Efficient Recognition Algorithms

Pedro Felzenszwalb
University of Chicago

Algorithms for Recognition

- Recognition often posed in terms of optimization
  - Define a function measuring the quality of an interpretation
  - Search over interpretations
  - Separation between problem formulation and computation
- Many objects, many pixels - inherently complex problem
  - We want linear time algorithms

Issues

- General algorithms would be great, but more specific algorithms are always faster
- There is a fine line between tractable and intractable problem
  - Can often formulate vision problem in different ways
  - We should leverage this fact (carefully)
- Often reduce a problem to one with known efficient algorithm
  - Solving segmentation with min-cut
  - Learning pictorial structure model with MST
  - Matching sets of features with Hungarian algorithm

Intractability

- Optimal solutions not intractable just because there are exponentially many possibilities and non-convex objective
- Sorting n numbers: n! possibilities, non-convex objective
  - Easy problem
- Some surprises related to vision:
  - O(n log n) algorithm for Convolution
  - Solving binary MRFs
  - Pictorial structures: free spatial relations in simple models
- We can make REAL progress through algorithms

Discrete vs. Continuous Optimization

- Discrete optimization:
  - Search space has finite or countably many solutions
  - Try to get guarantees about solution quality
- Continuous optimization:
  - Search space has uncountably many solutions
  - Example: looking for a set of real numbers
  - Often get guarantees about convergence to local minimum
- Relaxations transform a discrete problem into a continuous one

Toolbox of Efficient Algorithms

- Branch and bound / Coarse-to-fine
- Dynamic programming
- Graph algorithms
- Linear programming
- Belief propagation
- Spectral methods
- Fast Fourier Transform
- Fast approximate nearest-neighbors

Generic techniques

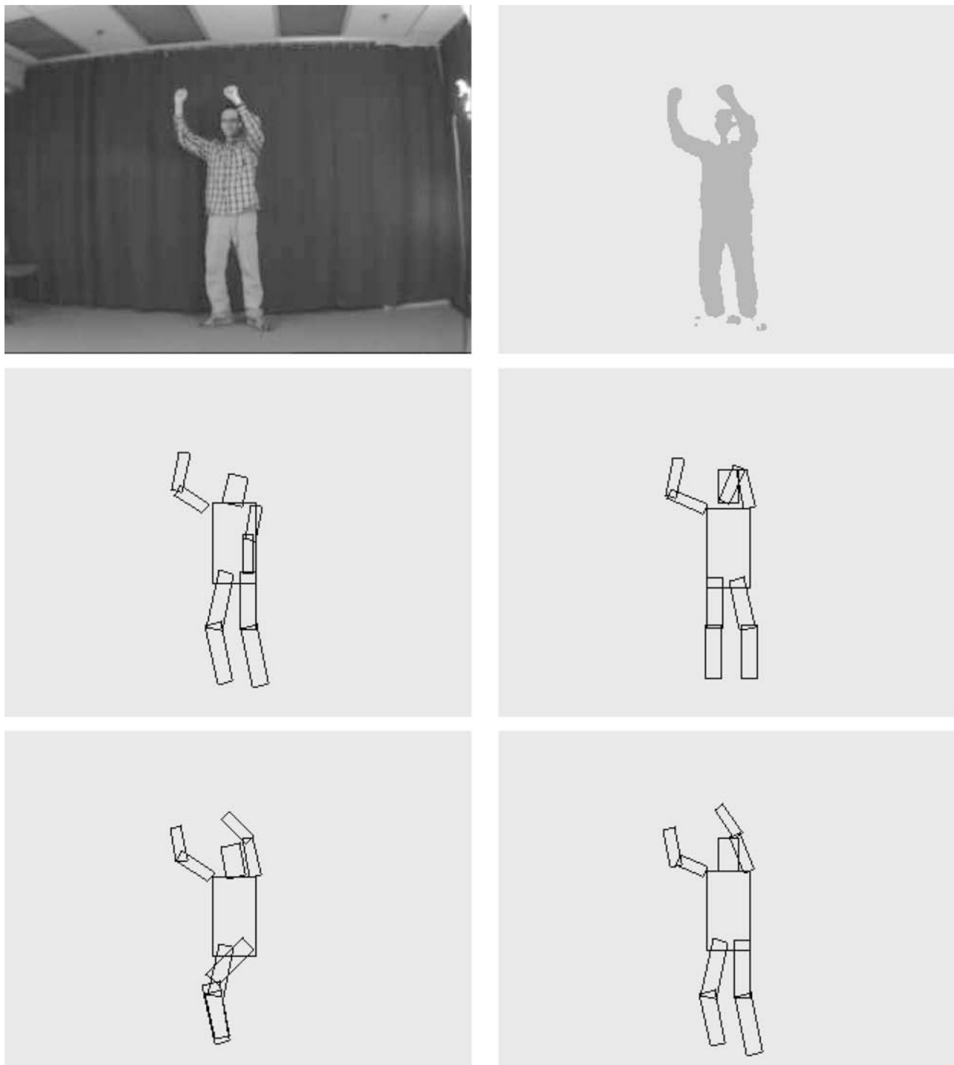
↓

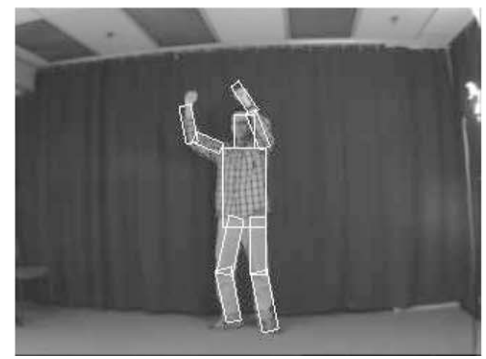
Specific problems

Efficient Recognition Architectures

- Recognition in time sub-linear in the number of models
  - Feature detection + nearest-neighbor methods
  - Shared parts among objects
- Branch and bound search with multiscale models
  - If we make models multiscale, its easy to compute bounds
- Hypothesize and test
  - Use efficient algorithms to find good hypothesis and a separate refinement/verification step
- Feed-forward methods

Efficient methods for generating Hypothesis





Weak Methods

- Local search
  - Example: gradient descent
  - Problem: get stuck in local minima
  - Can be improved with a stochastic component
- Markov Chain Monte Carlo
  - Metropolis, Gibbs sampling, Simulated annealing
  - Hard to prove mixing times (either lower or upper bound)
  - In general equivalent to exhaustive search

Limitations

- Efficient algorithms often solve a simplified problem
  - Can be hard to modify without losing guarantees
- Worst case analysis ignores easy situations
- Approximation algorithms are hard to come by
- Heuristics often work in particular problems