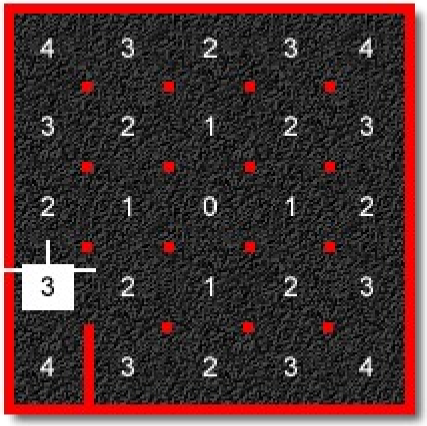
menu.gif

Introduction to the MicroMouse contest
[Contest Rules]
[Building a MicroMouse]
Suggestions on the software
Photos and video of MicroMouse robots

Software suggestions

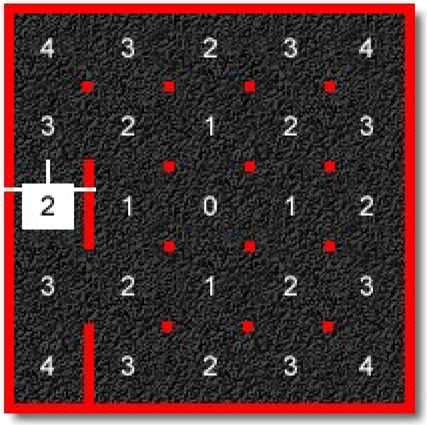
The modified flood-fill algorithm

The modified flood-fill algorithm is similar to the regular flood-fill algorithm in that the mouse uses distance values to move about the maze. The distance values, wich represent how far the mouse is from the destination cell, are followed in decending order until the mouse reaches its goal.



4	3	2	3	4
3	2	1	2	3
2	1	0	1	2
3	2	1	2	3
4	3	2	3	4

As the MicroMouse finds new walls during its exploration, the distance values need to be updated. Instead of flooding the entire maze with values, as is the case with the regular flood-fill, the modified flood-fill only changes those values which need to be changed. Let's say our mouse moves forward one cell and discovers a wall:

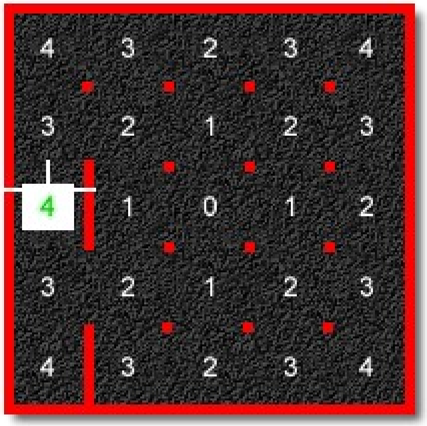


4	3	2	3	4
3	2	1	2	3
2	1	0	1	2
3	2	1	2	3
4	3	2	3	4

The robot cannot go West and it cannot go East, it can only travel North or South. But going North or South means going up in distance values which we do not want to do. So the cell values need to be updated. When we encounter this, we follow this rule:

If a cell is not the destination cell, its value should be
one plus the minimum value of its open neighbors.

In the example above, the minimum value of its open neighbors is 3. Adding 1 to this value results in 3 + 1 = 4. The maze now looks like this:



4	3	2	3	4
3	2	1	2	3
4	1	0	1	2
3	2	1	2	3
4	3	2	3	4

There are times when updating a cell's value will cause its neighbors to violate the "1 + minimum value" rule and so they must be checked as well. We can see in our example above that the cells to the North and to the South have neighbors whose minimum value is 2. Adding a 1 to this value results in 2 + 1 = 3 therefore the cells to the North and to the South do not violate the rule and the updating routine is done.

Now that the cell values have been updated, the mouse can once again follow the distance values in decending order.

So our modified flood-fill procedure for updating the distance values is:

Update the distance values (if necessary)

Make sure the stack is empty

Push the current cell (the one the robot is standing on) onto the stack

Repeat the following set of instructions until the stack is empty:

```
{
Pull a cell from the stack
Is the distance value of this cell = 1 + the minimum value of its open neighbors?
No -> Change the cell to 1 + the minimum value of its open neighbors and
push all of the cell's open neighbors onto the stack to be checked
Yes -> Do nothing
}
```

You can also see a 3 minute animated explanation of the algorithm.

Summary:

Most of the time, the modified flood-fill is faster than the regular flood-fill. By updating only those values which need to be updated, the mouse can make its next move much quicker.

Every time it arrives in a cell, have the mouse:

(1) Update the wall map

(2) Update the distance values (only if necessary)

(3) Decide which neighboring cell has the lowest distance value

(4) Move to the neighboring cell with the lowest distance value