Perspective and Scene Graph

Dr. Zhigang Deng

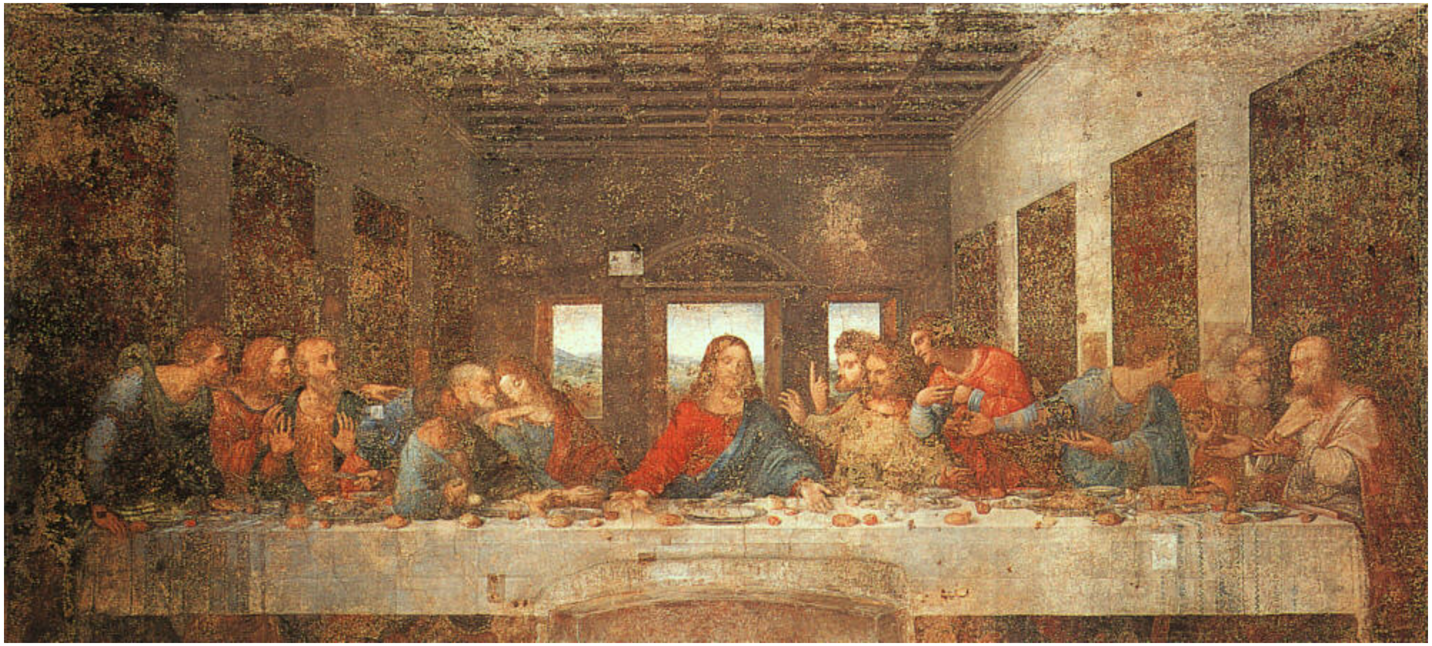
History of projection

- Ancient times: Greeks wrote about laws of perspective
- Renaissance: perspective is adopted by artists

Duccio c. 1308

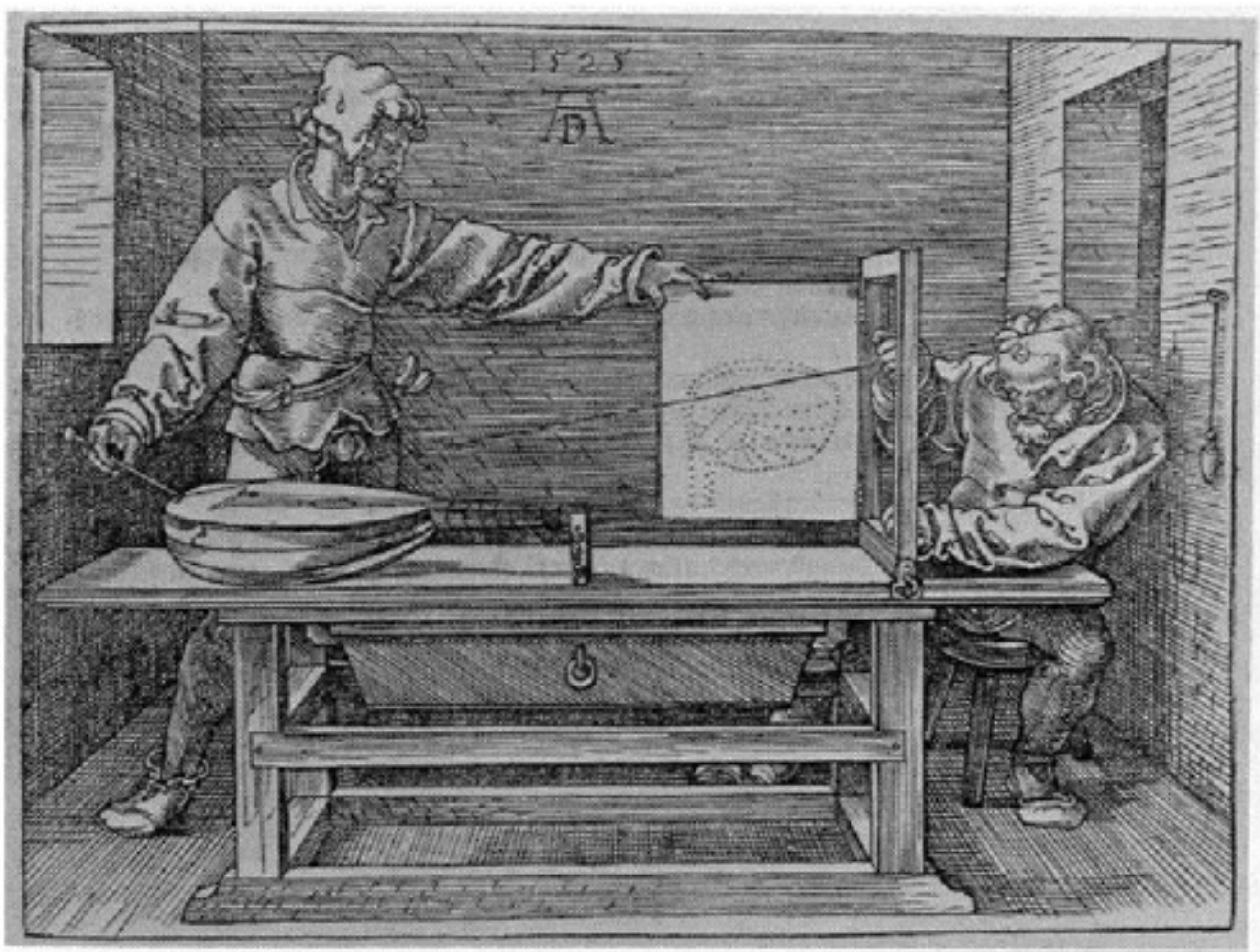
History of projection

- Later Renaissance: perspective formalized precisely



da Vinci c. 1498

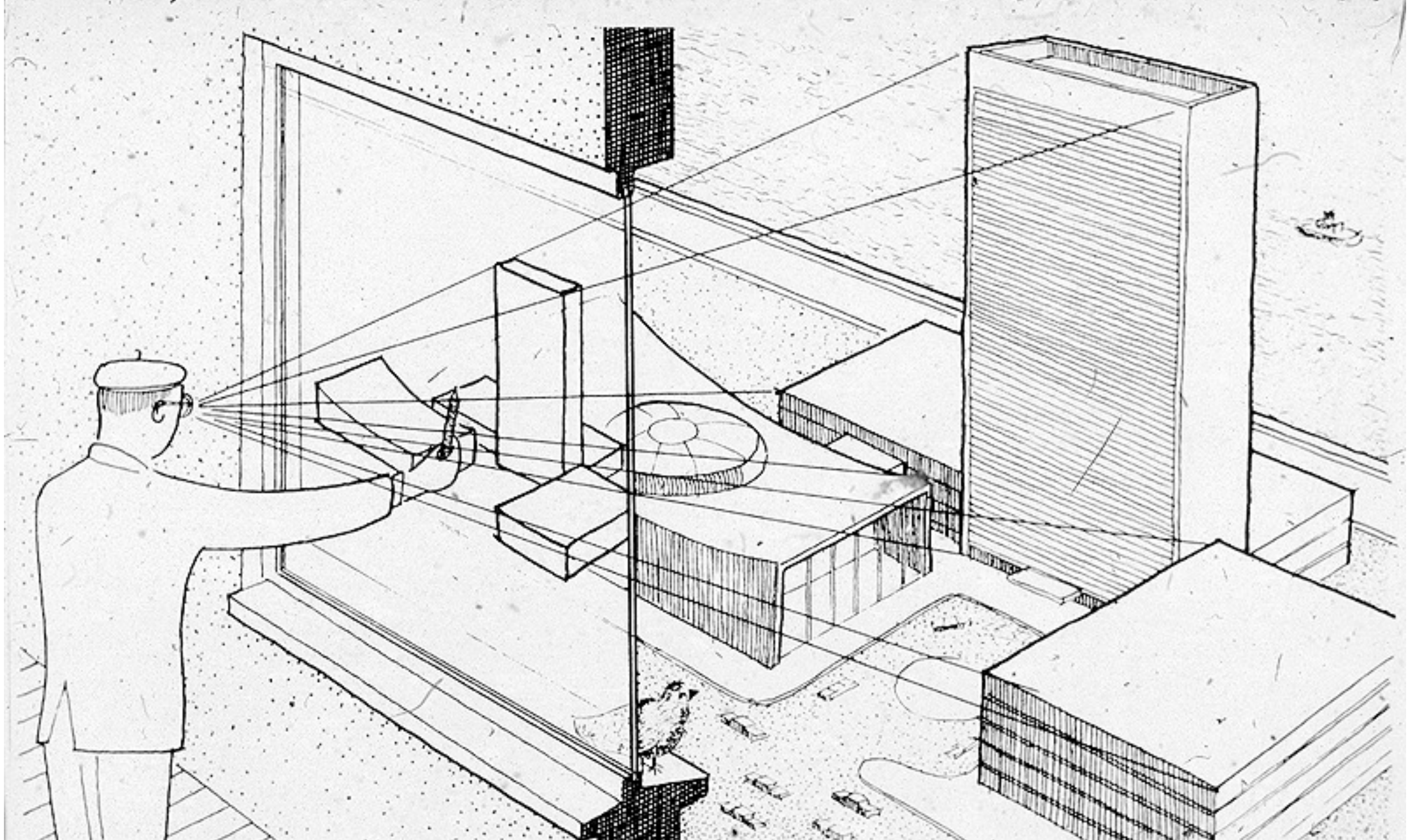
Plane projection in drawing



[Carlbom & Paciorek 78]

Plane projection in drawing

Basis Of Perspective – Lines Of Sight Through A Picture Plane [19]

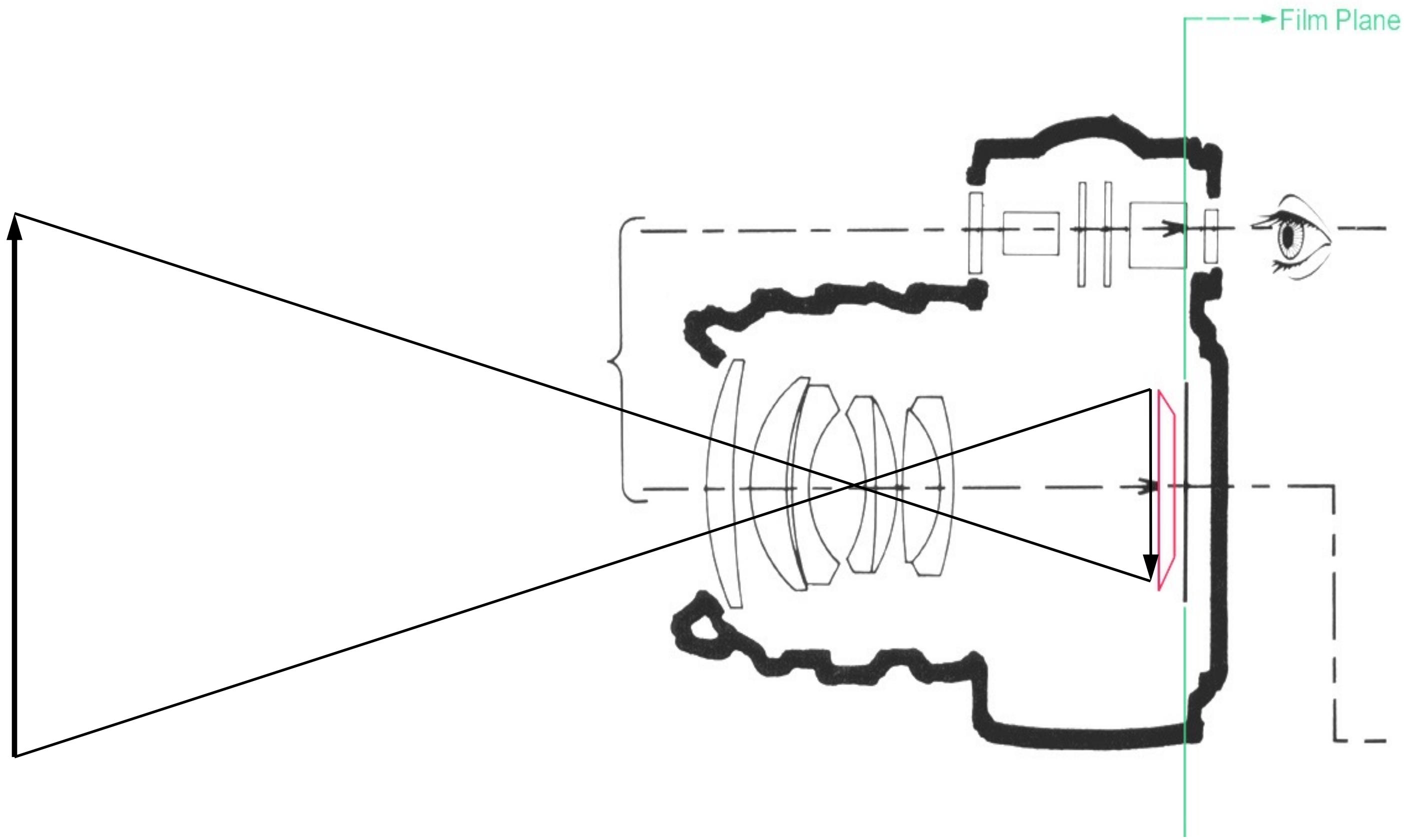


The concept of the picture plane may be better understood by looking through a window or other transparent plane from a fixed viewpoint. Your lines of sight, the multitude of straight lines leading from your eye to the subject, will all intersect this plane. Therefore, if you were to reach out with a grease pencil and draw the image of the subject on this plane you would be "tracing out" the infinite number of points of intersection of sight rays and plane. The result would be that you would have "transferred" a real three-dimensional object to a two-dimensional plane.

[CS 417 Spring 2002]

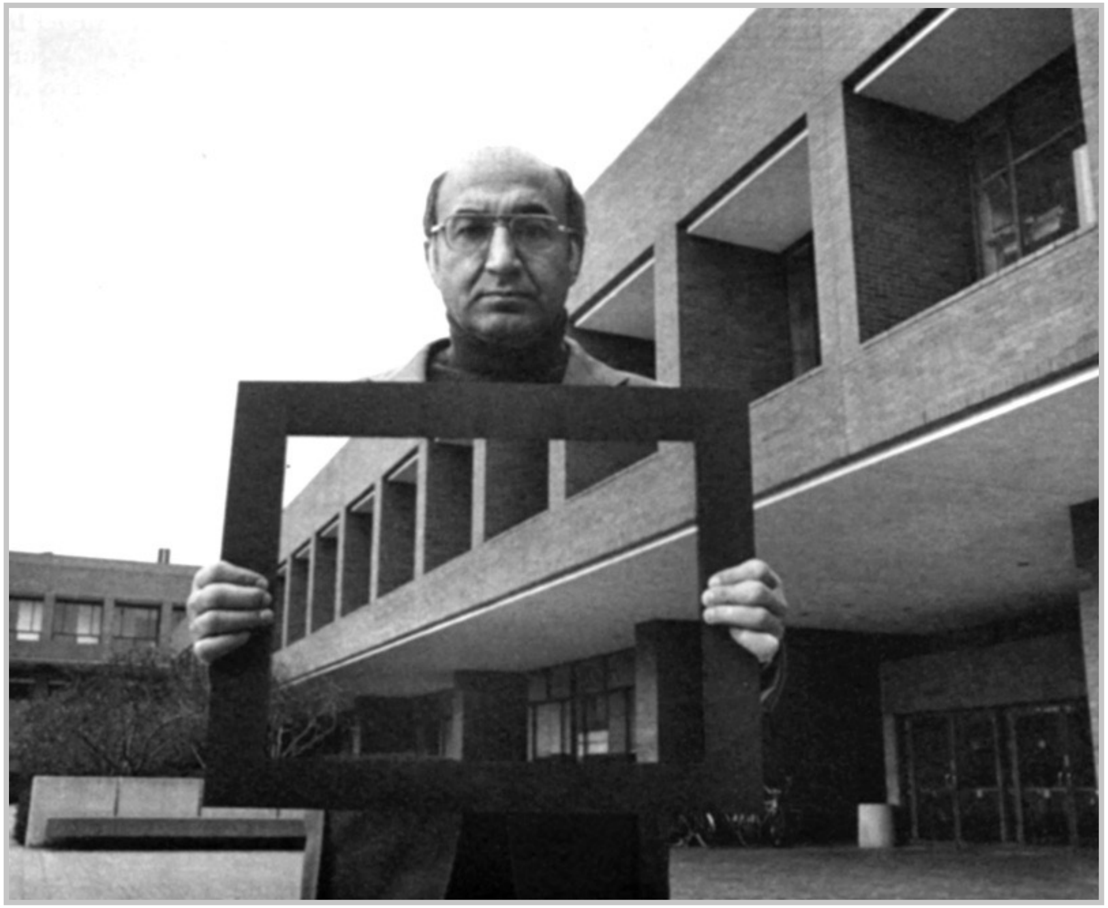
Plane projection in photography

- This is another model for what we are doing
  - applies more directly in realistic rendering



[Source unknown]

Plane projection in photography



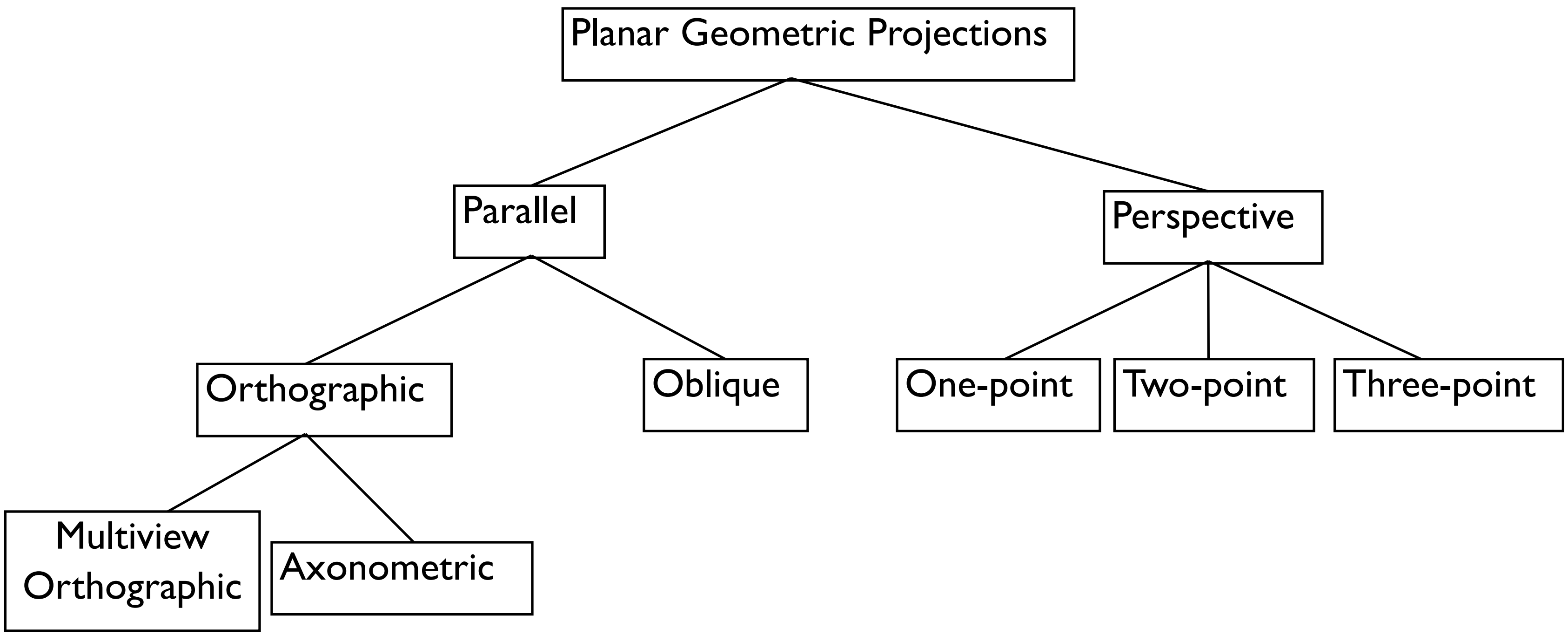
[Richard Zakia]

Ray generation vs. projection

- Viewing in ray tracing
  - start with image point
  - compute ray that projects to that point
  - do this using geometry
- Viewing by projection
  - start with 3D point
  - compute image point that it projects to
  - do this using transforms
- Inverse processes
  - ray gen. computes the preimage of projection

Classical projections

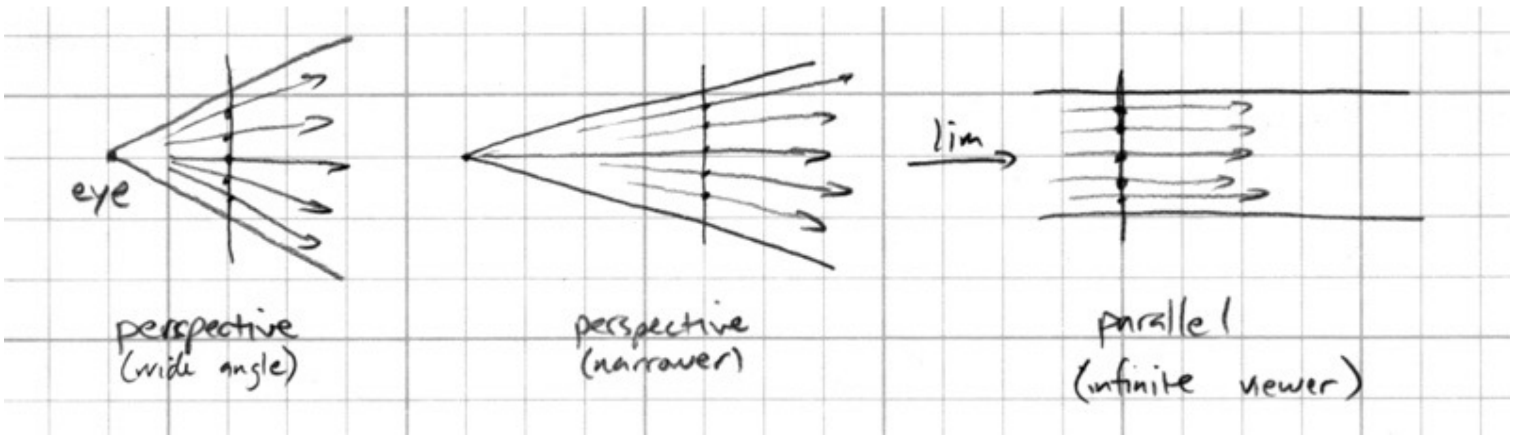
- Emphasis on cube-like objects
  - traditional in mechanical and architectural drawing



[after Carlbom & Paciorek 78]

Parallel projection

- Viewing rays are parallel rather than diverging
  - like a perspective camera that's far away



Multiview orthographic

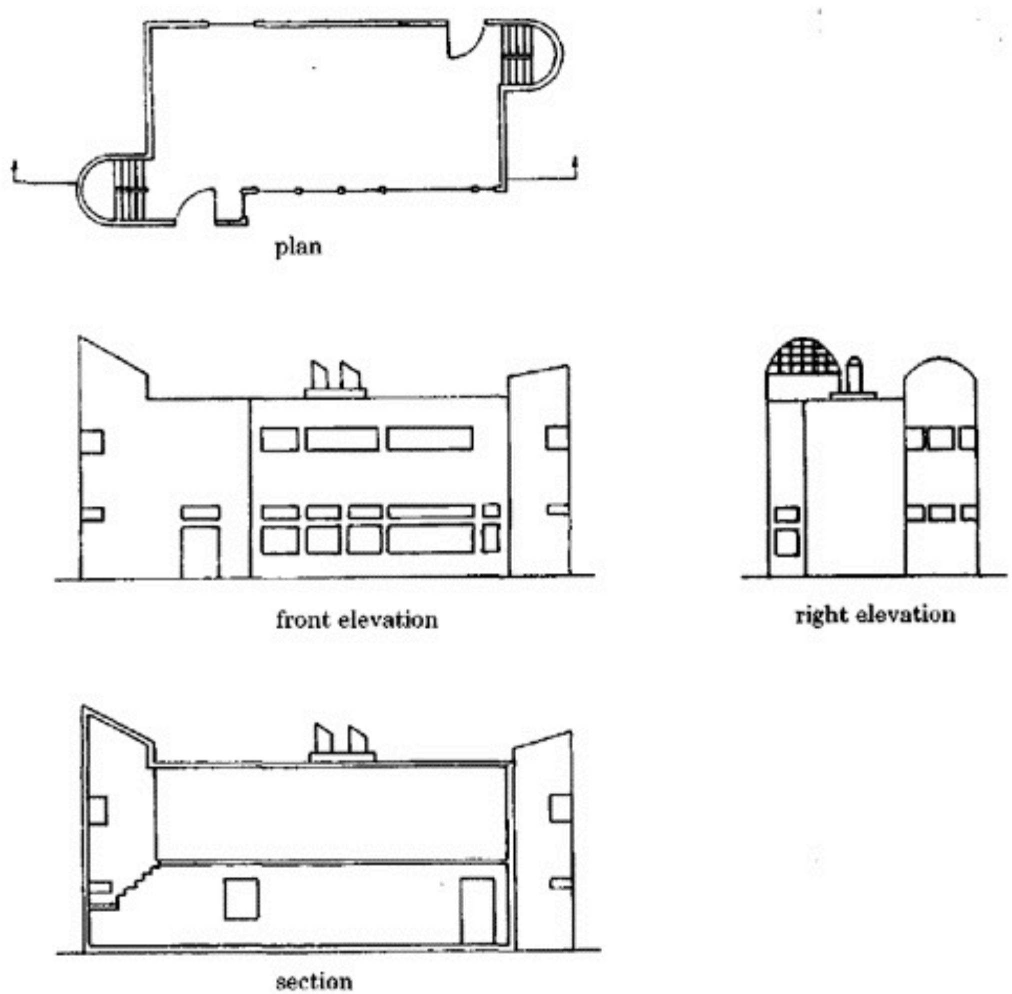
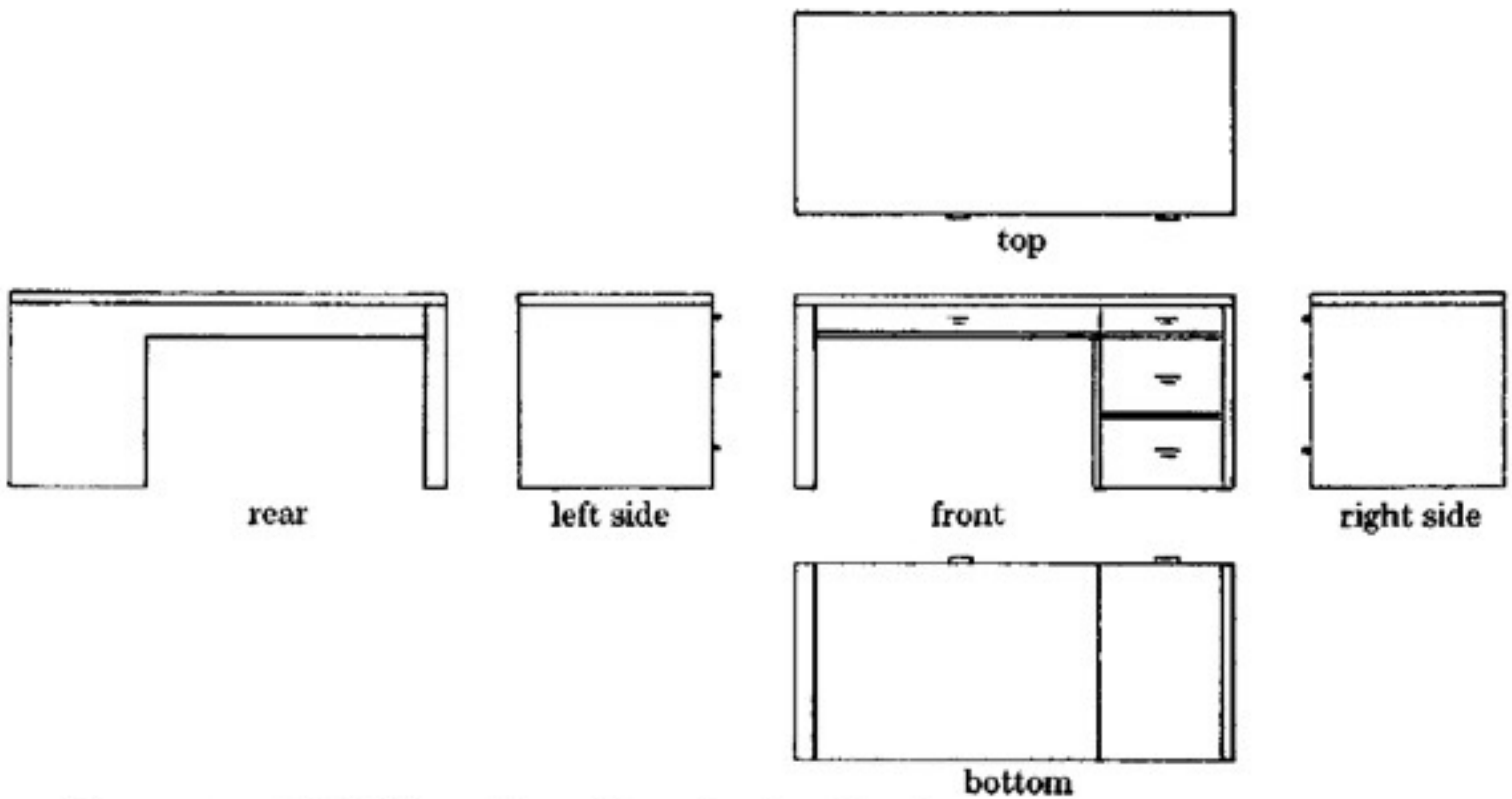


FIGURE 2-1. Multiview orthographic projection: plan, elevations, and section of a building.

[Carlbom & Paciorek 78]

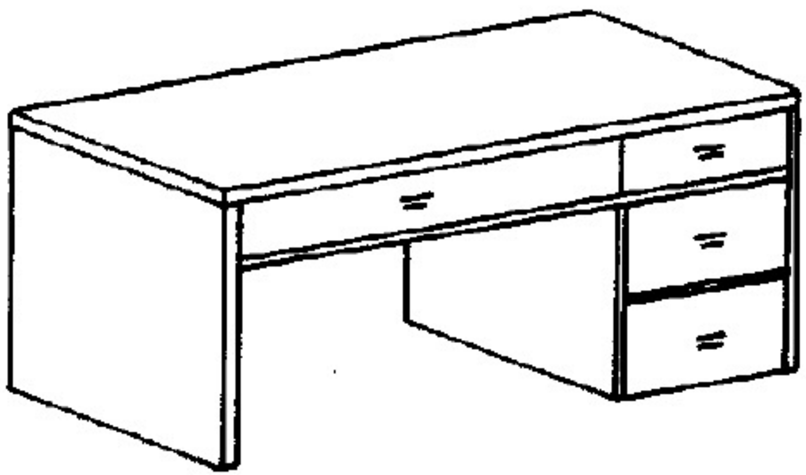
Multiview orthographic

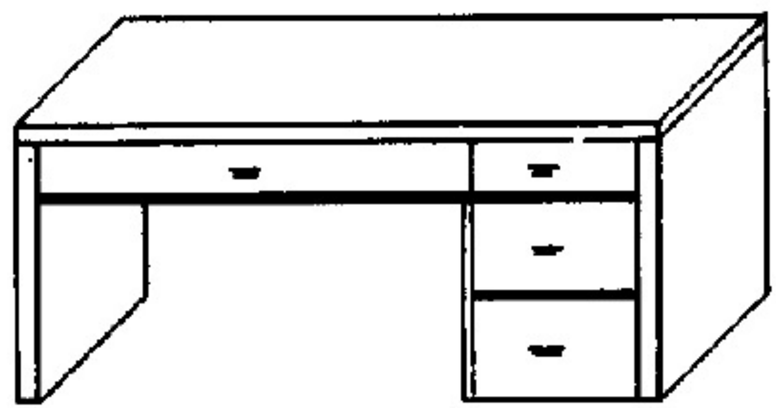


[Carlbom & Paciorek 78]

– projection plane parallel to a coordinate plane
– projection direction perpendicular to projection plane

Off-axis parallel





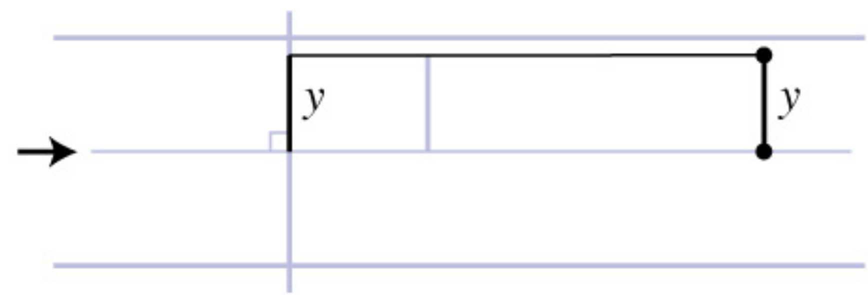
[Carlbom & Paciorek 78]

axonometric: projection plane perpendicular to projection direction but not parallel to coordinate planes

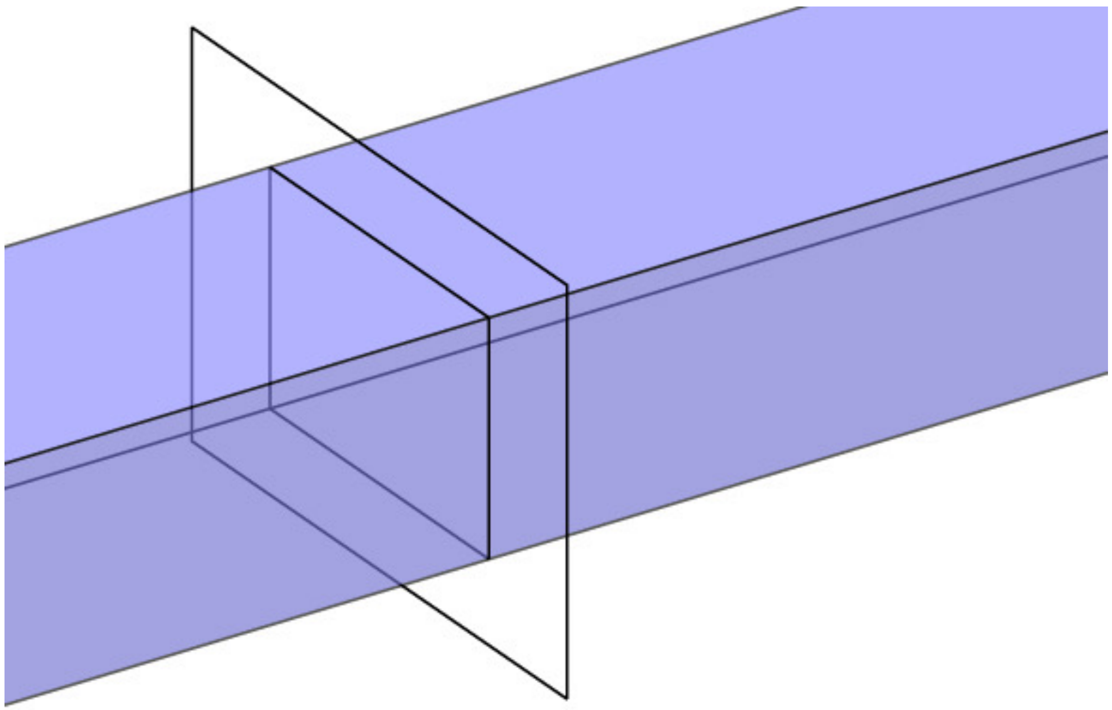
oblique: projection plane parallel to a coordinate plane but not perpendicular to projection direction.

"Orthographic" projection

- In graphics usually we lump axonometric with orthographic
  - projection plane perpendicular to projection direction
  - image height determines size of objects in image

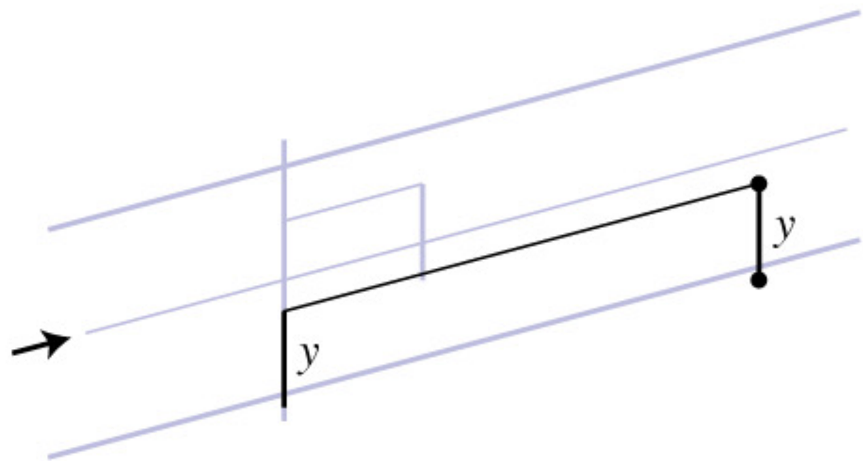


View volume: orthographic



Oblique projection

- View direction no longer coincides with projection plane normal (one more parameter)
  - objects at different distances still same size
  - objects are shifted in the image depending on their depth

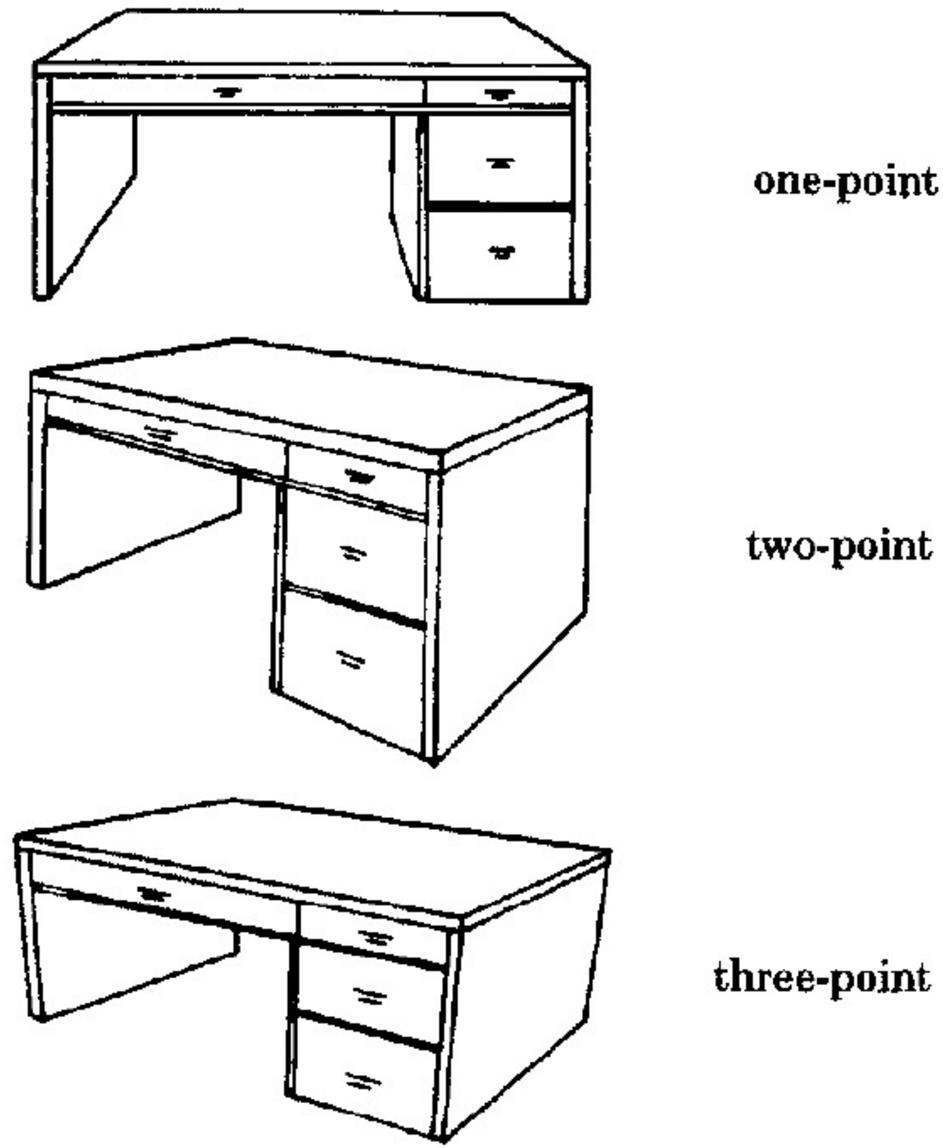


Perspective

one-point: projection plane parallel to a coordinate plane (to two coordinate axes)

two-point: projection plane parallel to one coordinate axis

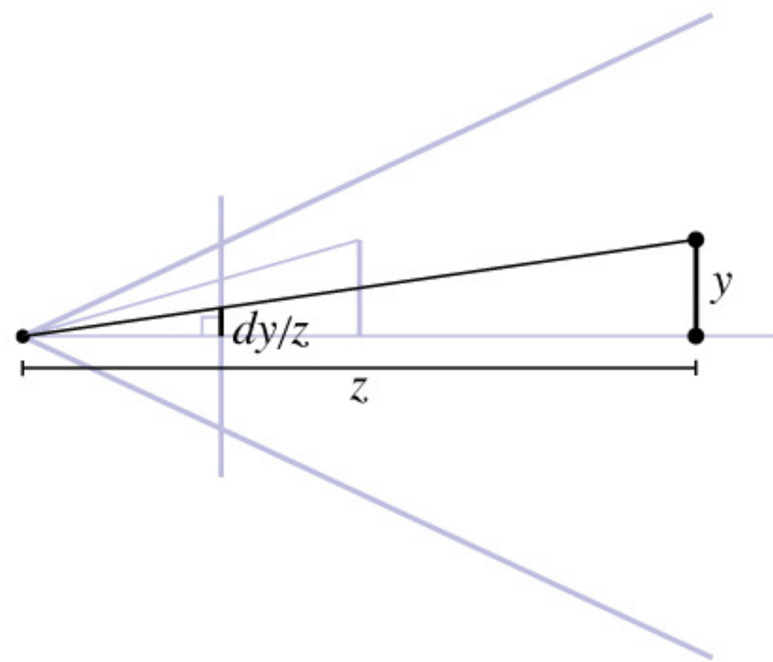
three-point: projection plane not parallel to a coordinate axis



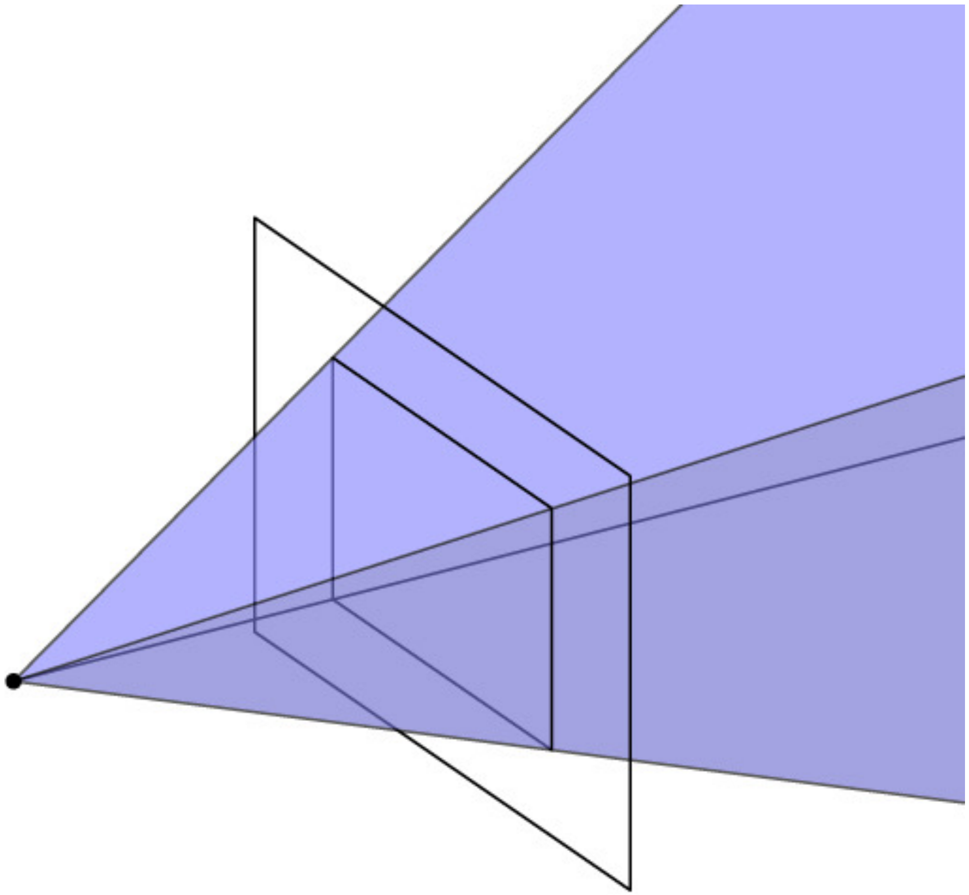
[Carlbom & Paciorek 78]

Perspective projection (normal)

- Perspective is projection by lines through a point; "normal" = plane perpendicular to view direction
  - magnification determined by:
    - image height
    - object depth
    - image plane distance
  - f.o.v. $\alpha = 2\ \mathrm{atan}(h/(2d))$
  - $y' = d\,y / z$
  - "normal" case corresponds to common types of cameras



View volume: perspective



Field of view (or f.o.v.)

- The angle between the rays corresponding to opposite edges of a perspective image
  - easy to compute only for "normal" perspective
  - have to decide to measure vert., horiz., or diag.
- In cameras, determined by focal length
  - confusing because of many image sizes
  - for 35mm format (36mm by 24mm image)
    - 18mm = 67° v.f.o.v. — super-wide angle
    - 28mm = 46° v.f.o.v. — wide angle
    - 50mm = 27° v.f.o.v. — "normal"
    - 100mm = 14° v.f.o.v. — narrow angle ("telephoto")

Field of view

- Determines "strength" of perspective effects

close viewpoint
wide angle
prominent foreshortening

far viewpoint
narrow angle
little foreshortening

[Ansel Adams]

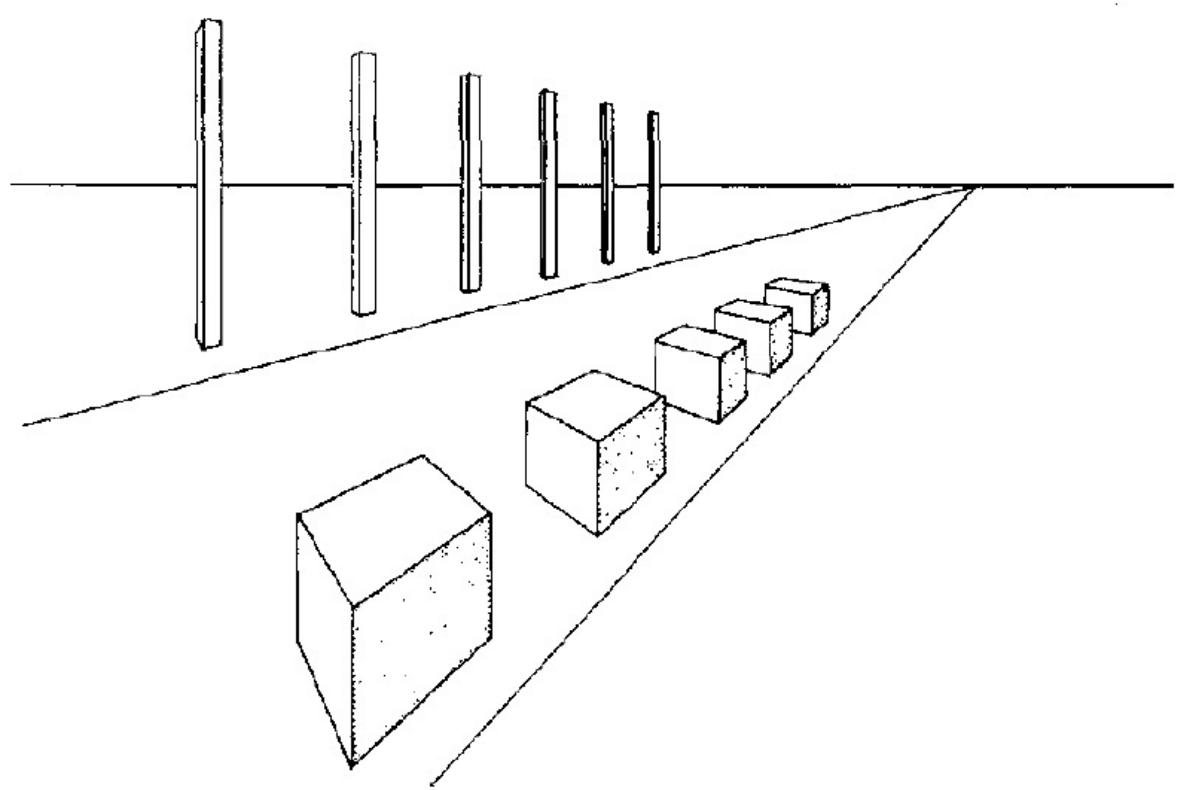
Choice of field of view

- In photography, wide angle lenses are specialty tools
  - "hard to work with"
  - easy to create weird-looking perspective effects
- In graphics, you can type in whatever f.o.v. you want
  - and people often type in big numbers!

[Ken Perlin]

Perspective distortions

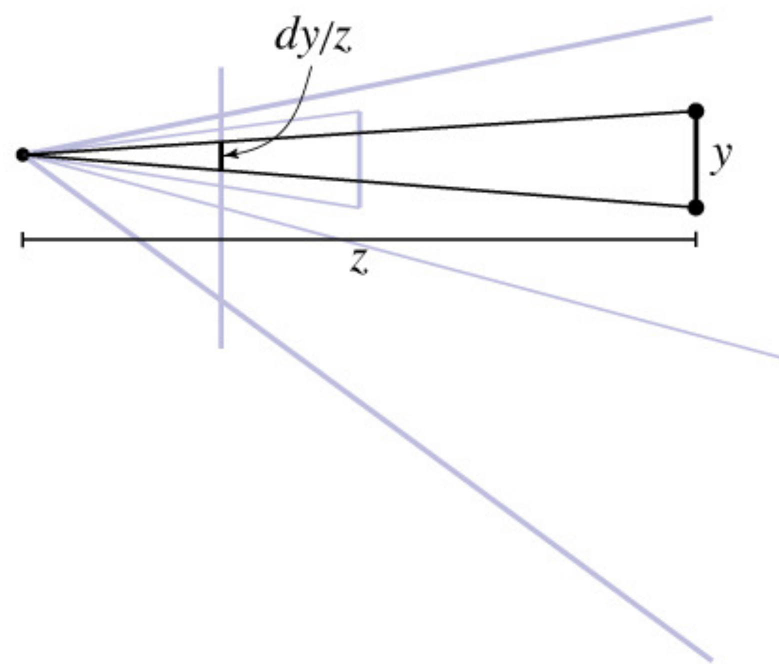
- Lengths, length ratios



[Carlbom & Paciorek 78]

Shifted perspective projection

- Perspective but with projection plane not perpendicular to view direction
  - additional parameter: projection plane normal
  - exactly equivalent to cropping out an off-center rectangle from a larger “normal” perspective
  - corresponds to *view camera* in photography



Why shifted perspective?

- Control convergence of parallel lines
- Standard example: architecture
  - buildings are taller than you, so you look up
  - top of building is farther away, so it looks smaller
- Solution: make projection plane parallel to facade
  - top of building is the same distance *from the projection plane*
- Same perspective effects can be achieved using post-processing
  - (though not the focus effects)
  - choice of *which* rays vs. arrangement of rays in image

[Philip Greenspun]

camera tilted up: converging vertical lines

[Philip Greenspun]

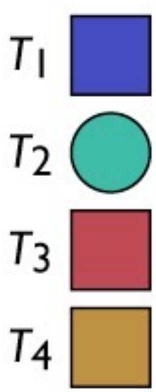
lens shifted up: parallel vertical lines

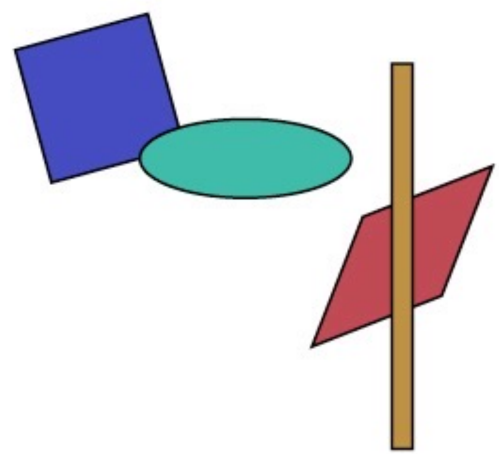
Specifying perspective projections

- Many ways to do this
  - common: from, at, up, v.f.o.v. (but not for shifted)
- One way (used in ray tracer):
  - viewpoint, view direction, up
    - establishes location and orientation of viewer
    - view direction is the direction of the center ray
  - image width, image height, projection distance
    - establishes size and location of image rectangle
  - image plane normal
    - can be different from view direction to get shifted perspective

Data structures with transforms

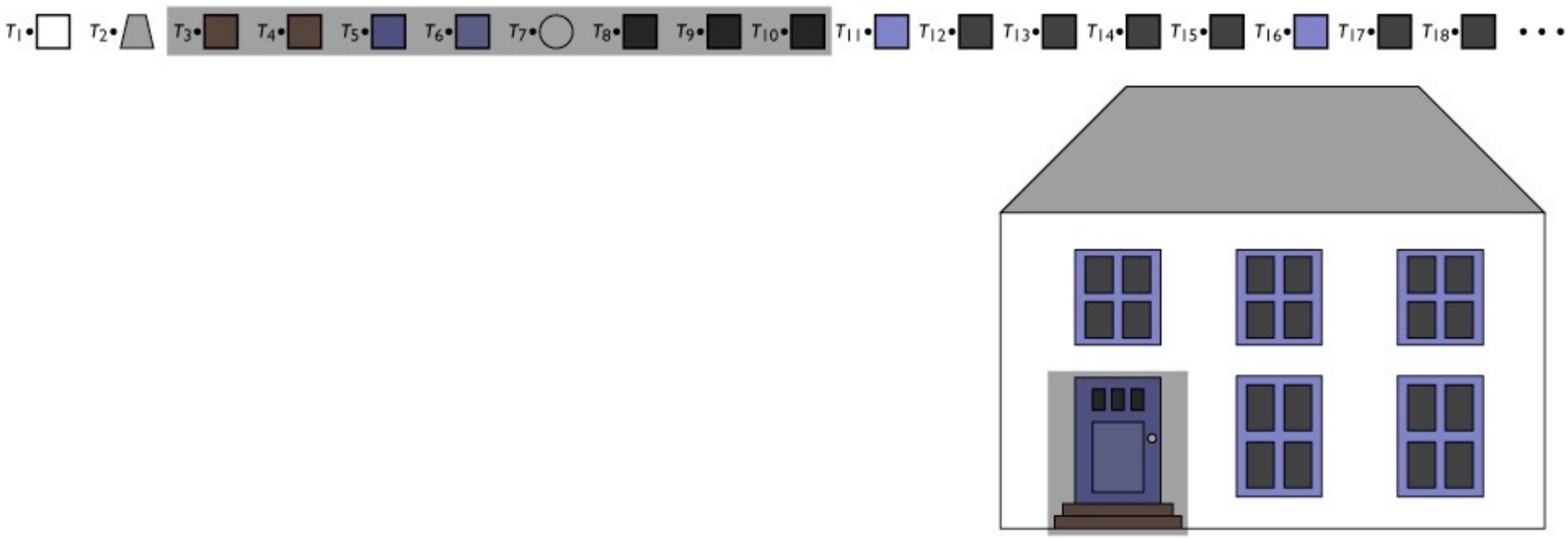
- Representing a drawing ("scene")
- List of objects
- Transform for each object
  - can use minimal primitives: ellipse is transformed circle
  - transform applies to points of object





Example

- Can represent drawing with flat list
  - but editing operations require updating many transforms

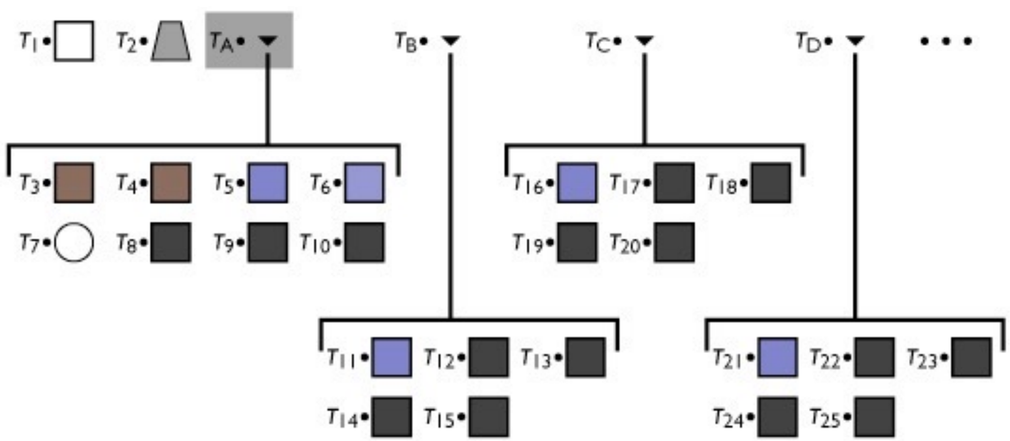


Groups of objects

- Treat a set of objects as one
- Introduce new object type: group
  - contains list of references to member objects
- This makes the model into a tree
  - interior nodes = groups
  - leaf nodes = objects
  - edges = membership of object in group

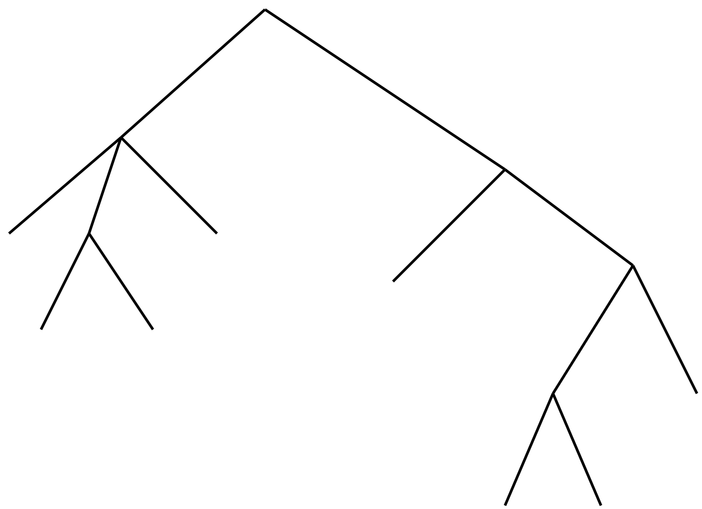
Example

- Add group as a new object type
  - lets the data structure reflect the drawing structure
  - enables high-level editing by changing just one node



The Scene Graph (tree)

- A name given to various kinds of graph structures (nodes connected together) used to represent scenes
- Simplest form: tree
  - just saw this
  - every node has one parent
  - leaf nodes are identified with objects in the scene



Concatenation and hierarchy

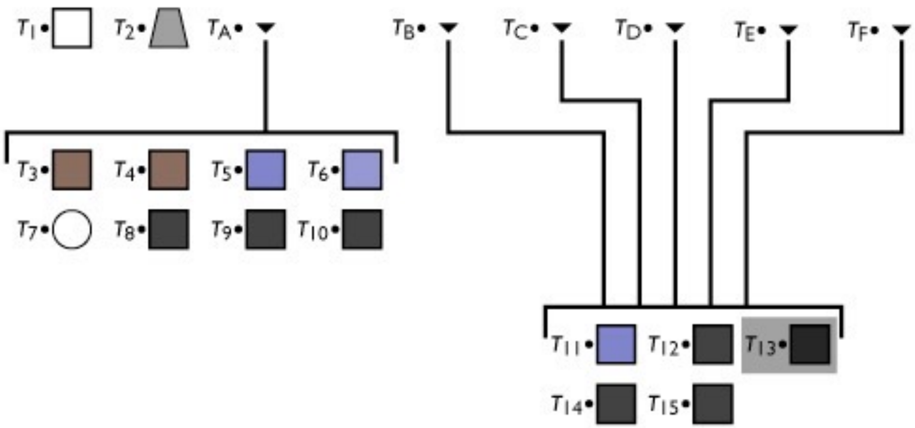
- Transforms associated with nodes or edges
- Each transform applies to all geometry below it
  - want group transform to transform each member
  - members already transformed—concatenate
- Frame transform for object is product of all matrices along path from root
  - each object's transform describes relationship between its local coordinates and its group's coordinates
  - frame-to-canonical transform is the result of repeatedly changing coordinates from group to containing group

Instances

- Simple idea: allow an object to be a member of more than one group at once
  - transform different in each case
  - leads to linked copies
  - single editing operation changes all instances

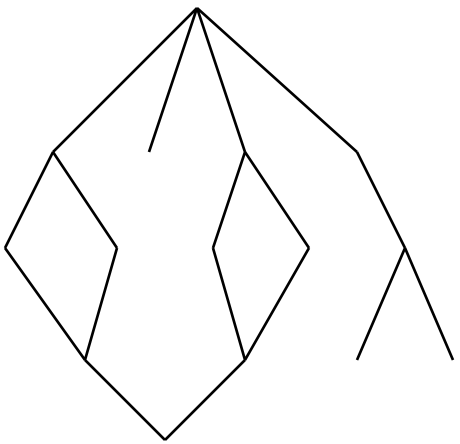
Example

- Allow multiple references to nodes
  - reflects more of drawing structure
  - allows editing of repeated parts in one operation



The Scene Graph (with instances)

- With instances, there is no more tree
  - an object that is instanced multiple times has more than one parent
- Transform tree becomes DAG
  - directed acyclic graph
  - group is not allowed to contain itself, even indirectly
- Transforms still accumulate along path from root
  - now *paths* from root to leaves are identified with scene objects



Implementing a hierarchy

- Object-oriented language is convenient
  - define shapes and groups as derived from single class

```
abstract class Shape {
   void draw();
}

class Square extends Shape {
   void draw() {
      // draw unit square
   }
}

class Circle extends Shape {
   void draw() {
      // draw unit circle
   }
}
```

Implementing traversal

- Pass a transform down the hierarchy
  - before drawing, concatenate

```
abstract class Shape {
   void draw(Transform t_c);
}

class Square extends Shape {
   void draw(Transform t_c) {
      // draw t_c * unit square
   }
}

class Circle extends Shape {
   void draw(Transform t_c) {
      // draw t_c * unit circle
   }
}
```

```
class Group extends Shape {
   Transform t;
   ShapeList members;
   void draw(Transform t_c) {
      for (m in members) {
         m.draw(t_c * t);
      }
   }
}
```

Basic Scene Graph operations

- Editing a transformation
  - good to present usable UI
- Getting transform of object in canonical (world) frame
  - traverse path from root to leaf
- Grouping and ungrouping
  - can do these operations without moving anything
  - group: insert identity node
  - ungroup: remove node, push transform to children
- Reparenting
  - move node from one parent to another
  - can do without altering position

Adding more than geometry

- Objects have properties besides shape
  - color, shading parameters
  - approximation parameters (e.g. precision of subdividing curved surfaces into triangles)
  - behavior in response to user input
  - …
- Setting properties for entire groups is useful
  - paint entire window green
- Many systems include some kind of property nodes
  - in traversal they are read as, e.g., “set current color”

Scene Graph variations

- Where transforms go
  - in every node
  - on edges
  - in group nodes only
  - in special Transform nodes
- Tree vs. DAG
- Nodes for cameras and lights?