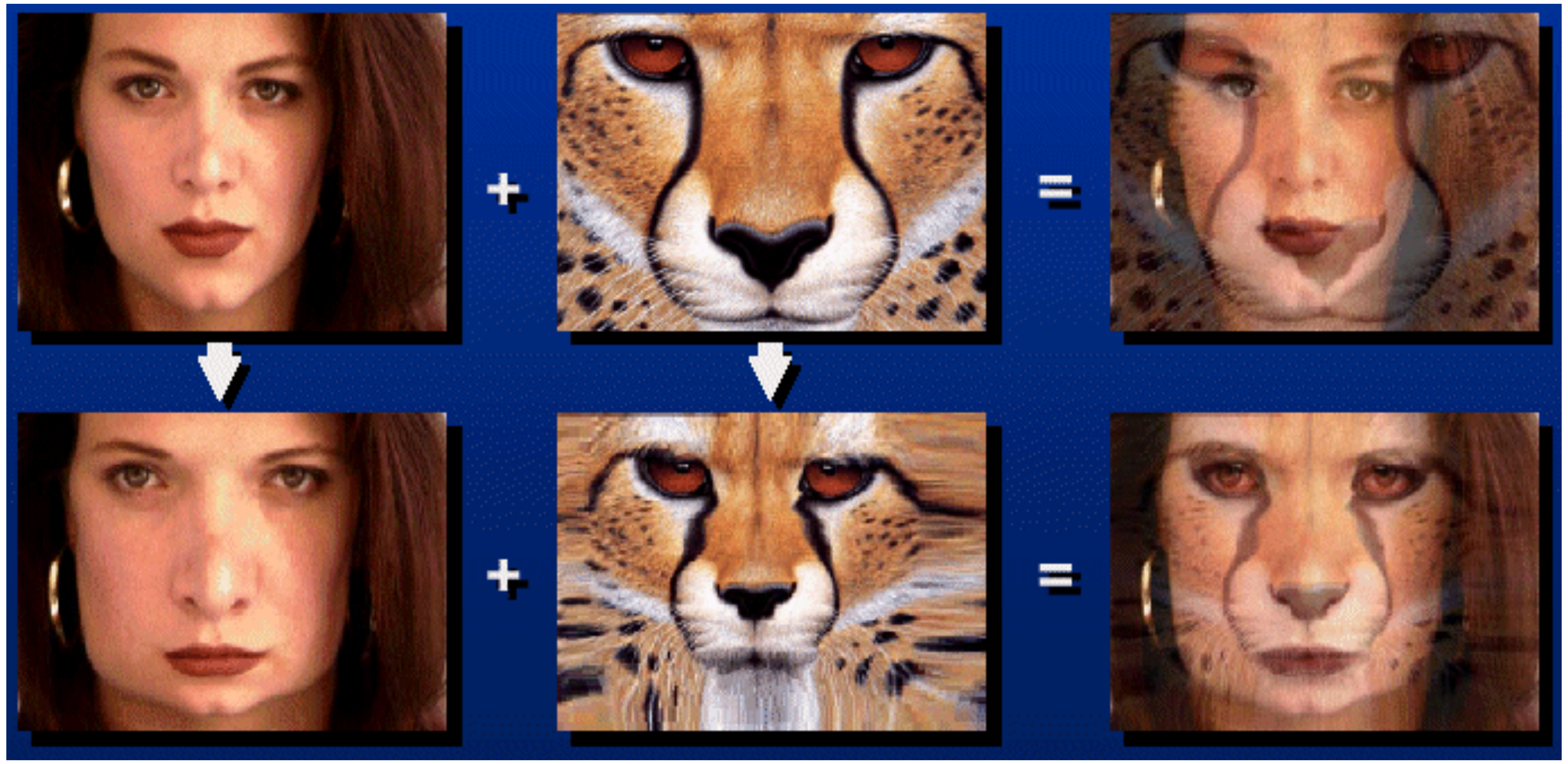
Morphing



For each intermediate frame I_t:

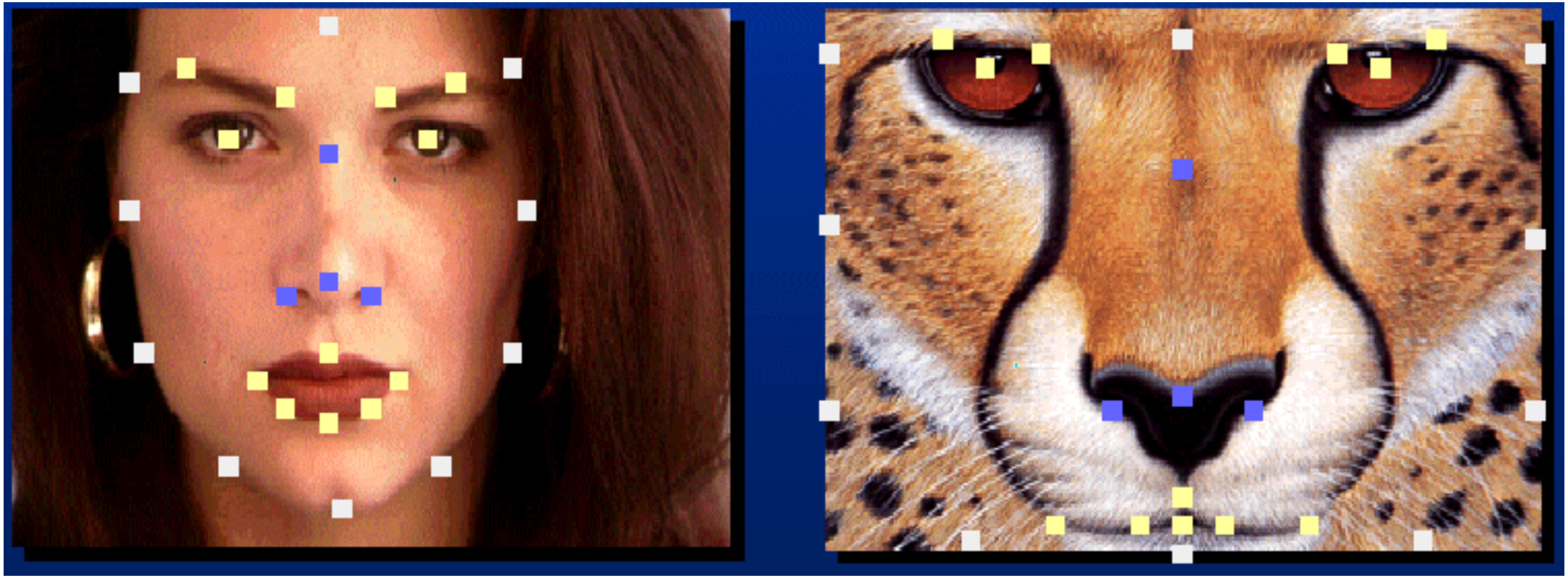
1. Interpolate feature locations

$$\mathbf{p}_i^t = (1-\alpha)\mathbf{p}_i^0 + \alpha\mathbf{p}_i^1$$

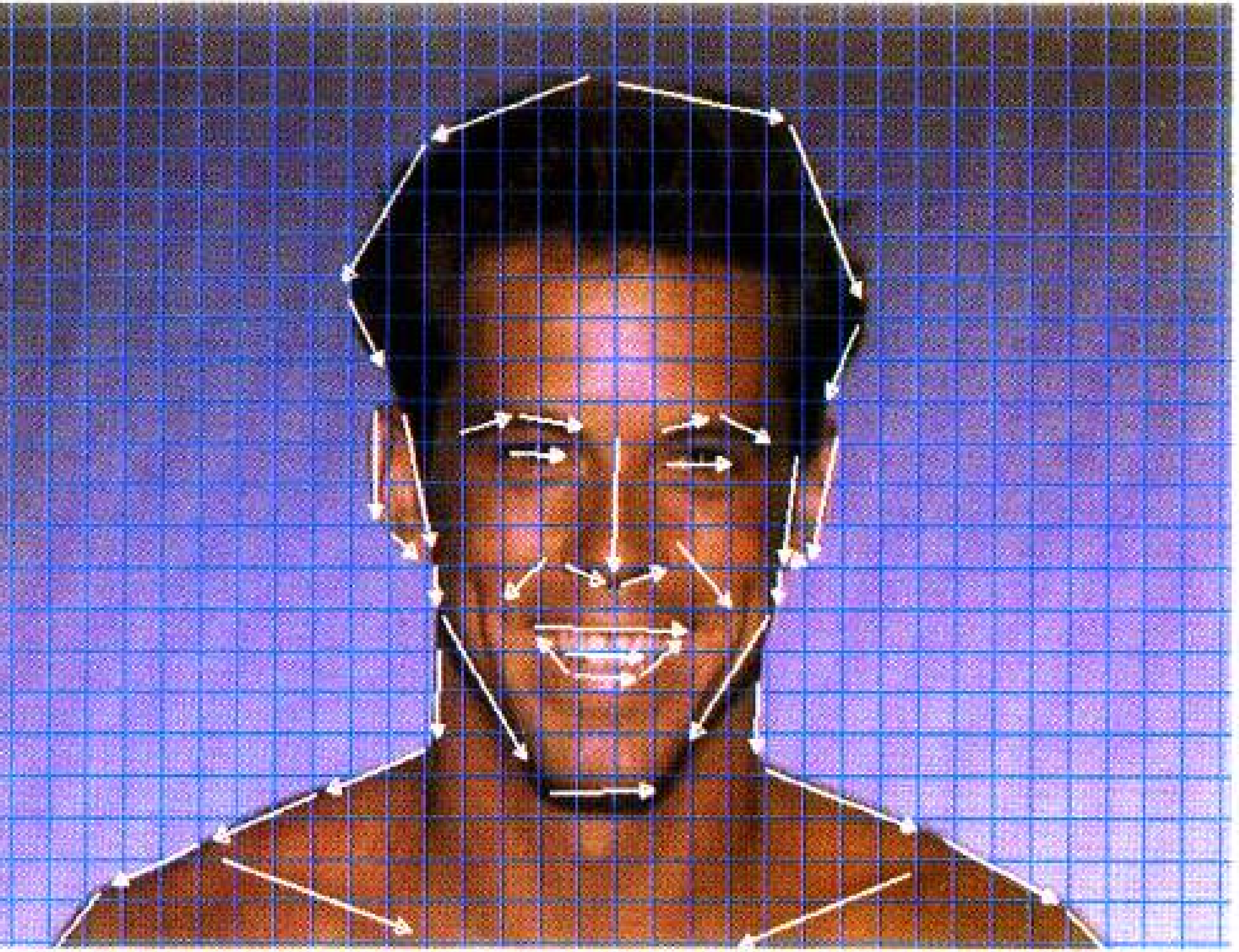
2. Perform two warps: one for I_0, one for I_1
    - Deduce a dense warp field from the pairs of features
    - Warp the pixels
3. Linearly interpolate the two warped images

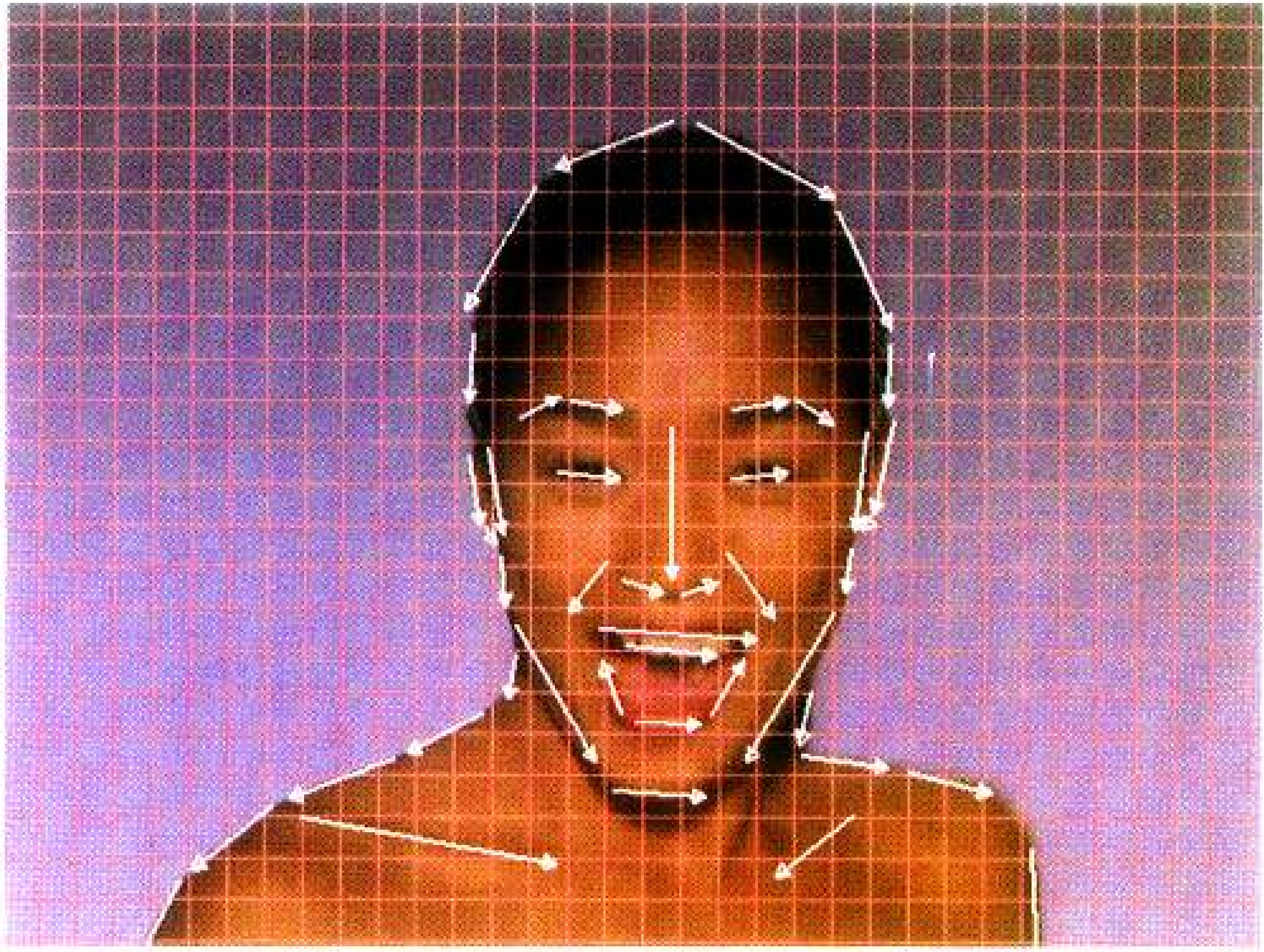
How can we specify the warp?

- Specify corresponding points
- *interpolate* to a complete warping function
- How do we do it?



Slide Alyosha Efros





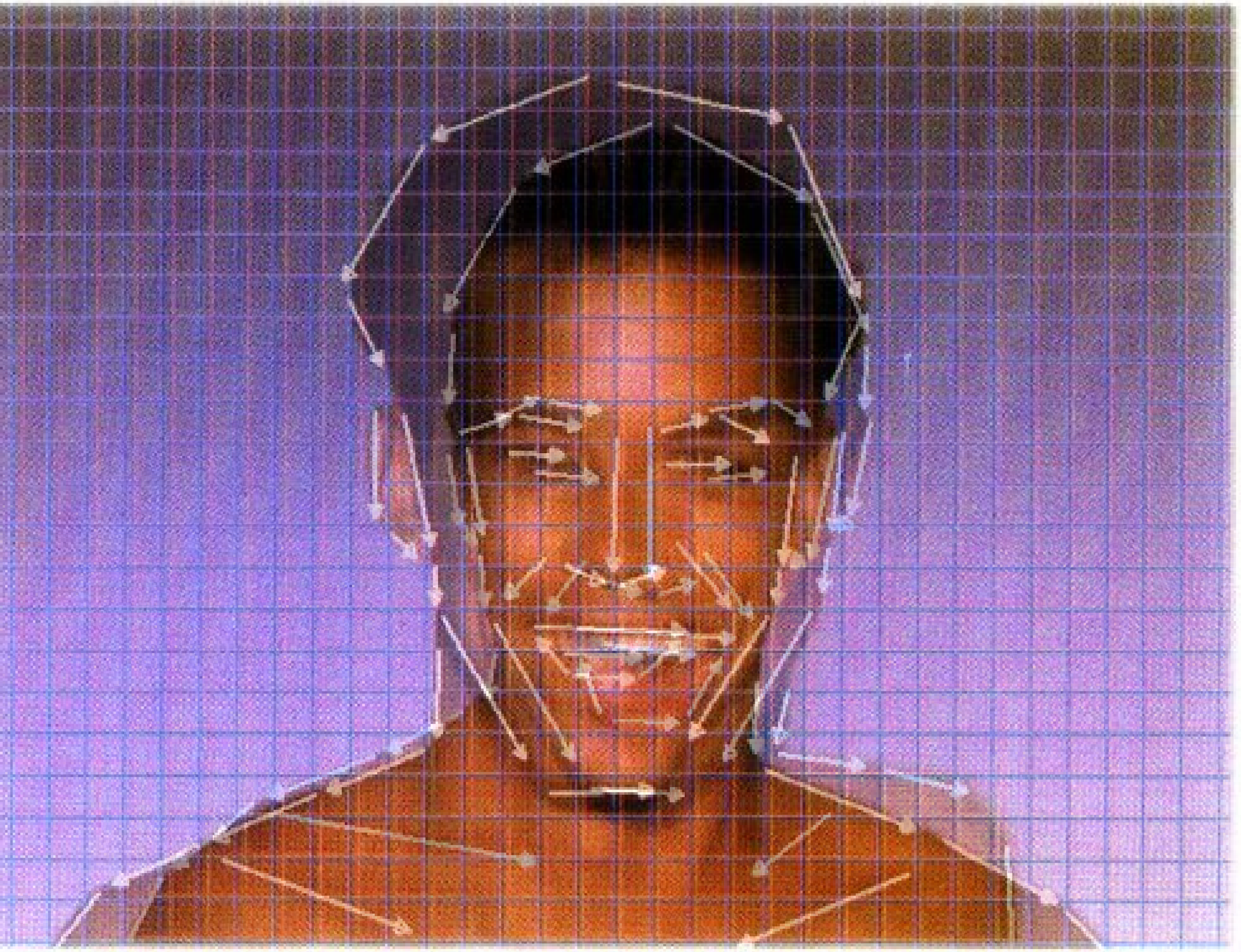


Image_0

Warp_0

Result

Image_1

Warp_1

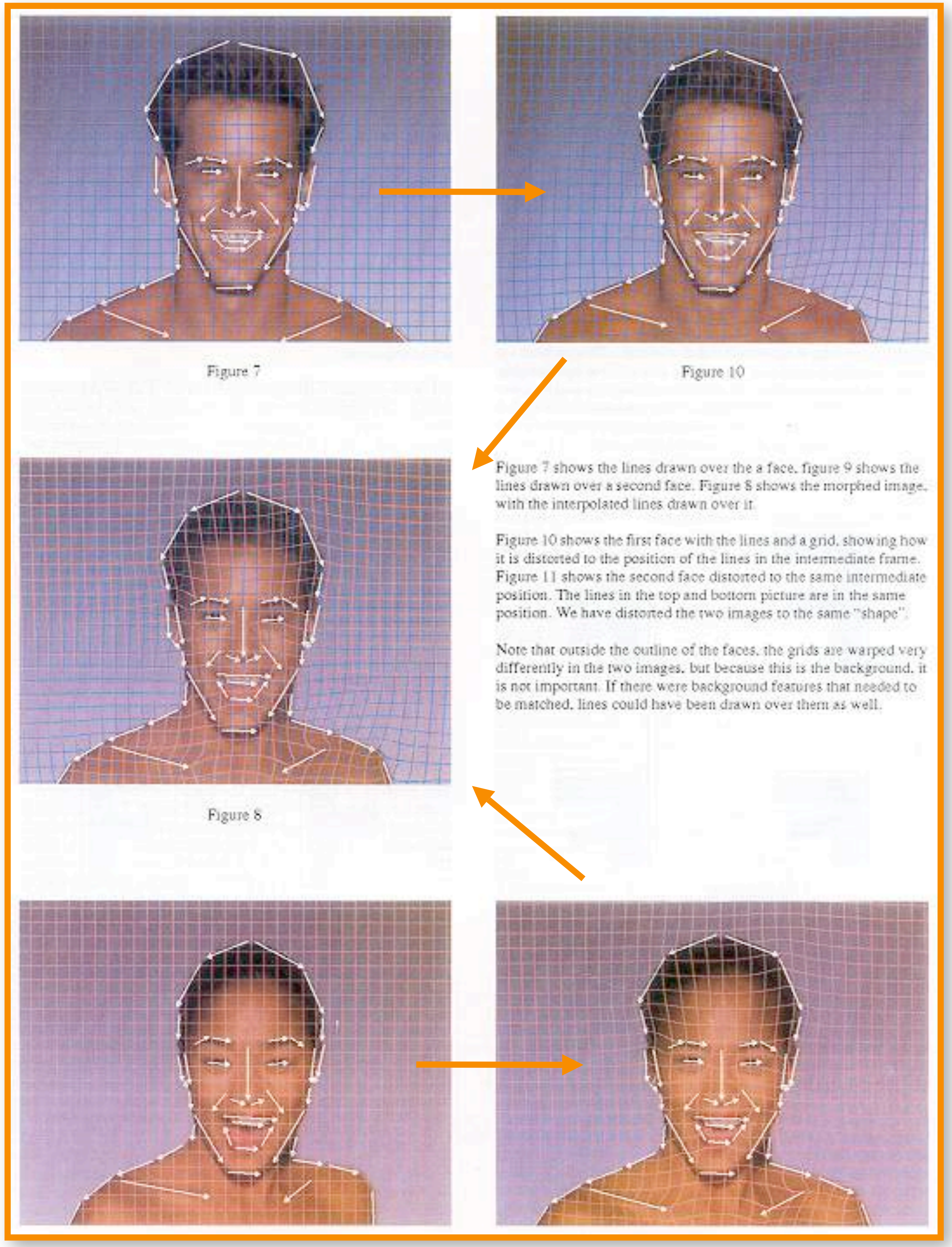


Figure 7

Figure 10

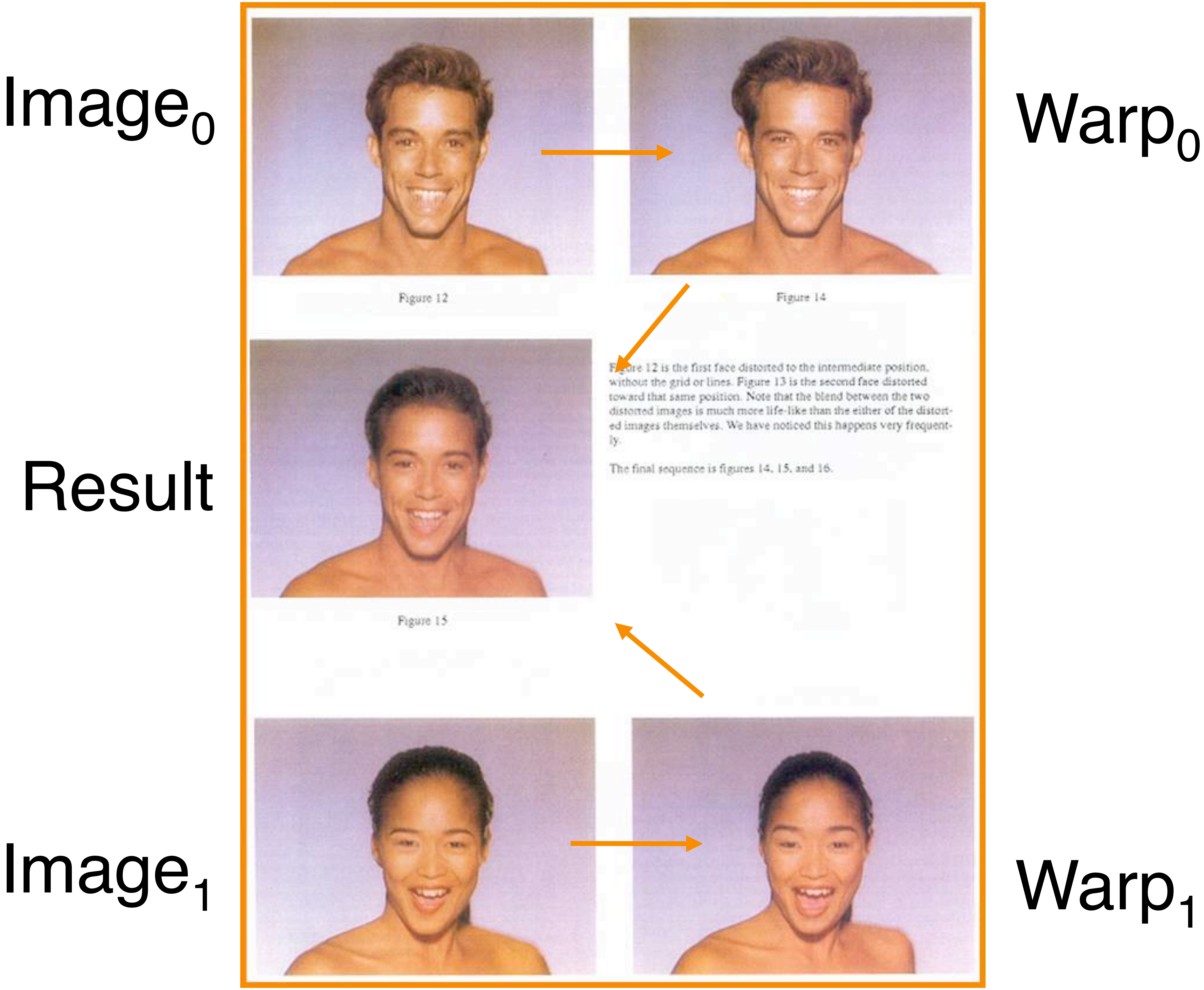
Figure 7 shows the lines drawn over the a face, figure 9 shows the lines drawn over a second face. Figure 8 shows the morphed image, with the interpolated lines drawn over it.

Figure 10 shows the first face with the lines and a grid, showing how it is distorted to the position of the lines in the intermediate frame. Figure 11 shows the second face distorted to the same intermediate position. The lines in the top and bottom picture are in the same position. We have distorted the two images to the same "shape".

Note that outside the outline of the faces, the grids are warped very differently in the two images, but because this is the background, it is not important. If there were background features that needed to be matched, lines could have been drawn over them as well.

Figure 8

slide from: http://www.cs.virginia.edu/~gfx/Courses/2010/IntroGraphics/Lectures/4-Image.pdf

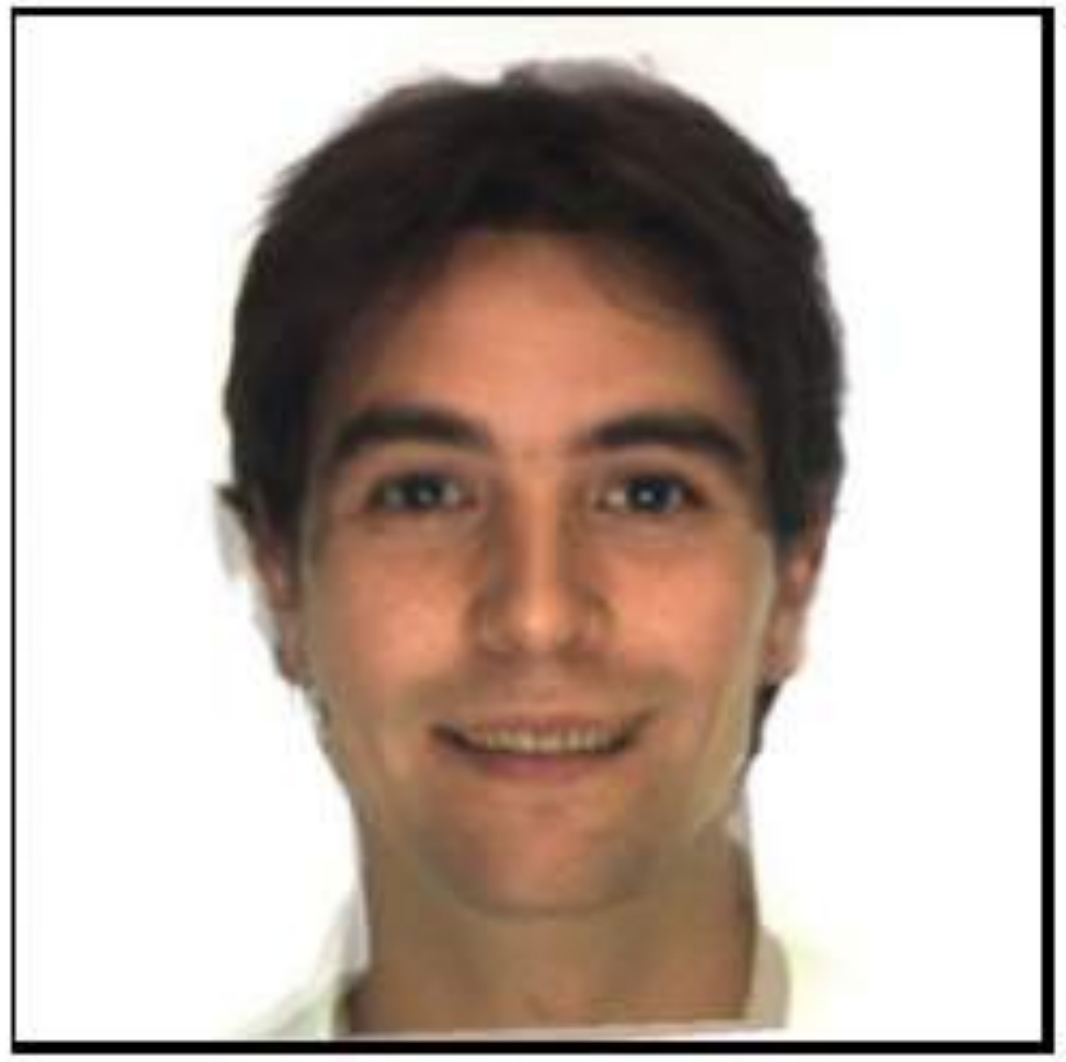


slide from: http://www.cs.virginia.edu/~gfx/Courses/2010/IntroGraphics/Lectures/4-Image.pdf

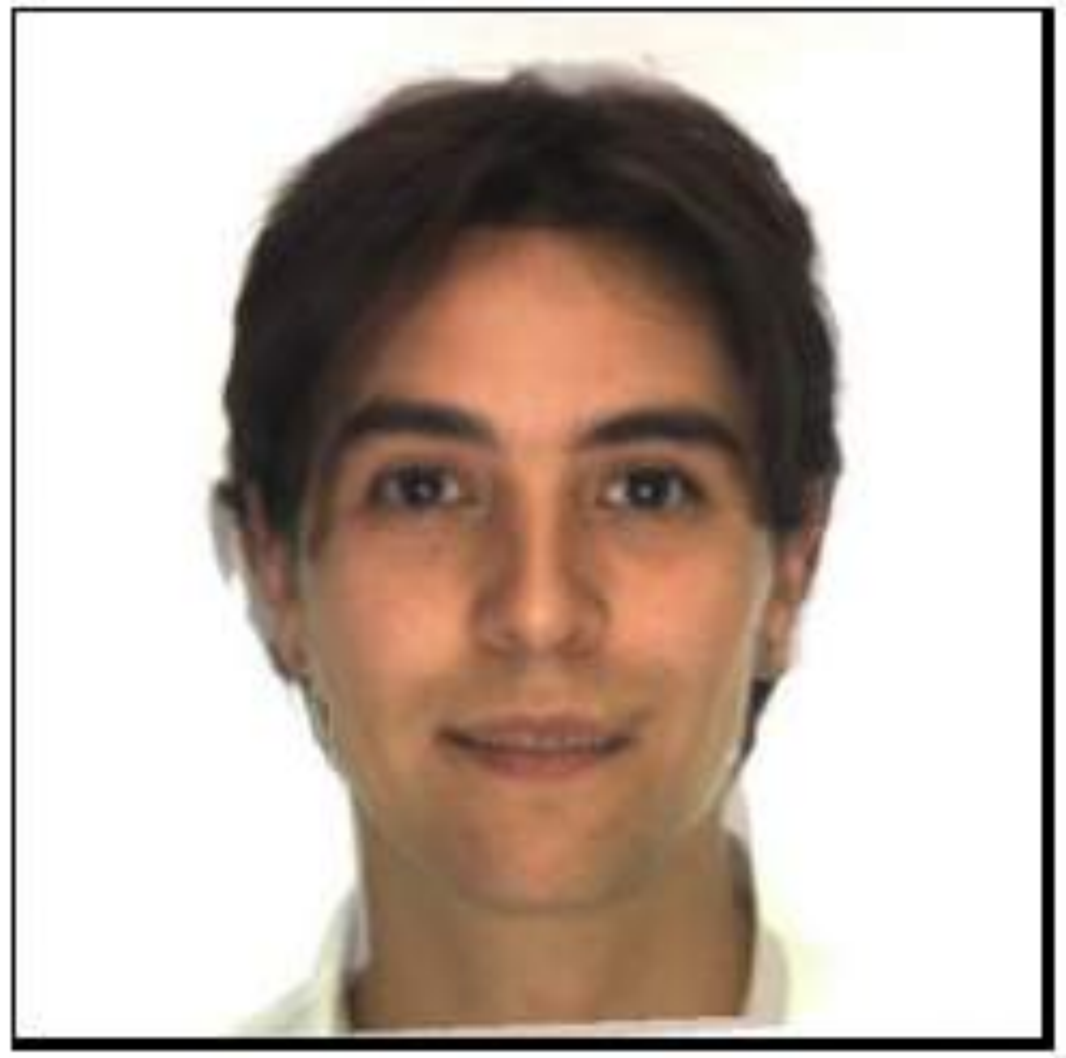
Extract foreground to avoid artifacts in the background

(c) $\alpha = 0.0$

(d) $\alpha = 0.2$



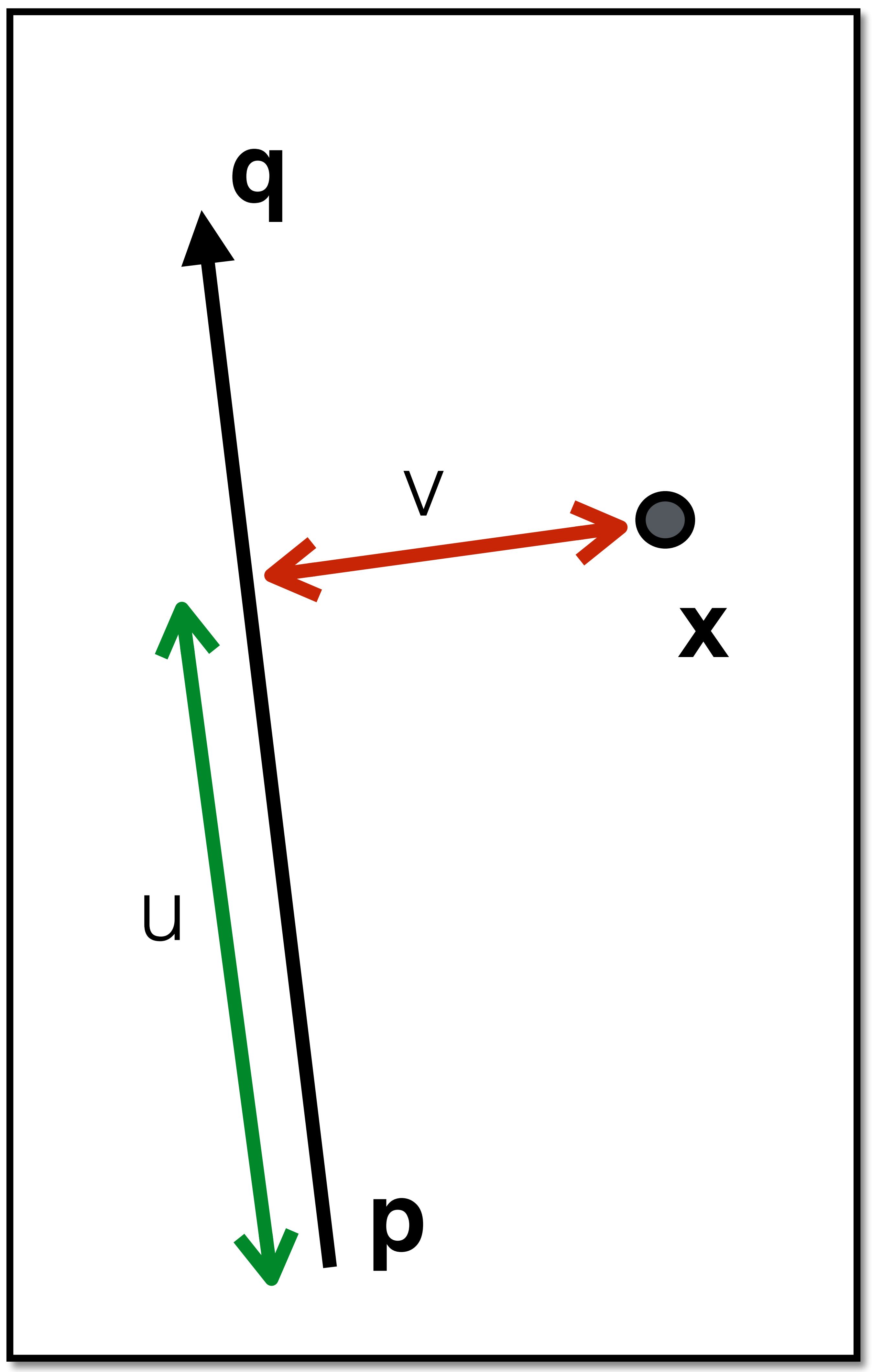
(e) $\alpha = 0.4$



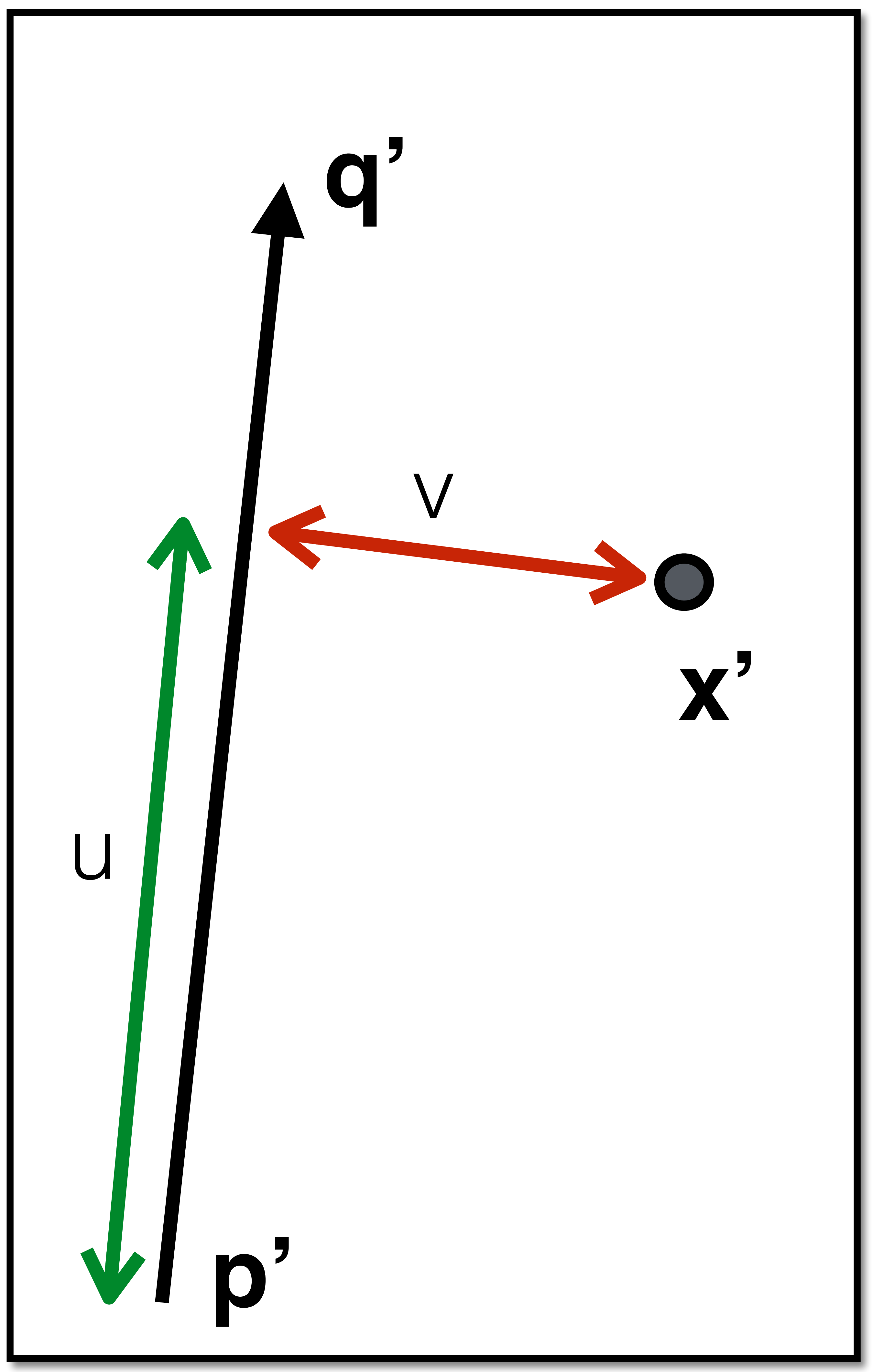
(f) $\alpha = 0.6$

(g) $\alpha = 0.8$

(h) $\alpha = 1.0$

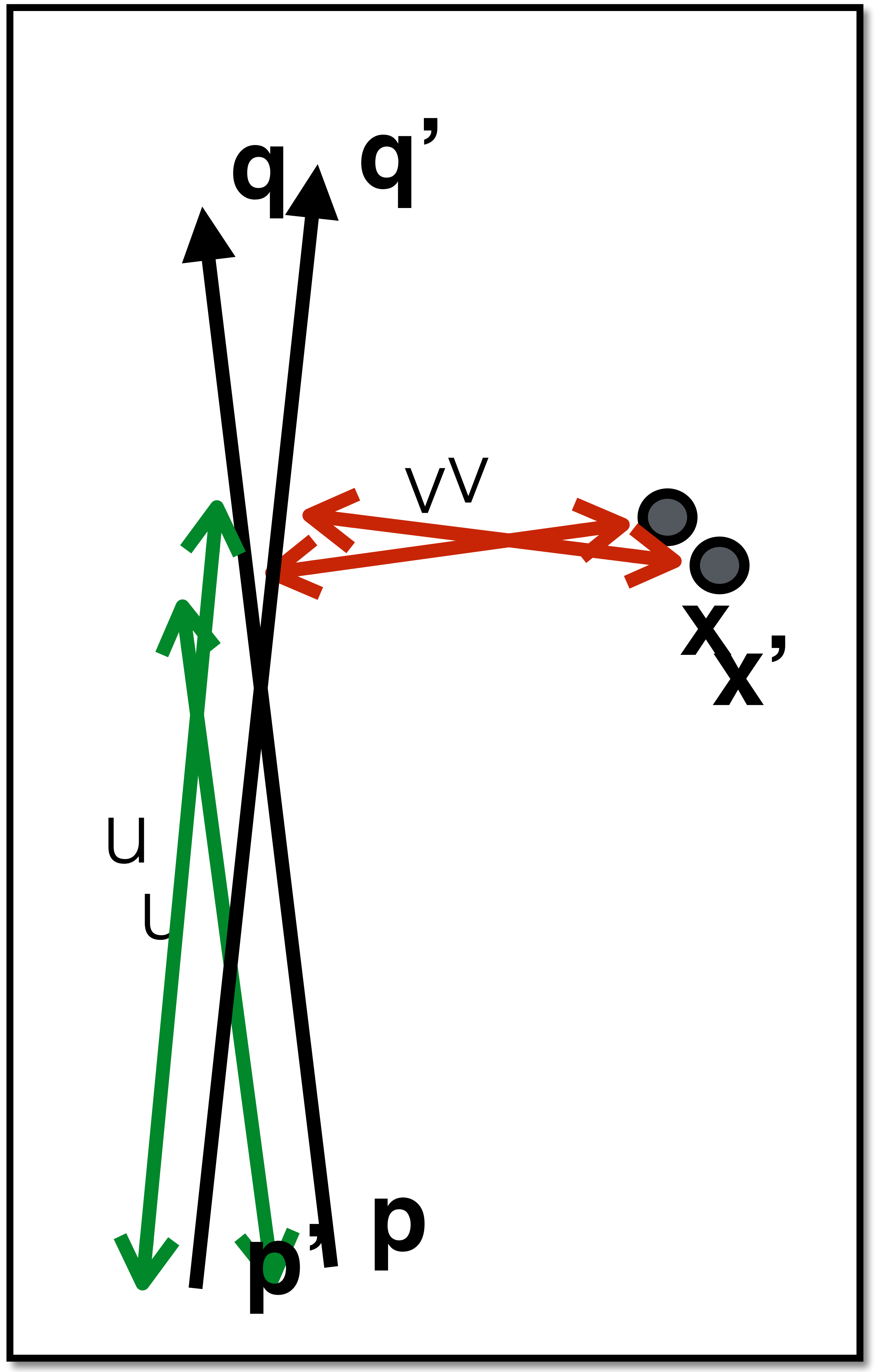


destination image



source image

u is a fraction
v is a length (in pixels)



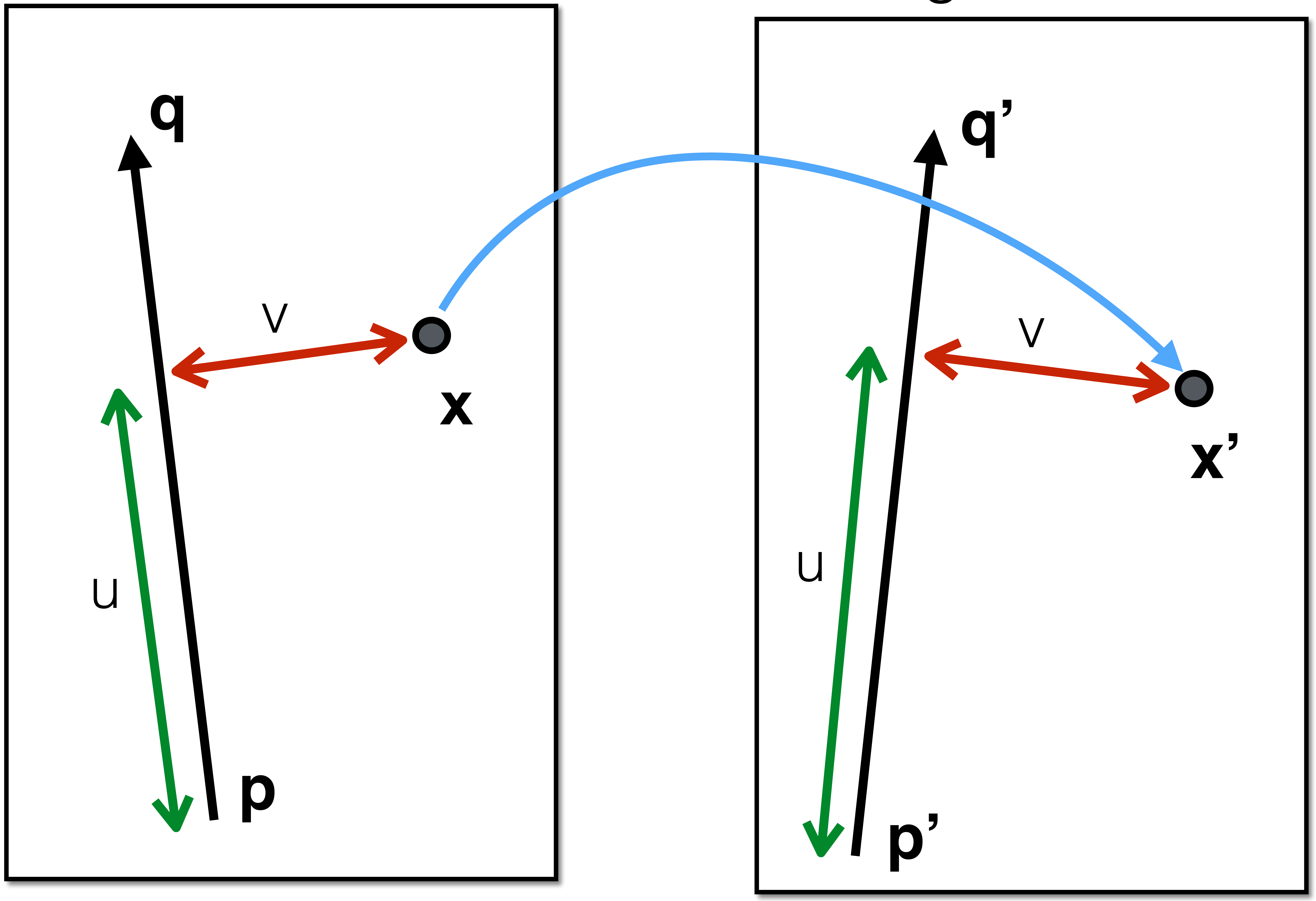
source image and

destination image superimposed

u is a fraction

v is a length (in pixels)

Given x in the destination image, where is x' in the source image?

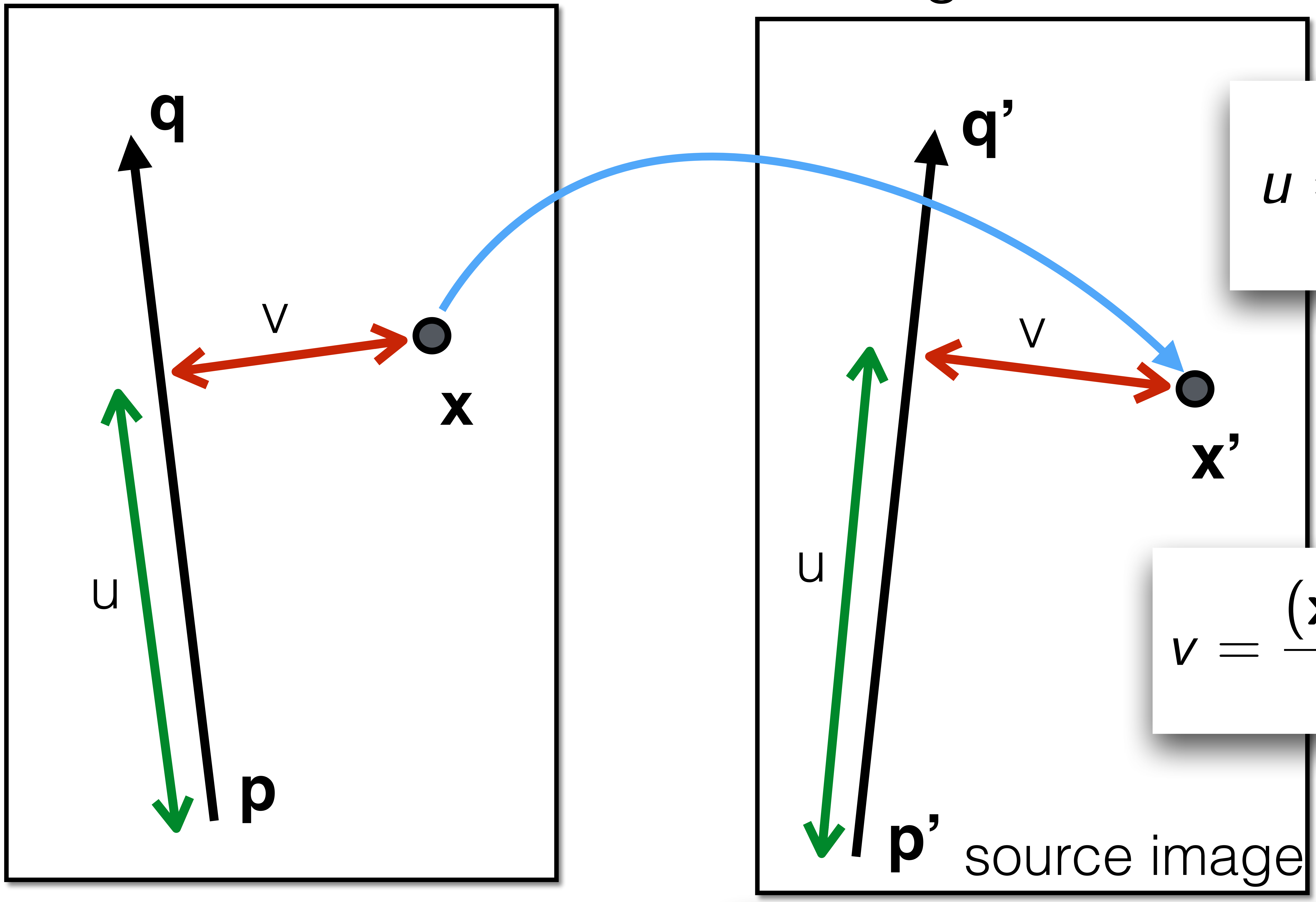


destination image

source image

u is a fraction
v is a length (in pixels)

Given x in the destination image, where is x’ in the source image?



destination image

$$u = \frac{(\mathbf{x} - \mathbf{p}) \cdot (\mathbf{q} - \mathbf{p})}{\|\mathbf{q} - \mathbf{p}\|^2}$$

$$v = \frac{(\mathbf{x} - \mathbf{p}) \cdot \mathrm{Perpendicular}\,(\mathbf{q} - \mathbf{p})}{\|\mathbf{q} - \mathbf{p}\|}$$

$$x' = \mathbf{p}' + \mathbf{u} \cdot (\mathbf{q}' - \mathbf{p}') + \frac{\mathbf{v} \cdot \mathrm{Perpendicular}\,(\mathbf{q}' - \mathbf{p}')}{\|\mathbf{q}' - \mathbf{p}'\|}$$

$$u = \frac{(\mathbf{x} - \mathbf{p}) \cdot (\mathbf{q} - \mathbf{p})}{\|\mathbf{q} - \mathbf{p}\|^2}$$

$$v = \frac{(\mathbf{x} - \mathbf{p}) \cdot \mathrm{Perpendicular}\,(\mathbf{q} - \mathbf{p})}{\|\mathbf{q} - \mathbf{p}\|}$$

$$x' = \mathbf{p}' + \mathbf{u} \cdot (\mathbf{q}' - \mathbf{p}') + \frac{\mathbf{v} \cdot \mathrm{Perpendicular}\,(\mathbf{q}' - \mathbf{p}')}{\|\mathbf{q}' - \mathbf{p}'\|}$$

Weighting Effect of Each Line Pair

- To weight the contribution of each line pair, Beier & Neeley use:

$$\boxed{weight[i] = \left(\frac{length[i]^p}{a + dist[i]} \right)^b}$$

Where:

- *length[i]* is the length of L[i]
- *dist[i]* is the distance from X to L[i]
- *a, b, p* are constants that control the warp

slide from: http://www.cs.virginia.edu/~gfx/Courses/2010/IntroGraphics/Lectures/4-Image.pdf

Warping Pseudocode

```
WarpImage(Image, L’[…], L[…])
begin
    foreach destination pixel p do
        psum = (0,0)
        wsum = 0
        foreach line L[i] in destination do
            p’[i] = p transformed by (L[i],L’[i])
            psum = psum + p’[i] * weight[i]
            wsum += weight[i]
        end
        p’ = psum / wsum
        Result(p) = Image(p’)
    end
end
```

slide from: http://www.cs.virginia.edu/~gfx/Courses/2010/IntroGraphics/Lectures/4-Image.pdf

Morphing Pseudocode

```
GenerateAnimation(Image_0, L_0[…], Image_1, L_1[…])
begin
    foreach intermediate frame time t do
        for i = 1 to number of line pairs do
            L[i] = line t-th of the way from L_0 [i] to L_1 [i]
        end
        Warp_0 = WarpImage(Image_0, L_0, L)
        Warp_1 = WarpImage(Image_1, L_1, L)
        foreach pixel p in FinalImage do
            Result(p) = (1-t) Warp_0 + t Warp_1

    end
end
```

slide from: http://www.cs.virginia.edu/~gfx/Courses/2010/IntroGraphics/Lectures/4-Image.pdf

Feature-Based Warping

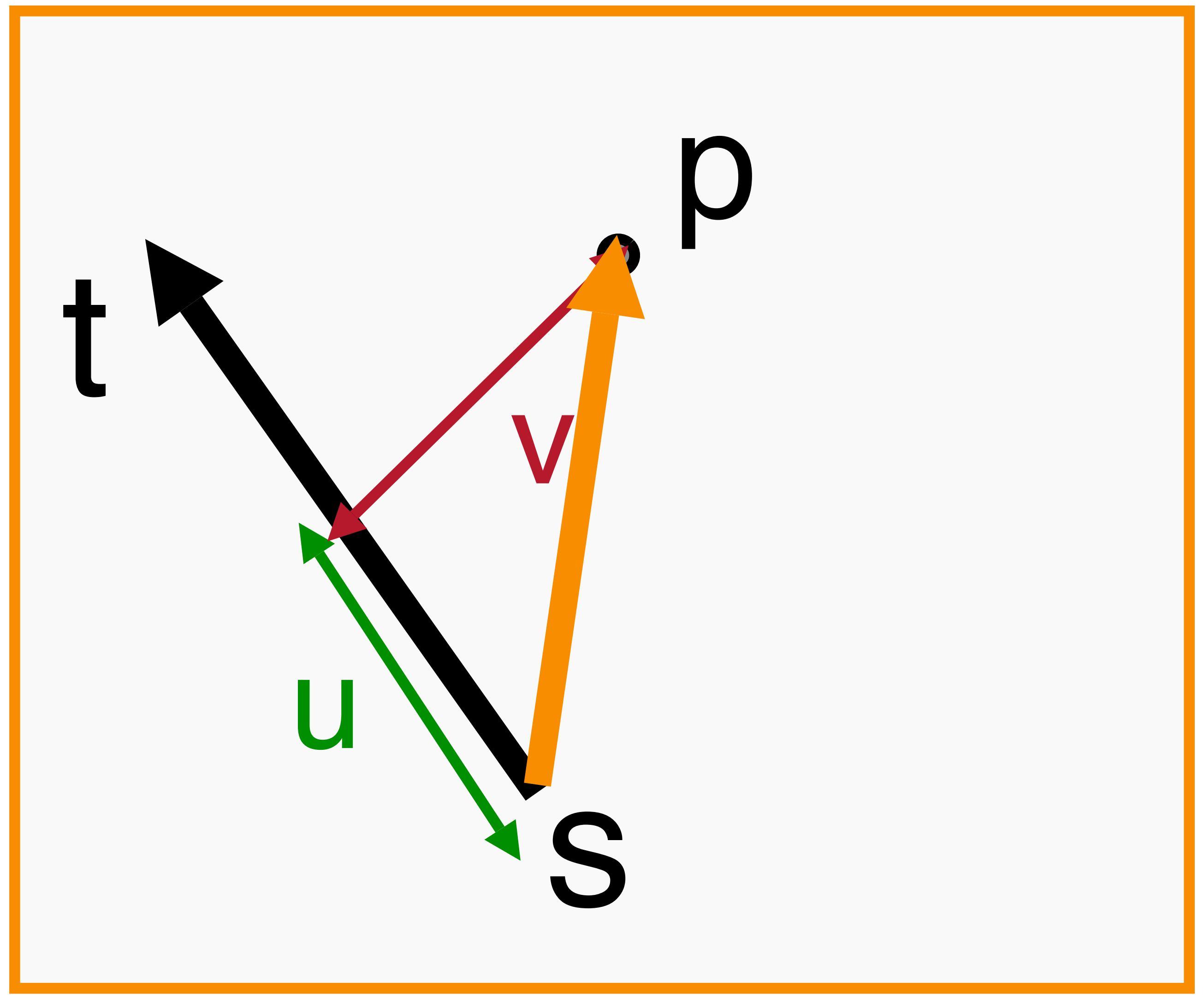
How do I calculate dist? Dist is either…

- abs(v) if u is >= 0 and <= 1

  OR

- distance to the closest endpoint i.e.

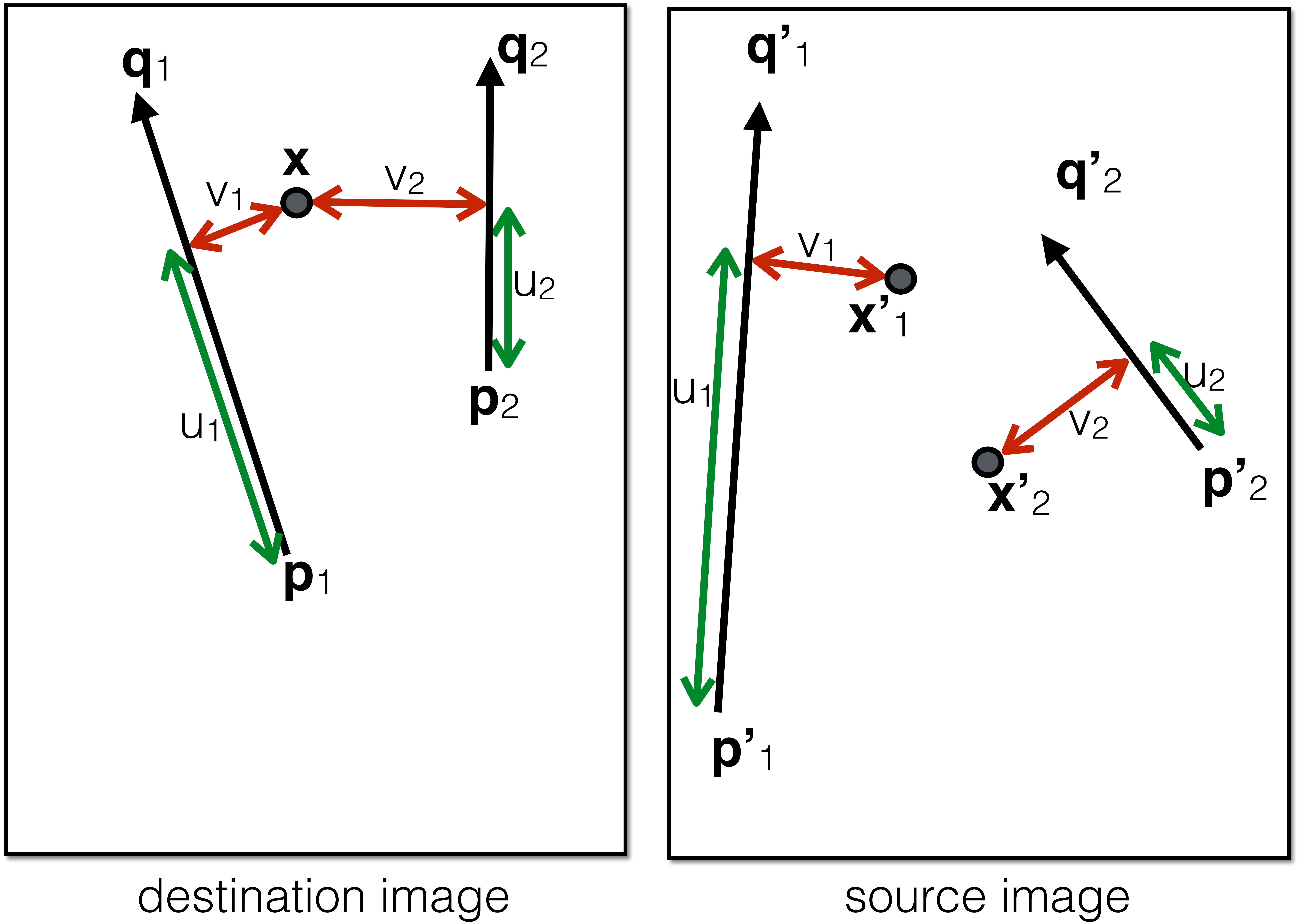
$$Min(\| p - s \|, \| p - t \|)$$



slide from: http://www.cs.virginia.edu/~gfx/Courses/2010/IntroGraphics/Lectures/4-Image.pdf

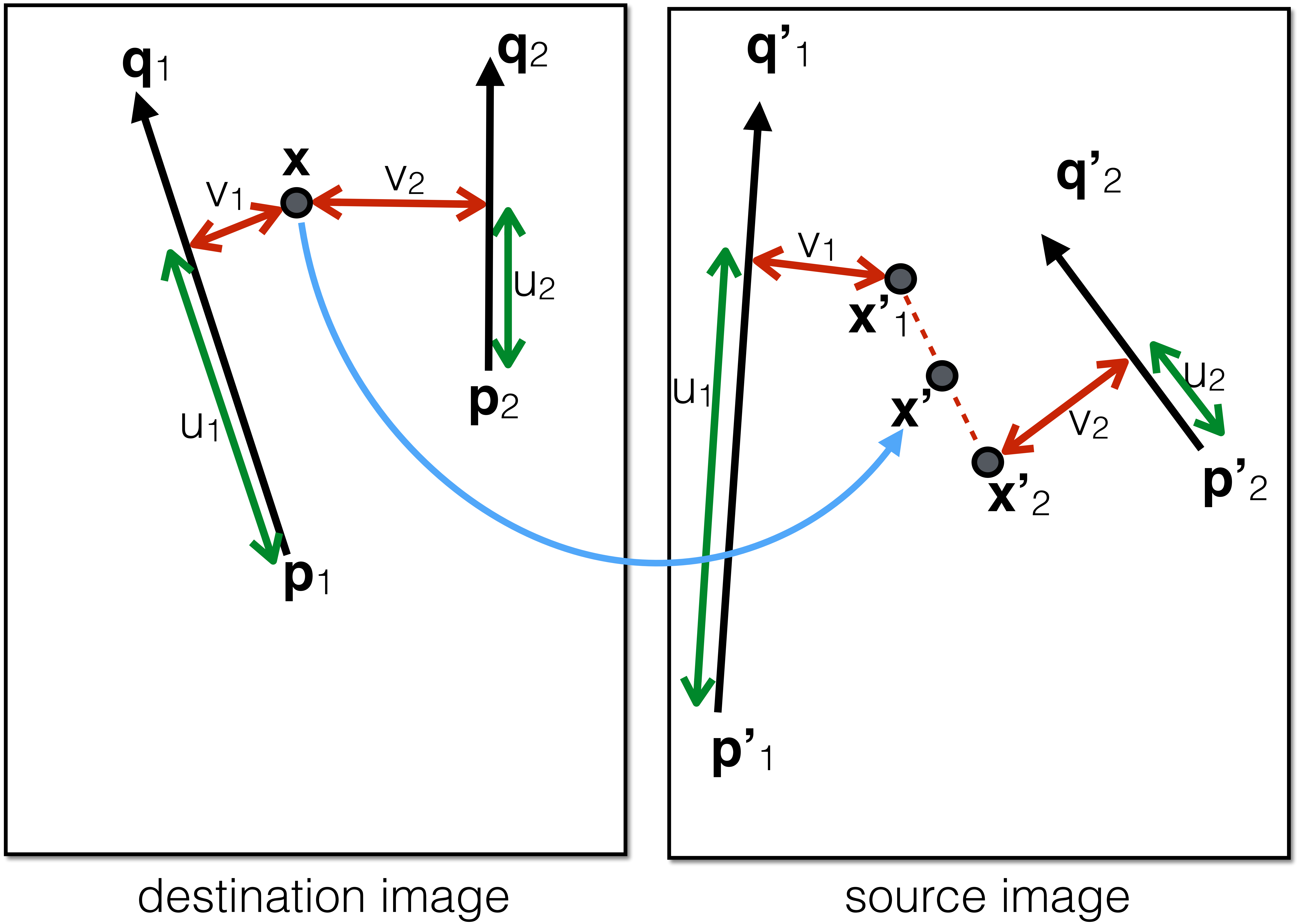
Pairs of lines

- Each line produces one mapping (i.e., warping) for a point **x**.
- As a result, two locations, $\mathbf{x}'_1$ and $\mathbf{x}'_2$, are produced on the source image.



Pairs of lines

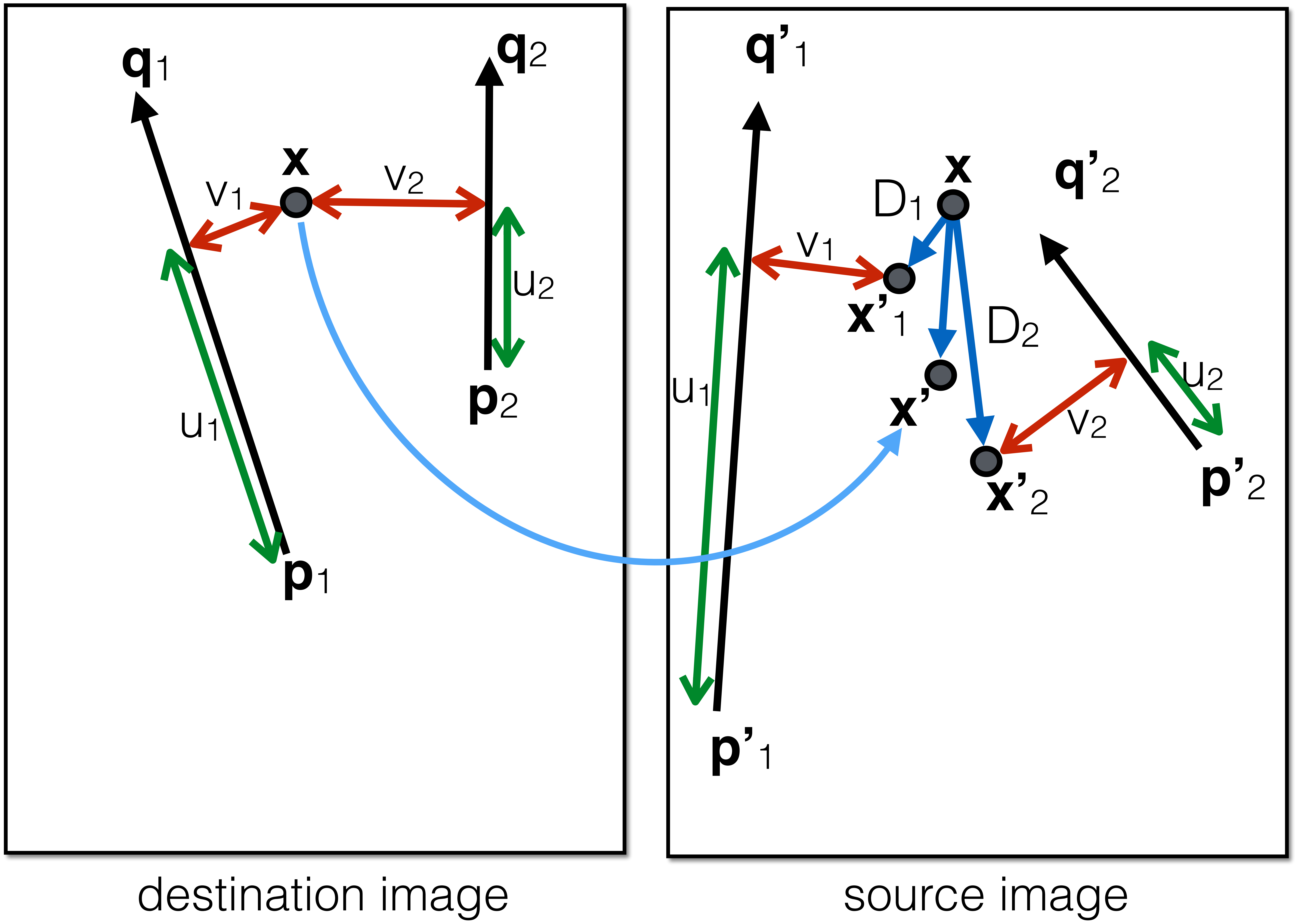
- Select the mapped location, **x**', as an average of **x**'$_1$ and **x**'$_2$.



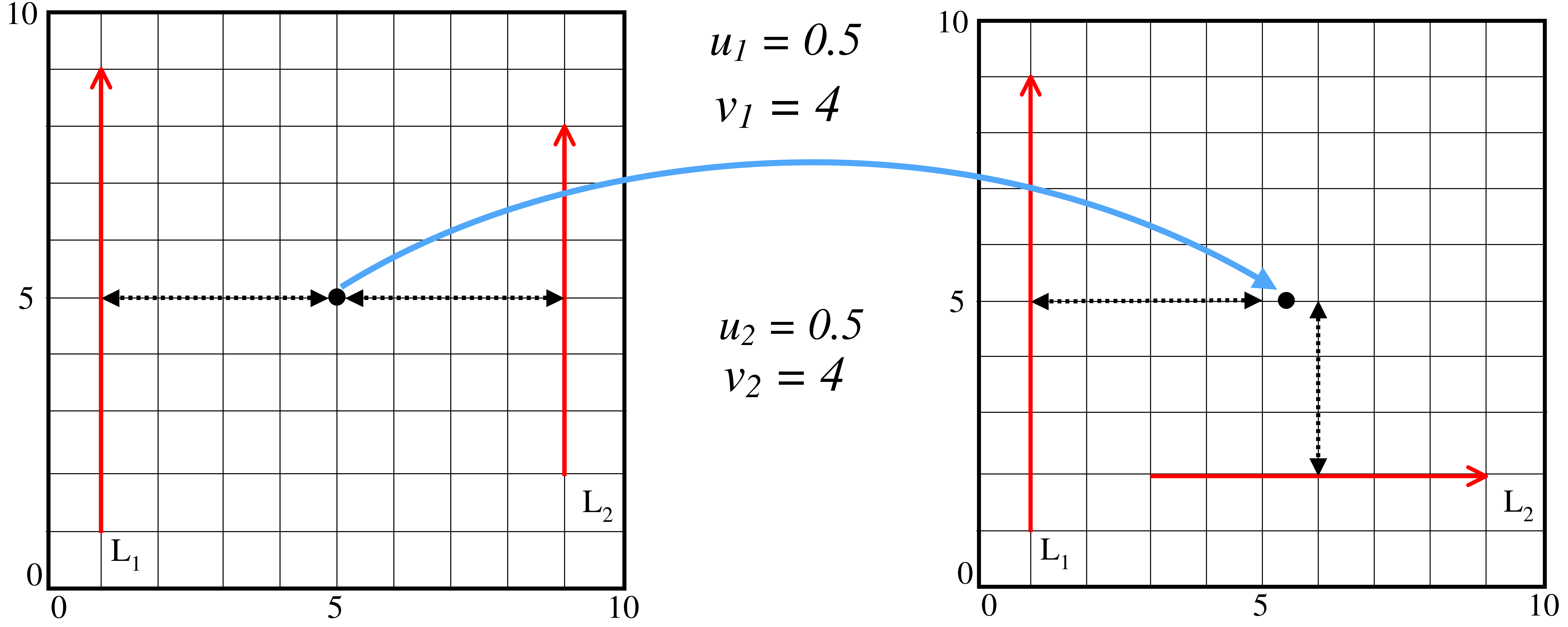
Pairs of lines

- Select the mapped location, **x**', as an average of **x**'$_1$ and **x**'$_2$.

- Use a weighted average, using the distance from the **x** to the lines.



Assume that the constants controlling the warping are $a = b = p = 1$.



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

For L_1' *the weight* $w_1 = 8 / (1 + 4) = 1.6$

for L_2' *the weight* $w_2 = 6 / (1 + 3) = 1.5.$

$$\mathbf{x'} = \frac{\sum_i w_i \mathbf{p}_i}{\sum_i w_i} = \frac{1.6 \cdot \begin{pmatrix} 5 \\ 5 \end{pmatrix} + 1.5 \cdot \begin{pmatrix} 6 \\ 5 \end{pmatrix}}{3.1} = \begin{pmatrix} 5.48 \\ 5 \end{pmatrix}$$

Source: http://www.doc.ic.ac.uk/~dfg/graphics/graphics2010/GraphicsTutorialSolution09.pdf