HW1

THE OHIO STATE UNIVERSITY

Preparation

Please make sure you have read Lectures 2 & 3 slides

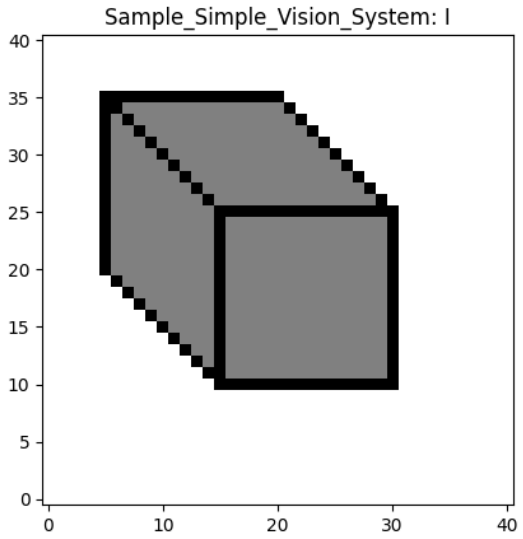
Please make sure you have read the Textbook chapter 2

Please make sure you check page 20 of this PPT.

The notations in this slide deck are a bit different from the lecture slides! The goal is to make implementation easier.

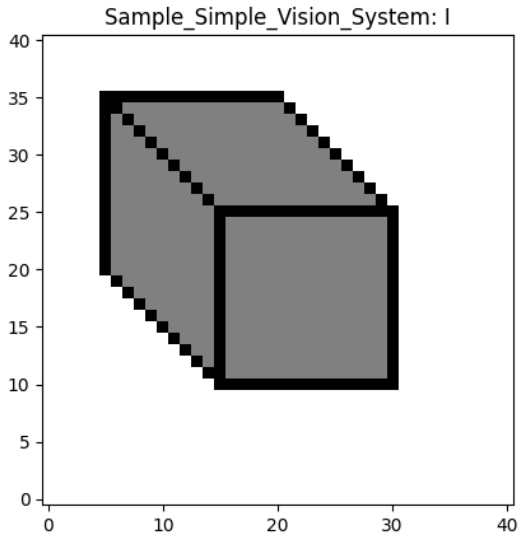
Problem overview

Given

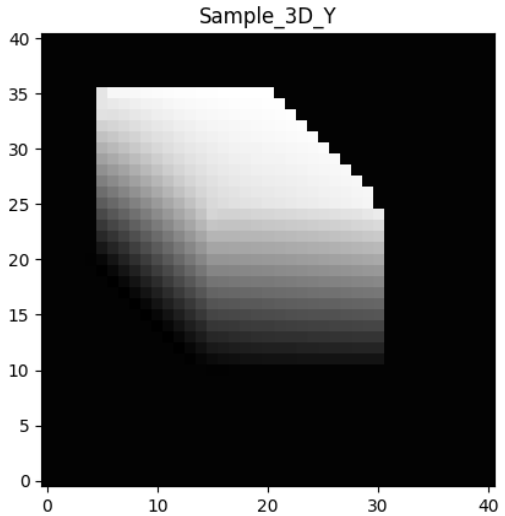


Problem overview

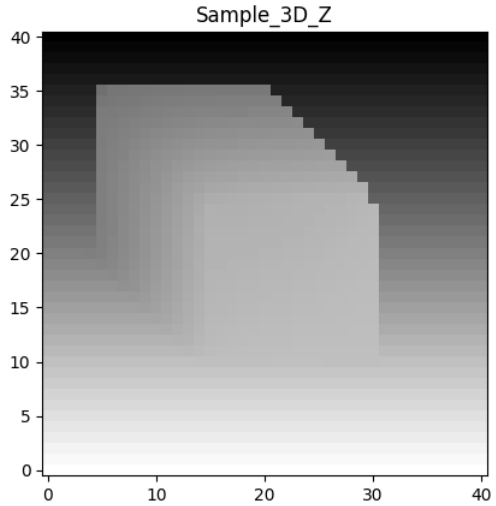
Given



Goal: Generate



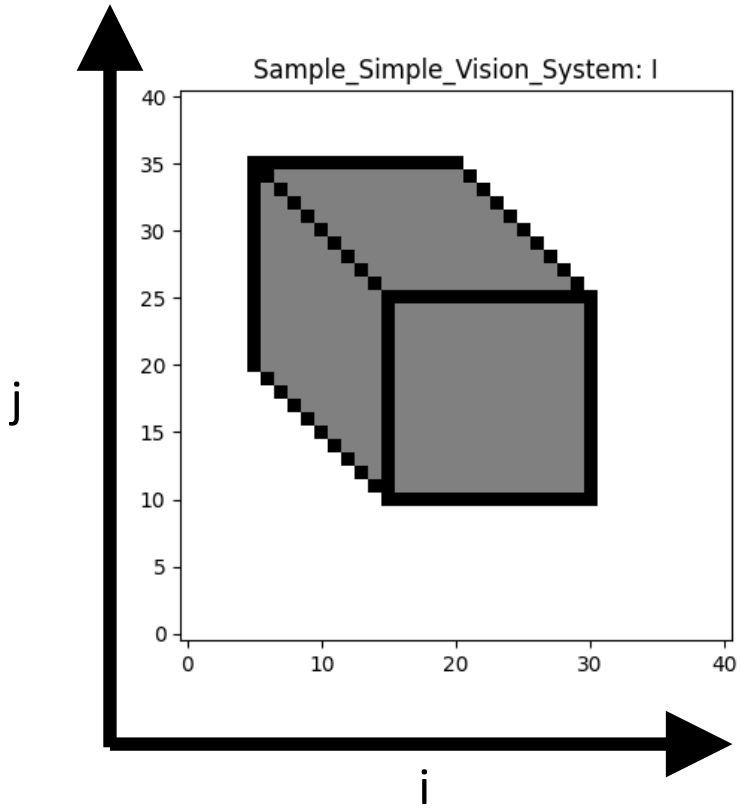
Y: 3D height



Z: 3D depth

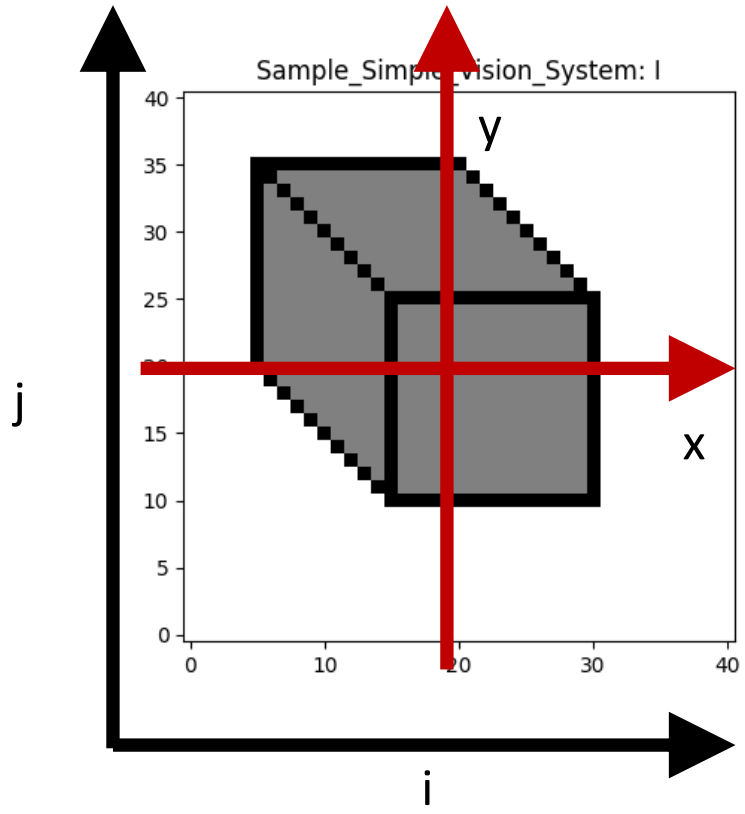
Convention

- In this homework, given a map (or a matrix), say I
  - I[i, j] means the i-th horizontal index (left-right) and j-th vertical index (bottom-up)
  - i >= 0, j >= 0

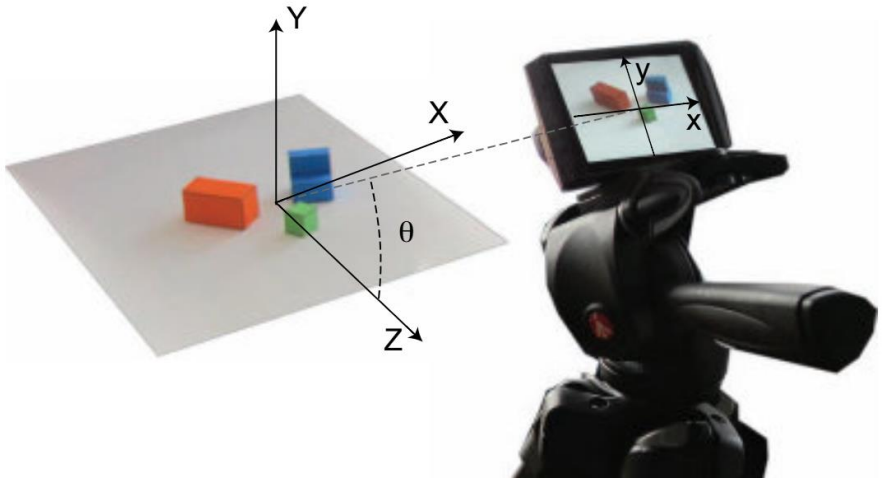


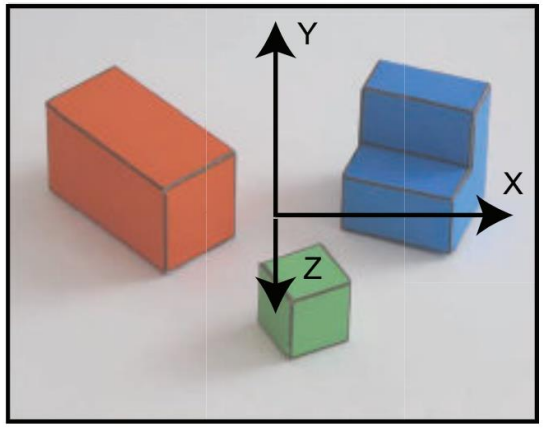
Convention

- There are differences in coordinate systems between (i, j) and (x, y)
- We thus provide a function x, y = image_plane(args, I)
  - x[i, j] is the 2D x location of pixel (i, j)
  - y[i, j] is the 2D y location of pixel (i, j)



Recall (from lectures)





We want to know X(x, y), Y(x, y), and Z(x, y) from the given image!

What we know:

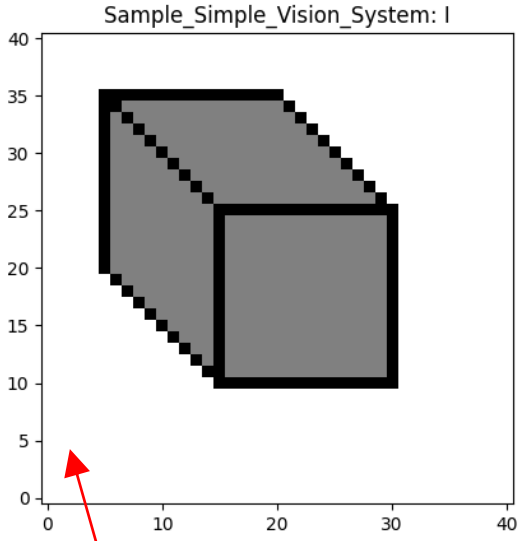
$$x = X$$

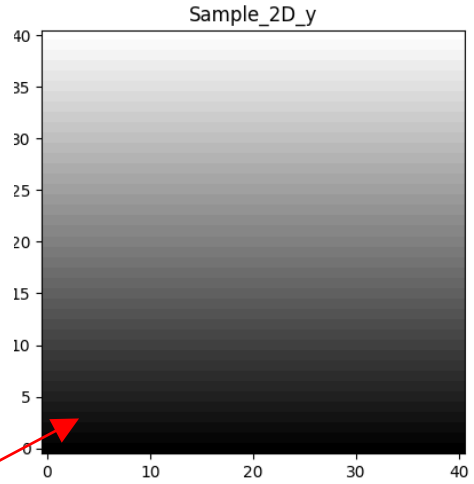
$$y = \cos(\theta)Y - \sin(\theta)Z$$

We need some cues from images and the 3D world!

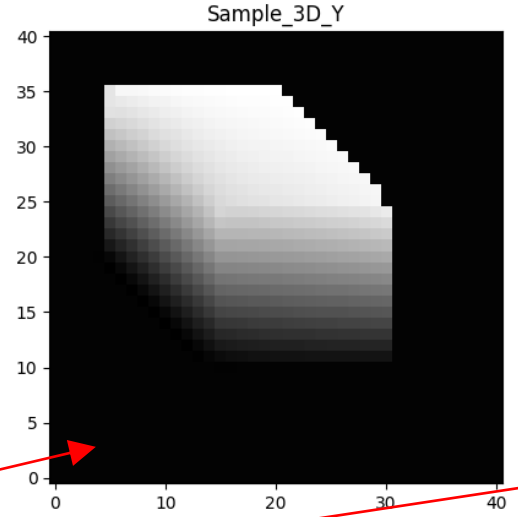
[Figure credit: A. Torralba, P. Isola, and W. T. Freeman, Foundations of Computer Vision.]

For implementation, locations are indexed by [I, j]

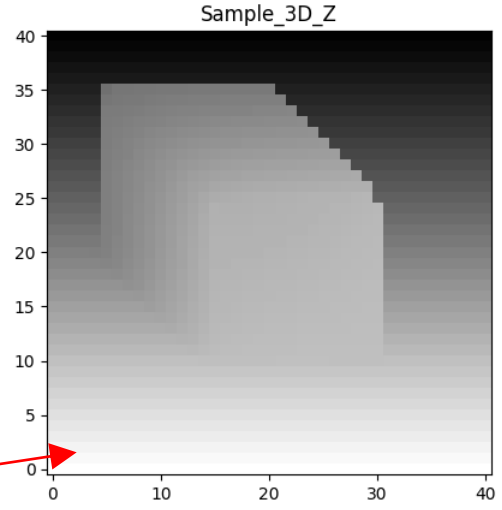




y: 2D vertical



Y: 3D height



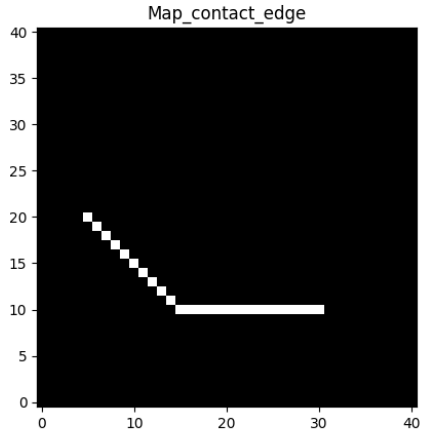
Z: 3D depth

(i, j)

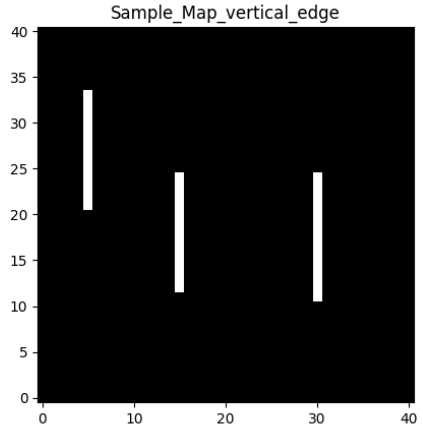
$$y = \cos(\theta)Y - \sin(\theta)Z \quad \Rightarrow \quad y[i,j] = \cos(\theta)Y[i,j] - \sin(\theta)Z[i,j]$$

Cue 1: edges (white pixels mean edges)

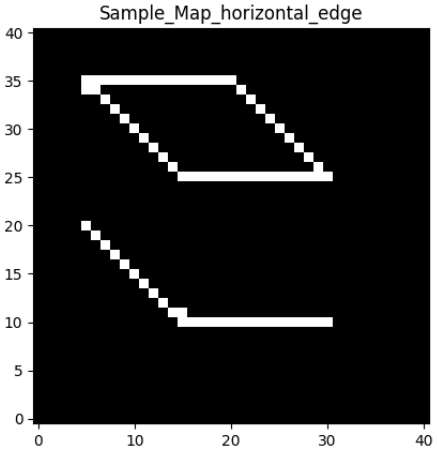
All edges



Contact edges



Vertical edges

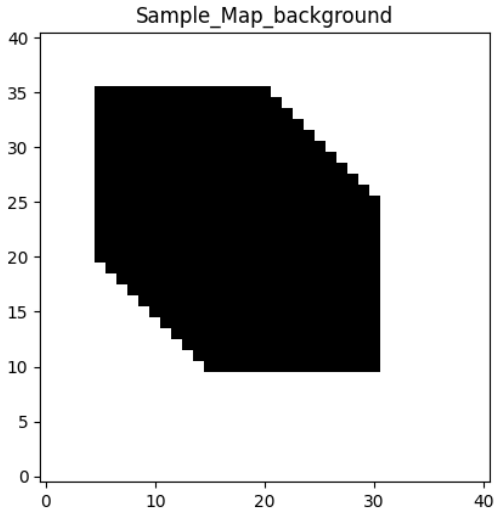


Horizontal

You need to find edge locations!

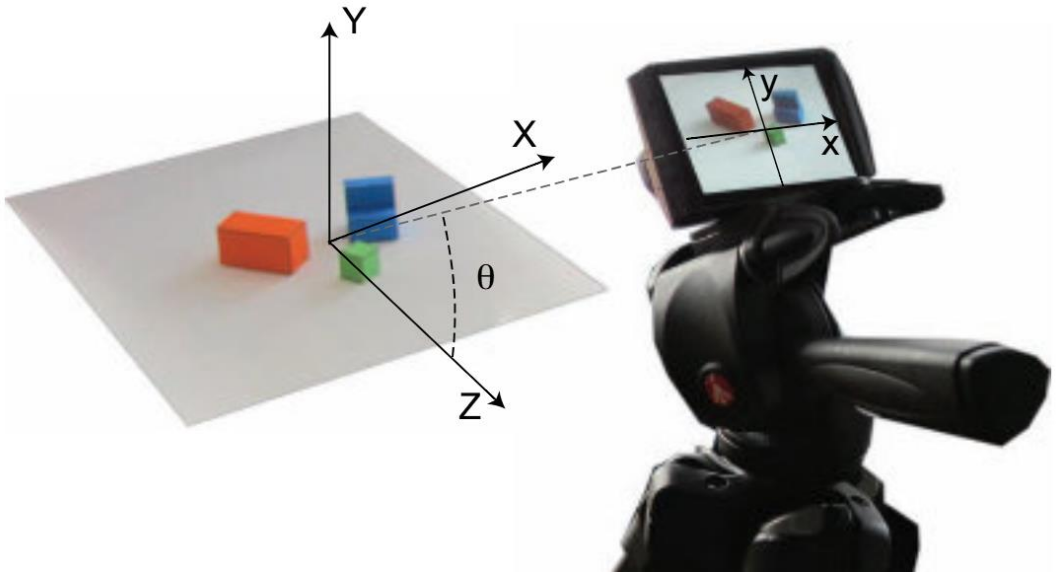
Cue 2: Surfaces & Cue 3: properties from 3D to 2D

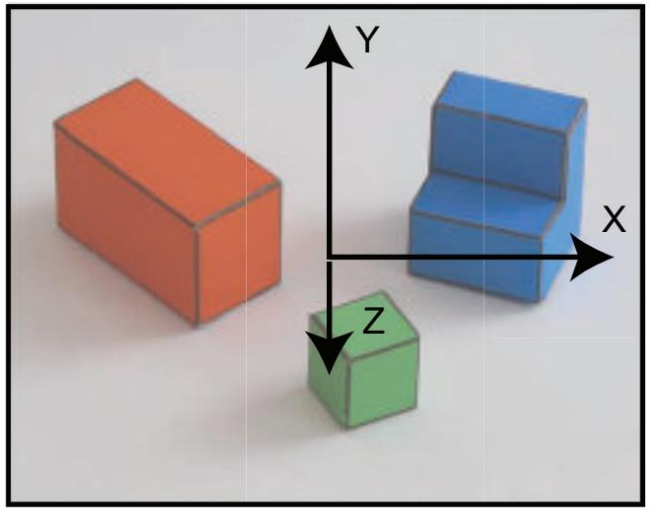
- Separate into **foregrounds (figures)/backgrounds**



- **Not always true, but let's assume it is true**
  - Vertical edges in 2D mean vertical in 3D
  - Non-vertical edges in 2D means horizontal in 3D

Recall (from lectures)





We want to know X(x, y), Y(x, y), and Z(x, y) from the given image!

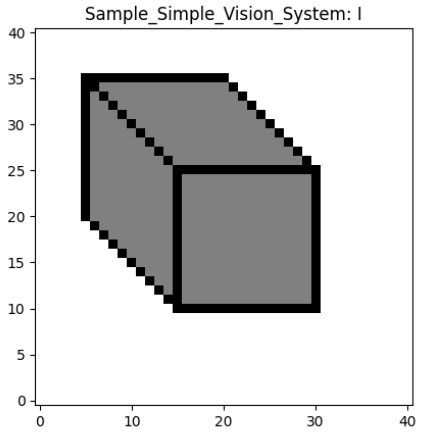
$$x = X$$

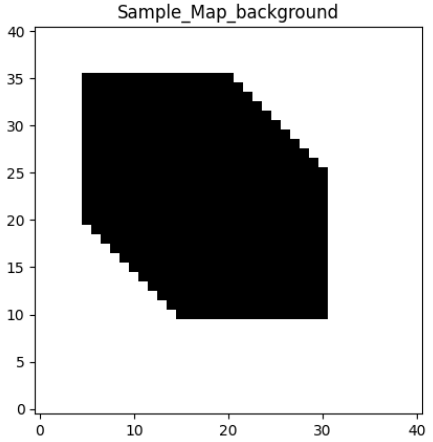
$$y = \cos(\theta)Y - \sin(\theta)Z$$

If we know Y(x, y), we know Z(x, y)

[Figure credit: A. Torralba, P. Isola, and W. T. Freeman, Foundations of Computer Vision.]

Estimating Y[I, j]: cues from the background





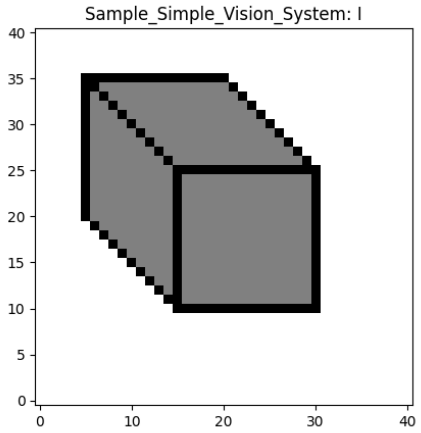
$$\tilde{Y} = \begin{bmatrix} Y(1,1) \\ Y(1,2) \\ \dots \\ Y(1,N) \\ Y(2,1) \\ Y(2,2) \\ \dots \\ Y(2,N) \\ \dots \\ Y(M,N) \end{bmatrix}$$

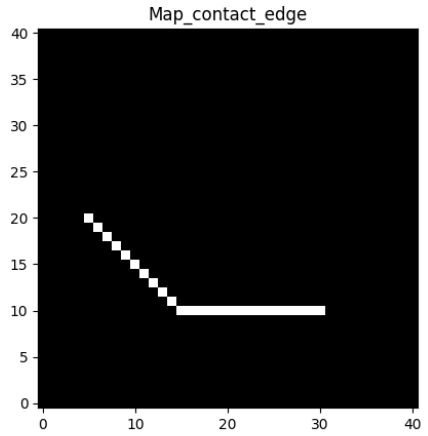
$$[0, \dots, 0, 1, \dots, 0] \begin{bmatrix} Y(1,1) \\ Y(1,2) \\ \dots \\ Y(1,N) \\ Y(2,1) \\ Y(2,2) \\ \dots \\ Y(2,N) \\ \dots \\ Y(M,N) \end{bmatrix} = 0$$

One row for each background pixel

Meaning: these locations have height 0

Estimating Y[I, j]: cues from contact edges





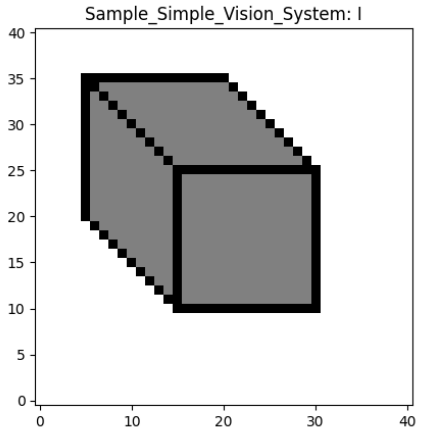
$$\tilde{Y} = \begin{bmatrix} Y(1,1) \\ Y(1,2) \\ \dots \\ Y(1,N) \\ Y(2,1) \\ Y(2,2) \\ \dots \\ Y(2,N) \\ \dots \\ Y(M,N) \end{bmatrix}$$

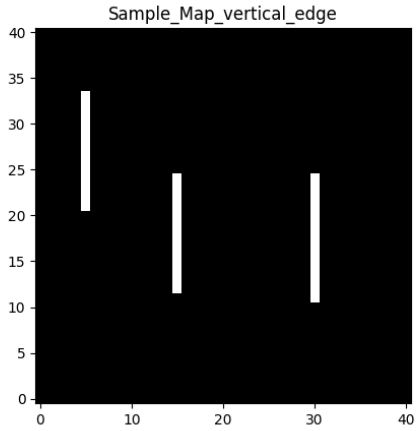
$$[0, \dots, 0, 1, \dots, 0] \begin{bmatrix} Y(1,1) \\ Y(1,2) \\ \dots \\ Y(1,N) \\ Y(2,1) \\ Y(2,2) \\ \dots \\ Y(2,N) \\ \dots \\ Y(M,N) \end{bmatrix} = 0$$

One row for each contact edge pixel

Meaning: these locations have height 0

Estimating Y[I, j]: cues from vertical edges





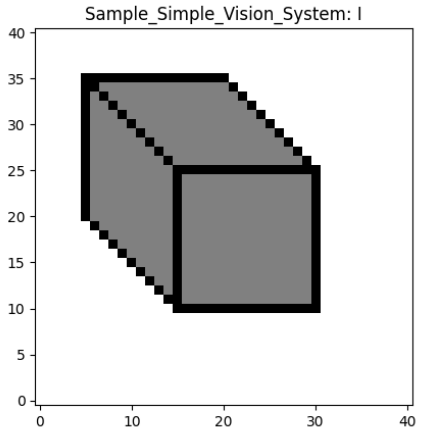
$$Y[i,j] - Y[i,j-1] = 1/\cos(\theta)$$

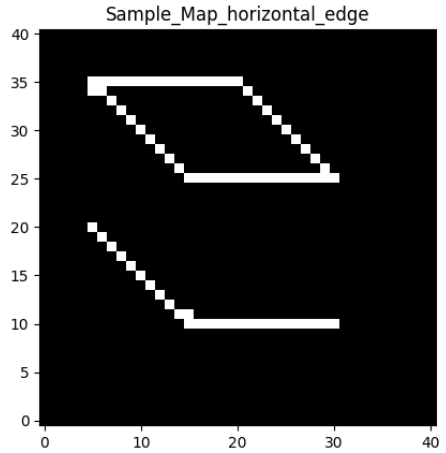
$$[0, \dots, 0, -1, 1, \dots, 0] \begin{bmatrix} Y(1,1) \\ Y(1,2) \\ \dots \\ Y(1,N) \\ Y(2,1) \\ Y(2,2) \\ \dots \\ Y(2,N) \\ \dots \\ Y(M,N) \end{bmatrix} = 1/\cos(\theta)$$

One row for each vertical edge pixel

Meaning: two vertically consecutive pixels have 3D height difference $1/\cos(\theta)$

Estimating Y[I, j]: cues from horizontal edges

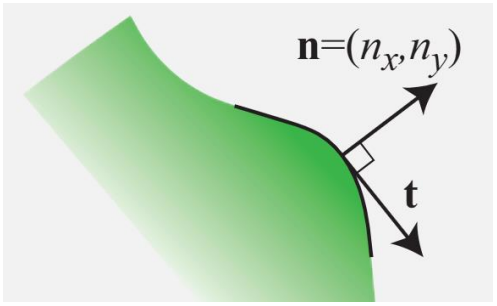




Horizontal edges: Y won't change along the edge

$$-n_y[i,j](Y[i,j]-Y[i-1,j])+n_x[i,j](Y[i,j]-Y[i,j-1])=0$$

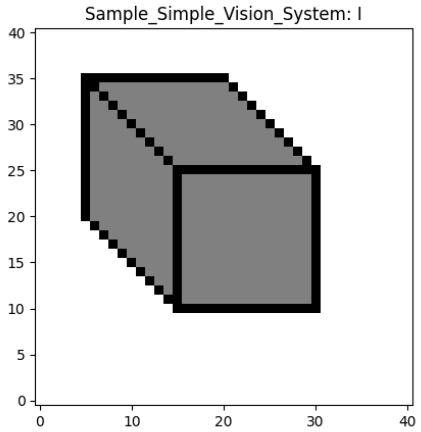
In the code, we have computed n_y and n_x for you!

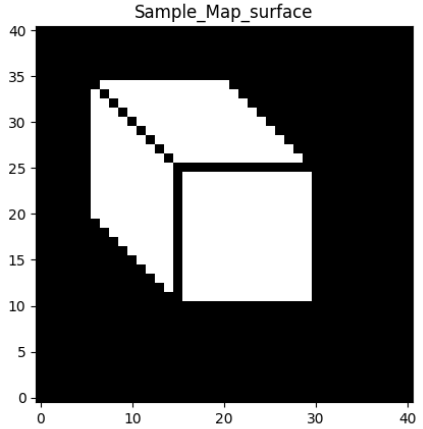


$$[0,\dots -n_x,(-n_y+n_x),0\dots 0,n_y,\dots,0]\begin{bmatrix}Y(1,1)\\Y(1,2)\\\dots\\Y(1,N)\\Y(2,1)\\Y(2,2)\\\dots\\Y(2,N)\\\dots\\Y(M,N)\end{bmatrix}=0$$

One row for each horizontal edge pixel

Estimating Y[I, j]: cues from surfaces





Three rows for each surface pixel

Surfaces: flat, not curved

$$2Y[i,j] - Y[i+1,j] - Y[i-1,j] = 0$$

$$2Y[i,j] - Y[i,j+1] - Y[i,j-1] = 0$$

$$Y[i,j] - Y[i,j-1] - Y[i-1,j] + Y[i-1,j-1] = 0$$

$$[0, \dots, -1, 0 \dots, 2, 0, \dots -1, 0, \dots]\tilde{Y} = 0$$

$$[0, \dots, -1, 0 \dots, 2, 0, \dots -1, 0, \dots]\tilde{Y} = 0$$

$$[0, \dots, -1, 0 \dots, 1, \dots 0, \dots 1, 0, \dots -1, 0, \dots]\tilde{Y} = 0$$

Put all the constraints together

$$\begin{bmatrix} - & \boldsymbol{a}_1 & - \\ - & \boldsymbol{a}_2 & - \\ & \dots & \end{bmatrix} \tilde{Y} = \boldsymbol{b}$$

$$\boldsymbol{A}\tilde{Y} = \boldsymbol{b}$$

Least square solution!

$$\tilde{Y} = \left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\mathrm{T}}\,\boldsymbol{b}$$

For example,

$$\boldsymbol{A} = \begin{bmatrix} \vdots \\ 0, \dots -n_x, (-n_y + n_x), 0 \dots 0, n_y, \dots, 0 \\ \vdots \\ 0, \dots, 0, -1, 1, \dots, 0 \\ \vdots \\ 0, \dots, -1, 0 \dots, 2, 0, \dots -1, 0, \dots \\ \vdots \end{bmatrix}$$

$$\boldsymbol{b} = \begin{bmatrix} \vdots \\ 0 \\ \vdots \\ 1/\cos(\theta) \\ \vdots \\ 0 \\ \vdots \end{bmatrix}$$

A is like 2300-by-1681

Estimating Z from Y and y

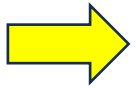
$$x = X$$

$$y = \cos(\theta)Y - \sin(\theta)Z$$

$$y[i,j] = \cos(\theta)Y[i,j] - \sin(\theta)Z[i,j]$$

Why linear system? Try this toy example

1	2	3
4	?	6
7	8	9



$$\tilde{Y} = \begin{bmatrix} Y(1,1) \\ Y(1,2) \\ Y(1,3) \\ Y(2,1) \\ Y(2,2) \\ Y(2,3) \\ Y(3,1) \\ Y(3,2) \\ Y(3,3) \end{bmatrix}$$

$$\begin{bmatrix} 1,0,0,0,0,0,0,0,0 \\ 0,1,0,0,0,0,0,0,0 \\ 0,0,1,0,0,0,0,0,0 \\ 0,0,0,1,0,0,0,0,0 \\ 0,0,0,0,0,1,0,0,0 \\ 0,0,0,0,0,0,1,0,0 \\ 0,0,0,0,0,0,0,1,0 \\ 0,0,0,0,0,0,0,0,1 \\ 0,0,0,-1,2,-1,0,0,0 \\ 0,-1,0,0,2,0,0,-1,0 \\ 1,-1,0,-1,1,0,0,0,0 \end{bmatrix} \begin{bmatrix} Y(1,1) \\ Y(1,2) \\ Y(1,3) \\ Y(2,1) \\ Y(2,2) \\ Y(2,3) \\ Y(3,1) \\ Y(3,2) \\ Y(3,3) \end{bmatrix} = \begin{bmatrix} 7 \\ 8 \\ 9 \\ 4 \\ 6 \\ 1 \\ 2 \\ 3 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

By solving it through $\tilde{Y} = \left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\mathrm{T}}\,\boldsymbol{b}$,

$Y(2,2) \approx 5$

Additional hints on Part 4

- See an example implementation for including rows corresponding to the background pixels below. I've also implemented the least square solution at the end. What you need to do is to fill in similar for loops for the vertical, horizontal, and contact edges, as well as the surface pixels.

```
#### Your job 4 starts here: generate Y ####
A = np.zeros((0, I.size))
b = np.zeros((0, 1))

for i in range(I.shape[0]):
    for j in range(I.shape[1]):
        if Map_background[i, j] == 1:
            temp = np.zeros((1, I.size))
            temp[0, i + j * I.shape[0]] = 1
            A = np.concatenate((A, temp), axis=0)
            b = np.concatenate((b, np.zeros((1, 1))), axis=0)

# You need to fill in similar for loops for contact edges, horizontal edges, vertical edges, and surface here

Y = np.matmul(np.linalg.inv(np.matmul(A.transpose(), A) + 0.0000001 * np.identity(A.shape[1])), np.matmul(A.transpose(),b))
Y = Y.reshape((I.shape[1], I.shape[0])).transpose()
#### Your job 4 ends here: generate Y ####
```