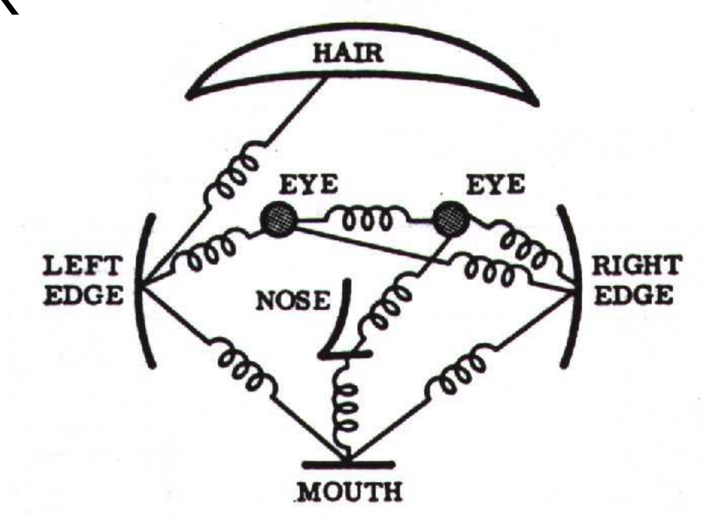
CS 664
Flexible Templates

Daniel Huttenlocher

Cornell University
Faculty of Computing and Information Science

Flexible Template Matching

- Pictorial structures
  - Parts connected by springs and appearance models for each part
  - Used for human bodies, faces
  - Fischler&Elschlager, 1973 – considerable recent work

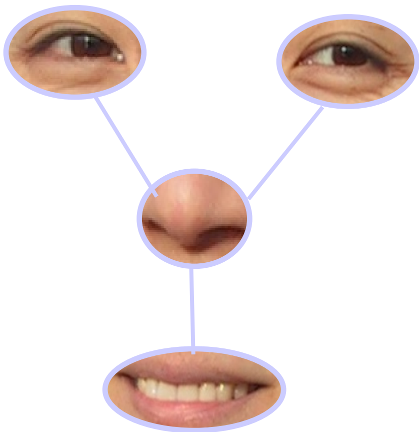


Formal Definition of Model

- Set of parts $V=\{v_1, \ldots, v_n\}$
- Configuration $L=(l_1, \ldots, l_n)$
  - Specifying locations of the parts
- Appearance parameters $A=(a_1, \ldots, a_n)$
  - Model for each part
- Edge e_{ij}, $(v_i,v_j) \in E$ for connected parts
  - Explicit dependency between part locations l_i, l_j
- Connection parameters $C=\{c_{ij} \mid e_{ij} \in E\}$
  - Spring parameters for each pair of connected parts

Flexible Template Algorithms

- Difficulty depends on structure of graph
  - Which parts connected and form of constraint
- General case exponential time
  - Consider special case in which parts translate with respect to common origin
    - E.g., useful for faces



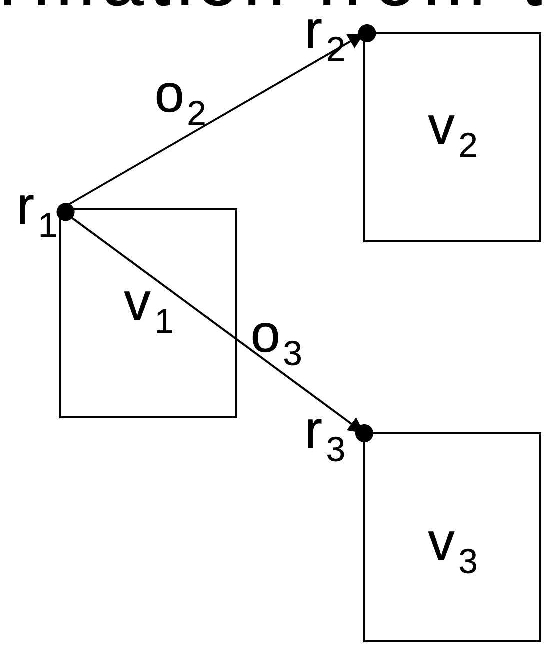
- Parts $V = \{v_1, \dots v_n\}$
- Distinguished central part v_1
- Spring c_{i1} connecting v_i to v_1
- Quadratic cost for spring

Efficient Algorithm for Central Part

- Location $L=(l_1, \ldots, l_n)$ specifies where each part positioned in image
- Best location $\min_L \left(\Sigma_i\, m_i(l_i) + d_i(l_i,l_1)\right)$
  - Part cost $m_i(l_i)$
    - Measures degree of mismatch of appearance a_i when part v_i placed at each of h locations, l_i
  - Deformation cost $d_i(l_i,l_1)$
    - 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
  - $\| l_j - T_j(l_1) \|^2$



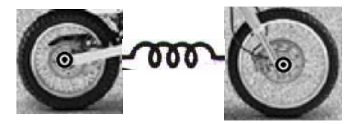
Consider Case of 2 Parts

- $\min_{l_1, l_2} (m_1(l_1) + m_2(l_2) + \| l_2 - T_2(l_1) \|^2)$
  - Where $T_2(l_1)$ transforms l_1 to ideal location with respect to l_2 (offset)
- $\min_{l_1} (m_1(l_1) + \min_{l_2} (m_2(l_2) + \| l_2 - T_2(l_1) \|^2))$
  - But $\min_x (f(x) + \| x - y \|^2)$ is a distance transform
- $\min_{l_1} (m_1(l_1) + D_{m_2}(T_2(l_1)))$
- 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

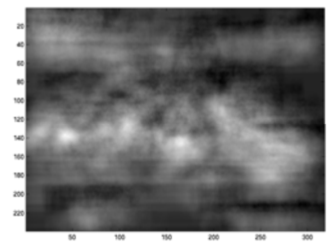
Cornell University

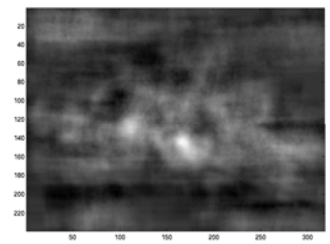
Overall Computation for 2 Parts

- Image and model (translation)



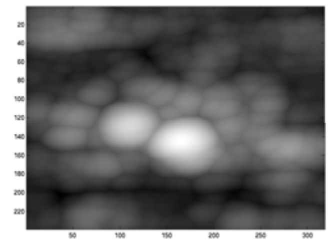
- Match cost of each part $m_1(l_1)$, $m_2(l_2)$





- Distance transform of $m_2(l_2)$

+



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

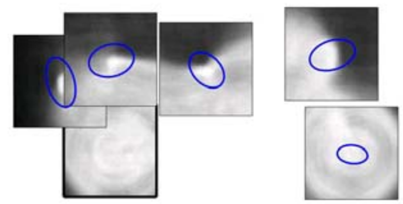
Star Graph – Central Reference Part

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

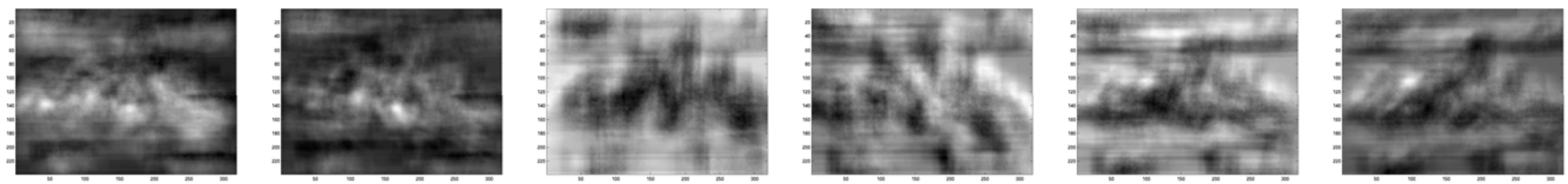
Star Graph

- 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

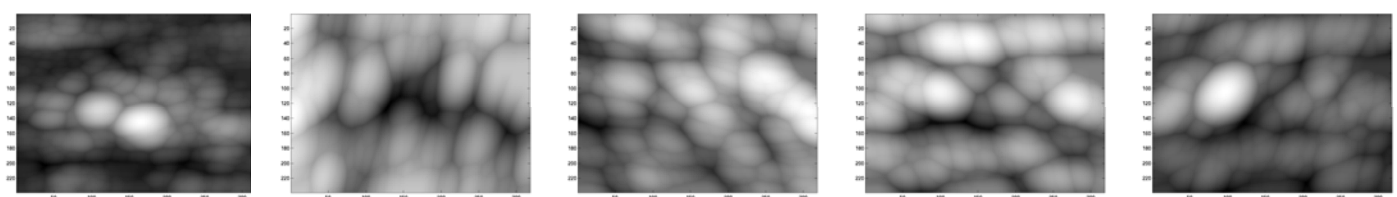
Overall Computation for Star Graph



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



- Distance transform non-reference part costs, sum to get MAP location, O(mn) 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 algorithm
  - Parameterization of distance transform more complex – transformation T_{ij} for each connected pair of parts

General Form of Problem

- Best location can be viewed in terms of probability or cost (negative log prob.)
  - $\max_L p(L|I,\Theta) = \text{argmax}_L p(I|L,A)p(L|E,C)$
  - $\min_L \Sigma_V m_j(l_j) + \Sigma_E d_{ij}(l_i,l_j)$
    - $m_j(l_j)$ – how well part v_j matches image at l_j
    - $d_{ij}(l_i,l_j)$ – how well locations l_i,l_j agree with model (spring connecting parts v_i and v_j)
- Difficulty of maximization/minimization depends on form of graph and pairwise cost

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 a DT $D_f(T_{ij}(l_i))$
  - Cost function
    - $f(y) = m_j(T_{ji}^{-1}(y)) + \sum_{Cj} B_c(T_{ji}^{-1}(y))$
  - 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
  - Each can be computed in O(hD) time
    - D is number of dimensions to parameter space but is fixed (D generally 2 to 4)

Sampling the Posterior

- Generate good possible matches as hypotheses
  - Locations where posterior $p(L|I,\Theta)$ large
  - 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
  - Running time under a minute for person model

Sampling Approach

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

  $p(l_r|I,\Theta) = \sum_{L\setminus lr} (\prod_V p(I|l_i,a_i) \prod_E p(l_i,l_j|c_{ij}))$

- Can be computed efficiently due to tree structured dependencies

  $p(l_r|I,\Theta) \propto p(I|l_r,a_r) \prod_{Ch} s_c(l_r)$

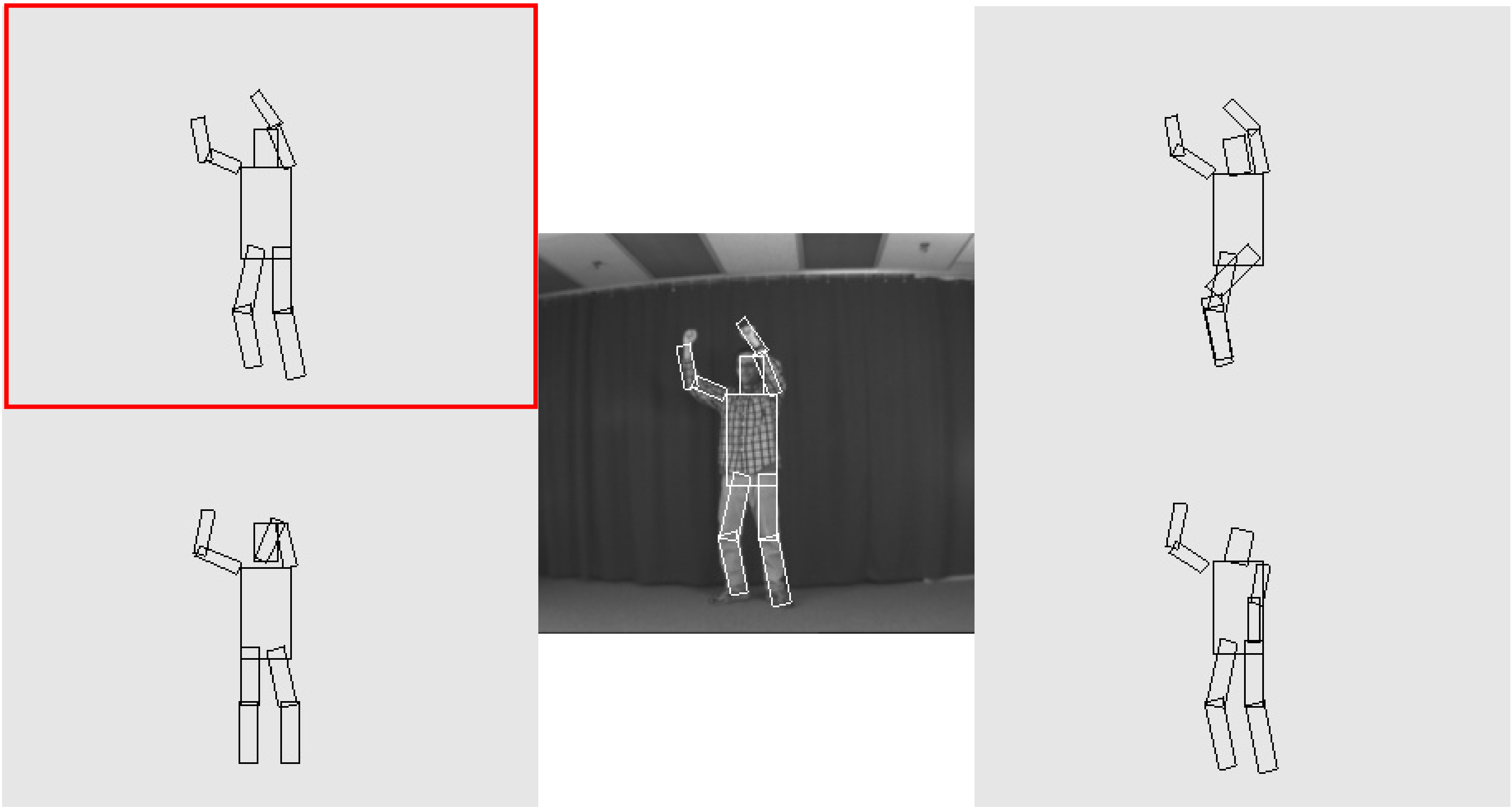
  – And fast convolution when $p(l_i,l_j|c_{ij})$ Gaussian

  $s_j(l_i) \propto \sum_{lj} (p(I|l_j,a_j)\ p(l_i,l_j|c_{ij}) \prod_{Ch} s_c(l_j))$

- Sample location for root from marginal

  – Sample from root to leaves using $p(l_j|l_i,I,\Theta)$

Samples From Posterior



Sampling from Proposal Distribution

- Can use to address limitations of models
  - Non-Gaussian pairwise constraints
  - Non-independence of individual part appearance
- Use model that factors to propose high probability answers according to a simpler model
- Maximize a less tractable criterion only for those sample configurations

Weakly Supervised Learning

- Consider large number of initial patch models to generate possible parts
  - Ranked by likelihood of data given part
- 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 models
  - One per reference set

Learning Spatial Model

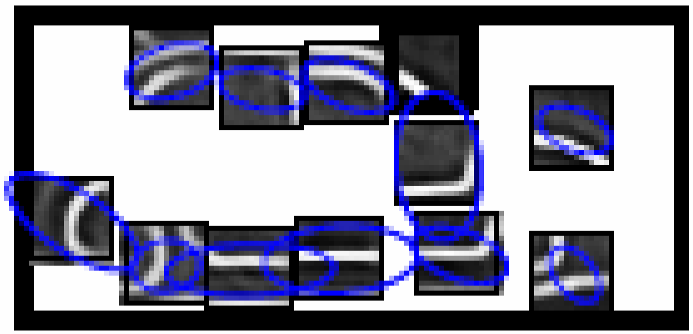
- Estimate pairwise spatial models for all pairs of patches – maximum likelihood
- Consider all k-tuples as root sets
- Use pairwise models to approximate true spatial model
  - Exact for 2-cliques (1-fan, star graph)
- Use EM to update model
  - Iteratively improve both appearance and spatial models

A More Accurate Form of Model

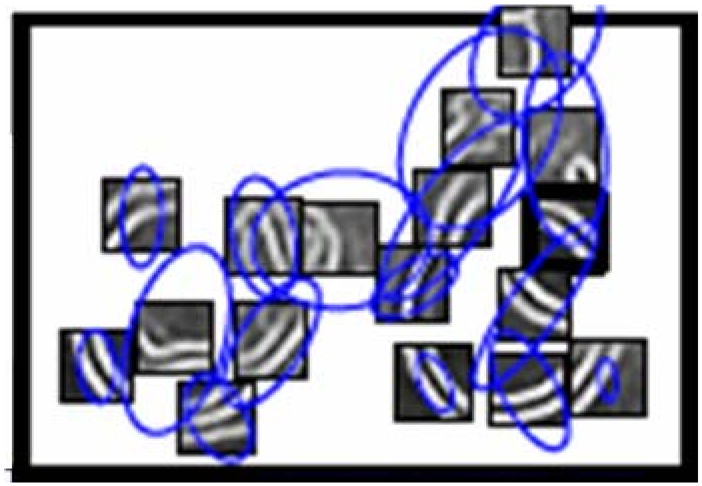
- Independent part appearance can over count evidence when parts overlap
  - Address by changing form of image likelihood
- POP – patchwork of parts [AT07]
  - More accurate model that accounts for overlapping parts
  - Average probabilities of patches that overlap
    - Distribution does not factor, can't compute efficiently
    - Can sample efficiently from factored distribution and then maximize POP criterion

Example Learned Models

- Star graph (one fan)
  - 24x24 patches
  - Reference part in bold box
  - Blue ellipse 2σ level set of Gaussian



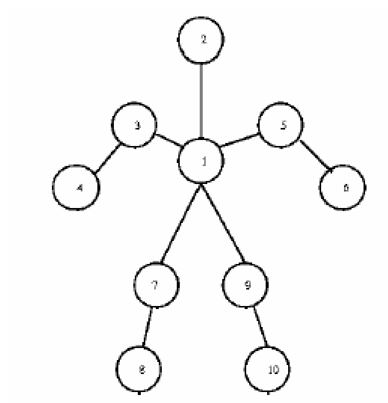
Side View of Car

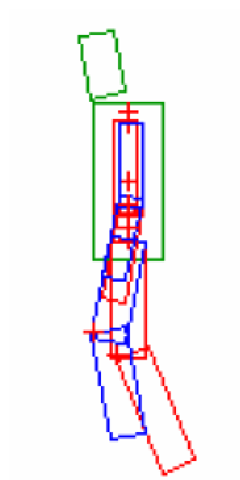


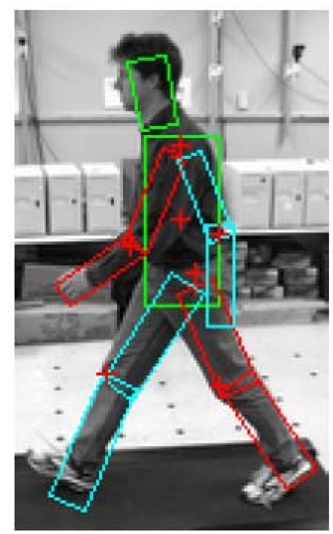
Side View of Bicycle

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

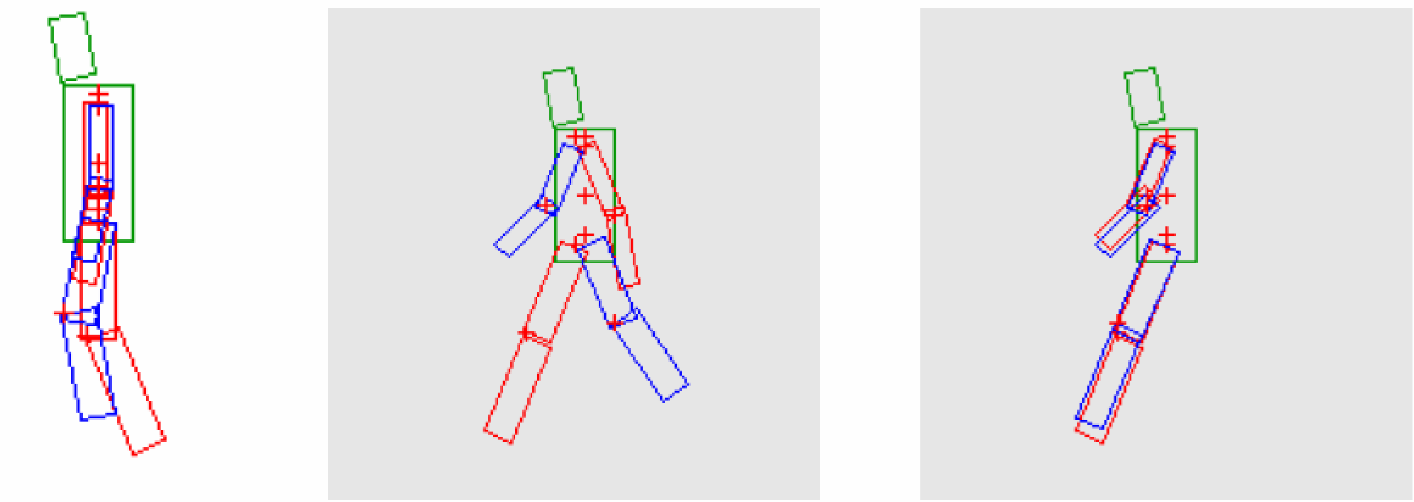






Limitations of Kinematic Trees

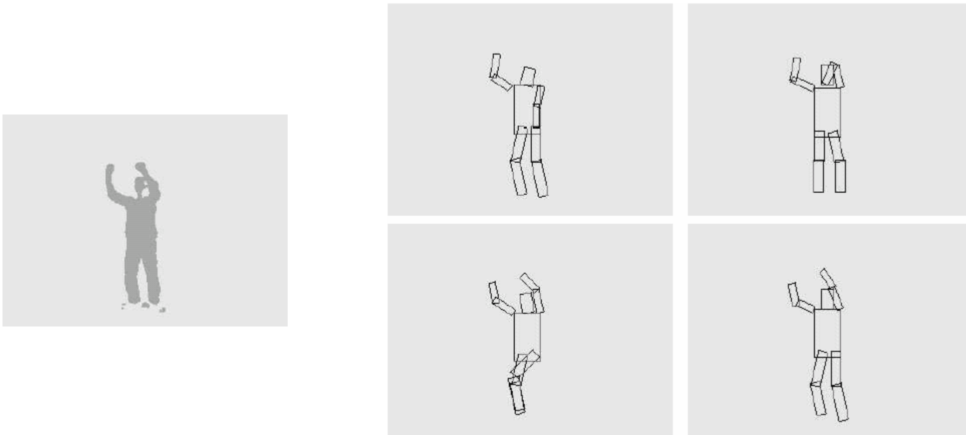
- Only represent relationships between connected parts
- Coordination between limbs not encoded
  - Critical for balance and many activities



Equally good under tree model

Addressing Limitations

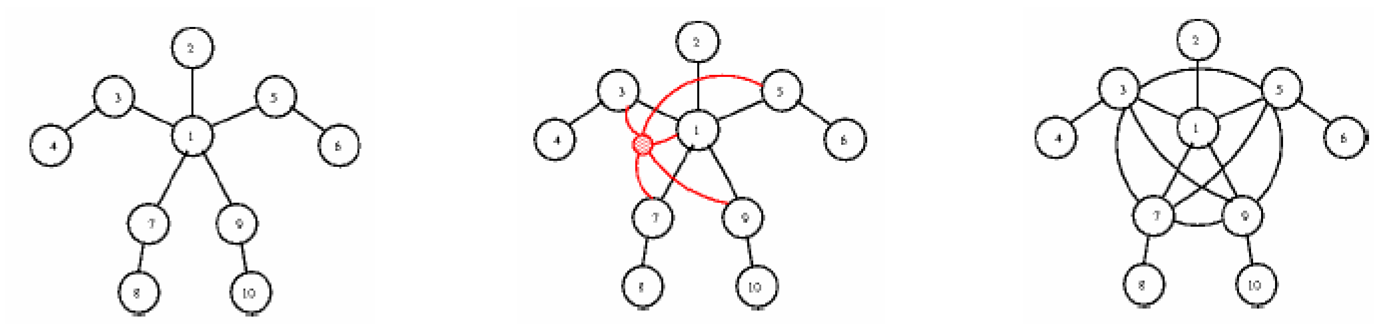
- Sampling based approaches
  - Probabilistic model
  - Sample high posterior probability poses and verify using other means (e.g, IF01, FH05)
    - Tractable because posterior factors



- Conditional random fields

Our Approach: Richer Spatial Model

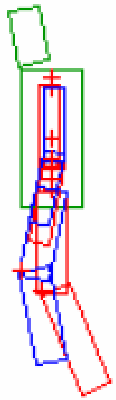
- Latent variables to encode additional relationships – e.g., between upper limbs
  - Low order (small cliques) to ensure efficient optimization/inference
- In contrast to simply adding constraints which can result in large clique
  - Running time exponential in clique size



Cornell University

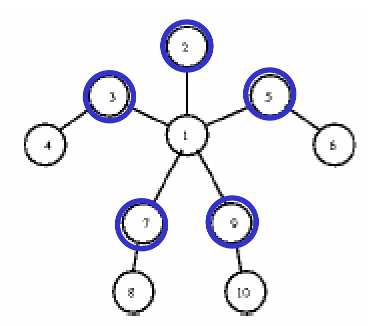
Learning Latent Variable Models

- First learn tree model [FH00,FH05]
  - Maximum likelihood estimation
  - Learn connections between parts and spatial relations
- Yields kinematic tree automatically
  - Lowest variability connections between parts
- Example using 240 labeled side-walking frames in CMU HumanID dataset
  - Shown at mean pose



Identify Violations of Tree Model

- Conditional independence
  - Parts with common "parent" should have uncorrelated locations given location of parent
- Consider simple 2D human body model
  - Pairwise relations parameterized by position, orientation and scale

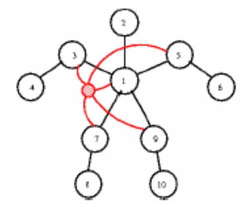


	Head	Lf. Arm	Lf. Leg	Rt. Arm	Rt. Leg
Head	1.00	0.00	-0.00	-0.06	0.00
Lf. Arm	0.00	1.00	-0.58	-0.83	0.67
Lf. Leg	-0.00	-0.58	1.00	0.61	-0.43
Rt. Arm	-0.06	-0.83	0.61	1.00	-0.59
Rt. Leg	0.00	0.67	-0.43	-0.59	1.00

Correlation in orientation given torso location

Test for Underlying Explanation

- Violations of conditional independence correspond to additional constraints
  - But don't want to model with large clique



- Determine whether simple parametric characterization of these constraints
  - Use factor analysis to identify common factor

    $$Y = \mathcal{N}(AX, \Lambda)$$

  - Factor loading vector A controls how scalar factor X affects variables Y
  - For human walking yields a single highly predictive gait-cycle parameter ("swing")

Summary of Model Learning

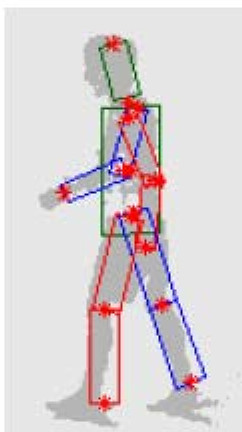
- Learn a tree model from labeled training data (max likelihood estimation)
- Identify parts that violate conditional independence of tree model
  - With respect to common parent
- Use factor analysis to discover underlying control variable(s)
- Introduce these latent variable(s) into the tree model
  - Yielding tree-like model

Inference Using These Models

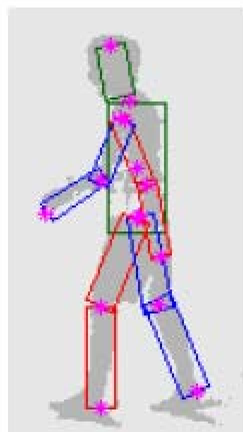
- When value of latent variable is fixed, have a tree
  - Efficient exact inference using Viterbi, forward or belief propagation algorithms
- Optimize over range of values of latent variable
- Use generalized distance transform methods to accelerate running time
  - Still exact estimation (global optimum)

Examples Using Brown MOCAP Data

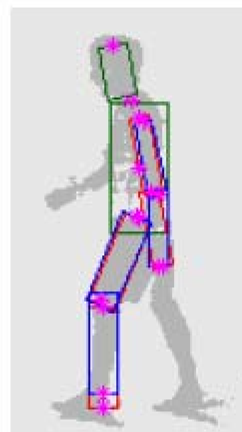
- MAP estimate of best pose, single frame



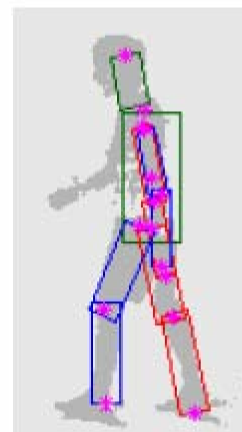
Ground Truth



Common Factor Model



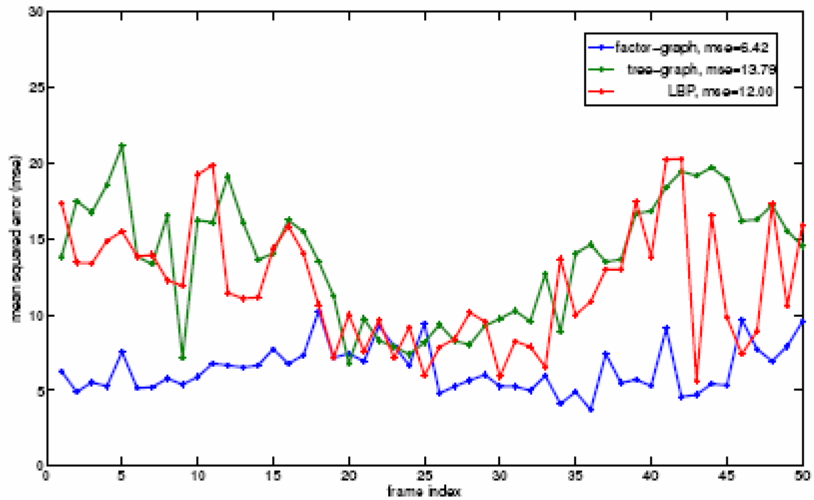
Tree Model



Clique Model Using LBP

Results on Brown Sequence

- Per frame error, averaged over joints

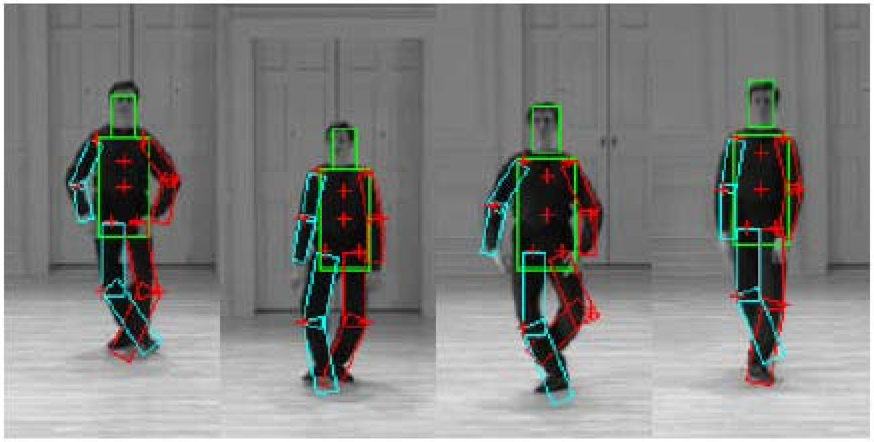


- Per joint error, averaged over frames

	shoulder	elbow	wrist	hip	knee	ankle
Factor	4.8	5.5	8.6	4.2	4.4	5.4
Tree	9.1	11.1	19.4	6.4	6.6	28.6
LBP	9.9	11.9	20.5	6.4	5.3	20.5

Examples

- Common factor model



- Tree model