Learning deformable shape models from images
Goal: localize boundaries of new class instances

Training data

Test image

Training: bounding-boxes

Testing: object boundaries

[Ferrari, Jurie, Schmid, IJCV09]
Learn a shape model from training images

Training data

prototype shape

deformation model
Match it to the test image
Challenges for learning

**Main issue**
Which edgels belong to the class boundaries?

**Complications**
- intra-class variability
- missing edgels
- produce point correspondences (learn deformations)
Challenges for detection

- scale changes
- intra-class variability
- clutter
- fragmented and incomplete contours
Local contour features

**PAS**
Pair of Adjacent Segments

+ *robust*
  connect also across gaps

+ *clean*
  descriptor encodes the two segments *only*

+ *invariant*
  to translation and scale

+ *intermediate complexity*
  good compromise between repeatability and informativity
Local contour features

**PAS**
Pair of Adjacent Segments

two PAS in correspondence

→ translation+scale transform

→ use in Hough-like schemes

Clustering descriptors

→ codebook of *PAS* types
(here from mug bounding boxes)
Learning: overview
Learning: finding model parts

**Intuition**

PAS on class boundaries reoccur at similar locations/scales/shapes

Background and details specific to individual examples don’t
Learning: finding model parts

Algorithm
1. align bounding-boxes up to translation/scale/aspect-ratio
2. create a separate voting space per PAS type
3. soft-assign PAS to types
4. PAS cast ‘existence’ votes in corresponding spaces
Learning: finding model parts

**Algorithm**
1. align bounding-boxes up to translation/scale/aspect-ratio
2. create a separate voting space per PAS type
3. soft-assign PAS to types
4. PAS cast ‘existence’ votes in corresponding spaces
5. local maxima → model parts
Learning: finding model parts

*Model parts*

- location + size (wrt canonical BB)
- shape (PAS type)
- strength (value of local maximum)
Learning: finding model parts

Why does it work?
Unlikely unrelated PAS have similar location and size and shape

→ form no peaks!

Important properties
+ see all training data at once

→ robust

+ linear complexity

→ efficient large-scale learning
Learning: assembling an initial shape

Cool, but not a shape yet
- multiple strokes
- adjacent parts don’t fit together

Why?
- parts are learnt independently

Let’s try to assemble parts into a proper whole

We want single-stroked, long continuous lines!
Learning: shape refinement

Idea

treat shape as deformable point set
and _match it back_ onto training images

How?

- robust non-rigid point matcher: TPS-RPM
  (thin plat spline – robust point matching)
- strong initialization:
  align model shape BB over training BB
  → likely to succeed

Chui and Rangarajan, *A new point matching algorithm for non-rigid registration*, CVIU 2003
Learning: shape refinement

Shape refinement algorithm

1. Match current model shape back to every training image
   
   backmatched shapes are in full point-to-point correspondence!

2. Set model to mean shape

3. Remove redundant points

4. If changed → iterate to 1
Learning: shape refinement

*Final model shape*

- clean (almost only class boundaries)
- smooth, connected lines
- generic-looking
- fine-scale structures recovered (handle arcs)
- accurate point correspondences spanning training images
Learning: shape deformations

From backmatching
intra-class variation examples,
in complete correspondence

\[ \text{Apply Cootes' technique} \]
1. shapes = vectors in 2p-D space
2. apply PCA

Deformation model
\[ \text{top } n \text{ eigenvectors covering } 95\% \text{ of variance} \]
\[ \text{associated eigenvalues } \lambda_i \text{ (act as bounds)} \]

\[ \rightarrow \text{valid region of shape space} \]

Tim Cootes, *An introduction to Active Shape Models*, 2000
Learning completed!

Automatic learning of shapes, correspondences, and deformations from unsegmented images
Object detection: overview

**Goal**

given a test image, localize class instances down to their boundaries

**How?**

1. Hough voting over PAS matches → rough location+scale estimates

2. use to initialize TPS-RPM

   *combination enables true pointwise shape matching to cluttered images*

3. constrain TPS-RPM by learnt deformation model → better accuracy
Object detection: Hough voting

Algorithm

1. soft-match model parts to test PAS

2. each match
   → translation + scale change
   → vote in accumulator space

3. local maxima
   → rough estimates of object candidates

Leibe and Schiele, DAGM 2004; Shotton et al, ICCV 2005; Opelt et al. ECCV 2006
Object detection: Hough voting

**Algorithm**

1. soft-match model parts to test PAS
2. each match
   - translation + scale change
   - vote in accumulator space
3. local maxima
   - rough estimates of object candidates

initializations for shape matching!

Leibe and Schiele, DAGM 2004; Shotton et al, ICCV 2005; Opelt et al. ECCV 2006
Object detection: shape matching by TPS-RPM

**Initialize**
get point sets V and X

**Goal**
find correspondences M and TPS mapping
M = (|X|+1)x(|V|+1) soft-assign matrix

**Algorithm**

1. Update M based on
   \[ \text{dist}(\text{TPS},X) + \text{orient}(\text{TPS},X) + \text{strength}(X) \]

2. Update TPS:
   - Y = MX
   - fit regularized TPS to V

Deterministic annealing:
iterate with T decreasing
   → M less fuzzy (looks closer)
   → TPS more deformable

Chui and Rangarajan, *A new point matching algorithm for non-rigid registration*, CVIU 2003
TPS-RPM in action!

Original V and X

Transformed V + X

Transformed V + X

TPS Warping

Estimated Shape Y=MX
Object detection: constrained TPS-RPM

*Output of TPS-RPM*

nice, but sometimes inaccurate or even not mug-like

*Why?*

generic TPS deformation model (prefers smoother transforms)

*Constrained shape matching*

constrain TPS-RPM by learnt *class-specific* deformation model

+ only shapes similar to class members
+ improve detection accuracy
Object detection: constrained TPS-RPM

General idea
constrain optimization to explore only region of shape space spanned by training examples

How to modify TPS-RPM?
1. Update M
2. Update TPS:
   - $Y = MX$
   - $Y \leftarrow Y^c$
   - fit regularized TPS to $V \leftarrow Y$

hard constraint, sometimes too restrictive
Object detection: constrained TPS-RPM

**General idea**

constrain optimization to explore only region of shape space spanned by training examples

**Soft constraint variant**

1. Update M
2. Update TPS:
   - $Y = MX$
   - $Y \leftarrow Y + \frac{\mathbf{T}}{\mathbf{T}_{\text{dist}}}(Y' - Y)$
   - fit regularized TPS to $V \leftarrow Y$

*soft constraint, Y is attracted by the valid region*
Soft constrained TPS-RPM in action!
Object detection: constrained TPS-RPM

Soft constrained TPS-RPM

+ shapes fit data more accurately
+ shapes resemble class members
+ in spirit of deterministic annealing!
+ truly alters the search
  (not fix a posteriori)

Does it really make a difference?
when it does, it’s really noticeable
(about 1 in 4 cases)
Datasets: ETHZ Shape Classes

• 255 images from *Google-images*, and *Flickr*
  - uncontrolled conditions
  - variety: indoor, outdoor, natural, man-made, …
  - wide range of scales (factor 4 for swans, factor 6 for apple-logos )

• all parameters are kept fixed for all experiments
• training images: 5x random half of positive; test images: *all* non-train
Datasets: INRIA Horses

• 170 horse images + 170 non-horse ones
  - clutter, scale changes, various poses

• all parameters are kept fixed for all experiments
• training images: 5x random 50; test images: all non-train images
Results: all learned models
Results: all learned models
Results: all learned models
Results: apple logos
Results: mugs
Results: giraffes
Results: bottles
Results: swans
Results: horses
Results: detection-rate vs false-positives per image

- **Swans**
  - full system (>20% intersection)
  - full system (PASCAL:  \( \cap/ \cup >50\% \))
  - Hough alone (PASCAL)
  - accuracy: 3.0

- **Bottles**
  - accuracy: 2.4

- **Apple logos**
  - accuracy: 1.5

- **Mugs**
  - accuracy: 3.1

- **Giraffes**
  - accuracy: 3.5

- **INRIA Horses**
  - accuracy: 5.4
Results: Hand-drawings

Same protocol as Ferrari et al, ECCV 2006: match each hand-drawing to all 255 test images
Results: detection-rate vs false-positives per image

- Our approach
- Ferrari, ECCV06
- Chamfer (with orientation planes)
- Chamfer (no orientation planes)
Conclusions

1. learning shape models from images

2. matching them to new cluttered images

+ detect object boundaries while needing only BBs for training
+ effective also with hand-drawings as models
+ deals with extensive clutter, shape variability, and large scale changes

- can’t learn highly deformable classes (e.g. jellyfish)
- model quality drops with very high training clutter/fragmentation (giraffes)