Digital Image Processing

Feature Based Morphing Using Lines

Brent M. Dingle, Ph.D. 2015
Game Design and Development Program
Mathematics, Statistics and Computer Science
University of Wisconsin - Stout

Material in this presentation is largely based on/derived from
presentation(s) and book: The Digital Image by Dr. Donald House at Texas A&M University

Lecture Objectives

- Previously
  - Interpolation
  - Warping
  - Morphing

- Today
  - Feature Based Morphs
    - Details of using line pairs

Morphing

- A smooth transition from one shape and coloring to another
- An image morph involves
  - warping both images to some intermediate shape such that they can be superimposed on each other
  - blending the two images together to produce a third image

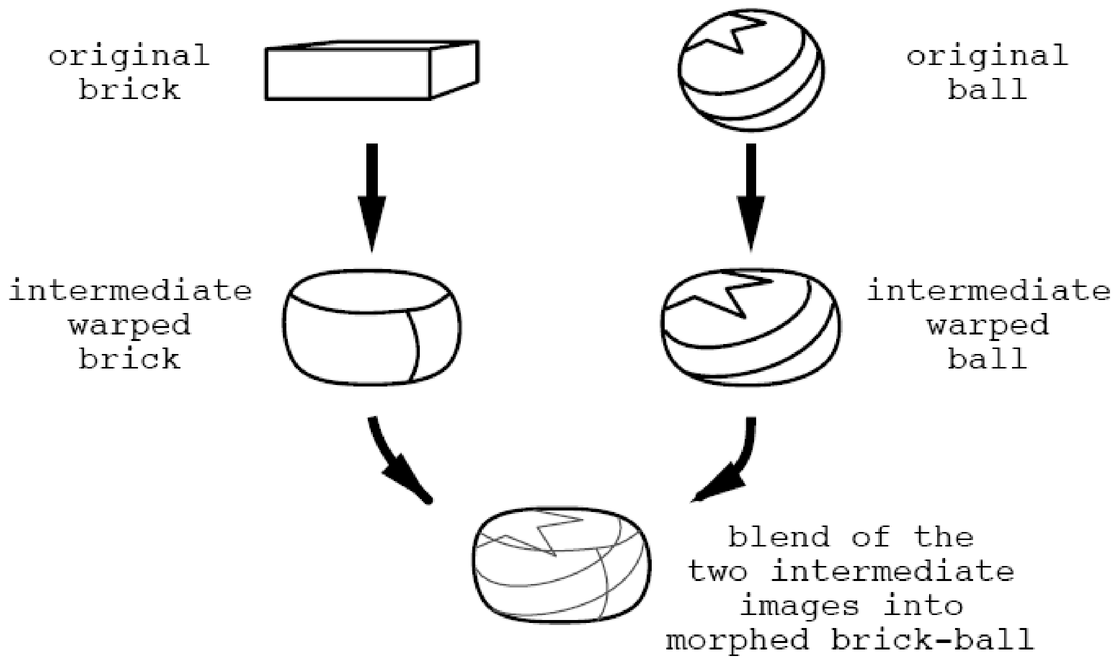


Figure 13.1: A Step in a Morph of a Brick into a Ball

Sample Steps

- Blend is controlled by giving more strength to the original image that is closer to the deformed shape

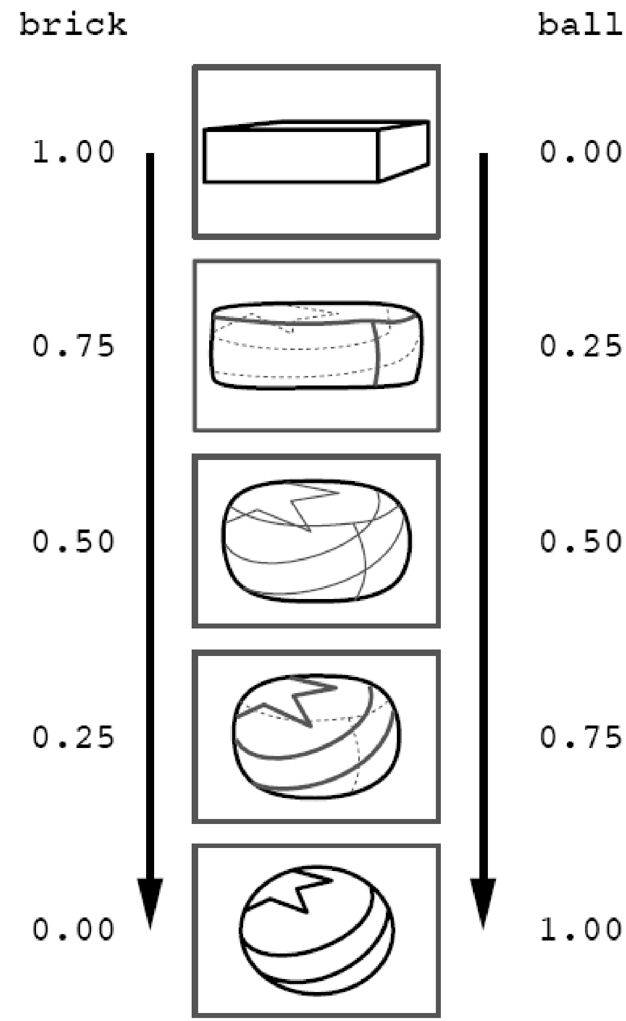


Figure 13.2: Steps in Morph Sequence from Brick to Ball

Feature-based Image Metamorphosis

- Concept:
  - Morph one image into another using an inverse map by specifying corresponding features in both images using directed lines

Beier, T & Neely, S. (1992). Feature-based image metamorphosis Computer Graphics 26 (2): 35-42. doi: 10.1145/133994.134003

-- Identify features via line-pairs

3 Examples: Each Using One Pair of Lines

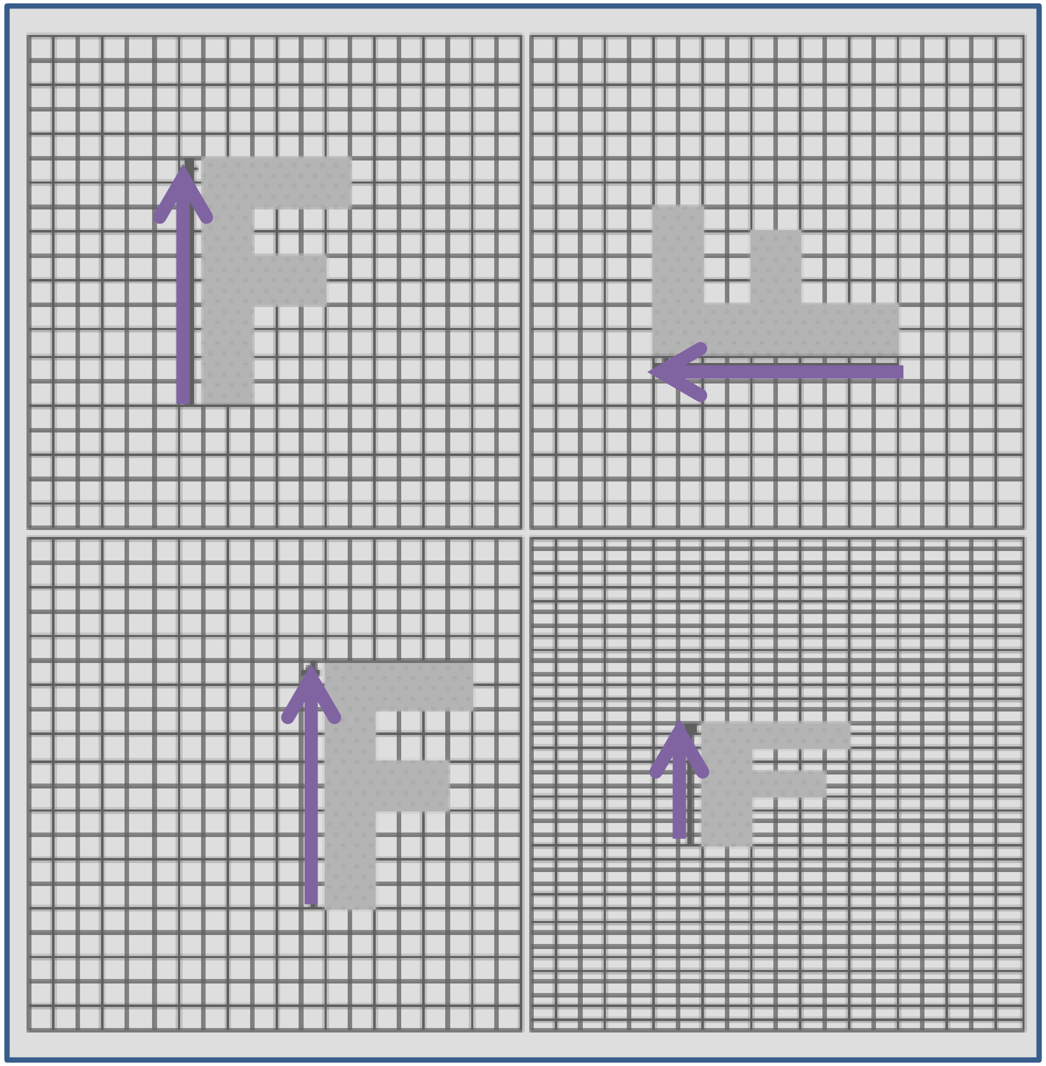
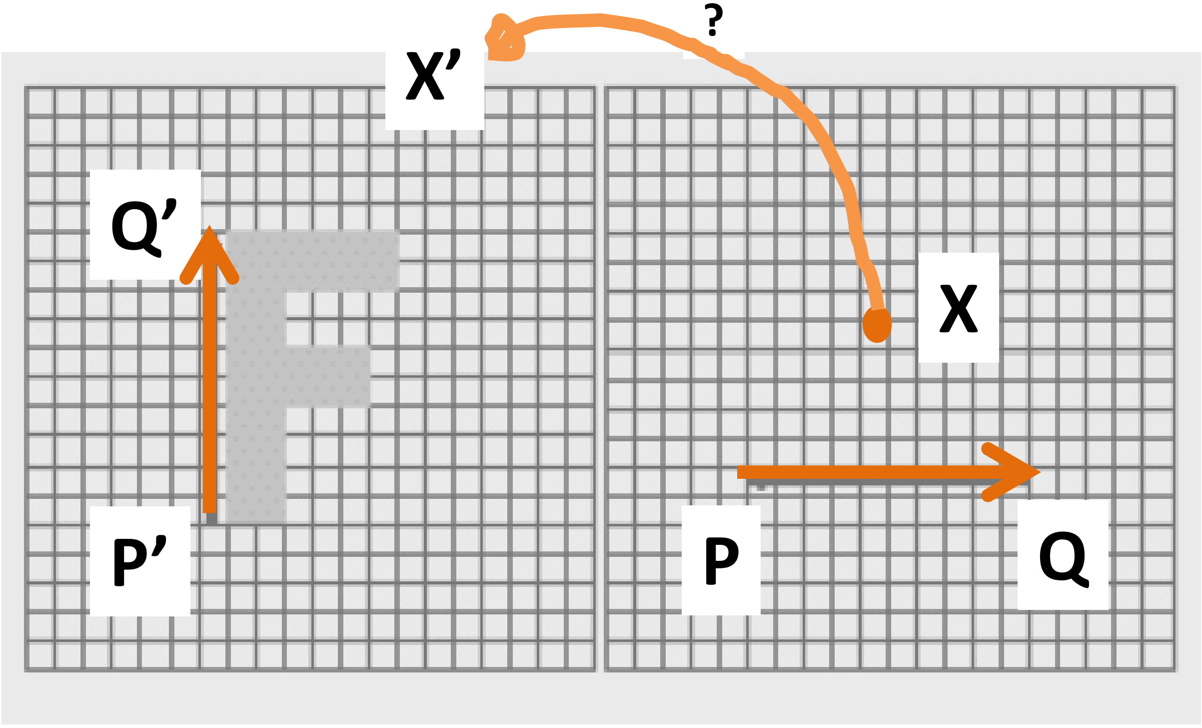


Figure 2: Single line pair examples

Transform via 1 pair of lines

- Given a line pair: PQ and P'Q'
  - Determine which pixel X' in the source image do we sample for the X pixel in the destination image



Transform via 1 pair of lines

- Solution Design

```
For each pixel X in the destination image
        find the corresponding u,v
        find the X' in the source image for that u,v
        destinationImage(X) = sourceImage(X')
```

First:
  Calculate parameters u and v

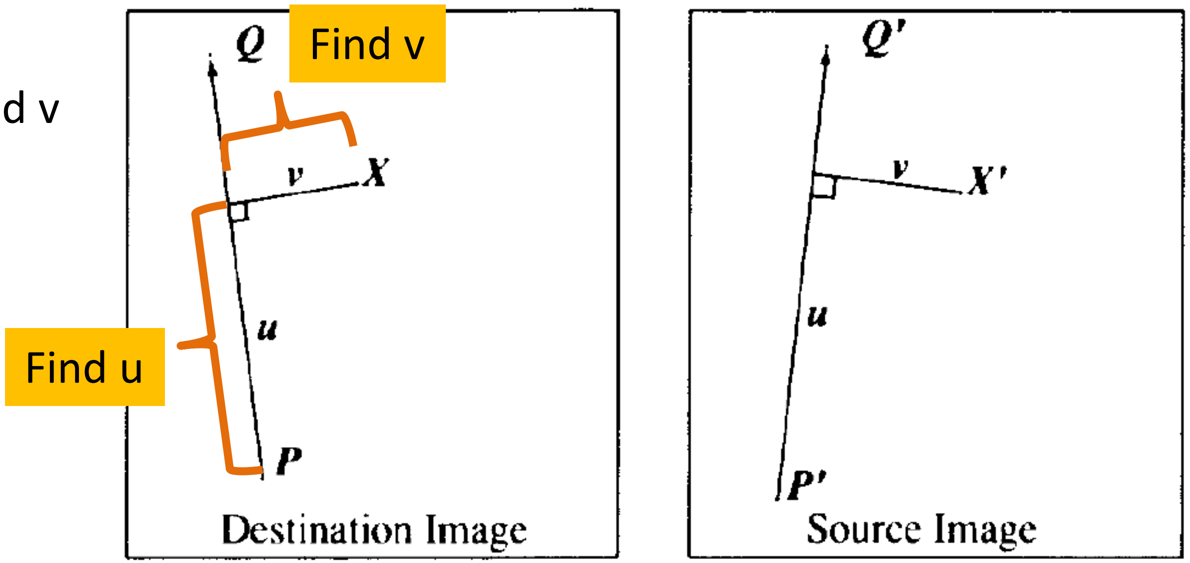


Figure 1: Single line pair

Transform via 1 pair of lines

- Solution Design

```
For each pixel X in the destination image
        find the corresponding u,v
        find the X' in the source image for that u,v
        destinationImage(X) = sourceImage(X')
```

First:
Calculate parameters u and v

Then
Use u, v to get X’

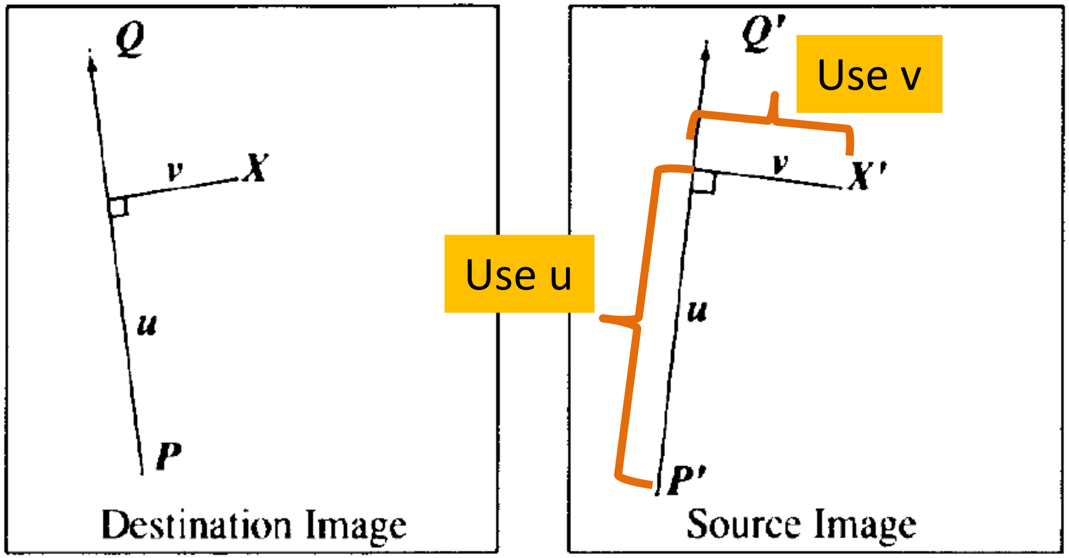
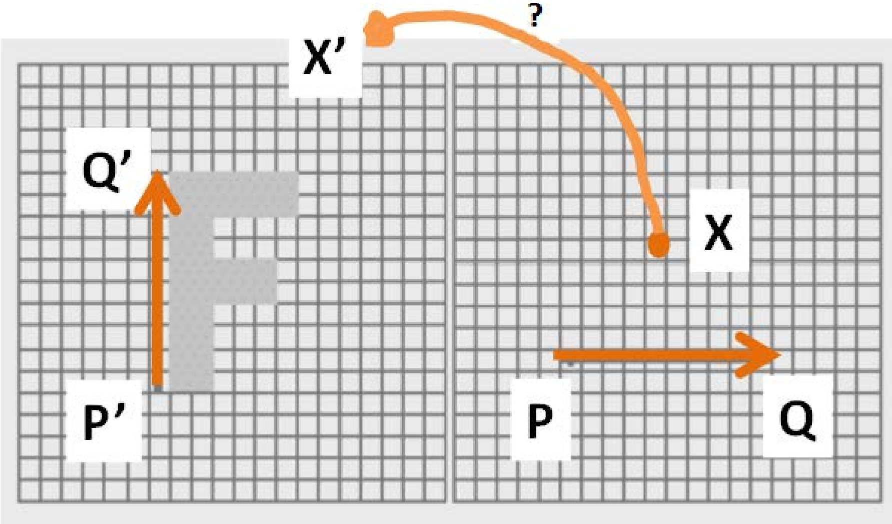


Figure 1: Single line pair

Implementation



$$u = \frac{(X - P) \cdot (Q - P)}{\| Q - P \|^2} \quad (1)$$

walk-thru of this follows

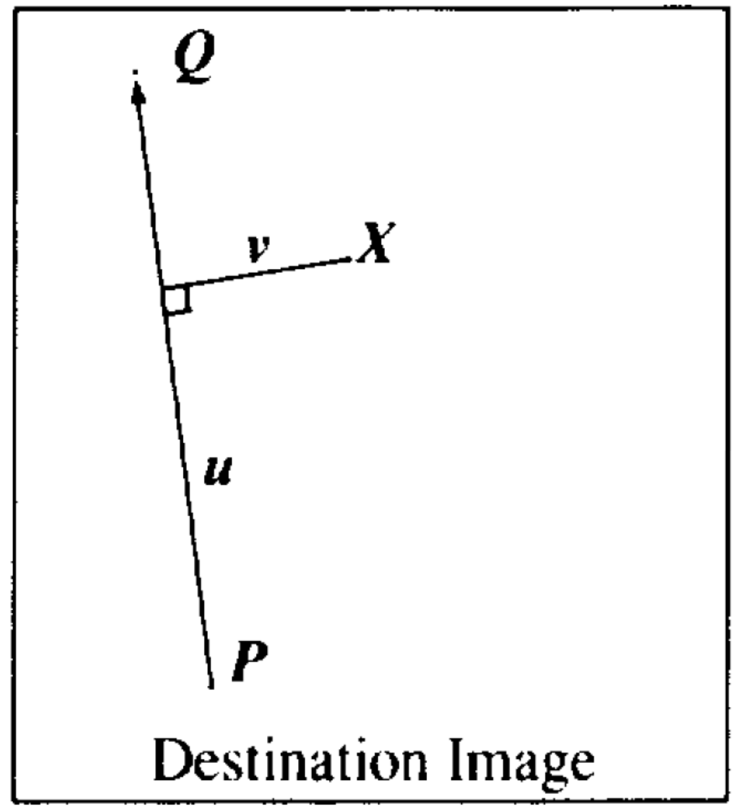
$$v = \frac{(X - P) \cdot Perpendicular(Q - P)}{\| Q - P \|} \quad (2)$$

$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot Perpendicular(Q' - P')}{\| Q' - P' \|} \quad (3)$$

Walk-Thru

$$u = \frac{(X - P) \cdot (Q - P)}{\| Q - P \|^2}$$

dot product and L2 norm



Example of Dot Product

$$(3,2) \cdot (4,5) = (3 * 4) + (2 * 5) = 12 + 10$$

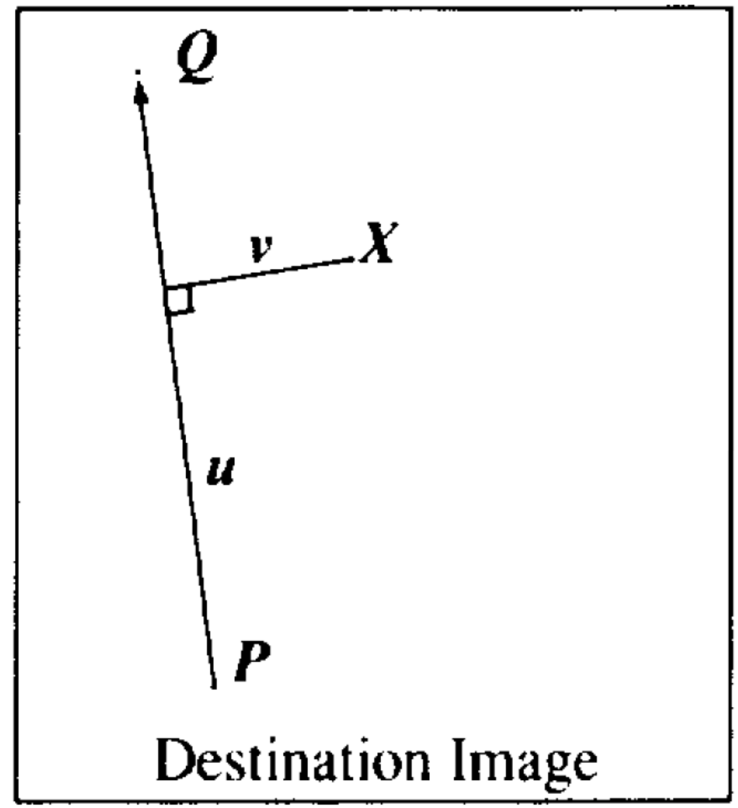
L2 norm

is the length/magnitude of the vector

Walk-Thru

$$u = \frac{(X - P) \cdot (Q - P)}{\| Q - P \|^2}$$

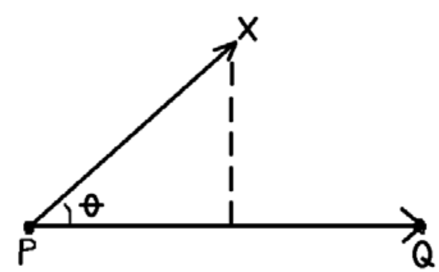
dot product and L2 norm



In words:

- u is the position along the line PQ
- u is a fraction
  - like a time parameter
  - u = 0 → P
  - u = 1 → Q

u gives position of X projected onto line PQ

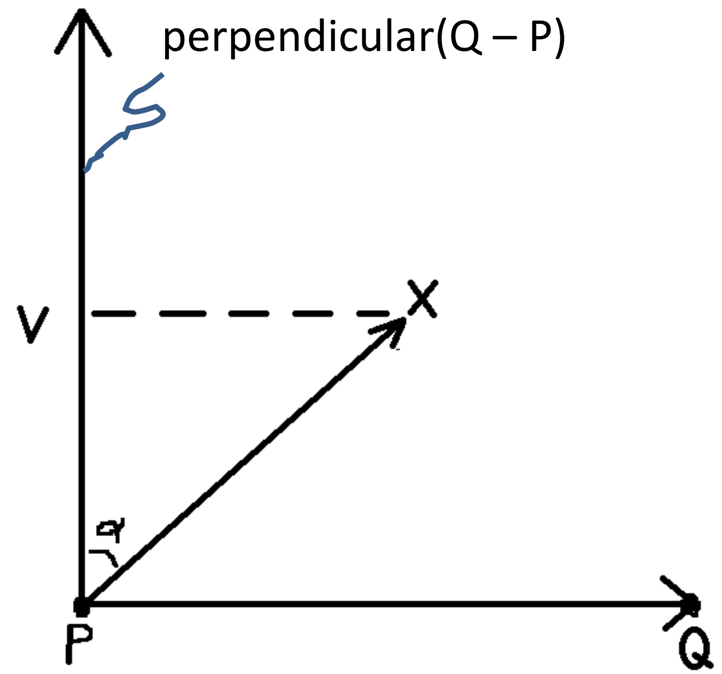


$$u = \frac{|PX||PQ|cos\theta}{|PQ||PQ|}$$

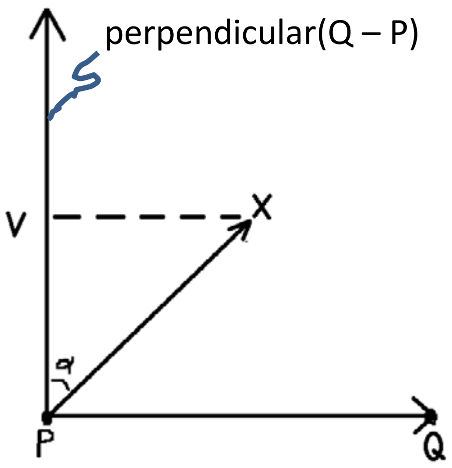
Walk-Thru

$$v = \frac{(X - P) \cdot Perpendicular(Q - P)}{\| Q - P \|}$$

Perpendicular()
returns the vector perpendicular to,
and the same length as, the input vector



Aside Example: Application of Transforms



Calculation of Perpendicular(PQ)

assume P is (0, 0) and Q at (x, y)

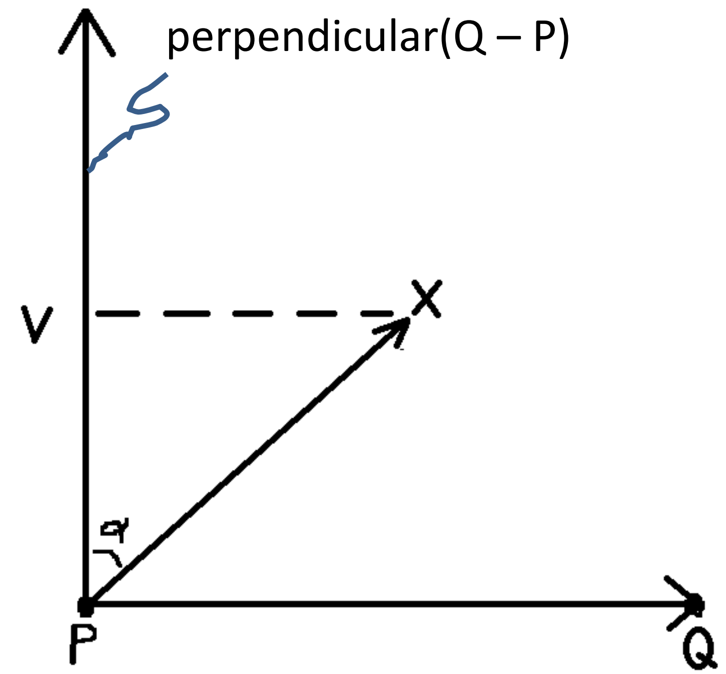
$$\begin{bmatrix} \cos\frac{\pi}{2} & -\sin\frac{\pi}{2} \\ \sin\frac{\pi}{2} & \cos\frac{\pi}{2} \end{bmatrix} \overrightarrow{PQ} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -y \\ x \end{bmatrix}$$

Walk-Thru

$$v = \frac{(X - P) \cdot Perpendicular\,(Q - P)}{\| Q - P \|}$$

$$= \frac{|PX||perpendicular(PQ)|(cos\alpha)}{|PQ|}$$

$$= |PX|(cos\alpha)$$



v is NOT normalized = distance in pixels from the PQ line

Walk-Thru

Using u and v, we now compute X’

$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot Perpendicular(Q' - P')}{\| Q' - P' \|}$$

Walk-Thru

Using u and v, we now compute X'

$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot Perpendicular(Q' - P')}{\| Q' - P' \|}$$

$$= P' + u \cdot (P'Q') + v \left(\frac{perpendicular(P'Q')}{|P'Q'|} \right)$$

note: this is a unit vector

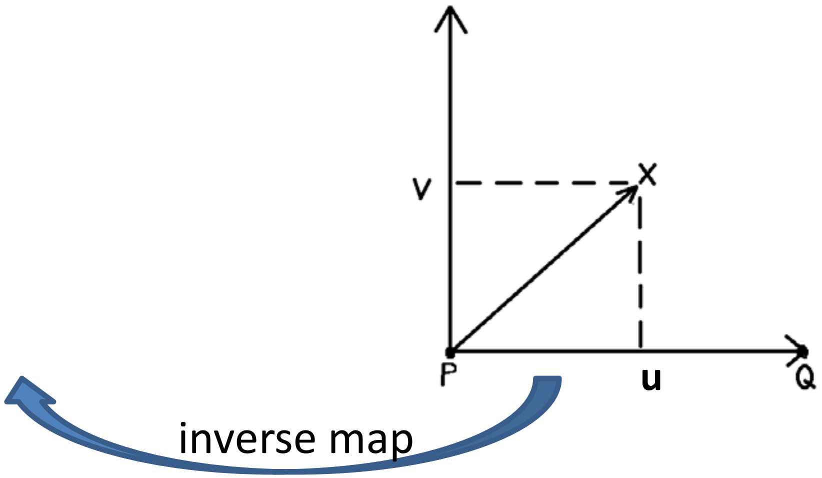
Walk-Thru

Using u and v, we now compute X'

$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot Perpendicular(Q' - P')}{\| Q' - P' \|}$$

$$= P' + u \cdot (P'Q') + v\left(\frac{perpendicular(P'Q')}{|P'Q'|}\right)$$

note: this is a unit vector



Walk-Thru

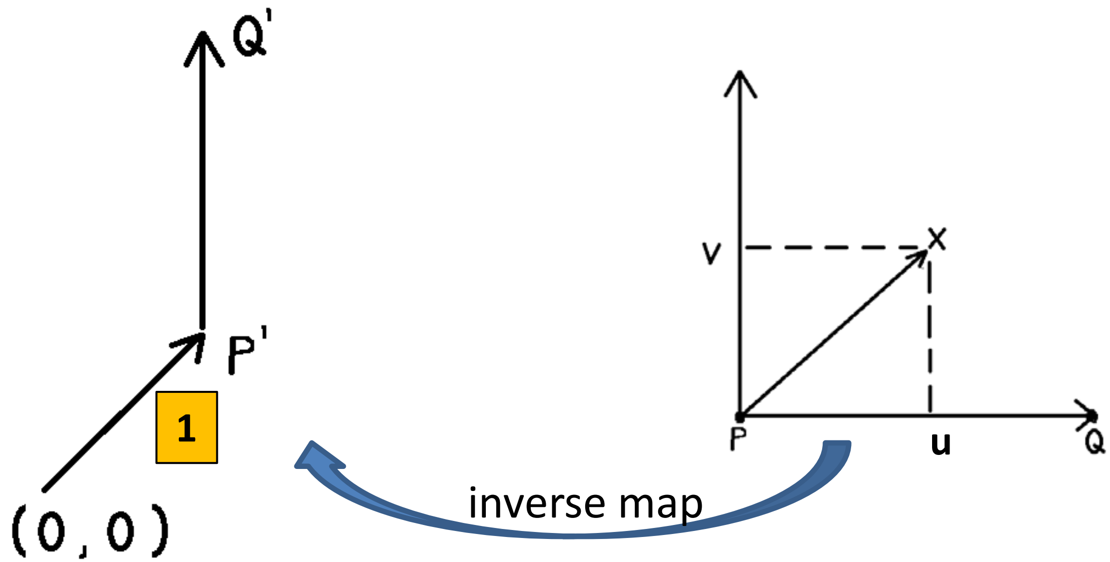
Using u and v, we now compute X'

$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot Perpendicular(Q' - P')}{\| Q' - P' \|}$$

$$= P' + u \cdot (P'Q') + v\left(\frac{perpendicular(P'Q')}{|P'Q'|}\right)$$

1

note: this is a unit vector



Walk-Thru

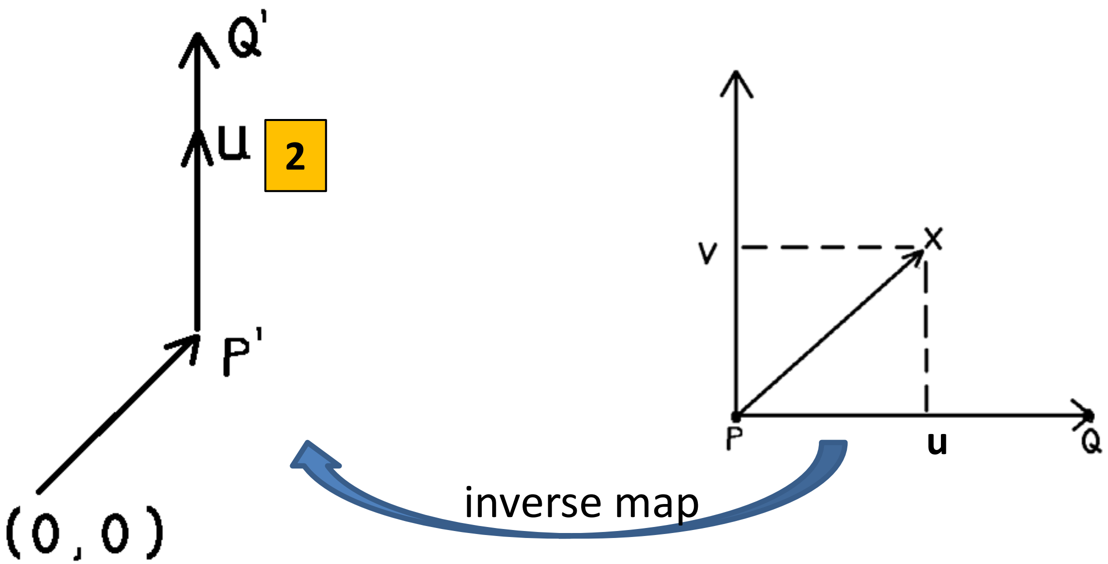
Using u and v, we now compute X'

$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot Perpendicular(Q' - P')}{\| Q' - P' \|}$$

$$= P' + u \cdot (P'Q') + v\left(\frac{perpendicular(P'Q')}{|P'Q'|}\right)$$

2

note: this is a unit vector



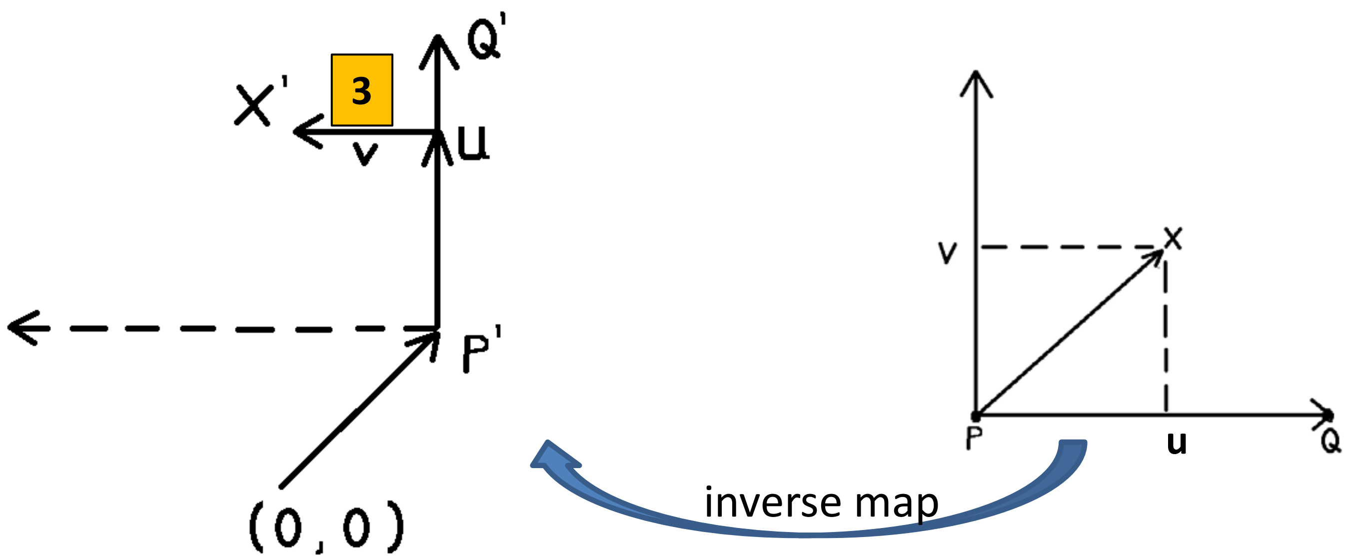
Walk-Thru

Using u and v, we now compute X'

$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot Perpendicular(Q' - P')}{\| Q' - P' \|}$$

$$= P' + u \cdot (P'Q') + v\left(\frac{perpendicular(P'Q')}{|P'Q'|}\right)$$

3



Finished: Transform with 1 Line Pair

```
For each pixel X in the destination image
        find the corresponding u,v
        find the X' in the source image for that u,v
        destinationImage(X) = sourceImage(X')
```

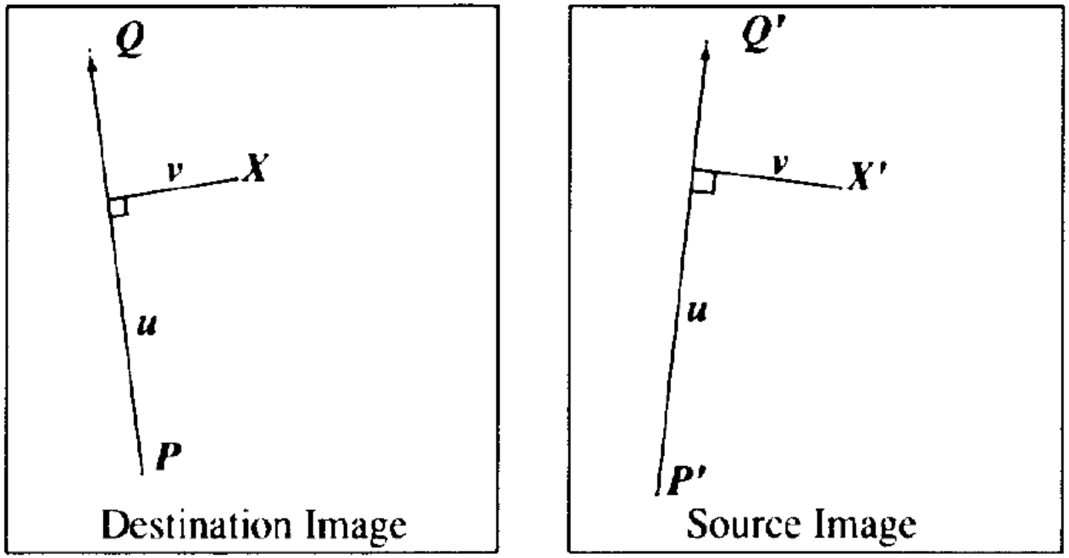


Figure 1: Single line pair

Results: Each Using One Pair of Lines

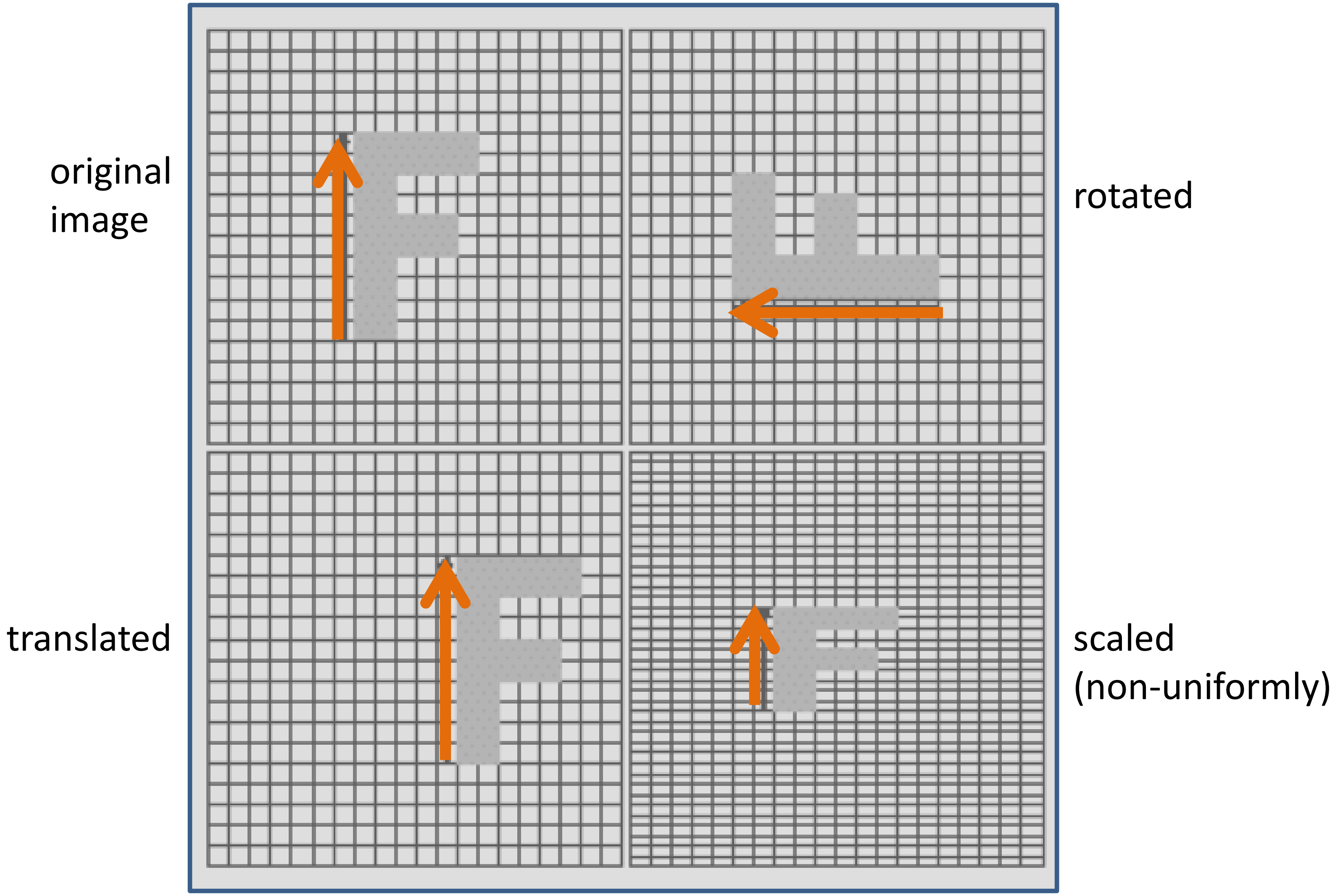


Figure 2: Single line pair examples

Transform with Multiple Line Pairs

- More complex and makes use of fields of influence from each line to identify which pixel X’ in the source image that must be sampled for each pixel X in the destination image

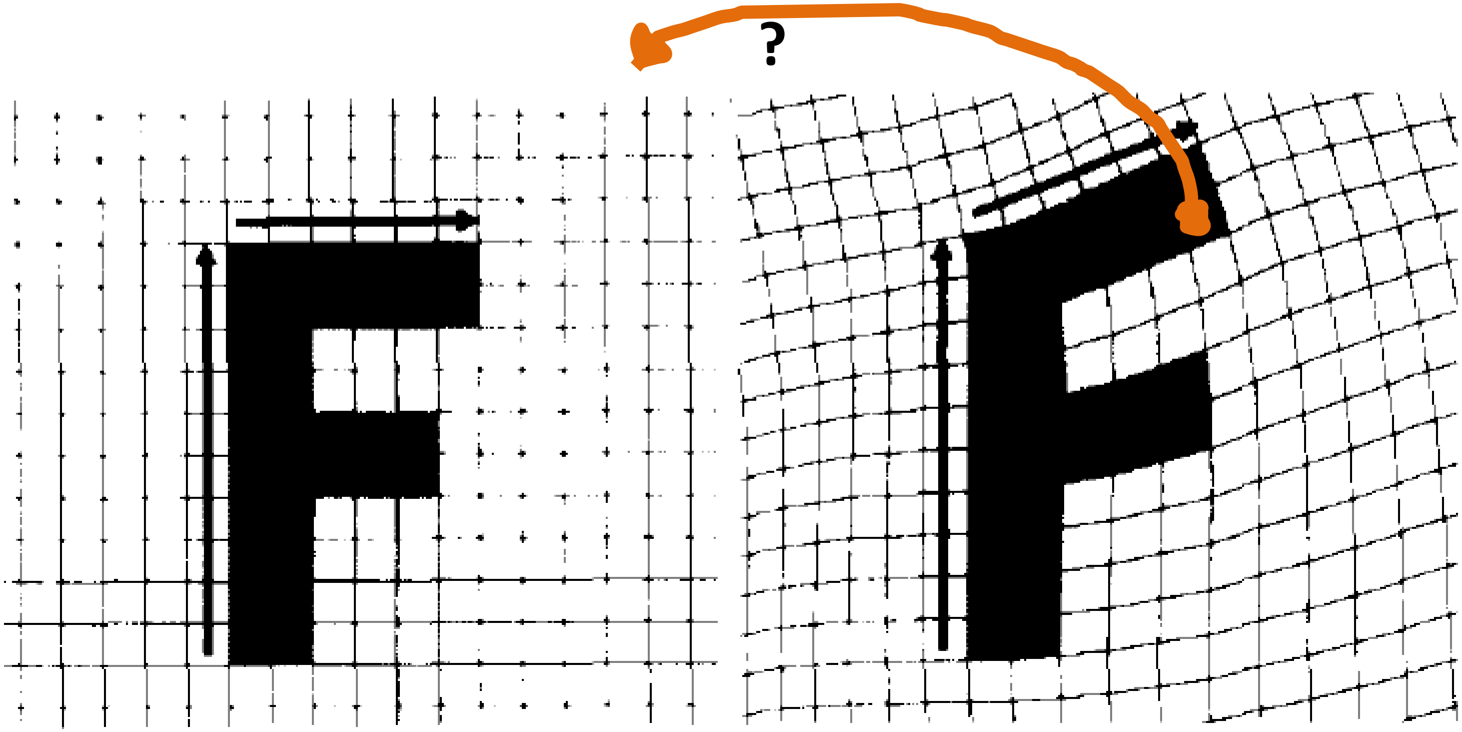
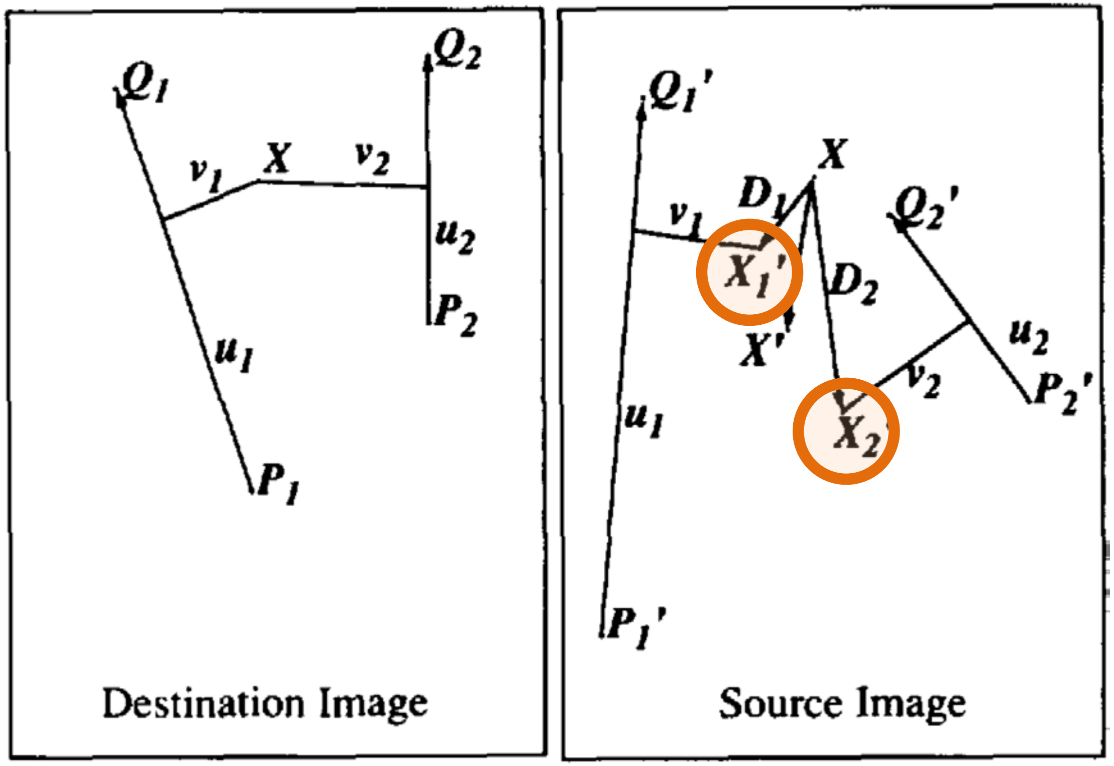


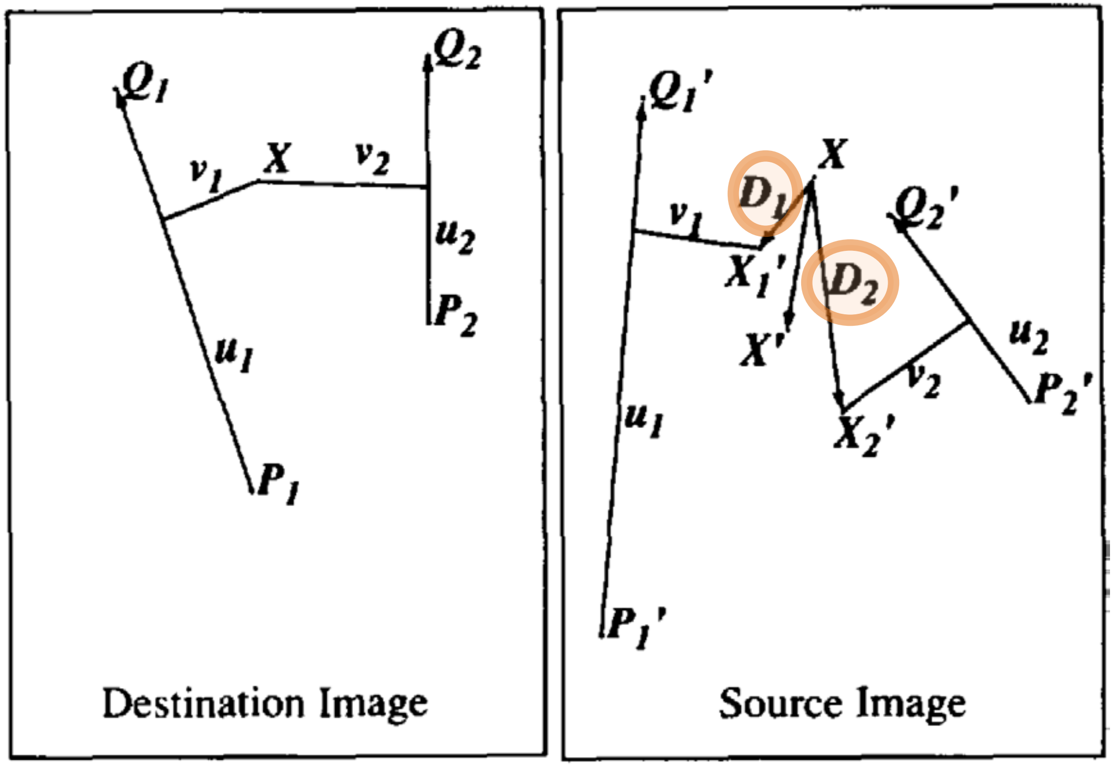
Figure 4: Multiple line pair example

Transform with Multiple Line Pairs



X'_i are obtained by single line pairs

Transform with Multiple Line Pairs

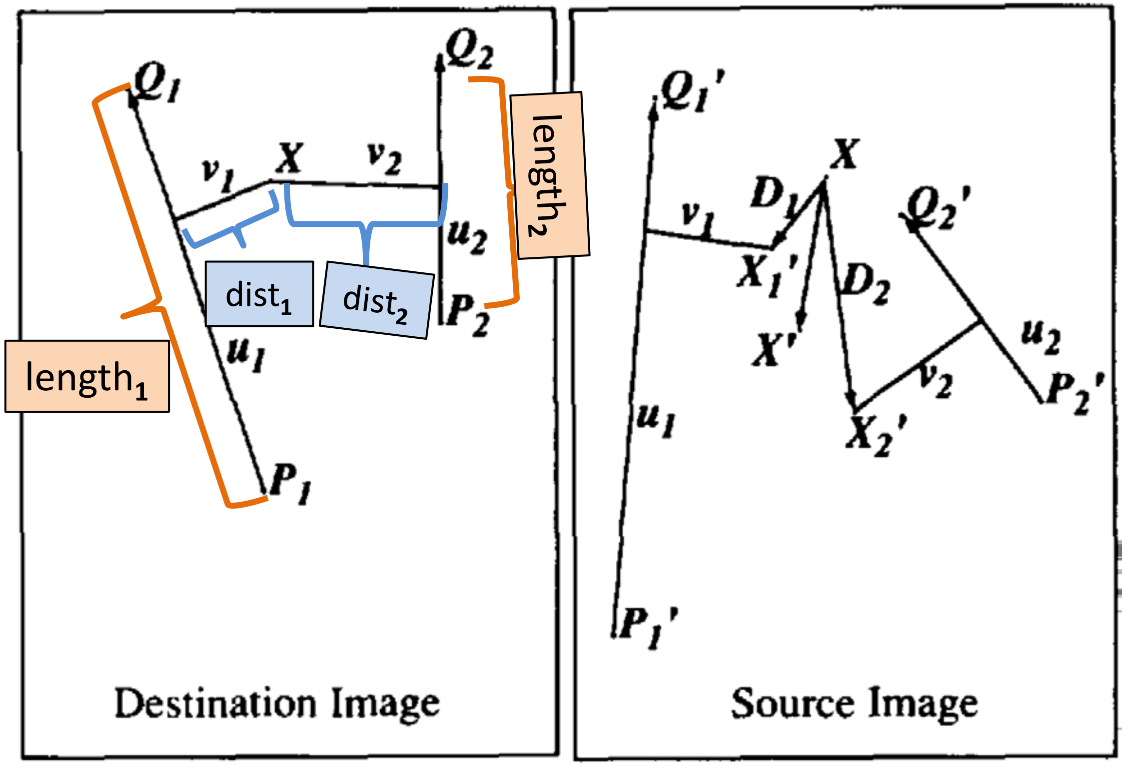


X'$_i$ are obtained by single line pairs

$D_i = X'_i - X$

displacement

Transform with Multiple Line Pairs



X'$_i$ are obtained by single line pairs

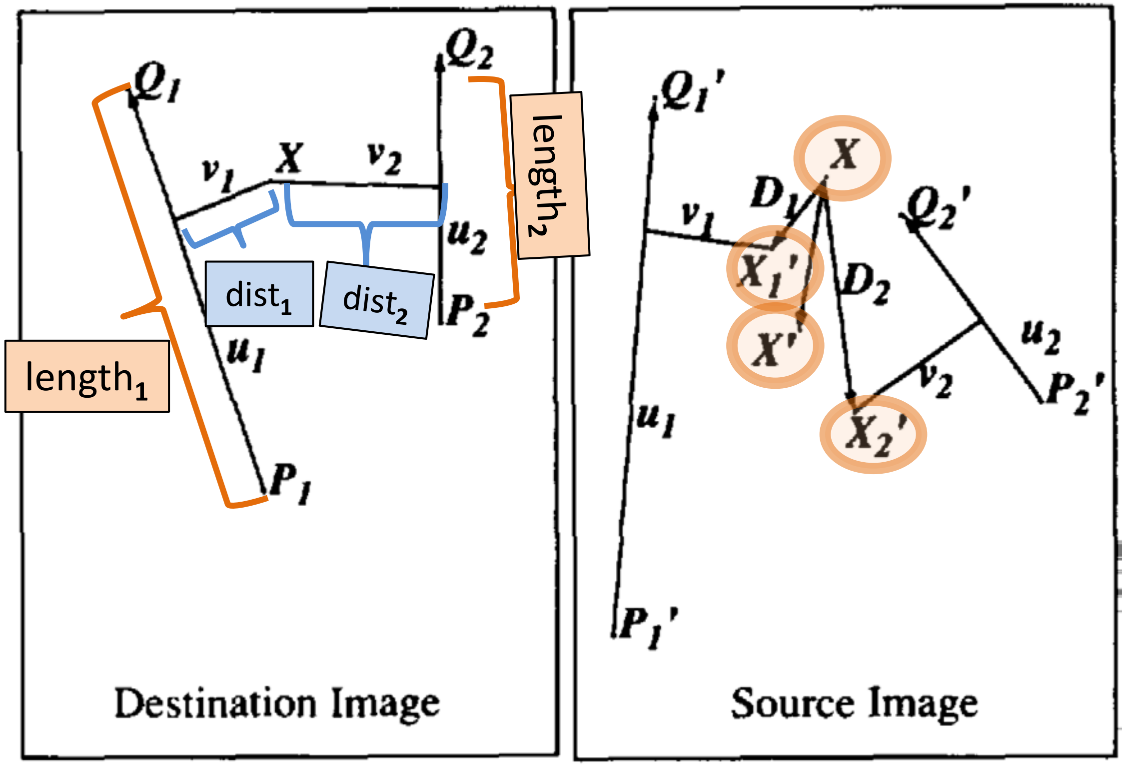
$D_i = X'_i - X$

displacement

$$weight_i = \left(\frac{(length_i)^p}{a + dist_i}\right)^b$$

a, p, and b
are constants that can be used to change the relative effect of the lines

Transform with Multiple Line Pairs



X'_i are obtained by single line pairs

$D_i = X'_i - X$

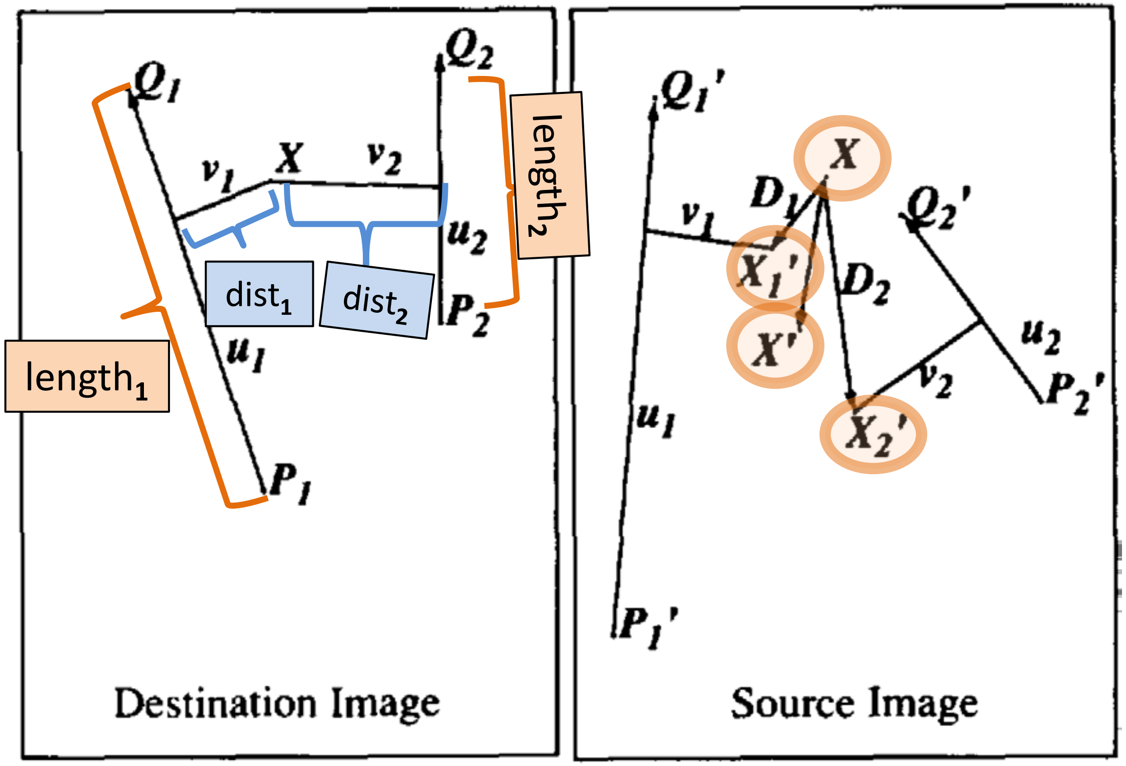
displacement

$$weight_i = \left(\frac{(length_i)^p}{a + dist_i}\right)^b$$

a, p, and b
are constants that can be used to change the relative effect of the lines

$$X' = X + \frac{\sum(weight_i * D_i)}{\sum weight_i}$$

Transform with Multiple Line Pairs



X'$_i$ are obtained by single line pairs

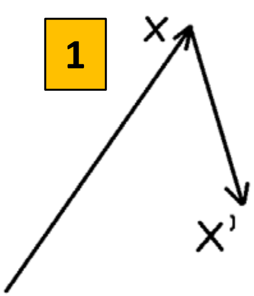
$D_i = X'_i - X$

displacement

$$weight_i = \left(\frac{(length_i)^p}{a + dist_i}\right)^b$$

a, p, and b
are constants that can be used to change the relative effect of the lines

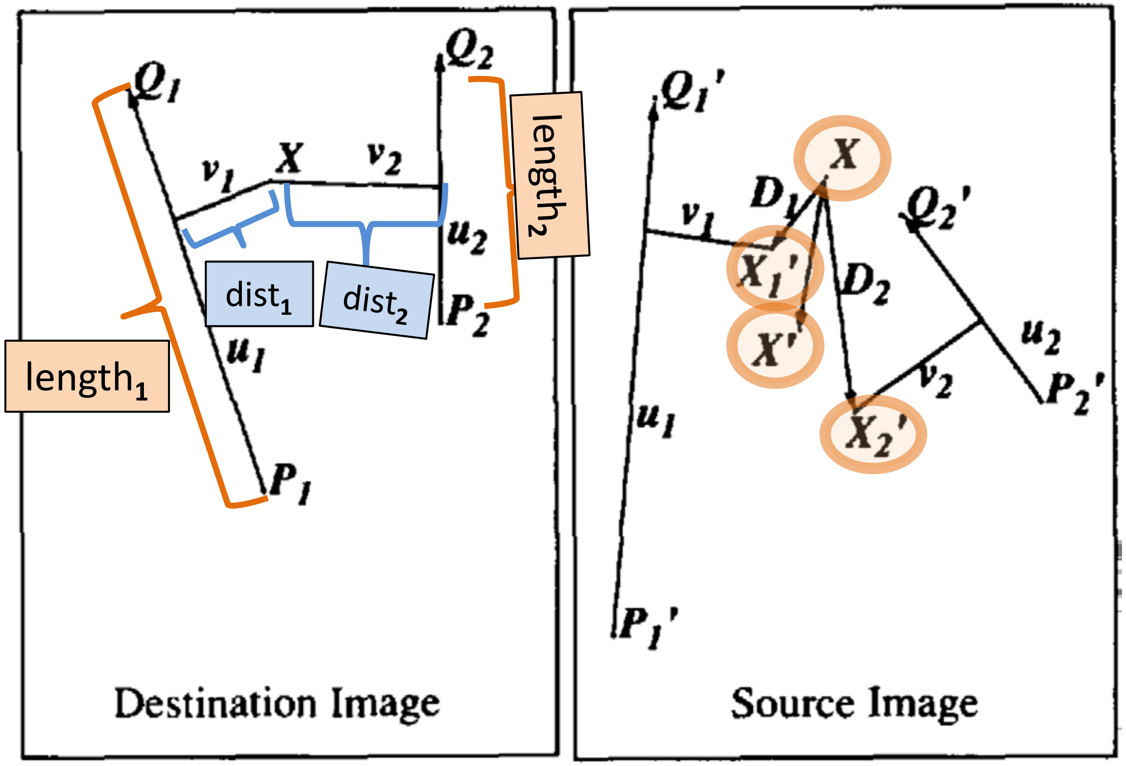
1



$$X' = X + \frac{\sum(weight_i * D_i)}{\sum weight_i}$$

1

Transform with Multiple Line Pairs

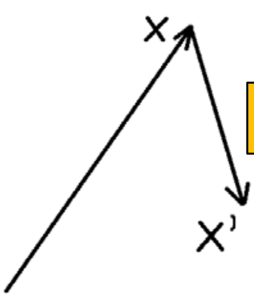


X'_i are obtained by single line pairs

$D_i = X'_i - X$
displacement

$$weight_i = \left(\frac{(length_i)^p}{a + dist_i}\right)^b$$

a, p, and b
are constants that can be used to change the relative effect of the lines



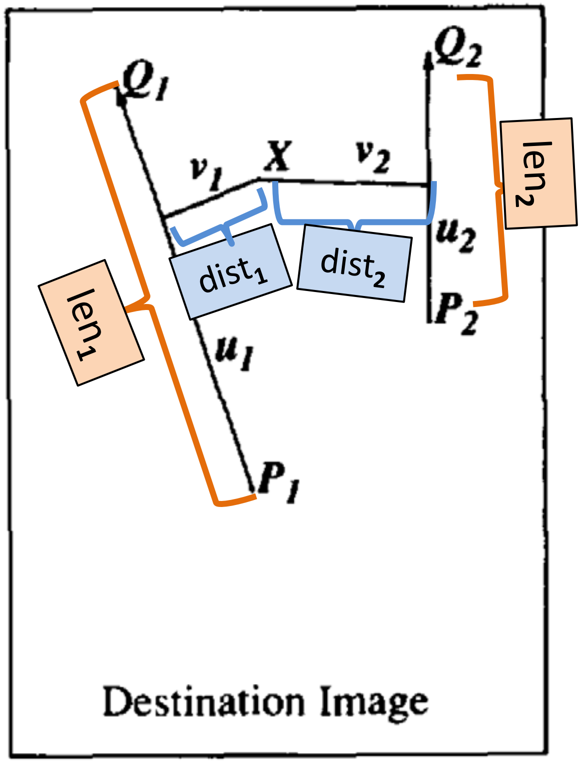
2

plus the weighted sum of D_i
(displacement)

$$X' = X + \frac{\sum(weight_i * D_i)}{\sum weight_i}$$

2

Constant Details



$$weight_i = \left(\frac{(len_i)^p}{a + dist_i}\right)^b$$

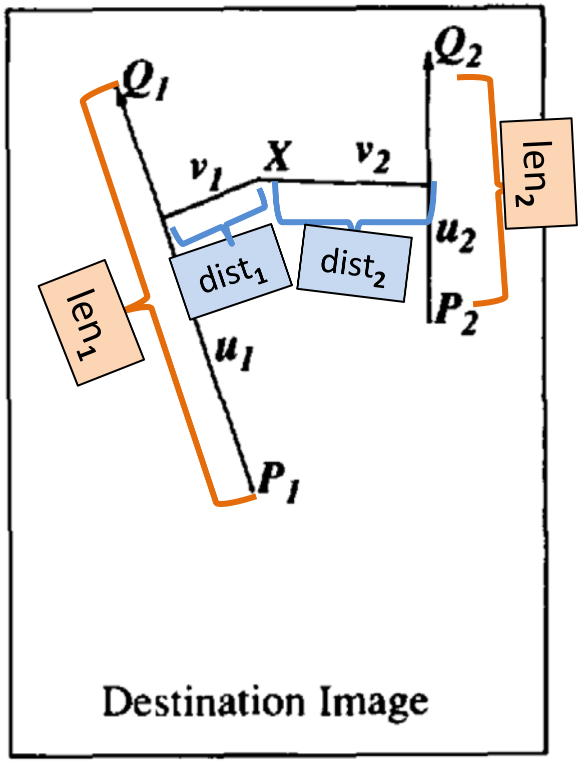
a – adjust the influence of distance; when a is larger, distance has smaller influence. When a is close to 0, pixels on a line go exactly to corresponding line as the line's influence is infinite. $a + dist_i \rightarrow 0$ then $weight_i \rightarrow \infty$

b – adjust the influence of length/distance ratio; when b is larger, longer and nearer line has larger influence. When b is 0, all lines have the same influence on each pixel.

p – adjust the influence of length; when p is larger, longer lines have greater influence than shorter lines. When p is 0, different lengths have the same influence.

a, p, and b are constants that can be used to change the relative effect of the lines

Constant Details



$$weight_i = \left(\frac{(len_i)^p}{a + dist_i} \right)^b$$

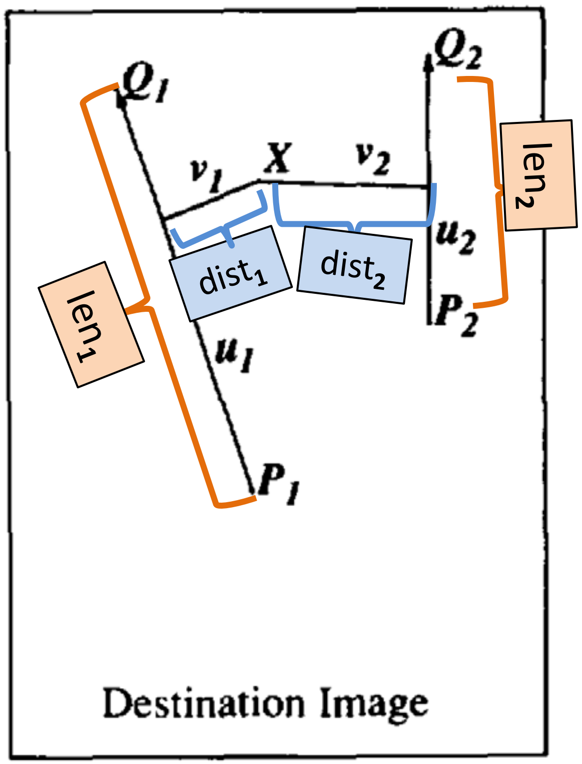
a – adjust the influence of distance; when a is larger, distance has smaller influence. When a is close to 0, pixels on a line go exactly to corresponding line as the line's influence is infinite.

b – adjust the influence of length/distance ratio; when b is larger, longer and nearer line has larger influence. When b is 0, all lines have the same influence on each pixel.

p – adjust the influence of length; when p is larger, longer lines have greater influence than shorter lines. When p is 0, different lengths have the same influence.

a, p, and b are constants that can be used to change the relative effect of the lines

Constant Details



$$weight_i = \left(\frac{(len_i)^p}{a + dist_i}\right)^b$$

a – adjust the influence of distance; when a is larger, distance has smaller influence. When a is close to 0, pixels on a line go exactly to corresponding line as the line's influence is infinite.

b – adjust the influence of length/distance ratio; when b is larger, longer and nearer line has larger influence. When b is 0, all lines have the same influence on each pixel.

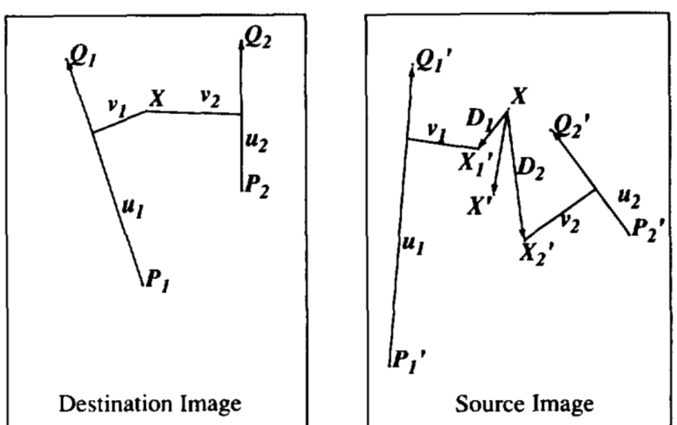
p – adjust the influence of length; when p is larger, longer lines have greater influence than shorter lines. When p is 0, different lengths have the same influence.

a, p, and b are constants that can be used to change the relative effect of the lines

Pseudocode: Multiple Line Pairs

```
For each pixel X in the destination
        DSUM = (0,0)
        weightsum = 0
        For each line Pi Qi
                calculate u,v based on Pi Qi
                calculate X'i based on u,v       and Pi'Qi'
                calculate displacement Di = Xi' - Xi for this line
                dist = shortest distance from X to Pi Qi
                weight = (lengthp / (a + dist))b
                DSUM += Di *    weight
                weightsum += weight
        X' = X + DSUM / weightsum
        destinationImage(X) = sourceImage(X')
```

$$X' = X + \frac{\sum(weight_i * D_i)}{\sum weight_i}$$



Results with Multiple Line Pairs

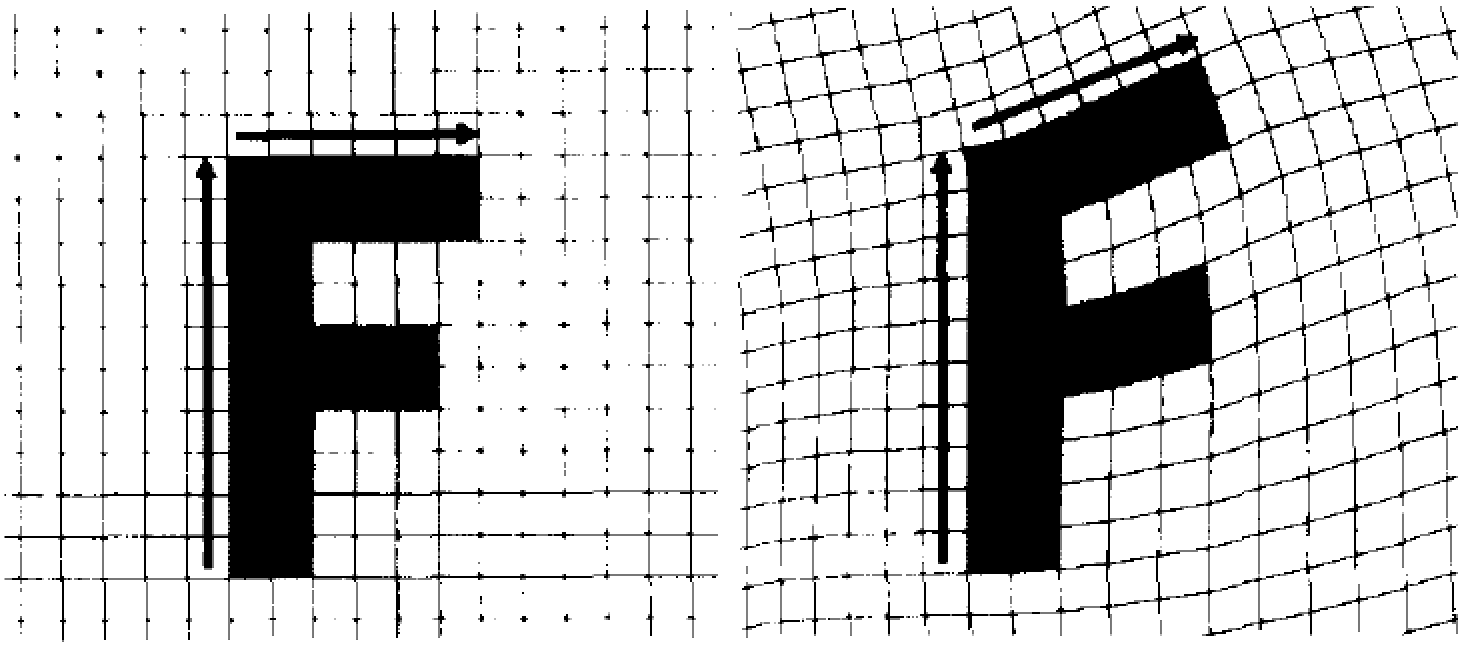


Figure 4: Multiple line pair example

Morphing Between 2 Still Images

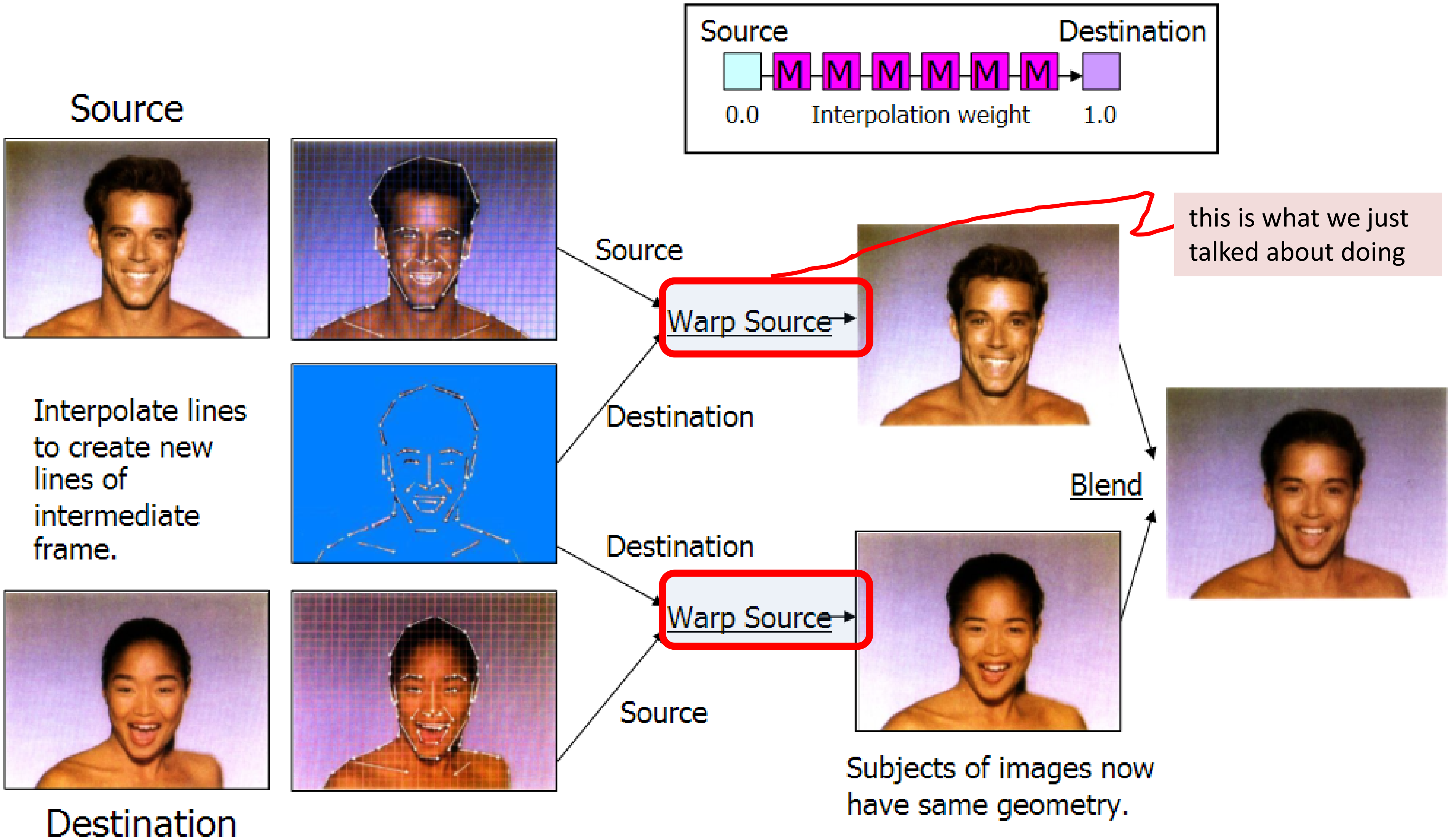
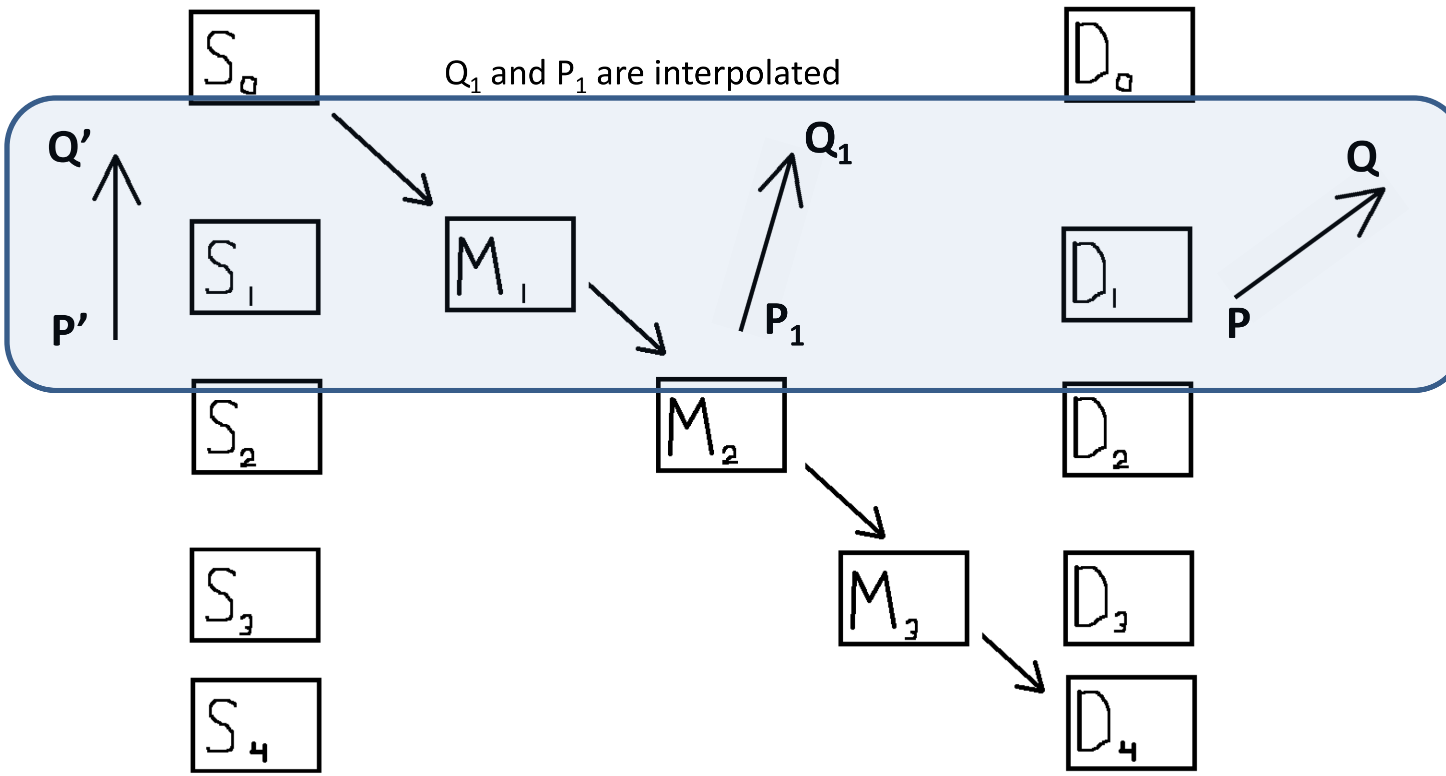


Image source: http://www.cs.mcgill.ca/~kaleigh/graphics/graphics767/beierneely_talk/frame.htm

Animated Sequences

Specify lines on keyframes and interpolate lines on other frames

Example:
$S_{1.5}$, $D_{1.5}$ would be interpolated between Q_1, P_1 and Q_2, P_2.



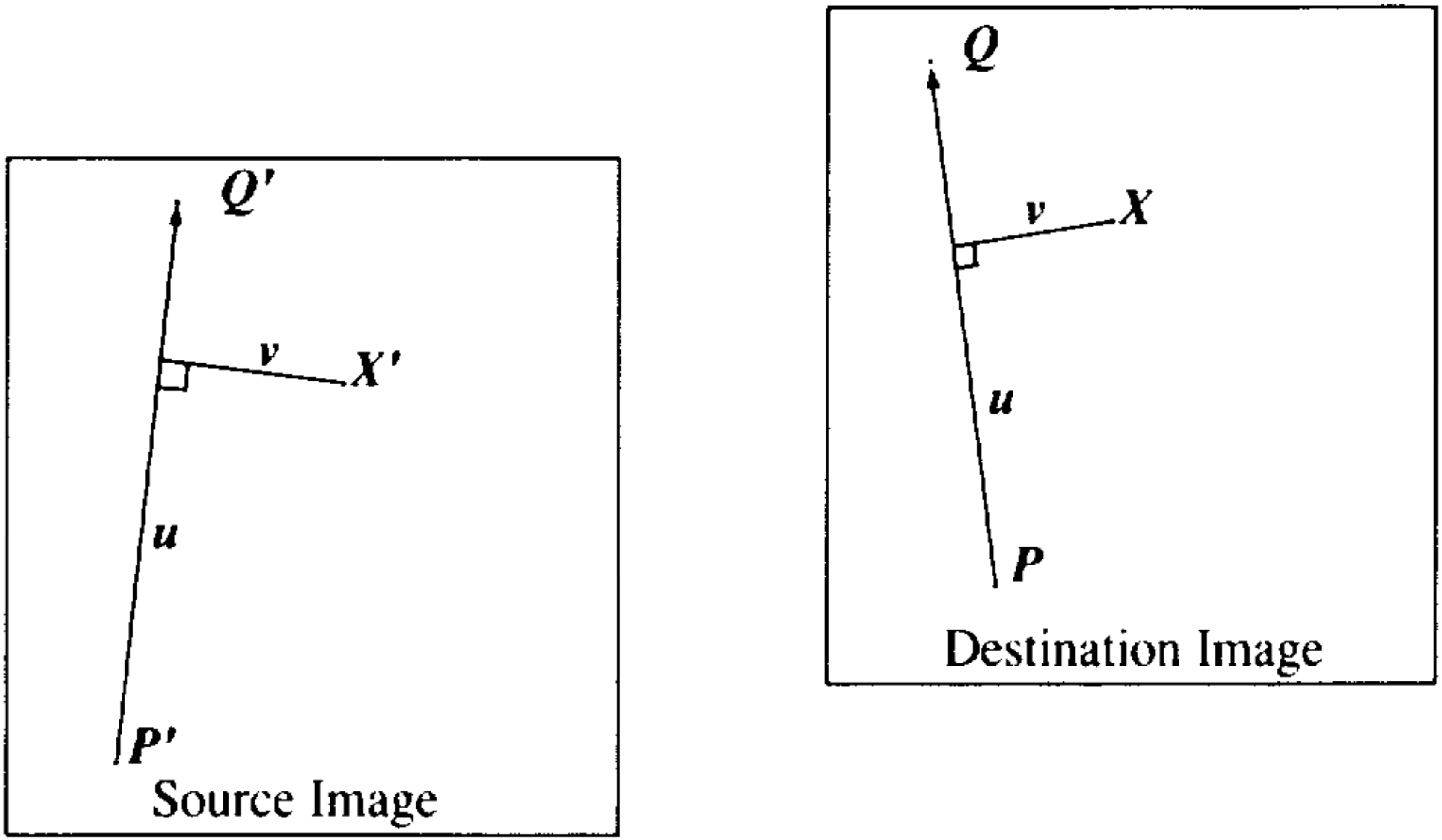
Note:
Interpolation can be
1. on the end points of line pairs
OR
2. on the center, orientation and length of line pairs

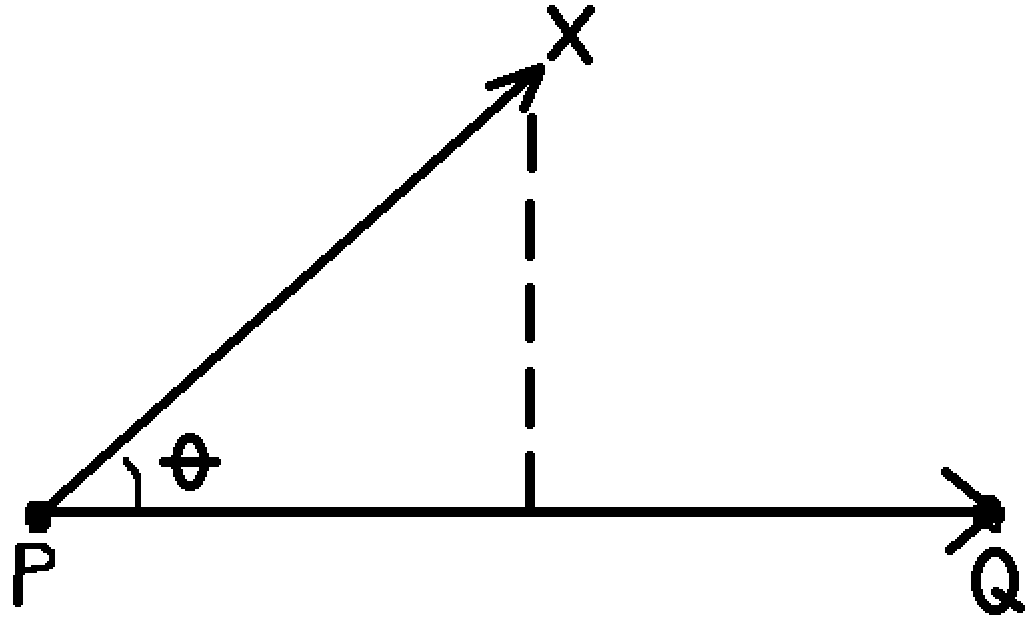
Questions?

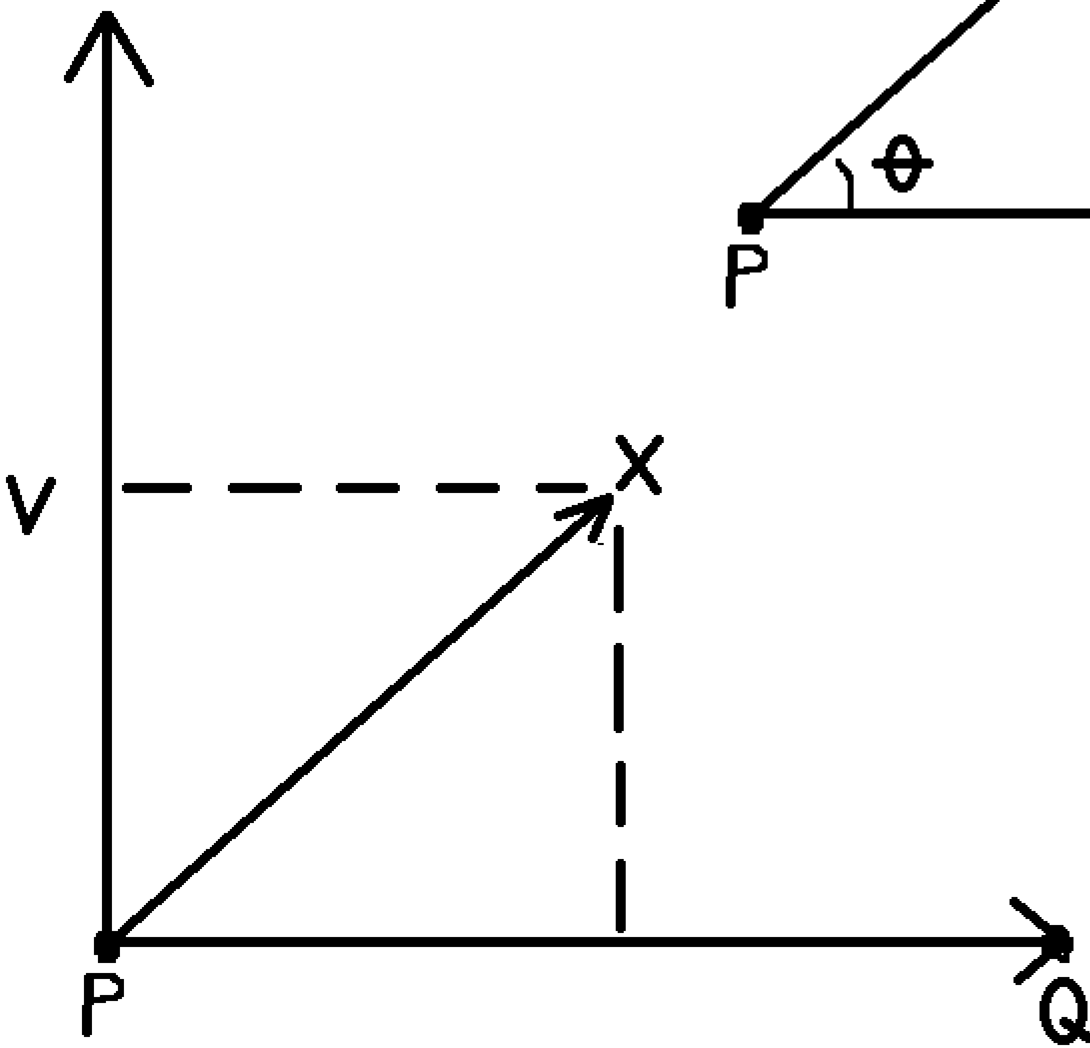
- Beyond D2L
  - Examples and information can be found online at:
    - *http://docdingle.com/teaching/cs.html*

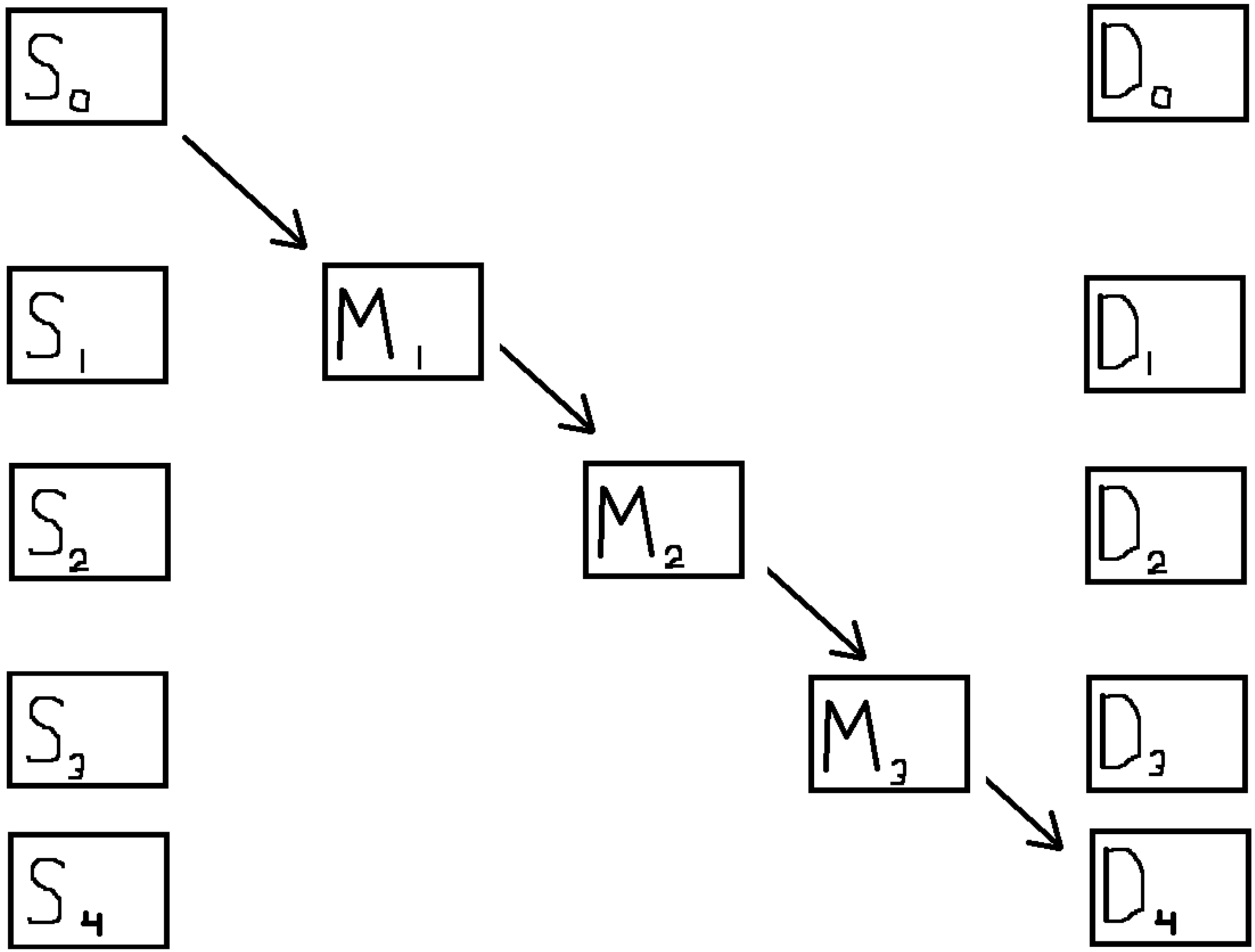
    - *Continue to more stuff as needed*

Extra Reference Stuff Follows


Q'
v
X'
u
P'
Source Image
Q
v
X
u
P
Destination Image


X
θ
P
Q


V
X
P
Q



S0
S1
S2
S3
S4
M1
M2
M3
D0
D1
D2
D3
D4

Credits

- Much of the content derived/based on slides for use with the book:
  - *Digital Image Processing,* Gonzalez and Woods

- Some layout and presentation style derived/based on presentations by
  - Donald House, Texas A&M University, 1999
  - Bernd Girod, Stanford University, 2007
  - Shreekanth Mandayam, Rowan University, 2009
  - Igor Aizenberg, TAMUT, 2013
  - Xin Li, WVU, 2014
  - George Wolberg, City College of New York, 2015
  - Yao Wang and Zhu Liu, NYU-Poly, 2015
  - Sinisa Todorovic, Oregon State, 2015