Interactive Techniques

Ben Fry’s Zipdecode

http://acg.media.mit.edu/people/fry/zipdecode/
Polling vs. interrupts
Event model
Picking
Design patterns
- Model-view-controller
- Game loops
Interactive techniques

Polling and Interrupts

Polling - Periodically read the state of the devices
- Mouse sends messages
  - 100 times per second
  - State of the buttons
  - Relative motion of the mouse
- Joystick/Gamepad interface
  - glutJoyStick( int buttonmask, int x, int y, int z )
  - STJoystick supports multiple joysticks, axes, balls, hats, and buttons

Interrupts - Send changes when the occur
- Keyboard sends “make” and “break” messages
Event Queue

Operating system responds to interrupt
Creates event record and places in a queue
- Time-stamp
- Queues are first-in first-out (FIFO)
- May store state of other devices at time of event
e.g. polled devices like the mouse

Callback manager
- Event priority
  - Reorder events based on priority
e.g. keyboard has priority over PostRedisplay
- Event coalescing
  - Combine mouse motions so current position is returned

Demo of input.py

Picking

Three issues
- How to find objects under the cursor
- How to assign priority to these objects
- How to handle hierarchy

Hit testing / picking
- Bounding rectangle
- Hit method for objects
- Pick support during rendering (OpenGL)
- Object tags
Flash Button Tracking

Tracking: Convert in/out state to enter/leave events

Flash MenuButton Tracking

Tracking: Convert in/out state to enter/leave events
Design Patterns

Model-View-Controller Design Pattern

```java
m = new Model();
m.addView(v1);
m.addView(v2);

v1 = new ViewA(m, c);
v2 = new ViewA(m, c);
v2 = new ViewB(m, c);
```

v1 = new ViewA(m, c);  c = new Controller(m);
v2 = new ViewA(m, c);
v2 = new ViewB(m, c);
Model-View-Controller Design Pattern

v.onChange() = function() {
    v.c.setName(v.value);
}

c.setName = function(s) {
    c.m.setName(s);
    c.m.onUpdate();
}

v.onChange() = function() {
    v.c.setName(v.value);
    c.m.setName(s);
    c.m.onUpdate();
}
m.onUpdate = function() {
    for v in m.views:
        v.onUpdate();
}

v.onChange() = function() {
    v.c.setName(v.value);
    c.m.setName(s);
    c.m.onUpdate();
}

c.setName = function(s) {
    c.m.setName(s);
    c.m.onUpdate();
}

m.onUpdate = function() {
    for v in m.views:
        v.onUpdate();
}

v.onUpdate() = function() {
    m.draw(v);
    c.m.setName(s);
    c.m.onUpdate();
}

c.setName = function(s) {
    c.m.setName(s);
    c.m.onUpdate();
}
Interactive Techniques

Basic Techniques

Excerpt from Sutherland video

Dragging
- Flash demo of editing shapes

Position control
- OpenGL asteroid program
- Game design pattern
Things to Remember

Flow from input to events to method invocation
- Inputs cause interrupts which generate events
- Events are routed to objects
- Pointer events require “hit testing”
- Model-view-controller
- Game simulation design pattern

Interactive techniques
- How to assemble events into interactive techniques
- The flash button model is a nice example of a low level behavior