# Seminar: A Probabilistic Programming Language for Scene Perception

Yu-Zhe Shi

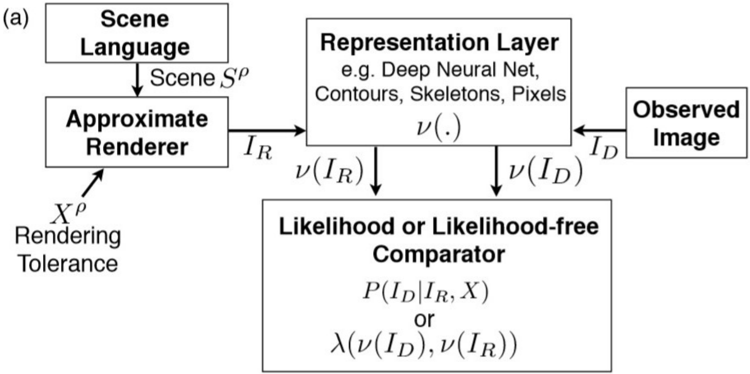
May 3, 2020

# Auther Profiles

- Joshua B. Tenenbaum, Prof, The Computational Cognitive Science Group, MIT.http://web.mit.edu/cocosci/josh.html
- Tejas D. Kulkarni, PhD, The Computational Cognitive Science Group, MIT.https://tejasdkulkarni.github.io/
- Pushmeet Kohli, Microsoft Research.https://sites.google.com/site/pushmeet/
- Vikash Mansinghka, Prof, Computer Science and Artificial Intelligence Lab, MIT.https://mit.academia.edu/VikashMansinghka

# Framework Overview

- Objective: Find a image scene from hypothesis sapce: $P(S^\rho | I_D)$



# Picture Language Overview

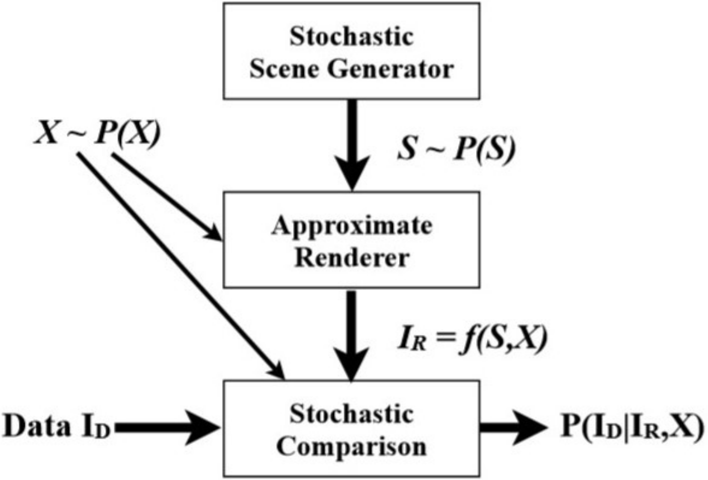
```
function PROGRAM(MU, PC, EV, VERTEX_ORDER)
  # Scene Language: Stochastic Scene Gen
  face=Dict();shape = []; texture = [];
  for S in ["shape", "texture"]
   for p in ["nose", "eyes", "outline", "lips"]
    coeff = MvNormal(0,1,1,99)
    face[S][p] = MU[S][p]+PC[S][p].*(coeff.*EV[S][p])
   end
  end
  shape=face["shape"][:]; tex=face["texture"][:];
  camera = Uniform(-1,1,1,2); light = Uniform(-1,1,1,2)

  # Approximate Renderer
  rendered_img= MeshRenderer(shape,tex,light,camera)

  # Representation Layer
  ren_ftrs = getFeatures("CNN_Conv6", rendered_img)

  # Comparator
  #Using Pixel as Summary Statistics
  observe(MvNormal(0,0.01), rendered_img-obs_img)
  #Using CNN last conv layer as Summary Statistics
  observe(MvNormal(0,10), ren_ftrs-obs_cnn)
end
```

# Motivation: dilemma of conventional generative models

- Problem-specific, need hand-crafted engineering.
- Poor scalability, data-dependence.

# Generative Probabilistic Graphics Programs

- Refer to: *Approximate Bayesian Image Interpretation using Generative Probabilistic Graphics Programs*, NIPS'13.



# Generative Probabilistic Graphics Programs

- Map a image as

$$I_R = f(S, X) \tag{1}$$

where $S$ is scene and $X$ is variables controlling the fidelity of the renderer.

- Image interpretation as sampling posterior of image

$$P(S|I_D) \propto P(S)P(X)\delta_{f(S,X)}(I_R)P(I_D|I_R, X) \tag{2}$$

- Decomposition of random variables $X$ into $X_j$ with priors $P(X_j)$:

$$P(S) = \prod_i S_i,\ q(S_i', S_i) = P(S_i'),\ P(X) = \prod_j X_j,\ q(X_j', X_j) = P(X_j') \tag{3}$$

# Metrobolis-Hastings

- Proposal kernel $q((S,X) \to (S',X'))$ for re-render $I'_R = f(S',X')$

$$q((S,X) \to (S',X')) = \delta_{S_{-i}}(S')P(S'_i)\delta_X(X') \tag{4}$$

- Calculate accept ratio $\alpha_{MH}((S,X) \to (S',X'))$ by MCMC

$$\min\left(1, \frac{P(I_D|f(S',X'),X')P(S')P(X')q((S',X') \to (S,X))}{P(I_D|f(S,X),X)P(S)P(X)q((S,X) \to (S',X'))}\right) \tag{5}$$

# Picture Program

- A Picture program:
$$f : S \rightarrow I_R \tag{6}$$
- Program traces
$\rho = \{\rho_i\} \in \{Multinomial, Uniform, Poisson, Gaussian\}$, which specifies a generated rendering respectively.

# Sence Language

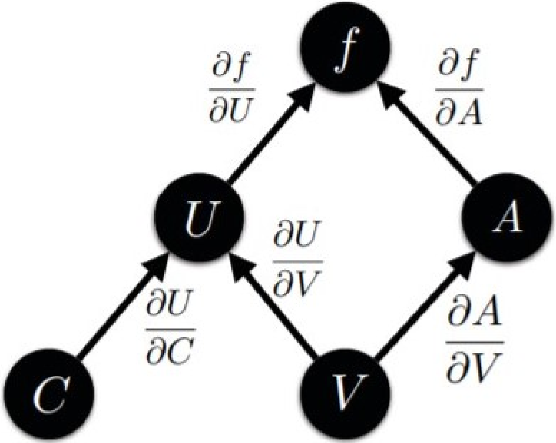
- Description of 3D object, e.g. z-map, mesh, volumetric; camera information, illumination.
- Expressed as probabilistic code.

Scene Representation $S$:

```
light_source { <0, 199, 20>
               color rgb<1.5,1.5,1.5> }
camera { location <30,48,-10> angle 40
         look_at <30,44,50> }

object{leg-right vertices ...
       trans  <32.7,43.6,9>}
object{arm-left vertices scale 0.2
       ... rotate x*0}
...
object{arm-left texture}
```

# Approximate Renderer

- Rendering tolarance $X^{\rho}$ adds structured noise.
- Refer to: *OpenDR: An Approximate Differentiable Renderer*, ECCV'14.



# Representation Layer

- Hierachical abstract features, e.g. contour, keypoint. Not only used for mapping the two exemplars into a same embedding space, but also serves as dimensional reduction.
- $v(I_D; \theta_v)$, $v(I_R; \theta_v)$ shares the same parameters of original image $I_D$ and rendered image $I_R$,

# Discriminator

- Calculate likelyhood $L = P(I_D | I_R)$.
- When the likelyhood function is not closed, exploit metrics to measure similarity
$$\lambda(v(I_D), v(I_R)) \tag{7}$$

# Details of Discriminator

- Likelyhood-free similarity, let $\varepsilon \in X^{\rho}$ be addition tolarance variables of original image.

$$\max\left(1, \frac{P_{\varepsilon}(v(I_D) - v(I'_R))P(S'^{\rho})P(X'^{\rho})q((S', X') \to (S, X))}{P_{\varepsilon}(v(I_D) - v(I_R))P(S^{\rho})P(X^{\rho})q((S, X) \to (S', X'))}\right) \tag{8}$$

# Inputs

| Modules | Functional Description |
|---|---|
| Scene Representation $S$: | |

```
light_source { <0, 199, 20>
               color rgb<1.5,1.5,1.5> }
camera { location <30,48,-10> angle 40
         look_at <30,44,50> }

object{leg-right vertices ...
       trans  <32.7,43.6,9>}
object{arm-left vertices scale 0.2
       ... rotate x*0}
...
object{arm-left texture}
```

| | |
|---|---|
| Program trace: | $\rho = \{\rho_i\}$ |
| Rendering tolerance: | $X^\rho \in \rho$ |
| Stochastic Scene: | $S^\rho \in \rho$ |
| Approximate Rendering: | $I_R$ |
| Approximate Renderer: | $render : (S, X) \rightarrow I_R$ |
| Image data: | $I_D$ |
| Data-driven Proposals: | $(f, T, \nu_{dd}, \theta_{\nu_{dd}}) \rightarrow q_{data}(.)$ |
| Data representations: | $\nu(I_D)$ and $\nu(I_R)$ |
| Comparator: | $\lambda : (\nu(I_D), \nu(I_R)) \rightarrow \mathbb{R}$ <br> $P(\nu(I_D) \vert \nu(I_R), X)$ |
| Rendering Differentiator: | $\nabla_{S^\rho_{real}} : \rho \rightarrow$ <br> $grad\_model\_density(S_{real}; I_D)$ |

# Inference

- Given program trace $\rho$, update $q((S^\rho, X^\rho) \to (S'^\rho, X'^\rho))$ until convergence.
- $K = |\{S^\rho\}| + |\{X^\rho\}|$
- Estimate image state from the entire search space $\Theta$ of $I_R$

$$P(S^\rho | I_D) \propto \int_\Theta P(S^\rho) P_\varepsilon(v(I_D) - v(I_R)) P(I_R | S^\rho) dI_R \tag{9}$$

- Exploit MCMC accept ratio

$$\min\left(1, \frac{L' P(S'^\rho) P(X'^\rho) q((S', X') \to (S, X)) K P(S^\rho_{del})}{L P(S^\rho) P(X^\rho) q((S, X) \to (S', X')) K' P(S^\rho_{new})}\right) \tag{10}$$

# Proposal Kernels

- Proposal kernel $q$ is defined from priori:

$$q((S^\rho, X^\rho) \rightarrow (S'^\rho, X'^\rho)) = \prod_{\rho'_i \in (S^\rho, X^\rho)} P(\rho'_i) \tag{11}$$

- Gradient Proposal: $\nabla S_{real} \rho,\ S_{real} \in S^\rho$,

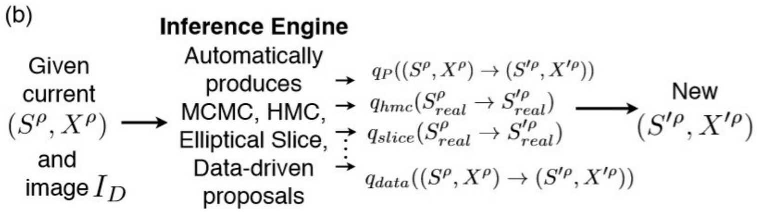
$$q_{hmc}(S^\rho_{real} \rightarrow S'^\rho_{real}) \tag{12}$$

- Elliptical Slice Proposal:

$$S'_{real} = \sqrt{1-\alpha^2} S_{real} + \alpha\theta,\ \theta \sim \mathcal{N}(0, \Sigma) \tag{13}$$

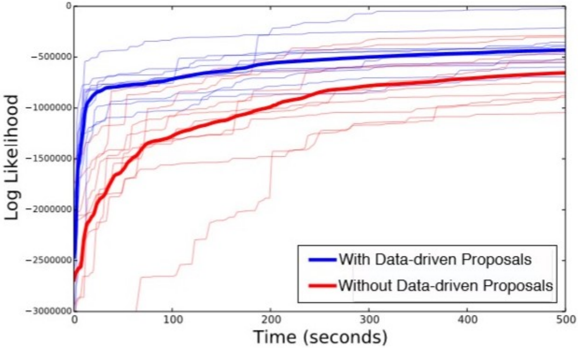
- Data-driven Proposal: fine-tune the representation layer supervised by labelled data.

# Inference Engine Pipeline



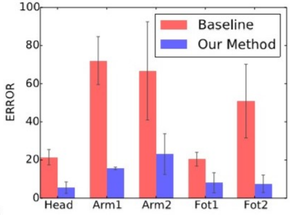
# Experiment 1

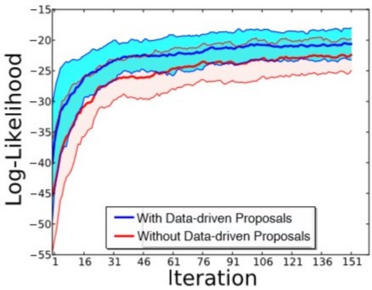
- 3D Face Analysis.



# Experiment 2

- 3D Human Pose Estimation. (Baseline: Deformable Parts Model)



# Experiment 3

- 3D Object Reconstruction.

| Quantitative Metrics | | |
|---|---|---|
| METHOD | Z-MAE | N-MSE |
| Barron et al.[2] | 15.19 | $2.407 \times 10^{-3}$ |
| Picture | 11.40 | $1.704 \times 10^{-3}$ |

# Summary and Inspiration

- Generative Model: Variations on image scenes.
- Markov Chain Monte Carlo: calculating posteriori in essential.
- Approximate Bayesian Calculating: approximate likelyhood-free posterior.