Graphcut Textures: Image and Video Synthesis using Graph Cuts (Kwatra, et al)

presented by

Adeel Bhutta

Outline

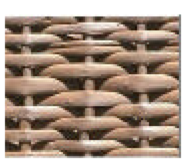
- Texture Synthesis
  - What is Texture?
  - How to Synthesize?
- Graphcut Textures
  - Main Idea, Contribution
  - Patch Placement and Matching Techniques
  - Patch Fitting
  - Refinements and Extensions
- Video Textures

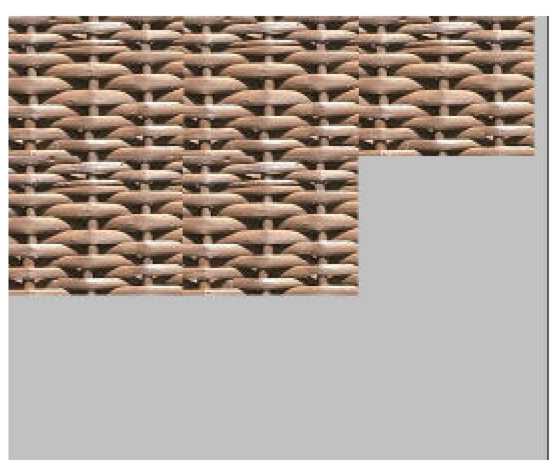
Texture

- Generate a large image from smaller Image or longer video from smaller one.
- How to do it?
  - Copy patches (or pixels) from input to output.
- Problems
  - Artifacts (e.g., boundaries of patches)

Solution

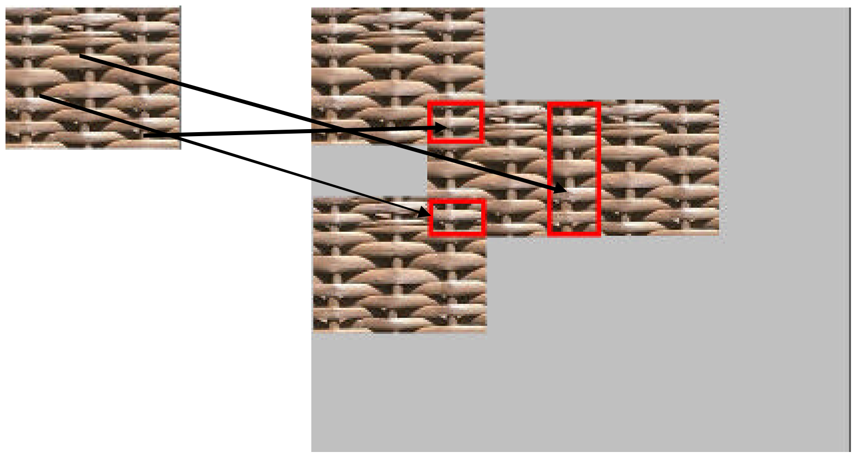
- Copy patches from input to output.



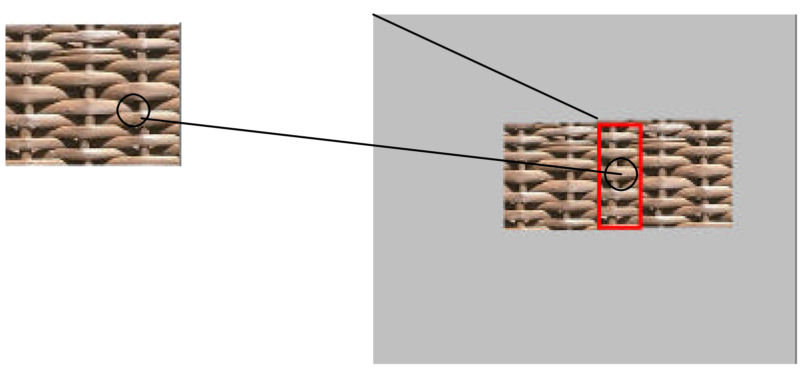


Solution

- Copy patches from input to output with overlap.



Definitions



- ***Where*** to position the input texture (e.g., translation) called *Offset*
- ***Which*** part of the input texture to transfer called *Seam*

The Synthesis Process

- Step1: Patch Placement and Matching (*Choose Candidate patches or offset*)
  - Random Placement
  - Entire Patch Matching
  - Sub-patch Matching

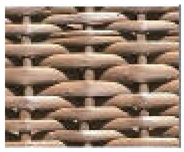
- Step2: Patch Fitting (*Choose optimal portion or seam*)
  - Only those pixels are copied that are chosen by graph-cut algorithm.
  - Cost of graph-cut is a measure of similarity.

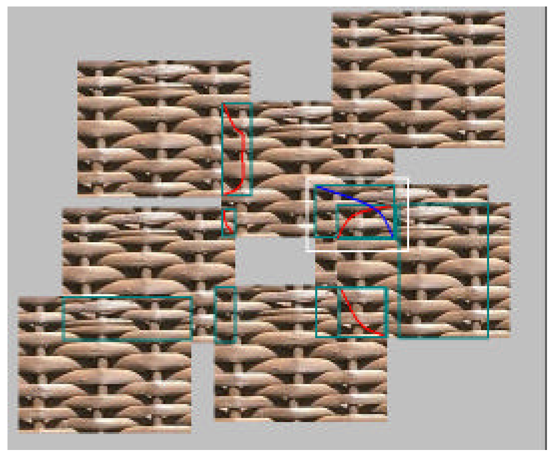
Step1. Patch Placement

Can be seen as a '*translation*' applied to input.

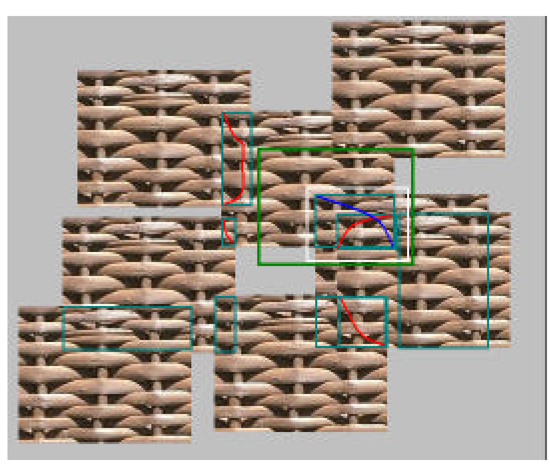
- **Random Placement**
  - Entire input image is translated to random location in the output image
  - Good results for random textures
- **Patch Matching (Entire or Sub)**
  - Used when we already have some patches in the output image (*refinement*).
  - Every seam has a cost (min graph cut cost)
  - Uses *error region*

Error Region (Contd.)





Error Region (Contd.)



Error Region

- Error = seam cost
  - Sum of costs along minimum cut path
- Choose a pixel with largest error
- Select a region around that pixel, called *error region*
- Patch Matching (entire or sub) will select those patches that completely cover our error region.

Patch Placement (Contd.)

- **Entire Patch Matching**
  - Search for translated input versions and choose that gives best match

Matching criteria:

Normalized SSD:

$$C(t) = \frac{1}{|A_t|} \sum_{p \in A_t} |I(p) - O(p+t)|^2$$

C*: cost of translation*
T*: translation*
A*: overlapping input*
I*: Input*
O*: Output*

- C(t): The smaller, the better (means similar).

Patch Placement (Contd.)

- Compute cost for all possible *offsets*. Cost is inversely proportional to similarity.
- Choose the cost that has the highest probability of resulting in a similar region.

$$P(t) \propto e^{-\frac{C(t)}{ks^2}}$$

S*: SD of pixel value*
***k**: Randomness*
k>>0 Random selection
k<<1 Matching with output
0.001 < k < 1.0

Good results for structured and semi-structured textures

Patch Placement (Contd.)

- **Sub-Patch Matching**
  - Pick a small sub-patch from output
  - Search for output-patch in input texture or look for translations of input sub-patch.

Matching criteria:

$$C(t) = \sum_{p \in So} \left| I(p-t) - O(p) \right|^2$$

S_0: Output sub patch
C: Cost of translation

- Use the same probability function
- Best results for unstructured regions or video textures (fire, waves, smoke, etc.)

2. Patch Fitting

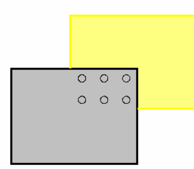
- Make graph for the overlap region
  - Every pixel is a node.
  - Edge weights:

$$M(s,t,A,B) = \|A(s) - B(s)\| + \|A(t) - B(t)\|$$

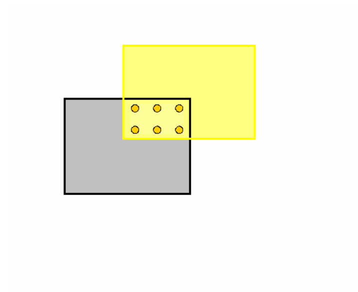
***s, t**: Adjacent pixels*
***A, B**: Old and New patches*
(i.e., color values)

- Associate weight with each edge
- Find minimum graph cut

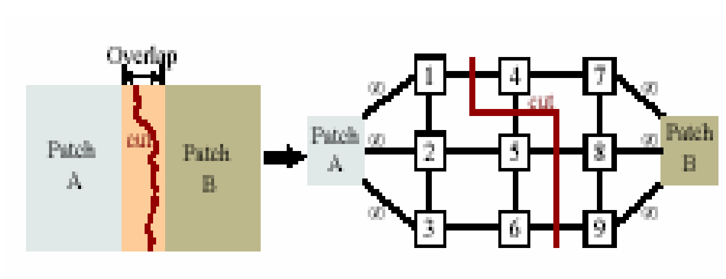
Construction of a Graph



Construction of a Graph



Finding Min Cut Path



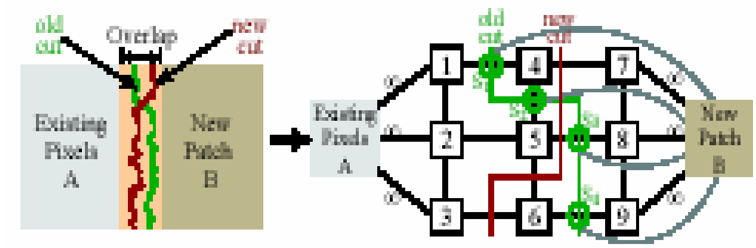
- Add two additional nodes representing patches A and B

$$M(s,t,A,B) = \|A(s) - B(s)\| + \|A(t) - B(t)\|$$

- Find Minimum Cut path for remaining nodes

Patch Fitting (Contd.)

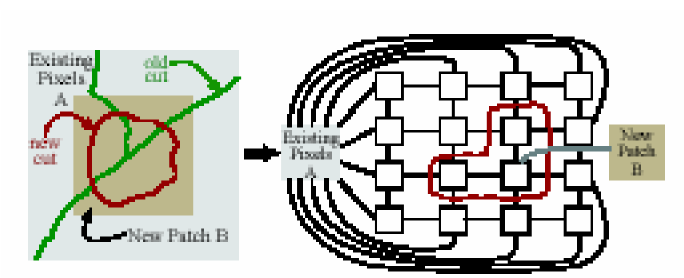
- Accounting for old seams



- For each seam add another node (*seam node*).
- Connect seam node to patch B. Weight is $M(1,4,A_1,A_4)$.
- Connect node1 to s1. Weight is $M(1,4,A_1,B)$.
- Connect s1 to node 4. Weight is $M(1,4,B,A_4)$.
- Find new Min Cut path.

Patch Fitting (Contd.)

- Surrounding Regions: To overwrite a potentially visible seams in the area that is already covered by earlier steps.



- All border pixels are connected to existing patches.
- Follow the same step (add additional nodes, connect to B, calculate new costs, calculate new minimum cut)

Refinements

- Modified Matching cost function

$$M'(s,t,A,B) = \frac{M(s,t,A,B)}{\left\|G_A^d(s)\right\| + \left\|G_A^d(t)\right\| + \left\|G_B^d(s)\right\| + \left\|G_B^d(t)\right\|}$$

d: direction of gradient (edge between s and t)

G_A^d: Gradient in patch A along the direction d

M': penalizes seams going through low frequency regions more than those going through high frequency regions.

- Search across all translations is costly (use FFT)

$$C(t) = \sum_p I^2(p-t) + \sum_p O^2(p) - \boxed{2\sum_p I(p-t)O(p)}$$

- Convolution (FFT)

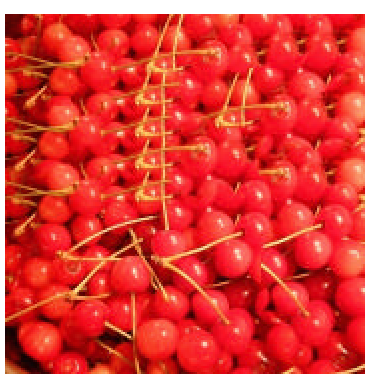
Extensions

- Translation to Transformation
  - Rotation, Scaling, Affine or Projection.
- Interactive Merging and Blending
  - Many source Images
  - User specifies position & constraints pixels
  - Algorithm finds best seam.
  - SIGGRAPH banner.

Results

Input

Results



Iteration1 Result

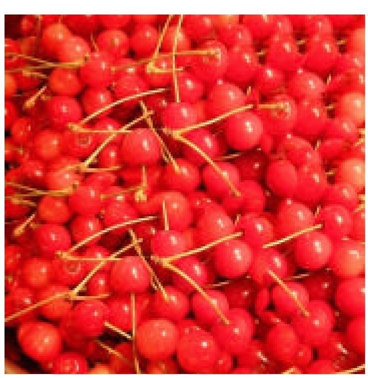
Error

Results

Iteration2 Result

Error

Results



Iteration3 Result

Error

Results

Iteration4 Result

Error

Results

Iteration5 Result

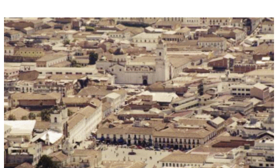
Error

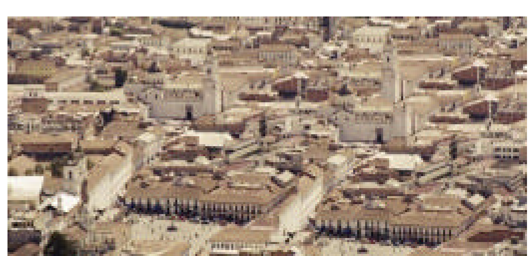
Results

Iteration6 Result

Error

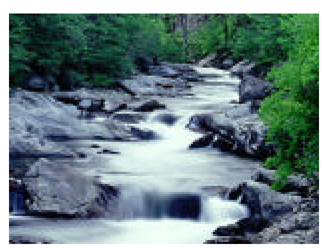
Results

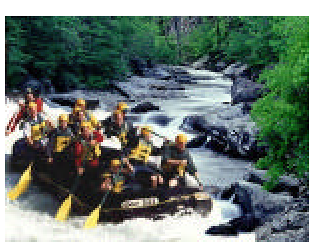


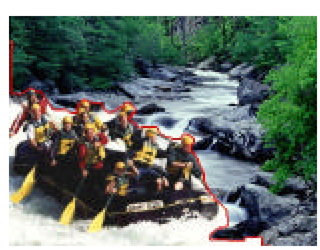


Results

Results







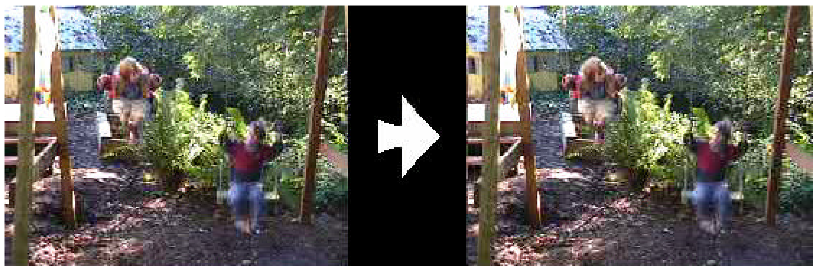
Results

Extensions (Contd.)

Goal is to loop the video forever.

- Video Texture
  - One way is to find the pair of similar looking frames and use them to repeat the video.
- Video Synthesis using Graphcut
  - Find time of transition on pixel-wise basis

Video Textures



video clip video texture

Video Texture - GraphCut

- Find *good* transition between pair of images
- Take a window around transition (60 frames)
- Construct the graph by connecting a pixel to its neighbors in space and time
- Min cut will give you time of transition on per pixel basis
- Use translation in time and space both

Video Synthesis (Contd.)

- To loop the video, add k frames in start and end of video (10 frames), constraint these frames to stay the same, graph is generated and best seam is found, then k frames are removed.
- Videos: Clouds, River, and Water-Fall, Grass, Pond, Fountain, and Beach

Results

Links

- More results and videos

http://www.cc.gatech.edu/cpl/projects/graphcuttextures/

- Graph-cut code available at

http://www.cs.cornell.edu/People/vnk/software.html