Calvin Pugmire
CS 470, section 001

Programming Assignment (Bayesian Filtering)

```
// TODO: update the probabilities of where the AI thinks it is based on the action selected and the new sonar readings
//       To do this, you should update the 2D-array "probs"
// Note: sonars is a bit string with four characters, specifying the sonar reading in the direction of North, South, East, and West
//       Ex: the sonar string 1001, specifies that the sonars found a wall in the North and West directions, but not in the South and East directions
void updateProbabilities(int action, String sonars) {
   // your code

   double[][] newprobs = new double[mundo.width][mundo.height];
   double newprobssum = 0.0;
   for (int i = 1; i < mundo.width-1; i++) {
      for (int j = 1; j < mundo.height-1; j++) {
         if (mundo.grid[i][j] == 1) {
            newprobs[i][j] = 0;
         } else {
            newprobs[i][j] = sensorProbability(sonars, i, j) * transitionProbability(action, i, j);
         }
         newprobssum += newprobs[i][j];
      }
   }

   for (int i = 0; i < mundo.width; i++) {
      for (int j = 0; j < mundo.height; j++) {
         probs[i][j] = newprobs[i][j] / newprobssum;
      }
   }

   myMaps.updateProbs(probs); // call this function after updating your probabilities so that the
   //  new probabilities will show up in the probability map on the GUI
}

double transitionProbability(int action, int i, int j) {
   double newprob = 0.0;
   for (int k = 0; k < mundo.width; k++) {
      for (int l = 0; l < mundo.height; l++) {
         if ((Math.abs(i - k) < 2 && j - l == 0) || (i - k == 0 && Math.abs(j - l) < 2)) {
            if (action == EAST && i == k + 1) {
               newprob += moveProb * probs[k][l];
            } else if (action == WEST && i == k - 1) {
               newprob += moveProb * probs[k][l];
            } else if (action == NORTH && j == l - 1) {
               newprob += moveProb * probs[k][l];
            } else if (action == SOUTH && j == l + 1) {
               newprob += moveProb * probs[k][l];
            } else if (action == STAY && i == k && j == l) {
               newprob += moveProb * probs[k][l];
            } else {
               newprob += ((1 - moveProb) / 4) * probs[k][l];
            }
```

```
            }
            //newprob += (probability of current=[i,j] given action and last=[k,l])*(probs[k][l])
        }
    }
    return newprob;
}

double sensorProbability(String sonars, int i, int j) {
    int[] sensors = new int[sonars.length()];
    for (int o = 0; o < sonars.length(); o++) {
        sensors[o] = Character.getNumericValue(sonars.charAt(o));
    }

    int[] expects = new int[sonars.length()];
    if (mundo.grid[i][j - 1] == 1) {
        expects[0] = 1;
    } else {
        expects[0] = 0;
    }
    if (mundo.grid[i][j + 1] == 1) {
        expects[1] = 1;
    } else {
        expects[1] = 0;
    }
    if (mundo.grid[i + 1][j] == 1) {
        expects[2] = 1;
    } else {
        expects[2] = 0;
    }
    if (mundo.grid[i - 1][j] == 1) {
        expects[3] = 1;
    } else {
        expects[3] = 0;
    }

    int m = 0;
    int n = 0;
    for (int o = 0; o < sonars.length(); o++) {
        if (sensors[o] == expects[o]) {
            m += 1;
        } else {
            n += 1;
        }
    }

    return (Math.pow(sensorAccuracy,m))*(Math.pow((1-sensorAccuracy),n));
    //senseprob = probability of sonars given current=[i,j]
}
```

Everything was clear.