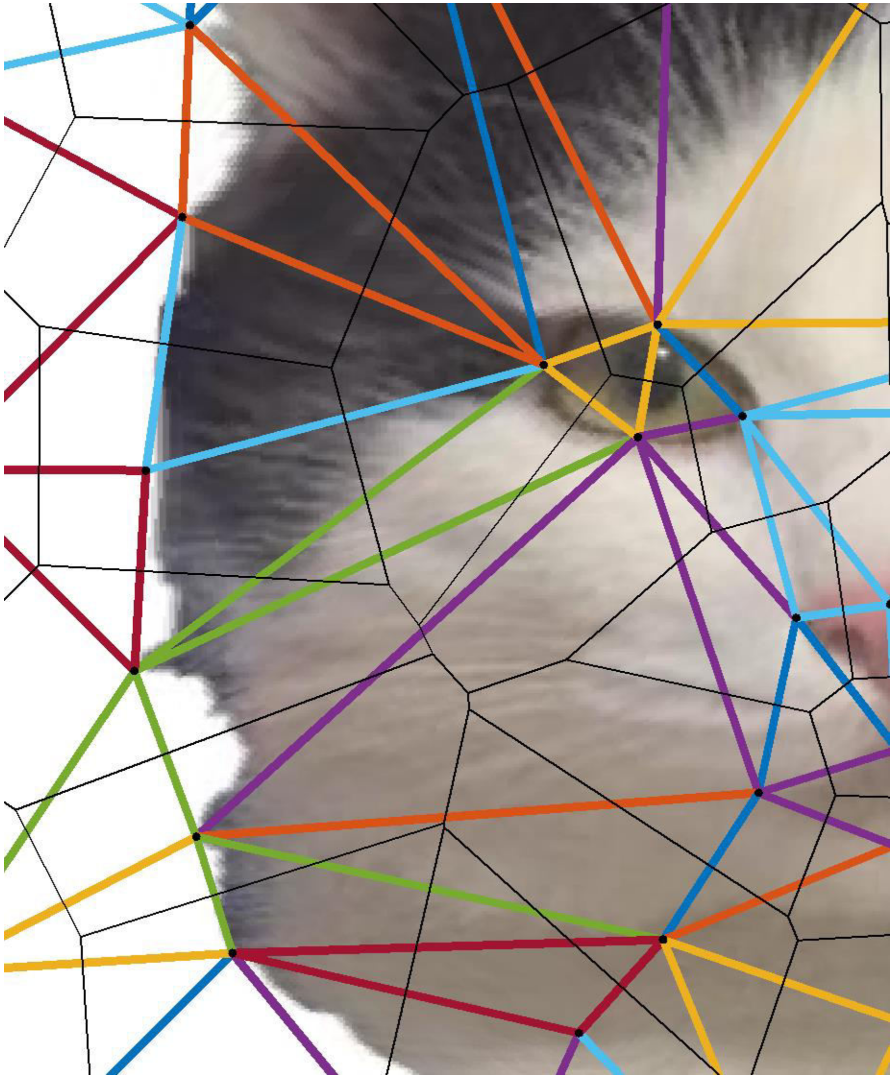
Video 7.1

Jianbo Shi

Image Morphing



Morphing = Object Averaging

"an average" between two objects

Not an average of two *images of objects*…

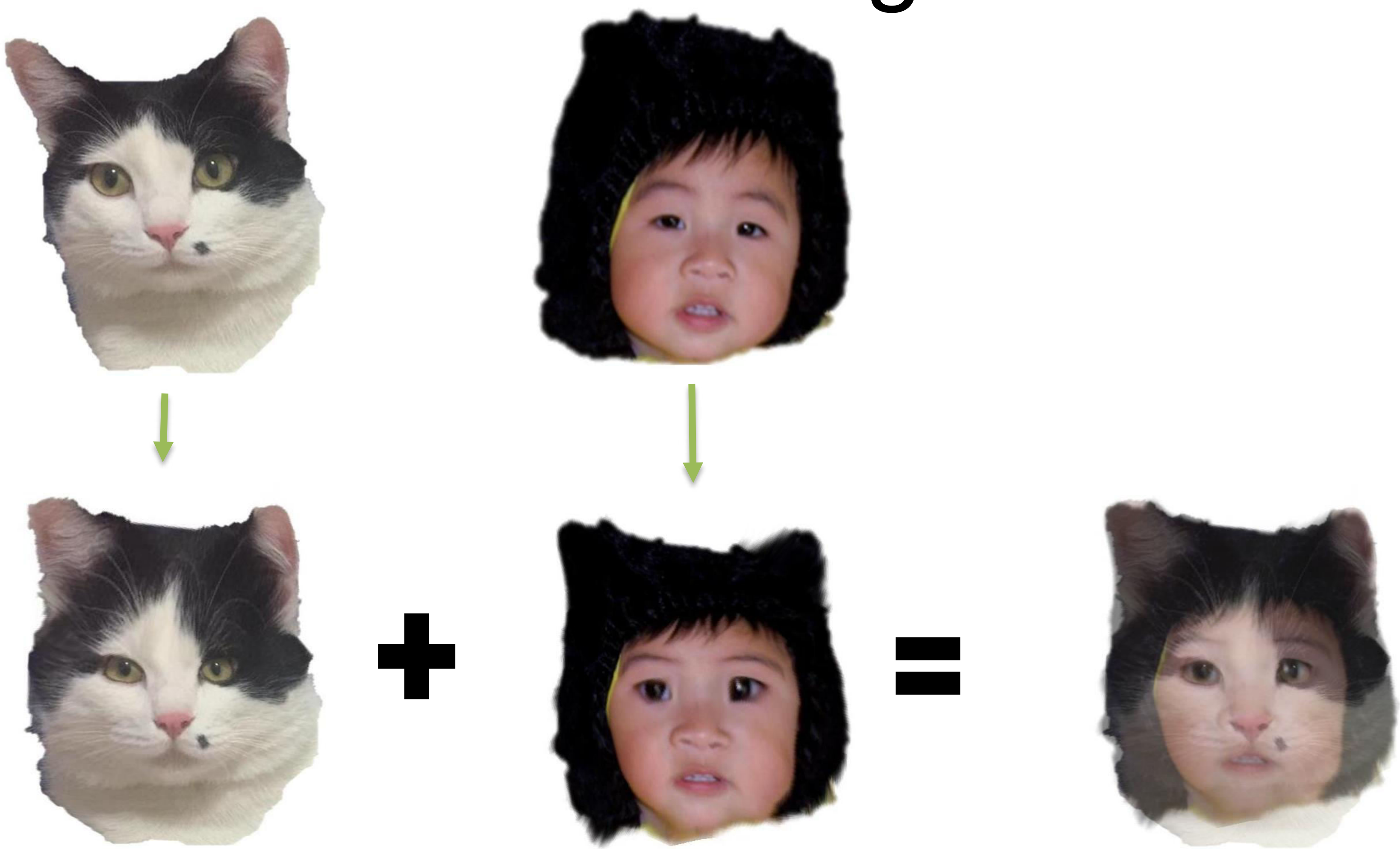
…but an image of the *average object*!

Morphing = Object Averaging

How do we know what the average object looks like?

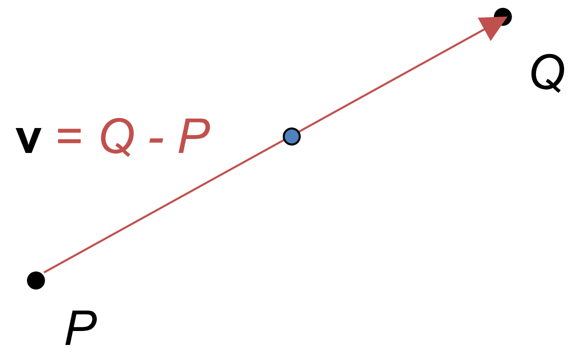
- We haven't a clue!
- But we can often fake something reasonable

Morphing = Warping + Cross Dissolving



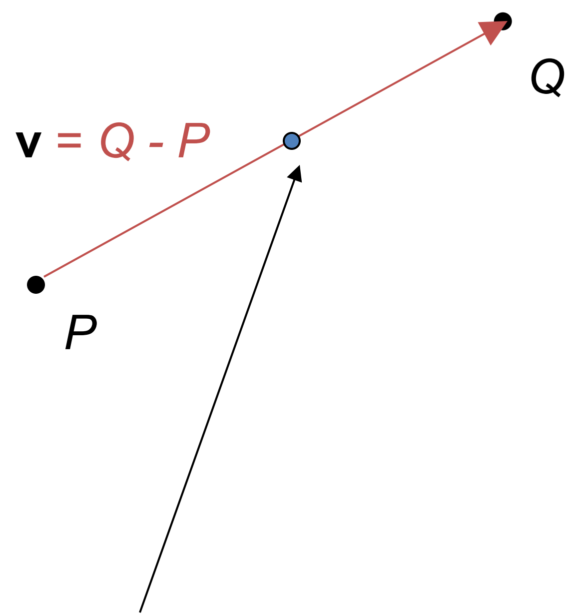
Averaging Points

What's the average of P and Q?



Averaging Points

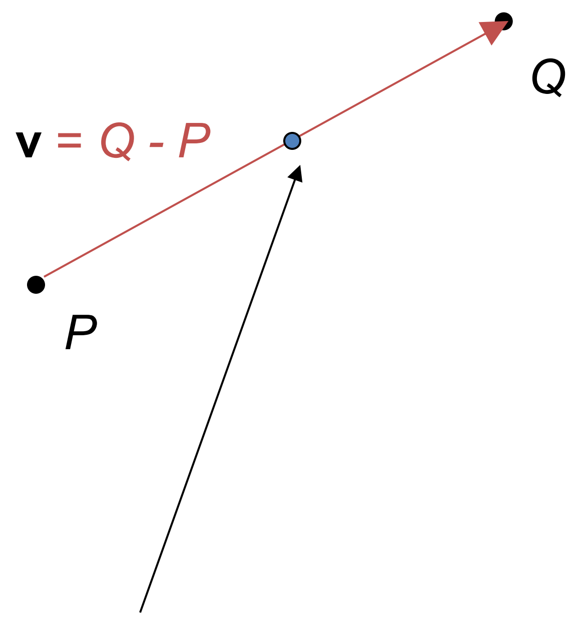
What's the average of P and Q?



$P + 0.5\mathbf{v}$
$= P + 0.5(Q - P)$
$= 0.5P + 0.5\,Q$

Averaging Points

What's the average of P and Q?



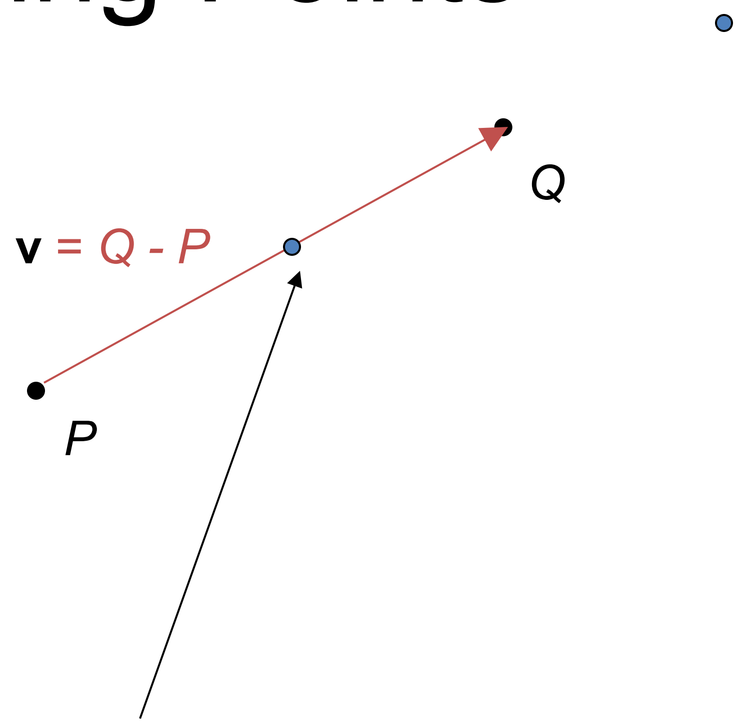
$P + 0.5\mathrm{v}$

$= P + 0.5(Q - P)$

$= 0.5P + 0.5Q$

Linear Interpolation (Affine Combination):
New point $aP + bQ$,
defined only when $a+b = 1$
So $aP+bQ = aP+(1-a)Q$

Averaging Points

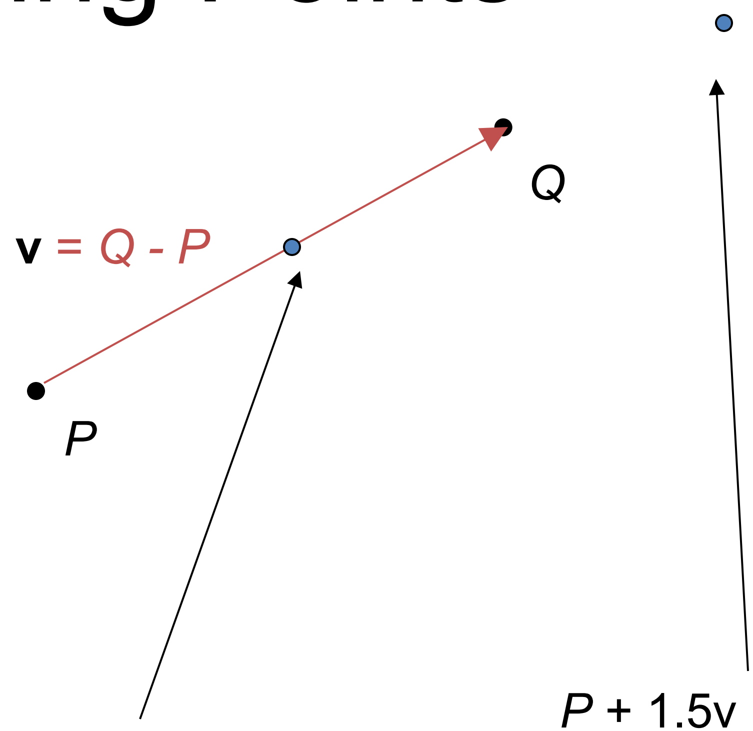


$P + 0.5\mathrm{v}$

$= P + 0.5(Q - P)$

$= 0.5P + 0.5\,Q$

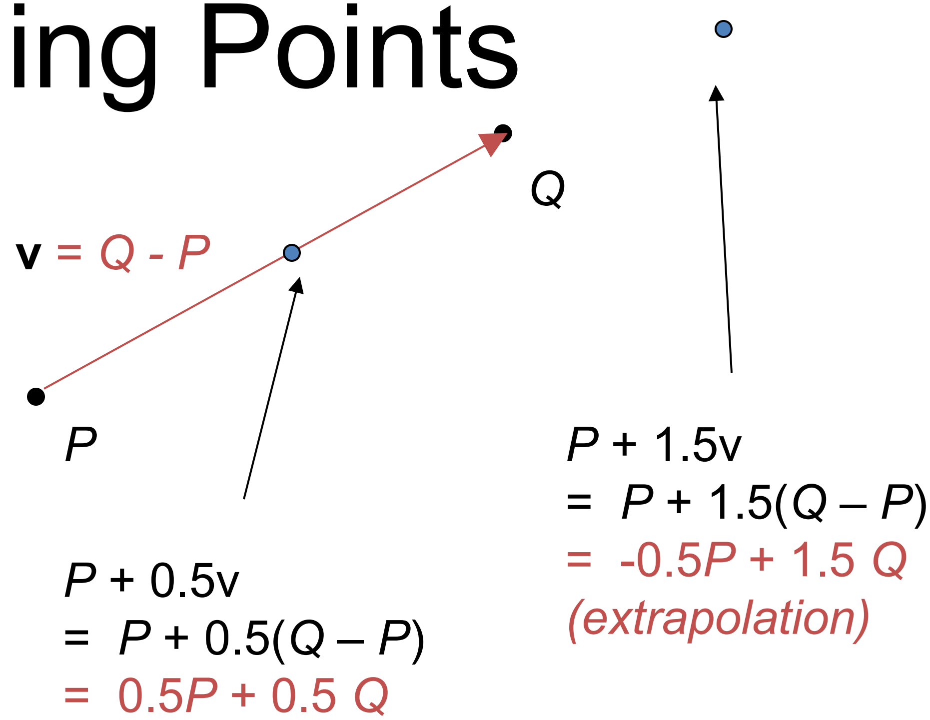
Averaging Points



$P + 0.5\mathrm{v}$
$= P + 0.5(Q - P)$
$= 0.5P + 0.5\,Q$

$P + 1.5\mathrm{v}$
$= P + 1.5(Q - P)$
$= -0.5P + 1.5\,Q$
(extrapolation)

Averaging Points



- P and Q can be anything:
  - points on a plane (2D) or in space (3D)
  - Colors in RGB or HSV (3D)
  - Whole images (m-by-n D)… etc.

Averaging Images: Cross-Dissolve

Interpolate whole images:

$$\text{Image}_{\text{halfway}} = (1\text{-}t)*\text{Image}_1 + t*\text{image}_2$$

This is called **cross-dissolve** in film industry

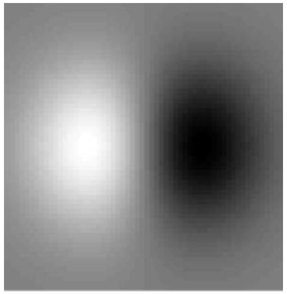
Averaging Images: Cross-Dissolve

Interpolate whole images:

$$\text{Image}_{\text{halfway}} = (1\text{-}t)^*\text{Image}_1 + t^*\text{image}_2$$

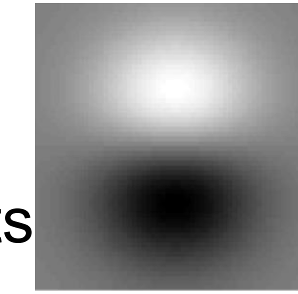
This is called **cross-dissolve** in film industry

Averaging Images



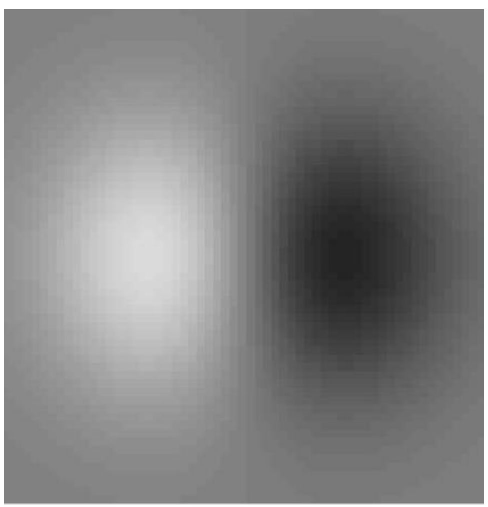
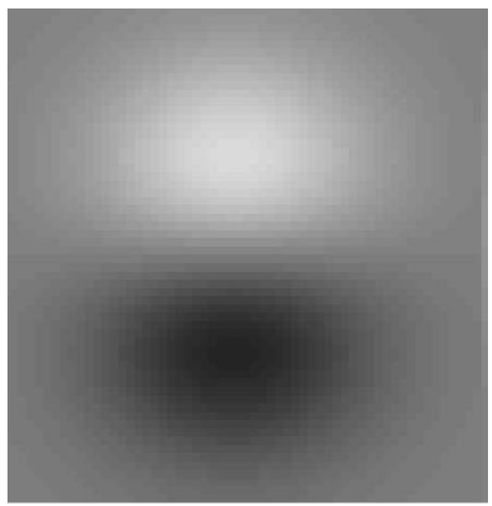
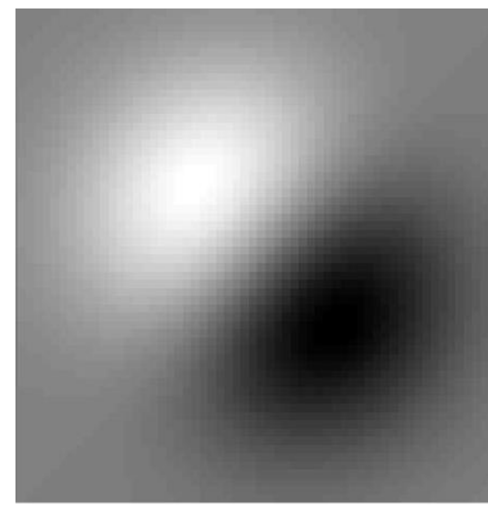
Image$_1$

Averaging Images = Rotating Objects



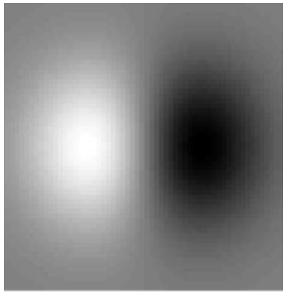
Image$_2$

t = 0 t = 0.5 t = 1

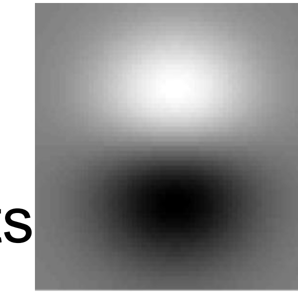
 +  = 

(1-t)*Image$_1$ + t*Image$_2$ = Image$_{halfway}$

Averaging Images

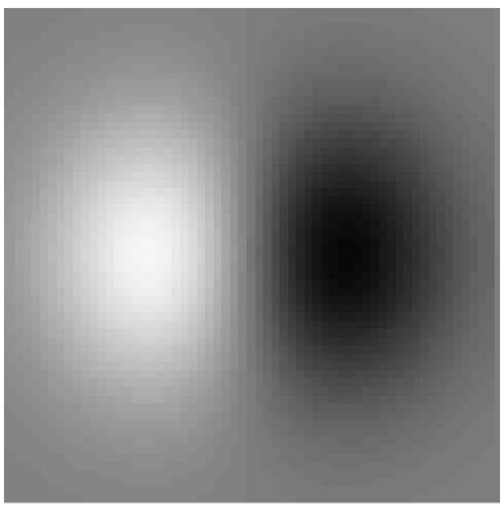
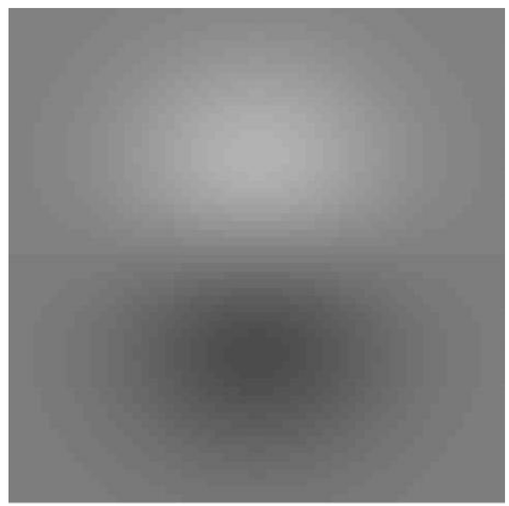
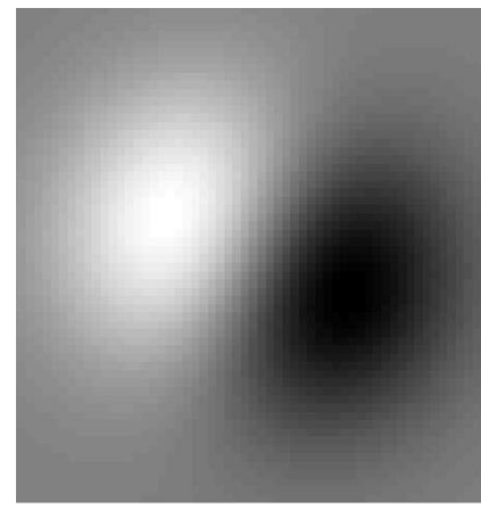


Averaging Images = Rotating Objects

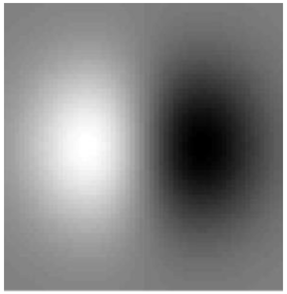


Image$_1$ Image$_2$

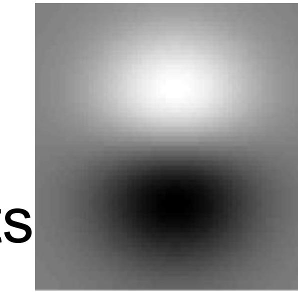
t = 0 t = 0.3 t = 1

 +  = 

(1-t)*Image$_1$ + t*Image$_2$ = Image$_{halfway}$



Averaging Images



Averaging Images = Rotating Objects

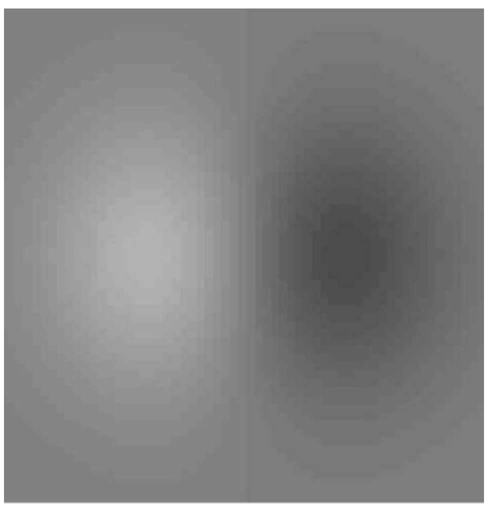
Image$_1$

Image$_2$

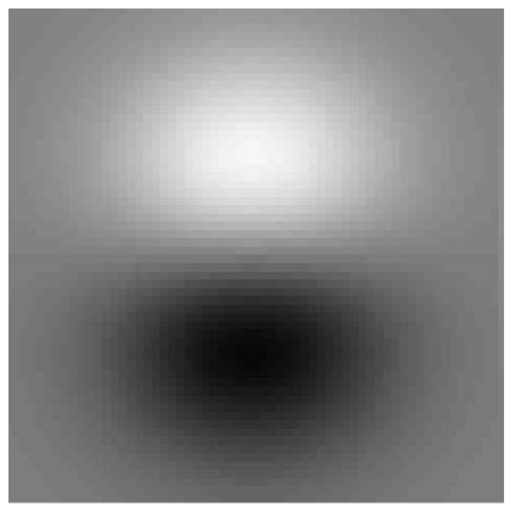
t = 0

t = 0.7

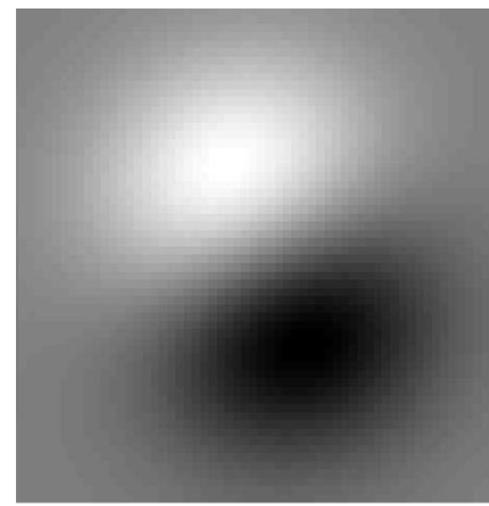
t = 1



+



=



(1-t)*Image$_1$ + t*Image$_2$ = Image$_{halfway}$

Averaging Images

Image_1 Image_2

+

= ?

Averaging Images

Image$_1$ Image$_2$

+

?

=

Averaging Images

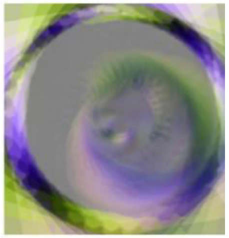
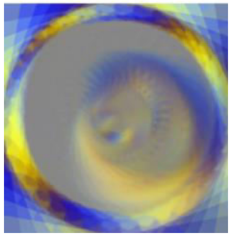
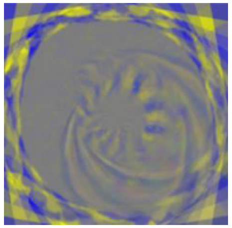
Image_1 Image_2

+

=

Averaging Images != Rotating Complex Obje

Averaging Images

 +  +  = 

Averaging ‘Eigen’ Images = Rotating Objects

Cat-Baby Averaging

Object Averaging with feature matching!

- Nose to nose, eye to eye, mouth to mouth, etc.
- This is a non-parametric **warp**

Cat-Baby Averaging

Object Averaging with feature matching (warping)!

- Nose to nose, eye to eye, mouth to mouth, etc.
- This is a non-parametric warp

Warping, then cross-dissolve

Morphing procedure:

1. Find the average shape
2. Non-parametric warping
3. Find the average color
   – Cross-dissolve the warped images

Video 7.2
Jianbo Shi

Image warping – non-parametric

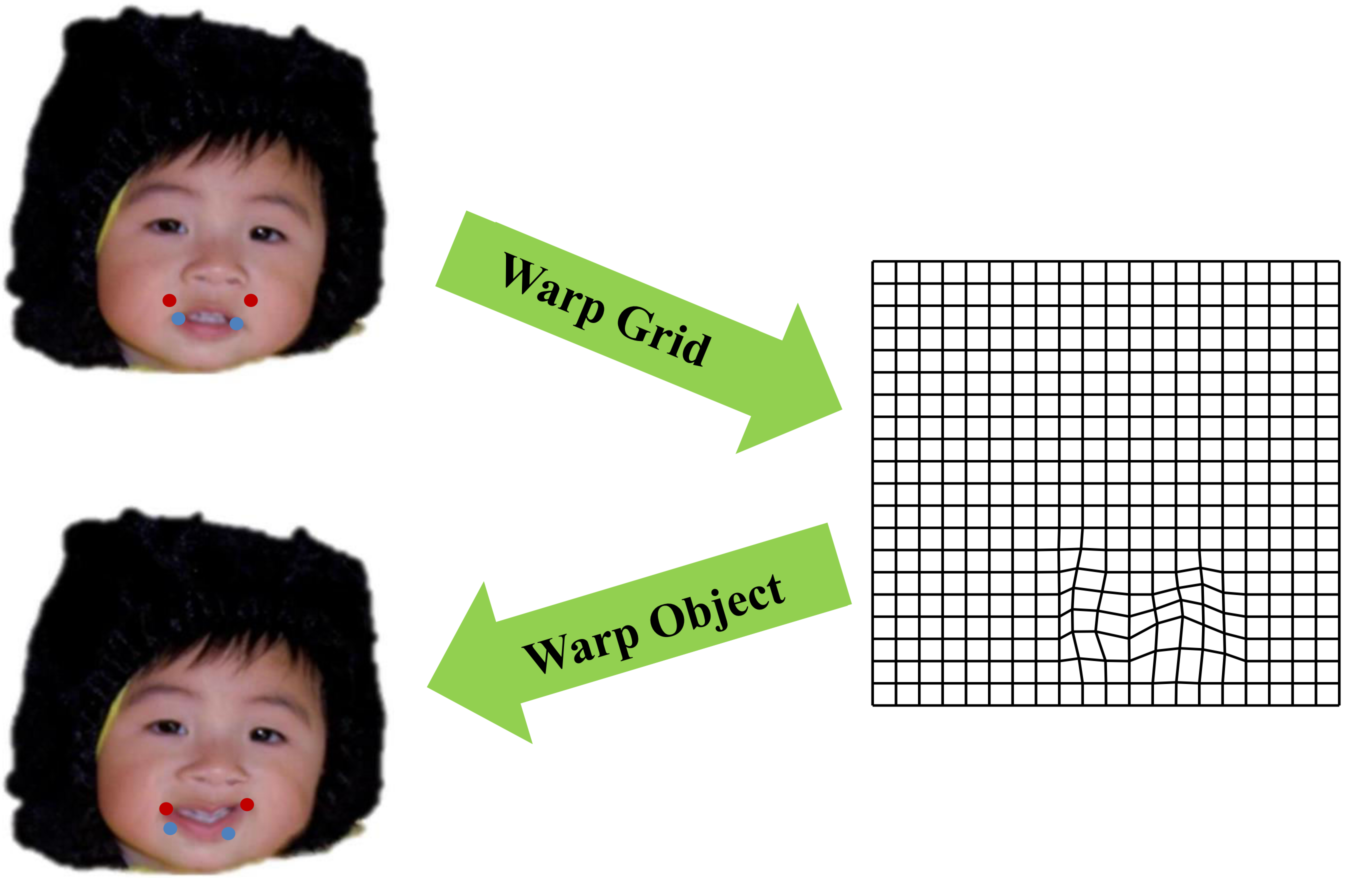
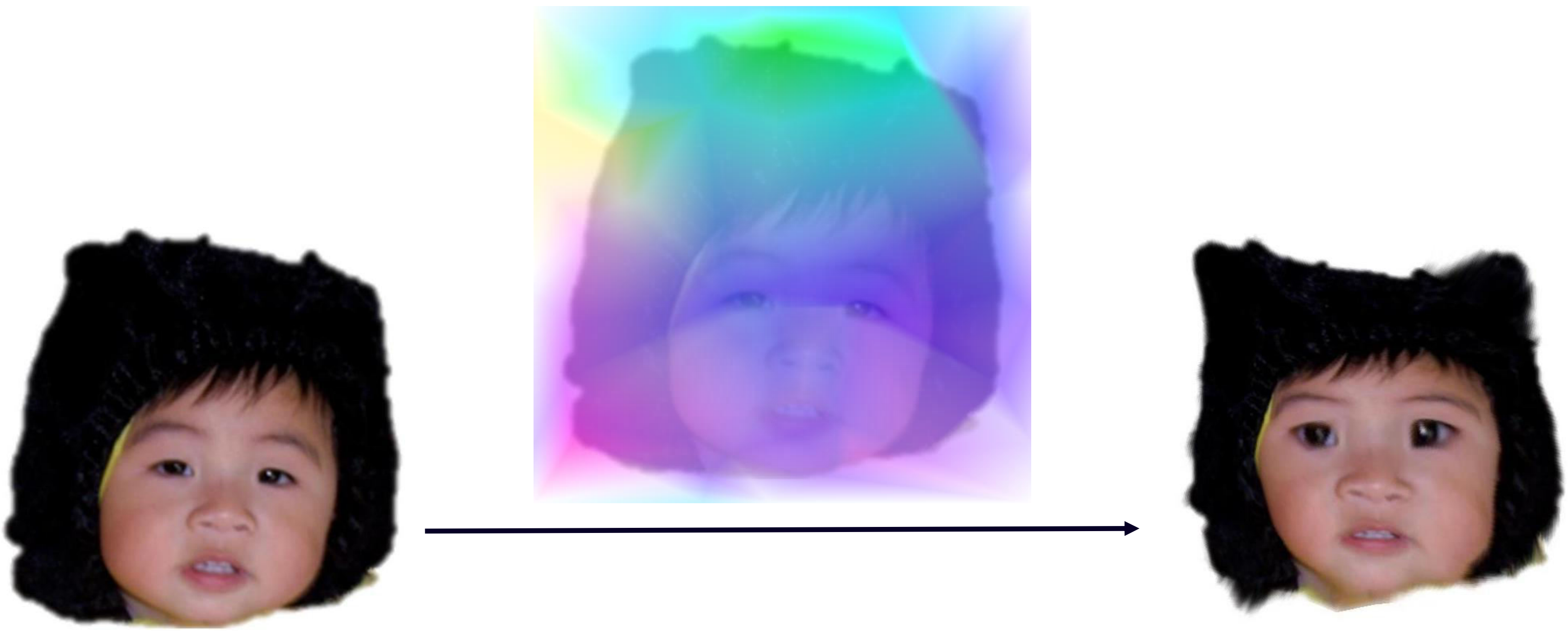


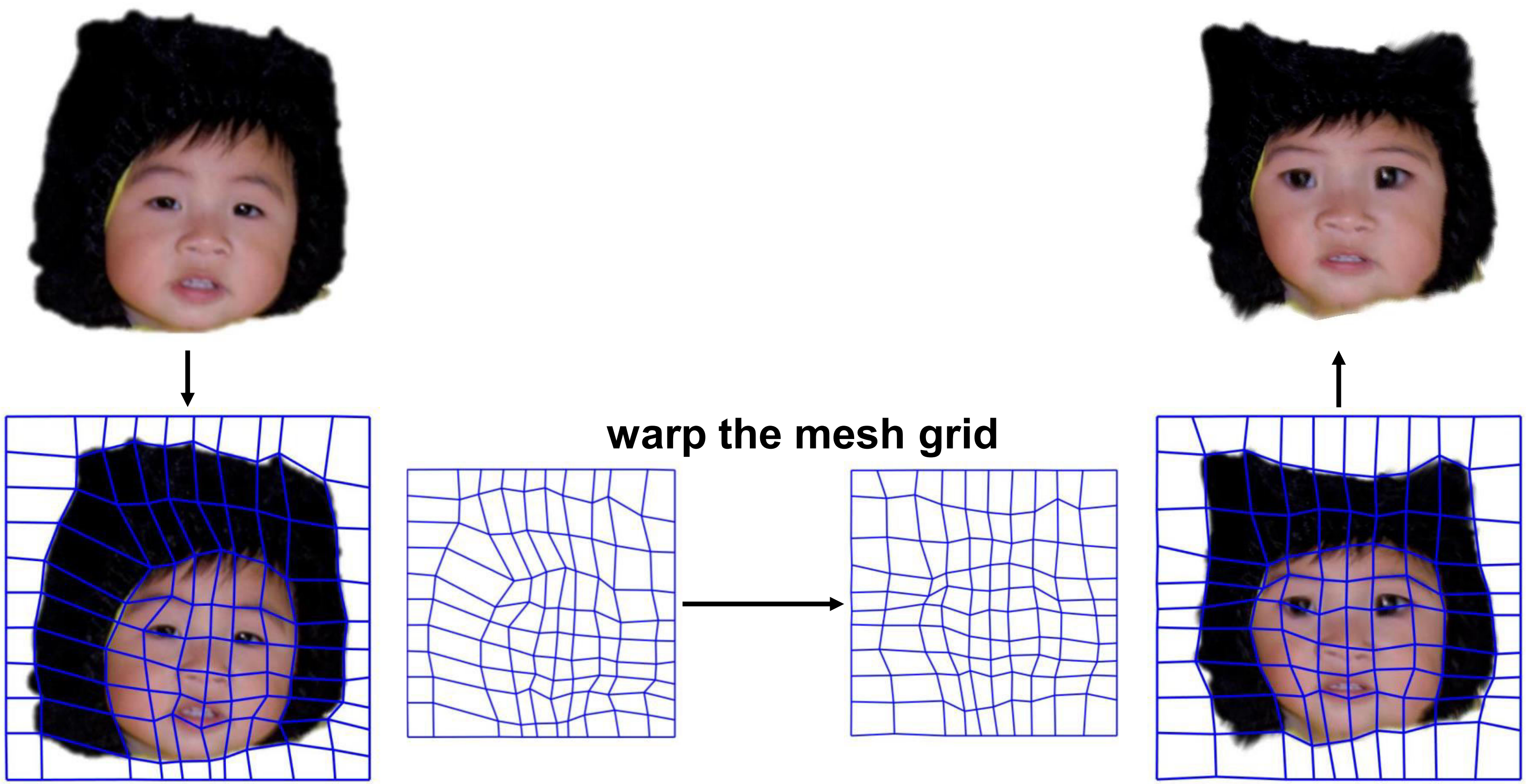
Image warping idea 1: dense flow



Displacement vector (u,v) for each pixel.

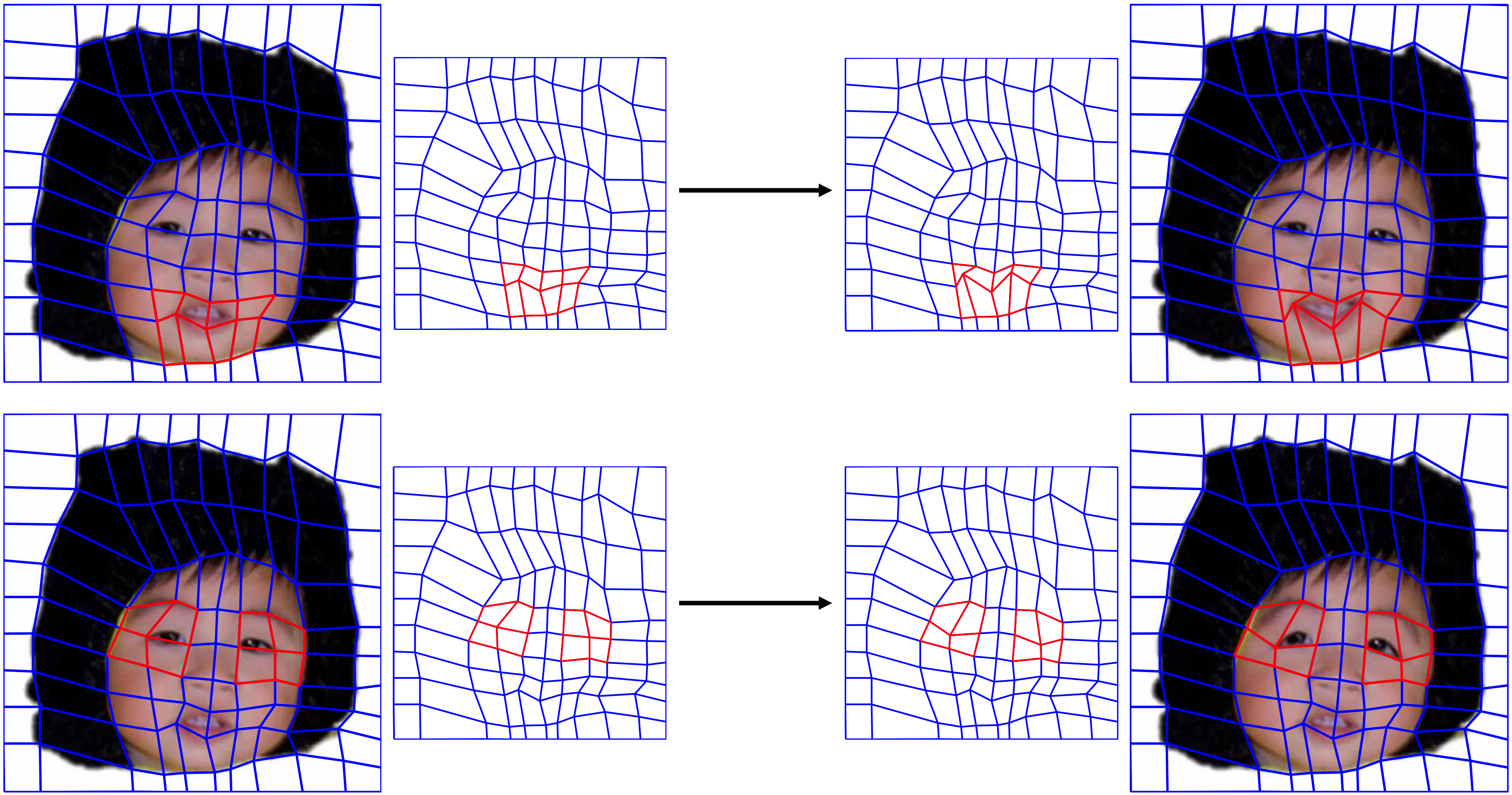
Great details… but too much work, let’s simply it to mesh grid

Image warping idea 2 : dense grid



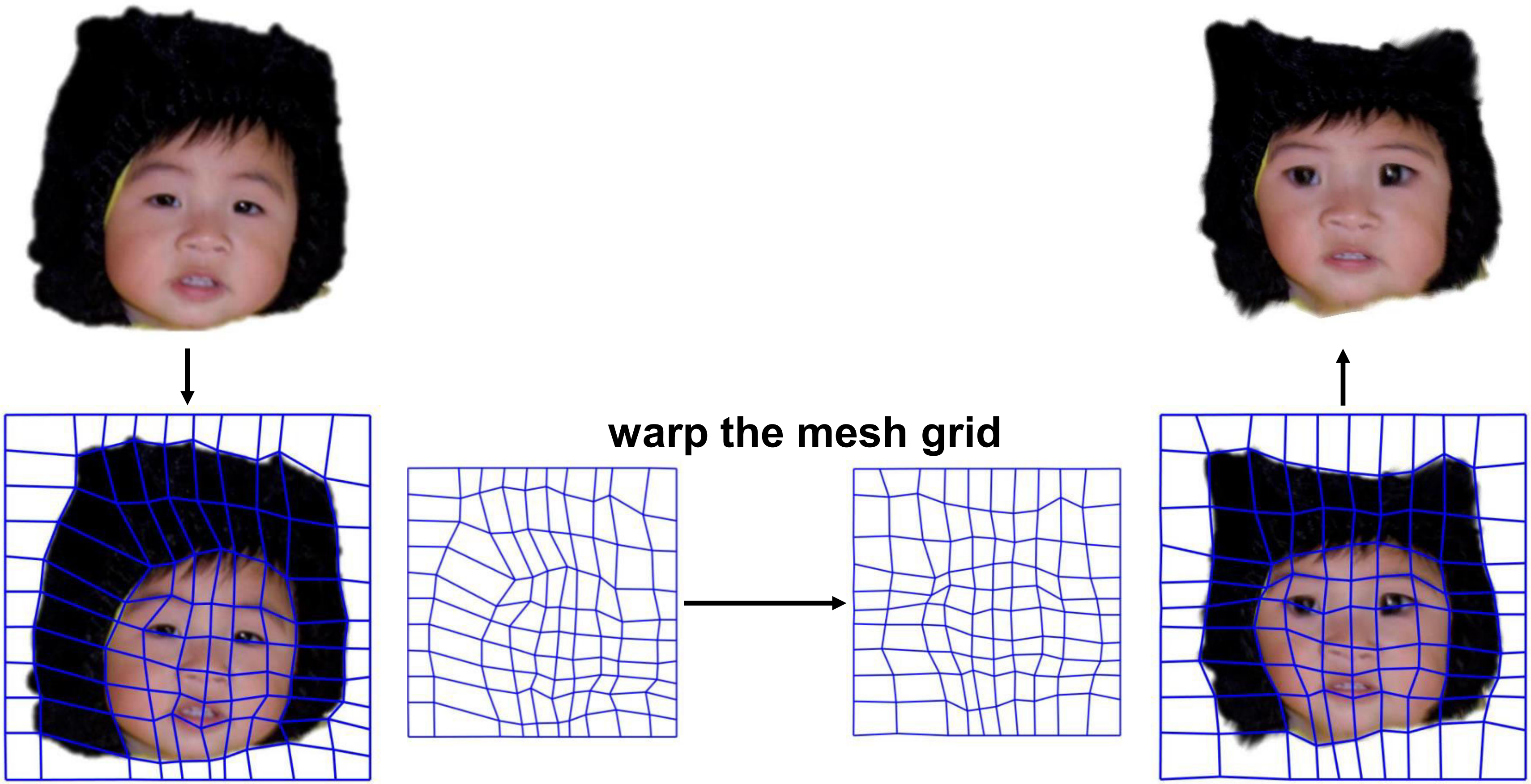
Define and manipulate the mesh grid

Image warping idea 2 : dense grid



Grid deformation generates expression change

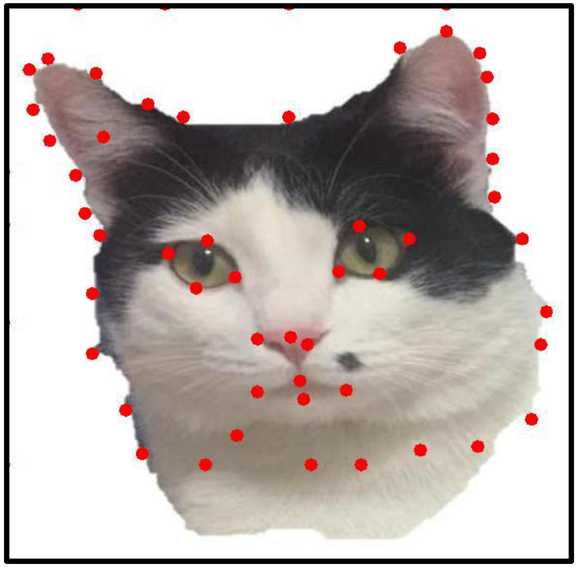
Image warping idea 2 : dense grid

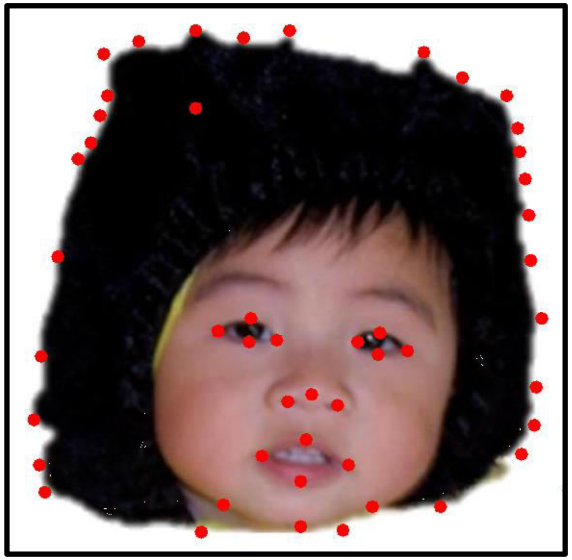


Still too much work…

simplify it to sparse control points and triangles

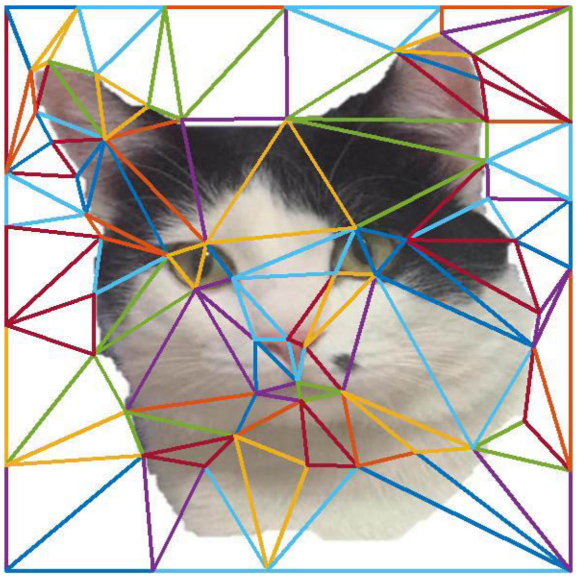
Image warping idea 3 : sparse points

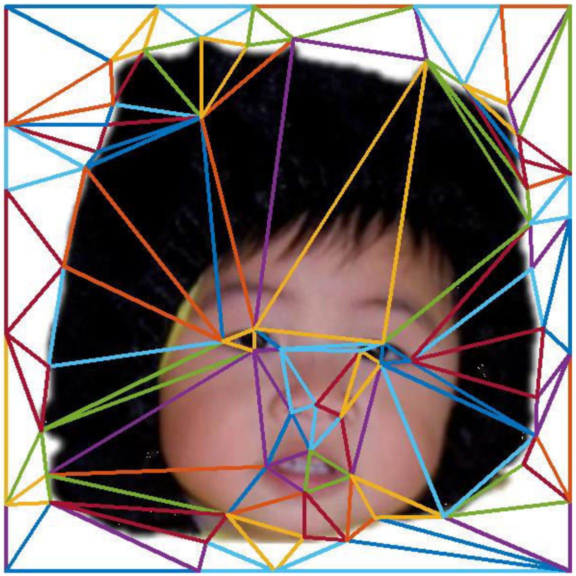




Specify sparse points and their correspondence

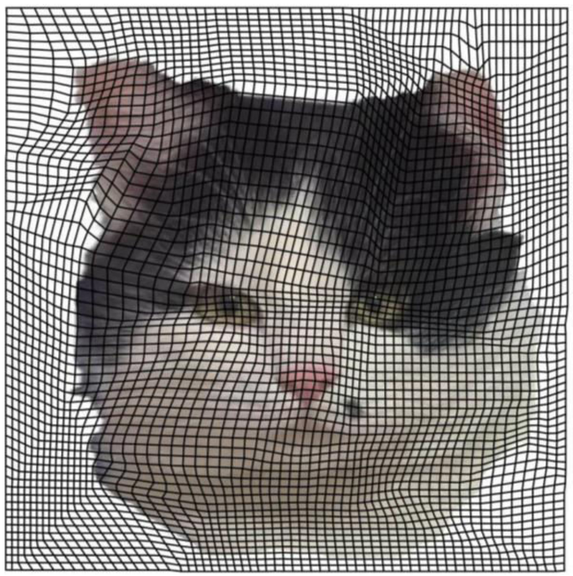
Image warping idea 3 : sparse points

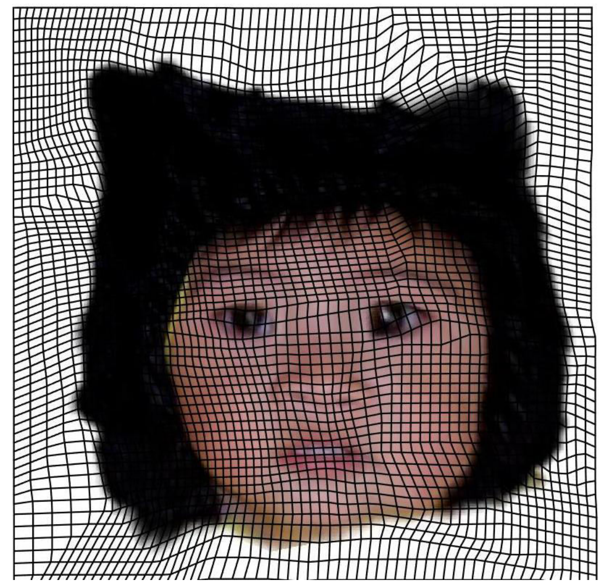




- Define a triangular mesh over the feature points
- Triangle-to-triangle correspondences
- Warp each triangle separately from source to destination

From sparse points to dense grid

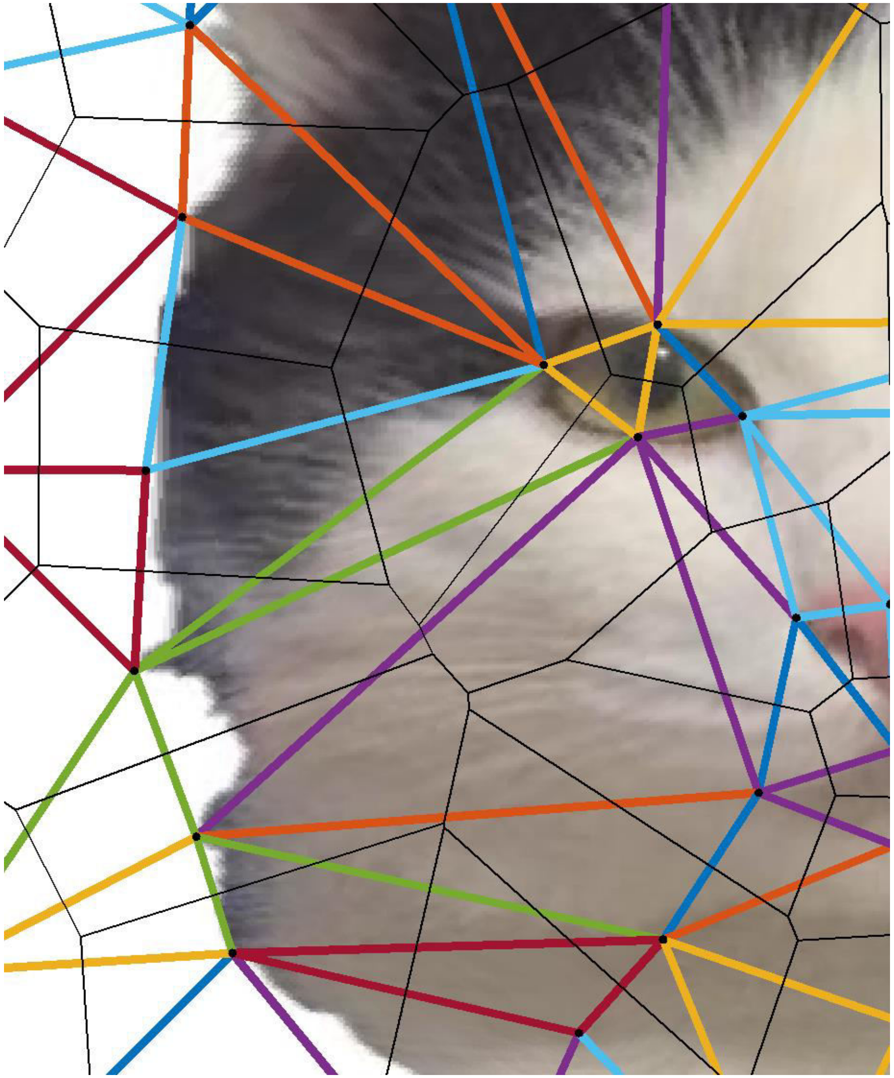




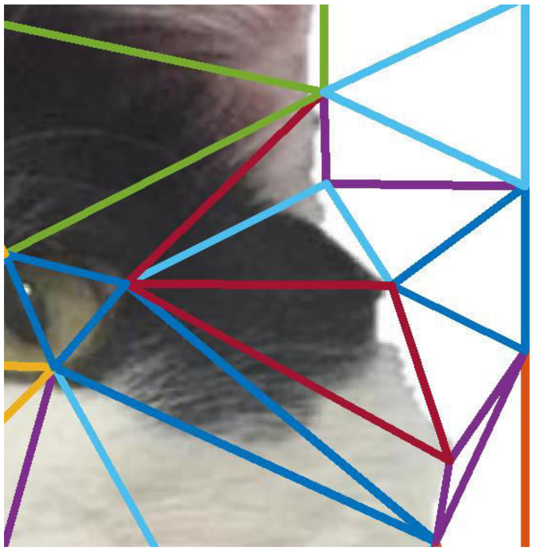
- Warping on triangulation corresponds to warping on dense grid, and dense pixel flow

Delaunay Triangulation

- Draw the dual to the Voronoi diagram by connecting each two neighboring sites in the Voronoi diagram.
- The DT may be constructed in O(nlogn) time.
- This is what Matlab's <u>delaunay</u> function uses.



What is good feature points



Good

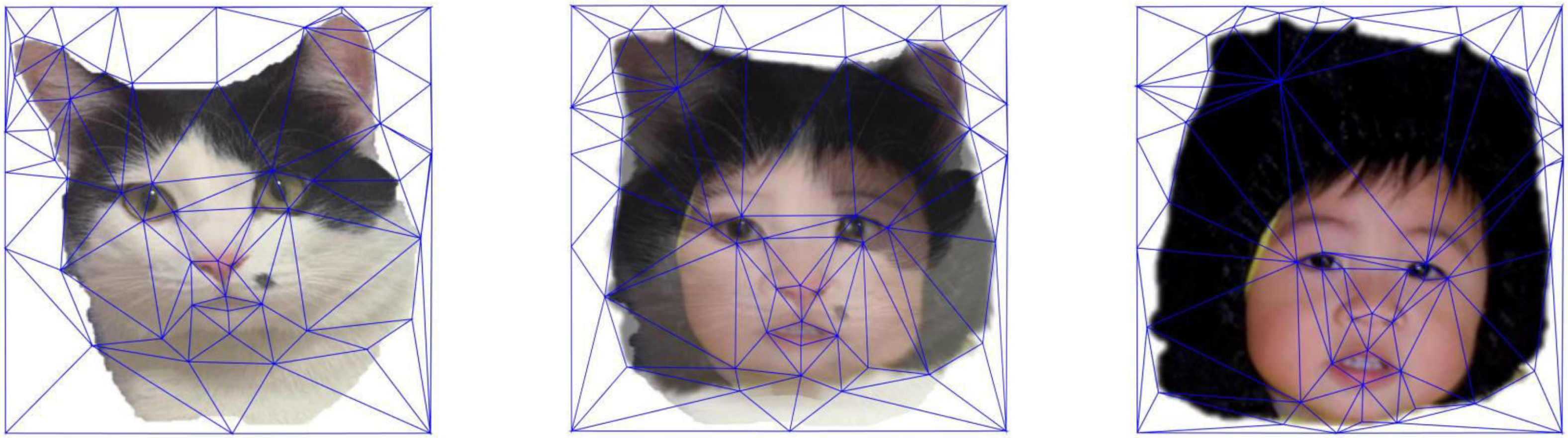
Bad

- The triangulation is consistent with image boundary
  - Texture regions won't fade into the background when morphing
- Maintain the relationship between parts

Video 7.3

Jianbo Shi

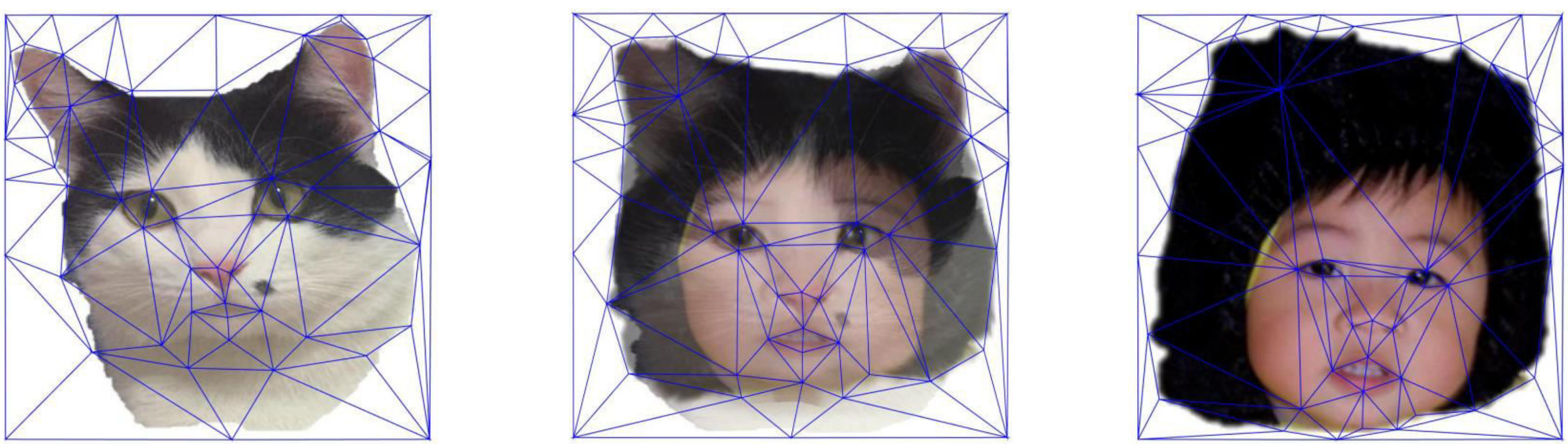
Triangular Mesh



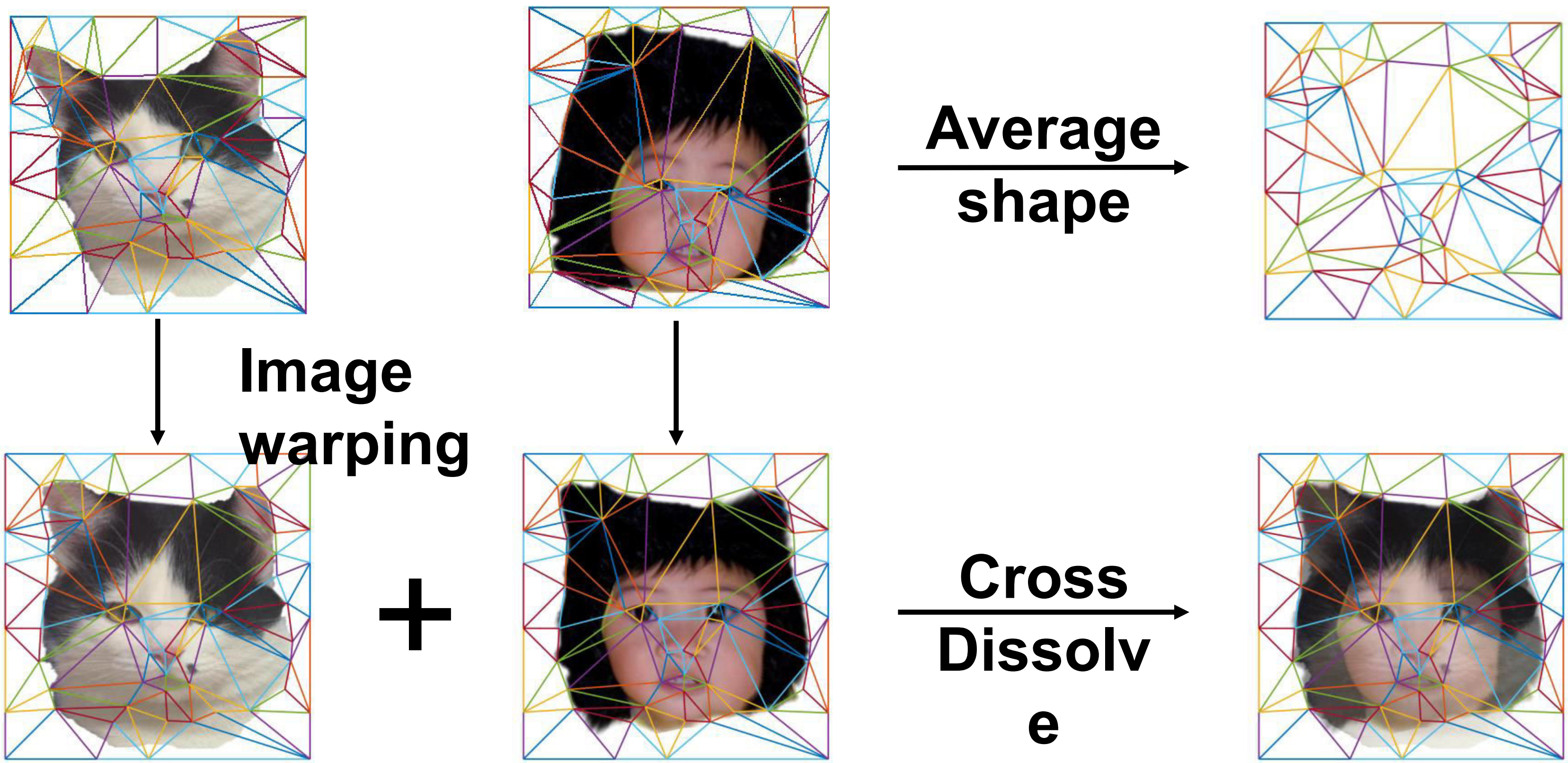
1. Input correspondences at key feature points
2. Define a triangular mesh over the points
   - Same mesh in both images!
   - Now we have triangle-to-triangle

Warp interpolation

- How do we create an intermediate warp at time t?
  - Assume t = [0,1]
  - Simple linear interpolation of each feature pair

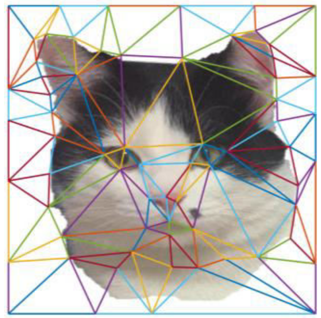


Morphing = Warping + Averaging

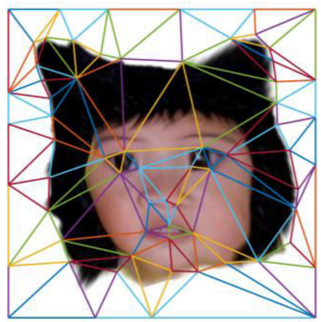


Morphing Sequence

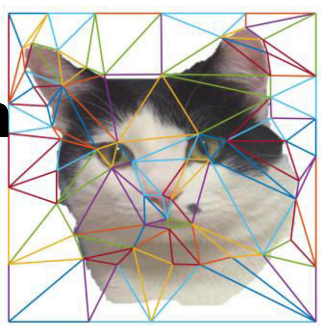
warped
image 1



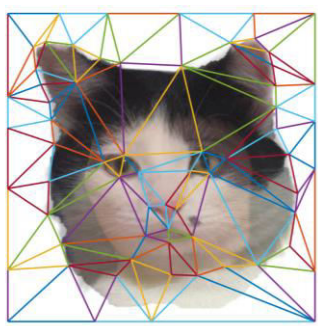
warped
image 2



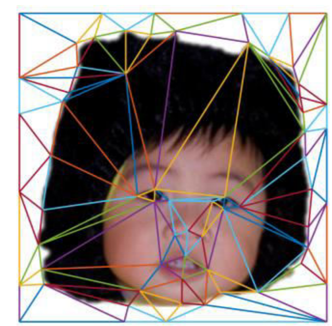
morph
result



t=0



t=0.3

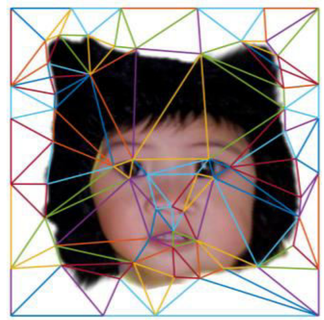


t=1

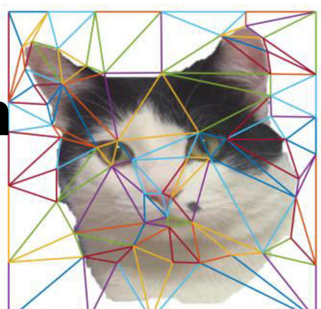
Morphing Sequence

warped image 1

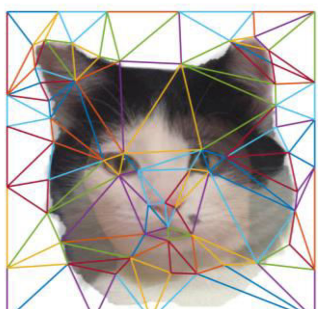
warped image 2



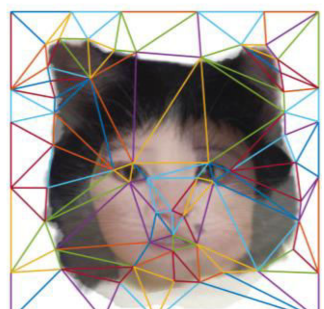
morph result



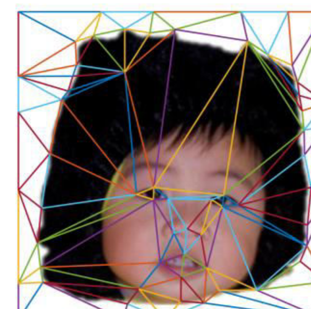
t=0



t=0.3



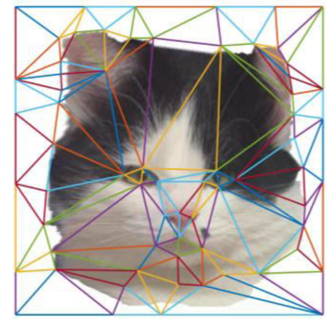
t=0.5



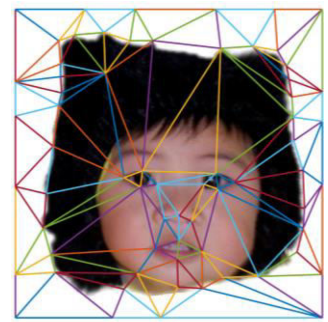
t=1

Morphing Sequence

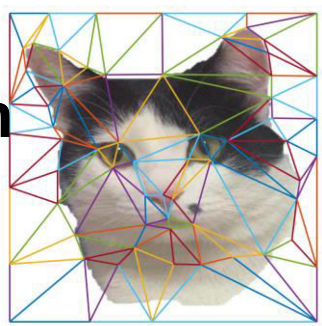
warped image 1



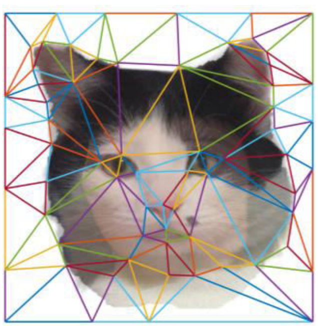
warped image 2



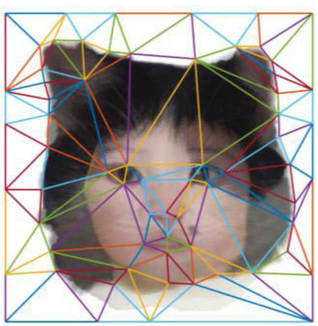
morph result



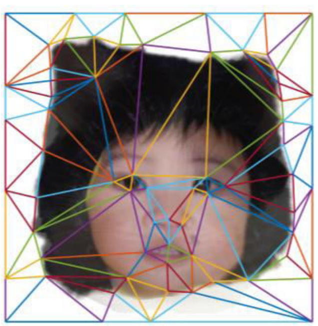
t=0



t=0.3



t=0.5



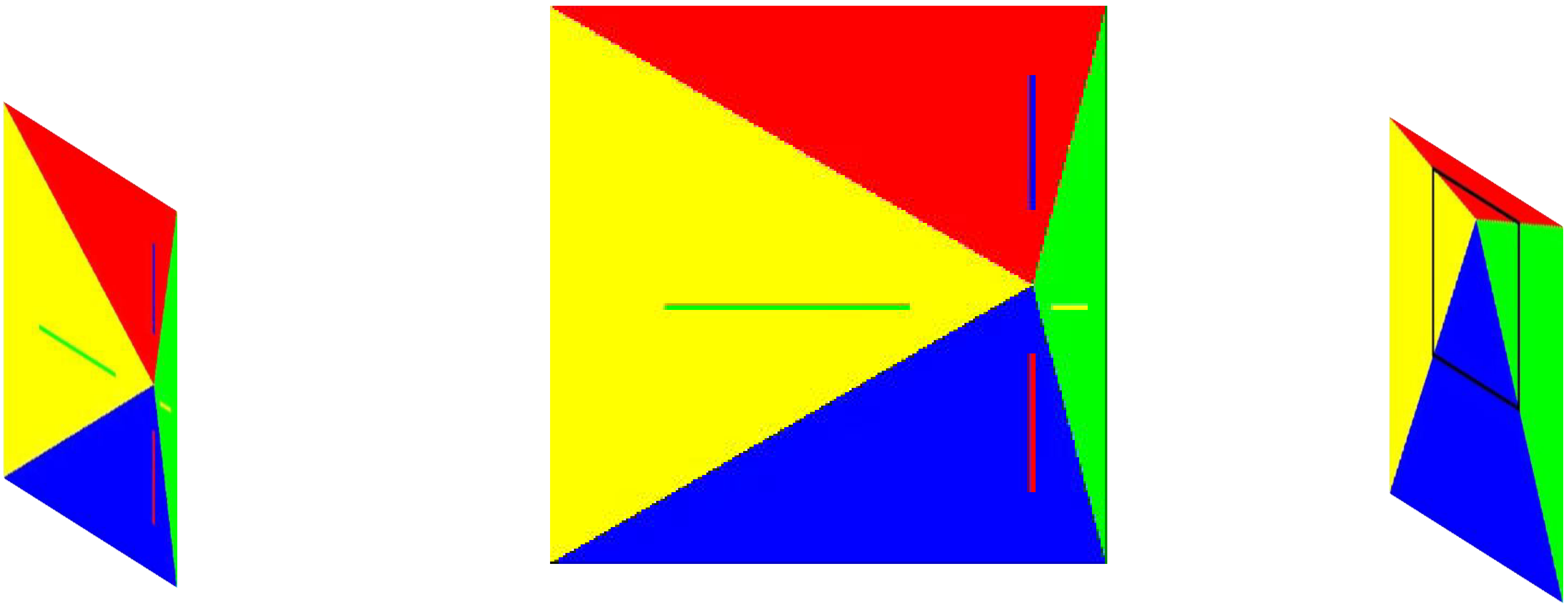
t=0.7

t=1

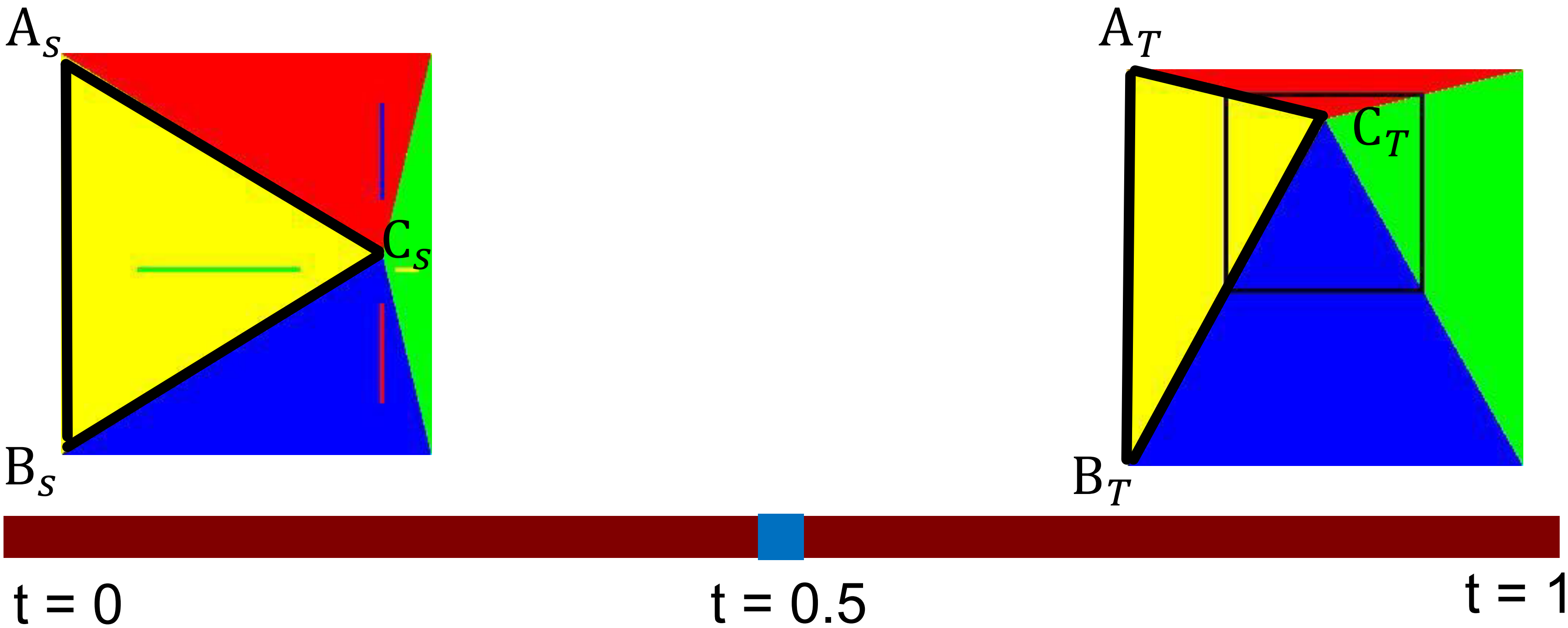
Morphing = Object Averaging

Video 7.4
Jianbo Shi

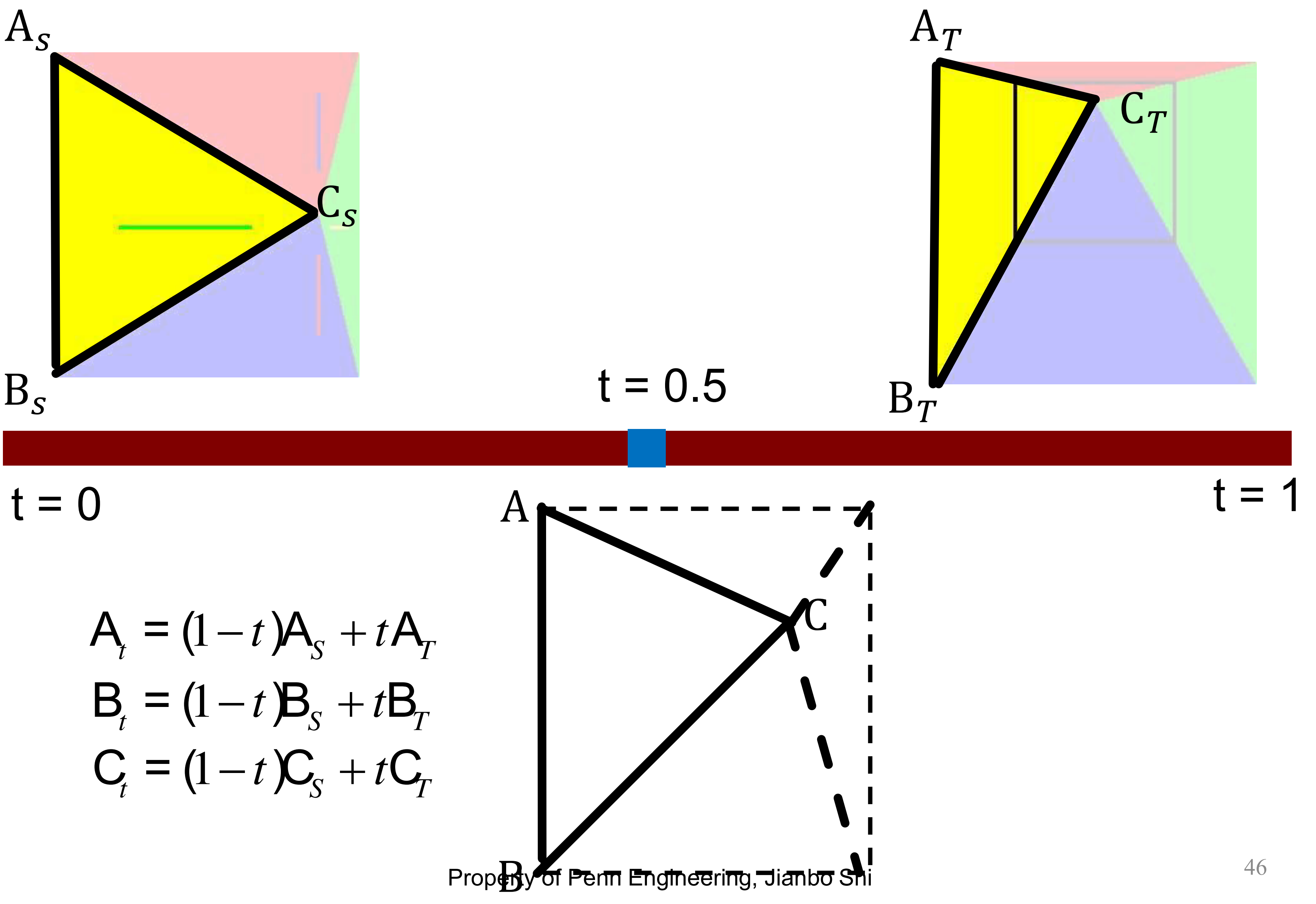
An Example



Morphing



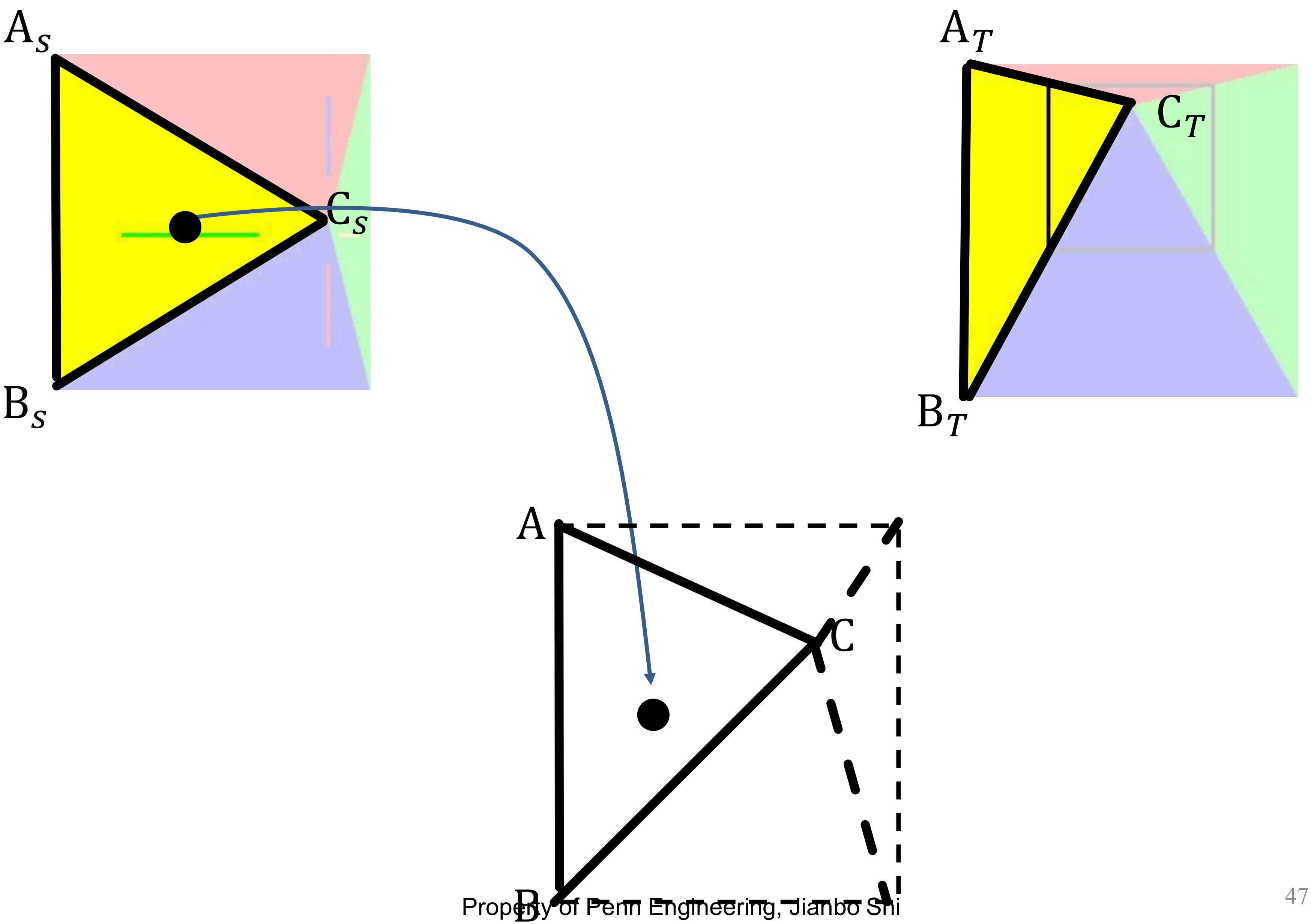
Step 1: Triangle interpolation



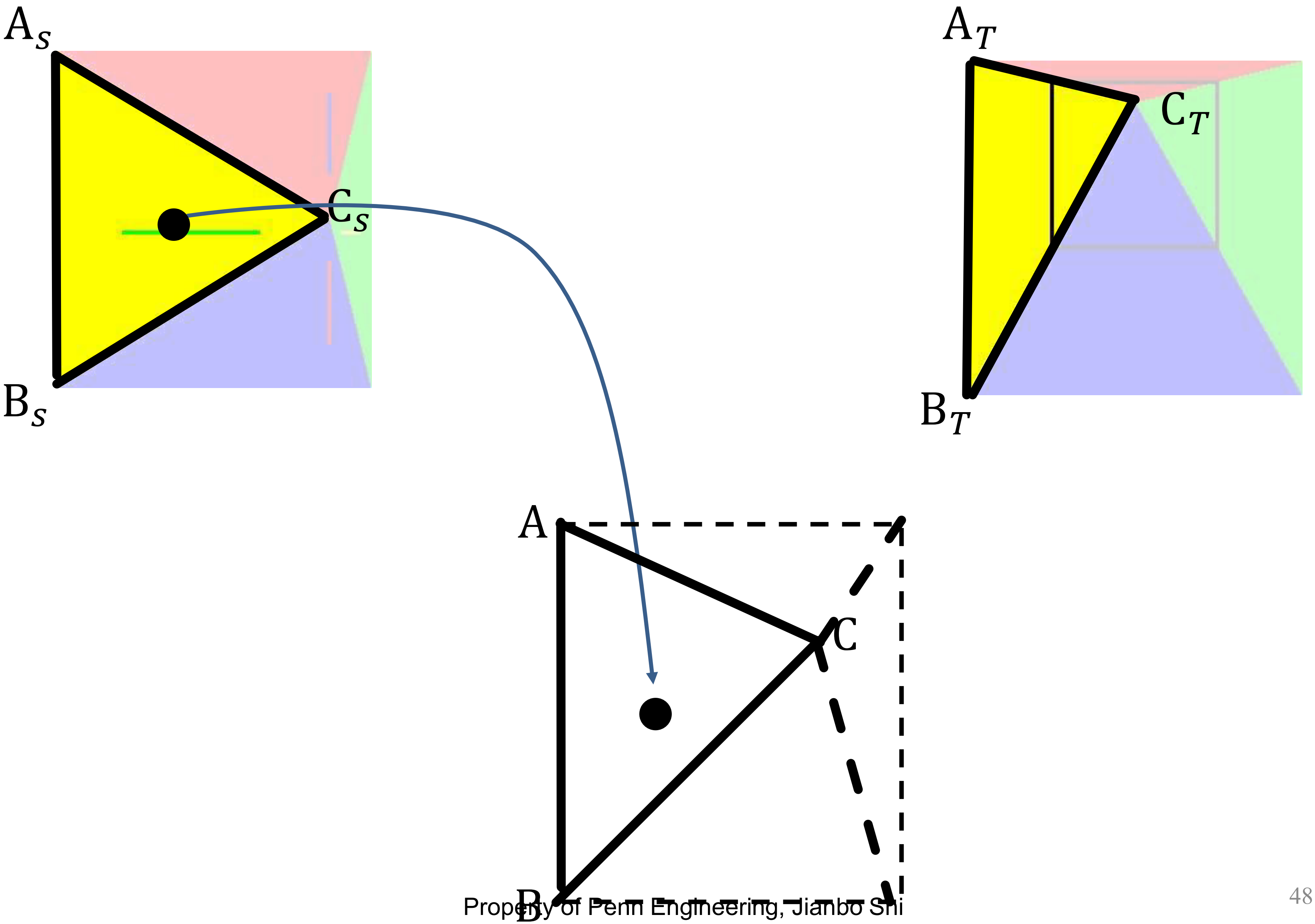
$$A_t = (1-t)A_S + tA_T$$
$$B_t = (1-t)B_S + tB_T$$
$$C_t = (1-t)C_S + tC_T$$

Property of Penn Engineering, Jianbo Shi

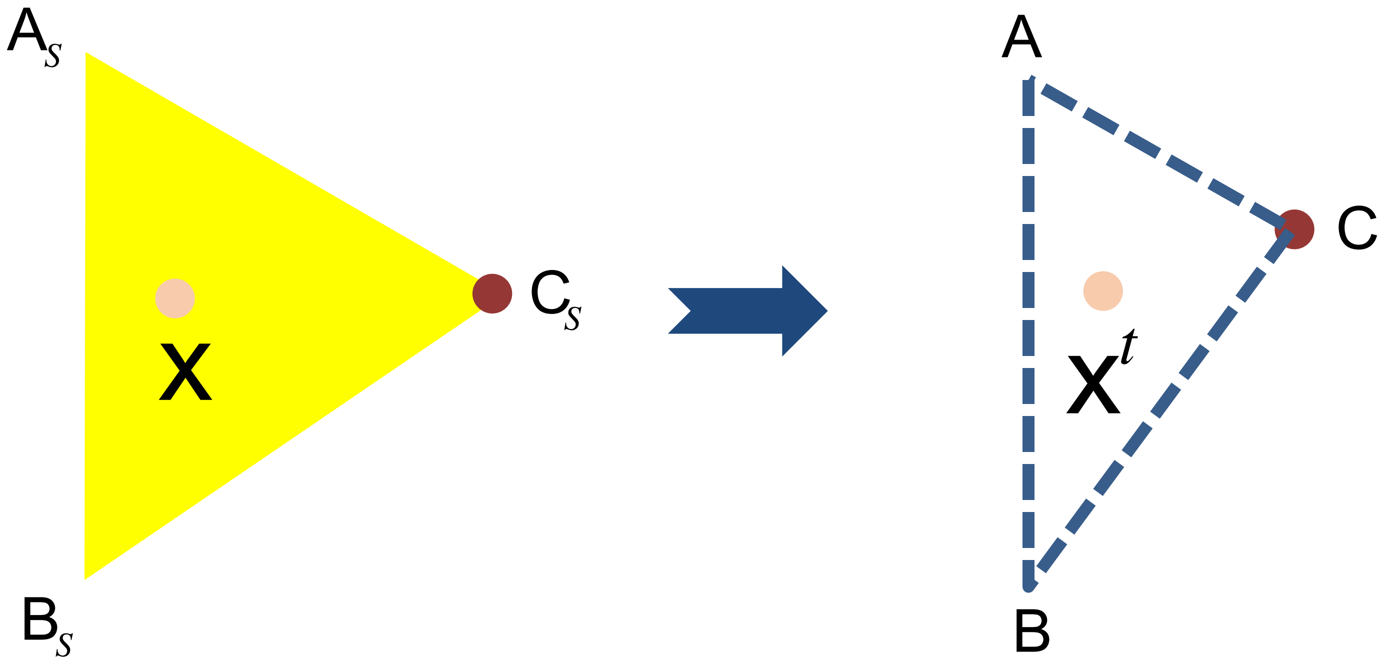
Step 2: Warping



Step 2: Warping

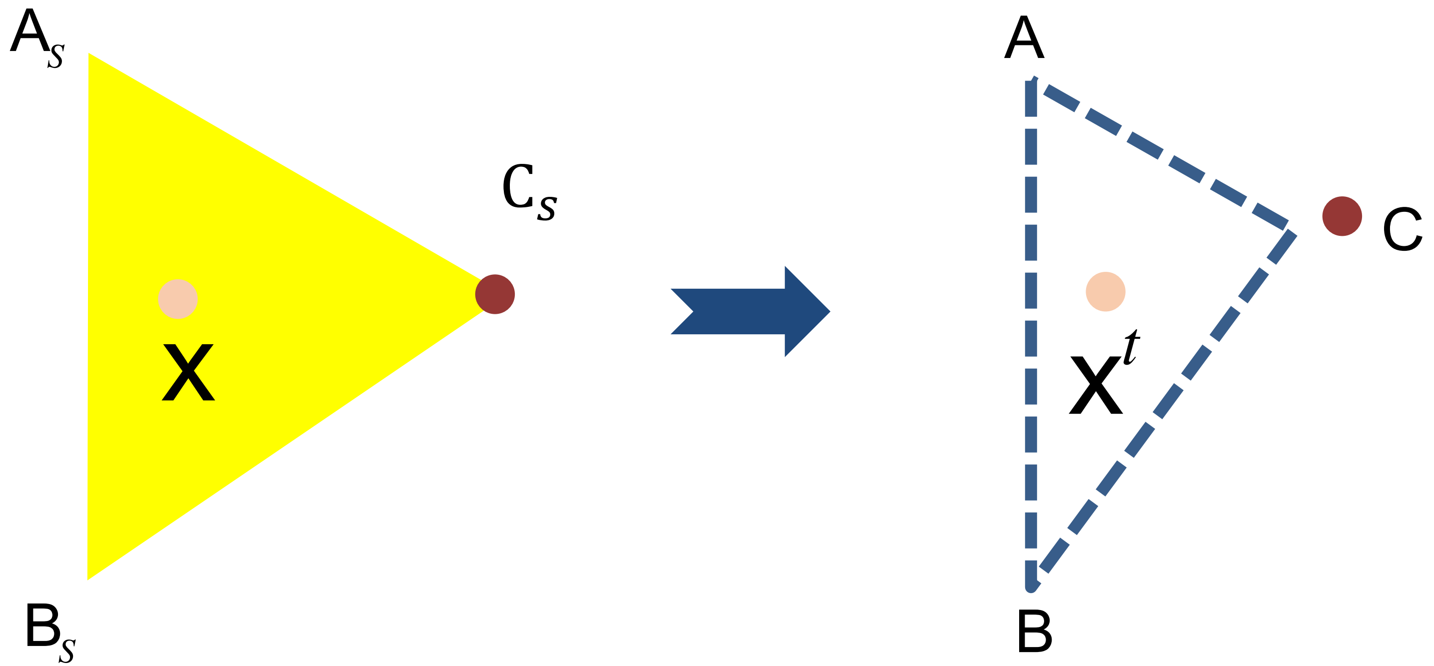


Triangle warping = Affine transform



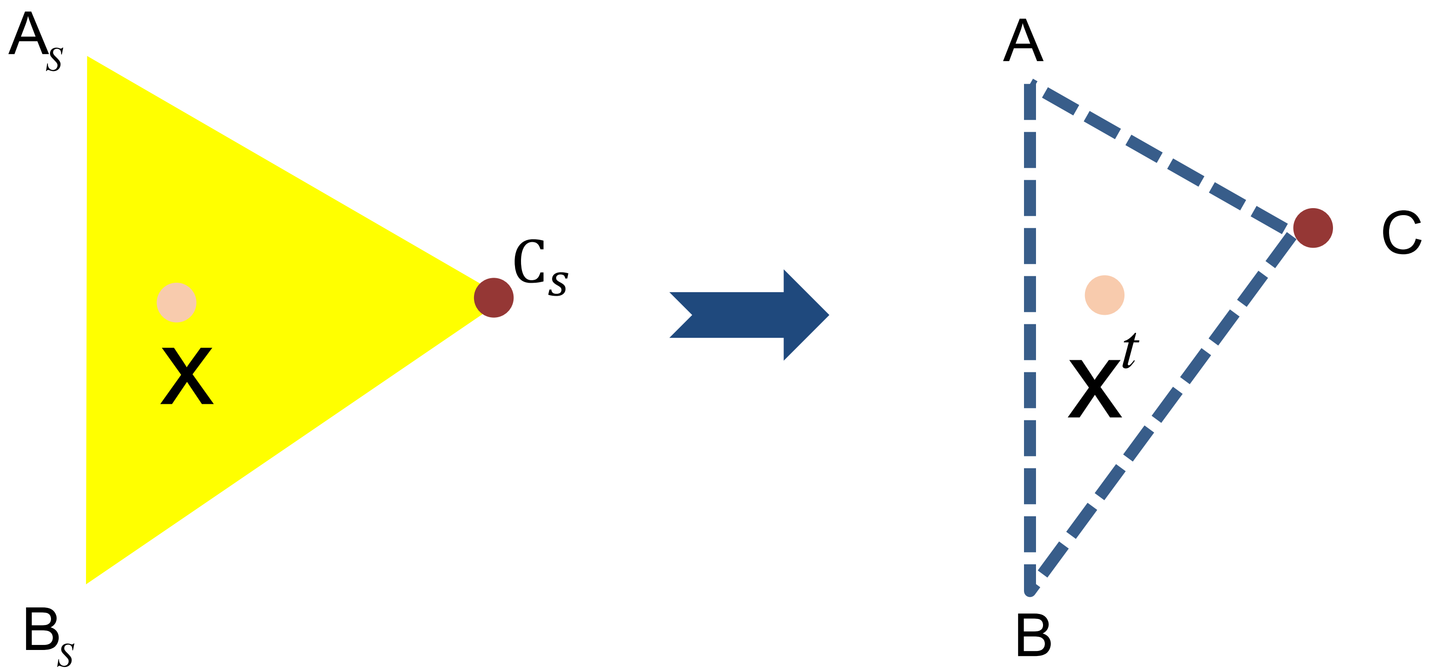
Affine transform is a pixel transportation $X \rightarrow X^t$
It is controlled by the movement of the three vertices of the triangle

Barycentric Coordinates



Each point X has an invariant representation w.r.t. the three vertices.

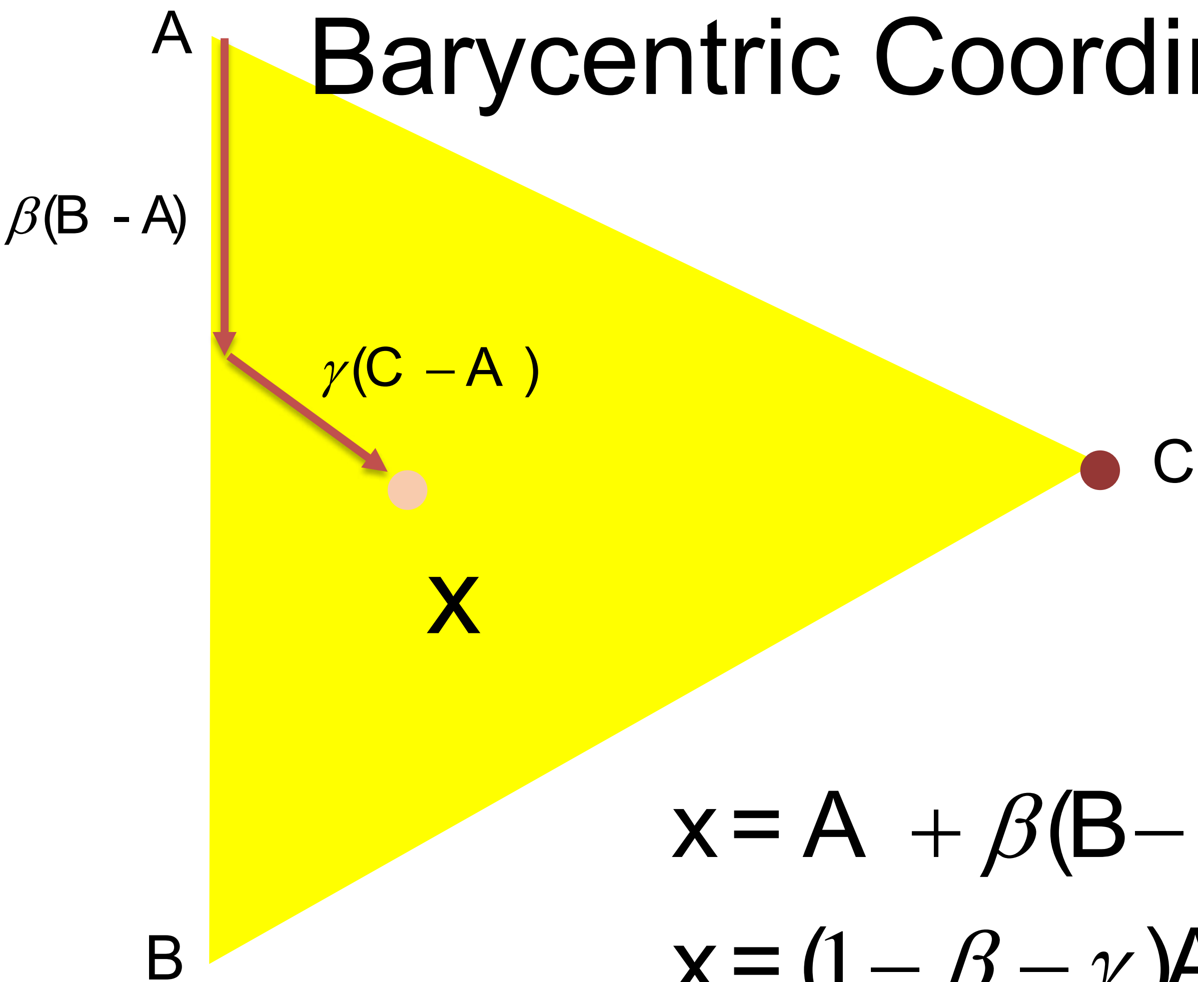
Barycentric Coordinates



$$x = \alpha A_s + \beta B_s + \gamma C_s \qquad x^t = \alpha A_t + \beta B_t + \gamma C_t$$

$$\alpha + \beta + \gamma = 1$$

Barycentric Coordinates



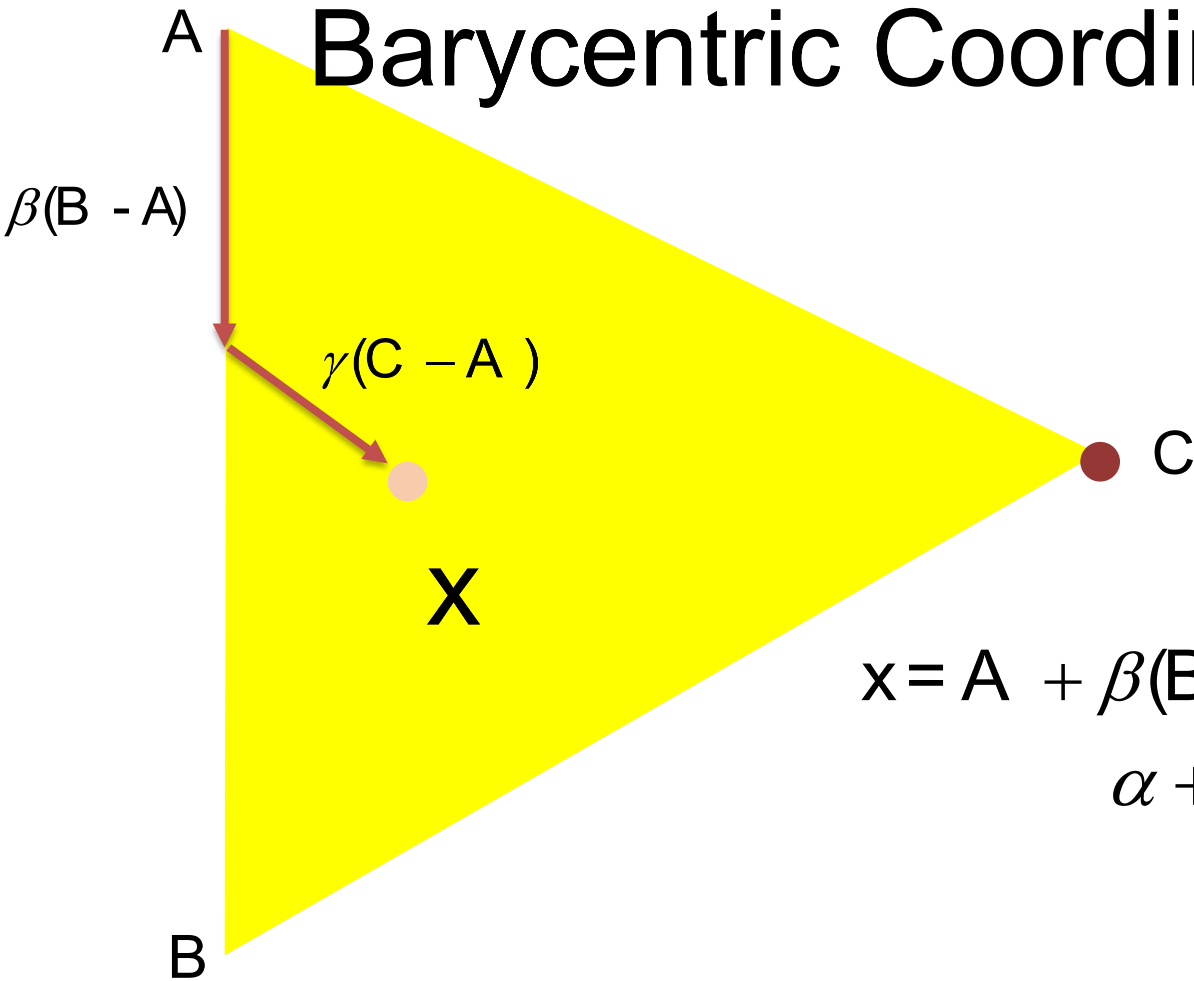
$$x = A + \beta(B - A) + \gamma(C - A)$$

$$x = (1 - \beta - \gamma)A + \beta B + \gamma C$$

$$\Rightarrow$$

$$x = \alpha A + \beta B + \gamma C \qquad \alpha + \beta + \gamma = 1$$

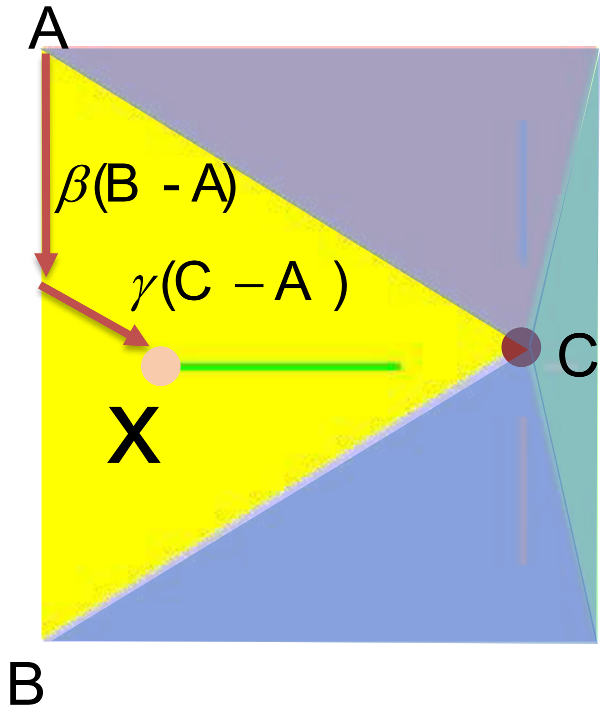
Barycentric Coordinates



$$x = A + \beta(B - A) + \gamma(C - A)$$

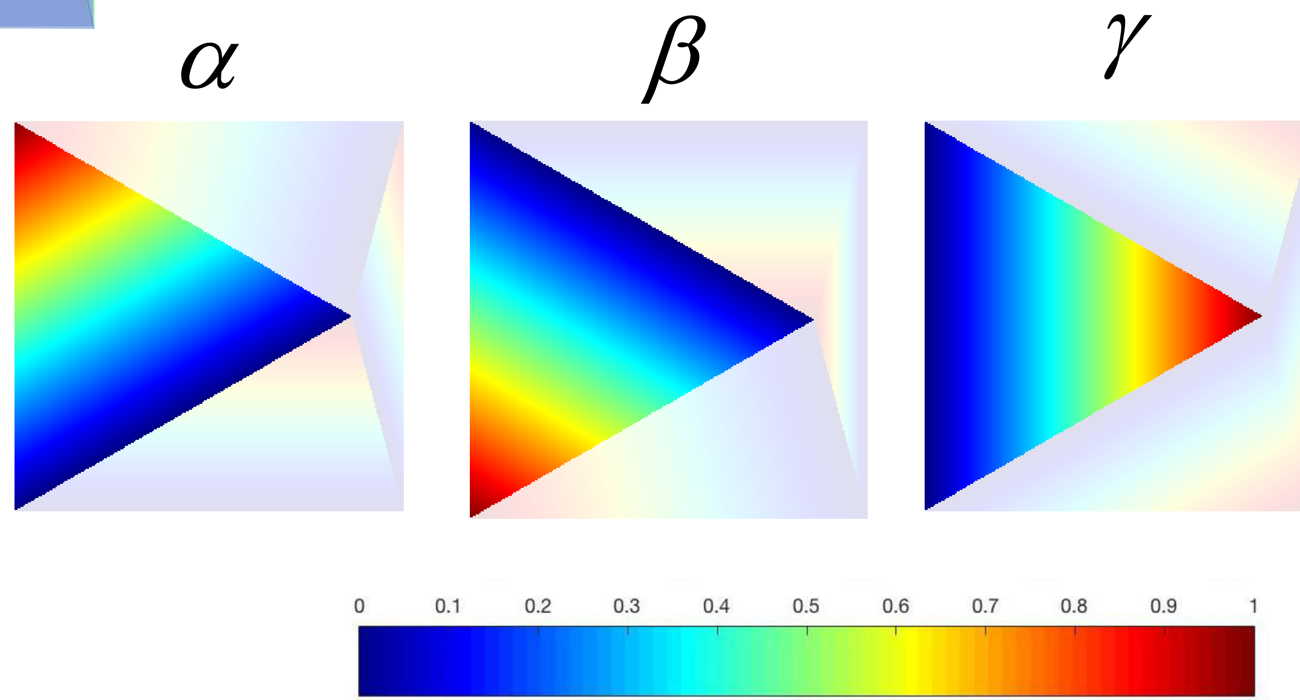
$$\alpha + \beta + \gamma = 1$$

$$\begin{bmatrix} A_x & B_x & C_x \\ A_y & B_y & C_y \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \lambda \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$ linear equations in 3 unknowns

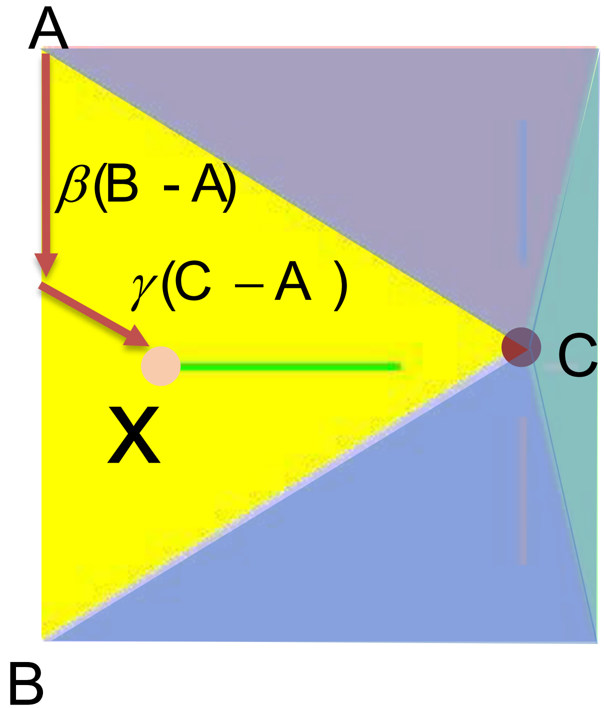


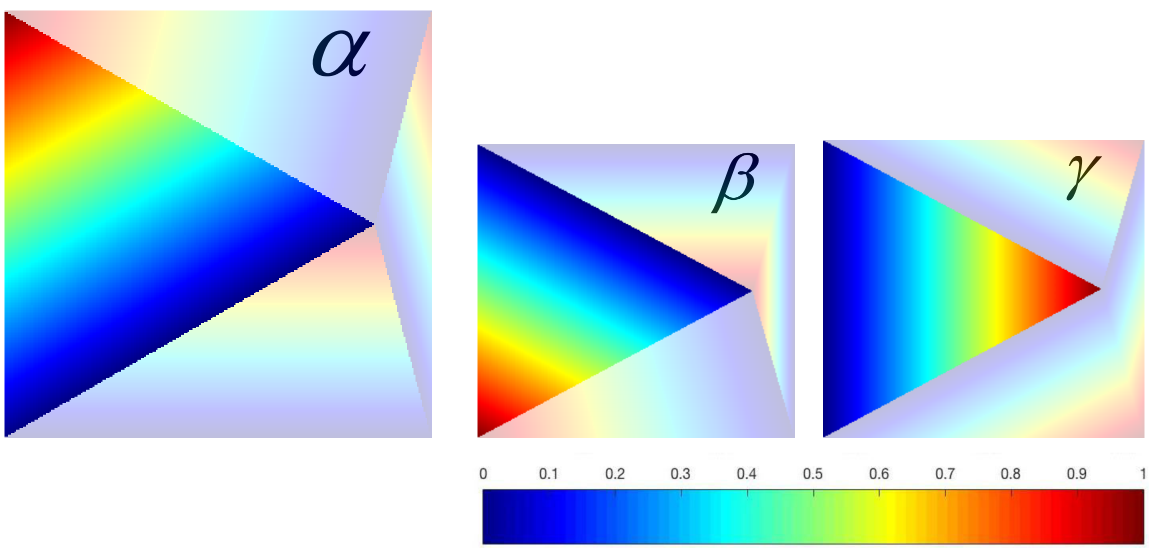
Barycentric coordinate

$$\begin{bmatrix} A_x & B_x & C_x \\ A_y & B_y & C_y \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \lambda \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



Barycentric coordinate





α is the distance to the vertex A, in the direction perpendicular to edge B-C.

$\alpha = 1$, then X is at A

$\alpha = 0$, $x = \beta B + \gamma C$ X is on the line between B-C

$$\beta + \gamma = 1$$

Penn Engineering

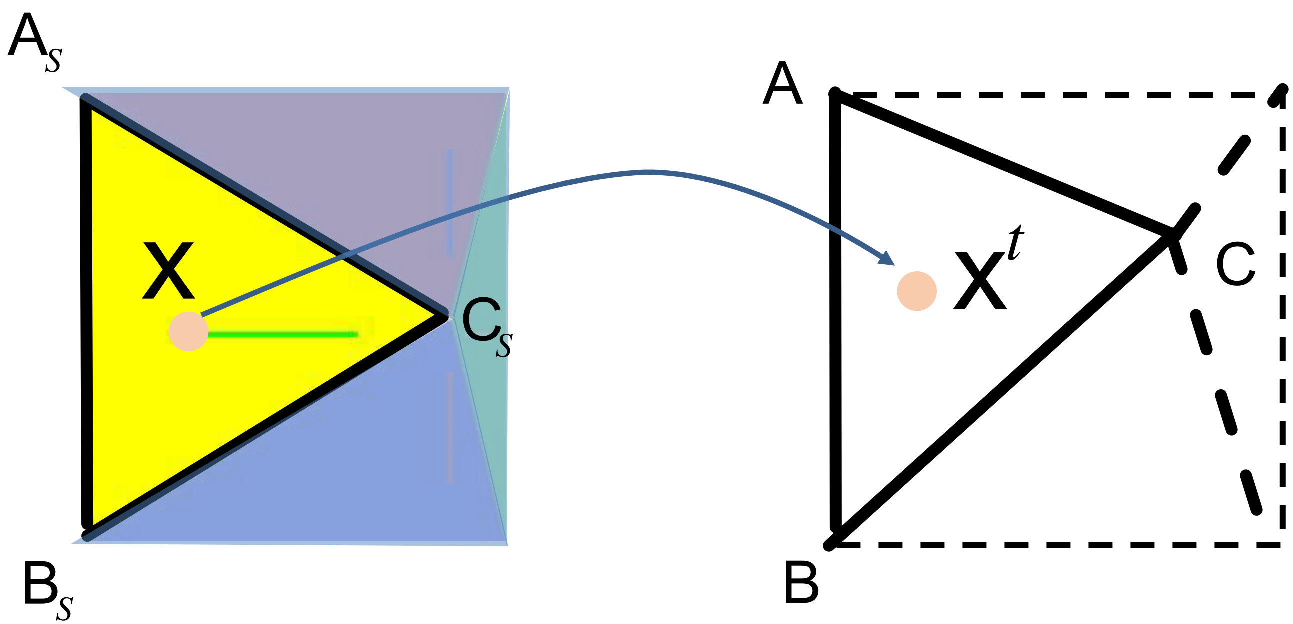
ONLINE LEARNING

Video 7.5

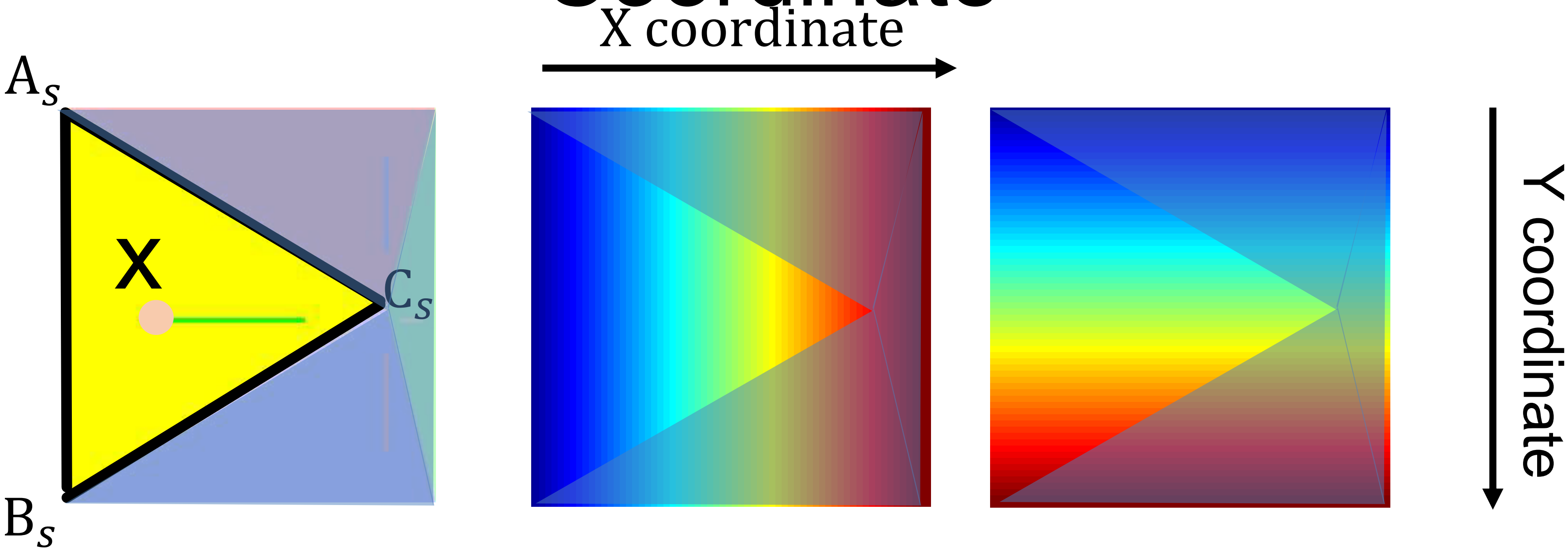
Jianbo Shi

Warping with Barycentric Coordinate

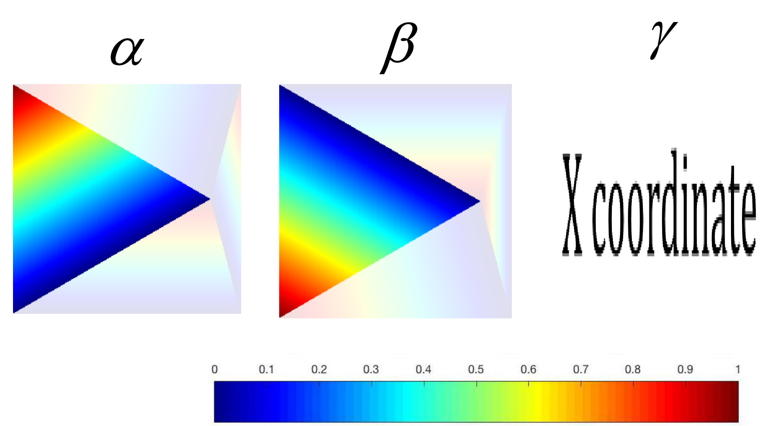
$$x = \alpha A_s + \beta B_s + \gamma C_s \qquad x^t = \alpha A + \beta B + \gamma C$$

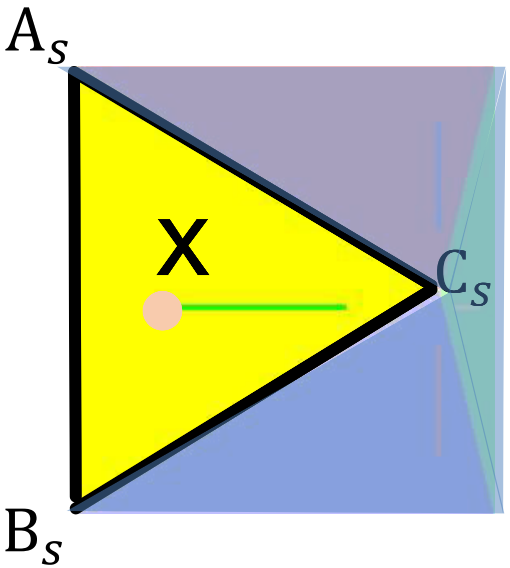


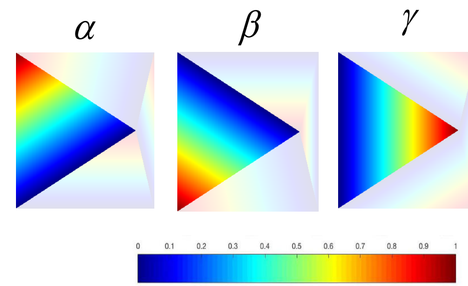
Warping with Barycentric Coordinate



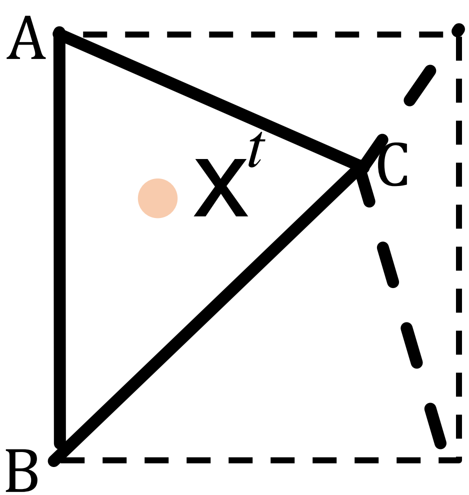
$$\begin{bmatrix} A_x & B_x & C_x \\ A_y & B_y & C_y \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \lambda \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

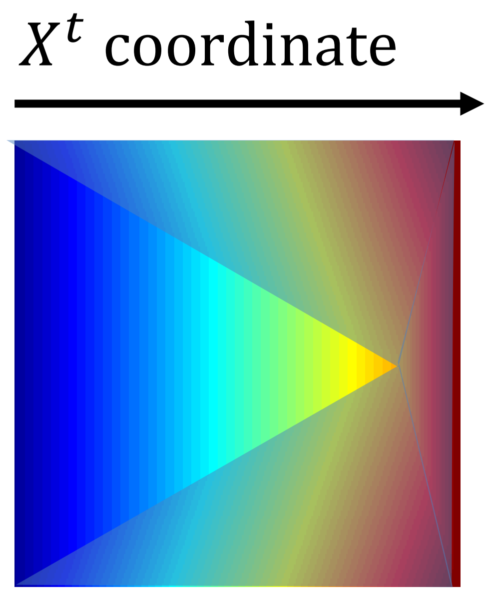




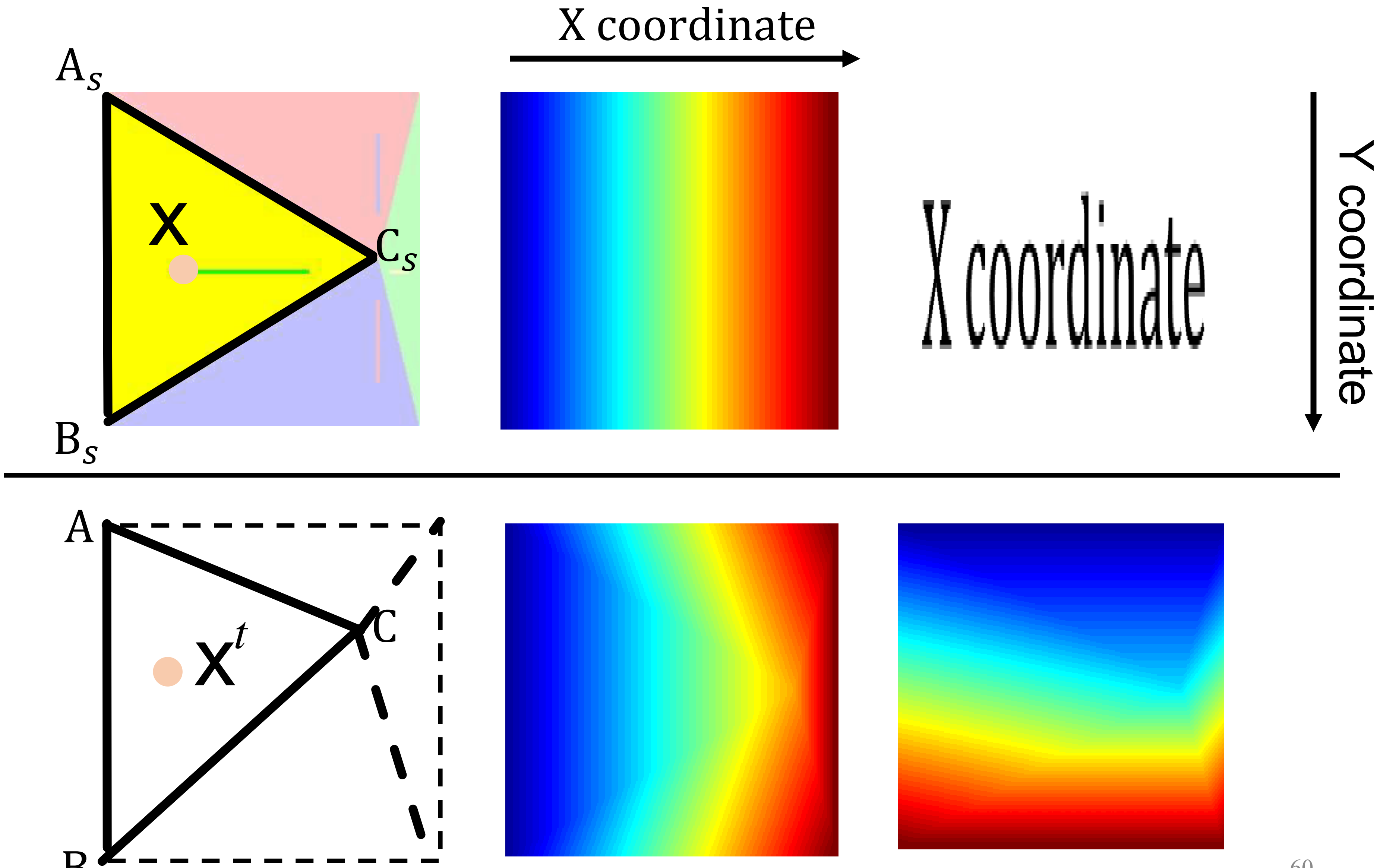


$$x^t = \alpha A + \beta B + \gamma C$$

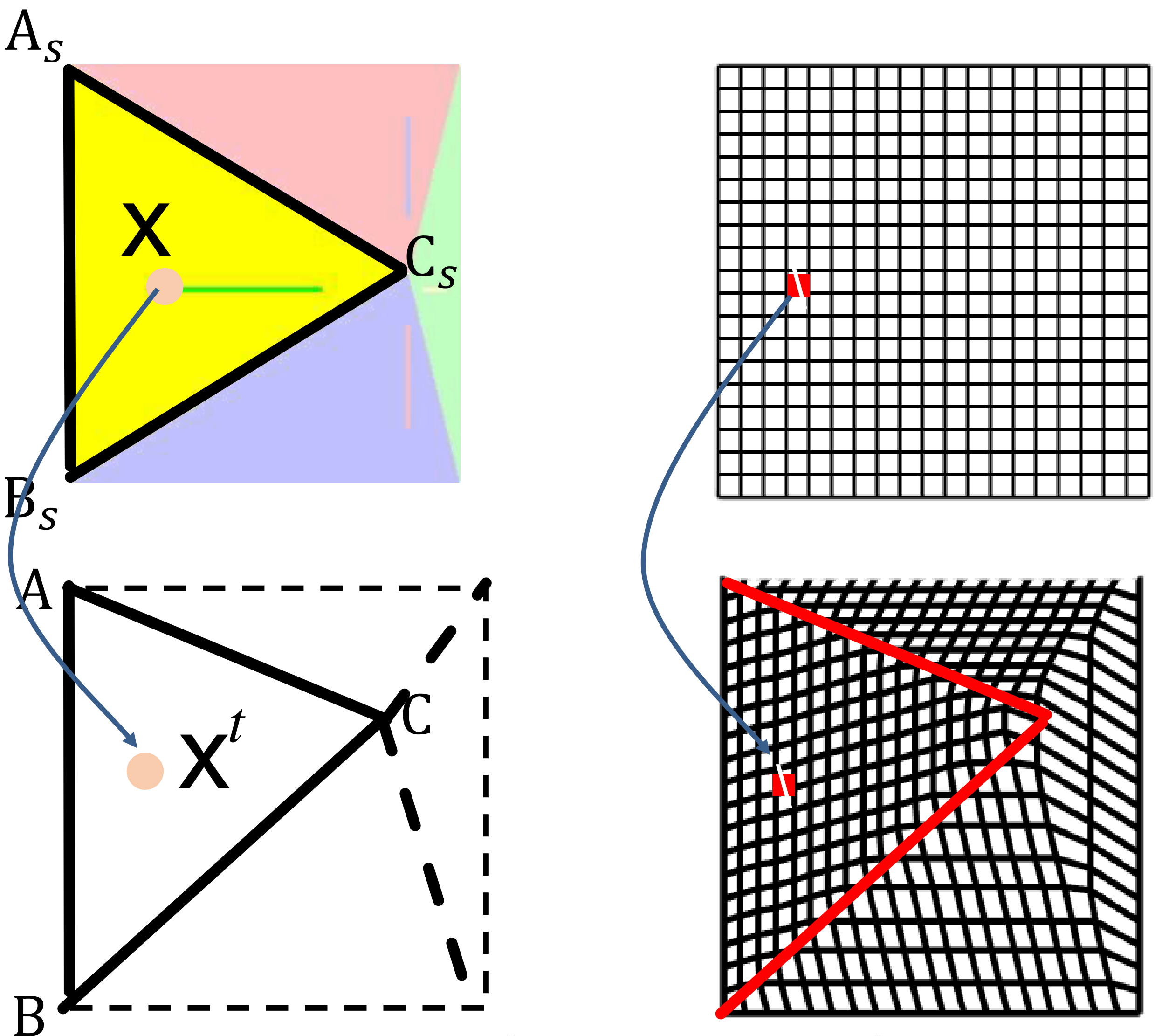




Warping with Barycentric Coordinate



Grids before and after warping



Grids before and after warping

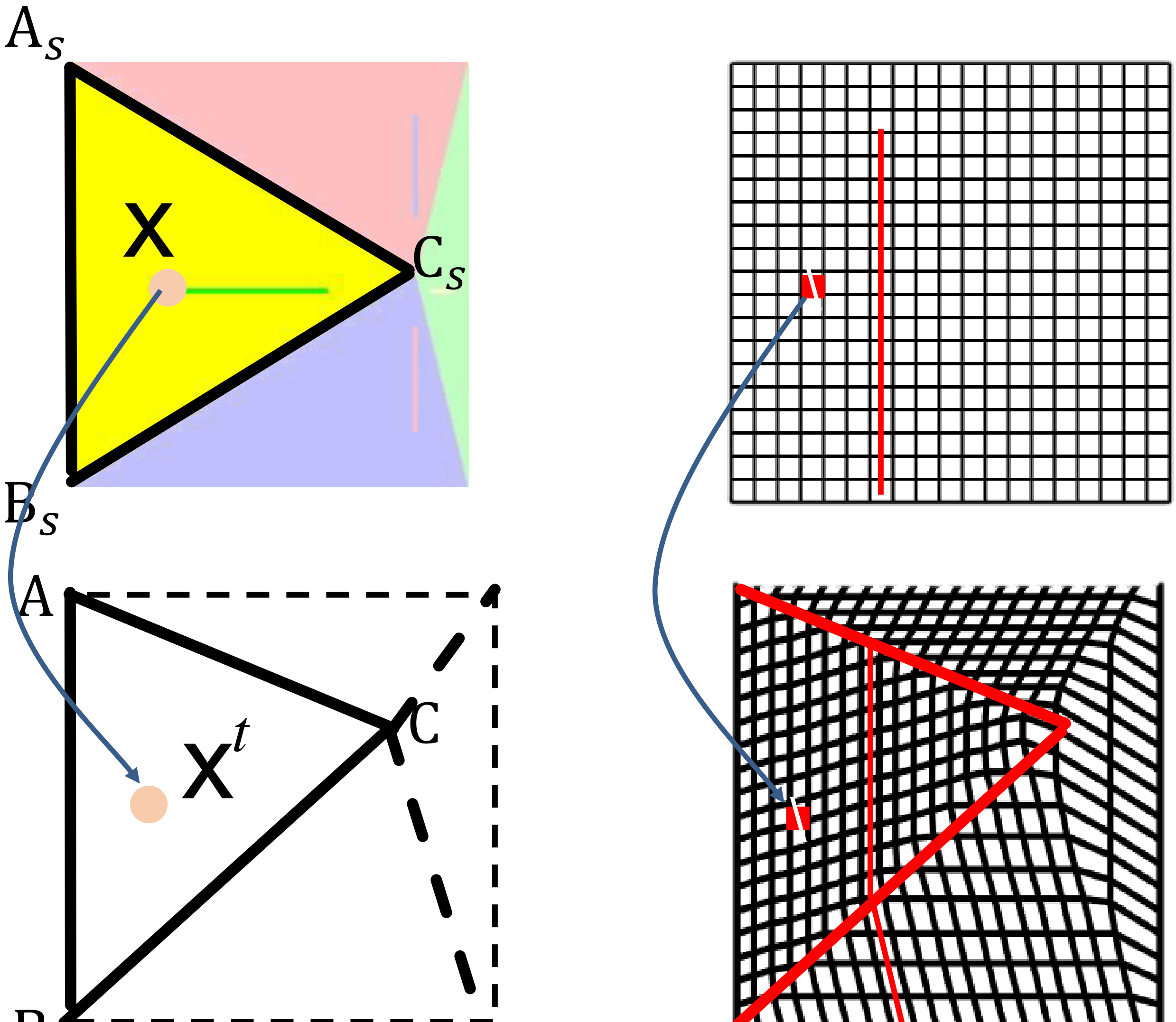


Image Warping: pixel transportation

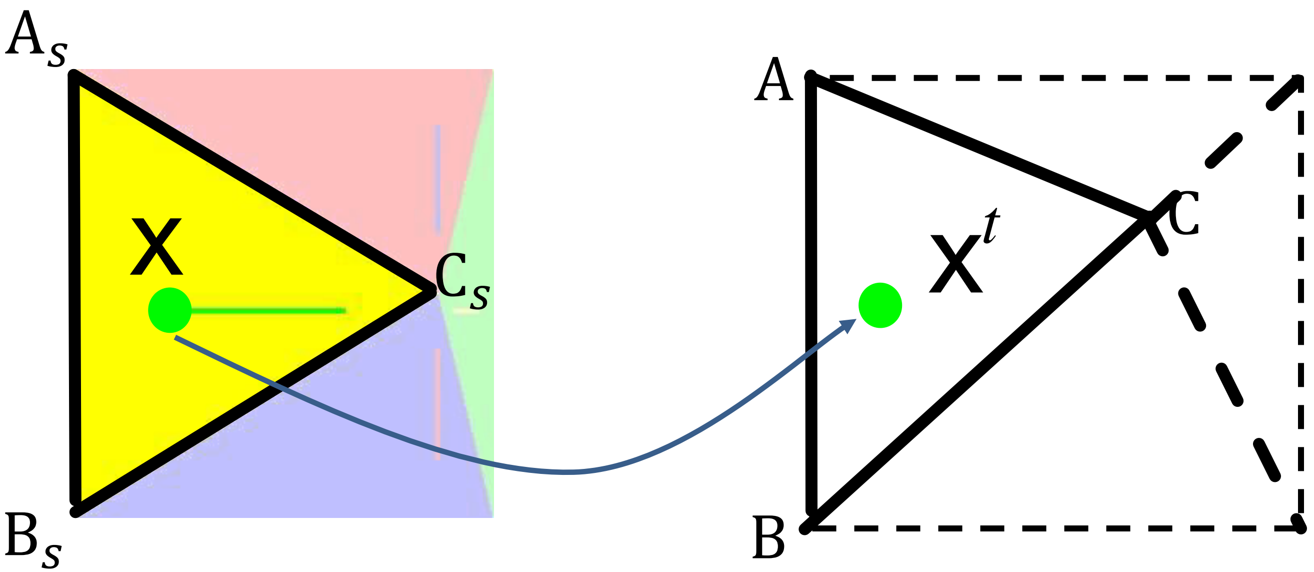


Image Warping: pixel transportation + color copying

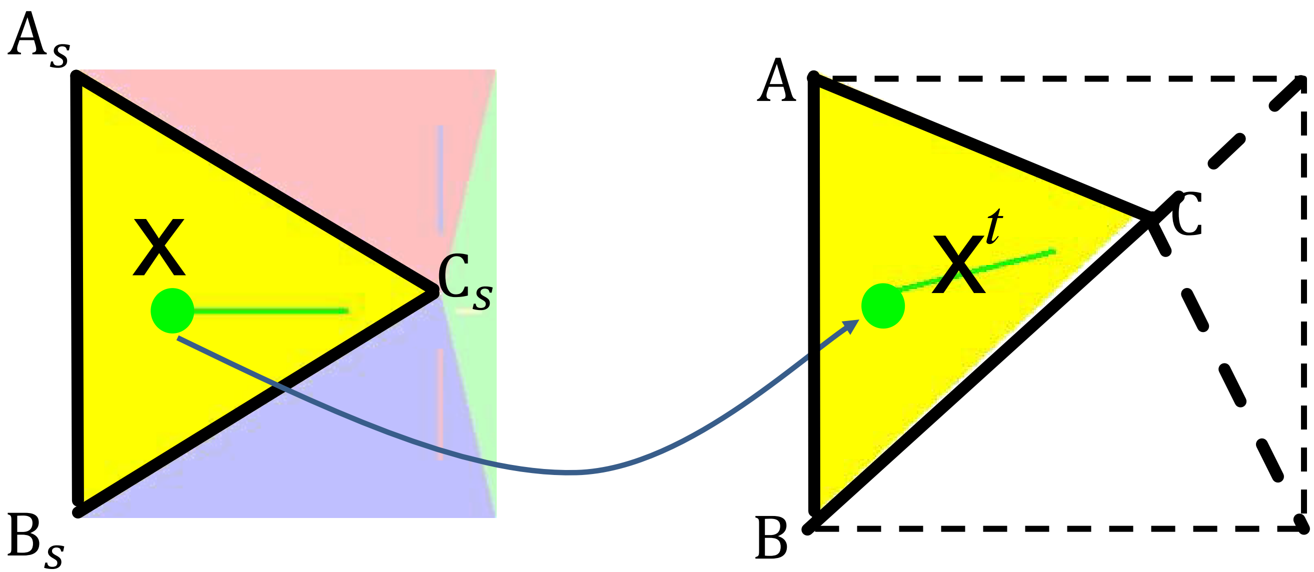
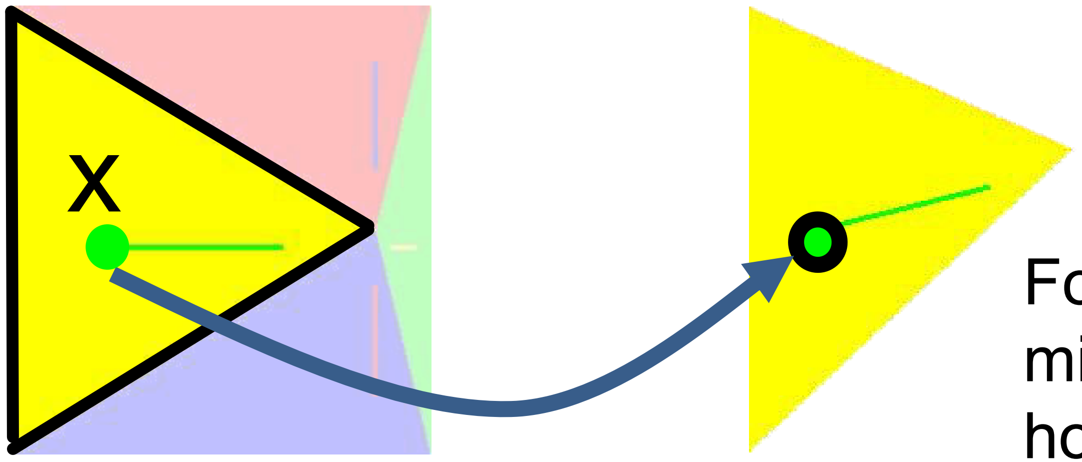
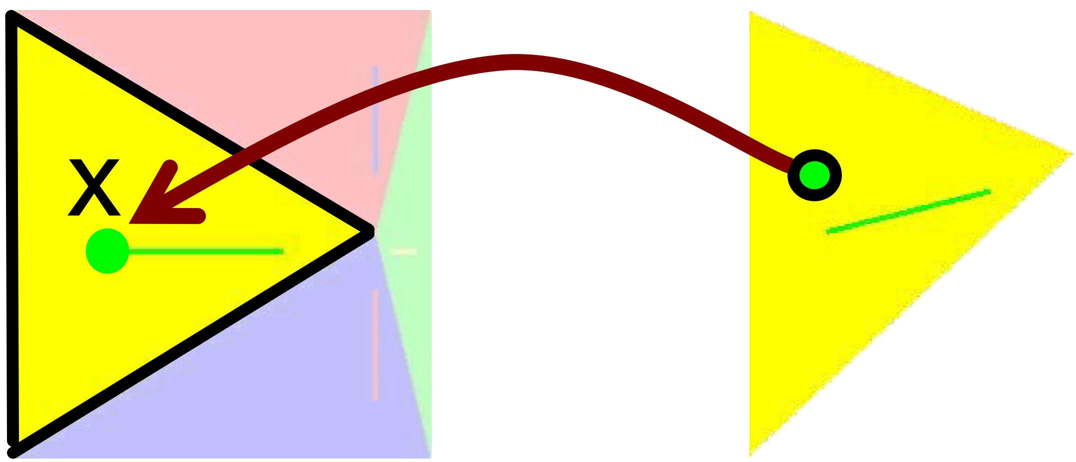


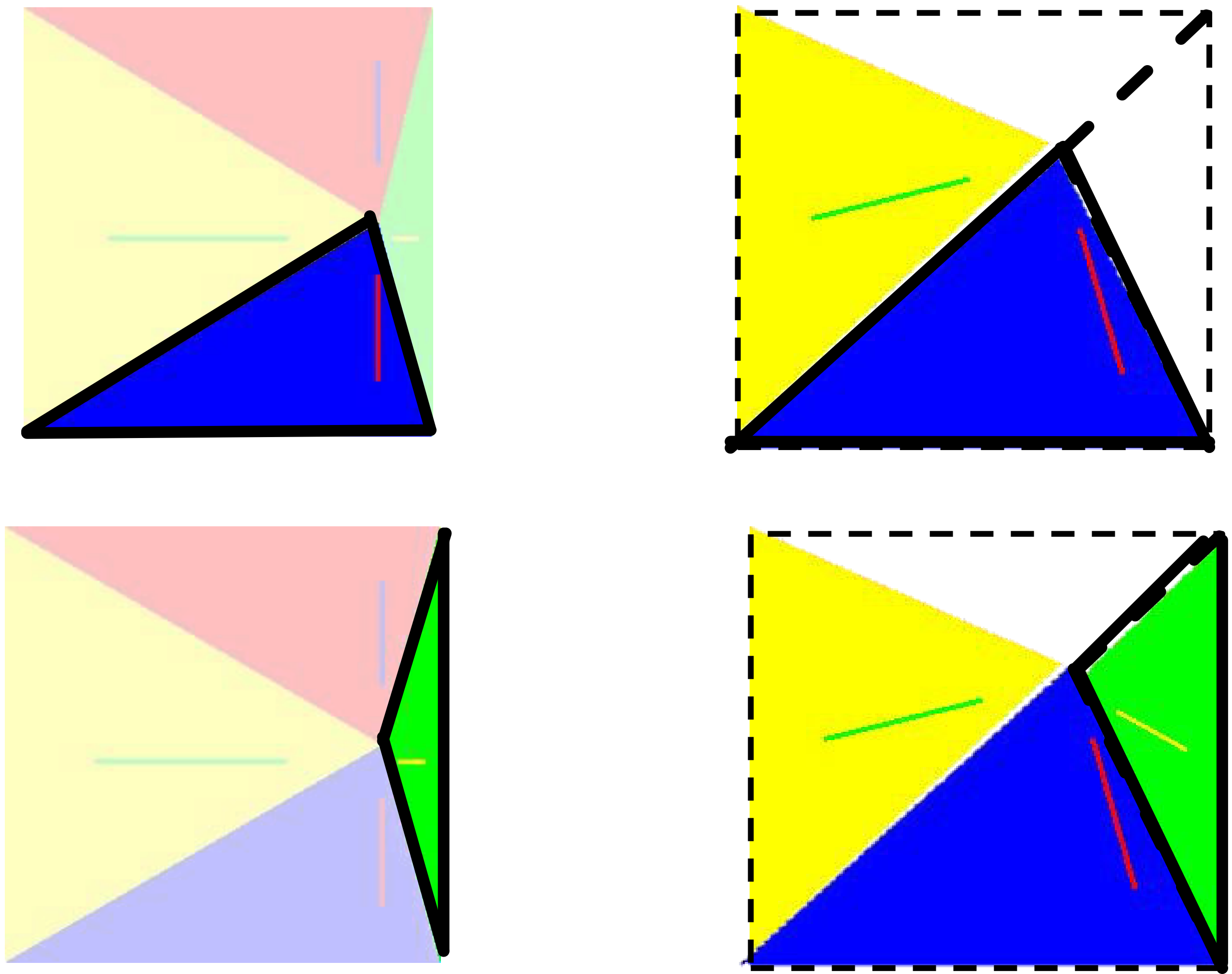
Image Warping: forward vs. backward



Forward Warping might create holes

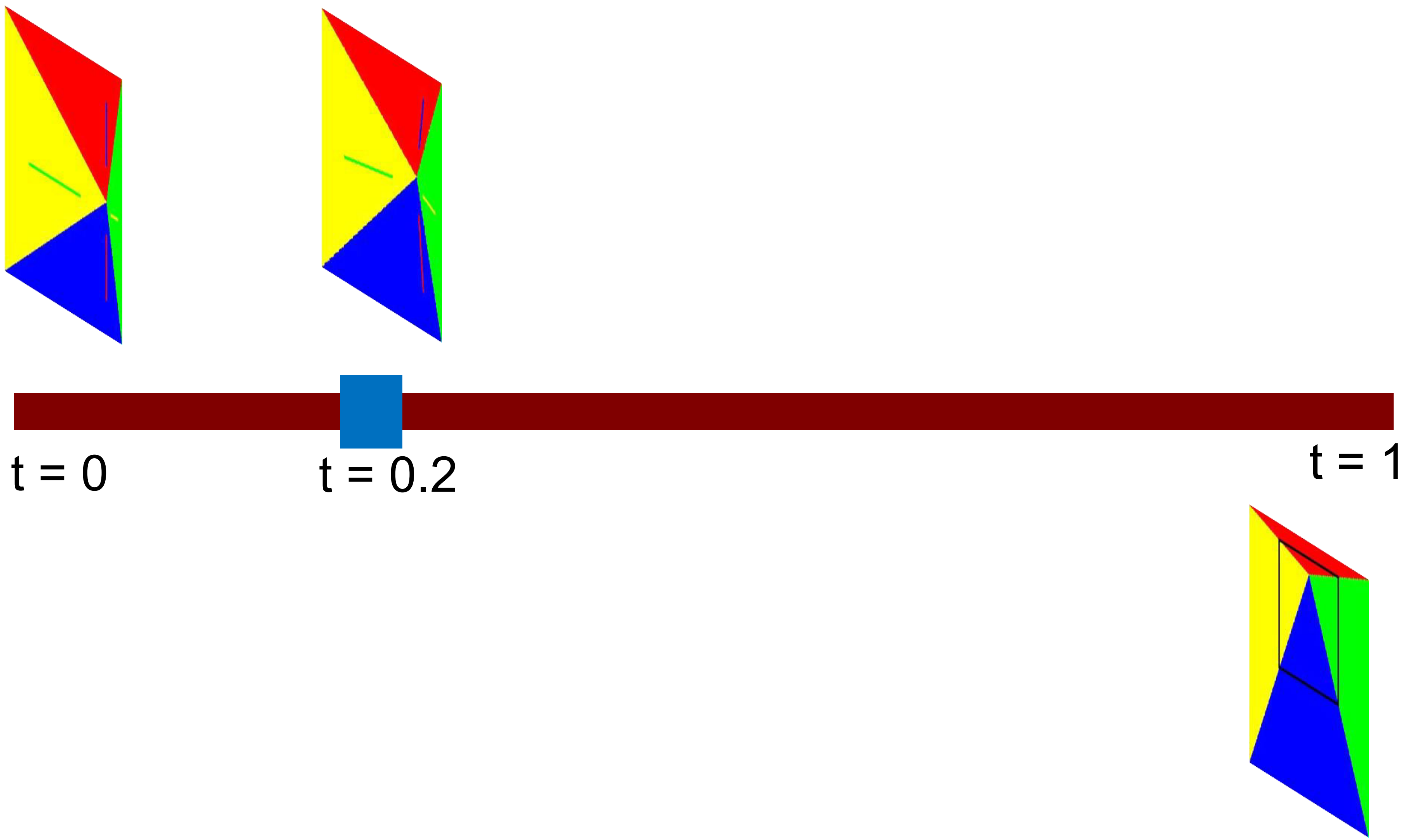


Iterate over all triangles

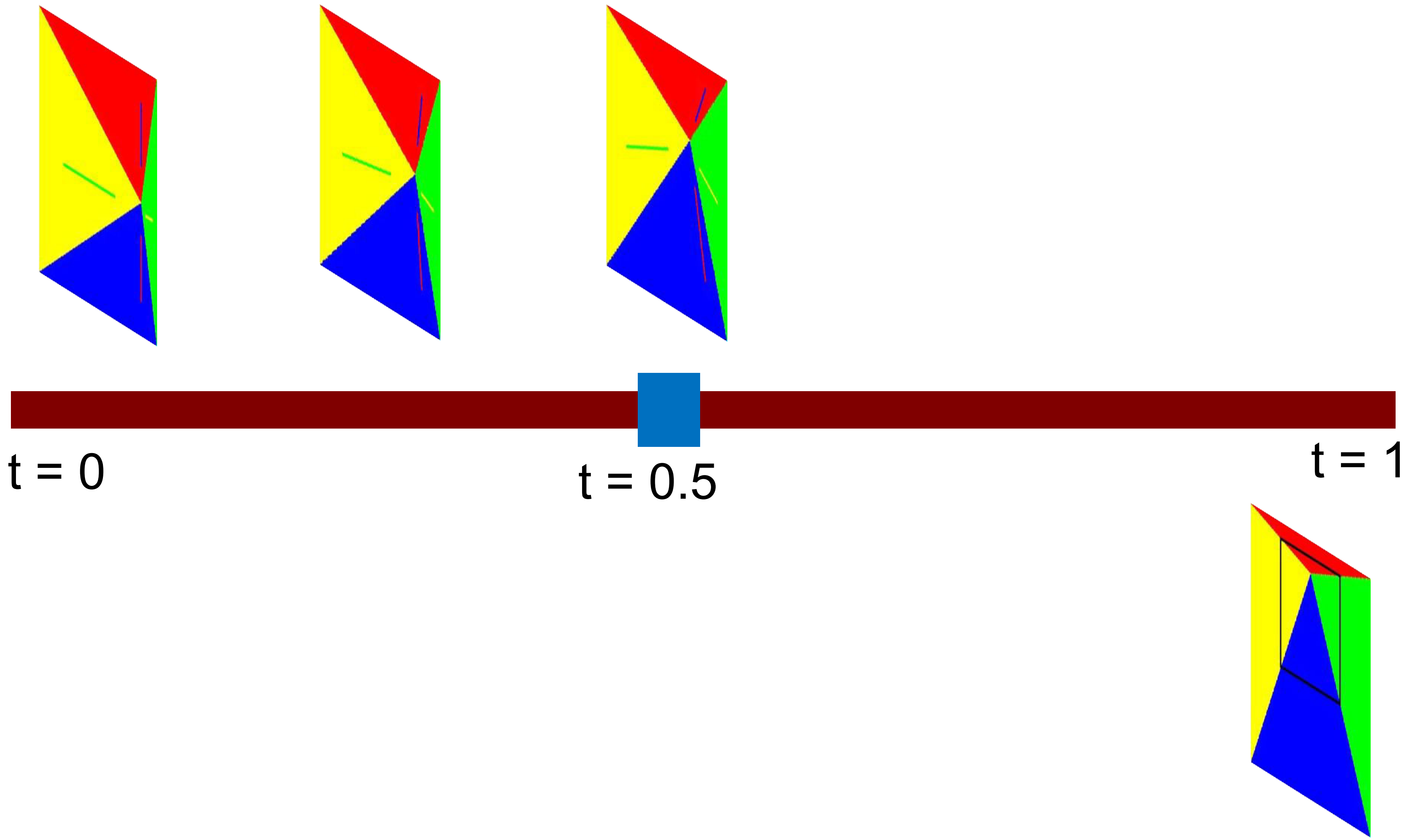



t = 0
t = 1

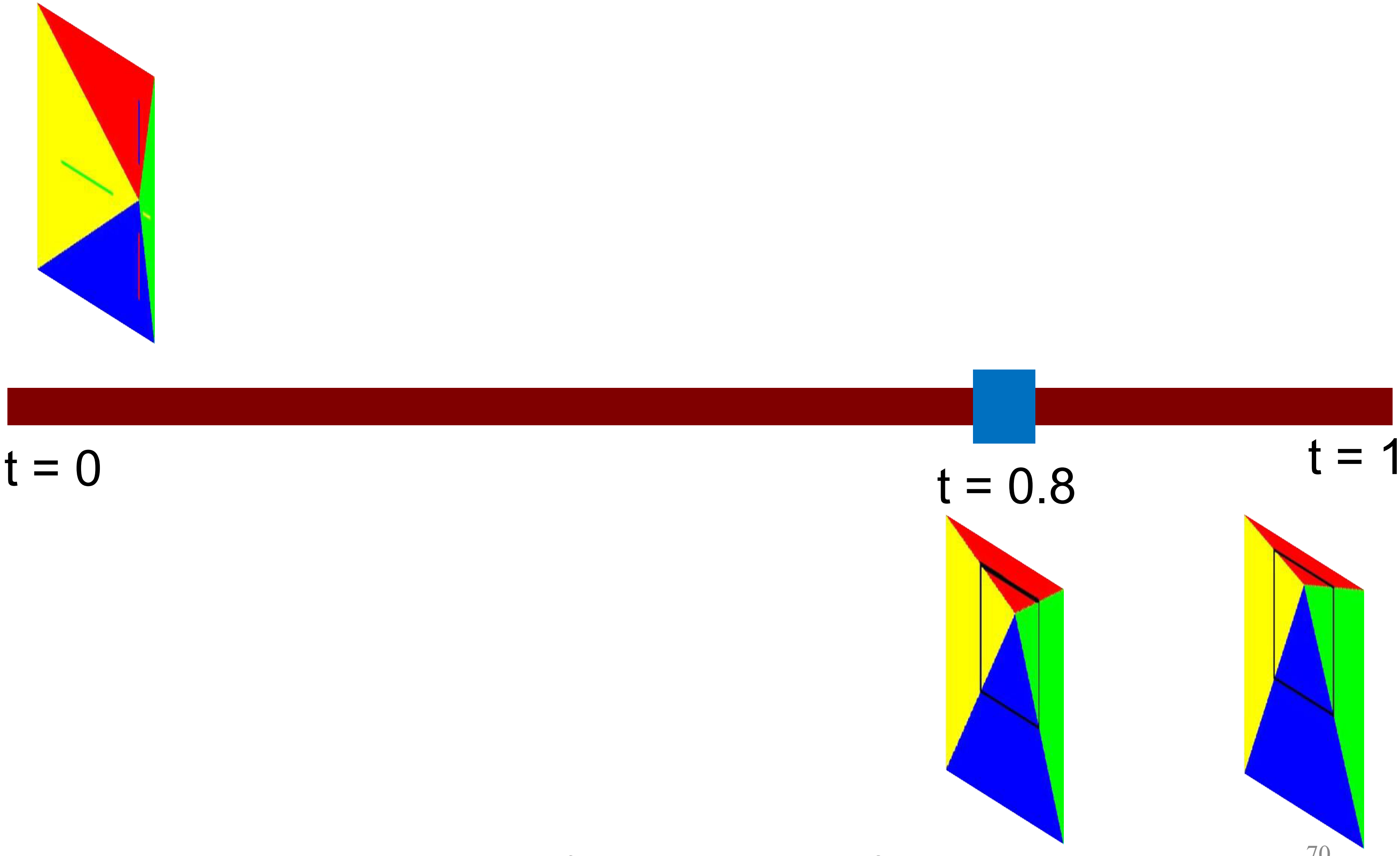
Warping forward



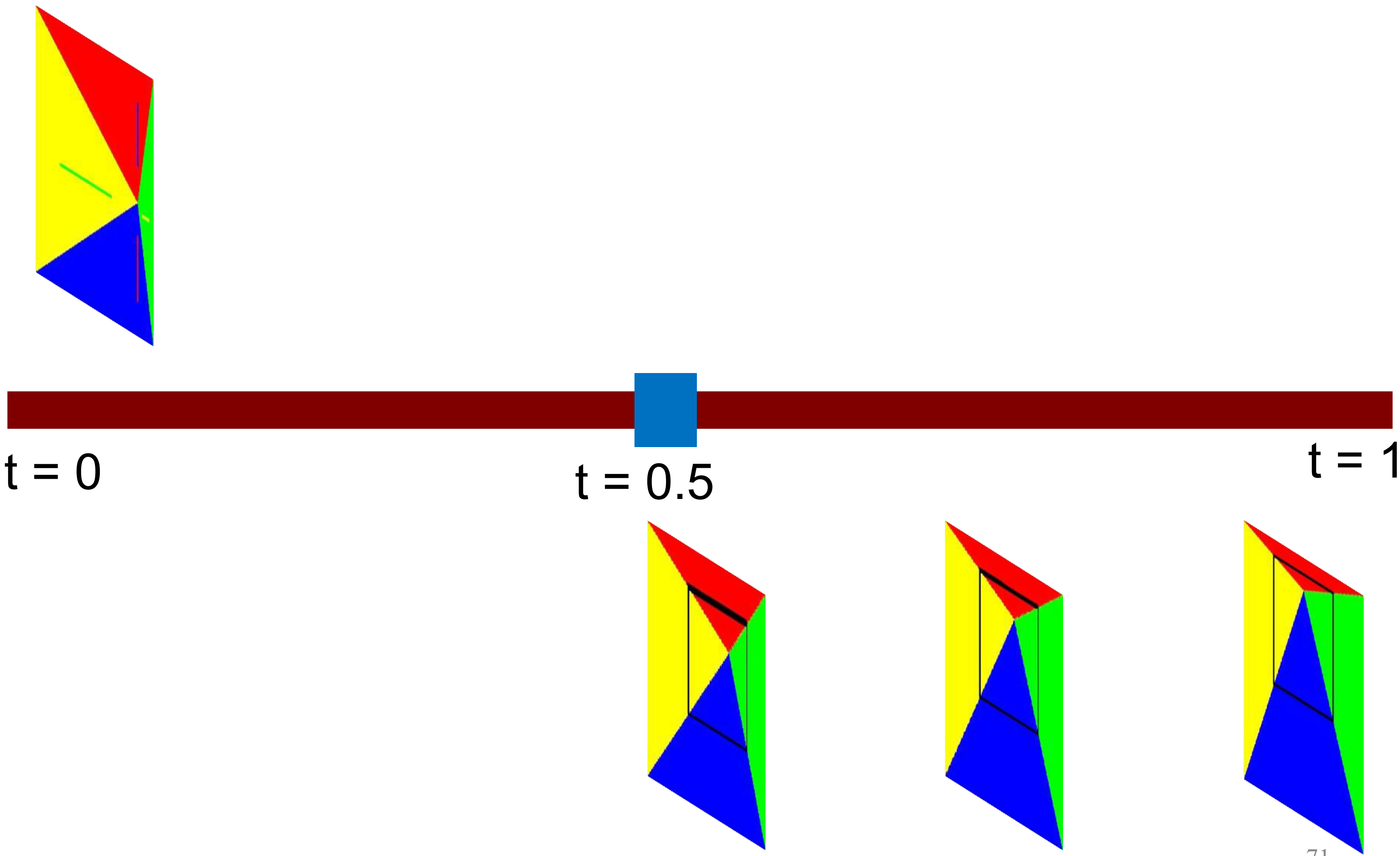
Warping foward



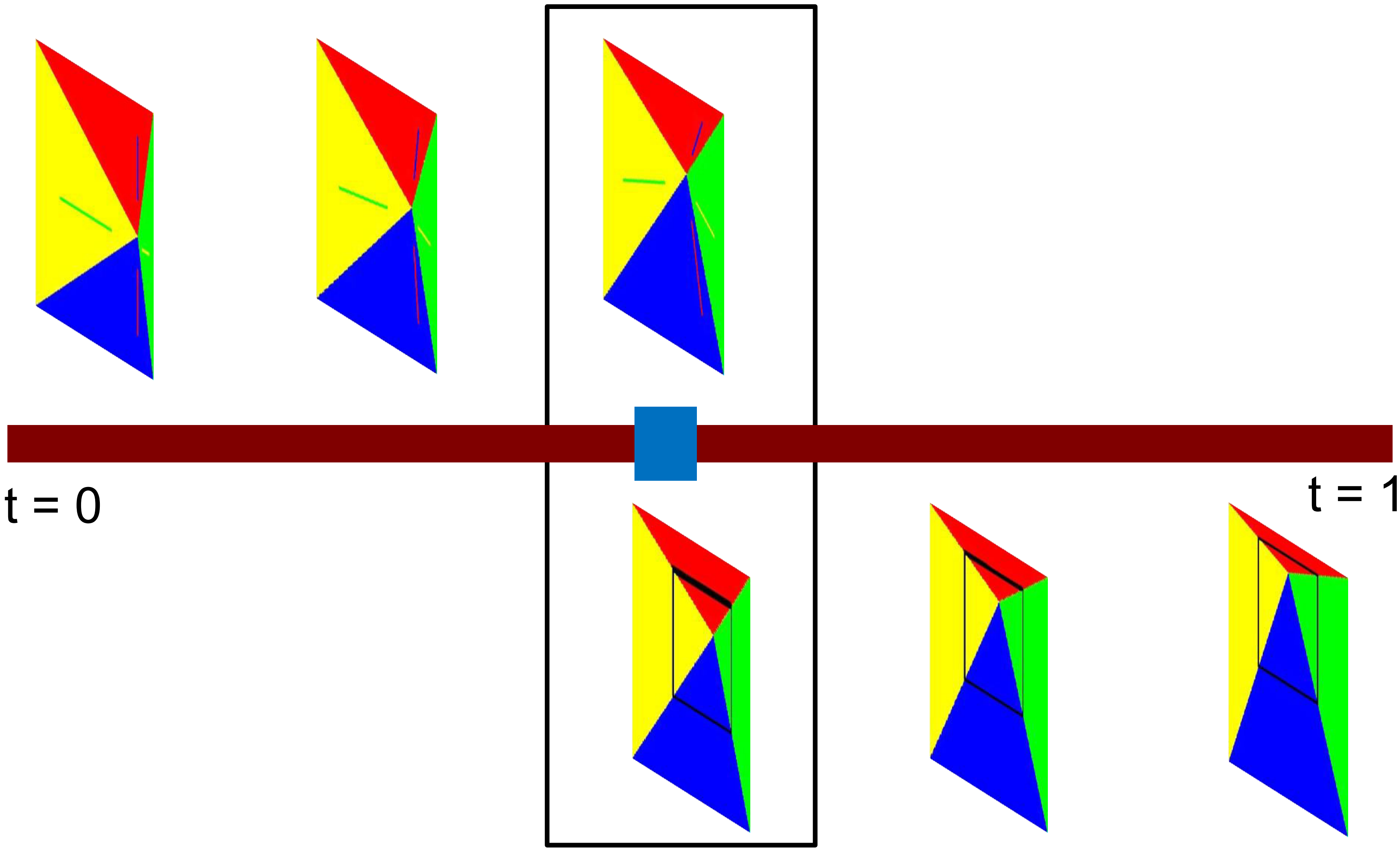
Warping backward



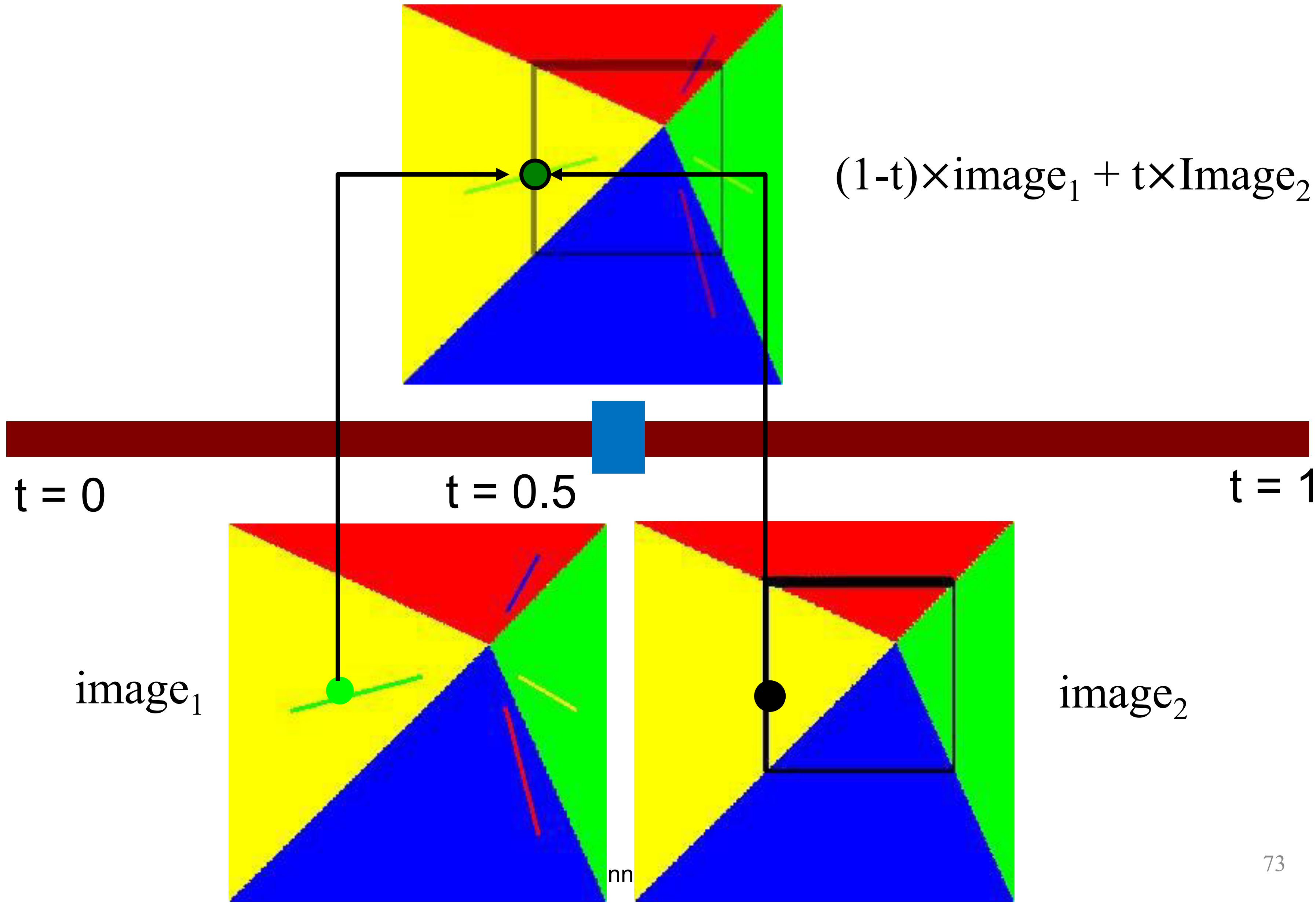
Warping backward



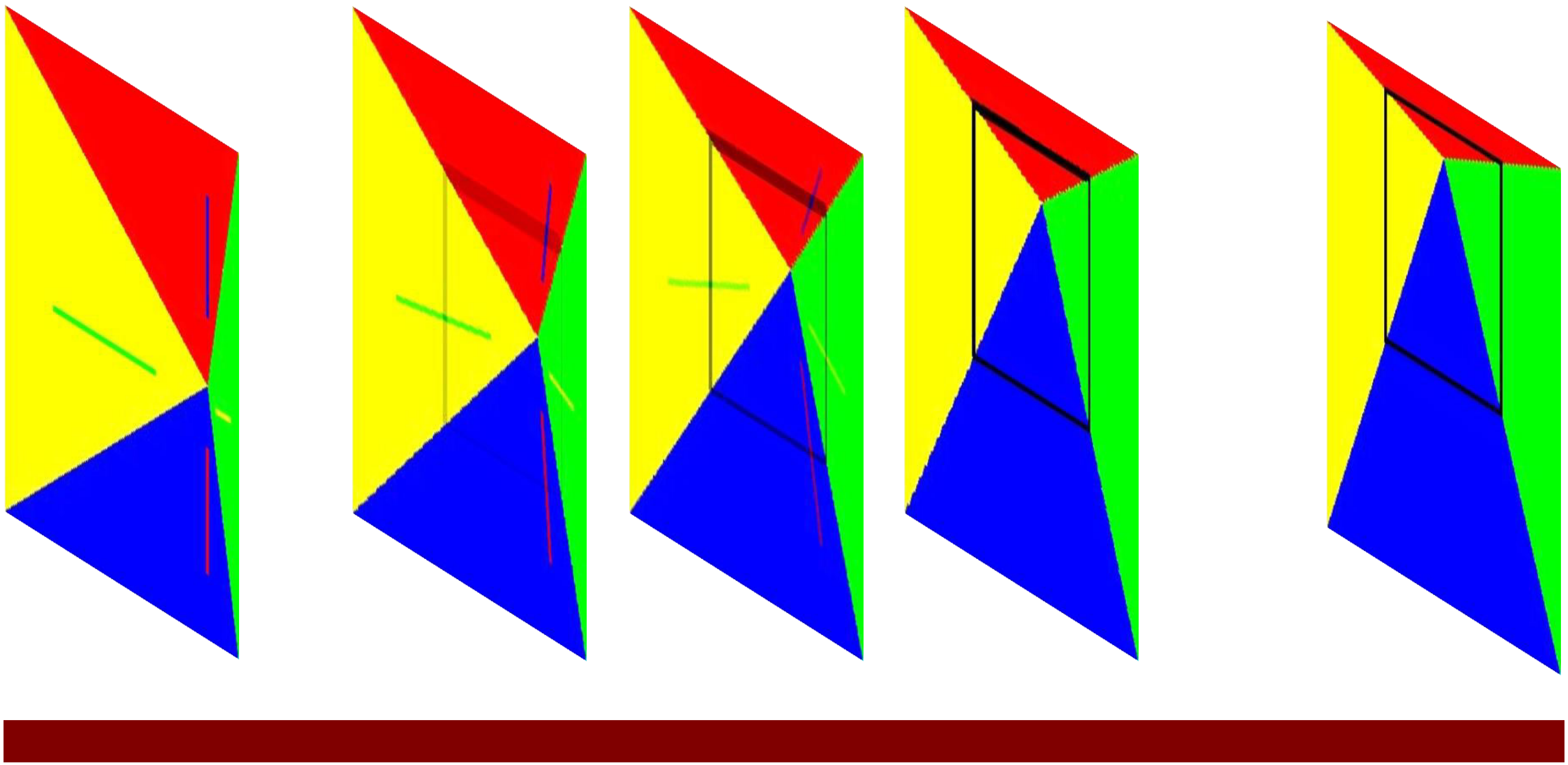
Step 3: Average warped image



Averaging Warped Images



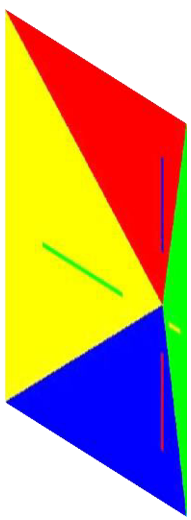
Morphing = Warping + Averaging

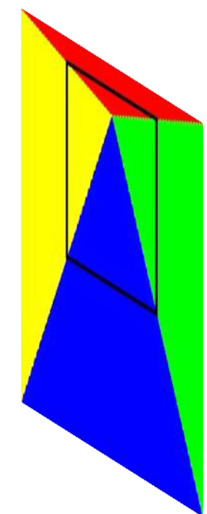


t = 0

t = 1

Morphing Result





Morphing = Warping + Averaging

Morphing procedure:

for every t,

1. Find the average shape
2. Non-parametric warping
3. Find the average color
   – Cross-dissolve the warped images