NUS National University of Singapore | School of Computing

CS5340 Uncertainty Modeling in AI Class Assignment

Xie Yaqi and Li Chen

AY 2019/20

Semester 1

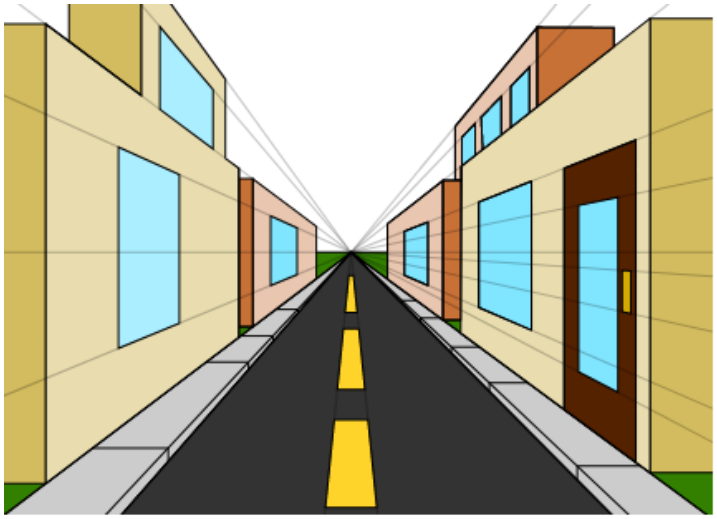
1 Vanishing Point Detection

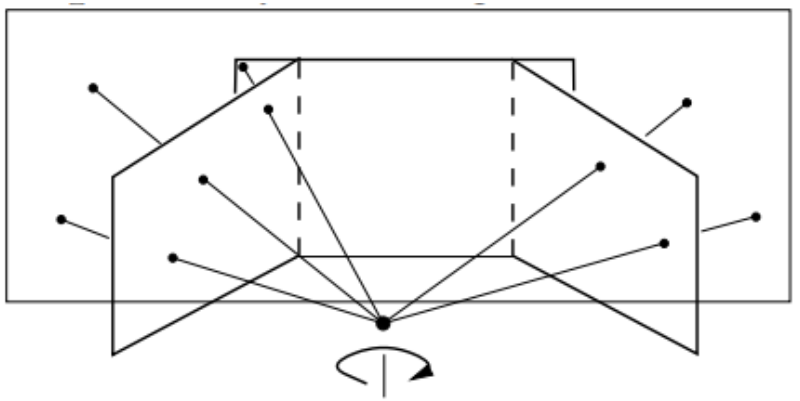
Objective : Estimate vanishing point locations by using EM algorithm.

Images which satisfy the Mahattan World assumption

Vanishing Point

- Definition: A point on the image plane where parallel lines in three-dimensional space appear to converge



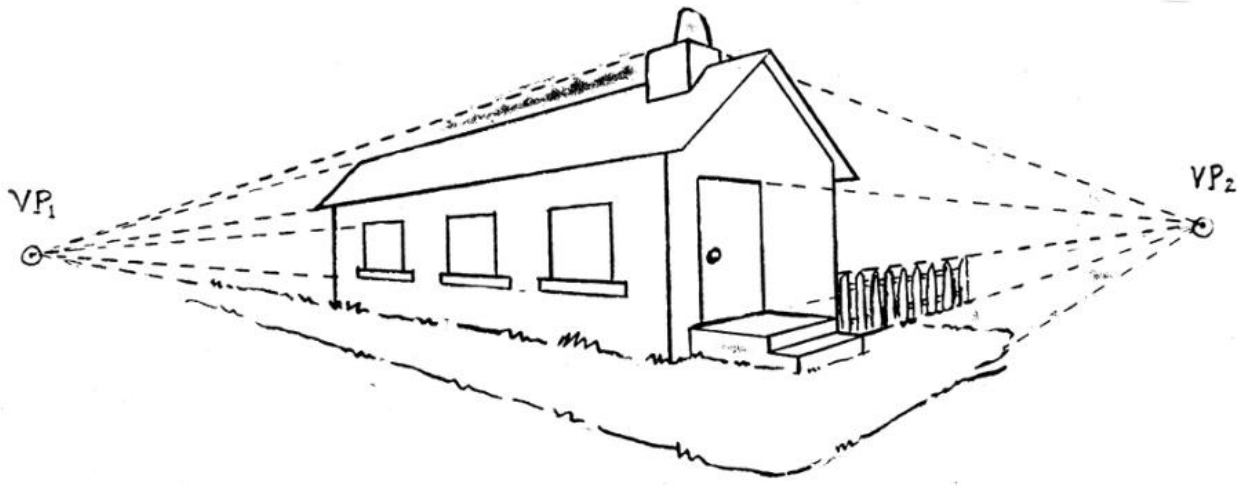


- Reason: the projective property of camera
- Application: Camera calibration $\mathbf{v} = \mathbf{K} * \mathbf{R} * \mathbf{V}$

VP location VP direction

Manhattan World

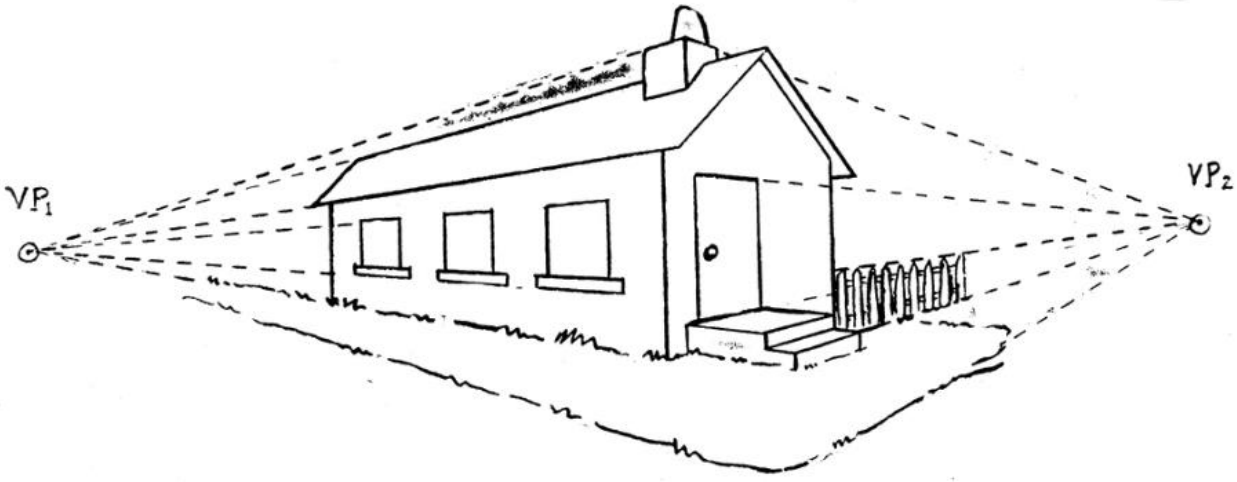
Manhattan world: the scene has a natural Cartesian x, y, z coordinate system.



The edge orientation is determined by the orientation of the viewer (or the camera rotation) with respect to the coordinate system.

Manhattan World

Manhattan world: the scene has a natural Cartesian x, y, z coordinate system.

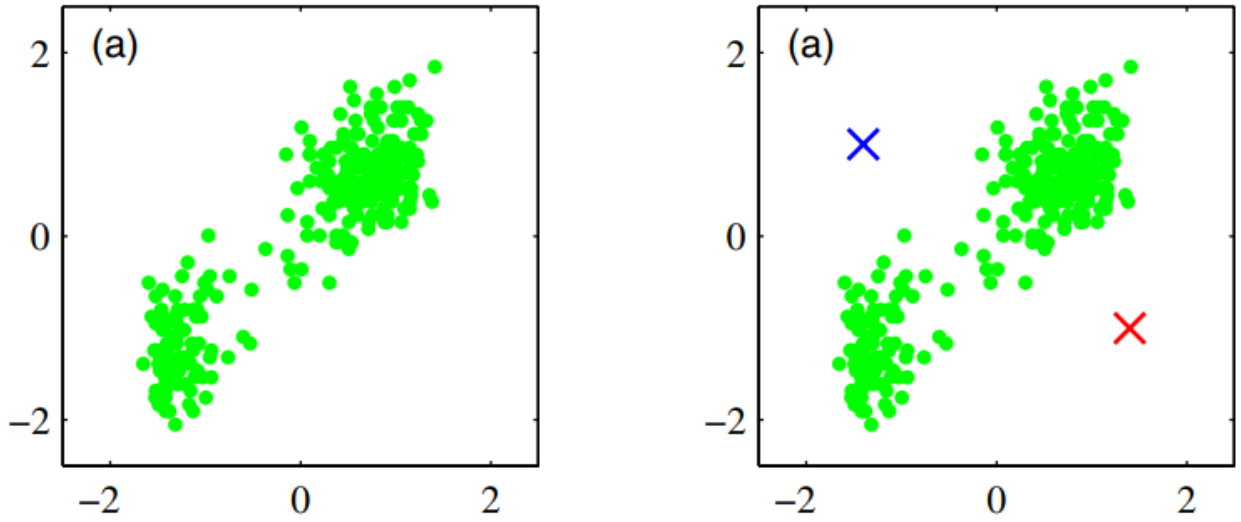


if we know the vanishing point (or camera rotation), we can know which pixels are assigned to it, and vice versa.

What if both are unknown?

Expectation-Maximization algorithm

Example: K-means Clustering



Goal: Find the assignment and center to minimize the distance

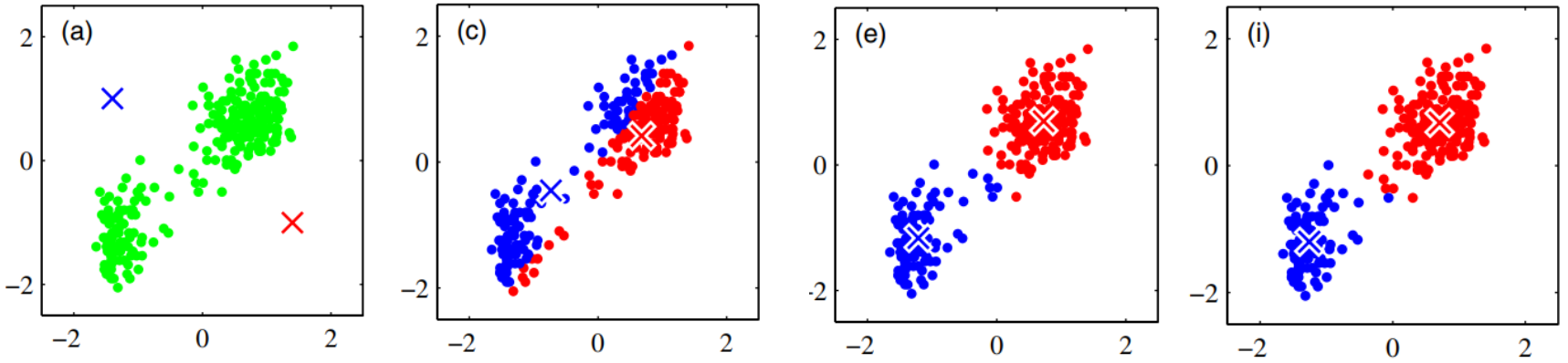
$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

Assignment Center

Image source: C.M.Bishop, 2006

Expectation-Maximization algorithm

Example: K-means Clustering



- Iterative procedure

Assignment

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{otherwise.} \end{cases}$$

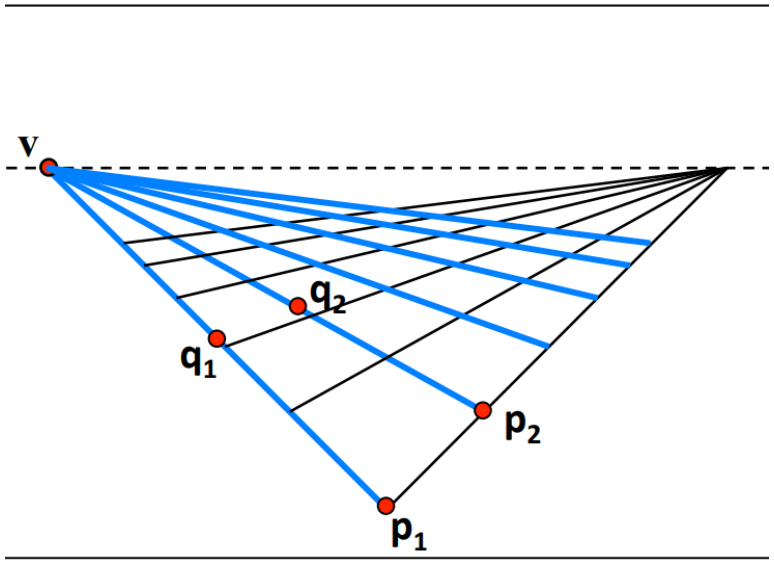
Optimization

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}}$$

Image source: C.M.Bishop, 2006

Expectation-Maximization algorithm

EM algorithm [1]: compute the assignment of each pixel and camera rotation alternatively



- Initialization: initialize the camera rotation matrix $\mathbf{R}$

Equivalent to initialize VP locations

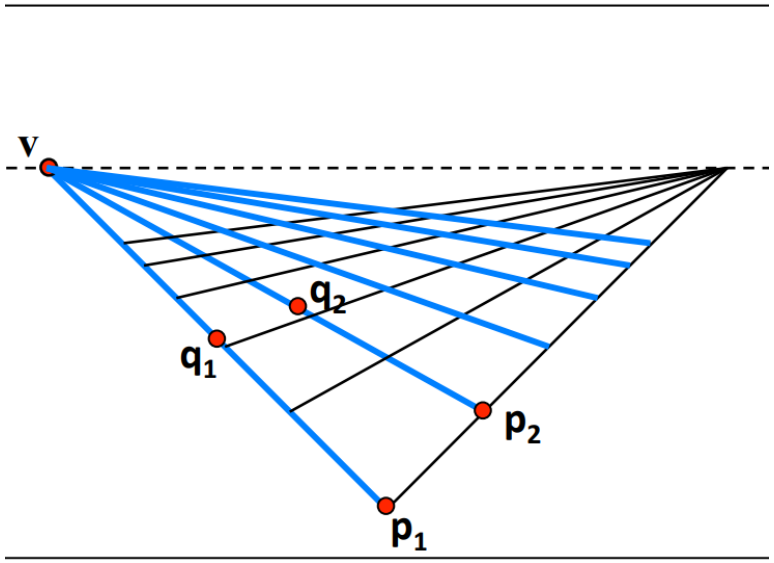
$$\mathbf{v} = \mathbf{K} * \mathbf{R} * \mathbf{V}$$

VP directions: $\mathbf{V} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$

[1] G. Schindler and F. Dellaert, "Atlanta world: an expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments," CVPR 2004.

Expectation-Maximization algorithm

EM algorithm: compute the assignment of each pixel and camera rotation alternatively



- E-step: solve the assignment problem, posterior a pixel belongs to different vanishing points M

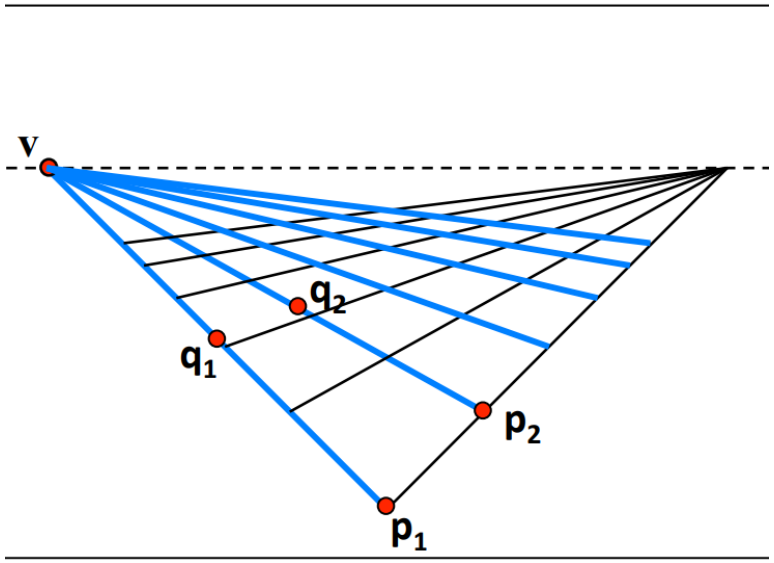
$$P(M \mid G, \Psi^{old}) \propto P(G \mid M, \Psi^{old}) P(M)$$

(Bayesian rule) likelihood prior

$\mathbf{R}$ is fixed, likelihood is computed based on edge orientation

Expectation-Maximization algorithm

EM algorithm: compute the assignment of each pixel and camera rotation alternatively



- M-step: Optimize over camera rotation parameters

$$Q(\Psi, \Psi^{old}) \triangleq \langle \log P(G \mid M, f(\Psi)) \rangle + \log P(\Psi)$$

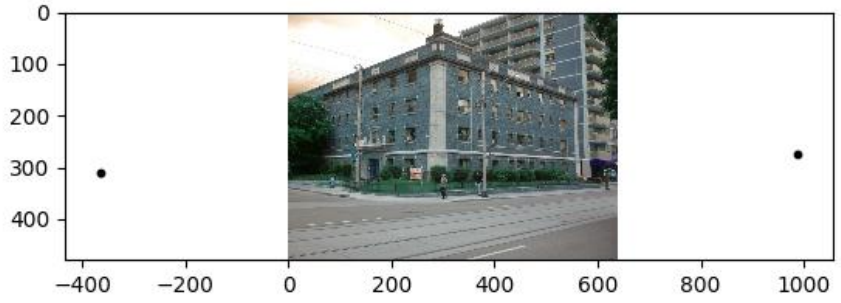
$$\Psi_{new} = \underset{\Psi}{\operatorname{argmax}}\, Q(\Psi; \Psi^{old})$$

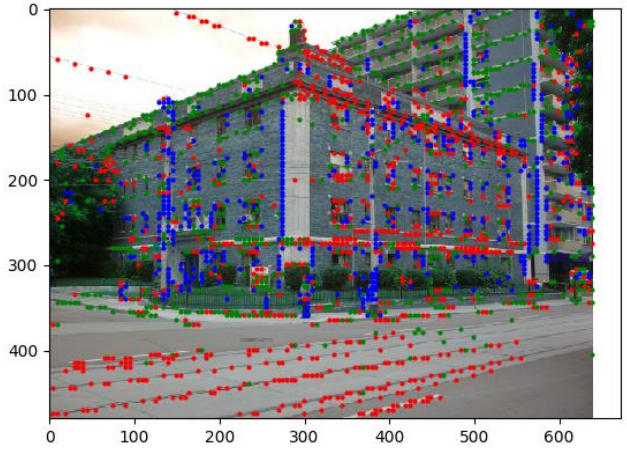
Assignment is fixed, optimize $\mathbf{R}$

The two steps are iterated until converge, get both the VP locations and assignment of each pixel.

Task

Objective : Find the VP locations and assignment of each pixel by using EM



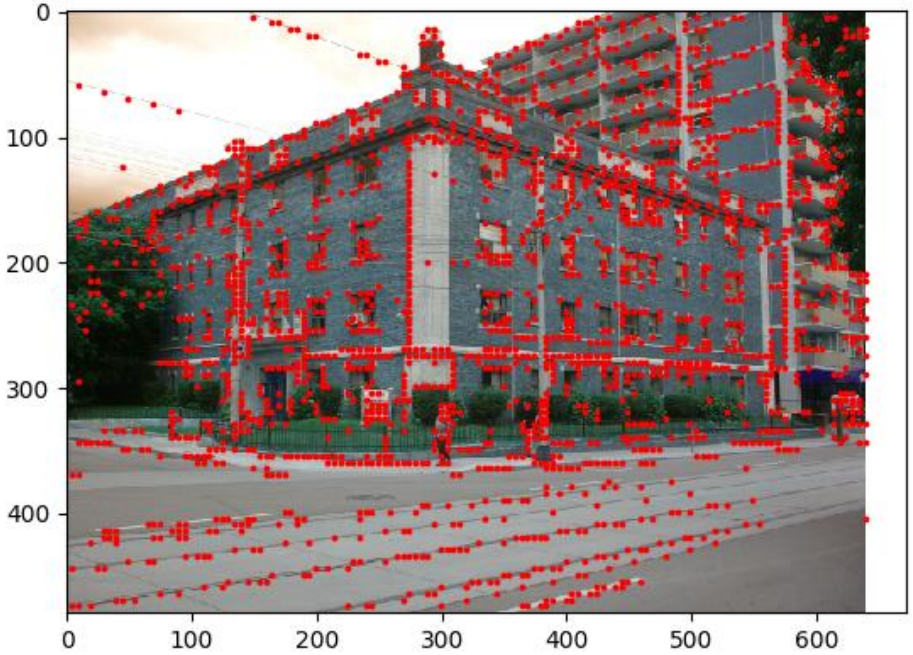


Steps

- Compute the gradient magnitude and orientation

  Convert to gray scale image first : cv2.cvtColor, cv2.Sobel

  Down sampling the image for computational efficiency



Steps

- EM algorithm needs good initialization

  Bayesian approach[2] : perform a brute-force search over the camera parameters

- E-step: solve the assignment problem based on parameters from the previous step

- M-step: maximize over the camera parameters

[2] James M. Coughlan , A. L. Yuille. Manhattan world: orientation and outlier detection by Bayesian inference. Neural Computation

Help Functions

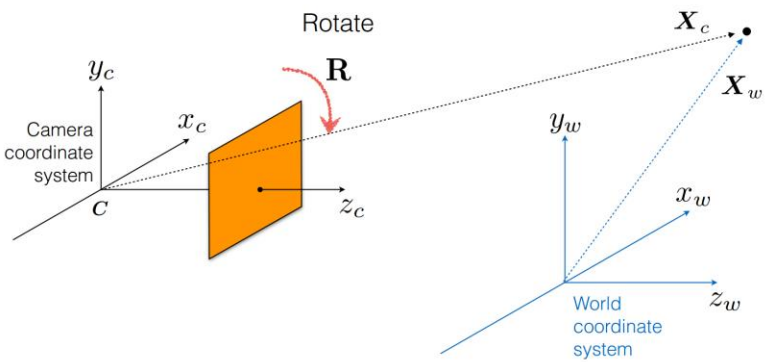
- How to compute vanishing point locations in image plane

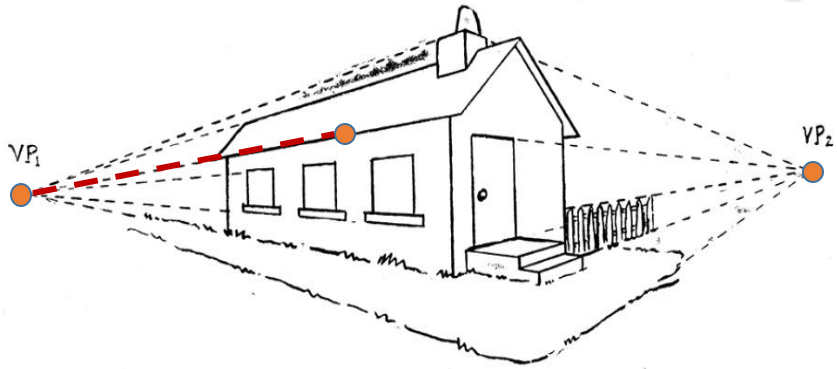
$$\mathbf{v} = \mathbf{K} * \mathbf{R} * \mathbf{V}$$

- How to compute the predicted edge direction at each pixel

$$v_m \times u = [x, y, 1]^T$$

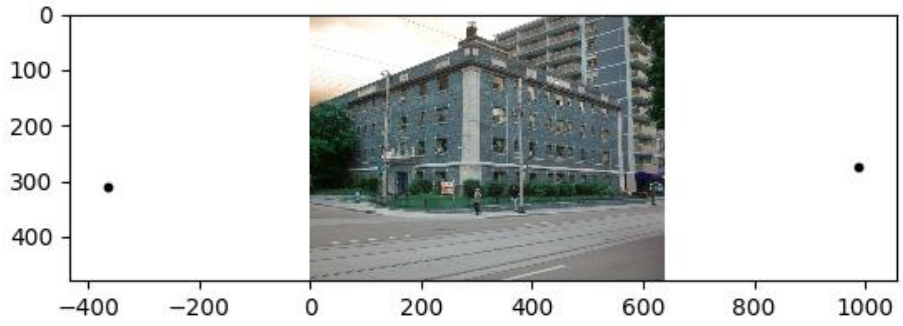
- Other functions that related to geometry

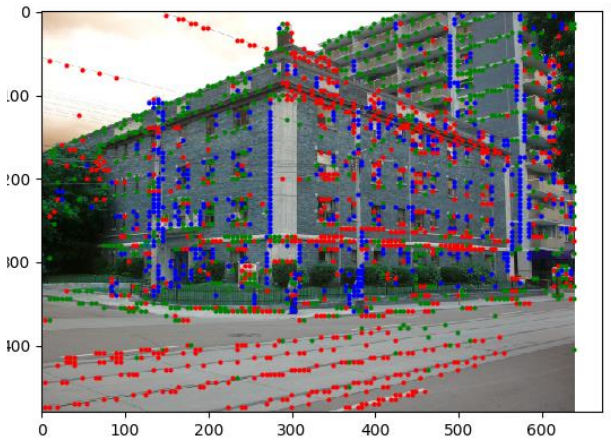




Submission

- Source code and results



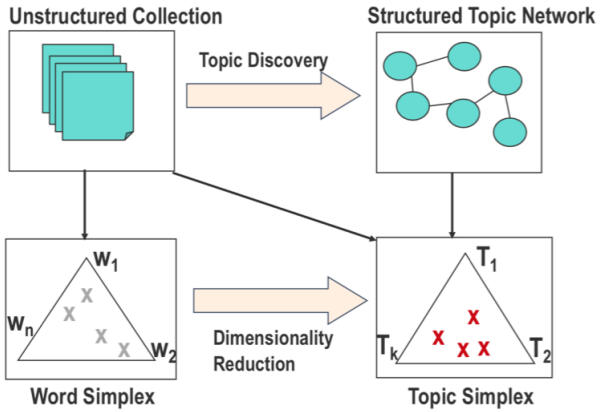


- Report
  - Describe how to apply the EM algorithm to vanishing point estimation
  - Show the results and brief analysis

2. Latent Dirichlet Allocation (LDA)

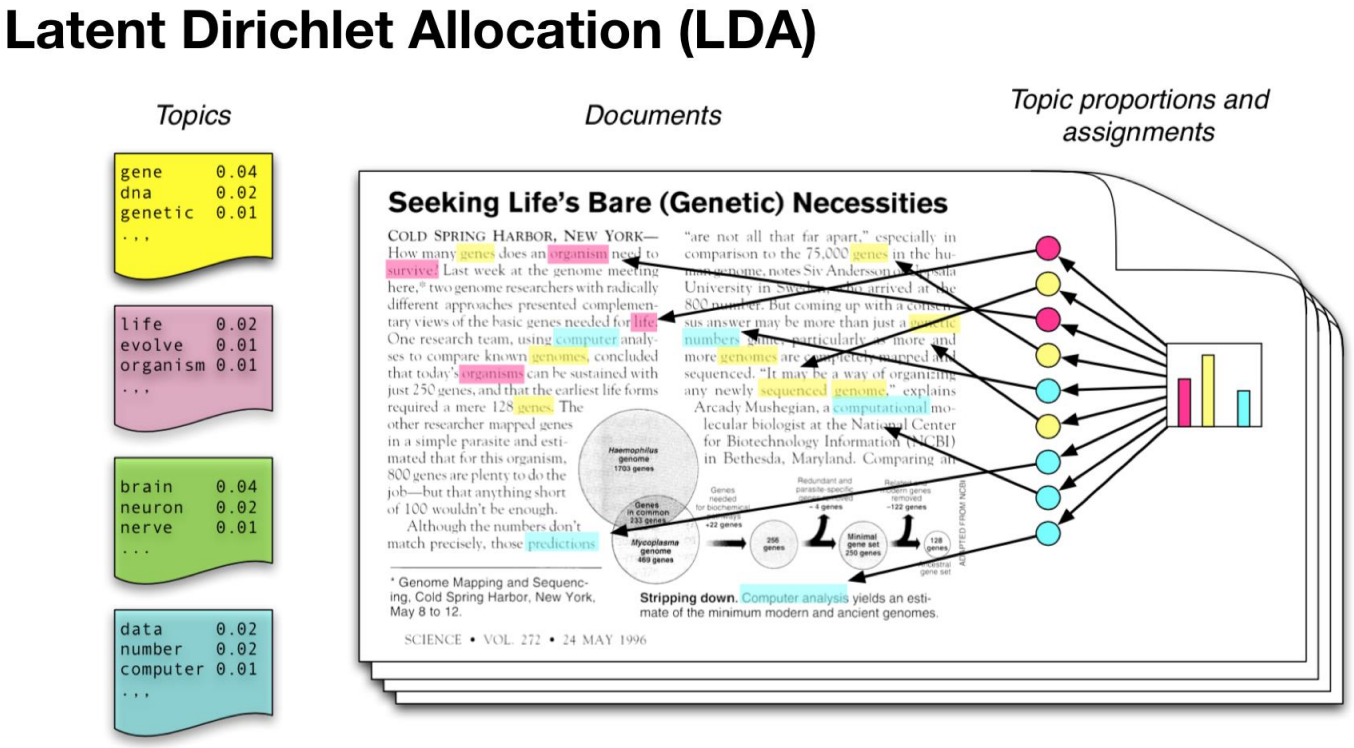
Topic Model

- Why? e.g. Summary document
  - Bag of words (distribution of words)
    - simple representation
    - order is ignored
    - too many possible words



  - Use topics -> has semantic meanings

2. Latent Dirichlet Allocation (LDA)



Each **topic** is a distribution over words

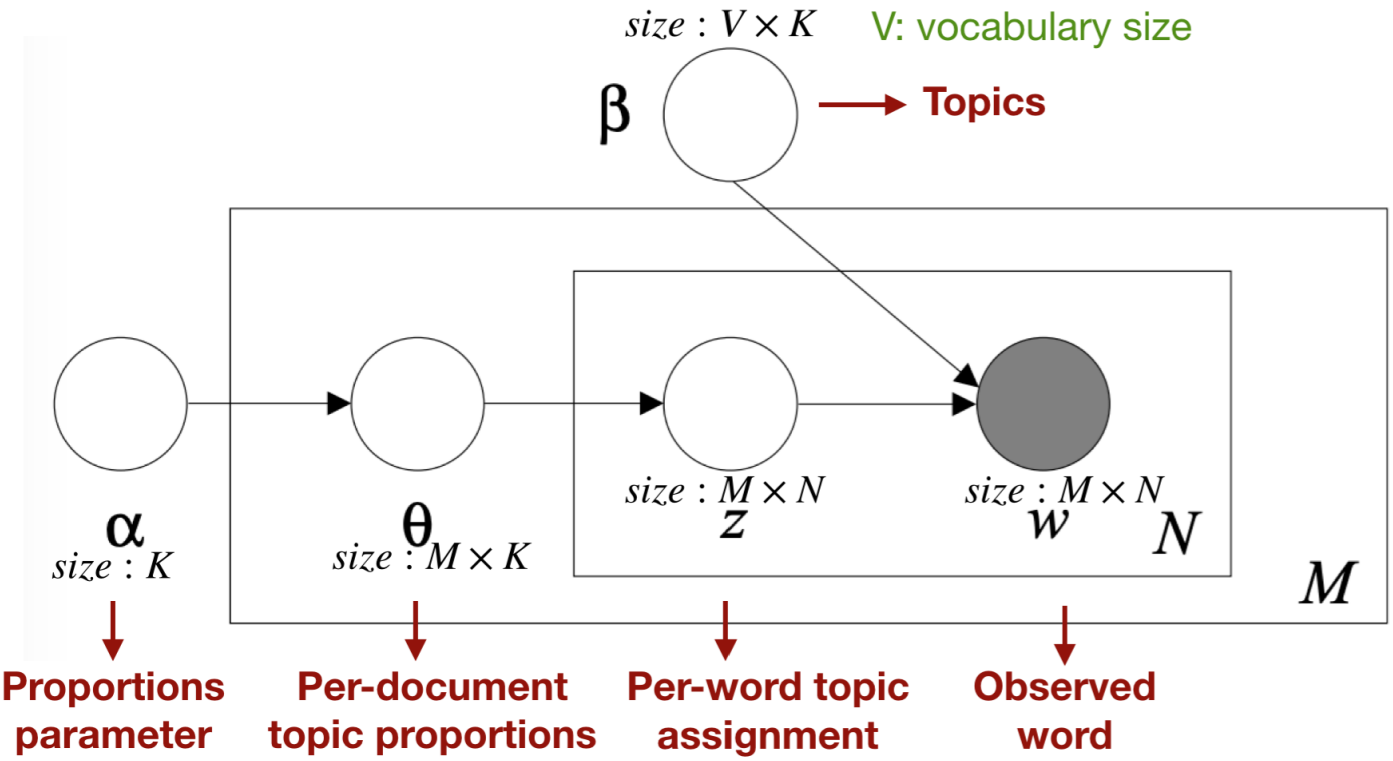
Each **document** is a mixture of corpus-wide topics

Each **word** is drawn from one of those topics

2. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

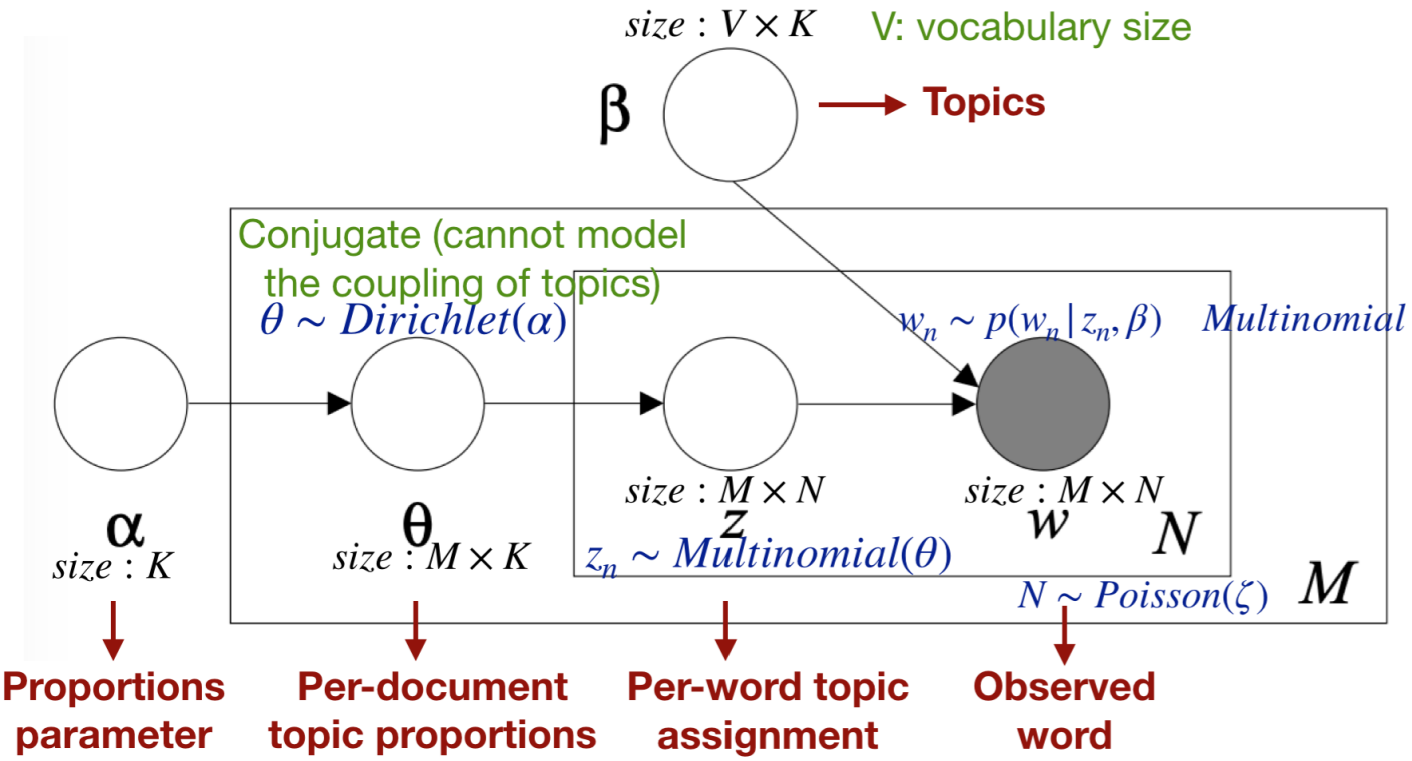
- Generative model - three-level hierarchical Bayesian model



2. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

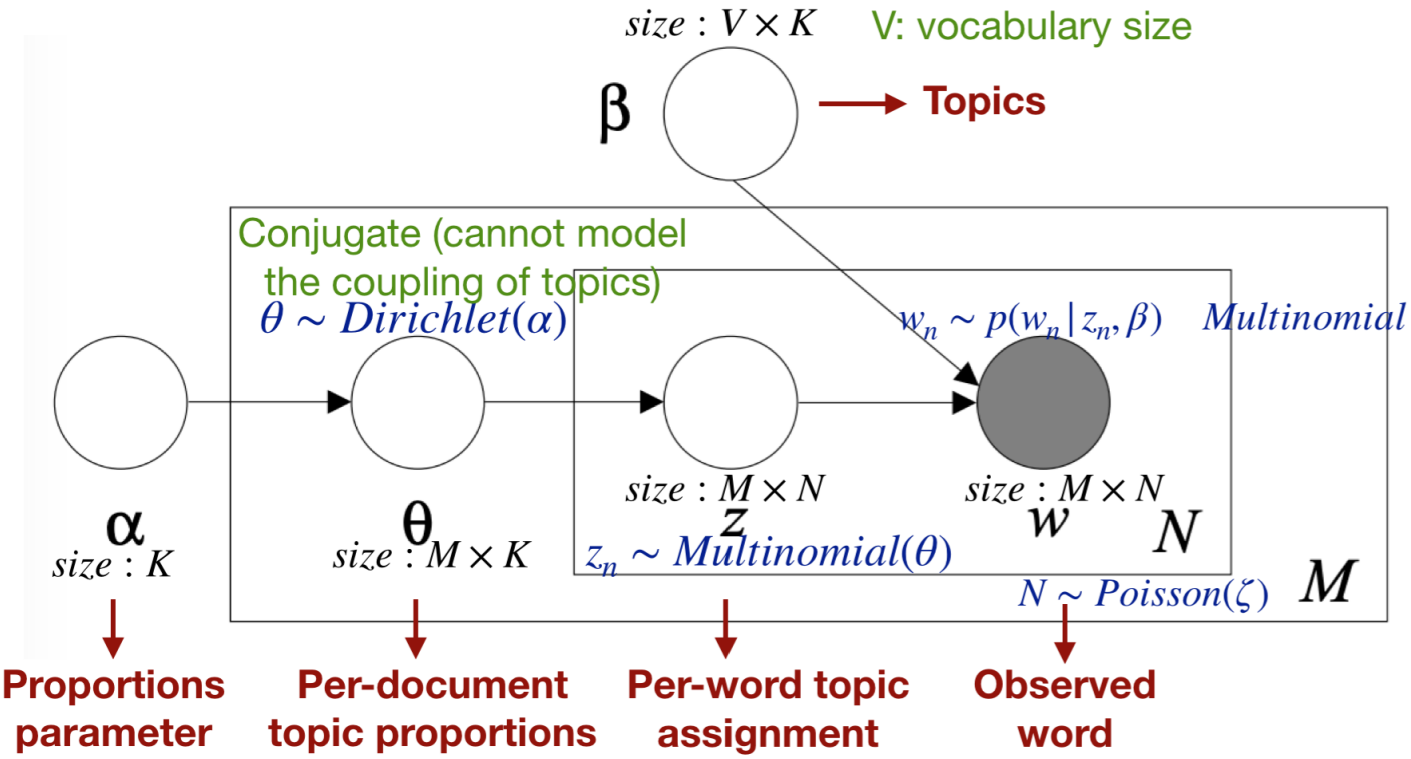
- Generative model - three-level hierarchical Bayesian model



2. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

Joint Probability: $p(\theta, z, w \mid \alpha, \beta) = p(\theta \mid \alpha) \prod_{n=1}^{N} p(z_n \mid \theta) p(w_n \mid z_n, \beta)$



2. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

- Inference and learning are both intractable -> too many configurations, the normalizing term of posterior is intractable
- **Problem:** Joint distribution $p(X, Z)$ is easy to compute, posterior distribution $p(Z|X)$ is intractable
- Approximate inference: **Variational Inference**
  - Turn inference into optimization problem
  - Given a joint distribution $p(X, Z)$, our goal is to find an approximation $q(Z)$ for the posterior distribution $P(Z|X)$ which is intractable

2. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

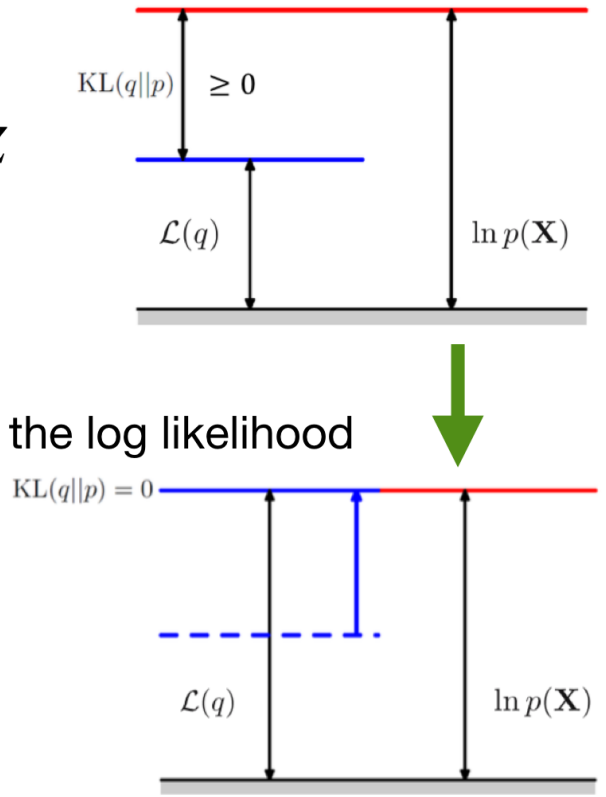
- **Variational Inference**
  - **Objective:** Minimize the KL divergence

$$KL(q||p) = -\int q(Z) ln\{\frac{p(Z|X)}{q(Z)}\} dZ$$

  - Hard since $p(Z|X)$ is intractable
  - **Solution:** maximize the lower bound of the log likelihood

$$L(q) = \int q(Z) ln\{\frac{p(X,Z)}{q(Z)}\} dZ$$

  - $log\ p(x) = KL(q||p) + L(q)$



2. Latent Dirichlet Allocation (LDA)

Mean Field assumption

- True posterior: $p(\theta, z \mid w, \alpha, \beta) = p(\theta, z, w \mid \alpha, \beta)/p(w \mid \alpha, \beta)$
- Break the dependency using fully factorized distributions; the coupling between θ, β is removed

$$q(\theta, z \mid \gamma, \phi) = q(\theta \mid \gamma) \prod_{n=1}^{N_i} q(z_n \mid \phi_n)$$

- Free variational parameters: Dirichelet parameter γ and multinomial parameters $(\phi_1, \phi_2 \ldots, \phi_{N_i})$

2. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

- **Mean Field Variational Inference**
  - $(\gamma^*, \phi^*) = argmin_{(\gamma,\phi)} D(q(\theta, z \mid \gamma, \phi) \mid\mid p(\theta, z \mid w, \alpha, \beta))$
  - Minimization can be achieved by iterative fixed-point method
  - $\phi_{ni} \propto \beta_{iw_n} exp\{E_q log(\theta_i) \mid \gamma\} \propto \beta_{iw_n} exp\{\Phi(\gamma_i) - \Phi(\sum_{j=1}^{K} \gamma_j)\}$, where Φ is the first derivative of $log\ \Gamma$ function (hint: check out *scipy.special.digamma* function)
  - $\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni}$

2. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

- **Mean Field Variational Inference**
  - Note that the parameters $(\gamma^*(w), \phi^*(w))$ are document-specific

initialize $\phi_{ni}^0 := 1/k$ for all i and n
initialize $\gamma_i := \alpha_i + N/k$ for all i
repeat
  for $n = 1$ **to** N
    for $i = 1$ **to** k
      $\phi_{ni}^{t+1} := \beta_{iw_n} \exp(\Psi(\gamma_i^t))$
    normalize ϕ_n^{t+1} to sum to 1.
  $\gamma^{t+1} := \alpha + \sum_{n=1}^{N} \phi_n^{t+1}$
until convergence

2. Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

- Discovering human ancestry - classify individuals into populations according to their genotypes
  - K ancestor population
  - Individual is an admixture of the ancestor populations
  - **Task:** implement variational inference to infer the population mixture (θ) and genotype ancestry assignments (z) for each individual

2. Latent Dirichlet Allocation (LDA)

Dataset

- M = 100 individuals (documents)
- N = 2000 genotype loci (words)
- K = 4 ancestor populations (topics)
- β matrix of size $N \times K$
- $\alpha = 0.1$

2. Latent Dirichlet Allocation (LDA)

Your task

- 1. Report variational inference update equation for γ, ϕ
- 2. For individual 1, run LDA inference to find ϕ for each genotype locus, store it as a matrix of size $n_1 \times K$ (where $n_1 := \sum_{1,j} I(D_{1j} \neq 0)$), $I(\cdot)$ being the indicator function, is the number of non-zero genotypes represent in individual 1.
- 3. Construct a matrix Θ of size $M \times K$ to represent the ancestor assignments for all individuals in the population. For each individual i, run LDA inference to find γ, and store it as row of Θ, i.e. $\Theta_i = \gamma$. (hint: update probabilities in log-space to avoid overflow and underflow issues)

2. Latent Dirichlet Allocation (LDA)

Your task

- 4. Report the number of iterations and time taken to get to convergence for running inference on all M individuals. (convergence criteria: absolute change in each value of γ, ϕ is less than $e = 1e^{-3}$)

- 5. Repeat the experiments for $\alpha = 0.01, \alpha = 1, \alpha = 10$, and discuss the change in the ancestor population assignments to the individuals Θ, and iterations required for convergence change as α change

2. Latent Dirichlet Allocation (LDA)

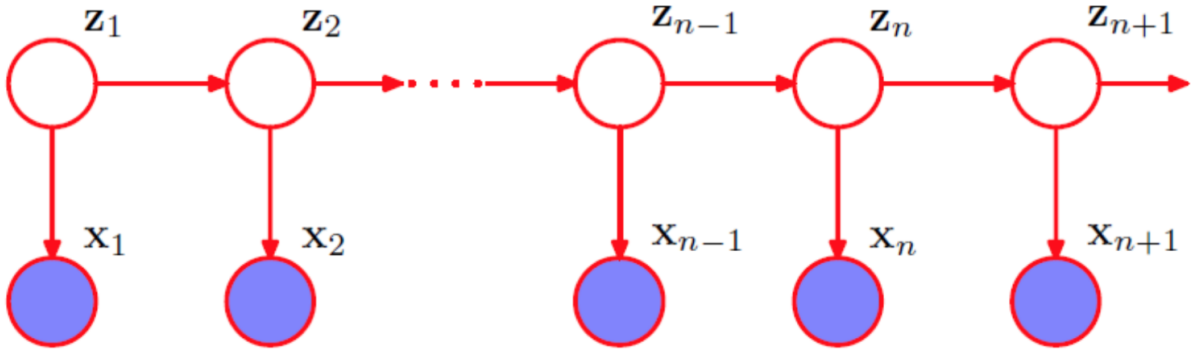
Implementation

- **Allowed packages:** numpy, scipy, pandas, date time
- **Function to load matlab data**
  - data = spcipy.io.loadmat(path)
- **You can use** scipy.special.diagamma **to calculate** $\Phi(\cdot)$

3. Seam Carving as HMM

Hidden Markov Model (HMM)

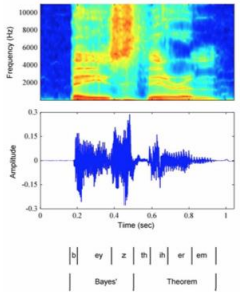
- State space model
- Markovian assumption (conditional independence): $z_{n-1} \perp z_{n+1} \mid z_n$
- Latent variables must be discrete (K possible states), observed variable can be either discrete or continuous



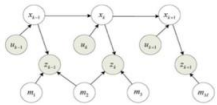
3. Seam Carving as HMM

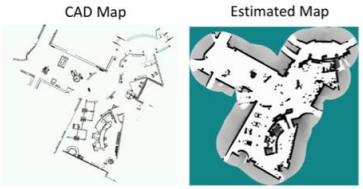
Hidden Markov Model (HMM)

- **Applications:**
  - Speech recognition



  - Natural Language Processing: part of speech tagging
  - Robot SLAM



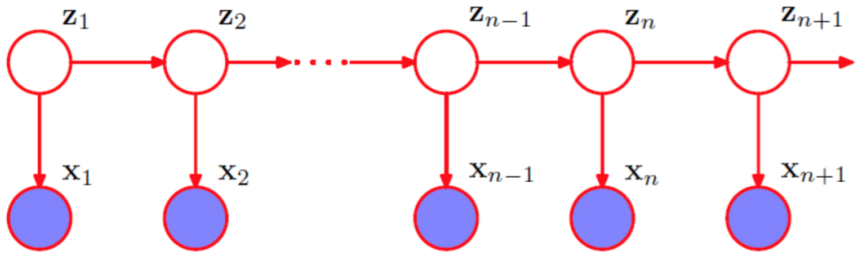


  - Financial forecasting

3. Seam Carving as HMM

Hidden Markov Model (HMM)

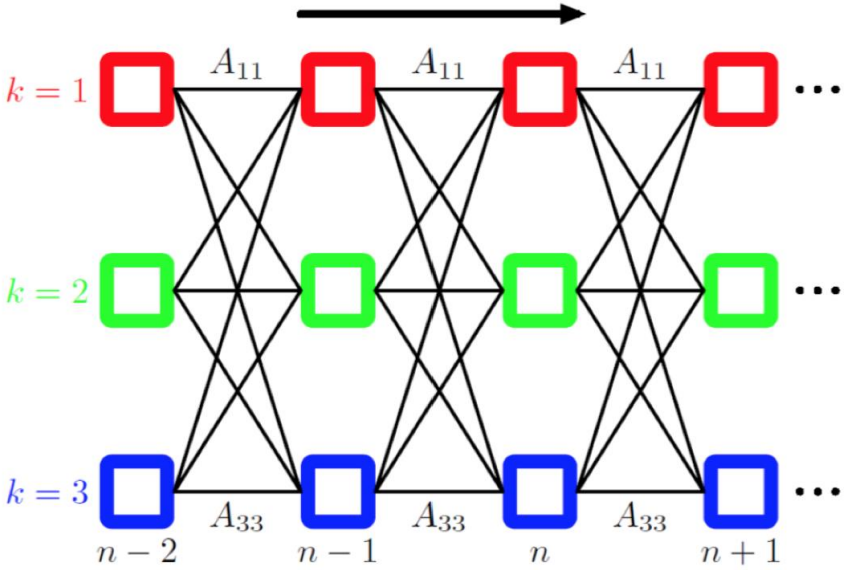
- Transition probability: $p(z_n \mid z_{n-1})$
- Emission probability: $p(x_n \mid z_n)$
- Joint probability: $p(X, Z) = \prod_{n=1}^{N} p(z_n \mid z_{n-1}) p(x_n \mid z_n)$, note that we can add dummy start node z_0



3. Seam Carving as HMM

Hidden Markov Model (HMM) - Task

- Assume probabilities are known, find the most probable sequence of hidden states for a given observation seq
- Viterbi Algorithm - dynamic programming

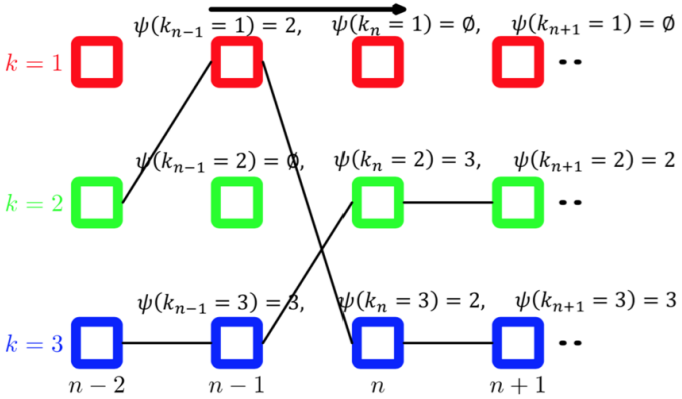


3. Seam Carving as HMM

Viterbi Algorithm

- Compute recursively (work with log probabilities):
$$w(z_{n+1}) = ln\ p(x_{n+1} \mid z_{n+1}) + max_{z_n}\{ln\ p(z_{n+1} \mid z_n) + w(z_n)\}$$

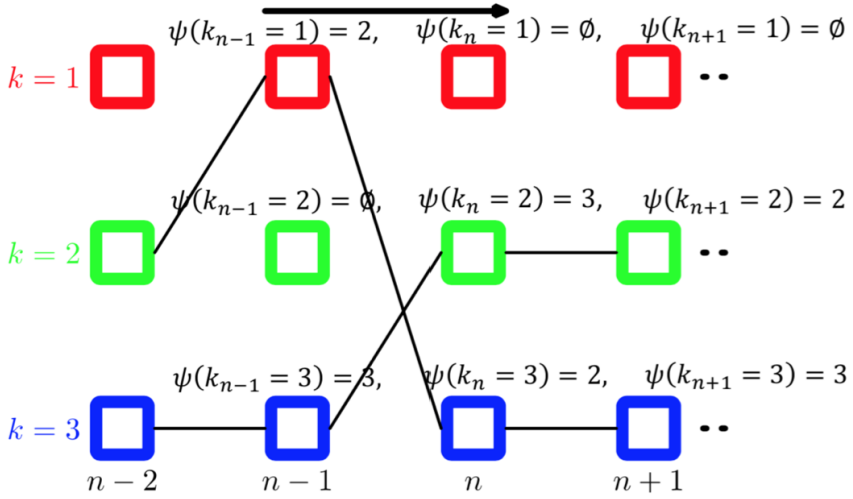
- Back-tracking: Keep a record of the values of z_n that correspond to the maxima for each value of the K values of z_{n+1}, denoted by $\phi(k_n)$ where $k = 1,...,K$



3. Seam Carving as HMM

Viterbi Algorithm

- We get $\phi(k_N)$ when we reach the end of the chain z_N
- The sequence of latent variable values that corresponds to the maximal probability can be obtained by backtracking the chain recursively: $k_n^{max} = \phi(k_{n-1}^{max})$



3. Seam Carving as HMM

Seam Carving

- Context aware image resizing (a change in size of the image by modifying the least noticeable pixels in an image)

Original

Scaling ✗

Cropping ✗

Seam Carving ✓

3. Seam Carving as HMM

Scaling down

3. Seam Carving as HMM

Scaling up

3. Seam Carving as HMM

Object Removal

3. Seam Carving as HMM

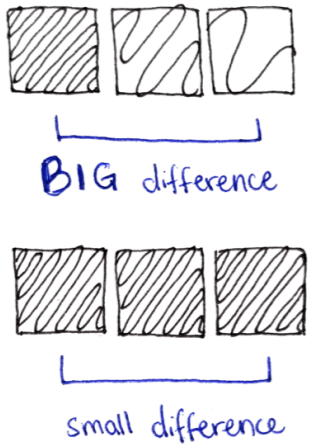
Seam Carving

- **Seams:** A vertical (horizontal) seam is a path of pixels connected from top (left) to bottom (right) in an image with one pixel in each row (column)

- Find the seam with minimal energy (sum of the energy of the pixels on the seam)

- **Assign energy to each pixel:**

$$|\Delta x|^2 = (\Delta r_x)^2 + (\Delta g_x)^2 + (\Delta b_x)^2$$
$$|\Delta y|^2 = (\Delta r_y)^2 + (\Delta g_y)^2 + (\Delta b_y)^2$$
$$e(x,y) = |\Delta x|^2 + |\Delta y|^2$$



3. Seam Carving as HMM

Seam Carving as HMM

- For example, finding vertical seams
  - Find each pixel per row
  - No. of columns = no. of possible states for hidden variable K
  - Transition probability (ensure continuity):

$$p(z_n = k' \mid z_{n-1} = k) = \begin{cases} 1 & \text{if } |k - k'| \leq 1 \\ \infty & \text{else} \end{cases}$$

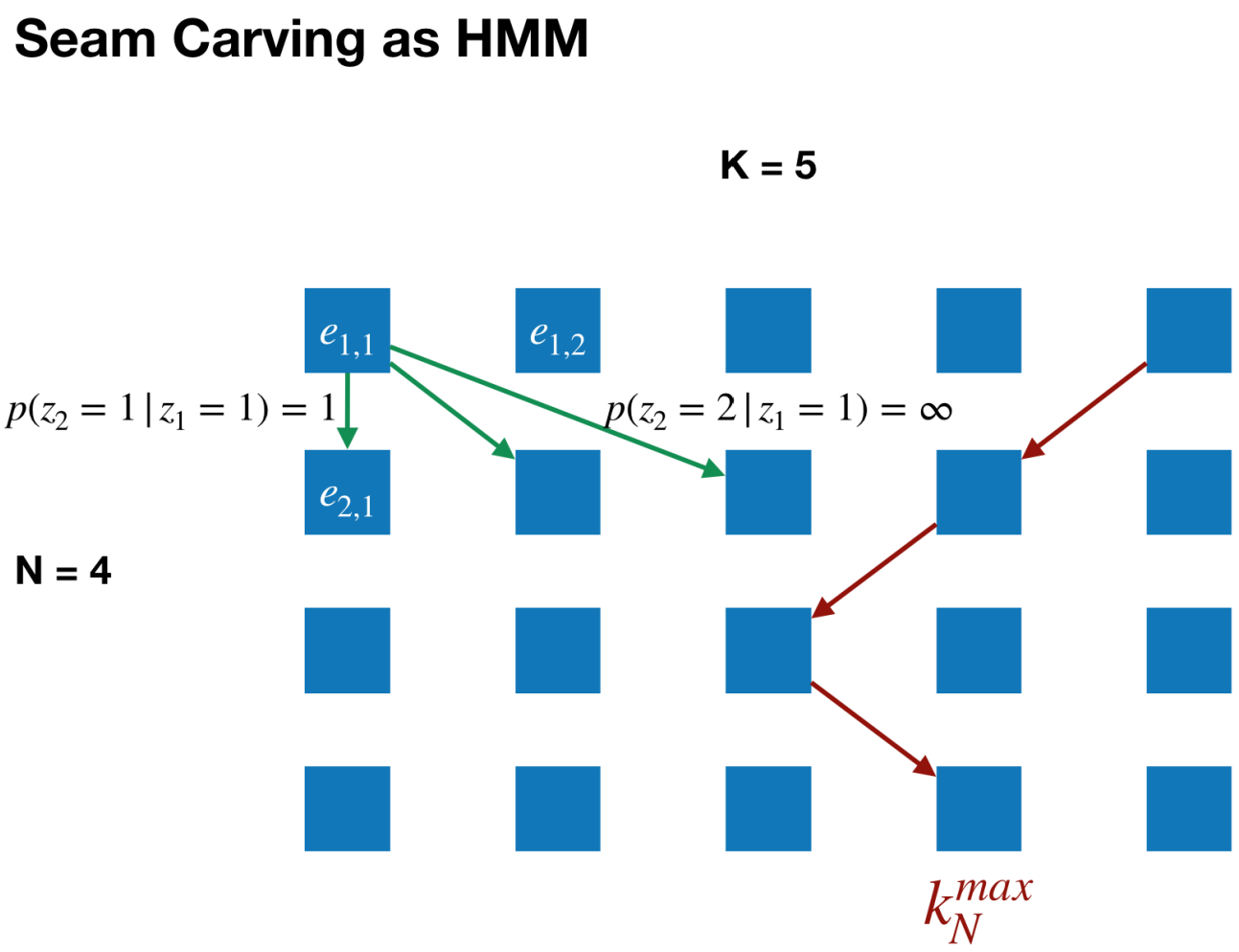
  - Emission probability: $p(x_n \mid z_n = k) = e(n, k)$

3. Seam Carving as HMM

Seam Carving as HMM

- Use Viterbi algorithm
  - Replace max with min, i.e. find the sequence of latent variables corresponds to the max probability -> corresponds to the min energy

3. Seam Carving as HMM



3. Seam Carving as HMM

Your task

- Input: image
- Output: context aware resized image

3. Seam Carving as HMM

Implementation

- Allowed packages: numpy, cv2
- Functions for reading, writing images, calculating pixel energy, removing seams are already implemented for you. You only need to fill in the Viterbi algorithm part

Submission

- **Deadline:** 17 Nov 2019, 2359hrs via NUS Luminus. (25% of the total score will be deducted for each day of late submission)
- **Group:** a maximum of 2 students (3 is allowed with written approval) per group
- **Submission format:**