PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing

Connelly Barnes Eli Shechtman Adam Finkelstein Dan B Goldman

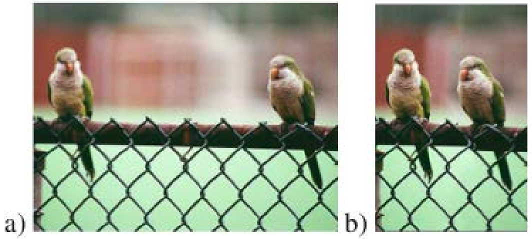
CS 294-69 Paper Presentation

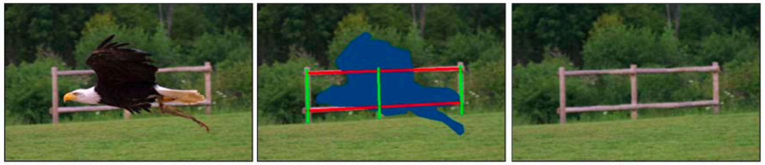
Jiamin Bai (Presenter)

Stacy Hsueh (Discussant)

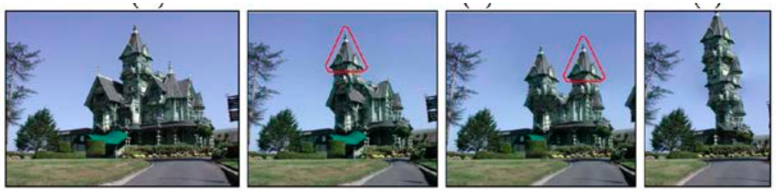
Structural Image Editing

- Synthesize complex texture and image structures that resembles input imagery
  - – Image retargeting
  - – Image completion
  - – Image reshuffling



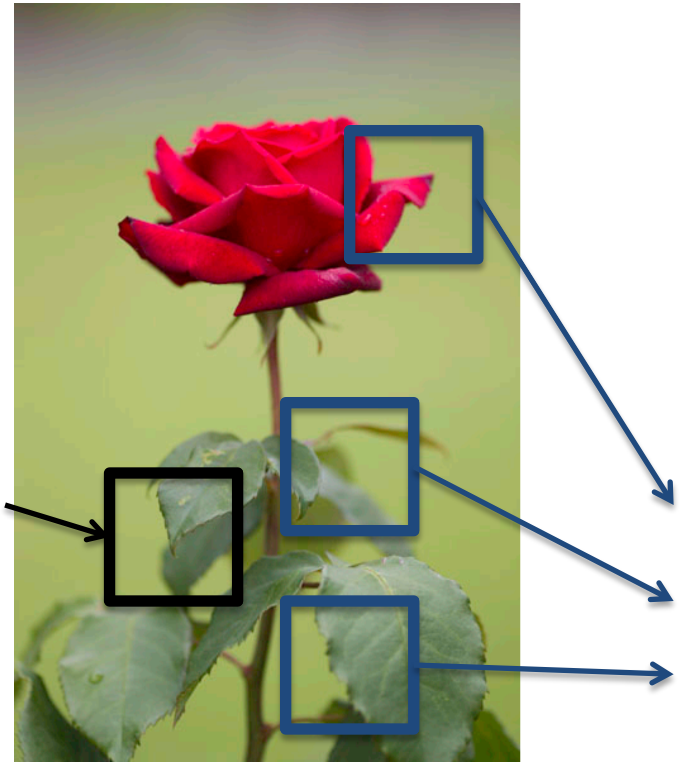


(a) input (b) hole and guides (c) completion result



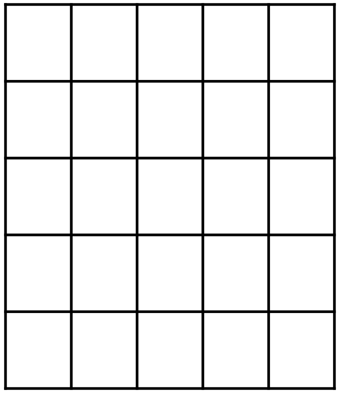
Matching Patches

- Nearest Neighbor Search



Patch p

Which patch is most similar?



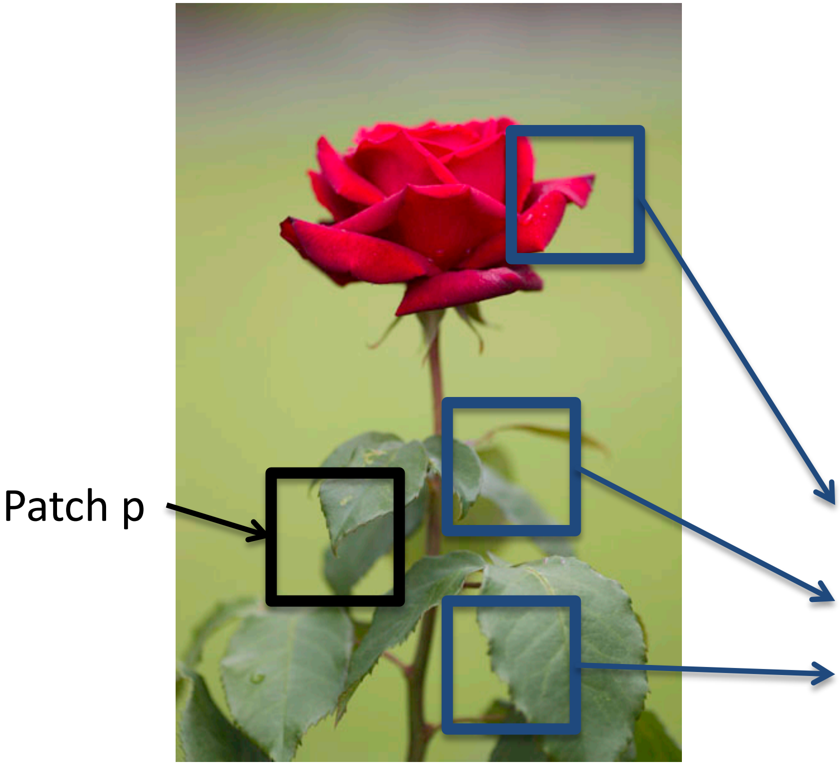
Patch p with 25 dimensions

Matching Patches

- Naïve Approach

Sample every possible patch to find best match!

$O(mM^2)$



Which patch is most similar?

Key Ideas

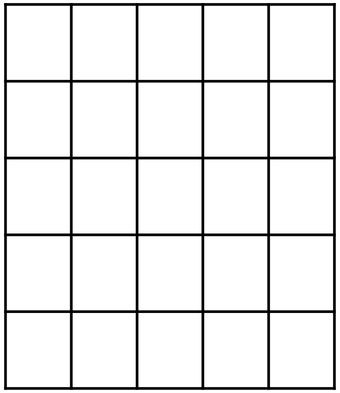
- Search space
  - Patch offsets vs Patches

- Neighboring pixels have coherent matches

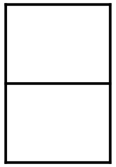
- Large number of random sampling will yield some good guesses.

Key Ideas

Patches vs. Patch offsets



Patch p with 25 dimensions

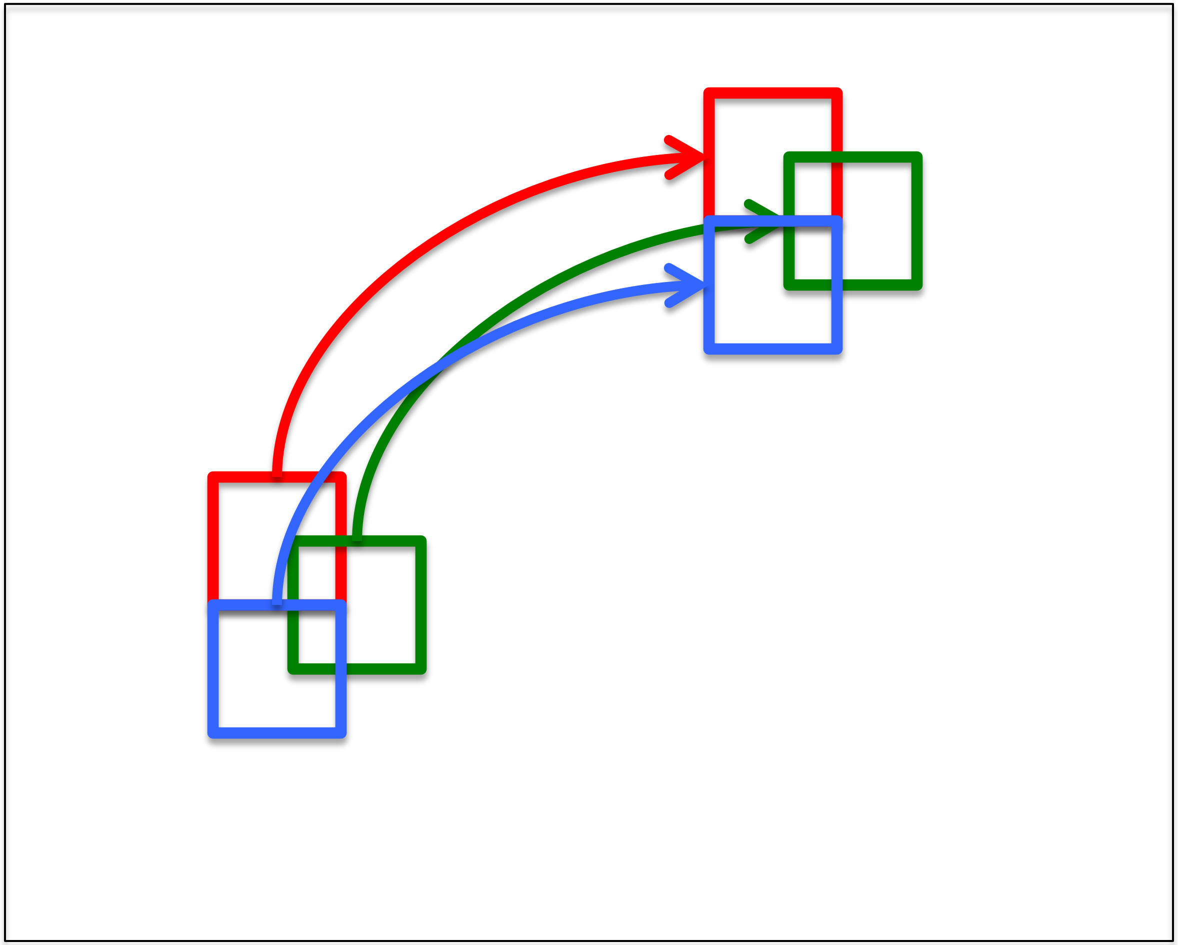


(x and y displacements)

Patch p offset search
with 2 dimensions

Key Ideas

Coherent matches with neighbors



Key Ideas

Large numbers of guesses

$\boldsymbol{M}$ number of total pixels

Probability of correct random guess: $1/\boldsymbol{M}$

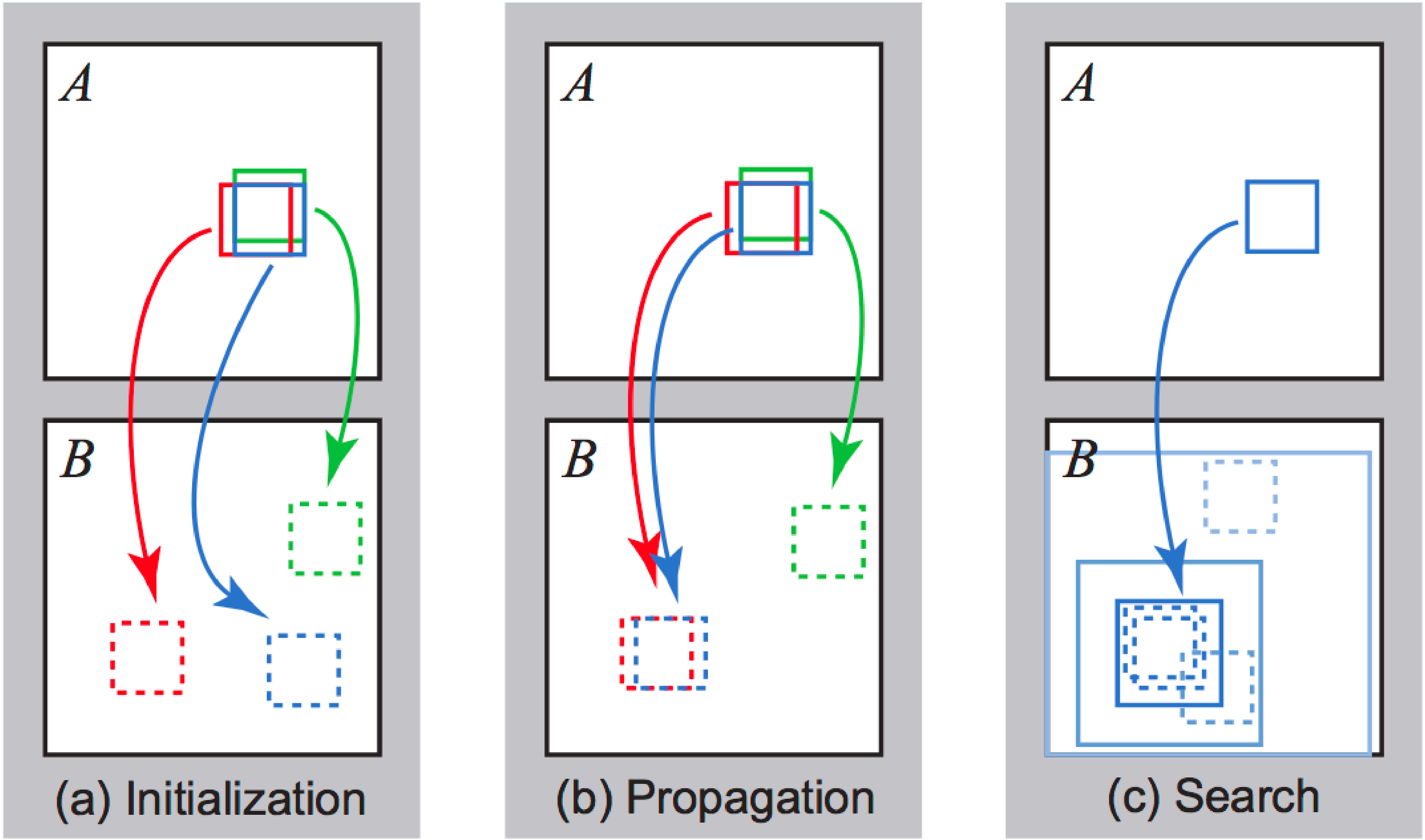
Probability of incorrect random guess: $1 - 1/\boldsymbol{M}$

Probability of all pixels with incorrect guess: $(1 - 1/\boldsymbol{M})^{\boldsymbol{M}}$ [approximately 0.37]

$\Rightarrow$Probability of at least 1 pixel with correct guess : $1 - (1 - 1/\boldsymbol{M})^{\boldsymbol{M}}$

$\Rightarrow$Probability of at least 1 pixel with good enough guess: $1 - (1 - C/\boldsymbol{M})^{\boldsymbol{M}}$

Algorithm – 3 steps



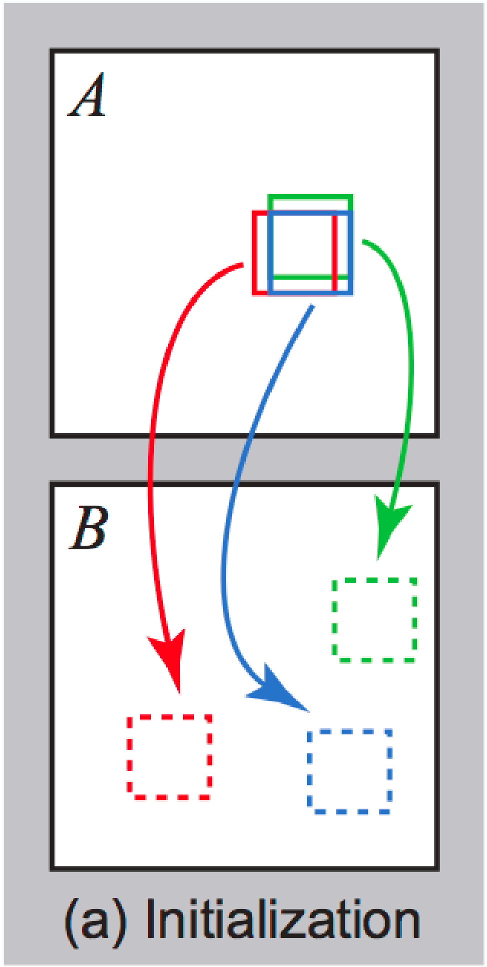
(a) Initialization

(b) Propagation

(c) Search

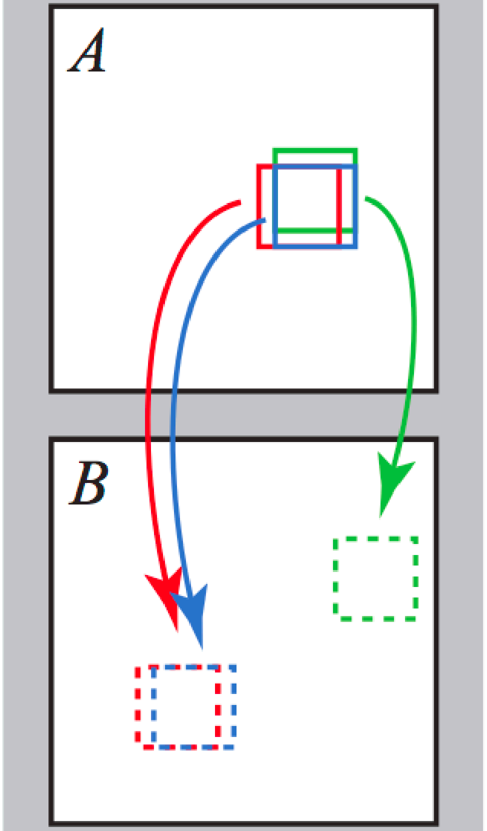
Algorithm – Initialization

- Each pixel is given a random patch offset as initialization



Algorithm – Propagation

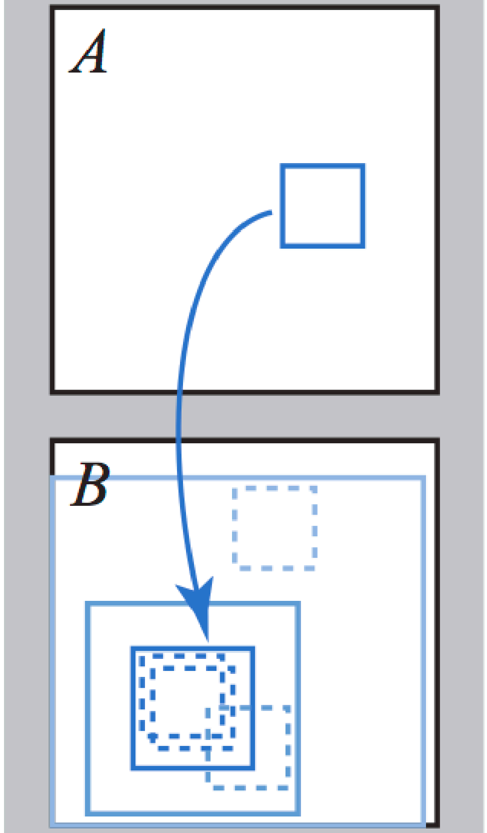
- Each pixels checks if the offsets from neighboring patches give a better matching patch. If so, adopt neighbor's patch offset.



(b) Propagation

Algorithm – Search

- Each pixels searches for better patch offsets within a concentric radius around the current offset.
- The search radius starts with the size of the image and is halved each time until it is 1.



(c) Search

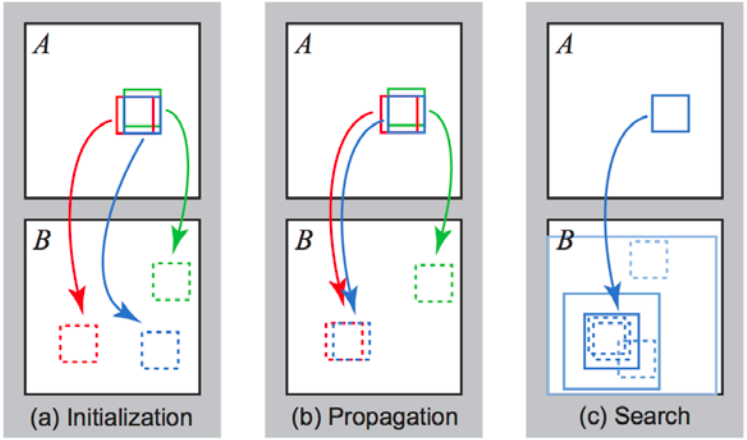
Algorithm

1. Initialize pixels with random patch offsets

2. Check if neighbors have better patch offsets

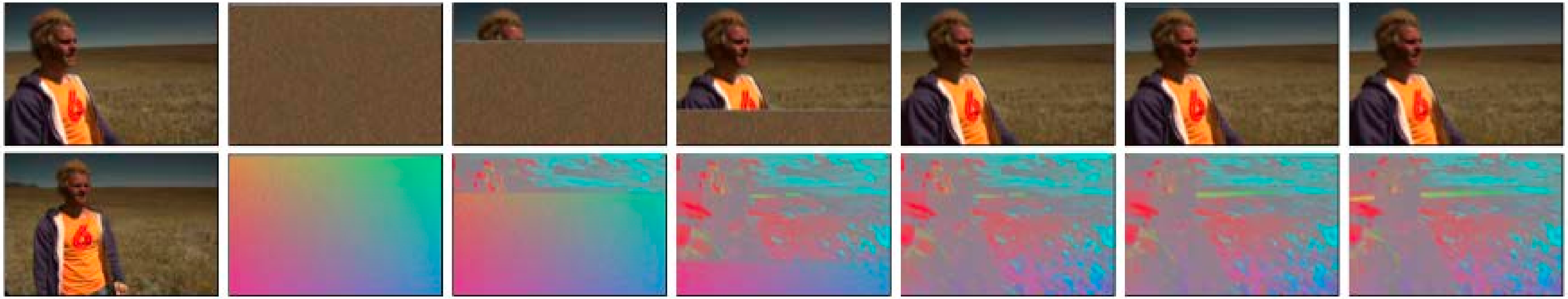
3. Search in concentric radius around the current offset for better better patch offsets

4. Go to Step 2 until converge.



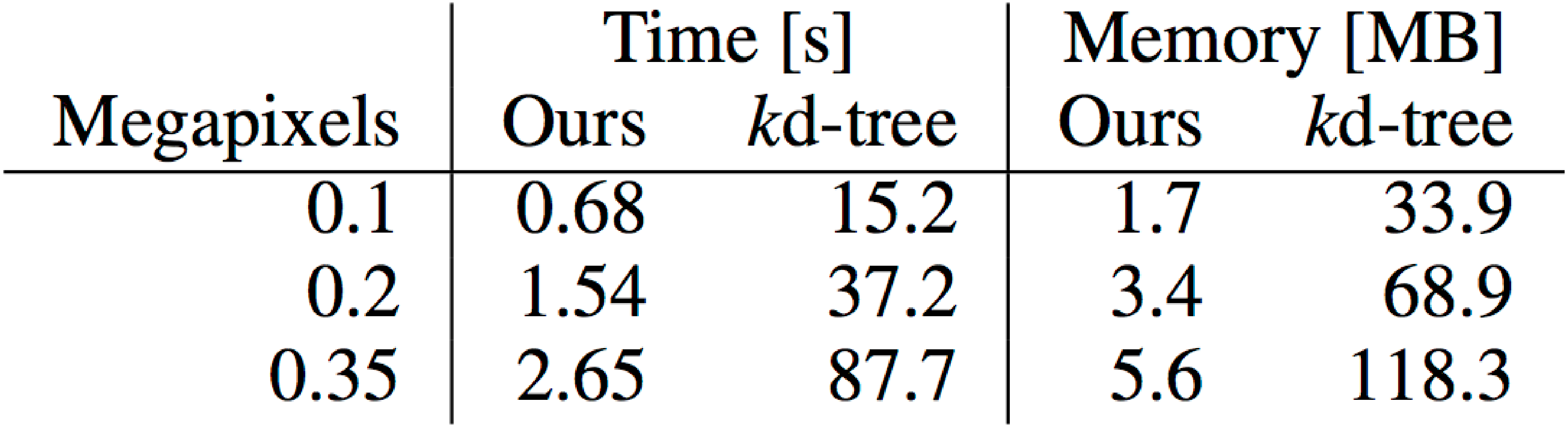
O(mMlogM)

Algorithm



(a) originals (b) random (c) $\frac{1}{4}$ iteration (d) $\frac{3}{4}$ iteration (e) 1 iteration (f) 2 iterations (g) 5 iterations

Speed Improvements



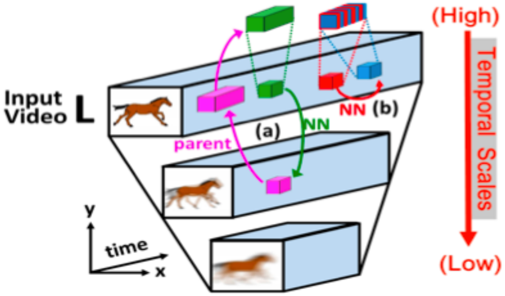
	Time [s]		Memory [MB]	
Megapixels	Ours	*k*d-tree	Ours	*k*d-tree
0.1	0.68	15.2	1.7	33.9
0.2	1.54	37.2	3.4	68.9
0.35	2.65	87.7	5.6	118.3

Impact

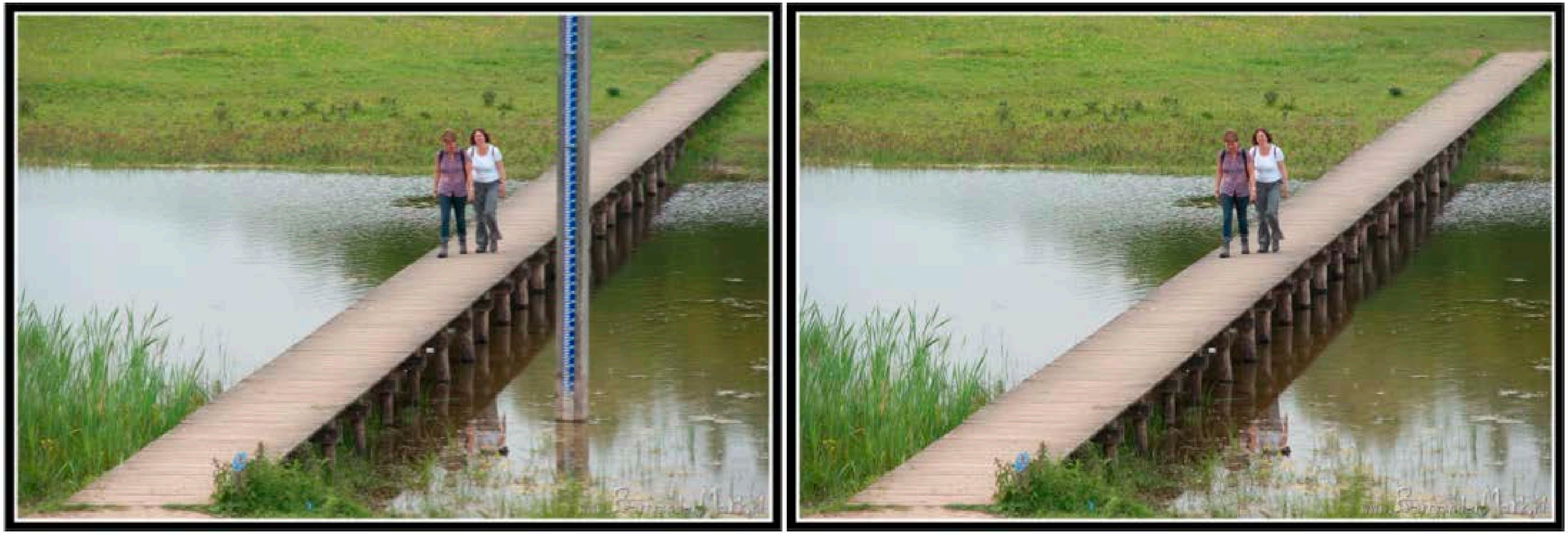
- Not only used in graphics, but in vision
  - Non-local means denoising
  - Image forensics
  - Object detection

- Video Tapestries

- Videos: Patchmatch in 3D
  - Temporal super-resolution



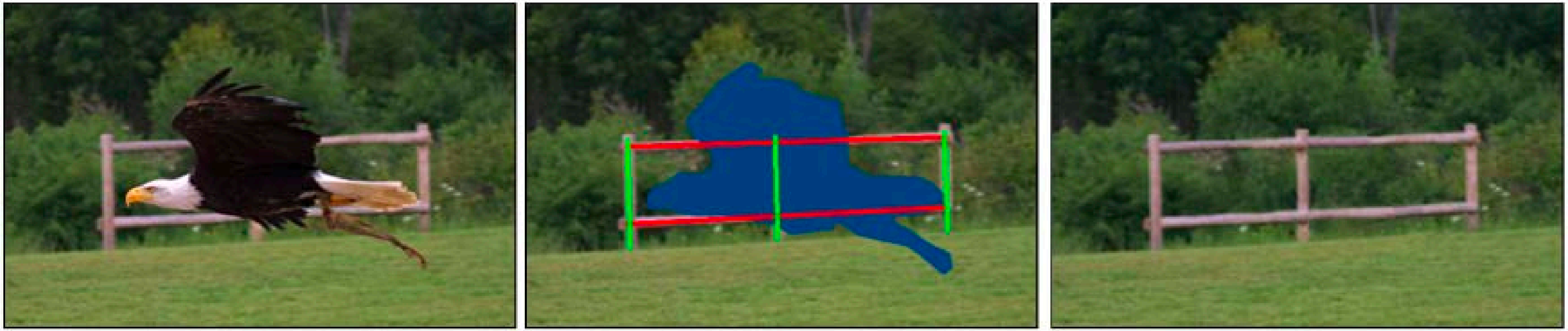
Results



(a) input

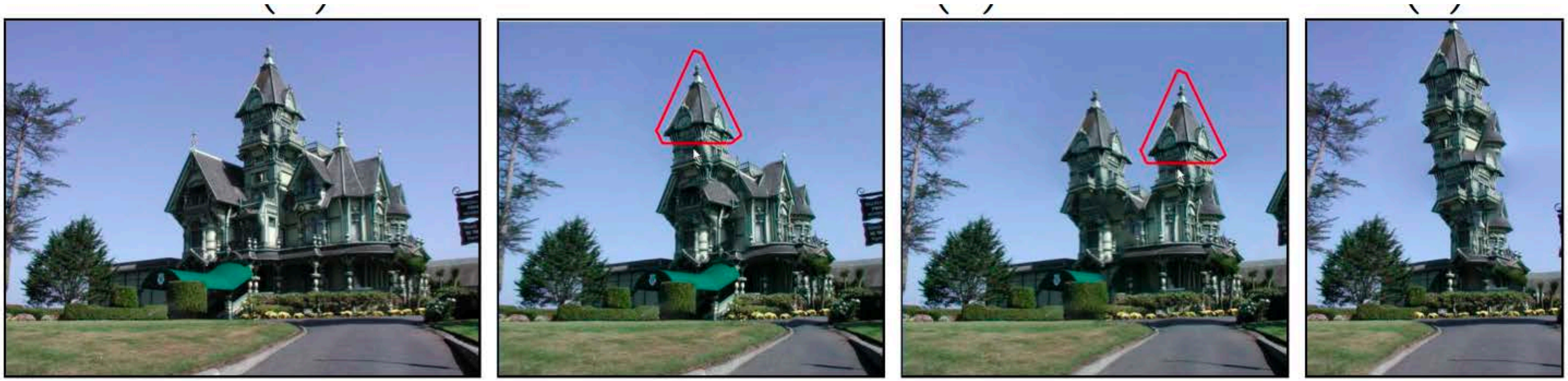
(b) result

Results

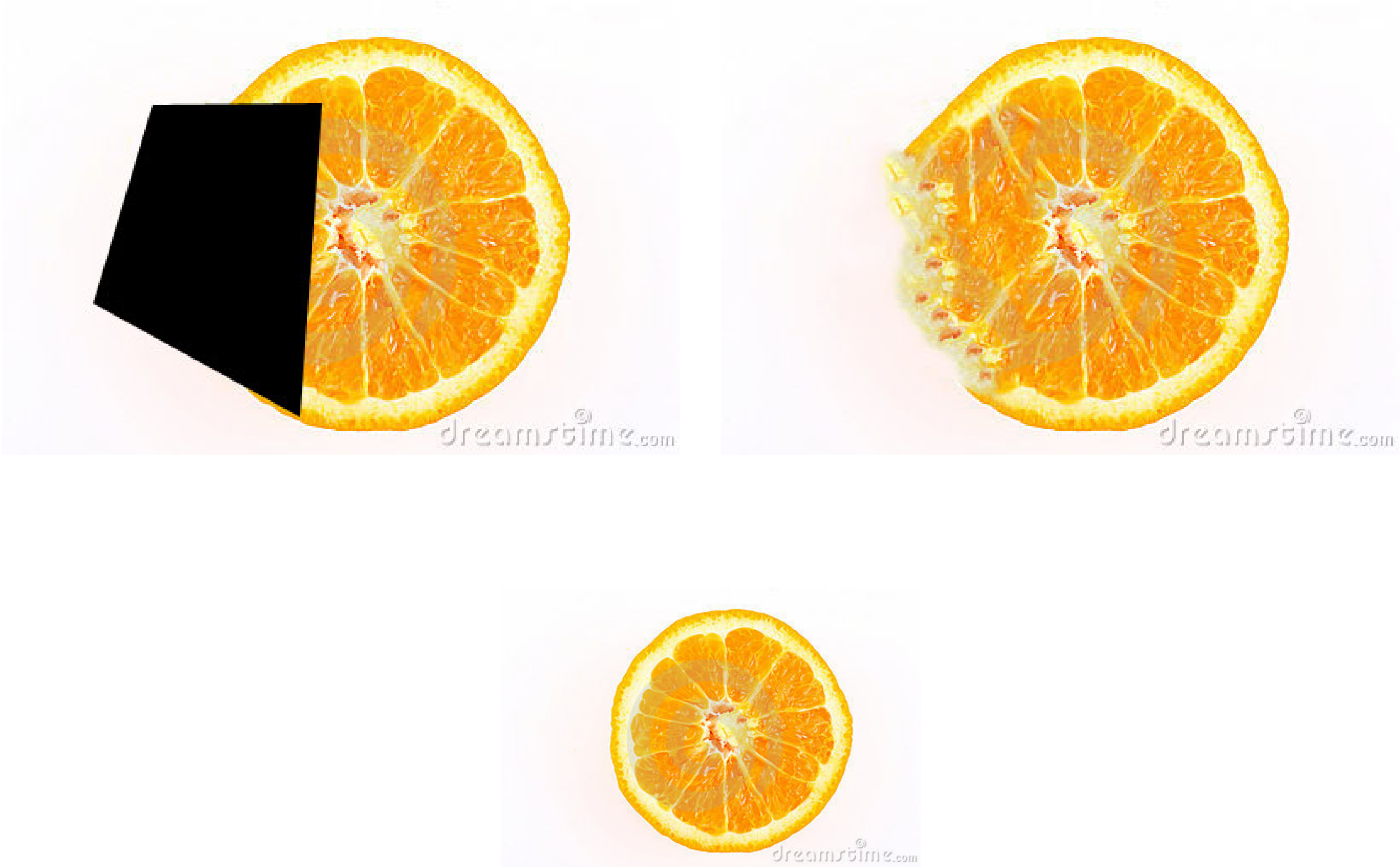


(a) input (b) hole and guides (c) completion result

Results



Results (Failure)



Results (Failure)