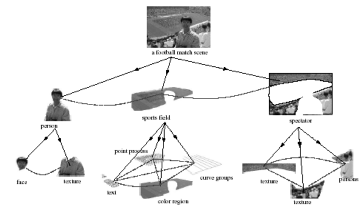
Image Parsing & DDMCMC.

Alan Yuille (Dept. Statistics. UCLA)

(Largely based on: Tu, Chen, Yuille & Zhu ICCV 2003, IJCV 2005. Yuille & Kersten 2006).

Image Parsing.

- (I) Image are composed of visual patterns:
- (II) Parse an image by decomposing it into patterns.

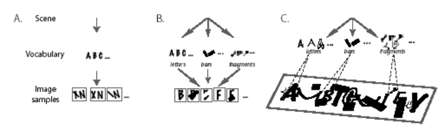


Analysis by Synthesis (AS).

- Ability to use models P(I|W) & P(W) to synthesize images of objects.
- This is an internal ability of the brain – dream of objects, simulate the environment, mental images?
- AS (1): synthesize until the observed images is identical to the synthesized image.
- AS(2): use proposals: low-level cues propose objects, that can be validated or rejected by synthesis.

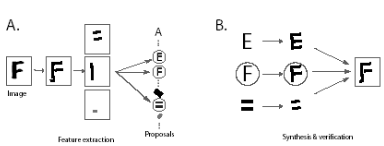
Generating an Image

- Generate an Image in terms of vocabularies of features.
- Simple vocabularies give rise to little ambiguity and easy inference.



Inference: interpreting images.

- Low-level features propose hypotheses that are validated or rejected by high-level generative models.



If the brain was simple, then we couldn't understand it. John Mayhew.

Part I: How to Generate an Image.

- Stochastic grammar for generating images in terms of ***visual patterns***.
- Visual patterns can be ***generic*** (texture/shading) or ***objects***.
- Hierarchical Probability model – probability on graphs.

Part II: How to Parse an Image

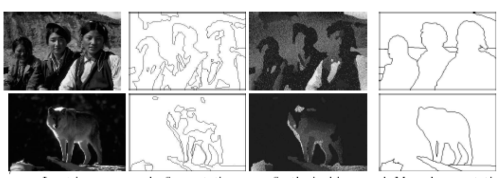
- Interpreting an image corresponds to constructing a ***parse graph***.
- Set of ***moves*** for constructing the parse graph.
- Dynamics for moves use bottom-up & top-down visual processing.
- Data-Driven Markov Chain Monte Carlo (DDMCMC).
- Discriminative Models to drive Generative models.

Part I: Generative Models.

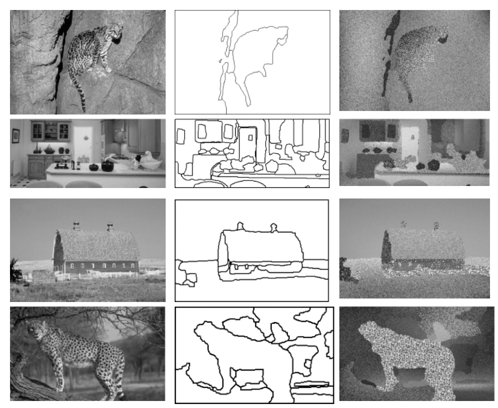
- Previous related work by our group:
- Zhu & Yuille 1996 (Region Competition).
- Tu & Zhu 2002. Tu & Zhu 2003.
- These theories assumed *generic visual patterns* only.

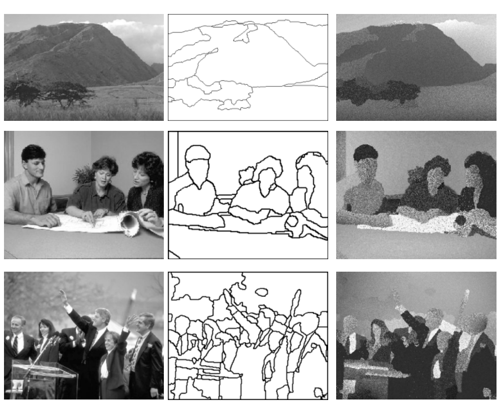
Generic Patterns & Object Patterns.

- Limitations of Generic Visual Patterns.
- Object patterns enable us to unify segmentation & recognition.



a. Input image b. Segmentation c. Synthesized image d. Manual segmentation





Stochastic Grammar: Parsing Graph.

- Nodes represent visual patterns. Child nodes to image pixels.

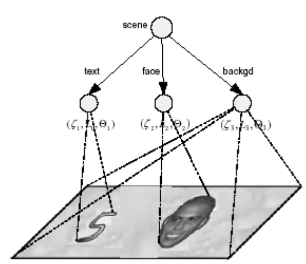


Image Patterns.

- Node attributes: $\zeta_i, L_i, \Theta_i.$
- ***Zeta***: Pattern Type – 66
  (I) Gaussian,
  (II) Texture/Clutter,
  (III) Shading.
  (IV) Faces,
  (V– LXVI) Text Characters.
- ***L*** – shape descriptor (image region modeled).
- ***Theta***: Model parameters.

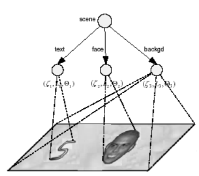
$$W = (K, \{(\zeta_i, L_i, \Theta_i) : i = 1, 2, ..., K\}).$$

Generative Model:

- Likelihood: $p(\mathbf{I}|W) = \prod_{i=1}^{K} p(\mathbf{I}_{R(L_i)}|\zeta_i, L_i, \Theta_i).$
- Prior: $p(W) = p(K) \prod_{i=1}^{K} p(L_i) p(\zeta_i|L_i) p(\Theta_i|\zeta_i).$
- Samples:

Stochastic Grammar Summary.

- Graph represents: $W = (K, \{(\zeta_i, L_i, \Theta_i) : i = 1, 2, ..., K\}).$
- Sampling from the graph generates an image.

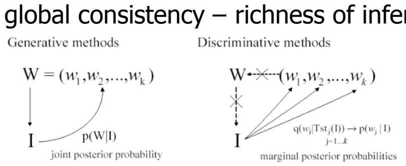


Part II: Inference Algorithm.

- We described a model to generate image: P(I|W) & P(W).
- Need an algorithm to infer W* from P(I|W) & P(W).

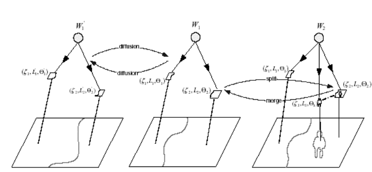
Inference Strategy:

- Discriminative versus Generative.
- Discriminative methods (eg. SVM, AdaBoost, etc.) can be very fast.
- But generative methods are better for global consistency – richness of inference.



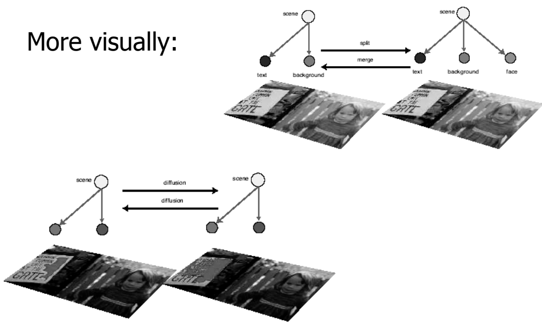
Inference & Parse Graph.

- Inference requires constructing a parse graph.
- Dynamics to create/delete nodes and alter node attributes:



Inference Dynamics

More visually:



Moves:

- Birth & Death of Text.
- Birth & Death of Faces.
- Splitting & Merging of Regions.
- Switching Node Attributes (Model Parameters & Pattern Types).
- Moving Region Boundaries.

Markov Chain Monte Carlo.

- Design a Markov Chain (MC) with transition kernel
$$\mathcal{K}(W'|W:\mathbf{I})$$
- Satisfies Detailed Balance.
$$p(W|\mathbf{I})\mathcal{K}_a(W'|W:\mathbf{I}) = p(W'|\mathbf{I})\mathcal{K}_a(W|W':\mathbf{I}).$$
- Then repeated sampling from the MC will converge to samples from the posterior P(W|I).

Moves & Sub-kernels.

- Implement each move by a transition sub-kernel:
$$\mathcal{K}_a(W'|W:\mathbf{I})$$
- Combines moves by a full kernel:
$$K(W,W') = \sum_i \alpha_i(I) K_i(W,W'), \quad \sum_i \alpha_i(I) = 1$$
- At each time-step – choose a type of move, then apply it to the graph.
- Kernels obey:
$$\sum_W K(W,W')P(W|I) = P(W'|I)$$

Data Driven Proposals.

- Use data-driven proposals to make the Markov Chain efficient.
- Metropolis-Hastings design:
$$K_i(W,W') = Q_i(W,W'|Tst_i(\mathbf{I}))\,\min\{1, \frac{P(W'|\mathbf{I})}{P(W|\mathbf{I})}\frac{Q_i(W,W'|Tst_i(\mathbf{I}))}{Q_i(W,W'|Tst_i(\mathbf{I})}\}.$$
- Proposal probabilities are based on discriminative cues.
$$Q_i(W,W'|\mathbf{I})?$$

Discriminative Methods:

- Edge Cues
- Binarization Cues.
- Face Region Cues (AdaBoost).
- Text Region Cues (AdaBoost).
- Shape Affinity Cues.
- Region Affinity Cues.
- Model Parameters & Pattern Type.

Design Proposals I.

- How to generate proposals $Q_i(W, W'|\mathbf{I})$?

$S_i(W)$ is the scope of W– states that can be reached from W with one move of type i.

- *Ideal proposal*:

$$Q_i(W, W'|\mathbf{I}) = \frac{P(W'|\mathbf{I})}{\sum_{W'' \in S_i(W)} P(W''|\mathbf{I})}, \quad W' \in S_i(W)$$

$$Q_i(W, W'|\mathbf{I}) = 0, \quad \text{otherwise}$$

Design Proposals II.

- Re-express this as:

$$Q_i(W, W'|\mathbf{I}) = \frac{P(W'|\mathbf{I})/P(W|\mathbf{I})}{\sum_{W'' \in S_i(W)} P(W''|\mathbf{I})/P(W|\mathbf{I})}$$

for $W' \in S_i(W)$

$$Q_i(W, W'|\mathbf{I}) = 0, \quad \text{otherwise}$$

- Set Q(W,W'|I) to approximate P(W'|I)/P(W|I) and *be easily computatable*.

Example: Create Node by Splitting

- Select region (node) R_k.
- Propose a finite set of ways to split R_k based on discriminative cues (edges and edge linking).
- Consider split R_k to R_i & R_j.

$$\frac{p(W'|\mathbf{I})}{p(W|\mathbf{I})} = \frac{p(\mathbf{I}_{R_i}|\zeta_i,L_i,\Theta_i)p(\mathbf{I}_{R_j}|\zeta_j,L_j,\Theta_j)}{p(\mathbf{I}_{R_k}|\zeta_k,L_k,\Theta_k)} \cdot \times \frac{p(\zeta_i,L_i\Theta_i)p(\zeta_j,L_j,\Theta_j)}{p(\zeta_k,L_k,\Theta_k)} \cdot \frac{p(K+1)}{p(K)}$$

$\phi_i = (\zeta_i, L_i, \Theta_i)$ are node attributes.

Example: Create Node by Splitting.

- Create (Split).

$$Q_i(W, W'|\mathbf{I}) \approx \frac{p(\mathbf{I}_{R_i}|\phi_i)p(\mathbf{I}_{R_j}|\phi_j)}{p(\mathbf{I}_{R_k}|\phi_k)} \cdot \times \frac{p(\phi_i)p(\phi_j)}{p(\phi_k)} \cdot \frac{p(K+1)}{p(K)}$$

- Denominator is known, we are in state W.

$Q_{ij} \approx p(\mathbf{I}_{R_i}|\phi_i)p(\mathbf{I}_{R_j}|\phi_j)$ Q_{ij} independent of ϕ_i, ϕ_j

- Use an affinity measure

Example: Create Face

- Create Face.
- Bottom-up proposal for face driven by AdaBoost & edge detection.
- This will require splitting an existing region R_k into R_i (face) and R_j (remainder).
- Parameters for R_k & R_j are known (the same).

$$Q_i(W, W'|\mathbf{I}) \approx \frac{p(\mathbf{I}_{R_i}|\zeta_i,L_i,\Theta_i)p(\mathbf{I}_{R_j}|\zeta_j,L_j,\Theta_j)}{p(\mathbf{I}_{R_k}|\zeta_k,L_k,\Theta_k)} \cdot \times \frac{p(\zeta_i,L_i,\Theta_i)p(\zeta_j,L_j,\Theta_j)}{p(\zeta_k,L_k,\Theta_k)} \cdot \frac{p(K+1)}{p(K)}$$

Examples: Death by Merging.

- Death (Merge).
- Select regions R_i & R_j to merge based on a similarity measure (as for splitting).
- Same approximations as for splitting to approximate.

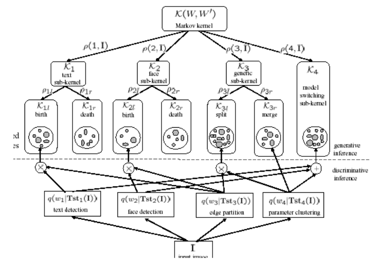
$$Q_i(W', W|\mathbf{I}) \approx \frac{p(\mathbf{I}_{R_k}|\zeta_k,L_k,\Theta_k)}{p(\mathbf{I}_{R_i}|\zeta_i,L_i,\Theta_i)p(\mathbf{I}_{R_j}|\zeta_j,L_j,\Theta_j)} \cdot \times \frac{p(\zeta_k,L_k,\Theta_k)}{p(\zeta_i,L_i,\Theta_i)p(\zeta_j,L_j,\Theta_j)} \cdot \frac{p(K)}{p(K+1)}$$

Node Splitting/Merging.

- Causes for split/merge.
- (i) Bottom-up cue – there is probably a face or text here (AdaBoost + Edge Detection).
- (ii) Bottom-up cue – there is an intensity edge splitting the region.
- (iii) Current state W – model for region I fits data poorly.
- (iv) Current state W -- two regions are similar (by affinity measure).

Full Strategy:

- Integration:



The bottom-up proposals are increasing good...

- Text Detection and Binarization.

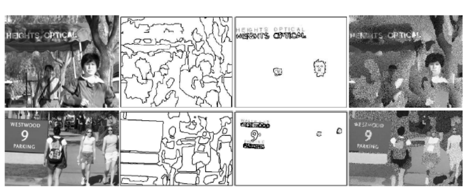
AdaBoost – Conditional Probs.

- Supervised Learning.

$$q(\ell = +1|\mathbf{I}) = \frac{e^{\ell<\alpha_{\mathrm{Ada}}, \mathrm{Tst}_{\mathrm{Ada}}(\mathrm{I}_t)>}}{e^{<\alpha_{\mathrm{Ada}}, \mathrm{Tst}_{\mathrm{Ada}}(\mathrm{I}_t)>} + e^{-<\alpha_{\mathrm{Ada}}, \mathrm{Tst}_{\mathrm{Ada}}(\mathrm{I}_t)>}}.$$

Experiments:

- Competition & Cooperation.



a. Input image b. Segmentation c. Object recognition d. Synthesized image

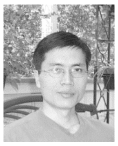
How to Go Forward?

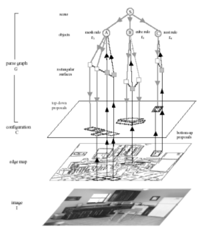
- Need more sophisticated grammatical models. Hierarchical models. And/Or graphs.

- Learn the models from data:
  (i) Supervised: The Lotus Hill Dataset.
  (ii) Unsupervised (next week).

Zhu's Program.

- Zhu & Mumford:

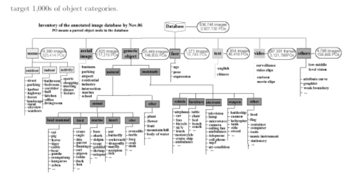


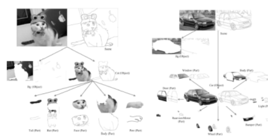


- www.stat.ucla.edu/~sc**zhu**/papers/**Grammar**_quest.pdf

Lotus Hill Dataset.

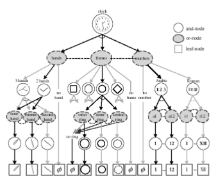
- *Construct grammars by interactive parsing.*

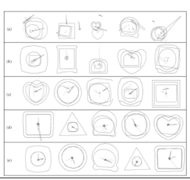


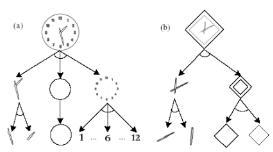


Clock Example

- *Grammar (AND/OR)*
- *Samples from models*
- *Parses*







Unsupervised Learning

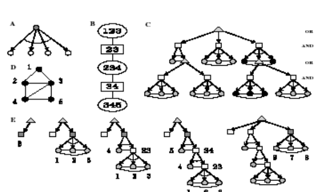
- Constellation models (Perona's Caltech group).
- Compositional models (Geman).
- Can we learn these models in an upsupervised/semi-supervised manner?

Learning Graphical Models.

- Learn a "grammar" defined on interest points in the image.

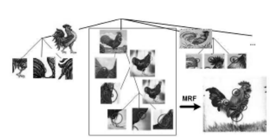
Triangles: Or nodes. Triplets as basic building blocks.

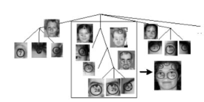
Combination of triples gives Junction-tree representations, enables rapid computation. Inference: 1 second.

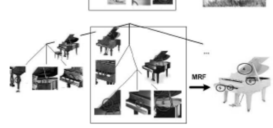


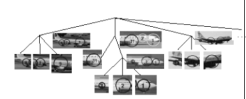
Example Models

- Grand Piano, Rooster, Faces, Motorbikes, Airplanes.



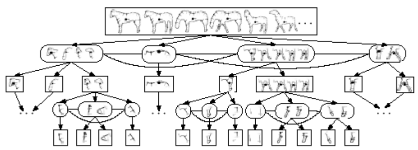






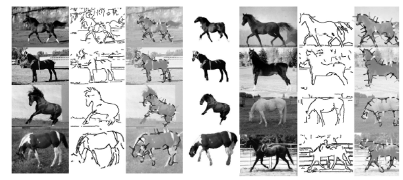
Horses as AND/OR Graphs.

- Horse decomposition.



Parsed Results for AND/OR graph

- The OR nodes enable the model to account for different configurations of the horse.



Key Ideas of Image Parsing:

- Generative Models for Visual Patterns & Stochastic Grammars.
- Inference: set of "moves" on the parse graph implemented by Kernels.
- Discriminative Models – bottom-up – drive top-down Generative Models.
- Proposals and validation.

The Future:

- More sophisticated representations learnt from large datasets.
- Stochastic Grammars, visual vocabularies, re-useable parts, compositionality.
- Bottom-up/top-down processing.