Image Inpainting using Wexler and PatchMatch algorithm

Anass Denguir
Luca Noel
Trong Quy Phan
Jieren Wu

Table of Contents

1. Introduction
2. GUI
3. Inpainting
   - 3.1 Main algorithm
   - 3.2 Reconstruction with Wexler
   - 3.3 Reconstruction enhancements
4. PatchMatch
   - 4.1 Step of algorithm
   - 4.2 Results of algorithm
5. Conclusion

Table of Contents

1. Introduction

2. GUI

3. Inpainting
   - 3.1 Main algorithm
   - 3.2 Reconstruction with Wexler
   - 3.3 Reconstruction enhancements

4. PatchMatch
   - 4.1 Step of algorithm
   - 4.2 Results of algorithm

5. Conclusion

Introduction

The purpose of this project is to

1. Define a mask of arbitrary shape by user. Then use this mask to eliminate any part of the picture.
2. Having an image and a missing part of it, we can then use Wexler's method to fill in the missing region with reference to surrounding background.
3. Use PatchMatch to speed up the process.

Table of Contents

1. Introduction

2. GUI

3. Inpainting
   - 3.1 Main algorithm
   - 3.2 Reconstruction with Wexler
   - 3.3 Reconstruction enhancements

4. PatchMatch
   - 4.1 Step of algorithm
   - 4.2 Results of algorithm

5. Conclusion

GUI

Using the Matlab function *ginput*, we are able to define any shape by mouse. To generate the mask, an image of certain size is used as a raw material. And we use *ginput* to select part of it.

After selecting the part out of the original image, we have a shape now.

Now we can use the original image to subtract this part then reverse the result. We then get a mask and its binary value is stored in a parameter. This is the user-define mask. Now by multiplying the mask and images, we can eliminate the sections we want out of the pictures.

Table of Contents

1. Introduction

2. GUI

3. Inpainting
   - 3.1 Main algorithm
   - 3.2 Reconstruction with Wexler
   - 3.3 Reconstruction enhancements

4. PatchMatch
   - 4.1 Step of algorithm
   - 4.2 Results of algorithm

5. Conclusion

Main algorithm

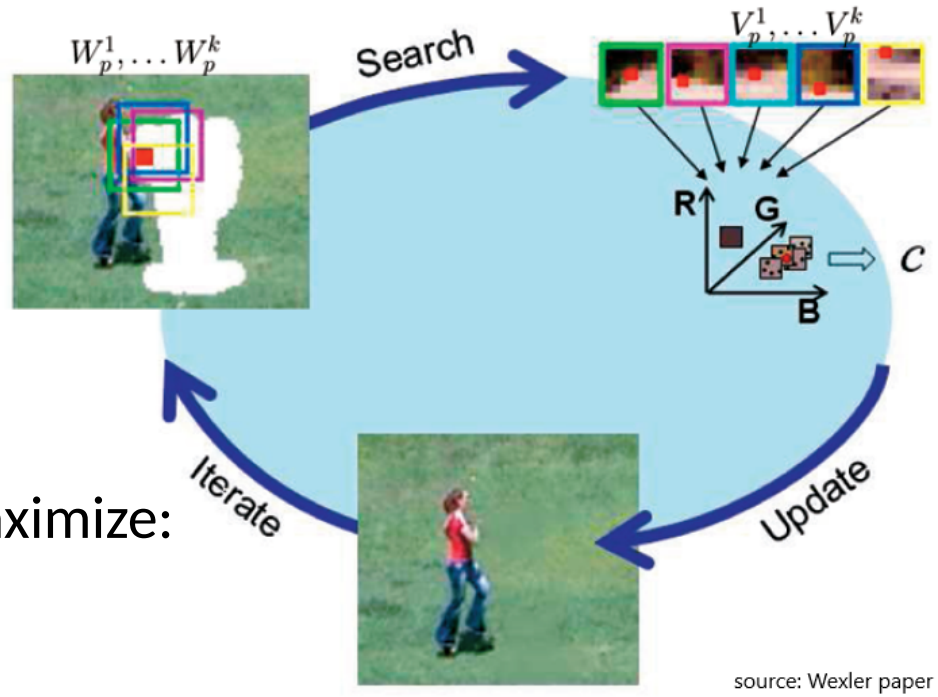
Algorithm 1 Inpainting algorithm

Input: image, binary mask

Output: inpainted image

```
Downscale image to the coarsest level L
Initial reconstruction with Onion-peel at level L
for level of pyramid k = L to 2 do
    while Convergence and max iterations not reached do
        Reconstruction
    end while
    Upscale for next level k
end for
For last level k = 1, apply FinalReconstruction
```

Determine the appropriate color c for a pixel p_i



source: Wexler paper

1. Find windows W_p^i
2. Find windows V_p^i that maximize:

   $sim(W_p, V_q) = e^{-\frac{d(W_p, V_q)}{2\sigma^2}}$
3. Compute $c = \frac{\sum_i \omega_p^i c^i}{\sum_i \omega_p^i}$

   with $\omega_p^i = \alpha_p^i sim(W_p^i, V^i)$

Hole reconstruction

Rather than processing pixel by pixel and finding the set $\{W_p^i\}_{i=1}^N$ and $\{V_p^i\}_{i=1}^N$ for every pixel of the hole, another approach was used to reduce the number of search operations:

1. Slide a window W'.
2. If the windows W' intersect the hole H then find V' that maximizes $sim(W', V')$.
3. The window V' will only contribute in the vote of the pixels contained in W' and H.
4. Redo step 1 until end of the image.

Pyramidal reconstruction: Downscaling

At first, we downscale the occluded image by *L* levels:

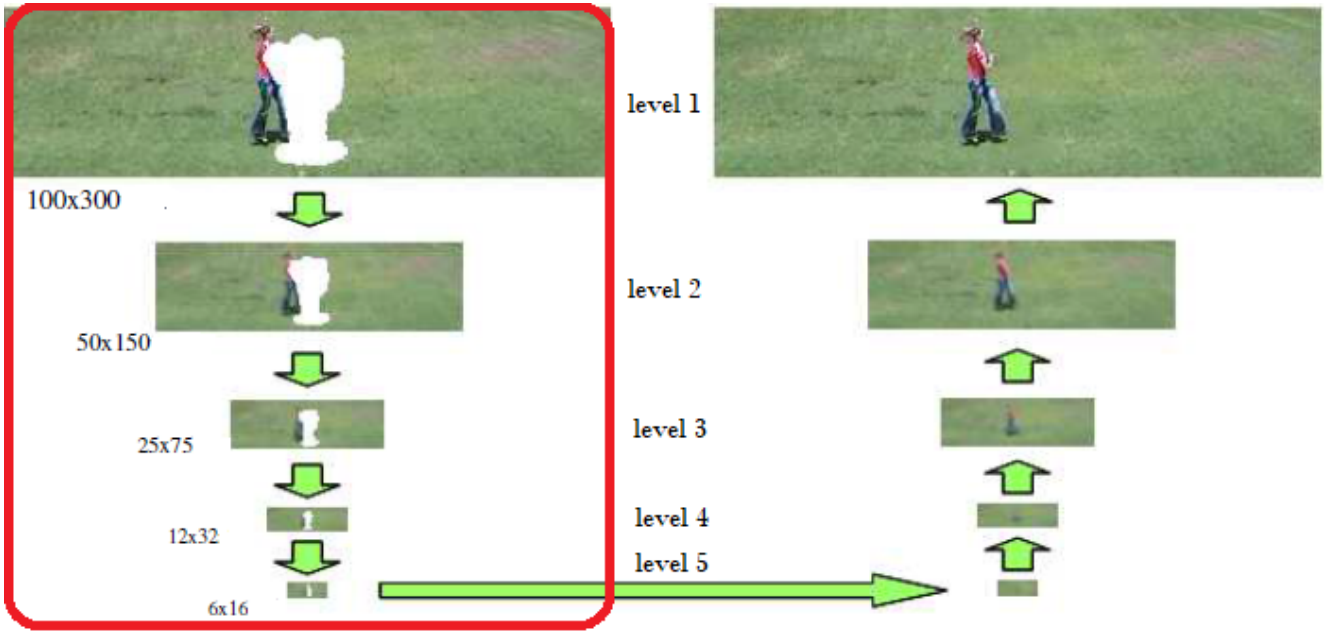


Figure: Pyramidal resolution reconstruction (source: Wexler paper)

Initial Reconstruction: Onion-peel

At the coarsest level L, we fill the hole H layer by layer:

$$W_p^* = \{W_p^i \mid (W_p^i \cap \partial H_k \neq \varnothing) \wedge (W_p^i \cap \partial H_{k+1} = \varnothing)\}$$

For the kth layer, We choose windows W^* containing at least one value k in the distance transform AND no value that is greater than k

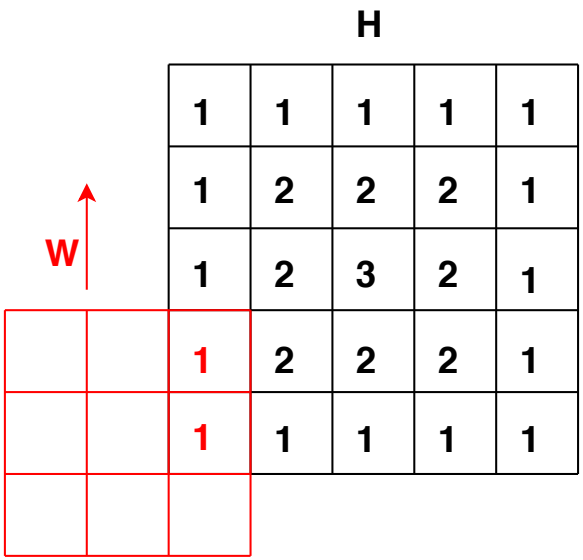


Figure: Distance transform of image

Pyramidal reconstruction: Upscaling

For each pyramid level $l > 1$:

1. Reconstruct downscaled hole H_l as shown before
2. $H_{l-1} \leftarrow upscale(H_l)$

At the finest pyramid level $l = 1$: $c = \arg\min_{q \in S} d(W_p, V_q)$

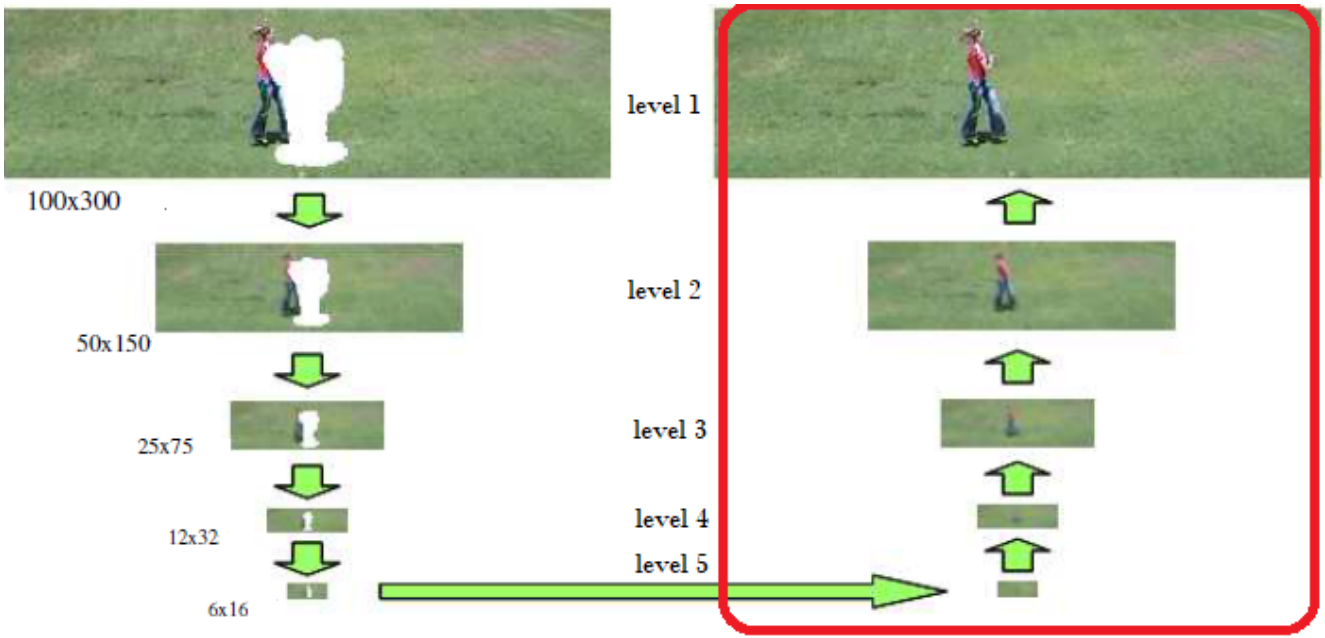


Figure: Pyramidal resolution reconstruction (source: Wexler paper)

Results of reconstruction

(a) Original picture

(b) Our result

Figure: Image completion using PatchMatch (590x591 pixels)

Results of reconstruction

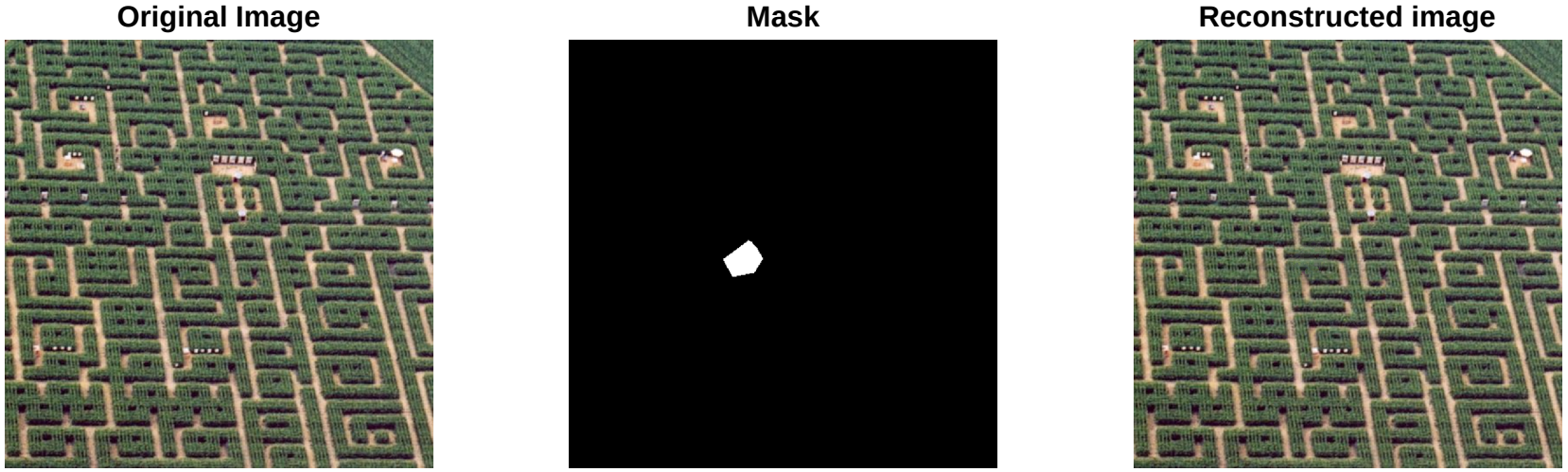


Figure: Completion of a structured image

Table of Contents

1. Introduction

2. GUI

3. Inpainting
   - 3.1 Main algorithm
   - 3.2 Reconstruction with Wexler
   - 3.3 Reconstruction enhancements

4. PatchMatch
   - 4.1 Step of algorithm
   - 4.2 Results of algorithm

5. Conclusion

Step of PatchMatch

The *nearest-neighbor field* (NNF) is defined as the function $f{:}A \mapsto R^2$ of offset

$$f(x, y) = v$$

Where v is an offset in image B. In the program,

$$(x, y) \mapsto (x, y) + v$$

Step of PatchMatch

The aim of the algorithm is to create a correspondence table between an source image A and destination image B.

1. Initialization of random coordinates in NNF table (same dimension as A)
2. Propagation process: checking the corresponding coordinates in B of pixel (i,j) and its neighbor (i-1,j), (i,j-1). Depending on the similarity of each patch and their corresponding patch in B, the new value of (i,j) is updated with the best matching offset.
3. Search of a random patch which minimize euclidean distance in shrinking window in B

Step of PatchMatch

Propagation:

$$f(x, y) = argmin[D((f(x, y)), D(f(x, y-1)), D(f(x-1, y))]$$

In reverse direction for even iterations.

Random search:

$$u_i = v_0 + w * \alpha^i * R_i$$

u_i : new offset, v_0 : offset after propagation, w: largest dimension of B, α is 0.5 and R_i is a random 2D vector with element between -1 and 1.

Step of PatchMatch

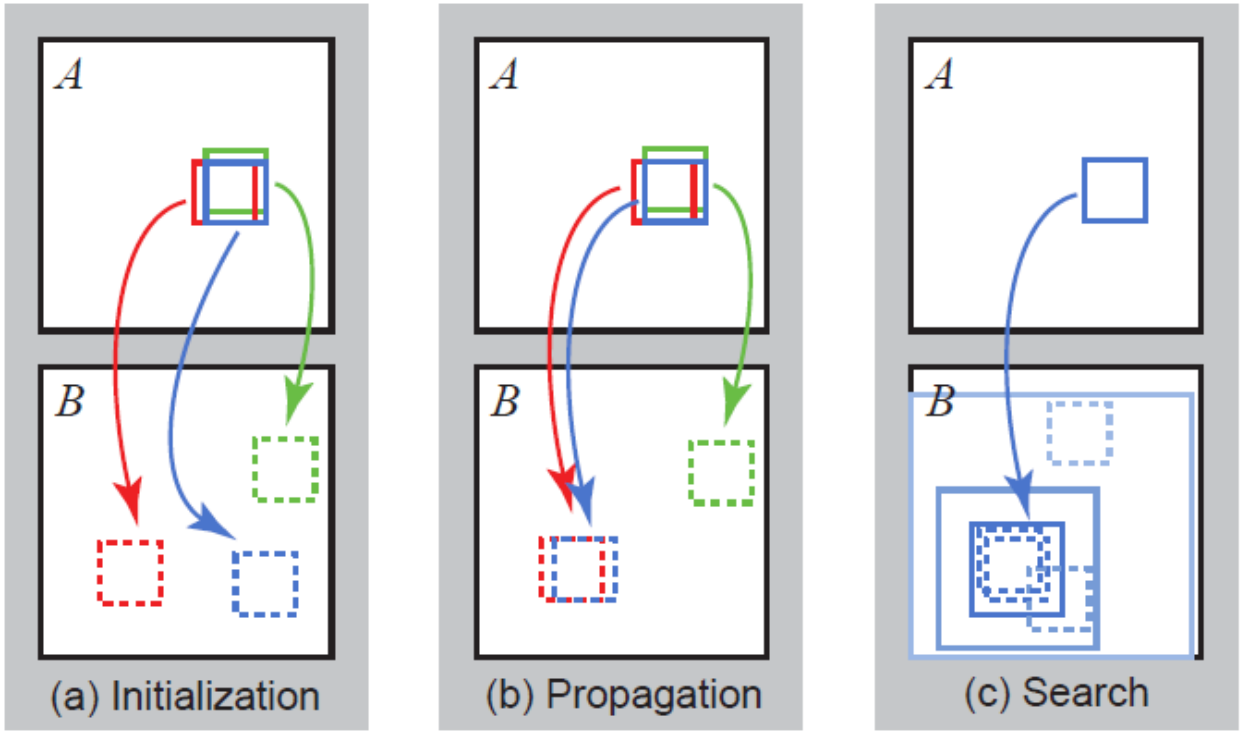


Figure: Step of patchmatch algorithm

Results of algorithm

Figure: Results of PatchMatch algorithm

Table of Contents

1. Introduction

2. GUI

3. Inpainting
   - 3.1 Main algorithm
   - 3.2 Reconstruction with Wexler
   - 3.3 Reconstruction enhancements

4. PatchMatch
   - 4.1 Step of algorithm
   - 4.2 Results of algorithm

5. Conclusion

Conclusion

- Inpainting algorithm implemented using a patch based multi-scaled reconstruction method
- Brute force research of coherent patches computationally heavy
- PatchMatch algorithm implemented to speed up patch search