Image Warping

CS 355: Introduction to Graphics and Image Processing

Geometric Operations

- Transformations (Shift, Rotation, etc.)
- Resizing
- Adding/Correcting a Warp
- Texture Mapping
- Morphing

Texture Mapping

- Mapping an image onto the surface of a geometric model
- Increased realism
- Requires warping image onto the object surface

Image Morphing

- Warp a pair of images based on corresponding points
- Combination of warp/cross-dissolve

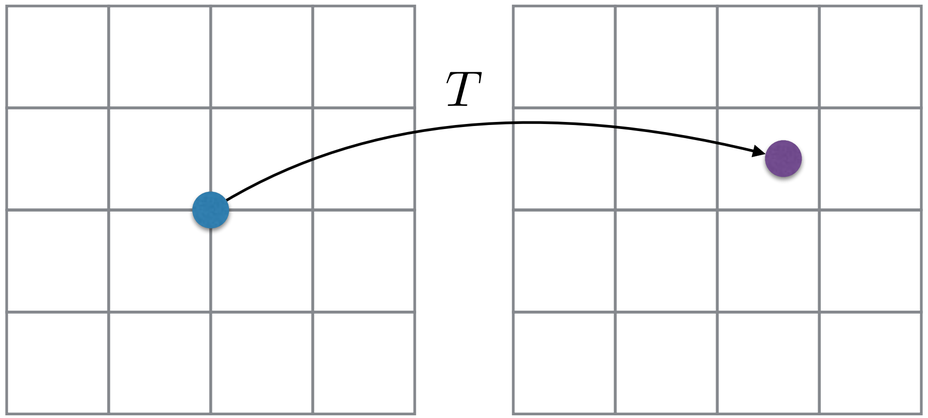
Transformations

- Define what pixels go where
  - Translate (shift image)
  - Rotate around point
  - Scale (resize)
  - Affine
  - Perspective
  - Or anything else…

$$\mathbf{p}' = T(\mathbf{p})$$

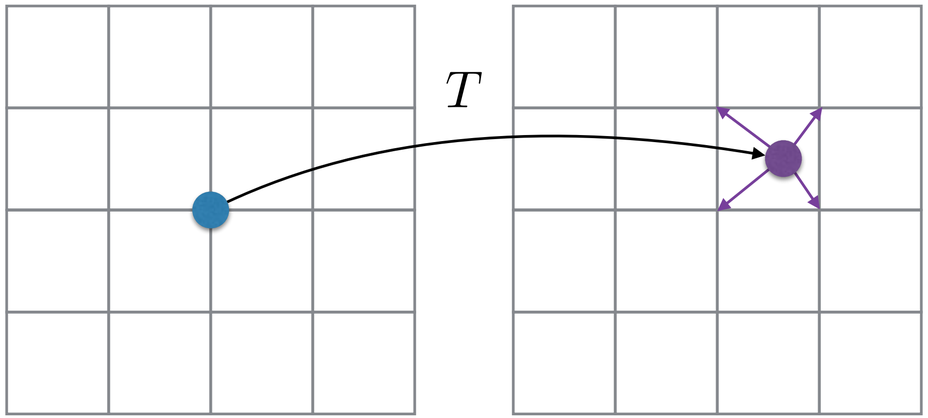
Forward Mapping

$$I'(T(\mathbf{p})) = I(\mathbf{p})$$



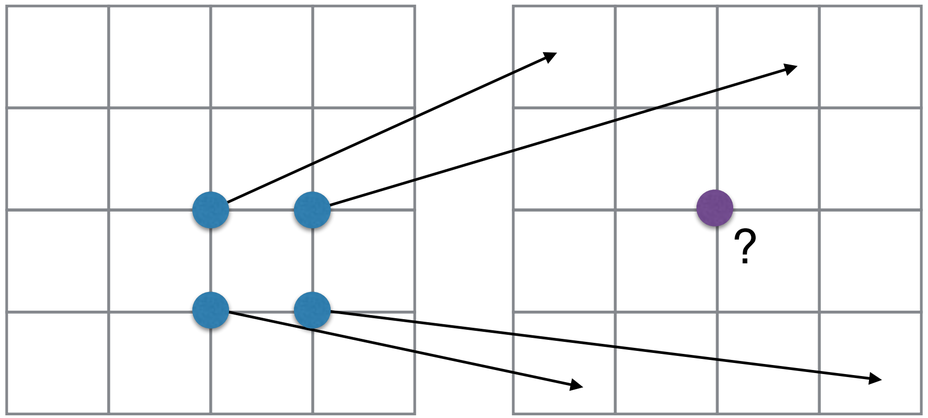
Forward Mapping

Problem: doesn't map to discrete pixel locations



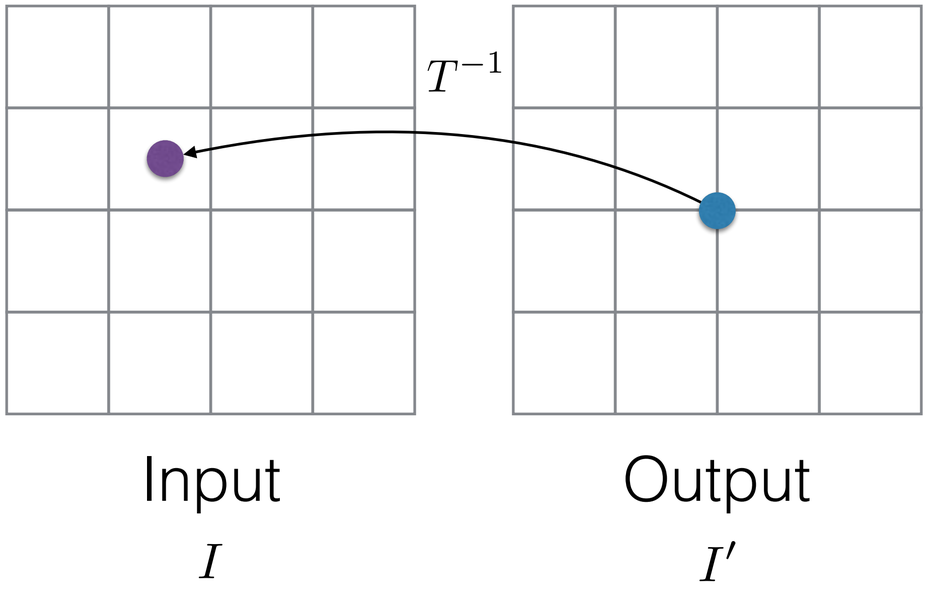
Forward Mapping

Problem: may produce holes in the output



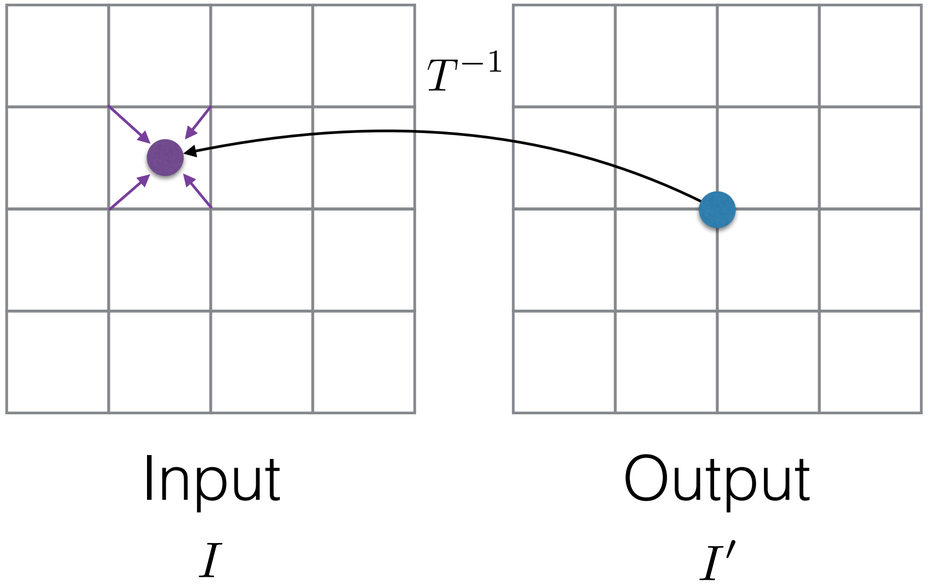
Backward Mapping

$$\mathbf{p} = T^{-1}(\mathbf{p}')$$



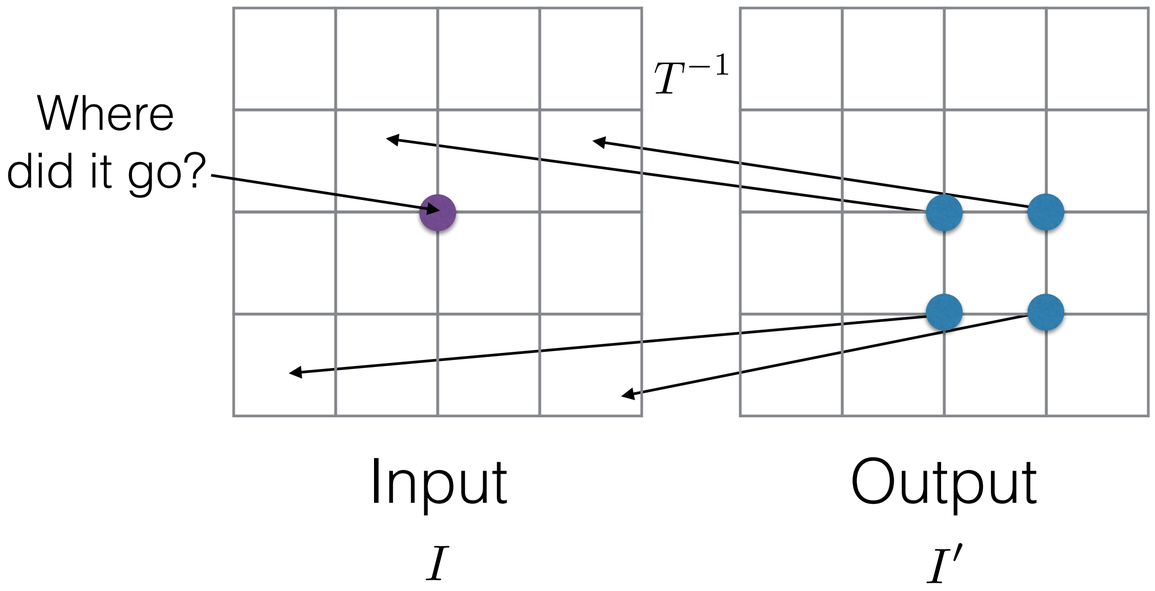
Backward Mapping

Problem: doesn't map *from* discrete pixel locations



Backward Mapping

Problem: may miss things in the input



Backward Mapping

- Reducing an image (even locally) can cause aliasing and Moiré patterns just like insufficient sampling
- May have to blur first

Forward vs. Backward

Forward mapping:

- Uses forward transformation
- Have to spread effect ("splatting")
- Parallel writes (slow)
- May cause holes in output

Backward mapping:

if you can!

- Requires inverse transformation
- Standard interpolation (easy)
- Parallel reads (fast)
- May miss fine detail and/or cause aliasing

Repeated Warping

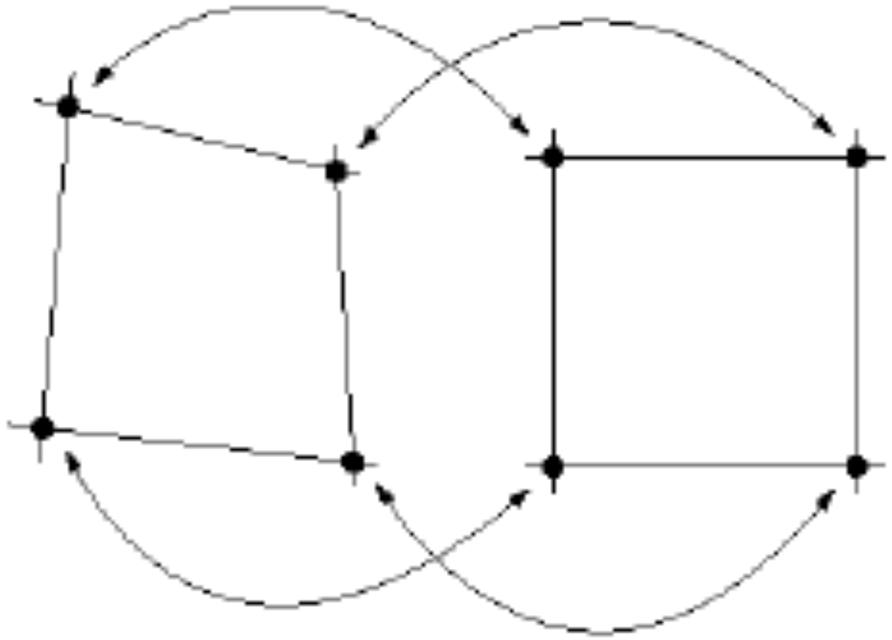
- Each spatial transformation requires interpolation, which means some error (blurring, jaggies, etc.)
- Repeated transformations accumulate this error
- Better idea:
  - keep track of transformations
  - composite transformations
  - warp image once

Transformations Revisited

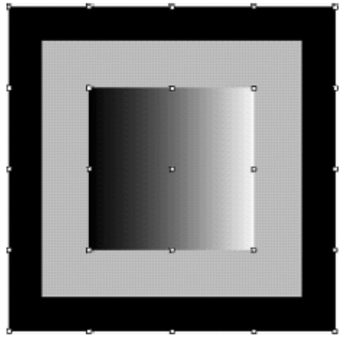
- Can be a single, closed-form transformation:
  - Scale, translation, rotation, skew,
    other affine, or even perspective
  - But all of the pixels undergo the same global transformation
- What about locally different transformations?

Warp Meshes

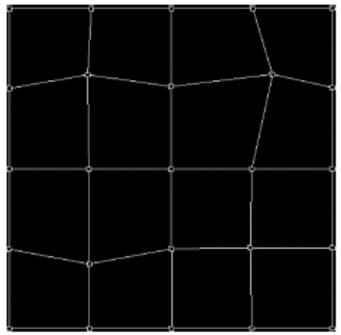
- Break source/destination images into meshes of quadrilaterals
- Map between corresponding “tiepoints”



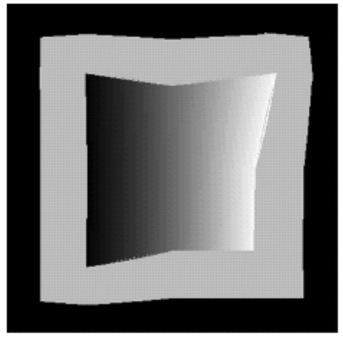
Warp Meshes



Source with tie points

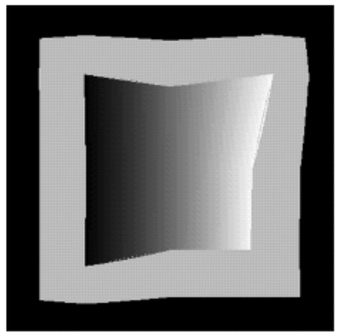


Destination mesh

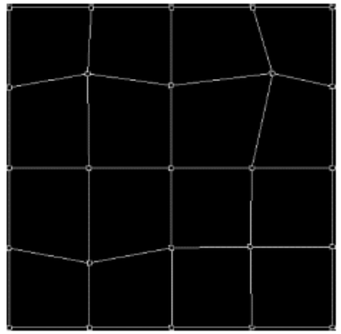


Result

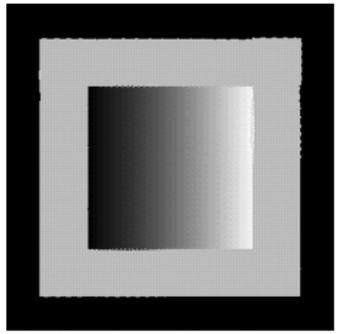
Correcting Distortion



Distorted source



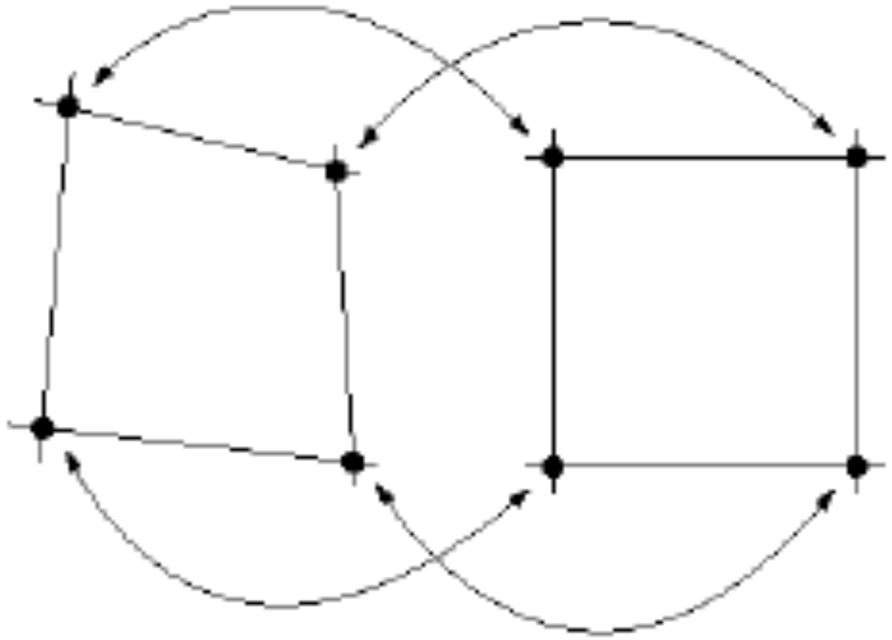
Overlaying source mesh



Restored result

Interpolating Transformations

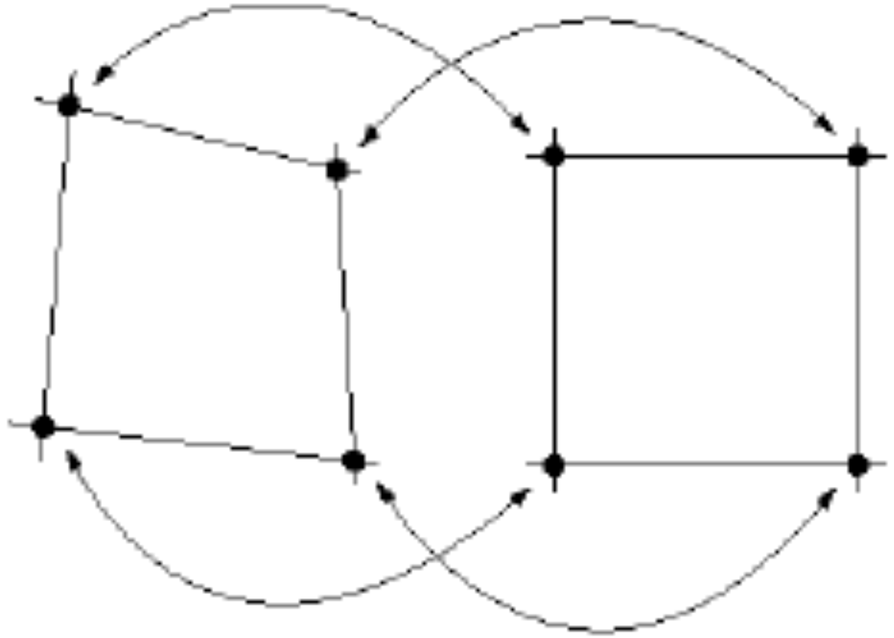
- To get the transformation from one quadrilateral to the corresponding one, interpolate the transformation
- Same idea as interpolating *values*, but for *positions* (x,y)
- Already seen how to do bilinear interpolation on quadrilaterals



Bilinear Transformations

Solve two systems of four equations each:

$$x' = a_1 x + b_1 y + c_1 xy + d_1$$
$$y' = a_2 x + b_2 y + c_2 xy + d_2$$



Key idea:
If you can write the transformation in terms of some number of unknowns, you can solve for it with enough corresponding points

Higher-Order Warps

- Can extend idea of bilinear interpolation of the transformation to higher-order
- More neighboring mesh points
- More computation
- Smoother results
- See Photoshop’s “Liquify” tool

Coming up...

- Perspective warping
- Back to more geometry:
  - Representing lines, curves, surfaces
  - Geometric tests