Introduction to Graphics

Lecture 2:

Worlds in 2D and 3D

Lecture Overview

- Planar Polyhedra
- Object Representation
- Wire Frame Models
- Vectors Review (based on Dr Bradley's notes)
- Planar Projections
- Ortographic Projections
- Perspective Projection
- Vanishing Points

Three Dimensional Scenes

- Three dimensional graphics scenes can be made up of many diverse objects:

  Spheres

  Cones

  Cubes

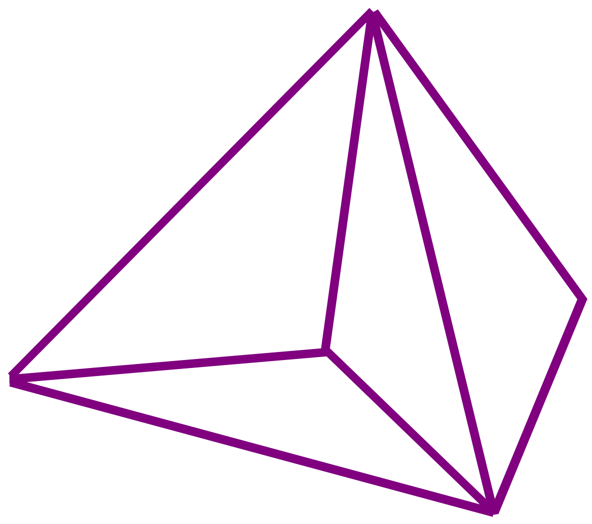
  Smooth Surfaces

  etc.

- For simplicity we will concentrate on planar polyhedra

Planar Polyhedra

These are three dimensional objects whose faces are all *planar polygons* often called *facets*.



Representing Planar Polygons

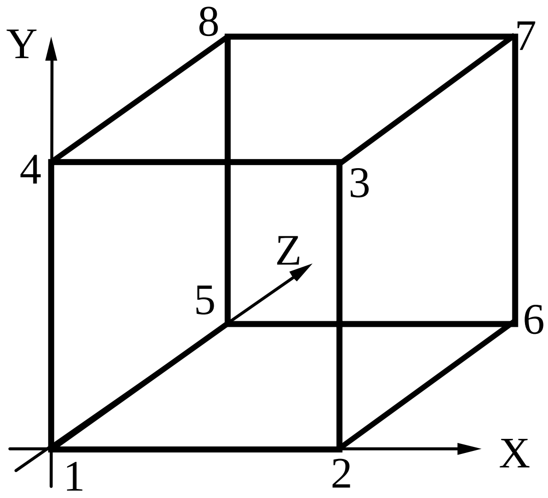
A mixture of numerical and topological data is required to represent planar polygons in a computer.

Numerical Data

Actual coordinates of vertices, etc.

Topological Data

Details of what is connected to what… **Edges/Faces**



NUMERICAL DATA	TOPOLOGICAL DATA	
Points	Lines	Faces
1. [0,0,0] 2. [2,0,0] 3. [2,2,0] 4. [0,2,0] &c	1. 1>>2 2. 1>>4 3. 1>>5 4. 3>>4 5. 3>>2 &c	1,2,4,5 1,3,7,8 &c

Diagram 2.1: Representing 3D objects

Object Representation

Static Data Structures:

The point data is stored in arrays

The topological data is stored in arrays of (point) array indices

Dynamic Data Structures:

The topological data is implied by the data structure

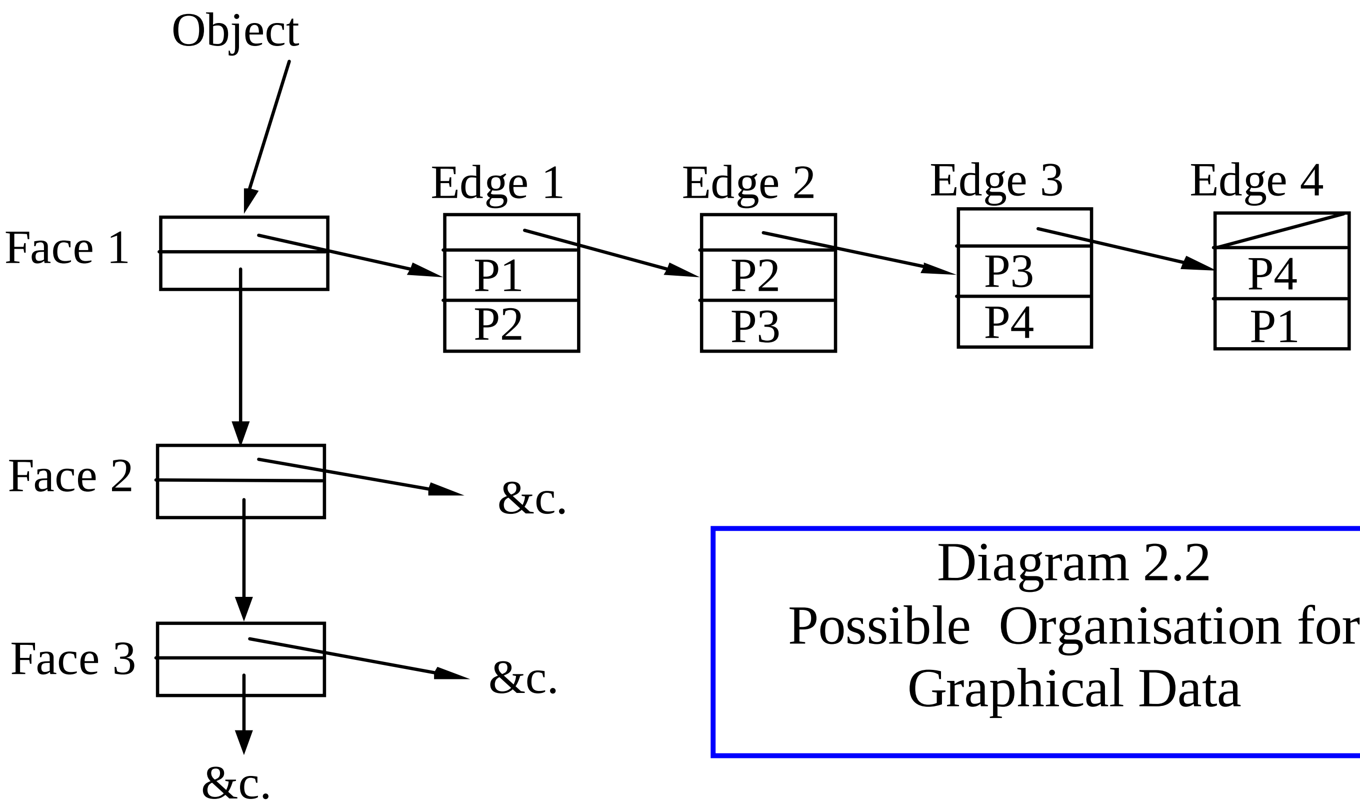


Diagram 2.2
Possible Organisation for Graphical Data

Vectors in Computer Graphics

A vector is used in Computer Graphics to represent the position coordinates for a point.

A *position vector* is simply another name for a coordinate in cartesian space. It differs from *directional vectors* in that it is always assumed to start from the origin.

Vector representation and notation

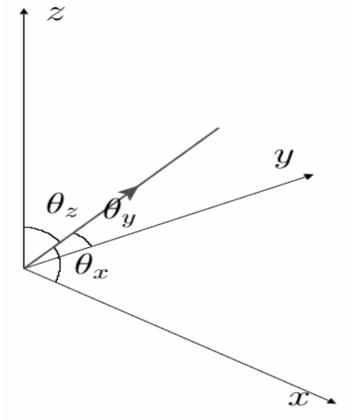
A vector conveys **both** direction and magnitude.

Row vector

$$\vec{p} = (p_1, p_2, p_3)$$

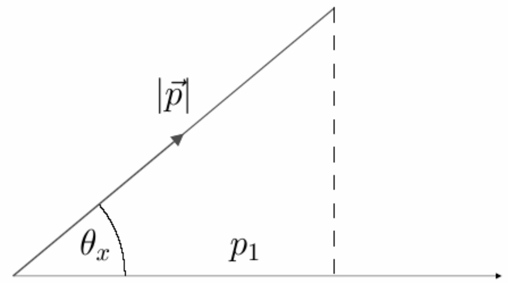
Column vector

$$\vec{p} = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}$$



$$|\vec{p}| = \sqrt{p_1^2 + p_2^2 + p_3^2}$$

Vector magnitude



$$\cos(\theta_x) = p_1 / |\vec{p}|$$

$$\cos(\theta_y) = p_2 / |\vec{p}|$$

$$\cos(\theta_z) = p_3 / |\vec{p}|$$

Vector direction

Unit Vectors

All vectors in 3D can be expressed as a weighted sum of the unit vectors $\vec{i}, \vec{j}, \vec{k}$:

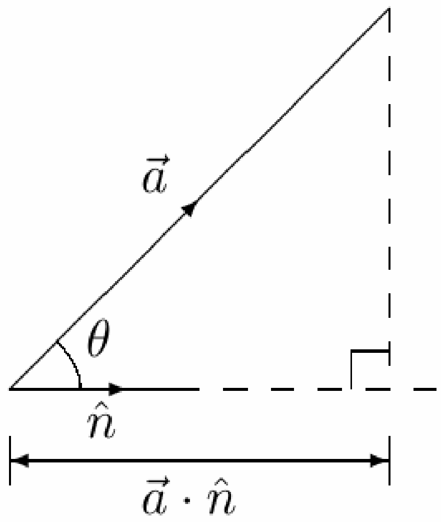
$$\vec{p} = (p_1, p_2, p_3) \equiv \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \equiv p_1\vec{i} + p_2\vec{j} + p_3\vec{k}$$

$$\left| p_1\vec{i} + p_2\vec{j} + p_3\vec{k} \right| = \sqrt{p_1^2 + p_2^2 + p_3^2}$$

Unit vectors are often used for specifying directions.

By convention, *i* = [1,0,0], *j* = [0,1,0] and *k* = [0,0,1] refer to the unit vectors in the directions of the Cartesian axes.

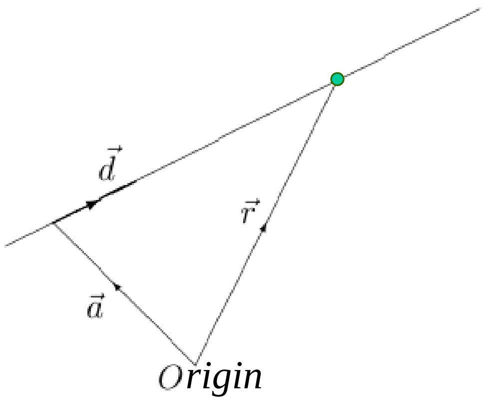
Vector Projection



$\hat{n}$ is a unit vector, i.e. $|\hat{n}| = 1$

$\vec{a} \cdot \hat{n} = |\vec{a}| \cos\theta$ represents the *amount* of $\vec{a}$ that points in the $\hat{n}$ direction

Equation of a Line



For a general point, $\vec{r}$, on the line:

$$\vec{r} = \vec{a} + \lambda\vec{d}$$

where: $\vec{a}$ is a point on the line and $\vec{d}$ is a vector parallel to the line

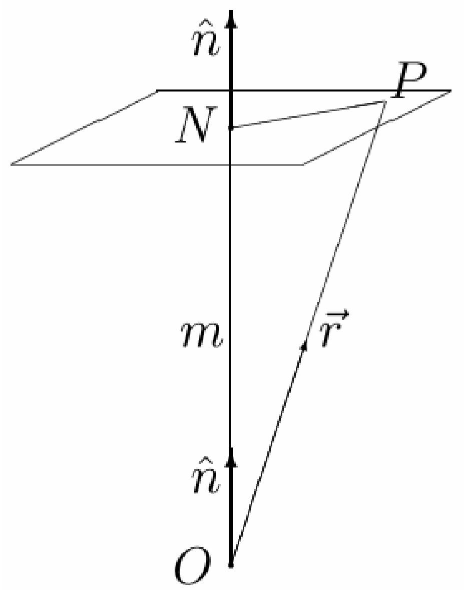
Equation of a Plane

Equation of a plane. For a general point, $\vec{r}$, in the plane, $\vec{r}$ has the property that:

$$\vec{r}.\hat{n} = m$$

where:

- $\hat{n}$ is the unit vector perpendicular to the plane
- $|m|$ is the distance from the plane to the origin (at its closest point)



Vector & Matrices Algebra

See:

`VectorAlgebraSummary.pdf`

<u>and</u>

`MatricesTutorial.pdf`

in additional material directory (~fernando/MMG/additional)

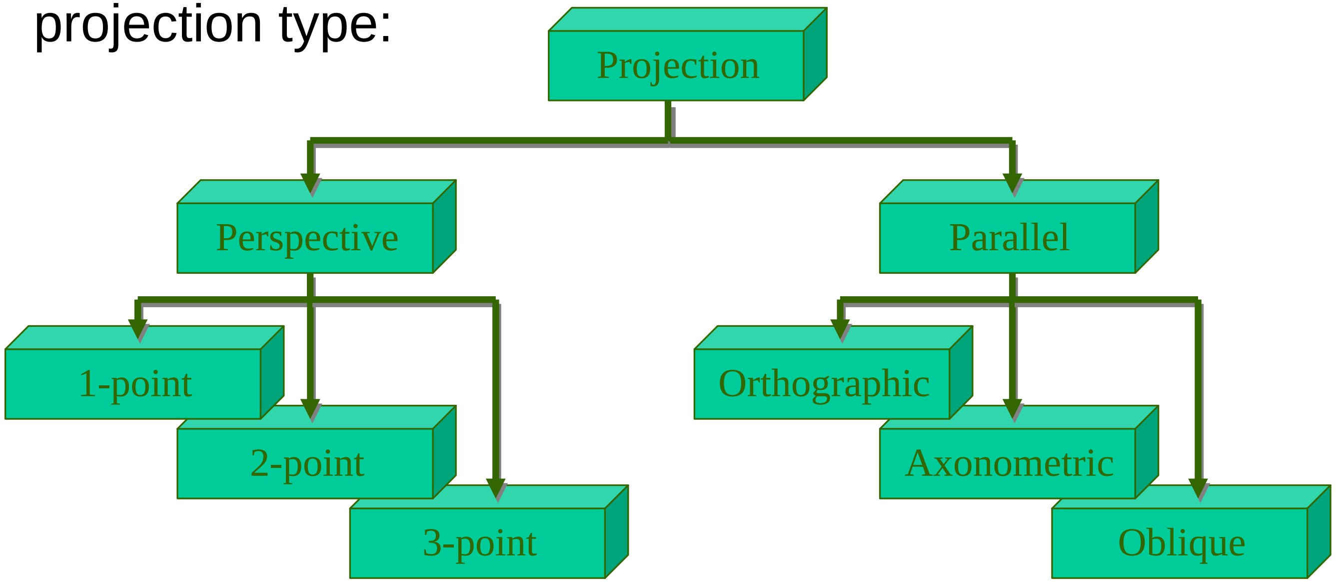
Projections of Wire Frame Models

- Wire frame models simply include points and lines (no faces).
- To draw a 3D wire frame model the **points** must first be converted to a 2D representation. Simple drawing primitives can then be used to draw them.
- The conversion from 3D into 2D is a form of **projection**.

3D → 2D Projection

Type of projection depends on a number of factors:

- *location* and *orientation* of the viewing plane (*viewport*)
- direction of projection (described by a vector)
- projection type:



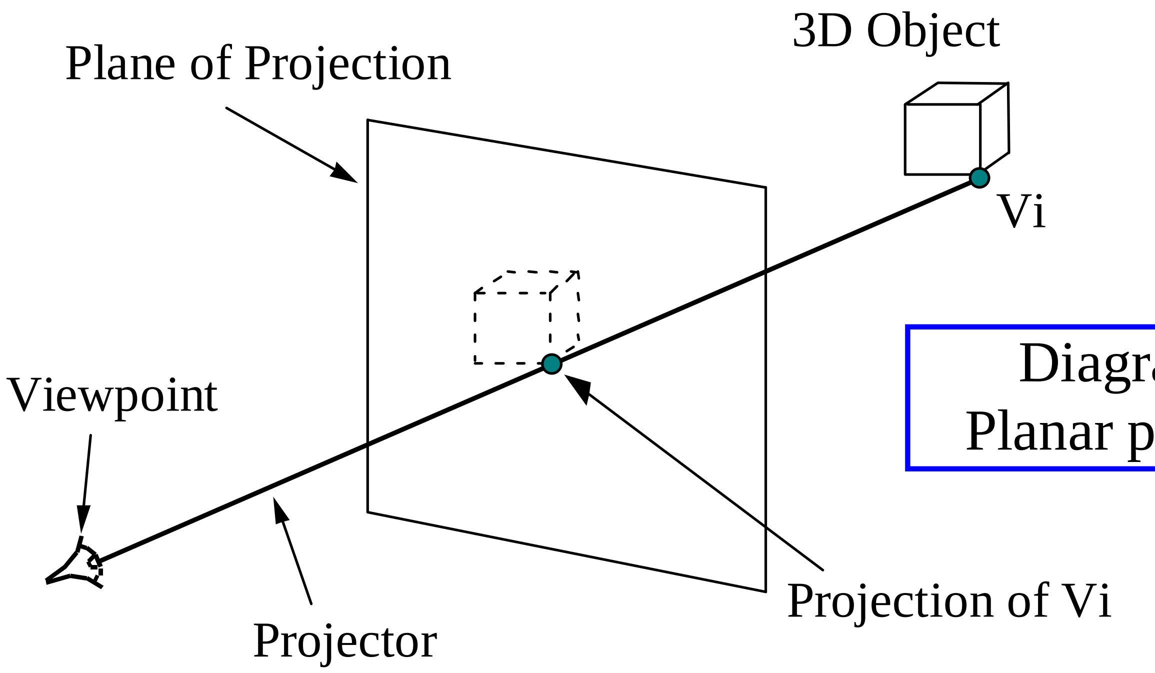
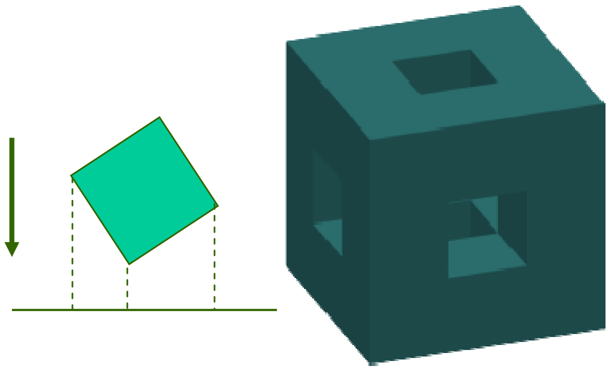


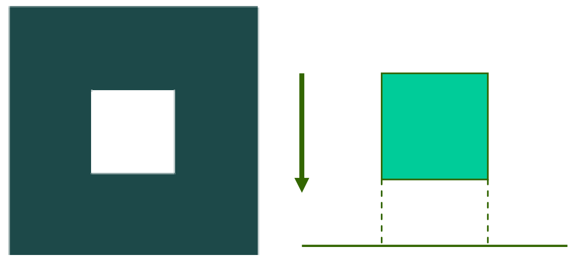
Diagram 2.3
Planar projection

Projection = Intersection of a line (Projector) with a surface (Plane of Projection)

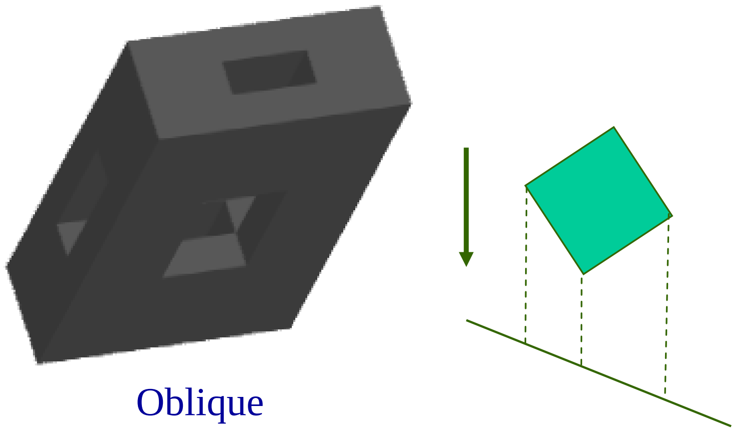
Parallel Projections



Axonometric

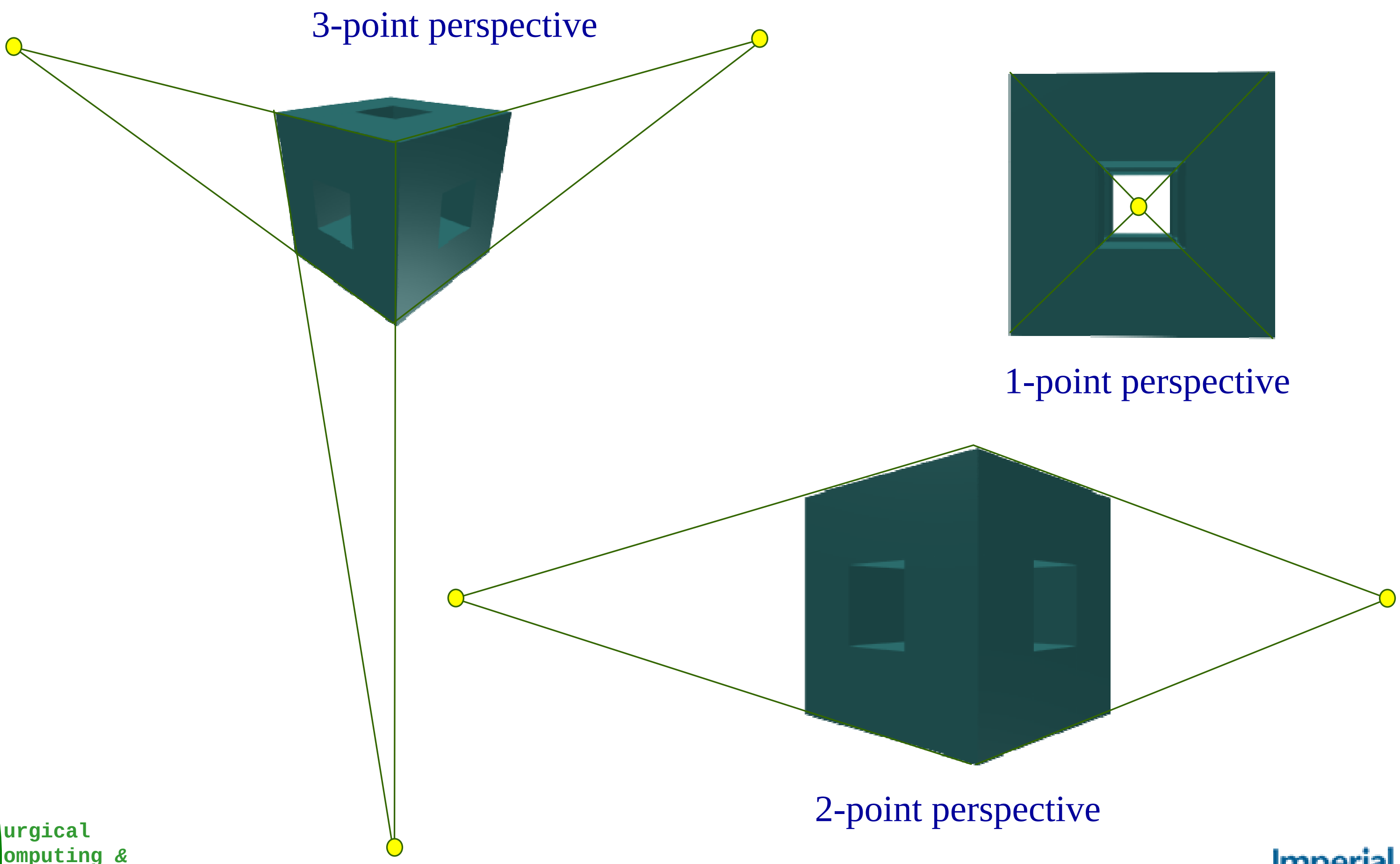


Orthographic



Oblique

Perspective Projections



Non Linear Projections

In general it is possible to project onto any surface:

- Sphere
- Cone
- etc

or to use curved projectors, for example to produce lens effects.

We will only consider planar linear projections.

Normal Orthographic Projection

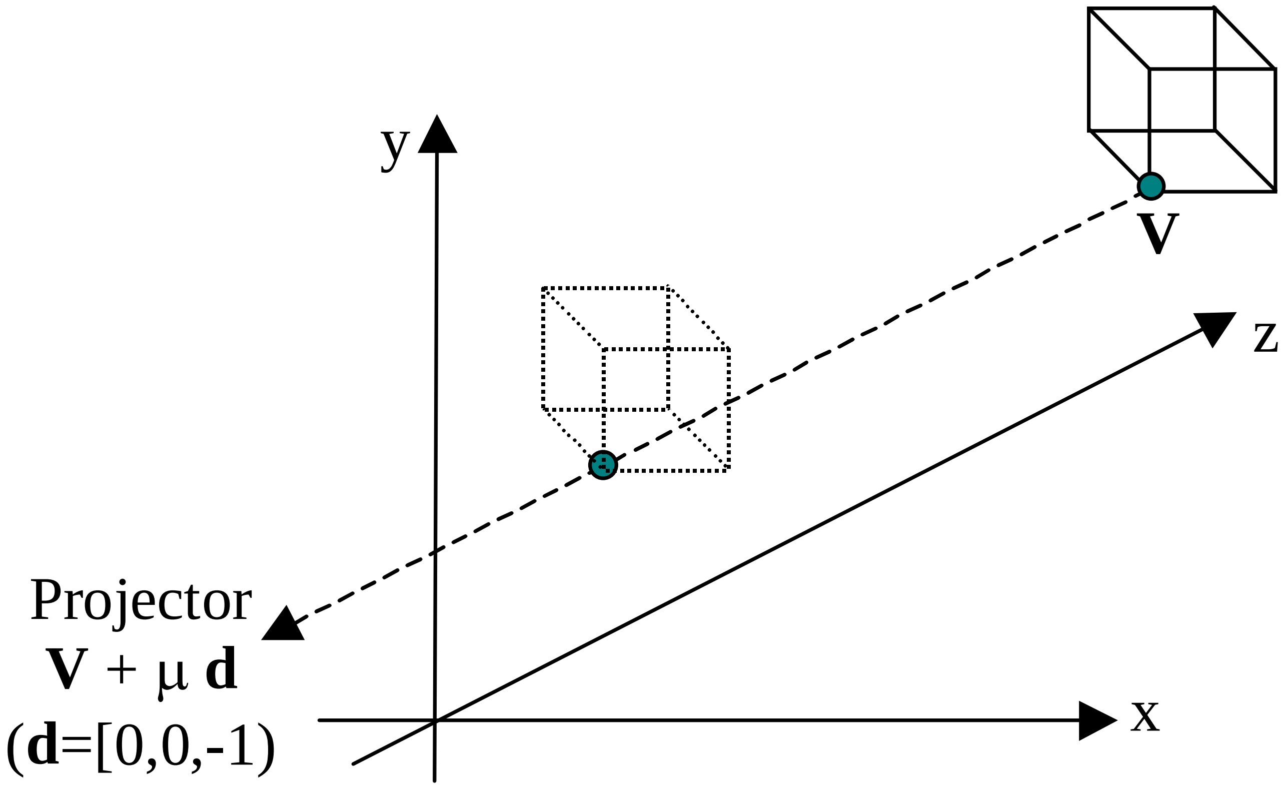
Simplest form of projection, and effective in many cases.

The viewpoint is at $z = -\infty$

The plane of projection is z=0

so

All projectors have direction d = [0,0,-1]


y
V
z
Projector
V + μ d
(d=[0,0,-1)
x

Calculating an Orthographic Projection

Projector Equation for each point P in 3D object:

$\mathbf{P} = \mathbf{V} + \mu\, \mathbf{d}$

Substitute direction of projection **d** = [0,0,-1]

Yields cartesian form

$P_x = V_x + 0 \quad P_y = V_y + 0 \quad P_z = V_z - \mu$

The projection plane is z=0 so the projected coordinate is

[Vx,Vy,0] or V2D = [Vx,Vy]

i.e. take the 3D x and y components of the vertex

See sample Orthographic Projection:

`orthoProj.wrl`

in additional material directory

(~fernando/MMG/additional)

Perspective Projection

- **Orthographic projection** is fine in cases depth is not important (i.e. most objects at same distance from viewer).

- However for depth sensitive work (e.g. computer games) it is not sufficient.

- Instead we use **Perspective Projection**

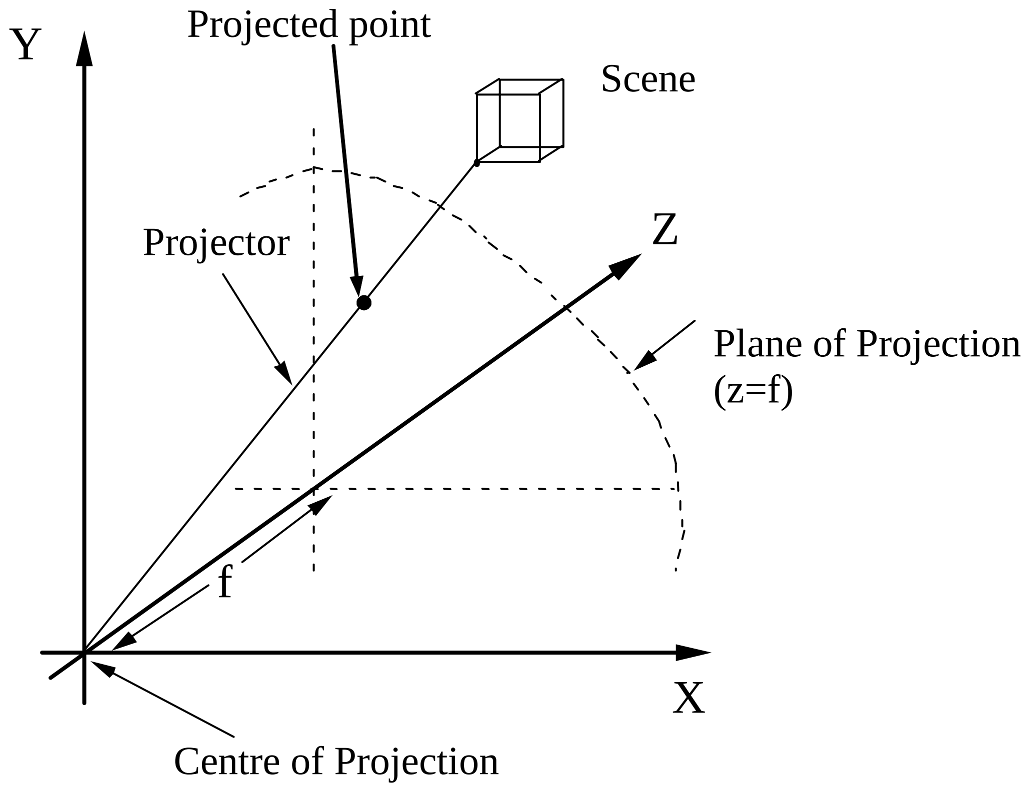


Diagram 2.5: Canonical form for Perspective projection

Calculating a Perspective Projection

Projector Equation:

$\mathbf{P} = \mu \mathbf{V}$ (all projectors go through the origin)

At the projected point Pz=f

$\mu_p = Pz/Vz = f/Vz$

$Px = \mu_p Vx$ and $Py = \mu_p Vy$

Thus

$Px = f\, Vx/Vz = \mu_p Vx$ and $Py = f\, Vy/Vz = \mu_p Vy$

μ_p is known as the **Fore-shortening Factor**

Perspective projection as similar triangles

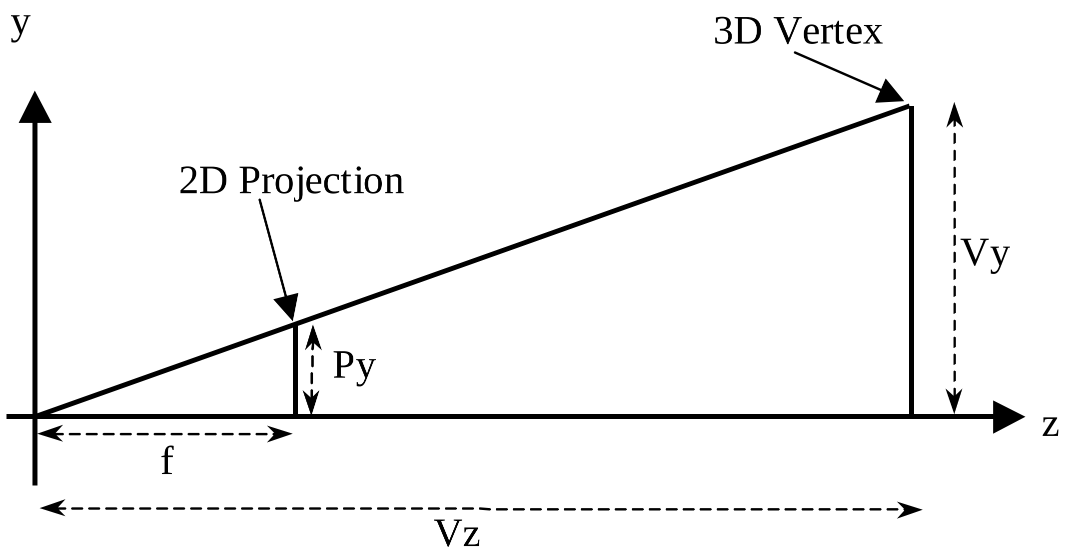


Diagram 2.6:
Perspective projection by similar triangles

See sample Perspective Projection:

`perspectiveProj.wrl`

in additional material directory

(~fernando/MMG/additional)

Characteristics of Perspective Projection

An interesting feature of perspective projection is that parallel lines are not necessarily parallel any more.

Images of parallel lines which are parallel to projection surface **WILL** remain parallel.

Others will meet at *vanishing points*.

Consider two parallel lines in 3D

$\mathbf{P}_1 = \mathbf{V_1} + \mu\ \mathbf{d}$

$\mathbf{P}_2 = \mathbf{V_2} + \mu\ \mathbf{d}$

Consider a perspective projection of these lines

Perspective projection of parallel lines

The projection of the X coordinate is:

$P_1x = f\,(V_{\mathbf{1}}x + \mu\ dx)/(V_{\mathbf{1}}z + \mu\ dz)$

now let $\mu \rightarrow \infty$

$P_1x = Ix = f\ dx/dz$ and similarly $P_1y = Iy = f\ dy/dz$

The projected point is independent of **V**.

All lines in direction **d** converge to same point in the image plane -- the Vanishing Point

Every point in plane is a VP for some set of lines

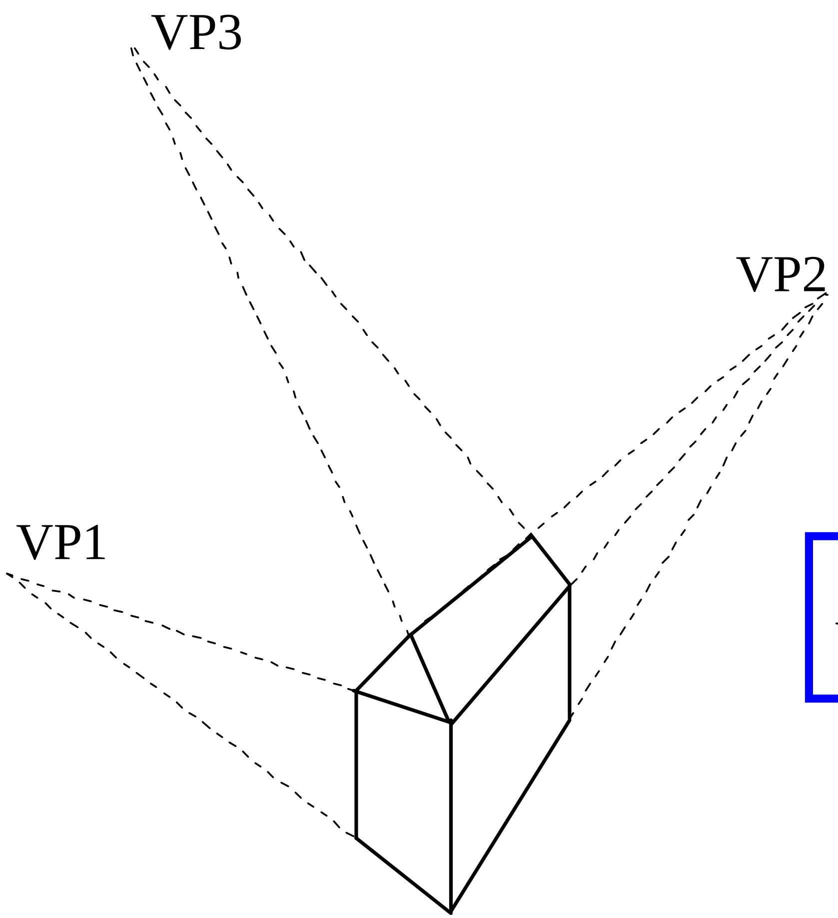


Diagram 2.7
Vanishing Points

The Vanishing Point

The vanishing point is a fixed point, which may or may not appear in the image. It is given by:

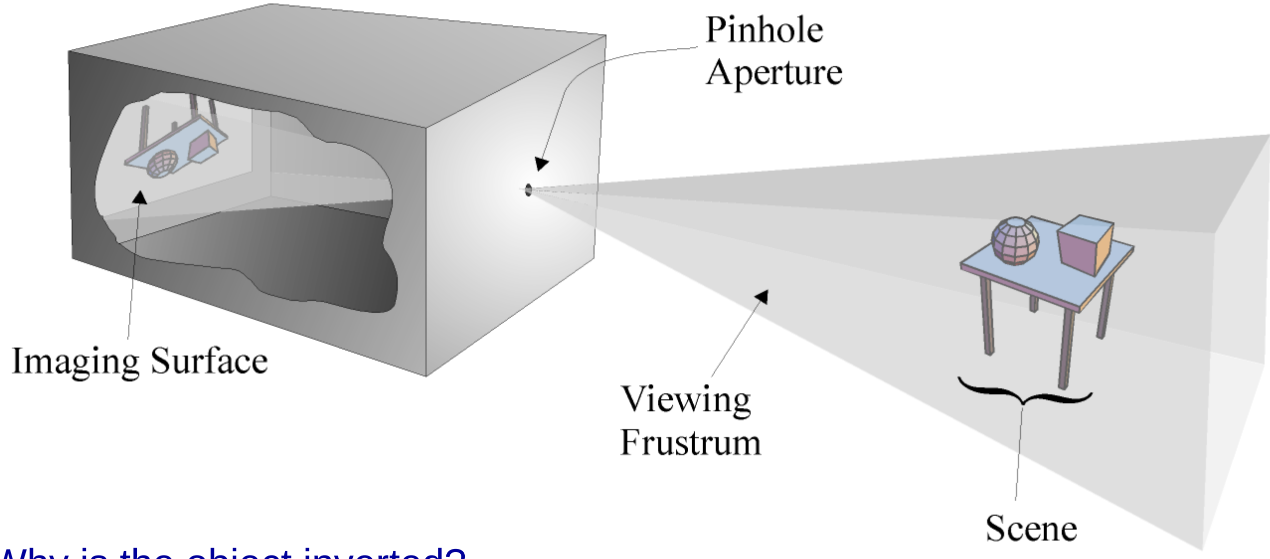
$$Ix = f \ d_x/d_z$$

and

$$Iy = f \ d_y/d_z$$

Vanishing points are used by architects to construct drawings.

Pinhole Camera Model



Why is the object inverted?

Where is the Centre of Projection?
Where is the plane of projection?