Resizing Images

- Recall images are a 2D array of RGB values
- I[y][x] gives access to pixel (x,y)
- I[y][x][c] gives access to colour c of pixel (x,y)
- See code in assignment framework
- To resize image Ia from size (Xa,Ya) to image Ib with size (Xb,Yb) we need to find a value for every colour component of every pixel in image Ib
- In other words, we need to find Ib[y][x][c] for all x,y,c where x in [0,Xb-1], y in [0,Yb-1], c in [0,1,2]=[R,G,B] (or [B, G, R] depending on which way around the bytes are stored

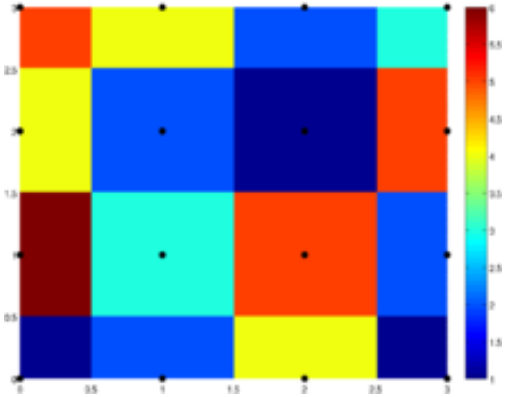
Resizing Images: Nearest Neighbour

- Use a loop:

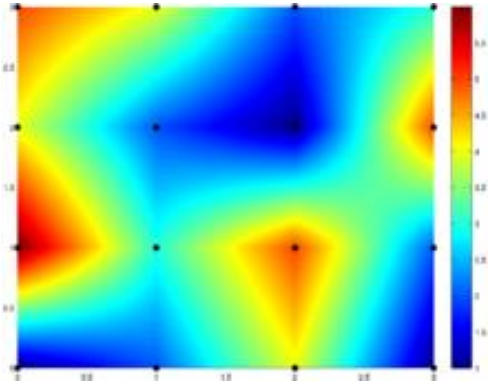
```
for j=0 to Yb-1
   for i=0 to Xb-1
      for c=0 to 2
         y=j*Ya/Yb <- make sure this is done using floats
         x=i*Xa/Xb <- same
         Ib[j][i][c]=Ia[y][x][c]
```

- Advantages:
  - Easy to code
  - Fast to compute (only look up 1 old pixel for each new pixel)
- Disadvantage:
  - Poor quality because each pixel is effectively made bigger

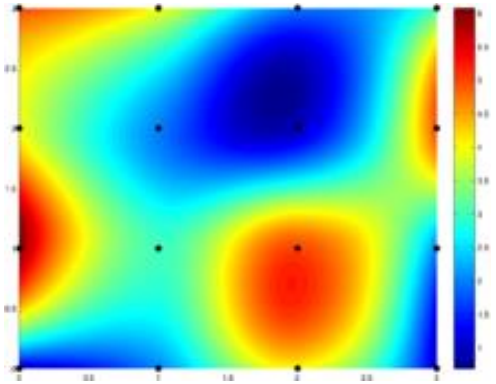
Comparing Interpolation Algorithms [source wikipedia]



- Dots have single colour which is used for nearest neighbouring pixels



- Bilinear interpolation used
- This function is smooth between points, but the derivative is not smooth at boundaries

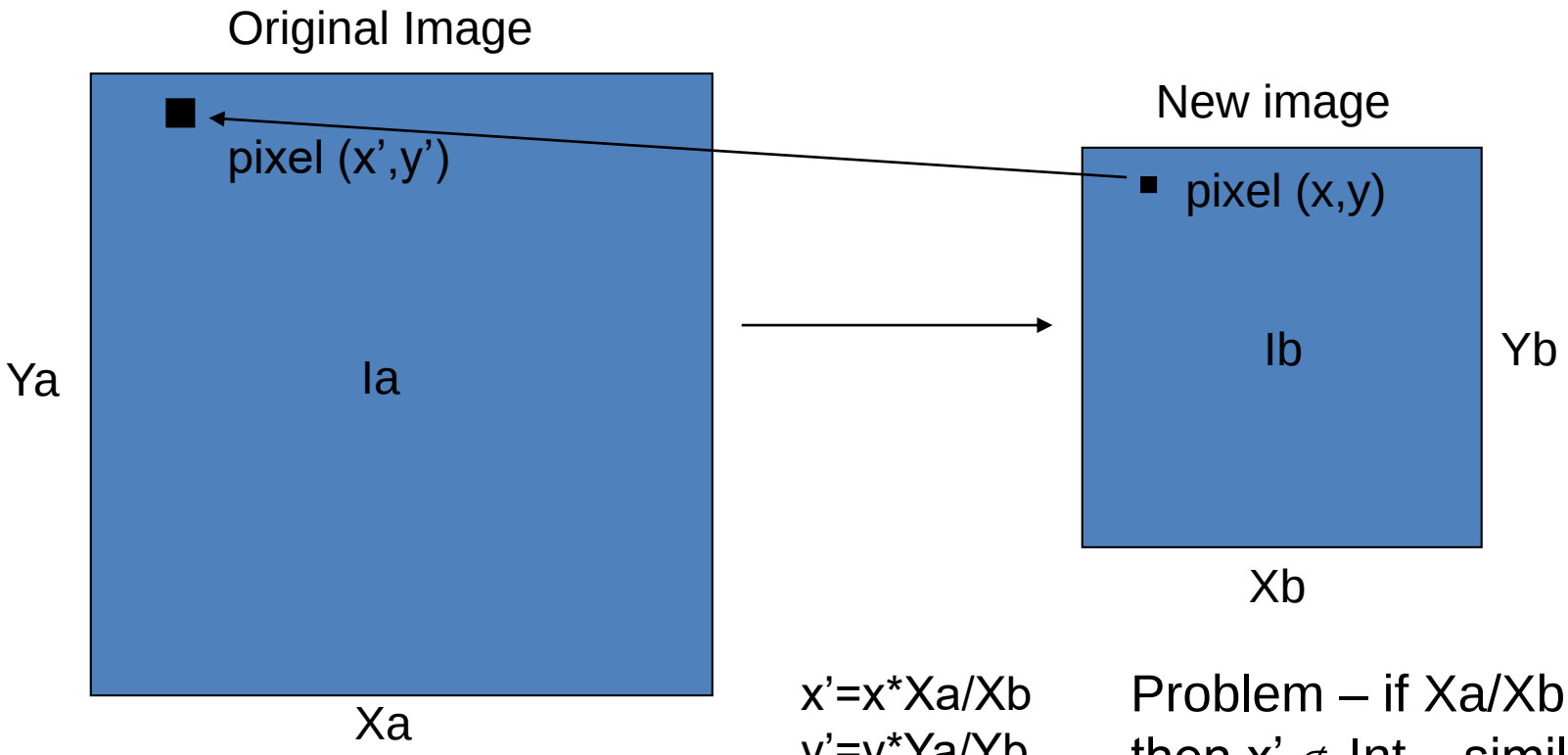


- Bicubic interpolation used
- This function is smooth between points and the derivative is smooth at the boundaries

All these techniques are demonstrated in Photoshop

Resizing Images: Bilinear Interpolation

- Scaling images

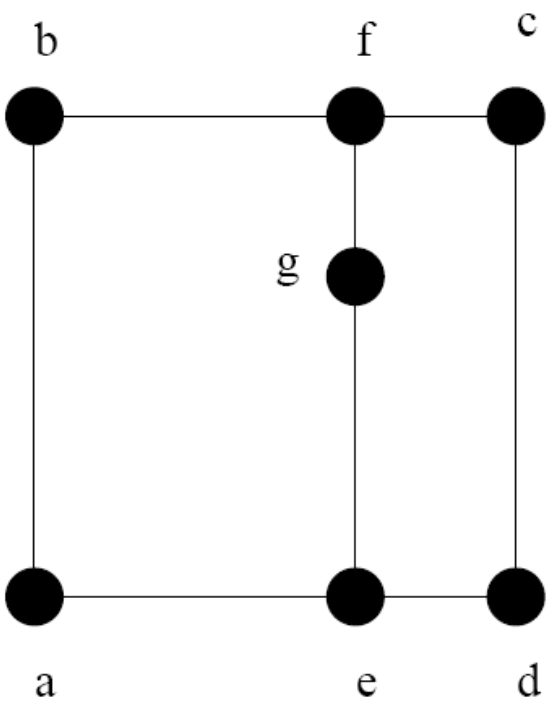


x'=x*Xa/Xb
y'=y*Ya/Yb

Problem – if Xa/Xb $\notin$ Int,
then x' $\notin$ Int – similar for y'

Finding Pixel Colour

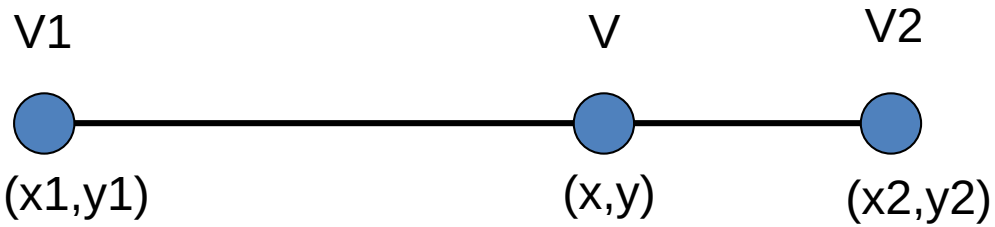
- How do we find a colour for a pixel (x',y') where x',y' $\in$ Real?



- e.g. we know the colours at integer pixels a, b, c and d.
- We want to find out the colour at non-integer pixel g.
- Solution – we find out colours at e and f using LINEAR INTERPOLATION, and then g the same

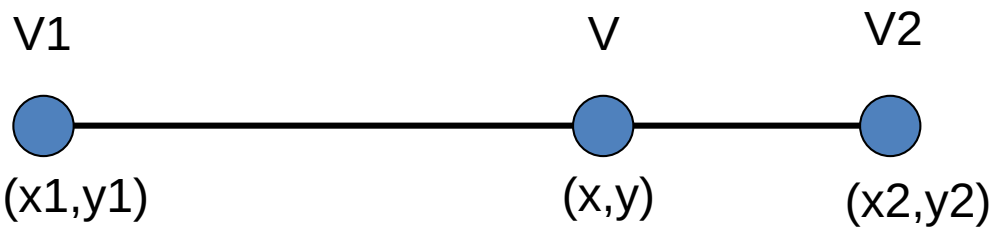
Linear Interpolation Equations

- To find value (e.g. colour) given position (on the y-axis)
- $v = v_1 + (v_2 - v_1)\frac{(y-y_1)}{(y_2-y_1)}$
- To find value (e.g. colour) given position (on the x-axis)
- $v = v_1 + (v_2 - v_1)\frac{(x-x_1)}{(x_2-x_1)}$

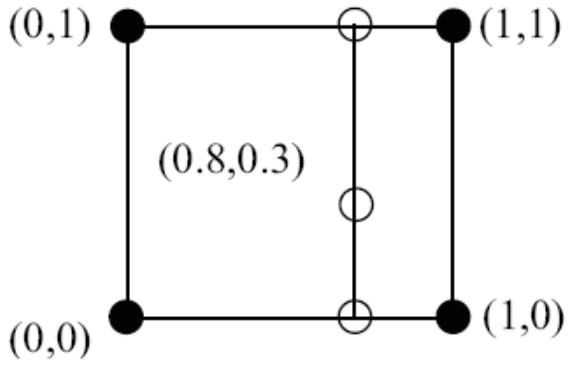


Linear Interpolation Equations

- To find position given colour
- $x = x_1 + (x_2 - x_1)\frac{(v - v_1)}{(v_2 - v_1)}$
- $y = y_1 + (y_2 - y_1)\frac{(v - v_1)}{(v_2 - v_1)}$



Bilinear Interpolation
Exam Question May 2007



Point	Intensity
(0, 0)	0
(1,0)	120
(0,1)	80
(1,1)	140

Describe linear and bilinear interpolation (giving equations where appropriate) and demonstrate their use to calculate the intensity at positions (0.8, 0) and (0.8, 0.3) in the above square.

equations for linear interpolation and, bilinear interpolation and application
[3 marks]. (0.8,0)=96, (0.8,1)=128, (0.8,0.3)=105.6 [2 marks]

Further reading

- Wikipedia, Linear interpolation https://en.wikipedia.org/wiki/Linear_interpolation (relate the equation and diagrams to these notes)
- Wikipedia, Bilinear interpolation https://en.wikipedia.org/wiki/Bilinear_interpolation (particularly the section on Application in image processing)