EECS 332 Digital Image Analysis

Texture Synthesis and Dynamic Programming

Ying Wu

Dept. Electrical Engineering and Computer Science.
Northwestern University
Evanston, IL 60208

http://www.ece.northwestern.edu/~yingwu
yingwu@ece.northwestern.edu

What are textures?



- Some kind of repetitive visual patterns
- Easy to recognize, but hard to define and describe

2

Three topics in textures

- Texture modeling

  how can we model textures mathematically?

  - How to represent each of this repetitive visual patterns (texton)?
  - How to describe their spatial layout?

- Texture analysis (such as shape from texture)

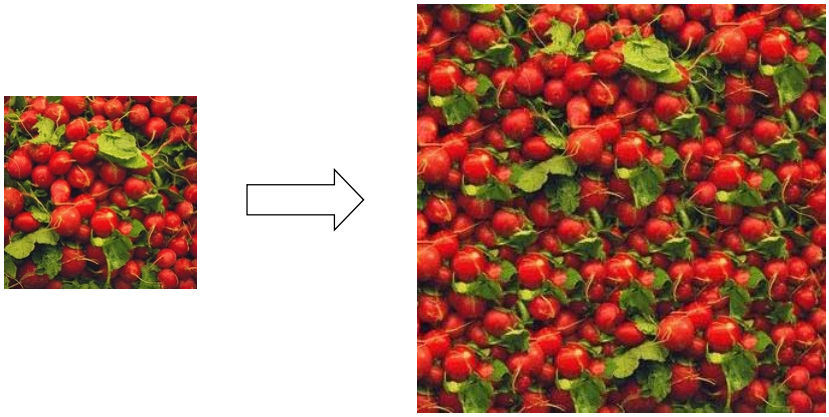
  how can we recover the 3D surface shape given the 2D texture image?

- Texture synthesis

  given a small texture image, how can we construct a texture image as large as we want?

3

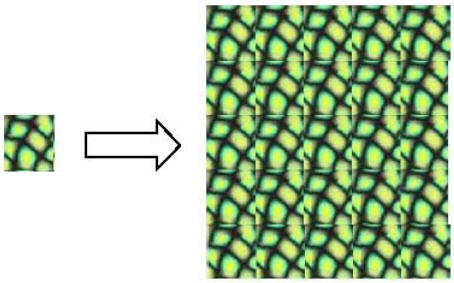
Texture Synthesis



4

An intuitive method

- Tiling small texture image patches one by one

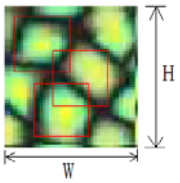


- The border of each unit can be easily identified
- We only have one candidate unit from input, and obviously the left/top border of this unit doesn't have to look similar to the right/bottom border of itself

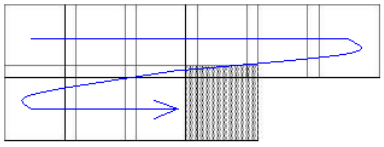
5

A Better ideas

- Image quilting (Efors & Freeman)
- Get more candidate units? How?
  random sampling



- Take these candidate units as building blocks, tile them in raster scanning order with some overlaps to do texture synthesis



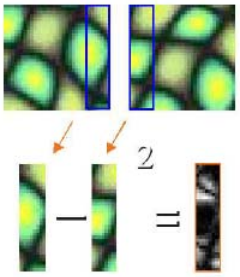
to continuously synthesize for the next block position, make sure select a candidate block which has a similar border to the adjoining borders of its neighboring blocks that have already been synthesized

6

Similarity of two neighboring blocks

- Sum-of-squared difference (SSD)

$$D = \sum_x \sum_y [B_1(x,y) - B_2(x,y)]^2$$



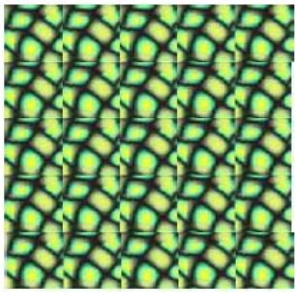
the smaller SSD is, the more similar the neighboring texture blocks are

Figure http://graphics.cs.cmu.edu/people/efros/research/quilting.html

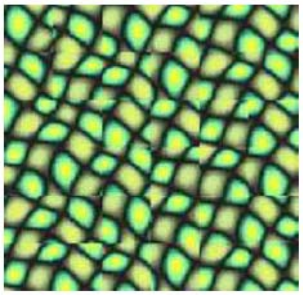
Texture synthesis algo. (I)

1. Pick size of block and size of overlap
2. Build the set of candidate texture blocks by randomly cutting from the original input sample texture
3. Synthesize blocks in raster order
4. Search from the candidate set to find one block that minimize the SSD of overlap regions with its neighboring blocks (above and left), which are already synthesized.
5. Paste this new block into resulting texture image

Comparison with previous simple tiling



simple tiling



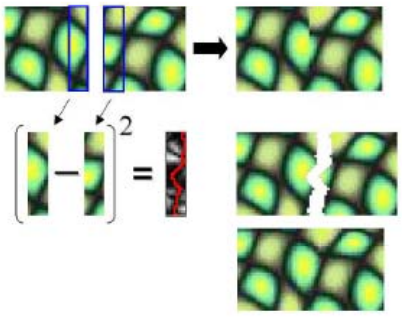
algo. (I)

- much better result is achieved in algo. (I)
- the borders between blocks are still quite noticeable
- How can we get better borders?

9

Image Blending

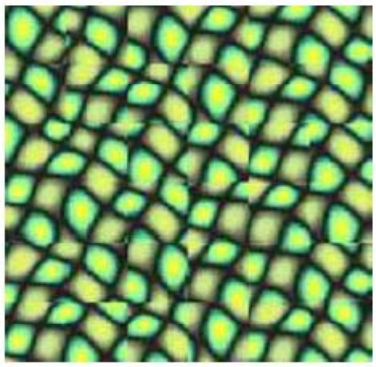
- Smooth the overlap region by finding an optimal ragged borders to combine the neighboring blocks



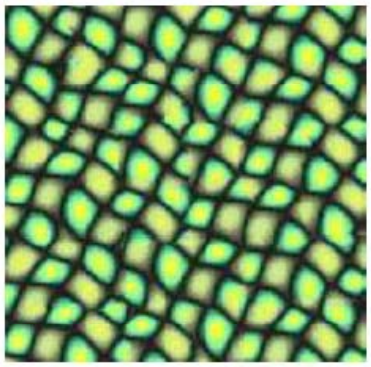
how can we find an optimal ragged border?
- dynamic programming

10

Texture synthesis w/ image blending



algo. (I)



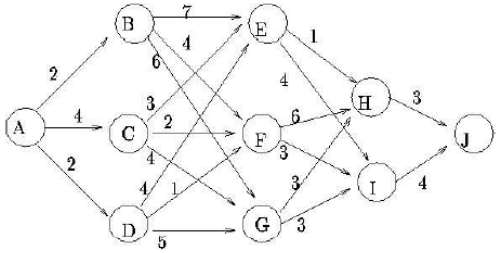
algo. (I) w/ image blending

Figure http://graphics.cs.cmu.edu/people/efros/research/quilting.html

11

Dynamic Programming (DP)

- We won't put too much math here, but borrow an illustrative example from M. A. Trick's tutorial on DP



stage 1 2 3 4 5

Can you find the shortest path from node A to node J?

12

Exhaustive Enumeration

- Exhaustively search for all feasible paths from node A to node J, then choose one with shortest distance
  - Combinatorial complexity
    1 * 3 * 3 * 2 * 1 = 18 possible paths

- Better ideas?
  - Look closely at two paths (A,B,E,H,J) and (A,B,E,I,J)
    contain the same sub-path (A,B,E)
    waste the computation on calculating this sub-path distance twice

13

A Better solution: DP

- A recursive calculation

$$f_j(S) = \min_{node Z in stage j+1} (c_{SZ} + f_{j+1}(Z))$$

$f_j(S)$ the shortest distance from node S to the destination J

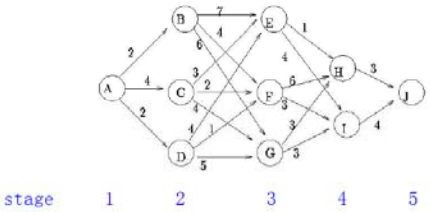
c_{sz} the length of arc SZ, i.e. the distance between S and Z

Intuition:

If the shortest path from node S to node J includes node Z, then the remaining path from node Z to node J in this shortest path must also be the shortest path from node Z to node J, there is no other paths from node Z to node J shorter than this one.

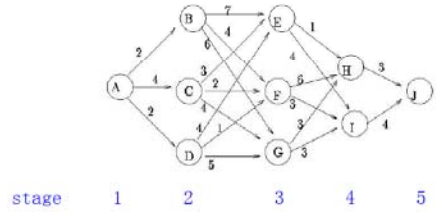
14

Procedure of DP



- Stage 5: $f_5(J) = 0$
- Stage 4:
  - $f_4(H) = c_{HJ} + f_5(J) = 3$
  - $f_4(I) = c_{IJ} + f_5(J) = 4$

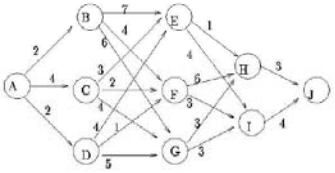
Procedure of DP



- Stage 3:

S_3	$c_{S_3 Z_3} + f_4(Z_3)$		$f_3(S_3)$	Decision
	H	I		Go to
E	4	8	4	H
F	9	7	7	I
G	6	7	6	H

Procedure of DP



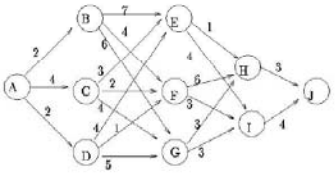
stage 1 2 3 4 5

- Stage 2:

S_2	$cs_2z_2 + f_3(Z_2)$			$f_2(S_2)$	Decision
	E	F	G		Go to
B	11	11	12	11	**E or F**
C	7	9	10	7	**E**
D	8	8	11	8	**E or F**

17

Procedure of DP



stage 1 2 3 4 5

- Stage 1:

S_1	$cs_1z_1 + f_2(Z_1)$			$f_2(S_1)$	Decision
	B	C	D		Go to
A	13	11	10	10	**D**

18

Principle of DP

- DP is a very useful technique, widely used to solve many engineering problems

- When can the DP be applied?
  - ❖ Properties:
    - the problem can be divided into stages
    - each stage has a number of states associated with it
    - the optimal solution has the optimal sub-path property

  - ❖ Benefits:

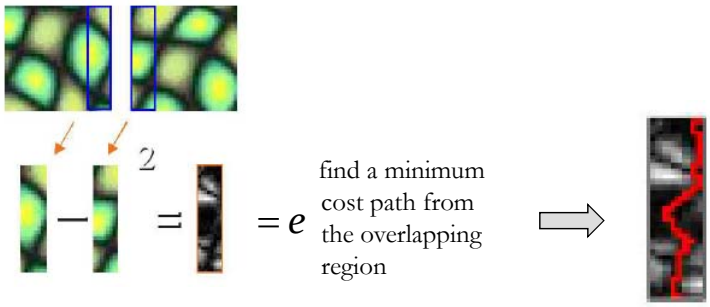
    Suppose a N stage problem, each stage has P states

    Exhaustive enumeration: $O(P^N \times (N-1))$ basic operations

    Dynamic Programming: $O(P^2 \times (N-1))$ basic operations

19

Image blending using DP

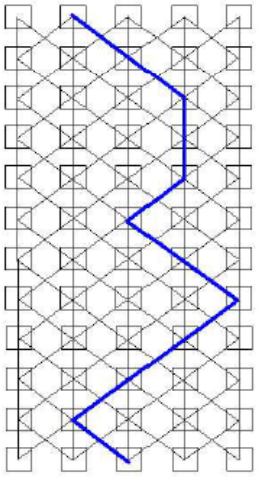


- The recursion

$$E_{i,j} = e_{i,j} + \min(E_{i-1,j-1}, E_{i-1,j}, E_{i-1,j+1})$$

20

Image blending using DP

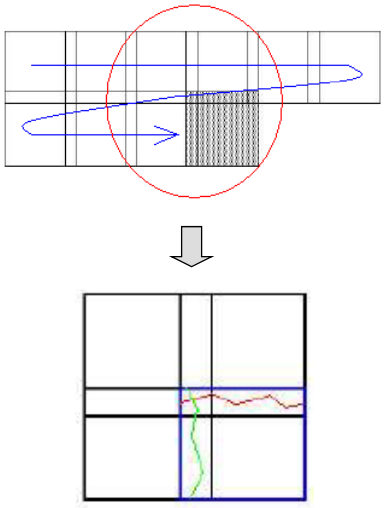


1. the minimum value of the last row in E indicate the end of the minimal cost path through the overlapping surface, and the path can be obtained by tracing back using the recorded information during DP

2. classify the pixels in the overlapping region according to the found minimal cost path

21

Texture synthesis w/ image blending



1. two minimal cost paths, one for vertical overlap, the other for horizontal overlap

2. for each pixel in these two overlap regions, if its location is either on the left side of the vertical path or on the top side of the horizontal path, its value should be taken from the blocks that have already been synthesized, otherwise its value should be from the new block

22

Texture synthesis w/ image blending

1. Pick size of block and size of overlap
2. Build the set of candidate texture blocks by randomly cutting from the original input sample texture
3. Synthesize blocks in raster order
4. Search from the candidate set to find one block that minimize the SSD of overlap regions with its neighboring blocks (above and left), which are already synthesized.
5. Paste this new block into resulting texture image according to the minimal error paths obtained by dynamic programming

23

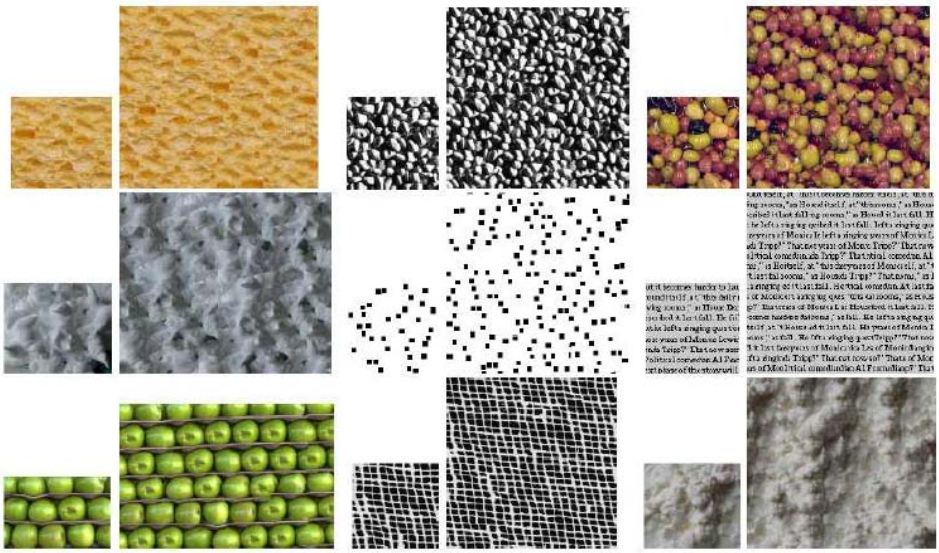
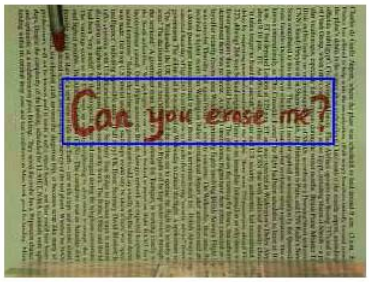
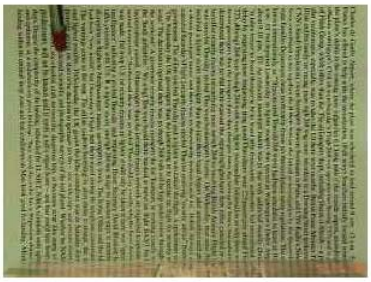


Figure http://graphics.cs.cmu.edu/people/efros/research/quilting.html

24

Texture Impainting





the only difference is that now we might need to consider the {above, left, right} or {above, left, bottom} or {above, left, right, bottom} overlapping region cases

25

References

- D. A. Forsyth and J. Ponce. Computer Vision: A Modern Approach. Prentice Hall, 2002.
- A. A. Efros and W. T. Freeman. Image Quilting for Texture Synthesis and Transfer. Proceedings of SIGGRAPH '01, Los Angeles, California, August, 2001. http://graphics.cs.cmu.edu/people/efros/research/quilting.html
- M. A. Trick. A Tutorial on Dynamic Programming. http://mat.gsia.cmu.edu/classes/dynamic/dynamic.html

26