



Computer Vision

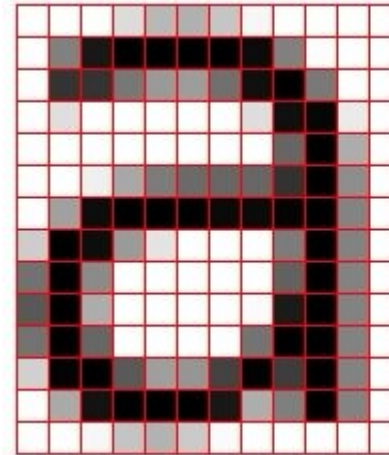
Spring 2026

Overview

- Filters & Convolutions
- Panoramas
- Stereo
- Object Tracking

Images

- 2D matrix/grid of values
 - 1 = light, 0 = dark
 - Alt: 0 to 256
- Color images with 3 channels (red, green, blue)



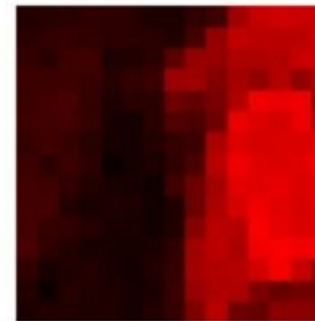
=

1.0	1.0	1.0	0.9	0.6	0.6	0.6	1.0	1.0	1.0	1.0				
1.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.0	1.0				
1.0	0.2	0.2	0.5	0.6	0.6	0.5	0.0	0.0	0.5	1.0				
1.0	0.9	1.0	1.0	1.0	1.0	1.0	1.0	0.9	0.0	0.0	0.9			
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.0	0.5				
1.0	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.4	0.0	0.5				
1.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5				
0.9	0.0	0.0	0.6	1.0	1.0	1.0	1.0	0.5	0.0	0.5				
0.5	0.0	0.6	1.0	1.0	1.0	1.0	1.0	0.5	0.0	0.5				
0.5	0.0	0.7	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.5				
0.6	0.0	0.6	1.0	1.0	1.0	1.0	0.5	0.0	0.0	0.5				
0.9	0.1	0.0	0.6	0.7	0.7	0.5	0.0	0.5	0.0	0.5				
1.0	0.7	0.1	0.0	0.0	0.0	0.1	0.9	0.8	0.0	0.5				
1.0	1.0	1.0	0.8	0.8	0.9	1.0	1.0	1.0	1.0	1.0				

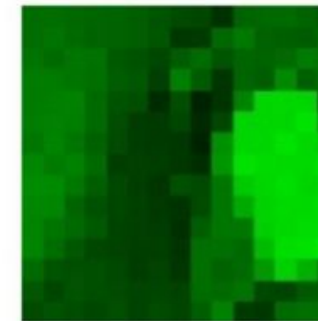


color image patch

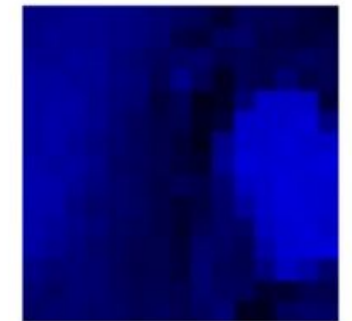
red



green



blue



Point Operation

- do some operation to each pixel value in the image

original



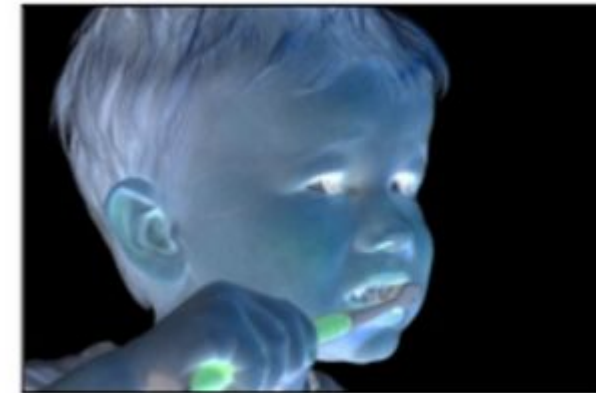
x

darken



$x - 128$

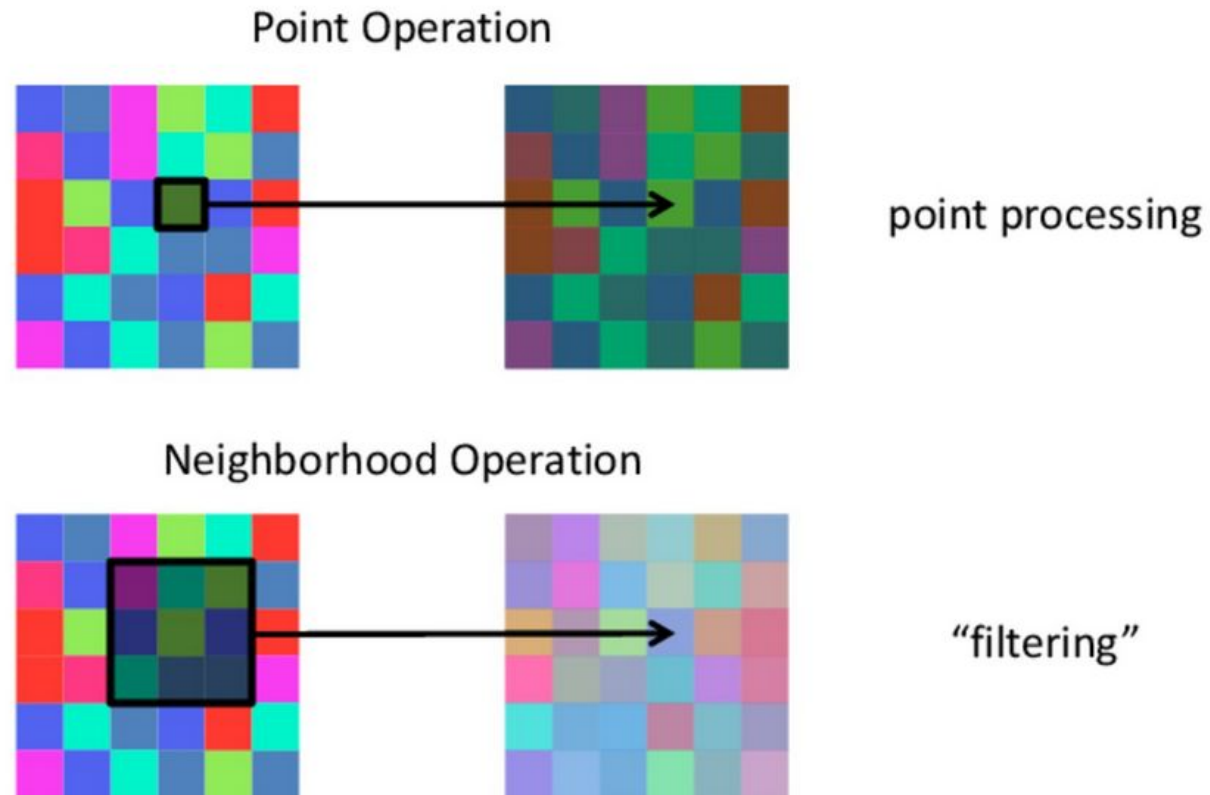
invert



$255 - x$

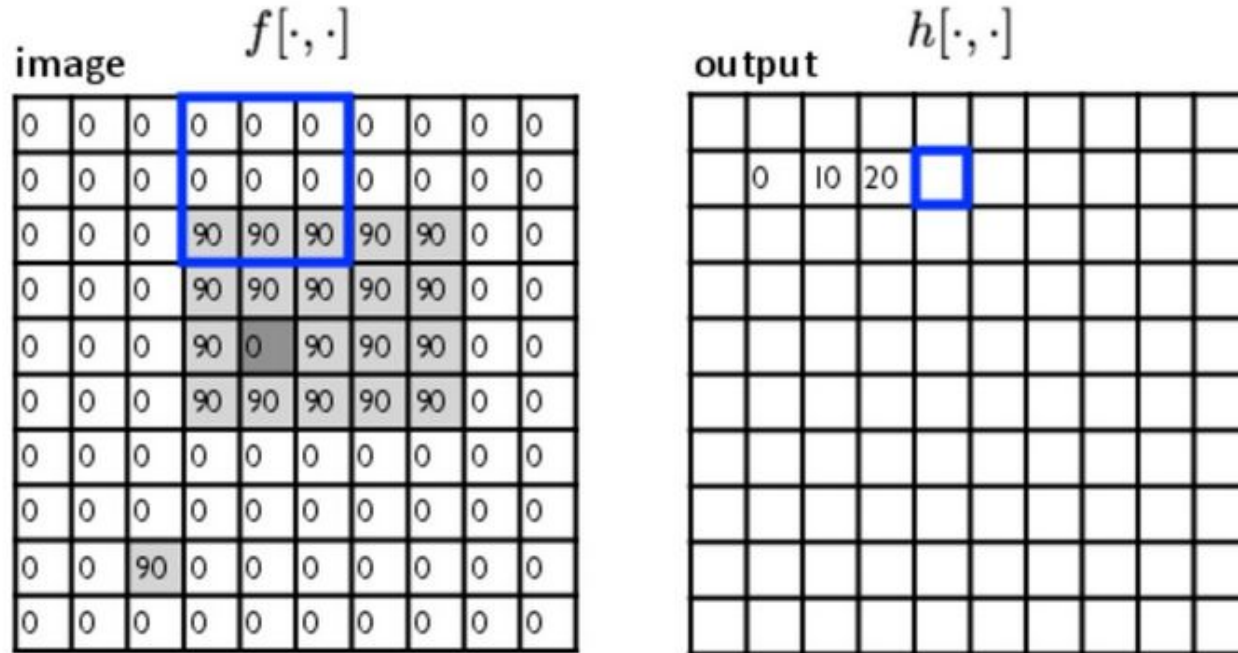
Filtering

- neighborhood operation



Box Filter

$$\frac{1}{9} \begin{matrix} g[\cdot, \cdot] \\ \text{kernel} \\ \begin{array}{|c|c|c|} \hline | & | & | \\ \hline | & | & | \\ \hline | & | & | \\ \hline \end{array} \end{matrix}$$

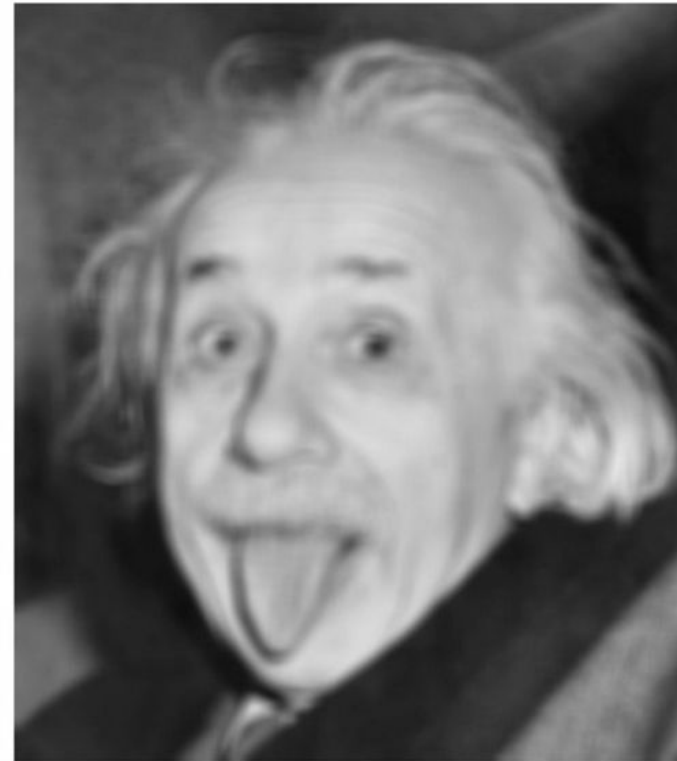
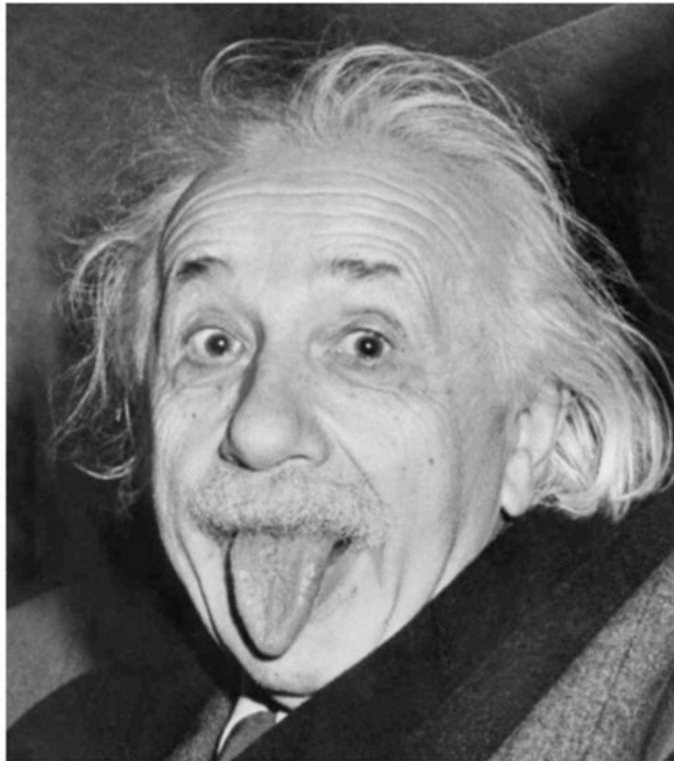


$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

output
 k, l
filter
image (signal)

Candy Question

What does this filter do?



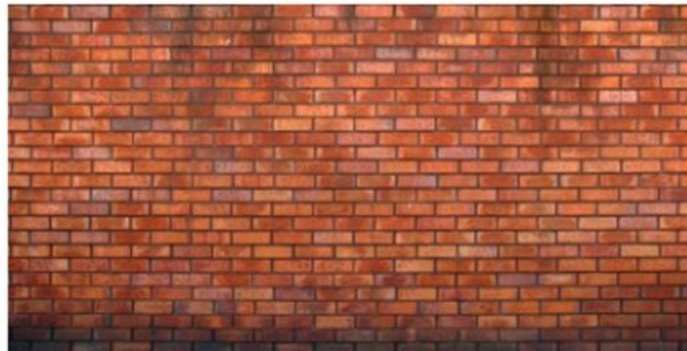
Sobel Filter

Horizontal Sober filter:

1	0	-1
2	0	-2
1	0	-1

Vertical Sobel filter:

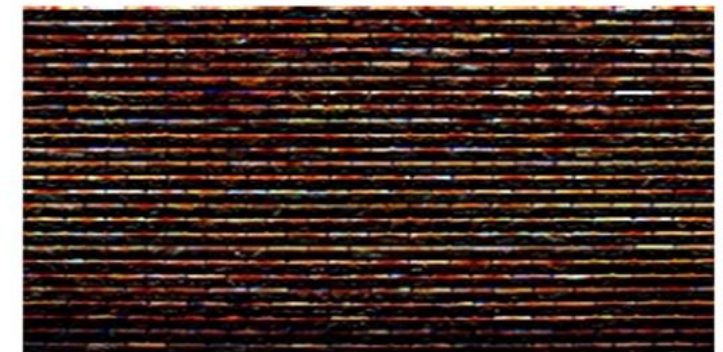
1	2	1
0	0	0
-1	-2	-1



original

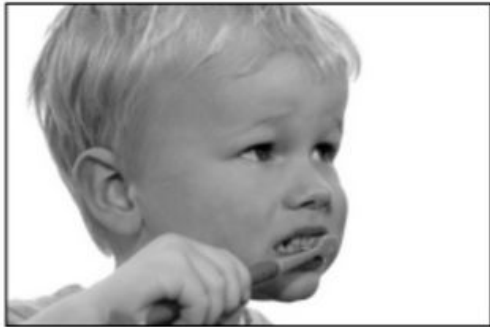


horizontal Sobel filter

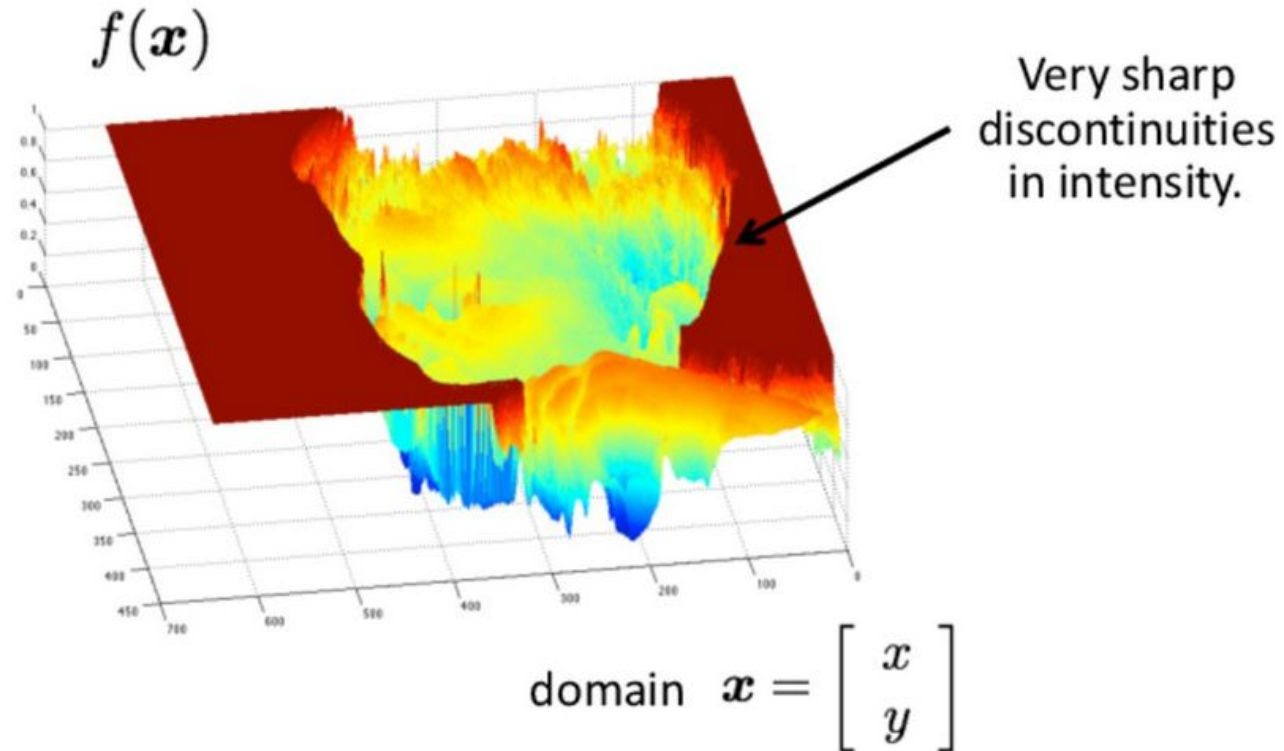


vertical Sobel filter

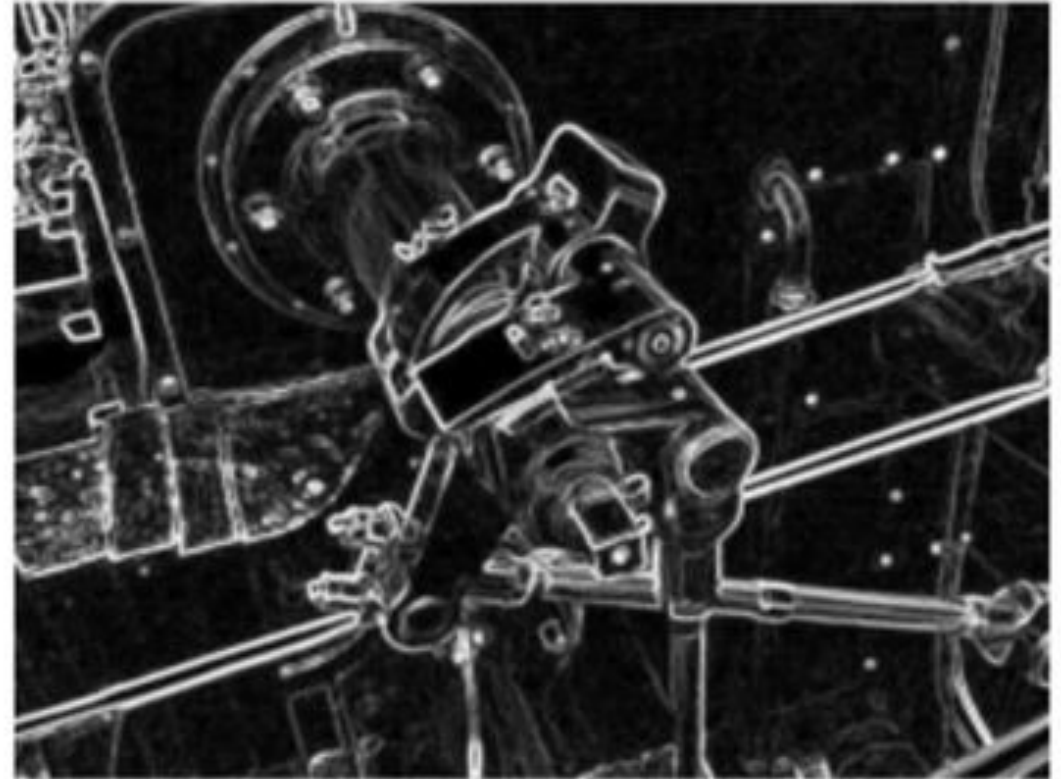
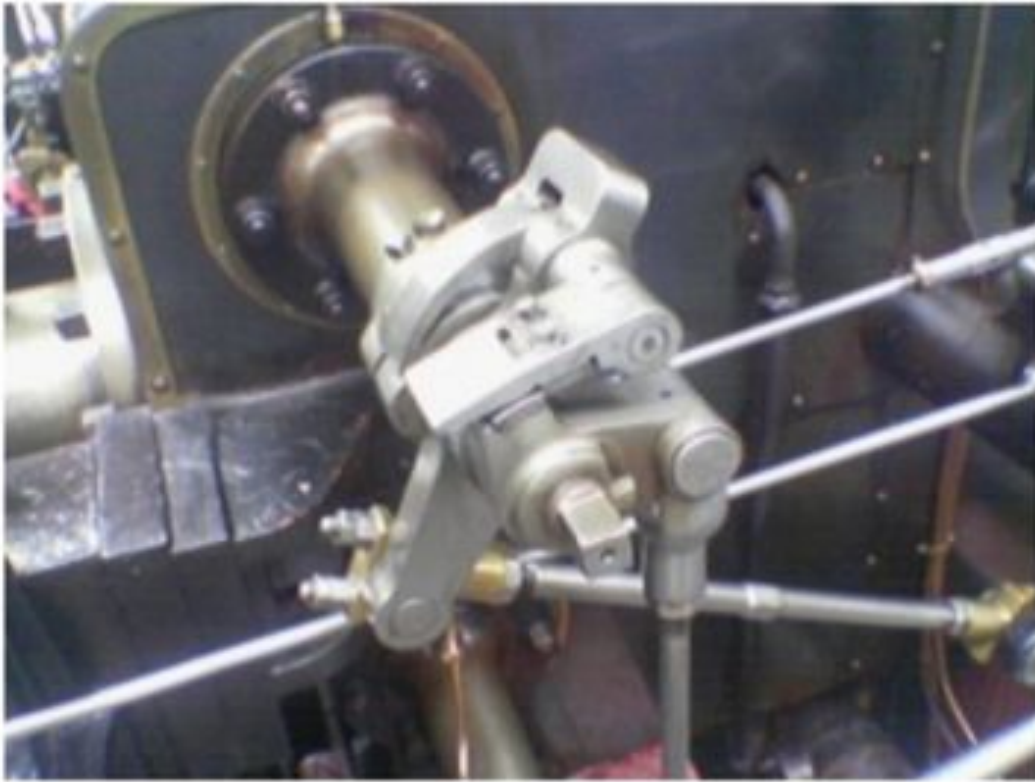
Gradient = Edge



grayscale image

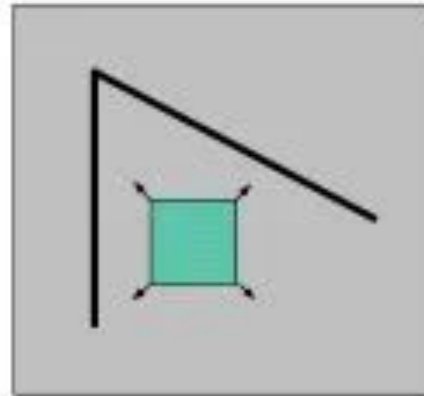


Line Detection

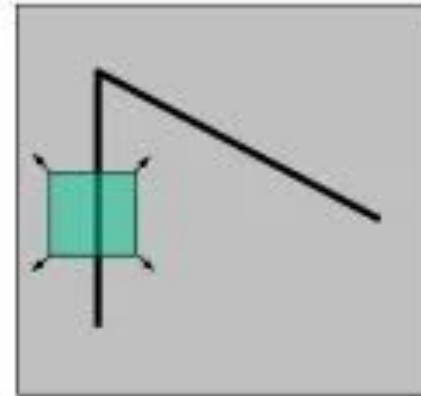


Corner Detection

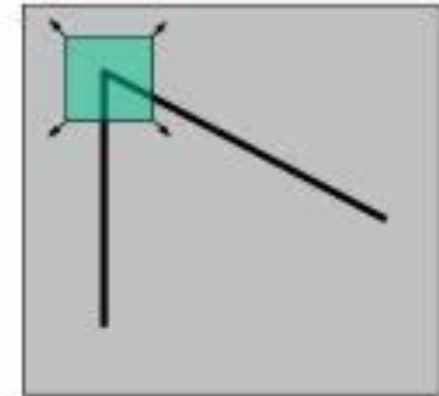
- Sobel Vertical & Horizontal Filter + Linear Algebra = Corners!
- Points of Interest (for CV algorithms)



"flat" region:
no change in all
directions

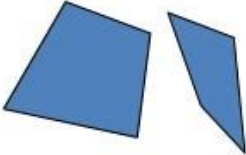
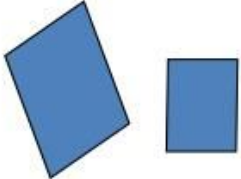
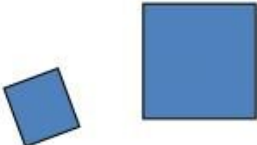
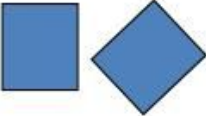


"edge":
no change along the
edge direction



"corner":
significant change in
all directions

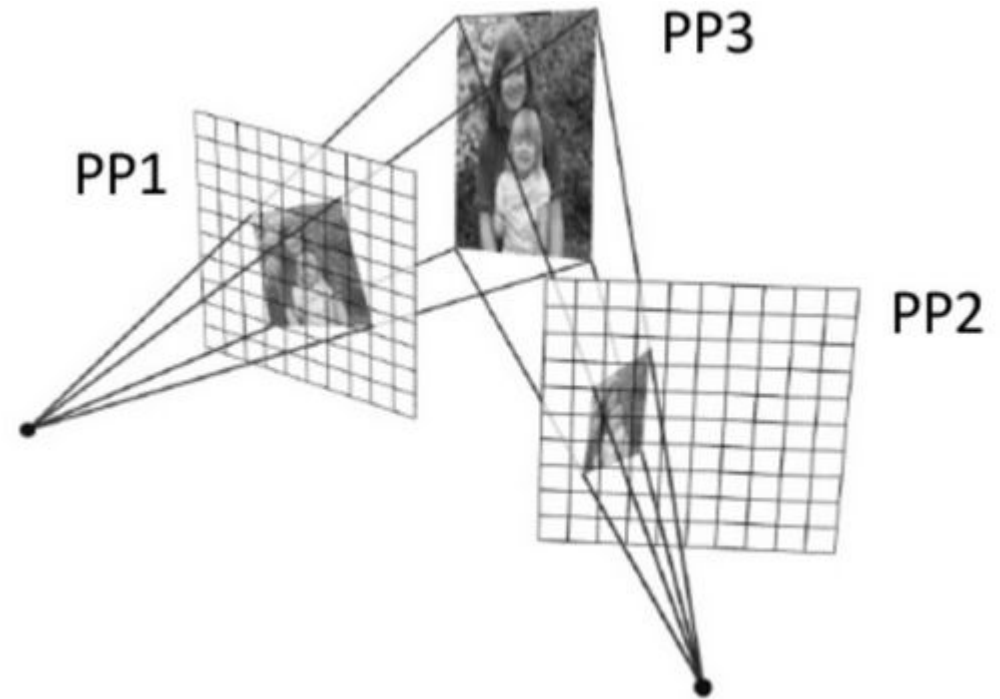
Transformations

Projective 8 dof	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$		Views of a plane from different viewpoints, any view of a scene from the same viewpoint.
Affine 6 dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Images of a “far away” object under any rotation
Similarity 4 dof	$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Camera looking at an assembly line w/ zoom.
Euclidean 3 dof	$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Camera looking at an assembly line.

Computer Vision, Robert Pless

Homography

- Projective
- A 3x3 matrix that maps 2D pts from different projective planes to 3D and vice versa.
- Think of it as a function that maps a 3D point (on the object) into the 2D pt on an image
- What points? Corners!



Application: Panorama

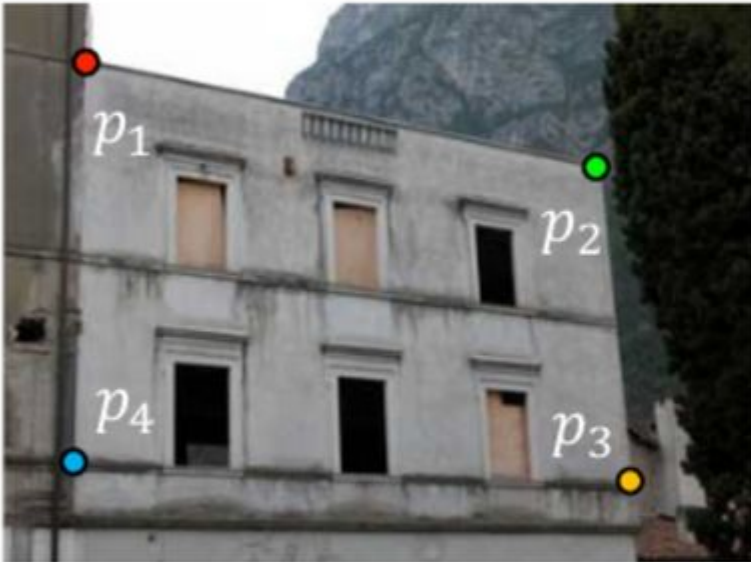
How do we stitch images from different viewpoints?



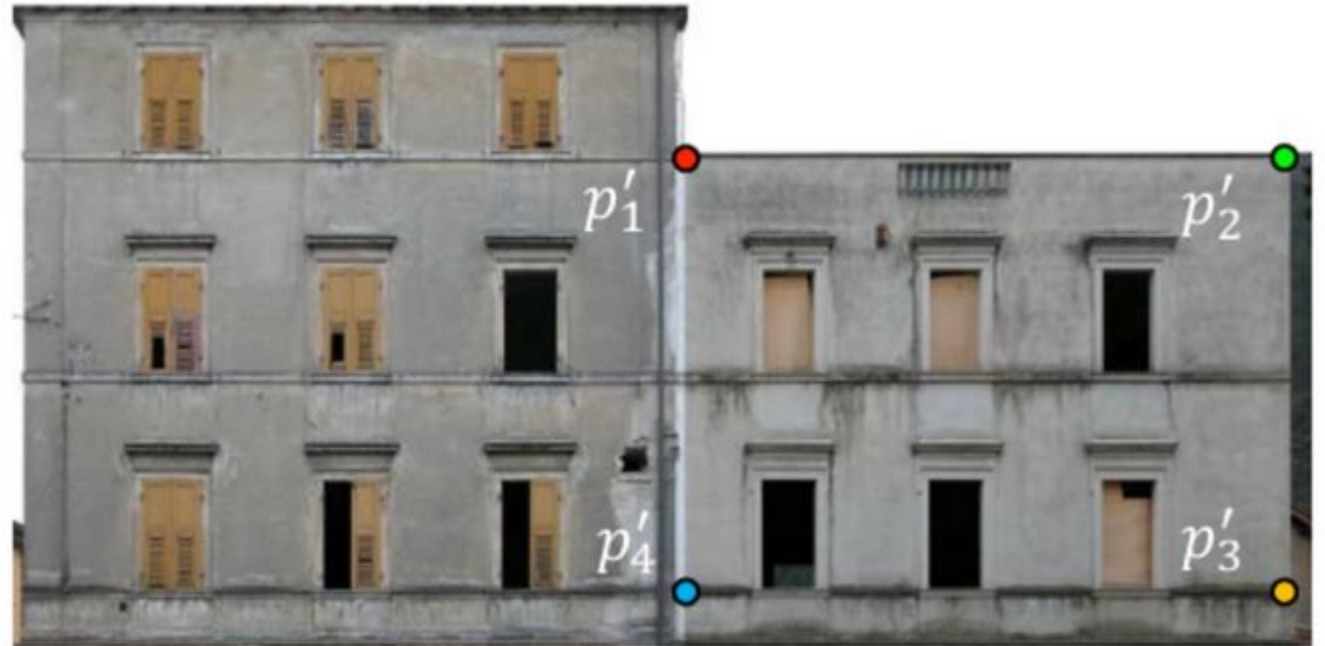
Use image homographies.



Application: Panoramas



original image



target image

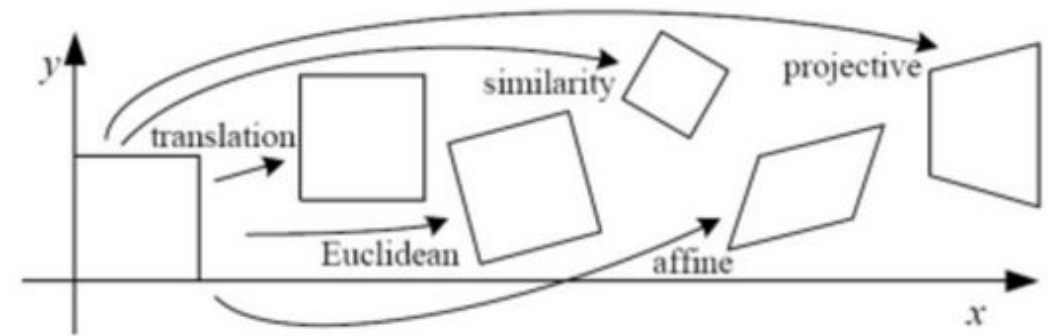
Application: AR



Object Tracking

Optical Flow: Tracking movement of pixels in a template(ex. a car) from frame to frame in a video

Limits to affine transformations =
Less work(less degrees of freedom)



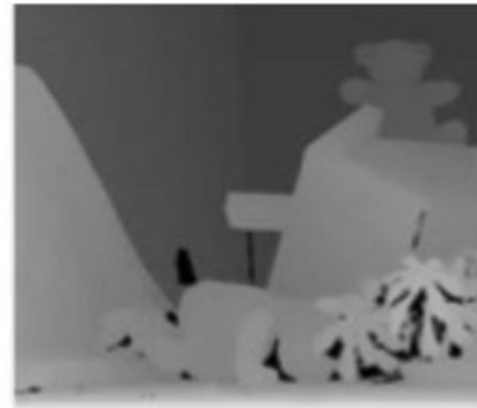
Small Demo

```
Activities Terminal Mar 14 21:22
la-shallot@linux-~/f6385/assign6/python
(56.243680219574316, 134.88169842076618, 142.62531216361685, 169.78226483047706)
M [[ 0.99798253 0.00350079 -0.52994844]
 [ -0.00194436 0.99059651 0.64787838]]
Plotting: [ 56.42099303 134.5323459 142.86099898 169.65027058]
frame***** 21
TEMP
86
34
Start Iter
(56.243680219574316, 134.88169842076618, 142.62531216361685, 169.78226483047706)
/home/la-shallot/16385/assign6/python/LucasKanadeAffine.py:118: FutureWarning: 'rcond' parameter will change to the default of machine precision times 'max(M, N)' where M and N are the input matrix dimensions.
To use the future default and silence this warning we advise to pass 'rcond=None', to keep using the old, explicitly pass 'rcond=-1'.
(delta_p, _, _) = np.linalg.lstsq(A, B)
Start Iter
(57.0079538157988, 135.59930414081106, 144.32378031913774, 171.11933815299668)
Start Iter
(58.04932385512691, 136.0641059050553, 145.2390487243482, 171.41044305711844)
Start Iter
(58.705835328917885, 136.22905927055774, 145.27149058149885, 171.3245750139285)
Start Iter
(58.8280217826682, 136.24107285616145, 145.2454128762916, 171.305368511619)
Start Iter
(58.84077172690909, 136.24212620428992, 145.2383414353066, 171.30379800159554)
M [[ 0.98252171 0.04371679 -2.31646054]
 [ 0.00739521 0.98631241 2.79069967]]
Plotting: [ 56.24368022 134.88169842 142.62531216 169.78226483]
frame***** 22
TEMP
86
35
Start Iter
(58.84077172690909, 136.24212620428992, 145.2383414353066, 171.30379800159554)
/home/la-shallot/16385/assign6/python/LucasKanadeAffine.py:118: FutureWarning: 'rcond' parameter will change to the default of machine precision times 'max(M, N)' where M and N are the input matrix dimensions.
To use the future default and silence this warning we advise to pass 'rcond=None', to keep using the old, explicitly pass 'rcond=-1'.
(delta_p, _, _) = np.linalg.lstsq(A, B)
Start Iter
(59.788996100195966, 137.1051465616653, 146.29223136382114, 172.28663775925142)
Start Iter
(60.08259254948007, 137.19327554358395, 146.35785468697293, 172.22500438079362)
Start Iter
(60.034016872639305, 137.18241577854513, 146.34993274999627, 172.22462679600704)
Start Iter
(60.042870145979286, 137.18336122098816, 146.35049974398242, 172.22554297213287)
M [[ 9.92275808e-01 1.64784186e-02 -5.87321943e-01]
 [ 7.95931340e-04 9.97482819e-01 1.23734792e+00]]
Plotting: [ 58.84077173 136.2421262 145.23834144 171.303798 ]
^CTraceback (most recent call last):
  File "/home/la-shallot/16385/assign6/python/test_lk_affine.py", line 55, in <module>
    plt.pause(0.01)
  File "/home/la-shallot/.local/lib/python3.10/site-packages/matplotlib/pyplot.py", line 545, in pause
    canvas.start_event_loop(interval)
  File "/home/la-shallot/.local/lib/python3.10/site-packages/matplotlib/backends/backend_qt.py", line 409, in start_event_loop
    with _maybe_allow_interrupt(event_loop):
  File "/usr/lib/python3.10/contextlib.py", line 142, in __exit__
    next(self.gen)
  File "/home/la-shallot/.local/lib/python3.10/site-packages/matplotlib/backends/qt_compat.py", line 261, in _maybe_allow_interrupt
    old_sigint_handler(*handler_args)
KeyboardInterrupt

la-shallot@linux-~/f6385/assign6/python
```

Stereo

The amount of horizontal movement is
inversely proportional to ...



... the distance from the camera.

2 Images ->
Depth
Perception!

Stereo: Result



Depth Estimation via Stereo Matching



Candy Question:

What are some applications of depth perception via stereo you can think of?

Answers

Stereoscopes: A 19th Century Pastime



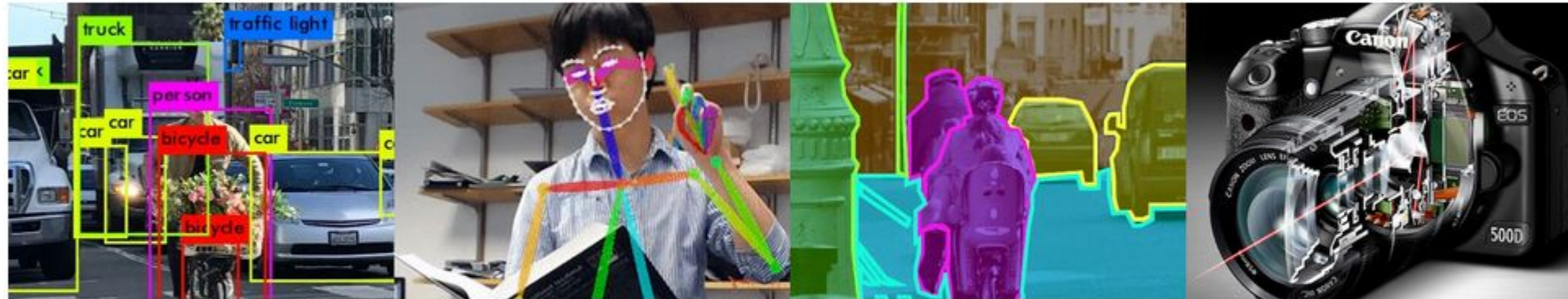
Subaru
Eyesight system

Pre-collision
braking



Want to Learn More?

Computer Vision (CMU 16-385)



This course provides a comprehensive introduction to computer vision. Major topics include image processing, detection and recognition, geometry-based and physics-based vision and video analysis. Students will learn basic concepts of computer vision as well as hands on experience to solve real-life vision problems.

more at this link! <http://vision.cs.cmu.edu/courses.html>