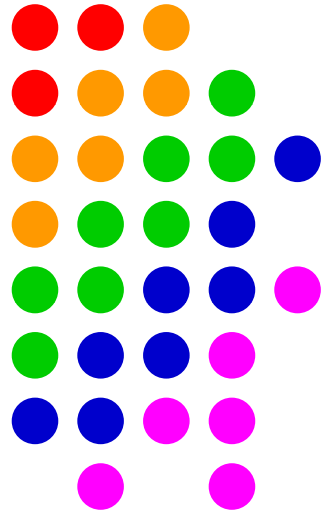
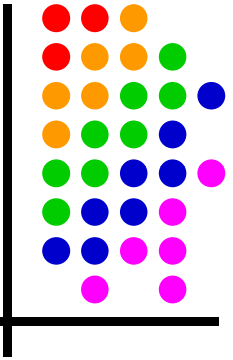
Separating Foreground from Backgrounds

Minglun Gong
Dept. of Computer Science
Memorial Univ. of Newfoundland
St. John's, NL, Canada



Outline



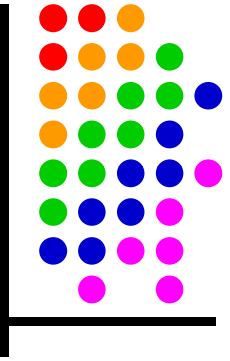
- Foreground extraction problem
  - Existing approaches
  - Background subtraction
  - Background cut
- Matte extraction problem
  - Existing approaches
  - Matting by triangulation
  - Poisson matting

Foreground Extraction

Images from Sun, et al. 2006

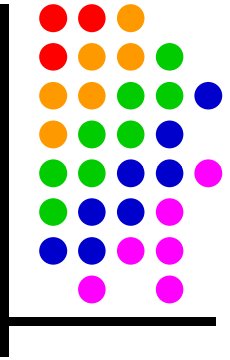
- **Objective:**
  - **Extract foreground object from the images/videos with natural background**
  - **Generate a mask that labels all the foreground pixels**
- **Application:**
  - **Live background substitution**

Existing Techniques



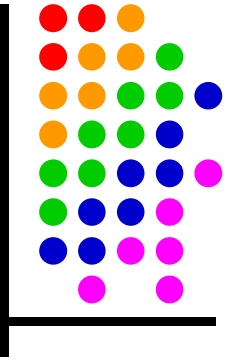
- **Static background:**
  - **Background subtraction**
  - **Background cut [Sun, et al. 2006]**
- **Dynamic background:**
  - **Online 1-SVM [Li, et al. 2006]**
  - **Bilayer segmentation for stereo video [Kolmogorov, et al. 2005]**

Background Subtraction



- A straightforward & efficient approach:
  - Assume a pure background image is available
  - Make decision based on the difference between the current image & the background image
  - Local per-pixel based optimization
- Limitations:
  - Result quality relies on the threshold value
  - Noise & illumination changes may cause false positive
  - Color similarity between foreground & background may cause false negative

Global Optimization



- **Find a label assignment L that maximizes the following posterior probability:**

$$L^* = \arg\max_L p(L \mid I)$$

  - **Based on Bayes' theorem:**

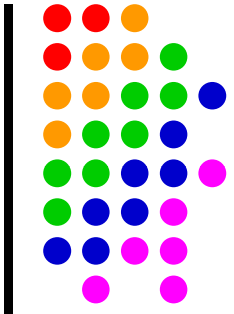
$$p(L \mid I) = \frac{p(L, I)}{p(I)} = \frac{p(I \mid L)p(L)}{p(I)} \propto p(I \mid L)p(L)$$

  - **Gibbs energy is defined as:**

$$E(L) = -\log p(L \mid I) = -[\log p(I \mid L) + \log p(L)]$$

  - **Maximizing probability is equivalent to minimizing Gibbs energy**

Background Cut

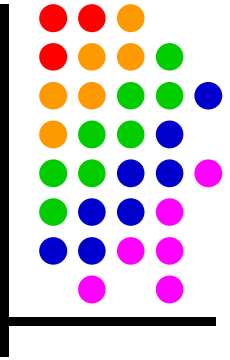


- Use color/contrast based model:
  - Model background using color distribution
  - Model interaction using contrast
- Assign a label $L_s \in \{B,F\}$ to each pixel s that minimizes a global energy function:

$$L^* = \arg\min_L \sum_{s \in I} E_{color}(L_s) + \lambda \sum_{(s,t) \in \Omega} E_{contrast}(L_s, L_t)$$

  - E_{color} : the color term that measures the labeling agrees with the measurements
  - $E_{contrast}$: the contrast term that encourages adjacent pixels with similar colors to have the same label

Global Background Modeling

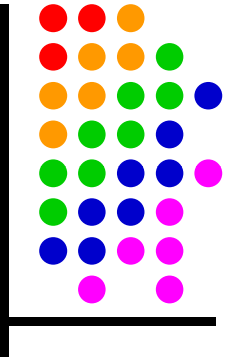


- **Pre-captured background image can be used to initialize a global background model**
  - **Model all background colors using a mixture of K Gaussians:**

$$p^G\left(I_s \middle| L_s = B\right) = \sum_{k=1}^{K} \omega_k^B N\left(I_s \middle| \mu_k^B, \Gamma_k^B\right)$$

  - **ω_k : the weight of the kth Gaussian**
  - **μ_k : the mean color of the the kth Gaussian**
  - **Γ_k : the covariance matrix of the kth Gaussian**
- **The parameters can be updated as more background colors are observed**

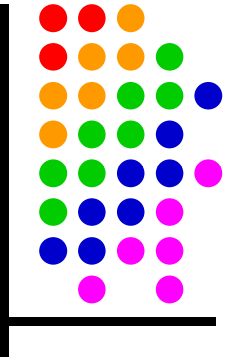
Local Background Modeling



- Even for static background, the observed colors at a given pixel location may change
  - Image capture noise
  - Illumination & exposure changes
- To adapt to temporal changes, the color variation is modeled using a single Gaussian
  - Each pixel has its own Gaussian model

$$p^L(I_s) = N(I_s | \mu_s, \Gamma_s)$$

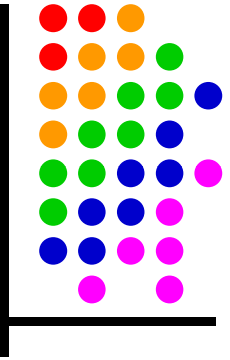
Foreground Modeling



- First classify all pixels into 3 classes using per-pixel background color model:
  - Definitely background, if $p^L(s) > T_B$
  - Definitely foreground, if $p^L(s) < T_F$
  - Uncertainty region, otherwise
- The global foreground color model is then learned from pixels labels as definitely background
  - Model all foreground colors using a mixture of K Gaussians:

$$p^G\left(I_s \middle| L_s = F\right) = \sum_{k=1}^{K} \omega_k^F N\left(I_s \middle| \mu_k^F, \Gamma_k^F\right)$$

Compute Color Term



- **Per-pixel color term is calculated using the foreground & background color models:**
  - **The energy (cost) of assigning pixel p to foreground is based on the global foreground model**

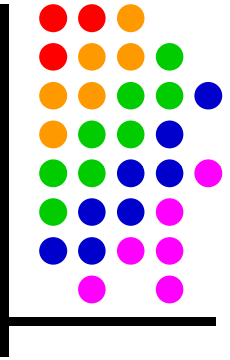
$$E_{color}(L_s = F) = -\log p^G(I_s | L_s = F)$$

  - **The cost of assigning pixel p to background is computed using an adaptive mixture of local & global background models**

$$E_{color}(L_s = B) = -\log\left(\alpha \cdot p^G(I_s | L_s = B) + (1-\alpha) \cdot p^L(I_s)\right)$$

    - ***Weight α depends on the difference between the global foreground & background models***

Compute Contrast Terms

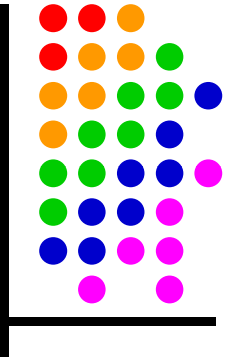


- **Per-pixel-pair contrast term is calculated using pixel color difference:**
  - **The term only penalizes a pixel pair if it has different labels for the 2 pixels**
  - **The amount of penalty depends on the color difference between the 2 pixels**

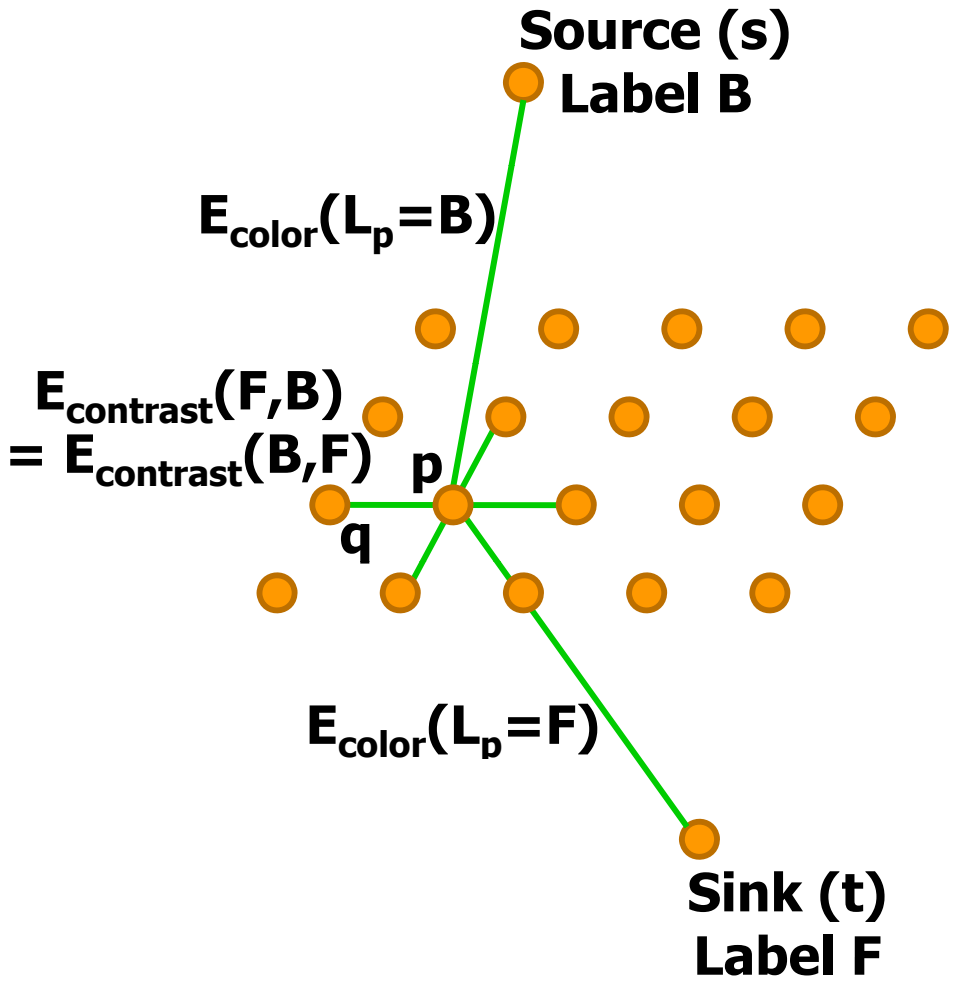
$$E_{contrast}(L_s, L_t) = |L_s - L_t| \cdot e^{\frac{\|I_s - I_t\|^2}{-2\sigma^2}}$$

    - ***Give low penalties to high contrast pixel pairs***
- **The background contrast is attenuated to remove artifacts caused by sharp edges in background**

Find Global Optimal Labeling using Graph Cut

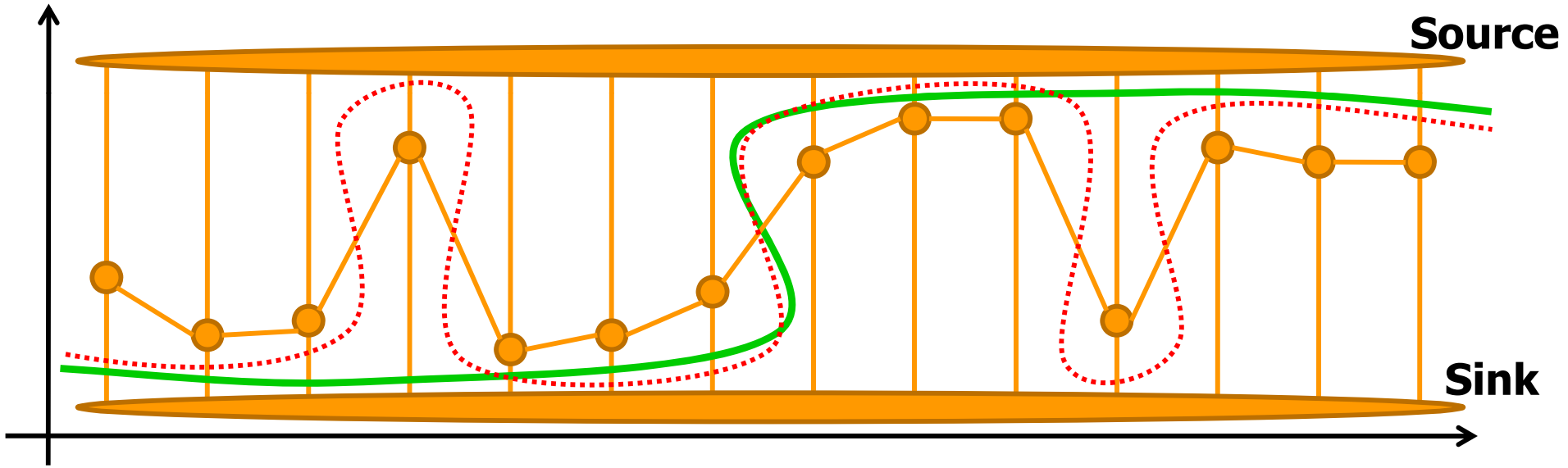


- Construct a 2D lattice graph G = (V,E):
  - One node for each pixel
  - 2 special nodes: source & sink represent 2 labels
- A pixel node connects to:
  - Source & sink, with capacities set based on the data term
  - Its 4 neighbors, with capacities set based on the connection term



The Min-cuts

- **Any cut separating the source & sink corresponds to an binary labeling assignment**
  - **The total capacities of edges being cut equals to the cost of the label assignment**
  - **The min-cut corresponds to the global optimal solution with minimum energy**



Matte Extraction Problem

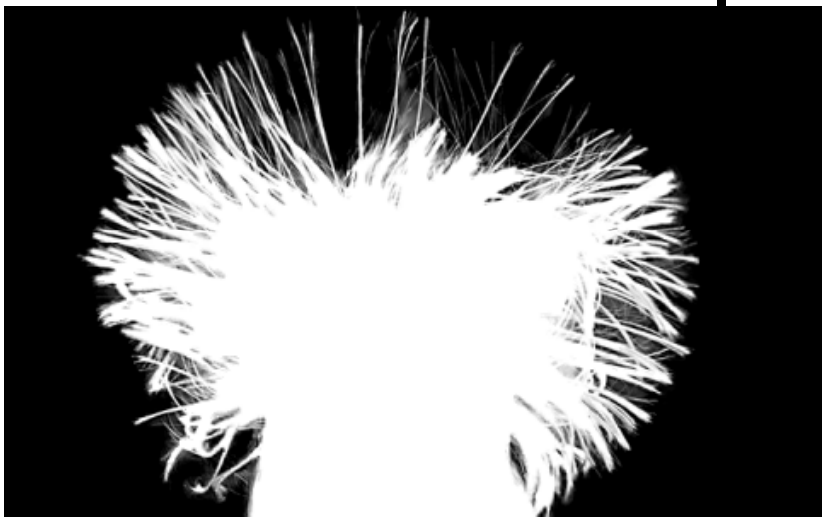


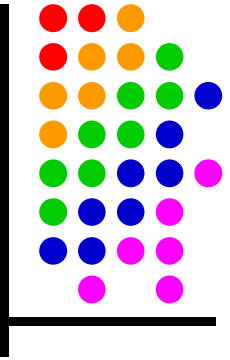
Image from Wang & Cohen 2007

- Given an image (I), extract the alpha channel (α), foreground object color (F), & background color (B)
  - The inverse problem of alpha compositing

$$I = \alpha F + (1-\alpha)B$$

- Application:
  - High quality background substitution

Existing Techniques



- Two known backgrounds:
  - Matting by triangulation [Smith & Blinn 1996]
- Unknown background:
  - Bayesian matting [Chuang, et al. 2001]
  - Poisson matting [Sun, et al. 2004]
  - Close form matting [Levin, et al. 2006]

Matting by Triangulation

- **The matte can be easily extracted if the images of the object in front of 2 known backgrounds are available:**

$$\begin{cases} I_1 = \alpha F + (1-\alpha)B_1 \\ I_2 = \alpha F + (1-\alpha)B_2 \end{cases}$$

$$\Rightarrow \alpha = 1 - \frac{I_1 - I_2}{B_1 - B_2}$$

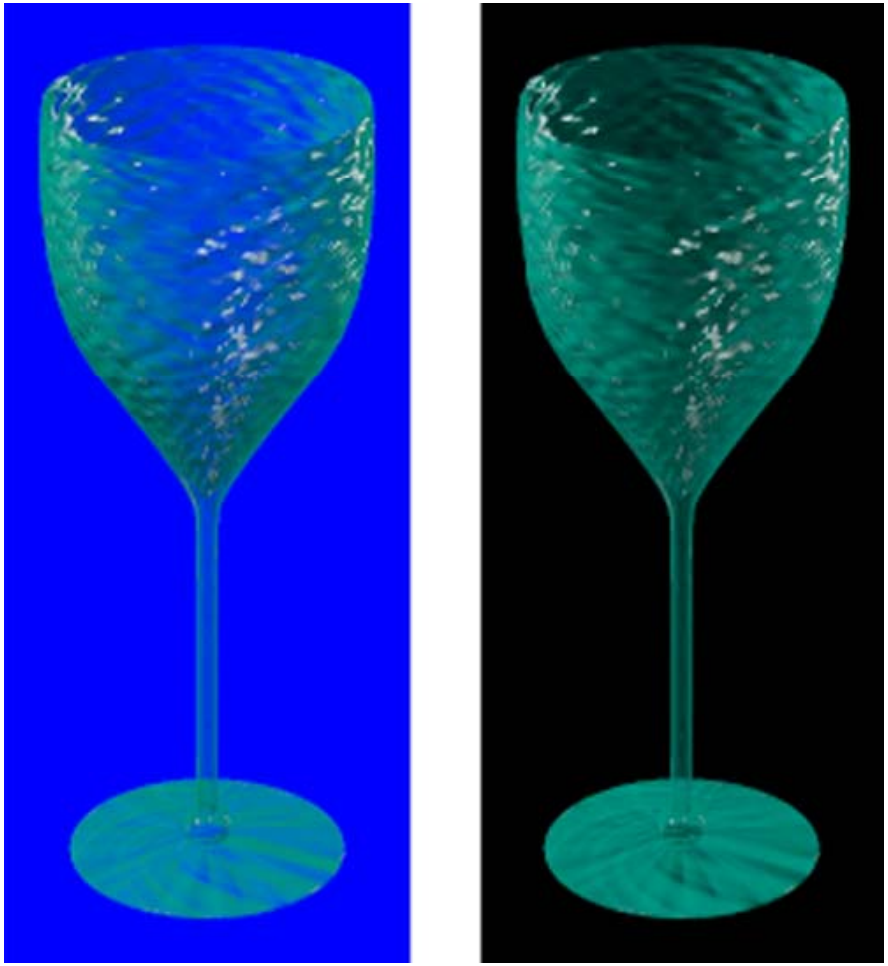
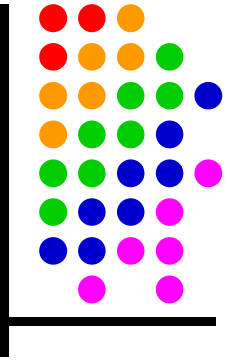


Image from Smith & Blinn 1996

Poisson Matting

- A 2-step approach:
  - Compute an approximate gradient field of matte
  - Solve the matte using Poisson's equation
- Require foreground & background labeled using trimap
  - Foreground in white
  - Background in black
  - Unknown in grey

Compute Gradient Field



- **Take the partial derivatives on both sides of the matting equation:**

$$I = \alpha(F - B) + B \Rightarrow$$

$$\nabla I = (F - B)\nabla\alpha + \alpha\nabla F + (1 - \alpha)\nabla B$$

$$\text{where } \nabla f = \text{Gradient}(f) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

- **Assume both F & B are smooth**

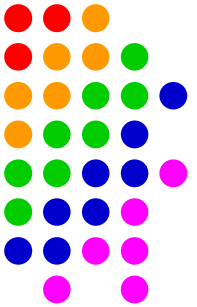
$$\nabla F \approx \nabla B \approx 0 \Rightarrow \nabla\alpha \approx \frac{\nabla I}{(F - B)} \Rightarrow \nabla \cdot \nabla\alpha \approx \nabla \cdot \left(\frac{\nabla I}{(F - B)}\right)$$

$$\text{where } \nabla \cdot \mathbf{f} = \text{Divergence}(\mathbf{f}) = \frac{\partial \mathbf{f}_x}{\partial x} + \frac{\partial \mathbf{f}_y}{\partial y}$$

Approximate Foreground & Background

- The accurate F & B for pixels in grey area are unknown
- Based on smoothness assumption, neighboring pixels' colors are used
  - Set F to the color of the closest pixel in white
  - Set B to the color of the closest pixel in black
  - Can be obtained using morphology operation

Approximate Gradient of Matte

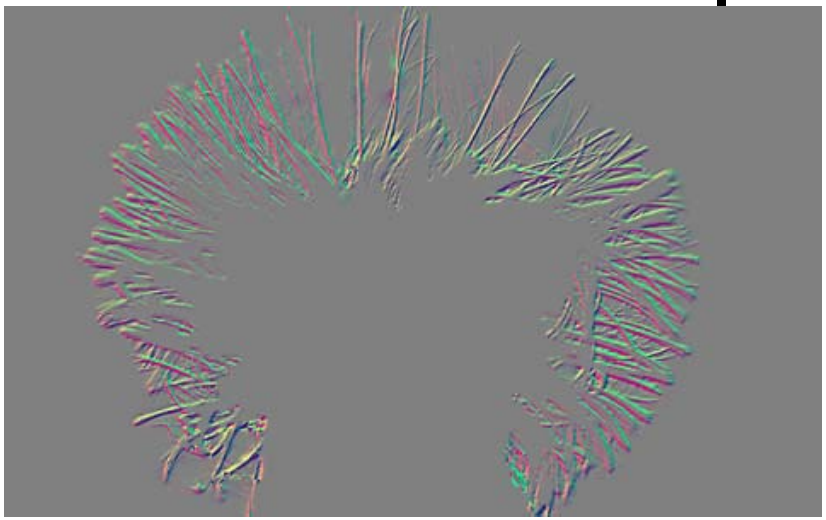


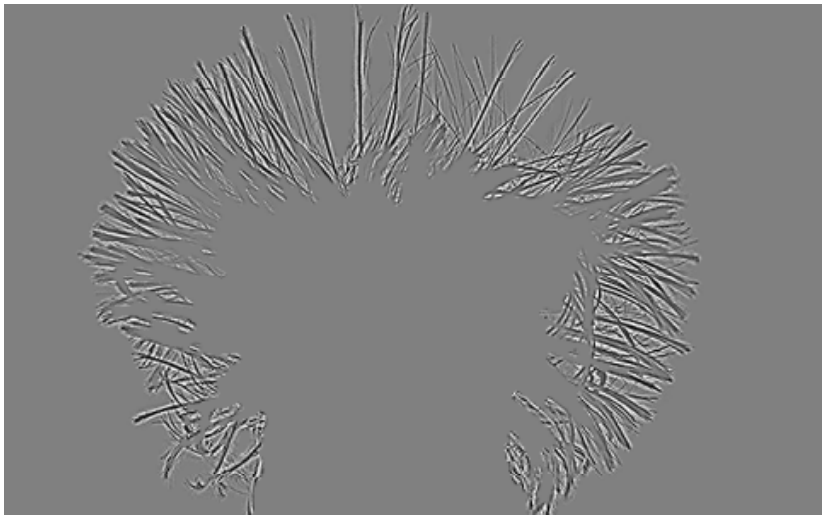
- **Compute an approximate gradient of matte using estimated F & B:**

$$\nabla\alpha \approx \frac{\nabla I}{(F - B)}$$

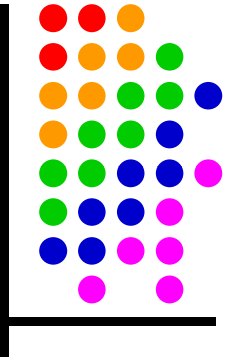
- **Calculate the divergence of the approximate gradient of matte:**

$$\nabla \cdot \nabla\alpha = \frac{\partial(\nabla\alpha)_x}{\partial x} + \frac{\partial(\nabla\alpha)_y}{\partial y}$$





Solve Poisson Equation



- **A partial differential equation of form: Δf = b**
  - **Δ is the Laplace operator (or Laplacian), defined as the divergence of the gradient field of a function**

$$\Delta f = \nabla^2 f = \nabla \cdot \nabla f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

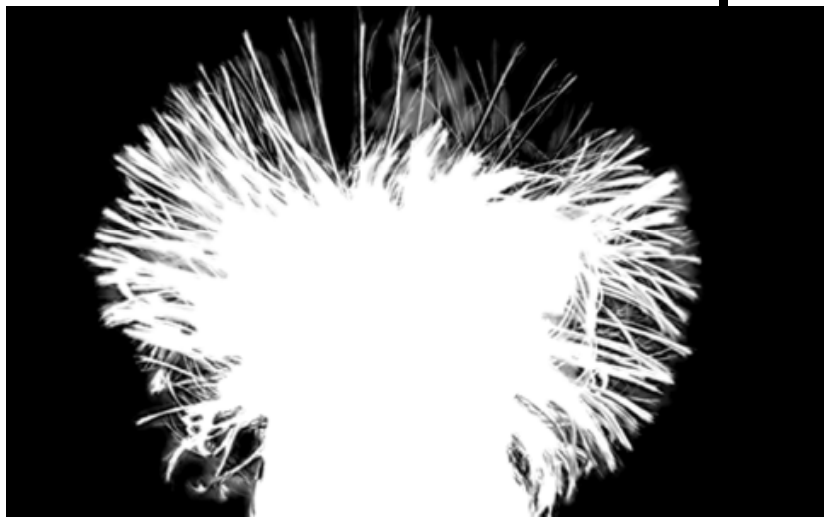
- **Can be solved using Jacobi or Gauss–Seidel method**
  - **Jacobi method is simple but converge slowly**

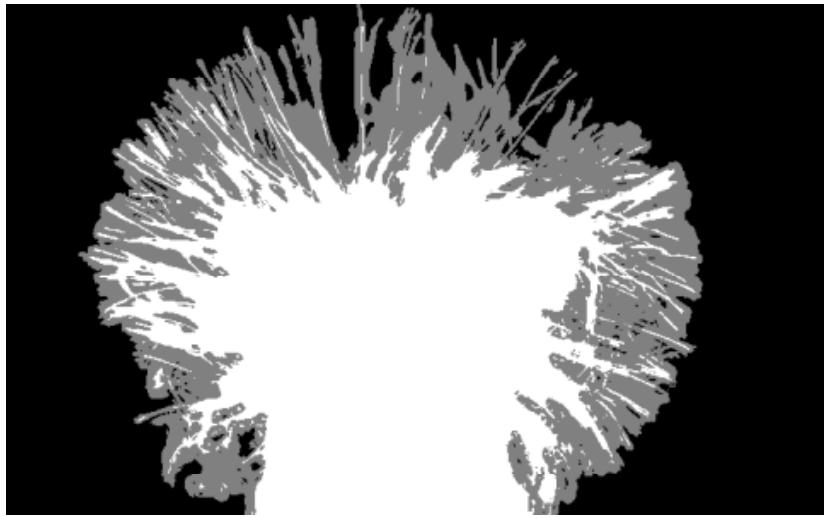
$$\nabla^2 f = b \xrightarrow{\text{discretize}} f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - f_{i,j} \times 4 = b_{i,j}$$

$$f_{i,j}^{0} = 0,\ f_{i,j}^{k+1} = \frac{f_{i+1,j}^{k} + f_{i-1,j}^{k} + f_{i,j+1}^{k} + f_{i,j-1}^{k} - b_{i,j}}{4}$$

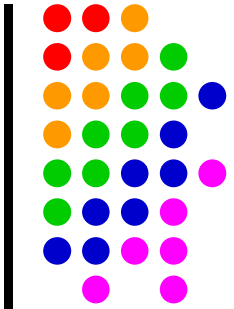
Update Trimap and Repeat

- Solving the Poisson equation gives an alpha matte estimation
- A more accurate trimap can be generated using multilevel thresholding
  - Allow better foreground & background estimations, which help to obtain better matte
- The process is repeated until it converges





Background Substitution Results



Without matting

With matting

Questions?

Thank you for your attention!