Shape Matching Based on Fully Automatic Face Detection on Triangular Meshes

Wolfram von Funck, Holger Theisel, and Hans-Peter-Seidel

MPI Informatik, 66123 Saarbruecken, Germany,
{wfunck,theisel,hpseidel}@mpi-inf.mpg.de

Abstract. This paper tackles a particular shape matching problem: given a data base of shapes (described as triangular meshes), we search for all shapes which describe a human. We do so by applying a 3D face detection approach on the mesh which consists of three steps: first, a local symmetry value is computed for each vertex. Then, the symmetry values in a certain neighborhood of each vertex are analyzed for building sharp symmetry lines. Finally, the geometry around each vertex is analyzed to get further facial features like nose and forehead. We tested our approach with several shape data bases (e.g. the Princeton Shape Benchmark) and achieved high rates of correct face detection.

1 Introduction

Due to the fast development of new 3D scanning and modelling techniques, the number and complexity of surfaces which Computer Graphics deals with is currently dramatically increasing. Because of this, the retrieval and search of shapes in the internet becomes an important and challenging issue. Shape matching aims in choosing shapes of certain characteristics out of a shape data base. These characteristics are usually the similarity to a given shape which is described either as a particular surface, a sketching, or a rather abstract description [1]. Recently, a number of shape matching approaches have been developed which are based on shape histograms [2], extended Gaussian images [3], spherical extend functions [4, 5], shape distributions [6], spherical harmonics [7], light fields [8], 3D Fourier transforms [9], the topology of Reeb graphs [10], or anisotropy [11]. A part-in-whole approach was introduced by [12] which allows searching 3D models with parts matching a query.

The problem we want to tackle in this paper is a particular shape matching problem which can be formulated as follows:

Problem 1. Given a data base of shapes, get all which describe a human.

The main application of this problem relates to the applications of shape matching in general: imagine a user who wants to build up a virtual 3D scene of many different humans in different positions. Instead of modelling them, he or she may search the internet for appropriate shapes.

Figure 1 shows a number of shapes which obviously describe humans in different positions and states of completeness. Contrary, figure 3 shows a number of shapes which do not describe humans. Note that the above-mentioned shape matching approaches detect significantly different shapes in the examples of figure 1. This is due to the fact that we did not make any assumption about the position of the human body. Arms and legs may be outstretched (figure 1 (a)) or bent (figure 1 (b)), merged with the rest of the body (figure 1 (c)) or even non-existing at all (figure 1 (d), (e)). Since shape matching algorithms are sensitive against these features, they tend to give a higher similarity for instance between the shapes in figures 1 (a) and 3 (c) than between the shapes 1 (a) and 1 (c). Hence, the above-mentioned shape matching approaches are not suitable for problem 1.

Our starting point for a solution of problem 1 lies in the assumption that a shape describing a human should contain the human's face. (In fact, this seems to be the only property which all examples of figure 1 have in common.) Hence we can reformulate problem 1 to

Problem 2. Given a data base of shapes, find all shapes which contain a human's face.

Note that problem 2 does not make any assumption about size and location of the face. Neither it does about size and resolution of the model. The only assumption we use is that – if the shape describes a human – only one human (and nothing more) is contained in the shape. We also assume that the shapes are described as triangular meshes, and – if a face is present – the part of the mesh representing the face has a disc-like topology, i.e., it is a manifold without holes. Then problem 2 appears to be a *face detection problem* for triangular meshes.

A variety of algorithms for detecting faces in images has been developed which roughly can be classified into knowledge base methods [13, 14], feature based methods [15, 16], template matching [17], and appearance based methods. Having these algorithms available, a straightforward approach to detect faces on meshes is to render the meshes from different view points and then apply 2D face detection methods on the resulting images. However, this approach appears to be not reliable because of two reasons: First, there is no control about how many and which view points to choose for rendering. Second, there is no texture information in the mesh which gives for instance different colors for a face and the surrounding hair. Because of this, we have to apply face detection approaches which work directly on the meshes.

A well-researched problem on triangular meshes is the problem of face recognition [18–21]. For this class of problems an a-priory knowledge about the location of a face is assumed. In this sense, our problem 2 can be considered as a preceding step of 3D face recognition.

The face of a human is approximately mirror-symmetric. This fact – already used for face detection in 2D images [22] – gives the key of our approach: we search for face symmetry lines as shown in figure 2 (a).

The paper is organized as follows: Section 2 describes our detector for the case that the size (given by a radius) of the face is known. Section 3 extends this to the case of an unknown radius of influence. In section 4 we apply our algorithm to several representative data sets. Section 5 draws conclusions and mentions issues in future research.

2 Our Approach - Single Search Radius

Symmetry is a feature which is well-researched in Computer Vision both for images and for 3D objects. Generally, two kinds of symmetry can be distinguished: rotation and mirror symmetry where for our purposes we are interested in the last-named.

For 2D images, most of the existing work considers symmetry as a binary feature (i.e., and object is symmetric or not [23, 24]). In addition, [25, 26, 22, 27] compute symmetry as a local feature and apply it to detect faces in 2D images. For 3D objects, mirror symmetry is usually considered as a global feature. This means that a main symmetry plane is searched [28], or all symmetry planes through the center of gravity are evaluated [29]. What we need for our purpose is a local symmetry detection on a surface. This means that we need two pieces of information for each surface point: the strength of local symmetry ("how symmetric is the surface at a certain point?"), and the best symmetry axis. These values depend on the choice of a search radius r: only the parts of the surface with a distance smaller than r are incorporated in the local symmetry analysis. Thus, the evaluation of local symmetry is always connected to a particular choice of r. The best detection of a symmetry line in a face can be expected if r is approximately half the diameter of the face. If r is larger, other parts of the human body influence the analysis while a smaller r detects too many symmetry lines on a face.

In this section we describe our symmetry-based face detection approach for the case of a particular given symmetry radius r. This means that we decide whether or not the mesh contains a face of approximately the diameter $2r$. For doing so, we start to compute symmetry values for each vertex.

2.1 Computing the Symmetry Field

Let M be a triangle mesh with vertices $V(M)$. For each vertex $v \in V(M)$, $\mathbf{p}_v$ denotes the vertex position, while $\mathbf{n}_v$ is the (exact or estimated) normal in v. Then let $\operatorname{dist}(\mathbf{p}, \mathbf{d})$ denote the minimum (signed) distance of point $\mathbf{p}$ along direction vector $\mathbf{d}$ to the surface defined by M, where $\operatorname{dist}(\mathbf{p}, \mathbf{d}) = \infty$ if there is no intersection. We compute $\operatorname{dist}(\mathbf{p}, \mathbf{d})$ by a raytracing approach using a kd-tree.

For each vertex v of the mesh, we sample a height field on a circle in the tangent plane through $\mathbf{p}_v$ by measuring the distance in normal direction to the mesh at a number of sample points (figure 2 (b)). In order to discard small-scale variations of the normals, we use an average surface normal $\tilde{\mathbf{n}}_v$ of all vertices which have an Euclidian distance smaller than r:

$$\tilde{\mathbf{n}}_v = \frac{\sum_{u \in N_v} w_u \mathbf{n}_u}{\| \sum_{u \in N_v} w_u \mathbf{n}_u \|} \text{ with } N_v = \{u \in V(M) : \|\mathbf{p}_v - \mathbf{p}_u\| \leq r\}, \tag{1}$$

where w_u is the total area of the triangle fan surrounding vertex u. $\tilde{\mathbf{n}}_v$ can be considered as a strong smoothing at v. Now, we choose two orthogonal normal vectors $\mathbf{x}_v$ and $\mathbf{y}_v$ which are perpendicular to $\tilde{\mathbf{n}}_v$. We sample the height field in this plane on n_C concentric circles where n_R sample points are placed equidistantly on each circle (we used are $n_C = 8$ and $n_R = 64$). This way we get all sample points as

$$\mathbf{s}_v(i_C, i_R) = \mathbf{p}_v + \frac{i_C + 1}{n_C} \cdot r \cdot \cos\left(\frac{i_R}{n_R} \cdot 2\pi\right) \cdot \mathbf{x}_v + \frac{i_C + 1}{n_C} \cdot r \cdot \sin\left(\frac{i_R}{n_R} \cdot 2\pi\right) \cdot \mathbf{y}_v \tag{2}$$

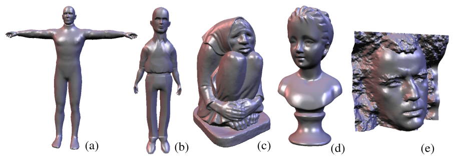

Fig. 1. A collection of human shapes.

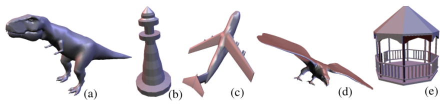

Fig. 3. Non-human shapes.

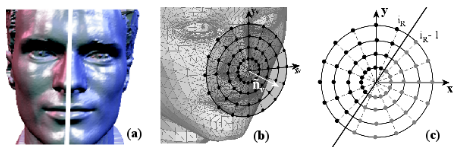

Fig. 2. (a) A face with a symmetry axis. (b) Sampling the surface around vertex v. (c) The mirror axis corresponding to rotation index i_R.

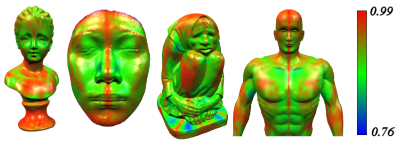

Fig. 4. Some meshes and their symmetry fields.

for $i_C = 0,...,n_C - 1$ and $i_R = 0,...,n_R - 1$. We can compute the height map h_v at each sample point as

$$h_v(i_C, i_R) = \mathrm{dist}(\mathbf{s}_v(i_C, i_R), -\tilde{\mathbf{n}}_v). \tag{3}$$

Using this, measuring the symmetry of the surface surrounding v is quite straightforward. As illustrated in figure 2 (c), each rotation index corresponds to a mirror axis that can be used for symmetry analysis. Basically, we simply have to compare the sample values on the one side of the mirror axis with the values on the other side. Since we want to constrain the analysis to the surface close to v, we limit the valid height map values to the range $[-r, r]$. Hence for each mirror axis, defined by rotation index i_R, we define a set of valid mirror pairs $M_v(i_R) = \{(i_C, i_{R1}, i_{R2}) : h_v(i_C, i_{R1}) \in [-r, r] \wedge h_v(i_C, i_{R2}) \in [-r, r] \wedge i_{R2} = 2i_R - i_{R1} - 1\}$. Now we can measure the error between both sides of the mirror axis by computing the mean difference between the sample values of all mirror pairs:

$$e_v(i_R) = \frac{1}{|M_v(i_R)|} \sum_{(i_C, i_{R1}, i_{R2}) \in M_v(i_R)} \|h_v(i_C, i_{R1}) - h_v(i_C, i_{R2})\| \tag{4}$$

To get a meaningful symmetry measure from this value, we normalise it by dividing it by the maximum difference of valid height map values. We define the symmetry measure of v as:

$$s_v = 1 - \frac{\min\limits_{i_R \in \{0,...,n_R-1\}} e_v(i_R)}{\max\limits_{(i_C, i_R) \in D_v} h_v(i_C, i_R) - \min\limits_{(i_C, i_R) \in D_v} h_v(i_C, i_R)} \tag{5}$$

where $D_v = \{(i_C, i_R) \in \{0,...,n_C - 1\} \times \{0,...,n_R - 1\} : h(i_C, i_R) \in [-r, r]\}$ is the set of valid samples. Furthermore, we can compute the normal of the corresponding mirror plane as follows: First, we get the rotation index $i_v = \mathrm{argmin}_{i_R \in \{0,...,n_R-1\}} e_v(i_R)$ and the corresponding rotation angle $\alpha_v = (i_v - \frac{1}{2}) \cdot \frac{2\pi}{n_R} + \frac{\pi}{2}$. Finally, we get the symmetry plane

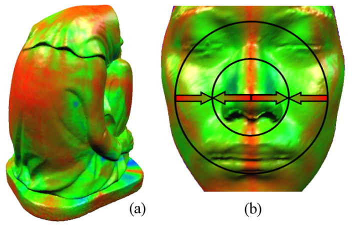

Fig. 5. (a) Smooth surfaces are highly symmetric. (b) Symmetry decrease in human faces.

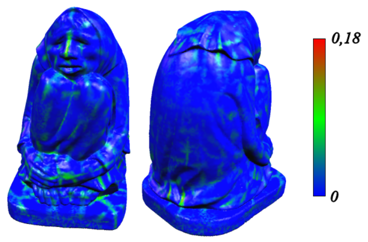

Fig. 6. The extracted symmetry lines.

normal:

$$\mathbf{m}_v = \cos(\alpha_v) \cdot \mathbf{x}_v + sin(\alpha_v) \cdot \mathbf{y}_v \tag{6}$$

2.2 Analysis of the Symmetry Field

As expected, the symmetry values are high for vertices located close to the horizontal center of a human face (see figure 4). Furthermore, the symmetry values are very low for most of the other vertices on the face. This results in a sharp line of high-symmetry vertices running from the forehead over the nasal bone to the mouth. Figure 5 (a) shows that other areas are detected to have a high symmetry as well. In the following section, we provide an algorithm that can be used to extract those significant areas.

2.3 Extracting Symmetry Features

Let $\text{sym}(\mathbf{p},\mathbf{d})$ denote the linearly interpolated symmetry value at the intersection point between the mesh surface and the ray from $\mathbf{p}$ along $\mathbf{d}$. Analogically to the height map h_v, we can sample the symmetry values around a vertex v:

$$\sigma_v(i_C, i_R) = \text{sym}(\mathbf{s}_v(i_C, i_R), -\tilde{\mathbf{n}}_v) \tag{7}$$

This function uses the same sample distribution as h_v. We want to extract those parts of the symmetry field that form narrow areas of high symmetry, bordered by areas of low symmetry. More precisely, we want to measure how much the symmetry values decrease in both directions orthogonal to the symmetry plane at vertex v. We can project every sample point into the symmetry plane normal $\mathbf{m}_v$ (i.e. compute its signed distance to the symmetry plane) and normalise it with respect to the symmetry radius r, obtaining a scalar value:

$$d_v(i_C, i_R) = \frac{(\mathbf{s}_v(i_C, i_R) - \mathbf{p}_v) \cdot \mathbf{m}_v}{r} \tag{8}$$

For each side of the mirror plane, we can define a set of two-dimensional points:

$$\begin{aligned} R_v &= \{(d_v(i_C,i_R), s_v - \sigma_v(i_C,i_R)) : (i_C,i_R) \in D_v \wedge i_C < \frac{n_C}{2} \wedge d_v(i_C,i_R) > 0\} \\ L_v &= \{(d_v(i_C,i_R), \sigma_v(i_C,i_R) - s_v) : (i_C,i_R) \in D_v \wedge i_C < \frac{n_C}{2} \wedge d_v(i_C,i_R) < 0\} \end{aligned} \tag{9}$$

The first coordinate of each point corresponds to the normalised distance of sample (i_C, i_R) to the symmetry plane. The second coordinate is the difference between the symmetry value of v and the symmetry value of sample (i_C, i_R), where this difference is negated in R_v. The reason for this negation will become clear in the next step. Note that only samples whose circle index i_C is smaller than $\frac{n_C}{2}$ are considered because it has turned out that the decrease of symmetry in human faces ranges from the face center approximately to half of the symmetry radius (see figure 5 (b)). In the next step, we fit a line through origin to the points in $R_v \cup L_v$. Since we negated the symmetry coordinates of the points in R_v, the gradient of the line will be positive if the symmetry decreases in both directions. The value of the gradient is computed as follows:

$$g_v = \frac{\sum_{(d,s)\in R_v \cup L_v} d \cdot s}{\sum_{(d,s)\in R_v \cup L_v} d^2} \tag{10}$$

The complete extraction process is depicted in figure 7. Expressed graphically, g_v weakens the large high-symmetry areas of the symmetry field mentioned above and intensifies the narrow lines of high symmetry as found in human faces (figure 6)). Interestingly, a common threshold seems to exist for all human faces that can be used to classify a vertex (more precisely its surrounding surface) as symmetric or non-symmetric. Provided that the symmetry radius r matches approximately the size of the face, by marking only those vertices whose gradient value g_v exceeds the threshold, a complete line of vertices running from the root of the nose to the nose tip is marked for all kinds of human faces (see figure 11). In our implementation, we used the threshold $t_{Sym} = 0.06$. This way, the number of potential face vertices has been decimated by a large amount and we even have an indication for the orientation of the face (symmetry plane direction m_v) as well as for the size of the face (radius r).

2.4 Analysis of the Face Geometry

In the following section, we examine the surrounding surfaces of all vertices that have been classified symmetric. More precisely, we try to find out if these vertices are located on the nose tip of a human face. Given a potential "nose tip vertex" v, we know that the corresponding face has two possible up-directions:

$$\mathbf{u}_v^1 = \tilde{\mathbf{n}}_v \times \mathbf{m}_v \text{ and } \mathbf{u}_v^2 = -\tilde{\mathbf{n}}_v \times \mathbf{m}_v \tag{11}$$

From now on, the up-vector is simply referred to as $\mathbf{u}_v^a$ since the algorithm works analogically for $\mathbf{u}_v^1$ and $\mathbf{u}_v^2$.

First we analyse the curves running horizontally from the nose over the cheeks as illustrated in figure 8 (a). Since the curves may run both over the left and the right side, we define a direction vector $\mathbf{d}_v^b$ with $\mathbf{d}_v^1 = \mathbf{m}_v$ and $\mathbf{d}_v^2 = -\mathbf{m}_v$. Given the number of curves n_Y, the i_Y-th curve is defined by

$$c_v^{a,b}(i_Y, x) = \frac{1}{r}\mathrm{dist}(\mathbf{p}_v + \frac{i_Y}{n_Y - 1} \cdot \frac{r}{2} \cdot \mathbf{u}_v^a + x \cdot \mathbf{d}_v^b, -\tilde{\mathbf{n}}_v). \tag{12}$$

First of all, we measure how far the potential nose sticks out of the face with respect to the cheeks (figure 8 (b)):

$$noseheight_v^{a,b} = \frac{1}{n_Y} \sum_{i_Y=0}^{n_Y-1} (c_v^{a,b}(i_Y, \frac{r}{2}) - c_v^{a,b}(i_Y, 0)) \tag{13}$$

Next, we measure the mean nose width. For each curve, we define the width as the position within the range $[0, \frac{3}{4}r]$ where the curve gradient is maximal (figure 8 (c)). Given n_X samples per curve, we get the mean nose width

$$nosewidth_v^{a,b} = \frac{1}{n_Y} \sum_{i_Y=0}^{n_Y-1} w_{i_Y} \tag{14}$$

with $w_{i_Y} = \frac{3}{4n_X} \mathrm{argmax}_{i_X \in \{1,\ldots,n_X\}} (c_v^{a,b}(i_Y, \frac{i_X}{n_X} \cdot \frac{3}{4}r) - c_v^{a,b}(i_Y, \frac{i_X - 1}{n_X} \cdot \frac{3}{4}r))$.

All curves should begin with a bulge and end with a relatively flat region. Figure 10 (a) shows how we can measure two heights on the curve whose difference gives us a meaningful value. By computing the mean value of all curves we get

$$\begin{aligned} nosecurve_v^{a,b} = \frac{1}{n_Y} \sum_{i_Y=0}^{n_Y-1} (c_v^{a,b}(i_Y, w_{i_Y} + \frac{r}{4}) - c_v^{a,b}(i_Y, 0) \\ -|c_v^{a,b}(i_Y, w_{i_Y} + \frac{r}{2}) - c_v^{a,b}(i_Y, w_{i_Y} + \frac{r}{4})|). \end{aligned} \tag{15}$$

The nose and the cheeks are quite smooth and contain no cracks. We cope with this fact by fitting a quadratic B-spline to each curve and measuring the error, as illustrated in figure 10 (b). Let $\mathrm{splinedist}_v^{a,b}(i_Y)$ denote the maximum height difference between curve i_Y and its corresponding B-spline. Then the mean smoothness error is defined as:

$$facesmoothness_v^{a,b} = \frac{1}{n_Y} \sum_{i_Y=0}^{n_Y-1} \mathrm{splinedist}_v^{a,b}(i_Y) \tag{16}$$

The nose bridge is relatively smooth and contains no cracks. Hence we measure the maximum distance between the nose profile and the straight line from the nose tip to the root (figure 10 (c)). Given the nose profile as height function $\mathrm{nose}_v^a(y) = \frac{1}{r}\mathrm{dist}(p_v + y \cdot \mathbf{u}_v^a, -\tilde{\mathbf{n}}_v)$, we can define the line as

$$noseline_v^a(y) = y \cdot \frac{\mathrm{nose}_v^a(\frac{3}{4}r)}{\frac{3}{4}r} \tag{17}$$

We suspect the nose root of being located at a distance of $\frac{3}{4}r$ upwards the nose tip. Then we can measure the error with

$$nosesmoothness_v^a = \mathrm{maxdist}(\mathrm{nose}_v^a, \mathrm{noseline}_v^a) \tag{18}$$

Noses stick out of the face, especially with respect to the region directly below the nose tip (figure 10 (d)). Thus, we measure the minimum distance to this spot.

$$nosebottom_v^a = \frac{1}{r} \min_{i_X \in \{-n_X,\ldots,n_X\}} \mathrm{dist}(\mathbf{p}_v - 0.2 \cdot r \cdot \mathbf{u}_v^a + \frac{i_X}{2n_X} r \cdot \mathbf{m}_v, -\tilde{\mathbf{n}}_v) \tag{19}$$

Next, we measure the smoothness of the forehead by sampling the heights of a rectangular region on the forehead (figure 9 (a)):

$$headsmoothness_v^a = \frac{\max H - \min H}{r} \tag{20}$$

with $H = \mathrm{dist}(\mathbf{p}_v + (1.3 + 0.2\frac{i_Y}{n_Y - 1} r)\mathbf{u}_v^a + 0.5\frac{i_X}{n_X}\mathbf{m}_v, \tilde{\mathbf{n}}_v) : (i_X, i_Y) \in \{-n_X, ..., n_X\} \times \{0, ..., n_Y - 1\}\}$.

Another feature of human faces is the convexity of the forehead (figure 9 (b)). Hence we compute the difference between the central and outer height of the forehead:

$$headconvexity_v^{a,b} = \frac{1}{r}(\mathrm{dist}(\mathbf{p}_v + 1.3 \cdot \mathbf{u}_v^a + 0.8 \cdot \mathbf{d}_v^b, -\tilde{\mathbf{n}}_v) - \mathrm{dist}(\mathbf{p}_v + 1.3 \cdot \mathbf{u}_v^a, -\tilde{\mathbf{n}}_v)) \tag{21}$$

The last property we examine are the eyes. Depending on the mesh resolution the eyes are described more or less detailed, but the eye-sockets should always be present. As figure 9 (c) shows, the eye-sockets are located deeper in the head than the forehead. We scan the region where the eye is assumed and compute the difference between the largest height value and the height of the forehead center:

$$\begin{aligned} eyedepth_v^{a,b} = \frac{1}{r}(\max\{\mathrm{dist}(\mathbf{p}_v + (0.5 + 0.4\frac{i_Y}{n_Y - 1} r)\mathbf{u}_v^a + (0.2 + 0.3\frac{i_X}{n_X - 1})\mathbf{d}_v^b, -\tilde{\mathbf{n}}_v) :& \\ (i_X, i_Y) \in \{0, ..., n_X - 1\} \times \{0, ..., n_Y - 1\}\}& \\ -\mathrm{dist}(\mathbf{p}_v + \mathbf{u}_v^a, -\tilde{\mathbf{n}}_v))& \end{aligned} \tag{22}$$

Using all the measures defined above, we impose the following set of constraints that need to be fulfilled if a vertex v is located on the nose tip of a human face:

$$\begin{aligned} noseheight_v^{a,1} > c_{NH} \wedge noseheight_v^{a,2} > c_{NH} \wedge \\ nosewidth_v^{a,1} < c_{NW} \wedge nosewidth_v^{a,2} < c_{NW} \wedge \\ nosecurve_v^{a,1} > c_{NC} \wedge nosecurve_v^{a,2} > c_{NC} \wedge \\ facesmoothness_v^{a,1} < c_{FS} \wedge facesmoothness_v^{a,2} < c_{FS} \wedge \\ eyedepth_v^{a,1} > c_{ED} \wedge eyedepth_v^{a,2} > c_{ED} \wedge \\ headconvexity_v^{a,1} > c_{HC} \wedge headconvexity_v^{a,2} > c_{HC} \wedge \\ nosesmoothness_v^a < c_{NS} \wedge nosebottom_v^a > c_{NB} \wedge headsmoothness_v^a < c_{HS} \end{aligned} \tag{23}$$

The parameters c_{NH}–c_{HC} are supposed to be constant for all meshes and can be trained (manually) with a database of human and non-human meshes (see section 4).

3 Our Approach - All Radii

Up to now our face detection approach was based on a particular choice of the search radius r: this way faces of the approximate diameter $2r$ are detected (or excluded) on a mesh. In fact, all thresholds and parameters of the approach are tuned to depend exclusively on r. For the complete solution of problem 2, we would have to apply the algorithm for all r. However, the following observations lead to the results that only a certain number of search radii have to be checked: First, the face detection algorithm appears to be rather stable against small variations of r. In fact, a face with a diameter fd is generally detected for any choice of r between $0.7\frac{fd}{2}$ and $1.2\frac{fd}{2}$. Second, giben the size d of the whole mesh (which we estimate by the length of the diagonal of the minimal enclosing bounding box), the diameter fd of the face is limited to a certain interval. If the mesh describes a complete stretched-out human, we can estimate the size of the face to be not smaller than 5% of the size of the mesh ($fd \geq 0.05d$). On the other hand, if the mesh describes only a face, then the size of the mesh and the face coincide ($fd \leq d$). Because of this, for each mesh we check 32 different search radii $r_0,...,r_{31}$ which are chosen as $r_0 = 0.05\,\frac{d}{2}$, $r_{31} = 0.7\,\frac{d}{2}$, and the remaining r_i are placed in a quadratic distribution between r_0 and r_{31} allowing a higher density for smaller radii. This way our algorithm becomes independent of any parameter.

4 Applications and Results

We trained the parameters c_{NH}–c_{HC} of the geometry constraints manually using the Princeton Shape Benchmark and found the following configuration: $c_{NH} = 0.2$, $c_{NW} = 0.4$, $c_{NC} = 0.2$, $c_{FS} = 0.2$, $c_{NS} = 0.1$, $c_{NB} = 0.1$, $c_{ED} = 0.1$, $c_{HS} = 0.2$, $c_{HC} = 0.01$. In order to test our approach, we applied it to several shape databases: The Princeton Shape Benchmark [30] (our training database), the CCCC database [5], the Utrecht database [31] and the aim@shape database [32]. Altogether, we tested 4429 meshes. Most of the databases provide shape classifications like “human”, “human_arms_out”, “head”, “face”. However, these classifications are inappropriate for our purpose due to the following reasons: many of the shapes classified as human have holes (figure 14 (b)) or don’t contain human faces (figure 14 (c)). Therefore, we identified the human faces in each database manually in order to evaluate the algorithm.

The Princeton Shape Benchmark consists of 1814 meshes. We identified 141 human faces without holes. Many of these faces are very coarse and have non-human features (figure 14 (a)). For this database, the algorithm detected 51 meshes to be human. All detected shapes are indeed human, i.e. no non-human mesh was found. The CCCC database contains 1841 meshes. We identified 49 valid faces, our algorithm detected 20. Again, no “wrong” face was detected. The Utrecht database consists of 684 meshes and contains no human face. The algorithm correctly detected no face in this database. Finally, we applied the algorithm to 90 high-resolution meshes of the aim@shape repository. 16 meshes have human faces, where 6 faces contain holes or are incomplete. The algorithm detected 12 faces (figure 13) – although two detected faces contain holes – and there was no incorrect detection.

The evaluation shows that the algorithm is able to detect humans in different states of completeness: complete bodies (figure 12 left), incomplete bodies, heads and single faces (figure 12 right). For a better visualisation, the application automatically displays “glasses” on each detected face and marks the nose tip red. Furthermore, there was no incorrect detection in all tested meshes. However, the algorithm cannot detect faces with non-human features like non-convex foreheads, or faces hidden by masks, glasses or hair. There were totally 11 non-detected meshes with one of these properties. The remaining meshes that have not been detected are very coarse (figure 14 (a) shows some examples): the number of face triangles lies between 50 and 400 and averages to 150. In contrast, the average triangle number of the detected faces amounts approximately to 4000, i.e. the algorithm requires a certain resolution of the face meshes in order to work reliably. The computing time for our approach is approximately linear to the number of vertices in the mesh: an AthlonXP 2400+ processor with 512 MB RAM took approx. 15 minutes for a mesh with 20000 triangles, 40 minutes for 60000 triangles, and 70 minutes for 100000 triangles. Since in our application scenario each mesh of a data base has to be checked only once and the result can be stored with the mesh, our algorithm can be considered as a preprocess of a web-search for meshes describing humans.

5 Conclusions

In this paper we made the following contributions: We introduced a domain-specific shape matching approach which is based on a fully-automatic face detection on triangular meshes. To decide whether a mesh contains a human face, each vertex undergoes a three-step test for being part of a face. This test is repeated for different radii of influence to ensure that faces of arbitrary scaling are detected. We trained the parameters of our algorithm and applied it to a number of different shape databases. No wrong face was detected. In general, we detected most faces as long as they did not contain holes and had a sufficently high triangular resolution. We conclude that, given that the meshes describe human faces in enough detail, the algorithm is able to differentiate between human and non-human meshes very reliably. For future research we intend to make the algorithm more robust against holes in the meshes, since a number of meshes comes with holes in the eye and mouth regions.

References

1. Funkhouser, T., Min, P., Kazhdan, M., Chen, J., Halderman, A., Dobkin, S.: A search engine for 3D models. ACM Transactiond on Graphics **22**(1) (2003) 83–105
2. Ankerst, M., Kastenmüller, G., Kriegel, H.P., Seidl, T.: Nearest neighbor classification in 3D protein data bases. In: Proc. 7th International Conference on Intelligent Systems for Molecular Biology. (1999) 34–43
3. Kang, S., Ikeuchi, K.: Determining 3-D object pose using the complex extended gaussian image. In: IEEE Conf. on Comp. Vision and Patt. Recog. (1991) 580–585
4. Saupe, D., Vranic, D.: 3D model retrieval with spherical harmonics and moments. In: Proceedings of the 23rd DAGM-Symposium on Pattern Recognition. (2001) 392–397
5. Vranic, D.: An improvement of rotation invariant 3D shape descriptor based on functions on concentric spheres. In: Proc. IEEE International Conference on Image Processing (ICIP 2003),. (2003) 757–760
6. R.Osada, T.Funkhouser, B.Chazelle, D.Dobkin: Shape distributions. ACM Trans. Graph. **21**(4) (2002) 807–832
7. Kazhdan, M., Funkhouser, T., Rusinkiewicz, S.: Rotation invariant spherical harmonic representation of 3D shape descriptors. In: SGP '03: Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing. (2003) 156–164
8. Chen, D., X. Tian, Y., Ouhyoung, M.: On visual similarity based 3d model retrieval. Computer Graphics Forum **23**(3) (2003) 223–232 (Proceedings Eurographics 2003).

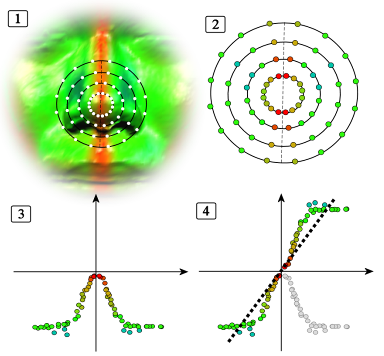

Fig. 7. 1.+2. Sampling of symmetry values. 3. Each sample is transformed to a 2D point (x = distance to symmetry plane, y = symmetry difference). 4. Fitting a line.

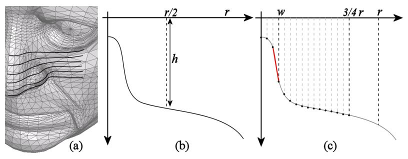

Fig. 8. (a) The curves that are analysed. (b) Computing the nose height. (c) The nose width w.

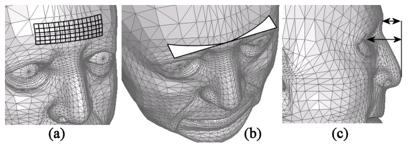

Fig. 9. (a) Measuring the forehead smoothness. (b) The human forehead is convex. (c) The eyes are located deeper in the head than the forehead.

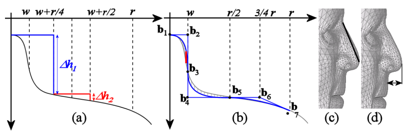

Fig. 10. (a) $\Delta h_1 - \Delta h_2$ is used to compute $nosecurve_v^{a,b}$. (b) We fit a quadratic B-spline to the curve. (c) The deviation between the nose profile and the straight line is very low. (d) The nose tip sticks out.

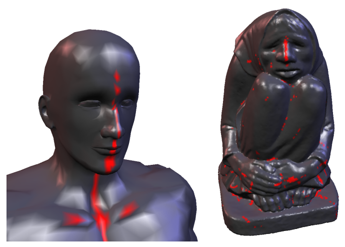

Fig. 11. Marked vertices.

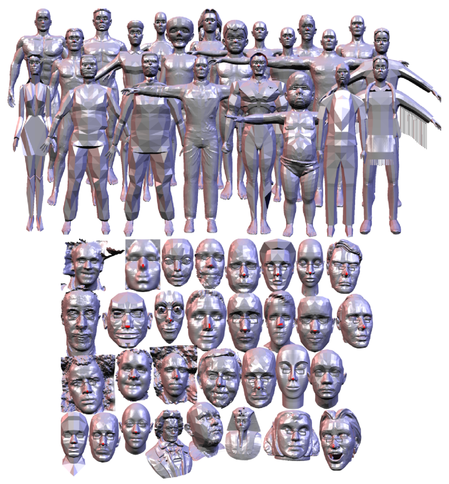

Fig. 12. Faces detected in complete bodies (top) and incomplete bodies (bottom).

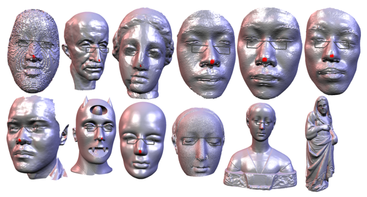

Fig. 13. Faces in the aim@shape database.

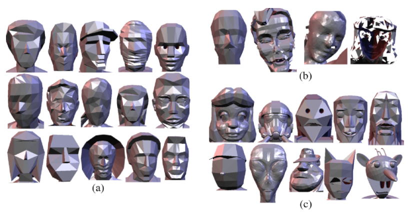

Fig. 14. Faces that could not be detected.

9. Vranic, D., Saupe, D.: 3D shape descriptor based on 3D fourier transform. In: Proc. ECMCS 2001. (2001) 271–274
10. Hilaga, M., Shinagawa, Y., Kohmura, T., Kunii, T.: Topology matching for fully automatic similarity estimation of 3D shapes. In: Proc. SIGGRAPH. (2001) 203–212
11. Kazhdan, M., Funkhouser, T., Rusinkiewicz, S.: Shape matching and anisotropy. In: Proc. SIGGRAPH. (2004) 623–629
12. Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., Dobkin, D.: Modeling by example. ACM Transactions on Graphics (SIGGRAPH 2004) (2004)
13. Yang, G., Huang, T.: Human face detection in a complex background. Pattern Recognition **27**(1) (1994) 53Ű–63
14. Kotropoulos, C., Pitas, I.: Rule-based face detection in frontal views. In: ICASSP '97: Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97) -Volume 4, IEEE Computer Society (1997) 2537
15. Leung, T., Burl, M., Perona, P.: Finding faces in cluttered scenes using random labeled graph matching. In: Proc. Fifth International Conference on Computer Vision. (1995) 637–Ű644
16. Yow, K., Cipolla, R.: Feature-based human face detection. Technical report, Department of Engineering, University of Cambridge, England (1996)
17. Sinha, P., Torralba, A.: Detecting faces in impoverished images. Journal of Vision **2**(7) (2002) 601a
18. Gordon, G.: Face recognition based on depth and curvature features. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition. (1992) 108–110
19. Hallinan, P., et al.: Two- and Three-Dimensional Patterns on the Face. AK Peters, Natick (1999)
20. Huang, J., Heisele, B., Blanz, V.: Component-based face recognition with 3D morphable models. In: Proc. of the 4th Int. Conf. on Audio- and Video-Based Biometric Person Authenticitation. (2003) 27–34
21. Blanz, V., Vetter, T.: Face recognition based on fitting a 3D morphable model. IEEE Trans. Pattern Anal. Mach. Intell. **25**(9) (2003) 1063–1074
22. Zabrodsky, H., Peleg, S., Avnir, D.: Symmetry as a continuous feature. IEEE Transactions on Pattern Analysis and Machine Intelligence **17**(12) (1995) 1154–1166
23. Marola, G.: On the detection of the axes of symmetry of symmetric and almost symmetric planar images. IEEE Trans. Pattern Anal. Mach. Intell. **11**(1) (1989) 104–108
24. Shen, D., Ip, H., Cheung, K., Teoh, E.: Symmetry detection by generalized complex (gc) moments: A close-form solution. IEEE Transactions on Pattern Analysis and Machine Intelligence **21**(5) (1999) 466–476
25. Reisfeld, D., Wolfson, H., Yeshurun, Y.: Detection of interest points using symmetry. In: ICCV90. (1990) 62–65
26. Reisfeld, D., Wolfson, H., Yeshurun, Y.: Robust facial feature detection using local symmetry. In: Proc. International Conference on Pattern Recognition. (1990) 117–120
27. Kovesi, P.: Symmetry and asymmetry from local phase. In: AI'97. (1997) 185–190
28. Sun, C., Sherrah, J.: 3D symmetry detection using the extended gaussian image. IEEE Trans. Pattern Anal. Mach. Intell. **19**(2) (1997) 164–168
29. M.Kazhdan, Chazelle, B., Dobkin, D., Finkelstein, A., Funkhouser, T.: A reflective symmetry descriptor. In: ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part II. (2002) 642–656
30. Shilane, P., Min, P., Kazhdan, M., Funkhouser, T.: The princeton shape benchmark. In: Proc. Shape Modeling International. (2004) 167–178
31. Tangelder, J., Veltkamp, R.: Polyhedral model retrieval using weighted point sets (2003)
32. aim@shape: (2004) http://www.aimatshape.net.