

OPENGL - ANIMATING INTERACTIVE PROGRAMS

3.10 Animating interactive programs

The points $x=\cos \theta$, $y=\sin \theta$ always lies on a unit circle regardless of the value of θ .

- In order to increase θ by a fixed amount whenever nothing is happening, we use the idle function

```
void(idle)
{
    theta+=2;
    If (theta > 360.0) theta -= 360.0;
    glutPostRedisplay();
}
```

- In order to turn the rotation feature on and off, we can include a mouse function as follows :

```
Void mouse(int button, int state, intx, int y)
{
    if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
        glutIdleFunc(idle);
    if (button == GLUT_RIGHT_BUTTON && state == GLUT_DOWN)
        glutIdleFunc(NULL);
}
```

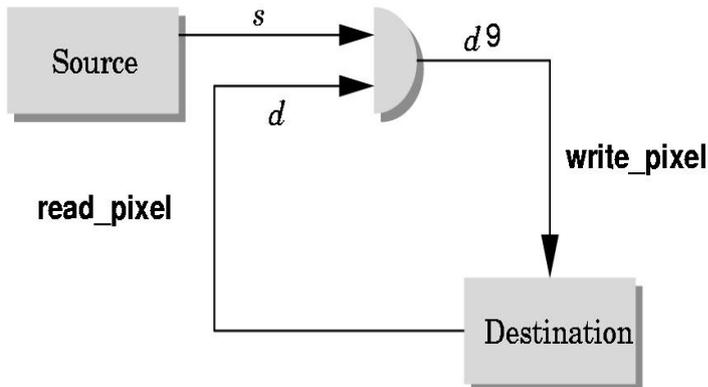
Double Buffering

- We have 2 color buffers for our disposal called the Front and the Back buffers.
- Front buffer is the one which is always displayed.
- Back buffer is the one on which we draw
- Function call to swap buffers :
- `glutSwapBuffers();`
- By default openGl writes on to the back buffer.
- But this can be controlled using

```
glDrawBuffer(GL_BACK);  
glDrawBuffer(FRONT_AND_BACK);
```

Writing Modes

XOR write



- Usual (default) mode: source replaces destination ($d' = s$)
 - Cannot write temporary lines this way because we cannot recover what was “under” the line in a fast simple way
- Exclusive OR mode (XOR) ($d' = d \oplus s$)
 - $x \oplus y \oplus x = y$
 - Hence, if we use XOR mode to write a line, we can draw it a second time and line is erased!

Rubberbanding

- Switch to XOR write mode
- Draw object
 - For line can use first mouse click to fix one endpoint and then use motion callback to continuously update the second endpoint
 - Each time mouse is moved, redraw line which erases it and then draw line from fixed first position to to new second position
 - At end, switch back to normal drawing mode and draw line
 - Works for other objects: rectangles, circles

XOR in OpenGL

- There are 16 possible logical operations between two bits

- All are supported by OpenGL
 - Must first enable logical operations
 - **glEnable(GL_COLOR_LOGIC_OP)**
 - Choose logical operation
 - **glLogicOp(GL_XOR)**
 - **glLogicOp(GL_COPY)** (default)

Source : <http://elearningatria.files.wordpress.com/2013/10/cse-vi-computer-graphics-and-visualization-10cs65-notes.pdf>