Noname manuscript No.
(will be inserted by the editor)

Using Bounded Diameter Minimum Spanning Trees to Build Dense Active Appearance Models

Robert Anderson · Björn Stenger · Roberto Cipolla

Received: date / Accepted: date

Abstract We present a method for producing dense Active Appearance Models (AAMs), suitable for video-realistic synthesis. To this end we estimate a joint alignment of all training images using a set of pairwise registrations and ensure that these pairwise registrations are only calculated between similar images. This is achieved by defining a graph on the image set whose edge weights correspond to registration errors and computing a bounded diameter minimum spanning tree (BDMST). Dense optical flow is used to compute pairwise registration and a flow refinement method to align small scale texture is introduced. Further, given the registration of training images, vertices are added to the AAM to minimise the error between the observed flow fields and the flow fields interpolated between the AAM mesh points. We demonstrate a significant improvement in model compactness.

Keywords Active Appearance Models · Groupwise registration · Minimum spanning trees

1 Introduction

Active Appearance Models (AAMs) are statistical models of both shape and appearance. Since their introduction fifteen years ago [6], they have been used extensively for tracking as they allow robust and efficient registration over a variety of different object classes [7, 25, 26]. More recently, AAMs have been growing in popularity for synthesis of faces, for example in emotion synthesis [1], expression transfer [36] and visual text-to-speech applications [12]. In order to train an AAM, a set of points must be consistently labelled in a collection of training images. Since this is usually carried out by hand the number of points is small (less than 100), leading to models such as the one in Figure 1(a). While automatic model building methods have been proposed previously [3, 28, 39], these do not produce results of sufficient accuracy for the synthesis of high-resolution images.

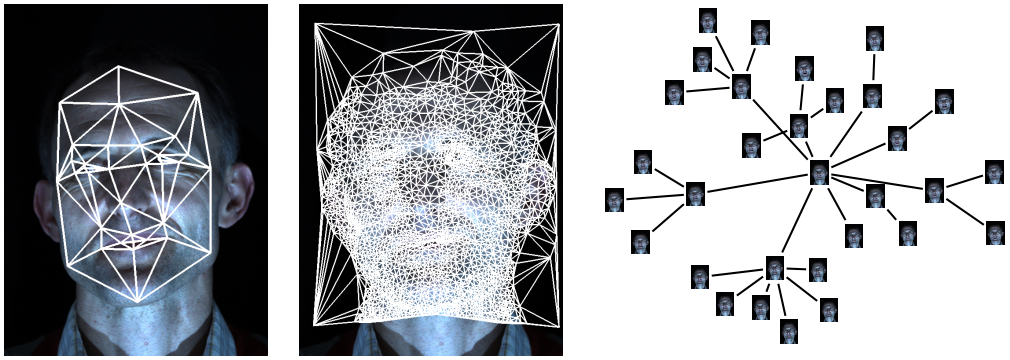
The task addressed in this paper is building dense AAMs, such as the example shown in Figure 1(b), in order to generate new video-realistic synthetic sequences. The underlying problem that needs to be solved in order to build such models is one of joint non-rigid image alignment. There exists a large body of work on this problem [8–10, 14, 16, 20, 24, 32], the majority of which registers each image to an iteratively updated base model. In this paper we propose a method that instead of registering all images to a base model registers images in a pairwise fashion. We find a bounded diameter minimum spanning tree (BDMST) on a graph containing all of the images, where each image is a node and each edge represents a warp between the two images it connects (see Figure 1(c)). The motivation for this approach is the fact that with current pairwise registration methods a low alignment error can only be achieved between similar images. Given the spanning tree all images can be registered to a common reference frame, solving the joint alignment problem.

In this paper we use a dense optical flow algorithm to align two images. Current methods use a coarse-to-

R. Anderson
Cambridge University Engineering Department
Cambridge, CB2 1PZ, UK
E-mail: ra312@cam.ac.uk

B. Stenger
Toshiba Research Europe Ltd
Cambridge, CB4 0GZ, UK

R. Cipolla
Cambridge University Engineering Department
Cambridge, CB2 1PZ, UK



(a) (b) (c)

Fig. 1: Overview: *From a sparse, 37 point AAM (a) the proposed method creates a dense, 1000 point AAM (b) suitable for image synthesis tasks. To achieve the dense registration required to construct the AAM we find a bounded diameter minimum spanning tree on a graph defined on the training images (c). On the digital version it is possible to zoom in to see the individual training images.*

fine approach, which often fails to register small scale texture. We therefore introduce an optical flow refinement technique that is particularly suited to registering regions containing only fine texture. Once the joint alignment is computed we present a method for densifying AAMs that applies ideas from digital terrain modelling. To summarise, the contributions of this paper are:

1. An application of BDMSTs to the joint alignment problem.
2. A method of optical flow refinement suited to regions which contain fine texture.
3. A method for densifying an AAM given dense correspondences between training images.

The paper is organised as follows: §1.1 reviews prior work and §2 describes the use of the BDMST to obtain a joint alignment. BDMSTs are contrasted against shortest path trees in the context of this problem in §3. Details of the optical flow algorithm used are given in §4 and densification of an AAM given a joint alignment of all training images is presented in §5. Subsequently, §6 shows the effect of different parameter settings, compares the proposed technique with competing methods and demonstrates the effectiveness of our approach for the AAM synthesis task. Conclusions and directions for future work are given in §7.

1.1 Related work

There exists a significant body of work on automating the process of registering images for model building. The two main approaches are (1) to iteratively refine the model itself [3,28,38] or (2) to solve a joint alignment problem and simultaneously build a model from the registered images [8,9]. We briefly review each of these.

Model-based approaches. The work of Vetter *et al.* [38] builds linear models automatically by using optical flow to register each new training image to the closest image that can be generated by the current linear model. The same technique has been applied to 3D data to automatically build morphable models [4].

Automatic AAM construction was formulated as an image coding problem by Baker *et al.* [3]. While this approach is theoretically appealing, it was only demonstrated on a dataset with little non-rigid deformation. Ramnath *et al.* [28] incrementally densify an AAM by using standard AAM fitting techniques but allowing for shape modes spanning the whole space of possible motions. This technique produces good results but is dependent on the initial input mesh as points are penalised for moving from their hand labelled positions.

While most techniques work on images given in an arbitrary order, some are designed specifically for ordered sequences where temporal constraints can be used [31, 39]. The problem can then be treated as tracking with an adaptive template. These approaches have the advantage of placing additional constraints on the registration problem, but they are not applicable to all scenarios.

The method of Tong *et al.* [37] solves a similar problem to AAM densification in which points are hand labelled in a few training images and these points are then propagated to more unlabelled images, whilst also removing noise on the initial hand labelled data. This can significantly reduce the amount of hand labelling necessary to build an AAM.

Joint alignment approaches. There are a large number of methods for pairwise image registration, for a survey see [43]. It has been shown, however, that there is often an advantage in registering sets of images *jointly* instead of in a pairwise manner [8], taking advantage of all the information present in the image data. Current state-of-the-art results for automatic model building are achieved by performing joint image alignment and warping points from a reference frame onto all images [8,9]. Cootes *et al.* [8] formulate joint image alignment as a Minimum Description Length encoding problem in which an efficient encoding of the image set represents a good registration. This approach produces very good results, however if features only appear in a small subset of the images then problems arise since all images are registered to a mean image which may not contain these features.

Marsland *et al.* [24] demonstrate an iterative joint alignment process in which all training images are compared to one of the training images instead of their mean. The choice of this reference image may change during the joint alignment process. However, even the optimal choice of reference image may be unsuitable for data sets exhibiting significant variation.

Sidirov *et al.* [32] have demonstrated promising results by directly optimising an objective function that minimises the difference between each image and the mean of all other images warped into a common reference frame. This objective function is highly non-convex and they use a stochastic technique to optimise it which randomly selects a set of control points within the image space at each iteration, helping to avoid local minima. More recently the method was extended to registering texture mapped surfaces [33].

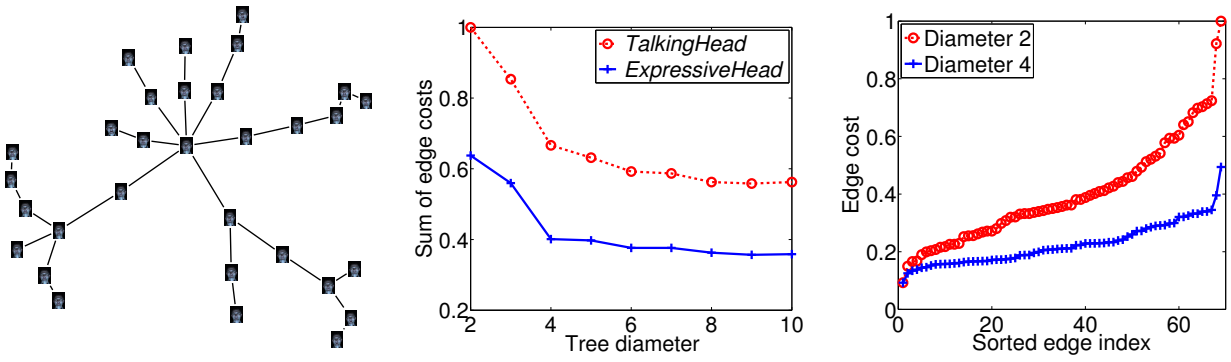
The so-called *congealing* approach presents another alignment method by optimising an allowable transformation for each image with the goal of minimising the entropy of the set of transformed images [20]. These allowable transformations are general in that they can be spatial transforms such as an affine warp, brightness or colour transforms, however this approach has not been demonstrated on warps with high enough dimensionality to express an arbitrary dense deformation field. A related approach is the RASL method of Peng *et al.*[27] which enforces sparsity to find a low rank representation of a set of images whilst simultaneously aligning them. By enforcing a low rank representation this method treats features that only appear in a few images as outliers, making it very well suited to challenging datasets that contain gross errors but less well suited to sets of images which have been carefully selected to avoid outliers, as is the case when building a dense AAM of a single target. While the initial work of [27] used affine transformations to align images the approach has been extended to allow for non-rigid deformation [35,42], representing deformation by a triangulated mesh of control points on the face.

Related uses of trees. Some types of tree have already been used in the joint alignment problem, Cristinacce and Cootes [10] use a shortest path tree over clusters of images. The tree is used to determine what order images are added to a joint alignment rather than for concatenating deformations as we do. The most similar approaches to the one proposed are the geodesic methods from the medical literature. Hamm *et al.*[14] use a k-nearest neighbour graph to approximate a manifold upon which valid images lie. Images are then registered by concatenating diffeomorphisms along the shortest path in the tree as an approximation to following the shortest geodesic distance over this manifold. Hernandez *et al.* [15] present an alternative method which is not tree based but relies upon the same underlying concept. Minimum spanning trees have also been used for registration in the medical literature as a method of estimating Rényi entropy [23,29] which is used as a measure of image alignment quality. Recently tree structures have been applied to the problem of registration over long sequences by Klaudiny and Hilton [18]. The proposed traversal tree reduces drift over long sequences by aligning images non-sequentially.

2 Joint alignment using a BDMST

We densify AAMs using a two-stage process; first a joint alignment is found between all of the training images $\{I_i\}_{i=1}^{N}$, then additional mesh vertices are added automatically to one training image and propagated to all others using the joint alignment. In order to find the joint alignment between all training images we calculate pairwise registrations between selected training images in the form of flow fields. All images are registered to a common frame by concatenating these flow fields.

The advantage of using pairwise registration is that when the correct image pairs are chosen each image is only registered with images that are similar in appearance. This is particularly important when there are features that only appear in a few images as it ensures that these features are well registered between the images in which they appear, whereas if these images were registered to a mean image lacking these features then registration is likely to fail in these regions.



(a) (b) (c)

Fig. 2: Effects of changing the tree diameter. *(a) A minimum spanning tree with diameter $D = 10$ for the ExpressiveHead dataset. (b) Sum of edge costs for a range of minimum spanning tree diameters. (c) Individual edge costs in a minimum spanning tree of diameter 2 (red) and diameter 4 (blue) arranged in ascending order for the ExpressiveHead dataset.*

2.1 Choice of registrations to compute

Joint alignment through concatenating pairwise flows can be viewed in a graphical form. Each node in the graph represents an image and each edge represents a dense mapping computed between two images. To be able to jointly align all of the images a spanning tree must be found so that all of the images are part of the same connected graph.

In order to minimise registration errors we wish to register between images that are as similar as possible. We assign each edge a cost C_{ij} for registering images I_i and I_j where C_{ij} is a measure of error between the two images after alignment. In order to register all images whilst minimising the sum of the pairwise registration errors we can now find a minimum spanning tree of the graph.

The problem with using a standard minimum spanning tree is that to register all images to a common image it may be necessary to concatenate a large number of flow fields along a path. This leads to error accumulation and results in a poor joint alignment. In order to limit the number of flow fields that need to be concatenated to reach the reference image from any other image we place a bound on the diameter of our minimum spanning tree. The diameter D is defined as the largest number of edges in any path between any two nodes in the spanning tree, for example the minimum spanning tree in Figure 2(a) has a diameter of 10. For comparison, the tree in Figure 1(c) has a diameter of 4, leading to lower accumulation of error when all of the images are registered to a single frame. We register all images to the image at the root of the tree, requiring concatenation of at most $\frac{D}{2}$ flow fields. The task of choosing which pairwise registrations to compute now becomes one of choosing a diameter D and finding a corresponding BDMST.

2.2 Finding a bounded diameter minimum spanning tree

The theory of BDMSTs has been well studied in the graph literature. For trees containing N nodes and with a diameter of D, where $4 \leq D < N - 1$, finding an optimal tree is NP-hard [13] and hence a number of heuristic methods have been proposed [2,17,34]. For trees with a small diameter it has been shown that randomised approaches give better results than greedy ones [17]. This motivates our use of the randomised centre-based tree reconstruction algorithm proposed by Julstrom [17]. This method repeatedly constructs BDMSTs in a semi-randomised way and retains the tree with the lowest sum of edge weights.

To allow a tree to be built we need a measure of how well two images can be registered. To approximate this cost we register two images to the same frame using optical flow and compute the L^2-norm of the image difference. In order to make the cost symmetrical we sum the scores achieved by warping to both image I_i and image I_j:

$$C_{ij} = |I_i - W_{ij}(I_j)|^2 + |I_j - W_{ji}(I_i)|^2, \quad (1)$$

where W_{ij} represents the warp from image I_j to image I_i computed by optical flow.

2.3 Choice of tree diameter

Figure 2(b) shows the total weight of all the edges in a spanning tree calculated for different diameters using the method above. The input comes from two different datasets, *TalkingHead* and *ExpressiveHead*, details

of which are given in §6. It can be seen that the decrease in the sum of the edge costs and hence the gain made through registering similar images starts to drop off rapidly after $D = 4$, suggesting that this is a reasonable choice for the tree diameter.

Figure 2(c) shows the edge weights for one of the datasets for a graph with a diameter of $D = 2$ (which corresponds to registering all images to one base image) and $D = 4$. It can be seen that the average edge cost is lower for the case where $D = 4$, indicating that each individual registration problem has a lower error in this case.

3 Comparison with shortest path trees

The above approach is based upon the the assumption that the best registration is achieved by minimising the sum of the weights of all edges in the spanning tree used for registration. Another aim could be to reduce the average path cost from the root node of the tree to each of the other nodes. This can be achieved by finding the shortest path tree either in the original graph or in a k-nearest neighbour (KNN) graph derived from the original graph.

Here we wish to present a toy example which demonstrates why minimising the total weight of the edges in the graph and hence using a BDMST is more appropriate than minimising the average path cost to the root node and therefore using a shortest path tree. The graph shown in Figure 3(a) represents an example of the common case in which there are some images which are similar between themselves but which are dissimilar to the rest of the graph. In this case A and B are similar to each other but dissimilar to C and D. In such a case the best registration will be achieved if C and D are registered to each other and A and B are registered to each other and only one of the high cost links (which is likely to represent an inaccurate registration) is used to join the two pairs. The minimum spanning tree shown in Figure 3(b) successfully achieves this, whilst the spanning tree found using a shortest path tree shown in Figure 3(c) does not. The root of the shortest path tree was chosen by trying all roots and selecting the one which gave the lowest average path cost. Assuming that a low edge cost corresponds to an accurate registration then when the BDMST in Figure 3(b) is used A and B will be well registered while when the tree in Figure 3(c) found using the shortest path approach is used A and B will be poorly registered despite being very similar.

4 Optical flow refinement

To register images we choose dense flow fields as a transformation model. While significant advances have been made in the development of robust optical flow algorithms, one of their drawbacks is that the commonly employed coarse-to-fine approach does not allow for the registration of structures which are displaced by a distance greater than their own size, due to the structure being completely blurred away at the pyramid level that would allow its matching [5]. To build an accurate, dense AAM we need to register structures that are only a few pixels in size but which may be displaced by several pixels within the training images. To overcome this limitation we therefore propose a two-stage approach. An initial estimate of the flow is computed using a modern optical flow algorithm and is used to approximately align two input images; we use the implementation of Liu [21]. A refinement step is then carried out calculating flow between the two partially aligned images over a small range of $\pm r$ pixels, where we have empirically set $r = 15$.

We formulate the flow refinement problem as a Markov Random Field (MRF) optimisation. Since we allow for displacements of $\pm r$ pixels in both the horizontal and vertical directions this results in $(2r+1)^2$ possible integer displacements. Optimising simultaneously over all displacements would result in a large number of labels that is impractical for current standard MRF optimisers. To make the problem computationally tractable, we estimate the horizontal and vertical components of the flow field independently, reducing the number of labels to $(2r + 1)$ in each case. In the following we outline the method for solving for horizontal flow $\mathbf{u}$, the method for calculating vertical flow being analogous.

We define an MRF with one node per pixel, connected in a 4-neighbourhood with the following energy terms. The pairwise term P_{ij} between two connected nodes, i and j, takes the form of a truncated quadratic,

$$P_{ij} = \alpha \min\left(\left|u_i - u_j\right|^2, P_{max}\right), \tag{2}$$

where P_{max} is the point at which the quadratic is truncated and α is a weighting term. This pairwise term encourages piecewise smoothness.

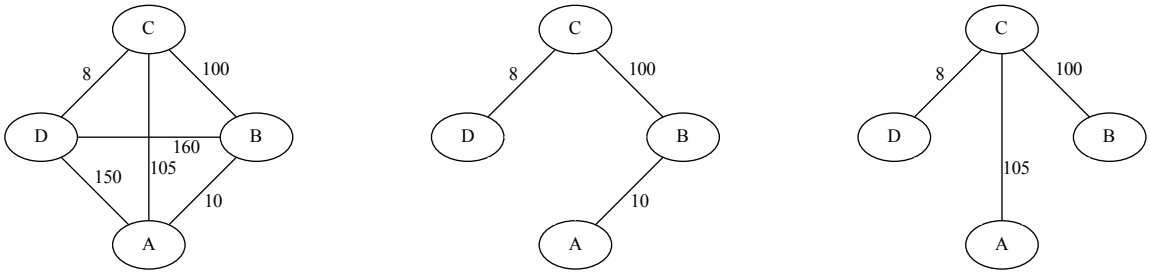
The unary term, $U(a, b, u)$, for a displacement of u pixels for the pixel in column a and row b is given by

$$U(a,b,u) = \min_{-r \le v \le r} \sum_{x=a-w}^{a+w} \sum_{y=b-w}^{b+w} D(x, y, x+u, y+v), \tag{3}$$

where

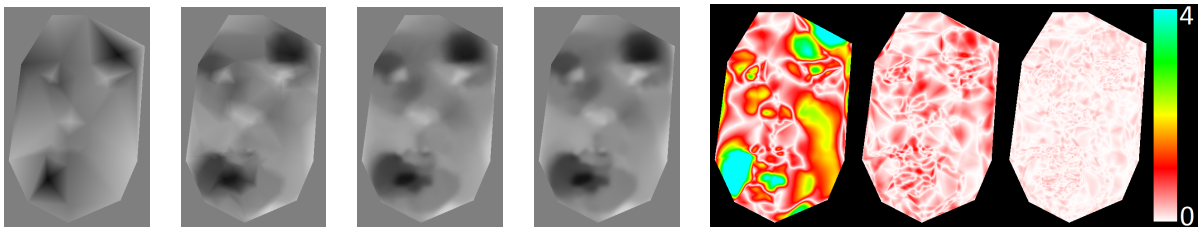
$$D(x_0, y_0, x_1, y_1) = \left|I_1(x_0, y_0) - I_2(x_1, y_1)\right|^2. \tag{4}$$

This is the minimum sum of squared differences (SSD) error, over a window of width $2w + 1$, that can be



(a) (b) (c)

Fig. 3: Toy example. *An example of a graph (a) on which a BMDST (b) will provide better registration than a shortest path tree (c) as only one costly (and hence inaccurate) edge is used.*



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

Fig. 4: Flow field improvement by AAM densification. *(a-c) Approximated horizontal flow fields for one image using meshes with 50, 250 and 1000 vertices respectively, (d) target horizontal flow field, (e) errors between the target flow field and approximations using 50, 250 and 1000 vertices.*

achieved for a horizontal displacement of u for any vertical displacement v within the allowed range of displacements $\pm r$. This term is made robust by thresholding the maximum cost at U_{max}. In order to efficiently compute the SSD scores for all images we take advantage of integral images. The constants were set empirically and were fixed at $\alpha = 100$, $w = 7$, $U_{max} = 2500$ and $P_{max} = 25$ for all experiments. The tree-reweighted message passing algorithm is used for optimisation [19]. This method optimises over a range of displacements without using any smoothing allowing for the alignment of fine texture. In our experience, the global nature of the MRF avoids the local minima created by the separation of the horizontal and vertical flow components.

5 Mesh densification

Once correspondence has been established between all training images, additional vertices are added to the model in order to increase its descriptive power. We aim to add points in a way that minimises the difference between the flow fields we have calculated and those which the mesh is able to model by using linear interpolation of flow between vertices. We aim to minimise at each pixel p the error function

$$E_p = \sum_{i=1}^{N} w_p \left| \mathbf{x}_i - \mathbf{y}_i \right|^2, \tag{5}$$

where $\mathbf{y}_i$ is the linear approximation to the flow field for training image i provided by the mesh, $\mathbf{x}_i$ is the calculated flow field for image i and w_p is a per-pixel weight. The weight w_p is used to increase the mesh density in regions with large variation and decrease it in smooth regions, such as the background. It is given by the sum of the local variation of each training image warped into the base image's reference frame.

In order to solve this problem we draw on ideas from the construction of digital terrain models. Instead of approximating a scalar height field we aim to approximate a vector field formed by pixelwise concatenation of the 2D flow vectors of all training images. We use the established, greedy method of DeFlorian [11] to solve this problem: The original vertices are used as an initial mesh and an approximation to the flow fields is constructed using linear interpolation between these points. Vertices are added iteratively at the point with the greatest error E_p and the mesh approximation to the flow fields is recalculated. Since this involves only

Fig. 5: Sample training images. *Training images from the TalkingHead dataset (top) and the ExpressiveHead dataset (bottom)*

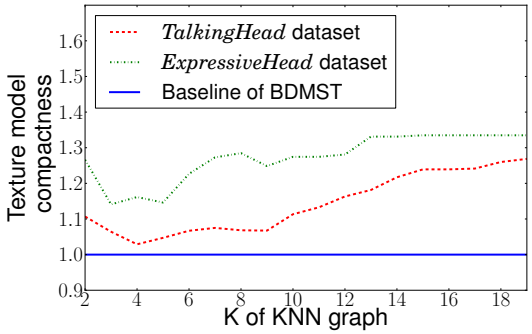


Fig. 6: Comparison with shortest path trees. *Texture model compactness achieved when using shortest path trees found on KNN graphs with varying values of K. The compactness values have been normalised relative to the values achieved using a BDMST of optimum diameter 4.*

triangles containing the inserted point this is an efficient local operation.

Figure 4 shows an example of the evolution of one component of a single flow field as the number of vertices is increased. We continue the densification until a fixed number of points have been added.

6 Experiments

To evaluate our approach we collected two datasets of data, both consisting of images of 800×600 resolution of a single subject:

1. *TalkingHead* - A set of 70 images. The AAMs built for this model are used to track an hour long sequence and train a visual text-to-speech (VTTS) model. The initial sparse AAM has 53 points.
2. *ExpressiveHead* - A sequence of 31 images demonstrating a greater degree of expressiveness than the *TalkingHead* dataset. The initial sparse AAM has 37 points.

Sample images from both datasets can be seen in Figure 5.

In order to compare the resulting models we use the model compactness measure used in [30]. This approximates the compactness of a Gaussian distributed model by the determinant of the model's covariance matrix, where a low value corresponds to a more compact model. Since the shape model component of the AAM is significantly more compact than the texture model we only report texture model compactness. In all cases the model compactness is scaled so that the baseline model has a compactness score of 1.

Before any processing we first high-pass filter all input images to reduce the effect of lighting variation. We subsequently warp all images into approximate correspondence using the initial hand-labelled sparse AAM points, making no further use of the hand labelling after this point.

Comparison with shortest path trees In § 3 our rationale for using a BDMST over a shortest path tree was given. We verify here that for the datasets used this is indeed the correct choice. Shortest path trees can be calculated either on an original fully connected graph or on a KNN graph derived from the original graph. Varying the value of K when computing the KNN graph has a similar effect to varying the diameter of a BDMST, with a larger value of K giving a more compact tree. A range of values were used for K on both datasets and the results can be seen in Figure 6 where the model compactness has been normalised against the best results achieved using a BDMST with a diameter of 4. Other than the choice of which edges to use to construct the tree, exactly the same process is used to build the models as for the BDMST. It can be seen that while for the *TalkingHead* dataset the difference between the best performing shortest path tree and the best performing BDMST is small, for the case of the more challenging *ExpressiveHead* dataset the difference is more pronounced.

Effect of tree diameter Figure 7(a) shows the effect of tree diameter on model compactness. There is a sharp decrease between a diameter of $D = 2$, corresponding to registering all images to the same base image, and $D = 4$, demonstrating that the spanning tree allows for more accurate registration. Note that a diameter value of $D = 4$ allows clusters of similar images to form (as seen in figure 1(c)). Beyond this diameter value there is a slow increase in error as the paths in the tree become

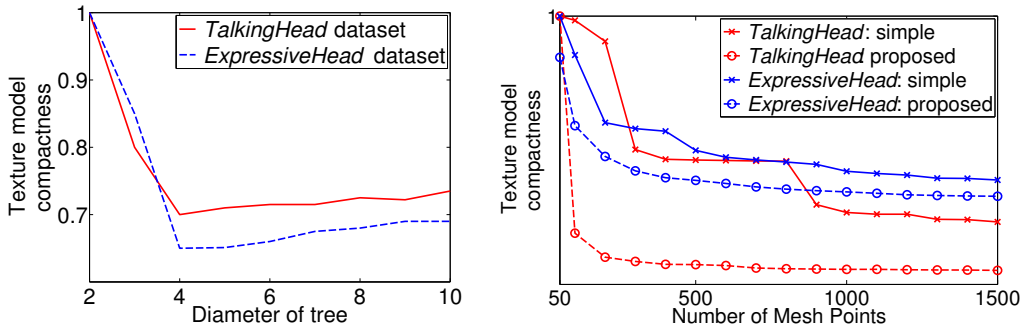


Fig. 7: Model compactness for varying parameters. *Graphs showing the effect of (a) varying tree diameter and (b) varying mesh density with simple densification (crosses) and the proposed densification (circles). Lower is better.*

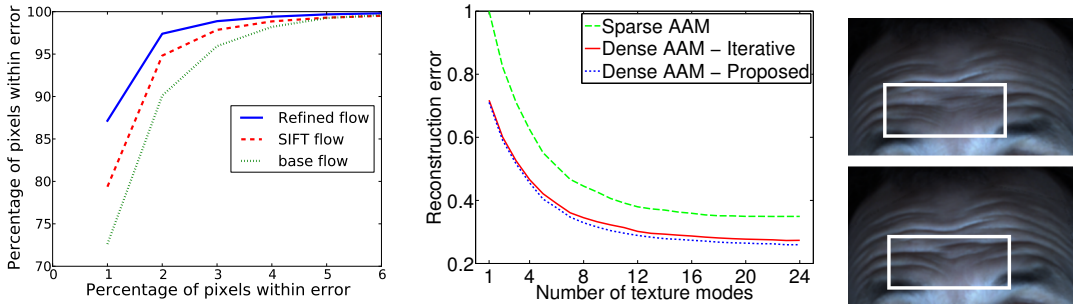


Fig. 8: Effects of flow refinement and varying number of AAM modes. *(a) Percentage of pixels for which flow was correctly calculated to within a given error, using the flow of [21] with (green) and without (red) our refinement. (b) Errors in reconstruction of a sequence using an AAM built using different approaches. (c) One frame of synthesis using an AAM built using iterative registration to a mean image (top) and the proposed method (bottom).*

longer, leading to error accumulation when concatenating flows.

Effect of densification method In order to demonstrate the effectiveness of the proposed mesh densification method presented in §5, we compare it to a simple densification approach where at each iteration a vertex is added at the centre of the largest triangle and then the mesh is re-triangulated. An example of a model densified using the proposed method can be seen in in Figure 1(b). Note the increased mesh density in areas which exhibit significant deformation, such as regions around the eyes, as opposed to more rigid regions like the nose. As can be seen from Figure 7(b) for both datasets the proposed method results in a more compact model for a given number of mesh points.

Effect of optical flow refinement To demonstrate the value of the proposed optical flow refinement method we compare it to the flow method of [21], which we use for initialisation. We also compare it to the SIFT flow approach [22] which is an existing method that deals well with large displacements. Due to the challenge of obtaining ground truth data for this type of non-rigid flow problem we set up the following experiment. Each image in the *ExpressiveHead* dataset is warped by a flow field calculated between two other random images in the data set. This procedure yields a set of warped images with known flow fields. Gaussian noise is added to the images and we then apply both SIFT flow and the flow method of [21], with and without the refinement proposed in section 4, and measure the per-pixel error in flow fields for the face region. In the case of SIFT flow we ran the code provided by the original authors with its default settings. Figure 8(a) shows the resulting errors as a cumulative percentage of pixels with errors below a given threshold. It can be seen that flow refinement results in significant error reduction, e.g. the percentage of pixels with a < 1 pixel error increased from 73% to 87%. The proposed method also outperforms SIFT flow for this task. This improvement in flow also leads to more compact models as fine texture is better aligned. In a separate experiment we find that flow refinement reduces the model compactness score by 17% on the *ExpressiveHead* dataset and 26% on the *TalkingHead* dataset.

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

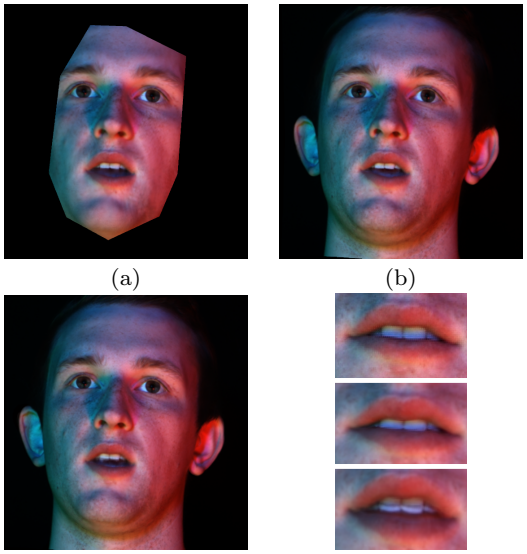
Fig. 9: Building low resolution models. *(a) A sample input image from which an AAM was automatically built (b). Plus and minus two standard deviations of the first (c) and second (d) modes of variation of the AAM are shown.*

Synthesis of an expressive sequence To demonstrate the advantage of using dense AAMs for synthesis we build an AAM of the *ExpressiveHead* dataset and use it to track a 450 frame long sequence and resynthesise it. We compare the original sparse model, a dense model built using the proposed method, and a dense model built using the popular method of jointly aligning all images to an iteratively updated mean image (similar to [8]). The dense models were constrained to have the same convex hull as the original model. Figure 8(b) shows the L^2-norm of the errors between the original image sequence that was tracked and the synthesised sequences. There is a large increase in accuracy when using a dense AAM instead of a sparse one. Figure 8(c) shows a close-up view of one the synthesised frames in which the model built by registering to a mean image (top) produces blending artefacts due to poor registration of the wrinkles that are only present in two of the training images. The model built using the proposed method shows no blending artefacts (bottom).

Results on lower resolution images To demonstrate the applicability of this method to lower resolution images and to show that in this case models can be learned completely automatically a model was built using a publicly available interview from YouTube. To train the model 30 images of resolution 480×320 (an example of which can be seen in Figure 9(a)) were extracted from the sequence and the model training approach was applied without using any initial correspondences. All of the training images were correctly registered and the first two modes of the resulting AAM can be seen in Figure 9. The mesh structure seen in Figure 9(b) shows that the approach does not waste vertices modelling the background.

Using a dense AAM for visual text-to-speech One application of dense AAMs is in visual text-to-speech (VTTS) systems [40]. To demonstrate the advantage of a dense AAM we train a VTTS system using a sparse AAM and a dense AAM built using the proposed method. Both models are used to track an hour long training video and the AAM parameters are used along with audio features to train a hidden Markov model based text-to-speech (HMM-TTS) system [41]. At synthesis time the HMM-TTS system generates audio and a set of new AAM parameters which are rendered using the AAM. As can be seen in Figure 10 the synthetic rendering using the dense AAM is significantly sharper than that obtained with the sparse AAM. We also built a dense AAM without providing an initial sparse mesh. This resulted in good results in most regions (Figure 10(c)) but was not able to fully handle the complex occlusions and disocclusions around the mouth, resulting in artefacts during synthesis (Figure 10(d)). Initial labelling in this region is currently still required in order to produce high-quality models on high resolution imagery.

Processing time All experiments were run on a single core of an 2.2GHz i7 processor. The total time to



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

Fig. 10: Synthesising novel speech using VTTS. *The same frame of synthesis using (a) the original sparse AAM, (b) a dense AAM built by densifying the sparse AAM and (c) a dense AAM built fully automatically. (d) top - the mouth region of a target image, middle - reconstruction using densified AAM, bottom - reconstruction using AAM built fully automatically. The model built fully automatically exhibits reconstruction artefacts.*

process the *TalkingHead* dataset was 3 hours and 10 minutes while the time to process the *ExpressiveHead* dataset was 1 hour and 12 minutes. The flow refinement step proposed in §4 dominates the running time, taking almost two minutes to process each 800×600 image pair. Calculating pairwise flows between all images to estimate edge weights for the BDMST scales with the square of the number of training images. Because of this, and since an accurate flow is not needed at this stage, the flow refinement step was not applied when estimating edge weights and quarter resolution images were processed. The time required for the other stages of the algorithm scales linearly with the number of images in the dataset.

7 Conclusions and future work

We have shown that concatenation of pairwise registrations can give good results on the joint image alignment problem when the correct registrations are chosen. We have demonstrated the use of a BDMST as a method for choosing these registrations. An MRF-based optical flow refinement technique and a method for mesh densification were demonstrated to improve the results in terms of model compactness. As a target application we have shown synthesis results of person specific active appearance models.

In the future we would like to investigate using graph structures other than trees to solve this problem, making use of more of the information made available when calculating pairwise registrations. We feel that the first step towards this would be to develop an improved error metric for measuring the accuracy of the pairwise registrations. We are also interested in allowing different regions within each image to be registered to different neighbours, allowing for each individual region to be aligned to more closely matching regions in the other images.

Acknowledgements We would like to thank Iain Waugh and Norbert Braunschweiler for allowing us to model their faces. We would also like to thank everyone in the Speech Technology Group at Toshiba Research Europe for their help with the visual text-to-speech component of this work.

References

1. Abboud, B., Davoine, F., Dang, M.: Facial expression recognition and synthesis based on an appearance model. Signal Processing: Image Communication **19**(8), 723–740 (2004)
2. Abdalla, A., Deo, N., Gupta, P.: Random-tree diameter and the diameter constrained MST. Congressus Numerantium pp. 161–182 (2000)
3. Baker, S., Matthews, I., Schneider, J.: Automatic construction of Active Appearance Models as an image coding problem. PAMI **26**(10), 1380–1384 (2004)
4. Blanz, V., Vetter, T.: A morphable model for the synthesis of 3D faces. SIGGRAPH pp. 187–194 (1999)
5. Brox, T., Bregler, C.: Large displacement optical flow. CVPR pp. 41–48 (2009)
6. Cootes, T., Edwards, G., Taylor, C.: Active appearance models. ECCV **2**, 484–498 (1998)
7. Cootes, T., Taylor, C.: Statistical models of appearance for medical image analysis and computer vision. In: SPIE Medical Imaging, vol. 4322, pp. 236–248 (2001)
8. Cootes, T., Twining, C., Petrović, V., Babalola, K., Taylor, C.: Computing accurate correspondences across groups of images. PAMI **32**(11), 1994–2005 (2010)
9. Cootes, T., Twining, C., Petrović, V., Schestowitz, R., Taylor, C.: Groupwise construction of appearance models using piece-wise affine deformations. BMVC pp. 879–888 (2005)
10. Cristinacce, D., Cootes, T.: Facial motion analysis using clustered shortest path tree registration. MLVMA Workshop (ECCV) (2008)
11. De Floriani, L.: A pyramidal data structure for triangle-based surface description. IEEE Computer Graphics and Applications **9**(2), 67–78 (1989)
12. Deena, S., Hou, S., Galata, A.: Visual speech synthesis by modelling coarticulation dynamics using a non-parametric switching state-space model. In: ICMI-MLMI, pp. 1–8 (2010)
13. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco (1979)
14. Hamm, J., Hye Ye, D., Verma, R., Davatzikos, C.: Gram: A framework for geodesic registration on anatomical manifolds. Medical Image Analysis **14**(5), 633–642 (2010)
15. Hernandez, M., Bossa, M., Olmos, S.: Registration of anatomical images using paths of diffeomorphisms parameterized with stationary vector field flows. IJCV **85**(3), 291–306 (2009)
16. Hill, D., Batchelor, P., Holden, M., Hawkes, D.: Medical image registration. Physics in Medicine and Biology **46**(3), R1 (2001)
17. Julstrom, B.: Greedy heuristics for the bounded diameter minimum spanning tree problem. J. Exp. Algorithmics **14**, 1:1.1–1:1.14 (2009)
18. Klaudiny, M., Hilton, A.: Towards optimal non-rigid surface tracking. ECCV pp. 743–756 (2012)
19. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. PAMI **28**(10), 1568–1583 (2006)
20. Learned-Miller, E.: Data driven image models through continuous joint alignment. PAMI **28**(2), 236–250 (2006)
21. Liu, C.: Beyond pixels: Exploring new representations and applications for motion analysis. Doctoral Thesis, MIT (2009)
22. Liu, C., Yuen, J., Torralba, A., Sivic, J., Freeman, W.: Sift flow: dense correspondence across different scenes. In: ECCV, pp. 28–42 (2008)
23. Ma, B., Hero, A., Gorman, J., Michel, O.: Image registration with minimum spanning tree algorithm. International Conference on Image Processing **1**, 481–484 (2000)
24. Marsland, S., Twining, C., Taylor, C.: Groupwise non-rigid registration using polyharmonic clamped-plate splines. MICCAI **2879**, 771–779 (2003)
25. Matthews, I., Baker, S.: Active appearance models revisited. IJCV **60**(2), 135–164 (2004)

26. Mittrapiyanuruk, P., DeSouza, G., Kak, A.: Calculating the 3d-pose of rigid-objects using active appearance models. In: International Conference on Robotics and Automation, vol. 5, pp. 5147–5152 (2004)
27. Peng, Y., Ganesh, A., Wright, J., Xu, W., Ma, Y.: Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images. In: CVPR, pp. 763–770 (2010)
28. Ramnath, K., Baker, S., Matthews, I., Ramanan, D.: Increasing the density of active appearance models. CVPR pp. 1–8 (2008)
29. Sabuncu, M., Ramadge, P.: Using spanning graphs for efficient image registration. IEEE Transactions on Image Processing **17**(5), 788–797 (2008)
30. Saragih, J., Goecke, R.: Learning active appearance models from image sequences. HCSNet workshop on Use of vision in human-computer interaction pp. 51–60 (2006)
31. Saragih, J., Goecke, R.: Monocular and stereo methods for aam learning from video. CVPR pp. 1–8 (2007)
32. Sidorov, K., Richmond, S., Marshall, D.: An efficient stochastic approach to groupwise non-rigid image registration. CVPR pp. 2208–2213 (2009)
33. Sidorov, K., Richmond, S., Marshall, D.: Efficient groupwise non-rigid registration of textured surfaces. CVPR pp. 2401–2408 (2011)
34. Singh, A., Gupta, A.: Improved heuristics for the bounded-diameter minimum spanning tree problem. Soft Computing11 pp. 911–921 (2007)
35. Smith, B., Zhang, L.: Joint face alignment with non-parametric shape models. In: ECCV, pp. 43–56 (2012)
36. Theobald, B., Matthews, I., Cohn, J., Boker, S.: Real-time expression cloning using appearance models. Proc. ACM Int. Conf. Multimodal Interfaces pp. 134–139 (2007)
37. Tong, Y., Liu, X., Wheeler, F., Tu, P.: Automatic facial landmark labeling with minimal supervision. Computer Vision and Pattern Recognition pp. 2097–2104 (2009)
38. Vetter, T., Jones, M., Poggio, T.: A bootstrapping algorithm for learning linear models of object classes. CVPR pp. 40–46 (1997)
39. Walker, K., Cootes, T., Taylor, C.: Automatically building appearance models from image sequences using salient features. BMVC pp. 463–562 (1999)
40. Wang, L., Han, W., Soong, F., Huo, Q.: Text driven 3D photo-realistic talking head. In: Interspeech, pp. 3307–3308 (2011)
41. Zen, H., Tokuda, K., Black, A.: Statistical parametric speech synthesis. Speech Communication **51**(11), 1039–1154 (2009)
42. Zhao, C., Cham, W., Wang, X.: Joint face alignment with a generic deformable face model. In: CVPR, pp. 561–568 (2011)
43. Zitova, B., Flusser, J.: Image registration methods: a survey. Image and Vision Computing **21**(11), 977–1000 (2003)