Computer Vision homework 2

Pan Changxun

March 2025

1 3D Location Transformation

a) The intrinsic matrix is given by:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

When $f_x = f_y = f = 721.5mm$ and $c_x, c_y = (609.6, 172.9)mm$, the intrinsic matrix becomes:

$$K = \begin{bmatrix} 721.5 & 0 & 609.6 \\ 0 & 721.5 & 172.9 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

b) The equation of the ground plane in camera's coordinate system is $y = 1.7m$.

c) When given a 2D point (x,y) in the image and suppose it lies on the ground (y=1.7m), we can find the corresponding 3D point in the camera's coordinate system using the following equations:

$$Y = 1.7m \tag{3}$$

$$Z = \frac{Y \cdot f_y}{y - c_y} = \frac{1.7m \cdot 721.5mm}{y - 172.9mm} \tag{4}$$

$$X = Y \cdot \frac{x - c_x}{y - c_y} = 1.7m \cdot \frac{x - 609.6mm}{y - 172.9mm} \tag{5}$$

where (X, Y, Z) are the coordinates of the corresponding 3D point in the camera's coordinate system and c_x, c_y equals to (609.6, 172.9)mm.

2 Road Analyzing

a) The depth maps are stored in the folder "/src/problem2_analyzing/data/depth/" with corresponding colorbar. Here is an example of the depth map:

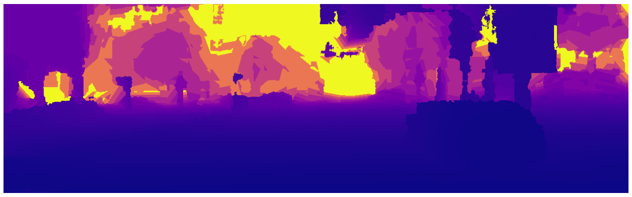


Figure 1: 004945_depth map

And here is the colorbar for the depth map:

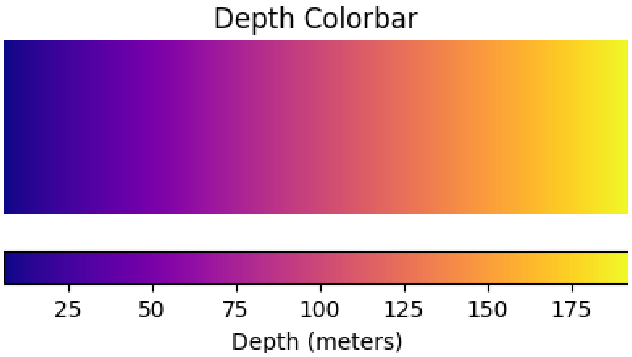


Figure 2: 004945_depth map colorbar

b) Visualize the bounding boxes in the images are stored in the folder ”/src/problem2_analyzing/data/bbox/”. Here is an example of the bounding box visualization: And the 3D coordinates of the center of the bounding box are stored in the folder ”/src/problem2_analyzing/data/3d_coordinates/”. Here are examples of the 3D coordinates of 004945:

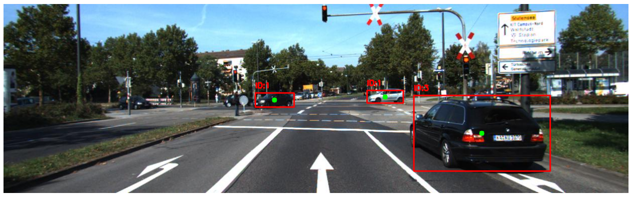


Figure 3: 004945 bounding box visualization

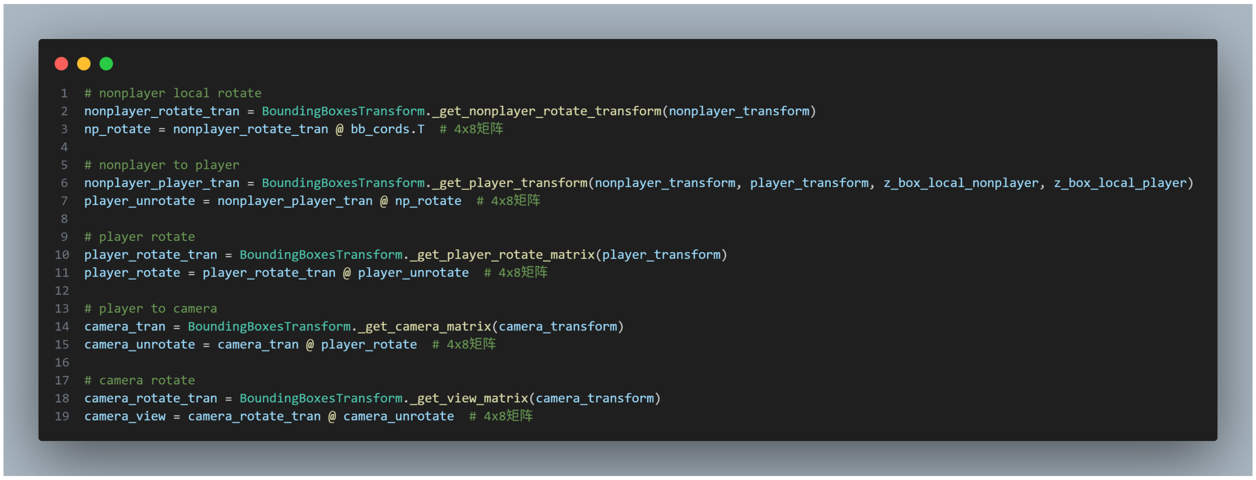
(a) ID: 3, original 2D coordinates: [950 256], 3D Coordinates: [3.2385147 0.79094404 6.863781]

(b) ID: 1, original 2D coordinates: [758 184], 3D Coordinates: [9.884517 0.7422009 48.046467]

(c) ID: 1, original 2D coordinates: [538 190], 3D Coordinates: [-3.4654994 0.8303526 34.942886]

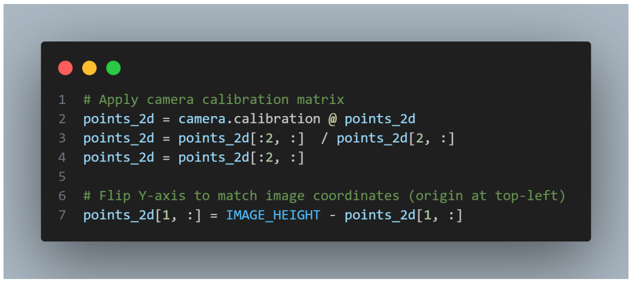
3 Self-driving Detection (Bonus)

Here are the keypoints of the self-driving detection algorithm:



```python
# nonplayer local rotate
nonplayer_rotate_tran = BoundingBoxesTransform._get_nonplayer_rotate_transform(nonplayer_transform)
np_rotate = nonplayer_rotate_tran @ bb_cords.T  # 4x8矩阵

# nonplayer to player
nonplayer_player_tran = BoundingBoxesTransform._get_player_transform(nonplayer_transform, player_transform, z_box_local_nonplayer, z_box_local_player)
player_unrotate = nonplayer_player_tran @ np_rotate  # 4x8矩阵

# player rotate
player_rotate_tran = BoundingBoxesTransform._get_player_rotate_matrix(player_transform)
player_rotate = player_rotate_tran @ player_unrotate  # 4x8矩阵

# player to camera
camera_tran = BoundingBoxesTransform._get_camera_matrix(camera_transform)
camera_unrotate = camera_tran @ player_rotate  # 4x8矩阵

# camera rotate
camera_rotate_tran = BoundingBoxesTransform._get_view_matrix(camera_transform)
camera_view = camera_rotate_tran @ camera_unrotate  # 4x8矩阵
```

Figure 4:



```python
# Apply camera calibration matrix
points_2d = camera.calibration @ points_2d
points_2d = points_2d[:2, :]  / points_2d[2, :]
points_2d = points_2d[:2, :]

# Flip Y-axis to match image coordinates (origin at top-left)
points_2d[1, :] = IMAGE_HEIGHT - points_2d[1, :]
```

Figure 5:

Here are the results of the self-driving detection algorithm: (3 figures)

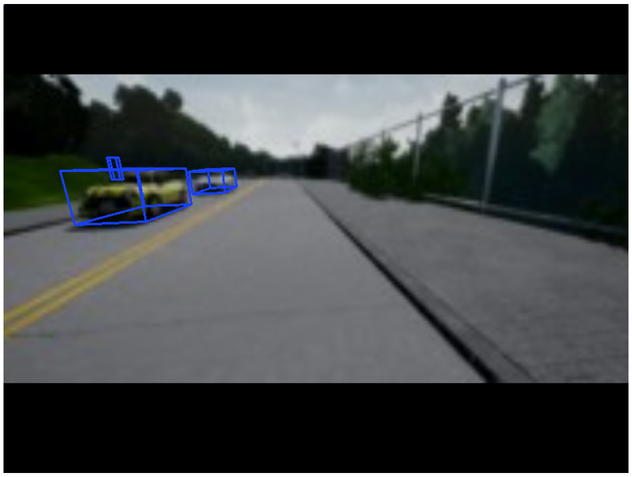


Figure 6:

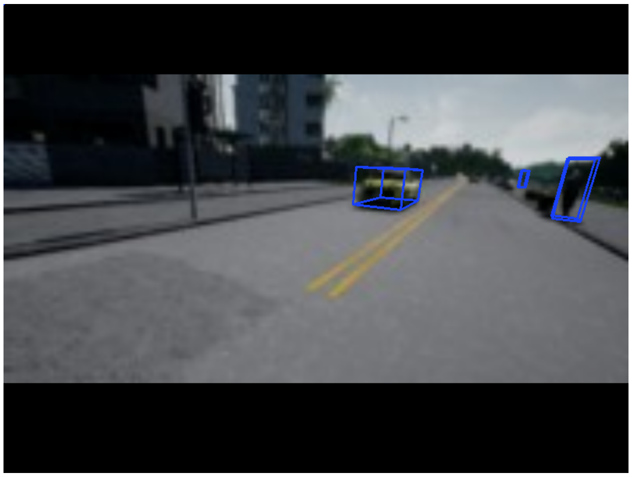


Figure 7:

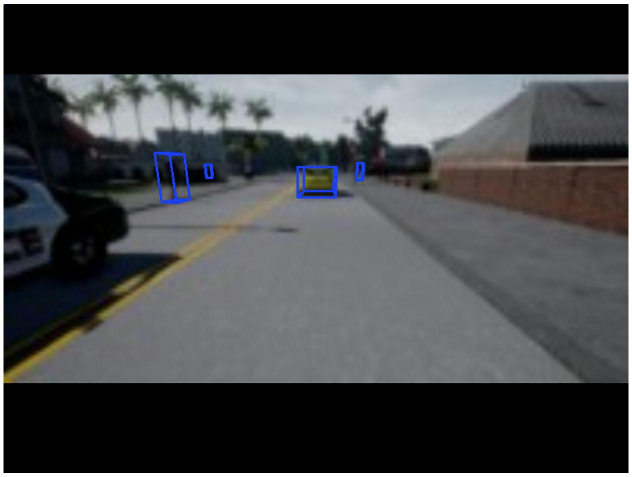


Figure 8: