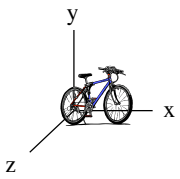
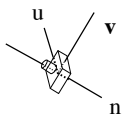
First step of geometric camera model

- Rewrite world coordinates as camera centric coordinates
  - Note that the origins are not the same and the axis are not aligned
  - Note that our rotation matrices are about an axis.
  - Hence we need to translate the world coordinates, and then rotate them.





Transform object expressed in world coords to camera coords

Step 1. Translate the camera at O_C to the world origin. Call this T_1.

Translation vector is simply negative O_C.

(We are changing the coordinate system of the world, which is the same thing mathematically as moving the camera. We want object world coordinates to **change** so that the camera location **becomes** the origin).

Transform object expressed in world coords to camera coords

Step 2. Rotate camera coordinate frame (in w.c.) so that so that **u** is **x**, **v** is **y**, and **n** is **z**. The matrix is:

$$\begin{vmatrix} \mathbf{u}^T & 0 \\ \mathbf{v}^T & 0 \\ \mathbf{n}^T & 0 \\ 0\ 0\ 0 & 1 \end{vmatrix}$$

(why?)

Transform object expressed in world coords to camera coords

$$\begin{vmatrix} \mathbf{u}^T & 0 \\ \mathbf{v}^T & 0 \\ \mathbf{n}^T & 0 \\ 0\ 0\ 0 & 1 \end{vmatrix} \mathbf{u} = \begin{vmatrix} 1 \\ 0 \\ 0 \\ 0 \end{vmatrix}$$

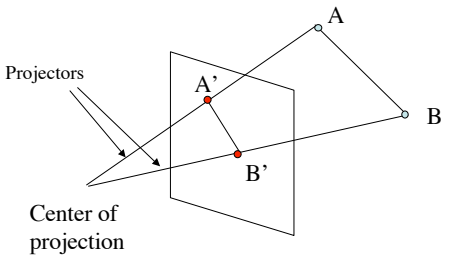
In the current coords (world shifted so that VPR is at origin): **u** maps into the X-axis unit vector (1,0,0,0) which is what we want.

(Similarly, **v**-->Y-axis unit vector, **n**-->Z-axis unit vector)

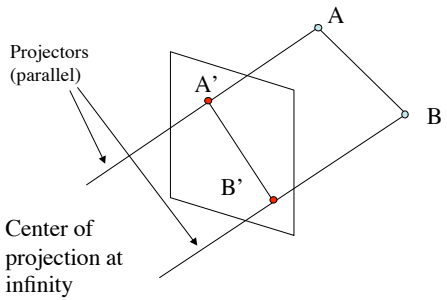
Projections

- Want to think about geometric image formation as a mathematical transformation taking points in the 3D world and mapping them into an image plane.
- Mathematical definition of a projection: PP=P
- (Doing it a second time has no effect).
- Generally rank deficient (non-invertable)--exception is P=I
- Transformation looses information (e.g., depth)
- Given a 2D image, there are many 3D worlds that could have lead to it.

Projections



Parallel Projection

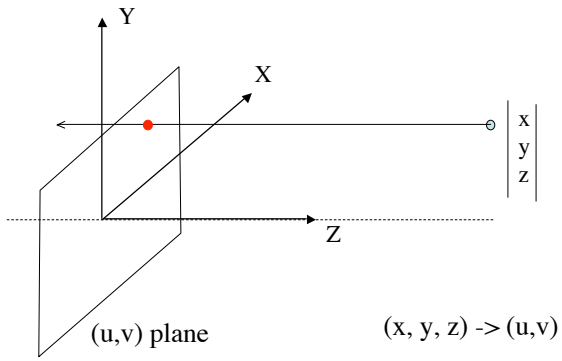


Parallel Projection

Parallel lines remain parallel, some 3D measurements can be made using 2D picture

If projection plane is perpendicular to projectors the projection is orthographic

Orthographic example (onto z=0)



(x, y, z) -> (u,v)

The equation of projection (orthographic, onto z=0)

- In homogeneous coordinates

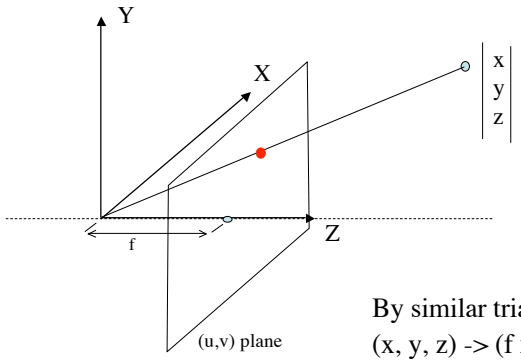
$$(x,\ y,\ z,\ 1) \Rightarrow (x,\ y,\ 1)$$

- Graphics course survivors: You might notice slight changes in style to be consistent with the book. Perhaps most notably we will explicitly, rather than implicitly, ignore the third projected coordinate, so projection matrices will be 3 by 4 not 4 by 4.

The projection matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Perspective example (onto z=f)



By similar triangles,
(x, y, z) -> (f x/z, f y/z, f)

The equation of projection

- In homogeneous coordinates

$$(x,\ y,\ z,\ 1) \Rightarrow (f\frac{x}{z},\ f\frac{y}{z},\ 1)$$

- Equivalently

$$(x,\ y,\ z,\ 1) \Rightarrow (x,\ y,\ \frac{z}{f})$$

- Homogeneous coordinates are being used to store foreshortening

- Note that using regular coordinates does **not** yield a linear transformation (inconvenient!).

The projection matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{bmatrix}$$

Camera matrix, *M*

Actual pixel coords are (u,v) = (U/W, V/W)

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} \text{Transformation} \\ \text{representing} \\ \text{intrinsic parameters} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \text{Transformation} \\ \text{representing} \\ \text{extrinsic parameters} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Projection. We use **f=1** and let the intrinsic parameters absorb the focal length.

First part makes it so that we are in standard camera coords where we know how to project.

Camera parameters (§2.2)

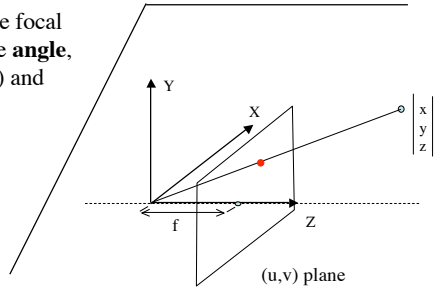
- Extrinsic parameters
  - position of the camera (3)
  - orientation of the camera (3)
- Intrinsic parameters (more natural)
  - focal length (1)
  - aspect ratio (ratio of pixel horizontal size to vertical size) (1)
  - principal point (intersection of viewing direction with camera plane) (2)
  - angle between axes of image plane---usually very close to 90 degrees (1)
- Alternative intrinsic parameters (more convenient)
  - focal length in units of horizontal pixels (1)
  - focal length in units of vertical pixels (1)
  - principal point (intersection of viewing direction with camera plane) (2)
  - angle between axes of image plane---usually very close to 90 degrees (1)

Intrinsic parameters (focal length)

Recall that $u = f * (x/z)$ and $v = f * (y/z)$

The natural, easy to measure, units for (u,v) are pixels.

This means that the focal length transfers the **angle**, as encoded in (x/z) and (y/z) into **pixels**.



Intrinsic parameters (focal length)

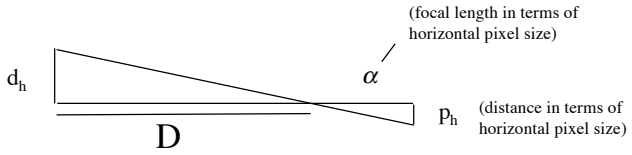
To transform a focal length in meters to pixels you would need to know the size of a pixel in meters. But you can easily measure the focal length in pixels (which is usually what you want)*.

However pixels are not always square. The ratio of width to height is called the aspect ratio.

Hence it is common to instead use two other parameters, α and β, which are the focal length in term of horizontal and vertical pixel units.

* If you have the focal length both in pixels and in meters, then you can compute the size of a pixel (if you wanted it for some reason)

Measuring focal lengths



$$\alpha = D\frac{p_h}{d_h}$$

Intrinsic parameter matrix

Conversion of projected coords into pixel units is achieved by simple **scalings** based on α and β.

Translate coords to so that line through pinhole (or center of lens) perpendicular to projection plane is at origin.

Compensate for non-perpendicular pixel axis (if needed, usually this is OK) using a **shear** transformation.

Since these operations are all achievable with matrices, we see that a camera can be modeled, as claimed, with M composed of the three parts (intrinsic, projection, extrinsic).

Camera calibration (§3)

- Want to find out:
  - what is the camera matrix? (intrinsic+extrinsic)
  - what are intrinsic parameters of the camera?
- General strategy:
  - view calibration object
  - identify image points
  - obtain camera matrix by minimizing error
  - obtain intrinsic parameters from camera matrix
- Error minimization:
  - Linear least squares
    - easy problem numerically
    - solution can be rather bad
  - More robust methods exist, including ones that don't require a calibration object

Typical setup for calibration

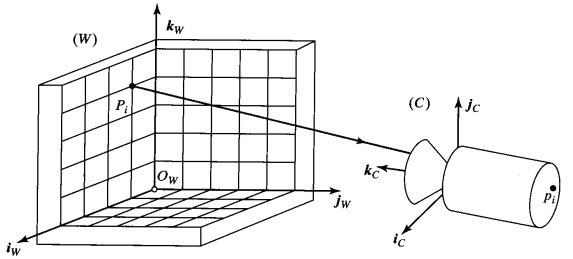


Figure 3.1 Camera calibration setup: In this example, the calibration rig is formed by three grids drawn in orthogonal planes. Other patterns could be used as well, and they may involve lines or other geometric figures.

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} \text{Transformation} \\ \text{representing} \\ \text{intrinsic parameters} \end{pmatrix} \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{matrix} \begin{pmatrix} \text{Transformation} \\ \text{representing} \\ \text{extrinsic parameters} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Camera matrix, M

Goal one: find M from image of calibration object

Goal two: given M, find the two matrices