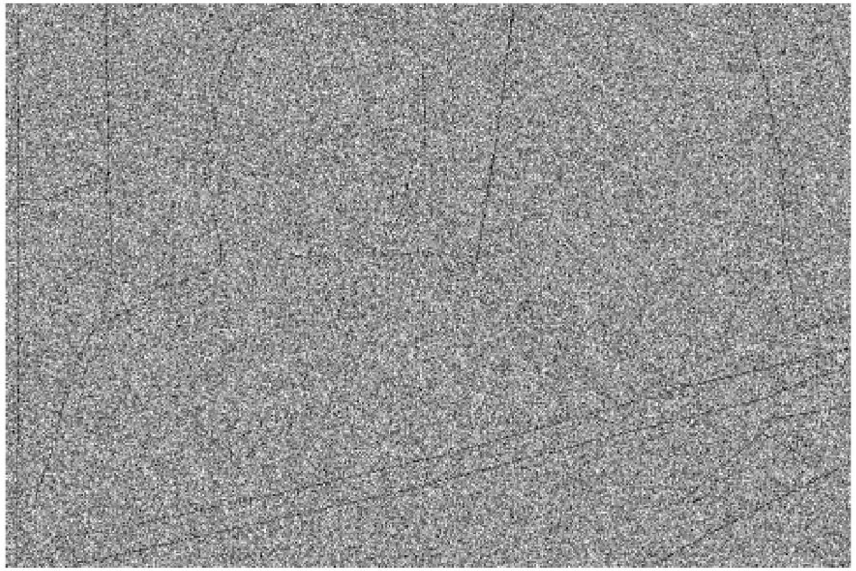
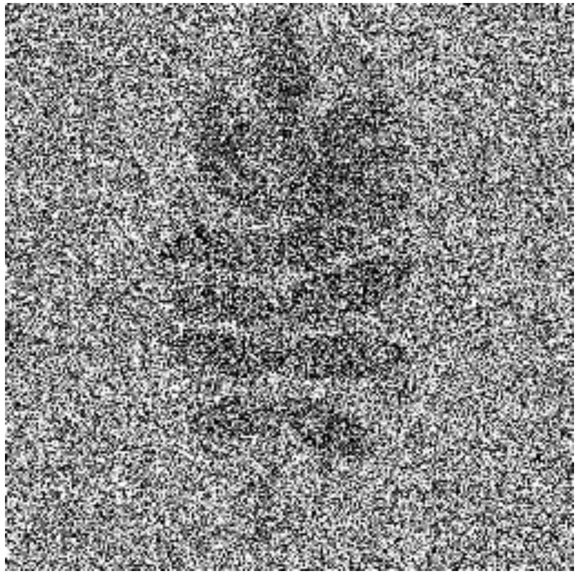
Probabilistic Scene Grammars: A General-Purpose Framework for Scene Understanding

Jeroen Chua

Advisor: Pedro Felzenszwalb





Why a general-purpose framework?

- Improvements can boost performance on many scene understanding tasks simultaneously
- Less engineering/research work for future scene understanding tasks
- Scene understanding tasks inform each other
- Scientifically interesting: can all (most) of scene understanding be understood as the same fundamental problem?

This thesis

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Notes** on other possible inference algorithms
- **Connections** to related work
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

This talk

- Motivation for a general scene understanding framework
- Background/related work
- Representation for general scene understanding tasks
- Efficient approximate inference algorithm
- ~~Notes on other possible inference algorithms~~
- ~~Connections to related work~~
- Learning algorithm to estimate model parameters
- Experimental evaluation
- Extensions for larger/more complex tasks
- Directions for future research

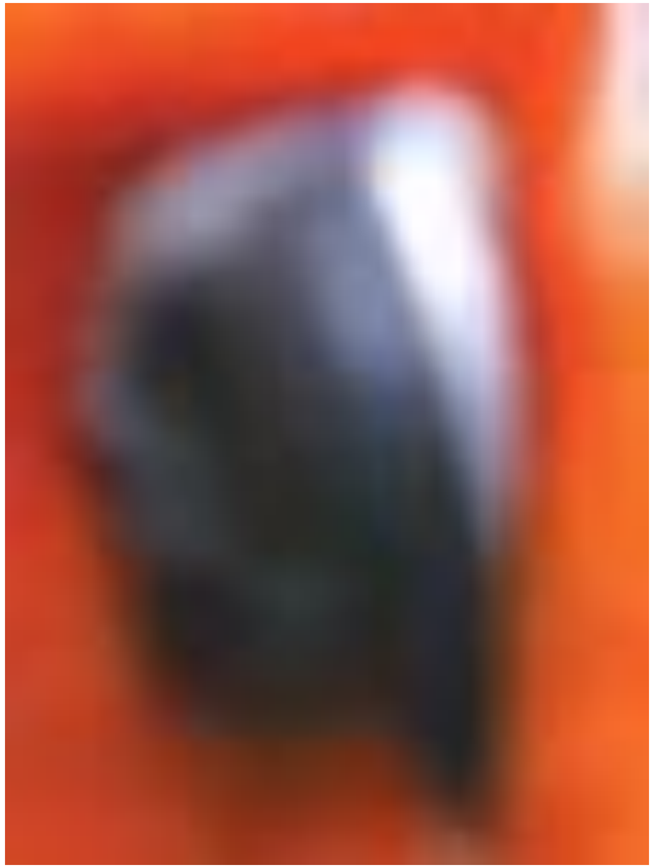
This talk

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

Contextual information is often useful

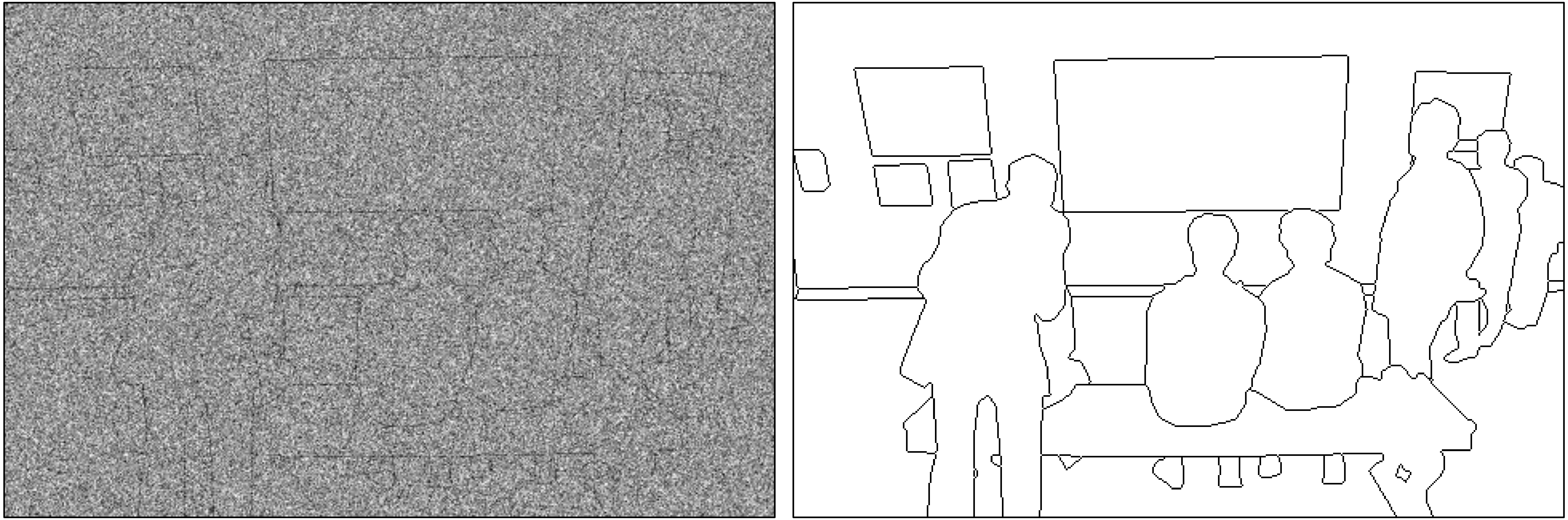
Context helps for ___________________

Context helps for **object recognition**

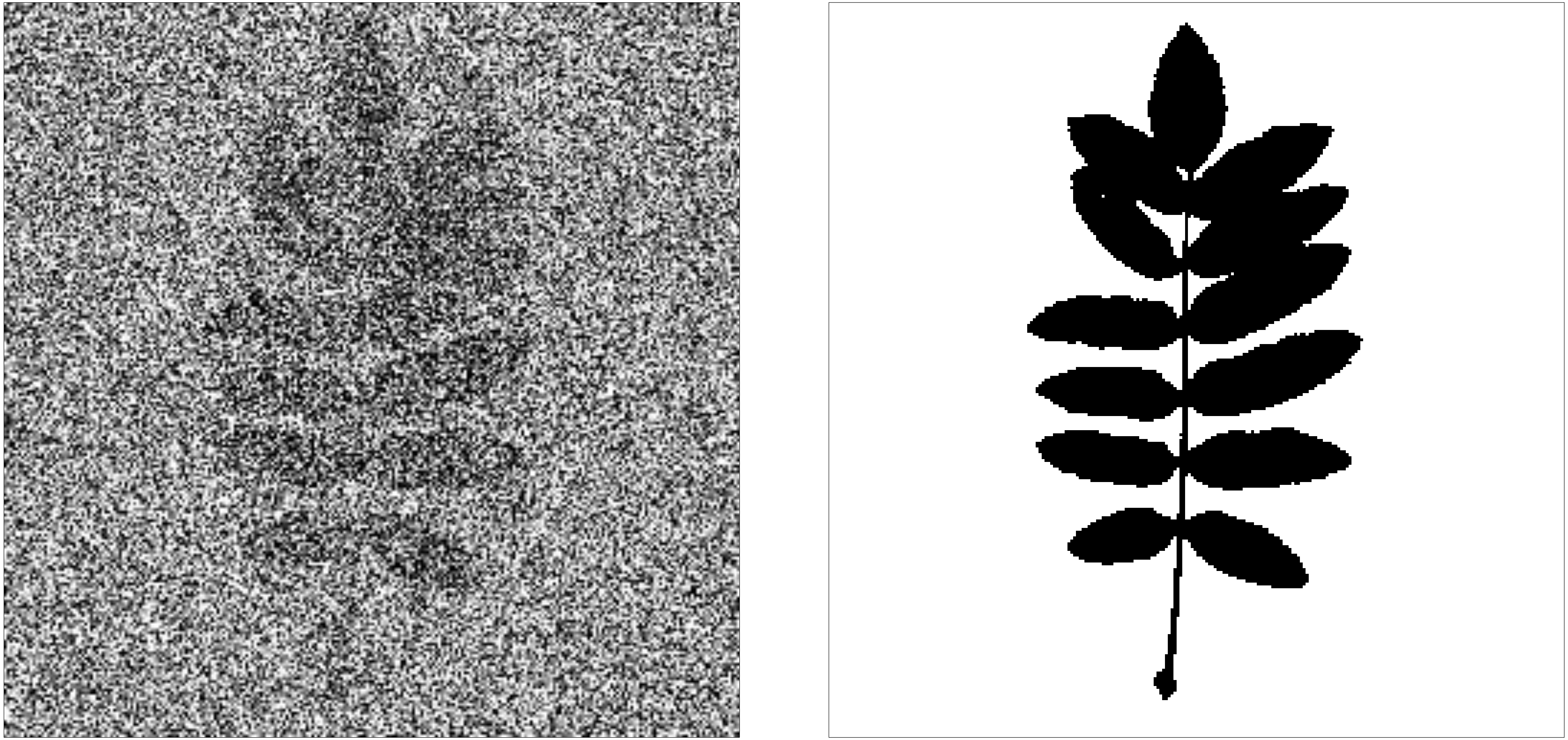


Context helps for **object recognition**

Context helps for **contour detection**



Context helps for **image segmentation**



Related work

- Contextual information
  - Oliva and Torralba[1,2], Efros [3]
  - Empirical studies [4]
  - Gestalt Theory [5]
- Motivation
  - Probabilistic Programming Languages [6,7,8]
- Modelling
  - Pictorial Structures [8]
  - DPM [9]
  - “Markov backbone” model [10]
- Probabilistic inference for compositional models
  - Markov-chain Monte Carlo
  - Heuristics

[1] “Modelling the shape of the scene: a holistic representation of the spatial envelope”, IJCV 2001

[2] “The role of context in object recognition”, Trends in Cognitive Sciences, 2007

[3] “Unsupervised visual representation learning by context prediction”, ICCV 2015

[4] “An empirical study of context in object detection”, CVPR 2009

[5] Vision science: Photons to phenomenology, volume 1. MIT press, 1999

[6] “Picture: A probabilistic programming language for scene perception”, CVPR 2015

[7] “Edward: A library for probabilistic modeling, inference, and criticism”, arXiv 2016

[8] “The design and Implementation of Probabilisitic Programming Languages”, http://dippl.org 2014

[9] “Pictorial Structures for object recognition”, IJCV 2005

[10] “Object detection with grammar models”, NIPS 2011

[11] “Context and hierarchy in a probabilistic image model”, CVPR 2006

This talk

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

This talk

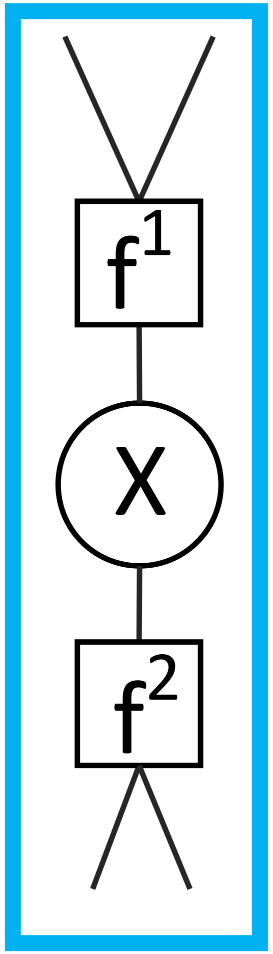
- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

PSG

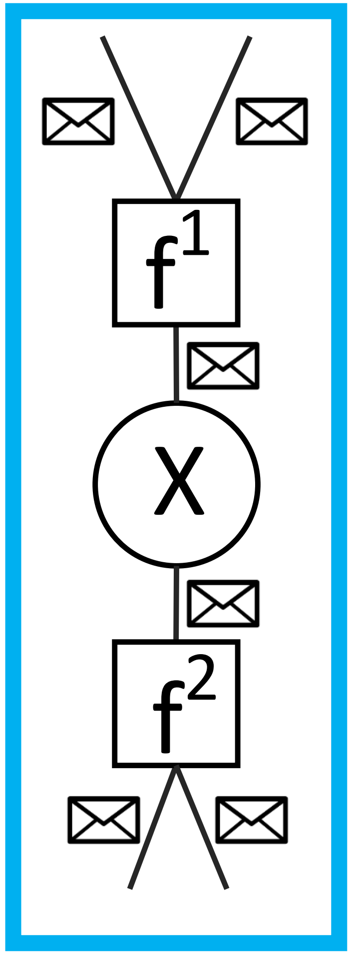
The Probabilistic Scene Grammar Framework

Probabilistic Scene Grammar

Factor graph



Inference

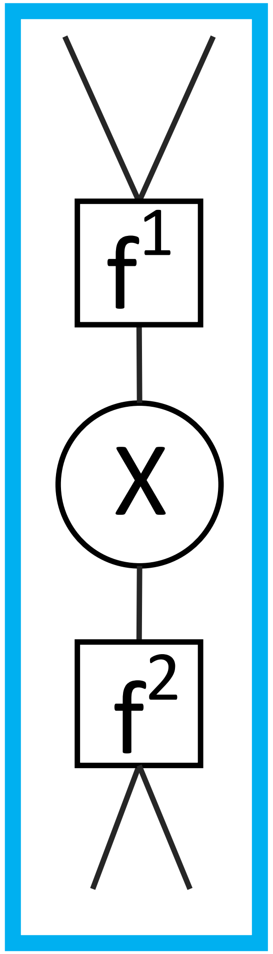


Learning

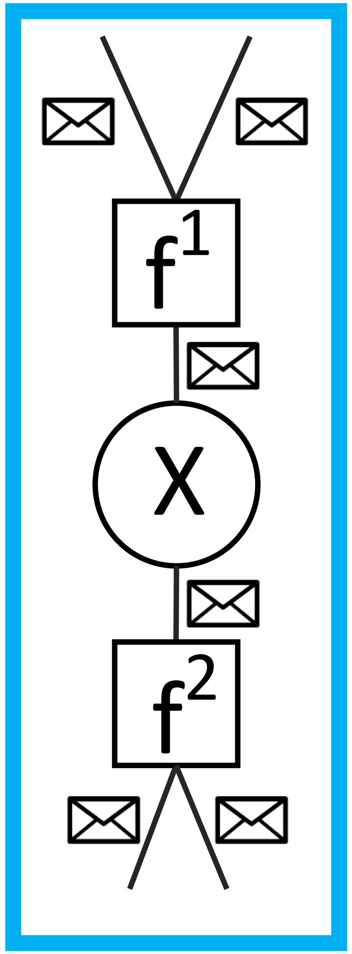
The Probabilistic Scene Grammar Framework

Probabilistic Scene Grammar

Factor graph



Inference



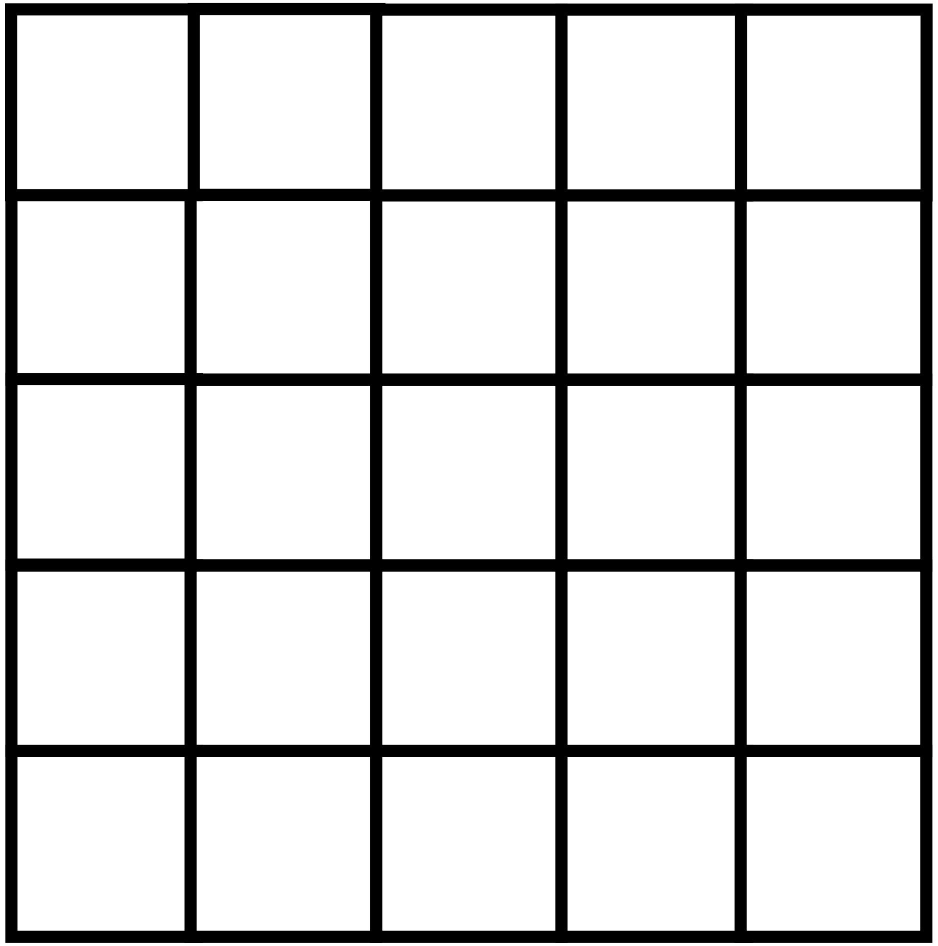
Learning

Probabilistic Scene Grammars

- Context-free stochastic grammar
  - Symbols, *eg. {Face, eye, conversation}*
  - Pose space, *eg. {location, orientation, scale}*
  - Production rules, *eg. Face → {Eye,Eye,Nose,Mouth}*
  - Production rule probabilities
- Geometric relationships between objects/parts
- “Self-rooting” parameter

Generative example: Faces

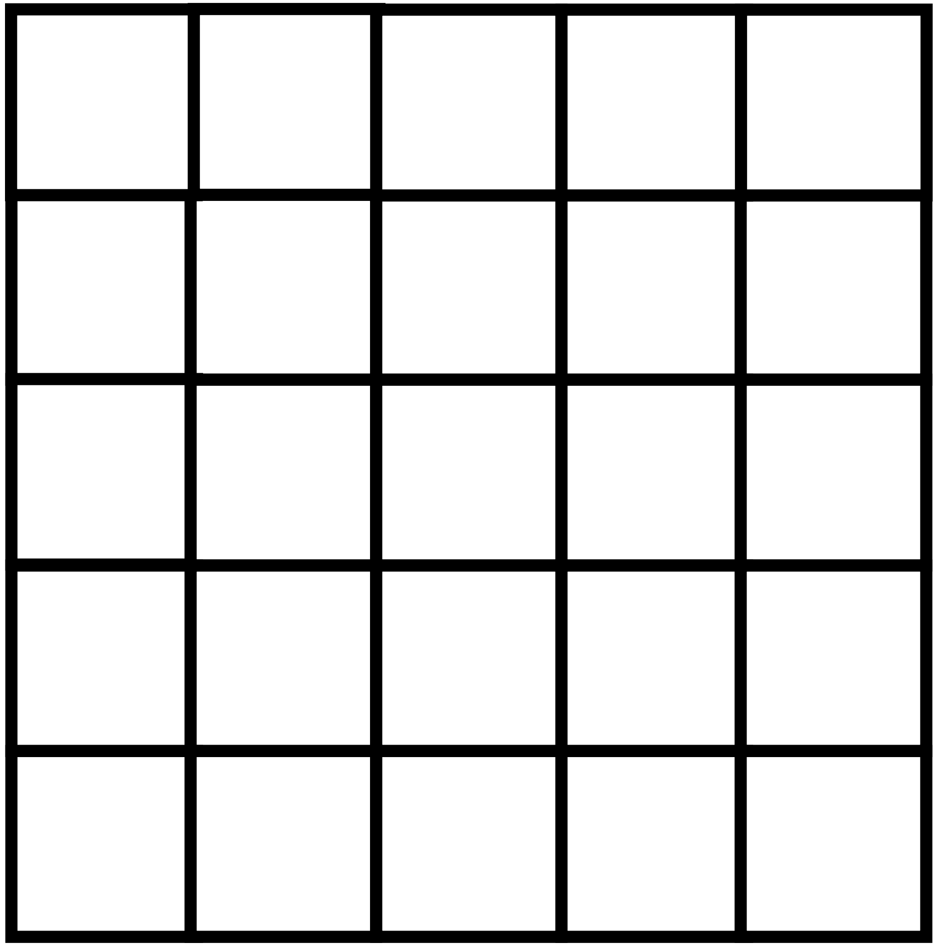
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type

Generative example: Faces

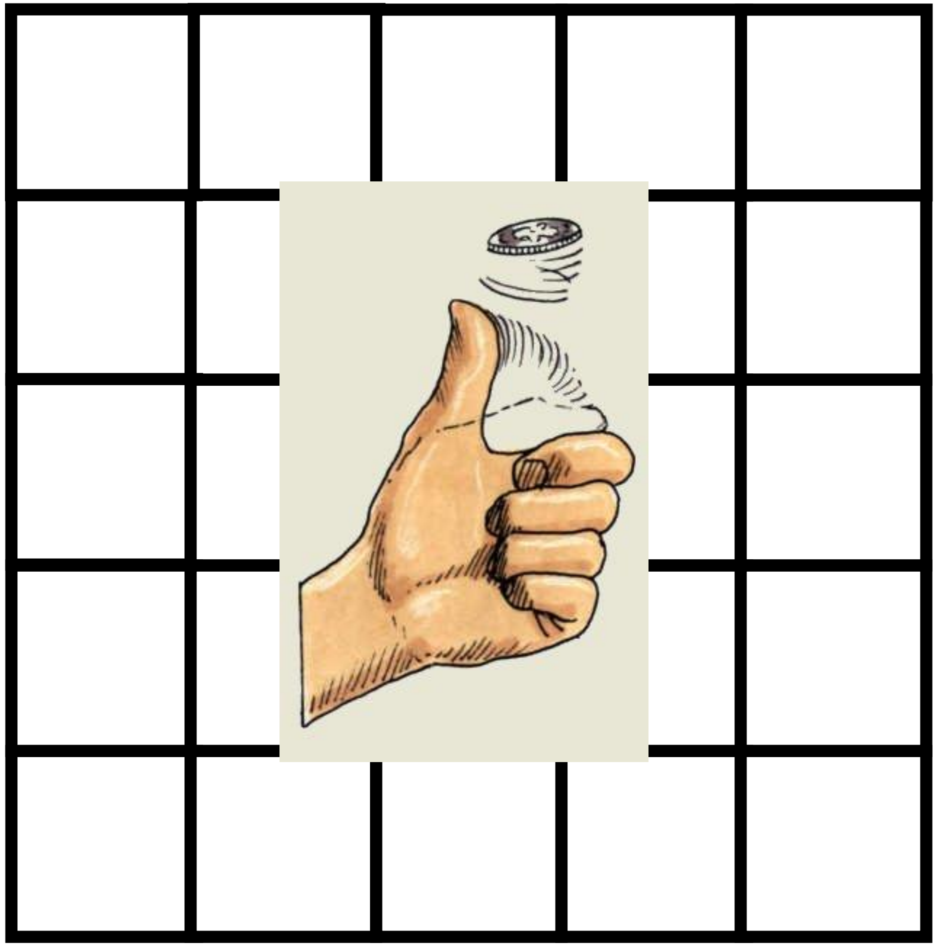
Symbols: $\boxed{\text{Face (F)}}$, Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type

Generative example: Faces

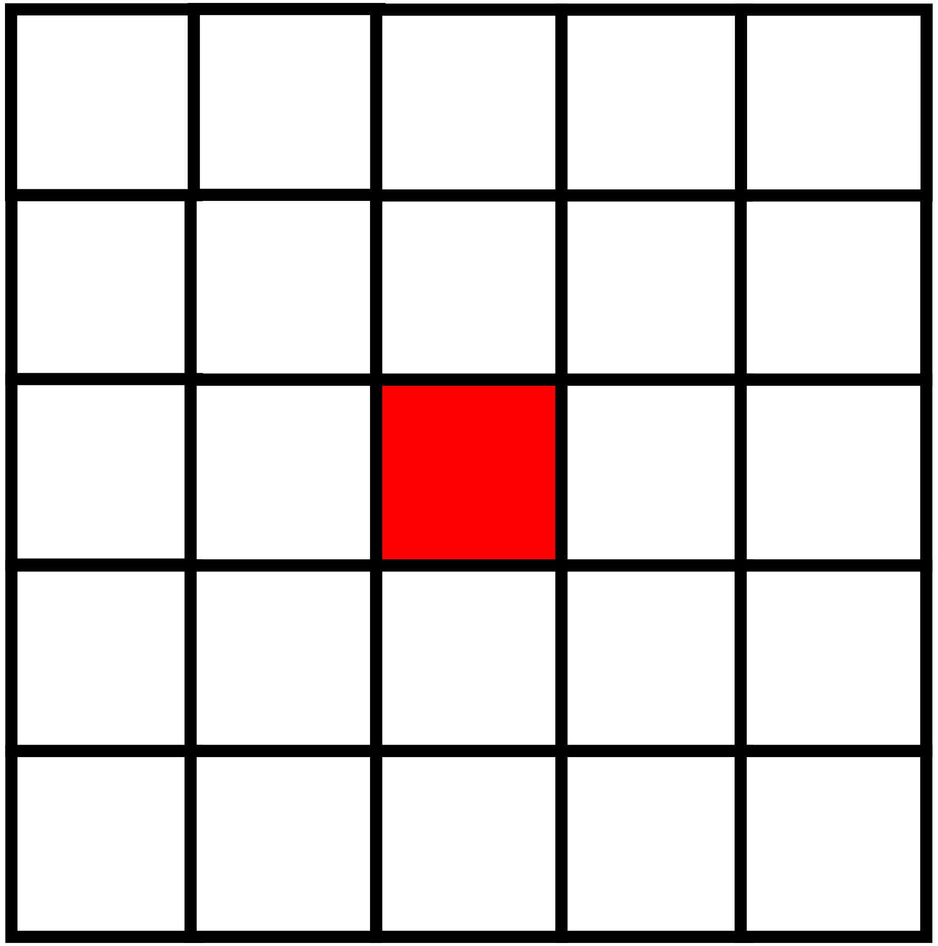
Symbols: $\boxed{\text{Face (F)}}$, Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type

Generative example: Faces

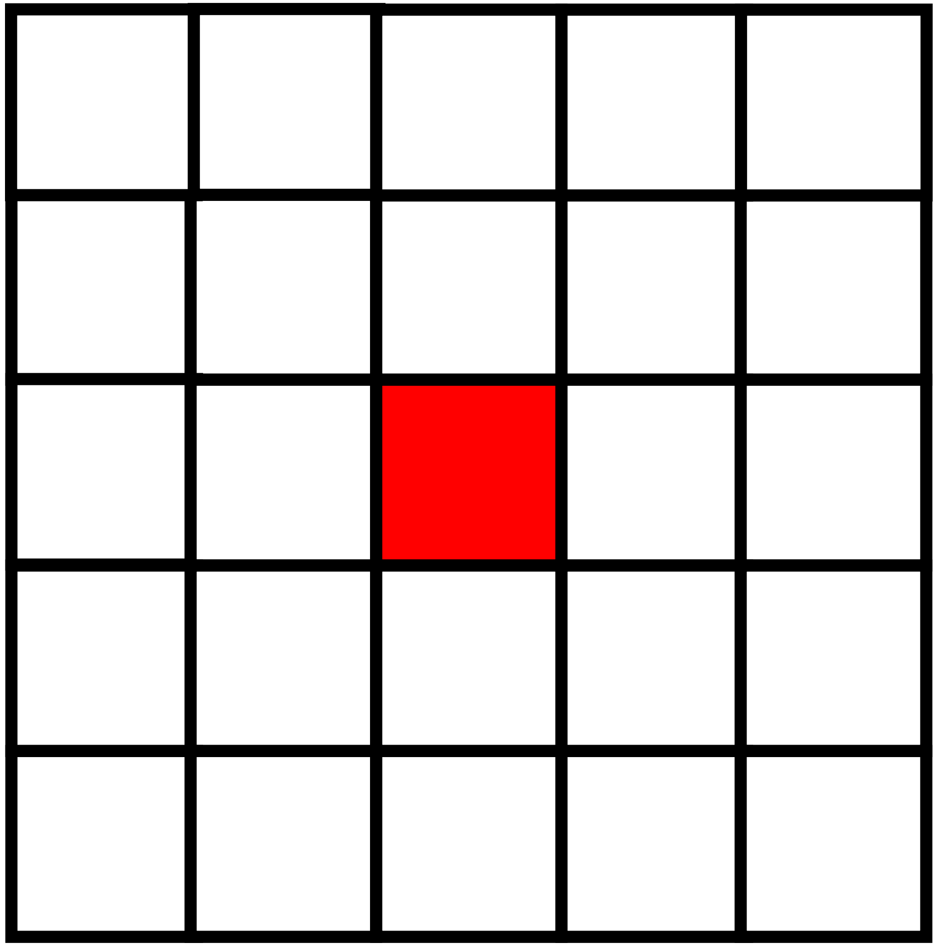
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type

Generative example: Faces

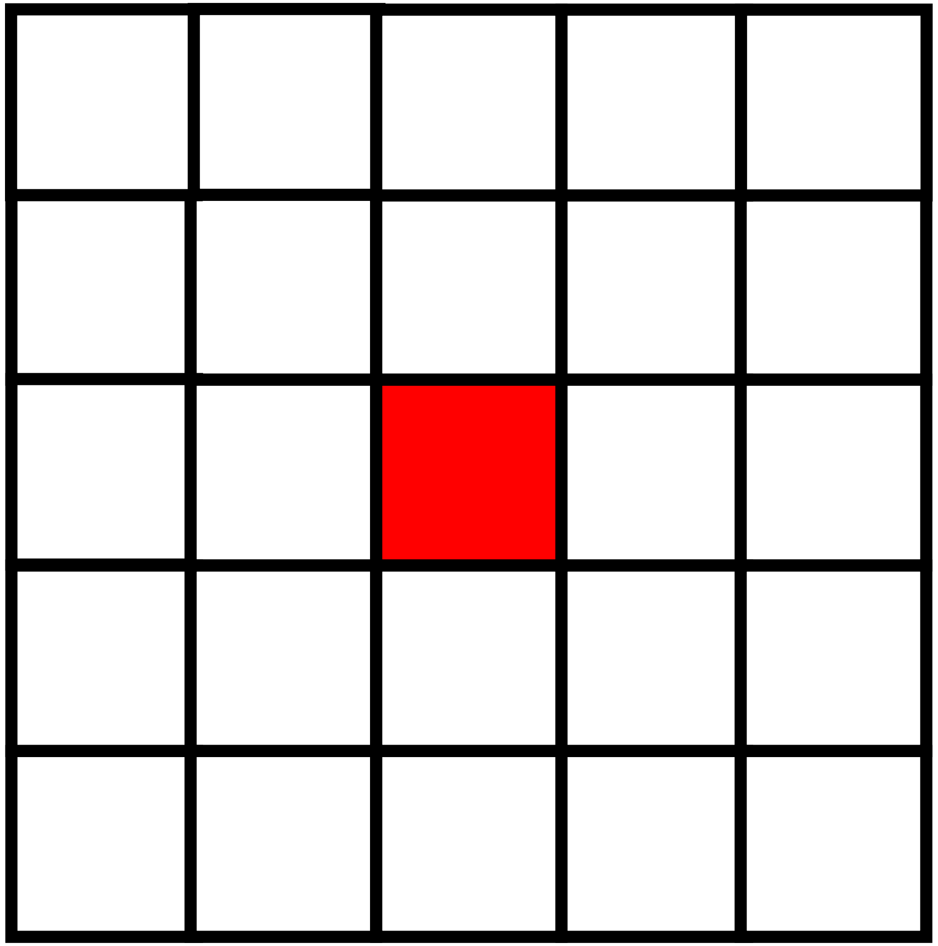
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

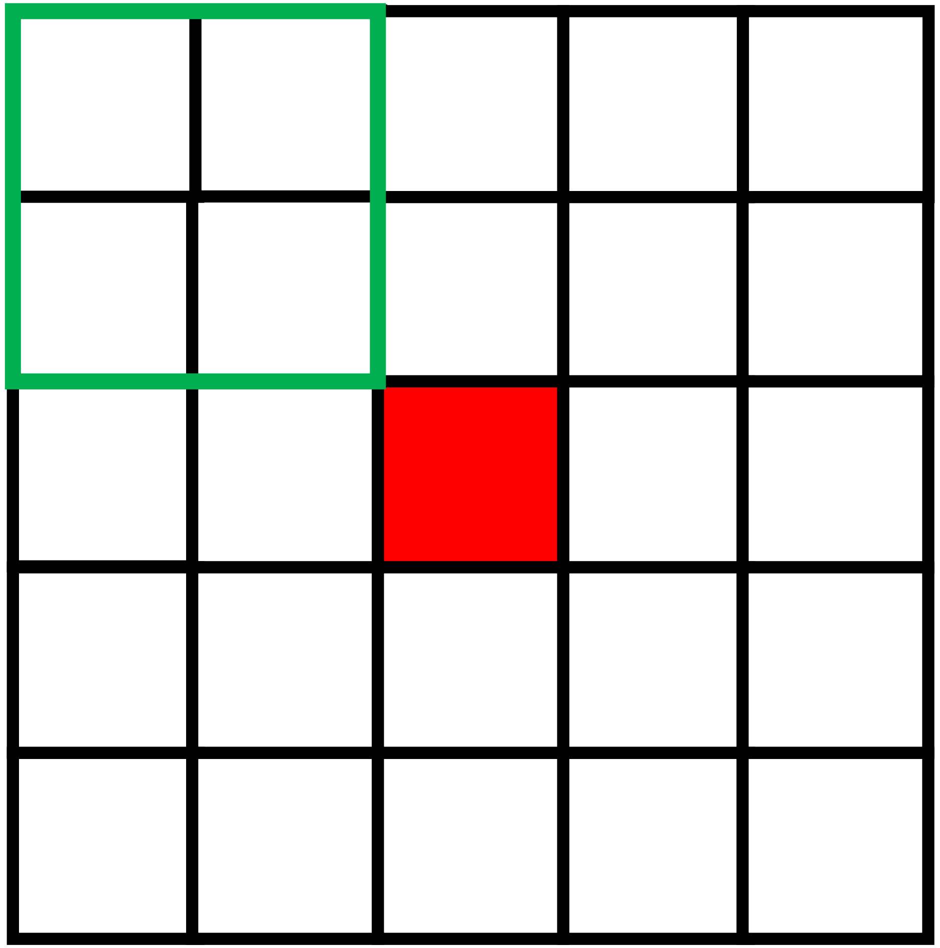
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

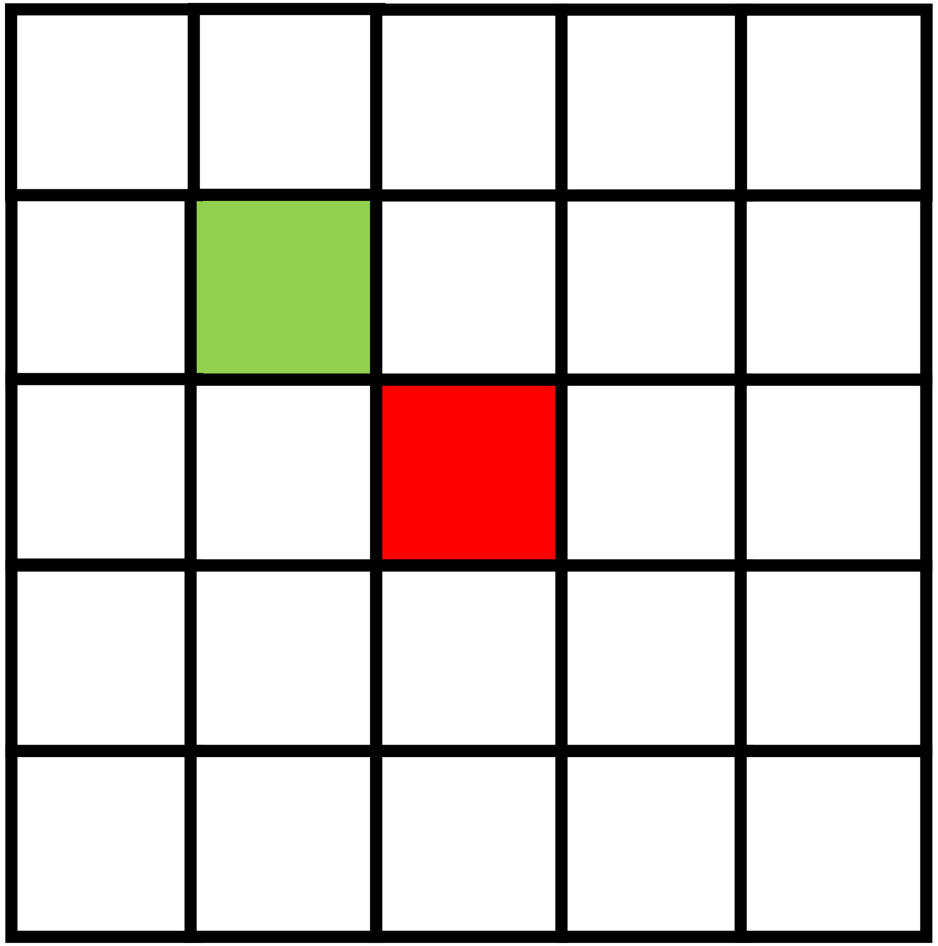
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

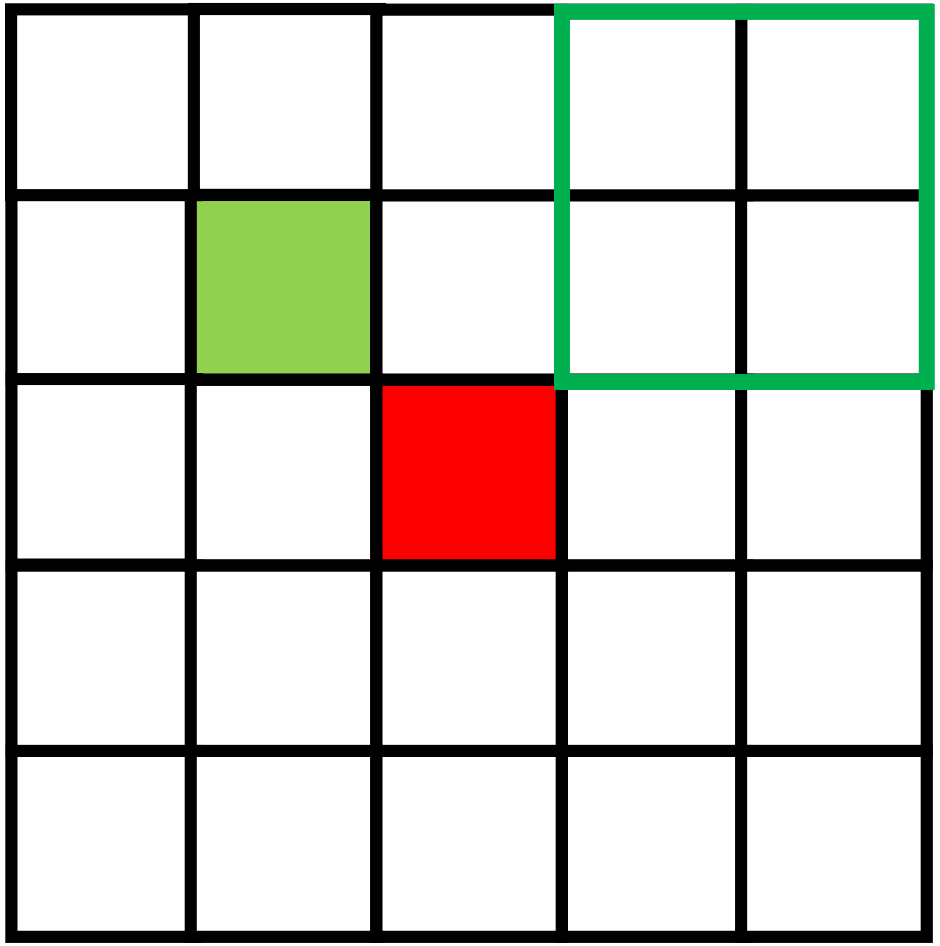
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

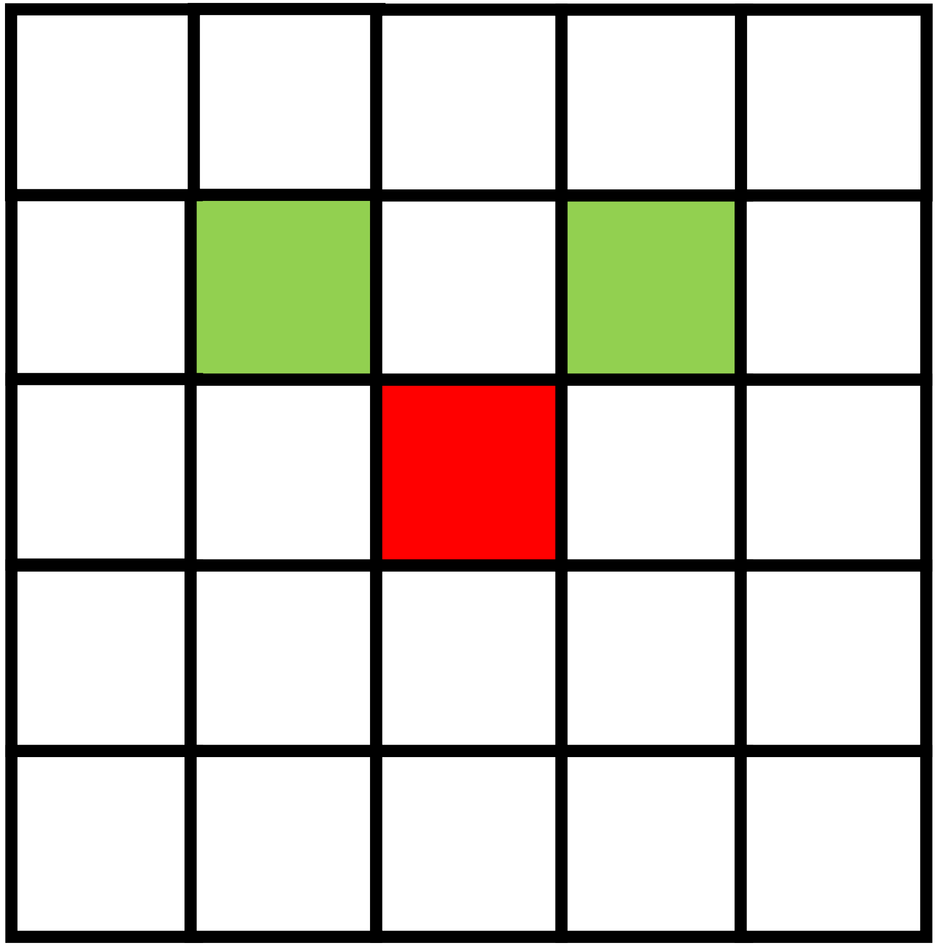
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

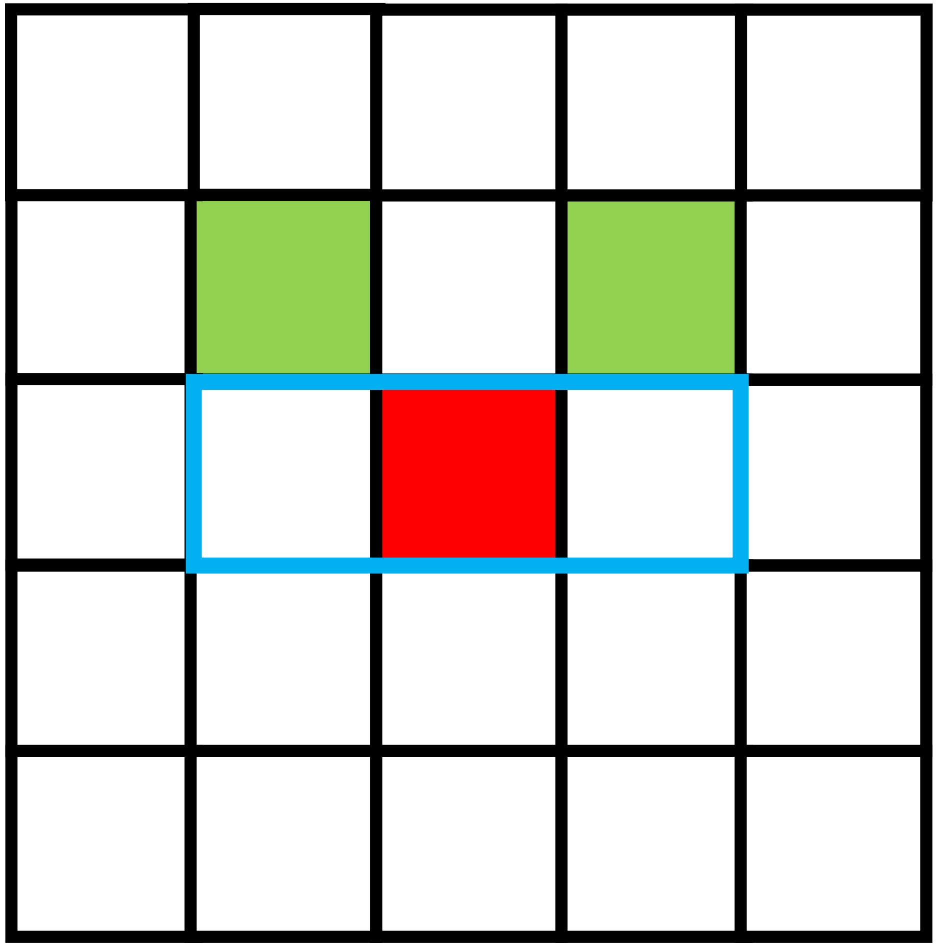
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

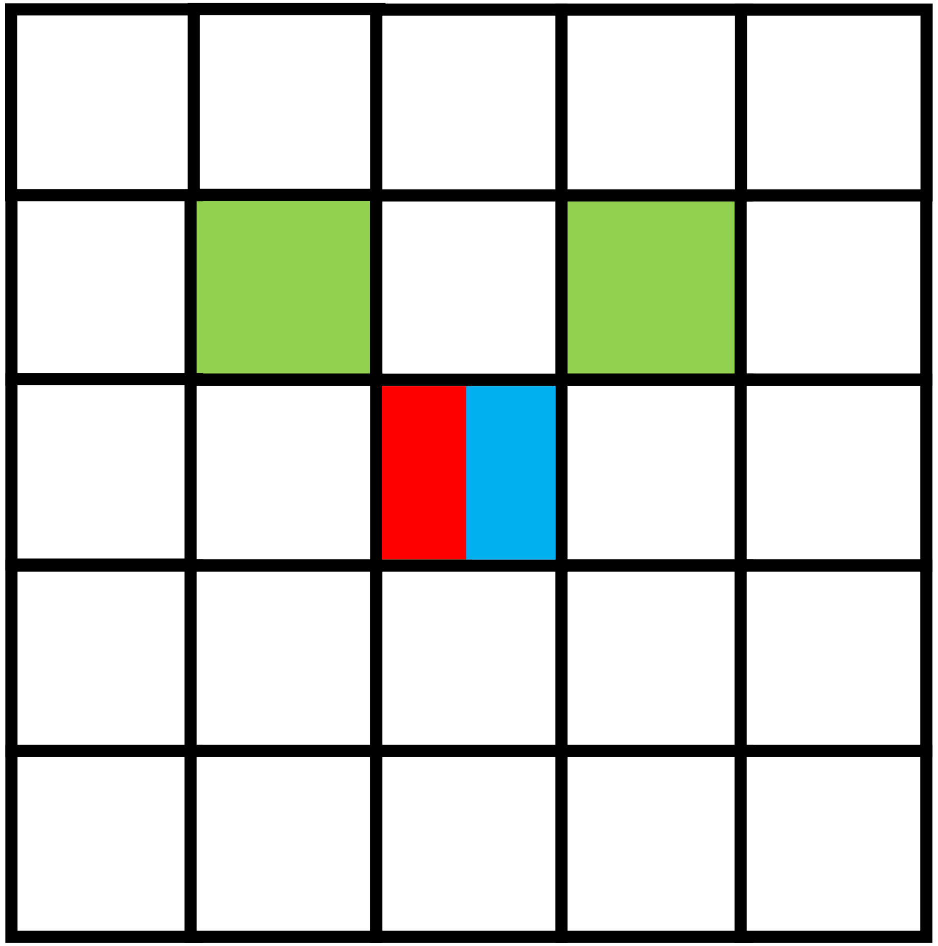
Symbols: $\boxed{\text{Face (F)}}$, Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,**N**,M	1.0	**Uniform**

Generative example: Faces

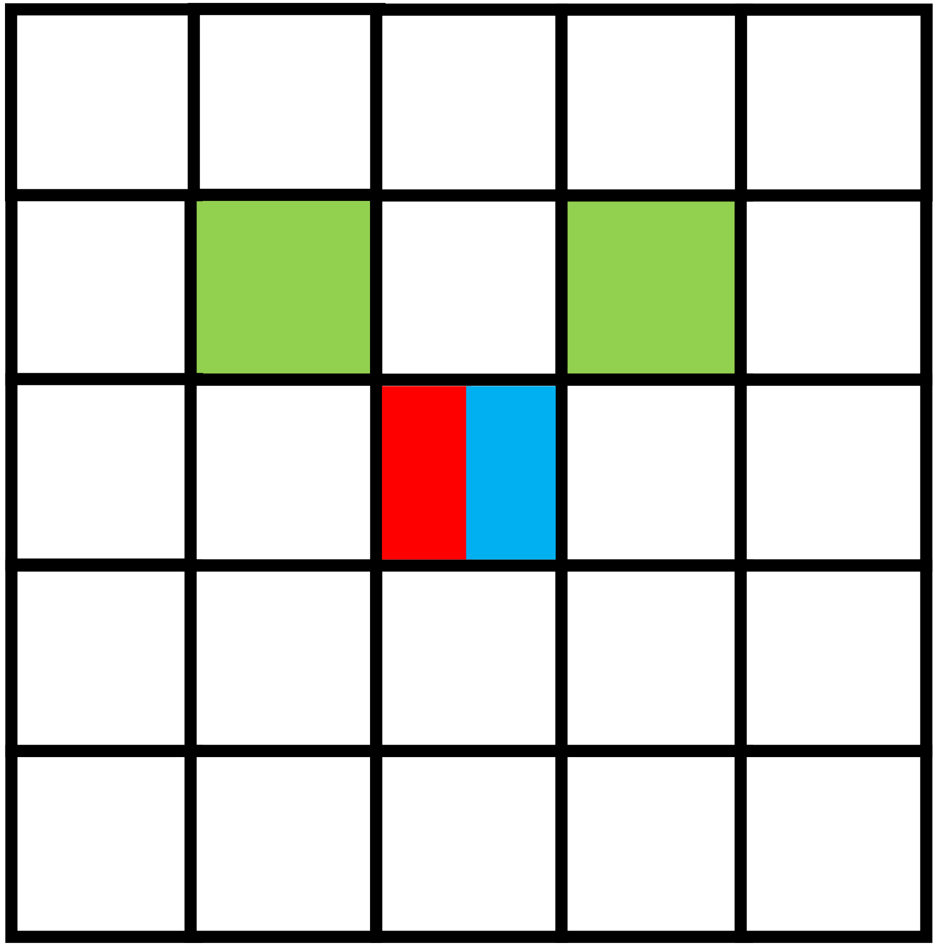
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

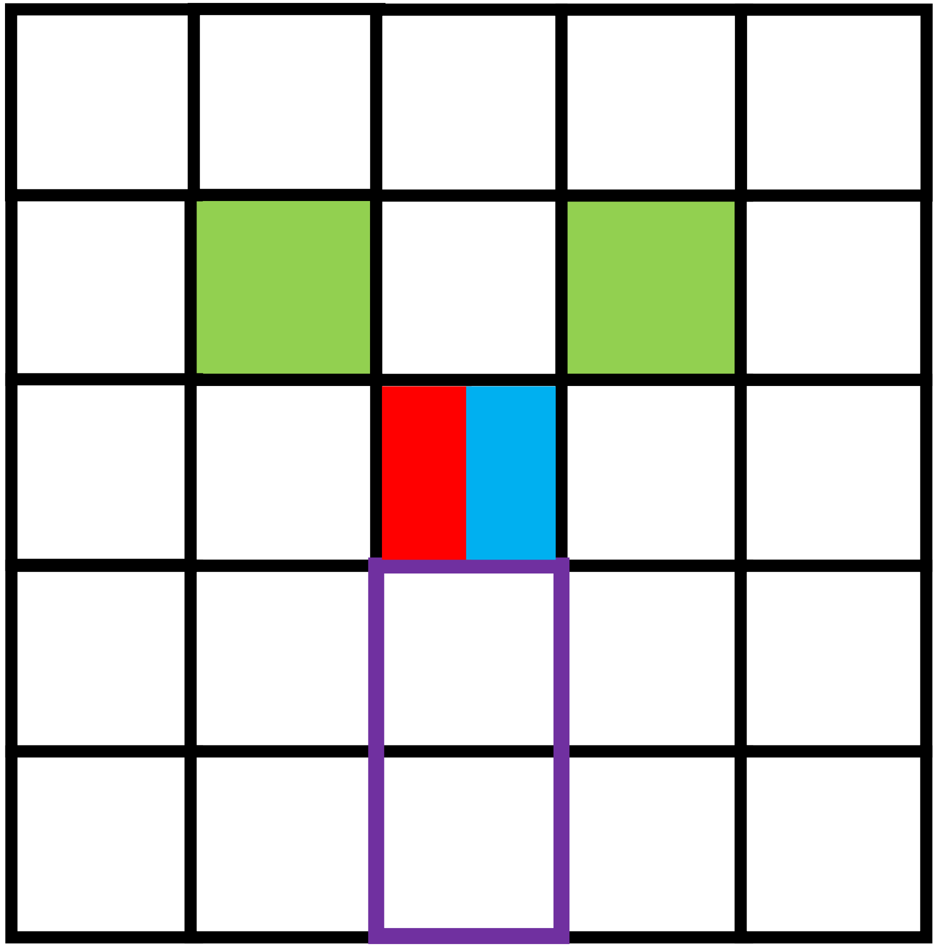
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,**M**	1.0	Uniform

Generative example: Faces

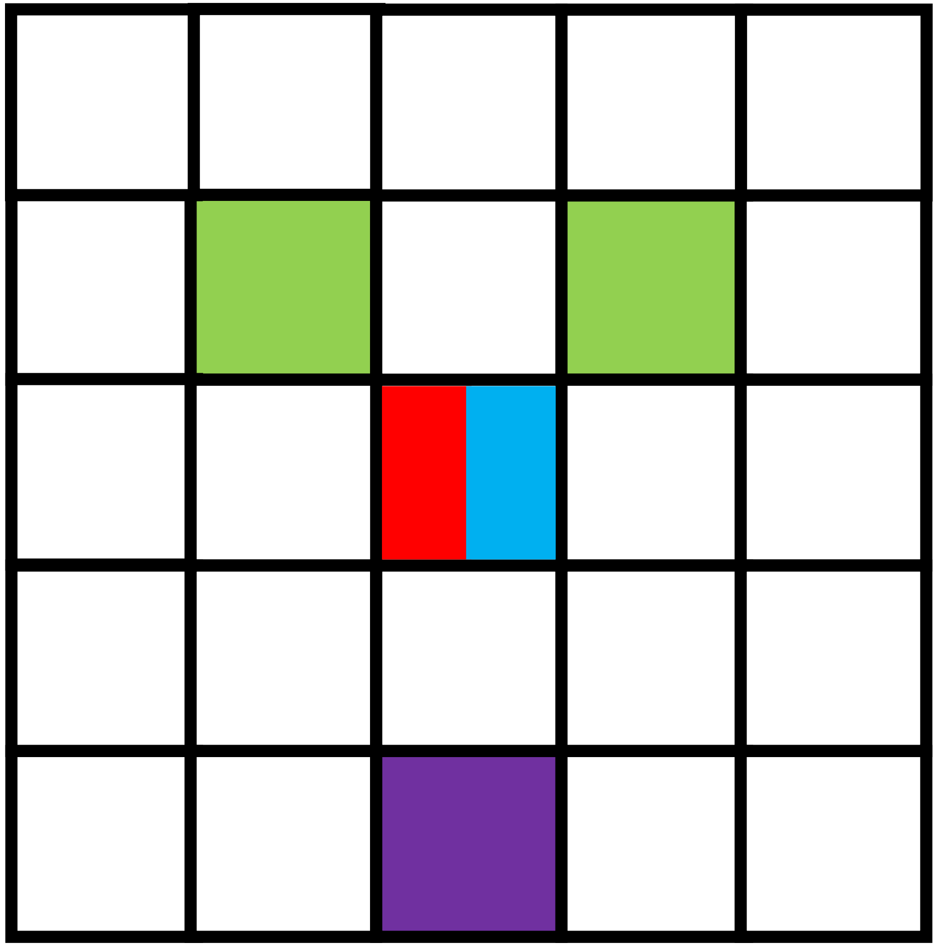
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,**M**	1.0	**Uniform**

Generative example: Faces

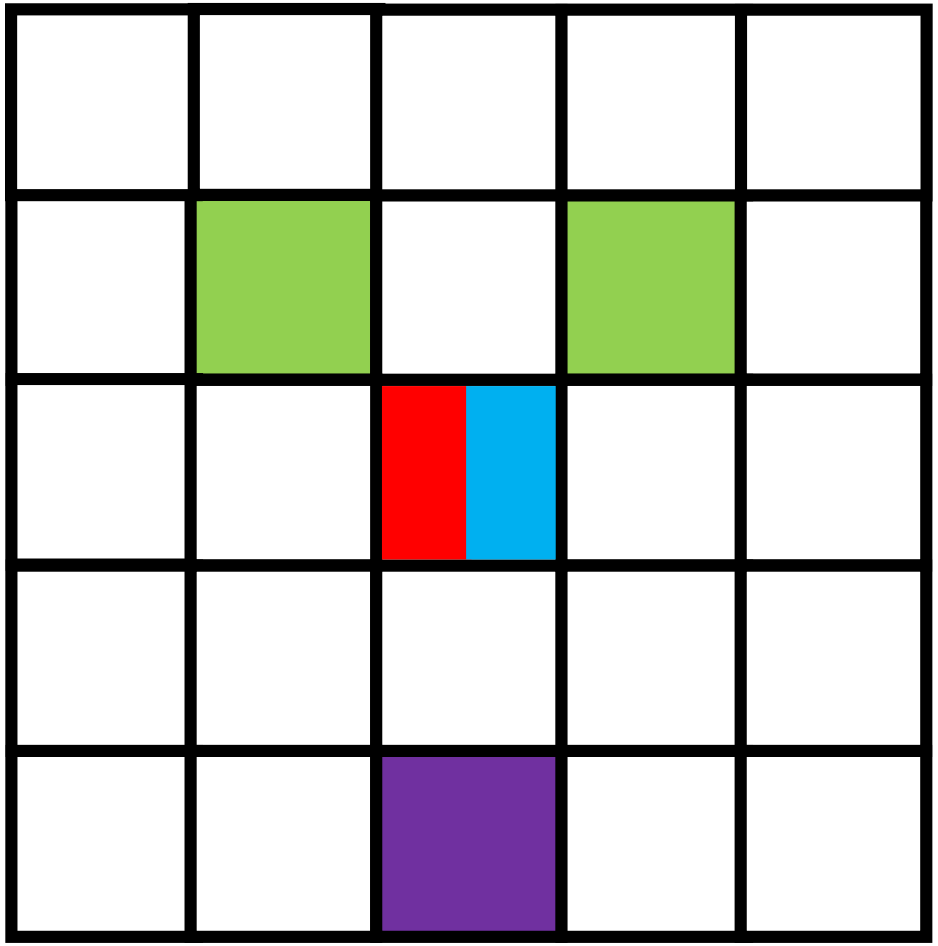
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,**M**	1.0	Uniform

Generative example: Faces

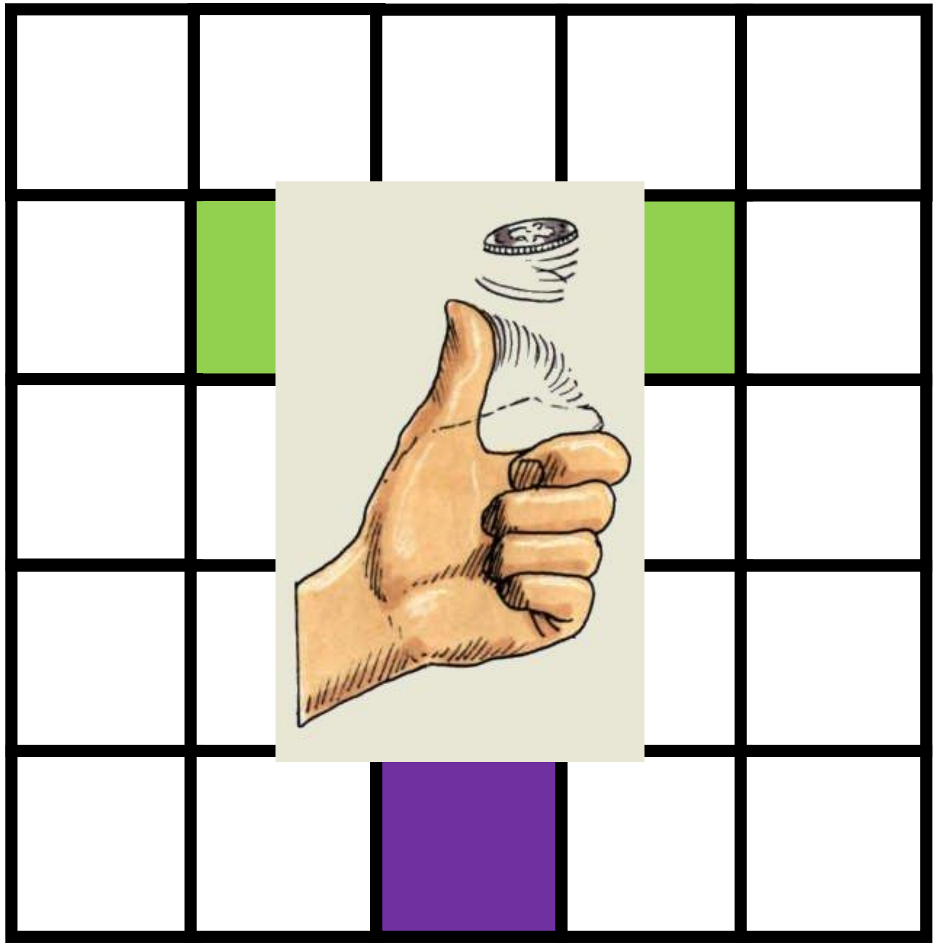
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

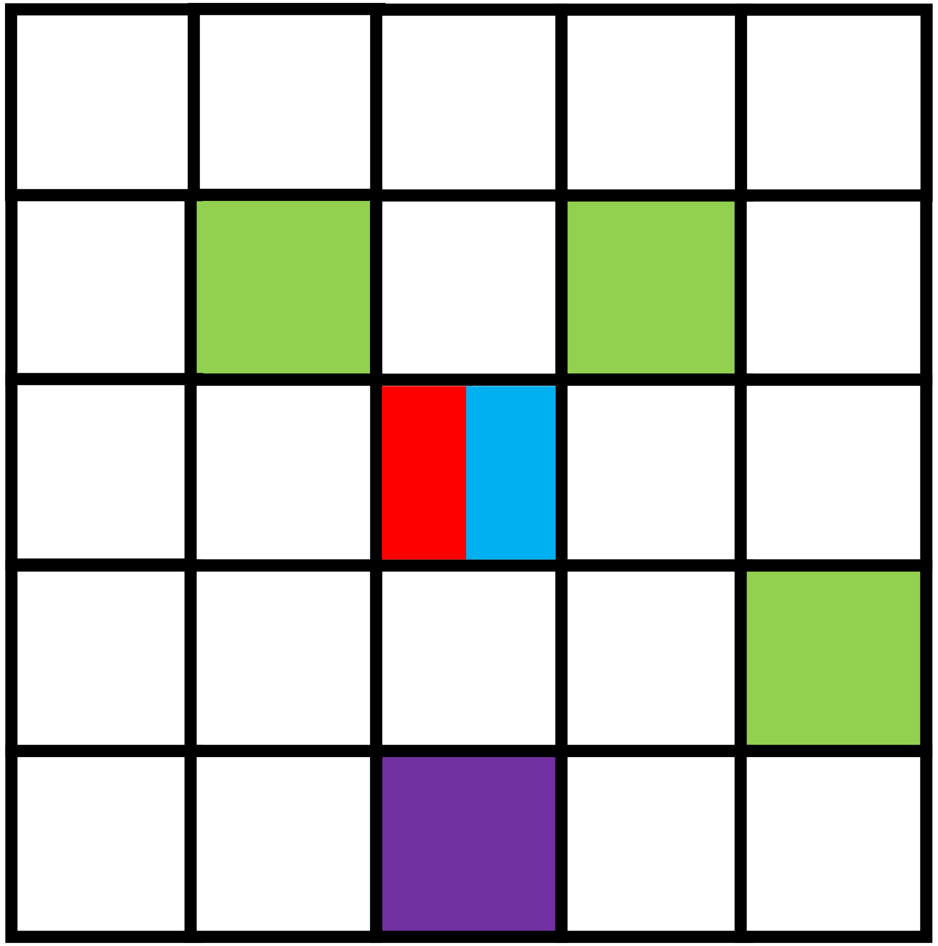
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

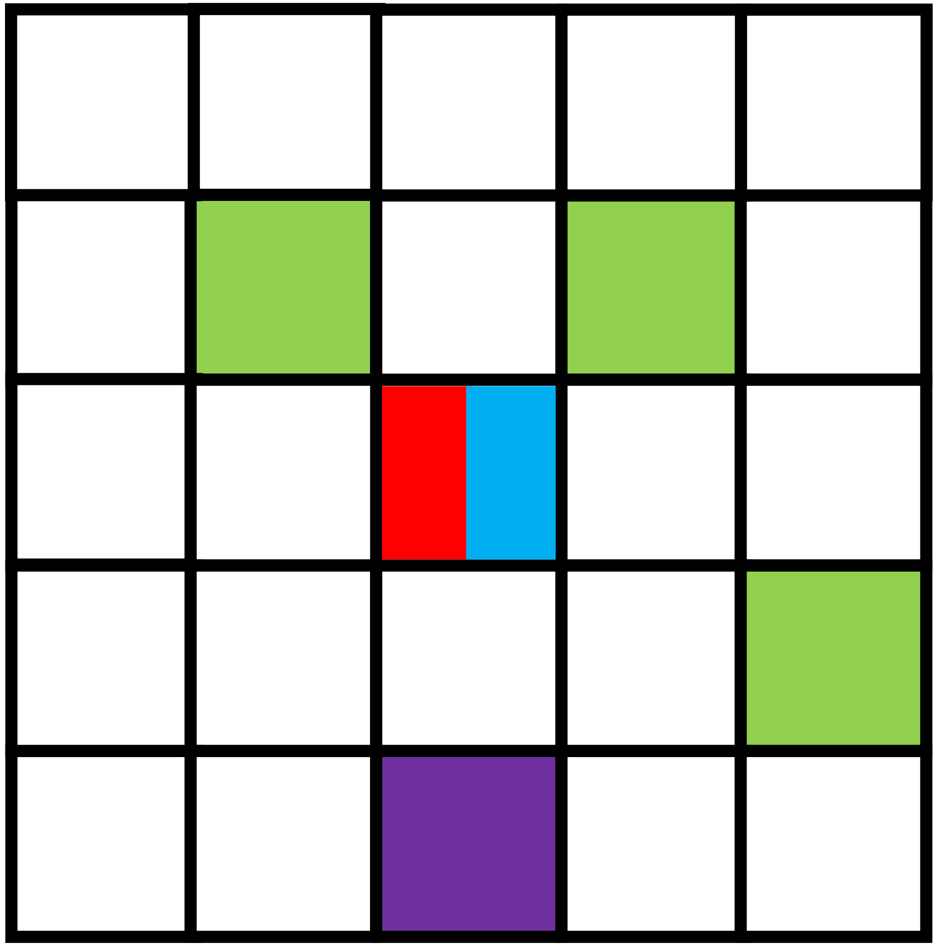
Symbols: Face (F), $\boxed{\text{Eye (E)}}$, Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform

Generative example: Faces

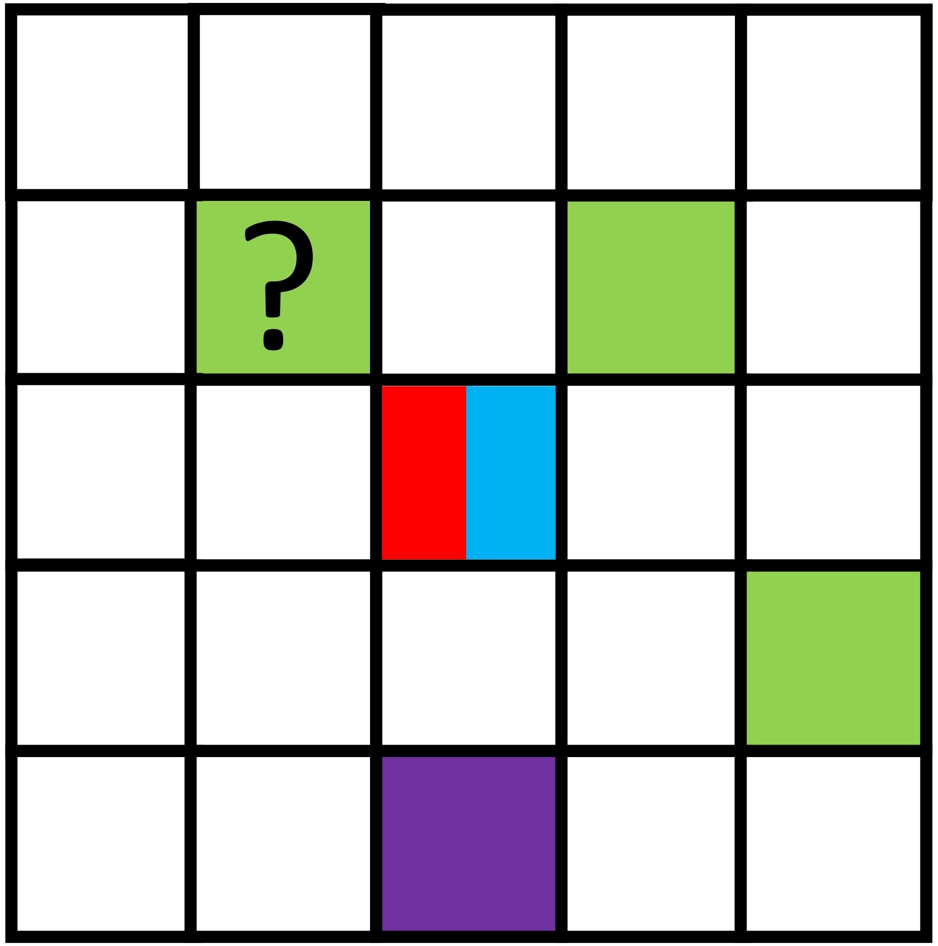
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	
E→∅	0.5	-

Generative example: Faces

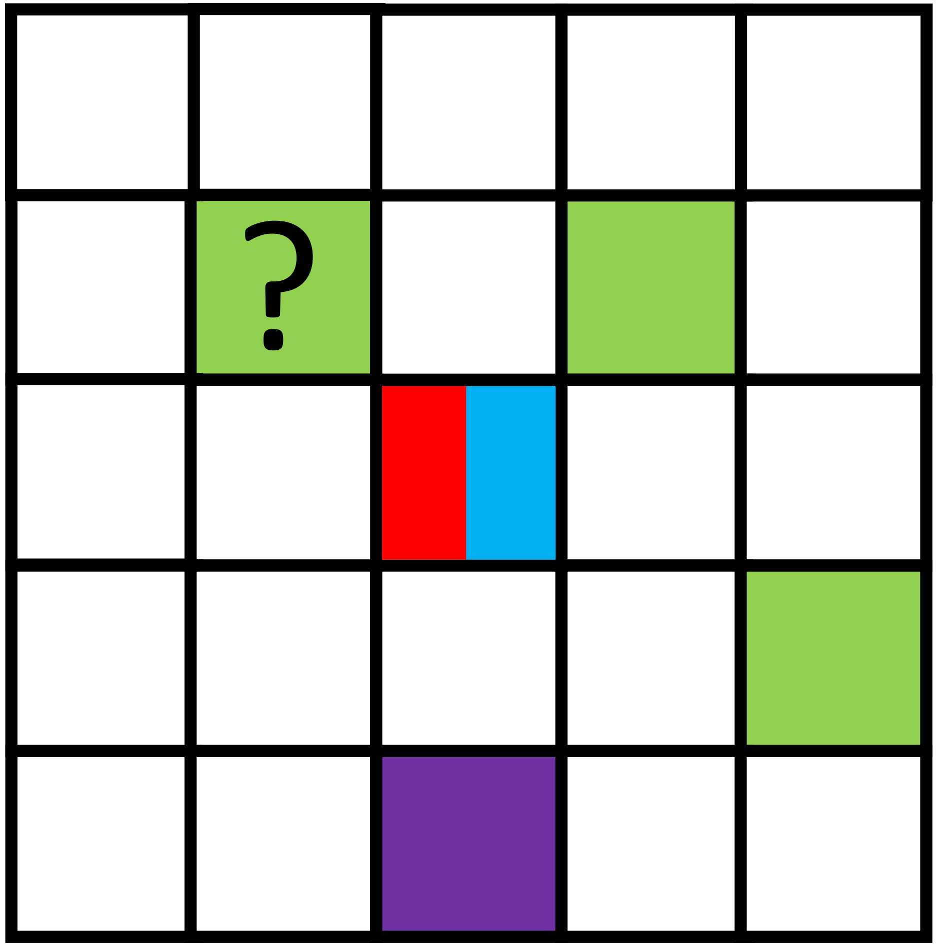
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	
E→Ø	0.5	-

Generative example: Faces

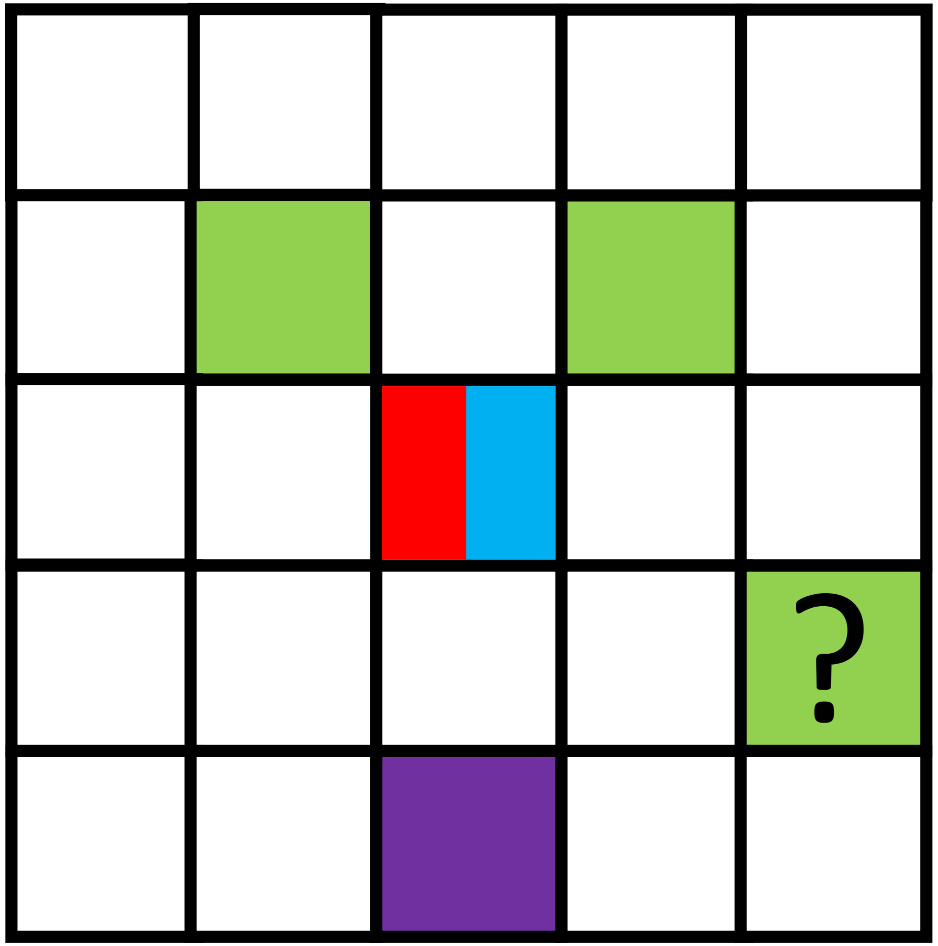
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	
E→∅	0.5	-

Generative example: Faces

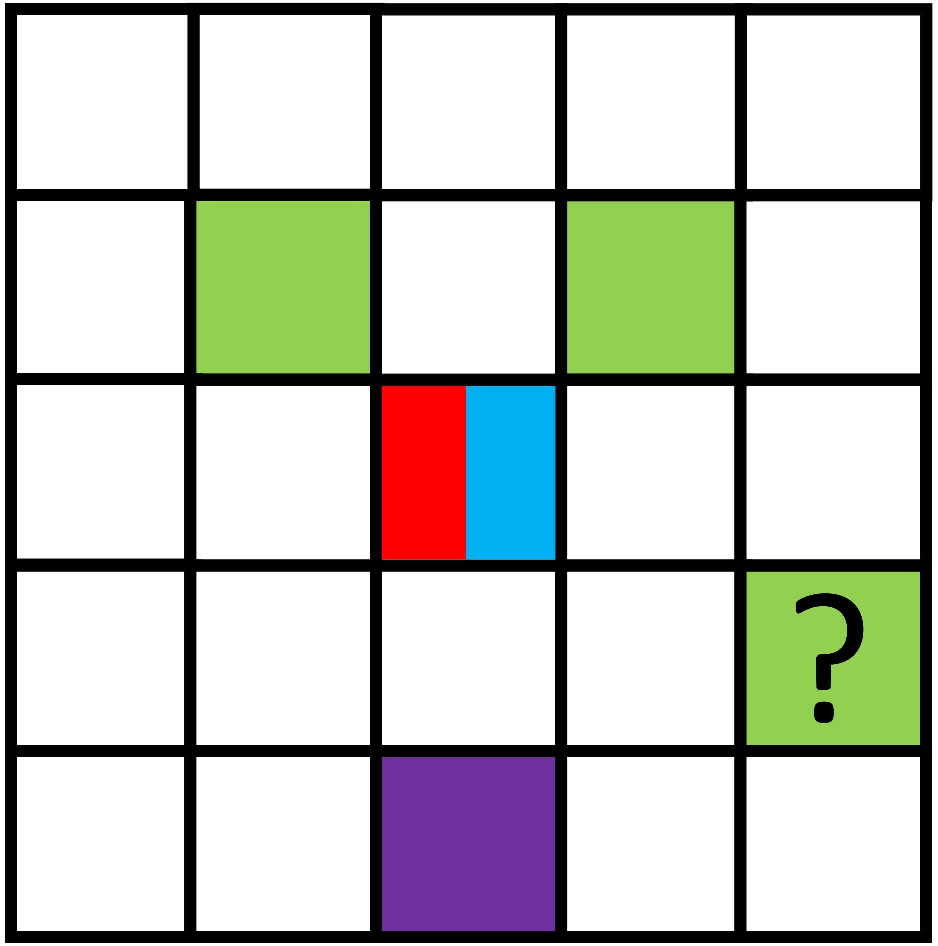
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	
E→∅	0.5	-

Generative example: Faces

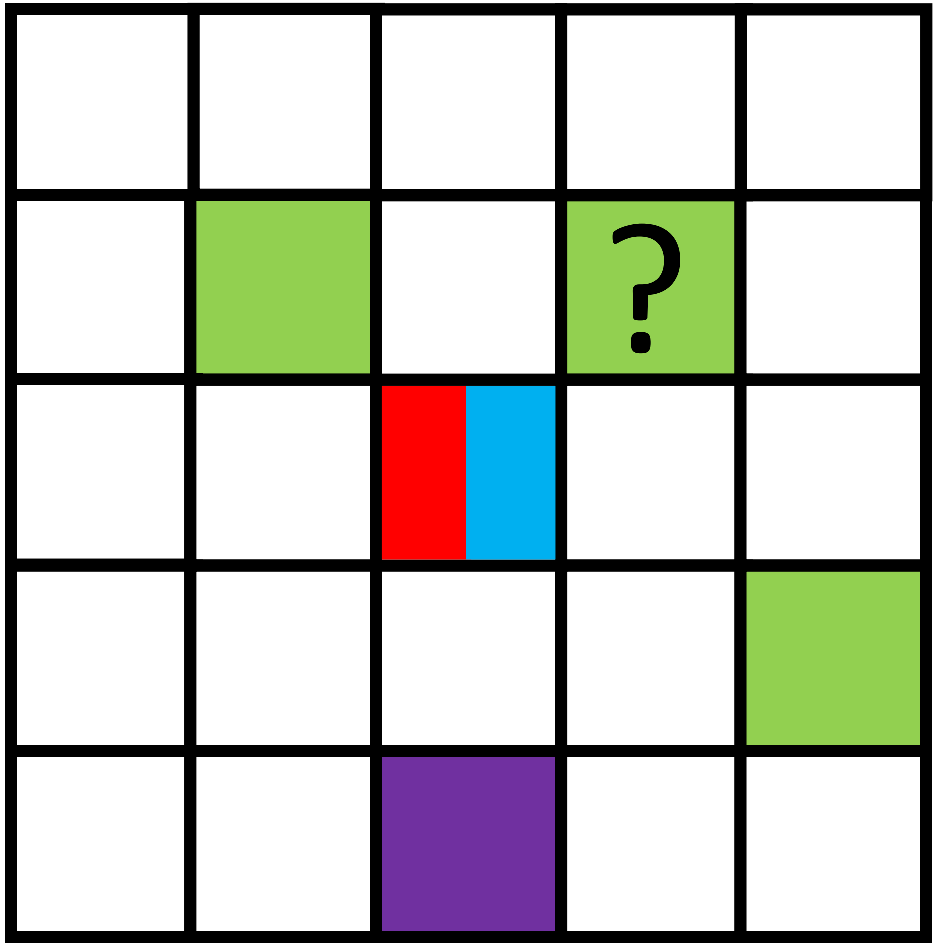
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	
E→Ø	0.5	-

Generative example: Faces

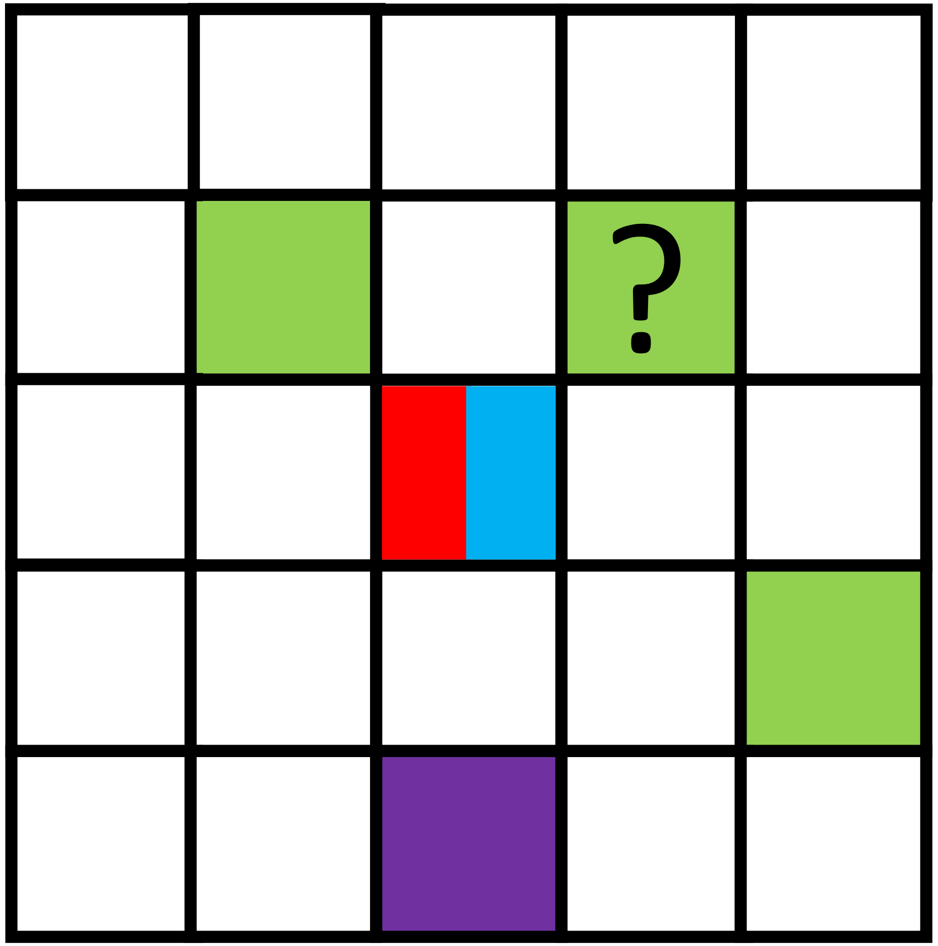
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	
E→∅	0.5	-

Generative example: Faces

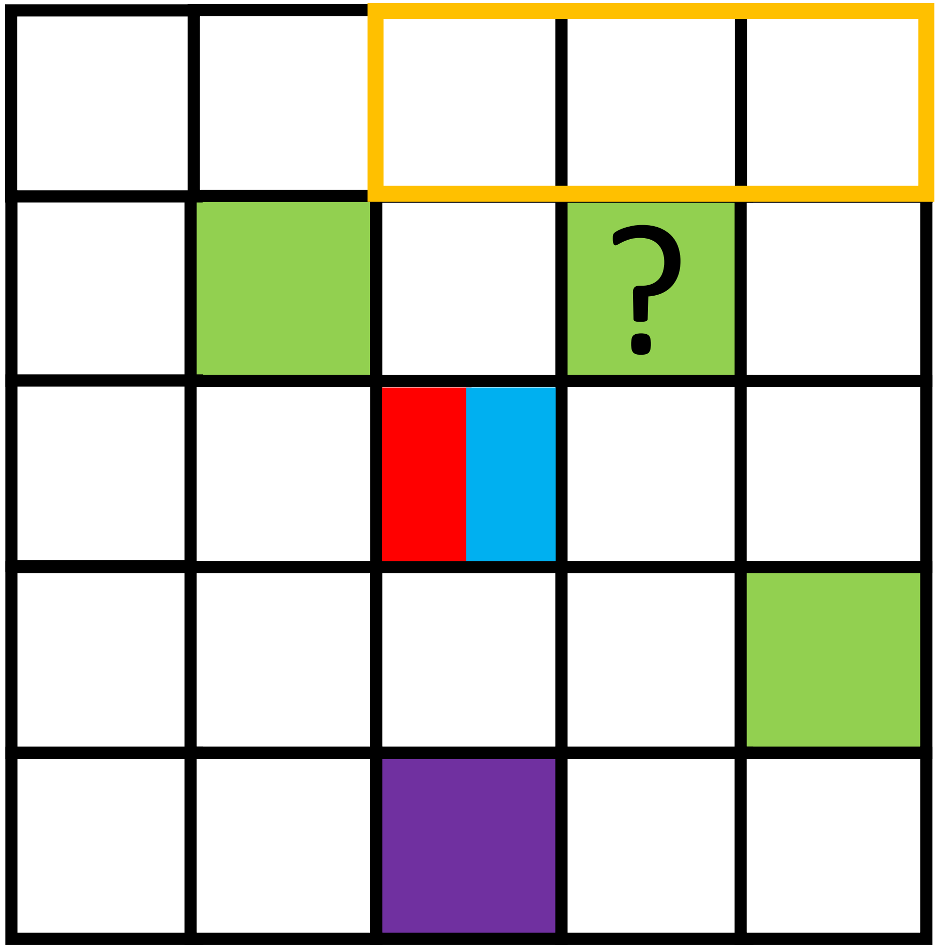
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	
E→Ø	0.5	-

Generative example: Faces

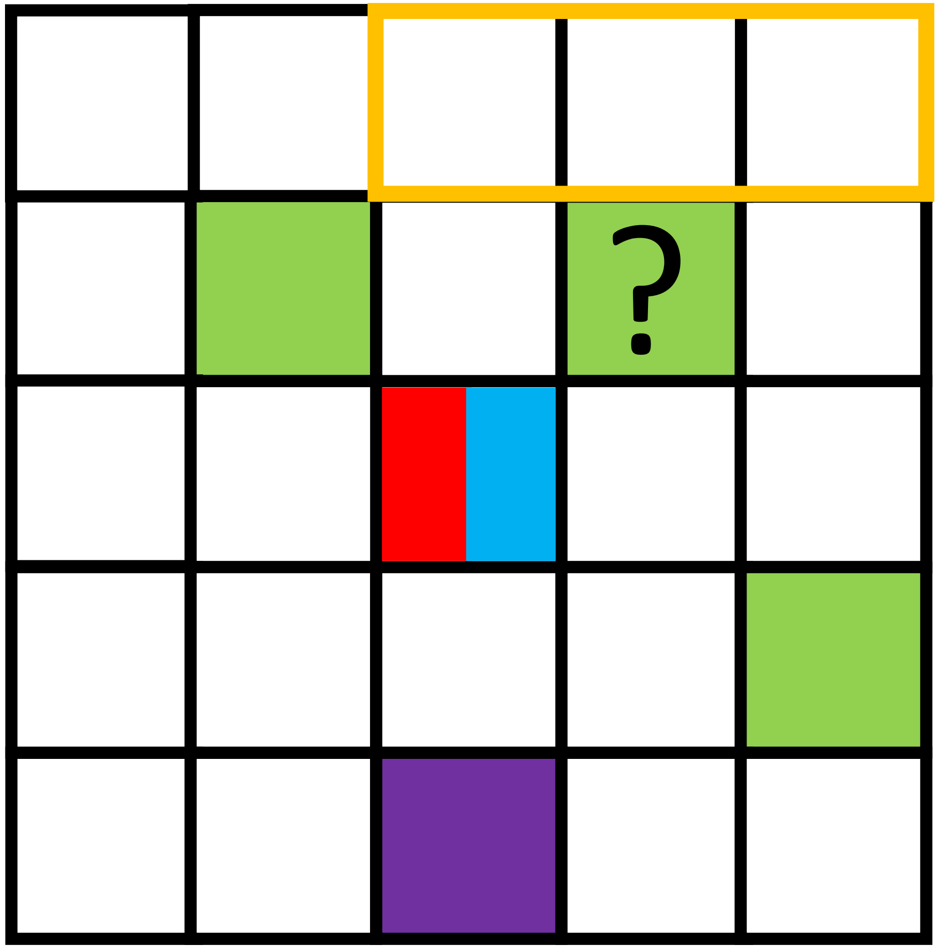
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	
E→∅	0.5	-

Generative example: Faces

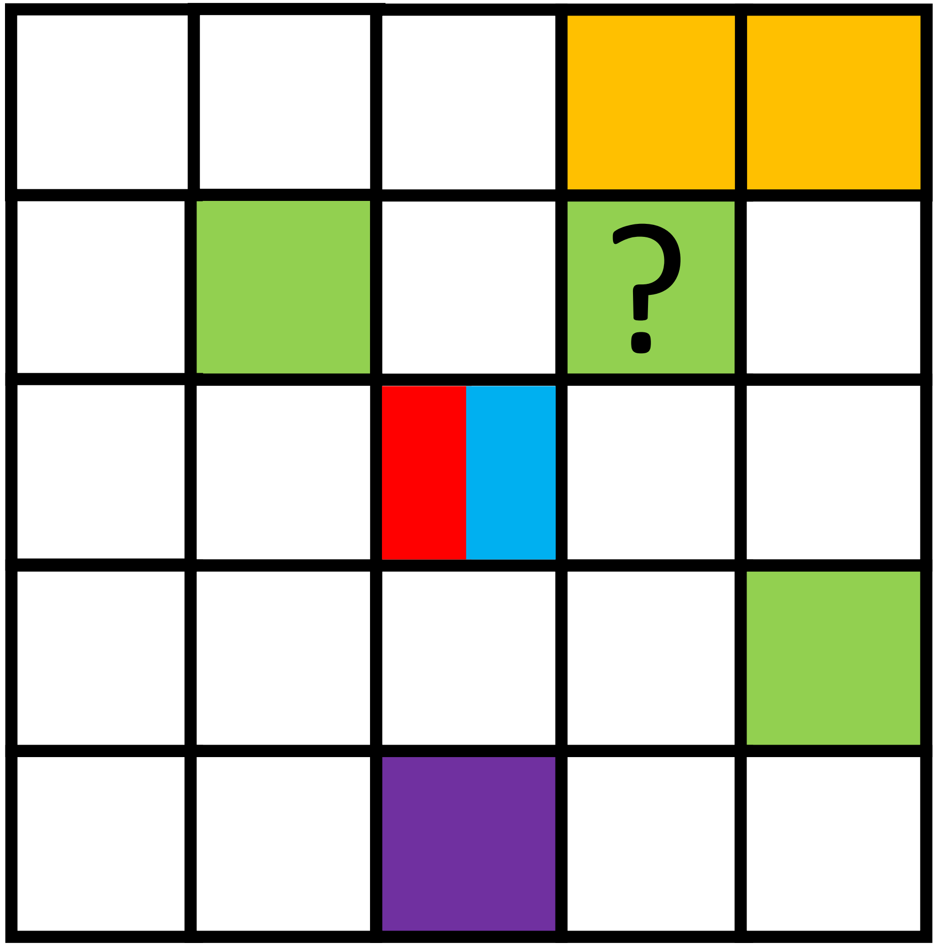
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	Indep. Bernoullis, 50%
E→Ø	0.5	-

Generative example: Faces

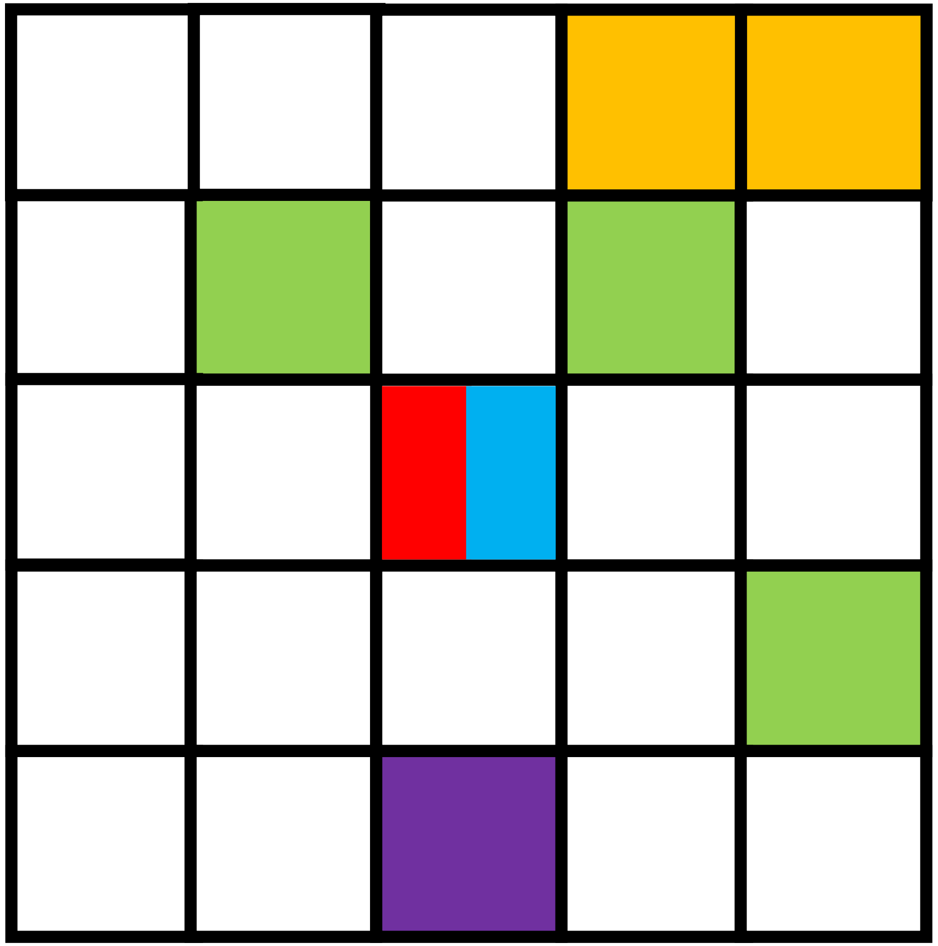
Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	Indep. Bernoullis, 50%
E→Ø	0.5	-

Generative example: Faces

Symbols: Face (F), Eye (E), Nose (N), Mouth (M), Eyelashes (L)



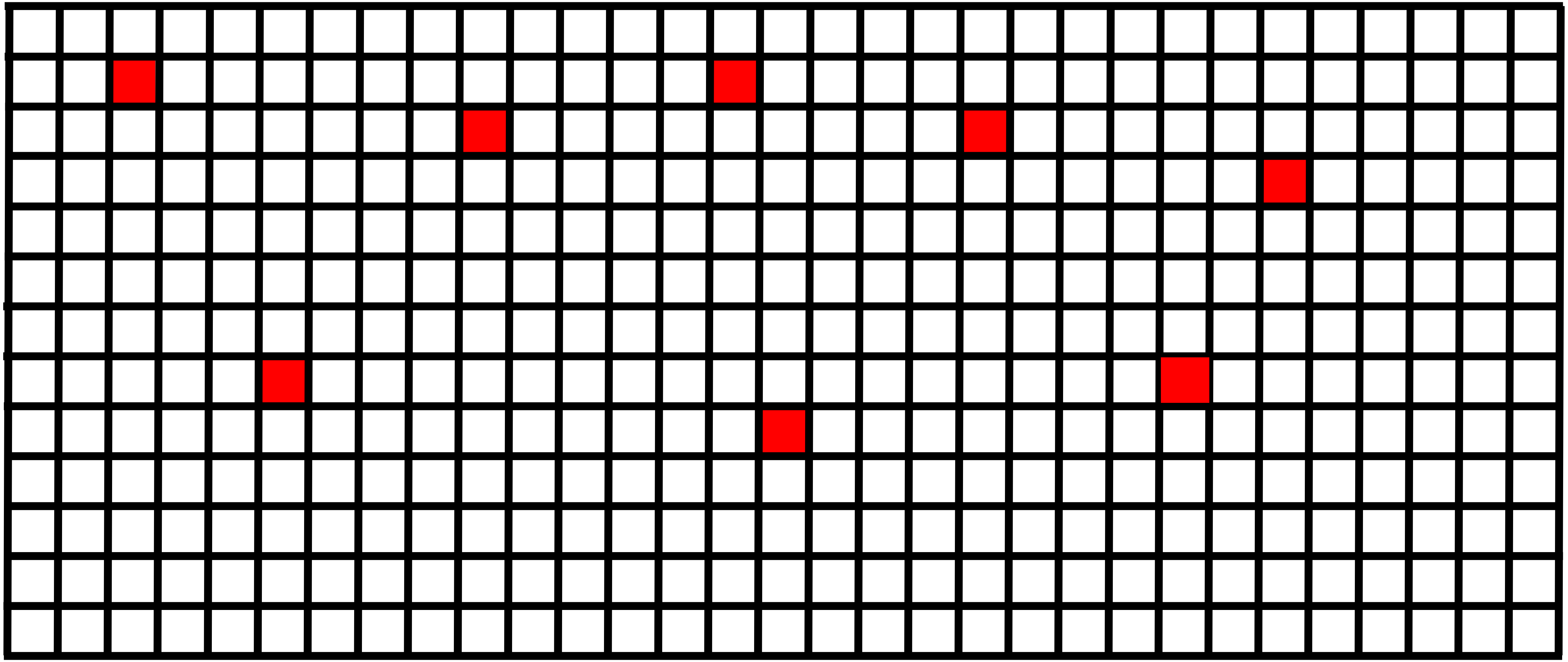
Rule	P(rule)	Spatial distribution type
F→E,E,N,M	1.0	Uniform
E→L	0.5	Indep. Bernoullis, 50%
E→Ø	0.5	-
N→Ø	**1.0**	**-**
M→Ø	**1.0**	**-**
L→Ø	**1.0**	**-**

Generative example: Faces

Symbols: Face (F), eye (E), nose (N), mouth (M), eyelashes (L)

Generative example: Faces

Symbols: Face (F), eye (E), nose (N), mouth (M), eyelashes (L)



Probabilistic Scene Grammar Specification

Face localization grammar

$\Sigma = \{\mathsf{FACE}, \mathsf{EYE}, \mathsf{NOSE}, \mathsf{MOUTH}\}$.

$\forall A \in \Sigma,\ \Omega_A = [N] \times [M]$.

Rules:

$$
\begin{array}{lll}
1.0, & (\mathsf{FACE}, \omega) & \rightarrow (\mathsf{EYE}, \mathrm{UniformRect}(\omega + a_1, \omega + b_1)), \\
 & & \quad (\mathsf{EYE}, \mathrm{UniformRect}(\omega + a_2, \omega + b_2)), \\
 & & \quad (\mathsf{NOSE}, \mathrm{UniformRect}(\omega + a_3, \omega + b_3)), \\
 & & \quad (\mathsf{MOUTH}, \mathrm{UniformRect}(\omega + a_4, \omega + b_4)) \\
1.0, & (\mathsf{EYE}, \omega) & \rightarrow \emptyset \\
1.0, & (\mathsf{NOSE}, \omega) & \rightarrow \emptyset \\
1.0, & (\mathsf{MOUTH}, \omega) & \rightarrow \emptyset
\end{array}
$$

$\epsilon_{\mathsf{FACE}} = 10^{-4}$,

$\epsilon_{\mathsf{EYE}} = \epsilon_{\mathsf{NOSE}} = \epsilon_{\mathsf{MOUTH}} = 10^{-5}$.

Contour detection grammar

$\Sigma = \{\mathsf{CURVE}, \mathsf{INK}\}$.

$\Omega_{\mathsf{CURVE}} = [N] \times [M] \times [8]$.

$\Omega_{\mathsf{INK}} = [N] \times [M]$.

Rules:

$$
\begin{array}{lll}
0.65, & (\mathsf{CURVE}, (x, y, \theta)) & \rightarrow (\mathsf{INK}, \delta((x, y))), (\mathsf{CURVE}, \delta(((x, y) + \mathrm{Round}(T_\theta(1, 0)), \theta))) \\
0.10, & (\mathsf{CURVE}, (x, y, \theta)) & \rightarrow (\mathsf{INK}, \delta((x, y))), (\mathsf{CURVE}, \delta(((x, y) + \mathrm{Round}(T_\theta(1, -1)), \theta))) \\
0.10, & (\mathsf{CURVE}, (x, y, \theta)) & \rightarrow (\mathsf{INK}, \delta((x, y))), (\mathsf{CURVE}, \delta(((x, y) + \mathrm{Round}(T_\theta(1, +1)), \theta))) \\
0.05, & (\mathsf{CURVE}, (x, y, \theta)) & \rightarrow (\mathsf{CURVE}, \delta((x, y, \theta + 1))) \\
0.05, & (\mathsf{CURVE}, (x, y, \theta)) & \rightarrow (\mathsf{CURVE}, \delta((x, y, \theta - 1))) \\
0.05, & (\mathsf{CURVE}, (x, y, \theta)) & \rightarrow (\mathsf{INK}, \delta((x, y))), \\
1.00, & (\mathsf{INK}, (x, y)) & \rightarrow \emptyset
\end{array}
$$

$\epsilon_{\mathsf{CURVE}} = \epsilon_{\mathsf{INK}} = 10^{-4}$.

Binary image segmentation grammar

$\Sigma = \{\mathsf{SEED}, \mathsf{FG}\}$.

$\Omega_{\mathsf{SEED}} = [1] \times [1]$.

$\Omega_{\mathsf{FG}} = [N] \times [M]$.

Rules:

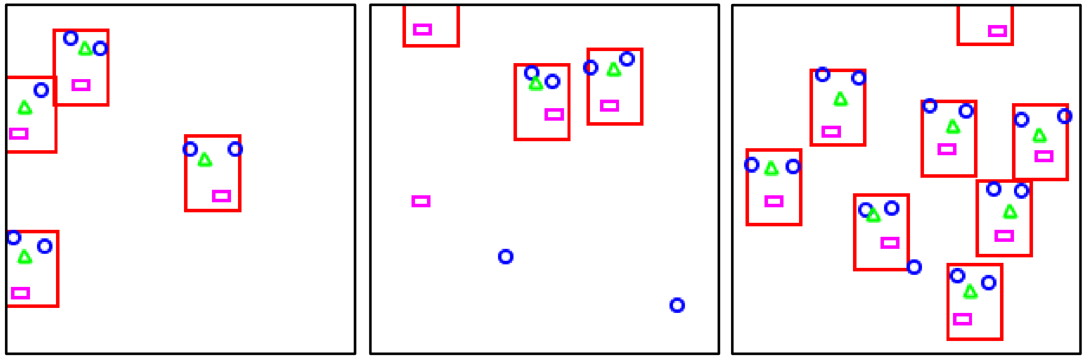
$$
\begin{array}{lll}
1.0, & (\mathsf{SEED}, \omega) & \rightarrow (\mathsf{FG}, \mathrm{UniformRect}((1, 1), (N, M))) \\
1.0, & (\mathsf{FG}, \omega) & \rightarrow (\mathsf{FG}, \mathrm{UniformBern}(\mathrm{Rect}(\omega - (1, 1), \omega + (1, 1)) \setminus \omega, 0.25))
\end{array}
$$

$\epsilon_{\mathsf{SEED}} = 1$,

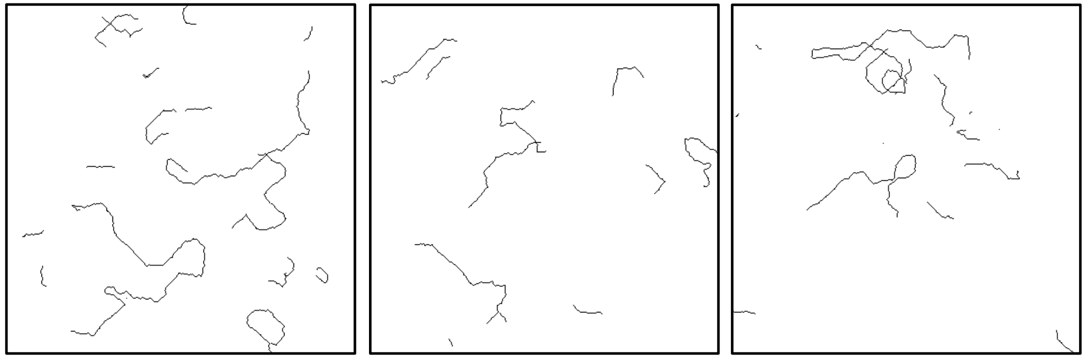
$\epsilon_{\mathsf{FG}} = 0$.

Probabilistic Scene Grammar Samples

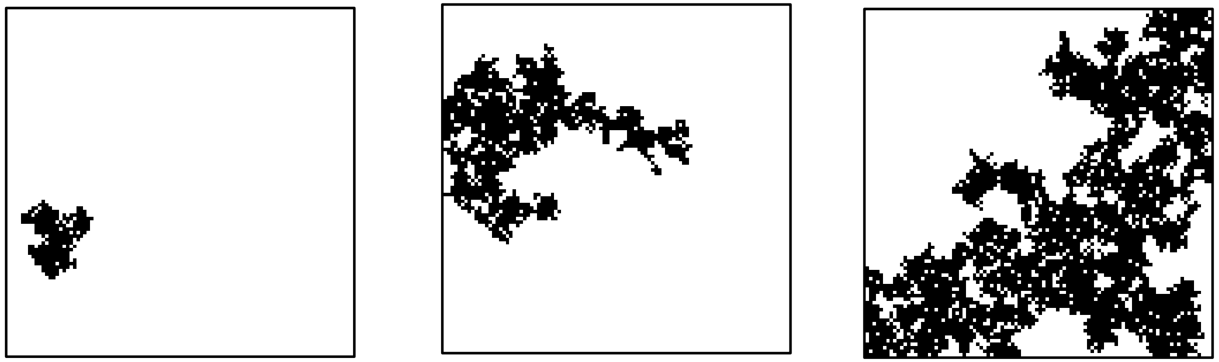
Face localization grammar



Contour detection grammar



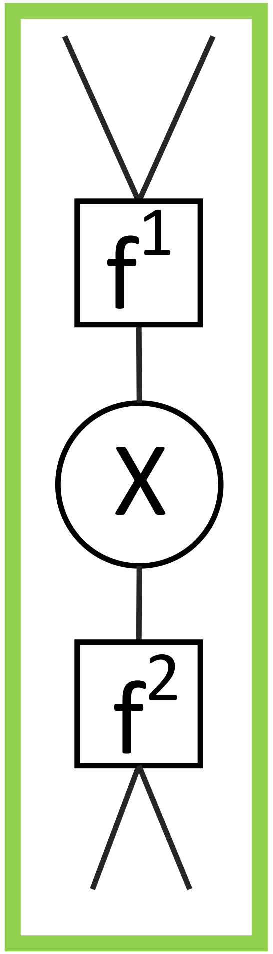
Binary image segmentation grammar



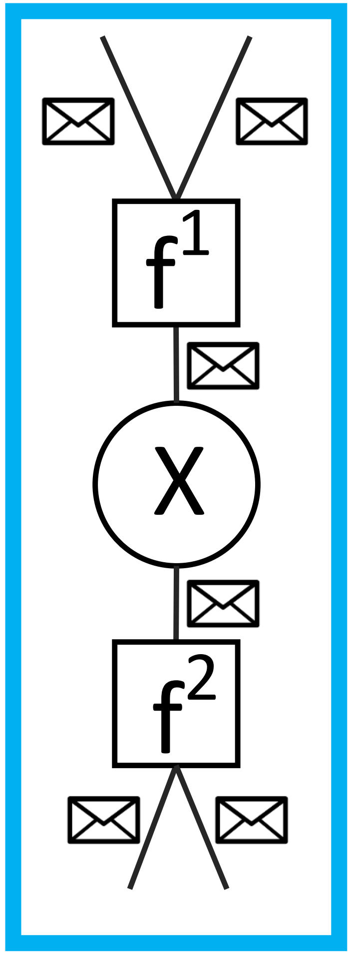
The Probabilistic Scene Grammar Framework

Probabilistic Scene Grammar

Factor graph

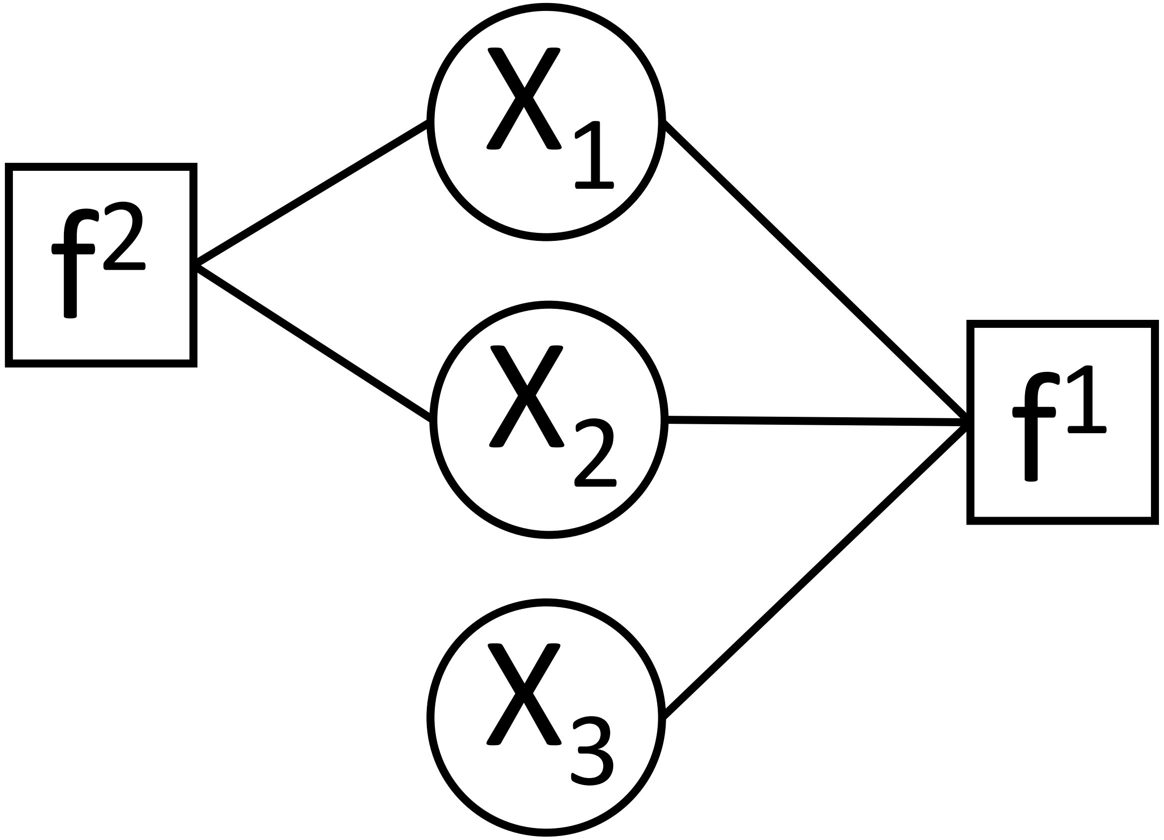


Inference



Learning

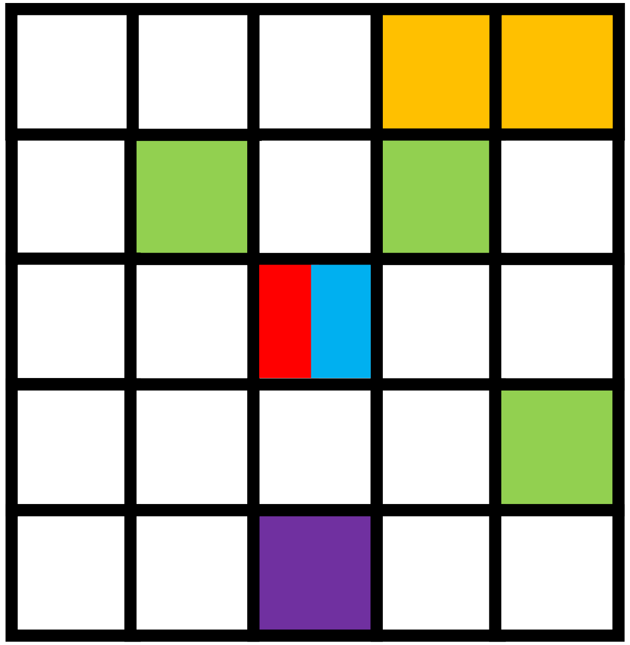
Factor Graphs



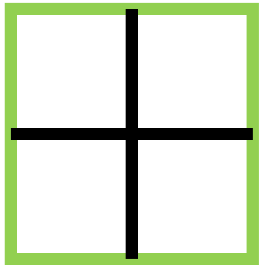
Encodes a distribution over random variables

Distributions of the PSG

Three types of distributions: 1) Categorical



Which child to choose ?

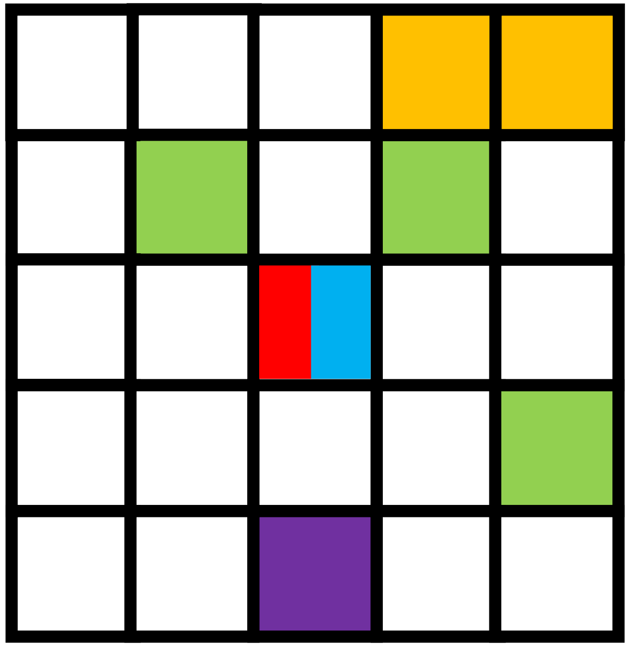


Which production rule?

E→L	0.5	-
E→Ø	0.5	-

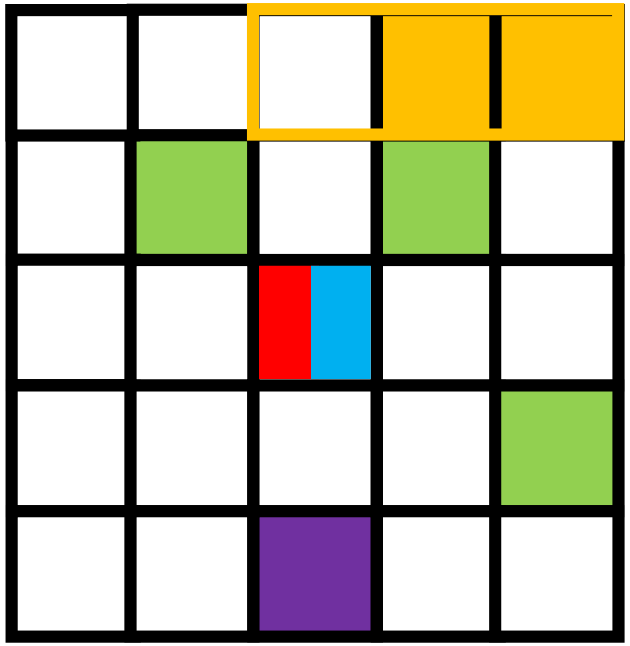
Distributions of the PSG

Three types of distributions: 2) Independent Bernoullis



Distributions of the PSG

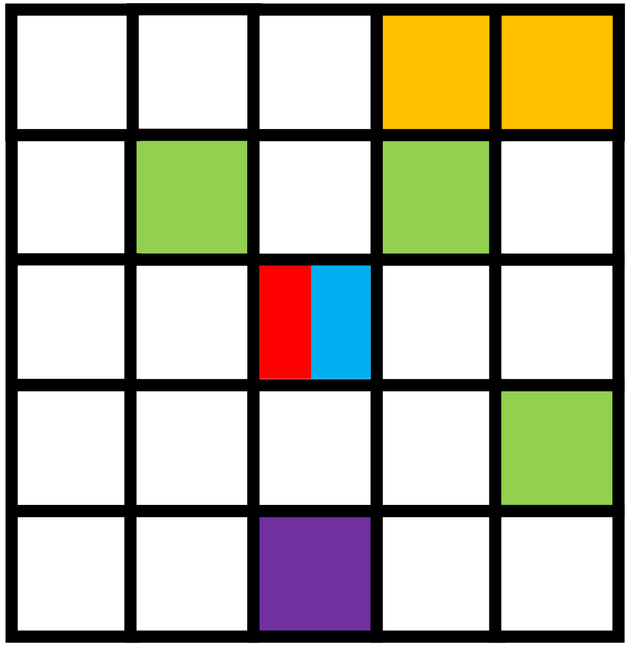
Three types of distributions: 2) Independent Bernoullis



Which children to choose ?

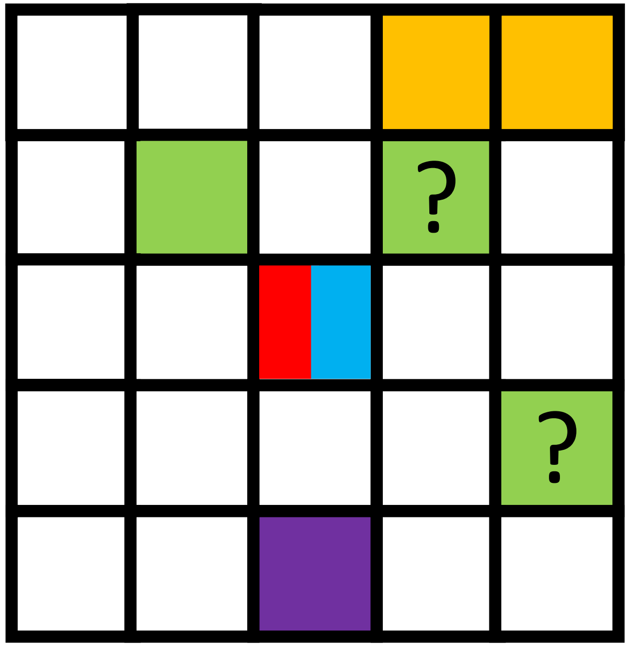
Distributions of the PSG

Three types of distributions: 3) Leaky-or

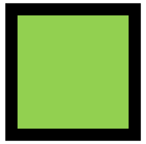


Distributions of the PSG

Three types of distributions: 3) Leaky-or

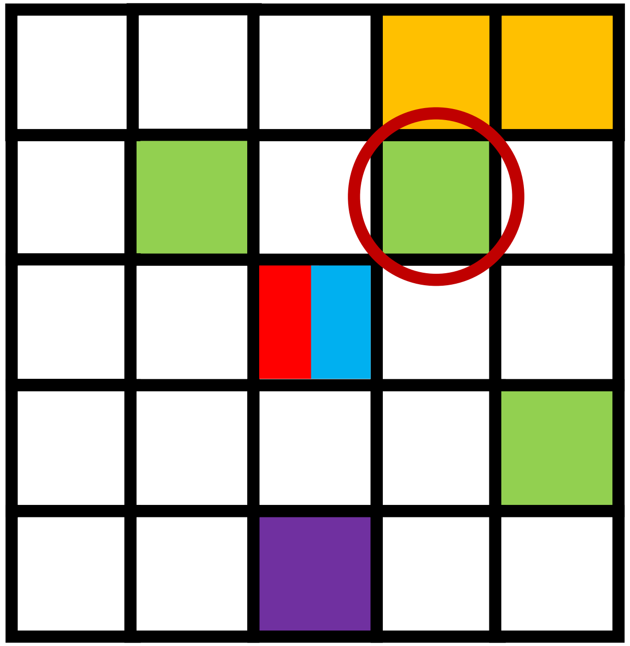


Object present?

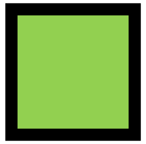


Distributions of the PSG

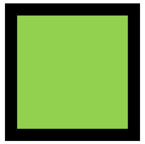
Three types of distributions: 3) Leaky-or



Object present?



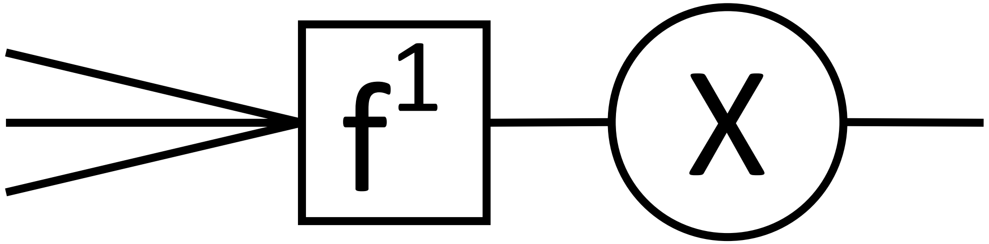
Framework: As a Factor Graph

 =

Binary random variables:

Factors:

Framework: As a Factor Graph

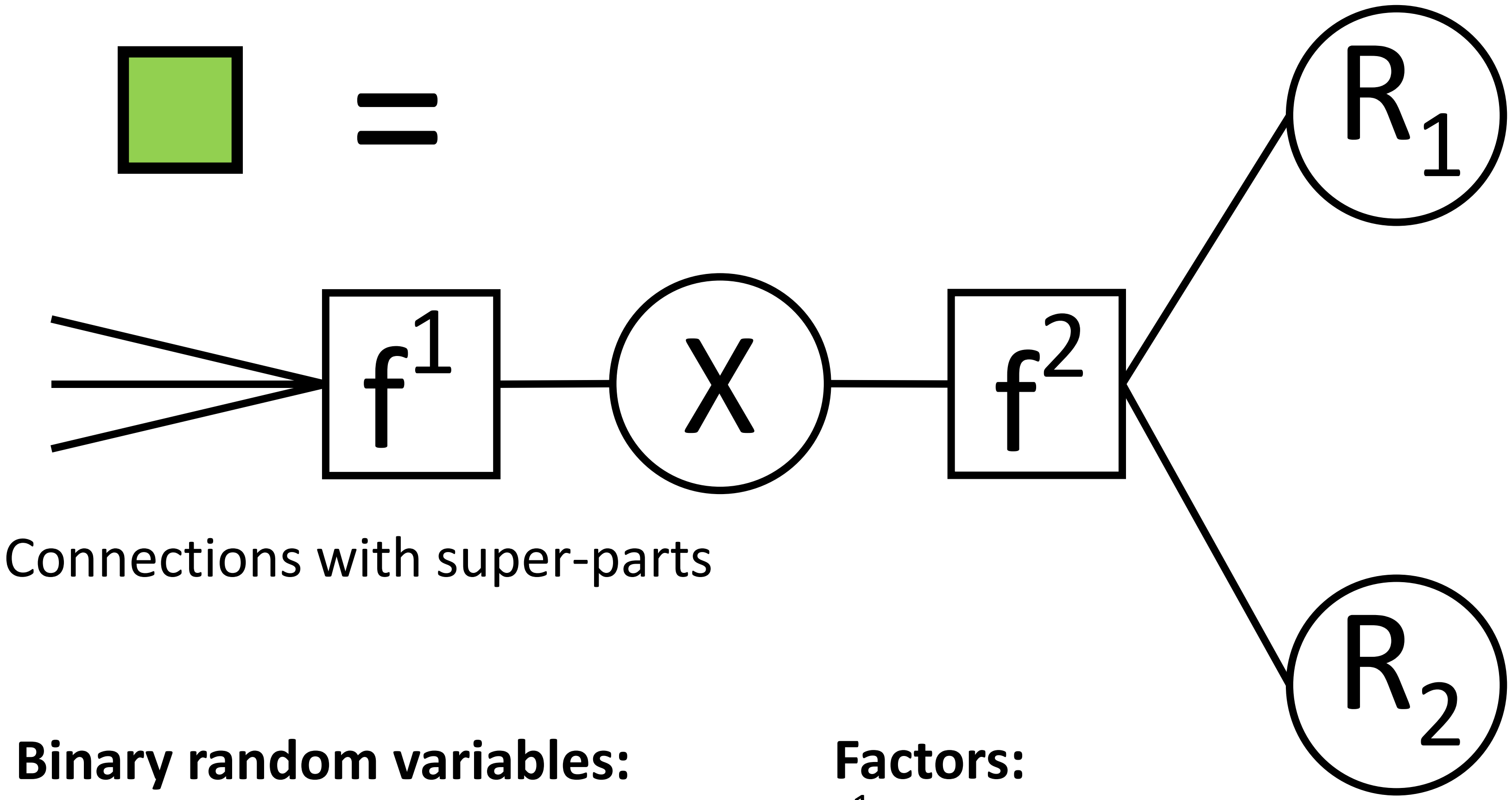


Connections with super-parts

Binary random variables:
X: Presence/absence of object

Factors:
f^1: Leaky-or factor

Framework: As a Factor Graph



Binary random variables:

X: Presence/absence of object

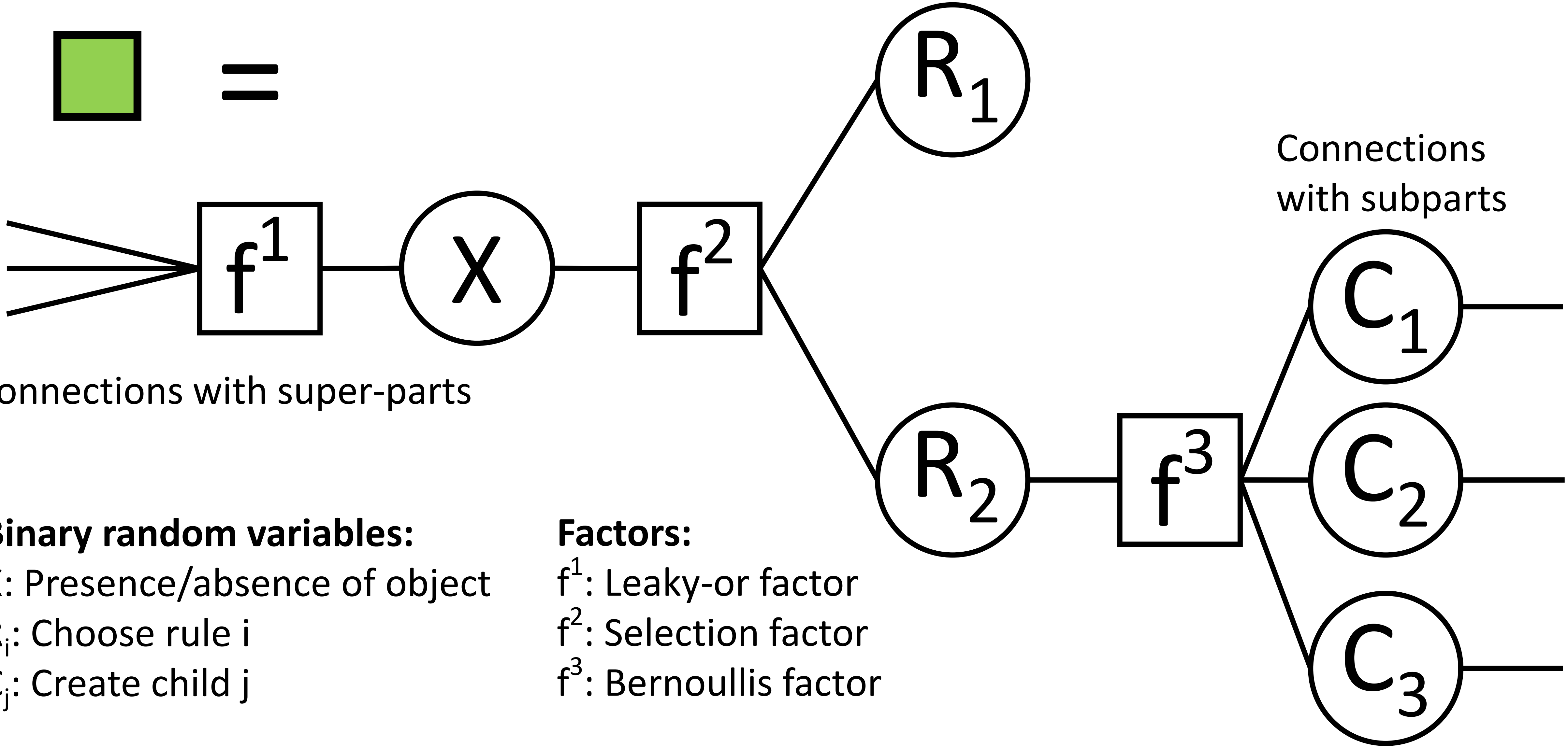
R_i: Choose rule i

Factors:

f^1: Leaky-or factor

f^2: Selection factor

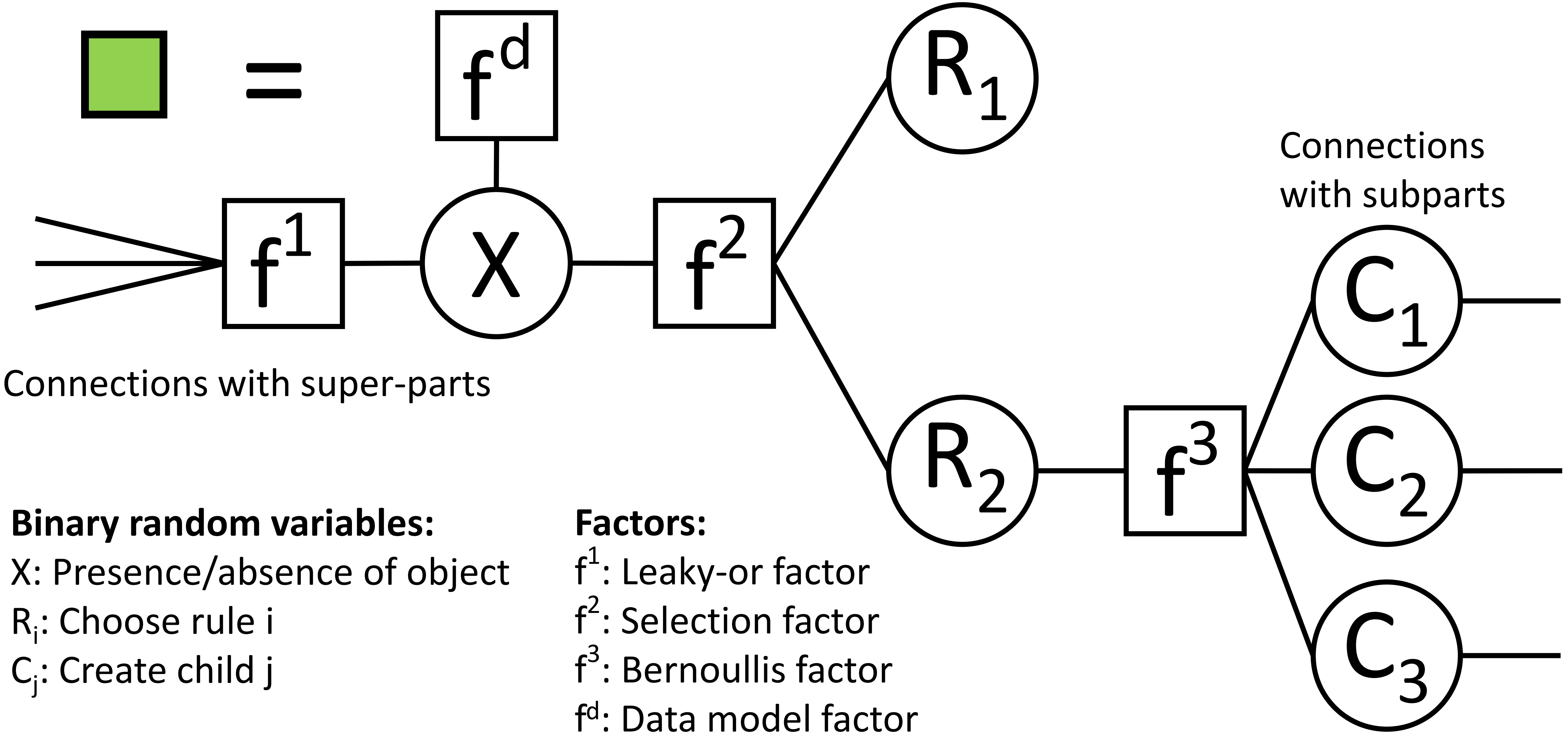
Framework: As a Factor Graph



Binary random variables:
X: Presence/absence of object
R_i: Choose rule i
C_j: Create child j

Factors:
f^1: Leaky-or factor
f^2: Selection factor
f^3: Bernoullis factor

Framework: As a Factor Graph



Binary random variables:

X: Presence/absence of object

R_i: Choose rule i

C_j: Create child j

Factors:

f^1: Leaky-or factor

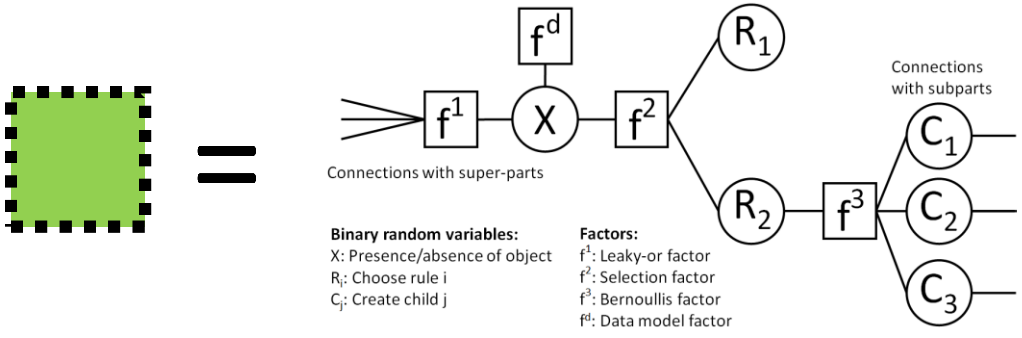
f^2: Selection factor

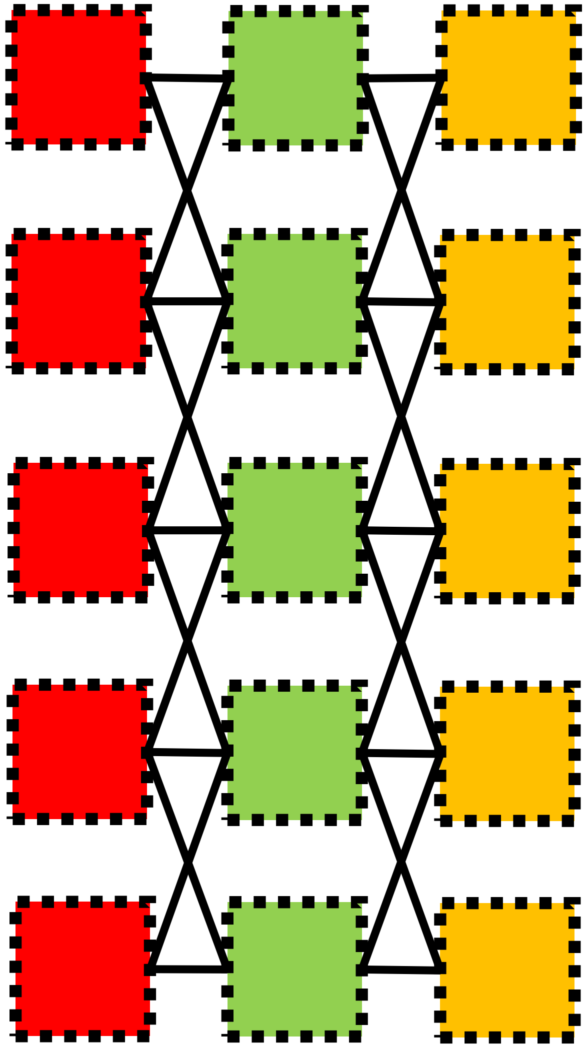
f^3: Bernoullis factor

f^d: Data model factor

Framework: As a Factor Graph

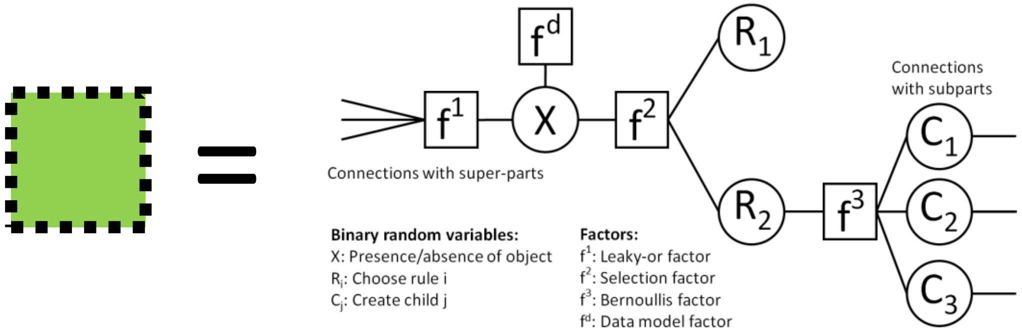
- Symbols: Face, Eye, Eyelashes
- 1D pose spaces
- Spatial neighbourhoods:
  - Face(y) → Eye(y'), y' = {y-1,y,y+1}
  - Eye(y) → EyeLashes(y'), y' = {y-1,y,y+1}

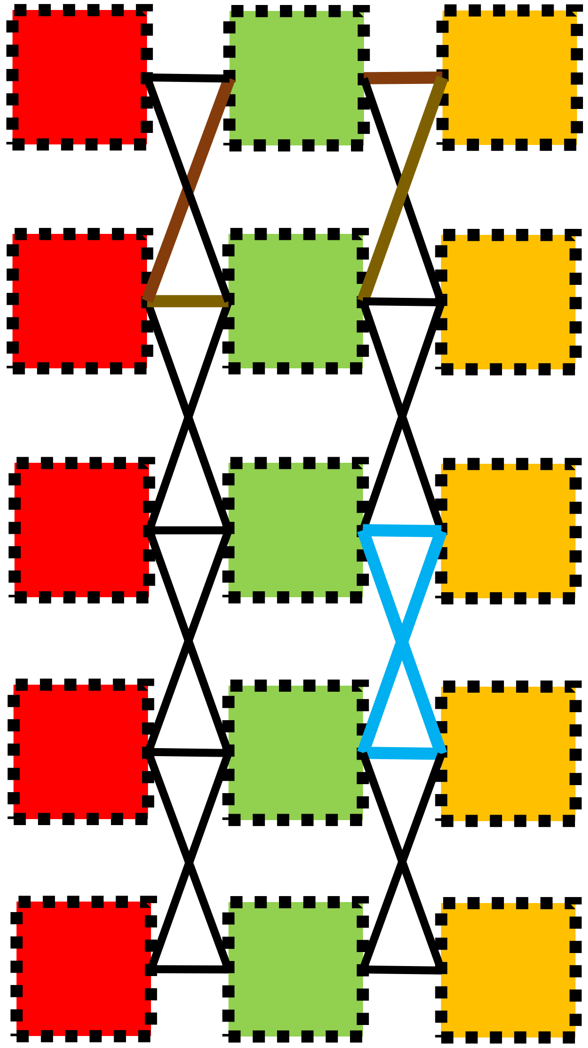




Framework: As a Factor Graph

- Symbols: Face, Eye, Eyelashes
- 1D pose spaces
- Spatial neighbourhoods:
  - Face(y) → Eye(y’), y’ = {y-1,y,y+1}
  - Eye(y) → EyeLashes(y’), y’ = {y-1,y,y+1}

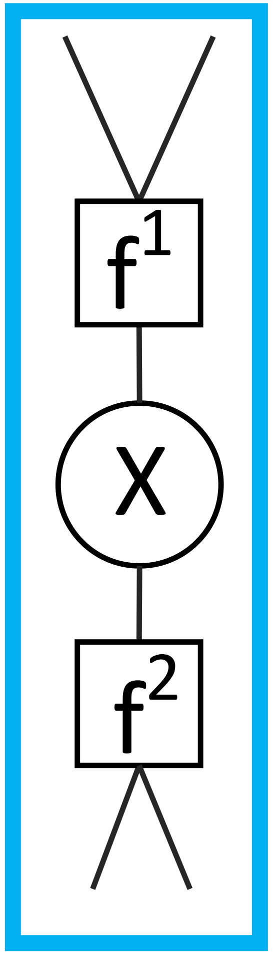




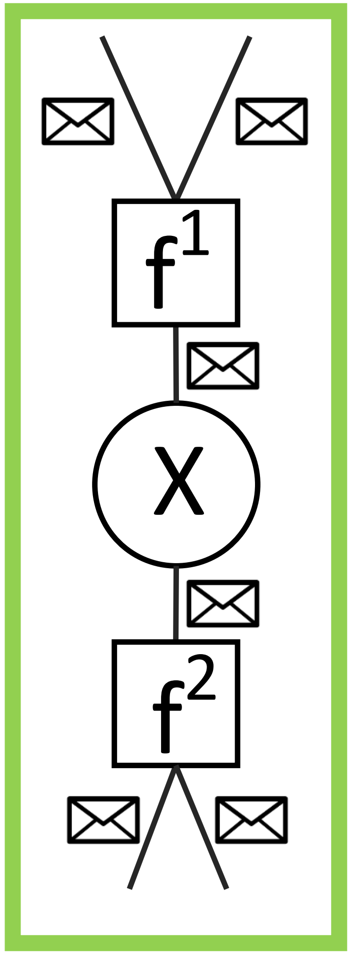
The Probabilistic Scene Grammar Framework

Probabilistic Scene Grammar

Factor graph



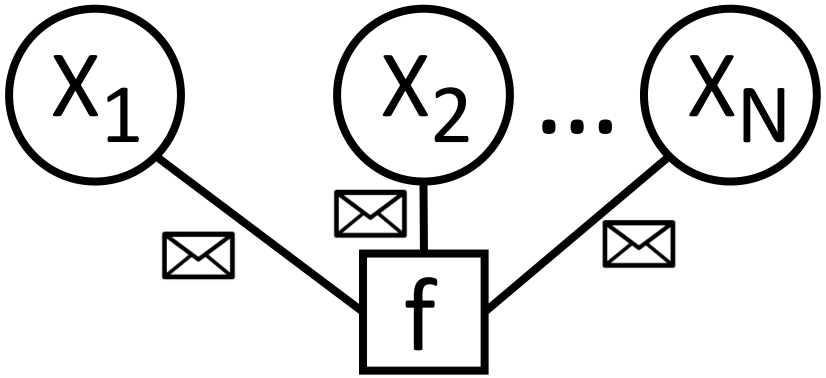
Inference



Learning

Approximate Inference

- Given an image, what objects are in the image and what are their parts?
- Run Loopy Belief Propagation on the factor graph
  - Compute (approximate) posterior quantities, $\hat{p}(\cdot|\text{Image})$



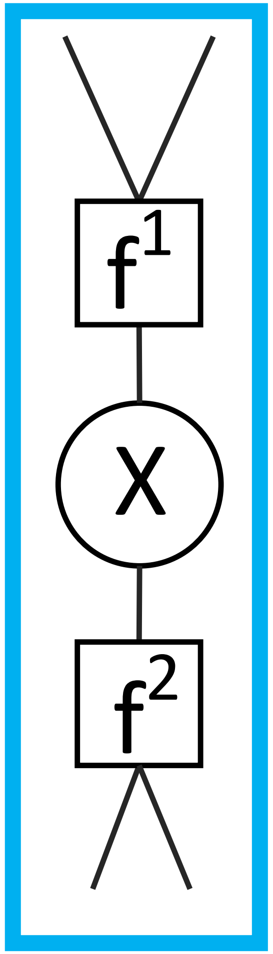
$$\mu_{f \to x_i}(x_i) = \sum_{x_1} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_N} f(x_1, \ldots, x_N) \prod_{j \neq i} \mu_{x_j \to f}(x_i)$$

  - In general: **exponential** time. Our case: **linear** time.

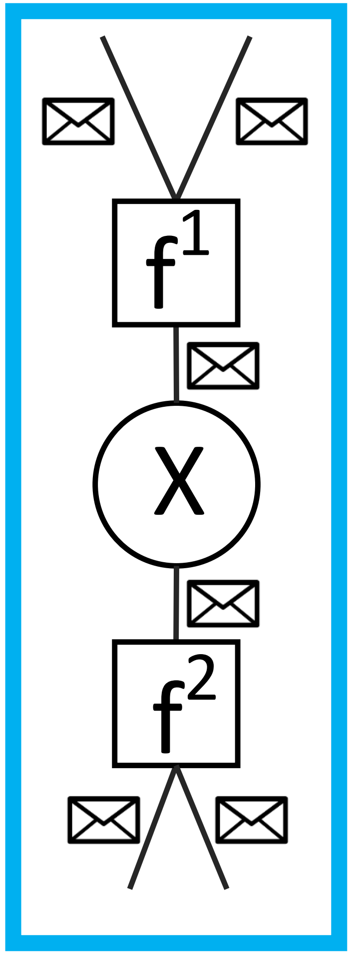
The Probabilistic Scene Grammar Framework

Probabilistic Scene Grammar

Factor graph



Inference



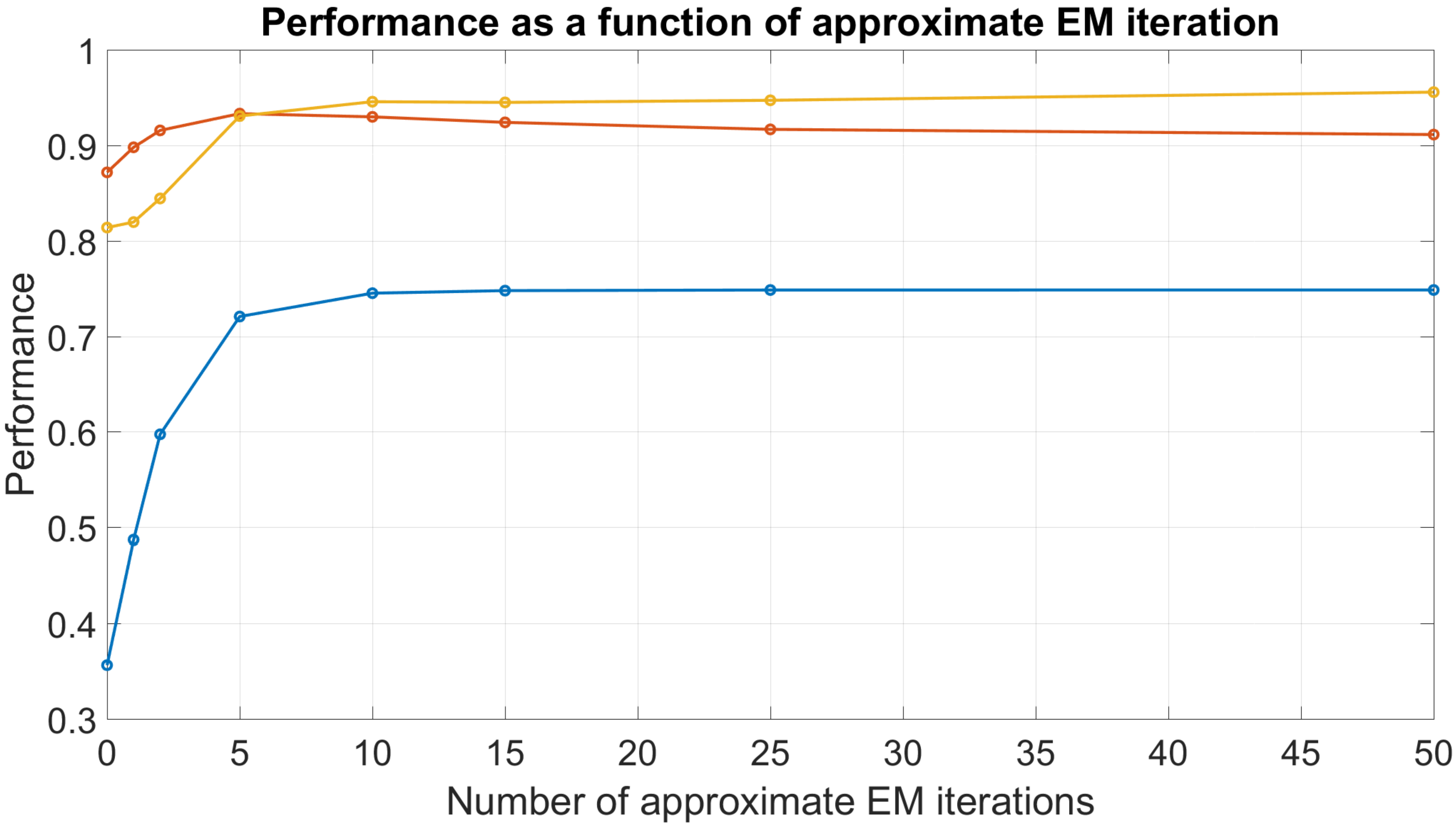
Learning

Learning: Approximate EM algorithm

- Goal: Estimate model parameters
  - Rule probabilities
  - Geometric relationships
  - Leaky-or parameter

- **Maximization-step**: sums involving posterior quantities
- Exact posterior quantities are intractable to compute
- **Expectation-step**: use approximate posteriors computed by Loopy Belief Propagation

Learning: Approximate EM algorithm



This talk

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

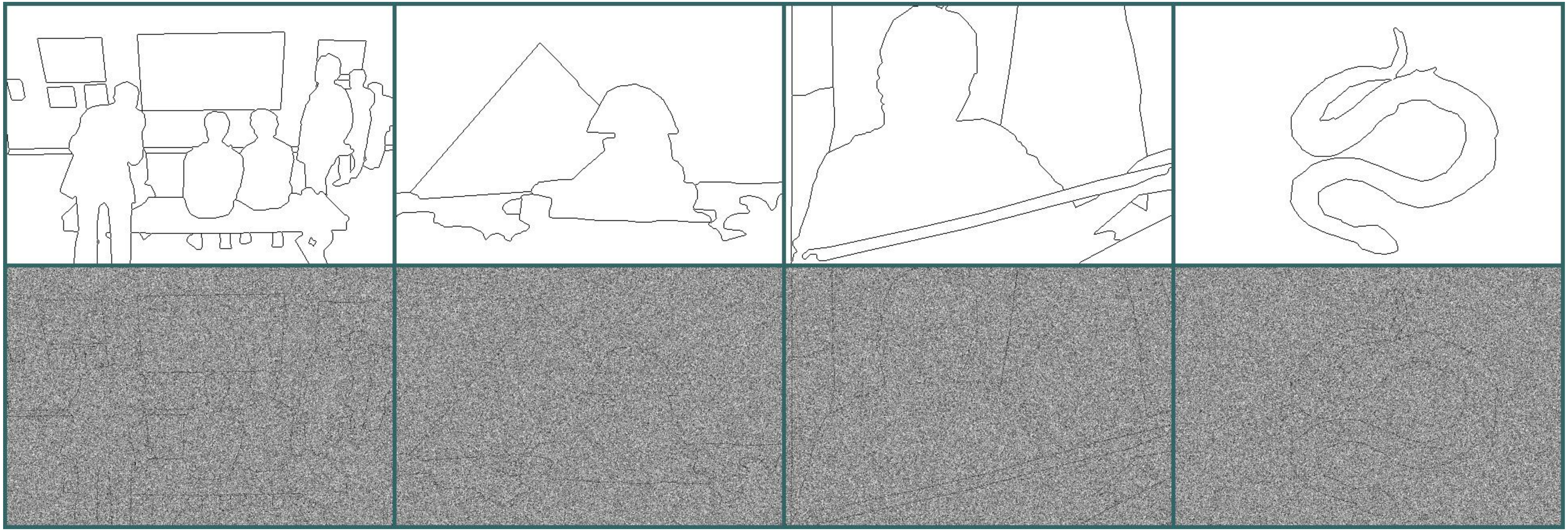
PSG

This talk

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

PSG framework is competitive with some specialized frameworks

Application: contour detection



Application: contour detection

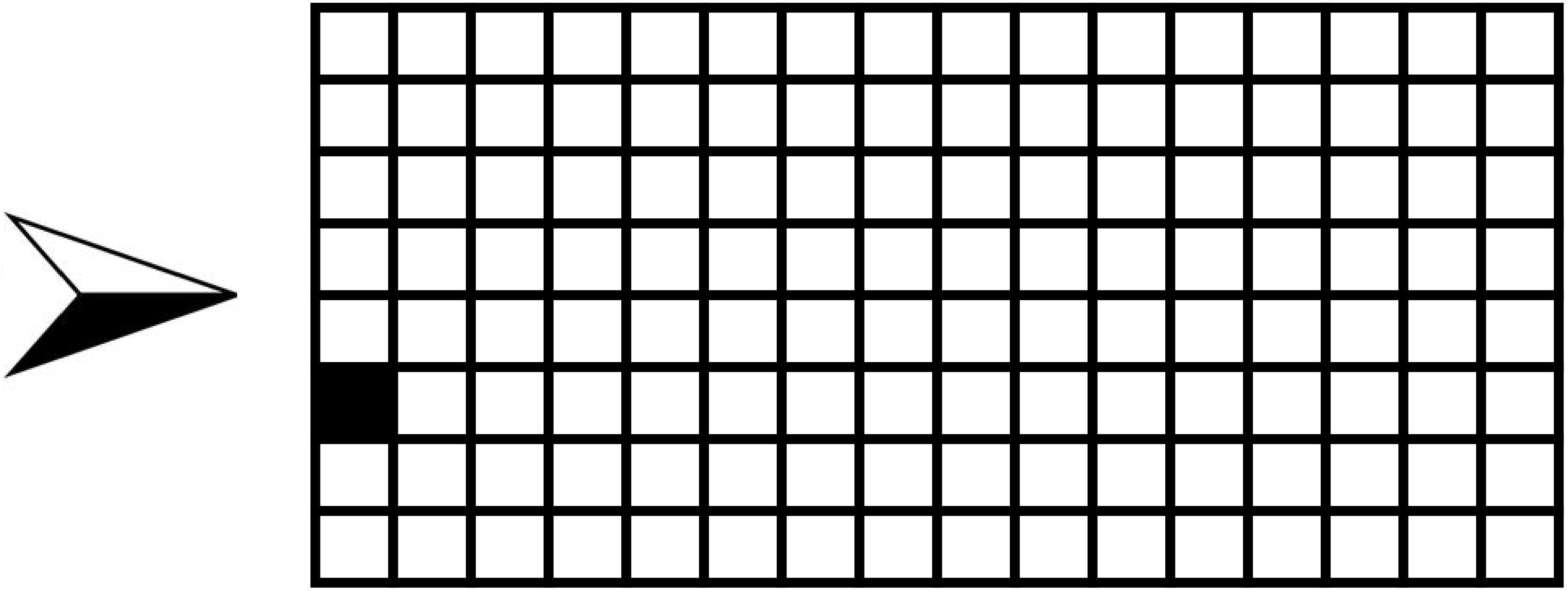
- Dataset:
  - Ground truth: human-drawn object boundary contours from Berkeley Segmentation Dataset [1]
  - B(x,y): Binary value of whether this pixel belongs to a contour
  - D(x,y): Pixel intensity
  - Data-model: $D(x,y) \sim N(\mu_{B(x,y)}, \sigma)$

[1] Arbelaez, Maire, Fowlkes, Malik. "Contour Detection and hierarchical image segmentation", PAMI 2011.

Application: contour detection

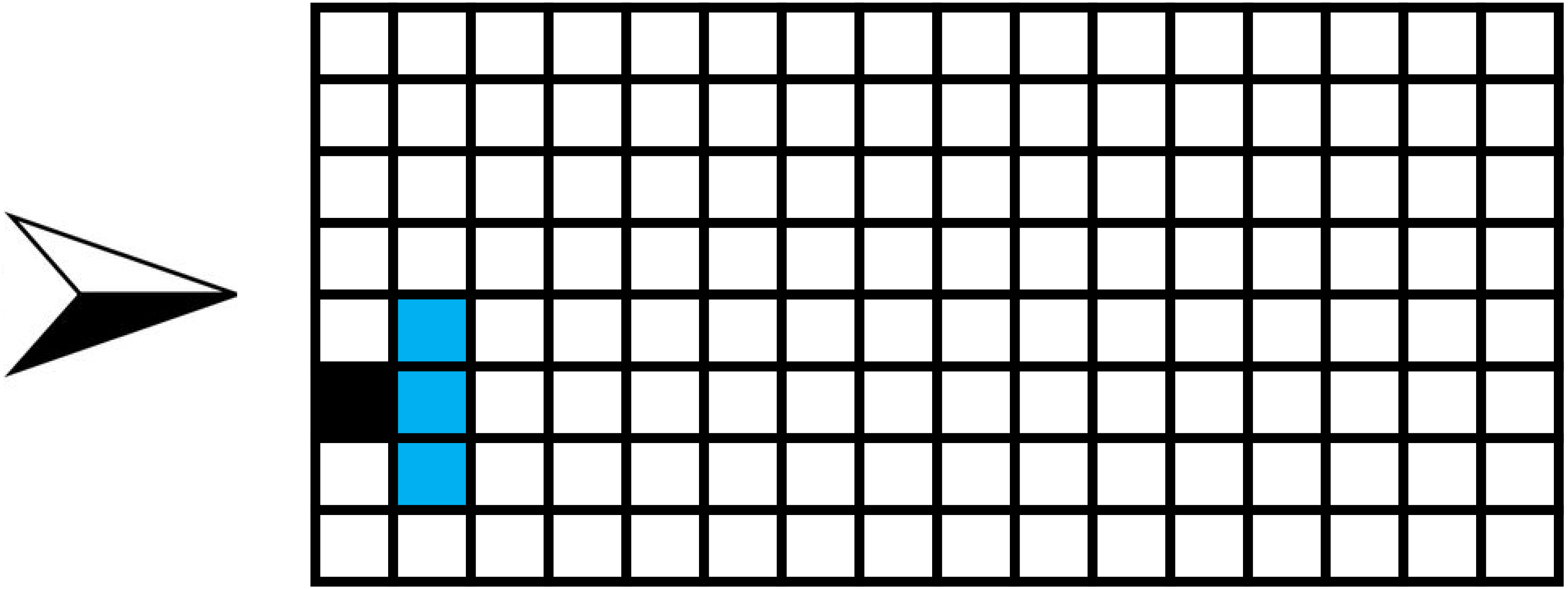
- Model:
  - Simple grammar model (next slides)
  - Factor graph contains $\sim$50M edges
- Training:
  - Model parameters estimated using approximate EM algorithm
  - 200 train, 200 test

Generating a curve

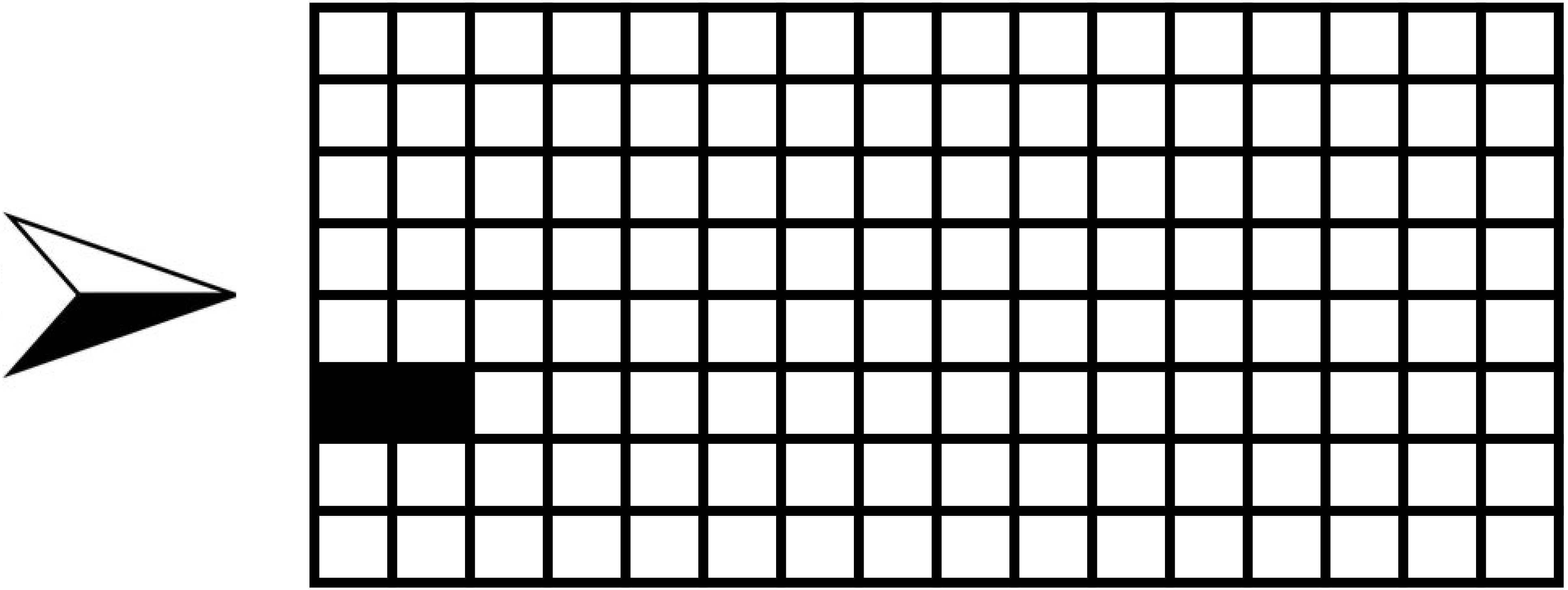


Generating a curve

Choose: continue contour, change orientation, or STOP

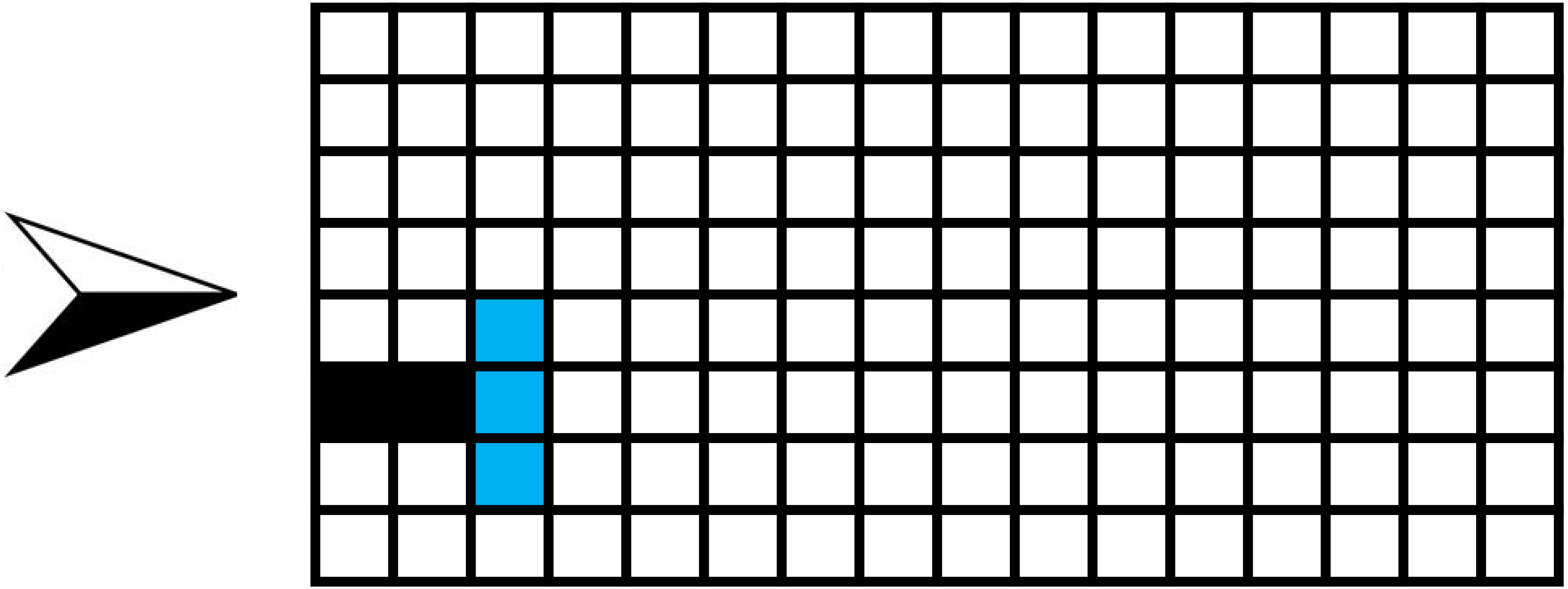


Generating a curve

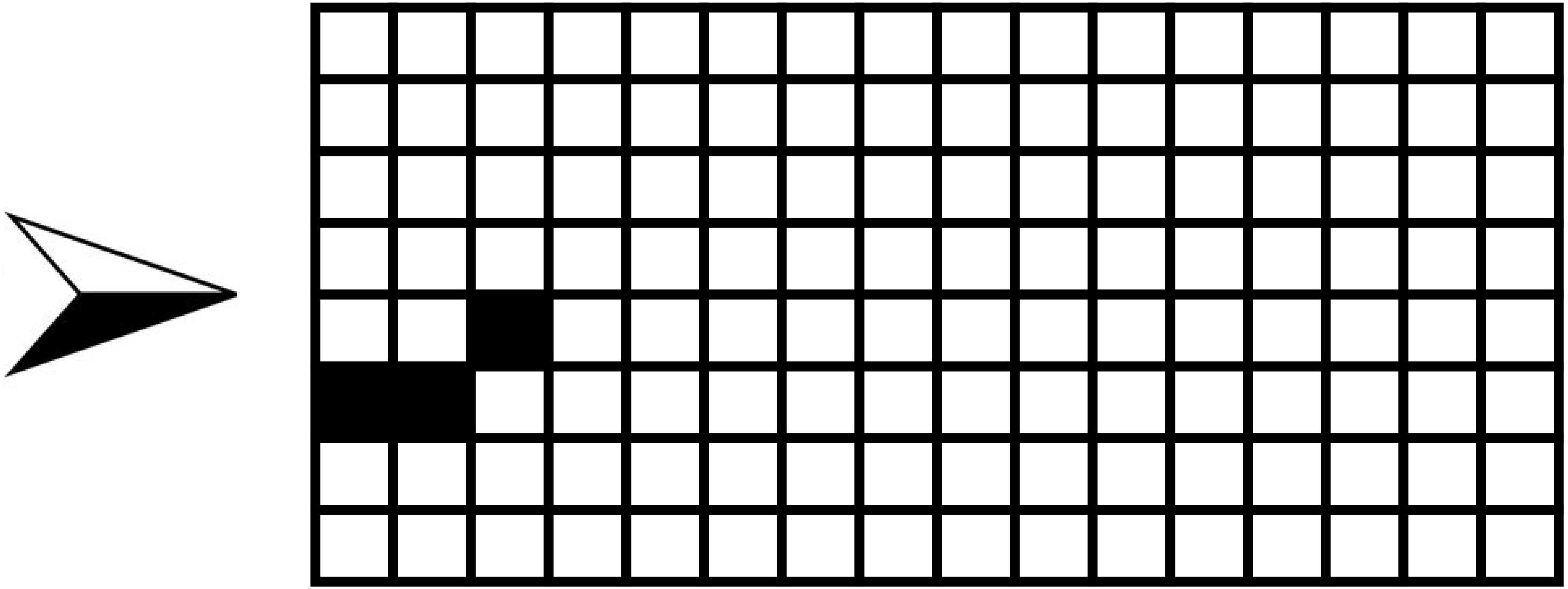


Generating a curve

Choose: continue contour, change orientation, or STOP

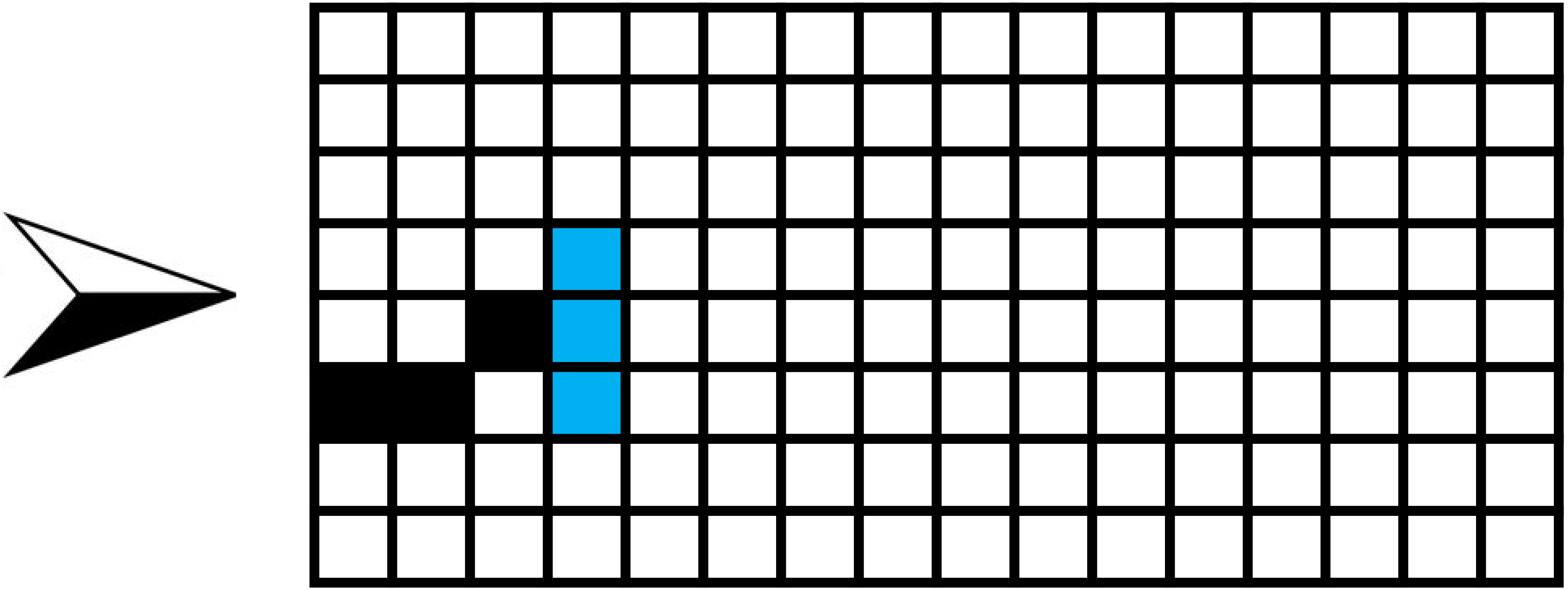


Generating a curve



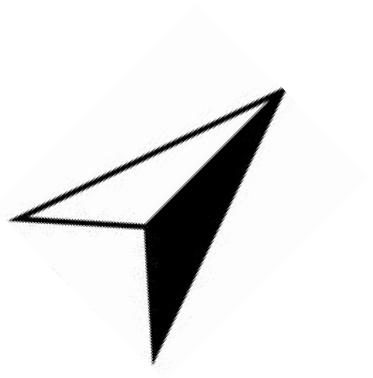
Generating a curve

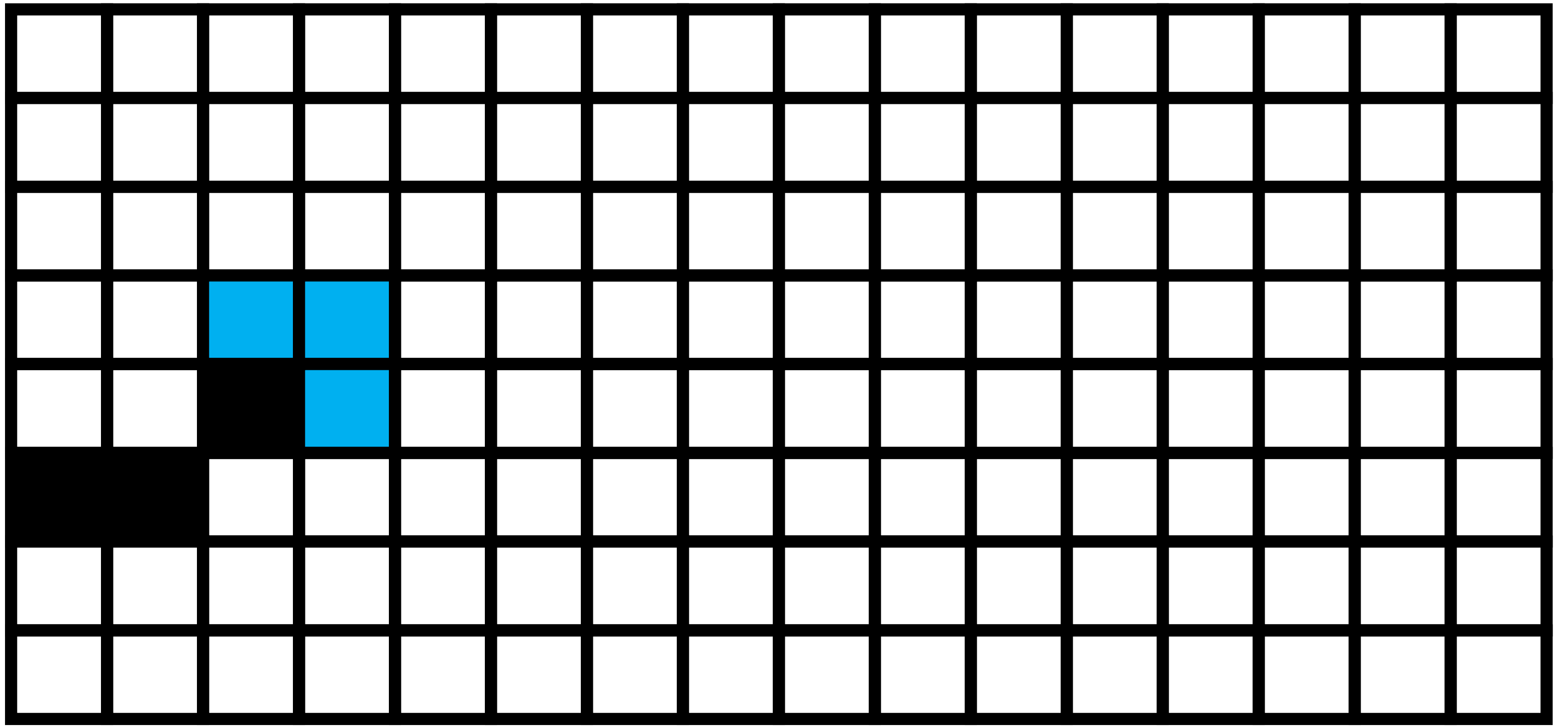
Choose: continue contour, change orientation, or STOP



Generating a curve

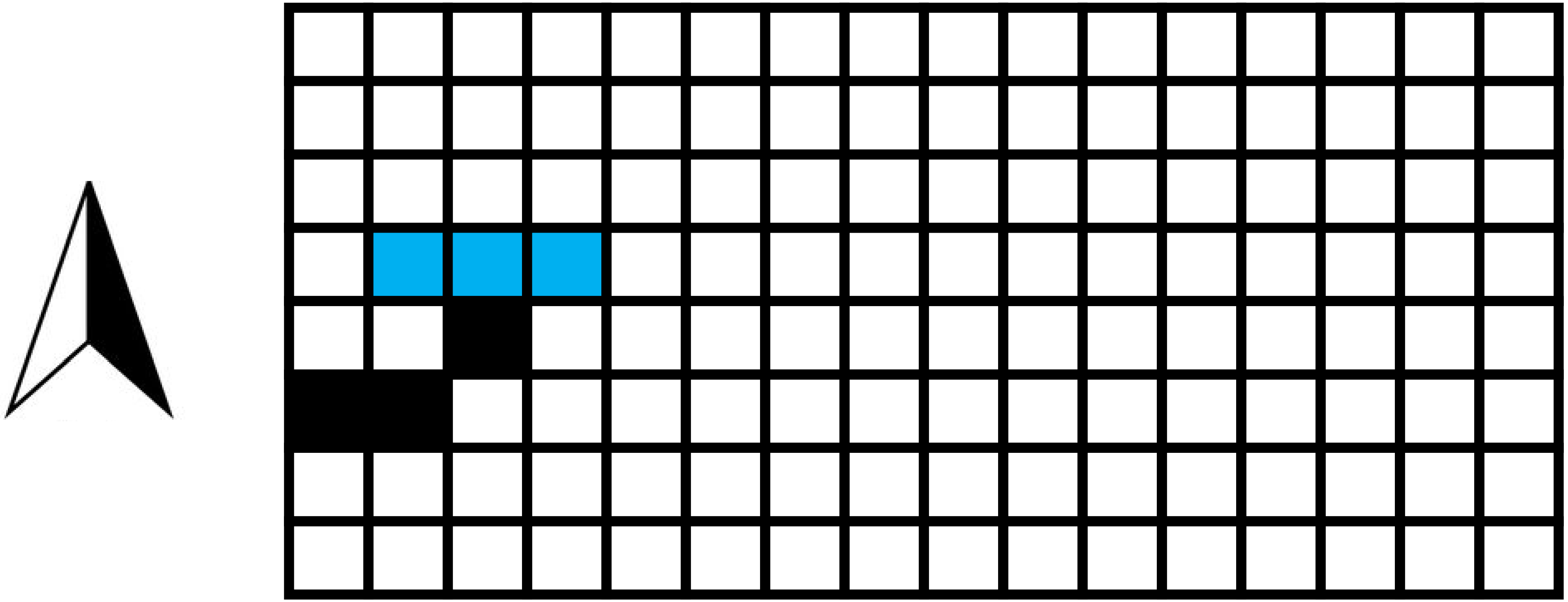
Choose: continue contour, change orientation, or STOP



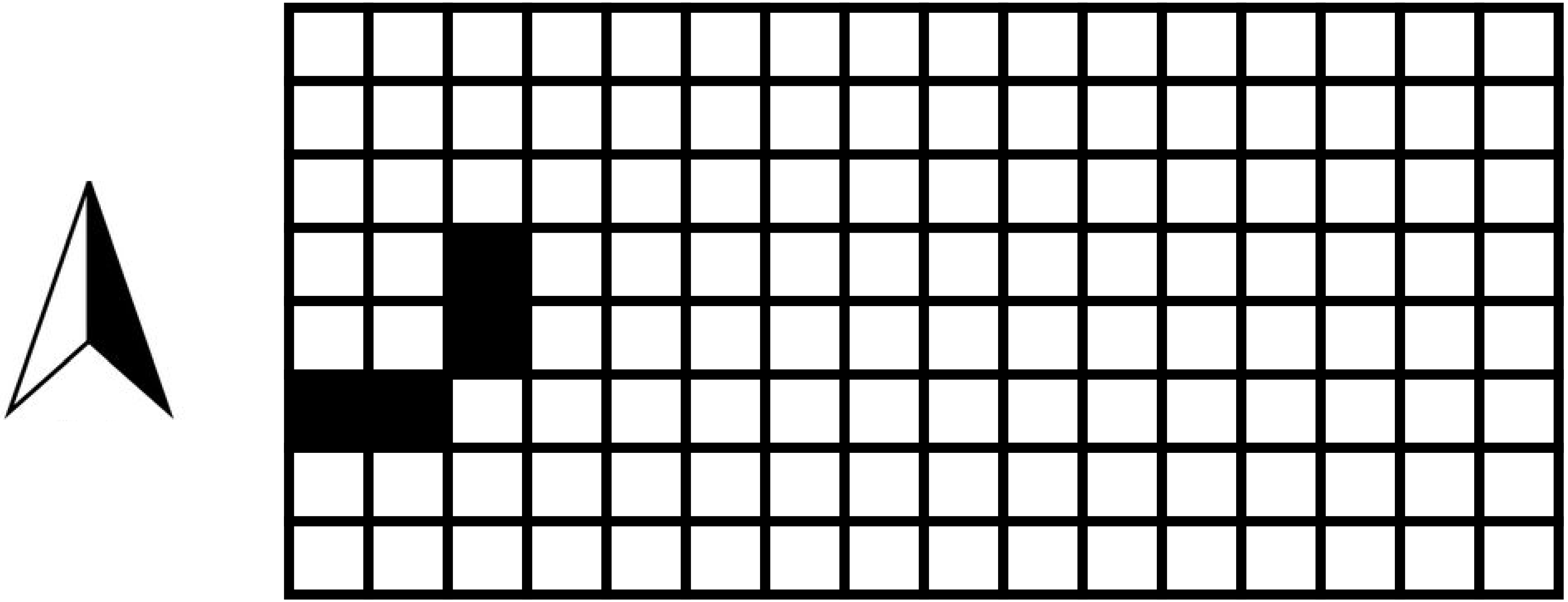


Generating a curve

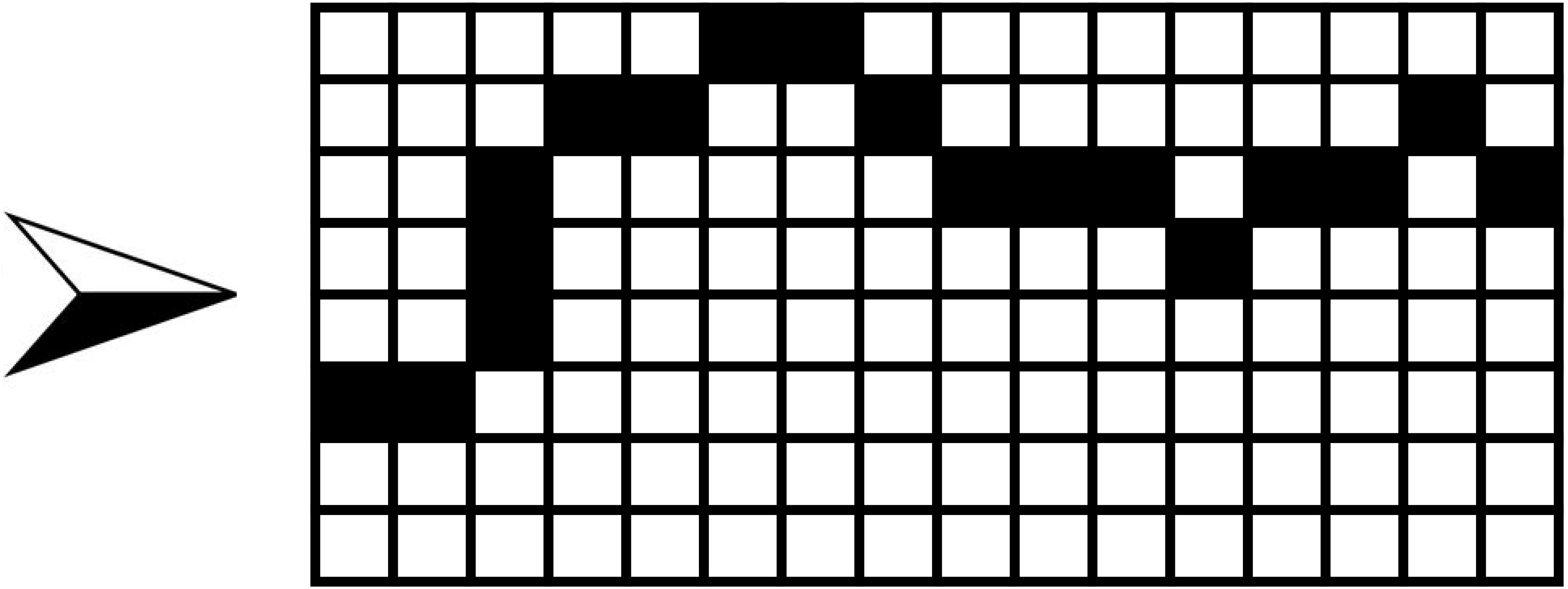
Choose: continue contour, change orientation, or STOP



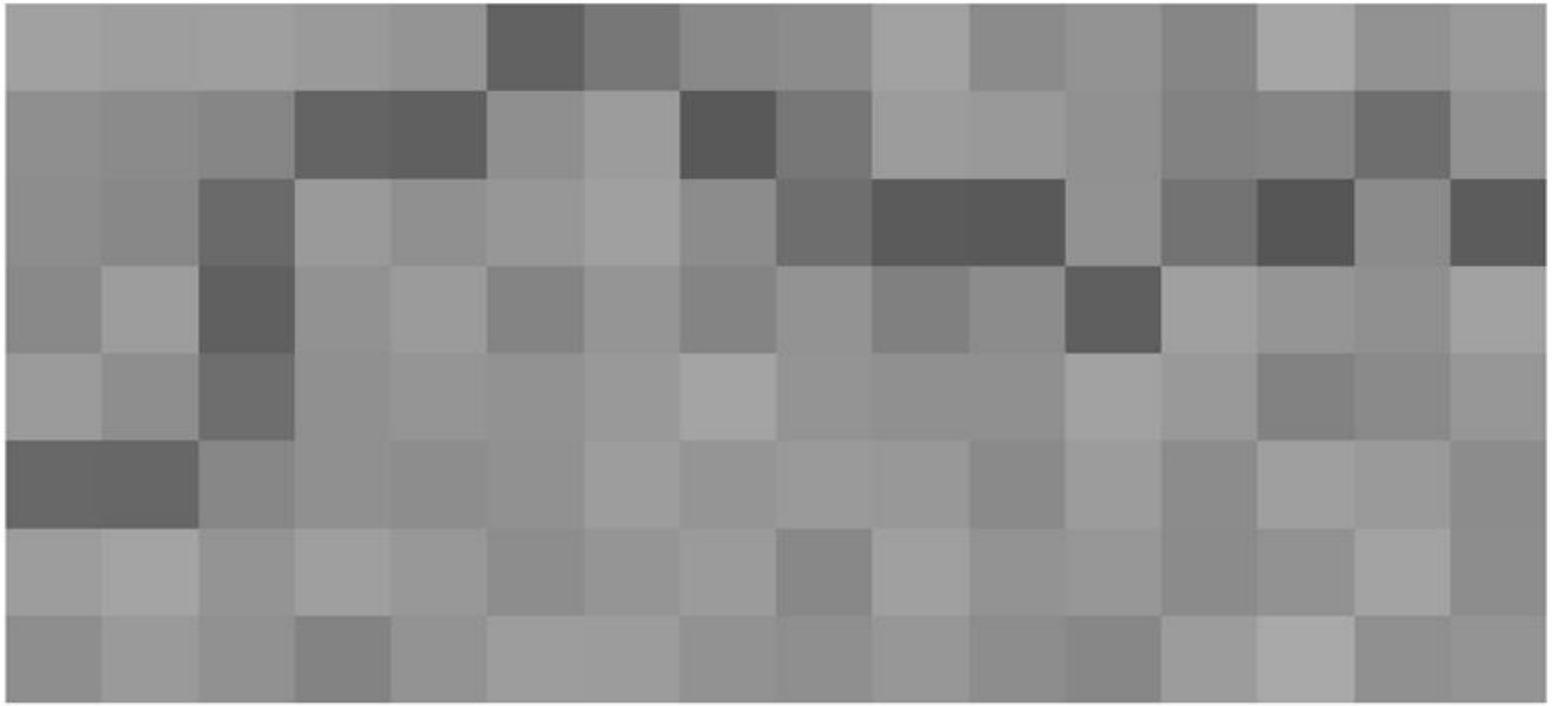
Generating a curve



Generating a curve



Generating a curve

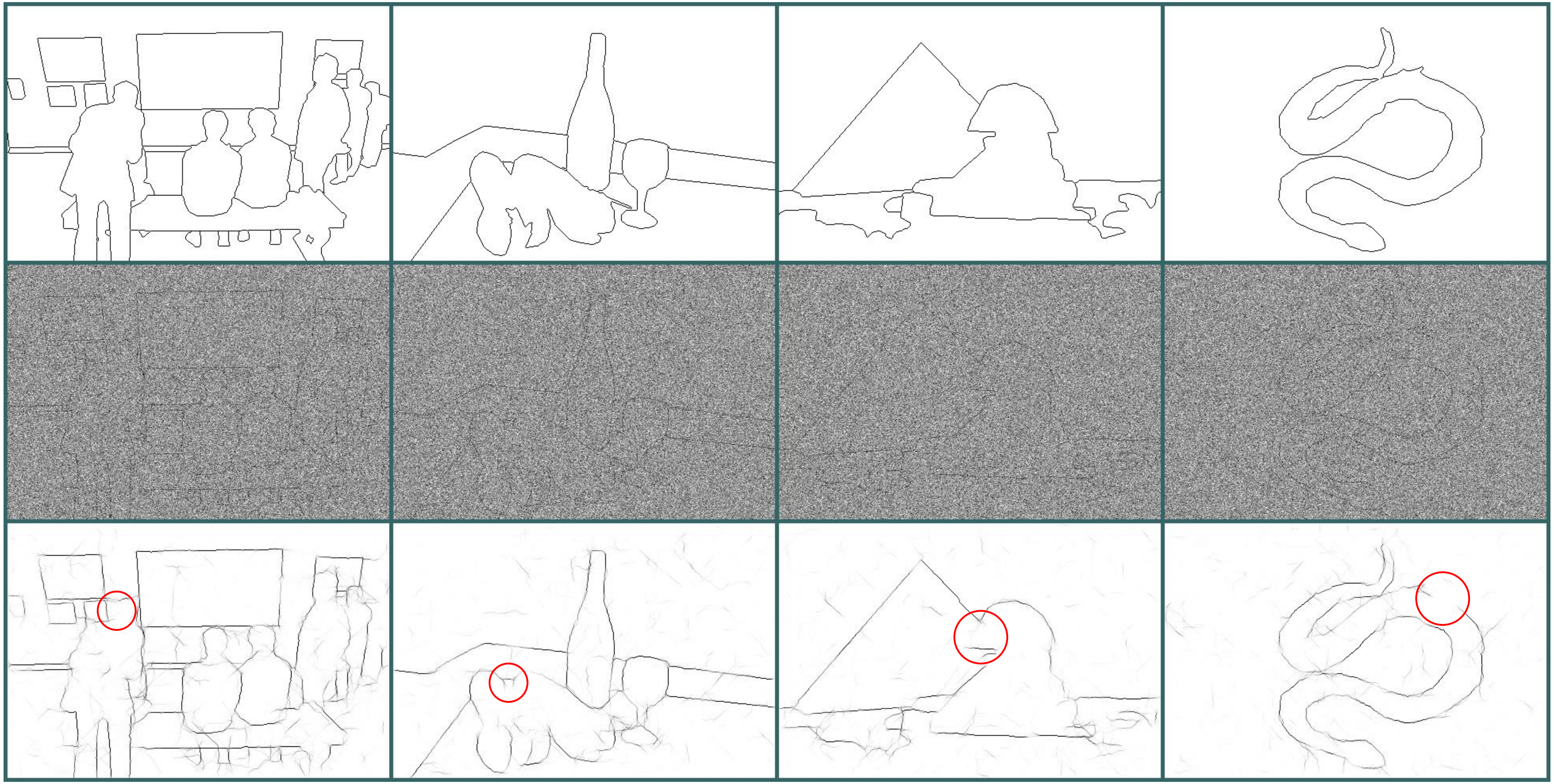


Inferred contours

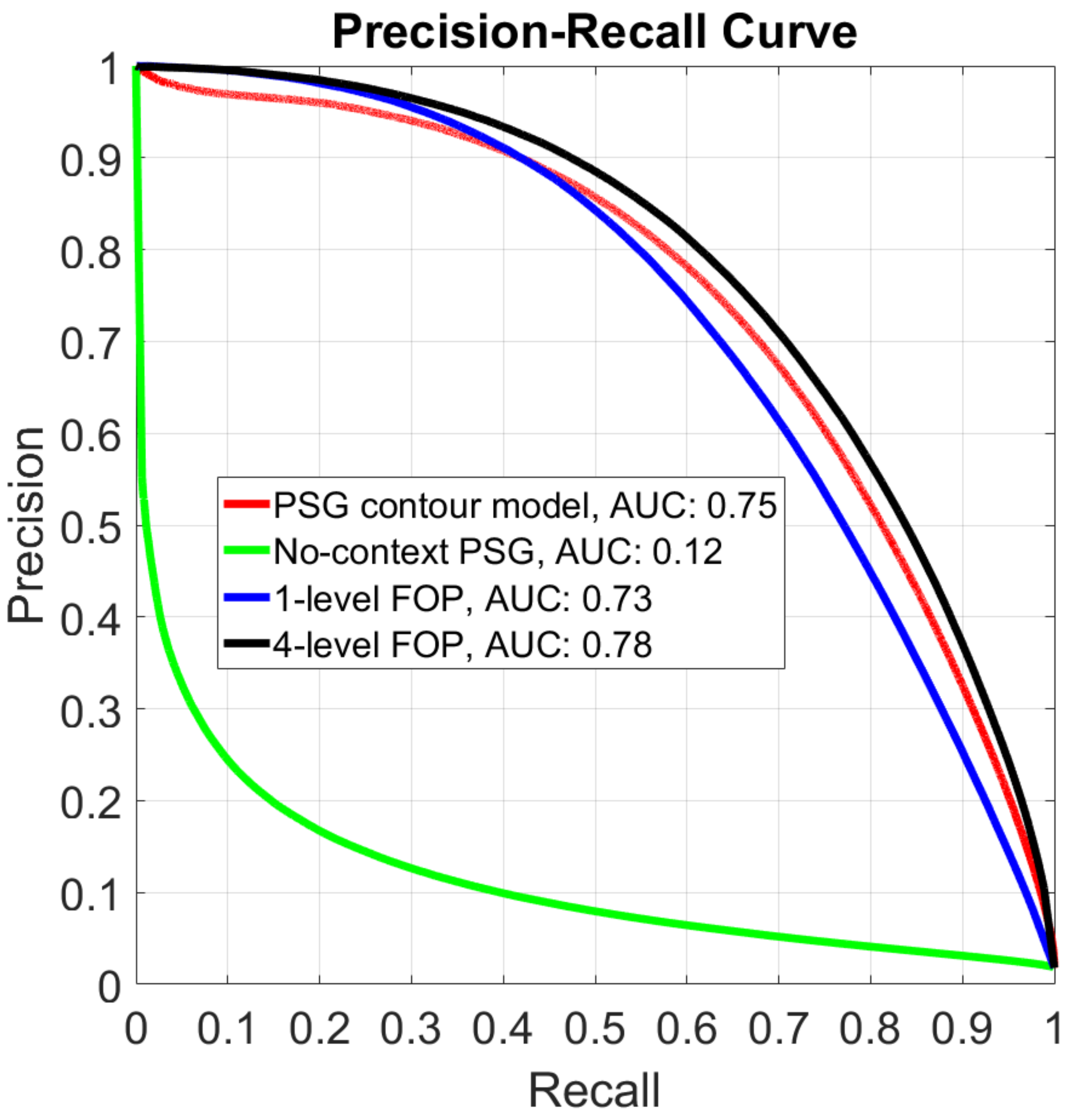
Ground truth

Image data

P(B = 1 | Image)



Contour detection: Comparison

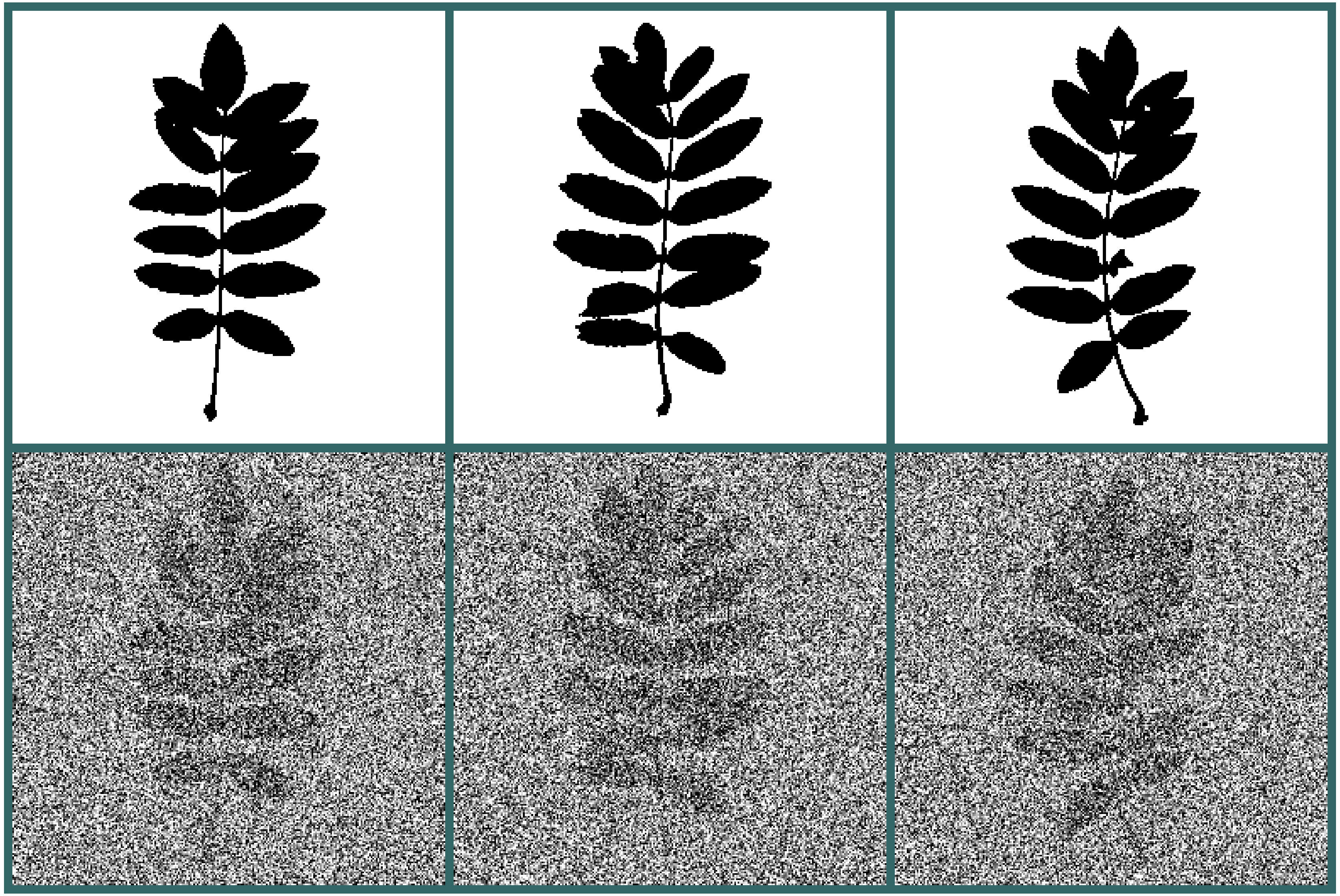


Context is important!

PSG framework competitive with Field-of-Patterns (FOP)

1-level FOP and 4-level FOP from: "Multiscale Field of Patterns", Felzenszwalb, Oberlin. NIPS 2014.

Application: binary image segmentation



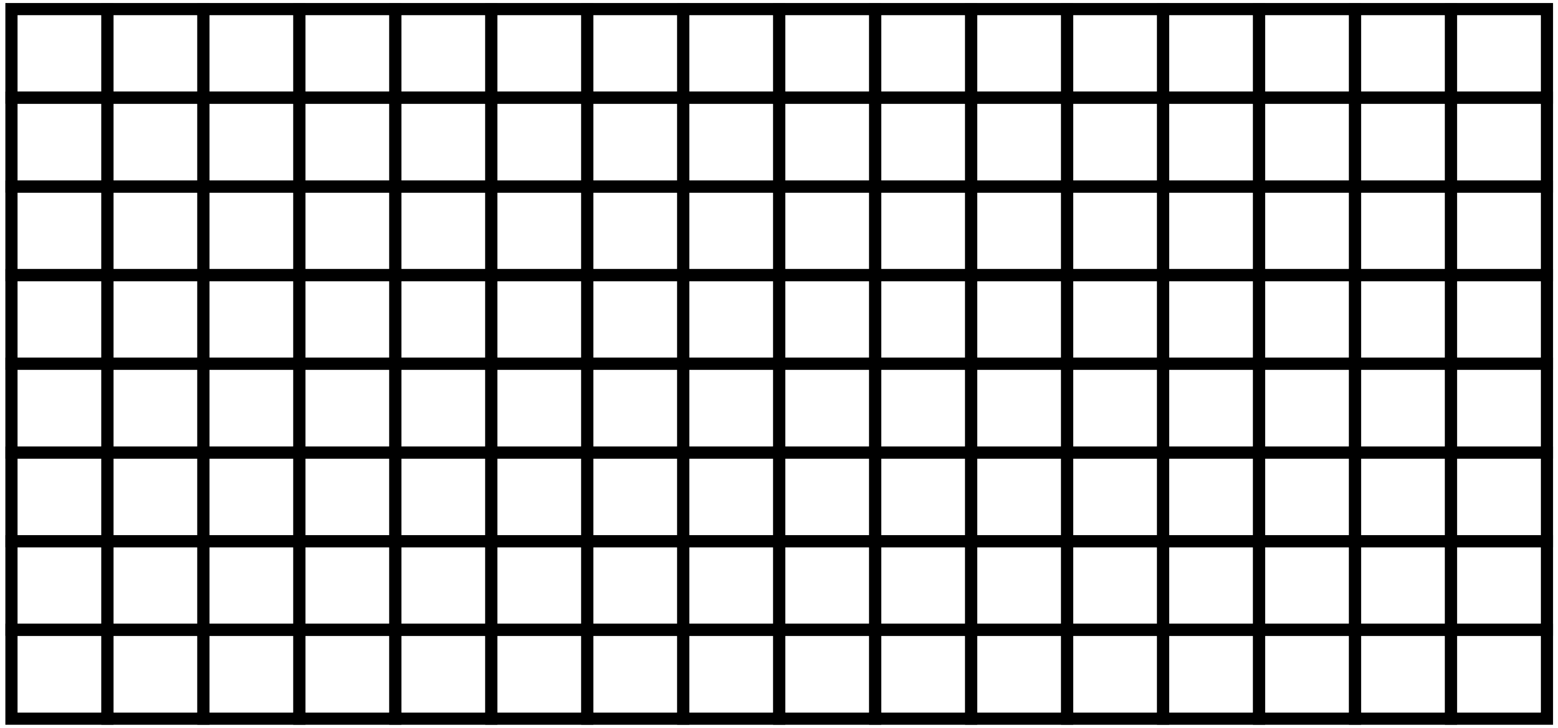
Application: binary image segmentation

- Dataset:
  - Ground truth: binary leaf masks [1]
  - $B(x,y)$: Binary value of whether this pixel belongs to a leaf
  - $D(x,y)$: Pixel intensity
  - Data-model: $D(x,y) \sim N(\mu_{B(x,y)}, \sigma)$

[1] Soderkvist. “Computer vision classification of leaves from Swedish trees”, Master’s thesis 2011.

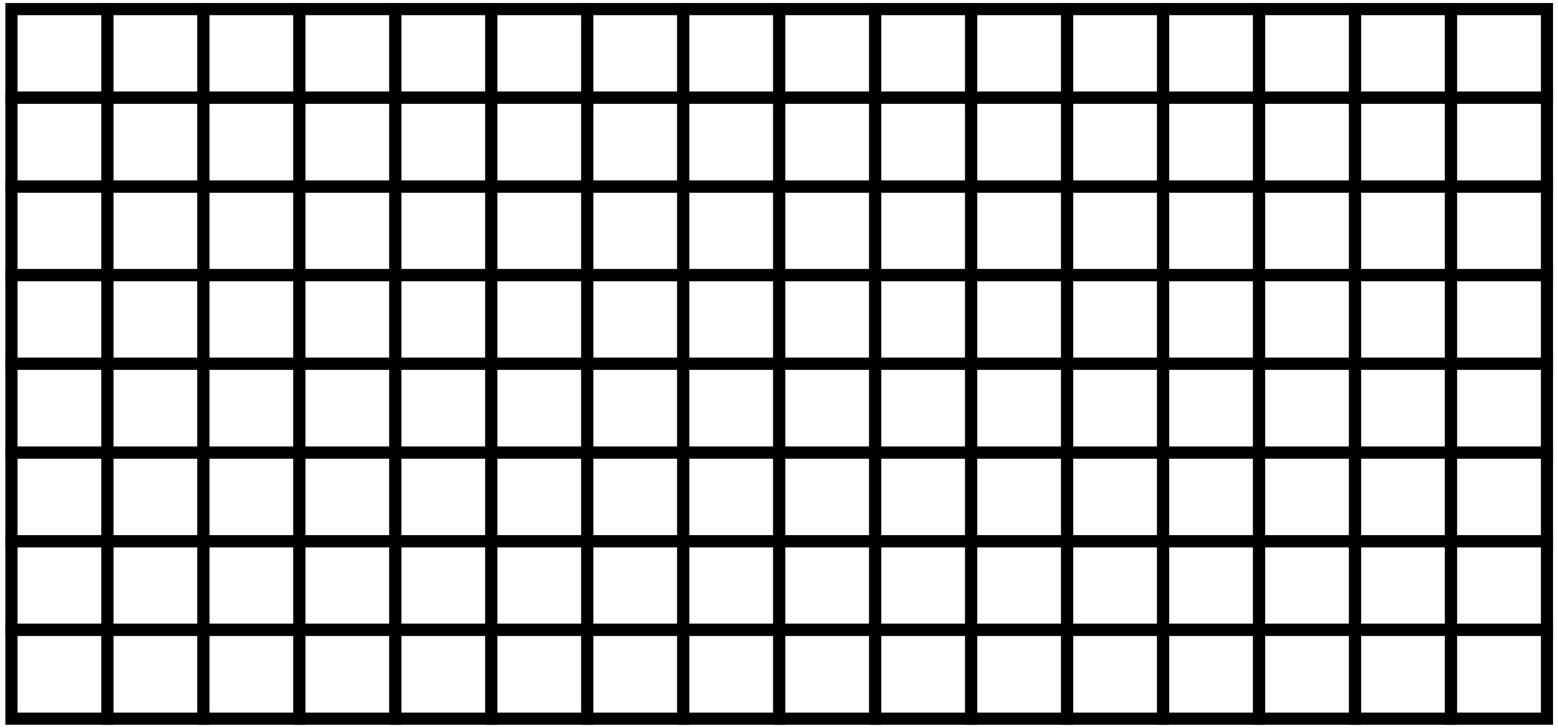
Simple process to generate a binary mask

“Grow” a foreground. Red/black pixels are part of foreground.



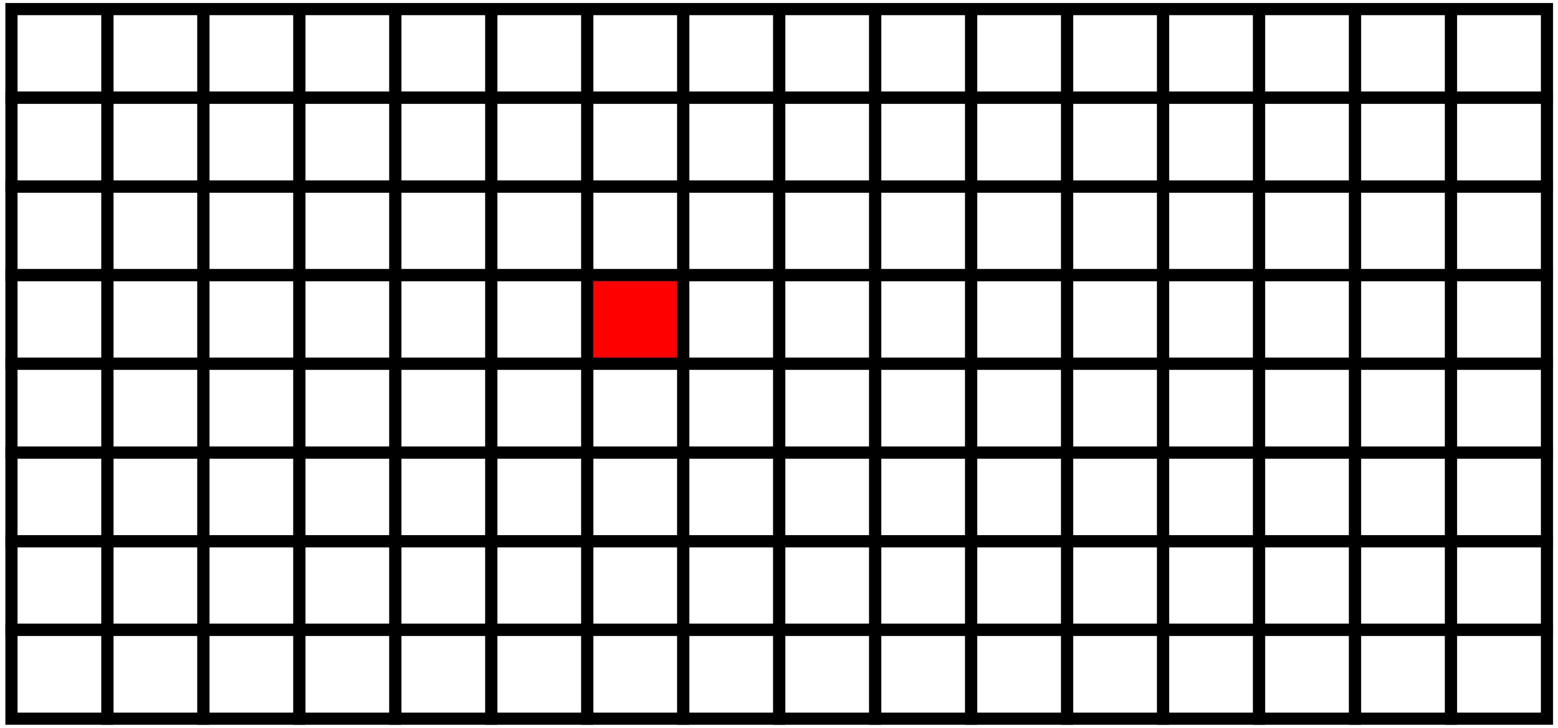
Simple process to generate a binary mask

Choose a pixel to be part of the foreground. Colour it red.



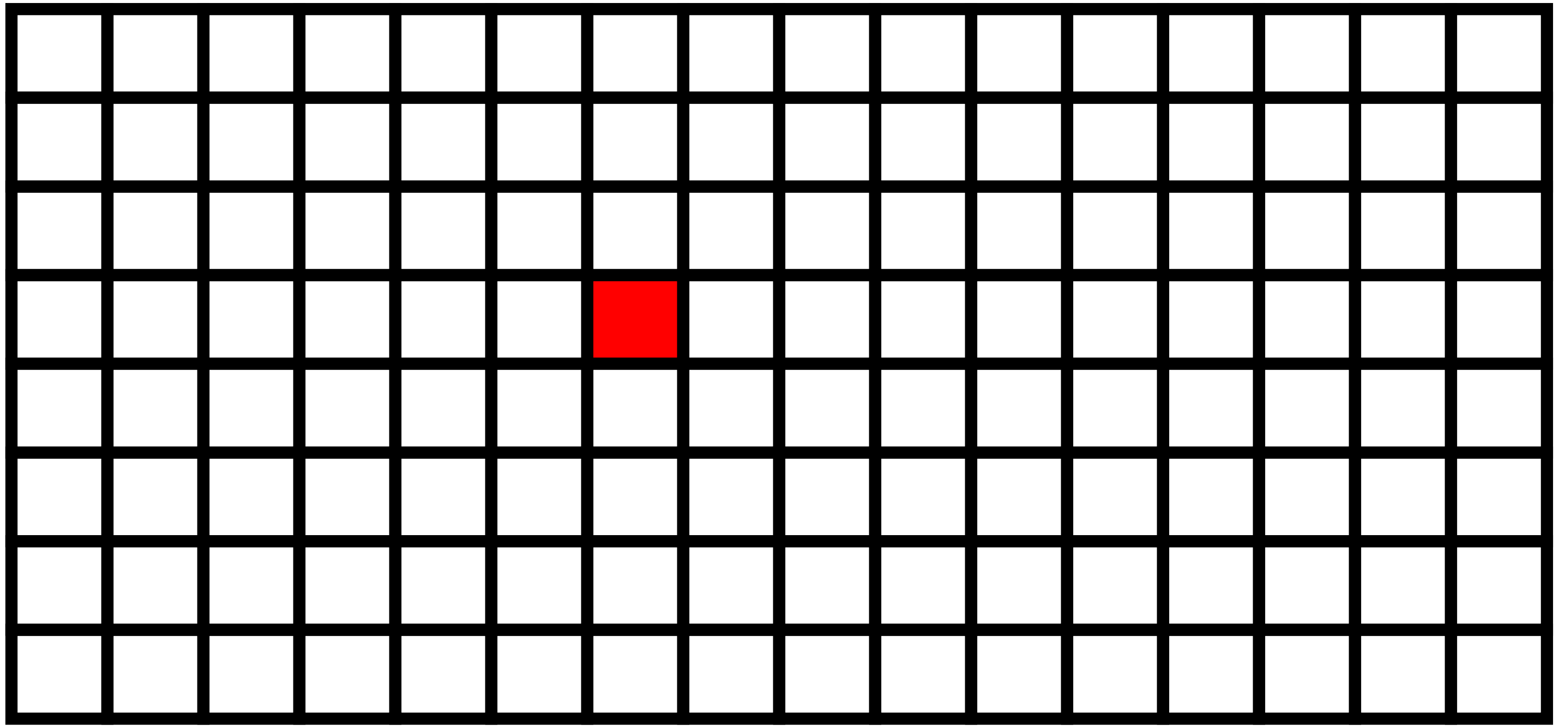
Simple process to generate a binary mask

Choose a pixel to be part of the foreground. Colour it red.



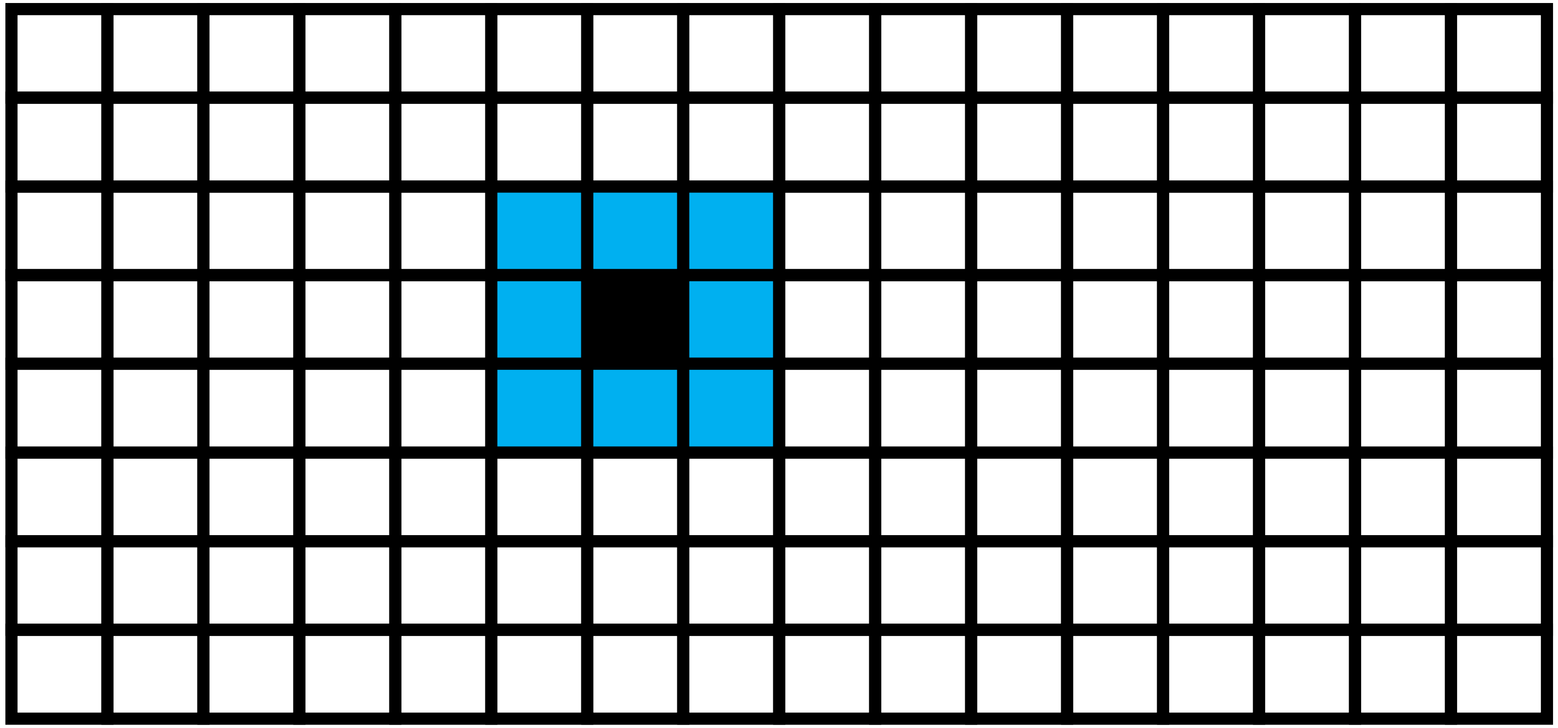
Simple process to generate a binary mask

Pick a red pixel and change its colour to black. Select some neighbours to be part of the foreground, if they are not already, and colour them red.



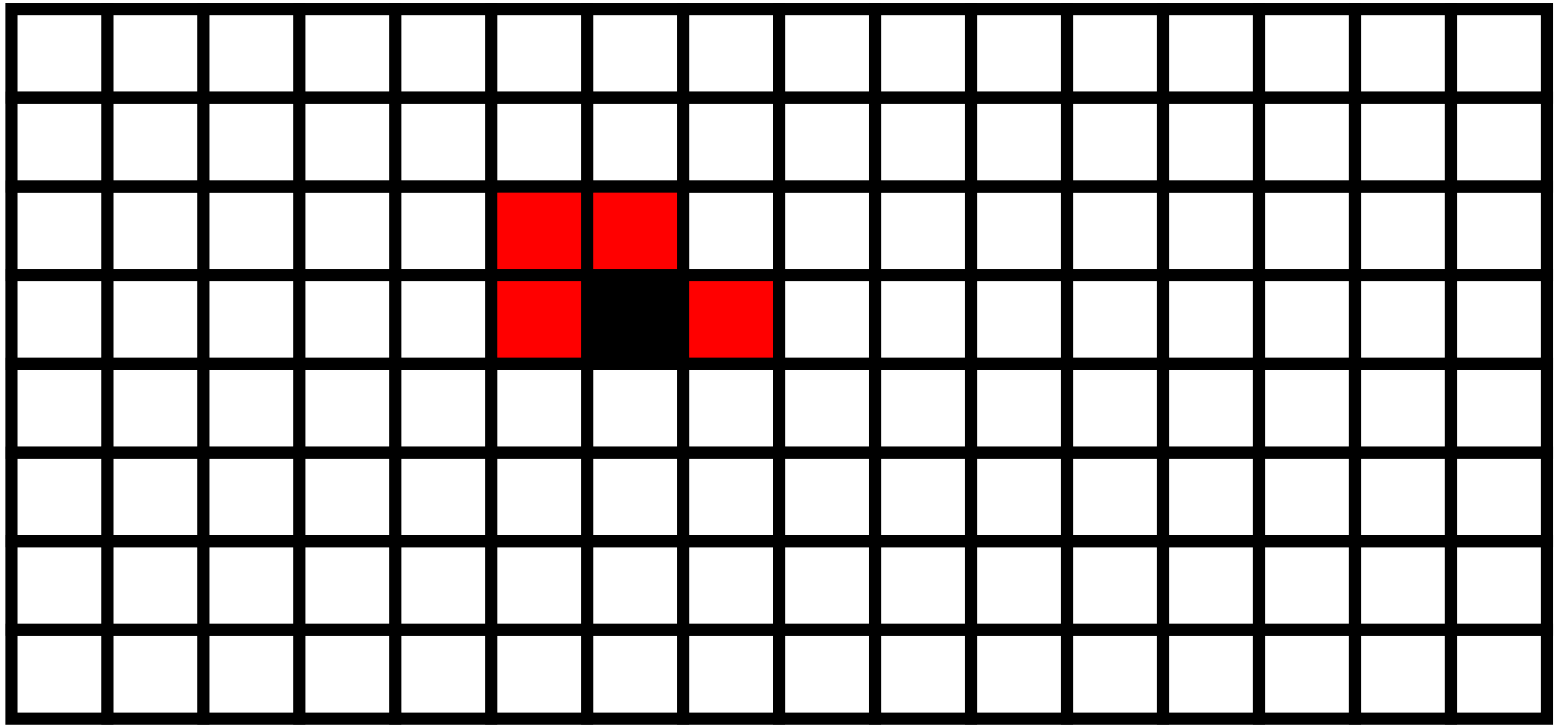
Simple process to generate a binary mask

Pick a red pixel and change its colour to black. Select some neighbours to be part of the foreground, if they are not already, and colour them red.



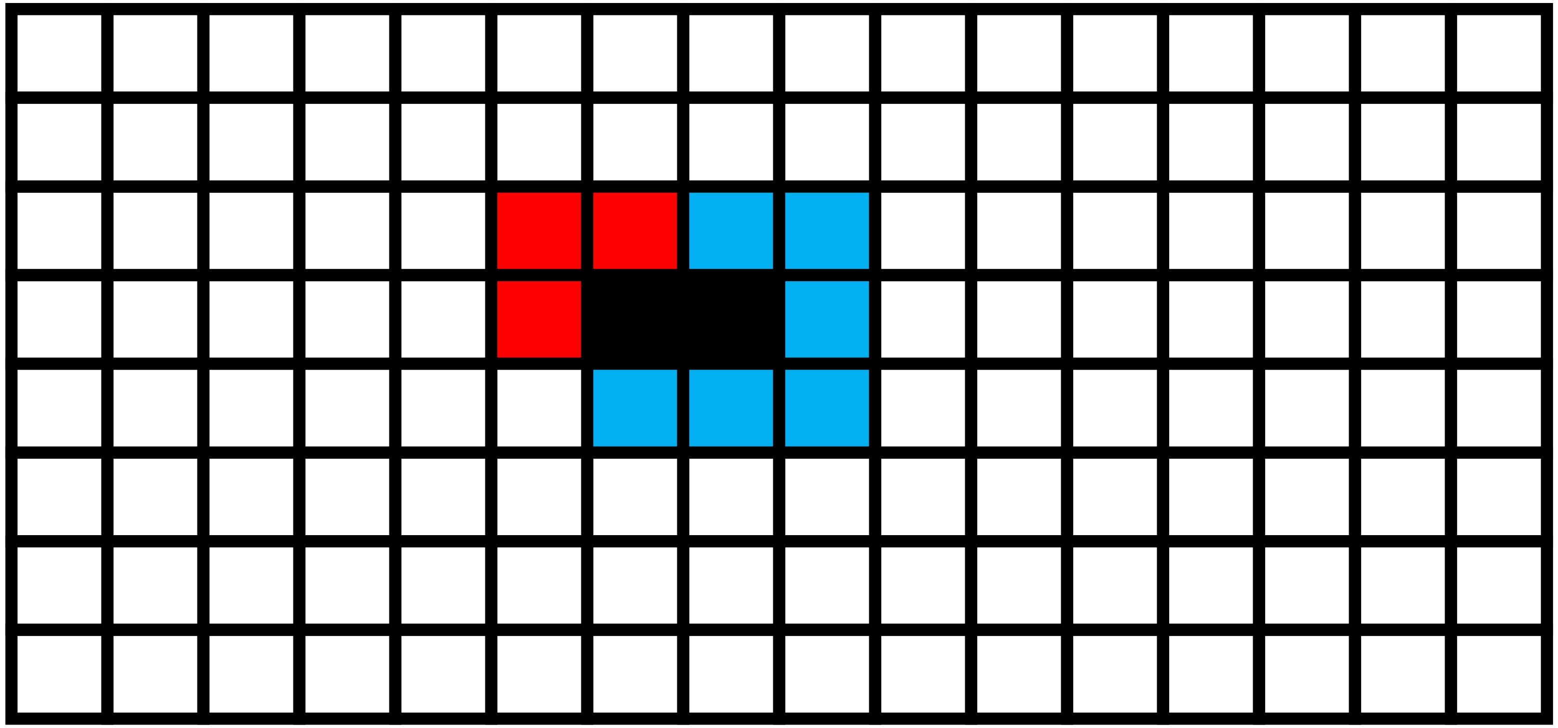
Simple process to generate a binary mask

Pick a red pixel and change its colour to black. Select some neighbours to be part of the foreground, if they are not already, and colour them red.



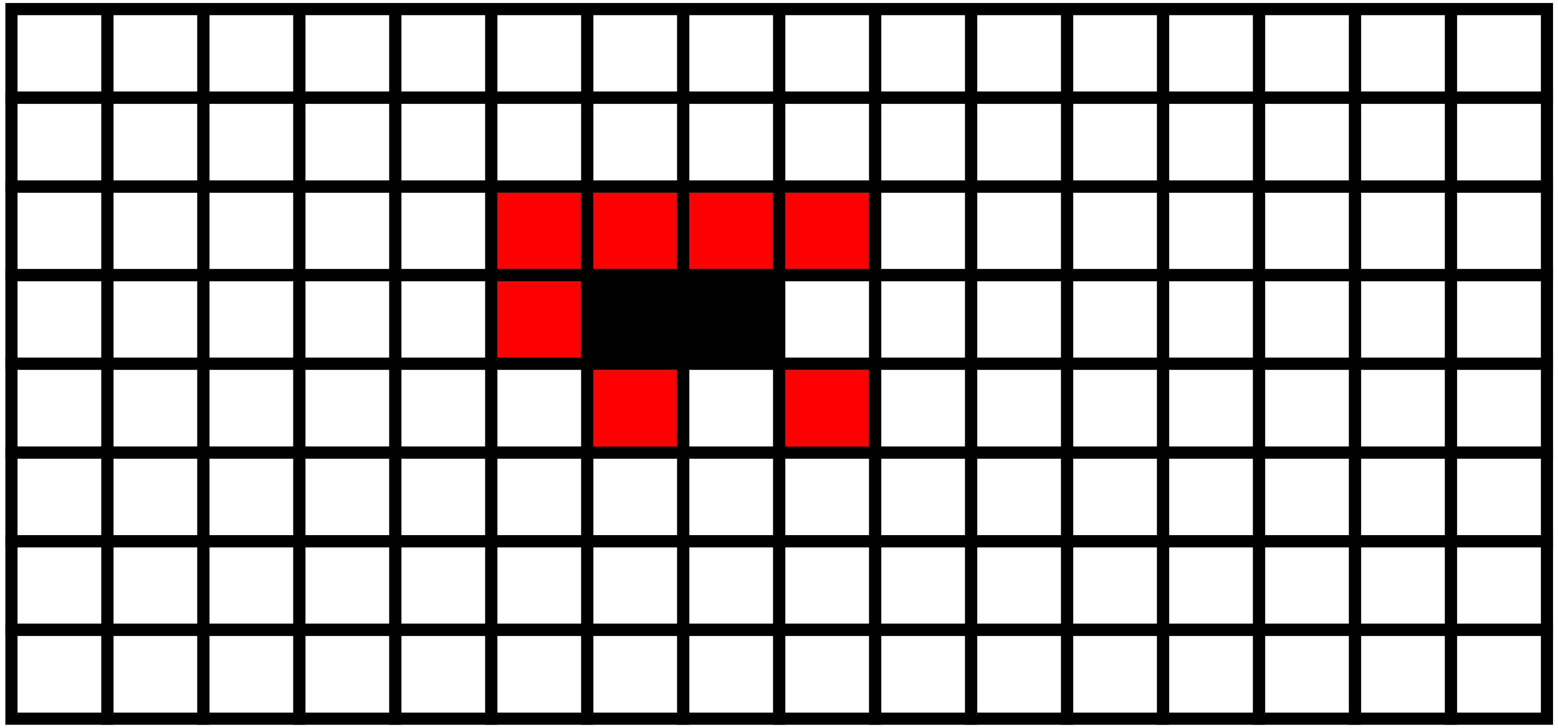
Simple process to generate a binary mask

Pick a red pixel and change its colour to black. Select some neighbours to be part of the foreground, if they are not already, and colour them red.

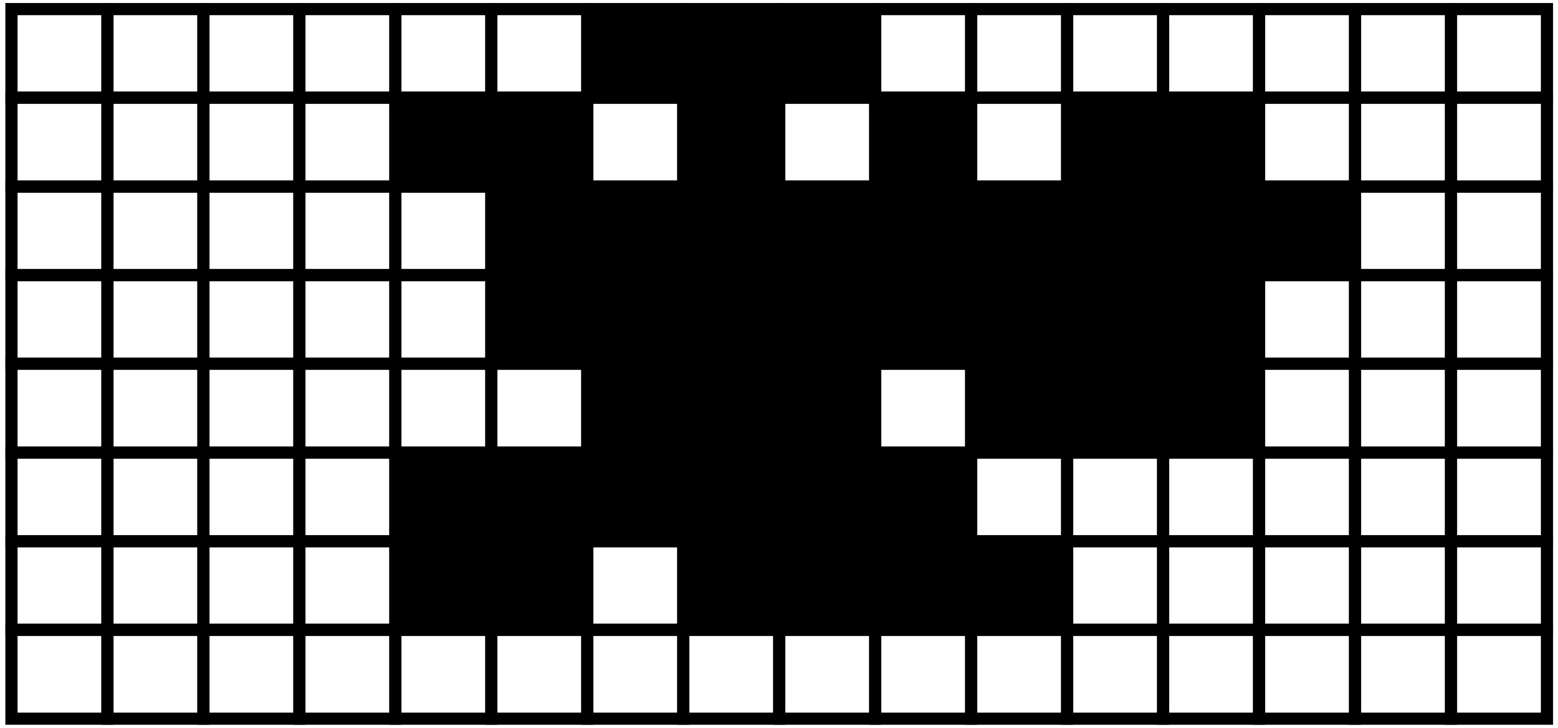


Simple process to generate a binary mask

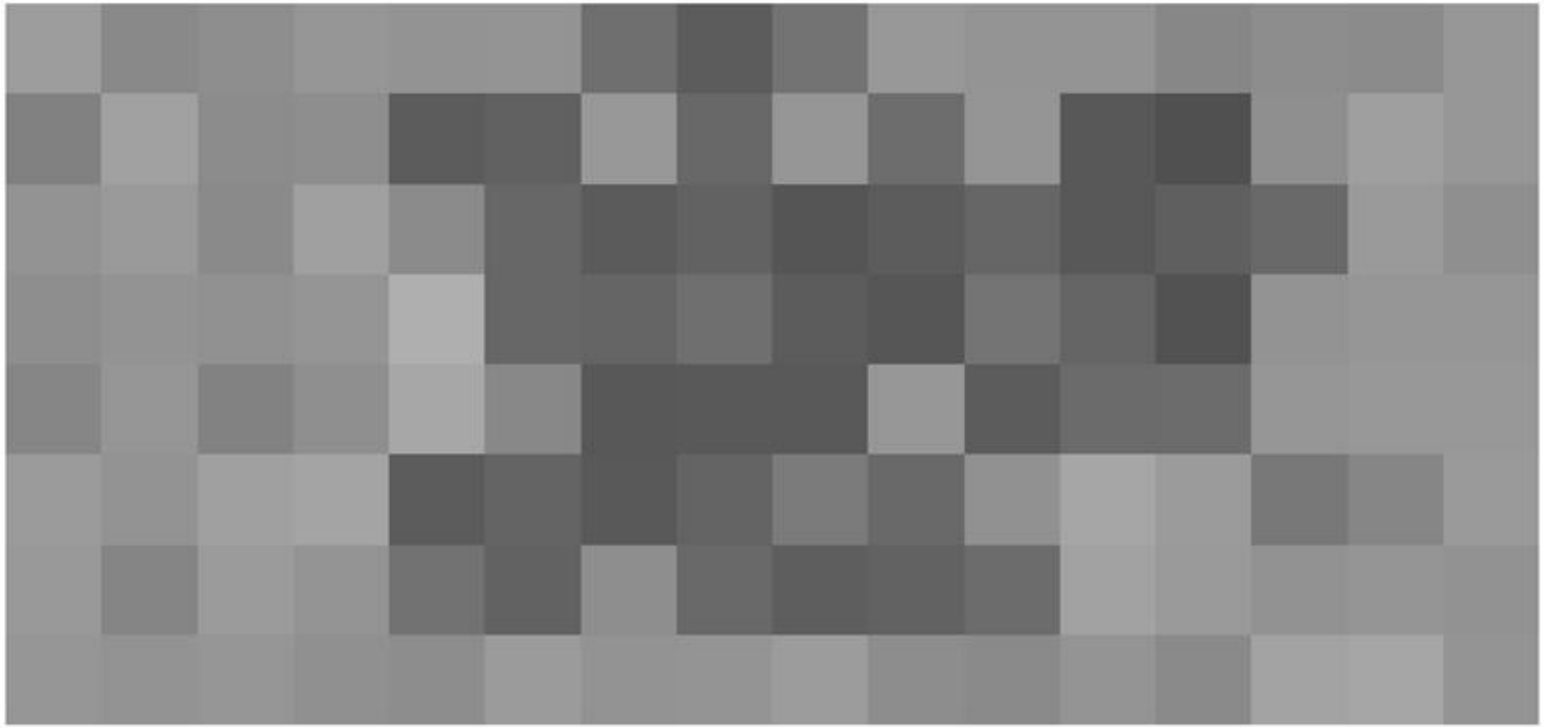
Pick a red pixel and change its colour to black. Select some neighbours to be part of the foreground, if they are not already, and colour them red.



Simple process to generate a binary mask

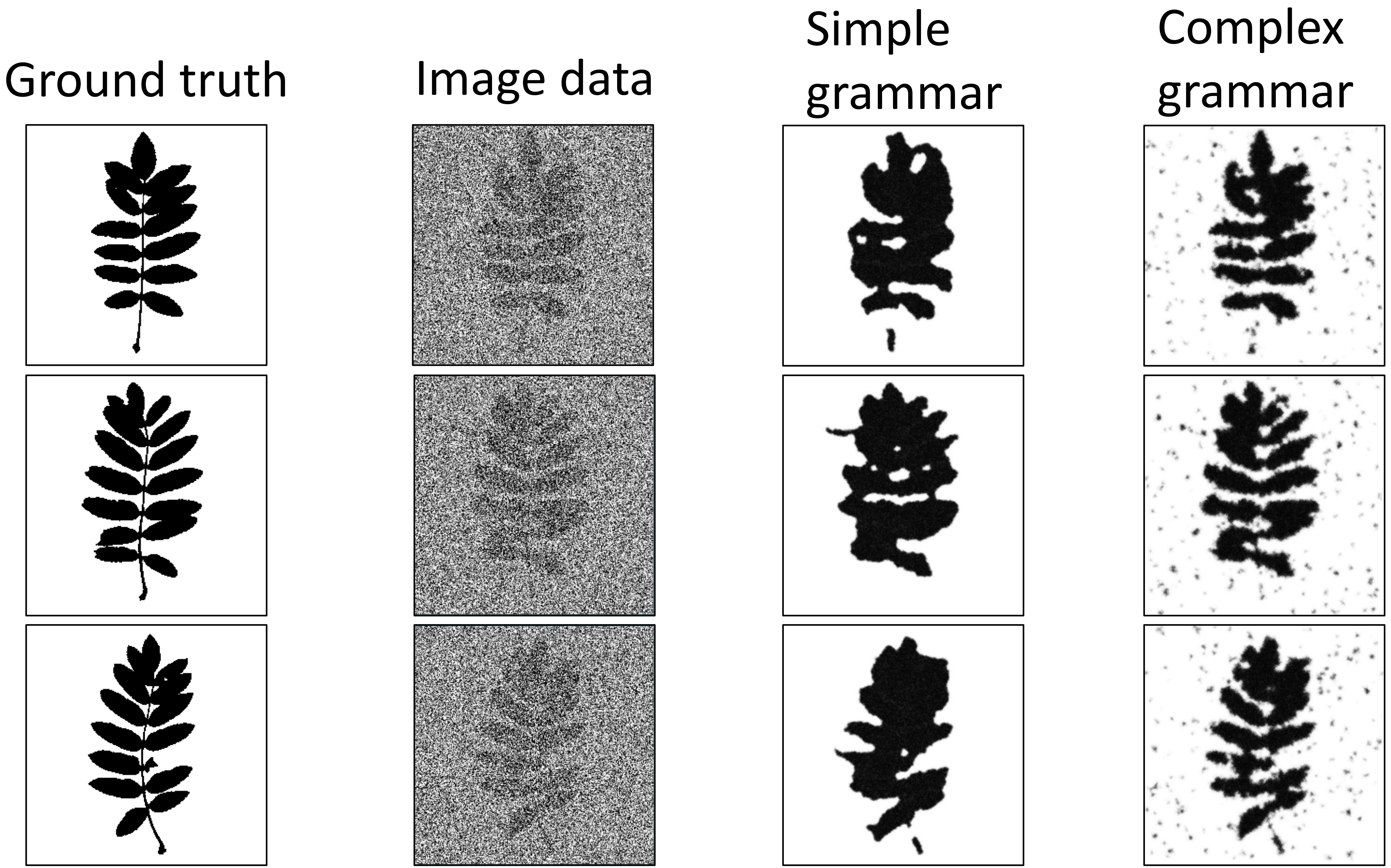


Simple process to generate a binary mask



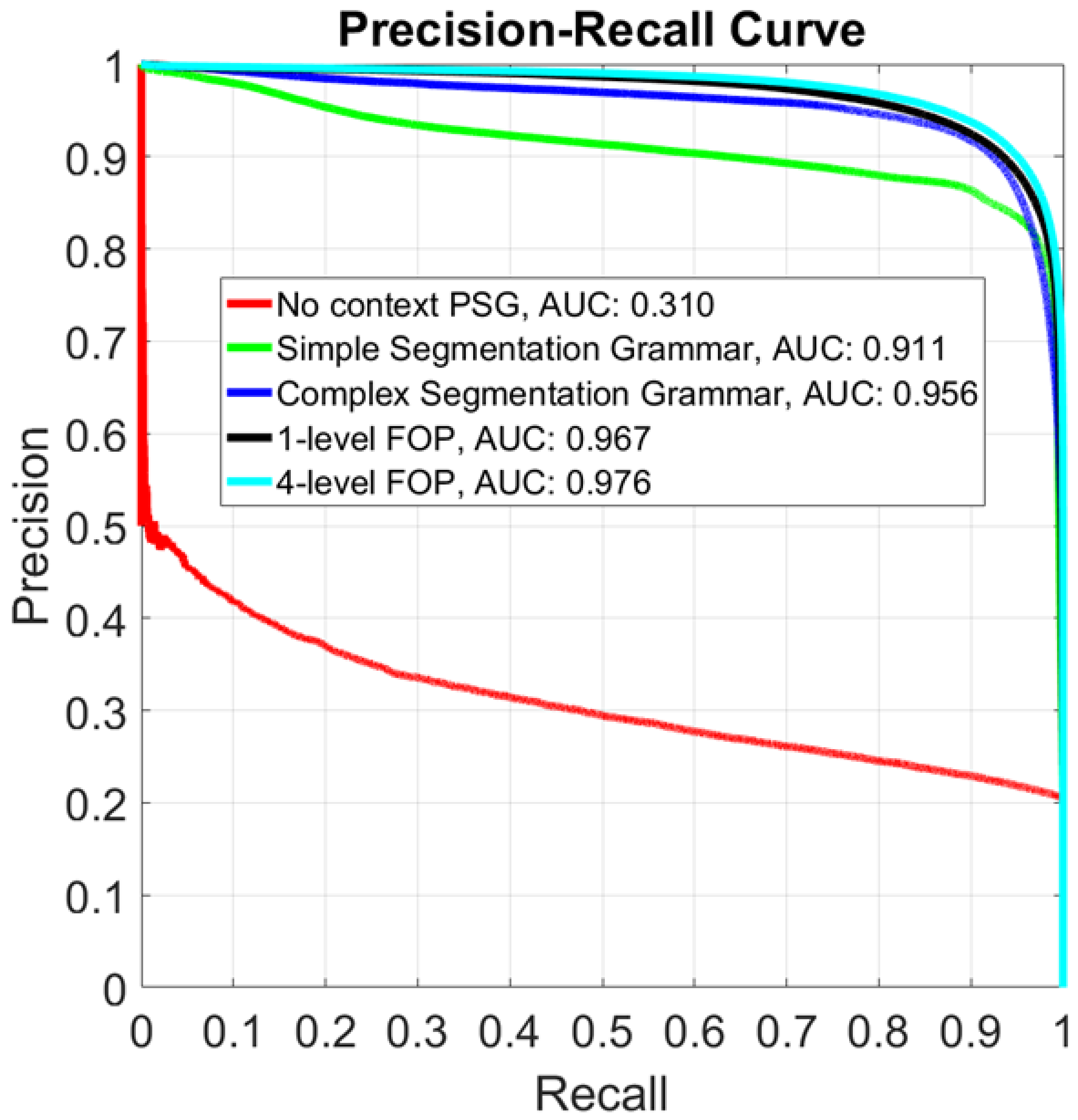
Application: binary image segmentation

- Models:
  - Simple segmentation grammar (previous slides)
  - More complex segmentation grammar
- Training:
  - Model parameters estimated using approximate EM algorithm
  - 50 train, 25 test



Ground truth
Image data
Simple grammar
Complex grammar

Binary image segmentation: Comparison



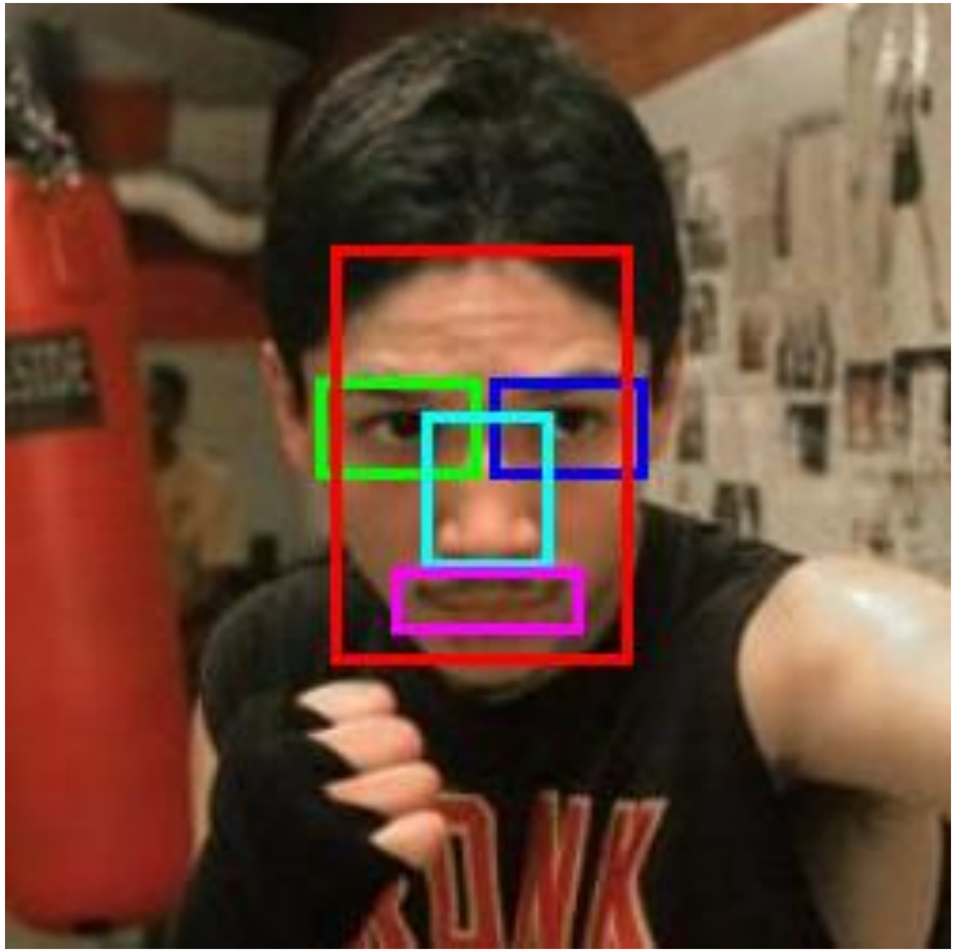
Context is important!

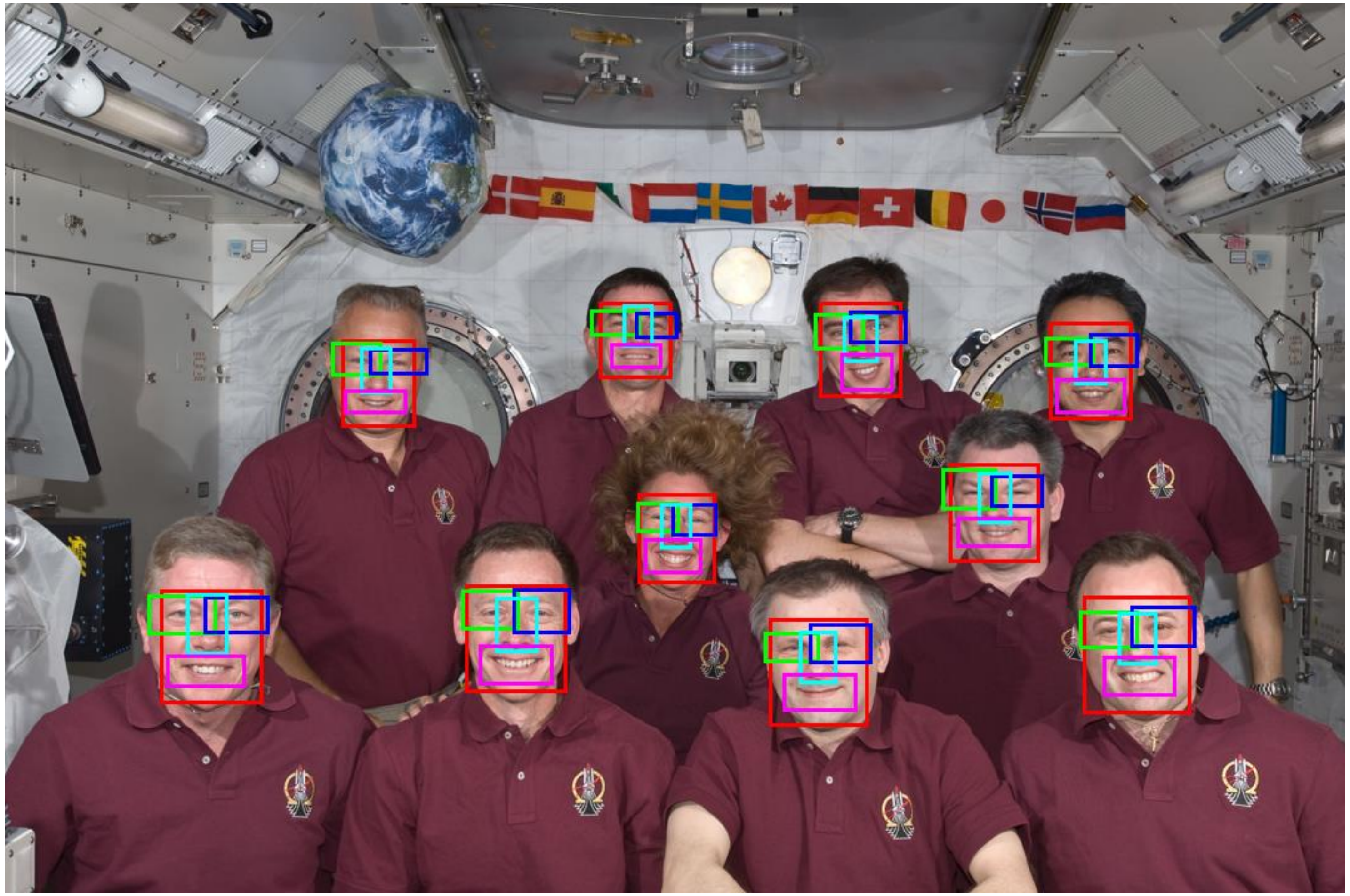
Complex segmentation grammar competitive with Field-of-Patterns (FOP)

1-level FOP and 4-level FOP from: “Multiscale Field of Patterns”, Felzenszwalb, Oberlin. NIPS 2014.

Application: Face localization

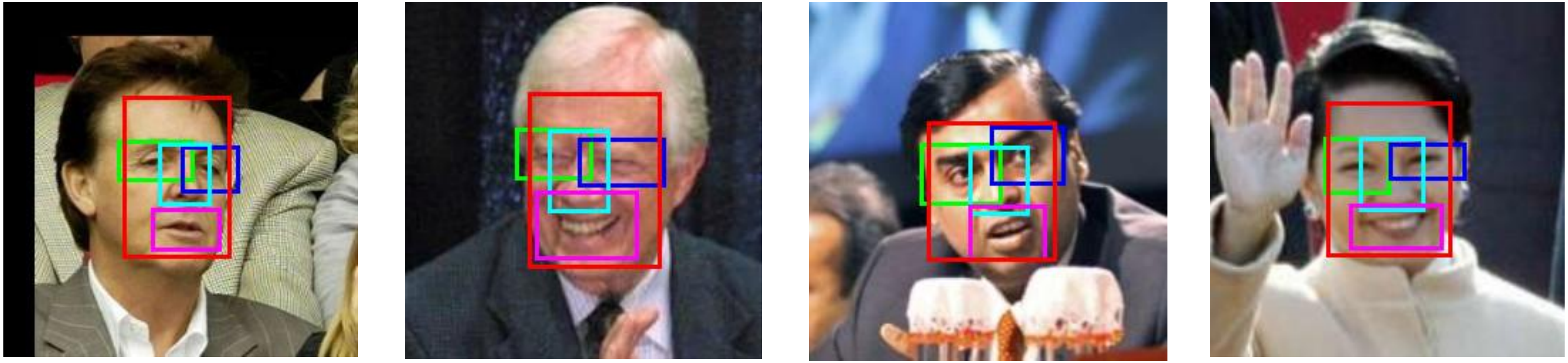
Application: Face localization





Face localization: Single-face Dataset

- Labelled Faces in the Wild [1]
- Manually annotated 300 images with bounding box information for face, left eye, right eye, nose, mouth
- 200 training, 100 test



[1] Huang et al., “Labeled faces in the wild: A database for studying face recognition in unconstrained environments.”, Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

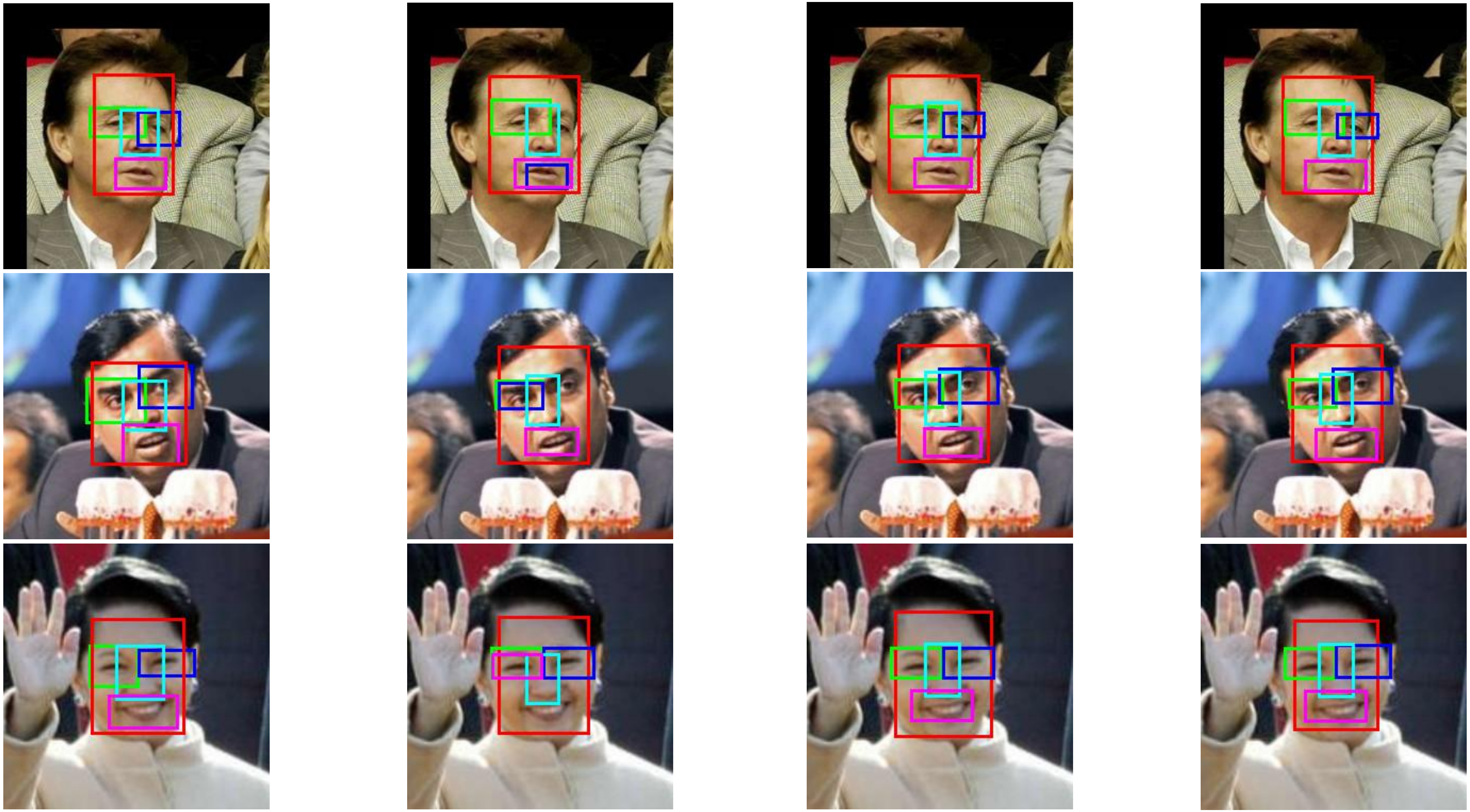
PSG Face Grammar

- Symbols: Face (F), Left eye (L), Right eye (R), Nose (N), Mouth (M)
- Pose space: (x,y,scale)
- Mechanism to handle help suppress false positives
- Geometric model learned from labelled data
- Data-model: calibrated HOG filter scores.
- Factor graph has ~3M edges

Face localization: Baselines

- HOG Filter scores: calibrated HOG filter scores only
- “Pictorial Structures for object recognition”, IJCV 2005
  - Fast, exact inference
  - Assumes 1 object of each type per scene

Ground truth	HOG Filters	Pictorial Structures	PSG Face Grammar



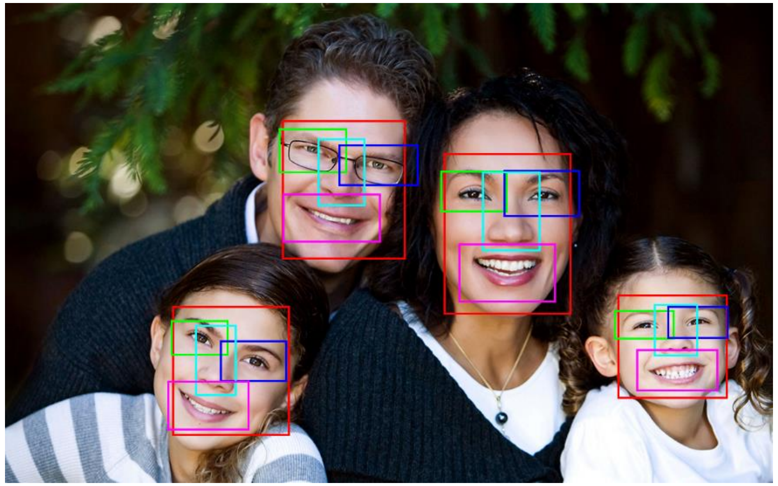
Face Localization: Performance comparison

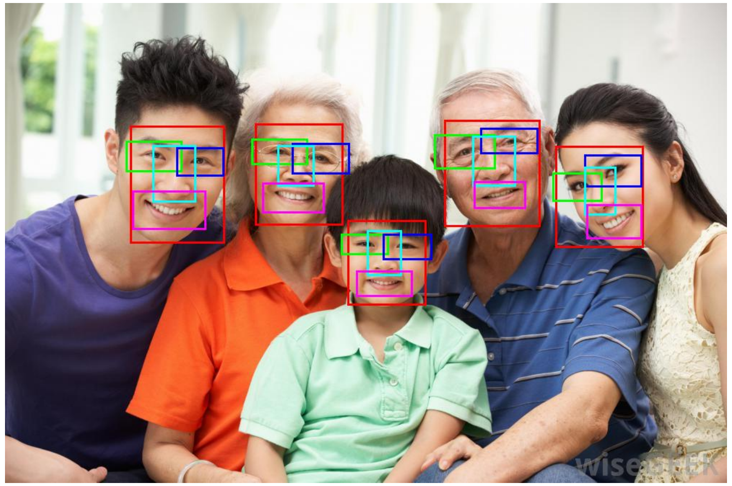
Model	FACE	LEFT-EYE	RIGHT-EYE	NOSE	MOUTH	**Average**
HOG Filters	1.00	0.76	0.65	0.96	0.60	0.80
Pictorial Structures	1.00	0.97	0.93	0.98	0.90	0.96
PSG Face Grammar	1.00	0.98	0.92	0.98	0.92	0.96

Area under the precision-recall curve

Face localization: Family Portraits Dataset

- Dataset collected from the Internet
- Manually annotated 40 images with bounding box information
- Average of 5.9 faces per image
- Similar as trained model from LFW, but with more scales
- Use all 40 images for testing





Ground truth

Pictorial Structures

PSG Face Grammar

Face Localization: Performance Comparison

Model	FACE	LEFT-EYE	RIGHT-EYE	NOSE	MOUTH	**Average**
HOG Filters	0.95	0.50	0.48	0.90	0.32	0.63
Pictorial Structures	0.97	0.78	0.69	0.96	0.73	0.82
PSG Face Grammar	0.97	0.81	0.81	0.96	0.80	0.87

Area under the precision-recall curve

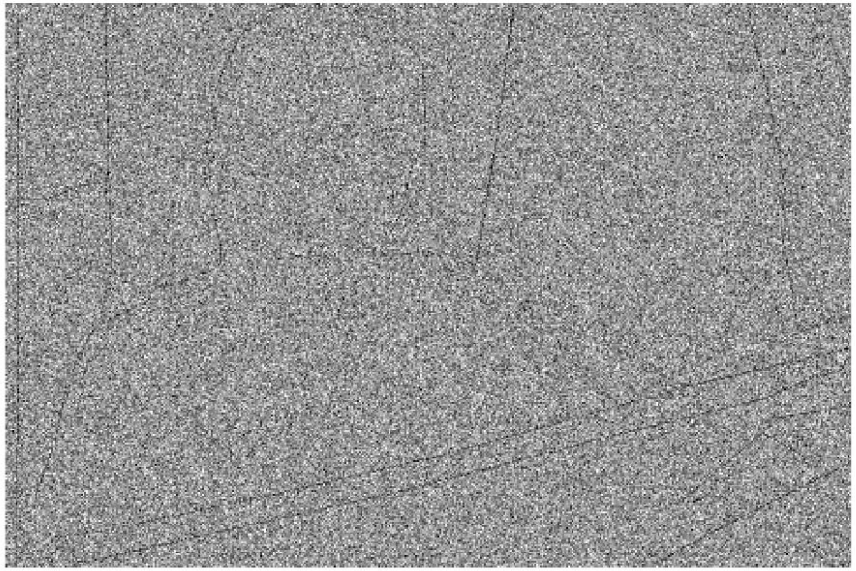
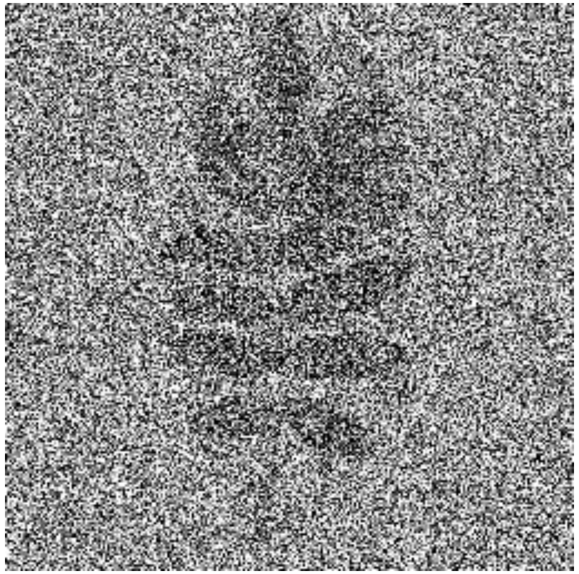
This talk

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

This talk

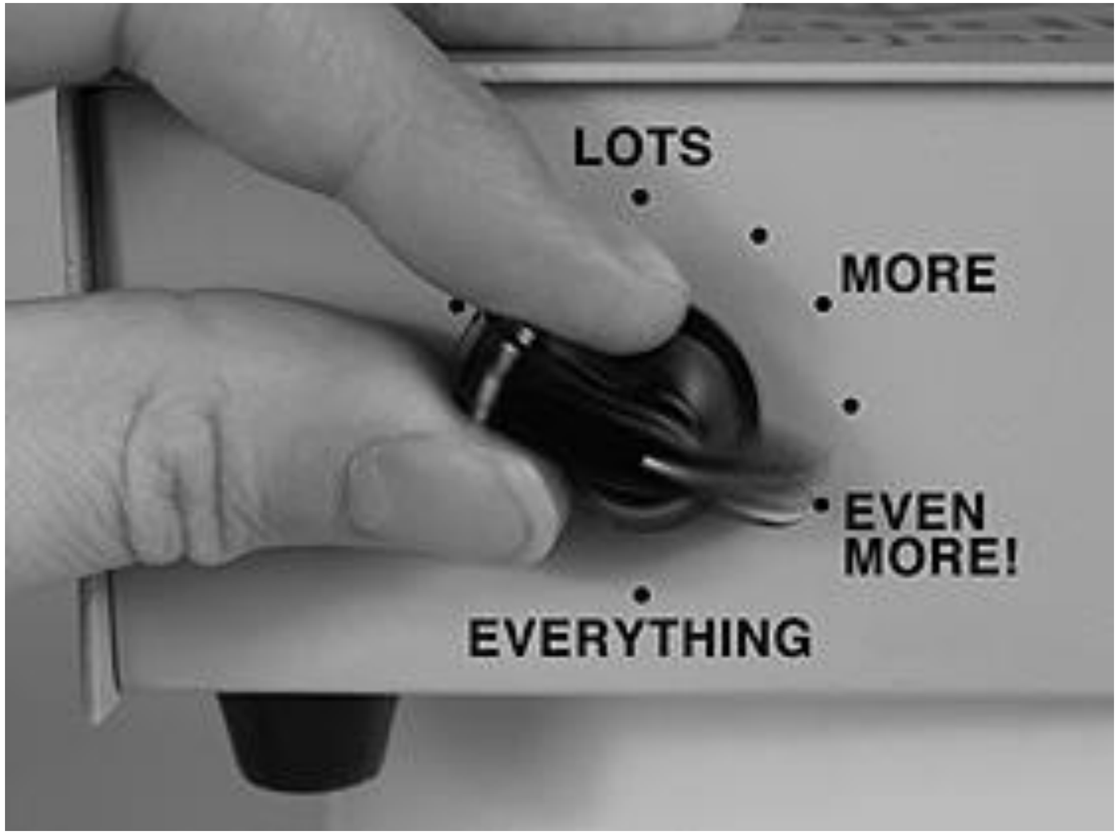
- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research



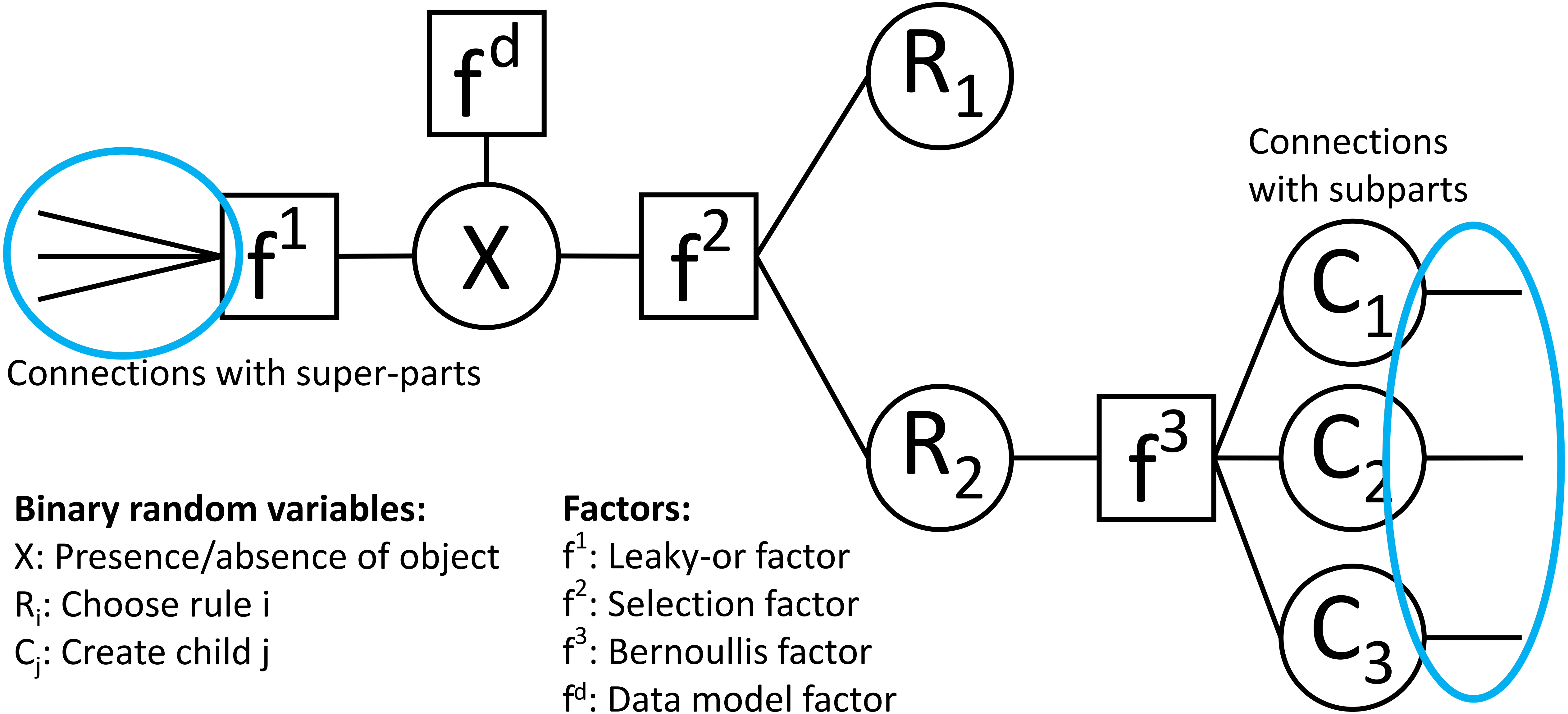




Scaling up

- Bigger images! More objects! Larger grammars!



Not so fast …



Binary random variables:

X: Presence/absence of object

R_i: Choose rule i

C_j: Create child j

Factors:

f^1: Leaky-or factor

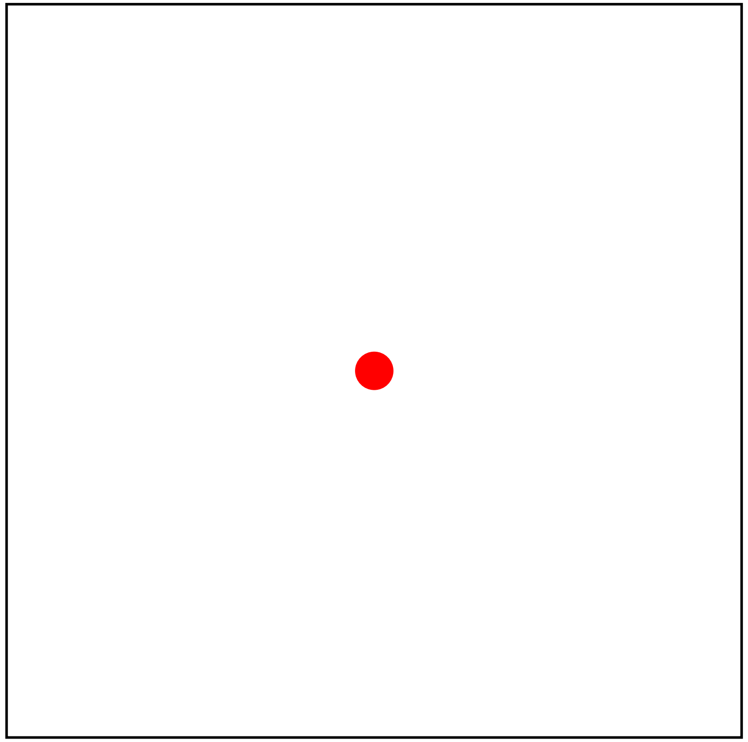
f^2: Selection factor

f^3: Bernoullis factor

f^d: Data model factor

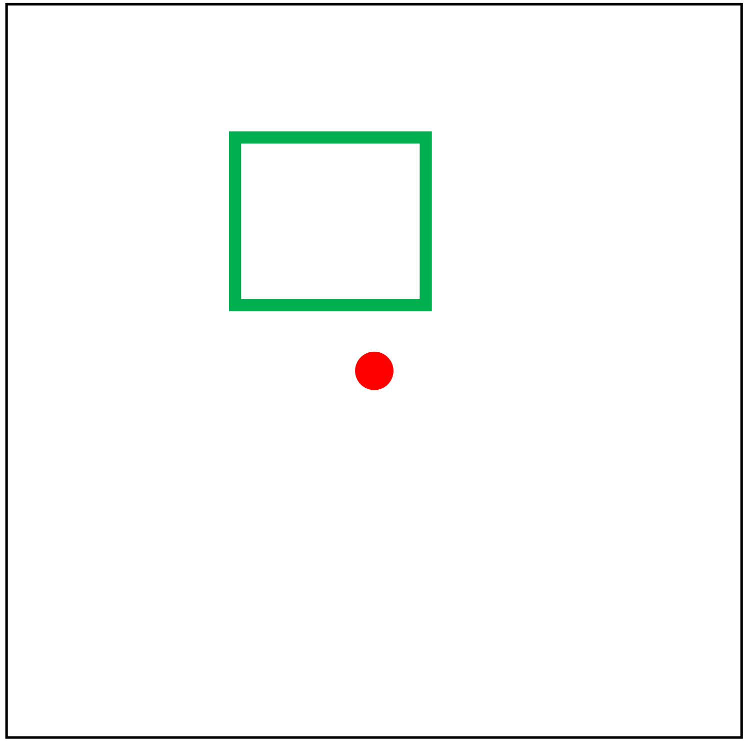
Graph has too many edges!

- Most edges are inter-object
- Example: Eye of a **particular** face



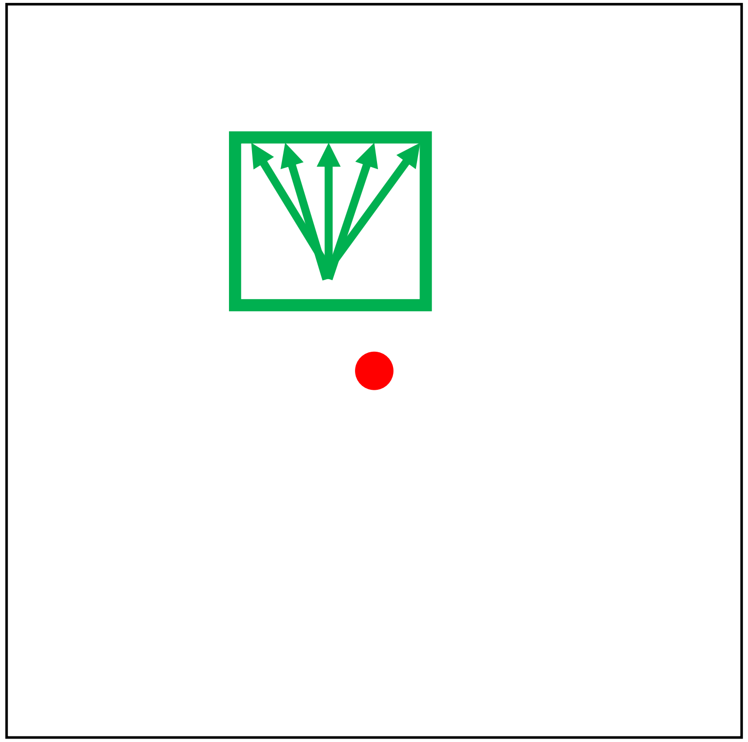
Graph has too many edges!

- Most edges are inter-object
- Example: Eye of a **particular** face
- $\underbrace{(15 \text{ x } 15)}_{\text{Size of image region}}$



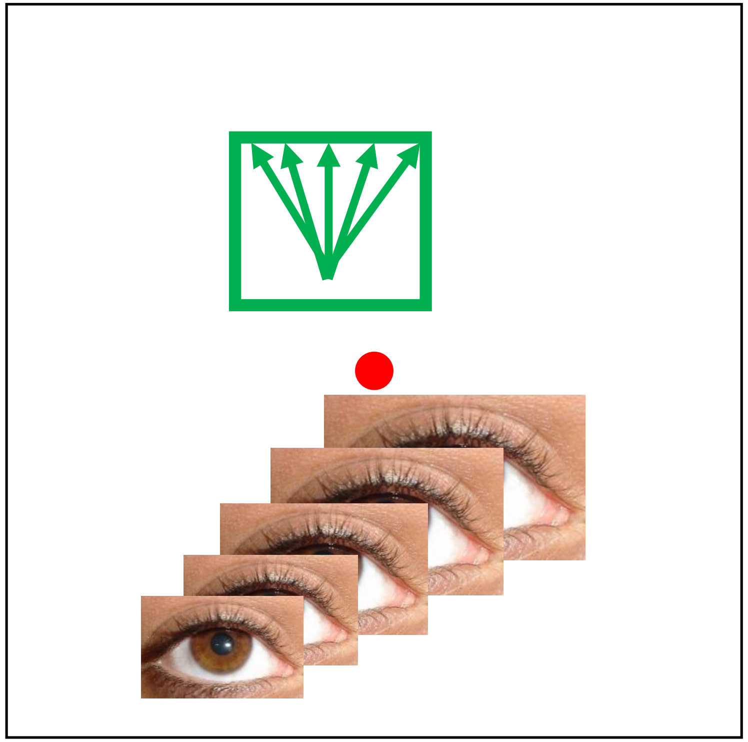
Graph has too many edges!

- Most edges are inter-object
- Example: Eye of a **particular** face
- $\underbrace{(15 \times 15)}_{\text{Size of image region}} \times \underbrace{5}_{\text{\# orientations}}$



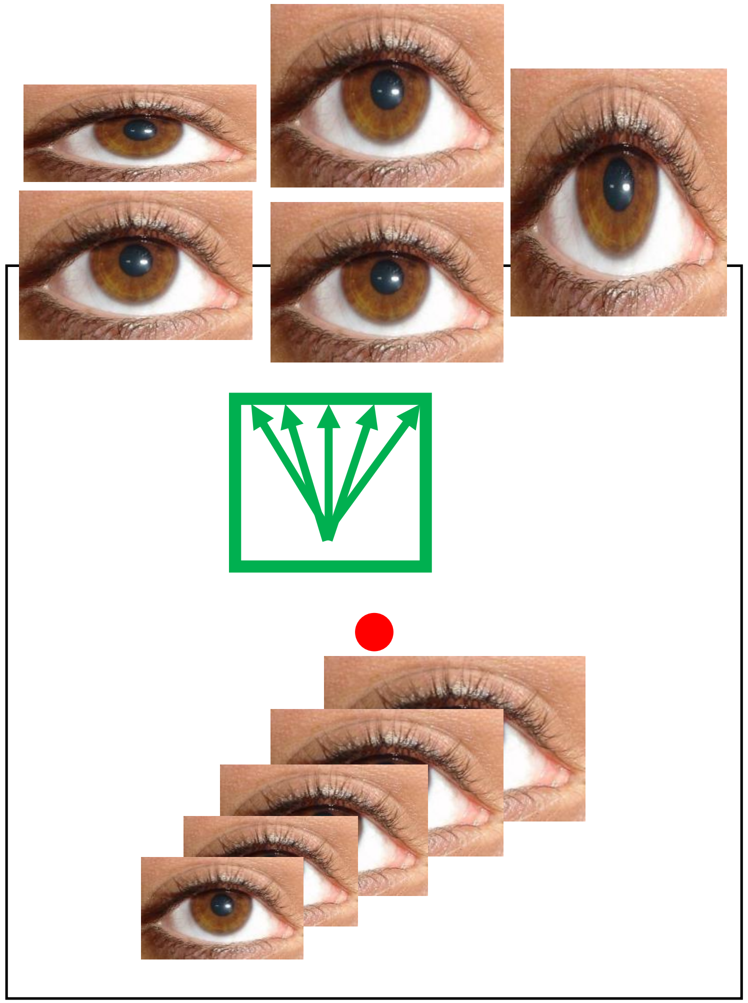
Graph has too many edges!

- Most edges are inter-object
- Example: Eye of a **particular** face
- $\underbrace{(15 \text{ x } 15)}_{\text{Size of image region}} \text{ x } \underbrace{5}_{\text{\# orientations}} \text{ x } \underbrace{5}_{\text{\# scales}}$



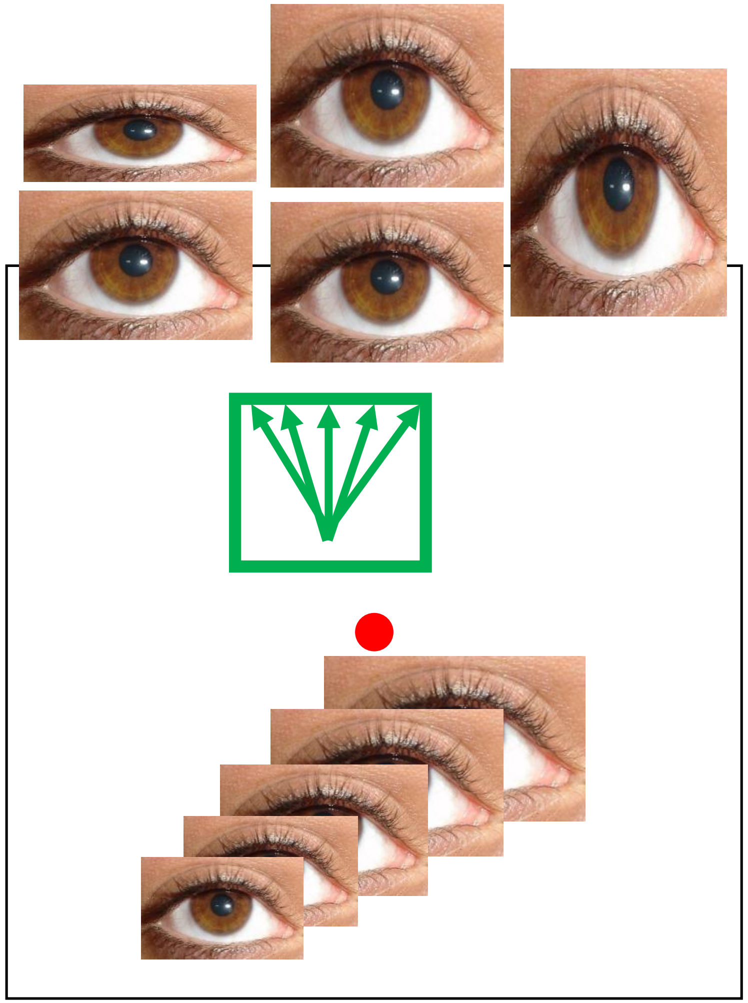
Graph has too many edges!

- Most edges are inter-object
- Example: Eye of a **particular** face
- $\underbrace{(15 \times 15)}_{\text{Size of image region}} \times \underbrace{5}_{\text{\# orientations}} \times \underbrace{5}_{\text{\# scales}} \times \underbrace{5}_{\text{\# aspect ratios}}$



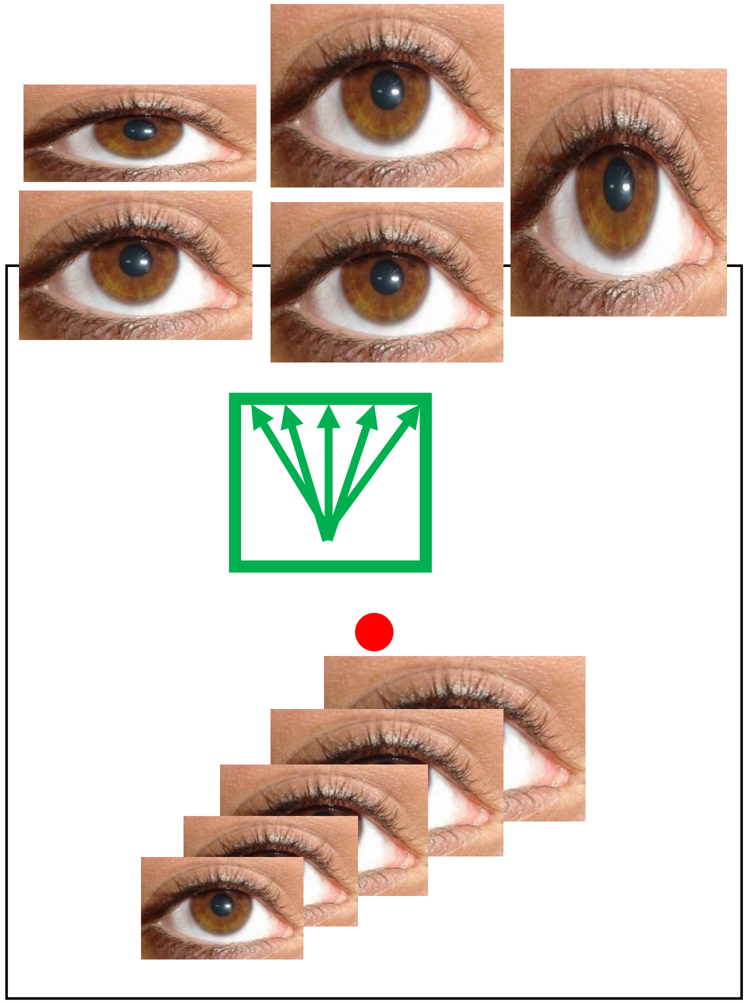
Graph has too many edges!

- Most edges are inter-object
- Example: Eye of a **particular** face
- $\underbrace{(15 \times 15)}_{\text{Size of image region}} \times \underbrace{5}_{\text{\# orientations}} \times \underbrace{5}_{\text{\# scales}} \times \underbrace{5}_{\text{\# aspect ratios}} = 28125$ edges



Graph has too many edges!

- Most edges are inter-object
- Example: Eye of a **particular** face
- $\underbrace{(15 \times 15)}_{\text{Size of image region}} \times \underbrace{5}_{\text{\# orientations}} \times \underbrace{5}_{\text{\# scales}} \times \underbrace{5}_{\text{\# aspect ratios}} = 28125$ edges



This is for a **single** face, and a **single** eye!

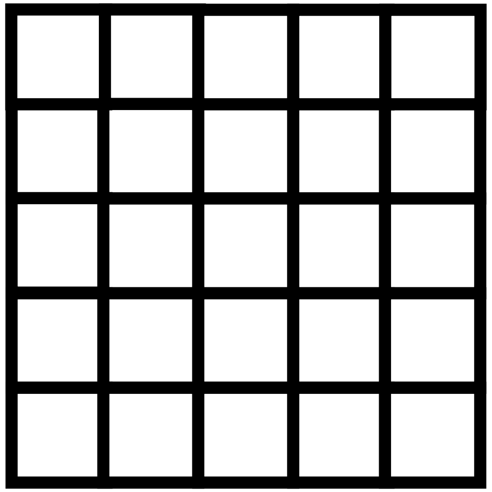
Key issue: Size of distribution's support

Reducing the number of factor graph edges

- Reduce total support of probability distributions →Reduce # of factor graph edges
- Approximate N-D distribution as N one-dimensional distributions
- Decompose 1-D Uniform distribution into a set of Categorical distributions

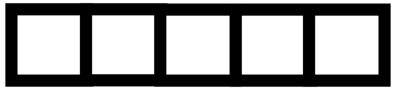
Approximating an N-D distribution

Where to put eye? 1 of 25 choice

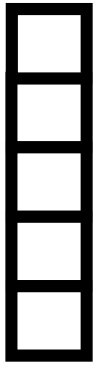


Cost: 25 edges

x-coordinate of eye? 1 of 5 choice



y-coordinate of eye? 1 of 5 choice

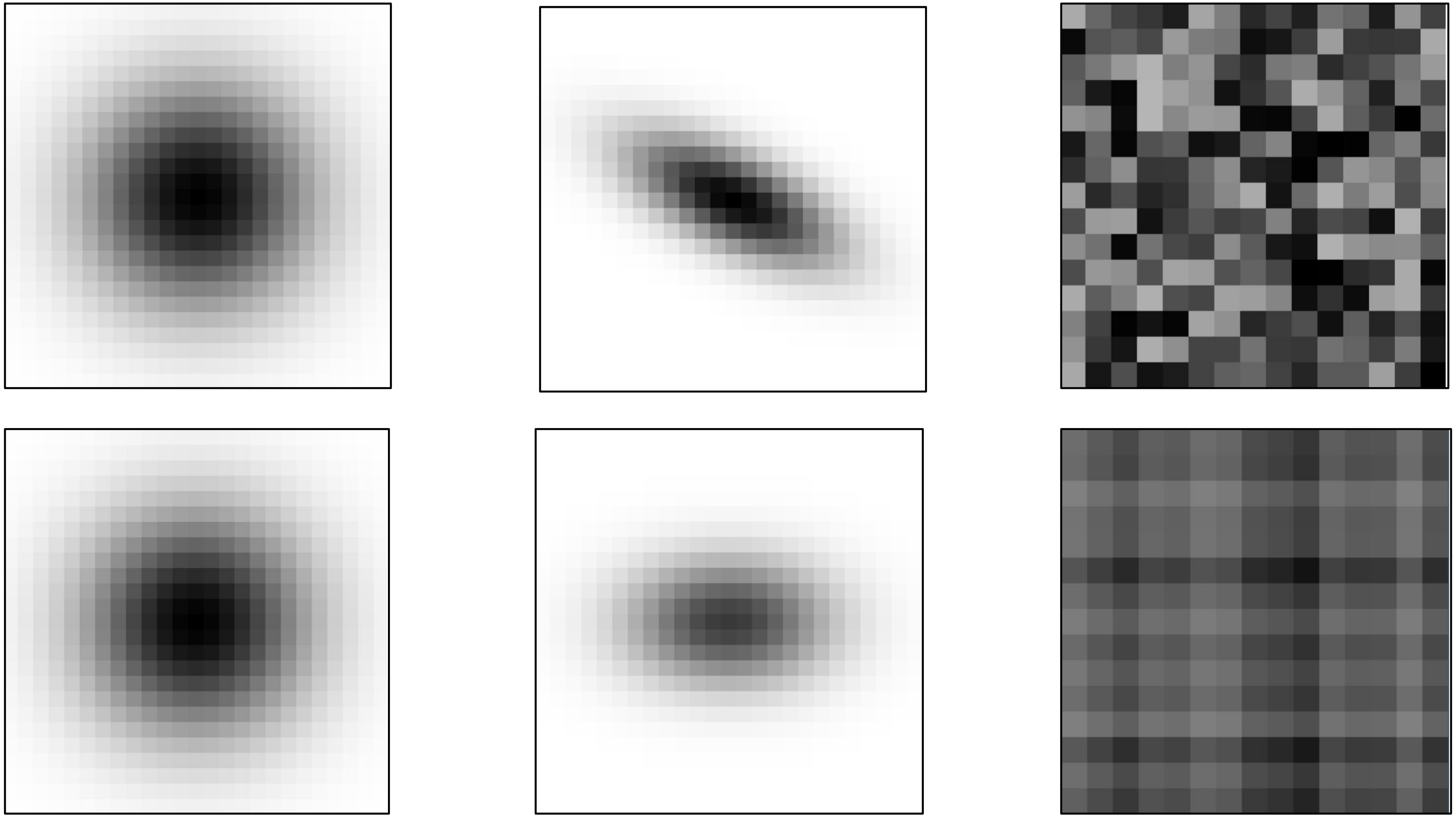


Cost: 10 edges

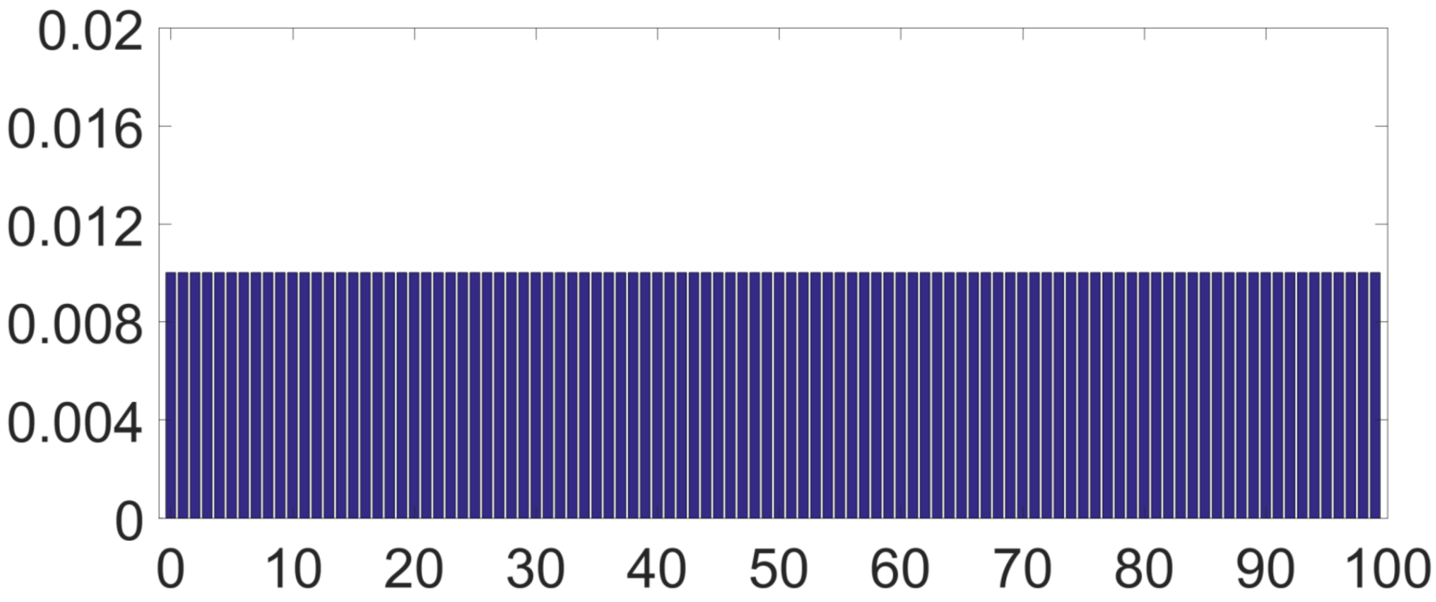
Approximating an N-D distribution

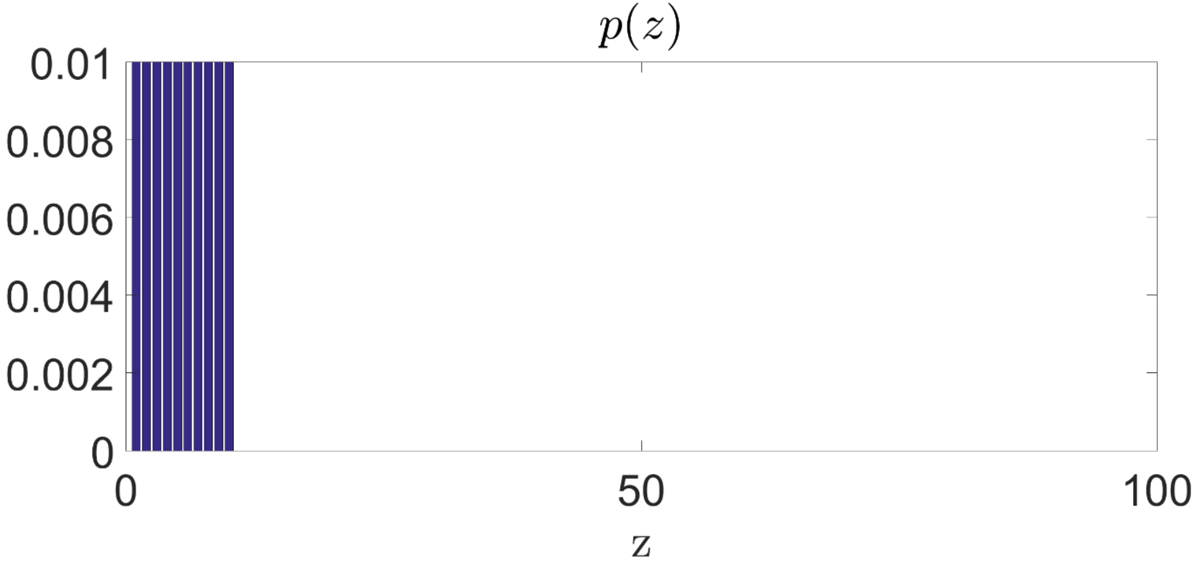
Original distribution

Approximated distribution

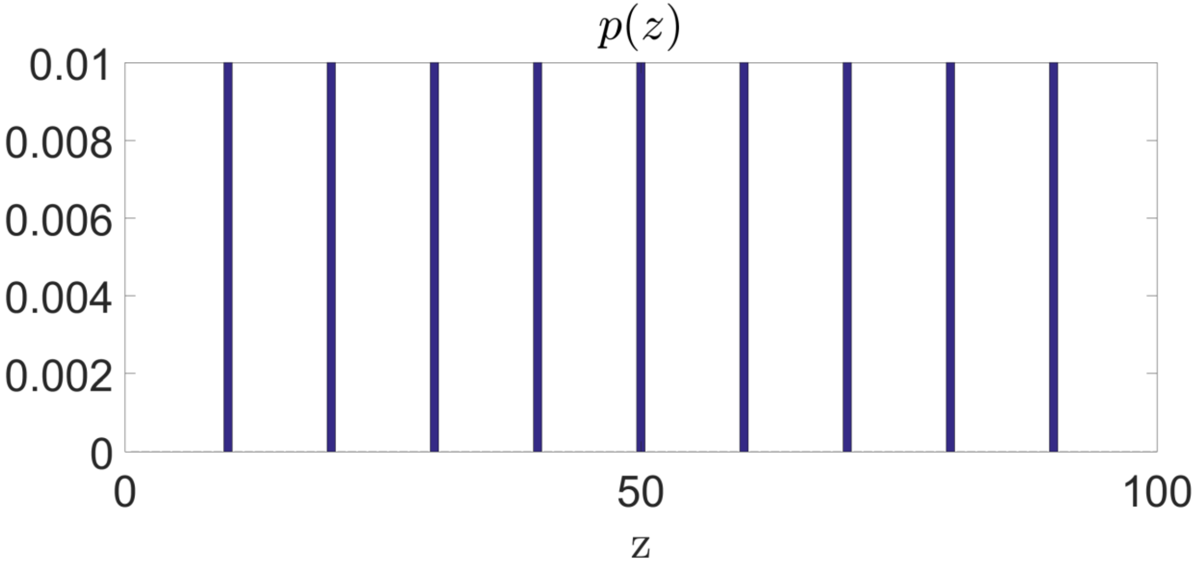


Decomposing a Uniform distribution

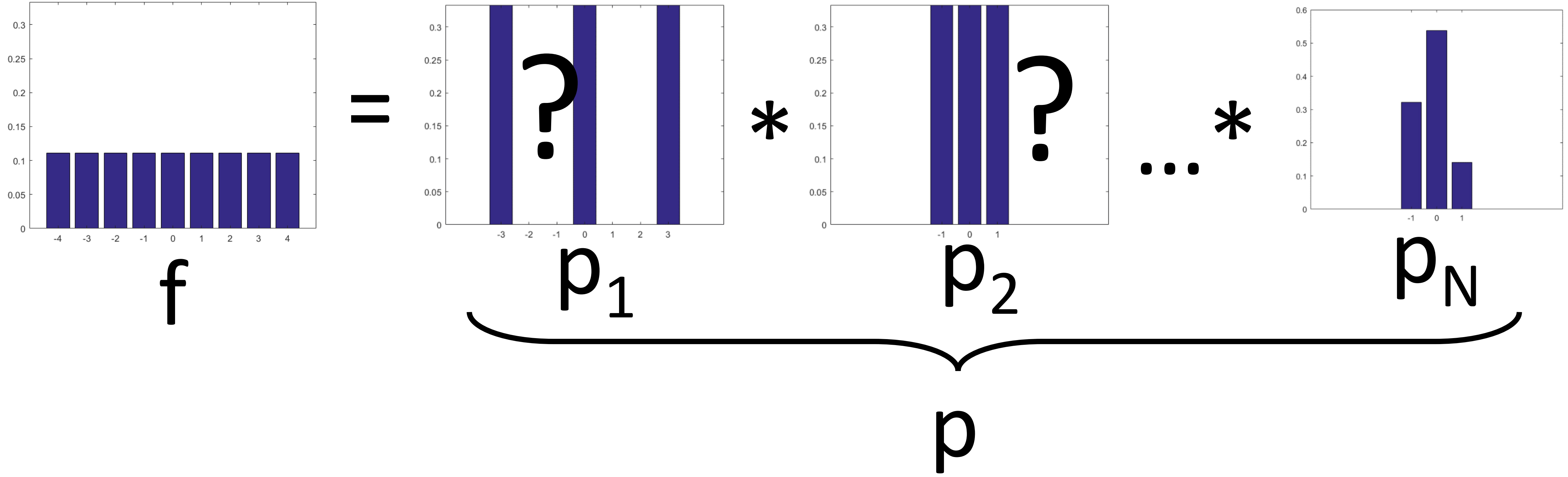




*

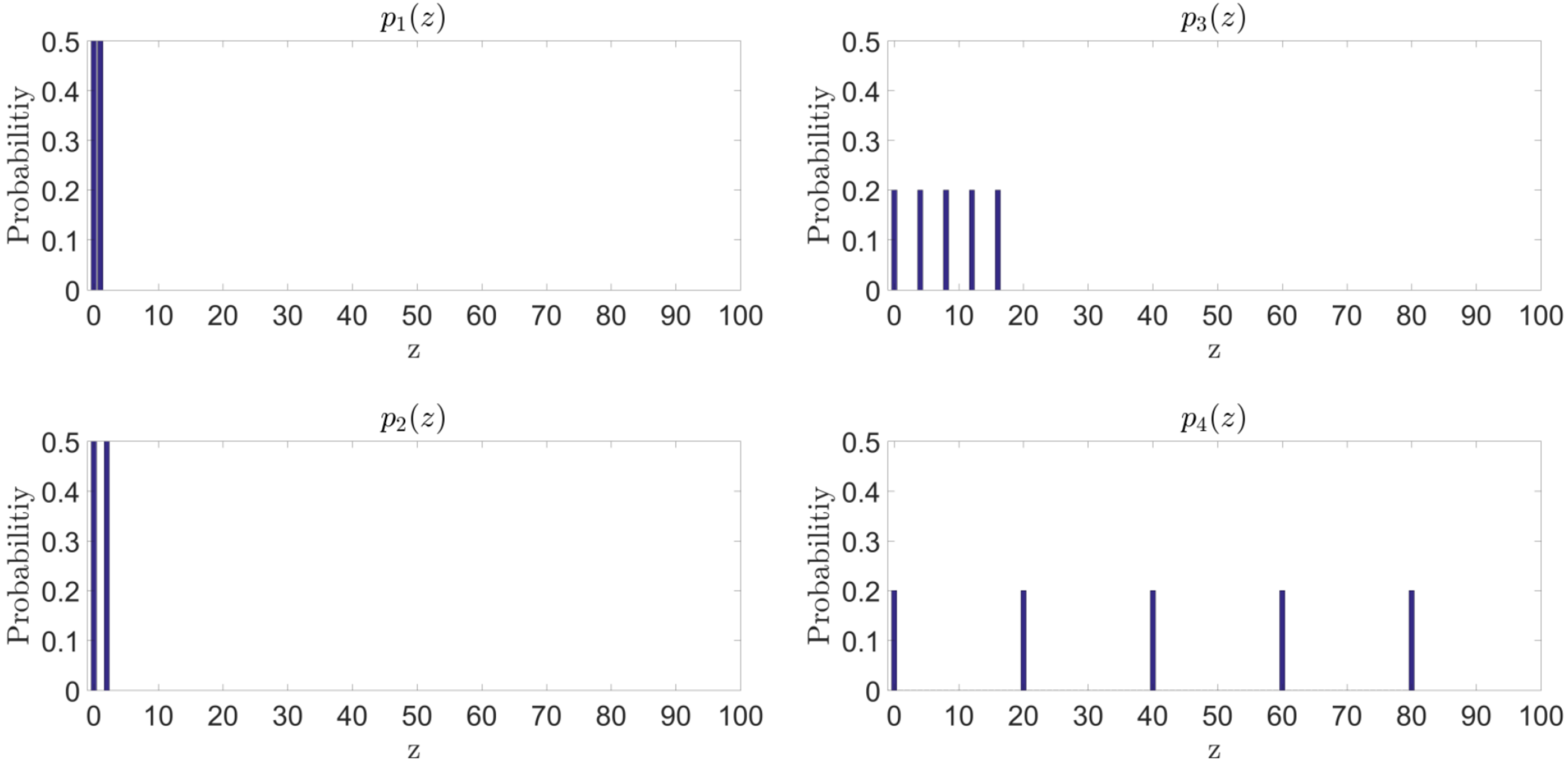


Decomposing a Uniform distribution



- Decomposition can be phrased as a series of convolutions
- Find N and p_i's such that f=p and $\sum|p_i|_0$ is minimized.
- Minimum value of $\sum|p_i|_0$ = sum of prime factorization of $|f|_0$
- Construction algorithm for p_i's in thesis

Decomposing a Uniform distribution on set {0,…,99}



Total support size: 14

Applying edge reductions

Rule	Spatial distribution	Size of region
Face→Nose	Uniform	[25,25]

625 edges

Rule	Spatial distribution	Size of region
Face→Nose-Y	Uniform	[1,25]
Nose-Y→Nose	Uniform	[25,1]

50 edges

Rule	Spatial distribution	Size of region
Face→Nose-Y1	Uniform	[1,5]
Nose-Y1→Nose-Y	Uniform	[1,5]
Nose-Y→Nose-Y2	Uniform	[5,1]
Nose-Y2→Nose	Uniform	[5,1]

20 edges

This talk

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

This talk

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

Directions for Future Research

- More scene understanding tasks
- Integration with Deep Learning
- Grammar learning

This talk

- **Motivation** for a general scene understanding framework
- **Background/related work**
- **Representation** for general scene understanding tasks
- Efficient **approximate inference algorithm**
- **Learning algorithm** to estimate model parameters
- **Experimental evaluation**
- **Extensions** for larger/more complex tasks
- **Directions** for future research

Thanks!

Backup slides start

Face localization **without** a face data model

PSG Face Grammar

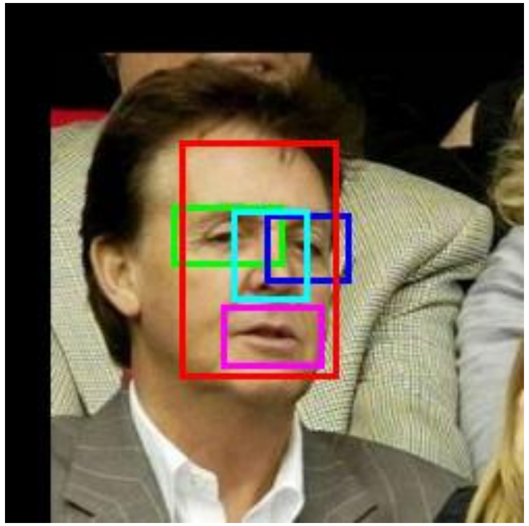
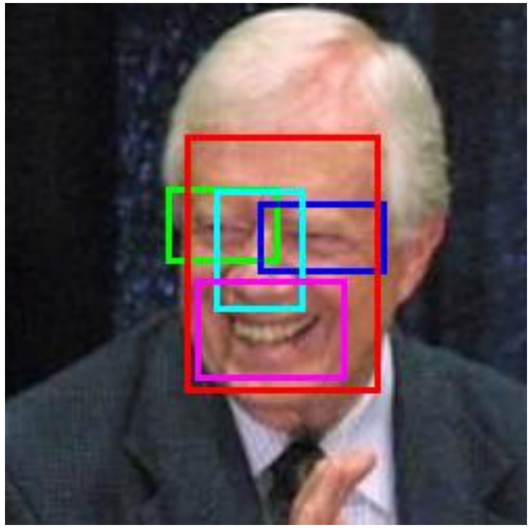
Faceless Grammar

Model	FACE	LEFT-EYE	RIGHT-EYE	NOSE	MOUTH	Average
PSG Face Grammar	0.97	0.81	0.81	0.96	0.80	0.87
Faceless Grammar	0.93	0.78	0.80	0.95	0.76	0.84

Family Portraits: Area under the precision-recall curve

Face localization: 0-1 Face Dataset

- Labelled Faces in the Wild [1] + images from VOC2012[2] without faces
- 200 training, 200 test
- 100 test image have one face, 100 images have no faces.

[1] Huang et al., “Labeled faces in the wild: A database for studying face recognition in unconstrained environments.”, Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[2] Everingham, et al., “The PASCAL Visual Object Classes Challenge 2012 {(VOC2012)} Results”, 2012.

Face localization: 0-1 Face Dataset

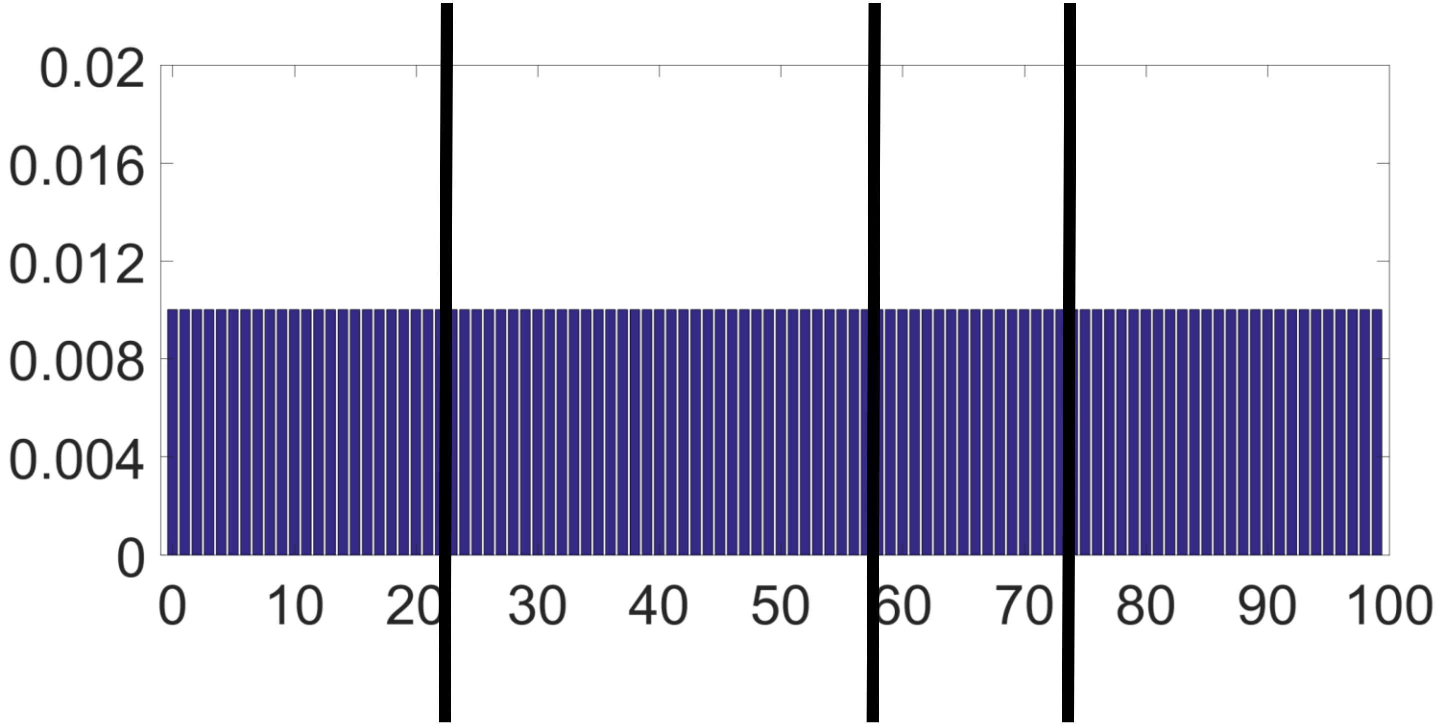
Model	Face	Left eye	Right eye	Nose	Mouth	Average
Pictorial Structures	0.86	0.94	0.86	0.81	0.84	0.86
PSG Face Grammar	1.00	0.98	0.95	0.99	0.93	0.97

Area under the precision-recall curve for the 0-1 Face Dataset

Model	Face	Left eye	Right eye	Nose	Mouth	Average
Pictorial Structures	1.00	0.97	0.93	0.98	0.90	0.96
PSG Face Grammar	1.00	0.98	0.92	0.98	0.92	0.96

Area under the precision-recall curve for the Single-Face Dataset

Decomposing a Uniform distribution with prime support



Search over partitions. Dynamic programming?