Robot Simulation of Clique Behavior

Clique-072.jpg

Carla Galarza  MissyGg15@yahoo.com
Sonia Luna  Luna_s46@hotmail.com
Christina Rodriguez  dreamangel1680@hotmail.com
Alejandra Estevez  Mementomori00@aol.com

The Young Women’s Leadership School
105 East 106th Street
New York, NY 10029

Mr. David Lewine, Teacher/Advisor
(917) 628-8627
Davlew@mac.com
# Table of Contents

I. Abstract............................................................................................................... 3

II. Introduction/Background............................................................................... 3

III. Hypothesis....................................................................................................... 4

IV. Method............................................................................................................. 5

  A. Anatomy of Robot...................................................................... 5

  B. Program........................................................................................ 6

  C. Testing.......................................................................................... 9

  D. Analyzing Cliques.................................................................... 10

V. Results.............................................................................................................. 11

VI. Conclusion..................................................................................................... 15

VII. Bibliography................................................................................................ 17

Appendix A: Data Collected................................................................................. 18

Appendix B: RunManyPortsx........................................................................... 21

Appendix C: 1Colors_50Robots.world............................................................. 22

Appendix D: RobotProgram.............................................................................. 25
I. Abstract

Our team researched emergent behavior that would result due to simple algorithms in a set of simulated robots. Our main focus was on how cliques emerged and what factors can influence each robot, such as color. In our experiment we simulated a population of 50 robots interacting in a bounded world using the player/stage software. In our experiment we varied the number of colors of robots. This led to very interesting results, we found that the more colors of robots there were, the fewer and larger the cliques became. These results can be interpreted as suggesting that diversity or heterogeneity leads to greater order and complexity. Our results can be applied to many different situations, including human social behavior, which is what initially inspired our interest in this experiment.

II. Introduction/Background

What exactly is it that leads to cliques? What factors cause for people to come together? Cliques are part of society in one way or another. This is where our curiosity and interest for this experiment arose. We wondered what would result if we programmed robots with simple algorithms - Would cliques emerge? How would the robots react to their surroundings and the world we programmed for them? What behavior would result? Would we notice the same behavior as in real life, virtual realities, or other such experiments?

We were also influenced by other experiments involving emergence. Some of the experiments we found included insects and moths as agents. [2, 3] In the project involving the moths, the moths flew into any light source that was around them. People may think that the moth is following complex goal-oriented thinking. But in reality the moth flying into the light source is a behavior that emerges from the moth following simple rules. The moths rule is to fly in a straight line, yet still be able to see their light source. To do this though, the moths would have to keep rotating themselves as they move straight ahead to remain at the same angle to the light source. Basically what the moths are doing is turning towards the light gradually. The way they turn makes them fly in the spiral path towards the light, eventually they would follow their simple rule into the light, making it look like this was their goal all along.

Another really interesting project that influenced the idea for our experiment was the work of Craig Reynolds in his project “Boids.” [4] This experiment modeled the flocking behavior of birds. In this project the robots were also following a simple algorithm, based on the information about the robots nearest them. Eventually, by all following the same rules, the robots moved similarly to birds flocking. The robots seemed, just like birds, that they were organized.

We decided to conduct an experiment similar to Mr. Reynolds, where we would see if individual robots following simple rules could result in organized behavior on the part of the whole population. Our robots though, would be replicating a student body dividing into cliques. Robotics is an excellent way to conduct experiments testing real life ideas since robots are easy to give simple instructions to. We chose this experiment because we wanted to investigate something that we ourselves experienced: Cliques. Cliques are apparent throughout any society, but most importantly amongst students in high school, something we experience commonly.
Originally we were curious to see whether our simple algorithms would lead to any clique behavior between the robots. Our initial goal was to plot the final positions of the robots on a coordinate system to judge whether or not cliques were formed. Afterwards we would ask how other people defined cliques, as a way to better judge our results. However, very early into our experiment we found that cliques did form, there was no question about it, and they also formed rather quickly. Our focus then changed to, not whether cliques would form, and how to judge if they had, but rather how they formed and what exactly influenced they way that the formed?

We decided that we could focus on the way that the color of each individual robot would affect the formation if cliques. The robots would avoid or be attracted to other robots based on their color. We chose color because it is a distinct trait that students also see in our diverse school but which could stand for any reason of attraction or avoidance such as: interests, personality, or looks. We formed a new hypothesis were we stated that the color of the robots would affect how the cliques formed. We made a prediction that the fewer colors used (many of the robots with the same color) would result in fewer and bigger cliques. But just as before, with fast forming clique behavior, our results turned out different that expected.

III. Hypothesis

When first beginning our research and experiment, our simple hypothesis questioned if any cliques or group-like behavior would form at all from the interactions of our robots. Each robot would follow a simple algorithm, in which the robots either ignore or become attracted to the other robots surrounding it, based on color. But during our experiment we found that cliques did form quickly. Our attention was then brought to another question: How and why cliques would form. We decided to focus on the effect of the number of colors of robots in clique formation. Our hypothesis then became, that colors matter and that more colors create more cliques.
IV. Method

A. Anatomy of Robot

Our robots are fairly simple. They have a camera that is on a pan-tilt-zoom, two sonar sensors, one on the left and one on the right, a laser and a bumper. For our experiments we mainly used the camera and the two sonar sensors. It is an octagon. The following lines of program describe one of our robots, an instance of the rover2dwc class of robots:

rover2dwc
  (  
    pose [-9.640 -0.020 360.000]  
    color "red"  
    name "sebastian"  
    sick_laser( samples 361 laser_sample_skip 4 )  
  )

The pose describes where the robot is placed in the world and what direction it faces. The world is on a coordinate grid, where each robot has a set of coordinate points; these are the first two digits in the brackets. The last digit in the brackets defines the angle the robot faces. The next line defines the color and in the next line the robot is given a name. Each robot has to have a different name. The last line defines the laser. In this experiment we did not use the laser. The Rover2dwc class is defined as follows. The definition specifies the size, shape and features (bumper, camera, pan-tilt-zoom, etc.) of robots of that class.

define rover2dwc position
  (  
    # actual size  
    size [.3 .3]  
    # the rover's center of rotation is offset from its center of area  
    origin [-0.04 0.0 0]  
    # draw a nose on the robot so we can see which way it points  
    gui_nose 1  
    # estimated mass in KG
mass 1.0

# this polygon approximates the shape of a pioneer
polygons 1
polygon[0].points 8
polygon[0].point[0] [ 0.23  0.05 ]
polygon[0].point[1] [ 0.15  0.15 ]
polygon[0].point[2] [-0.15  0.15 ]
polygon[0].point[3] [-0.23  0.05 ]
polygon[0].point[4] [-0.23 -0.05 ]
polygon[0].point[5] [-0.15 -0.15 ]
polygon[0].point[6] [ 0.15 -0.15 ]
polygon[0].point[7] [ 0.23 -0.05 ]

#bumper( bcount 1
#        blength 0.4
#        bpose[0] [.5 0  0]
#        bpose[1] [0 0.165 -90]
#        bpose[2] [-0.26  0 0]
#        #blength[2] 0.1 # set the length of a single bumper
#

rover_sonar()

   ptz(
      blobfinder(
         channel_count 10
         channels [ "red" "blue" "green" "cyan" "yellow" "magenta" "orange" "gray" "pink" "purple"]
      )
   )
)

B. Program

In order to begin our experiment we had to enable player/stage using a Linux operating system with Fedora. Player/stage is a simulation program in which we could create a world that our robots could interact in. Player/stage works with a world file and configuration file but is able to work with other files. The world file defines our robots and the world they interact in, while the configuration file links communication ports to specific robots.

Our robots interact in a two dimensional world with no obstacles bordered with solid white lines that form a rectangle. The dimensions of our world are 285 mm by 175 mm; the robots are 5mm by 5 mm. The dimensions we chose for the world are big enough for 50 robots to move around in but small enough to ensure interaction between the robots.
In using these programs we could put as many robots as we wanted to. For this experiment we used 50 robots in each trial. In our world file we could decide on the robot’s color, starting position and name. The robots were initially spread out evenly throughout the world. For every trial the robots started in the same position. (See Figure 2)

A separate program, “RobotProgram,” chooses the robot’s color preference randomly every time we run the program. A random generator picks a number one or two. Each robot has a fifty percent chance of liking or disliking a color. If a one is chosen the robot will have a preference toward that color and will move toward it. If a two is chosen the robot will not like that color and will avoid it. The same program is in charge of deciding how and where the robot moves. The robot looks at the biggest blob in the field of view. The biggest blob is usually the closest robot. If the robot likes that color it will move toward it and if it does not the robot will move away from it. If the robot does not find the color it wants it will continue to search randomly. Note that a robot’s color has no effect on its color preference.

“RobotProgram” has the structure of a Read-Think-Act loop. First the robot reads the sonar sensors and the camera. The robot then decides what to do regarding the biggest blob in its camera. What the robot does depends on the color and position of the biggest blob. If the robot likes that color it will move toward it, but if it does not the robot will move away from it. (See Appendix D for full programming code of “Robot Program”)

![Figure 2: Starting Positions](image)
Figure 3 demonstrates the pink and blue robots reaching a state of equilibrium. The robot is reacting to the biggest blob in its field of view, which is the color it likes. Since the blob is centered they have no reason to turn away. The two robots are close enough to each other so that their velocity is zero and they have stopped. We can tell that the two have reached a state of equilibrium because their cameras are facing each other. They are within the minimum distance of each other and have stopped.

Figure 4 is demonstrating the process in which a robot “makes the decision.” The magenta robot is looking at both the blue, green and magenta robots but is reacting to the green robot because it is the biggest blob in its field of view. It will turn and move towards the green robot if it has color preference for green is positive. It will turn away if the colors robot preference for green is negative.
C. Testing

In order to collect our data we decided to run 100 trials. We ran 25 trials with 1 color, 25 trials with 2 colors, 25 trials with 5 colors and 25 trials with 10 colors. We ran each trial for approximately 7 minutes using a cell phone stop watch. We observed the robots during their 7 minute trial.

For each trial we used a data collecting sheet to organize our data. (See figure 5) Using the sheet we recorded the file name, number of cliques, largest and smallest clique number, average clique size and number of pairs and solos. In our data sheets we did not count pairs as cliques, but in the graph tables we incorporated the pairs as cliques to obtain a better correlation in our results.

Running 100 trials was time consuming. Unfortunately, we only had one computer for most of the experiment in which we could install the Linux operating system. For each trial, we had to issue the “RobotProgram” command 50 times, once for each robot. The command “RobotProgram” is given together with the port number of the robots to be controlled. Each robot has its own port number. We computerized this procedure by writing a Linux Shell Program, “RunManyPorts” that would deliver the “RobotProgram” commands automatically, 50 times. This loop program runs “RobotProgram” over and over again for the different ports. (See Appendix B for programming code of “RunManyPorts”)

D. Analyzing Cliques

<table>
<thead>
<tr>
<th>File name: ______________________</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total # of robots:</strong> _________</td>
</tr>
<tr>
<td># of red robots: _________</td>
</tr>
<tr>
<td># of purple robots: _________</td>
</tr>
<tr>
<td># of pink robots: _________</td>
</tr>
<tr>
<td># of gray robots: _________</td>
</tr>
<tr>
<td># of green robots: _________</td>
</tr>
<tr>
<td># of yellow robots: _________</td>
</tr>
<tr>
<td># of orange robots: _________</td>
</tr>
<tr>
<td># of blue robots: _________</td>
</tr>
<tr>
<td># of cyan robots: _________</td>
</tr>
<tr>
<td># of magenta robots: _________</td>
</tr>
</tbody>
</table>

| **Number of cliques:** __________ |
| # of robots in largest clique: ___|
| # of robots in smallest clique: ___|
| Average size of cliques: ________ |

| **Number of pairs:** __________  |
| Number of solos: __________     |

| Elapsed time: __________        |

Figure 5: Data Collecting Sheet
It was a difficult task when it came to decide which robots had accomplished a clique formation. Sometimes there were many robots together in the same area. This was hard to decipher. We defined a clique as a group of 3 or more robots. Solos and partners were counted in a different category. A clique has to be a cluster of robots that are connected in some way.

We found that sometimes the robots formed trains and other times the robots were looking at each other. In order to determine whether a robot is part of a clique that we were unsure of we had to look at the position of the robots camera i.e. the direction the robot was looking. For a robot to be part of a clique, it must be within the minimum distance of the other robots and the robot has to be looking at a robot in the clique or must be looked at by a robot in the clique in order to be considered part of the clique. Below are some examples on how we divided cliques.

In figure 6 above, the pair in the lower right is counted as such and not as part of the large clique next to it because even though one of the robots is facing the bigger clique it is too far away to be associated with it.

**Figure 6: Clique-008.jpg**

5 colors (Purple, Pink, Green, Gray, and Red)

Data Collected:
- Number of Cliques: 6
- Largest Clique: 22
- Smallest Clique: 3
- Average size: 7.8
- Number of Pairs: 1
- Number of Solos: 1
V. Results

We analyzed our data by using an Analysis of Variance (ANOVA) test. This test compares the variance of the group averages to the variance of each group. We decided to use this test because we needed to compare the averages of our four groups of data: 1 color, 2 colors, 5 colors and 10 colors. We compared the averages of the number of cliques, the size of the largest clique, the number of pairs, the number of solos and the average size of cliques. We found that there was a statistically significant correlation between the number of colors and all of these variables. More specifically the number of colors was negatively correlated with the number of pairs, the number of cliques and the number of solos, and positively correlated with the average size of cliques and the size of the largest clique. In other words as the number of colors increased there were fewer but larger cliques.

Figure 7: Clique-052.jpg
10 colors (Red, Purple, Blue, Pink, Orange, Gray, Yellow, Magenta, Green, Cyan)
Data Collected:
   Number of Cliques: 6
   Largest Clique: 13
   Smallest Clique: 4
   Average size: 7.2
   Number of Pairs: 3
   Number of Solos: 1

In figure 7 above, the solo at the top is within the minimum distance of the clique below it. None the less, it is not counted as part of that clique because it is facing away from that clique and none of the robots in that clique are facing it.
Test of all cases

Analysis of Variance

Response attribute (numeric): pairs
Grouping attribute (categorical): number_of_colors

Alternative hypothesis: The population means of pairs grouped by number_of_colors are not equal

If it were true that the population means of pairs were equal (the null hypothesis) and the sampling process were performed repeatedly, the probability of getting a value for F greater than or equal to the observed value of 12.6152 would be < 0.0001.
Number of Solos

<table>
<thead>
<tr>
<th>number_of_colors</th>
<th>1</th>
<th>2</th>
<th>5</th>
<th>10</th>
<th>2.85</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.52</td>
<td>3.56</td>
<td>0.92</td>
<td>0.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.24191</td>
<td>1.95959</td>
<td>0.996661</td>
<td>0.57735</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.648383</td>
<td>0.391918</td>
<td>0.199332</td>
<td>0.11547</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S1 = mean ( )
S2 = stdDev ( )
S3 = stdError ( )

Test of all cases
Response attribute (numeric): solos
Grouping attribute (categorical): number_of_colors

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>SS</th>
<th>MS</th>
<th>F</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Groups</td>
<td>3</td>
<td>592.510</td>
<td>197.503</td>
<td>50.394</td>
<td>0.0000</td>
</tr>
<tr>
<td>Error</td>
<td>96</td>
<td>376.240</td>
<td>3.919</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>99</td>
<td>968.750</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ho: Population means of solos grouped by number_of_colors are equal
**Size of Largest Clique**

### Number of Clique

**Box Plot**

**Test of all cases**

**Analysis of Variance**

<table>
<thead>
<tr>
<th>Source</th>
<th>Degrees of Freedom</th>
<th>Sum of Squares</th>
<th>Mean Square</th>
<th>F</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Groups</td>
<td>3</td>
<td>161.84</td>
<td>53.9467</td>
<td>13.877</td>
<td>0.0000</td>
</tr>
<tr>
<td>Error</td>
<td>96</td>
<td>373.200</td>
<td>3.8875</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>99</td>
<td>535.040</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Alternative hypothesis: The population means of number_of_cliques grouped by number_of_colors are not equal.

If it were true that the population means of number_of_cliques were equal (the null hypothesis) and the sampling process were performed repeatedly, the probability of getting a value for F greater than or equal to the observed value of 13.877 would be < 0.0001.
Average Size of Clique

**Response attribute (numeric):** Avg_Size

**Grouping attribute (categorical):** number_of_colors

Alternative hypothesis: The population means of Avg_Size grouped by number_of_colors are not equal.

If it were true that the population means of Avg_Size were equal (the null hypothesis) and the sampling process were performed repeatedly, the probability of getting a value for F greater than or equal to the observed value of 10.5257 would be < 0.0001.
VI. Conclusion

For this robotics simulation we did many trials. We noticed that the amount of colors of the robots in the world affected the clique behavior greatly. The more colors we used in our simulation led to larger cliques; with fewer colors for the robots we had more and smaller cliques. Using the definition of complexity and order related to the concept of entropy, we can say that a state of our robot system that has fewer and larger cliques is a state that is more complex and ordered than one that has many small cliques.[1,5] Thus our experiment suggests that an increase in diversity in the population of the robots led to a more complex, and more ordered structure. This was an interesting result because we felt that with more colors there would be less integration among the robots. As we saw with the one color simulation though, there were many small cliques and less unification among the robots. Below is an example of a 2 color trial with many scattered solos and cliques and an example of a 10 color trial showing a much more highly organized structure with fewer groupings.

![Figure 8: Clique-048.jpg](image)

2 colors (Red and Purple)
Data Collected:
- Number of Cliques: 7
- Largest Clique: 14
- Smallest Clique: 3
- Average size: 5.9
- Number of Pairs: 1
- Number of Solos: 7
Our experiment and the analytical break down of its results can be applied to many other sciences such as: chemistry, physics, biology, and the social sciences. The social sciences are where we focused on applying our results. Since the robots were created to copy clique behavior of humans, we wanted to think about how it relates to humanistic complexities. We could compare the robot structures to cities and small towns. In cities there are many people, and lots of diversity, so it may seem chaotic; in reality though a city has much more structure and order than many small towns. The small towns have people dispersed all around in smaller groups, and do not have as much complexity.

Our experiment suggests that a diverse population helps promote large complex social structures such as cities. This means if you value diversity on the one hand and large complex
unified social structures like cities on the other hand, you do not have to worry about those two values conflicting. You do not have to plan for how to prevent diversity from fragmenting society. Rather, you have to plan for how to remove the obstacles, such as ethnocentricity, that get in the way of the unifying effect of diversity.

But to really understand the connection between diversity and complexity that our experiment suggests we must look at what would happen if we varied factors other than colors. There were many factors that could affect the type of clique formations we saw. The field of view of the camera on the robots limited the sight of the rest of the environment around the robots: they could not look behind them, or have an all around field of view. The robots only reacted to the closest robot. Thus changing the field of view or changing how the robot reacts to the nearest robots may change our results. We would like to run additional trials changing these variables to see if the number of colors continues to lead to more complex structures.

Another variation we are anxious to try is to see what would happen if the robots did not have a color picked out as randomly as we set it up. There could be a test where the robot has a seventy-five percent chance of liking its own color, and then a twenty-five percent chance of liking another color. This could be interpreted as ethnocentricity, and we might not have results the same as before.

This idea of using robot simulation to study human behaviors is relatively new. Since it is hard to study humans and get truthful results, robot simulation is a great advancement.

VII. Bibliography


## Appendix A: Data Collected

### One Color

<table>
<thead>
<tr>
<th>File Name</th>
<th># of Cliques Greater Than 3</th>
<th># of Cliques (including pairs)</th>
<th>Largest Clique</th>
<th>Smallest Clique</th>
<th>Avg. Size</th>
<th># of pairs</th>
<th># of solos</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clique-076.jpg</td>
<td>5</td>
<td>10</td>
<td>10</td>
<td>3</td>
<td>5.6</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>Clique-077.jpg</td>
<td>8</td>
<td>10</td>
<td>8</td>
<td>3</td>
<td>5.2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>Clique-078.jpg</td>
<td>6</td>
<td>8</td>
<td>14</td>
<td>4</td>
<td>6.8</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Clique-079.jpg</td>
<td>3</td>
<td>10</td>
<td>14</td>
<td>3</td>
<td>7.7</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Clique-080.jpg</td>
<td>5</td>
<td>8</td>
<td>15</td>
<td>3</td>
<td>6.2</td>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>Clique-081.jpg</td>
<td>8</td>
<td>10</td>
<td>8</td>
<td>3</td>
<td>5.4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Clique-082.jpg</td>
<td>8</td>
<td>12</td>
<td>6</td>
<td>3</td>
<td>4.3</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Clique-083.jpg</td>
<td>7</td>
<td>10</td>
<td>7</td>
<td>3</td>
<td>4.7</td>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>Clique-084.jpg</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>3</td>
<td>5.3</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>Clique-085.jpg</td>
<td>8</td>
<td>14</td>
<td>6</td>
<td>3</td>
<td>3.8</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Clique-086.jpg</td>
<td>5</td>
<td>8</td>
<td>11</td>
<td>3</td>
<td>8.2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Clique-087.jpg</td>
<td>4</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>7.3</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>Clique-088.jpg</td>
<td>6</td>
<td>10</td>
<td>12</td>
<td>3</td>
<td>6.4</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Clique-089.jpg</td>
<td>5</td>
<td>11</td>
<td>10</td>
<td>3</td>
<td>6.4</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Clique-090.jpg</td>
<td>6</td>
<td>11</td>
<td>11</td>
<td>3</td>
<td>6.2</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Clique-091.jpg</td>
<td>5</td>
<td>8</td>
<td>22</td>
<td>3</td>
<td>7.6</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Clique-092.jpg</td>
<td>9</td>
<td>12</td>
<td>8</td>
<td>3</td>
<td>4.6</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Clique-093.jpg</td>
<td>7</td>
<td>13</td>
<td>7</td>
<td>3</td>
<td>4.6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Clique-094.jpg</td>
<td>6</td>
<td>12</td>
<td>8</td>
<td>4</td>
<td>5.6</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>Clique-095.jpg</td>
<td>5</td>
<td>7</td>
<td>20</td>
<td>3</td>
<td>8</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>Clique-096.jpg</td>
<td>4</td>
<td>11</td>
<td>8</td>
<td>3</td>
<td>6.3</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>Clique-097.jpg</td>
<td>9</td>
<td>10</td>
<td>9</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Clique-098.jpg</td>
<td>5</td>
<td>10</td>
<td>14</td>
<td>3</td>
<td>7.6</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Clique-099.jpg</td>
<td>5</td>
<td>9</td>
<td>8</td>
<td>4</td>
<td>5.8</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>Clique-100.jpg</td>
<td>9</td>
<td>12</td>
<td>6</td>
<td>3</td>
<td>4.5</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

### Two Colors

<table>
<thead>
<tr>
<th>File Name</th>
<th># of Cliques Greater Than 3</th>
<th># of Cliques (including pairs)</th>
<th>Largest Clique</th>
<th>Smallest Clique</th>
<th>Avg. Size</th>
<th># of pairs</th>
<th># of solos</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clique-026.jpg</td>
<td>5</td>
<td>10</td>
<td>18</td>
<td>3</td>
<td>7.4</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Clique-027.jpg</td>
<td>7</td>
<td>9</td>
<td>12</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Clique-028.jpg</td>
<td>5</td>
<td>8</td>
<td>18</td>
<td>3</td>
<td>7.6</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Clique-029.jpg</td>
<td>6</td>
<td>10</td>
<td>12</td>
<td>3</td>
<td>6.3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Clique-030.jpg</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>3</td>
<td>5.8</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Clique-031.jpg</td>
<td>6</td>
<td>9</td>
<td>12</td>
<td>3</td>
<td>6.8</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Clique-032.jpg</td>
<td>6</td>
<td>8</td>
<td>16</td>
<td>3</td>
<td>7.2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Clique-033.jpg</td>
<td>5</td>
<td>7</td>
<td>16</td>
<td>3</td>
<td>8.6</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>File Name</td>
<td># of Cliques Greater Than 3</td>
<td># of Cliques (including pairs)</td>
<td>Largest Clique</td>
<td>Smallest Clique</td>
<td>Avg. Size</td>
<td># of pairs</td>
<td># of solos</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------</td>
<td>--------------------------------</td>
<td>----------------</td>
<td>-----------------</td>
<td>-----------</td>
<td>------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Clique-001.jpg</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>4</td>
<td>5.9</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-002.jpg</td>
<td>5</td>
<td>5</td>
<td>25</td>
<td>6</td>
<td>10.2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Clique-003.jpg</td>
<td>6</td>
<td>7</td>
<td>12</td>
<td>4</td>
<td>7.8</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-004.jpg</td>
<td>4</td>
<td>5</td>
<td>33</td>
<td>3</td>
<td>12.3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-005.jpg</td>
<td>6</td>
<td>7</td>
<td>17</td>
<td>3</td>
<td>7.8</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-006.jpg</td>
<td>6</td>
<td>9</td>
<td>20</td>
<td>3</td>
<td>7.2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-007.jpg</td>
<td>4</td>
<td>4</td>
<td>37</td>
<td>3</td>
<td>12.5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Clique-008.jpg</td>
<td>6</td>
<td>7</td>
<td>22</td>
<td>3</td>
<td>7.8</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-009.jpg</td>
<td>6</td>
<td>8</td>
<td>14</td>
<td>4</td>
<td>7.6</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Clique-010.jpg</td>
<td>6</td>
<td>6</td>
<td>19</td>
<td>4</td>
<td>8.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Clique-011.jpg</td>
<td>6</td>
<td>9</td>
<td>21</td>
<td>3</td>
<td>6.8</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Clique-012.jpg</td>
<td>7</td>
<td>10</td>
<td>13</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Clique-013.jpg</td>
<td>6</td>
<td>8</td>
<td>13</td>
<td>4</td>
<td>7.8</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Clique-014.jpg</td>
<td>5</td>
<td>8</td>
<td>14</td>
<td>4</td>
<td>8.2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Clique-015.jpg</td>
<td>5</td>
<td>6</td>
<td>17</td>
<td>3</td>
<td>9.6</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-016.jpg</td>
<td>6</td>
<td>9</td>
<td>18</td>
<td>3</td>
<td>7.3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Clique-017.jpg</td>
<td>7</td>
<td>8</td>
<td>15</td>
<td>3</td>
<td>6.6</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-018.jpg</td>
<td>6</td>
<td>7</td>
<td>13</td>
<td>4</td>
<td>6.3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-019.jpg</td>
<td>4</td>
<td>5</td>
<td>16</td>
<td>8</td>
<td>12</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-020.jpg</td>
<td>7</td>
<td>14</td>
<td>17</td>
<td>3</td>
<td>6.7</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>Clique-021.jpg</td>
<td>7</td>
<td>13</td>
<td>14</td>
<td>3</td>
<td>6</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Clique-022.jpg</td>
<td>5</td>
<td>7</td>
<td>14</td>
<td>5</td>
<td>8.8</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Clique-023.jpg</td>
<td>3</td>
<td>9</td>
<td>19</td>
<td>7</td>
<td>12.6</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>Clique-024.jpg</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Clique-025.jpg</td>
<td>4</td>
<td>4</td>
<td>28</td>
<td>4</td>
<td>12.3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>File Name</td>
<td># of Cliques Greater Than 3</td>
<td># of Cliques (including pairs)</td>
<td>Largest Clique</td>
<td>Smallest Clique</td>
<td>Avg. Size</td>
<td># of pairs</td>
<td># of solos</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------</td>
<td>---------------------------------</td>
<td>----------------</td>
<td>-----------------</td>
<td>-----------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>Clique-051.jpg</td>
<td>8</td>
<td>11</td>
<td>10</td>
<td>4</td>
<td>5.5</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Clique-052.jpg</td>
<td>6</td>
<td>9</td>
<td>13</td>
<td>4</td>
<td>7.2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Clique-053.jpg</td>
<td>7</td>
<td>9</td>
<td>10</td>
<td>4</td>
<td>6.4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Clique-054.jpg</td>
<td>6</td>
<td>7</td>
<td>15</td>
<td>3</td>
<td>8</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-055.jpg</td>
<td>7</td>
<td>8</td>
<td>12</td>
<td>4</td>
<td>6.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-056.jpg</td>
<td>5</td>
<td>6</td>
<td>24</td>
<td>3</td>
<td>9.4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-057.jpg</td>
<td>6</td>
<td>6</td>
<td>20</td>
<td>3</td>
<td>8.2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Clique-058.jpg</td>
<td>5</td>
<td>5</td>
<td>23</td>
<td>4</td>
<td>9.2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Clique-059.jpg</td>
<td>6</td>
<td>7</td>
<td>17</td>
<td>3</td>
<td>8</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-060.jpg</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>4</td>
<td>6.6</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Clique-061.jpg</td>
<td>7</td>
<td>9</td>
<td>13</td>
<td>3</td>
<td>6.4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Clique-062.jpg</td>
<td>6</td>
<td>7</td>
<td>16</td>
<td>3</td>
<td>7.8</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Clique-063.jpg</td>
<td>5</td>
<td>6</td>
<td>17</td>
<td>4</td>
<td>9.6</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-064.jpg</td>
<td>5</td>
<td>6</td>
<td>35</td>
<td>3</td>
<td>9.6</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-065.jpg</td>
<td>6</td>
<td>6</td>
<td>25</td>
<td>3</td>
<td>8.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Clique-066.jpg</td>
<td>6</td>
<td>6</td>
<td>17</td>
<td>3</td>
<td>8.2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Clique-067.jpg</td>
<td>7</td>
<td>12</td>
<td>15</td>
<td>3</td>
<td>5.4</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Clique-068.jpg</td>
<td>7</td>
<td>8</td>
<td>14</td>
<td>3</td>
<td>6.9</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-069.jpg</td>
<td>4</td>
<td>8</td>
<td>19</td>
<td>3</td>
<td>10.5</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Clique-070.jpg</td>
<td>6</td>
<td>6</td>
<td>16</td>
<td>4</td>
<td>8.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Clique-071.jpg</td>
<td>6</td>
<td>7</td>
<td>23</td>
<td>3</td>
<td>8</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-072.jpg</td>
<td>3</td>
<td>4</td>
<td>31</td>
<td>4</td>
<td>16</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Clique-073.jpg</td>
<td>5</td>
<td>5</td>
<td>25</td>
<td>3</td>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Clique-074.jpg</td>
<td>4</td>
<td>6</td>
<td>21</td>
<td>3</td>
<td>11.5</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Clique-075.jpg</td>
<td>5</td>
<td>7</td>
<td>22</td>
<td>3</td>
<td>9.2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>
Appendix B: RunManyPortsX

#!/bin/bash
export LD_LIBRARY_PATH=/usr/local/lib

f=$3

n=$4

r=$RANDOM

while test $f -le $n
do
r=$RANDOM
$1 $2 $f $r &
f=$(($f+1))
done
Appendix C: 1Colors_50Robots.world

include "/usr/local/share/stage/worlds/pioneer.inc"
include "/usr/local/share/stage/worlds/sick.inc"
include "/usr/local/share/stage/worlds/map.inc"

interval_sim 100  # milliseconds per update step
interval_real 100 # real-time milliseconds per update step

# The rover sonar array
define rover_sonar ranger
  (scount 2
    # define the pose of each transducer [xpos ypos heading]
    spose[0] [ 0.150 0.1 45 ]
    spose[1] [ 0.150 -0.1 -45]

    # define the field of view of each transducer [range_min range_max view_angle]
    sview [0 5.0 90]

    # define the size of each transducer [xsize ysize] in meters
    ssize [0.01 0.05]
  )

# a rover position 2 or 3 in standard configuration
define rover2dwc position
  ( size [.3 .3]
    # the rover's center of rotation is offset from its center of area
    origin [-0.04 0.0 0]

    # draw a nose on the robot so we can see which way it points
    gui_nose 1

    # estimated mass in KG

mass 1.0

# this polygon approximates the shape of a pioneer
polygons 1
polygon[0].points 8
polygon[0].point[0] [ 0.23 0.05 ]
polygon[0].point[1] [ 0.15 0.15 ]
polygon[0].point[2] [-0.15 0.15 ]
polygon[0].point[3] [-0.23 0.05 ]
polygon[0].point[4] [-0.23 -0.05 ]
polygon[0].point[5] [-0.15 -0.15 ]
polygon[0].point[6] [ 0.15 -0.15 ]
polygon[0].point[7] [ 0.23 -0.05 ]

# bumper( bcount 1
#   blength 0.4
#   bpose[0] [.5 0 0]
#   bpose[1] [0 0.165 -90]
#   bpose[2] [-0.26 0 0]
#   blength[2] 0.1
# )

rover_sonar()

ptz(
  blobfinder(
    channel_count 10
    channels [ "red" "blue" "green" "cyan" "yellow" "magenta" "orange" "gray" "pink" "purple"]
  )
)

model
(
  pose [-10.200 0.000 0.000]
  size [.1 13]
  color "white"
  line_count 4
  line[0] [-1 -1 -1 1]
  line[1] [-1 1 -.9 1]
  line[2] [-.9 1 -.9 -1]
  line[3] [-.9 -1 -1 -1]
)

model
(
  pose [10.200 0.000 0.000]
  size [.1 13]
  color "white"
  line_count 4
  line[0] [-1 -1 -1 1]
  line[1] [-1 1 -.9 1]
  line[2] [-.9 1 -.9 -1]
  line[3] [-.9 -1 -1 -1]
)
model
(  
  pose [0.000 6.450 0.000]
  size [20.4 .1]
  color "white"
  line_count 4
  line[0] [-1 -1 -1 -.9]
  line[1] [-1 -.9 1 -.9]
  line[2] [1 -.9 1 -1]
  line[3] [1 -1 -1 -1]
)

model
(  
  pose [0.000 -6.450 0.000]
  size [20.4 .1]
  color "white"
  line_count 4
  line[0] [-1 -1 -1 -.9]
  line[1] [-1 -.9 1 -.9]
  line[2] [1 -.9 1 -1]
  line[3] [1 -1 -1 -1]
)

rover2dwc
(  
  pose [0.000 0.000 0.000]
  color "red"
  name "ben"
  sick_laser( samples 361 laser_sample_skip 4 )
)

rover2dwc
(  
  pose [5.000 0.000 90.000]
  color "red"
  name "jona"
  sick_laser( samples 361 laser_sample_skip 4 )
)

rover2dwc
(  
  pose [3.000 2.000 190.000]
  color "red"
  name "Gemma"
  sick_laser( samples 361 laser_micesample_skip 4 )
)

.
.
.

(This file actually included 47 more blocks of code with the same format as above.)
Appendix D: Robot Program

#include <iostream>
#include "usr/local/include/player-2.0/libplayerc++/playerc++.h"
#include "usr/local/share/player/examples/libplayerc++/args.h"

using namespace PlayerCc;

const int numcolors = 10;
double turnratearray[numcolors];
double blobsizearray[numcolors];
double colorarray[numcolors];
int colorprefarray[numcolors];
double newturnrate = -20;
double turnrate = -20;
double newspeed = 0.1;
char avoid;
uint blobindex = 0;

bool blobsofinterest = false;
bool avoiding = false;

int randnum1();
void sonaravoid();
void uturn();
double setturturnrate(int index, int xvalue);
uint largestblob();

int blobx;

PlayerClient* robot;
BlobfinderProxy* bp;
SonarProxy* sp;
Position2dProxy* pp;

int main(int argc, char *argv[])
{
parse_args(argc, argv);

robot = new PlayerClient(gHostname, gPort);
pp = new Position2dProxy(robot, 0);
bp = new BlobfinderProxy(robot, 0);
sp = new SonarProxy(robot, 0);

colorarray[0] = 16711680; //red
colorarray[1] = 10494192; //purple
colorarray[2] = 65280; // green
colorarray[3] = 12500670; // grey
colorarray[4] = 16761035; // pink
colorarray[5] = 255; // blue
colorarray[6] = 65535; // cyan
colorarray[7] = 16776960; // yellow
colorarray[8] = 16753920; // orange
colorarray[9] = 16711935; // magenta

// seed rand() function with random number generated in RunManyPorts
long int randseed = atof(argv[3]) * atof(argv[3]);
srand48(randseed);
std::cout << "randseed = " << randseed << std::endl;

colorprefarray[0] = randnum1() + 1;
std::cout << "red preference: " << colorprefarray[0] << std::endl;
colorprefarray[1] = randnum1() + 1;
std::cout << "purple preference: " << colorprefarray[1] << std::endl;
colorprefarray[2] = randnum1() + 1;
std::cout << "green preference: " << colorprefarray[2] << std::endl;
colorprefarray[3] = randnum1() + 1;
std::cout << "grey preference: " << colorprefarray[3] << std::endl;
colorprefarray[4] = randnum1() + 1;
std::cout << "pink preference: " << colorprefarray[4] << std::endl;
colorprefarray[5] = randnum1() + 1;
std::cout << "blue preference: " << colorprefarray[5] << std::endl;
colorprefarray[6] = randnum1() + 1;
std::cout << "cyan preference: " << colorprefarray[6] << std::endl;
colorprefarray[7] = randnum1() + 1;
std::cout << "yellow preference: " << colorprefarray[7] << std::endl;
colorprefarray[8] = randnum1() + 1;
std::cout << "orange preference: " << colorprefarray[8] << std::endl;
colorprefarray[9] = randnum1() + 1;
std::cout << "magenta preference: " << colorprefarray[9] << std::endl;

pp->SetMotorEnable(true);

for (;;) // read-think-act loop
{

blobsofinterest = false;

robot->Read();
if (bp->GetCount() > 0)
{

blobindex = largestblob();

blobx = bp->GetBlob(blobindex).x;

switch (bp->GetBlob(blobindex).color) /* cases depend on color of blob. Set turnrate is where */
{
    case 16711680:
        
    case 16711680:
        
    case 16711680:
        
    case 16711680:
        
}
// std::cout << "case red" << std::endl;
turnrate = setturnrate(0, blobx);
break;
}

case 10494192:
{
// std::cout << "case purple" << std::endl;
turnrate = setturnrate(1, blobx);
break;
}

case 65280:
{
// std::cout << "case green" << std::endl;
turnrate = setturnrate(2, blobx);
break;
}

case 12500670:
{
// std::cout << "case grey" << std::endl;
turnrate = setturnrate(3, blobx);
break;
}

case 16761035:
{
// std::cout << "case pink" << std::endl;
turnrate = setturnrate(4, blobx);
break;
}

case 255:
{
// std::cout << "case pink" << std::endl;
turnrate = setturnrate(5, blobx);
break;
}

case 65535:
{
// std::cout << "case pink" << std::endl;
turnrate = setturnrate(6, blobx);
break;
}

case 16776960:
{
// std::cout << "case pink" << std::endl;
turnrate = setturnrate(7, blobx);
break;
}

case 16753920:
//   std::cout << "case pink" << std::endl;
    turnrate = setturnrate(8, blobx);
  break;
}

case 16711935:
{
  //   std::cout << "case pink" << std::endl;
    turnrate = setturnrate(9, blobx);
  break;
}

default:
{
  // std::cout << "case default" << std::endl;
    break;
}
}

if (blobsofinterest)//blobsofinterest gets set during switch block of setturnrate.
{                //set newturnrate based on turnrate and blobsize arrays.
  newturnrate = turnrate;
}
else newturnrate = 40*(2*(rand()/(RAND_MAX+1.0))-1);//random turnrate from -40 to 40
//std::cout << "turnrate = " << newturnrate << std::endl;

sonaravoid();//set newspeed depending on proximity to obstacles.

  /*
   if (newspeed < 0) //if backing up reverse direction of turn
       newturnrate = newturnrate * -1;
   */

  pp->SetSpeed(newspeed, dtor(newturnrate));
}
return 0;
}

//Functions

int randnum1() //pick a random integer from 0 to 1
{
  return drand48()*2;
}

uint largestblob() //get index of largest blob
{
  int oldwidth = 0;
  int newwidth = 0;
  int index = 0;
}
//robot->Read();
uint count = bp->GetCount(); //count = the number of blobs in field of view

for (uint j=0; j < count; ++j) //set largeblob equal to index of largest non-white blob
{
    newwidth = bp->GetBlob(j).right - bp->GetBlob(j).left;
    if (newwidth > oldwidth)
    {
        oldwidth = newwidth;
        index = j;
    }
}

return index;
}

void sonaravoid()
//set speed based on whether there are robots blocking the way
{
    double min_front_dist = 0.4;
    double avoid_dist = .5;

    robot->Read();

    // if (!avoiding)
    // {
    //    if((sp->GetScan(0) < min_front_dist) ||
    //       (sp->GetScan(1) < min_front_dist))
    //    {
    //        if (bp->GetCount() == 0)
    //            uturn();
    //        else
    //            newspeed = 0;
    //        avoiding = true;
    //    }
    //    else
    //    {
    //        newspeed = .2;
    //    }
    // }
    /*
    }*/
    else // robot is in the middle of avoidance manoeuvre.
    {
        if((sp->GetScan(0) < avoid_dist) || //continue backing up if robot
           (sp->GetScan(1) < avoid_dist)) // is within avoid_dist of obstacle.
        {
            newspeed = -0.200;
        }
        else
        {
            newspeed = .2; //if not within avoid_dist
        }
    }
double setturnrate(int index, int xvalue)
// set the turn rate for blobs of a color based on robot's
// preference for the color.
{
    double rate = 0;

    switch (colorprefarray[index])
    {
        case 1://robot likes blob
        {
            rate = (40 - xvalue);// turn towards blobs
            blobsofinterest = true;
            break;
        }
        case 2://robot doesn't like blob
        {
            if (xvalue <= 40) rate = -40;// always turn quickly away even if blob is at
            else rate = 40; // edge of field of view.
            blobsofinterest = true;
            break;
        }
        default://robot is indifferent to blob
        break;
    }
    return rate;
}

void uturn()
{
    double begin;
    double end;

    robot->Read();
    begin = pp->GetYaw();
    if (begin >= 0)
        end = begin - 3.14*(rand()/(RAND_MAX+1.0));
    else
        end = begin + 3.14*(rand()/(RAND_MAX+1.0));

    while ( fabs(end - pp->GetYaw()) > .2)
    {
        robot->Read();
        pp->SetSpeed(0, dtor(-30));
    }
}