# PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing

Ajay Shrihari - 20171097

Chaitanya Kharyal - 20171208

Rahul Sajnani - 20171056

Anoushka Vyas - 20171057

TA- Meher Shashwat Nigam

GitHub URL- https://github.com/Digital-Image-Processing-IIITH/project-strawhat

# Motivation

# Problem Statement

The main objective of this project is to present interactive image editing tools using a new randomized algorithm for quickly finding approximate nearest neighbor matches between image patches. This algorithm forms the basis for a variety of tools – that can be used together in the context of a high-level image editing application.

One more feature is additional intuitive constraints on the synthesis process that offer the user a level of control unavailable in any other methods. Previous methods in this field are generally very slow and could not be ported into applications where user input was allowed. This method aims to allow for the use of tools in a real time scenario.

# Method

# Algorithm

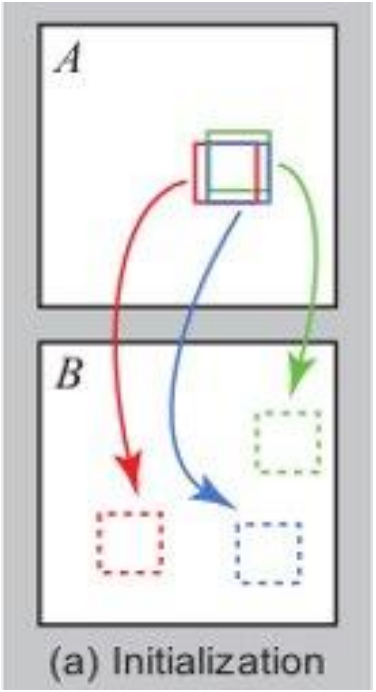
## Notations

- Here, ***A[x,y]*** and ***B[x,y]*** determine the patch with center at ***(x,y)*** in image ***A*** and image ***B***.
- ***f(x, y)*** is the distance between the location of the patch in image ***B*** that is closest to the patch at the location ***(x, y)*** in image ***A***.
- ***D(A[x, y], B[f(x,y)])*** determines the distance (can be mean *L1* cost) between the patches of image ***A*** and image ***B***.

# Algorithm

## Initialization

- The algorithm initializes the patch locations by sampling from a uniform random distribution of locations from image ***B*** and assigns the initial patch matching error.
- A few early iterations of the algorithm using a random initialization is performed and then merged with the upsampled initialization only at patches where ***D*** is smaller, and then perform the remaining iterations.

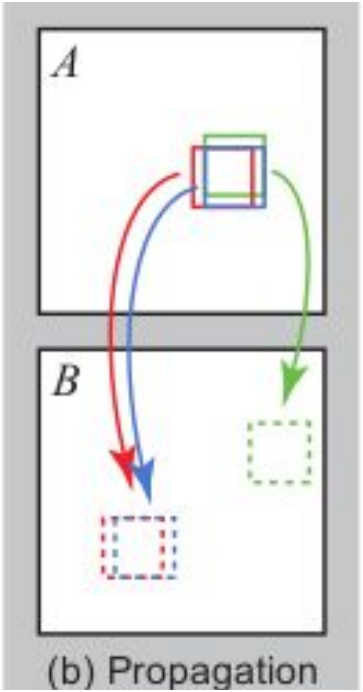


# Algorithm

## Propagation

- In this step, we look at the locations of ***f(x-1, y)*** and ***f(x, y - 1)*** (in even iteration), and ***f(x + 1, y)*** and ***f(x, y + 1)*** (in odd iteration) for closest patch to ***(x,y)***.
- The idea here is that the patch for the neighbors of ***(x, y)*** should be close to the patch at ***(x,y)***.
- The final distance or the new value for ***f(x, y)*** is:

***D = min{D(A[x, y],B[f(x,y)]), D(A[x, y],B[f(x - 1,y)]), D(A[x, y],B[f(x,y - 1)])}.***

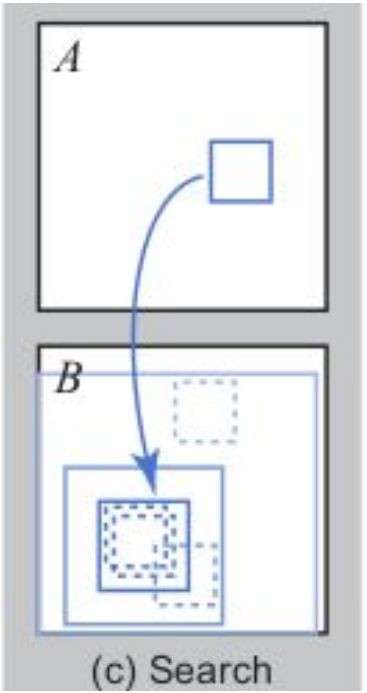


# Algorithm

## Random Search

To prevent non-optimal convergence random search is performed.This involves testing a set of offset candidates at an exponentially decreasing distance.

- If $v_0 = f(x, y)$, $v_{new} = v_0 + w\,\alpha^i R_i$
- $v_{new}$ gives us the new offset. $R_i$ is uniform random in $[-1, 1] \times [-1, 1]$, $w$ is a large maximum search “radius” and $\alpha = ½$.
- We search for patches $i = 0,1,2....$ until $w\,\alpha^i$ is smaller than $1$ pixel.




Higher
level intuitive
understanding
of the topic
Maths
imgflip.com

# Why does this work?

- Given an image with ***M*** pixels the probability of obtaining a correct pixel location patch from image ***B*** is $\frac{1}{M}$.
- Probability of all pixels with incorrect random guess is $\left(1-\frac{1}{M}\right)^{M} \approx 0.37$ for large ***M***.
- Let's consider a small patch region of ***C*** pixels where the pixel is close to its actual position. This makes the probability of obtaining a correct pixel location from patch ***B*** equal to $\frac{C}{M}$.
- Probability of one or more pixels to be assigned the correct location is $1-\left(1-\frac{C}{M}\right)^{M}$.

For ***C = 20*** and ***M = 50000*** the probability is approximately is ***0.999*** (***500*** pixels in image).

Since the above probability is so high after random initialization and propagation we have a large number of pixels with their nearest patch initialized very near to the actual location.

# Application- Hole Filling/Object Removal

# Bidirectional Distance Measure

***S*** - Source Image, ***T***- Target Image

$$d_{BDS}(S,T) = \overbrace{\frac{1}{N_S}\sum_{s \subset S}\min_{t \subset T} D(s,t)}^{d_{complete}(S,T)} + \overbrace{\frac{1}{N_T}\sum_{t \subset T}\min_{s \subset S} D(t,s)}^{d_{cohere}(S,T)}$$

1. The completeness term ensures that the output image contains as much visual information from the input as possible.
2. The coherence term ensures that the output is coherent w.r.t. the input and that new visual structures (artifacts) are penalized.
3. The distance measure is defined simply as the sum of the average distance of all patches in ***S*** to their most similar (nearest-neighbor) patches in ***T*** and vice versa.
4. ***D*** is the SSD (sum of squared differences) of patch pixel values in ***L*a*b**** color space.

# Algorithm

***S*** - Filled image, ***T***- Hole image

- Given an the output image, $T_0$, this distance is iteratively minimized by an EM-like algorithm (omitting the completeness term).
- In the E step of each iteration ***i,*** the NN-fields are computed from ***S*** and $T_i$ and “patch-voting” is performed to accumulate the pixel colors of each overlapping neighbor patch.
- In the M step, all the color “votes” are averaged to generate a new image $T_{i+1}$.
- ***T*** and ***S*** are gradually upsampled to finer resolutions, followed by more EM iterations, until the final fine resolution is obtained to avoid getting stuck in bad local minima.

# Results - PatchMatch

“Only if I could patch my sleep schedule in the way the source image patches the target image”

# Test Image 1

Source Image

Target Image

Reconstructed Image

# Test Image 2

Source Image

Target Image

Reconstructed Image

# Test Image 3

Source Image

Target Image

Reconstructed Image

# Middlebury Stereo

Source Image

Target Image

Reconstructed Image

# Middlebury Stereo

Source Image

Target Image

Reconstructed Image

# Caltech 256


Source Image

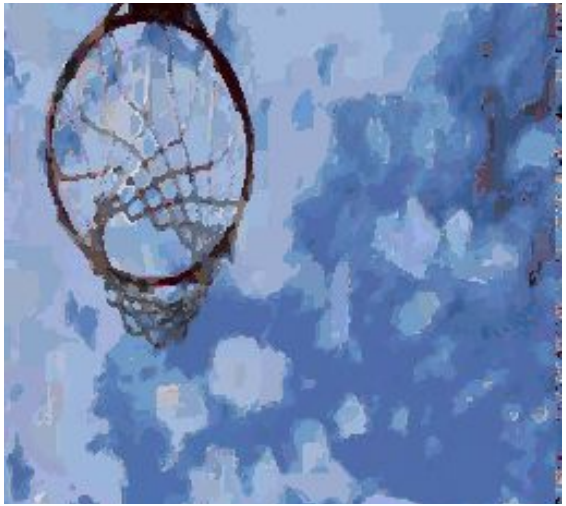

Target Image


Reconstructed Image

# Caltech 256

Source Image

Target Image



Reconstructed Image

# Frames of Running Video

Source Image

Target Image

Reconstructed Image

# Frames of Running Video

Source Image

Target Image

Reconstructed Image

# Our image

Source Image

Target Image

Reconstructed Image

# Results- Object Removal / Hole Filling

# Our image

Source Image

Hole Image

Reconstructed Image

# Our image

Source Image

Hole Image

Reconstructed Image

# Our image

Source Image

Hole Image

Reconstructed Image

# Experiments

# Experiment 1

Effect of varying loss function for reconstruction

# Results

Source image

Target image

Max L1 (error = 13.73)

Mean L1 (error = 12.30)

L2 (error = 12.13)

# Results

Source image

Target image

Max L1 (error = 4.87)

Mean L1 (error = 4.25)

L2 (error = 4.23)

# Experiment 2

Why LAB colorspace?

# Results

Source image

Target image

LAB patchmatch (error = 6.99)

RGB patchmatch (error = 16.54)

# Results

Source image

Target image

LAB patchmatch (error = 12.90)

RGB patchmatch (error = 28.93)

# Experiment 3

Varying search size in random search

# Results

Source image


Target image


Alpha = 0.1 (error = 8.80)

Alpha = 0.3 (error = 8.49)

Alpha = 0.5 (error = 8.57)

Alpha = 0.7 (error = 8.91)

# Results

Source image

Target image

Alpha = 0.1 (error = 15.12)

Alpha = 0.3 (error = 14.77)

Alpha = 0.5 (error = 15.20)

Alpha = 0.7 (error = 14.89)

# Experiment 4

Varying patch size

# Results

Source image

Target image

Size = 3 (error = 13.20)

Size = 5 (error = 15.77)

Size = 7 (error =17.04)

Size = 9 (error =18.35)

# Results

Source image

Target image

Size = 3  (error = 14.10)

Size = 5  (error = 17.96)

Size = 7  (error = 20.42)

Size = 9  (error = 23.29)

# Limitations

# Limitations

- The worst case for **exact** matching occurs when we have a highly repetitive image. In this case, PatchMatch algorithm gives suboptimal patch locations from the source image.
- PatchMatch entirely depends on reducing the patch wise photometric error. This is prone to lightning effects and can perform bad in such cases.
- Object removal tasks displayed in the paper are time consuming as they require a lot of iterations to complete.

# Division of Work

- **Normal PatchMatch and result generation:** Chaitanya Kharyal, Ajay Shrihari
- **Ablation study in Normal PatchMatch:** Rahul Sajnani
- **Hole Filling:** Anoushka Vyas, Rahul Sajnani
- **GUI for Hole Filling:** Anoushka Vyas, Rahul Sajnani, Chaitanya Kharyal, Ajay Shrihari