University of British Columbia

Chapter 2

Math Review + OpenGL

Vectors

- Notations
  - column vectors

$$a_{col} = \begin{bmatrix} a_1 \\ a_2 \\ \dots \\ a_n \end{bmatrix} \qquad a_{col}^T = a_{row}$$

  - row vectors

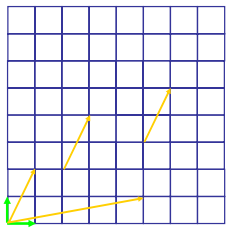
$$a_{row} = [a_1 \quad a_2 \quad \dots \quad a_n]$$

- Simple operations:
  - sum (subtract), multiply by scalar

University of British Columbia

Vectors

- Geometric meaning – oriented segment in nD space



University of British Columbia

Coordinate Systems

Right-handed Coordinate System

Left-handed Coordinate System

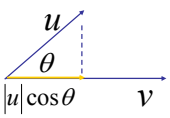
University of British Columbia

Dot Product

- Or *inner product*

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \bullet \begin{bmatrix} a \\ b \\ c \end{bmatrix} = x*a + y*b + z*c \qquad P \bullet N$$

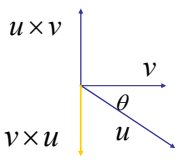
$$u \bullet v = \|u\|\|v\|\cos\theta$$



University of British Columbia

Cross Product

$$[x \quad y \quad z] \times [a \quad b \quad c] = [yc - za \quad za - xc \quad xb - ya]$$



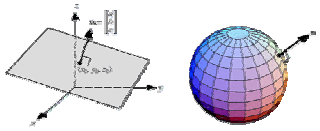
Right Handed Coordinate System

(curl fingers from u to v; thumb points to u x v)

$$\|u \times v\| = \|u\|\|v\|\sin\theta$$

University of British Columbia

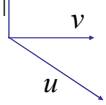
Normal



Normal: unit vector perpendicular to surface
Unit vector – vector of length 1

$$\frac{u \times v}{\| u \times v \|}$$

Normal to plane containing u **&** v



Convention: list vertices of polygon counterclockwise around normal

University of British Columbia

Math Review

- Coordinate Systems

Right-handed Coordinate System

$z = x \times y$
using right-hand rule

Left-handed Coordinate System

$z = x \times y$
using left-hand rule

University of British Columbia

Math Review

- matrix vector multiplication
  - points as column vectors

$$\begin{bmatrix} x' \\ y' \\ z' \\ h' \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ h \end{bmatrix} \qquad P' = MP$$

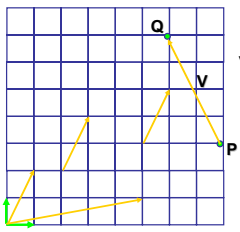
- points as row vectors

$$\begin{bmatrix} x' & y' & z' & h' \end{bmatrix} = \begin{bmatrix} x & y & z & h \end{bmatrix} \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}^T \qquad P'^T = P^T M^T$$

University of British Columbia

Math Reviw

- Points and Vectors



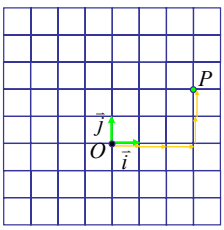
vector space
vectors are invariant under translation
affine space:
allows vector-to-point addition

$$P + V = Q$$
$$Q - P = V$$

University of British Columbia

Math Reviw

- Coordinate System vs Frame



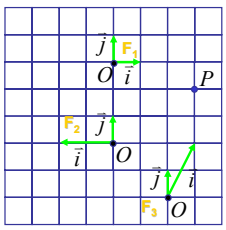
coordinate system: basis vectors
frame: basis vectors + Origin
allows for points

$$P = O + x\vec{i} + y\vec{j}$$

University of British Columbia

Math Review

- Working with Frames



$$P = O + x\vec{i} + y\vec{j}$$

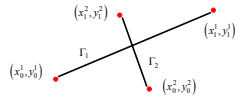
F_1 P(3,-1)

F_2 P(-1.5,2)

F_3 P(1,2)

University of British Columbia

Line-Line Intersection



$$G_1 = \begin{cases} x^1(t) = x_0^1 + (x_1^1 - x_0^1)t \\ y^1(t) = y_0^1 + (y_1^1 - y_0^1)t \end{cases} \quad t \in [0,1] \qquad G_2 = \begin{cases} x^2(r) = x_0^2 + (x_1^2 - x_0^2)r \\ y^2(r) = y_0^2 + (y_1^2 - y_0^2)r \end{cases} \quad r \in [0,1]$$

Intersection: *x* & *y* values equal in both representations - two linear equations in two unknowns (*r*,*t*)

$$\begin{aligned} x_0^1 + (x_1^1 - x_0^1)t &= x_0^2 + (x_1^2 - x_0^2)r \\ y_0^1 + (y_1^1 - y_0^1)t &= y_0^2 + (y_1^2 - y_0^2)r \end{aligned}$$

Question: What is the meaning of $r,t < 0$ or $r,t > 1$?

University of British Columbia

Linear Operators

Definition: *linear* operator $\Gamma : A \to B$ satisfies :

$$\Gamma(aX + bY) = a\Gamma(X) + b\Gamma(Y)$$

Examples

Differentiation

$$D(h(x)) = h'(x)$$

$$\begin{aligned} D(af(x) + bg(x)) &= D(af(x)) + D(bg(x)) \\ &= aD(f(x)) + bD(g(x)) \end{aligned}$$

Scaling

$$S(h(x)) = sh(x)$$

$$\begin{aligned} S(af(x) + bg(x)) &= s(af(x) + bg(x)) \\ &= asf(x) + bsg(x) \\ &= aS(f(x)) + bS(g(x)) \end{aligned}$$

Question: Is $F(X) = \alpha X + \beta$ a linear operation ?

University of British Columbia

Linear Operators (continued)

Answer: No !!

$$\begin{aligned} & F(aX + bY) \\ &= \alpha(aX + bY) + \beta \\ &= \alpha(aX) + \alpha(bY) + \beta \\ &= a\alpha X + b\alpha Y + \beta \\ &= a(\alpha X + \beta) + b(\alpha Y + \beta) + (1 - a - b)\beta \\ &= aF(X) + bF(Y) - (1 - a - b)\beta \end{aligned}$$

Definition: *affine* operator *A* satisfies:

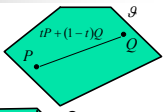
$$A(aX + bY) = aA(X) + bA(Y), \qquad for \;\; a + b \equiv 1$$

Question: Is $F(X) = \alpha X + \beta$ an affine operator ?

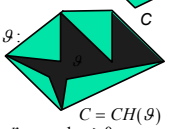
University of British Columbia

Convexity

Object ϑ is **convex** iff for any two points $P, Q \in \vartheta, tP + (1-t)Q \subseteq \vartheta, t \in [0,1]$.



Convex hull $CH(\vartheta)$, of object ϑ: minimal convex shape C such that $\vartheta \subseteq C$



$C = CH(\vartheta)$

For n points $\{P_i\}_{i=1}^n$ & $\{a_i\}_{i=1}^n, \Sigma_{i=1}^n a_i = 1, a_i \geq 0.$

$P = \Sigma_{i=1}^n a_i P_i$ is an **affine combination** of $\{P_i\}_{i=1}^n$.

$P \in CH\left(\{P_i\}_{i=1}^n\right)$

For $n = 2, P \in \overline{P_1 P_2}$

University of British Columbia

Triangle

- Normal

$$n = \frac{(P_1 - P_0) \times (P_2 - P_0)}{\left\| (P_1 - P_0) \times (P_2 - P_0) \right\|}$$

- Area

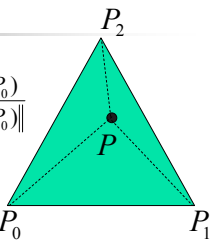
$$A = \frac{1}{2} \left\| \overrightarrow{P_0 P_1} \times \overrightarrow{P_0 P_2} \right\|$$

- Barycentric coordinates

$$a_0 = A_{P_1 P_2 P} / A, a_1 = A_{P_2 P_0 P} / A,$$

$$a_2 = A_{P_0 P_1 P} / A,$$

$$P = a_0 P_0 + a_1 P_1 + a_2 P_2$$

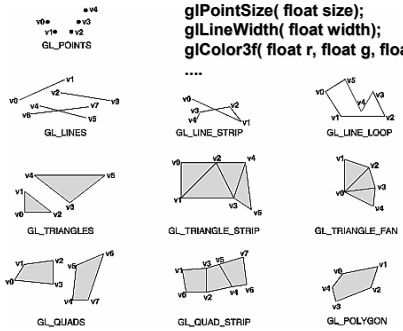


- Use for assigning properties

University of British Columbia

Open GL: Primitives

glPointSize(float size);
glLineWidth(float width);
glColor3f(float r, float g, float b);
....



University of British Columbia

GLUT: OpenGL Utility Toolkit

- The basics...

```
int main(int argc, char **argv)
{
      glutInit( &argc, argv );
      glutInitDisplayMode( GLUT_RGB |
                           GLUT_DOUBLE | GLUT_DEPTH);
      glutInitWindowSize( 640, 480 );
      glutCreateWindow( "openGLDemo" );
      glutDisplayFunc( DrawWorld );
      glutIdleFunc(Idle);
      glClearColor( 1,1,1 );
      glutMainLoop();

      return 0;       // never reached
}
```

University of British Columbia

GLUT Example

```
void DrawWorld() {
      glMatrixMode( GL_PROJECTION );
      glLoadIdentity();
      glMatrixMode( GL_MODELVIEW );
      glLoadIdentity();
      glClear( GL_COLOR_BUFFER_BIT );
      angle += 0.05;
      glRotatef(angle,0,0,1);
      ...  // draw triangle
      glutSwapBuffers();
}
```

University of British Columbia

GLUT Example

- TRIANGLE...

```
glColor3f(0,1,0);
glBegin( GL_TRIANGLES );
  glVertex3f( 0.0f, 0.5f, 0.0f );
  glVertex3f( -0.5f, -0.5f, 0.0f );
  glVertex3f( 0.5f, -0.5f, 0.0f );
glEnd();
```

University of British Columbia

GLUT Example

```
void Idle() {
      angle += 0.05;
      glutPostRedisplay();
}
```

University of British Columbia

GLUT Input Events

```
  // you supply these kind of functions
void reshape(int w, int h);
void keyboard(unsigned char key, int x, int y);
void mouse(int but, int state, int x, int y);


  // register them with glut
glutReshapeFunc(reshape);
glutKeyboardFunc(keyboard);
glutMouseFunc(mouse);
```

University of British Columbia

GLUT and GLU primitives

```
gluSphere(...)
gluCylinder(...)

glutSolidSphere(...)
glutWireSphere(...)

glutSolidCube(...)
glutWireCube(...)

glutSolidTorus(...)
glutWireTorus(...)

glutSolidTeapot(...)
glotWireTeapot(...)
```

University of British Columbia

Depth buffer

- for visibility
  - stores a z-value for every pixel
  - smaller z means "closer"

```
  // allocate depth buffer
glutInitDisplayMode( GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);

  // enabling the depth test
glEnable( GL_DEPTH_TEST );

  // clearing the depth buffer for each frame
glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
```

University of British Columbia

GLUT menus

```
glutCreateMenu(...)
glutSetMenu(...)
glutGetMenu(...)
glutDestroyMenu(...)
glutAddMenuEntry(...)
glutAddSubMenu(...)
glutAttachMenu(...)

  // Example usage
glutCreateMenu(demo_menu);
glutAddMenuEntry("quit", 1);
glutAddMenuEntry("Increase Square Size", 2);
glutAttachMenu(GLUT_RIGHT_BUTTON);
```

University of British Columbia

Assignment 0

- Target:
  - Experience OpenGL & GLUT
  - See "real" models – meshes in OBJ format
- Description:
  http://www.ugrad.cs.ubc.ca/~cs314/Vsep2004/a0/a0_desc.html
- Do NOT submit
- Basis for future assignments

University of British Columbia