High-level feature extraction: Deformable shape analysis

COS 791

Chapter 6

In this lecture:

- Deformable shape analysis
- Parts-based shape analysis
- Active contours (snakes)
- Shape skeletonization
- Flexible shape models
  - Active shape modeling

Deformable shape analysis

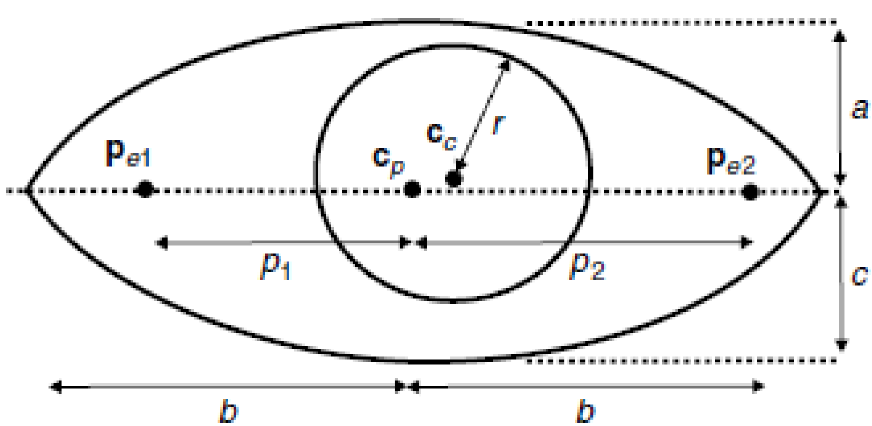
- The previous chapter found shapes by matching
  - This implies knowledge of a model of the target shape
  - The shape is fixed in the sense that it is only flexible in the choice of parameters that define the shape
- Sometimes it is not possible to model a shape with sufficient accuracy
  - The exact shape could be unknown or difficult to parameterize

Deformable shape analysis

- Need techniques that can evolve to the target solution or adapt their result to the data
- Implies use of flexible shape formulations
- If a shape is flexible (or deformable) so as to match the image data, we have a deformable template

Deformable templates

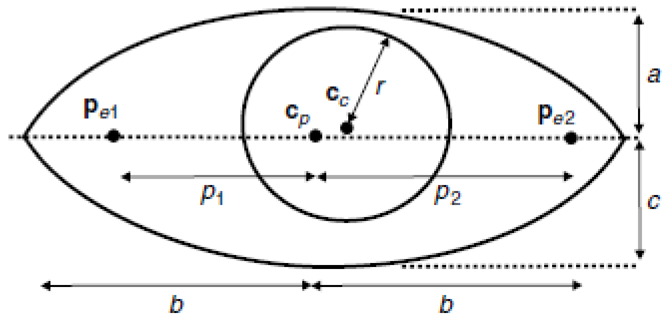
- Earlier approach by Yuille was aimed to find facial features for purposes of recognition
- An eye is comprised of an iris, which sits within the sclera
  - Can be modeled as a combination of a circle that lies within a parabola



Deformable templates

- The circle and parabola can separately be extracted the Hough transform, but not in combination
- When we combine the two shapes and allow them to change in size and orientation, while retaining their spatial relationship (the iris should reside within the sclera) we have a deformable template

Deformable templates



- The parabola is a shape described by a set of points (x,y) related by:

$$y = a - \frac{a}{b^2}x^2$$

Where a is the height of the parabola and b is its radius

- The circle is defined by a center coordinate c_c and radius r

Deformable templates

- We then seek values of the parameters which give a best match of this template to the image data
  - Fit the edge data to the template
- The set of values for the parameters which gives a template which matches the most edge points could then be deemed to be the best set of parameters describing an eye

Deformable templates

- Additional parameters can be added:
  - Values for parameters should be chosen that maximises the match between the edges and the template data as well as define that the inside of the circle should be dark and the inside of the parabola should be light
  - Weighted factors controls the influence of each factor on the eventual result
- Additional parameters and weights results in eleven parameters to seek for

Deformable templates

- One possibility is to simply cycle through every possible value
  - Given 100 possible values for each parameter, then we have to search 10^{22} combinations of parameters
- Two alternatives:
  - Using optimization techniques
  - Genetic algorithm approach

Parts-based shape analysis

- Rather than characterize an object by a single feature, objects are represented by a collection of parts arranged in a deformable structure
- Objects are modeled as a network of masses which are connected by springs
- The springs adds context to the position of the shape
  - The springs control the relationship between the objects and the object parts to move relative to one another

Parts-based shape analysis

- The extraction of the representation is then a compromise between the match of the features to the image and the relationships between the locations of the features

Parts-based shape analysis

- Say we have *n* parts (assume *n* = 3)
- $m_i(l_i)$ represents the difference from the image data when each feature f_1, f_2 and f_3 is placed at location l_i
- $d_{i,j}(l_i,l_j)$ is a function that measures the degree of deformation (by how much the springs extend) when features f_i and f_j are placed at location l_i and l_j respectively
- The best match of the model to the image is then:

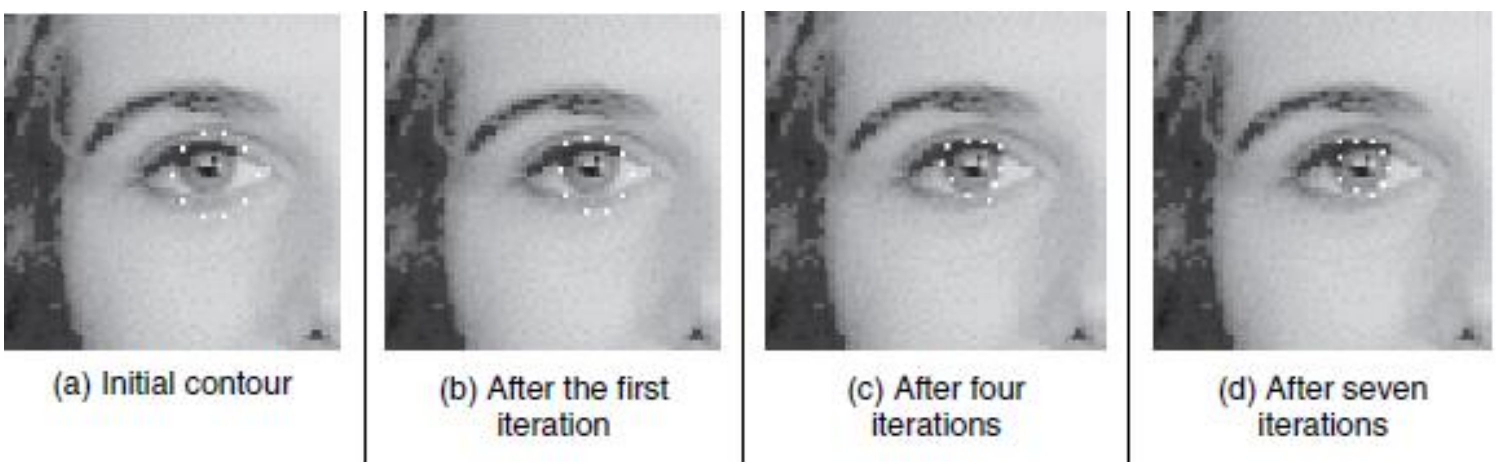
$$L^* = arg\left[min\left(\sum_{i=1}^{N} m_i(l_i) + \sum_{f_i,f_j \in \Re} d_{i,j}(l_i,l_j)\right)\right]$$

Parts-based shape analysis

- The components can be weighted
- The parameters derived from the previous function are the best compromise between the position of the parts and the deformation
- Determining these parameters is computationally difficult
- An alternative approach is to seek a technique that uses fewer parameters - snakes

Active contours (snakes)

- An active contour is a set of points which aims to enclose a target feature – the feature to be extracted
- An initial contour is placed outside the target feature and is then evolved so as to enclose it



(a) Initial contour

(b) After the first iteration

(c) After four iterations

(d) After seven iterations

Active contours

- Active contours are expressed as an energy minimization process
- The target feature is a minimum of a suitably formulated energy functional
- A snake represents a compromise between its ability to bend and stretch and the edge magnitude

Active contours

- The energy functional is the addition of a function of the contour's internal energy (E_{int}), its constraint energy (E_{con}) and the image energy (E_{image})
- The snake $\mathbf{v}(s)$ is the set of x and y coordinates of the points in the snake

$$E_{snake} = \int_{s=0}^{1} E_{int}\big(\mathbf{v}(s)\big) + E_{image}\big(\mathbf{v}(s)\big) + E_{con}\big(\mathbf{v}(s)\big)\mathrm{d}s$$

The **internal energy** controls the natural behaviour of the snake

The **image energy** attracts the snake to edge points

The **constraint energy** allows higher level information to control the snake's evolution

Active contours

- The aim of the snake is to evolve the previous equation
- New snake contours are those with lower energy and are a better match to the target feature than the original set of points from which the contour has evolved
- We thus seek to choose a set of points v(s) such that:

$$\frac{\mathrm{d}E_{snake}}{\mathrm{d}\mathbf{v}(s)} = 0$$

Active contours

- The energy functionals are expressed in terms of functions of the snake and of the image
- These functions contribute to the snake energy along with weighting coefficients
- The **internal image energy** is defined as a weighted summation of first- and second-order derivatives around the contour

$$E_{int} = \alpha(s)\left|\frac{\mathrm{d}\mathbf{v}(s)}{\mathrm{d}s}\right|^2 + \beta(s)\left|\frac{\mathrm{d}^2\mathbf{v}(s)}{\mathrm{d}s^2}\right|^2$$

Active contours

$$E_{int} = \alpha(s)\left|\frac{\mathrm{d}\mathbf{v}(s)}{\mathrm{d}s}\right|^2 + \beta(s)\left|\frac{\mathrm{d}^2\mathbf{v}(s)}{\mathrm{d}s^2}\right|^2$$

- The first-order differential measures the energy due to stretching – high values indicate a high rate of change
- The weighting coefficient α controls the point spacing
- Low values for α imply the points can change in spacing greatly and high values imply the snake aims for evenly spaced contour points

Active contours

$$E_{int} = \alpha(s)\left|\frac{\mathrm{d}\mathbf{v}(s)}{\mathrm{d}s}\right|^2 + \beta(s)\left|\frac{\mathrm{d}^2\mathbf{v}(s)}{\mathrm{d}s^2}\right|^2$$

- The second-order differential measures the energy due to bending or curvature
- The weighting coefficient β controls the contribution of point variation
- Low values for β imply that the contour can form corners in its perimeter, high values results in smooth contours
- α and β controls the shape the snake aims to attain

Active contours

- The **image energy** attracts the snake to low-level features
- Lines (E_{line}), edges (E_{edge}) and terminations (E_{term}) could contribute to the energy function
- The image energy is then:

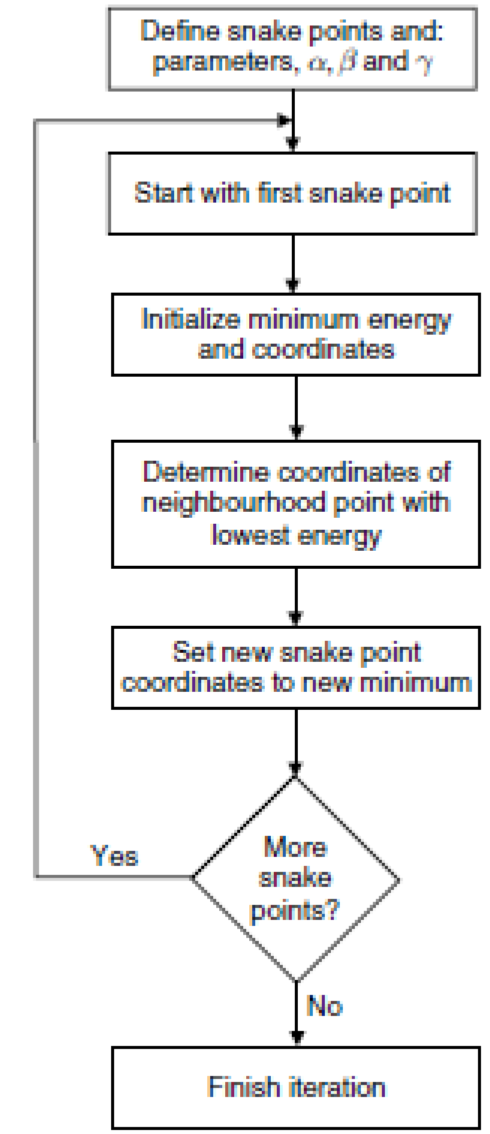
$$E_{image} = w_{line}E_{line} + w_{edge}E_{edge} + w_{term}E_{term}$$

Active contours

- The **line energy** can be set to the image intensity at a particular point
  - If black has a lower value than white then the snake will be attracted to dark features and vice versa
- The **edge energy** can be computed by an edge detection operator
  - For example the Sobel operator
- The **termination energy** can include the curvature of level image contours
- Most common is to use edge, but lines can also be used

Greedy algorithm for snakes

- Williams and Shah
- The energy minimization process is implemented as a discrete algorithm



Greedy algorithm for snakes

- Process starts by specifying an initial contour
  - Can be specified manually
- Greedy algorithm then evolves the snake iteratively by searching local neighbourhoods for a new contour point with a lower snake energy
- At each iteration all contour points are evolved

Greedy algorithm for snakes

- For a set of snake points $\mathbf{v}_s$, the energy functional minimized for each snake point is:

$$E_{snake}(s) = E_{int}(\mathbf{v}_s) + E_{image}(\mathbf{v}_s)$$

- This is expressed as

$$E_{snake}(s) = \alpha(s)\left|\frac{\mathrm{d}\mathbf{v}_s}{\mathrm{d}s}\right|^2 + \beta(s)\left|\frac{\mathrm{d}^2\mathbf{v}_s}{\mathrm{d}s^2}\right|^2 + \gamma(s)E_{edge}$$

- Each contour point has associated values for α, β and γ

Greedy algorithm for snakes

- The first-order differential is approximated as the modulus between the average spacing of contour points and the distance between the current image point and the next contour point
  - Euclidean distance is used to calculate the distance
- This controls the spacing between points
  - The first order differential drops to zero when the points are evenly spaced

$$\left|\frac{\mathrm{d}\mathbf{v}_s}{\mathrm{d}s}\right|^2 = \left|\sum_{i=0}^{S-1} \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i-1})^2}/S - \sqrt{(x_s - x_{s+1})^2 + (y_s - y_{s+1})^2}\right|$$

Greedy algorithm for snakes

- The second-order differential can be implemented as an estimate of the curvature between the next and the previous contour points, $\mathbf{v}_{s+1}$ and $\mathbf{v}_{s-1}$, and the point in the local neighbourhood of the current point $\mathbf{v}_s$

$$\begin{aligned}\left|\frac{\mathrm{d}^2\mathbf{v}_s}{\mathrm{d}s^2}\right|^2 &= |\mathbf{v}_{s+1} - 2\mathbf{v}_s + \mathbf{v}_{s-1}|^2 \\ &= (x_{s+1} - 2x_s + x_{s-1})^2 + (y_{s+1} - 2y_s + y_{s-1})^2\end{aligned}$$

Greedy algorithm for snakes

- E_{edge} can be implemented as the magnitude of the Sobel edge operator at point x,y
- Since the snake is a minimization process the edge image is inverted so that points with highest edge strength are given lowest edge values and vice versa
- The snake will thus be attracted to edge points with the greatest magnitude

Greedy algorithm for snakes

- The energy for each snake point is first determined and is stored as the minimum
  - To eliminate points with equally small energies
- The local 3 x 3 neighbourhood is then searched to determine if another point has a lower energy than the current contour point
- If it does then that point is returned as the new contour point
- This process repeats an x number of times

Greedy algorithm for snakes

- Effects of varying α and β:
- Setting α to zero removes influence of spacing and points become unevenly spaced
  - Can also be placed on top of one another
- Can be desirable for features with high localized curvature

(a) Initial contour | (b) After iteration 1 | (c) After iteration 2 | (d) After iteration 3

Greedy algorithm for snakes

- Setting β to zero removes influence of curvature and allows corners to form in the contour
- Can again be beneficial for features with high localized curvature

(a) Initial contour (b) After iteration 1 (c) After iteration 2 (d) After iteration 3

Greedy algorithm for snakes

- Disadvantages:
  - Final solution can oscillate between two sets of points with equally low energy
    - Can be prevented by detecting occurrence of oscillation
  - As the contour becomes smaller the number of points constrains the result
    - Cannot fit points into too small a space
    - Only solution is to resample the contour

Complete (Kass) snake implementation

- The Greedy method iterates around the snake to find local minimum energy at snake points
- This is an approximation and does not necessarily determine the “best” local minimum
- A complete snake implementation ensures that the snake moves to the best local energy minimum

Complete (Kass)snake implementation

- Complete snake implementation uses calculus of variation to create a set of equations that iteratively provide new sets of contour points
- This is a continuous formulation as opposed to the discrete implementation of Greedy

Complete (Kass) snake implementation

- One penalty is the need for matrix inversion which affects speed
- The benefits are that that coordinates are calculated for real functions and the complete set of new contour points is provided at each iteration

Active contours

- Difficulty in using snakes for practical feature extraction:
  - The choices for values α and β (and others) must be made very carefully
  - Result depends on where the initial contour is placed

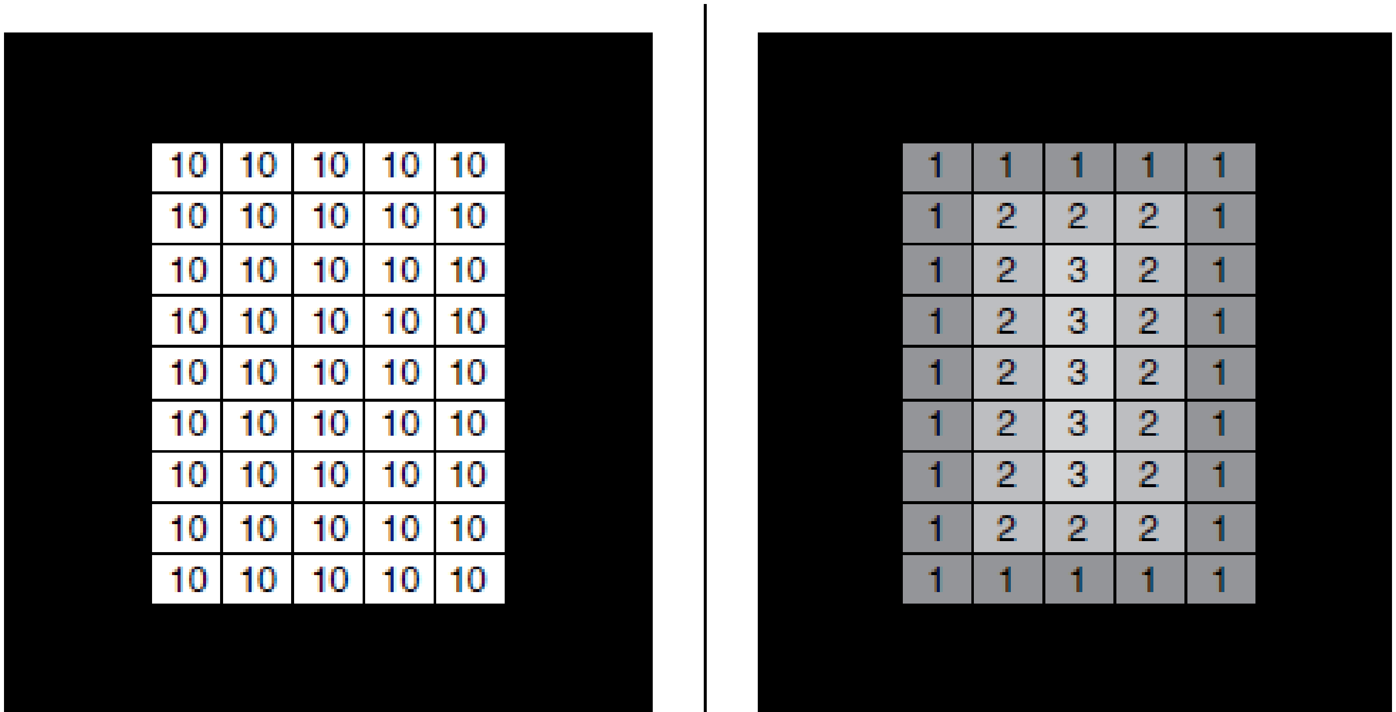
Shape skeletonization

- It is possible to describe a shape not just by its perimeter or its area but by its skeleton
  - A central axis to the shape
  - The axis which is equidistant from the borders of the shape
  - Central axis is determined by a distance transform
- Result is a representation that has the same topology, size and orientation of the shape, but now contains just the essence of the shape

Distance transforms

- The distance transform shows the distance from each point in an image to its central axis
  - Distance can be measured in coordinate values or Euclidean distance
- The distance transform can be achieved by successive erosion
- The pixels at the border of a shape will have a distance transform of 1, those adjacent will have a value of 2 etc.

Distance transforms

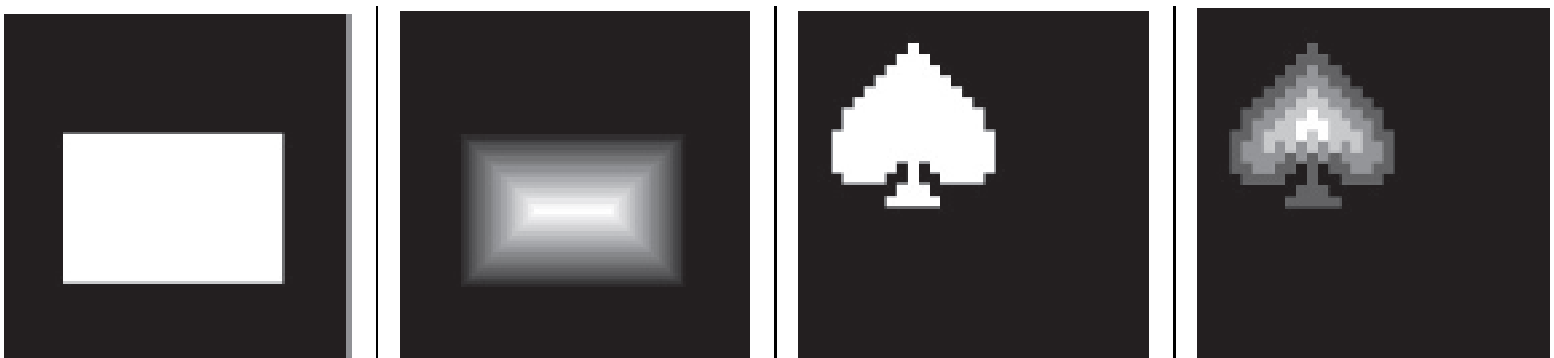


(a) Initial shape (b) Distance transform

- Here the central axis has a value of three as it takes that number of erosions to reach it from either side

Distance transforms

- When applied to images with a higher resolution:



(a) Rectangle (b) Distance transform (c) Card suit (d) Distance transform

Flexible shape models

- So far the approach has been a match to image data
  - Usually a match between a model and an image
- A completely different approach is to use a database that contains all possible variations of a shape
- The database can then form a model to match to the most likely version of a shape
- This deformable approach is guided by the statistics of the likely variation of the shape

Active shape models (ASMs)

- A new approach
- A model of a shape is made up of points
  - The variation in these points are called the point distribution model
- The chosen landmark points are labeled on the training images
  - The set of training images aim to capture all possible variations of the shape
- Each point describes a particular point on the boundary

ASMs

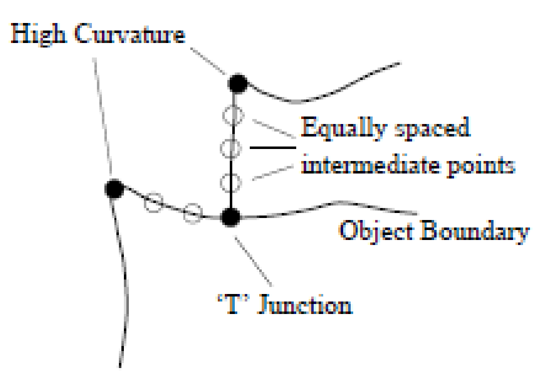
- Example applications include finding a human face in an image
- We have a known set of shapes and fixed interrelationships, but the detail can change
  - Shapes combined with distributions
- There is a lot of data concerned
  - If we choose lots of points and we have lots of training images we have an enormous amount of points

ASMs

- In order to locate structure of interest, we must first build a model of it – require a set of annotated images of typical examples
  – Model is trained from a set of images annotated by a human expert
- By analysing variations in shape and appearance over the training set, a model is built which can mimic this variation
- To interpret a new image we must find the parameters which best match a model instance to the image.

ASMs

- Must first decide on a set of suitable landmarks which describe the shape of the target and which can be found reliably on every training image
  - Good choices for landmark points are points at clear corners of object boundaries, T-junctions between boundaries or easily located biological landmarks – to make sure there are enough landmarks points we add points along boundaries where are arranged to be equally spaced



ASMs

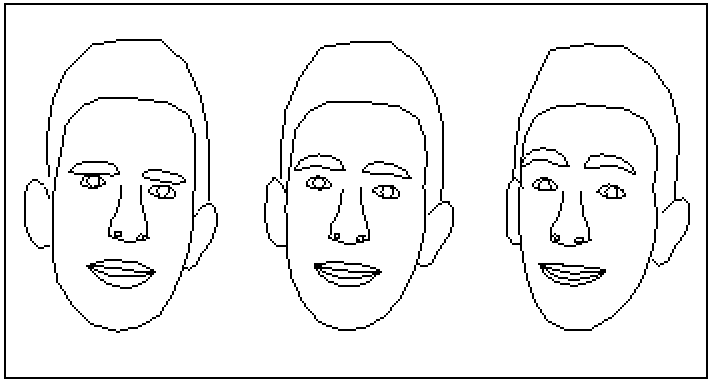
- To represent the shape we also record the connectivity defining how the landmarks are joined to form the boundaries of the image
  - This allows us to determine the direction of the boundary at a given point
- Suppose the landmarks along a curve are labelled $\{(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)\}$
- For a 2-D image we can represent the n landmarks as a 2n element vector x where

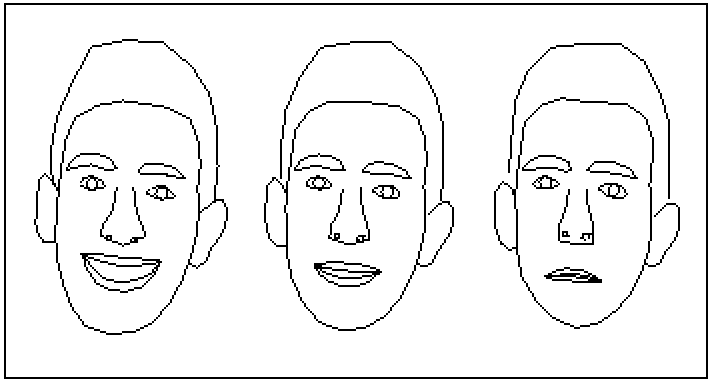
$$x = (x_1, \ldots x_n, y_1, \ldots y_n)^T$$

- If we have s training examples, we generate s such vectors

ASMs

- Effects of variance is then included in the model
- Effect of varying two shape parameters out of 169 point face model within 3 standard deviations from the mean





ASMs

- To find the best fit of an instance of such a model to a new image, we must find the shape parameters, position, orientation and scale which best match the model to the image
- Involves an iterative approach to bring about increasing match between the points in the model and the image
- Regions around model points are examined to determine the best nearby match
- Models only change to better fit the data and are controlled by the expected appearance of the shape

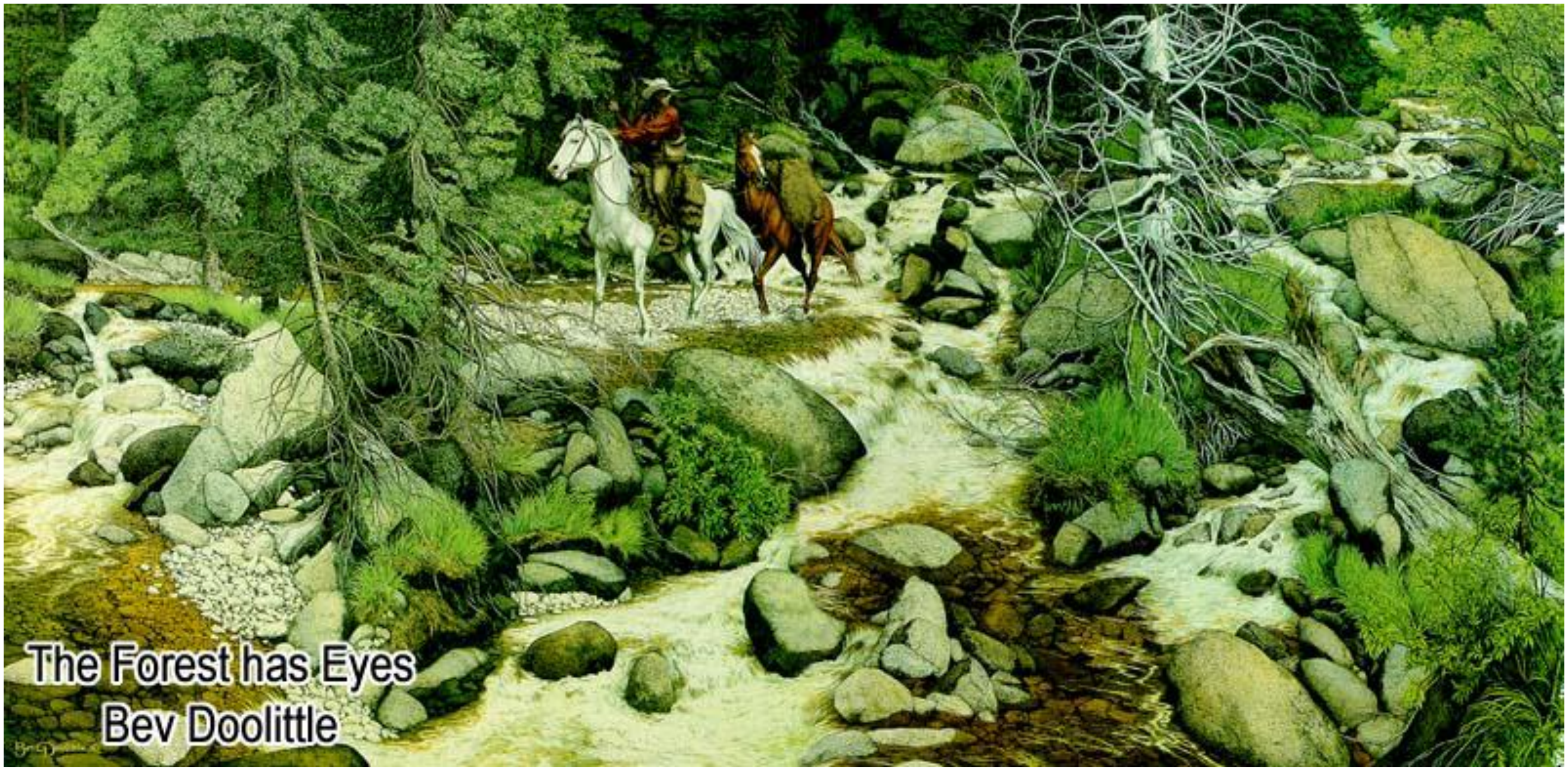
ASMs

(a) Initialization

(b) After five iterations

(c) At convergence, the final shapes

Best laid plans…




The Forest has Eyes
Bev Doolittle