One Possible Solution to "The Wandering Robot" Problem

1. IF the last action was not to rotate to the right, and the robot is not facing the finish point
   THEN ROTATE 1° LEFT
2. IF taking a step would not cause the robot to hit an obstacle or cross the boundary
   THEN STEP
3. IF taking a step would cause the robot to hit an obstacle or cross the boundary
   THEN ROTATE 1° RIGHT

Explanation:

The problem was broken down into 3 sub-problems:

1. Point the robot at the goal
2. Walk the robot toward the goal
3. Avoid obstacles and boundary

As shown in class, a simple first rule like:
   IF the robot is not facing the finish point
   THEN ROTATE 1° LEFT
would solve the first problem, and is contained in the solution's first rule.

If the robot is facing the finish point, then it would skip the first rule, so
   IF taking a step would not cause the robot to hit an obstacle or cross the boundary
   THEN STEP
will make the robot take steps up to an obstacle or the boundary.

If the robot is facing the finish point, then it would skip the first rule, and if an obstacle or the boundary is
immediately in the way, then it would skip the second rule, so the third rule must get it around the boundary. It
seems hard to get this in a single rule, so we subdivide the third sub-problem once again. In programming this
dividing problems into smaller and smaller sub-problems is called successive refinement, and is a very
important technique. Avoid obstacles and boundary could be divided into two sub-sub-problems:
   A. Point into a clear area as close to the obstacle or boundary as possible
   B. Step into the clear area

Now, sub-sub-problem A could be partly solved with rule 3 given in the solution:
   IF taking a step would cause the robot to hit an obstacle or cross the boundary
   THEN ROTATE 1° RIGHT
But it would only happen one time in a row with the simple rule:
   IF the robot is not facing the finish point
   THEN ROTATE 1° LEFT
That rule would immediately cause the robot to resume its previous orientation (pointing at the finish point),
destroying the effort to point into a clear area. So we modify the first rule so that when the robot is attempting to
point into a clear area, i.e. when it is rotating right, it cannot fire the first rule! Note that since we are using the
memory of a right rotation to identify the state of obstacle avoidance, we must only use right rotation there. Any
other rotations must be to the left!

We can now see that there is no need for an additional rule to handle sub-sub-problem B, because as soon as the
robot is pointed into a clear area, rule #2 will fire, pre-empting rule #3. This will cause a step, which will allow rule
#1 to fire as necessary on the next pass, to reorient the robot toward the goal.

Note that this solution is not necessarily the best solution possible. The robot tends to hunt for paths in a rightward
fashion. If a longer viable path was to its right, and a shorter viable path to its left, the robot would take the longer
one!