The Slitherlink Puzzle (continued)

Lecture 14

Robb T. Koether

Hampden-Sydney College

Wed, Sep 20, 2017

Outline

Outline

Drawing the Cells

- Each cell must be drawn individually.
  - Each cell is drawn as a triangle fan.
  - We must use the correct color (undercount, matching count, overcount).
- The edges are shared by adjacent cells. Therefore,
  - Vertical and horizontal edges must be drawn separately from the cells.
  - Each edge is drawn as a line.
- We must draw the numeral whenever it has been specified.

Outline

Coloring the Cells

Puzzle Class

```
class Puzzle
{
    Status status(int row, int col) const;
};
```

- Create three colors, one each for undercount, matching count, and overcount.
- The `status()` function will return one of the `Status` **enums** `UNDER`, `OK`, or `OVER`.
- When a cell is drawn, make the three-way decision that selects and sets the correct color.
- Then draw the cell.

Outline

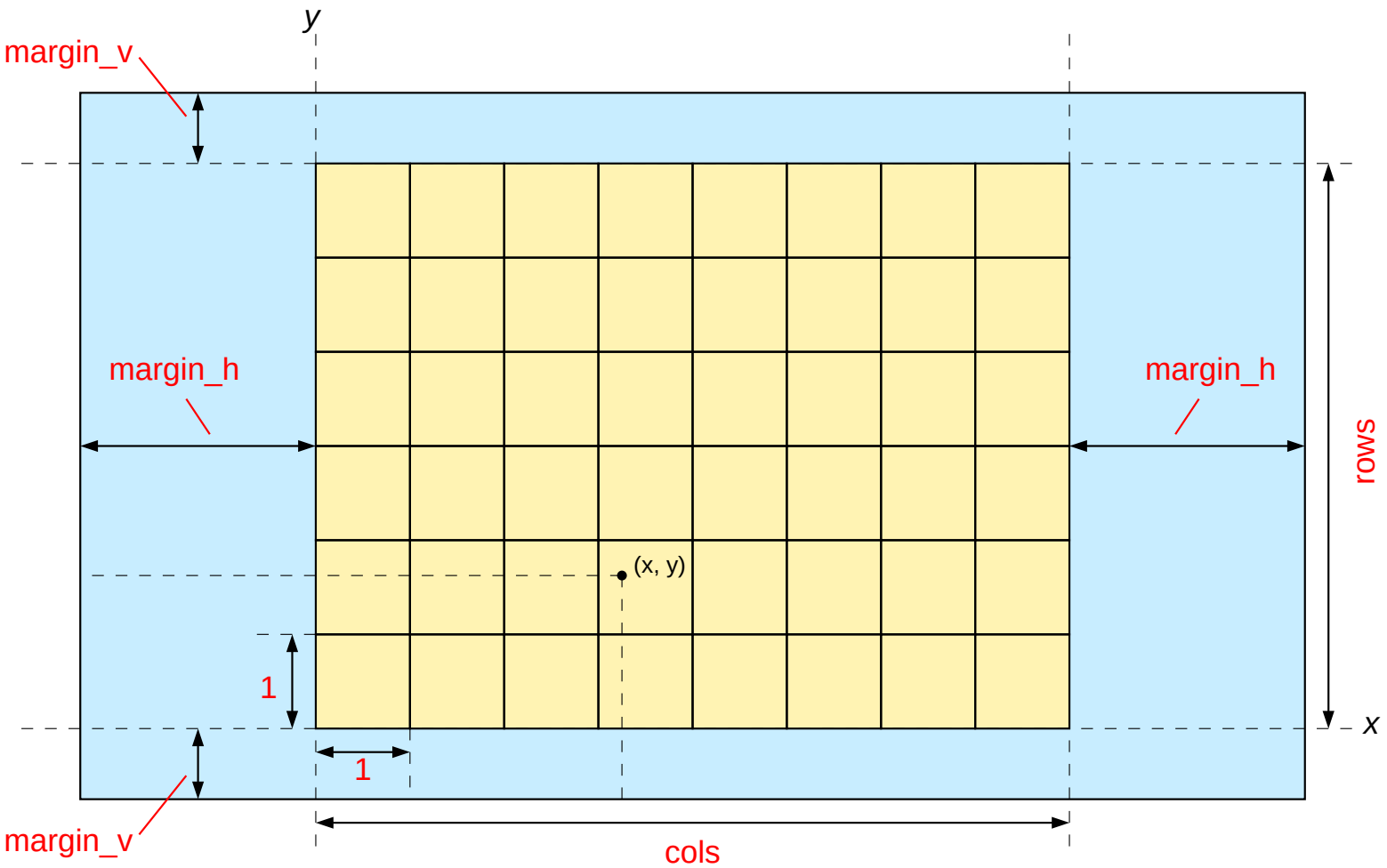
- I will provide you with a function `createNumerals()` that creates the numerals 0, 1, 2, and 3 and copies them to buffers.
- Each numeral is constructed as a set of line segments.
- You will write the functions that draw them (easy-peasy).

Outline

Handling Left-Clicks

- When the user left-clicks the mouse, we must first convert the screen coordinates to world coordinates.
- Then we must
  - Determine whether he clicked on the puzzle board.
  - If so, then on which edge did he click?
  - And, was it a horizontal edge or a vertical edge?
  - Toggle the edge.

Converting to World Coordinates



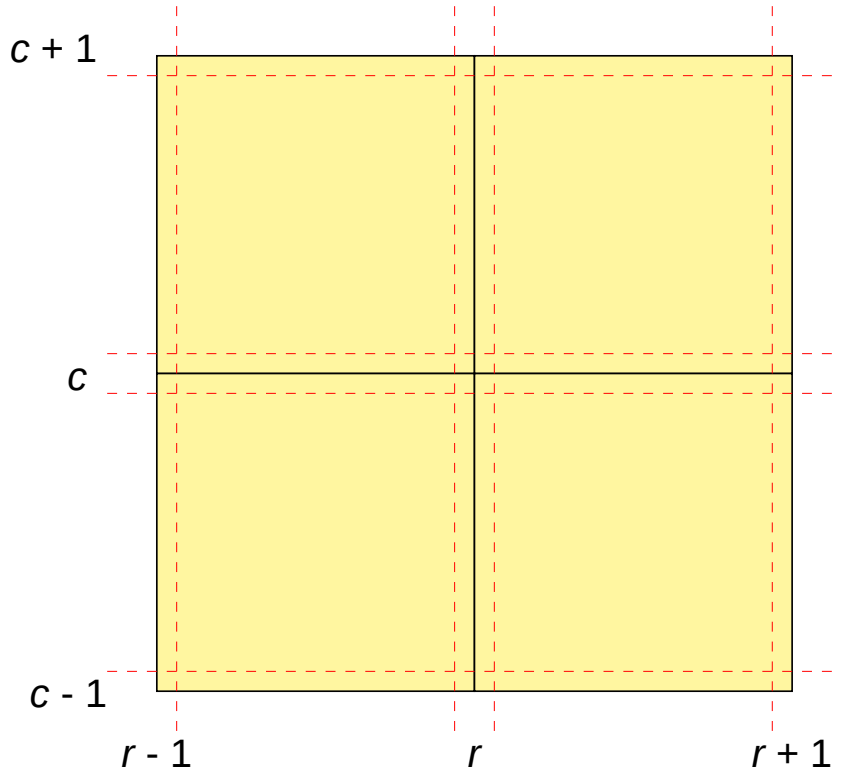
Converting to World Coordinates

- Let x be the x-coordinate in screen coordinates and let x' be the x-coordinate in world coordinates.
- We see that, proportionally,

$$\frac{x}{\text{width}} = \frac{\text{margin\_h} + x'}{\text{cols} + 2 \times \text{margin\_h}}$$

- We can solve this equation for x'.
- In a similar way, we can find y' in world coordinates.

Determining Which Vertical Edge



- We cannot expect the user to click *precisely* on an edge.
- We must allow a small tolerance (red dotted lines).

Determining Which Vertical Edge

- For vertical edges, find the nearest integer to x'.
- Then check whether x' is within the tolerance of that integer, either way.

Determining Which Vertical Edge

- For vertical edges, find the nearest integer to x'.
- Then check whether x' is within the tolerance of that integer, either way. (How do we do that?)

Determining Which Vertical Edge

- For vertical edges, find the nearest integer to x'.
- Then check whether x' is within the tolerance of that integer, either way. (How do we do that?)
- If so, then the click counts as on that vertical line.
- Register that fact (true or false) with a `bool on_vertical`.
- Do the same in the horizontal direction and set `on_horizontal`.
- If *exactly one* of them is true, then the user clicked on an edge.

Toggling the Edge

Puzzle Class

```
class Puzzle
{
    void toggleVEdge(int row, int col);
    void toggleHEdge(int row, int col);
};
```

- If a vertical edge was left-clicked on, then toggle that edge (row and column).
- If a horizontal edge was left-clicked on, then toggle that edge.
- How do we determine the parameters `row` and `col`?

Drawing the Edges

- When we draw each edge, we must make a three-way decision.
  - Is the edge visible? If not, then skip it.
  - Is the edge selected? If so, then draw it wider and in a prominent color.
  - If not, then draw it as a thin black edge (visible and unselected).

Drawing the Edges

Puzzle Class

```
class Puzzle
{
bool v_edge(int row, int col) const;
bool h_edge(int row, int col) const;
bool v_visible(int row, int col) const;
bool h_visible(int row, int col) const;
};
```

- The `Puzzle` class has inspector functions that help with those decisions.

Outline

Handling Right-Clicks

Puzzle Class

```
class Puzzle
{
    void toggleVVisible(int row, int col);
    void toggleHVisible(int row, int col);
};
```

- Handling right-clicks is the same as handling left-clicks except that we call a different pair of toggle functions.

Drawing the Edges

The `glLineWidth()` Function

```
void glLineWidth(GLfloat width);
```

- The width is expressed in pixels.
- The default is `1.0`.

Outline

Detecting a Win

- We will not attempt to detect a win.

Outline

Assignment

Assignment

- Assignment.