Project 1 Discussion

Tom Kelliher, CS 320

Feb. 9, 2005

1 Administrivia

Announcements

Assignment

Re-read Chapter 2.

From Last Time

OpenGL API.

Outline

1. Project 1 discussion.

Coming Up

2-D graphics and OpenGL lab.

2 Project 1

2.1 World/Window Coordinates

1. Let cell size be 1.0×1.0.

2. Use a one cell wide border around the maze. Special encoding to indicate these cells? Don't render border.

   Orthogonal projection; viewing volume.

3. Use a 500×500 window. The maze won't always be square. Also, deal with resizing. See `reshape()` in `quadric.c`.

   Fall backs:

   (a) Square maze.

   (b) Square window (window not resized).

2.2 Cell Data Structure

1. Maze is 2-D array of cells. Max maze: 100×100. Generalization: any size array, dynamically sized at run-time. Extra credit.

   Entrance lower-left bottom, exit upper-right top.

2. Needed for each cell: visited; north, south, east, west walls up.

   Redundancy: remove.

   Draw south, west maze walls.

3. `struct` declaration:

```
typedef struct cell
{
   int visited;
   int north;
   int east;
} cell;
```

2.3 Sketch of `main()`

```
cell maze[ROWS][COLS];

main(...)
{
   process args;

   set-up window;
   set-up world;

   initialize maze and seed random number generator;

   /* Create a random maze. */
   visit(start cell);

   /* Exit on mouse click. */
   register MouseFunc;

   /* On re-display clear window, redraw maze, re-compute
    *   and display path.
    */
   register DisplayFunc;

   /* IdleFunc unused!!! */

   /* "Prime the pump." */
   post a redisplay event;

   enter MainLoop;
}
```

2.4 `visit()` Pseudo-Code

```
visit(current cell)
{
   mark current cell visited;

   Randomly arrange the four compass directions

   for each of the four randomly arranged compass directions
      if new cell exists and is unvisited
      {
         remove appropriate wall;
         visit(new cell);
      }
}
```

2.5 `path()` Pseudo-Code

Idea: Get a sequence of cells from start cell to end cell on the call stack and draw path from end to start.

Don't forget to mark all maze cells as unvisited before first call!

Initial call: `path(start cell, end cell);`

```
path(current cell, target cell)
{
   mark current cell visited;

   if current cell == target cell
      return 1;

   for each of the four neighbor cells
      if new cell exists and is unvisited
         if (path(new cell, target cell))
         {
            draw segment from current cell to new cell;
            return 1;
         }

   return 0;
}
```

2.6 A Sample Maze

Need to draw entry/exit segments. Use two colors. 4×4 maze: