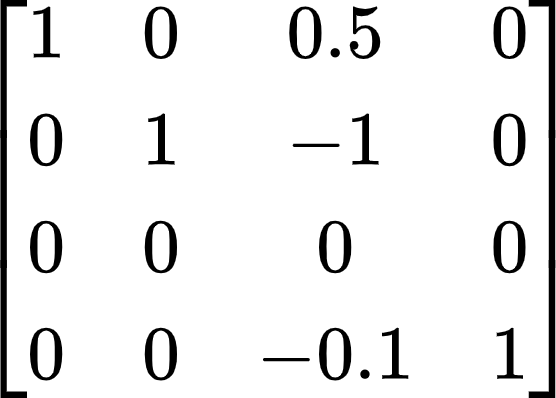
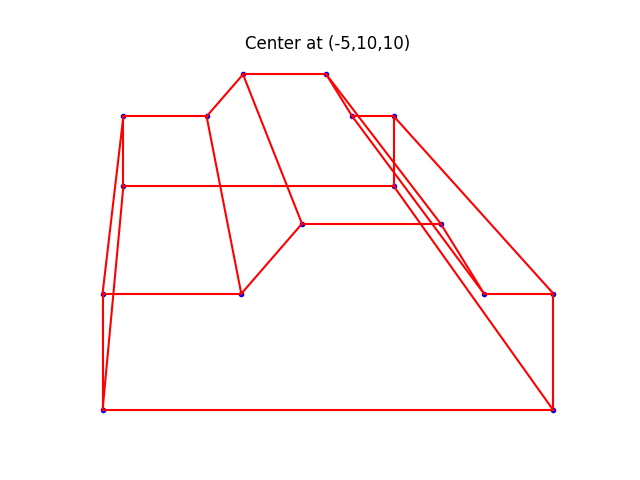
**Case Study: Computer Graphics in Automotive Design**

**By:Christopher Fallas Aguero and Dhruv Rana**

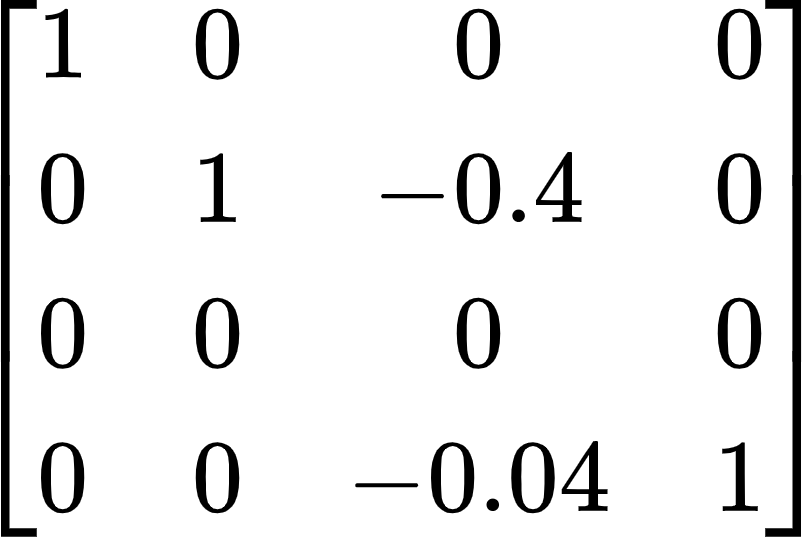
1. Perspective projection at (-5, 10, 10).
   1. What is the matrix of the perspective projection?

P = 

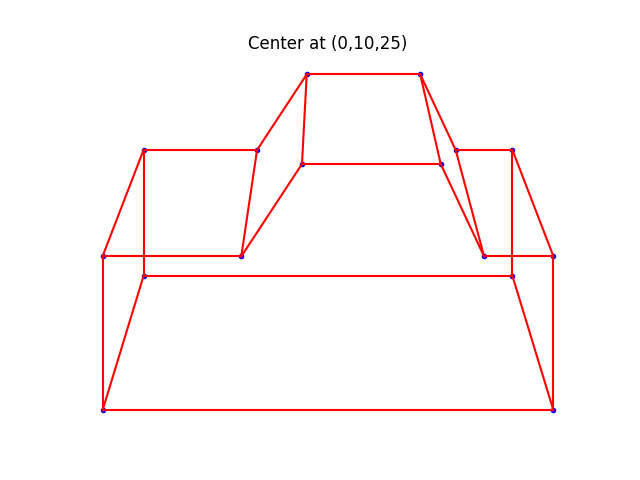
* 1. Figure:



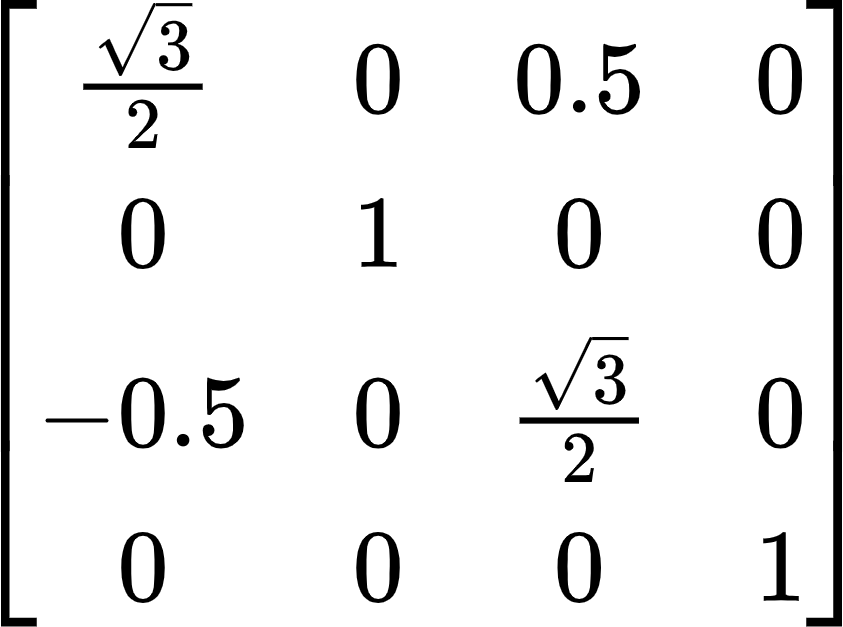
1. Perspective projection at (0, 10, 25).
   1. What is the matrix of the perspective projection?

P = 

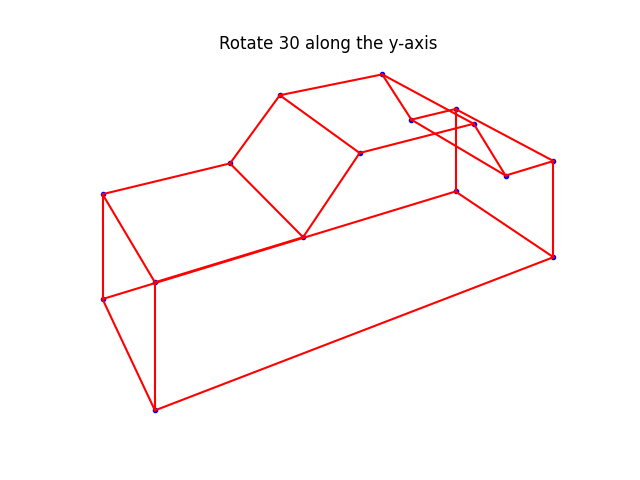
* 1. Figure:



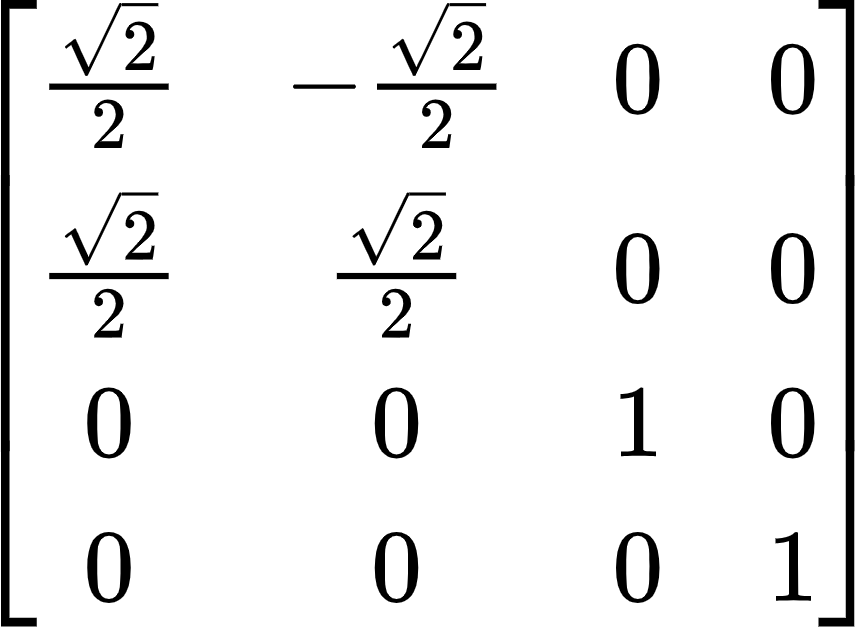
1. Rotate 30 degrees about the y-axis centered at (0, 10, 25).
   1. What is the matrix of the rotation?

Ay = 

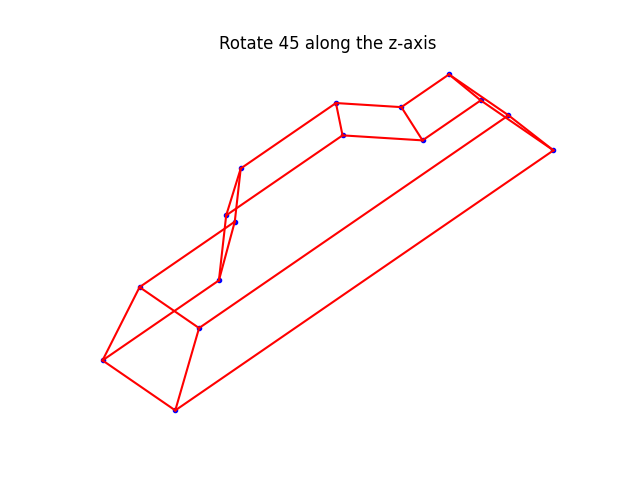
* 1. How does this figure compare with that in Question 2?
     1. The difference between this figure and the figure in Question 2 is that this figure is rotated 30 degrees along the y-axis. This allows us to get a better view of the back of the car, while also obstructing the view of the front of the car. In both figures, we see the car from an aerial view.
  2. Figure:



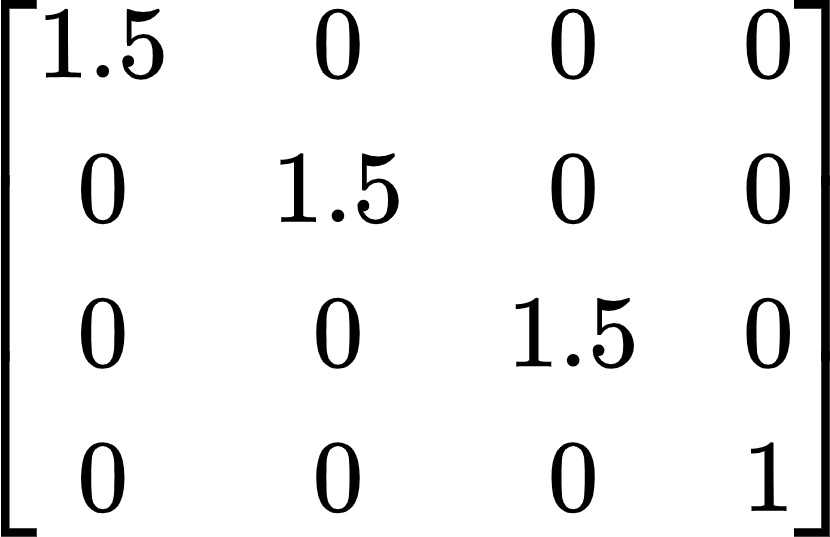
1. Rotate 45 degrees about the z-axis centered at (0, 10, 25).
   1. What is the matrix of the rotation?

Az = 

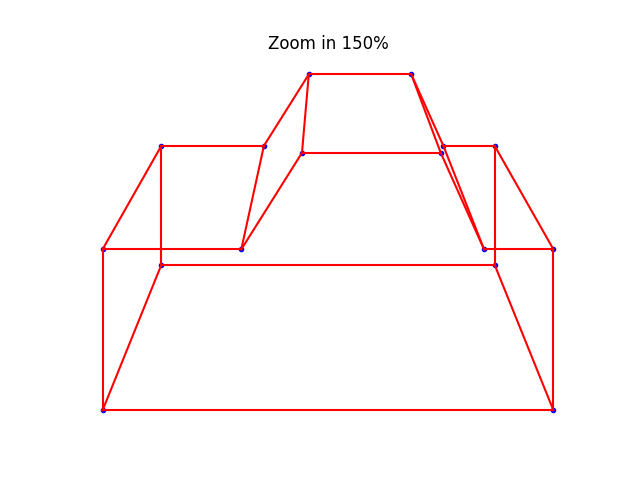
* 1. How does this figure compare with that in Question 2?
     1. The difference between this and the figure in Question 2 is that this figure is rotated 45 degrees along the z-axis. This allows us to get a better view of the side of the car, while the view of the front of the car is blocked. The view of this figure is more of a frontal view than an aerial view that of the figure in Question 2.
  2. Figure:



1. Zoom in 150% centered at (0,10, 25).
   1. What is the matrix of the zoom?

A = 

* 1. How does this figure compare with that in Question 2?
     1. The difference between this figure and the figure in Question 2 is that we see this figure more close up and get a more frontal or straight view than that of the figure in Question 2 which is a more aerial view. This figure looks almost like it got bigger, which shows we zoomed-in. This figure also seems more stretched out from the z-axis, showing more depth than the figure in Question 2.
  2. Figure:



1. Python Code as plain text.

########################################################

# Import statements necessary for program to run

import math

import numpy as np

import matplotlib.pyplot as plt

# from matplotlib.pyplot import plot, ion, show

# Above import not used

########################################################

# ion()

# Above code does not work depending on your IDE, use plt.show() at bottom

########################################################

########################################################

# Vertices and adjacency matrix to determine size and shape of graph

D = np.array([[-6.5, -6.5, -6.5, -6.5, -2.5, -2.5, -0.75, -0.75, 3.25, 3.25, 4.5, 4.5, 6.5, 6.5, 6.5, 6.5],

[-2, -2, .5, .5, .5, .5, 2, 2, 2, 2, .5, .5, .5, .5, -2, -2],

[-2.5, 2.5, 2.5, -2.5, -2.5, 2.5, -2.5, 2.5, -2.5, 2.5, -2.5, 2.5, -2.5, 2.5, 2.5, -2.5],

[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])

C = np.array([ [0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],

[1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],

[0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0],

[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1],

[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0]])

#####################################################################################

# Perspective Projection at (-5,10,10)

P2 = np.array([[1, 0, .5, 0],

[0, 1, -1, 0],

[0, 0, 0, 0],

[0, 0, -0.1, 1]]) # Projection matrix

PD2 = np.matmul(P2, D)

PD2[0, :] = PD2[0, :] / PD2[3, :]

PD2[1, :] = PD2[1, :] / PD2[3, :]

f, ax2 = plt.subplots(1)

ax2.plot(PD2[0, :], PD2[1, :], 'b.')

for i in range(16):

for j in range(i):

if C[i, j] == 1:

ax2.plot([PD2[0, i], PD2[0, j]], [PD2[1, i], PD2[1, j]], 'r-')

ax2.set\_title('Center at (-5,10,10)')

ax2.axis('off')

#########################################

# Perspective Projection at (0,10,25)

P3 = np.array([[1, 0, 0, 0],

[0, 1, -0.4, 0],

[0, 0, 0, 0],

[0, 0, -0.04, 1]]) # Projection matrix

PD3 = np.matmul(P3, D)

PD3[0, :] = PD3[0, :] / PD3[3, :]

PD3[1, :] = PD3[1, :] / PD3[3, :]

f, ax3 = plt.subplots(1)

ax3.plot(PD3[0, :], PD3[1, :], 'b.')

for i in range(16):

for j in range(i):

if C[i, j] == 1:

ax3.plot([PD3[0, i], PD3[0, j]], [PD3[1, i], PD3[1, j]], 'r-')

ax3.set\_title('Center at (0,10,25)')

ax3.axis('off')

#########################################

# Rotate by 30 of y-axis and projection at (0,10,25)

P4 = np.array([[(math.sqrt(3)/2), 0, 0.5, 0],

[0, 1, 0, 0],

[-.5, 0, (math.sqrt(3)/2), 0],

[0, 0, 0, 1]]) # Projection matrix

test = np.matmul(P3, P4) # Perspective matrix times rotation matrix

PD4 = np.matmul(test, D)

PD4[0, :] = PD4[0, :] / PD4[3, :]

PD4[1, :] = PD4[1, :] / PD4[3, :]

f, ax4 = plt.subplots(1)

ax4.plot(PD4[0, :], PD4[1, :], 'b.')

for i in range(16):

for j in range(i):

if C[i, j] == 1:

ax4.plot([PD4[0, i], PD4[0, j]], [PD4[1, i], PD4[1, j]], 'r-')

ax4.set\_title('Rotate 30 along the y-axis')

ax4.axis('off')

#########################################

# Rotate by 45 of z-axis and projection at (0,10,25)

P5 = np.array([[(math.sqrt(2)/2), -(math.sqrt(2)/2), 0, 0],

[(math.sqrt(2)/2), (math.sqrt(2)/2), 0, 0],

[0, 0, 1, 0],

[0, 0, 0, 1]]) # Projection matrix

test = np.matmul(P3, P5) # Perspective matrix times rotation matrix

PD5 = np.matmul(test, D)

PD5[0, :] = PD5[0, :] / PD5[3, :]

PD5[1, :] = PD5[1, :] / PD5[3, :]

f, ax5 = plt.subplots(1)

ax5.plot(PD5[0, :], PD5[1, :], 'b.')

for i in range(16):

for j in range(i):

if C[i, j] == 1:

ax5.plot([PD5[0, i], PD5[0, j]], [PD5[1, i], PD5[1, j]], 'r-')

ax5.set\_title('Rotate 45 along the z-axis')

ax5.axis('off')

#########################################

# Zoom in by 150% and projection at (0,10,25)

P6 = np.array([[1.5, 0, 0, 0],

[0, 1.5, 0, 0],

[0, 0, 1.5, 0],

[0, 0, 0, 1]]) # Projection matrix

PD6 = np.matmul((np.matmul(P3, P6)), D)

PD6[0, :] = PD6[0, :] / PD6[3, :]

PD6[1, :] = PD6[1, :] / PD6[3, :]

f, ax6 = plt.subplots(1)

ax6.plot(PD6[0, :], PD6[1, :], 'b.')

for i in range(16):

for j in range(i):

if C[i, j] == 1:

ax6.plot([PD6[0, i], PD6[0, j]], [PD6[1, i], PD6[1, j]], 'r-')

ax6.set\_title('Zoom in 150%')

ax6.axis('off')

plt.show()