3D-2D Coordinate Transforms

3D to 2D Perspective Transformation

- We can project 3D points onto 2D with a matrix multiplication

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

- Assuming that the 2D point is in homogeneous coordinates, we divide through by the last element

$$\tilde{\mathbf{x}} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} X/Z \\ Y/Z \\ 1 \end{pmatrix}$$

- Recall perspective projection (x = f X/Z, y = f Y/Z), so

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \quad x = x_1 / x_3,\ y = x_2 / x_3$$

If f=1, we sometimes call (x,y) "normalized image coordinates"

Intrinsic Camera Matrix

- We can capture all the intrinsic camera parameters in a matrix **K**

$$\mathbf{K} = \begin{pmatrix} f/s_x & 0 & c_x \\ 0 & f/s_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \quad \text{or} \quad \mathbf{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

- Recall that if f is the focal length in mm, then sx,sy is the size of a pixel in mm
- Alternatively, can just use fx,fy in pixels
- The optical center of the image is at pixel location cx, cy

- So to project 3D points in camera coordinates onto the pixel image

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \mathbf{K} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} {}^{C}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \qquad \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x_1/x_3 \\ x_2/x_3 \\ 1 \end{pmatrix}$$

Extrinsic Camera Matrix

- If 3D points are in world coordinates, we first need to transform them to camera coordinates

$$ {}^{C}\mathbf{P} = {}^{C}_{W}\mathbf{H}\,{}^{W}\mathbf{P} = \begin{pmatrix} {}^{C}_{W}\mathbf{R} & {}^{C}\mathbf{t}_{Worg} \\ \mathbf{0} & 1 \end{pmatrix} {}^{W}\mathbf{P} $$

- We can write this as an extrinsic camera matrix, that does the rotation and translation, then a projection from 3D to 2D

$$ \mathbf{M}_{ext} = \begin{pmatrix} {}^{C}_{W}\mathbf{R} & {}^{C}\mathbf{t}_{Worg} \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_X \\ r_{21} & r_{22} & r_{23} & t_Y \\ r_{31} & r_{32} & r_{33} & t_Z \end{pmatrix} $$

- Also note

$$ \mathbf{M}_{ext} = \begin{pmatrix} {}^{C}_{W}\mathbf{R} & {}^{C}\mathbf{t}_{Worg} \end{pmatrix} = \begin{pmatrix} {}^{C}_{W}\mathbf{R} & -{}^{C}_{W}\mathbf{R}\,{}^{W}\mathbf{t}_{Corg} \end{pmatrix} $$

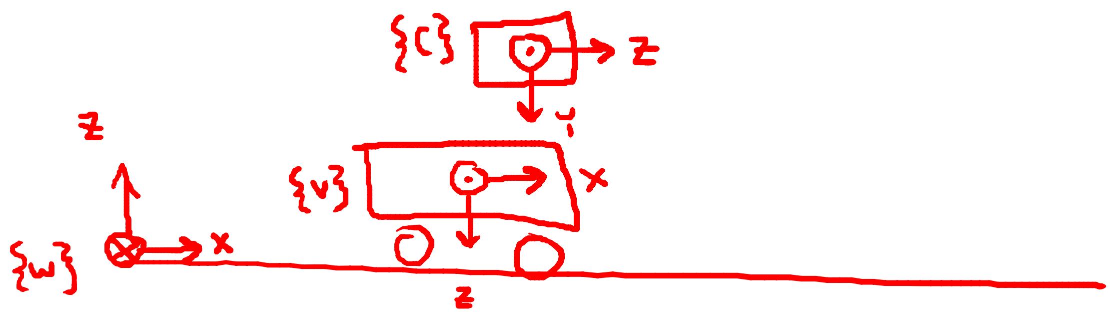
Complete Perspective Projection

- Projection of a 3D point ${}^{W}\mathbf{P}$ in the world to a point in the pixel image (x_{im},y_{im})

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \mathbf{K}\,\mathbf{M}_{ext} \; {}^{W}\!\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \qquad x_{im} = x_1 / x_3, \; y_{im} = x_2 / x_3$$

Example

- If the robot in the earlier example had a camera instead of a range sensor, what pixel would P project to?
- Assume f=512 pix, (cx,cy)=(256,256)



$${}^{W}P = \begin{bmatrix} 16 \\ 0 \\ -1 \\ 1 \end{bmatrix}$$

$${}^{W}_{V}H = \begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^{V}_{C}H = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$K = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$p = K M_{EXT} \, {}^{W}P$$

```
H_V_W = [ 1  0  0   5;
          0 -1  0   0;
          0  0 -1   1;
          0  0  0   1]

 H_S_V = [ 0  0  1   1;
           1  0  0   0;
           0  1  0  -2;
           0  0  0   1]

P_W = [ 16; 0; -1; 1];

K = [512  0   256;
      0  512  256;
      0   0    1 ];

R_C_V = [ 0  0  1;
          1  0  0;
          0  1  0];
R_V_C = R_C_V';

R_W_V = [1  0  0;
         0 -1  0;
         0  0 -1]';

R_W_C = R_V_C * R_W_V;

tCorg_V = [1; 0; -2; 1];

tCorg_W = H_V_W * tCorg_V;
tCorg_W = tCorg_W(1:3);

Mext = [ R_W_C   -R_W_C * tCorg_W ];

p = K * Mext * P_W;
p = p / p(3)
```

Weak Perspective

- Sometimes it is better to use an approximation to perspective projection, called "weak" projection or scaled orthography
- This works if the average depth Z_{avg} to an object is much larger than the variation in depth within the object
  - Instead of $x = f\, X/Z,\ y = f\, Y/Z$
  - use $x = f\, X/Z_{avg},\ y = f\, Y/Z_{avg}$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Z_{avg} \end{pmatrix} {}^{C}\!\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \qquad \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x_1 / x_3 \\ x_2 / x_3 \\ 1 \end{pmatrix}$$

- This makes the image coordinates (x,y) a linear function of the 3D coordinates (X,Y,Z)

Special Case

- Small planar patch
  - Often we want to track a small patch on an object
  - We want to know how the image of that patch transforms as the object rotates
- Assume
  - Size of patch small compared to distance -> *weak perspective*
  - Rotation is small -> *small angle approximation*
  - Patch is planar
- It can be shown that the patch undergoes affine transformation

$$\begin{pmatrix} x_B \\ y_B \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_A \\ y_A \\ 1 \end{pmatrix}$$