Implementing the Automatic Generation of 3D Statistical Shape Models with ITK

Release 1.00

Tobias Heimann[1], Ipek Oguz[2], Ivo Wolf[1], Martin Styner[2] and Hans-Peter Meinzer[1]

July 19, 2006

[1]Div. Medical and Biological Informatics, German Cancer Research Center,
t.heimann@dkfz-heidelberg.de
[2]University of North Carolina at Chapel Hill, ipek@cs.unc.edu

Abstract

We describe a method to automatically find the point correspondences between a collection of polygonal genus 0 meshes. This correspondence data is the key for building three-dimensional statistical shape models, which have a variety of applications in medical imaging. Our method is based on minimizing a cost function that describes the goodness of correspondence. Apart from a cost function derived from the description length (MDL) of the model[1], we also employ a cost function working with arbitrary local features. As an example, we present results using surface curvature measurements. The entire method is implemented in a collection of versatile and easy-to-use ITK classes. In addition to an overview of the implementation, we present results for a synthetic and a real-world dataset processed with the software.

Contents

[1]Establishing correspondences by minimizing the MDL is a method patented by the University of Manchester (`www.manchester.ac.uk/`).

1 Introduction

Since its introduction by Cootes et al. [1], Active Shape Models (ASMs) have become a popular segmentation method in medical imaging. The main drawback of the approach is the point correspondence problem in the model construction phase: On every training sample for the ASM, landmarks have to be placed in a consistent manner. While it is tedious and time-consuming work to label the training sets manually, this approach is a feasable solution for 2D models with a limited number of landmarks. In the 3D domain however, manual labelling is highly impractical: Not only is the required number of landmarks higher than in the 2D case, depending on the sample shapes it becomes increasingly difficult to identify and pinpoint corresponding points, even for experts. Several automated methods to find the correspondences in 3D have been presented so far. In a recent comparison [6], the approach by Davies et al. to minimize a cost function based on the minimum description length of the resulting statistical shape model [2] delivered the most promising results. The correspondence optimization of this method was later improved by Heimann et al. in [4], using a more efficient reparameterization scheme and a gradient descent optimization. In this paper, we present the implementation of this correspondence optimization for the Insight Toolkit (ITK) and explore the possibilities of alternative cost functions.

2 Material and Methods

2.1 Statistical shape models

Statistical shape models capture shape information from a set of labeled training data. A popular method to describe these shapes are point distribution models [1], where each training shape is specified by a set of n landmarks on the surface. Applying principal component analysis to the covariance matrix of all landmarks delivers the principal modes of variation $\mathbf{p}_m$ in the training data and the corresponding eigenvalues λ_m. Restricting the model to the first c modes, all valid shapes can be approximated by the mean shape $\bar{\mathbf{x}}$ and a linear combination of displacement vectors:

$$\mathbf{x} = \bar{\mathbf{x}} + \sum_{m=1}^{c} y_m \mathbf{p}_m \tag{1}$$

In general, c is chosen so that the model explains a certain amount of the total variance, usually between 90% and 99%. In order to describe the modeled shape and its variations correctly, landmarks on all training samples have to be located at corresponding positions.

2.2 Correspondence by optimization

A prerequisite for statistical shape models is a set of landmark points located at corresponding positions on all training shapes. In the MDL approach introduced by Davies et al. [2], these points are created by minimizing a cost function F which is based on the minimum description length of the generated model. In this work, we use a simplified version of the MDL as proposed by Thodberg [7], where F is defined as:

$$F = \sum_m \mathcal{L}_m \quad \text{with} \quad \mathcal{L}_m = \begin{cases} 1 + \log(\lambda_m/\lambda_{\text{cut}}) & \text{for } \lambda_m \geq \lambda_{\text{cut}} \\ \lambda_m/\lambda_{\text{cut}} & \text{for } \lambda_m < \lambda_{\text{cut}} \end{cases} \tag{2}$$

λ_{cut} is the threshold that determines which modes are considered as systematic variations and which ones as noise.

2.3 Alternative cost functions

Generally, the λ_m in Eq. 2 correspond to the eigenvalues of the landmark positions, i.e. their spatial locations. However, it is also possible to use any other local feature and minimze a cost function F based on the eigenvalues of these features. A good example for this are local curvature metrics as the ones presented in Koenderink [5], namely, the shape index S and the curvedness C. C and S can be computed as functions of the two principal curvatures of the surface. They basically are equivalent to a polar representation of the principal curvatures κ_1 and κ_2.

$$C = \frac{2}{\pi} ln \sqrt{(\kappa_1^2 + \kappa_2^2)/2} \tag{3}$$

$$S = -\frac{2}{\pi} arctan \frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2} \tag{4}$$

C and S improve the curvature measurement by decoupling the size and shape aspects of the curvature. C describes how curved an object is, and is closely related to the size. S, on the other hand, is indicating the shape of the surface in terms of concaveness and convexness. This pair of metrics is very suitable for measuring correspondence of two surfaces, since they provide a means of measuring shape in a very intuitive way.

It should be noted that, even though we only present results using C and S as metrics in this work, our implementation provides complete flexibility in the choice of features to be used. This is achieved by letting the user provide the number of features per point and feature values, without any constraints. The feature values should be computed offline and stored in a feature file for each object in the population.

2.4 Mesh parameterization

To define an initial set of correspondences and a means of manipulating them efficiently, we need a convenient parameter domain for our training shapes. In order to minimize complexity for the parameterization of 3D shapes, we will restrict the discussion to closed two-manifolds of genus 0 (i.e. surfaces without holes and self-intersections). Objects of this class are topologically equivalent to a sphere and most shapes encountered in medical imaging are of this type (e.g. liver, kidneys and lungs). The task is to find a one-to-one mapping which assigns every point on the surface of the mesh a unique position on the unit sphere, described by two parameters longitude $\theta \in [0..2\pi]$ and latitude $\phi \in [0..\pi]$.

Along with this article, we provide source code to create a conformal parameterization for a genus 0 input mesh based on the method described by Gu et al. in [3]. Alternatively, the classes for handling spherical harmonics from the UNC Neurolib (`www.ia.unc.edu/dev/`) can be used for the same purpose, or any other method to create a spherical parameterization.

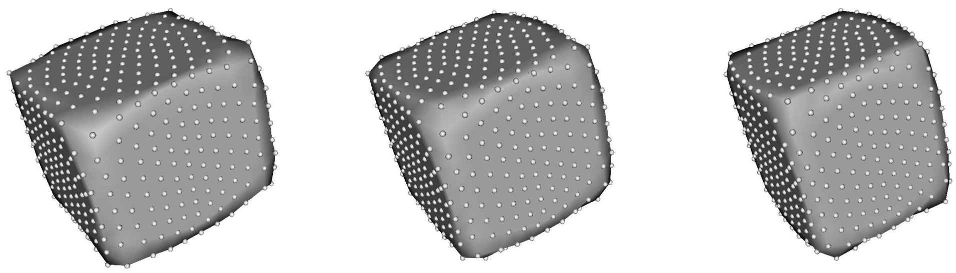

Figure 1: The first eigenmode of the synthetic cuboid dataset after landmark optimization (ranging from -2σ to $+2\sigma$).

2.5 Optimizing landmark correspondences

With an initial conformal parameterization ω_i for each training sample i, we can acquire the necessary landmarks by mapping a set of spherical coordinates to each shape. To optimize the point correspondences with respect to our cost function, two possibilities are available: We can either change the individual ω_i and maintain a fixed set of global landmarks or modify individual landmark sets Ψ_i.

In this work, we opted for the first alternative, which has the advantage that the correspondence is valid for any set of points placed on the unit sphere. The modification is performed by warping the parameterizations inside strictly local regions, modeled by a Gaussian envelope function. Direction and amplitude of the warp are determined by the gradients of the cost function F. For a detailed description of the approach, we refer the reader to [4].

3 Experiments and Results

3.1 Spatial location based optimization

An in-depth evaluation of the correspondence optimization using a cost function based on the 3D spatial locations of vertices for establishing correspondence was conducted in [4]. As an example, we supply the reader with one of the synthetic datasets from that paper, a collection of 20 cuboids with varying aspect ratio. The correspondence optimization of these meshes converges in less than 1000 iterations, which takes approximately 15 minutes on a modern desktop PC. The variation along the first mode of the resulting shape model is displayed in Figure 1 and shows the expected bahaviour.

3.2 Local curvature based optimization

The extended version of the presented method can use any number of local features for establishing correspondence. The feature values at each location are provided in input files. Here, we present results of an experiment where we used the previously presented local curvature metrics C and S as our features. Figures 2 and 3 show the results of this optimization, visualized such that corresponding locations across the population are colored in the same way. Figure 2 shows the ϕ value correspondence and Figure 3 shows the θ value correspondence, where ϕ and θ are the usual spherical coordinates.

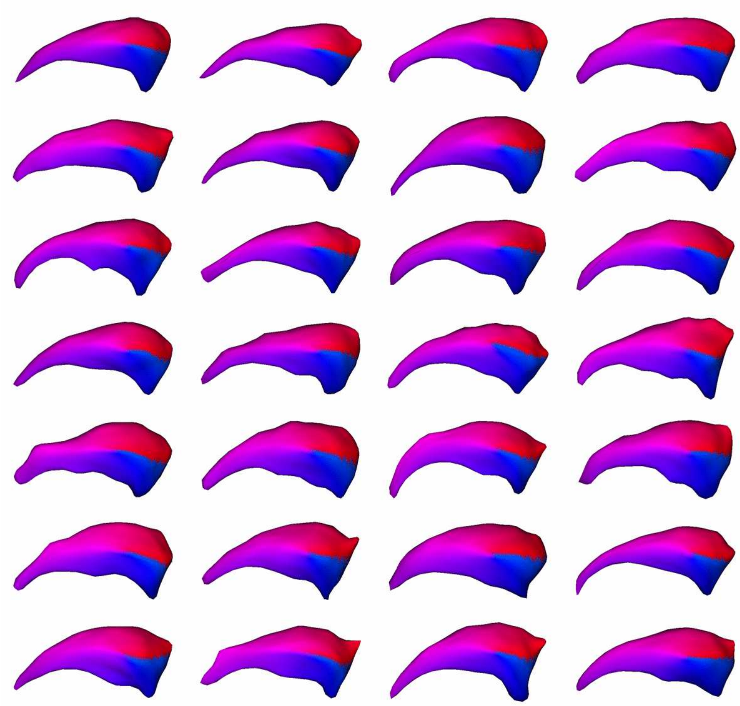

Figure 2: The ϕ values of our population after curvature based correspondence is run. Similar colors across the objects show corresponding ϕ values on each object.

Note that even though we present results using only C and S metrics, our tool allows the usage of any desired optimization metric, or any combination thereof. This is achieved by making the correspondence based on input read through a file, and not internal computations. This provides great flexibility and enables exploring various shape metrics and inspecting the quality of the correspondence they imply, without even modifying the code.

4 Implementation

Although the proposed algorithm is easier to implement than the original MDL optimization, doing so still is a challenging undertaking. One of the earliest problems encountered was that ITK, while offering a large variety of 2D and 3D image filters, provides only very limited mesh support. Most of the functionality necessary for parameterizing meshes — beginning with efficient access to vertices, edges and faces — had to be implemented from scratch in diverse subclasses of `itk::Mesh`. Consequently, there was a lot of work to do apart from designing the core components of the algorithm. An overview of how these classes act together in the algorithm for automatic model building is given in Fig. 4.

An example application to find corresponding landmarks over a set of training meshes is provided along with these classes as a ready-to-use tool. The only parameters to this tool are an input list file, a landmark file, and a model radius. The input list file is a simple text file including the paths for all the input mesh files representing the input objects in the population. For each object, there should be separate files containing the vertices, the faces, the parametrization, and the features (if the features rather than the spatial locations are to be used for the optimization). Each of these files should have the same name, but different extensions: *.pts* for the vertices, $.fce$ for the faces, *.par* for the parametrization, and *.txt* for the features. The landmark

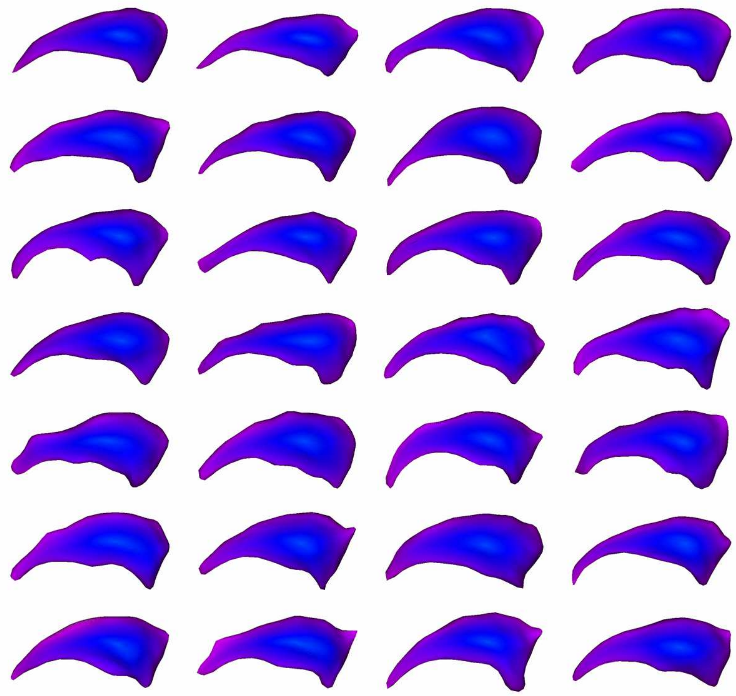

Figure 3: The θ values of our population after curvature based correspondence is run. Similar colors across the objects show corresponding θ values on each object.

file is a separate mesh file that holds the coordinates of the landmark prototypes on the unit sphere. One of the simplest methods to create a landmark mesh is to subdivide one of the platonic solids, e.g. the icosaeder. Two landmark meshes, consisting of 642 and 2562 points, respectively, are provided with the example data. The last parameter, the model radius, is used to determine the variance threshold for the cost function. Note that the radius has to be given in number of voxels to allow a valid interpretation of the noise in the training data. To run the correspondence optimization with the example cuboid data, the command line arguments should be `cuboids.txt landmarks642.pts 100`.

A detailed overview of how the example application works is useful to demonstrate how the various classes work together. Initially, an instance of the `StatisticalShapeModel3DCalculator` class is created and provided with a cost function, which is an instance of the `SimplifiedMDLCostFunction` class. Note that one can either choose the `StatisticalShapeModel3DCalculator` class itself and use spatial locations as a metric, or use the subclass `StatisticalShapeModel3DCalculatorWithFeatures` and use arbitrary local features. Next, the input meshes and the landmark mesh are loaded. If a parametrization file is not already provided along with the input meshes, a suitable initial parametrization is computed, either via conformal spherical parametrization or via spherical harmonics basis functions (the first method is used in the provided example application). Any other method that generates a spherical parametrization can be used as well. The resulting instances of the `SphericalParametrizedTriangleMesh` class are then provided to the `StatisticalShapeModel3DCalculator`. After the `StatisticalShapeModel3DCalculator` is updated, all that remains to do is to output the final versions of the meshes. Additionally, the final (corresponding) parameterization can be queried for all input samples. On these parameterizations, points with the same (ϕ,θ) values will be corresponding.

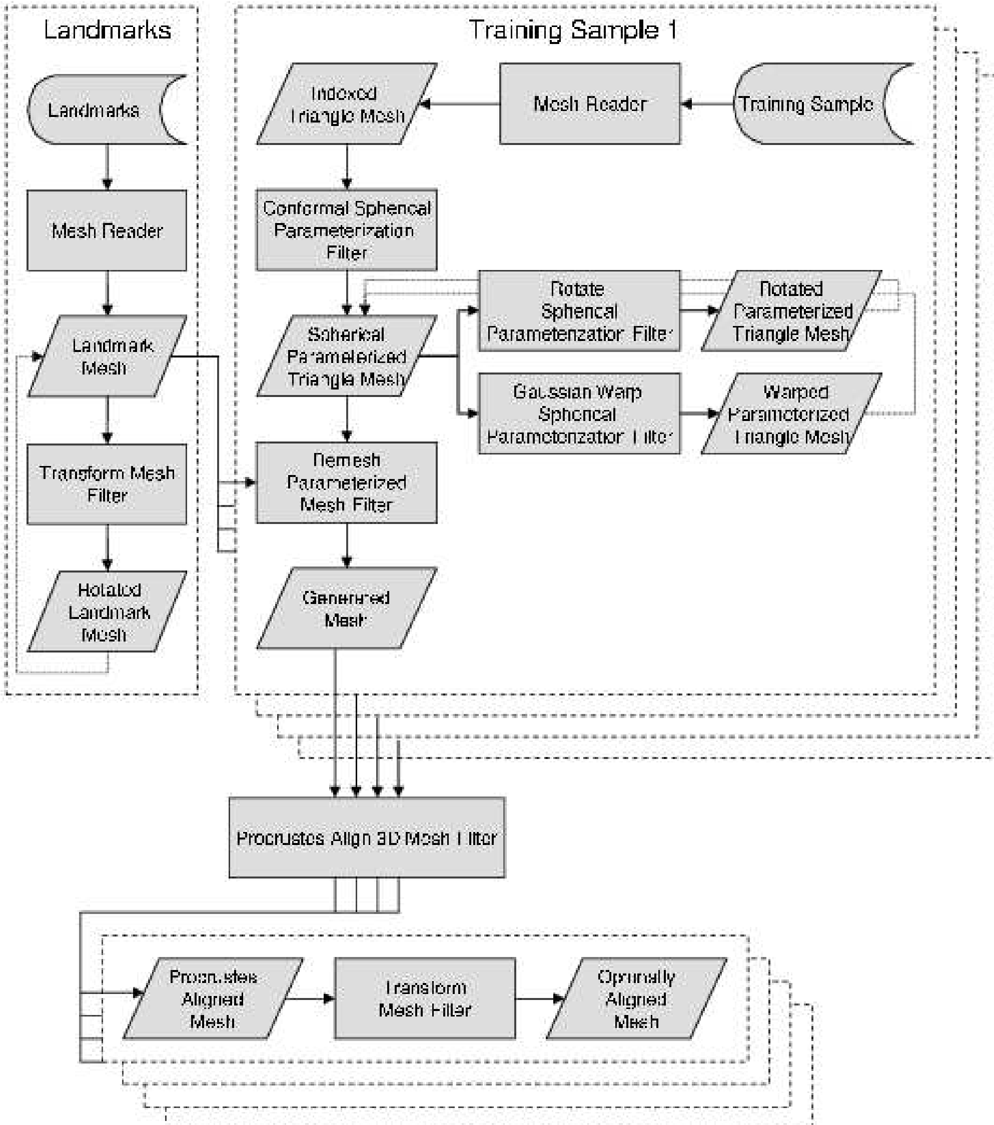

Figure 4: An overview of the pipeline involved in the algorithm for automatic model building. Each training sample is read from disk and parameterized conformally. Using a landmark mesh which is also read from disk, shapes with the same number of vertices are created. These are aligned by a Generalized Procrustes matching and scaled to tangent size. In each optimization step, all parameterizations are modified by the Gaussian warp filter and the results written back to the original data (dotted line). Subsequently, landmarks and parameterizations are rotated with the same transform (i.e. landmark positions on the generated meshes do not change), again overwriting the original values.

5 Conclusions

We have presented a method to automatically find the correspondences on a set of genus 0 meshes, which is the basis for building a 3D statistical shape model. For all necessary steps, from file IO over the creation of a parameterization up to the actual optimization of correspondences, we provide flexible, high-performance and easy-to-use ITK classes. In addition to the standard method of minimizing a cost function based on the spatial location of landmarks, we offer the possibility to use arbitrary features, e.g. curvature metrics. This collection of classes substantially eases the creation of 3D statistical shape models and should further propagate their use in medical image analysis.

6 Acknowledgements

We are thankful to Guido Gerig and Steve Pizer (UNC Chapel Hill) for highly valuable discussions in regard to shape correspondence and surface curvature. The caudate segmentations used in this manuscript were provided by Jim Levitt and Martha Shenton (Dept. Psychiatry, VA Boston Healthcare System, Brockton Division, Harvard, Boston). Ipek Oguz and Martin Styner are supported by the NIH Roadmap for Medical Research Grant U54 EB005149-01, UNC Neurodevelopmental Disorders Research Center HD 03110, and the NIH NIBIB grant P01 EB002779.

References

[1] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models – their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995. 1, 2.1

[2] Rhodri H. Davies, Carole J. Twining, Timothy F. Cootes, John C. Waterton, and Christopher J. Taylor. 3D statistical shape models using direct optimisation of description length. In *Proc. European Conference on Computer Vision, Part III*, pages 3–20. Springer, 2002. 1, 2.2

[3] Xianfeng Gu, Yalin Wang, Tony F. Chan, Paul M. Thompson, and Shing-Tung Yau. Genus zero surface conformal mapping and its application to brain surface mapping. In *Proc. IPMI*, pages 172–184, 2003. 2.4

[4] Tobias Heimann, Ivo Wolf, Tomos G. Williams, and Hans-Peter Meinzer. 3D active shape models using gradient descent optimization of description length. In *Proc. IPMI*, pages 566–577. Springer, 2005. 1, 2.5, 3.1

[5] J. J. Koenderink. *Solid Shape*. MIT Press, 1990. 2.3

[6] Martin Styner, Kumar T. Rajamani, Lutz-Peter Nolte, Gabriel Zsemlye, Gábor Székely, Christopher J. Taylor, and Rhodri H. Davies. Evaluation of 3D correspondence methods for model building. In *Proc. IPMI*, pages 63–75, 2003. 1

[7] Hans Henrik Thodberg. Minimum description length shape and appearance models. In *Proc. IPMI*, pages 51–62, 2003. 2.2