Minimum Description Length
Shape and Appearance Models

Hans Henrik Thodberg

Informatics & Mathematical Modelling,
Technical University of Denmark,
2800 Lyngby, Denmark
e-mail: hht@imm.dtu.dk, web: http://www.imm.dtu.dk/~hht

IMM TECHNICAL REPORT 2003-01
Submitted Feb 14, 2003 to *Information Processing in Medical Imaging 2003*

Keywords: Shape Modelling, Minimum Description Length, Appearance Models, Point Correspondence Problem, Unsupervised Vision, Image Segmentation.

Abstract. The Minimum Description Length (MDL) approach to shape modelling is reviewed. It solves the point correspondence problem of selecting points on shapes defined as curves so that the points correspond across a data set. An efficient numerical implementation is presented and made available as open source Matlab code. The problems with the early MDL approaches are discussed. Finally the MDL approach is extended to an MDL Appearance Model, which is proposed as a means to perform unsupervised image segmentation.

1. Introduction

In order to construct an Active Shape or Active Appearance Model [1,2] one needs a number of training examples, where the true location of the underlying shape is known. From thereon these models are automatically generated. This paper addresses the problem of constructing these training examples automatically.

The problem is divided into two: The first is to define the shapes in terms of contours. The second is to define marks on these contours. The marks should be defined so that marks on different examples are located at *corresponding* locations, hence the second problem is sometimes denoted *the point correspondence problem*, and it has been the subject of a series of papers by Taylor and collaborators [3,4,5] founded on MDL. This paper reviews the development, describes a simple and efficient implementation and demonstrates it on open and closed contours. The Matlab code and the examples are published to facilitate the dissemination of this technique in medical imaging and other applied fields.

Finally it is proposed to extend the MDL approach to solve also the first problem, defining the shape contours in the first place through unsupervised learning with the *MDL Appearance Model*.

2. History of Minimal Shape Modelling

The development of the MDL approach to the point correspondence problem is marked by three important papers.

Kotcheff and Taylor – 1998 [3]. In this paper the problem is formulated as finding a reparametrisations of each contour. The cost function Cost = $\Sigma \log \lambda_m$ is the sum over eigenvalues λ_m larger than some cut-off λ_{cut}, and the optimisation technique is a genetic algorithm. The contribution of the paper is conceptual while the algorithm performance is not spectacular.

Davies, Cootes and Taylor – 2001 [4]. This paper uses MDL as basis for the cost function. The paper computes the cost of transmitting the PCA model and the PCA-coded data, and the optimal trade-off between the precisions of the various components is derived. This leads to a description length expression, which allows a determination of the optimal number of principal components independent of the precision. A heuristic optimisation technique based on a coarse-fine strategy is introduced. The performance is impressive on several 2D data sets, and the computation time is practical – of the order of four hours in Matlab. In addition generalisation to 3D is possible. This paper attracted a lot of attention due to its combination of a principled approach and wide applicability. The work was presented at many conferences and received several awards.

Davies, Twining, Cootes, Waterton and Taylor – 2002 [5]. This is the first journal article on MDL shapes and contains several changes to the formalism. Gone is the full PCA MDL model, and now a slower optimisation based on genetic algorithms is used to avoid local minima. A master example is selected to have fixed parameterisation to avoid that the shapes collapse onto a small part of the contour.

Questions. Several questions come to mind in this development

- Why was the MDL approach of 2001 abandoned, and is there something wrong it?
- Can MDL be used to determine the number of principal components in shape modelling?
- Is it possible to run MDL with something faster than genetic algorithms and still avoid local minima?
- How does one prevent the reparametrisations to diverge (run away)? Is one fixed master example sufficient?
- How does the formalism apply to open curves?
- What is the best way to begin using MDL on 2D problems?

Outline

These questions are answered in this paper, which is organised as follows:

- Section 3 describes a simple and efficient version of the MDL approach to 2D shapes and demonstrates it on artificial data.
- Section 4 analyses the theoretical development of the MDL shape approach.
- Section 5 applies the method to medical shape data.
- Section 6 generalises the method to MDL Appearance Models.
- Section 7 contains the conclusions.

Matlab source code and test data are available on www.imm.dtu.dk/~hht.

3. An Efficient MDL Shape Algorithm

This section describes the efficient MDL shape algorithm used for the simulations in this paper. The algorithm applies to *a set of shapes* defined as *curves* in 2D space. Shape sets are classified into three kinds: *Closed curves*, *open curves with fixed end-points* and *open curves with free end-points.* Fixed end-points means that the shape has its end-points fixed at the curve end-points, while free end-points means that the "true" shape is an unknown subset of the open curve, i.e. the determination of the shape end-points is part of the task.

The curves are represented as a polyline i.e. an ordered list of points. The arc length along the curve is normalised to run from 0 to 1.

We are now seeking a set of 2^L+1 marks on each curve to represent the shape. They are called marks to indicate that they carry a meaning (like landmarks, post-marks, hallmarks etc). For closed shapes, the start- and end-points (number 0 and 2^L) are identical. The mark locations are specified in a hierarchical manner (as described by Davies 2001), on L levels. For closed curves with 65 marks, we specify on the first level the coordinates of mark 0 and 32 by their absolute arc length position. On the second level, mark 16 and 48 are specified by parameters between 0 and 1. For example mark 16 can be anywhere on the curve between mark 0 and 32, corresponding to the extremes 0 and 1. On the third level the marks 8, 24, 40 and 56 are specified in between already fixed marks. This is continued until level 6 so that all marks are specified.

For open *fixed*-end curves, level 1 places only mark 32, while for open *free*-end curves there are three marks on level 1, namely 0, 32 and 64.

The end-marks are defined by two positive parameters describing the distance of the end-marks from the ends of the curve.

The initial shape can be defined by marks placed evenly in arc length by setting all parameters to $a = 0.5$ (except for the end-points). Alternatively a priori knowledge of a good starting guess can be used. Closed curves should be roughly aligned initially.

Statistical shape analysis is now performed in the usual way. The number of marks is $N = 2^L$ for closed curves and $N = 2^L+1$ for open curves (free as well as fixed). First the shapes are centred and aligned to the mean shape normalised to one, i.e. the rms radius of the mean is $1/\sqrt{N}$. The mean is determined using Kent's method [6] by representing mark positions as complex numbers and diagonalising the hermitian N-by-N covariance matrix of the set; the mean is the leading eigenvector. If the number of shapes s is smaller than N, the "dual" s-by-s matrix is diagonalised instead.

The covariance matrix of the aligned shapes (normalised with the number of shapes, but not with the number of points) is then formed and principal component analysis is performed yielding the eigenvalue spectrum.

The optimisation does not need to be done on all marks, but only on marks up to a given level, typically we adjust level 1, 2, and 3. These active marks are called *nodes* because the curve reparametrisations evolve kinks at these marks. The optimisation adjusts the node parameters to optimise the correspondence of all the marks over the set of examples. The parameters of level 4, 5, 6 are frozen at 0.5 corresponding to even distribution in arc length to capture the shape variation between the nodes.

The objective function is derived from the MDL principle. The cost describes the information needed to transmit the PCA representation of the shapes, i.e. the principal components. For a mode m with large eigenvalue the cost is $\log(\lambda_m)$, while for smaller lambdas it should tend to a constant. We therefore introduce an important parameter λ_{cut} which separates these two regimes and use a cost expression from Davies 2002 in the low lambda region: $\log(\lambda_{\text{cut}}) + (\lambda_m / \lambda_{\text{cut}} - 1)$. (Davies' expression was simplified by approximating $(s+3) / (s-2)$ by 1). Adding the constant $1 - \log(\lambda_{\text{cut}})$ leads to our final choice for the total cost

$$\begin{aligned} &\text{Description Length} = \Sigma L_m \\ &L_m = 1 + \log(\lambda_m/\lambda_{\text{cut}}) \quad \text{for } \lambda_m \geq \lambda_{\text{cut}} \\ &L_m = \lambda_m/\lambda_{\text{cut}} \qquad\qquad\quad \text{for } \lambda_m < \lambda_{\text{cut}} \end{aligned} \qquad \textbf{(1)}$$

This cost has the attractive properties that it tends to zero when all eigenvalues tend to zero, and both L_m and $dL_m/d\lambda_m$ are continuous at the cut-off.

In plain words, when λ_m falls below λ_{cut}, the benefit of decreasing it further is no longer logarithm, but levels off and reaches a minimum, 1 unit below the transition point. A mode with eigenvalue λ_{cut} contributes on average a variance of λ_{cut}/N per mark, and since the rms radius of the aligned shapes is $1/\sqrt{N}$, the mode contributes a standard deviation per rms radius of $\sigma_{\text{cut}} = \sqrt{\lambda_{\text{cut}}}$. We specify λ_{cut} in terms of σ_{cut} and use $\sigma_{\text{cut}} = 0.003$ in all the simulations in this paper. This corresponds to a cut-off at 0.3 pixels for shapes with original rms radius 100 pixels.

A crucial requirement for the cost is that it be insensitive to N in the high-N limit: If N is doubled, the rms radius of the aligned shapes decreases by $\sqrt{2}$. This balances the doubling of all eigenvalues that would otherwise occur, with the result that the high end of the eigenvalue spectrum - and hence the cost - are unchanged.

The shape representation assigns the same weight to all marks in the Procrustes alignment and in the PCA; one could have a weight proportional with the spacing and with some prior, but that would complicate the algorithm. As a consequence, the centre of gravity and rms radius of a shape changes as the nodes shift. This gives rise to effects, which are often not desirable, but a consequence of the chosen simplicity.

One effect is that the marks can pile up in some areas and thereby avoid describing the rest and reach a small description length.

One way to avoid this run-away is to select a single shape as master example (as introduced by Davies 2002) for which the marks are not allowed to move. Its marks can be positioned at landmark position for instance by manual annotation by an expert, at conspicuous locations e.g. where the curvature is at a local maximum.

The iterative optimisation can then begin. The nodes, e.g. 8, are ordered according to ascending level. Each node is associated with a step length, initially set to 0.01. These 8 step lengths are automatically decreased by the algorithm. Now the parameters a_{node} for each node and each example are probed, one at a time according to the following pseudo-code, which runs over a number of passes, until the results has stabilised, typically 40 passes.

```
Loop over passes
  Loop over nodes
    Loop over 5 steps
      Loop over examples
        Loop over + and - step
           Probe a(node) = a(node) +- step of example
           Recompute marks of example
           Do Procrustes of set
           Do PCA of set
           Compute new MDL
           If new MDL is lower accept and break loop
           Undo a(node) change
        End of +- step loop
      End of example loop
      If <20% of a(node)'s changed, divide step(node) by 2
    End of step loop
  End of node loop
End of passes loop
```

The three lines in bold each account for approx 1/3 of the processing time. The step lengths are adaptive, so there are no parameters to tune.

For the master example a special code is used: The nodes of this example should not be moved, but if a node parameter is far out of correspondence with the rest, it is very slow for all the others to move individually to the lonely master. If Mohammed does not to come to mountain, then the mountain must come to Mohammed and therefore when the master example is encountered in the pseudo-code above, all the other examples are given a step simultaneously to allow for a collective move towards the master.

The convergence time increases with decreasing cut-off, so if a very low cut-off is desired, it can be effective to run the optimisation with larger cut-off initially, decreasing to the desired value at the end. This was not needed for the examples in this paper.

Example box-bump

As a first example, consider the 24 shapes in Figure 1 using $\sigma_{\text{cut}} = 0.003$, 40 passes, 64 marks and 8 nodes. The basic evaluation in this procedure consists of the reparametrisations and the diagonalisation of a complex 24-by-24 matrix and a real 24-by-24 real matrix. 140 evaluations are made per second on a 1.2 GHz PC in Matlab. The number of evaluations in 40 passes is 40 passes · 8 nodes · 5 steps · 24 examples · 2 signs = 80,000 evaluations, which takes approximately 10 minutes. Figure 2 shows the convergence of the node parameters, and Figure 3 illustrates the modes.

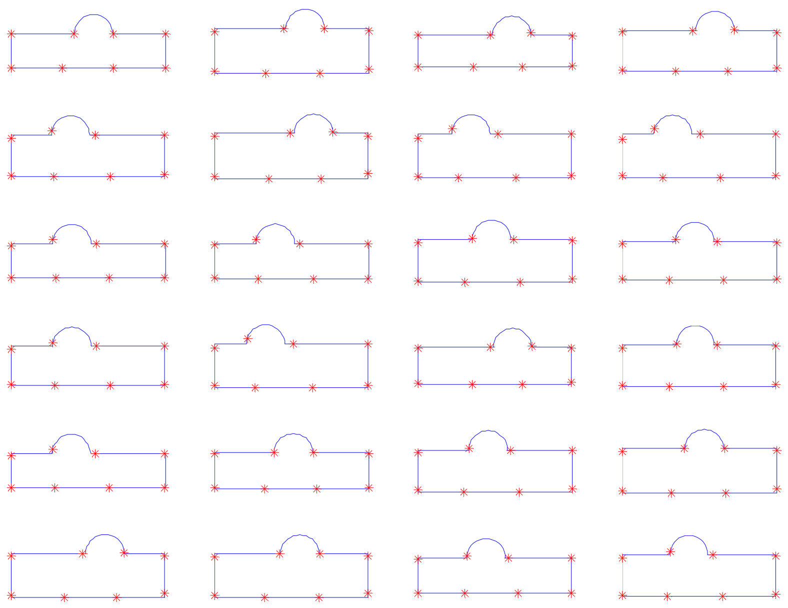

Fig. 1. The 24 ‘box-bump’ shapes created with a bump at a varying location and with varying aspect ratio of the box. Eight nodes are optimised, and the first example was annotated manually as shown, while the corresponding nodes were placed on the other examples by the MDL algorithm in 10 minutes.

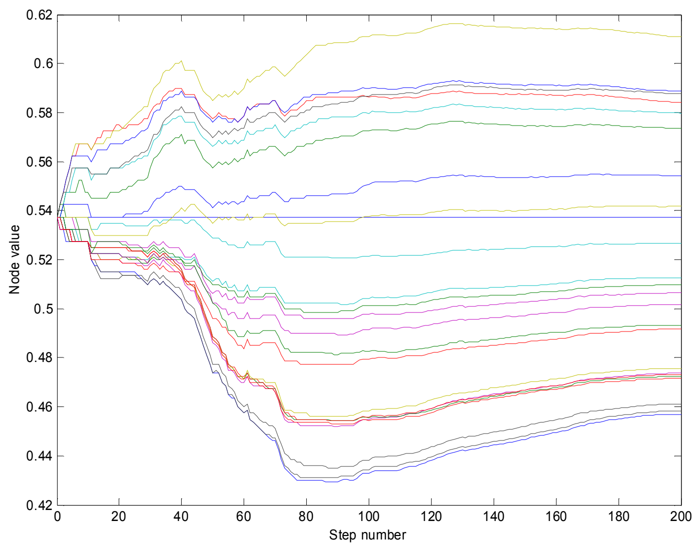

Fig. 2. The course of the optimisation of the parameter of the node to the right of the bump in the box-bump problem is shown for each example. The master example is kept fixed.

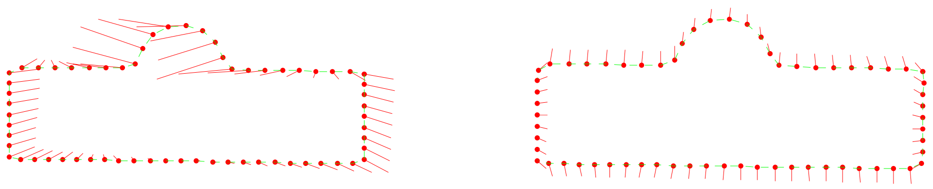

Fig. 3. The two first modes of the box-bump set after MDL. Shown is the mean shape in green with red marks. The whiskers emanating from the marks indicate three standard deviations of the principal components. The mean has indeed captured the shape of a small semicircular bump and the first component is close to a horizontal displacement of the bump. To keep the centre of gravity fixed, the movement of the bump is counterbalanced by an opposite movement of the box. The second principal component describes the aspect ratio.

Improved control of run-away

In some cases, a single fixed master example is not sufficient to keep the whole set in place. For example the free endpoints of open curves can drift systematically to one side or the other, neglecting the master. This is because the statistical weight of the majority can outweigh the single master and the gain of run-away exceeds the cost of a single outlier.

A remedy to this defection is to add a stabilising term to the MDL cost. Instead of fixing the node parameters of the master, one introduces a target a_i^{target} for the *average* parameter a_i^{average} for each node i by means of a quadratic cost:

$$\text{NodeCost} = \Sigma\,(a_i^{\text{average}} - a_i^{\text{target}})^2/T^2 \qquad \textbf{(2)}$$

where T is a chosen tolerance, so if the average drifts e.g. $T = 0.05$ away from the target, one unit is added to the cost.

The algorithm can in general be run in four modes: Mode 1 fixes the node parameters of one master example. Mode 2 fixes the averages of the node parameters as described above. Mode 1 maintains the pure learning idea of one labelled example among the unlabelled rest, while mode 2 has stability towards runaway and a symmetric treatment of all examples. Finally we introduce mode 3 and 4 where the dynamics is controlled by the node cost with "moving target" values, which are adjusted at the start of each pass. In mode 3, an annotated master example is provided, and the moving targets are adjusted such that the master case relaxes in agreement with the annotation. In mode 4, no annotated example is used. Instead the moving targets are adjusted such that the average node parameters relax on desired values – a neutral choice is 0.5 for all parameters, which leads the marks to be located on average evenly in arc length. All four modes are supported by the open source Matlab code.

Mode 3 is to be preferred over mode 1. It maintains the attractive learning paradigm of one labelled example among many unlabelled, but avoids the troublesome, unsymmetrical treatment of the master example of mode 1.

4. Theoretical Development of MDL Shape Models

Davies 2001 presents a fundamental MDL analysis of PCA models. The transmission of the shape data set is done by transmitting the mean value and t eigenvectors (the model) and for each data example transmitting the t principal component and the n residuals ($n = 2N$). The residuals stem from the use of a limited number t, plus an enhancement due to the finite precision used to transmit the model and the principal components. The enhancement factor is computed to $\alpha = ns \,/\, (n(s-1) - t\,(n-s))$.

The optimal number of principal components can be determined, i.e. MDL can be used to determine the proper model complexity for the data set at hand, and if the number of data examples increases, the optimal t increases. For a fixed number t, the MDL expression is of the form

$$\mathrm{DL} = \Sigma_{\lambda m \geq \lambda\mathrm{cut}} \log \lambda_m + K \log \left(\Sigma_{\lambda m < \lambda\mathrm{cut}}\, \lambda_m\right) \qquad \mathbf{(3)}$$

where K is a constant. This expression is similar to the Bayesian BIC expression in [7].

There are two problems with this MDL shape model.

Firstly, α was computed wrongly; the correct expression is $\alpha = ns \,/\, (ns - (n + tn + ts))$. This diverges when the dimension of the PCA coding (the mean value contributes n, the loadings vector tn and the scores ts) equals the dimension ns of the original data. Thus the total MDL cost can be computed only for $t < (ns - n)/(n + s)$ and not as in Davies 2001 all the way up to s (if $s<n$). It is easy to verify the corrected expression by reviewing the details in Davies 2001. The expression (3) is unaffected by the error in α.

The second and more serious problem is the inappropriateness of MDL for doing a complete PCA analysis of shapes. PCA is blind to the spatial arrangement of the inputs: Any permutation of the coordinate labels yields an identical PCA. A considerable amount of information is spent on transmitting the eigenvectors, but MDL ignores the correlations between eigenvector coordinates of nearby points. This means that the MDL analysis gets increasingly inadequate as n increases and the high-n limit is not consistent. For this reason it is strongly recommended to use the Davies 2002 version.

However, one should not through out the baby with the bathwater. With the corrected α, the MDL PCA is perfectly valid and could be a useful tool for data, which are not spatial or otherwise sampled in an ordered manner. For instance in analyses of demographic data, the method could be used to determine the number of components.

Davies 2002: When Twining joined the author list, the problem with MDL PCA was solved. The analysis is now greatly simplified by removing the transmission of the model from the cost, i.e. the mean and the eigenvectors are transmitted with infinite precision *for free*. What is left is the cost of transmitting the principal components. This is almost trivial since we are dealing with orthogonal variables transmitted one by one and assumed to be Gaussian. The cost of transmitting component m is $\log(\lambda_m)$ for $\lambda_m \geq \lambda_\mathrm{cut}$ where $\lambda_\mathrm{cut} = \sigma_\mathrm{cut}^2$ is related to the principal component discretization step

Δ through $\sigma_{\mathrm{cut}} = 2\Delta$. So for a shape with rms radius 100 pixels, $\sigma_{\mathrm{cut}} = 0.003$ corresponds to a discretization of the MDL representation of on the average 0.15 pixels. Below λ_{cut} the cost approaches a constant as described in Section 3.

Now there is little left of the delicate balance between model complexity and data misfit, which is the strength of MDL, and the optimal number of principal components is no longer determined. However, this is not needed for the point correspondence problem and this cost is sufficiently powerful to guide the search for the optimal shape parameterisation. As Ockham said, *one should not do with more, what one can do with less*, and in this sense, the simpler 2002 version is the true way of doing MDL shape analysis.

The new way is not only simpler, it is also more consistent. It can be considered as a reformulation of the problem in a proper reference frame - a transformation of the data to the natural coordinates of the problem. These coordinates, the principal components, are independent of the granularity n by which the shapes are sampled spatially, as demonstrated in Section 3. The formalism has one important and meaningful parameter σ_{cut}, which controls the desired level of detail in the shape modelling as explained in Section 3.

It is interesting to note the circular path of history of the point correspondence problem. The Davies 2002 paper returns to a cost, which is close to the original, intuitive notion of compactness of Kotcheff and Taylor 98, and the recommended optimisation method is again a genetic algorithm.

5. Examples from medical images

The MDL method is applied to a set of 24 contours of metacarpals deduced from standard projection radiographs of the hand in the posterior-anterior projection. We use 64 marks, 8 nodes and a master example, and the algorithm converges after 40 passes in 10 minutes - see Figure 4.

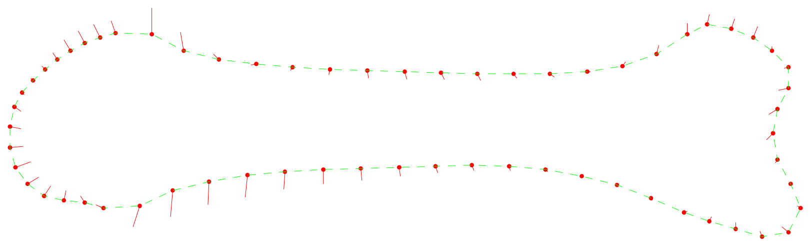

Fig. 4. MDL shape analysis of the second metacarpal. The mean and 3 standard deviations of the first principal component are displayed.

The method is also applied to a set of 32 contours of femurs in the supine projection deduced from projection X-rays (Figure 5). This is treated as an open contour with free end-points. Using a single master example causes a slight run-away, so the more powerful control method mode 2 is used instead with $T = 0.05$. The target parameters at both ends are set to 0.04 and the internal node parameter targets are 0.5, corresponding to marks distributed on average evenly in arc length. This is the most neutral choice, and the 9 nodes of the shapes will then in general not relax at conspicuous locations. If such marks are needed, e.g. for visual validation they can easily be constructed afterwards by interpolation between marks. The computation uses 65 marks, 9 nodes, 40 passes and takes 11 minutes.

In both examples is was checked that starting with different initial conditions leads to the same minimum, so there is no sign of problems with local minima, and therefore there is no need to use genetic algorithms for shapes of this kind.

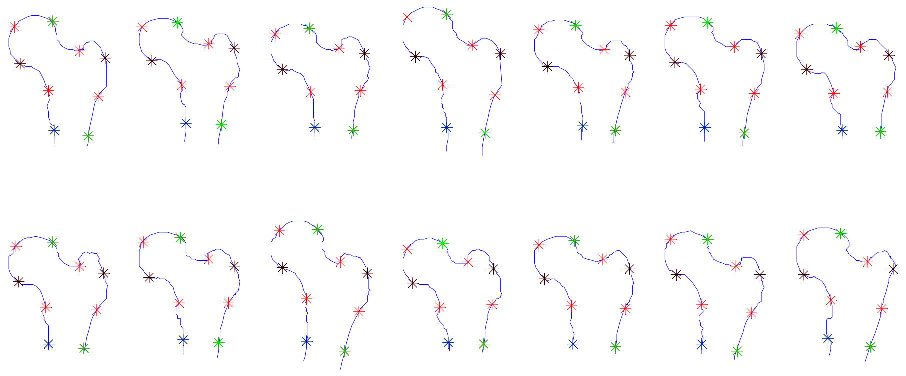

Fig. 5. Result of MDL analysis of femur contours. Here 14 of the 32 examples are shown with the optimised node positions. It is seen that they appear to be placed in a corresponding manner, and the free end-points have selected different portions of the available shafts.

6. Generalisation to MDL Appearance Model

This work was motivated by its use for creating training examples for ASM and AAM. In the introduction, this problem was divided into two parts: (1) Finding the shape contours and (2) defining marks on the contours, and only the latter problem was treated so far. In this section it is briefly proposed how the MDL approach could be extended to attack the first problem as well.

First notice that the point correspondence problem can be viewed as segmentation on a contour: the bump and the sides of the boxes are the segments of the object. These segments are annotated on the master example and the task of the optimisation is to locate the corresponding segments on the other examples. Thus we are performing *unsupervised segmentation with a single labelled example*. The problem is solved by brute force using a greedy algorithm minimizing the coding length of the shapes expressed in the PCA frame of reference.

Now consider the analogous problem of *unsupervised image segmentation*: Assume that we have 100 images of objects of a certain class, for instance X-ray images of a specific bone in a specific view. With just one example labelled by marks on the object boundary, it is often possible for humans to correctly segment the other 99 images in a corresponding way. However, this problem has not been solved in computer vision, despite the human proof-of-concept, despite the huge computer power available, and despite the obvious application e.g. for construction of AAMs for medical imaging. In addition it is an interesting cognitive question how humans manage to solve the task. It is suggested that Ockham's principle of minimum description length or economy in explanation can be the guiding principle of this process. If so, the human mind must have extra-ordinarily flexible and powerful optimisation skills. The successful and fast MDL shape model reviewed and improved in this paper naturally encourages us to apply a similar solution to the unsupervised object segmentation problem.

The following scheme is proposed: Match the master shape approximately onto the unlabeled examples using a rigid transformation. This can be done either manually or by some simple template matching method based on image correlation.

The shapes on each example are now allowed to reparameterise individually along the contour, exactly as in the MDL shape model as shown in Figure 6 left using the same hierarchical system of nodes. But the shapes are now also allowed to deform by displacements *perpendicular* to the curve, as shown in Figure 6 right. Again this is done in a coarse-fine manner using the same node hierarchy, where the displacement drops linearly to zero towards the neighbouring nodes on the same level.

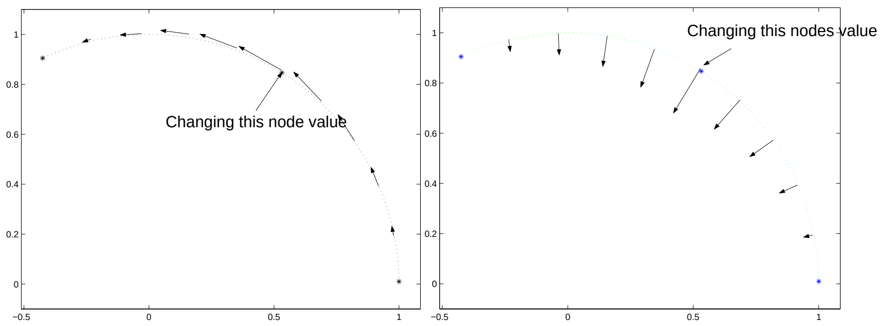

Fig. 6. The reparametrisation used in MDL shape model (left) and the additional reparametrisation (right) introduced in the MDL Appearance Model for unsupervised vision.

The cost function is based on an appearance model of the set of images. An image template is defined in terms of a triangulation controlled by the shape marks. This is designed on the labelled example. Auxiliary points are constructed from the shape marks in order to span a margin around the object as is often done in AAM [8], and the triangulation of the image template is defined using the marks and the auxiliary points as vertices. Image texture sampling points are defined relative to this mesh. A priori knowledge can be inserted into the modelling at this stage by increasing the density of sampling points in certain areas of the object, e.g. more points near the shape boundary and fewer points in irrelevant areas. In addition to sampling image intensities, edge intensities can be sampled [9] to emphasise that we want accurate modelling of the edges.

For any placement of the marks on the examples, a shape and a texture PCA model can be built and combined to an appearance model. The cost function is then defined in terms of the eigenvalue spectrum and a suitable cut-off value.

The algorithm is slower than the MDL shape method due to the sampling of some or all of say 10,000 texture values every time an example is reparameterised, but as demonstrated by Stegmann [10] this can be speeded up dramatically using modern graphics cards.

7. Conclusions

The MDL shape method was reviewed and the following original contributions were presented:

1) Correction of the 2001 PCA MDL formula.
2) Explanation of the problems of full MDL on PCA of shapes.
3) Efficient treatment of closed and open curves.
4) The "Mohammed and the mountain" trick.
5) Alternative control of run-away using average node parameters.
6) A new optimisation scheme with adaptive step length and a multiple coarse-fine strategy.
7) Efficient numerical method: 10 minutes on a 1.2 GHz machine in Matlab.
8) Open source Matlab code and test examples.
9) Generalisation to MDL Appearance Models (a kind of "Manchester United") for unsupervised image segmentation from one labelled example.

Correspondence with Carole Twining is acknowledged. Pronosco is acknowledged for providing the contours of metacarpals and femurs.

References

1. Cootes, T.F Hill, A., Taylor, C. J., Haslam, J.: The use of active shape models for locating structures in medical images. Image Vis. Comput. vol 12 (1994) 355-366.
2. Cootes, T.F., Edwards, G. J., and Taylor, C. J.: Active appearance models. In Proc. European Conf. on Computer Vision, volume 2, Springer (1998) 484–498.
3. Kotcheff, M. and Taylor, C.J.: Automatic construction of eigenshape models by direct optimisation, Med. Image Anal., vol 2 (1998) 303-314.
4. Davies, R.H., Cootes, T.F, and Taylor, C.J.: A Minimum Description Length Approach to Statistical Shape Modeling. 14th IPMI, Springer, 2001.
5. Davies, R.H., Twining, C.J, Cootes, T.F, Waterton, J. C and Taylor, C.J: A Minimum Description Length Approach to Statistical Shape Modelling, IEEE Trans Med. Imaging, Vol 21 (2002) 525-537.
6. Dryden, I. L., Mardia, K.V, Statistical Shape Analysis, Wiley (1998)
7. Minka, T.P: Automatic choice of dimensionality for PCA, Technical Report 514, MIT Media Laboratory, 2000
8. Thodberg, H. H.: Hands-on Experience with Active Appearance Models, Medical Imaging 2002: Image Proc., Eds. Sonka & Fitzpatrick, Proc. SPIE Vol. 4684 (2002), 495-506.
9. Cootes, T.F and Taylor, C. J.: On representing edge structure for model matching, Proc. IEEE CVPR, vol. 1, (2001) 1114–1119.
10. M. B. Stegmann, Analysis and Segmentation of Face Images using Point Annotations and Linear Subspace Techniques, IMM Technical Report IMM-REP-2002-22, (2002)