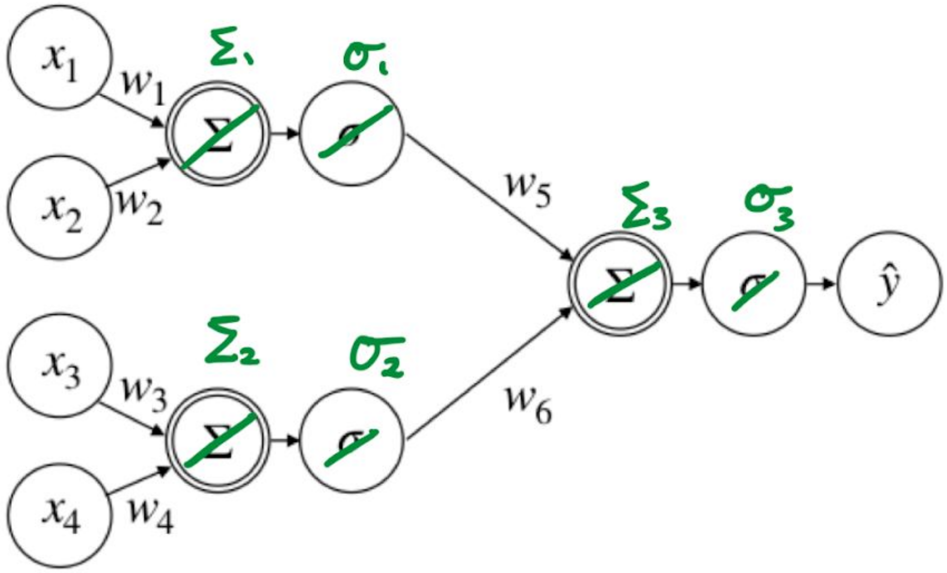
Question 1

For easier tracking of the notations, we will change the default notations of the diagram as follows:




$$\Sigma_1 = w_1 x_1 + w_2 x_2$$
$$= (0.9)(0.75) + (-1.1)(-0.63)$$
$$= 0.675 + 0.693$$
$$= 1.368$$

$\boxed{\text{Forward Pass}}$

$$\sigma_1 = \frac{1}{1+e^{-1.368}} = 0.797056831$$

$$\Sigma_2 = w_3 x_3 + w_4 x_4$$
$$= (-0.3)(0.24) + (-1.7)(0.8)$$
$$= -0.072 + (-1.36)$$
$$= -1.432$$

$$\sigma_2 = \frac{1}{1+e^{1.432}} = 0.192787252$$

$\Sigma_3 = w_5 \sigma_1 + w_6 \sigma_2$
$= 0.8\,(0.787056831) + (-0.2)\,(0.192787252)$
$= 0.599088014$

$$\sigma_3 = \frac{1}{1+e^{-0.599088014}} = 0.64544763$$

total error $= (0.5 - 0.64544763)^2$
$= 0.021155013$

## Backward Pass

$$\frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial \sigma_3} \cdot \frac{\partial \sigma_3}{\partial \Sigma_3} \cdot \frac{\partial \Sigma_3}{\partial \sigma_2} \cdot \frac{\partial \sigma_2}{\partial \Sigma_2} \cdot \frac{\partial \Sigma_2}{\partial w_3}$$

$L = (0.5 - \sigma_3)^2$

$\frac{\partial L}{\partial \sigma_3} = 2(0.5 - \sigma_3)(-1)$

$= -1 + 2\sigma_3$ — (1)

$\sigma_3 = \frac{1}{1+e^{-\Sigma_3}}$

$\frac{\partial \sigma_3}{\partial \Sigma_3} = \frac{-e^{-\Sigma_3}}{(1+e^{-\Sigma_3})^2}$ — (2)

$$\Sigma_3 = w_5 \sigma_1 + w_6 \sigma_2$$

$$\frac{\partial \Sigma_3}{\partial \sigma_2} = w_6 \qquad - (3)$$

$$\sigma_2 = \frac{1}{1 + e^{-\Sigma_2}}$$

$$\frac{\partial \sigma_2}{\partial \Sigma_2} = \frac{-e^{-\Sigma_2}}{(1 + e^{-\Sigma_2})^2} \qquad - (4)$$

$$\Sigma_2 = w_3 x_3 + w_4 x_4$$

$$\frac{\partial \Sigma_2}{\partial w_3} = x_3$$

$$\therefore \frac{\partial L}{\partial w_3} = (-1 + 2\sigma_3)\left(-\frac{e^{-\Sigma_3}}{(1+e^{-\Sigma_3})^2}\right)(w_6)\left(\frac{-e^{-\Sigma_2}}{(1+e^{-\Sigma_2})^2}\right)(x_3)$$

$$= 0.000613553421 \quad //$$

Question 2i

$$\begin{pmatrix} f & 0 & P_x \\ 0 & f & P_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_0 \\ Y_0 \\ Z_0 \end{pmatrix} = \begin{pmatrix} fX_0 + P_x Z_0 \\ fY_0 + P_y Z_0 \\ Z_0 \end{pmatrix}$$

Also since $\begin{pmatrix} fX_0 + P_x Z_0 \\ fY_0 + P_y Z_0 \\ Z_0 \end{pmatrix} = \begin{pmatrix} wx \\ wy \\ w \end{pmatrix}$

then $w = Z_0$

this implies $fX_0 + P_x Z_0 = Zx$

$$x = \frac{fX_0 + P_x Z_0}{Z_0}$$

$$x = \frac{fX_0}{Z_0} + P_x$$

and $fY_0 + P_y Z_0 = Z_0 y$

$$y = \frac{fY_0 + P_y Z_0}{Z_0}$$

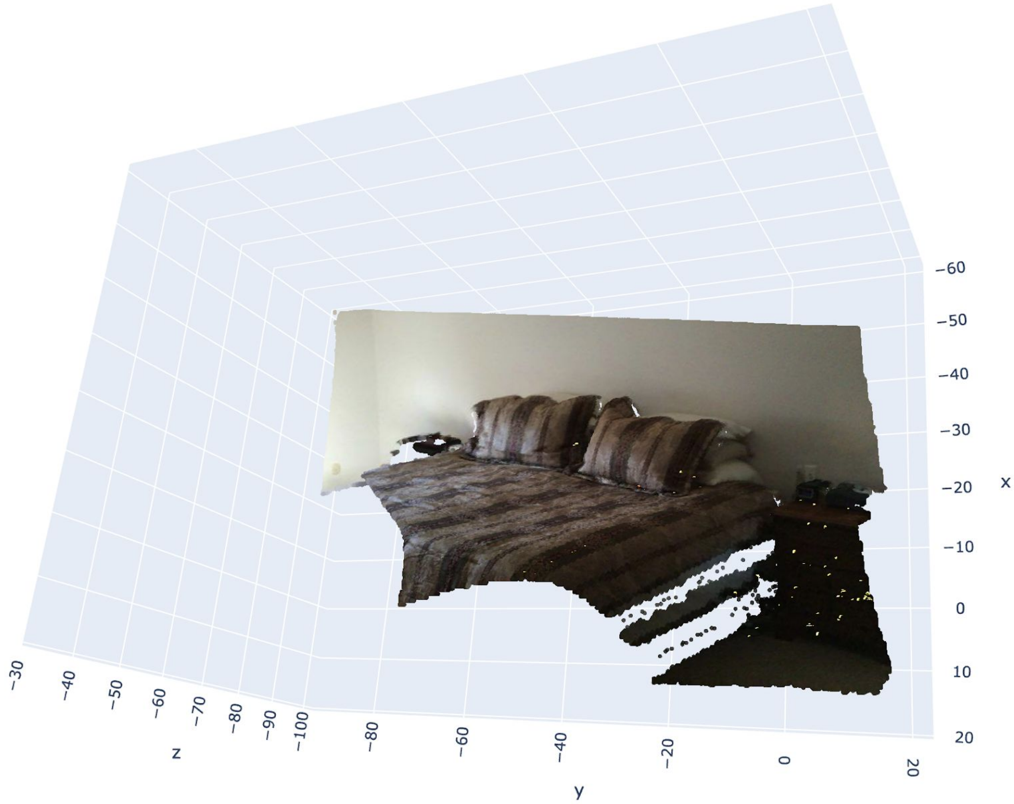
$$= \frac{fY_0}{Z_0} + P_y$$

※

Question 3

```python
#initilize the rotation matrix R 4x4
new_R = np.zeros((R.shape[0] + 1, R.shape[1] + 1))
for i in range(R.shape[0]):
    for j in range(R.shape[1]):
        new_R[i][j] = R[i][j]

new_R[new_R.shape[0]-1 ][new_R.shape[1]-1] = 1

#initilize [I | -c] 4x4
ic = np.identity(4)
for i in range (t.shape[0]):
    ic[i][3] = t[i]


xyz_c = np.zeros((4,1))

results = []

#loop through each pixel in depthImage and rgbImage
for i in range(depth.shape[0]):
    for j in range(depth.shape[1]):
        if depth[i][j] == 0:
            continue

        '''
        initialize the projection matrix
        [1 0 0 0]
        [0 1 0 0]
        [0 0 1 0]
        '''
        proj = np.array([[1,0,0,0], [0,1,0,0], [0,0,1,0]])

        #trace back the coordinate from the depthImage
        w_xy= np.array([[depth[i][j]*i],[depth[i][j]*j],[depth[i][j]]])


        #inverse the camera calibration matrix K
        inv_intrinsics = np.linalg.inv(intrinsics)

        #calculate the 3D camera coordinate system
        xyz_c = np.matmul(inv_intrinsics, w_xy)
        xyz_c_homo = np.append(xyz_c, [[1]], axis=0)

        #calculate the extrinsic matrix R[I | -c] and inversed it
        R_ic = np.matmul(new_R, ic)
        inv_R_ic = np.linalg.inv(R_ic)

        #calculate the 3D world coordinate
        xyz_w_homo = np.matmul(inv_R_ic, xyz_c_homo)


        output = [xyz_w_homo[0][0], xyz_w_homo[1][0], xyz_w_homo[2][0], rgb[i][j][0], rgb[i][j][1], rgb[i][j][2]]
        results.append(output)
```

Image 1 Result



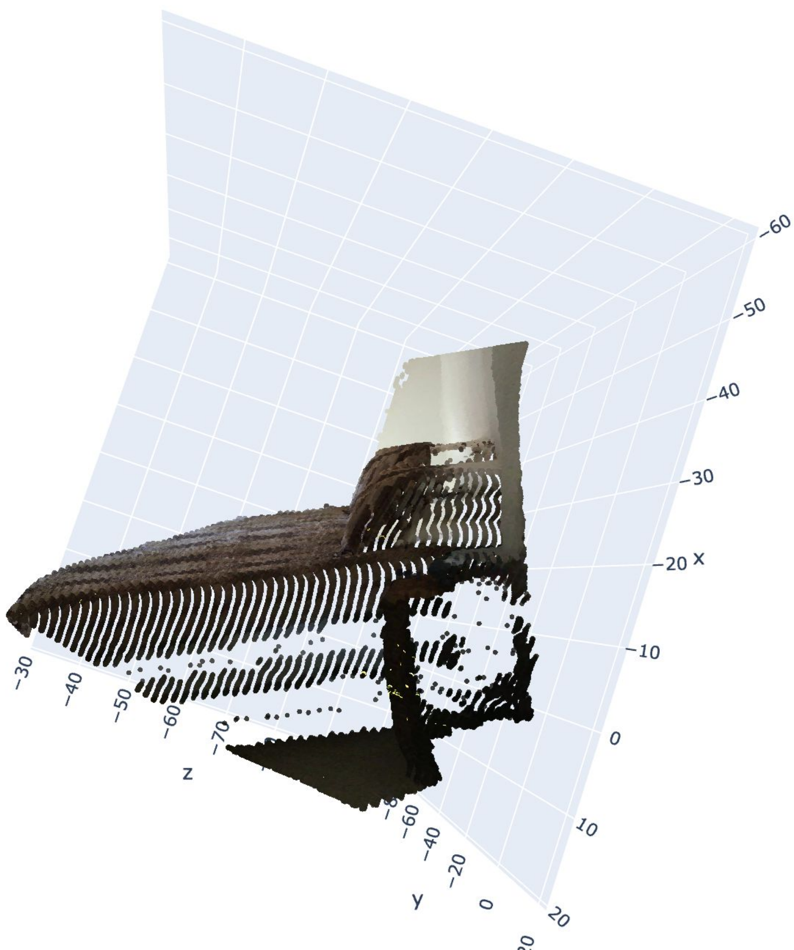
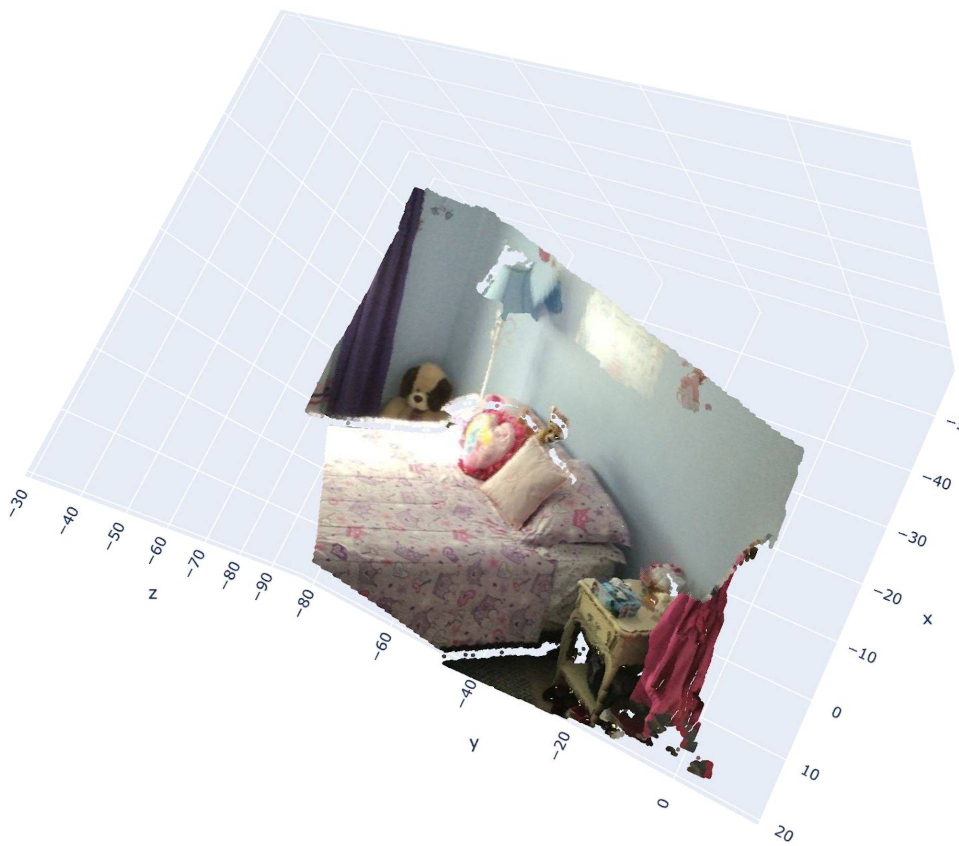



Image 2 Result



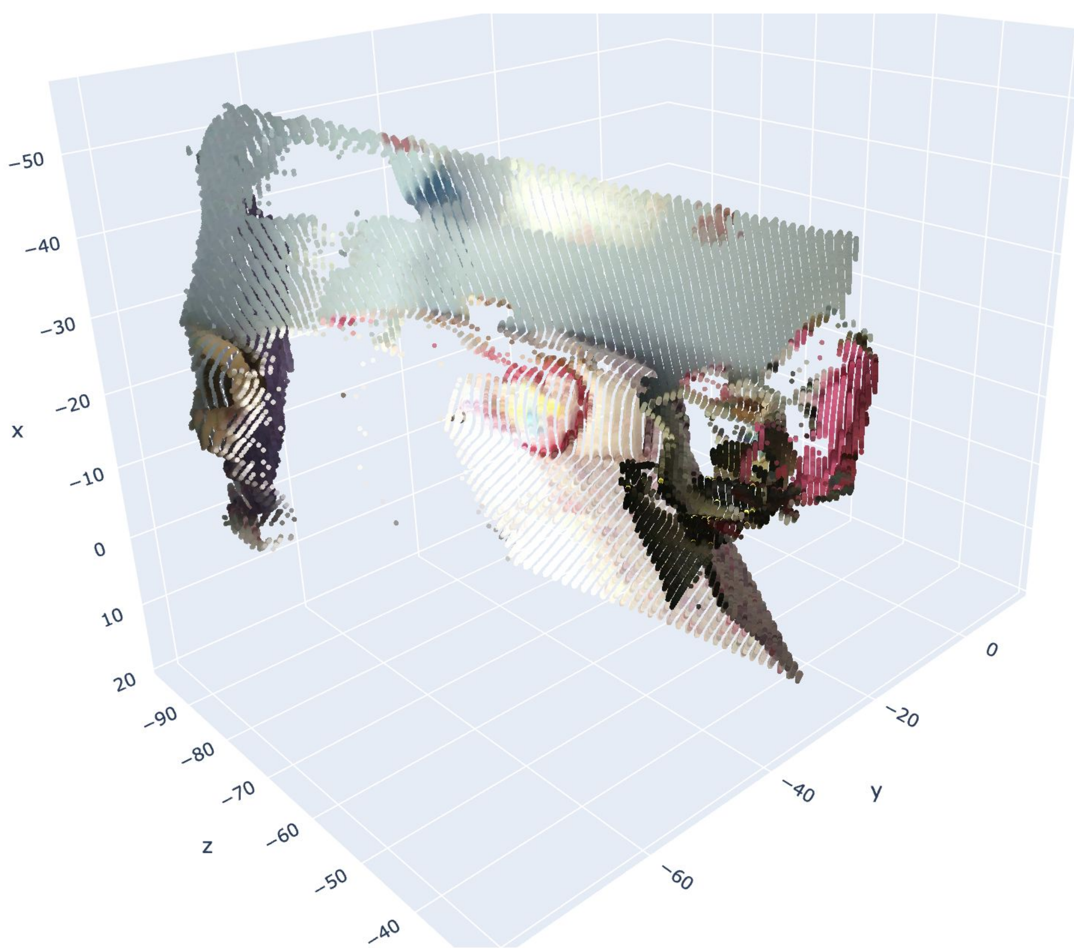



Image 3 Result