SCENE AND CONTEXT
SCENE UNDERSTANDING

What is goal of scene understanding:

- Build machine that can see like humans to automatically interpret the content of the images.

Comparing with traditional vision problem:

- Study on larger scale
- Human vision related tasks
LARGER SCALE

More image information. Context information.

focal length = 35 mm
HUMAN VISION RELATED TASK

More similar as the way that human understand the image
Infer more useful information from image
HOW DO HUMAN LEARN?

Bayesian Rules:

\[ P(A | B) = P(B | A) \cdot P(A) / P(B) \]

In practice: Infer abstract knowledge based on observation

\[ P(W | I) = P(I | W) \cdot P(W) / P(I) \]

Posterior probability

Likelihood: The probability of getting I given model W

Prior: The probability of W w/o seeing any observation
HOW DO HUMAN LEARN?

To teach human baby what is “horse”: show 3 pictures and let them learn by themselves.

They can be very successful to learn the correct concept.

But all the following concepts can explain the images:

• “horse” = all horse
• “horse” = all horse but not Clydesdales
• “horse” = all animal

$I =$

“horse”
Likelihood: What is the probability of sampling these 3 horse images given the concept?

\[ P(I \mid W_3) \approx P(I \mid W_2) > P(I \mid W_1) \]

Prior: How likely would the concept be the referent of a common word?

\[ P(W_1) \approx P(W_2) > P(W_3) \]

\[ W_2 = \operatorname{argmax}_w (P(I \mid W) \cdot P(W)) \]
TASK 1: IMAGE SEGMENTATION

\[ W_{\text{best}} = \arg \max_W (P(W \mid I)) = \arg \max_W (P(I \mid W) \cdot P(W)) \]

How to formulate the segmentation problem?

Warning: A huge wave of equations is approaching!
FORMULATION #1

Image Lattice: \[ \Lambda = \{(i, j) : 1 \leq i \leq L, 1 \leq j \leq H\} \]

Image: \[ I_\Lambda \]

For any point \( v \in \Lambda \) either \( I_v \in \{0, \ldots, G\} \)

Lattice partition into \( K \) disjoint regions:
\[
\Lambda = \bigcup_{i=1}^{K} R_i, \quad R_i \cap R_j = \emptyset, \quad \forall i \neq j
\]

Region is discrete label map: \[ R \subset \Lambda \]
Region Boundary is Continuous: \[ \Gamma_i = \partial R_i \]
FORMULATION #2

Each Image Region $I_R$ is a realization from a probabilistic model $p(I_R; \Theta)$

$\Theta$ are parameters of model indexed by $\ell$.

A segmentation is denoted by a vector of hidden variables $W$; $K$ is number of regions

$$W = (K, \{(R_i, \ell_i, \Theta_i); \ i = 1, 2, \ldots, K\})$$

Bayesian Framework:

$$W \sim p(W|I) \propto p(I|W)p(W), \ W \in \Omega.$$
\[ W \sim p(W|I) \propto p(I|W)p(W), \quad W \in \Omega. \]

**PRIOR OVER SEGMENTATIONS**

\[ p(W) \propto p(K) \prod_{i=1}^{K} p(R_i) p(\ell_i) p(\Theta_i | \ell_i) \]

- \( p(K) \propto e^{-\lambda_0 K} \)  \text{Want less regions}
- \( p(\Gamma) \propto e^{-\mu \int_{\Gamma} ds} \)  \text{Want round-ish regions}
- \( p(\ell) \sim \text{uniform} \)
- \( p(\Theta | \ell) \propto e^{-\nu|\Theta|} \)  \text{Want less complex models}
- \( p(A) \propto e^{-\gamma A^c} \)

\[ A = |R_i| \quad \text{Want small regions} \]

\[
p(W) \propto \exp \left\{ -\lambda_0 K - \sum_{i=1}^{K} \left[ \mu \int_{\partial R_i} ds + \gamma |R_i|^c + \nu |\Theta_i| \right] \right\}
\]
LIKELIHOOD FOR IMAGES

Visual Patterns are independent stochastic processes

\[ p(I|W) = \prod_{i=1}^{K} p(I_{R_i}; \Theta_i, \ell_i) \]

- \( \ell \): model-type index \( \ell \in \{g_1, g_2, g_3, g_4\} \)
- \( \Theta_i \): model parameter vector
- \( I_{R_i} \): image appearance in i-th region
FOUR GRAY-LEVEL MODELS

Uniform          Clutter           Texture         Shading

Gaussian          Intensity         Gabor Response   B-Spline
                  Histogram         Histogram

Given parameter $\Theta$ and region image $I$, we can measure $P(I_R | \Theta_i, I_i)$ by defined distance.
WHAT DID WE JUST DO?

Def. of Segmentation:

\[ W = (K, \{(R_i, \ell_i, \Theta_i); \ i = 1, 2, \ldots, K\}) \]

Score (probability) of Segmentation:

\[ W \sim p(W|I) \propto p(I|W)p(W), \ W \in \Omega. \]

Likelihood of Image = product of region likelihoods

\[ p(I|W) = \prod_{i=1}^{K} p(I_{R_i}; \Theta_i, \ell_i) \]

Regions defined by k-partition:

\[ \Lambda = \bigcup_{i=1}^{K} R_i, \quad R_i \cap R_j = \emptyset, \ \forall i \neq j \]
WHAT DO WE DO WITH SCORES?

Given the image $I$

Search

for optimal $W$
SEARCH THROUGH WHAT?
ANATOMY OF SOLUTION SPACE

Space of all k-partitions

\[ \pi_k = (R_1, R_2, \ldots, R_k), \quad \bigcup_{i=1}^{k} R_i = \Lambda, \quad R_i \cap R_j = \emptyset, \forall i \neq j \]

General partition space

\[ \varpi = \bigcup_{k=1}^{4} \varpi_{\pi_k} \]

Space of all segmentations

\[ \Omega = \bigcup_{k=1}^{\left| \Lambda \right|} \Omega_{k} = \bigcup_{k=1}^{\left| \Lambda \right|} \varpi_{\pi_k} \times \varpi_{\Theta} \times \cdots \times \varpi_{\Theta} \]
SEARCHING THROUGH SEGMENTATIONS

Exhaustive Enumeration of all segmentations
   Takes too long!

Greedy Search (Gradient Ascent)
   Local minima!

MCMC based exploration
   Described later!
WHY MCMC

What is it: Markov chain Monte Carlo

- A clever way of searching through a high-dimensional space
- A general purpose technique of generating samples from a probability

\[ W \sim p(W|I) \propto p(I|W)p(W), \quad W \in \Omega. \]

What does it do?

- Iteratively searches through space of all segmentations by constructing a Markov Chain which converges to stationary distribution
A Markov chain is a mathematical model for stochastic systems whose states, discrete or continuous, are governed by a transition probability. The current state in a Markov chain only depends on the most recent previous states, e.g. for a 1st order Markov chain.

\[ x_t | x_{t-1}, \ldots, x_0 \sim P(x_t | x_{t-1}, \ldots, x_0) = P(x_t | x_{t-1}) \]

The Markovian property means “locality” in space or time, such as Markov random fields and Markov chain. Indeed, a discrete time Markov chain can be viewed as a special case of the Markov random fields (causal and 1-dimensional).
**What is Markov Chain Monte Carlo?**

**MCMC** is a general purpose technique for generating fair samples from a probability in high-dimensional space, using random numbers (dice) drawn from uniform probability in certain range. A Markov chain is designed to have $\pi(x)$ being its stationary (or invariant) probability.

This is a non-trivial task when $\pi(x)$ is very complicated in very high dimensional spaces!
DESIGNING MARKOV CHAINS

Three Markov Chain requirements

**Ergodic:** from an initial segmentation \( W_0 \), any other state \( W \) can be visited in finite time (no greedy algorithms); ensured by jump-diffusion dynamics

**Aperiodic:** ensured by random dynamics

**Detailed Balance:** every move is reversible

\[
P(x)P(x \rightarrow x') = P(x')P(x' \rightarrow x)
\]
5 DYNAMICS

1.) Boundary Diffusion

\[
\frac{d \Gamma_{i,j}(s)}{dt}
\]

2.) Model Adaptation

\[
\frac{d \Theta_i}{dt}
\]

3.) Split Region

4.) Merge Region

5.) Switch Region Model

At each iteration, we choose a dynamic with probability \(q(1), q(2), q(3), q(4), q(5)\)
Diffusion* within $\pi_k$ between Regions i and j.

\[
\frac{d\Gamma_{ij}(s)}{dt} = f_{prior}(s) + \log \frac{p(I(x(s), y(s)); \Theta_i, \ell_i)}{p(I(x(s), y(s)); \Theta_j, \ell_j)} + \sqrt{2T(t)}dB \nabla(s)
\]

Temperature Decreases over Time

Brownian Motion Along Curve Normal

*Movement within partition space
DYNAMICS 2: MODEL ADAPTATION

Fit the parameters* of a region by steepest ascent

\[
\frac{d \Theta_i}{dt} = \frac{\partial \log p(I_{R_i}; \Theta_i, \ell_i)}{\partial \Theta_i}.
\]

*Movement within cue space
DYNAMICS 3-4: SPLIT AND MERGE

Split one region into two

\[ W = (K, (R_k, \ell_k, \Theta_k), W_-) \iff (K + 1, (R_i, \ell_i, \Theta_i), (R_j, \ell_j, \Theta_j), W_-) = W' \]

Conditional Probability of how likely chain proposes to move to \( W' \) from \( W \)

\[
G(W \rightarrow dW') = q(3)q(R_k)q(\Gamma_{ij}|R_k)q(\ell_i)q(\Theta_i|R_i, \ell_i)q(\ell_j)q(\Theta_j|R_j, \ell_j)dW'.
\]

Probability of acceptance

\[
\alpha(W \rightarrow dW') = \min\left(1, \frac{G(W' \rightarrow dW)p(W'|I)dW'}{G(W \rightarrow dW')p(W|I)dW'} \right)
\]

Standard Metropolis-Hastings Algorithm
DYNAMICS 3-4: SPLIT AND MERGE

Merge two Regions

\[ W = (K, (R_k, \ell_k, \Theta_k), W_-) \mapsto \]

\[ (K + 1, (R_i, \ell_i, \Theta_i), (R_j, \ell_j, \Theta_j), W_-) = W' \]

Probability of Proposed Merge

\[ G(W' \rightarrow dW) = q(4)q(R_i, R_j)q(\ell_k)q(\Theta_k|R_k, \ell_k)dW \]
DYNAMICS 5: MODEL SWITCHING

Change models

\[ W = (\ell_i, \Theta_i, W_-) \leftrightarrow (\ell'_i, \Theta'_i, W_-) = W' \]

Proposal Probabilities

\[ G(W \rightarrow dW') = q(5)q(R_i)q(\ell'_i)q(\Theta'_i| R_i, \ell'_i)dW', \]
\[ G(W' \rightarrow dW) = q(5)q(R_i)q(\ell_i)q(\Theta_i| R_i, \ell_i)dW. \]
**MOTIVATION OF DD**

Region Splitting: How to decide where to split a region?

Model Switching: Once we switch to a new model or generate a new region, what parameters do we jump to?
**K-PARTITION PARTICLES**

Edge detection gives us a good idea of where we expect a boundary to be located.

A candidate region \( R_k \) is superimposed on the partition maps at three scales for computing a candidate boundary \( \Gamma_{ij} \) for the pending split.

For \( s = 1, 2, 3 \), \( \forall k \):

\[
q^{(s)}(\pi_k) = \frac{1}{|\Pi_k^{(s)}|} \sum_{j=1}^{|\Pi_k^{(s)}|} G(\pi_k - \pi_{k,j}^{(s)})
\]

\( q(\Gamma | R) \)

All k-partition in s scale

\[
q(\pi_k) = \sum_s q(s)q^{(s)}(\pi_k), \quad \forall k.
\]

Prob. of choosing scale s
PRECOMPUTE MODEL PARTICLES

Get clusters of each model, and softly assign pixel to clusters with saliency:
PRECOMPUTE MODEL PARTICLES

For a region $R, \ell$,

$$p_i = \frac{1}{|R|} \sum_{v \in R} S_{i,v}^\ell, \quad i = 1, 2, \ldots, m, \quad \forall \ell.$$

Sampling in practice:

$$\Theta \sim q(\Theta|R, \ell) = \sum_{i=1}^{m} p_i G(\Theta - \Theta_i^\ell).$$

1. Choose a model according to $p$
2. Add some small disturbance
DATA DRIVEN METHODS

Focus on boundaries and model parameters derived from data: compute these before MCMC starts

Cue Particles: Clustering in Model Space
K-partition Particles: Edge Detection

Particles Encode Probabilities Parzen Window Style
CUE PARTICLES IN ACTION

Clustering in Color Space

Input 1  Color clusters and their saliency maps $S_{i}^{C1}, i = 1, \ldots, 6$
What is this particle business about?

A particle is just the position of a parzen-window which is used for density estimation.

*Parzen Windowing also known as: Kernel Density Estimation, Non-parametric density estimation
ARE YOU AWAKE: WHAT DID WE JUST DO?

Define scores (Probability of Segmentation) → Search

5 MCMC dynamics to sample for solution → solution space is so huge

Data-Driven Speedup (key to making MCMC work in finite time)
RESULT OF DDMCMC
**TASK 2: OBJECT (FACE) DETECTION**

**I**: Grayscale image

**W**: The objects in the image

\[
W_{\text{best}} = \arg \max_W (P(W \mid I)) = \arg\max_W (P(I \mid W) \cdot P(W))
\]
HOW TO MEASURE PRIOR: $P(W)$

Example: rectangle detection

\[ \begin{array}{c}
\text{rectangle} \\
\end{array} = \begin{array}{c}
\text{type I} \\
\end{array} + \begin{array}{c}
\text{type II} \\
\end{array} \]

\[ \begin{array}{c}
\text{rectangle} \\
\end{array} = \begin{array}{c}
\text{parallel line} \\
\text{junctions} \\
\text{linelet} \\
\end{array} + \begin{array}{c}
\text{circles} \\
\text{or node} \\
\end{array} \]
**HOW TO MEASURE PRIOR: \( P(W) \)**

Example: face detection

AoG representation of human face

Integrating \( \alpha, \beta, \gamma \) inference processes in face detection

- **\( \alpha \) (head-shoulder)**
- **\( \gamma \) (face)**
- **\( \beta \) (face)**
AND OR GRAPH PRIOR

Define the And/Or graph (AOG)

Each object is represented by a parse graph, which is an instance of the AOG by selecting parameter for each node.

\[ \mathcal{E}(pg) = - \sum_{<O,A> \in E_{or}^{pg}} \log p(A|O) \]

Or node

\[ - \sum_{<P,A> \in E_{dec}^{pg}} \log p(X(A)|X(P)) \]

And node: top down

\[ - \sum_{<C_i,C_j> \in E_{rel}^{pg}} \log p(X(C_i), X(C_j)) \]

And node: bottom up

All these probability distribution can be learned from training data
AND OR GRAPH LIKELIHOOD

\[ \log(P(I \mid W)) \]

The scores of part detector, trained separately.

**Inference:**

1. Find activation of each part
2. Convert head shoulder detection to face detection, merge if possible
3. Convert eye/nose/mouth detection, merge if possible
4. Select the top ones
WAKE UP

What have we learned up to now:

Bayesian framework: \[ P(W \mid I) \propto P(I \mid W) \cdot P(W) \]

A framework inferring model base on observation

1. likelihood: observation related
2. prior: prior knowledge about the problem

AOG: a graph model of prior knowledge

Inference:

1. If space of W is large: sampling or greedy search
2. or: sacrifice model and get global optima
AOG models relation between parts with graph

Problem: hand-coded context
1. miss the useful context in prior knowledge
2. add useless context and hard to calibrate

More flexible way of handling context information:

Auto Context
AUTO CONTEXT

Task: Binary segmentation

Method: Take feature from each pixel. Train a binary classifier to make decision for each pixel. (data unit: pixel)
**AUTO CONTEXT: IDEA**

Idea: Heavy use of context and let classifier choose

1. use feature from nearby region uniformly
2. select feature by decision tree
3. use response of weak classifier

Idea: Heavy use of context and let classifier choose

1. use feature from nearby region uniformly
2. select feature by decision tree
3. use response of weak classifier
AUTO CONTEXT

The context information in clarifying the ambiguities. This demonstrates the importance of using context features with the rest being the pixel for any label is similar to auto-context. The only difference is that if any to the auto-context algorithm, called auto-context-th which on a particular sample, then it is probably a waste to that if the previous classifier has already made a firm decision on the appearance, rather than probabilities, of the context pixels. Several observations we can make from this figure: (1) the appearance context does not improve the result (even slightly worse); (4) The second stage of the auto-context algorithm significantly improve the results over patch-based classification methods; (2) auto-context algorithm is widely used in computer vision, we illustrate the auto-context algorithm using a cascade of AdaBoost model algorithm [11]. Training takes about half a day for (4) the performance is slightly worse than auto-context using cascade and a couple of days for auto-context using PBT, with both having 5 stage of classifiers. The training time of auto-context-th does appear not choose any context features as they are not informative for any label. Starting from the second classifier, nearly if the previous classifier has already made a firm decision, as suggested in [28], to train the system with the high-recall area. The Weizmann dataset contains one horse in each image for horse segmentation. (b) gives the precision-recall curves by different algorithms for any label. Figure 3. (a) shows the training and test errors at different stages of auto-context; (b) shows the precision-recall curves for the different stages of the PBT algorithm, our algorithm outperforms many the existing algorithms reported so far [17, 11, 3, 26]. Also, auto-context algorithm is very general and easy to implement. There is no need to design specific features, which makes the system directly protable to a variety of other applications. However, pose is not to design a specific horse segmentation algorithm, we apply it on another problem, human body context algorithm trained on the Weizmann dataset [3]. (b) is the confusion matrix key word "horses". The second row displays the final probability map by the auto-context algorithm. The last row shows two images with the worst scores. As we can see, even these results are not too bad. Though our pursuit feature design by human intelligence is still a very interesting and promising direction.
BETTER FEATURE? BETTER CLASSIFIER?

Problem of auto-context:

1. may still lost important context information
2. weak feature: pixel value
3. no weight on context, heavily relies on classifier

To get powerful context information from SEGMENTATION

1. define extent of region for context
2. more informative than a single pixel
3. take reliability of segments as weight
GEOMETRIC CONTEXT

Task: Geometric Labeling (pixel-wise labeling)
GEOMETRIC CONTEXT

How to get context information?

How to get? Training
GEOMETRIC CONTEXT

Generate multiple hypotheses (segmentation) on training image, and label them as “ground”, “sky”, “vertical”, and “mixed”.

Compute feature for each hypothesis

Train adaboost (w. decision tree)

\[ P(v \mid h, x) : \text{score}(v), v = \text{“ground”}, \text{“sky”}, \text{“vertical”} \]

\[ P(h \mid x) : \text{score}(v), v = \text{“mixed”} \]

\[ C(y_i = v \mid x) = \sum_{j} P(y_j = v \mid x, h_{ji}) P(h_{ji} \mid x) \]
GEOMETRIC CONTEXT

Figure 4: Results on images representative of our data set. Two columns of {original, ground truth, test result}. Colors indicate the main class label (green=ground, red=vertical, blue=sky), and the brightness of the color indicates the confidence for the assigned test labels. Markings on the vertical regions indicate the assigned subclass (arrows indicate planar orientations, "X"=non-planar solid, "O"=non-planar porous). Our system is able to estimate the geometric labels in a diverse set of outdoor scenes (notice how different orientations of the same material are correctly labeled in the top row). This figure is best viewed in color.
GEOMETRIC CONTEXT: AUTO IMAGE POP UP

Pop up card
How to do it automatically?

Input image

GC

Fit polylines

3D model

Cut and fold
GEOMETRIC CONTEXT: AUTO IMAGE POP UP

Input

Labels

Novel View

Novel View

Input

Labels

Novel View

Novel View

Input

Labels

Novel View

Novel View

Input

Labels

Novel View

Novel View

Input

Labels

Novel View

Novel View

Input

Labels

Novel View

Novel View

Input

Labels

Novel View

Novel View
CAN WE GET MORE DETAILS?

From single image to more detailed reconstruction
MAKE3D: A MORE DETAILED CONSTRUCTION

With the same technique as GC, we can estimate the 3D position and orientation of each superpixel.

![Input image](image1.png) ![Estimated depth map](image2.png)

**Input image**  **Estimated depth map**
MAKE3D: A MORE DETAILED CONSTRUCTION

Pairwise constraint:
Connectivity
Co-planar
Co-linear

Input image
Estimated depth map
MRF optimization
MAKE3D: A MORE DETAILED CONSTRUCTION

![Image](image1.png)

**Fig. 11.** (a) Original Image, (b) Ground truth depthmap, (c) Depth from image features only, (d) Point-wise MRF, (e) Plane parameter MRF. (Best viewed in color.)

![Image](image2.png)

**Fig. 12.** Typical depthmaps predicted by our algorithm on hold-out test set, collected using the laser-scanner. (Best viewed in color.)

![Image](image3.png)

**Fig. 13.** Typical results from our algorithm. (Top row) Original images, (Bottom row) depthmaps (shown in log scale, yellow is closest, followed by red and then blue) generated from the images using our plane parameter MRF. (Best viewed in color.)

In addition, we manually labeled 50 images with 'ground-truth' boundaries to learn the parameters for occlusion boundaries and folds.

**B. Results and Discussion**

We performed an extensive evaluation of our algorithm on 588 internet test images, and 134 test images collected using the laser scanner.

In Table I, we compare the following algorithms:

- **(a) Baseline:** Both for pointwise MRF (Baseline-1) and plane parameter MRF (Baseline-2). The Baseline MRF is trained without any image features, and thus reflects a "prior" depthmap of sorts.
- **(b) Our Point-wise MRF:** with and without constraints (connectivity, co-planar and co-linearity).
- **(c) Our Plane Parameter MRF (PP-MRF):** without any constraint, with co-planar constraint only, and the full model.
- **(d) Saxena et al. (SCN), [6], [21] applicable for quantitative errors only.
MAKE3D: A MORE DETAILED CONSTRUCTION

Fig. 15. Typical results from our algorithm. Original image (top), and a screenshot of the 3-d flythrough generated from the image (bottom of the image). The 11 images (a-g, l-t) were evaluated as "correct" and the 4 (h-k) were evaluated as "incorrect."
GEOMETRIC CONTEXT: DETECTION

Refine object detection:

(a) Local Features Only  (b) Geometric Labels  (c) With Context

Figure 5: Failure examples. Two columns of {original, ground truth, test result}. Failures can be caused by reflections (top row) or shadows (bottom-left). At the bottom-right, we show one of the most dramatic failures of our system.

Figure 6: In difficult cases, every cue is important. When any set of features (d-g) is removed, more errors are made than when all features are used (b). Although removing location features (f) cripples the classifier in this case, location alone is not sufficient (c).

Figure 7: Improvement in Murphy et al. [20]’s detector with our geometric context. By adding a small set of context features derived from the geometric labels to a set of local features, we reduce false positives while achieving the same detection rate. For a 75% detection rate, more than two-thirds of the false positives are eliminated. The detector settings (e.g. non-maximal suppression) were tuned for the original detector.

Figure 8: Original image used by Liebowitz et al. [17] and two novel views from the scaled 3D model generated by our system. Since the roof in our model is not slanted, the model generated by Liebowitz, et al. is slightly more accurate, but their model is manually specified, while ours is created fully automatically [12]!

8
UNDERSTANDING AN IMAGE
TODAY: LOCAL AND INDEPENDENT
WHAT THE DETECTOR SEES
LOCAL OBJECT DETECTION

True Detection

False Detections

Missed

Local Detector: [Dalal-Triggs 2005]
WHAT DOES SURFACE AND VIEWPOINT SAY ABOUT OBJECTS?

Image

P(surfaces)

P(viewpoint)

P(object)

P(object | surfaces)

P(object | viewpoint)
OBJECT SIZE IN THE IMAGE

Let $y_{i}$ denote the object image height, $h_{i}$ the object 3D height, $y_c$ the camera height, $v_i$ the foot image position, and $v_0$ the horizon image position. The relationship between these variables is given by:

$$y_i = \frac{h_i y_c}{v_i - v_0}$$
WHAT DOES SURFACE AND VIEWPOINT SAY ABOUT OBJECTS?

P(object | surfaces, viewpoint)

Image

P(surfaces)

P(viewpoint)

P(object)

P(object | surfaces, viewpoint)
SCENE PARTS ARE ALL INTERCONNECTED
A DETECTION FRAMEWORK

This context model between: object, geometric, viewpoint can be used with any object detection system.

Pedestrian Detection

Local Detector from [Murphy-Torralba-Freeman 2003]

Pedestrian Detection

Local Detector: [Dalal-Triggs 2005] (SVM-based)
Hallucinated Humans as the Hidden Context for Labeling 3D Scenes

Yun Jiang, Hema Koppula and Ashutosh Saxena
Department of Computer Science, Cornell University.
{yunjiang,hema,asaxena}@cs.cornell.edu

Abstract
For scene understanding, one popular approach has been to model the object-object relationships. In this paper, we hypothesize that such relationships are only an artifact of certain hidden factors, such as humans. For example, the objects, monitor and keyboard, are strongly spatially correlated only because a human types on the keyboard while watching the monitor. Our goal is to learn this hidden human context (i.e., the human-object relationships), and also use it as a cue for labeling the scenes. We present Infinite Factored Topic Model (IFTM), where we consider a scene as being generated from two types of topics: human configurations and human-object relationships. This enables our algorithm to hallucinate the possible configurations of the humans in the scene parsimoniously. Given only a dataset of scenes containing objects but not humans, we show that our algorithm can recover the human object relationships. We then test our algorithm on the task of attribute and object labeling in 3D scenes and show consistent improvements over the state-of-the-art.

1. Introduction
We make the world we live in and shape our own environment.

Orison Swett Marden (1894).

For reasoning about cluttered human environments, for example in the task of 3D scene labeling, it is critical we reason through humans. Human context provides a natural explanation of why the environment is built in particular ways. Specifically, consider the scene in Fig. 1, with a chair, table, monitor and keyboard. This particular configuration that is commonly found in offices, can be naturally explained by a sitting human pose in the chair and working with the computer. Moreover, from the point of view of modeling and learning, this explanation is parsimonious and efficient as compared to modeling the object-object relationships \[1\] such as chair-keyboard, table-monitor, monitor-keyboard, etc.

For \(n\) objects, we only need to model how they are used by humans, i.e., \(O(n)\) relations, as compared with modeling \(O(n^2)\) if we were to model object to object context naively.

Figure 1: Left: Previous approaches model the relations between observable entities, such as the objects. Right: In our work, we consider the relations between the objects and hidden humans. Our key hypothesis is that even when the humans are never observed, the human context is helpful.

In fact, several recent works have shown promise in using human and object affordances to model the scenes. Jiang, Lim and Saxena \[14, 17\] used hallucinated humans for learning the object arrangements in a house in order to enable robots to place objects in human-preferred locations. However, they assumed that the objects have been detected. Our goal in this work is different, where we start with 3D point-clouds obtained from RGB-D sensors and label them using their shape, appearance and hallucinated human context. Gupta et al. \[10\] proposed predicting stable and feasible human poses given an approximate 3D geometry from an image. While inspired by these prior works, the key idea in our work is to hallucinate humans in order to learn a generic form of object affordance, and to use them in the task of labeling 3D scenes. While a large corpus of scenes with objects is available, humans and their interactions with objects are observed only a few times for some objects. Therefore, using hallucinated humans gives us the advantage of considering human context while not limited to data that contains real human interactions.

However, if the humans are not observed in the scene and we do not know the object affordances either (i.e., how humans use objects), then learning both of them is an ill-posed problem. For example, one trivial, but useless, solution would be having one human configuration for every object in the scene. The key idea in our work is to prefer parsimony in our model as follows. First, while the space of potential unobserved human configurations are large, only one
HUMAN-OBJECT CONTEXT

1. Label all objects
2. Infer a human pose interacting with objects
3. Refine object labeling base on the inferred human pose
GEOMETRIC CONTEXT: ROOM LAYOUT

Task: Estimate a box for the room

Clutter is a big problem
ROOM LAYOUT ESTIMATION

The # of possible room layout is huge, so sampling first.

Vanishing point
HYPOTHESIS GENERATION

Uniformly sample rays from two vanishing points

Two rays from each vp (4 rays) can generate a room layout hypothesis

The next step: select the best one
**STRUCTURE SVM**

Goal: \[ y^* = \arg\max_y f(x, y) = \arg\max_y \omega^T \phi(x, y) \]

Training:
\[
\min_w, \xi \frac{1}{2} ||w||^2 + C \sum_i \xi_i \\
\text{s.t. } \xi_i \geq 0 \forall i, \text{ and} \\
w^T \psi(x_i, y_i) - w^T \psi(x_i, y) \geq \Delta(y_i, y) - \xi_i \\
\forall i, \forall y \in \mathcal{Y} / y_i
\]

Line seg. belong to the two vanishing points of \( F_k \)

Feature:
\[
f_l(F_k) = \frac{\sum_{l_j \in C_k} |l_j|}{\sum_{l_j \in L_k} |l_j|}
\]

Line segments in \( F_k \)
WHERE IS GC?

Use Geometric context to label pixel with label: “left wall”, “right wall”, “middle wall”, “ceiling”, “floor”, and “object”.

Reduce the weight of line segments in “object” region.
PROBLEM

Room layout hypotheses are sampled uniformly, which cannot guarantee global (or even local) minimum.

What should we have to get global minimum?

1. a quick way of computing feature
   “Integral Geometry”

2. a quick way of searching
   “Branch and bound”
INTEGRAL GEOMETRY

Suppose we have a pixel-wise feature map $f(x, y)$, and

The feature of any box region equals to the sum of features of included pixels,

$$F(u_1, v_1, u_2, v_2) = \sum_{x=u_1}^{u_2} \sum_{y=v_1}^{v_2} f(x, y)$$

Compute feature for $M$ boxes with $N$ pixels would cost $O(MN)$

How to compute feature for any box in constant time? $O(M)$
1. Compute integral image in \(O(N)\)

\[ I(u, v) = \sum_{x=1}^{u} \sum_{y=1}^{v} f(x, y) \]

2. Compute feature for box region:

\[ F(u_1, v_1, u_2, v_2) = I(u_2, v_2) - I(u_1 - 1, v_2) - I(u_2, v_1 - 1) + I(u_1 - 1, v_1 - 1) \]
BRANCH AND BOUND

Task: Search for the rectangle give the best score

Branching. Dividing a space of candidate rectangles into subspaces

Bounding. Pruning subspaces with a highest possible score lower than some guaranteed score in other subspaces
To use branch-and-bound for given quality function $f$, we need to define upper bound function $\hat{f}$

1. $\hat{f}(\mathcal{R}) \geq \max_{R \in \mathcal{R}} f(R),$

2. $\hat{f}(\mathcal{R}) = f(R)$, if $R$ is the only element in $\mathcal{R}$.

Computation of upper bound has to be efficient to make the search efficient.
Assume the previous definition of box feature, and all dimension of feature are positive.

We trained a linear SVM

\[ B^* = \arg \max_B \omega^T F(B) \]

**Upper bound function:**

If \( R_{\text{max}} \) the largest rectangle and by \( R_{\text{min}} \) the smallest rectangle contained in a parameter region \( R \),

\[ \hat{f}(R) := f^+(R_{\text{max}}) + f^-(R_{\text{min}}) \]

All dimension with positive weight

\[ f(R) = f^+(R) + f^-(R) \]

\[ f^+(R_{\text{max}}) > f^+(R) \]

\[ f^-(R_{\text{min}}) > f^-(R) \]
Exactly the same as object detection.

Feature:

1. each pixel: geometric context trained before (all positive).
2. each wall: sum of feature of pixel inside
3. room layout: concatenation of feature of all walls

Feature of a region: integral image
ROOM LAYOUT EASTIMATION

Set of room layout: 4 ranges

Define maximal and minimal region for each wall

Branch and Bound

Usually 20 times faster, and slightly better

\[ f^+(x, y) = \sum_{(i, \alpha, r): i \in \{o, g\}, \alpha \in \mathcal{F}, w_i, \alpha, r > 0} w_i, \alpha, r \phi_i, \alpha, r(x, y_\alpha), \]

\[ f^-(x, y) = \sum_{(i, \alpha, r): i \in \{o, g\}, \alpha \in \mathcal{F}, w_i, \alpha, r \leq 0} w_i, \alpha, r \phi_i, \alpha, r(x, y_\alpha). \]

\[ \hat{y} = \arg\max_{y \in \mathcal{Y}} f^+(x, y) + f^-(x, y) \]
EXAMPLE: CONTEXT + BRANCH&BOUND

Task: Fit cuboids on RGBD image

— Diagram of a bedroom scene with cuboids overlaid on the RGBD image —
CUBOID MATCHING IN RGBD IMAGE

Fit cuboid directly from image & 3D information

How to find those correct ones?
Minimizing error will give trivial empty solution.
CUBOID MATCHING IN RGBD IMAGE

Context information for regularization:

1. surface coverage
   - the fitted cuboids should cover a large region.

2. volume exclusion
   - two cuboids shall not intersect, or the intersection should be minimized.

3. occlusion constraints
   - a cuboid shall not be fully occluded by others

4. small # of cuboids
Context Model: Mixed IP

\[
\min_x \left\{ U(x) + \lambda P(x) + \mu N(x) - \gamma A(x) + \xi O(x) \right\}
\]

\[
\min_z \min c^T x \quad y_k + \xi \sum_{i,j} q_{i,j} w_{i,j}
\]

s.t. \quad \forall \{i,\}
\quad y_k \leq x_i
\quad w_{i,j} \leq x_i
\quad \forall \{i, j\} \text{ that } x_i + x_j \leq
\quad x_i = 0 \text{ or } 1. \text{ All variables are nonnegative.}

Branch and Bound Mixed Integer Programing
Global Optimal Solution in 10 seconds (Code Available)
CONTEXT MODEL: MIXED IP

Branch and Bound:
1. Fix one $x_i$ to be 0 or 1
2. Lower bound function

$$f(X) = \min_c c^T x,$$
$$s.t. Ax < b,$$
$$x_i \in \{0,1\}, i \in I_z,$$
$$x_m = 0, m \in I_0, x_n = 1, n \in I_1,$$

$$\hat{f}(X) = \min_c c^T x,$$
$$s.t. Ax < b,$$
$$x_i \in \{0,1\}, i \in I_z,$$
$$x_m = 0, m \in I_0, x_n = 1, n \in I_1,$$

Typical linear programming
Call MATLAB