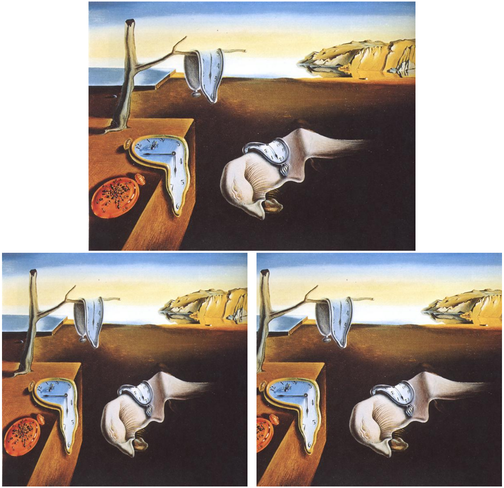
Content Aware Image Resizing

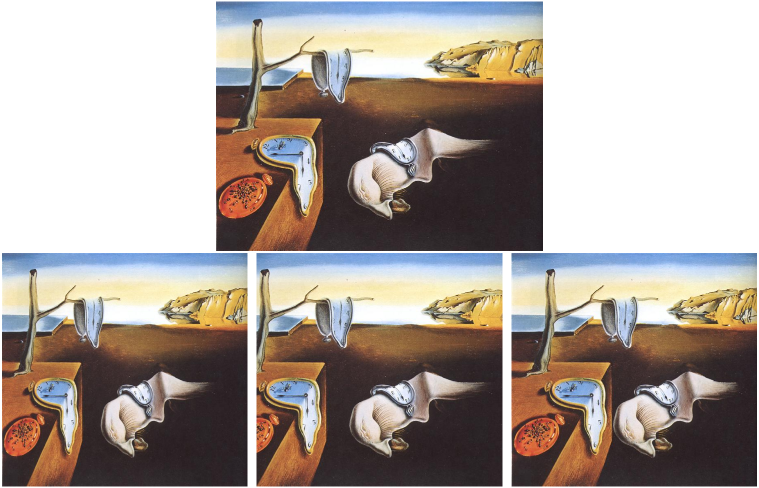
Nicholas Miehlbradt

June 8, 2022

How do we resize an image?



Content aware resizing

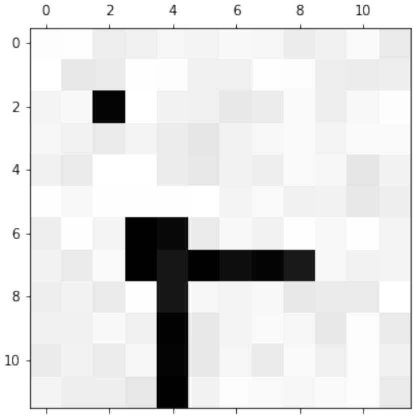


What are the steps of the problem?

1. Find the features in the image
2. Find a line that goes around these features to cut out
3. Cut out the line
4. Repeat 1-3 until we reach the size we want

Detecting Features

How do we indicate what parts of an image are important?



Edge Detection

We can do this by looking for edges in an image. Things in an image are outlined so we can find a line that passes through as few outlines as possible.

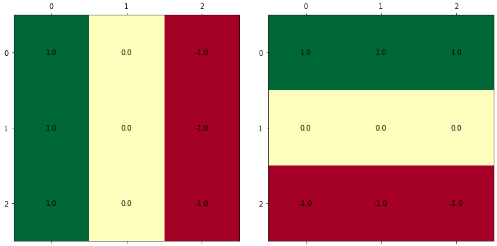
- What makes an edge?
- Can we express this as an algorithm?
- Can we do this quickly?
- Consider a simpler case, just take one row (or column) of the image.

Edge Detection

We want parts of the image that are similar to have a low value and parts where there is a big change to have a high value.
Thinking about images as functions

Edge Detection

We can express whether or not there is a edge as the weighted sum of the surrounding pixels. The weights are called a kernel.

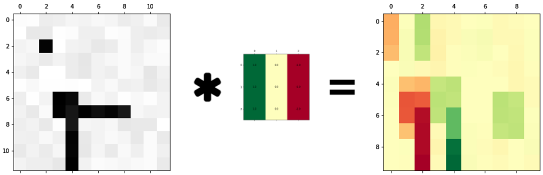


	0	1	2
0	1.0	0.0	-1.0
1	1.0	0.0	-1.0
2	1.0	0.0	-1.0

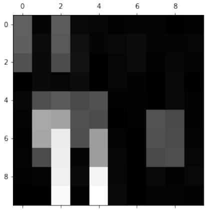
	0	1	2
0	1.0	1.0	1.0
1	0.0	0.0	0.0
2	-1.0	-1.0	-1.0

In practice we rotate the kernel by 180°.

Convolution



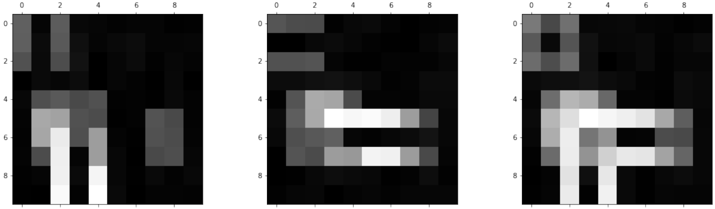
We don’t actually care about the sign of the result, just the magnitude.



Combining Edges

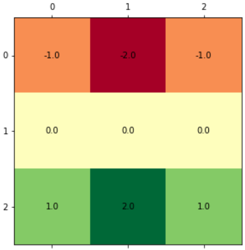
Once we have the horizontal and vertical edges we can combine them.

$$E = \sqrt{E_x^2 + E_y^2}$$

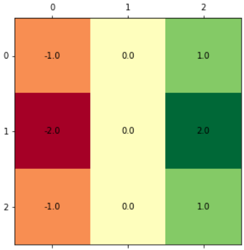


Sobel Filters

A common edge detecting kernel is is called the Sobel operator. It puts more emphasis on the pixels directly above and below (or to each side).



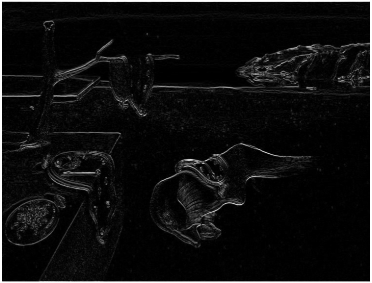
	0	1	2
0	-1.0	-2.0	-1.0
1	0.0	0.0	0.0
2	1.0	2.0	1.0

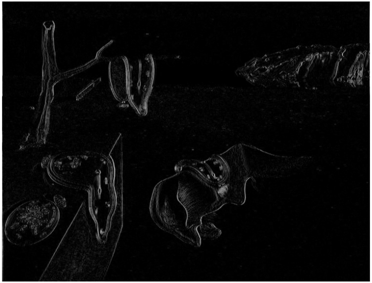


	0	1	2
0	-1.0	0.0	1.0
1	-2.0	0.0	2.0
2	-1.0	0.0	1.0

Sobel Filters

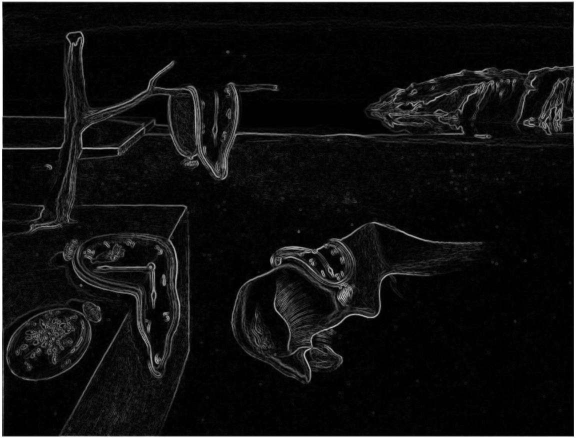
Convolving with the Sobel kernels.





Edges

Combining the horizontal and vertical edges of our image.



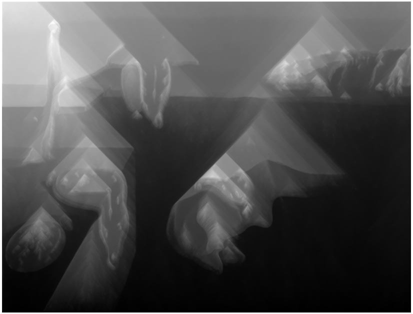
How do we find the minimum 'energy'?

1. Start at the bottom
2. For each row set each pixel to be the sum of the energy of that pixel plus lowest energy pixel below it
3. Repeat until we reach the top of the image

In practice we also want to keep track of the direction of the next pixel too.

Energy Map

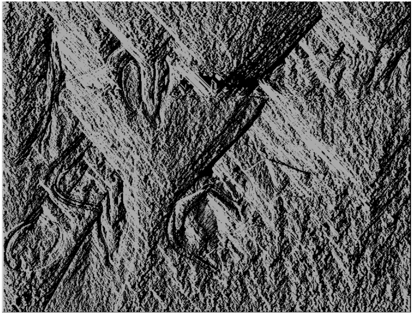
We can visualise the results. Areas that we don't want to cut through are brighter. These are around and above our objects.



Why the triangles?

Direction Map

We can also visualise the directions. Black means go left, white means go right, grey means go straight.



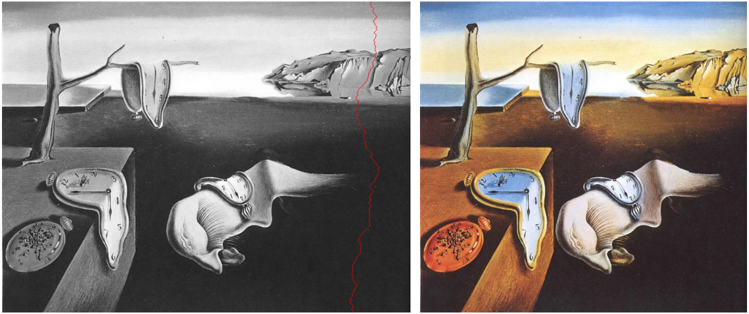
Cutting out the seam

First find the seam to cut out. Which pixels does it pass through?

1. Find the pixel in the top row of the energy map with the lowest value.
2. Work your way down using the direction map to tell you which way to go.

Then remove those pixels from the image.

Results



Links

- Github repo
- Original paper
- More examples
- Online code walk through
- Video explanation