Homework 1
KNN Matting

IMVFX Fall
October 12, 2023

Our Goal

Original

Trimap

Trimap

- 255 for the foreground, and 0 the for background.
- Given a trimap, we need to solve for the unknown areas.
- Then we can split the foreground from the background.

Original

Trimap

Quick Overview

1. Find KNN and compute the affinity matrix A accordingly.
2. Solve for the linear system to get the alpha matte.
   - Shown in the course slide. (In IMVFX_2_Matting_F23_s.pdf, P65~P70)
   - Check the original paper for more detail.
3. Attach your result onto another background.

Find KNN and construct the affinity matrix A

Evaluation for KNN matting

- Feature vector *X*, e.g.

$$X(i) = (\cos(h), \sin(h), s, v, x, y)_i$$

For, HSV color space with the spatial coordinate

$$X(i) = (R, G, B)_i$$

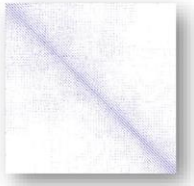
For, RGB color space

- Kernel function $k(i, j)$

$$k(i,j) = 1 - \frac{\|X(i) - X(j)\|}{C}$$

C is the least upper bound of $|X(i)-X(j)|$ to make $k(i, j) \in [0, 1]$.

$$A = \begin{bmatrix} k(1,1) & \cdots & k(1,n) \\ \vdots & \ddots & \vdots \\ k(n,1) & \cdots & k(n,n) \end{bmatrix}$$



A of KNN matting

69

Solve for the linear system

Evaluation for KNN matting (cont.)

- We want α of mark-up pixels fitting for users assignment.

$$\boldsymbol{M\alpha \approx v}$$

$\boldsymbol{\alpha}$: the vector of all α
$\boldsymbol{M}$: diag($\boldsymbol{m}$), indicates whether a pixel is marked-up.
$\boldsymbol{v}$: *indicates the assigned alpha*

- Besides, $\boldsymbol{D\alpha \approx A\alpha}$

$\boldsymbol{D}$: diag(D_i), an NxN diagonal matrix
$\boldsymbol{A}$: NxN affinity matrix composed of $k(i,j)$

- Our objective function becomes

$$Obj = (\boldsymbol{D - A})\boldsymbol{\alpha} + \lambda(\boldsymbol{M\alpha - v}) = (\boldsymbol{L} + \lambda\boldsymbol{M})\boldsymbol{\alpha} - \lambda\boldsymbol{v}$$

$\boldsymbol{L}$= ($\boldsymbol{D}$-$\boldsymbol{A}$)

λ: a weight controlling user's confidence on the markups

- The optimal $\boldsymbol{\alpha}$ can be estimated for the linear system.

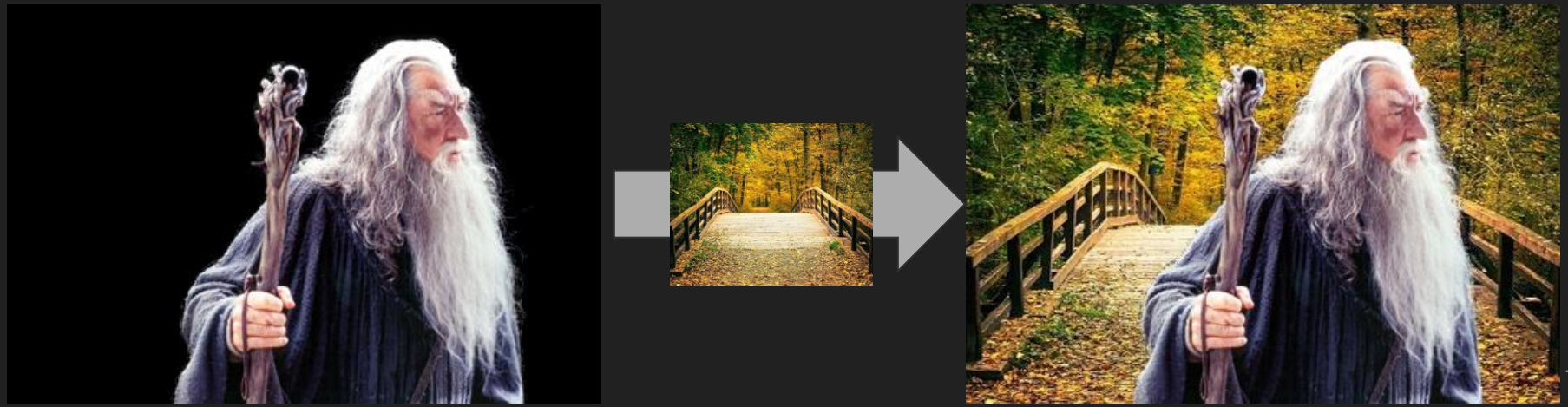
This work can simultaneously extract multiple image layers, but we focus on fg and bg here.

70

Composition

- After getting F, B and α, we can simply change the background and composite a new image using the formula shown below.

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



Requirements

1. Find KNN and construct the affinity matrix A.
   a. **scikit-learn** is allowed for finding KNN.
   b. Try to play with the number of K and see how it affects the result.
2. Solve for the linear system.
   a. **scipy** and **numpy** are allowed for matrix computation.
   b. **To solve linear system you can use numpy.linalg.solve, numpy.linalg.lstsq, scipy.linalg.solve, scipy.linalg.lstsq**, and so on.
   c. You can view the objective function as an Ax=b linear equation and solve it with $x = A^{-1}b$.
3. Composite some new images.
   a. Pick a landscape photo on your own and try to attach your result onto it.
4. Want more points?
   a. Solving for a large linear system is extremely time-comsuming, so find some other methods to speed it up and make a comparison on how long do these methods take.
   b. Try different ways of representing feature vectors.
   c. Work on more images (http://www.alphamatting.com/datasets.php)
   d. Implement KNN by yourself.
   e. Compare with other matting methods.
   f. Do anything not mentioned above to amaze us!

Score

- Finding KNN and constructing the affinity matrix A (30%)
- Solving for the linear system. (25%)
- Composition (9%)
- Quality of the result images (6%)
  - 2 points will be given if it looks natural.
  - 1 point will be given if there are some obvious artifacts.



- Report (15%)
  - Explanation for all parts of your code.
  - Description on what you have done in experiments and show the results. (e.g. Zoom in to see how it looks.)
  - Description on what you have done for the bonus.
- Bonus (at most 15%)
- 85 points will be given if you reach all the basic requirements. For the bonus, your score will be judged based on how much you outperform your classmates.

Reminder

- Please do not upload code that you have found on the internet.
- If you refer any code from GitHub or other open-source projects, you have to properly cite the source and comment on those code. Otherwise, you will get a penalty of 20 points or more.
- You should work on all the given images.
- It takes much time to process an image, so you'd better to start early.
- Feel free to modify codes provided by the TA.

Submission

- Every submission should consist of the followings:
  - Your code
  - A README describing how to run your code (If you use languages other than Python)
  - Report (.pdf)
- Please put everything into <your student ID>_<your name>_hw1.zip and submit through E3.