ME/CS 132: Homework # 2 Solutions

February 17, 2009

Problem # 1

Consider a fixed orientation slice through the configuration space obstacle associated with a convex polygon robot and a convex polygon obstacle. We know that the boundary of this slice is a polygon. Prove that the polygon is convex.

Answer:

- Let $\mathbf{R}$ be a convex set representing the convex polygon robot.
- Let $\mathbf{O}$ be a convex set representing the convex polygon obstacle.

We know that for a convex set, $\mathbf{C}$, the following must be true:

for $(x_i, x_j) \in \mathbf{C}$ then:

$$\lambda x_i + (1-\lambda)x_j \in \mathbf{C} \tag{1}$$

Since we can define the C-obstacle space by the Minkowski Difference, we can create a new set $\mathbf{H}$ defined as:

$$\mathbf{H} = \{r + o : r \in \mathbf{R},\ o \in \mathbf{O}\} \tag{2}$$

Now consider $(h_1, h_2) \in \mathbf{H}$. Let $h_1 = r_1 + o_1$ and let $h_2 = r_2 + o_2$, then:

$$\begin{aligned}
\lambda r_1 + (1-\lambda)r_2 &\in \mathbf{R} && (3)\\
\lambda o_1 + (1-\lambda)o_2 &\in \mathbf{O} && (4)\\
\lambda(r_1 + o_1) + (1-\lambda)(r_2 + o_2) &\in \mathbf{R} + \mathbf{O} && (5)\\
\lambda(r_1 + o_1) + (1-\lambda)(r_2 + o_2) &\in \mathbf{H} && (6)\\
\lambda h_1 + (1-\lambda)h_2 &\in \mathbf{H} && (7)
\end{aligned}$$

Therefor $\mathbf{H}$ is a convex set, and hence the C-obstacle is a convex polygon.

Problem # 2

The "silhouette" of a c-obstacle (for the case of planar objects and robot) is the projection, along the orientation axis, of the c-obstacle boundary onto the x-y plane. Sketch an algorithm for computing the silhouette of a planar polygonal robot and obstacle.

Answer:

1. Begin with a robot orientation, $\theta = 0$.
2. For the chosen value of θ, define the set of points which define the robot and obstacle boundaries.
3. Calculate the Minkowski difference between the set of robot and obstacle boundary points.
4. Determine the convex hull of the set of points resulting from the Minkowski difference and store these points as $\mathbf{P}_i$
5. Repeat steps 2-4 for values of $\theta \in [0, 2\pi]$, incremented by $\Delta\theta$.
6. Calculate the convex hull over the set $\{\mathbf{P}_i\}$ and the result will be the silhouette.

Problem # 3

Consider the case of a 2-dimensional configuration space populated by planar polygonal c-obstacles. In such c-spaces, the arcs of the Voronoi diagram are locally of three types, depending on the two local polygon features which are equidistant from the Voronoi arc:

1. the two closest polygon features are an edge and an edge
2. the two closest polygon features are a vertex and an edge
3. the two closest polygon features are a vertex and vertex

Derive formulas for the Voronoi arcs for the three cases described above.

Answer:

Edge-Edge

For an edge-edge configuration, let the first edge be given by the following equation for a line:

$$Ax + By + C = 0 \tag{8}$$

and let the second edge similarly be given by:

$$Dx + Fy + E = 0 \tag{9}$$

Now let (x_o, y_o) be the location of the Voronoi arc. We know that the Voronoi arc must be equidistant from both edges, so let us begin by defining the distance between (x_o, y_o) and either edge:

$$d_1 = \frac{Ax_o + By_o + C}{\sqrt{A^2 + B^2}} \tag{10}$$

$$d_2 = \frac{Dx_o + Fy_o + E}{\sqrt{D^2 + F^2}} \tag{11}$$

Finally, setting $d_1 = d_2$, we arrive at:

$$\frac{Ax_o + By_o + C}{\sqrt{A^2 + B^2}} = \frac{Dx_o + Fy_o + E}{\sqrt{D^2 + F^2}} \tag{12}$$

which can be reduced into the following form (after some algebraic manipulation):

$$y_o = mx_o + b \tag{13}$$

which is the general equation of a line.

Vertex-Edge

For a vertex-edge configuration, let us begin as before and define the edge by the following equation for a line:

$$Ax + By + C = 0 \tag{14}$$

Let the vertex be given by the following point:

$$(x_p, y_p) \tag{15}$$

Now let (x_o, y_o) be the location of the Voronoi arc. We know that the Voronoi arc must be equidistant from both the edge and the vertex, so let us begin by defining the distance between (x_o, y_o) and the edge and vertex:

$$d_1 = \frac{Ax_o + By_o + C}{\sqrt{A^2 + B^2}} \tag{16}$$

$$d_2 = \sqrt{(x_p - x_o)^2 + (y_p - y_o)^2} \tag{17}$$

Setting $d_1 = d_2$, we get:

$$\frac{Ax_o + By_o + C}{\sqrt{A^2 + B^2}} = \sqrt{(x_p - x_o)^2 + (y_p - y_o)^2} \tag{18}$$

which can be reduced into the following form (after some algebraic manipulation):

$$(\alpha_0 x_o + \alpha_1 y_o)^2 + \alpha_2 x_o + \alpha_3 y_o + \alpha_4 = 0 \tag{19}$$

which is the general equation of a parabola.

Vertex-Vertex

For a vertex-vertex configuration, let us begin as before and define the two vertices as:

$$(x_q, y_q) \ \ and \ \ (x_p, y_p) \tag{20}$$

Now let (x_o, y_o) be the location of the Voronoi arc. We know that the Voronoi arc must be equidistant from both vertices, so let us begin by defining the distance between (x_o, y_o) and either vertex:

$$d_1 = \sqrt{(x_q - x_o)^2 + (y_q - y_o)^2} \tag{21}$$

$$d_2 = \sqrt{(x_p - x_o)^2 + (y_p - y_o)^2} \tag{22}$$

Setting $d_1 = d_2$, we get:

$$\sqrt{(x_q - x_o)^2 + (y_q - y_o)^2} = \sqrt{(x_p - x_o)^2 + (y_p - y_o)^2} \tag{23}$$

which can be reduced into the following form (after some algebraic manipulation):

$$y_o = mx_o + b \tag{24}$$

which is the general equation of a line.

Problem # 4

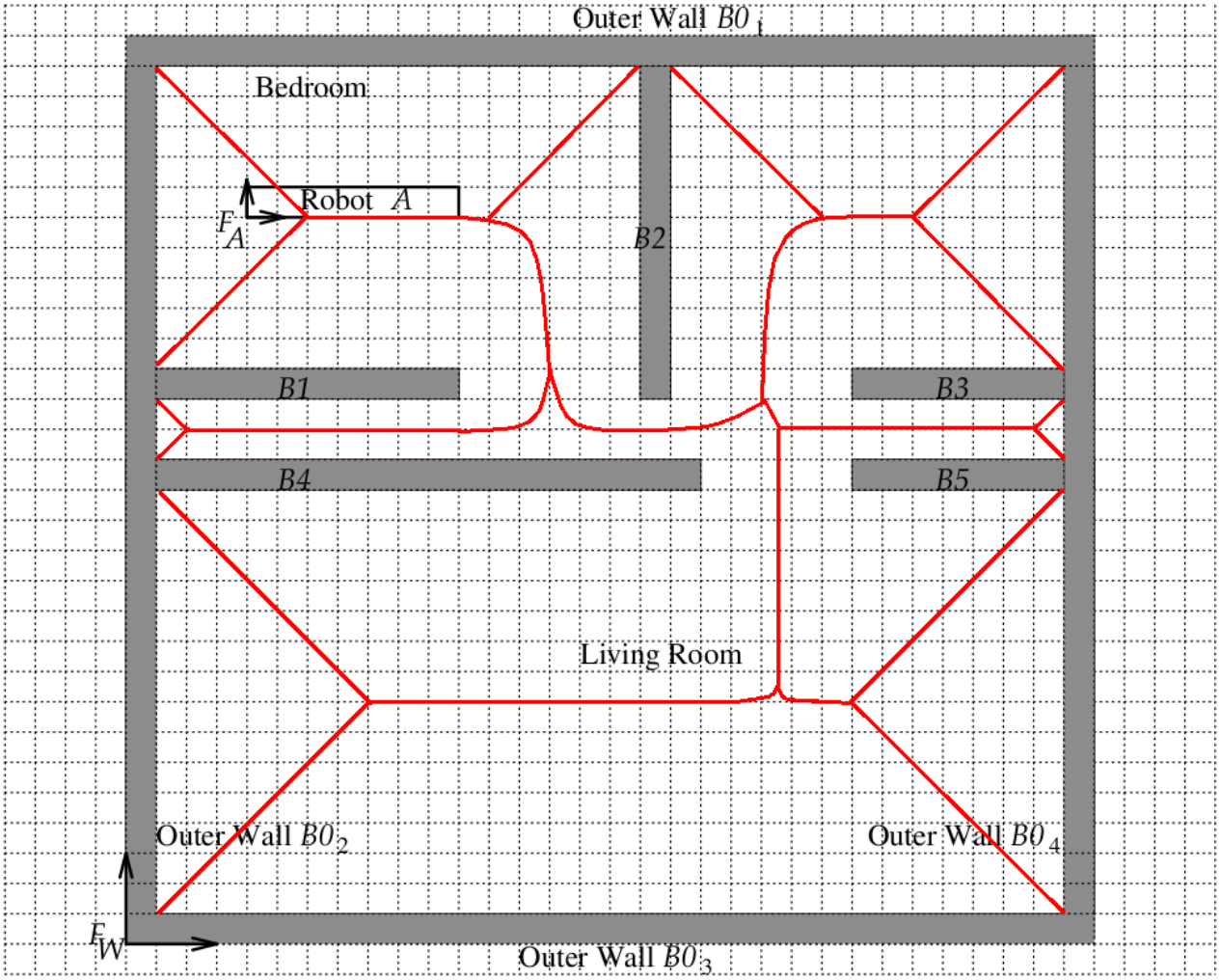

Figure 1: Voronoi diagram for the given workspace.

Problem # 5

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2: Print out of the workspace in the case that no cell has been visited. Instead of X's, 0's have been used and 1 represents the freespace.

```
% ----------------------------
% Problem 5 m-file
% ----------------------------
%
%====================
% Build the obstacles
%====================

% I chose to use 8 as the non-visited cell number just because it looks
% nicer in MATLAB; originally I tried -1 but it made the cells not line up
% because of the minus sign

M = 1*ones(30,32);

% Outer-Wall BO2
M(:,1) = 0;

% Outer-Wall BO1
M(1,:) = 0;

% Outer-Wall BO4
M(:,end) = 0;

% Outer-Wall BO3
M(end,:) = 0;

% Inner-Wall B1
M(12,1:11) = 0;

% Inner-Wall B2
M(1:12,18) = 0;

% Inner-Wall B3
M(12,25:end) = 0;

% Inner-Wall B4
M(15,1:19) = 0;

% Inner-Wall B5
M(15,25:end) = 0;

dlmwrite('problem5.txt',M,' ',10,10);
```

Problem # 6

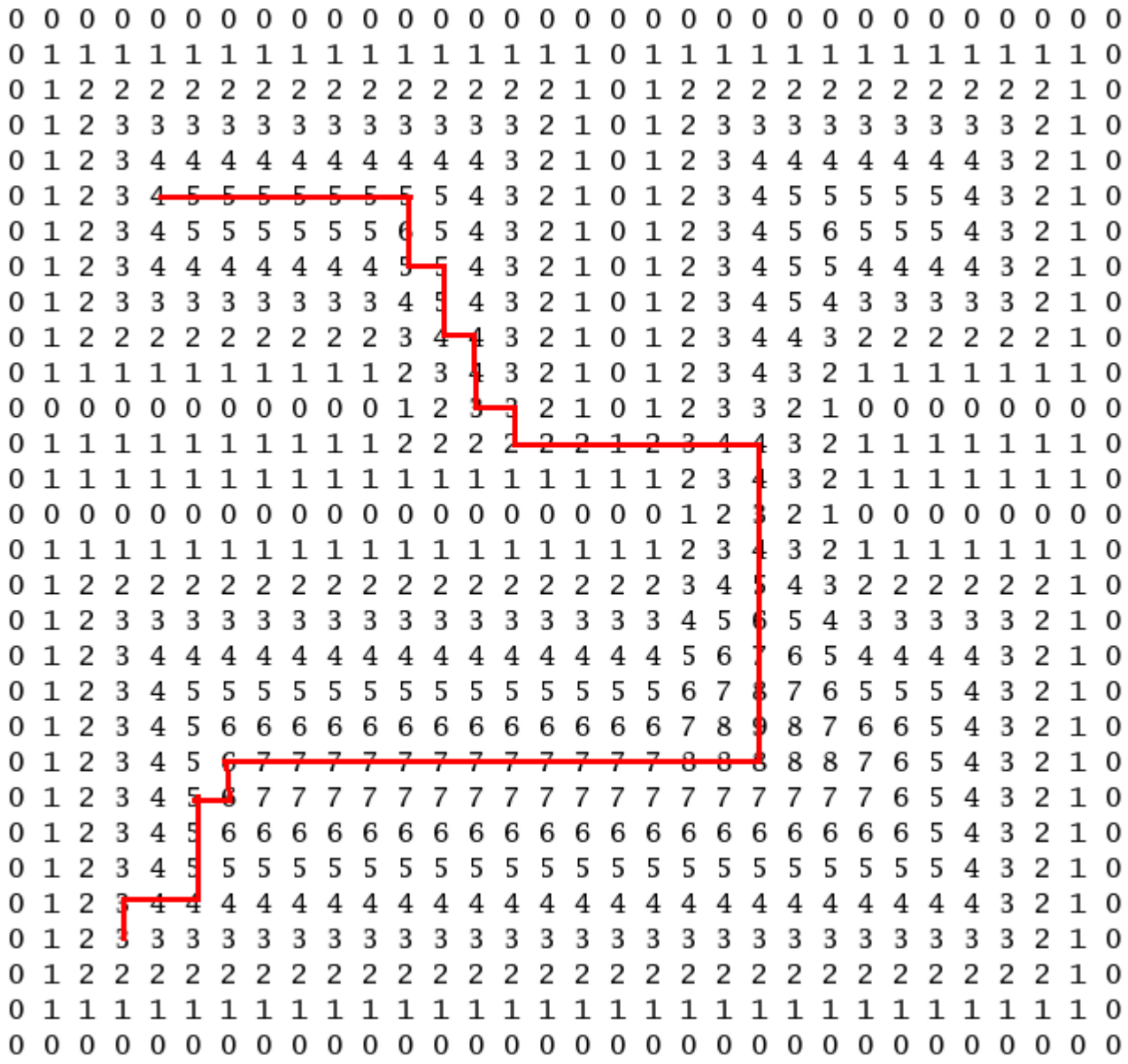

Figure 3: Approximated Voronoi Diagram with hand drawn path.

```
% ----------------------------
% Problem 6 m-file
% ----------------------------
%
%====================
% Build the obstacles
%====================

% I switched to using -1 for non-visited cells here because due to the
% large number of cells, it's quite possible that cells can reach integer
% values greater than or equal to 8 in the brushfire method; though no
% visited cells would ever have a a negative integer value

M = -1*ones(30,32);

% Outer-Wall BO2
M(:,1) = 0;

% Outer-Wall BO1
M(1,:) = 0;

% Outer-Wall BO4
M(:,end) = 0;

% Outer-Wall BO3
M(end,:) = 0;

% Inner-Wall B1
M(12,1:11) = 0;

% Inner-Wall B2
M(1:12,18) = 0;

% Inner-Wall B3
M(12,25:end) = 0;

% Inner-Wall B4
M(15,1:19) = 0;

% Inner-Wall B5
M(15,25:end) = 0;
```

```
%============================
% Now start Brushfire algorithm
%============================
NUMROWS = 30;
NUMCOLS = 32;
NUMITERS = 10;

for(k=1:NUMITERS)
    for(i=1:NUMROWS)
        for(j=1:NUMCOLS)
            if(M(i,j)<0)
                leftcell = M(i-1,j);
                rightcell = M(i+1,j);
                topcell = M(i,j+1);
                bottomcell = M(i,j-1);
                if(bottomcell==k-1 || topcell==k-1 || ...
                    rightcell==k-1 || leftcell==k-1)
                    M(i,j) = k;
                end
            end
        end
    end
end

dlmwrite('problem6.txt',M,' ',10,10);
```