments related by proximity and function to be sized simultaneously. For example, one might want to make the legs thicker, by scaling the width and depth of thighs and calves, using a single "leg thickness" parameter. The higher level body parts implemented include the *legs*, *arms*, *trunk*, *upper trunk*, and *lower trunk*.

For each of these parts, the same five sizing parameters discussed above are created (i.e., *width*, *length*, *depth*, *thickness*, and *size*). These five parameters are first decomposed into the primary three orthogonal parameters, *width*, *length* and *depth*, in much the same manner as in equation 4 above, only now the results are mapped into a $[-1..1]$ range:

$$\begin{bmatrix} p_w \\ p_l \\ p_d \end{bmatrix} = \begin{bmatrix} 2(p_w + p_t + p_s)/3 - 1 \\ p_l + p_s - 1 \\ 2(p_d + p_t + p_s)/3 - 1 \end{bmatrix} \tag{5}$$

The three resulting primary orthographic parameters, taken collectively for each of the higher level body parts (e.g. arms, legs) forms a $3 \times n_p$ matrix $\mathcal{P}$, where n_p is the number of higher level parts. Additionally, for each part, a *segment mask vector* is created of length n_s, where n_s is the number of segments. Each element of the vector indicates the degree to which a segment is a member of the higher level body part, and is in the range [0..1]. For example, the mask vector for an arm might contain a membership weight for each segment as follows:

$$M\text{arm} = \begin{bmatrix} 0 \\ 0 \\ 0.5 \\ 1 \\ 1 \\ \vdots \end{bmatrix} \begin{matrix} \text{//head} \\ \text{//torso} \\ \text{//shoulder} \\ \text{//upperarm} \\ \text{//lowerarm} \\ \vdots \end{matrix} \tag{6}$$

An $n_p \times n_s$ matrix $\mathcal{M}$ is then constructed with the segment masks as rows. Finally, a $3 \times n_s$ matrix $\mathcal{B}$, encoding the amount to modify each segment's width, length, and depth, according to the current part sizing parameters, is found by:

$$\mathcal{B} = \mathcal{P} \cdot \mathcal{M} \tag{7}$$

Each element of $\mathcal{B}$, b_{ij}, is at this point still in the range $[-1..1]$ and needs to be remapped using equation 4:

$$b'_{ij} = S((b_{ij}/2) + 0.5) \tag{8}$$

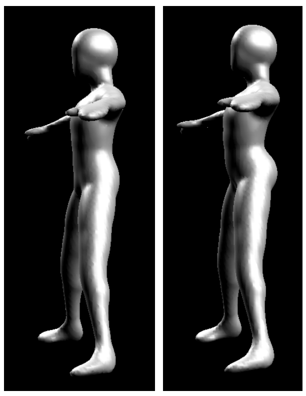

(a) (b)

Figure 4: Pelvic Tilt

The width, length and depth of each segment can then be multiplied by the elements in the corresponding column of $\mathcal{B}$.

2.4 Resting Posture

Additional parameters were added to demonstrate control of a few resting postural properties such as pelvic tilt and the arch of the back. The tilt of the pelvis greatly affects perception of gender (Figure 4), while the arch of the back can yield a lazy slouch or a proud chest. Changing one of these parameters modifies the base rotation of the affected postural joints. The postural joints required for these examples were single degree of freedom, and included three spinal joints and the hips.

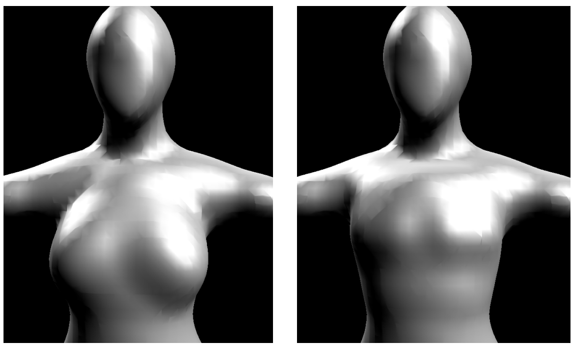
Given the set of all joints $J = \{j_0, j_1, \ldots, j_{n_j - 1}\}$ affected by one or more posture parameters, the set of posture parameters $P = \{p_0, p_1, \ldots, p_{n_p - 1}\}$, a vector D of default joint rotation values, and a pair of matrices $\mathcal{X}_{min}$ and $\mathcal{X}_{max}$ containing the minimum and maximum rotations caused by each postural parameter for each of the joints in J, a figure's postural joint rotations can be determined as follows:

$$j_i = D_i + \sum_k L(p_k, X_{min_{ik}}, D_i, X_{max_{ik}}) \tag{9}$$

with,

$$L(p, lo, d, hi) = \begin{cases} (2p-1)(d-lo) & \text{if } p < 0.5 \\ (2p-1)(hi-d) & \text{otherwise} \end{cases} \tag{10}$$

As was mentioned above, for this simple implementation, postural joints only needed to rotate around the X-axis. If arbitrary rotations were needed, then equation 9 would need to become a product of rotation matrices instead of the sum of joint angle changes, and the interpolation in equation 10 would be replaced by quaternion based spherical linear interpolation. Note also that while L could employ a smoother interpolation of the minimum default and extreme values if desired, such smoothing would bias the parameters values and is unnecessary for our purposes.



(a) (b)

Figure 5: Surface Deformation

2.5 Local Surface Deformations

Localized deformations of the surface geometry can be parameterized in a number of ways, depending on the desired effect and the primitive geometry being manipulated. While local deformations in the previous implementation of this work employed lattice based deformers [27], the flexibility granted by the blending of metaballs allows metaball sizing, placement, and shaping (via exponent adjustment) to be used to modify localized surface deformations. As an example, to facilitate the implementation of a "gender" parameter (described below), several chest deformation parameters were created which control the size, depth, orientation, and curvature of a pair of metaballs on the surface of the chest. This simplification gives sufficient variety to produce the suggestion of breasts or muscular pectorals of various sizes and shapes (Figure 5).

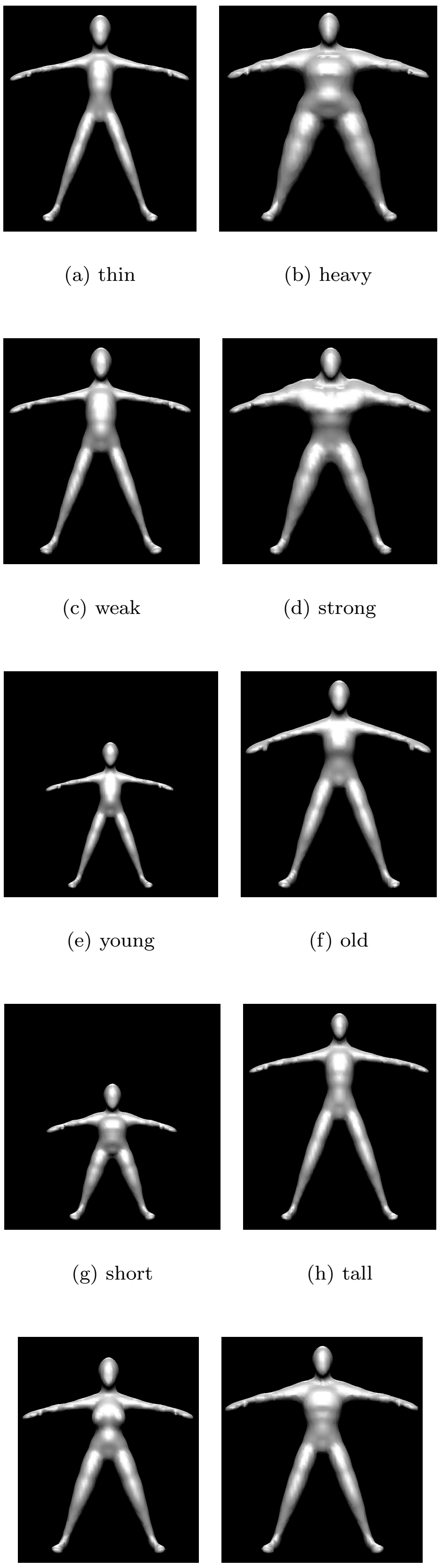
The method by which the chest deformation parameters are mapped into the scale, rotation, translation, and exponents of the metaballs will now be described. The process generalizes to any number of body attribute deformation parameters mapping to specific transformation values of geometric primitives. Just as in equation 9, the low level deformation attributes d (e.g., the Z-axis translation of a chest metaball) are found using a vector D of default values for d and matrices $\mathcal{X}_{min}$ and $\mathcal{X}_{max}$ containing minimum and maximum values for each low level deformation attribute for each deformation parameter. The weighted influence of each of the deformation parameters p is summed:

$$d_i = D_i + \sum_k w_{ik} \cdot L(p_k, X_{min_{ik}}, D_i, X_{max_{ik}}) \qquad (11)$$

The weight of influence of parameter p_k on d_i is given by w_{ik}, and L is from equation 10.

2.6 High-level Parameters

So far, low-level parameters have been described that allow individual body parts to be resized (e.g., "make the left forearm thicker") and it has been shown how localized regions can be reshaped parametrically. Mid-level parameters have been presented for sizing related groups of segments (e.g., "make the legs longer") as well as for reorienting certain groups. The implementation of higher level parameters such as weight, height, muscle, age, and gender, which modify



(a) thin (b) heavy

(c) weak (d) strong

(e) young (f) old

(g) short (h) tall

(i) female (j) male

Figure 6: High Level Parameters

subsets of the previously described parameters will now be described. Effects of such parameters are shown in figure 6.

Each of these high level parameters has affiliated with it a list of parameters to be affected. As was seen before with deformation and posture parameters, there are once again matrices $\mathcal{X}_{min}$ and $\mathcal{X}_{max}$ containing the minimum and maximum values by which each affected parameter should be modified when each high-level parameter changes. These limits represent aesthetic choices made by the designer of the prototype human body. These numbers are easily adjusted, as designers will no doubt have very different opinions about what ratio of hip-to-shoulder proportion is appropriate for perception of gender, for example.

The change in each parameter (Δp_i) based on the value of each high level parameter h_k is as follows:

$$\Delta p_i = \sum_k w_{ik} \cdot L(h_k, X_{min_{ik}}, 0, X_{max_{ik}}) \qquad (12)$$

The weight of the influence of parameter h_k on Δp_i is again specified by a matrix $\mathcal{W}$ of weights w_{ik}, and L is from equation 10. Note that the $\mathcal{X}$ matrices need not really contain a "minimum" or "maximum". Rather they are just the amount by which the parameter in question should be modified, as the h_k changes from 0.5 to either extreme.

Like the limits specified in $\mathcal{X}_{min}$ and $\mathcal{X}_{max}$, the weights in $\mathcal{W}$ are again chosen by the designer of the human prototype model. Although the lack of parameter clamping can potentially push parameters out of their normalized ranges, this is not a problem in practice, and it might even be seen as a benefit in this evolutionary design context since it ultimately broadens the low level parameter ranges and thus the scope of the potential search space. If it were to become an issue for a given prototype implementation, parameters could be clamped when the changes calculated in equation 12 are added.

3 Evolving Parameter Values

3.1 Concept

The total set of parameters defined for the prototype figure described in the previous section forms an N-dimensional, continuous parametric space $\mathcal{C}$ of possible human figures. Each point in this space maps to a specific (though not necessarily visually unique) human figure. Instead of searching for an interesting model by making small, single axis adjustments in $\mathcal{C}$ by adjusting one parameter at a time, several regions of $\mathcal{C}$ are browsed simultaneously. This is done by displaying a population of possible designs (each corresponding to a point in $\mathcal{C}$) and allowing the user to judge them by selecting those deemed the most interesting. Given the user's critique, the system can then generate a new population of design proposals.

To produce the next generation, members of the previous generation are "mated" with preference given to those marked attractive by the user. The mating process discussed below tends to endow children with many of their parents' parameters (e.g., if short, fat parents are selected, short, fat offspring are likely). This process allows the user to travel through parameter space in the direction of greatest aesthetic preference.

3.2 Algorithm and Interface

When the system is started, an initial population of human models is generated by creating the desired number of copies of the prototype and generating random normalized values for each copy's parameters. The copies are each displayed in a grid (Figure 8). The list of parameter values controlling the shape of each figure is called the *genotype*. Each individual parameter is referred to as a *gene*, and each gene's value is called an *allele*. The human figure model produced by applying the parameter values to the prototype model is called a *phenotype*. In this implementation of the human prototype, each figure's genotype contains ninety-six genes[2], each of which corresponds to one of the normalized parameters discussed in section 2.

Given that the initial default prototype body was created to be fairly "realistic" in terms of proportion and scale, the degree of deviation from the default parameter settings that is permitted will determine the realism of the resulting population as can be seen by comparing Figures 8(a) and 8(b). This deviation can be globally adjusted by mapping the alleles into a $[-1..1]$ range and then scaling them as desired.

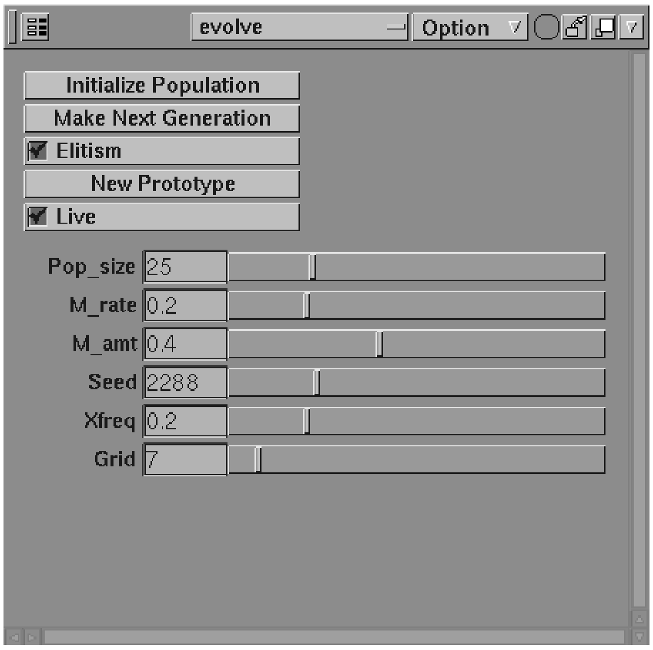
Additional control over the realism of the initial populations can be gained in general by controlling the amount that the body segments' relative sizes are constrained. The different subsets of genes discussed in the subsections 2.2 through 2.6 can be weighted to modify their relative influence. If more weight is given to the higher level parameters, and less to the low level segment sizing, then changes will largely be made in more realistic ways: bodies will be overall taller, thinner, stronger, etc., while maintaining more realistic proportions. If, on the other hand, low-level segment sizing is given more weight than the high level genes, then a much wider variety of large and small segments may result, yielding perhaps more potentially interesting character designs. The tuning of these additional weights gives yet another aesthetic choice to the designer of the human prototype.

Once the initial population has been generated and displayed, the user selects the members of the population found to be most pleasing by selecting them with the mouse. Each figure has a time-based expression in its Y rotation field $((\$F - 1) * 6)$ so that moving the animation time bar forward and backward will cause each figure to rotate in place, allowing for examination of the figures from all sides (Figure 8(c), selected figures are highlighted). For large populations, it may be desirable to move the camera through the space to examine subsets of the population more closely, from arbitrary viewing positions (Figure 8(e)).

Custom interface components were assembled to control the rate of evolution (Figure 7). The mutation amount and frequency, crossover frequency, and population size can all be adjusted. Mutation refers to random changes made to the genes of the population. Crossover is the means by which the genotypes of two selected parents are combined to form a child genotype. This is implemented by copying a subsequence of genes from one parent, then "crossing over" to the corresponding position in the other parent's genotype and continuing to copy the second parent's genes into the offspring's genotype. The "Crossover Frequency" controls the number of times that the copying process switches between the two genotypes when generating an offspring. Further details of the implementation of this process in Houdini's data flow networks are discussed in [26].

When the user presses the *Make Next Generation* button,

[2]The ninety-six parameters used here include sixty for segment sizing (i.e., five parameters for each of twelve unique segments), twenty-five for higher level part sizing (i.e., five parameters for each of five higher level parts), two posture, four deformation, and five high level parameters.



✓ Elitism
✓ Live
Pop_size 25
M_rate 0.2
M_amt 0.4
Seed 2288
Xfreq 0.2
Grid 7

Figure 7: Evolution Interface

the genotypes of the selected figures are stored as the mating pool for generating offspring for the next population. Mating is conducted by iteratively selecting random pairs of genotypes from the mating pool and performing crossover on the pair to generate an offspring to add to the next generation. Once the requested number of offspring are produced, some percentage of their genes may optionally be mutated, introducing small random changes to some of the genes. If the *elitism* option is activated, the figures chosen to be parents will survive into the next generation (i.e., they are copied unchanged).

As this process is repeated, the user typically reduces the mutation and crossover rates to slow movement through design space from generation to generation. The search is gradually narrowed until it converges into a specific region of the parametric space upon which the user's interest is focused (Figures 8(d) and 9). Any figure's parameters can be manually adjusted at any time if desired. Also any individual figure model's geometry can be saved.

4 Conclusions and Future Work

As can be seen in the examples, a wide range of body types can be easily generated using relatively simple parameters and selection-driven interaction. Arbitrarily complex figures can be produced with these techniques by increasing the number, complexity, and accuracy of primitives and parameters built into the human prototype, producing any desired degree of realism and flexibility of form, limited only by the prototype author's modeling skills and the computer's ability to display a population interactively. An interface like the one described can be used in an exploratory manner for brainstorming, to generate a large number of similar yet different entities quickly, or for creating finished geometry. The primary purpose of this work, however, was to further develop and explore the infrastructure required to support 3D interactive evolution problem domains such as this.

For future work, using techniques such as those introduced by Sims to evolve surface textures would be feasible [36]. Evolving character animation is also an interesting area [3]. There is also a great need to study ways to facilitate fast visual review of larger numbers of individuals. While it is possible to quickly create a very large population, examining the members of the population for fitness using the standard camera control interface is sometimes unwieldy. Finally, learning a user's aesthetic preferences by analyzing selections is a fascinating area for future work.

5 Acknowledgments

Thanks to Wayne Carlson for supporting this work. Also thanks to John Josephson, Beth Lewis, Flip Phillips, Side Effects Software, and Lawson Wade for valuable discussions and support.

Permission to display elements of the Houdini interface in figure 7 courtesy of Side Effects Software Inc. All product names mentioned are trademarks or registered trademarks of their respective holders.

References

[1] ActiveWorlds.com, Inc. Active worlds. http://www.activeworlds.com, 2000.

[2] Alias|Wavefront. Maya. http://www.aliaswavefront.com, 2000.

[3] Riccardo Antonini. Implementing an avatar gesture development tool as a creative evolutionary collaborative system. In P. J. Bentley and D. Corne, editors, *Proceedings of the AISB'99 Symposium on Creative Evolutionary Systems (CES)*. Morgan Kaufmann, 1999.

[4] Autodesk, Inc. 3d Studio MAX. http://www.discreet.com, 2000.

[5] Ellie Baker. Evolving line drawings. In *Proceedings of the Fifth International Conference on Genetic Algorithms*. Morgan Kaufmann, 1993.

[6] Matthew Beitler. H-Anim: Humanoid Animation Working Group. http://www.h-anim.org, 2000.

[7] Peter J. Bentley. *Evolutionary Design by Computers*. Morgan Kaufmann, 1999.

[8] Peter J. Bentley. From coffee tables to hospitals: Generic evolutionary design. In Peter J. Bentley, editor, *Evolutionary Design by Computers*, chapter 18, pages 405–423. Morgan Kaufmann, 1999.

[9] Blaxxun Interactive. Blaxxun community platform. http://www.blaxxun.com, 2000.

[10] Jim Blinn. Nested transformations and blobby man. *IEEE Computer Graphics And Applications*, pages 59–65, October 1987.

[11] Jules Bloomenthal, editor. *Introduction to Implicit Surfaces*. Morgan Kaufmann, 1997.

[12] Beth Blostein. Procedural generation of alternative formal and spatial configurations for use in architecture and design. Technical Report OSU-ACCAD-5/95/TR1, Advanced Computing Center for the Arts and Design, The Ohio State University, 1995.

[13] The Boeing Company. Boeing Human Modeling System: BMD-HMS. http://www.boeing.com/assocproducts/hms, 2000.

[14] R. Boulic et al. The HUMANOID environment for interactive animation of multiple deformable human characters. In *Proceedings of Eurographics '95, Maastricht*, pages 337–348, August 1995.

[15] Craig Caldwell and Victor S. Johnston. Tracking criminal suspect through "face-space" with a genetic algorithm. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 416–421, 1991.

[16] Cebas Computer, Inc. ANTHROPOS. http://www.cebas.com, 2000.

[17] Paul Coates. Using genetic programming and L-systems to explore 3d design worlds. In R. Junge, editor, *CAAD-Futures '97*. Kluwer Academic, Munich, 1997.

[18] Curious Labs, Inc. Poser. http://www.curiouslabs.com, 2000.

[19] Richard Dawkins. *The Blind Watchmaker.* Penguin Books, 1986.

[20] Emergent Design. Chair farm. http://www.emergent-design.com/dyn/chair.html, 2000.

[21] Epic Games, Inc. Unreal tournament. http://www.unrealtournament.com, 1999.

[22] Janine Graf and Wolfgang Banzhaf. Interactive evolution of images. In D. B. Fogel, editor, *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, pages 53–65, 1995.

[23] I. J. Graham, R. L. Wood, and K. Case. Evolutionary form design: The application of genetic algorithmic techniques to computer aided product design. In *Proceedings of the 15th National Conference on Manufacturing Research (NCMR), "Advances in Manufacturing Technology Vol.13"*, September 1999.

[24] Jano van Hemert. Mondriaan art by evolution. http://www.wi.leidenuniv.nl/∼jvhemert/mondriaan, 2000.

[25] id Software, Inc. Quake III Arena. http://www.idsoftware.com, 1999.

[26] Matthew Lewis. Aesthetic evolutionary design with data flow networks (to appear). In *Proceedings of Generative Art 2000, Milan, Italy*, 2000.

[27] Matthew Lewis. Evolving human figure geometry. May OSU-ACCAD-5/00-TR1, ACCAD, The Ohio State University, 2000.

[28] Matthew Lewis. Visual aesthetic evolutionary design links. http://www.cgrg.ohio-state.edu/∼mlewis/aed.html, 2000.

[29] Ik Soo Lim and Daniel Thalmann. Pro-actively interactive evolution for computer animation. In *Proceedings of Eurographics Workshop on Animation and Simulation '99 (CAS '99), Milan, Italy*, pages 45–52. Springer, 1999.

[30] MEET Factory. Avatar builder. http://angels.kiasma.fng.fi/avatarbuilder, 2000.

[31] Riccardo Poli and Stefano Cagnoni. Evolution of psuedo-colouring algorithms for image enhancement with interactive genetic programming. In *Proceedings of the Second International Conference on Genetic Programming, GP'97*, pages 269–277. Morgan Kaufmann, 1997.

[32] M. A. Rosenman. An exploration into evolutionary models for non-routine design. In D. Dasgupta and Z. Michalewicz, editors, *Evolutionary Algorithms in Engineering Applications*, pages 69–86. Springer-Verlag, 1997.

[33] Andrew Rowbottom. Evolutionary art and form. In Peter J. Bentley, editor, *Evolutionary Design by Computers*, chapter 11, pages 261–277. Morgan Kaufmann, 1999.

[34] Safework, Inc. Virtual man. http://www.safework.com/ contents.html, 2000.

[35] Side Effects Software, Inc. Houdini. http://www.sidefx.com, 2000.

[36] Karl Sims. Artificial evolution for computer graphics. *ACM Computer Graphics*, 25(4):319–328, 1991.

[37] Celestino Soddu. Argenic design: Chairs. http://soddu2.dst.polimi.it/stoc_sed.htm, 1999.

[38] Sven Technologies. Avatarmaker. http://www.sven-tech.com, 2000.

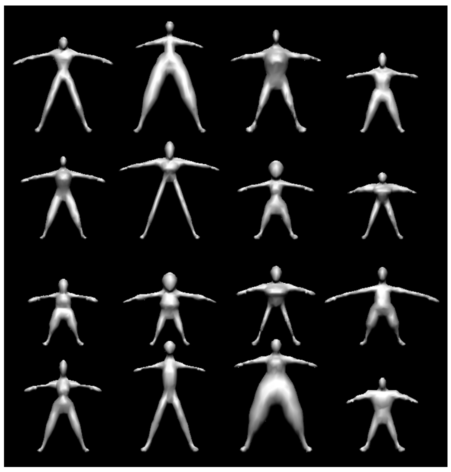
[39] P. Tabuada, P. Alves, J. Gomes, and A. Rosa. 3d artificial art by genetic algorithms. In *Proceedings of the Workshop on Evolutionary Design at Artificial Intelligence in Design - AID' 98*, pages 18–21, 1998.

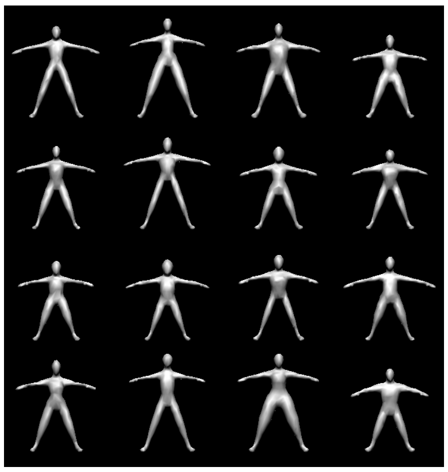
[40] Stephen Todd and William Latham. *Evolutionary Art and Computers.* Academic Press, 1992.

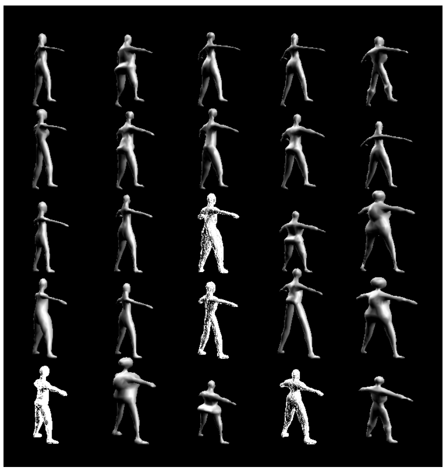
[41] Transom Technologies, Inc. Jack. http://www.transom.com, 2000.

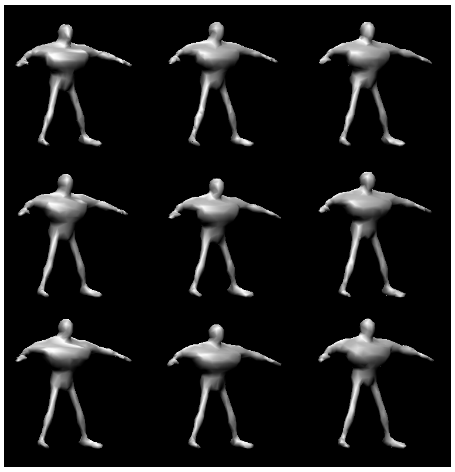
[42] Jeffrey Ventrella. Disney meets Darwin: The evolution of funny animated figures. In *Computer Animation '95 Proceedings*, Geneva, Switzerland, 1995.

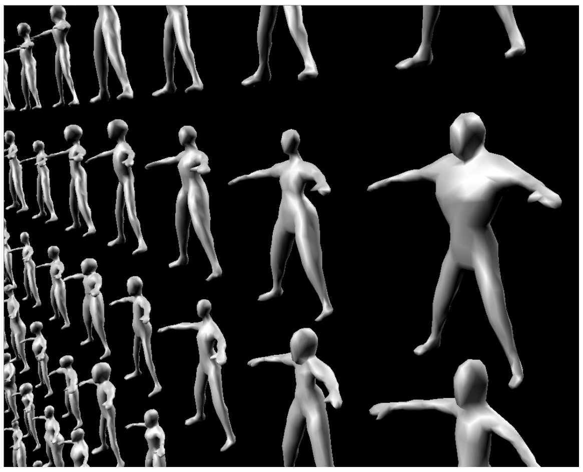
[43] Jeffrey Ventrella. Tweaks. http://www.ventrella.com, 2000.


(a)


(b)


(c)


(d)


(e)

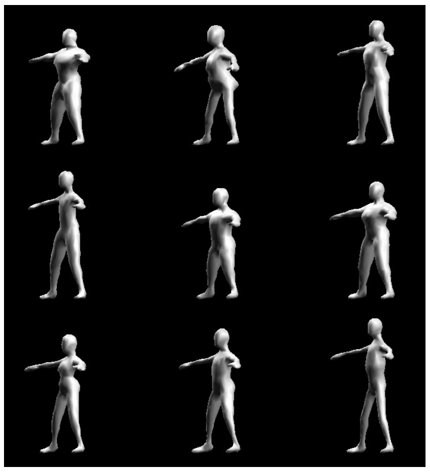
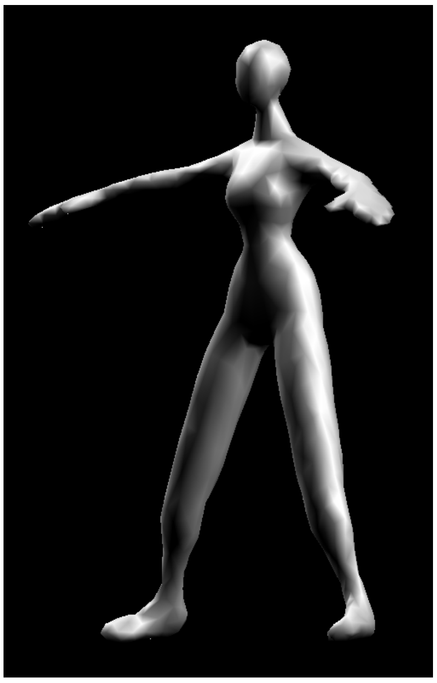
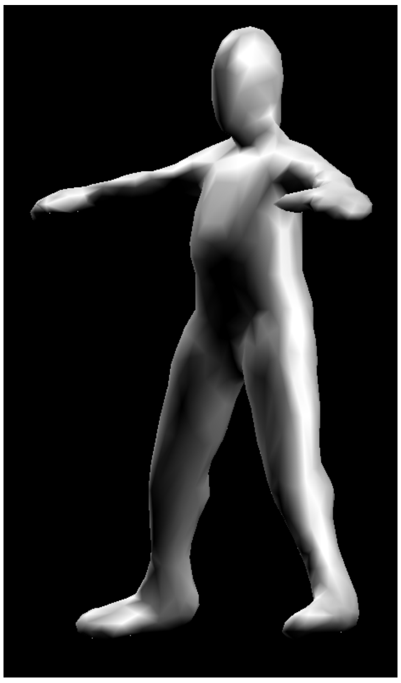

(f)

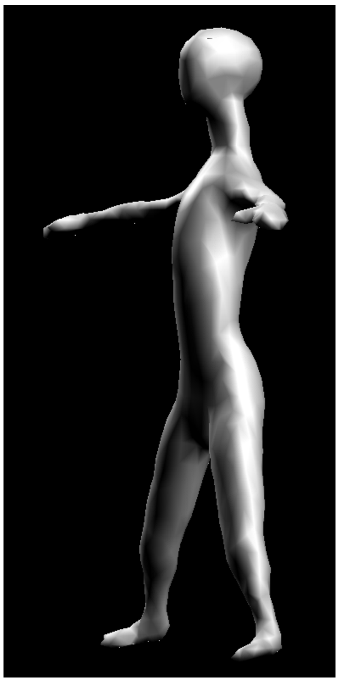
Figure 8: Populations



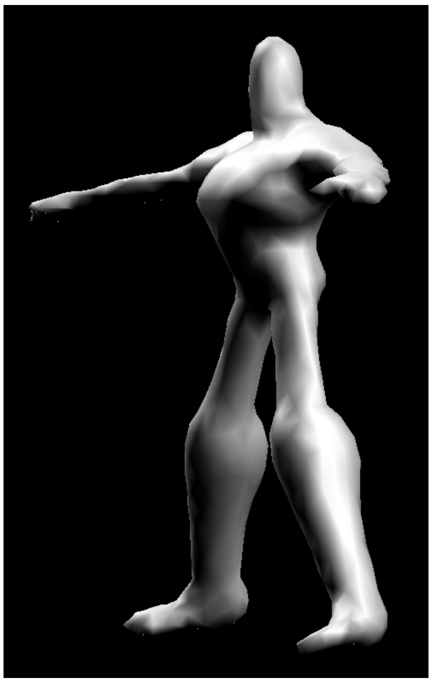
(a)



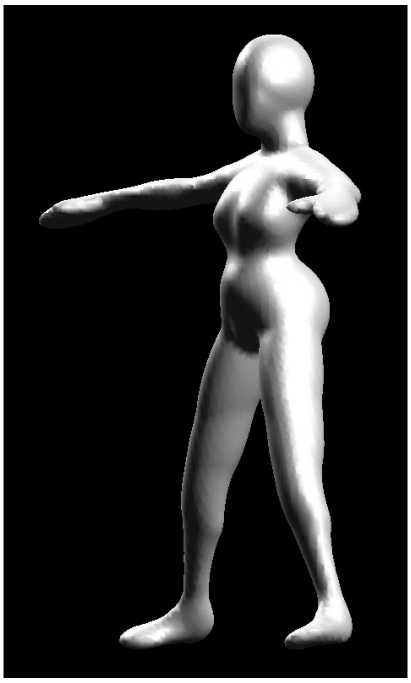
(b)



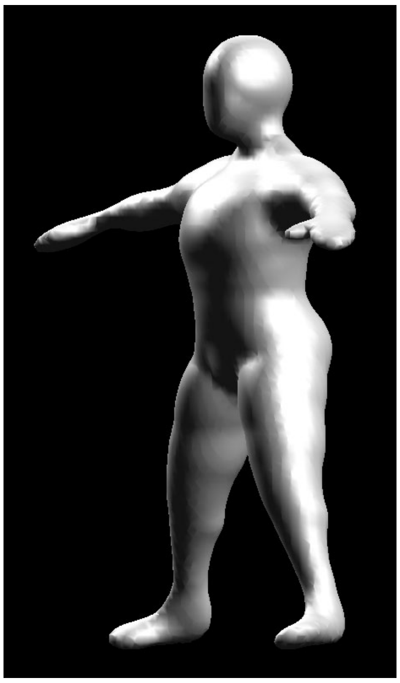
(c)



(d)



(e)



(f)

Figure 9: Evolved Bodies