3D Shape from Silhouette Points in Registered 2D Images Using Conjugate Gradient Method

Andrzej Szymczak[a], William Hoff[b] and Mohamed Mahfouz[c]

[a] Department of Mathematical and Computer Sciences, Colorado School of Mines, Golden, CO 80401, USA.
[b] Department of Engineering, Colorado School of Mines, Golden, CO 80401, USA.
[c] Department of Mechanical, Aerospace and Biomedical Engineering, University of Tennessee, Knoxville, TN 37996, USA.

ABSTRACT

We describe a simple and robust algorithm for estimating 3D shape given a number of silhouette points obtained from two or more viewpoints and a parametric model of the shape. Our algorithm minimizes (in the least squares sense) the distances from the lines obtained by unprojecting the silhouette points to 3D to their closest silhouette points on the 3D shape. The solution is found using an iterative approach. In each iteration, we locally approximate the least squares problem with a degree-4 polynomial function. The approximate problem is solved using a nonlinear conjugate gradient solver that takes advantage of its structure to perform exact and global line searches. We tested our algorithm by applying it to reconstruct patient-specific femur shapes from simulated biplanar X-ray images.

Keywords: 3D reconstruction, silhouette points, biplanar X-ray, PCA based shape model, optimization

1. INTRODUCTION

In bone-related Computer Aided Surgery applications, one is often interested in constructing a patient-specific 3D model of the bone of interest, which can be rigidly registered to the patient during the procedure and be displayed to guide the surgeon. Such a model can be built from a Computed Tomography (CT) scan, but this is a time-consuming process that exposes the patient to large amount of radiation. A cheaper and faster approach is to use a number of X-ray images.

Reconstructing 3D bone models from X-ray images is a challenging problem because of limited amount of information provided by the input. This paper describes a robust algorithm for a closely related problem of reconstructing 3D shape from silhouette points extracted from 2D images and a parametric model of the shape. Our method essentially searches for a shape defined by the parametric model and its pose that makes it as close to tangent as possible to a set of lines obtained by unprojecting the input silhouette points from the input 2D images to 3D. In contrast to[1,2] out approach does not require complete silhouette.

Like many other algorithms, our method is based on minimizing the distance between the unknown shape and the lines defined by the input silhouette points in the least squares sense. In contrast to methods described in,[3,4] our algorithm is based on silhouette points rather than apparent silhouettes. Apparent silhouettes tend to be larger and more complex than silhouettes and typically give rise to more local extrema in the objective function. We also use a different optimization method: a customized version of the conjugate gradient method. We take advantage of the structure of the problem to make it perform global and exact line searches. Even though this approach still does not qualify as a global optimization method, it is very effective at avoiding local minima. The need for a global approach was recognized in,[5] where genetic algorithm with intensity and contour based scoring metric is used for optimization. The method of[6] uses an approach based on simulated annealing and improves upon earlier approach[7] that minimizes likelihood energy obtained from edge potential field and a prior energy term defined in terms of the statistical model using the gradient descent method.

The method of[8] is also based on silhouette points, but it uses a different optimization method. Also, the goal of that work is to segment bone contours from X-ray images: results of 3D shape reconstruction from X-ray pairs are not reported.

2. PROBLEM STATEMENT

The input to our procedure consists of:

- a parametric shape model (base mesh and a small number of eigenshapes – Section 2.1)
- the initial pose (location and orientation in space – Section 2.2)
- a number of silhouette points of the unknown shape in a set of registered 2D images.

In our current implementation, the user specifies the initial pose interactively by rotating and translating the average shape so that it is roughly aligned with the expected output. The alignment does not have to be accurate - in our femur experiments described in Section 4 we sometimes used initial orientation about 45 degrees off the output orientation, and the algorithm still is able to converge to the right solution.

The silhouette samples are also selected manually. In many cases, complex background makes it hard to decide where the silhouette is. For examples discussed in Section 4 we simply avoided sampling these areas and focused on the parts of silhouette that were clearly correct.

The output is an estimate of the shape parameters and the pose consistent with the images from which the silhouette samples were obtained.

2.1 Parametric model of the shape

The parametric model of the shape used in this work consists of the base triangle mesh M and eigenshapes E_1, E_2, ..., E_N. The base mesh and the eigenshapes can be obtained by fitting a template mesh (typically, of simple and regular connectivity) to a family of shapes. Then, PCA analysis of the vertex coordinate vectors is applied to the fitted template meshes to compute the average shape and describe their most significant deformation modes. The base mesh represents the average shape in the family and the eigenshapes represent the most significant allowable deformations of the base mesh. In this work, we focus on the space of femur shapes and use the parametric model constructed using the method of.[9]

If the base mesh has n vertices $v_0, v_1, \ldots, v_{n-1}$ then the eigenshapes consist of n vectors: $E_k = \left(e_0^k, e_1^k, \ldots, e_{n-1}^k\right)$ for $k \in \{1, 2, \ldots, N\}$. The shape corresponding to parameter vector $\bar{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_N]$, denoted by $M(\bar{\alpha})$, is the triangle mesh obtained from M by changing the coordinates of its i-th vertex from v_i to $v_i + \sum_{k=1}^{N} \alpha_k e_i^k$ while keeping the connectivity fixed. Our procedure works by searching for a shape (in the N-dimensional affine space of all meshes of the form $M(\bar{\alpha})$) and its pose that lead to silhouettes matching the user-specified silhouette points.

2.2 Pose

Our algorithm estimates both the shape parameters and the pose information. The pose is represented as a rotation matrix R and a translation vector T. In what follows, if R is a 3×3 matrix and S is a 3D triangle mesh, by $R[S]$ we shall denote the result of applying the transformation represented by R to the mesh S, i.e. the mesh obtained from S by applying the matrix R to each vertex. Similarly, by $S + T$, where T is a 3D vector, we denote the mesh S translated by T.

Our algorithm computes a pose update for the mesh at each iteration. The pose update is a pair (R, T), where $R \in \mathrm{SO}(3)$ ($\mathrm{SO}(3)$ denotes the group of 3D rotation matrices) and T is a translation vector. Generally, updating the pose of a mesh S boils down to substituting the current mesh S with the mesh $R[S] + T$ and applying the rotation matrix R to all component vectors of the eigenshapes (to make their orientation consistent with that of the updates mesh).

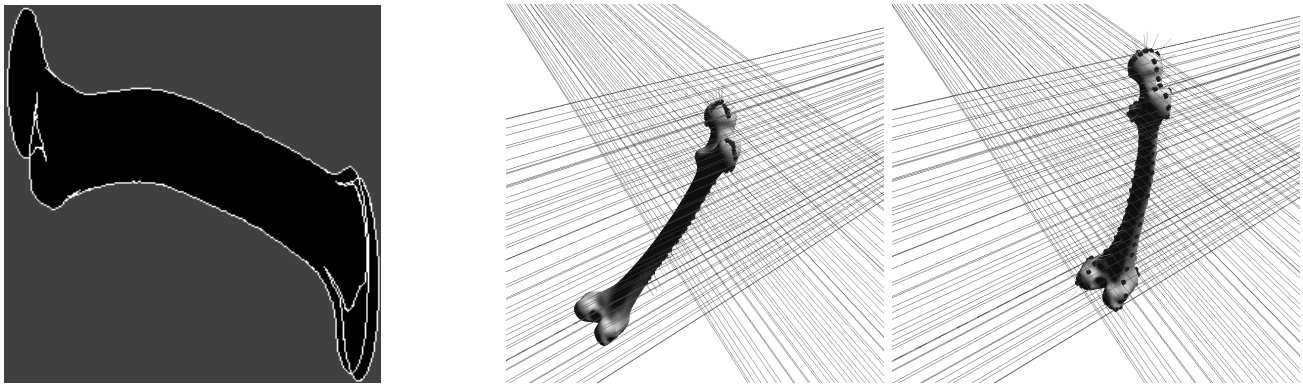

Figure 1. Left: An example of image used to determine the approximate silhouettes (with anisotropic pixels). Right: The initial shape and pose and shape and pose estimated by our algorithm. The lines were obtained from the silhouette points selected from the input images. The dark cubes indicate the location of silhouette points on the current shape that are closest to one of the lines.

3. OUR APPROACH

Our algorithm is based on least squares minimization of the distance between lines obtained by unprojecting the silhouette samples into the three-dimensional space to the silhouette of the unknown shape. In what follows, by L we denote the set of lines obtained from the silhouette samples. Each line $l \in L$ projects to a silhouette point in one of the input images. By e_l we denote the center of projection (X-ray source location) used to obtain that image.

The goal is to minimize the objective function equal to the sum (over all $l \in L$) of the squared distance from l to the silhouette of the mesh relative to e_l. The detailed description of the optimization method is given below.

3.1 Silhouettes

By a silhouette point of a surface S relative to a viewpoint e we mean any point p on S such that the intersection of the ray starting at e and passing through p intersects S at exactly one point. The set of all such points will be called *e-silhouette of S*. Exact computation of silhouette points can be computationally intensive. Therefore, our procedure uses a simple approximation that works very well in practice.

We scan-convert all *apparent silhouette edges*, i.e. edges that separate a front facing triangle from a back facing triangle relative to the viewpoint e. Let's say that the initial color of every pixel is black and all edges are drawn in white. The projection plane is selected so that all edges project to non-boundary pixels. Then, all pixels in the connected component of the set of black pixels that contains the boundary pixels are colored grey. Finally, we go over all edges in B and check if there is a gray neighbor for any pixel in the projection of the edge. If there is one, we declare *all* points on the edge as silhouette points. We are going to denote the approximate e-silhouette of a surface S by $\mathrm{Silh}(S, e)$. Note that it consists of edges of the surface S.

It would be easy to accelerate this procedure on a GPU, but our current implementation is entirely CPU-based. An example of image used to compute the approximate silhouette is shown in Figure 1.

3.2 Objective function

The objective function F that we minimize in this paper represents the least squares line-silhouette distance:

$$F(R, T, \bar{\alpha}) = \sum_{l \in L} \mathrm{dist}^2\big(l, \mathrm{Silh}(R[M(\bar{\alpha})] + T, e_l)\big). \tag{1}$$

Its domain is a Cartesian product $\mathrm{SO}(3) \times R^3 \times R^k$. Note that the distance between sets C and D (in the above equation, between a line and an approximate silhouette) is defined as the minimum distance between a point $c \in C$ and $d \in D$.

F is a complicated function that is costly to even *evaluate* for given arguments. Thus, we exploit its local approximation whose reliable minimization is much easier.

3.3 Local approximation

Notice the right hand side of Equation (1) is well defined if R is any matrix, not necessary a rotation. The tangent space $\mathcal{T}$ to SO(3) at the identity matrix $\mathbf{I}$ is the space of all anti-symmetric 3D matrices. This means that small rotations can be approximated (with quadratic accuracy) with matrices of the type $\mathbf{I} + A$ where A is an anti-symmetric matrix. The space of anti-symmetric matrices is isometric to R^3 and therefore easy to deal with. An example of a function that defines an isometry is

$$A(a,b,c) = \begin{bmatrix} 0 & a & b \\ -a & 0 & c \\ -b & -c & 0 \end{bmatrix}.$$

By replacing the SO(3) factor in the domain of F (Equation (1)) with the set of anti-symmetric matrices, we essentially flatten its domain, making it the $(k+6)$-dimensional Euclidean space.

The next goal is to limit the nonlinearity of F. Let $S = M(\bar{\alpha}_0)$ be the current approximation of the target shape. For each line $l \in L$ we find the point $q_l \in \mathrm{Silh}(S, e_l)$ that is closest to l by exhaustive search over all edges in $\mathrm{Silh}(S, e_l)$. We also find the vector f_l^k corresponding to q_l in the eigenshape E_k for $k = 1, 2, \ldots, N$. The point q_l is on an edge of S, and S has the same connectivity as the average shape M. Thus, q_l is a weighted average of the i-th and j-th vertex of S for some $i, j \in \{0, 1, \ldots, n-1\}$. f_l^k is obtained as the weighted average of e_i^k and e_j^k with the same weights.

The simplified objective function F_0 is defined by

$$F_0(a,b,c,T,\bar{\alpha}) = \sum_{l \in L} \mathrm{dist}^2\Big(l,\ [\mathbf{I} + A(a,b,c)](q_l + \sum_{k=1}^{N} \alpha_k f_l^k) + T\Big). \tag{2}$$

Since the line l passes through e_l, the squared distance from a given point p to l can be computed from the following formula

$$\mathrm{dist}^2(l,p) = |e_l - p|^2 - (\vec{e_l p} \cdot w_l)^2,$$

where w_l is a unit vector parallel to l. Note that the right hand side is a quadratic function of the coordinates of p. Thus, each term under the summation on the right hand side of Equation (2) is a quadratic function of the coordinates of the point

$$p = [\mathbf{I} + A(a,b,c)](q_l + \sum_{k=1}^{N} \alpha_k f_l^k) + T.$$

The point p depends on the variables of F_0 in a quadratic manner (evaluating the formula for p yields quadratic terms of the form $a\alpha_k$, $b\alpha_k$ and $c\alpha_k$). We conclude that F_0 is a degree-4 polynomial of $(N+6)$ variables. In particular, this highly simplifies, speeds up and increases the accuracy of the nonlinear conjugate gradient method for minimization of F_0. Recall that the conjugate gradient method works by performing a series of line searches, that find minima of the objective function along lines. Because the function F_0 is a degree-4 polynomial, line searches reduce to minimization of degree-4 polynomials in one variable, or solving cubic equations. In particular, this means that the line searches can easily be made *global* rather than local, which makes the method less likely to get trapped in a local minimum.

3.4 Algorithm

The algorithm proceeds as follows. We initialize the current surface S to be the average shape transformed to the user-specified initial pose (Section 2). Thus, if the pose update defined by the user is (R_0, T_0) then S is set to $R_0[M] + T_0$. We also apply R_0 to all component vectors of all eigenshapes to make them consistent with the current pose.

In a loop that is iterated until convergence, we compute the local approximation F_0 of the objective function as described in Section 3.3 and minimize it using the nonlinear conjugate gradient method. Let $(a_*, b_*, c_*, T_*, \bar{\alpha}_*)$ be the minimum, with $\bar{\alpha}_* = (\alpha_{*,i})_{k=1}^N$. We substitute S with $\exp(A(a_*, b_*, c_*))[S(\bar{\alpha}_*)] + T_*$, where $S(\bar{\alpha})$ is obtained from S by translating the i-th vertex by $\sum_{k=1}^N \alpha_k e_i^k$ (cf definition of $M(\bar{\alpha})$ in Section 2.1). We also apply the

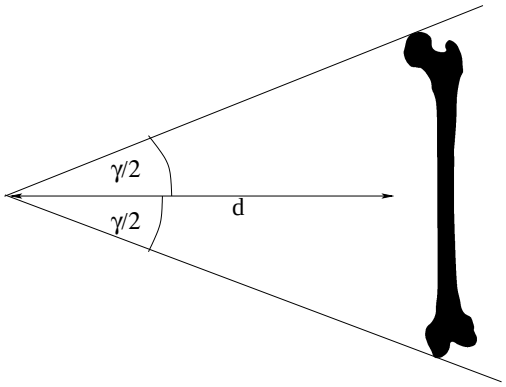

Figure 2. Estimate of the femur length assuming it is placed so that it is parallel to the projection plane; In order to obtain the scaling factor s in Section 3.5.1, we use $2d\tan(\gamma/2)$ as an approximation of the femur's length, where γ is the angular diameter of the femur.

rotation matrix $\exp(A(a_*, b_*, c_*))$ to all component vectors of all eigenshapes to make them consistent with the current pose.

The exponential map is a natural way to transform an antisymmetric matrix into a rotation matrix. This is because the derivative of the exponential map at the identity matrix is the identity: $\exp(A)$ is a rotation matrix very close to $\mathrm{Id} + A$ if the norm of A is small enough (the distance between the two is of the order of square of the norm of A).

3.5 Heuristics to improve convergence

The plain version of the algorithm described in Section 3.4 occasionally fails to converge to the global maximum. For the femur shapes, the most frequent reason for failure that we observed in our experiments was large length difference between the average shape and the output shape. The algorithm is made robust with two simple heuristics described below.

3.5.1 Approximate pre-scaling

Before executing the algorithm of Section 3.4, we apply an approximate scaling to the initial shape by changing the first shape parameter. The goal is to make the size of the starting surface closer to the size of the target surface. In most parametric shape models of the type used in this work (Section 2.1) the first eigenshape roughly controls the size of the shape (e.g. length of the femur). Let M_{α_1} be the shape corresponding to shape parameter vector $(\alpha_1, 0, 0, \ldots, 0)$. For the purpose of this discussion, we shall identify a shape with the $3n$-dimensional vector obtained by concatenating the coordinates of all vertices. In order to approximately scale the mesh M_{α_1} by a factor of s, we alter its shape parameter to α_1' such that the following inequality holds

$$M_{\alpha_1} \cdot M_{\alpha_1'} = sM_{\alpha_1} \cdot M_{\alpha_1}$$

where $\cdot$ denotes the $3n$-dimensional dot product. This equation states that the perpendicular projection of $M_{\alpha_1'}$ onto M_{α_1} (in the $3n$-dimensional space) is equal to sM_{α_1}. Since $M_\alpha = M + \alpha E_1$ and M and E_1 are known, the equation can easily be solved for α_1' leading to the following formula

$$\alpha_1' = \alpha_1 + (s-1)\frac{(M + \alpha_1 E_1)^2}{E_1 \cdot M + \alpha_1 E_1^2}.$$

In order to make use of it, we need to have an estimate of the desired scaling factor s. To obtain s, we perform a few iterations (3 were enough for examples discussed in Section 4) of the algorithm described in Section 3.4 to put the current surface S into roughly correct pose. These iterations involve only the pose parameters. Then, we compute the maximum angle γ between two rays passing through the approximate e-silhouette of S where e is one of the viewpoints. Let γ' be the maximum angle between rays originating from the viewpoint and passing

Figure 3. Four of the X-ray pairs used for evaluation.

through the silhouette points in the input image taken from viewpoint e. We use $s = \frac{\tan(\gamma'/2)}{\tan(\gamma/2)}$. This scaling factor is based on an estimate of femur's length assuming its perpendicular orientation to the projection plane and that the area near the middle of the femur projects close to the center of the image. If the femur's distance to the projection plane is assumed to be fixed, its length is proportional to the tangent of the half of its angular diameter (Figure 2). We consider γ and γ' estimates of the angular diameter of the current shape and the target shape (respectively). The user-selected silhouette points are used to estimate γ'. Note that this means that the user needs to select a pair of silhouette samples that are close to maximum possible distance between two silhouette points in the input image away.

The approximate scaling procedure is iterated until $|s - 1| < 0.001$ (it typically takes about $2 - 3$ iterations to make this condition true for a typical input).

Clearly, one can compute the estimate of s using a few (or even all) available viewpoints and average the results. However, if a particular view is to be used, the angle γ' must be close to the maximum angle defined by a pair of silhouette samples for that view. Thus, the user needs to select two silhouette samples that are roughly maximum possible distance away in the corresponding input image.

3.5.2 Gradual introduction of degrees of freedom

The second heuristic is to introduce the degrees of freedom associated with the shape parameters gradually. The impact of this heuristic is much smaller than that of the previously described one but we found a few synthetic examples in which it helps. First, a small number of iterations are done with just 6 degrees of freedom corresponding to pose update related variables. The following few iterations use the pose variables and only one shape parameter. We introduce a new shape parameter every fixed number of iterations (we found that adding a degree of freedom every 3 iterations works well for femur model examples).

4. EXPERIMENTAL RESULTS

In order to validate our algorithm, we used 23 simulated X-ray pairs obtained from CT scans (four of the pairs we used are shown in Figure 3). For each scan, the femur was segmented manually and the resulting 3D model was used as the ground truth for evaluating the result. The femur shapes computed using our procedure were aligned with the ground truth shape using the ICP method[10] and the mean, mean square and maximum errors between the two were measured. The average errors over all 23 cases as a function of the number of shape degrees of freedom used (i.e. N) are shown in Figure 4. For $N = 16$, the average mean, mean square and maximum errors were 0.74mm, 0.95mm and 4.52mm (respectively).

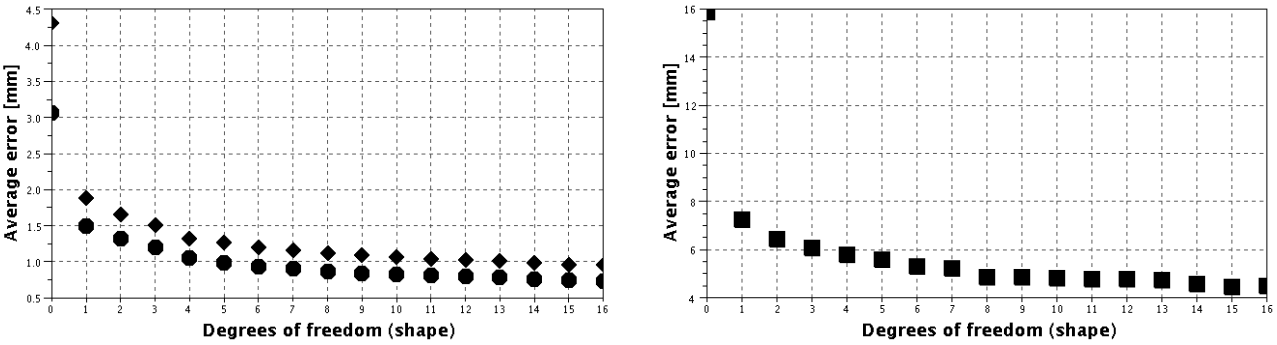

Figure 4. Average error of our algorithm for simulated X-ray pairs. Left: mean and mean square errors (circles and diamonds, respectively); Right: maximum error.

For each input dataset, manual selection of the silhouette points and specification of the initial alignment took a few minutes of user's time (almost all of which was spent on specifying the silhouette samples - 57 per X-ray image on average). The running time depends on the number of degrees of freedom and on a particular example and is around 5 minutes (single thread on a 2.4 GHz Core 2 processor) for $N = 16$. Moving the silhouette computation (Section 3.1) to the GPU could save about 60 percent of the running time.

A shortcoming of our method is that the objective function F defined by Equation (1) is discontinuous. This is because the silhouette point set is not a continuous function of the pose (this is true about apparent silhouettes too). This causes the algorithm to converge not to a well-defined point but rather to a more complex limit trajectory, which usually appears to be periodic. In some cases, the limit trajectory depends on the initial condition. The trajectories we observed are very small in diameter. We found estimates of the diameter by measuring distances between vertex vectors of the shapes corresponding to points in the limit trajectories for the same input dataset obtained using 3 runs with different starting pose. In all runs, we used 16 degrees of freedom for the shape ($N = 16$). The average diameter (in the maximum, mean and mean square sense, respectively) of the limit trajectories over all 23 input dataset was 0.42mm, 0.06mm and 0.08mm. The largest diameters for the 3 norms were observed for the same dataset and were 2.06mm, 0.33mm and 0.41mm (respectively). Curiously, the algorithm's performance for this particular dataset (i.e. the distance between the output and the ground truth) was slightly better than average.

Making the objective function more regular by making silhouettes depend smoothly on the pose could improve the convergence of the algorithm and may be an interesting topic for future work.

REFERENCES

[1] Kurazume, R., Nakamura, K., Okada, T., Sato, Y., Sugano, N., Koyama, T., Iwashita, Y., and Hasegawa, T., "3d reconstruction of a femoral shape using a parametric model and two 2d fluoroscopic images," in [*IEEE International Conference on Robotics and Automation'2007*], 3002–3008 (2007).

[2] Iwashita, Y., Kurazume, R., Nakamura, K., Okada, T., Sato, Y., Sugano, N., Koyama, T., and Hasegawa, T., "Patient-specific femoral shape estimation using a parametric model and two 2d fluoroscopic images," in [*ACCV'07 Workshop on Multi-dimensional and Multi-view Image Processing*], 59–65 (2007).

[3] Fleute, M. and Lavallée, S., "Nonrigid 3-d/2-d registration of images using statistical models," in [*MICCAI'99, LNCS 1679*], 138–147 (1999).

[4] Zheng, G., "Reconstruction of patient-specific 3d bone model from biplanar x-ray images and point distribution models," in [*Proc. IEEE Conf. on Image Processing'2006*], 1197–1200 (2006).

[5] Mahfouz, M., Emam, A. F., Dakhakhni, H. E., Tadross, R., and Komistek, R. D., "Three-dimensional bone creation and landmarking using two still x-rays," in [*Proc. Annual Meeting of American Academy of Orthopaedic Surgeons'2008*], (2008).

[6] Benameur, S., Mignotte, M., Parent, S., Labelle, H., Skalli, W., and Guise, J. A. D., "A hierarchical statistical modeling approach for the unsupervised 3d reconstruction of the scoliotic spine," in [*Proc. of the International Conference on Image Processing'2004*], 561–564 (2004).

[7] Benameur, S., Mignotte, M., Parent, S., Labelle, H., Skalli, W., and Guise, J. D., "3d biplanar reconstruction of scoliotic vertebrae using statistical models," in [*20th IEEE International Conference on Computer Vision and Pattern Recognition, CVPR'01*], 577–582 (December 2001).

[8] Dong, X., Ballester, M. A. G., and Zheng, G., "Automatic extraction of femur contours from calibrated x-ray images using statistical information," *Journal of Multimedia* **2**(5), 46–54 (2007).

[9] Merkl, B. C. and Mahfouz, M. R., "Unsupervised three-dimensional segmentation of medical images using an anatomical bone atlas," in [*IFMBE Proceedings ICBMEC 2005 "The 12th International Conference on Biomedical Engineering"*], **12** (2005).

[10] Besl, P. J. and McKay, N. D., "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**(2), 239–256 (1992).