Tutorial on Pictorial Structures for Matching and Recognition

Dan Huttenlocher

ICVSS
July 2009

Object (Category) Recognition

- Active research area, largely focused on "categories" rather than specific objects
  - E.g., bicycle

  - Question of what constitutes a category, not issue for specific objects
  - Most of history of object recognition is specific objects, 1960-2000+
- Increasing use of shared datasets – e.g., Caltech, PASCAL (10K images, 10 categories)

Classification vs. Localization

- Classification: presence or absence of an object category in an image
- Localization (detection): where objects and potentially subparts are in an image
- Image retrieval such as Web search often requires only classification
  - E.g., searching for photos of motorcycles
- Interpreting and interacting with the world generally requires localizing
  - Visual user interfaces, monitoring, navigation

Localizing Often Difficult

- Classification implicitly assumes that object is major part of image
  - E.g., classifying "Where's Waldo" images
- Detection specifies much more information
  - Many ways to be wrong: millions of possible locations vs. one presence/absence decision

Category Recognition Research

- Research largely focused on classification rather than detection
  - Both methods and evaluation criteria
    - Recent PASCAL challenge an exception, but still comparatively few entries for localization task
- Broad range of learning techniques readily applicable to classification
  - Detection not only a harder problem also fewer techniques to directly apply
- Yet localizing important for most applications other than retrieval

Category Recognition Approaches

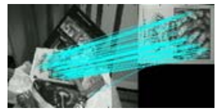
- Bag of features ("visual words") models
  - Capture little or no spatial information, less well suited to localization
- Pixel-based techniques – combination of segmentation and recognition
  - Little in way of explicit object representation
- Part-based statistical models
  - Data term and spatial prior
    - Constellation, pictorial structures (tree), kfans
      - Markov random field (MRF)
  - Best current localization techniques

Recognition Cues

- Appearance
  - Patterns of intensity or color, e.g., tiger fur
  - Generally measured locally over region
- Geometry
  - Spatial configuration of parts or local features
    - E.g., face has eyes above nose above mouth
- Early era relied on geometry (1960-80), later on appearance (1985-95), more recently both

Role of Sparse Features

- Recognition often viewed as bottom-up process
  - First detect a relatively small number of distinctive local features (e.g., SIFT [Lowe04])
  - Then match detected features to pre-existing model



Specific Object

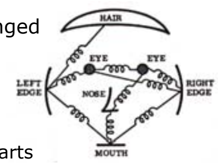
Object Category

Problems With Detecting Features

- Local decisions about presence or absence of features are difficult and error prone
  - E.g., often hard to determine whether a corner is present without more context

Recognition Without Feature Detection

- Pictorial structures [FE73]
  - Model consists of parts arranged in deformable configuration
    - Match cost function for each part (at all locations)
    - Deformation cost function for each connected pair of parts



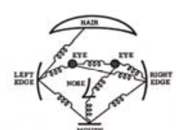
- Intuitively natural notion of parts connected by springs
  - “Wiggle until fits”, no individual feature detection
  - Recent work making computationally tractable

Modern Pictorial Structures

- Developed efficient algorithms for certain types of pictorial structure models
  - Gaussian spatial models with tree- or fan-like underlying graph structures [FH00,FH05,CFH05]
    - Dynamic programming techniques
- Addressed learning models [CH06]
  - Using weak supervision, where training data specifies coarse bounding box but not parts
- Spatial models learned using SVM and HOG templates for parts [FMR08]

Formal Definition of Pictorial Structure

- Object modeled by graph, M=(V,E)
  - Parts $V=(v_1, \ldots, v_n)$
  - Spatial relations $E=\{e_{ij}\}$
    - Gaussian on relative locations for pair of parts i,j
- Spatial prior $P_M(L)$ on configurations of parts $L=(\ell_1, \ldots, \ell_n)$
  - Where ℓ_i over discrete configuration space
    - E.g., translation, rotation, scale



7 nodes
9 edges
(out of 21)

Object Detection

- Given image I and model M
  - Prior $P_M(L)$ distribution of spatial configurations
  - Likelihood $P_M(I|L)=\prod P_M(I|\ell_i)$ of image given conf.
- Evidence over all configs L, marginalization

$$P_M(I) = \sum_L P_M(I|L)\, P_M(L) \propto \sum_L P_M(L|I)$$

- Or quality of best configuration (MAP est.)

$$\max_L P_M(I|L)\, P_M(L)$$

- Or sample from distribution $P_M(I|L)\, P_M(L)$
- Latter two localize parts, maximizer L^*

Fast Methods

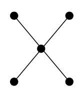
- Spatial term based on relative location of pairs, allows convolution-like operations

$$P_M(\ell_i,\ell_j) \propto \rho(\ell_i-\ell_j)$$

- Best match (MAP estimate) [FH00, FH05]
  - Linear time methods for min convolution yield O(mn) time, generalized distance transforms
- Marginalization over configs [FH05]
  - Can apply FFT yielding O(mnlogm) time
  - For Gaussian, binomial filters yield O(mn) time
  - Fast sampling of good candidate matches

Graphical Model View

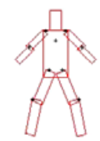
- Probabilistic model
  - Collection of random variables with explicit dependencies between certain pairs
- Graph structure
  - Reachability corresponds to conditional independence



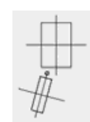
- Undirected edges – correlation rather than causality
  - Markov random field (MRF)
  - Distribution factors according to cliques

Tree Structured Models

- Kinematic structure of animate objects
  - Skeleton forms tree
  - Parts as nodes, joints as edges



- 2D image of joint
  - Spatial configuration for pair of parts
  - Relative orientation, position and scale (foreshortening) – x,y,s,o



Tree Factorization

- Spatial prior factors

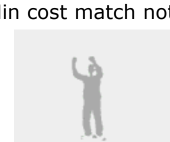
$$P_M(L) = \frac{\prod_{e_{ij}\in E} P_M(\ell_i,\ell_j)}{\prod_{v_i\in V} P_M(\ell_i)^{\deg(v_i)-1}}$$

- Denominator constant
  - No prior on absolute location

- Dynamic programming methods
  - Viterbi or Baum-Welch algorithms
  - Not very practical, $O(h^2n)$
    - Quadratic in number of locations per part, h

Best Match (MAP Estimate)

- All possible spatial configurations "considered" – most eliminated implicitly
  - Dynamic programming for min convolution
- Example using simple binary silhouette for appearance
  - Min cost match not always "best"

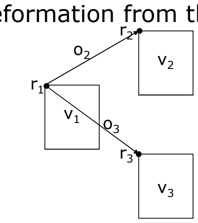


Efficient Algorithm for Central Part

- Location $L=(l_1, \ldots, l_n)$ specifies where each part positioned in image
- Best location $\min_L (\Sigma_i \, m_i(l_i) + d_i(l_i,l_1))$
  - Part cost $m_i(l_i)$ – negative log of likelihood
    - Measures degree of mismatch of appearance a_i when part v_i placed at each of h locations, l_i
  - Deformation cost $d_i(l_i,l_1)$ – negative log of prior
    - Spring cost c_{i1} of part v_i measured with respect to central part v_1
    - E.g., quadratic or truncated quadratic function
    - Note deformation cost zero for part v_1 (wrt self)

Central Part Model

- Spring cost c_{ij}: i=1, ideal location of l_j wrt l_1
  - Translation $o_j=r_j-r_1$
  - $T_j(x)=x+o_j$
- Spring cost deformation from this ideal
  - $\| l_j-T_j(l_1) \|^2$



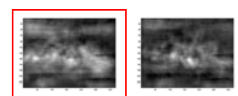
Consider Case of 2 Parts

- $\min_{l_1,l_2} (m_1(l_1) + m_2(l_2)+\| l_2-T_2(l_1) \|^2)$
  - Where $T_2(l_1)$ transforms l_1 to ideal location with respect to l_2 (offset)
- $\min_{l_1} (m_1(l_1) + \min_{l_2} (m_2(l_2)+\| l_2-T_2(l_1) \|^2))$
  - Second term is a generalized distance transform!
- $\min_{l_1} (m_1(l_1) + DT_{m_2}(T_2(l_1))$
- Sequential rather than simultaneous min
  - Don't need to consider each pair of positions for the two parts because a distance
    - Just distance transform the match cost function, m

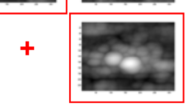
Overall Computation for 2 Parts

- Image and model (translation)

- Match cost of each part $m_1(l_1)$, $m_2(l_2)$



- Distance transform of $m_2(l_2)$



- $\min_{l_1}(m_1(l_1) + DT_{m_2}(T_2(l_1))$

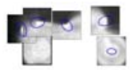
Star Graph – Central Reference Part

- $\min_L (\Sigma_i (m_i(l_i) + d_i(l_i,l_1)))$
- $\min_L (\Sigma_i \, m_i(l_i) + \| l_i - T_i(l_1) \|^2)$
  - Quadratic distance between location of part v_i and ideal location given location of central part
- $\min_{l_1} (m_1(l_1) +$ $\Sigma_{i>1} \min_{l_i} (m_i(l_i)+\| l_i-T_i(l_1) \|^2))$
  - i-th term of sum minimizes only over l_i
- $\min_{l_1} (m_1(l_1) + \Sigma_{i>1} DT_{m_i}(T_i(l_1)))$
  - Where $DT_f(x) = \min_y (f(y) + \| y-x \|^2)$

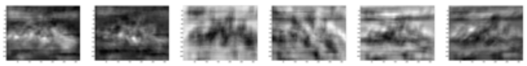
Star Graph

- Simple overall computation
  - Match cost $m_i(l_i)$ for each part at each location
  - Distance transform of $m_i(l_i)$ for each part other than reference part
    - Shifted by ideal relative location $T_i(l_1)$ for that part
  - Sum the match cost for the first part with the distance transforms for the other parts
  - Find location with minimum value in this sum array (best match)
- DT allows for flexibility in part locations

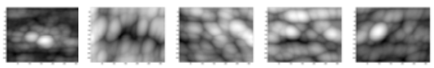
Overall Computation for Star Graph

- Part costs, O(h) time each, total O(hn)



- Distance transform non-reference part costs, sum to get MAP location, O(hn) time



More General Flexible Templates

- Efficient computation using distance transforms for any tree-structured model
  - Not limited to central reference part – star
- Two differences from reference part case
  - Relate positions of parts to one another using tree-structured recursion
    - Solve with Viterbi (or forward-backward for marginals) algorithm
  - Parameterization of distance transform more involved – transformation T_{ij} for each connected pair of parts

Minimizing Over Tree Structures

- Use dynamic programming to minimize $\Sigma_V m_j(l_j) + \Sigma_E d_{ij}(l_i,l_j)$
- Can express as function for pairs $B_j(l_i)$
  - Cost of best location of v_j given location l_i of v_i
- Recursive formulas in terms of children C_j of v_j
  - $B_j(l_i) = \min_{l_j} (m_j(l_j) + d_{ij}(l_i,l_j) + \Sigma_{C_j} B_c(l_j))$
  - For leaf node no children, so last term empty
  - For root node no parent, so second term omitted

Efficient Algorithm for Trees

- MAP estimation algorithm
  - Tree structure allows use of Viterbi style dynamic programming
    - $O(nh^2)$ rather than $O(h^n)$ for h locations, n parts
    - Still slow to be useful in practice (h in millions)
  - Couple with distance transform method for finding best pair-wise locations in linear time
    - Resulting O(nh) method
- Similar techniques allow sampling from posterior distribution in O(nh) time
  - Using forward-backward algorithm

O(nh) Algorithm for MAP Estimate

- Express $B_j(l_i)$ in recursive minimization formulas as $DT_f(T_{ij}(l_i))$
  - Cost function
    - $f(y) = m_j(T_{ji}^{-1}(y)) + \Sigma_{C_j} B_c(T_{ji}^{-1}(y))$
  - T_{ij} maps locations to space where difference between l_i and l_j is a squared distance
    - Distance zero at ideal relative locations
- Yields n recursive equations, one per part
  - Each can be computed in O(h) time
    - Where h number of parameter values (fixed dimensional parameter space)

Sampling the Posterior

- Generate good possible matches as hypotheses
  - Locations where posterior P(L|I) high
  - Validate using another technique
    - Here use a correlation-like measure (Chamfer)
- Computation similar to MAP estimation
  - Recursive equations, one per part
  - Ability to solve each equation in linear time
    - Linear time dynamic programming approximation to Gaussian using box filters
  - Fast running times (seconds)

Sampling Approach

- Marginal distribution for location l_r of (arbitrarily chosen) root part

  $p(l_r|I,\Theta) = \sum_{L\backslash lr} (\prod_V p(I|l_i,a_i) \prod_E p(l_i,l_j|c_{ij}))$
- Can be computed efficiently due to tree structured dependencies

  $p(l_r|I,\Theta) \propto p(I|l_r,a_r) \prod_{Ch} s_c(l_r)$
  - And fast convolution when $p(l_i,l_j|c_{ij})$ Gaussian

  $s_j(l_i) \propto \sum_{lj} (p(I|l_j,a_j)\, p(l_i,l_j|c_{ij}) \prod_{Ch} s_c(l_j))$
- Sample location for root from marginal
  - Sample from root to leaves using $p(l_j|l_i,I,\Theta)$

Samples From Posterior

Pictorial Structure as Proposal Distribution

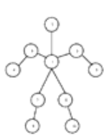
- Computationally simpler distribution
  - E.g., POP model, [AT07]
- Can use to address limitations of models
  - Non-Gaussian pairwise constraints
  - Non-independence of part appearances
- Use model that factors to propose high probability answers according to a simpler model
- Maximize a less tractable criterion only for those sample configurations

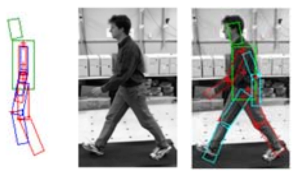
More Spatial Structure in Model

- While preserving computational tractability
- Adding latent spatial variable(s) to models
  - Correspond to overall model parameters rather than parts
  - Need to ensure no large cliques in resulting graph as computation increases exponentially
- K-fans
  - Generalization of star graph to root set of size k rather than single root node
  - Depth one graph of low tree width

Spatial Models for Human Pose

- Widespread use of kinematic tree models
  - Encode relationships between rigid parts connected by joints (2D and 3D)
  - Enables efficient exact inference/global optimization of pose given model and data





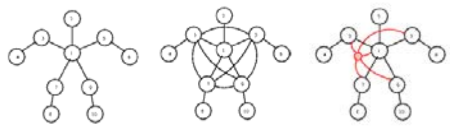
Limitations of Kinematic Trees

- Only represent relationships between connected parts (note still good proposals!)
- Coordination between limbs not encoded
  - Critical for balance and many activities

Equally good under tree model

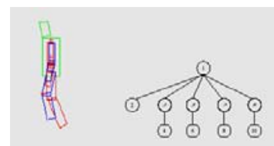
Non-Tree Models

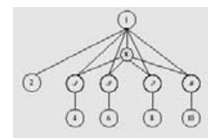
- Larger cliques, latent variables
  - Introduce additional variable corresponding to common factor of limb coordination
    - Does not correspond to any part
    - Dependencies among orientation parameters
  - Still relatively efficient inference for small clique



A Latent Gait Variable for Humans

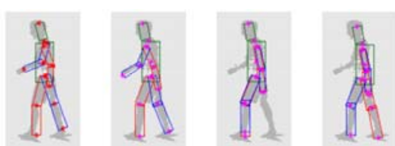
- Introduce additional variable corresponding to common factor [LH05]
  - Consistency between limb positions, not captured by kinematic (skeletal) model
    - Rather than directly connecting limbs which creates large clique





Example Using Brown MOCAP Data

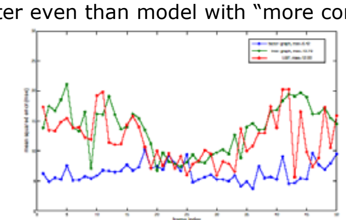
- MAP estimate of best pose, single frame
  - Loopy models, but with small cliques



Ground Truth | Latent Variable Model | Tree Model | Larger Clique Using LBP (Pairwise)

Latent Gait Variable Helps

- Comparison using ground truth (MOCAP)
  - Latent gait variable model, tree structured model, model with large clique (loopy graph)
  - Better even than model with "more constraint"

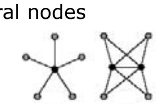


K-fan Models

- Prior factors according to graph of spatial constraints between parts

  $$P_M(L) = \prod_C \Psi_C(L_C)$$

  - Product over maximal cliques of triangulated graph, L_C locations of corresponding parts
- K-fan generalizes star graph structure
  - Cliques of size k+1 for k central nodes
  - Exact discrete inference in $O(nm^k)$ time for n parts and m locations per part, using fast convolution methods



1-fan 2-fan

Spatial Prior for k-Fan

- Let $R \subseteq V$ be set of reference parts, "center"

  $$P_M(L) = P_M(L_R) \prod_{v_i \in R'} P_M(\ell_i | L_R)$$

  - Where L_R vector of locations for R $L_R = (\ell_1, \ldots, \ell_k)$ for $R = (v_1, \ldots, v_k)$
- Makes explicit that part locations are independent conditioned on reference set
  - Product over non-reference parts, R'
- Geometric interpretation in terms of parts defining "reference frame"

Edge-Based Part Models

- Assume likelihood factors
  - Foreground product over parts
  - Background product over pixels

    $P_M(I|L) = \prod_i g_i(I,l_i) \prod_p b_p(I)$
- Foreground model simple edge template
  - Probability of an edge at each pixel
  - Use vector of probabilities for four possible orientations
  - Slight dilation to account for discretization

Overall Estimation Approach

- Overall estimation more accurate (and faster) than feature detection
  - E.g., optimization approach [CFH05,FPZ05] for star or 2-fan vs. feature detection for full joint Gaussian [FPZ03]
  - 6 parts under translation, Caltech-4 dataset
  - Single class, equal ROC error

	Airplane	Motorbike	Faces	Cars
Feat. Det. [FPZ03]	90.2%	92.5%	96.4%	90.3%
Est.-Star [FPZ05]	93.6%	97.3%	90.3%	87.7%
Est.-Fan [CFH05]	93.3%	97.0%	98.2%	92.2%

Learning the Models

- [FPZ05] uses feature detection to learn models under weakly supervised regime
  - Know only which training images contain instances of the class, no location information
- [CFH05] does not use feature detection but requires extensive supervision
  - Know locations of all the parts in all the positive training images
- [CH06] weak supervision without relying on feature detection – no part locations

Weakly Supervised Learning

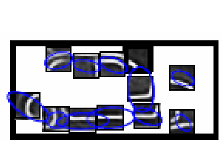
- Consider large number of initial patch models to generate possible parts
- Generate all pairwise models formed by two initial patches
- Consider all sets of reference parts for fixed k
- Greedily add parts based on pairwise models to produce initial model
- EM to iteratively improve model

Model Improvement

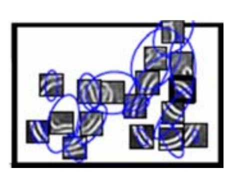
- Use EM to update model
  - Increase likelihood of training data by iteratively improving both appearance and spatial models
- POP – patchwork of parts [AT07]
  - More accurate model that accounts for overlapping parts
  - Average probabilities of patches that overlap
    - Distribution does not factor, can't compute efficiently
    - Sample from factored distribution and maximize POP criterion

Example Learned Models

- Star graph (one fan)
  - Reference part in bold box
  - Blue ellipse 2σ level set of Gaussian



Side View of Car



Side View of Bicycle

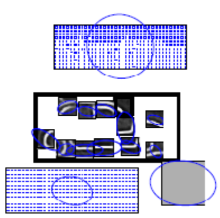
Adding Local Context to Models

- Spatial relations not only among parts of object but also object and background
  - E.g., vehicles on roads, often in front of buildings
  - Less predictable relative locations than object parts within a category
- Use coarser appearance models
  - Less predictable appearance of "scene parts"
- Augment spatial model using two-level hierarchy

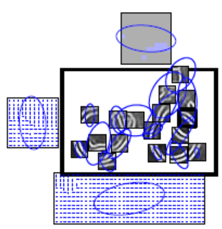
Contextual Model

- Learn part-based object category model as before
- Also learn spatial relationship between object bounding box and parts of scene
- Parts modeled using quantized colors and surface orientation
- Posterior that factors according to object and parts scene context "parts"

Example Learned Models



Side View of Car



Side View of Bicycle

Recognition Results

- Four categories from PASCAL 06 VOC
  - Manmade objects: bicycle, bus, car, motorbike
  - Detection task (localize object)
    - Standard measure used in VOC, overlap of detected location with ground truth > 50%
    - Average precision
- Training with weak supervision
  - Use object bounding box
    - For scene model
    - To separate multiple instances in images

Comparison of Results

- Scene information substantially increases accuracy
- Better accuracy – average precision – than entries in VOC challenge
  - One method rather than several different methods

Object class	Obj. model only	Scene + obj. model	Best VOC result
Bicycle	0.421	0.498	0.440
Bus	0.172	0.185	0.169
Car	0.429	0.458	0.444
Motorbike	0.342	0.388	0.390

Example Results

Summary of Basic Model

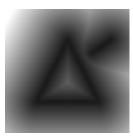
- Pictorial structures: detection without doing feature detection
  - For common object class datasets, faster and more accurate than spatial models using feature detection
- Role of spatial structure
  - Latent structural variable such as human "gait" can substantially improve localization
- Role of local context
  - Including scene parts in model can substantially improve localization

Distance Transform

- Map of distances from any point to nearest point of some type
  - Distances to object boundaries in computer graphics, robotics and AI
  - Distances to image features in computer vision
- Generally used for data on grid
  - Pixels or voxels, 2D or 3D
  - Related to exact algorithms for Voronoi diagrams
- Efficient algorithms for computing
  - Linear in number of pixels, fast in practice

Uses of Distance Transforms

- Proximity-based matching: Chamfer and Hausdorff distances
  - For each point of set A nearest point of set B
  - But not correspondence or one-to-one matching
  - Related to morphological dilation

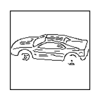


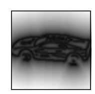
- Path planning and obstacle avoidance
  - Maximal clearance path
  - Re-compute if moving obstacles
    - But bound on how fast changes

Distance Transform Formula

- Set of points, P, and measure of distance

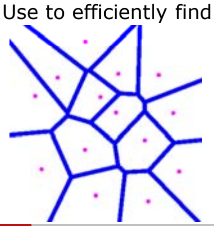
  $DT(P)[x] = \min_{y \in P} dist(x,y)$
- For each location x distance to nearest point y in P
  - Can think of "cones" rooted at each $y \in P$
  - Min over all the cones (lower envelope)

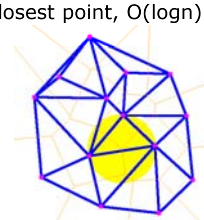




Relation to Voronoi Diagram

- Locus equidistant from two or more points
  - Dual of Delaunay triangulation
  - Compute in O(nlogn) time (Graham scan)
  - Use to efficiently find closest point, O(logn)

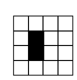




Grid Formulation of Distance Trans.

- Commonly computed on a grid Γ, for set of points $P \subseteq \Gamma$

  $DT(P)[x] = \min_{x' \in \Gamma} (dist(x,x') + 1_P(x'))$
- Where $1_P(x')$ indicator function for P
  - Value of 0 when $x' \in P$, ∞ otherwise
  - Can think of cone rooted at each grid point where indicator is 0
  - Cones for grid cells not corresponding to points in set P are infinite, don't contribute to min



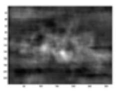
2	1	2	3
1	0	1	2
1	0	1	2
2	1	2	3

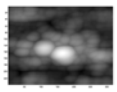
Generalized Distance Transform

- No need to limit only to indicator function with cost of zero or infinity
  - Applies to any cost map

$DT(P)[x] = \min_{x' \in \Gamma} (dist(x,x') + cost(x'))$

  - Can think of cone rooted at each point of grid rather than just where indicator 0 (binary grid)
  - Height of root of each cone corresponding cost at that location – min over cones





Naïve Computation

- For each point on the grid, explicitly consider each point and minimize
  - $O(n^2)$ method
- Combinatorial methods using explicit point set P not much better and also don't generalize
- Not very practical even for moderate size grids such as images
  - Even a low-resolution video frame has about 300K pixels
    - About 100 billion distance computations

Faster Methods on Grid

- 1D case, L_1 norm: $|x_1 - x'_1| + |x_2 - x'_2|$
  - Two passes:
    - Find closest point on left
    - Find closest on right if closer than one on left
  - Incremental:
    - Moving left-to-right, closest point on left either previous closest point or current point
    - Analogous for moving right-to-left
  - Can keep track of closest point as well as distance to it
    - Will illustrate distance only, less book-keeping

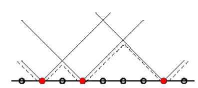
L_1 Distance Transform Algorithm

- Two pass O(n) algorithm for 1D L_1 norm (just distance and not source point)
  1. Initialize: For all j
     $D[j] \leftarrow 1_P[j]$
  2. Forward: For j from 1 up to n-1
     $D[j] \leftarrow \min(D[j],D[j-1]+1)$

1	0

  3. Backward: For j from n-2 down to 0
     $D[j] \leftarrow \min(D[j],D[j+1]+1)$

0	1

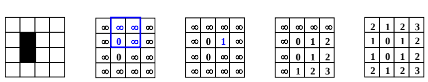


∞	0	∞	0	∞	∞	∞	0	∞
∞	0	1	0	1	2	3	0	1
1	0	1	0	1	2	1	0	1

L_1 Distance Transform

- 2D case analogous to 1D
  - Initialization
  - Forward and backward pass
    - Forward pass adds one to closest above and to left, takes min with self
    - Backward pass analogous below and to right

0	1
1	s



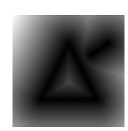
s	1
1	0

L_2 Distance Transform

- What about Euclidean distance $\text{sqrt}((x_1 - x'_1)^2 + (x_2 - x'_2)^2)$?
- Not linear function of location on grid
  - Simple local propagation methods not correct

$\sqrt{2}$	1
1	s

- Local propagation just approximation
  - Introduces considerable error, particularly at larger distances
  - Bigger neighborhood can help but not fix



Exact L_2 Distance Transform

- 1D case doesn't seem helpful
  - Same as L_1
  - But just saw 2D case not same as L_1
- Several quite complex methods
  - Linear or O(nlogn) time, but at edge of practical
- Revisit 1D
  - Decompose 2D into two 1D transforms
  - Yield relatively simple method, though not local
  - Requires more advanced way of understanding running time – amortized analysis

Squared Distance on 2D Grid

- Consider f(x,y) on grid
  - For instance, indicator function for membership in point set P, 0 or ∞
- Distance transform

  $D_f(x,y) = \min_{x',y'}((x-x')^2 + (y-y')^2 + f(x',y'))$
- First term does not depend on y'

  $= \min_{x'}((x-x')^2 + \min_{y'}((y-y')^2 + f(x',y')))$
- But then can view as 1D distance transform restricted to column indexed by x'

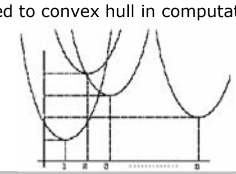
  $= \min_{x'}((x-x')^2 + D_{f|x'}(y))$

Approach for L_2 Distance Transform

- Start with point set on grid
- Initialize to $0,\infty$ cost function
- Perform 1D transform on columns of cost function
- Perform 1D transform on rows of <u>result</u>
  - Cascade results in each dimension
- Compute square roots if actual distance needed
  - Note, as does not change minima, often more efficient to leave as squared distance

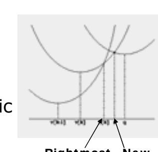
Computing 1D L_2^2 Transform Efficiently

- Compute $h(x)=\min_{x'} ((x-x')^2+f(x'))$
- Intuition: each value defines a constraint
  - Geometric view: in one dimension, <u>lower envelope</u> of arrangement of n quadratics
    - Each rooted at (x,f(x))
      - Related to convex hull in computational geometry



Algorithm for 1D Lower Envelope

- Incrementally add quadratics
  - Keep only those on lower envelope
    - Maintain ordered list of visible quadratics and the intersections of successive ones
- Consider in left-to-right order
  - Compare new intersection with rightmost quadratic to rightmost existing intersection
    - If to left, hides rightmost quadratic so remove and repeat



Running Time of LE Algorithm

- Consider adding each quadratic just once
  - Intersection and comparison constant time
  - Adding to lists constant time
  - Removing from lists constant time
    - But then need to try again
- Amortized analysis
  - Total number of removals O(n)
    - Each quadratic, once removed, never considered for removal again
- Thus overall running time O(n)

1D L_2^2 Distance Transform

```
static float *dt(float *f, int n) {
  float *d = new float[n], *z = new float[n];
  int *v = new int[n];
  int k = 0;
  v[0] = 0;
  z[0] = -INF;
  z[1] = +INF;
  for (int q = 1; q <= n-1; q++) {
    float s  = ((f[q]+square(q))-(f[v[k]]+square(v[k])))
                /(2*q-2*v[k]);
    while (s <= z[k]) {
      k--;
      s  = ((f[q]+square(q))-(f[v[k]]+square(v[k])))
           /(2*q-2*v[k]);    }
    k++;
    v[k] = q;
    z[k] = s;
    z[k+1] = +INF; }
```


DT Values From Intersections

```
k = 0;
  for (int q = 0; q <= n-1; q++) {
    while (z[k+1] < q)
      k++;
    d[q] = square(q-v[k]) + f[v[k]];
  }
  return d;
}
```

- 2D version easily runs at video rates
- No reason to approximate L_2 distance
  - Simple to implement as well as fast

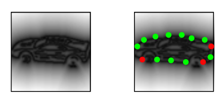
Distance Transforms in Matching

- Chamfer measure – asymmetric
  - Sum of distance transform values
    - "Probe" DT at locations specified by model and sum resulting values
- Hausdorff distance (and generalizations)
  - Max-min distance which can be computed efficiently using distance transform
  - Generalization to quantile of distance transform values more useful in practice
    - Max sensitive to even single outlier

Chamfer Measure

- Asymmetric comparison of two binary images A,B
  - Use points of A to select corresponding values in distance transform of B
  - Sum selected values

$$\text{chamf}(A,B) = \sum_{a\in A} \min_{b\in B} \|a-b\|$$
$$= \sum_{a\in A} D_B(a)$$



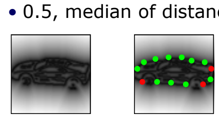
1+1+2+2+3+3+3+3+4+4+5+
12+14+15 = 72

Hausdorff Distance

- Classical definition
  - Directed distance (not symmetric)
    - $h(A,B) = \max_{a\in A} \min_{b\in B} \|a-b\|$
  - Distance (symmetry)
    - $H(A,B) = \max(h(A,B), h(B,A))$
- Minimization term simply a distance transform of B
  - $h(A,B) = \max_{a\in A} D_B(a)$
  - Maximize over selected values of DT
- Not robust, single "bad match" dominates

Hausdorff Matching

- Partial (or fractional) Hausdorff distance to address robustness to outliers
  - Rank rather than maximum
    - $h_k(A,B) = \text{kth}_{a\in A} \min_{b\in B} \|a-b\| = \text{kth}_{a\in A} D_B(a)$
  - K-th largest value of D_B at locations given by A
  - Often specify as fraction f rather than rank
    - 0.5, median of distances; 0.75, 75th percentile



1,1,2,2,3,3,3,3,4,4,5,12,14,15

.25 .5 .75 1.0

Applications of Pictorial Structures

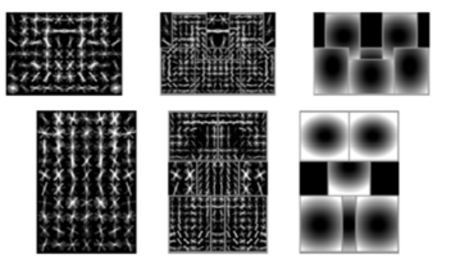
- PASCAL VOC 2008 [FMR08]
  - Discriminatively trained star-graph model
- Long-term arm and hand tracking [BEHZ08]
  - Hypothesize and test using hypotheses from pictorial structure model
- Person detection – full body and upper body models [ARS09]
  - Discriminatively learned part models

Top PASCAL08 Detection Method

- Mixture of star-graph spatial models [FMR08]
  - Given object category represented by several star graphs (e.g., encode multiple viewpoints)
  - Spatial deformation of fine-scale parts with respect to coarse-scale root part
  - Parts modeled using HOG templates
- Learned using weak supervision paradigm where object bounding box given but not part locations
- Discriminative training using SVM's

Form of Model

- Two component bicycle model with 6 parts



Score of Hypothesis

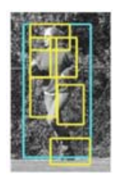
- Root w/n parts

Score of F at position p is $F \cdot \phi(p, H)$

"data term" "spatial prior"

$$\text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) - \sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)$$

filters — deformation parameters — displacements



$$\text{score}(z) = \beta \cdot \Psi(H, z)$$

concatenation filters and deformation parameters — concatenation of HOG features and part displacement features

Processing of Part Response

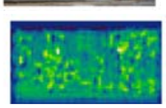
head filter

input image

Response of filter in l-th pyramid level

$$R_l(x, y) = F \cdot \phi(H, (x, y, l))$$

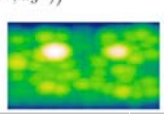
cross-correlation

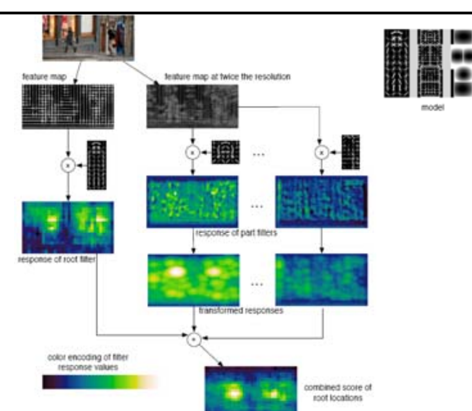


Transformed response

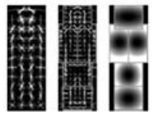
$$D_l(x, y) = \max_{dx,dy} \left(R_l(x + dx, y + dy) - d_i \cdot (dx^2, dy^2) \right)$$

max-convolution, computed in linear time (spreading, local max, etc)





Example Results



- After non-maximum suppression
- Fast: approx 1 sec to search all scales

Learning Models

- Learn root and part filters and spatial model from bounding boxes (positive and negative)
- Latent SVM
  - Classifiers that score an example x using
    $$f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z)$$
  - With model parameters β and latent variables z

Minimize

$$L_D(\beta) = \frac{1}{2}||\beta||^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i))$$

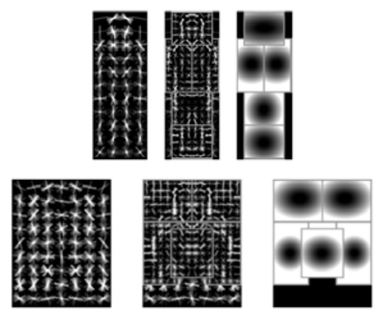
Training data $D = (\langle x_1, y_1 \rangle, \ldots, \langle x_n, y_n \rangle) \qquad y_i \in \{-1, 1\}$

Latent SVM Training

$$L_D(\beta) = \frac{1}{2}||\beta||^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i))$$

- Convex if fix z for positive examples
- Initialize β and iterate
  - Pick best z for each positive example
  - Optimize β via gradient descent
- Positive examples should have some z with high score, negative examples none

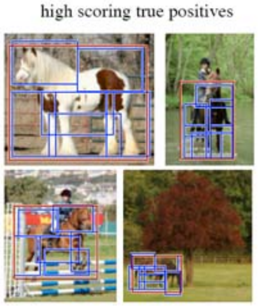
Learned Person Model (PASCAL)



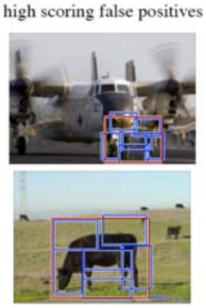
Coarse Root Fine Parts Spatial Constraint

Example Horse Detections (PASCAL)

high scoring true positives



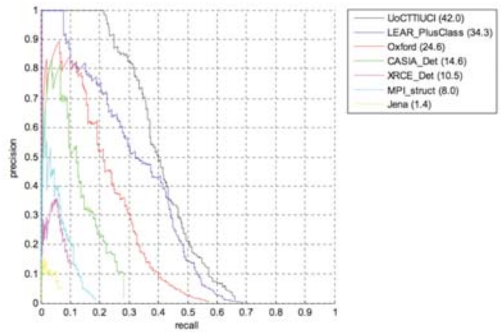
high scoring false positives



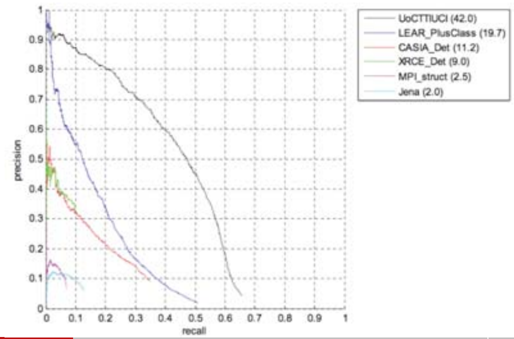
Quantitative Results

- 7 systems in 2008 PASCAL detection challenge
- 20 classes
  - First place in 7 classes, second place in 8 classes, lower rank in remaining 5 classes
  - No other system with as many first and second place rankings
- Much faster than most systems (algorithms!)
  - Approx 2 seconds to match a model to an image
  - Approx 4 hours to train a model

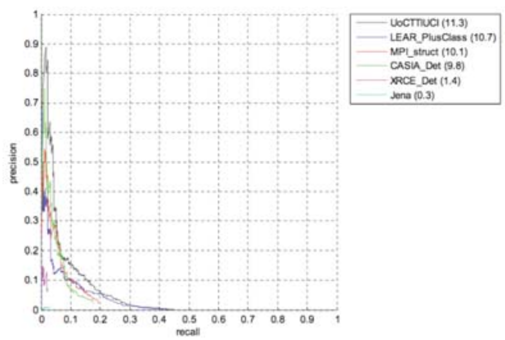
Bicycle Class PASCAL 2008



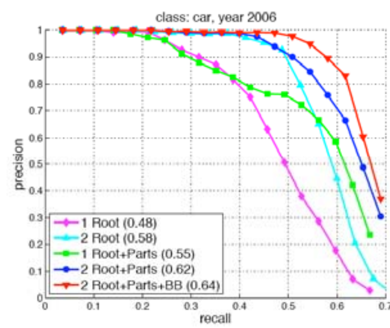
Person Class PASCAL 2008



Bird Class PASCAL 2008



Comparison of Car Models (2006 Data)



Arm and Hand Tracking

- For recognizing signing want long-term, accurate, computationally tractable tracking of arms and hands [BEHZ08]
- Find torso, then treat each arm as simple kinematic chain pictorial structure
- Empirically, sampling from max marginals gives better results than from marginals
  - Both better than straight MAP estimation
  - Good “test” criteria

Challenging Problem

- Long-term tracking of arms and hands in TV broadcasts is difficult

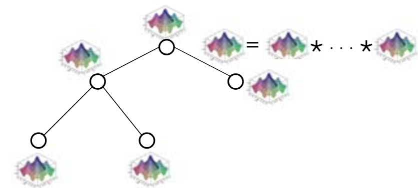
(a) Shading (b) Colour ambiguity (c) Motion blur (d) Hand ambiguity

Multiple Good Configurations

- Hypothesize and test paradigm
  - Postulate configurations by sampling pictorial structures with high posterior probability
  - Verify using other means
- For tree, factored distribution
  - Marginals or max marginals
    - Low dimensional table per part
  - Sample high probability location of "root" part
    - Posterior, fine if occluded (bad part likelihood)!
  - Then sample high (conditional) probability location for each child, and so on

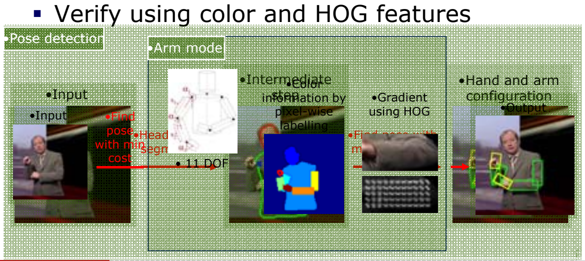
Computing (Max) Marginals

- Likelihoods and messages at each node over space of configurations
- Messages between nodes fast convolution (min conv.) of neighboring messages



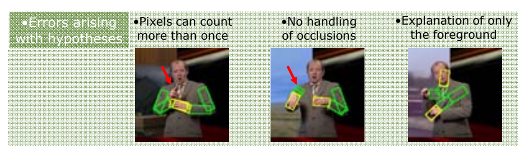
The Approach

- Root upper arms at shoulders of torso
  - Generate hypotheses
- Verify using color and HOG features



Sampling and Verifying

- Sample arm configurations using generative pictorial structure model
- Score each using color and HOG based match measure, keep best so far
- Scoring uses richer model



Sampling Example

Video illustrating sampling

Quantitative Results

Distribution of overlap scores for 296 frames:

Accuracy	Left arm	Right arm	Hands
Overlap ≥ 0.2	99.7%	100%	100%
Overlap ≥ 0.5	91.2%	99.7%	95.6%
Overlap ≥ 0.6	75.0%	82.8%	82.8%

→ **Qualitatively correct for nearly 100% of the frames**

Matching Quality

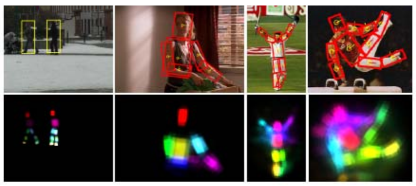
- Good enough arm and hand position for subsequent learning and recognition of signs
  - Using temporal information as well as individual frame matches
- Example video

Person Detection CVPR09

- State-of-art body pose estimation and detection (Andriluka et al, TU Darmstadt)
- Pictorial structure model
  - Use sum-product BP, but exact for acyclic graph
  - [FH05] 4D pairwise configuration x,y,s,o "loose limbed prior"
    - Found to be better than Ramanan05 spatial prior
  - Tree slightly better than star model
- Discriminatively trained part models
  - Densely sampled shape context previously used for pedestrian detection (BMVC 05)

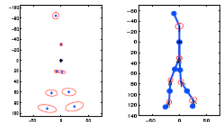
Example Results

- 10 part full body model, 8 part pedestrian model, 6 part upper body model
- Posterior rendered with color per part and intensity corresponding to probability

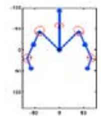


Spatial Priors

- 8 part pedestrian model, star versus tree

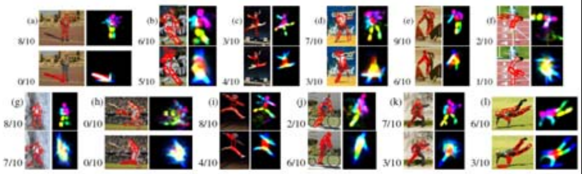


- 6 part upper body model



Full Body Pose Estimation

- Compared to Ramanan NIPS06



Method	Torso	Upper leg		Lower leg		Upper arm		Forearm		Head	Total
IIP [15], 1st parse (edge features only)	39.5	21.4	20	23.9	17.5	13.6	11.7	12.1	11.2	21.4	19.2
IIP [15], 2nd parse (edge + color feat.)	52.1	30.2	31.7	27.8	30.2	17	18	14.6	12.6	37.5	27.2
Our part detectors	29.7	12.6	12.1	20	17	3.4	3.9	6.3	2.4	40.9	14.8
Our inference, edge features from [15]	63.4	47.3	48.7	41.4	34.14	30.2	23.4	21.4	19.5	45.3	37.5
Our inference, our part detectors	**81.4**	**67.3**	**59**	**63.9**	**46.3**	**47.3**	**47.8**	**31.2**	**32.1**	**75.6**	**55.2**

Upper Body Pose Estimation

- Compared to Ferrari et al CVPR08

Method	Torso	Upper arm		Forearm		Head	Total
Progressive Search Space Reduction [6]	–	–	–	–	–	–	57.9
Our part detectors	18.9	6.6	7	3.3	2.9	47.2	14.3
Our part detectors and inference with generic prior	**90.7**	80.2	78.4	40.1	42.3	**95.9**	71.3
Our part detectors and inference with front/back view prior	**90.7**	**80.6**	**82.1**	**44.2**	**47.9**	95.5	**73.5**

Summary

- Pictorial structure models
  - Part-based appearance
  - Spatial constraints using small cliques of parts (pairs, triples)
- Simplifying models so that can do exact inference is good!
  - Apply dynamic programming can make fast
  - Currently best-performing object category recognition and person detection methods
- Even though discretizing parameters!

References

M: Method
A: Application

- M: P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. IJCV, 61(1):55-79, 2005.
- M: D. Crandall, P. Felzenszwalb, D. Huttenlocher. Spatial priors for part-based recognition using statistical models. CVPR 2005.
- M: P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. CVPR 2008.

- A: D. Ramanan. Learning to parse images of articulated objects. NIPS 2006.
- A: V. Ferrari, M. Marin, and A. Zisserman. Progressive search space reduction for human pose estimation. CVPR 2008.
- A: P. Buehler, M. Everingham, D. Huttenlocher and A. Zisserman. Long term arm and hand tracking for continuous sign language TV broadcasts. BMVC 2008.
- A: M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: people detection and articulated pose estimation. CVPR 2009.

Question 1

- What form of pictorial structure model learning works best in practice (2 answers)?
  - A) Maximum likelihood estimation of both the spatial model and part appearances
  - B) Maximum likelihood estimation of the spatial model and discriminative training for the part appearances
  - C) Discriminative training of the spatial model and maximum likelihood estimation of the part appearances
  - D) Discriminative training of both the spatial model and part appearances

Question 2

- For a pictorial structure model with n parts, h locations per part, and cliques of size k, how much does use of the distance transform speed up the running time?
  - A) A factor of k
  - B) A factor of n
  - C) A factor of h
  - D) A factor of h^k

Question 3

- In estimating human body pose using a pictorial structure model, for what part is the estimate least affected by occlusion of that part?
  - A) The root part
  - B) A leaf part
  - C) A part with many neighbors in the graph