3D Geometry

17 February 2009
CMPT370
Dr. Sean Ho
Trinity Western University

OpenGL state machine

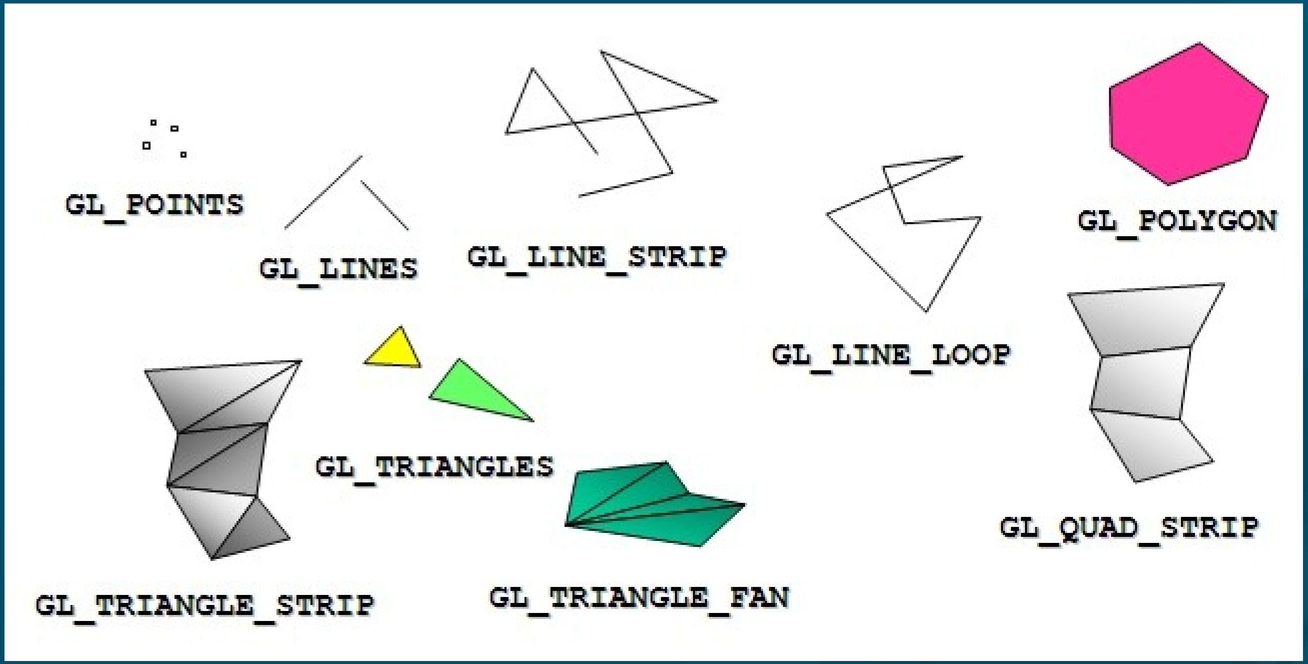
- Two kinds of OpenGL functions
  - Generate primitives
    - Vertex, line, triangle, polygon, etc.
  - Change state
    - Current colour
    - Material properties (shininess, etc.)
    - Transformations: view, model
- Each primitive gets whatever state was current when it was drawn

OpenGL functions

- Most core OpenGL functions look like this:
  - glVertex3f(x, y, z)
    - gl: belongs to core OpenGL library (glu for GLU)
    - Vertex: name of function
    - 3f: argument type: 3 floats
- Not overloaded, for efficiency
  - glVertex3fv(vec)
    - takes a pointer to an array of 3 floats
  - glVertex3i(x, y, z): ints
  - glVertex3d(x, y, z): doubles

OpenGL primitives

- glBegin(GL_*) starts a set of primitives



- Polygons must be simple: edges cannot cross
- Must be convex
- Must be flat: all vertices in the same plane

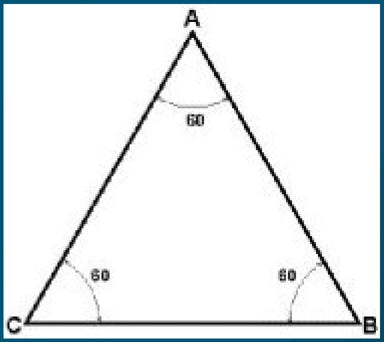
Drawing in OpenGL (cf CubeView)

- Start the set of primitives:
  - glBegin(GL_TRIANGLES);
- Set the colour and other attributes:
  - glColor3f(0.0, 0.0, 1.0);
- Create the vertices:
  - glVertex3f(0.0, 0.1, 0.2);
  - glVertex3f(0.1, 0.0, 0.2);
  - glVertex3f(0.0, 0.0, 0.2);
- End the set of primitives:
  - glEnd();

Projection matrix

- The coordinates of glVertex are in world coords
  - OpenGL converts to camera coords, then to screen coords
- The projection matrix specifies the camera:
  - glMatrixMode(GL_PROJECTION);
- Specify the viewing volume:
  - glLoadIdentity();
  - glOrtho(left, right, bottom, top, near, far)
  - Orthographic (parallel) projection

Coordinate-free geometry



- Cartesian geometry:
  - Points are locations in space (x,y,z)
  - Tied to a particular coordinate system
- Euclidean (coordinate-free) geometry:
  - Points exist regardless of the coordinate system
  - e.g.: two triangles are identical if all three legs are same length
    - Regardless of where in space the triangle is

Scalars, vectors, and points

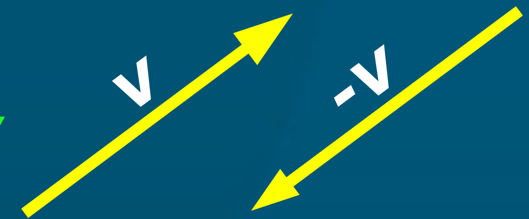
- Three basic elements in geometry
  - Scalars (α)
    - Addition, multiplication
    - Associativity, commutativity, etc. (field)
    - No geometric properties
    - E.g., reals, complex numbers
  - Vectors (v)
  - Points (P)

Vectors

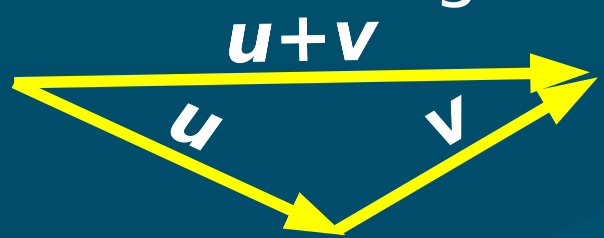
- Vectors have two attributes:
  - Direction
  - Magnitude
- No position
- Physically-inspired definition
  - e.g., force, velocity, directed line segments

Vector spaces

- Every vector *v* has an inverse *-v*



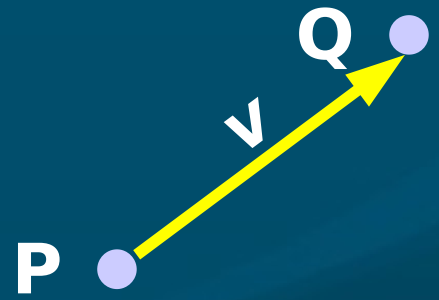
- There is a zero vector (zero magnitude)
- Adding two vectors gives another vector *u+v = w*



- Vectors can be multiplied by scalars α*v = w*

Points

- Location in space (no direction)
- Relationships between vectors and points:
  - Subtracting two points gives vector: $P - Q = v$
  - Adding a vector to a point: $Q + v = P$

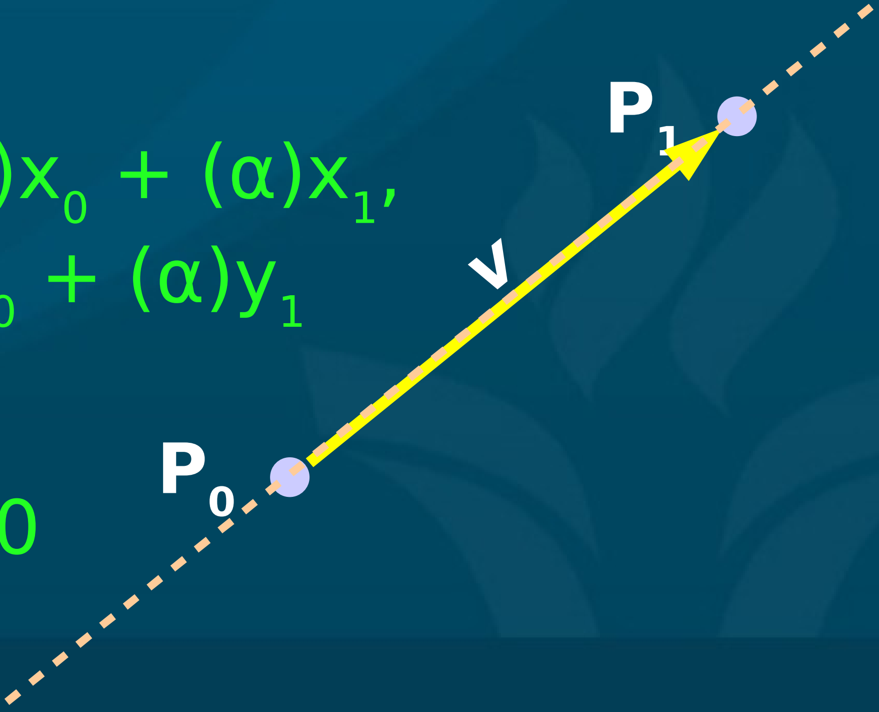


Affine spaces

- Combine a point (origin) with a vector space
  - Vector-vector addition: $u + v$
  - Scalar-vector multiplication: $\alpha * v$
  - Point-vector addition: $P + v$
  - Scalar-scalar operations: $\alpha+\beta*\gamma$
- For any point P, let
  - $1 * P = P$
  - $0 * P =$ (zero vector) 0

Lines

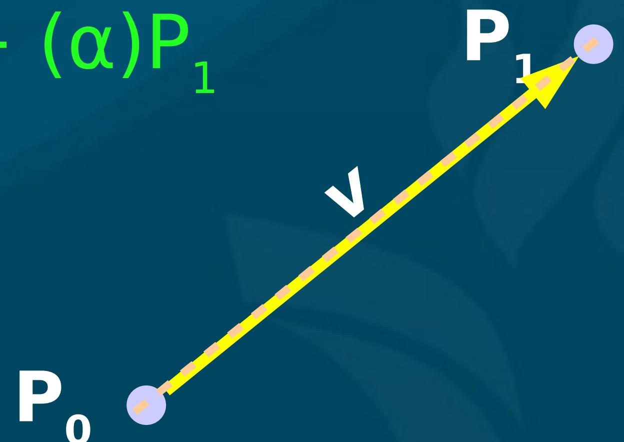
- Parametric definition of a line:
  - All points P that pass through a point P_0 in the direction of the vector v:
  - $P(\alpha) = P_0 + \alpha * v$
- Alternate forms in 2D:
  - Parametric: $x(\alpha) = (1-\alpha)x_0 + (\alpha)x_1$,
    $y(\alpha) = (1-\alpha)y_0 + (\alpha)y_1$
  - Explicit: $y = mx + h$
  - Implicit: $ax + by + c = 0$



Rays

- A ray is one side of a line:
  - All points $P(\alpha) = P_0 + \alpha * v$ for which $\alpha \geq 0$

- A line segment between points P_0 and P_1 is
  - All points $P(\alpha) = (1-\alpha)P_0 + (\alpha)P_1$ for which $0 \leq \alpha \leq 1$

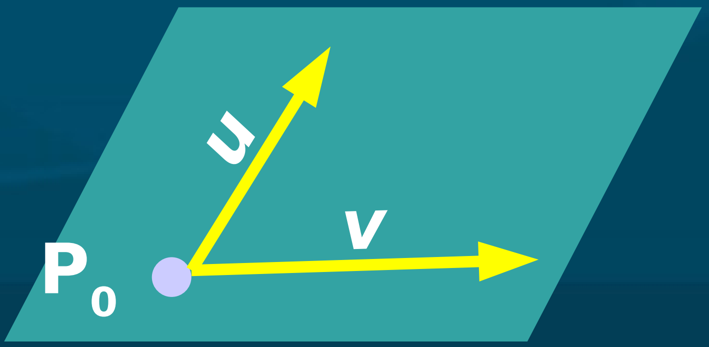


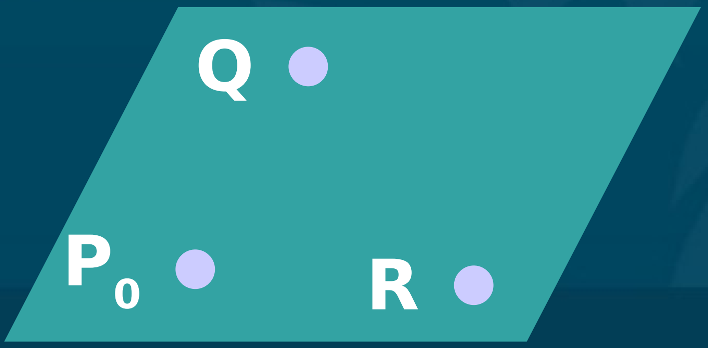
Curves

- We can generalize from lines to curves:
- Curves are one-parameter geometric entities $P(\alpha)$
  - Often have a starting point P_0
  - α can be thought of as time
    - Curve describes motion of a point through time

Surfaces

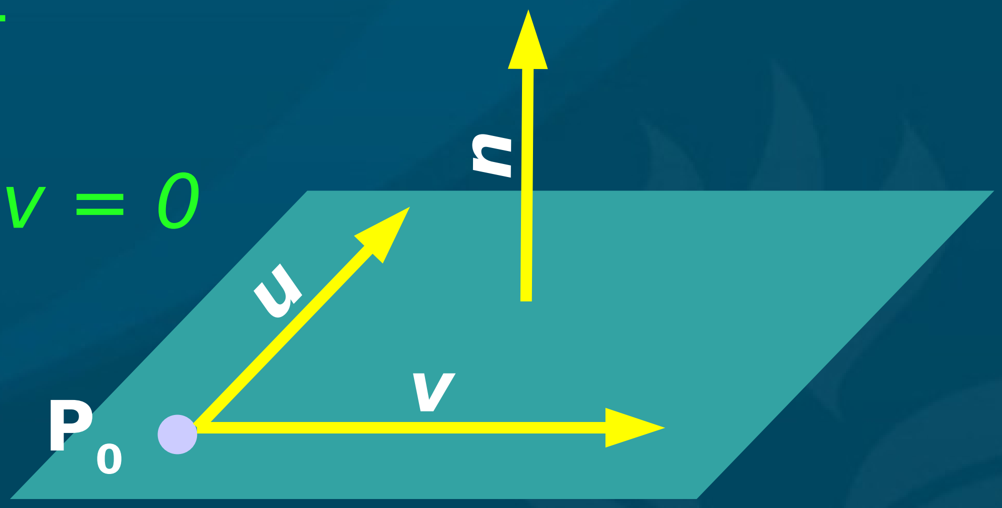
- Curves $P(\alpha)$ have one parameter α
- Surfaces $P(\alpha, \beta)$ have two parameters α, β
  - Linear functions of α, β give planes and polygons
- A plane in 3D can be defined by
  - Point + 2 vectors: $P(\alpha, \beta) = P_0 + \alpha u + \beta v$
  - Or 3 points: $P(\alpha, \beta) = P_0 + \alpha(Q-P_0) + \beta(R-P_0)$





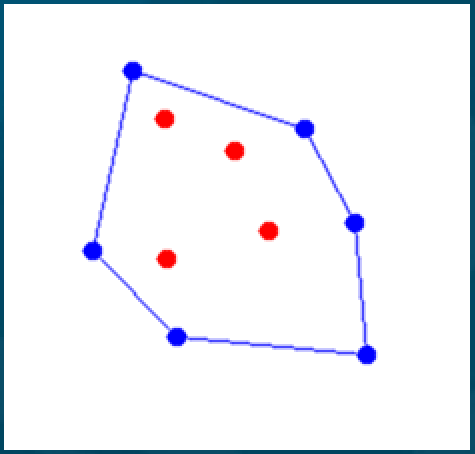
Normal vectors

- Every plane has a vector *n* which is normal (perpendicular, orthogonal) to it
- Use cross-product: $n = u \times v$
- Unit normal is the normal vector which has magnitude 1
- Perpendicular means dotproduct is zero: $n * v = 0$



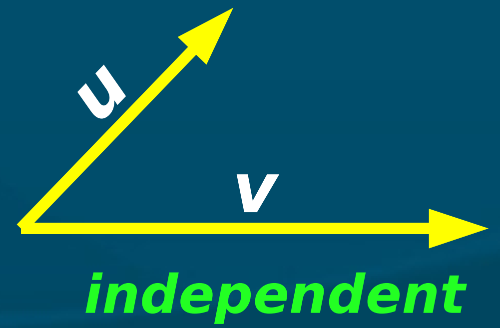
Convex hull

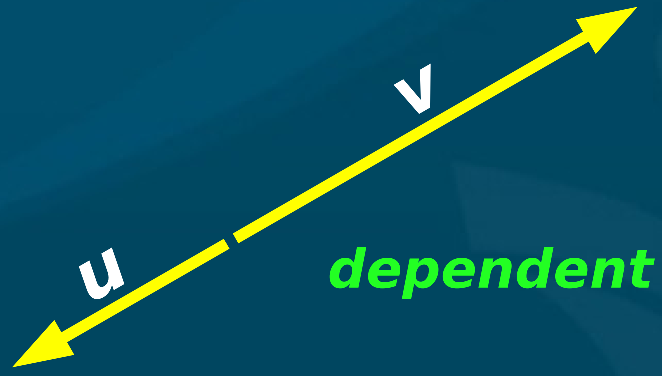
- The convex hull of a set of points $\{P_1, P_2, \ldots, P_n\}$ is the smallest convex area containing the points:
  - Convex: connect any two points in the area, the line segment is completely within the area
  - "Shrink-wrap" of the points
- Points are convex sums of $\{P_i\}$:
  - $\sum \alpha_i P_i$, where $\sum \alpha_i \leq 1$
- Triangles are the convex hulls of 3 points



Linear independence

- A set of vectors $\{v_1, v_2, \ldots, v_n\}$ is linearly independent if
  - $\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n = 0$ only when all the α_i are zero.
  - i.e., cannot represent one vector v_i as a linear combination of the others





Basis

- Any vector space has a dimension:
  - Max # of linearly independent vectors
- A basis for an n-D vector space is a set of n vectors $\{v_i\}$ such that any vector w in the space can be written as a combination of them:
  - $w = \alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n$
  - These $\{\alpha_i\}$ are a unique representation for the vector w with respect to this basis
- If the basis vectors are unit-length, this unit basis is usually written $\{e_i\}$

Frame

- A basis is enough to represent vectors, but
  - Holds no position information
- We use a frame to represent points
  - Basis + a point (origin): affine space
    - In 3D: frame $F = (P_0, e_1, e_2, e_3)$
  - Any vector $w = \alpha_1 e_1 + \alpha_2 e_2 + \ldots + \alpha_n e_n$
  - Any point $P = P_0 + \beta_1 e_1 + \beta_2 e_2 + \ldots + \beta_n e_n$
  - The representation is the scalar coefficients $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ or $(\beta_1, \beta_2, \ldots, \beta_n)$