Today’s Agenda

Input and Interaction

Geometry
OpenGL Sources

OpenGL website

https://www.opengl.org/
Event Types

Window: resize, expose, iconify
Mouse: click one or more buttons
Motion: move mouse
Keyboard: press or release a key
Idle: nonevent
  • Define what should be done if no other event is in queue
GLUT Callbacks

GLUT recognizes a subset of the events recognized by any particular window system (Windows, X, Macintosh)

- glutDisplayFunc
- glutMouseFunc
- glutReshapeFunc
- glutKeyboardFunc
- glutIdleFunc
- glutMotionFunc,
- glutPassiveMotionFunc

These call back functions except the reshape require posting redispays

    glutPostRedisplay();
Using the Keyboard

```c
void mykey(unsigned char key, int x, int y)
{
    if (key == 'Q' | key == 'q')
        exit(0);
}
```
Handling Multiple Key Inputs

• For ASCII character

Each key press of will trigger the key callback function
  • Use switch & case
  • Or use a buffer to store the key strokes
    buffer[key] = true

• For Non ASCII character

  • Function keys (e.g., F1) or directional keys (e.g. →)
    void glutSpecialFunc(void (*func)(int key, int x, int y));

  • State modifier keys (e.g., “Shift” and “Ctrl”)
    glutGetModifiers()
Manage Multiple Windows

- Create a second window
  
  ```c
  uint id = glutCreateWindow("second window");
  ```
- Set the window as the current window for rendering
  
  ```c
  glutSetWindow(id);
  ```
- Each window can have its own call back functions
Toolkits and Widgets

Most window systems provide a toolkit or library of functions for building user interfaces that use special types of windows called *widgets*.

Widget sets include tools such as

- Menus
- Slidebars
- Dials
- Input boxes

But toolkits tend to be platform dependent

GLUT provides a few widgets including menus.
Menus

GLUT supports pop-up menus
  • A menu can have submenus

Three steps
  • Define entries for the menu
  • Define action for each menu item
    – Action carried out if entry selected
  • Attach menu to a mouse button
  • Register a callback function for each menu
Defining a simple menu

In `main.c`

```c
menu_id = glutCreateMenu(mymenu);
glutAddmenuEntry("clear Screen", 1);
gluAddMenuEntry("exit", 2);
glutAttachMenu(GLUT_RIGHT_BUTTON);
```

Entries that appear when right button depressed

- clear screen
- exit

Identifiers
Menu Actions

- Menu callback

```c
void mymenu(int id)
{
    if(id == 1) glClear();
    if(id == 2) exit(0);
}
```

- Add submenus by

```c
glutAddSubMenu(char *submenu_name, submenu id)
```

entry in parent menu

Note Menu is a deprecated feature and will not work for a core profile
Reading Assignments

Chapter 2. of Angels et al

Chapter 2&3 Shreiner et al
Geometric Objects and Transformations
Basic Elements

Geometry is the study of the relationships among objects in an n-dimensional space
  - In computer graphics, we are interested in objects that exist in three dimensions

Want a minimum set of primitives from which we can build more sophisticated objects

We will need three basic elements
  - Points  → represented by uppercase letters, e.g., P, Q
  - Scalars  → represented by Greek letters, e.g., α,β
  - Vectors  → represented by lowercase letters, e.g., v,w
Points

- Fundamental geometric object
- Associated with location
- No size & shape
Scalars

Scalars can be defined as members of sets which can be combined by two operations (addition and multiplication) obeying some fundamental axioms (associativity, commutivity, inverses)

• Examples: the real and complex number systems

Scalars alone have no geometric properties
Vectors

Physical definition: a vector is a quantity with two attributes
• Direction
• Magnitude

Examples include
• Force
• Velocity
• Directed line segments
  – Most important example for graphics
  – Can map to other types
Vectors Lack Position

These vectors are identical
  • Same length and magnitude

Vectors spaces insufficient for geometry
  • Need points
Point-Vector Addition/Subtraction

Points define locations in space

Operations allowed between points and vectors

- Point-point subtraction yields a vector
- Point-vector addition yields a new point

\[ P = v + Q \]
\[ v = P - Q \]

Destination point
Start point

3D vectors representing points
3D vector representing displacement
Coordinate-Free Geometry

When we learned simple geometry, most of us started with a Cartesian approach
  • Points were at locations in space \( p=(x,y,z) \)

This approach was nonphysical
  • Physically, points exist regardless of the location of an arbitrary coordinate system
  • Most geometric results are independent of the coordinate system
    – Example: two triangles are identical if two corresponding sides and the angle between them are identical
Spaces

(Linear) vector space: scalars and vectors

Affine space: vector space + points

Euclidean space: vector space + distance
Vector Operations

Every vector has an inverse
• Same magnitude but points in opposite direction

Every vector can be multiplied by a scalar

There is a zero vector
• Zero magnitude, undefined orientation

The sum of any two vectors is a vector
• Use head-to-tail axiom
Linear Vector Spaces

Mathematical system for manipulating vectors

Operations

• Scalar-vector multiplication: \( \mathbf{u} = \alpha \mathbf{v} \)

• Vector-vector addition: \( \mathbf{w} = \mathbf{u} + \mathbf{v} \)

Expressions such as

\[ v = u + 2w - 3r \]

Make sense in a vector space
Linear Independence

A set of vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ is *linearly independent* if

$$\alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + \ldots + \alpha_n \mathbf{v}_n = 0 \text{ iff } \alpha_1 = \alpha_2 = \ldots = 0$$

If a set of vectors is linearly independent, we cannot represent one in terms of the others

If a set of vectors is linearly dependent, at least one can be written in terms of the others
Dimension, Basis, and Representation

**Dimension of the space:** the maximum number of linearly independent vectors

- Fixed for a space

In an \( n \)-dimensional space, any set of \( n \) linearly independent vectors form a *basis* for the space

**The basis for the space is not unique!**

Given a basis \( v_1, v_2, \ldots, v_n \), any vector \( v \) can be written as

\[
v = \alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n
\]

where the \( \{\alpha_i\} \) are unique
Changing Representation

The same vector $v$ can be represented differently given different bases

For a basis $v_1, v_2, \ldots, v_n$, a vector $v$ can be written as

$$v = \alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n$$

For a different basis $v'_1, v'_2, \ldots, v'_n$, $v$ can be written as

$$v = \alpha'_1 v'_1 + \alpha'_2 v'_2 + \ldots + \alpha'_n v'_n$$

Where

$$\begin{bmatrix}
\alpha_1 \\
\alpha_2 \\
\vdots \\
\alpha_n
\end{bmatrix} = M \begin{bmatrix}
\alpha'_1 \\
\alpha'_2 \\
\vdots \\
\alpha'_n
\end{bmatrix}$$
Affine Spaces

Point + a vector space

Operations

- Vector-vector addition
- Scalar-vector multiplication
- Scalar-scalar operations
- Point-vector addition
- Point-point addition
- Scalar-Point multiplication

For any point define

- $1 \cdot P = P$
- $0 \cdot P = 0$ (zero vector)
Consider all points of the form

- $P(\alpha) = P_0 + \alpha \, d$
- Set of all points that pass through $P_0$ in the direction of the vector $d$
- If $\alpha \geq 0$, then $P(\alpha)$ is the ray leaving $P_0$ in the direction $d$

This form is known as the parametric form of

- More robust and general than other forms
- Extends to curves and surfaces
Line Segments

If we use two points to define $v$, then

$$P(\alpha) = Q + \alpha v = Q + \alpha (R - Q) = \alpha R + (1 - \alpha)Q$$

For $0 \leq \alpha \leq 1$ we get all the points on the line segment joining $R$ and $Q$
Convexity

An object is convex iff for any two points in the object all points on the line segment between these points are also in the object.

A line segment is a convex object

\[ P(\alpha) = \alpha R + (1 - \alpha)Q = \alpha_R R + \alpha_Q Q \]

Can we extend it to N points?

Affine Sums and Convex Hull

Consider the “sum”

\[ P = \alpha_1 P_1 + \alpha_2 P_2 + \cdots + \alpha_n P_n \]

Can show by induction that this sum makes sense iff

\[ \alpha_1 + \alpha_2 + \cdots + \alpha_n = 1 \]

in which case we have the affine sum of the points \( P_1, P_2, \ldots, P_n \)

If, in addition, \( \alpha_i \geq 0 \), we have the convex hull of \( P_1, P_2, \ldots, P_n \)

Smallest convex object containing \( P_1, P_2, \ldots, P_n \)

Formed by “shrink wrapping” points