Generic Deformable Implicit Mesh Models for Automated Reconstruction

Slobodan Ilic, Pascal Fua*
Swiss Federal Institute of Technology
Computer Vision Laboratory
IN-J Ecublens, 1015 Lausanne, Switzerland
{Slobodan.Ilic, Pascal.Fua}@epfl.ch

Abstract

Deformable 3–D models can be represented either as explicit or implicit surfaces. Explicit surfaces, such as triangulations or wire-frame models, are widely accepted in the Computer Vision and Computer Graphics communities. However, for automated modeling purposes, they suffer from the fact that fitting to 2–D and 3–D image-data typically involves minimization of the Euclidean distance between observations and their closest facets, which is a non-differentiable distance function. By contrast, implicit surface representations allow fitting by minimizing an algebraic distance where one only needs to evaluate a differentiable field potential function at every data point. However, they have not gained wide acceptance because they are harder to meaningfully deform and render.

To combine the strength of both approaches, we propose a method that can turn a completely arbitrary triangulated mesh, such as one taken from the web, into an implicit surface that closely approximates its shape and can deform in tandem with it. This allows both graphics designers to deform and reshape the implicit surface by manipulating explicit surfaces using standard deformation techniques and automated fitting algorithms to take advantage of the attractive properties of implicit surfaces. We demonstrate the applicability of our technique for upper body—head, neck and shoulders— automated reconstruction.

1. Introduction

In the world of Computer Graphics, 3–D objects tend to be modeled as explicit surfaces such as triangulated meshes or parametric surfaces like spline patches. Because such representations are intuitive and easy to manipulate, they are widely accepted among graphics designers. These representations, however, are not necessarily ideal for fitting surfaces to data such as 3–D points produced by laser-scanners and stereo systems or 2–D points from image contours where the data are noisy and incomplete. This stems from the fact that fitting typically involves finding the facets that are closest to the 3–D data points or most likely being silhouette facets. This involves non-differentiable distance function, which degrades the convergence properties of most optimizers.

Implicit surfaces, known in the literature as *Blobby Molecules[4], Soft Objects[34] and Metaballs[19]*, have received substantial attention in both the Computer Graphics and Computer Vision communities. They are well-suited for simulating physically based processes and for modeling smooth objects. Because the algebraic distance to an implicit surface is computed by evaluating a differentiable function, they do not suffer from the drawbacks discussed above when it comes to fitting them to 2 and 3–D data [29, 22, 8]. However, they have not gained wide acceptance, in part because they are more difficult to deform and to render than explicit surfaces.

In short, explicit surface representations are well suited for graphics purposes, but less so for fitting and automated modeling. The reverse can be said of implicit surface representations. In earlier work [**?**], we proposed method for combining the strengths of both approaches while avoiding their drawbacks by converting explicit surfaces into *implicit meshes* whose shape closely approximates that of the original triangulations and deforming the implicit and the explicit surfaces in tandem for fitting purposes. To create the implicit mesh, we circumscribed each facet with a spherical volumetric primitive with its center being on the facet, as depicted by the middle row of Fig. 1. This approach is effective but has some limitations: It works best for fairly regular meshes like one shown in the middle row of Fig. 1, or high-resolution meshes such as one shown in Fig. 2(e, f), while it can produce lumpy implicit surfaces for irregular coarse ones, as depicted in Fig. 2(a, b).

* This work was supported in part by the Swiss National Science Foundation.

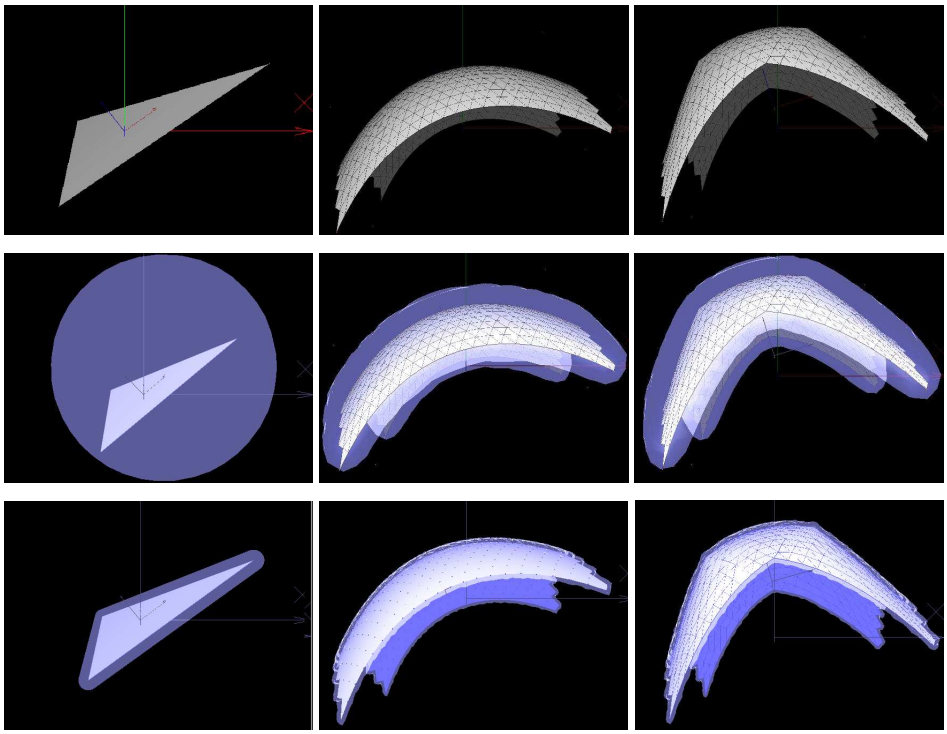


Figure 1. Converting an explicit surface into an implicit surface. *Top row:* Initial explicit meshes (facet, triangulated mesh patch and deformed mesh patch, from left to right respectively) . *Middle row:* Initial explicit surfaces from the top row converted to the spherical implicit meshes shown as transparent around explicit once. *Bottom row:* Initial explicit surfaces from the top row converted to the triangular implicit meshes shown again as transparent.

Here, we overcome these limitations by replacing spherical metaballs by *triangular metaballs* shown in the left hand side image of the bottom row in Fig. 1. Instead of computing the potential field as a function of the distance from the center of a facet, we take it to be the Euclidean distance from the whole triangle. As shown in the bottom row of Fig. 1 and in third and forth column of Fig. 2, the thickness of the implicit surface approximating the explicit surface can be arbitrarily small, whatever the mesh topology. The parameters of those metaballs are a function of the facet geometry. As a result, when a facet deforms, so does the corresponding metaball and the implicit and the explicit surfaces move in tandem. In this work, we use Dirichlet Free Form Deformation(*DFFD*) [18, 13] to control the shape, but in general, since we can turn any mesh into its implicit representation one could have chosen other methods, such as Free Form Deformations(*FFD*s) [26, 7], B-splines or PCA parameterization [3] to deform the explicit mesh and consequently the implicit one. *DFFD* had been chosen because it allows us to control any complex shapes with a relatively small number of parameters, then allows arbitrary control points deployment, produces local deformation and provides very natural way of deforming the objects for the graphics designers.

Our contribution is therefore an approach to surface fitting that allows to take an arbitrary explicit surface model of any complexity, for example one that has been obtained from the web and was not designed with fitting in mind, turn it into an implicit mesh, and deform it to obtain an optimal fit to image-data. In the automatic reconstruction implicit surface is just virtually present and it was fitted to the data, while actual explicit mesh was deformed along with it. Because of very close approximation of the mesh with its virtual implicit surface we can keep the deformed explicit mesh and use it instantly for rendering, furthermore provide it to the graphic designer with a optimal position of the control points for further modification and animation.

In the remainder of the paper, we first briefly review earlier approaches. We then introduce our approach to creating implicit meshes and deforming them, where we compare spherical and new triangular metaballs approach. Then, we describe our optimization framework, and finally demonstrate the applicability of our framework to the complex

case of fitting the upper-body including – head, neck and shoulders – to image-data, where we compare results obtained by fitting explicit mesh, spherical implicit mesh and triangular implicit mesh to stereo and silhouette data.

2. Previous Work

Three-dimensional surface reconstruction continues to be an important goal and many approaches relying on explicit surface representations, such as 3–D surface meshes [6, 31], parameterized surfaces [28, 17], local surfaces [10], and particle systems [30], have been proposed.

There has also been sustained interest in the use of volumetric primitives [16, 31, 20] and implicit surface representations [8, 29, 23] for fitting purposes. These methods, however, are tailored for specific shapes such as the human body and its skeleton and there is no generally accepted way to deform generic implicit surfaces.

A popular way to deform implicit surfaces is to twist, bend, and taper the space in which the model lives by choosing a suitable warping function [4, 2, 33]. However, these deformations are limited to parametric surfaces, such as spheres or cylinders, and there is no way to warp the space in a free form manner. In [1], simple superquadrics are parametrized using conventional FFDs for automatic heart reconstruction and deformation from medical images. Here the FFDs ability to deform parametric surfaces has been exploited, but only to reshape a single primitive. Our proposed implicit shells coupled with DFFDs [18, 13] go much further by allowing us to deform completely generic implicit surfaces. In spirit, the our implicit meshes are related to the earlier distance surfaces [5]. However, in this earlier work, the problems associated to bulges created by metaballs blending into each other are handled by a convolution mechanism that looses the algebraic nature of the distance function and makes the distance surfaces impractical for the kind of fitting we perform.

Radial basis functions (RBF) [14, 15, 32] are an interesting alternative to soft objects or metaballs [34, 19]. The shape of the resulting surface, however, is controlled not only by the position of the RBF centers but also by the RBF weights that have no geometric interpretation, which makes this approach also unsuitable for in-tandem deformation of explicit and implicit surface.

In short, both approaches to 3–D modeling have their strengths and weaknesses for the purpose of fitting noisy image-data. It is therefore important to be able to combine two kinds or representations and deform them in tandem.

3. Implicit Mesh Models

To create an implicit mesh model that can deform in tandem with the explicit surface, we must address two problems:

1. Creating an implicit surface that closely approximates the shape of the initial explicit mesh,
2. Controlling the object shape, in both its explicit and implicit forms, using the same set of parameters.

To convert an arbitrary triangulated surface into an implicit mesh, we create an implicit surface primitive or *metaball* for each facet. In earlier work [**?**] we used spherical metaballs, which are very simple but only suitable for fairly regular meshes or high resolution meshes as it is shown in the middle row of Fig. 1, where first one triangle, then ordinary regular and the deformed mesh patch are converted into spherical implicit mesh shown as transparent. Here, we replace them by the triangular metaballs, which are more complex but can handle arbitrarily irregular meshes and low resolution meshes and perform much closer approximation of the explicit surface, as depicted by the third row of Fig. 1. In this section, we first compare the two kinds of metaballs and then discuss our approach to shape deformation.

3.1. Spherical Metaballs

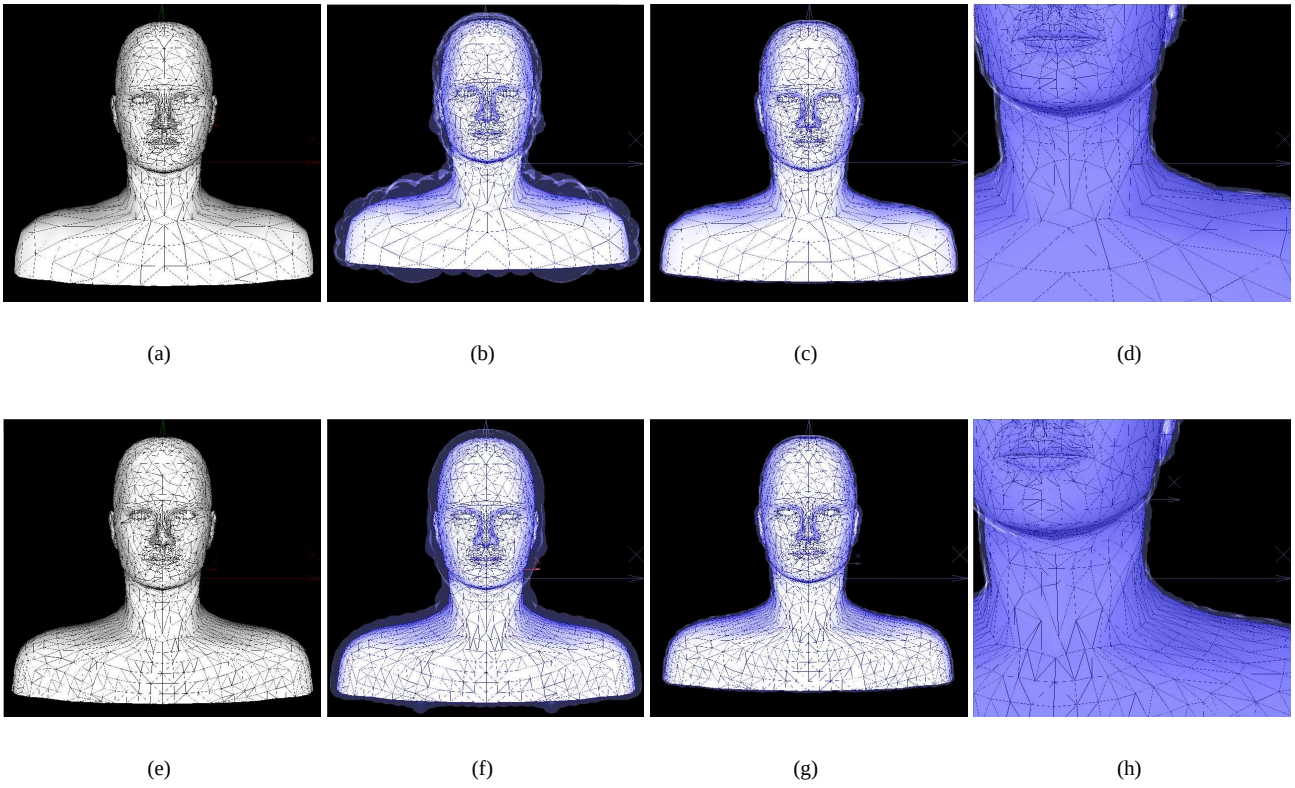
The spherical metaball [**?**] is created by circumscribing a spherical primitive around a facet in such a way that the sphere center lies on the facet and corresponds to the center of the circumscribed circle around the facet. It defines a potential field that can be expressed as:

$$f(\mathbf{x}) = exp(-k(r(\mathbf{x}) - r_0)) \tag{1}$$

where $\mathbf{x}$ is a 3–D point, r is the Euclidean distance to the sphere's center, r_0 is the radius of the spherical metaball and k is free coefficient defining slope of the potential field function. The implicit mesh, shown in gray in the middle row of Fig. 1, is then taken to be an isosurface of the sum of all these potential fields. Formally, it is defined as the set of 3–D points $\mathbf{x}$ that satisfy

$$F(\mathbf{x}) = T - \sum_{i=0}^{N} exp(-k(r_i(\mathbf{x}) - r_0)\,, \tag{2}$$

where T is an arbitrarily chosen isovalue. Usually we take T to be one, so that all points on the surface have a potential field value equal to zero and the values smaller then zero inside and greater then zero outside. Because the spherical metaballs are circumscribed around the facets their radius r_0 depends on the size of the triangle. As shown in the second row of Fig. 1, as long as the explicit mesh is relatively regular or high resolution, this yields a valid approximation. However, because large facets produce large primitives, the approximation becomes much less accurate when the explicit mesh has large facets. If we deal with low resolution irregular mesh as the one depicted in Fig. 2(a),



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2. Conversion of low and high resolution explicit meshes to implicit ones, using either spherical or triangular metaballs . *Left column(a, e):* Low and high resolution mesh. *Second column(b, f):* Corresponding implicit surfaces created with spherical metaballs, shown as transparent. *Third column(c, g):* Corresponding implicit surfaces created with triangular metaballs. *Last column(d, h):* Magnified implicit triangular metaballs surface in the neck and shoulders area, highlighting the approximation's quality.

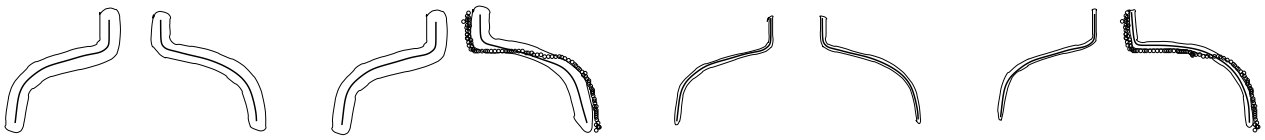
elongated facets produce an implicit surface whose thickness can change dramatically, as shown in Fig. 2(b). Up to a point, that can be remedied by retriangulating the mesh obtaining one depicted in Fig. 2(e), so that it consists of many smaller size facets and produce better approximation as shown in Fig. 2(f). This has been done in our previous work [**?**], but of course, that results in a substantial increase in computational cost.

Lack of close approximation of the explicit mesh with the implicit one may produce problems during the fitting. That is caused by the notion of two sides of the implicit surface which become important when implicit surface is thick like in a case of spherical implicit surface depicted in Fig. 3(a). If the model is not encapsulated inside the observation data and it intersects with the observations, that can cause fitting of the wrong side of the implicit surface to the data as it is shown in Fig. 3(b) in the neck area.

3.2. Triangular Metaballs

To solve these problems, and create implicit surfaces that more closely approximate arbitrary meshes, we propose to replace the spherical metaballs by *triangular* ones. This is done by replacing the Euclidean distance r to the facet' center in Eq. 1 by the actual distance d to the whole facet. In the bottom row of Fig. 1 you can see metaball created around the triangle which we call triangular metaball. The distance function is the Euclidean distance from the triangle expressed as function which defines distance either from the plane if the point projects on the triangle or the distance from the line or point if the point project outside the triangle.

Finally, distance function can be incorporated in the same potential field function as used for spherical metaballs:



(a) (b) (c) (d)

Figure 3. Influence of the explicit mesh approximation by the spherical or triangular implicit surface to fitting results. *(a)* Spherical implicit mesh. *(b)* Inner side of the spherical implicit mesh fitted to the data, depicted as small circles, in the neck area. *(c)* Triangular implicit mesh. *(d)* Correct fitting of triangular implicit mesh to the data because of the small thickness of the implicit surface.

$$f(\mathbf{x}) = exp(-k(r(\mathbf{x}) - d_0)) \tag{3}$$

that has almost the same form as before, but where $r(x)$ is distance of the point in space to the triangle and d_0 is a distance that represents the thickness of the implicit surface. Actually, all the points in space at the distance d_0 from the triangle have potential field value equal to zero. Again, complete implicit surface is obtained by summing all the field potentials that produces overall implicit surface expressed as:

$$F(\mathbf{x}) = T - \sum_{i=0}^{N} exp(-k(r_i(\mathbf{x}) - d_0)\ , \tag{4}$$

It is easy to spot that the potential field is now independent of the facet sizes and mesh resolution as depicted in Fig. 2(c, g, d, h) what is not the case for the spherical metaballs as shown in Fig. 2(b, f). Having control over the parameter d_0 allows us to approximate the explicit mesh with an arbitrarily thin implicit surface and in that way to relax the constraint of using fairly regular or high resolution mesh.

However, triangular implicit mesh might produces small bulges near vertices and along the edges, but since we have control over the slope of the exponential field function k it is easy to remove the bulges by tuning k to smaller value. Also, the problem of fitting to the wrong side of the implicit surface is now overcome by the very close approximation of explicit surface. Choosing d_0 to be arbitrary small thickness of the implicit surface, even if fitting is done to the inner side of it, fitting error is negligible as shown in Fig. 3(c, d).

3.3. Deforming Implicit and Explicit Meshes

We have shown that introducing DFFD control points is an effective way to deform explicit meshes [13]. Our idea of converting explicit mesh to implicit surface by close approximation allows to apply the same deformation mechanism based on DFFD control points to control the shape of both explicit and implicit surface.

3.3.1. Deforming Explicit Meshes Mayor advantage of DFFD over other FFDs [26, 7, 12], is obtained by releasing the constraint on the shape of the control mesh, which is the main conceptual geometric limitation of FFDs. Here rectangular local coordinates of FFDs are replaced by generalized natural neighbor coordinates of DFFD, also known as Sibson coordinates, and a generalized interpolant [9] is applied. The idea comes from the data visualization community that relies on data interpolation and, thus, heavily depends on local coordinates. This property of locality allows using sparse matrix computation in our optimization framework, what is not the case for other FFDs which are global deformation.

3.3.2. Computing Sibson Coordinates Every surface triangulation point is influenced by certain subset of control points. The magnitudes of these influences, known as Sibson coordinates [27], are computed only once before the optimization starts [18]. The displacement of each surface triangulation point is the linear combination of the displacements of the control points that influence it.

Let $P = \{P_1, P_2, ..., P_N\} \in R^3$ be the set of all control points and $Q = \{P_1, P_2, P_3, P_4\}$ be a subset of all control points, influencing surface triangulation point p, $Q \subset P, Q = \{P_k\}$, where $k = 1, \ldots, 4$ as shown in Fig 4(a). Subset Q of influencing control points has been

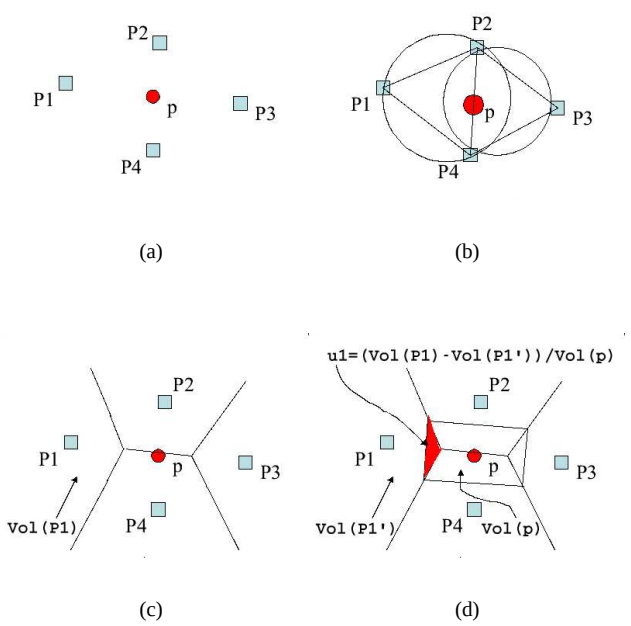


Figure 4. *(a)* Initial position of a subset of control points $Q = \{P_1, P_2, P_3, P_4\}$ and one mesh vertex p for which Sibson coordinates have to be computed. *(b)* Delaunay triangulation of the control point set with circumscribed spheres around each Delaunay facet. *(c)* Voronoi diagram of control points. *(d)* Voronoi diagram of control point set with added mesh vertex $Q^{'} = \{P_1, P_2, P_3, P_4, p\}$, and Sibson coordinate of control point P_1.

extracted from the overall set of control points P, by making Delaunay triangulation of all control points from P, and then searching to which circles(spheres in 3D) circumscribed around Delaunay facets, surface triangulation points p belongs to, as depicted in Fig. 4(b). The elements of Q are the natural neighbors of p and their influence is expressed by the Sibson coordinates u_k. Further step is to create Voronoi diagram out of control points from Q, what will produce one Voronoi cell for each control point that is some polygon in 2D or polyhedron in 3D as shown in Fig. 4(c). Finally, insert the surface triangulation point p to the set of control points Q and redo Voronoi diagram on set $Q' = \{P_1, P_2, P_3, P_4, p\}$. This will add new Voronoi cell corresponding to the inserted point p. It will take out parts of the volumes of the precedent Voronoi cells corresponding to control points from Q, as shown in Fig. 4(d). Simply, for control point P_1, part of its Voronoi cell taken by the new inserted Voronoi cell corresponding to p, marked as dark gray in Fig. 4(d), normalized with the whole volume of the cell p, is actual Sibson coordinate $u_1 = \frac{Vol(P_1) - Vol(P1')}{Vop(p)}$, as depicted on Fig. 4(d), which measures influence of control point $P_1$to surface point p.

Let the control points from Q be displaced from their initial positions by $\Delta P_k, k = 0, \ldots, N_c$, where N_c is number of influencing control points (in our example $N_c = 4$). The new position of the surface triangulation point becomes:

$$p^{new} = p + \sum_{k=0}^{N_c} u_k \Delta P_k, P_k \in Q \tag{5}$$

with $\sum_{k=0}^{N_c} u_k = 1$ and $u_k > 0$.

3.3.3. Deforming Implicit Meshes To deform implicit surface created from the explicit one either created from spherical or triangular metaballs it is sufficient to control parameters which define the shape of the metaballs. Let us consider one single triangle. In both cases important parameters which define shape of the metaball are distance function $r(\mathbf{x})$ and either radius of the metaball r_0 for the case of spherical metaball or thickness of the metaball d_0 in the case of triangular metaball. Distance function $r(\mathbf{x})$ in the case of triangular metaball is distance of the point $\mathbf{x} \in R^3$to the triangle, and in case of the spherical metaball distance from the center of the circumscribed circle around the triangle. Since triangle corners are controlled by the control points, and can be expressed as weighted linear combination of control points, as in Eq. 5, than the corresponding distance function $r(\mathbf{x})$ also depends on control points and can be expressed as:

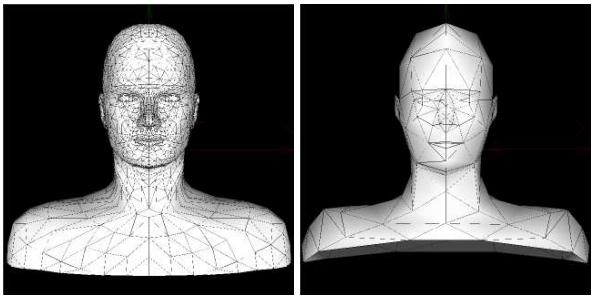
$$r(\mathbf{x}) = r(\mathbf{x}, P_{1,} P_2, ..., P_N) \tag{6}$$

In case of the spherical metaballs radius of the metaball r_0 also depends on the control points, while thickness of the triangular metaball d_0 is free parameter and is provided by the user. We can therefore rewrite the field potential function F of Eq. 4 that defines the implicit mesh as:

$$F(\mathbf{x}, P_1, .., P_N) = T - \sum_{i=1}^{N} exp(-k(r_i(\mathbf{x}, P_1, P_2, .., P_N) - d_0)) \tag{7}$$

where $\mathbf{x}$ is a point in R^3, $r_i(\mathbf{x}, P_1, P_2, ..., P_N)$ is the Euclidean distance to primitive i, d_0 is the implicit surface's thickness and N is number of facets.

Note that here we have chosen to use DFFD control points to deform the explicit mesh but, that in fact, the method is generic because allows us to express the vertices' positions as a function of a some other parameters.



(a) (b)

Figure 5. *(a)* Complete head and neck shoulders generic surface triangulation model which is converted to implicit mesh. *(b)* Control mesh representing triangulated DFFD control points, which appear to be optimization parameters.

4. Optimization framework

Our goal is to deform the implicit mesh so that it conforms to the image data which are made of 3–D points derived from stereo and silhouette information. Model is parametrized in terms of DFFD control points which are connected to a control mesh, as depicted on the Fig. 5(b), representing a reduced version of the original generic model shown in Fig. 5(a).

4.1. Objective Function

In standard least-squares fashion, for each data-point $\mathbf{x}_i$, we minimize following objective function:

$$\chi(S) = \frac{1}{2}\|D(\mathbf{x},S)\|^2 = \frac{1}{2}\sum_{i=1}^{N_{obs}} w_i^{type}\|d(\mathbf{x}_i,S)\|^2 \quad (8)$$

with weight w_i^{type}, where $type$ is one of the possible types of observations we use, such as stereo and silhouette observation, $S = \{dP_1, dP_2, ..., dP_{Nc}\}$ is a state vector that defines the surface shape and in our implementation represents displacements of the DFFD control points, d is the distance from the either stereo or silhouette point to the surface. In practice, we take $d(\mathbf{x}_j, S)$ to be the algebraic distance of $\mathbf{x}_j, j = 1, N_{obs}$ to the implicit surface defined by the field function of F of Eq. 7 and can be expressed as:

$$d(\mathbf{x}_j,S) = F(\mathbf{x}_j,S) = T - \sum_{i=1}^{N} f(r_i(\mathbf{x}_j,S),d_0), 1 \le j \le N_{obs} \quad (9)$$

Where T is a potential field threshold set to one, and N is a number of facets. To ensure that the minimization proceeds smoothly, the system automatically computes the w_i^{type} weights, so that the different kinds of observations have commensurate influence[13].

Because there are both noise and gaps in the image data, we still found it necessary to introduce a small regularization term based on the connectivity of the control mesh shown in Fig. 5(b). Since, we expect the deformation between the initial shape and the original one to be smooth, this can be done by preventing deformations at neighboring vertices of the control mesh to be too different. This is enforced by introducing a deformation energy E_D that approximates the sum of the square of the derivatives of displacements across the control surface. By treating the control triangulation facets as C^0 finite elements, we write

$$E_D = \Delta_x^t K \Delta_x + \Delta_y^t K \Delta_y + \Delta_z^t K \Delta_z \quad (10)$$

where K is a stiffness matrix and Δ_x, Δ_y and Δ_z are the vectors of the *x, y* and *z* coordinates of the control vertices' displacements. The term we actually optimize using Levenberg-Marquardt algorithm [24] becomes:

$$\chi(S) = \frac{1}{2}\|D(\mathbf{x},S)\|^2 + \lambda_D E_D \quad (11)$$

where λ_D is a small positive constant.

4.2. Stereo and Silhouette Observations

In this work, we concentrate on combining stereo and silhouette data. Because the field function F of Eq. 7 is both well-defined and differentiable, the observations and their derivatives can be computed both simply and without search.

Stereo Observations Disparity maps are used to compute clouds of noisy 3–D points such as those of Fig. 5. For each one of the kind we express the distance of the observation to the model defined by Eq. 9. Minimizing the norm defined by Eq. 11 tends to force the model to be as close as possible to the observations. Because of the long range effect of the exponential field function F of Eq.7, the fitting succeeds even when the model is not very close to the data. Also, during least-squares optimization, observations which appear to be outliers have smaller algebraic distance when they are further from the model. This means that the error measure approaches zero instead of becoming even greater with growing distance, what has the effect of filtering outliers.

Silhouette Observations A silhouette point in the image defines a line of sight tangential to the surface. Let S be an state vector. For each value S , we define the implicit surface:

Figure 6. Reconstruction from an uncalibrated video sequence. *Left column:* 3 of 6 images from a short video sequence with overlaid silhouettes on head, neck and shoulders. *Second column:* Disparity maps extracted from rectified consecutive image pairs using max flow-based stereo, after automated registration. *Third column:* Textured reconstructed model with triangular metaballs where overlaid silhouettes show correct fitting to silhouettes. *Forth column:* Animated reconstructed model.

$$L(S) = \left\{ \mathbf{x} \in R^3, F(\mathbf{x}, S) = T \right\} \tag{12}$$

Let $\mathbf{x}(S)$ be the point on the line of sight where it is tangential to $L(S)$. By definition, it must satisfy two constraints:

1. The point is on the surface, therefore $F(\mathbf{x}(S), S) = T$.
2. The normal to $L(S)$ is perpendicular to the line of sight at $\mathbf{x}(S)$.

We integrate silhouette observations into our framework by performing, before each minimization, a search along the line of sight to find the point that has the lowest field value and further must satisfy the second constraint as it is done in [21].

5. Results

Here we use the example of head, neck and shoulder modeling to demonstrate our method's applicability. We start with the generic head, neck and shoulder model as shown in Fig. (a) and Fig. 2(a) when fitting triangular implicit mesh that has a complex topology and is made of very irregular facets, and with remeshed high resolution model as shown in Fig. 2(e) when fitting spherical implicit mesh.

Reconstruction and Animation In Fig. 5 we show reconstruction results using stereo and silhouette data obtained from an initially uncalibrated 6–frame video sequence in which the camera was filming a moving subject. In the left column, we show the first, middle and last frames of the sequence. We used snakes to extract the silhouettes shown as white lines. In the absence of calibration information, we used a model-driven bundle-adjustment technique [11] to compute the relative motion and, thus, register the images. We then used a graph-cut technique [25] to derive disparity maps from consecutive images, such as those shown in the second column. In the third column, we show reconstructed model using triangular implicit mesh reprojected using the same camera corresponding to the images in the first column. Notice that outlined silhouettes which are taken from the original image precisely lay on the reprojected model.

Furthermore, in Fig. 7 we compare fitting quality of results obtained by fitting following models to stereo and silhouettes: the explicit mesh (first row), the implicit spherical mesh (second row), and the triangular implicit mesh (third column). Obtained results are reprojected back to the corresponding views of the camera and important details are zoomed in. First and the last end column represent two different camera views, while middle columns show close ups.

Notice that when fitting mesh because of non-differentiable distance function we have wrong result

Figure 7. Reconstruction results comparing fitting to stereo and silhouettes of the following models: Explicit mesh *(first row)*, Spherical implicit mesh *(second row)*, and Triangular implicit mesh *(third column)*

on the right shoulder and on the right side of the face as shown in the first row of Fig. 7. In the case of fitting spherical implicit mesh, as depicted in the second row of Fig. 7, even though we use high resolution model, the result obtained suffers from the problem of thick implicit surface, as explained in Fig. *3* where inner side of the implicit surface is fitted to the silhouettes in the neck area what produces wrong result on the left shoulder too. Finally, fitting of the triangular implicit mesh is depicted in the bottom row of Fig. 7. In this case well aligned re-projection of the reconstructed model with overlaid silhouettes indicates quality of the reconstruction, though we used low resolution irregular mesh. The texture-mapped models depicted on the right side column of Fig. 5 and the shaded views were generated by using only explicit surface, thereby underlining the importance of deforming explicit and implicit surfaces in tandem. It is important to stress that obtained avatar can be animated using optimal position of the control points obtained from our algorithm as a starting configuration. Some animation results are shown in forth column of Fig. .

6. Summary and Conclusions

We have presented an approach to switching from explicit surfaces to implicit that allows us to take advantage of the strengths of both kinds of approaches. To this end, we have proposed a technique for creating implicit meshes in such a way that their shape depends only on the explicit surfaces' shape and that they are both parametrized in the same way. Particularly we choose DFFD to deform the implicit and explicit models in tandem, but the way how implicit shells are created allows using of any other deformation method including direct mesh manipulation or indirect mesh manipulation using set of control points. This means that any FFD, or B-spline based approach of deforming meshes can be used to deform their implicit shells. Also, some other indirect methods for explicit surface deformation, such as PCA parmetrization, can be used to deform our implicit meshes.

We used the example of upper-body modeling using stereo and silhouette data to demonstrate the power of this approach. The explicit model we started from was not tailored for fitting purposes and has no man facets, but has a

complex topology, neither of which has a significant impact on the quality of the fitting or the complexity of the computation.

Our next step will be to explore the use of this method for tracking upper body motion form monocular video sequences what should automatically produce animation parameters of the model. We expect this to result in a completely generic approach for modeling and animation from images.

References

[1] E. Bardinet, L. Cohen, and N. Ayache. A Parametric Deformable Model to Fit Unstructured 3D Data. *Computer Vision and Image Understanding*, 71(1):39–54, 1998.

[2] A. H. Barr. Local and global deformations of solid primitives. pages 21–30, July 1984.

[3] V. Blanz and T. Vetter. A Morphable Model for The Synthesis of 3–D Faces. In *Computer Graphics, SIGGRAPH Proceedings*, Los Angeles, CA, August 1999.

[4] J. F. Blinn. A Generalization of Algebraic Surface Drawing. *ACM Transactions on Graphics*, 1(3):235–256, 1982.

[5] J. Bloomenthal and K. Shoemake. Convolution Surfaces. *SIGGRAPH*, 25(4):251–256, 1991.

[6] I. Cohen, L. D. Cohen, and N. Ayache. Introducing new deformable surfaces to segment 3D images. In *Conference on Computer Vision and Pattern Recognition*, pages 738–739, 1991.

[7] S. Coquillart. Extended Free-Form Deformation: A sculpturing Tool for 3D Geometric Modeling. 24(4):187–196, 1990.

[8] M. Desbrun and M. Gascuel. Animating Soft Substances with Implicit Surfaces. pages 287–290, 1995.

[9] G. Farin. Surface over Dirichlet Tessellations. *Computer Aided Design*, 7:281–292, 1990.

[10] F. P. Ferrie, J. Lagarde, and P. Whaite. Recovery of Volumetric Object Descriptions from Laser Rangefi nder Images. In *European Conference on Computer Vision*, Genoa, Italy, April 1992.

[11] P. Fua. Regularized Bundle-Adjustment to Model Heads from Image Sequences without Calibration Data. *International Journal of Computer Vision*, 38(2):153–171, July 2000.

[12] W. Hsu, J. Hugues, and H. Kaufman. Direct manipulation of Free-Form Deformations. *SIGGRAPH*, 26(2):177–184, 1992.

[13] S. Ilic and P. Fua. Using Dirichlet Free Form Deformation to Fit Deformable Models to Noisy 3-D Data. In *European Conference on Computer Vision*, Copenhagen, Denmark, May 2002.

[14] R. K. B. J. C. Carr. Reconstruction and Representation of 3D Objects with Radial Basis Functions. pages 67–76, August 2001.

[15] W. R. F. J. C. Carr and R. K. Beatson. Surface Interpolation with Radial Basis Functions for Medical Imaging. *IEEE Transactions on Medical Imaging*, 16(1):96–107, 1997.

[16] I. Kakadiaris and D. Metaxas. Model based estimation of 3d human motion with occlusion based on active multiviewpoint selection. In *Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, June 1996.

[17] D. G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(441-450), 1991.

[18] L. Moccozet and N. Magnenat-Thalmann. Dirichlet Free-Form Deformation and their Application to Hand Simulation. 1997.

[19] H. Nishimura and al. 1985.

[20] A. Pentland and S. Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:715–729, 1991.

[21] R. Plänkers. 3–D Human Body Modeling from Video Sequences. Diploma project, EPFL, 1998.

[22] R. Plänkers and P. Fua. Articulated Soft Objects for Video-based Body Modeling. In *International Conference on Computer Vision*, pages 394–401, Vancouver, Canada, July 2001.

[23] R. Plänkers and P. Fua. Tracking and Modeling People in Video Sequences. *Computer Vision and Image Understanding*, 81:285–302, March 2001.

[24] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes, the Art of Scientific Computing*. Cambridge U. Press, Cambridge, MA, 1986.

[25] S. Roy and I. Cox. A Maximum-Flow Formulation of the N-camera Stereo Correspondence Problem. In *International Conference on Computer Vision*, pages 492–499, Bombay, India, 1988.

[26] T. Sederberg and S. Parry. Free-Form Deformation of Solid Geometric Models. 20(4), 1986.

[27] R. Sibson. A vector identity for the Dirichlet Tessellation. In *Math. Proc. Cambridge Philos. Soc.*, pages 151–155, 1980.

[28] E. M. Stokely and S. Y. Wu. Surface parameterization and curvature measurement of arbitrary 3-d objects: fi ve practical methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):833–839, August 1992.

[29] S. Sullivan, L. Sandford, and J. Ponce. Using geometric distance fi ts for 3–d. object modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(12):1183–1196, December 1994.

[30] R. Szeliski and D. Tonnesen. Surface Modeling with Oriented Particle Systems. In *Computer Graphics, SIGGRAPH Proceedings*, volume 26, pages 185–194, July 1992.

[31] D. Terzopoulos and M. Vasilescu. Sampling and reconstruction with adaptive meshes. In *Conference on Computer Vision and Pattern Recognition*, pages 70–75, 1991.

[32] G. Turk and J. F. O'Brien. Shape transformation using variational implicit surfaces. pages 335–342, August 1999.

[33] B. Wyvill and K. van Overveld. Warping as a modelling tool for csg/implicit models. In *Shape Modelling Conference, University of Aizu, Japan*, pages 205–214. IEEE Society Computer Press ISBN0-8186-7867-4, March 1997. inivited.

[34] G. Wyvill and B. Wyvill. Data Structure for Soft Objects.