## Matlab Code (Appendix 3)

### ***pitch.m***

1. function pitch\_f
2. % define every part of the pitch and set value to them as variables
3. PitchLength=9;
4. PitchWidth=6;
5. GoalDepth=0.6;
6. GoalWidth=2.6;
7. GoalHeight=1.2;
8. GoalAreaLength=1;
9. GoalAreaWidth=3;
10. PenaltyMarkDistance=1.5;
11. CentreCircleDiameter=1.5;
12. BorderStripWidth=1;
13. PenaltyAreaLength=2;
14. PenaltyAreaWidth=5;
15. % put every variables into the new variables so that it can be easier to
16. % code in the rectangle functions
17. A=PitchLength;
18. B=PitchWidth;
19. C=GoalDepth;
20. D=GoalWidth;
21. E=GoalAreaLength;
22. F=GoalAreaWidth;
23. G=PenaltyMarkDistance;
24. H=CentreCircleDiameter;
25. I=BorderStripWidth;
26. J=PenaltyAreaLength;
27. K=PenaltyAreaWidth;
28. % use the rectangle function to draw different part of the pitch
29. axis([0 A+2\*I 0 B+2\*I]);
30. rectangle('Position',[0 0 2\*I+A 2\*I+B],'FaceColor','g','EdgeColor','w');
31. rectangle('Position',[I I A B],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
32. rectangle('Position',[I-C B/2+I-D/2 C D],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
33. rectangle('Position',[I+A B/2+I-D/2 C D],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
34. rectangle('Position',[I B/2+I-F/2 E F],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
35. rectangle('Position',[I+A-E B/2+I-F/2 E F],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
36. rectangle('Position',[I B/2+I-K/2 J K],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
37. rectangle('Position',[I+A-J B/2+I-K/2 J K],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
38. viscircles([I+A/2 I+B/2],H/2,'Color','w','LineWidth',0.75);
39. hold on;
40. plot(I+G,I+B/2,'w+');
41. plot(I+A/2,I+B/2,'w+');
42. plot(I+A-G,I+B/2,'w+');
43. line([I+A/2 I+A/2],[I+B I],'Color','w','LineWidth',1);

### ***BallFree\_f.m***

1. function BallFree\_f(Ballpos,Balltheta)
2. results=1;
3. Ball\_pos=Ballpos;
4. theta\_Ballmove=Balltheta;
5. v\_Ball=2;
6. t\_ball=3;
7. dis\_Ball=1/2\*v\_Ball\*t\_ball;
8. delta\_x\_Ball=dis\_Ball\*cos(theta\_Ballmove);
9. delta\_y\_Ball=dis\_Ball\*sin(theta\_Ballmove);
10. Ball\_aim=[Ball\_pos(1)+delta\_x\_Ball Ball\_pos(2)+delta\_y\_Ball];
11. t\_equal=dis\_Ball/(2\*0.2145);
12. num\_Ball=ceil(t\_equal);
13. for i=0:num\_Ball
14. run pitch.m
15. x=Ball\_pos(1)+i\*delta\_x\_Ball/num\_Ball;
16. y=Ball\_pos(2)+i\*delta\_y\_Ball/num\_Ball;
17. viscircles([x y],0.143/2,'Color','black','LineWidth',0.75);
18. pause(0.2);
19. if i<num\_Ball
20. clf;
21. end
22. end

### ***Player MoveInPitch\_f.m***

1. function MoveInPitch\_f(A1pos,A1aim,A1theta)
2. results=1;
3. A1\_pos=A1pos;
4. A1\_aim=A1aim;
5. theta\_A1=A1theta;
6. dis\_A1=sqrt((A1\_aim(1)-A1\_pos(1))^2+(A1\_aim(2)-A1\_pos(2))^2);
7. v\_A1=2\*0.2145;
8. t\_A1=dis\_A1/v\_A1;
9. num\_A1=ceil(t\_A1);
10. delta\_x\_A1=A1\_aim(1)-A1\_pos(1);
11. delta\_y\_A1=A1\_aim(2)-A1\_pos(2);
12. theta\_A1move=atan(delta\_y\_A1/delta\_x\_A1);
13. if theta\_A1~=theta\_A1move
14. delta\_theta=theta\_A1move-theta\_A1;
15. for i=0:2
16. run pitch.m
17. theta\_mid=theta\_A1+i\*delta\_theta/2;
18. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
19. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid+pi/3)],'Color','m','LineWidth',1);
20. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid-pi/3)],'Color','m','LineWidth',1);
21. pause(0.2)
22. if i<2
23. clf;
24. end
25. end
26. end
27. for i=0:num\_A1
28. run pitch.m
29. x=A1\_pos(1)+i\*delta\_x\_A1/num\_A1;
30. y=A1\_pos(2)+i\*delta\_y\_A1/num\_A1;
31. viscircles([x y],0.2145,'Color','r','LineWidth',0.75);
32. line([x x+0.2145\*cos(theta\_A1move+pi/3)],[y y+0.2145\*sin(theta\_A1move+pi/3)],'Color','m','LineWidth',1);
33. line([x x+0.2145\*cos(theta\_A1move-pi/3)],[y y+0.2145\*sin(theta\_A1move-pi/3)],'Color','m','LineWidth',1);
34. pause(0.2);
35. if i<num\_A1
36. clf;
37. end
38. end

### ***GoToBall\_f.m***

1. function GoToBall\_f(A1pos,Ballpos,A1theta)
2. results=1;
3. A1\_pos=A1pos;
4. Ball\_pos=Ballpos;
5. theta\_A1=A1theta;
6. A1\_aim=Ball\_pos;
7. dis\_A1=sqrt((A1\_aim(1)-A1\_pos(1))^2+(A1\_aim(2)-A1\_pos(2))^2);
8. v\_A1=2\*0.2145;
9. t\_A1=dis\_A1/v\_A1;
10. num\_A1=ceil(t\_A1);
11. delta\_x\_A1=A1\_aim(1)-A1\_pos(1);
12. delta\_y\_A1=A1\_aim(2)-A1\_pos(2);
13. theta\_A1move=atan(delta\_y\_A1/delta\_x\_A1);
14. if theta\_A1~=theta\_A1move
15. delta\_theta=theta\_A1move-theta\_A1;
16. for i=0:2
17. run pitch.m
18. theta\_mid=theta\_A1+i\*delta\_theta/2;
19. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
20. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
21. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid+pi/3)],'Color','m','LineWidth',1);
22. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid-pi/3)],'Color','m','LineWidth',1);
23. pause(0.5)
24. if i<2
25. clf;
26. end
27. end
28. end
29. for i=0:num\_A1
30. run pitch.m
31. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
32. x=A1\_pos(1)+i\*delta\_x\_A1/num\_A1;
33. y=A1\_pos(2)+i\*delta\_y\_A1/num\_A1;
34. viscircles([x y],0.2145,'Color','r','LineWidth',0.75);
35. line([x x+0.2145\*cos(theta\_A1move+pi/3)],[y y+0.2145\*sin(theta\_A1move+pi/3)],'Color','m','LineWidth',1);
36. line([x x+0.2145\*cos(theta\_A1move-pi/3)],[y y+0.2145\*sin(theta\_A1move-pi/3)],'Color','m','LineWidth',1);
37. pause(0.2);
38. if i<num\_A1
39. clf;
40. end
41. end

### ***PassBall\_f.m***

1. % initial the positions of A1, A2, ball and goal; set variables such as velocity, facing
2. % direction of A1 and A2
3. % set ball's velocity
4. A1\_pos=[1 2];
5. v\_A1=2\*0.2145;
6. theta\_A1=0;
7. Ball\_pos=[6 3];
8. v\_Ball=2;
9. t\_Ball=3;
10. A2\_pos=[2 7];
11. theta\_A2=0;
12. v\_A2=v\_A1;
13. Goal\_pos=[10 4];
14. % calculate the distances from A1 to ball and from A2 to ball
15. % and also calculate the distances between A2 and goal
16. % dis\_Ball means the range that allow players to pass the ball
17. dis\_A1\_Ball=sqrt((Ball\_pos(1)-A1\_pos(1))^2+(Ball\_pos(2)-A1\_pos(2))^2);
18. dis\_Ball=1/2\*v\_Ball\*t\_Ball;
19. dis\_A2\_Ball=sqrt((Ball\_pos(1)-A2\_pos(1))^2+(Ball\_pos(2)-A2\_pos(2))^2);
20. dis\_A2\_goal=sqrt((Goal\_pos(1)-A2\_pos(1))^2+(Goal\_pos(2)-A2\_pos(2))^2);
21. % calculate the time that A1 runs to the ball
22. t\_A1\_Ball=dis\_A1\_Ball/v\_A1;
23. num\_A1=ceil(t\_A1\_Ball);
24. % calculate the angles of A1 to ball and A2 to ball
25. % then use two frames to make A1 and A2 turn their directions toward ball
26. delta\_x\_A1\_Ball=Ball\_pos(1)-A1\_pos(1);
27. delta\_y\_A1\_Ball=Ball\_pos(2)-A1\_pos(2);
28. delta\_x\_A2\_Ball=Ball\_pos(1)-A2\_pos(1);
29. delta\_y\_A2\_Ball=Ball\_pos(2)-A2\_pos(2);
30. delta\_x\_A1\_A2=A2\_pos(1)-A1\_pos(1);
31. delta\_y\_A1\_A2=A2\_pos(2)-A1\_pos(2);
32. delta\_x\_A2\_Goal=A2\_pos(1)-Goal\_pos(1);
33. delta\_y\_A2\_Goal=A2\_pos(2)-Goal\_pos(2);
34. theta\_A1\_Ball=atan(delta\_y\_A1\_Ball/delta\_x\_A1\_Ball);
35. theta\_A2\_Ball=atan(delta\_y\_A2\_Ball/delta\_x\_A2\_Ball);
36. theta\_A1\_A2=atan(delta\_y\_A1\_A2/delta\_x\_A1\_A2);
37. theta\_A2\_Goal=atan(delta\_y\_A2\_Goal/delta\_x\_A2\_Goal);
38. if theta\_A1~=theta\_A1\_Ball || theta\_A2~=theta\_A2\_Ball
39. if delta\_x\_A1\_Ball>=0
40. delta\_theta\_A1\_Ball=theta\_A1\_Ball-theta\_A1;
41. else
42. delta\_theta\_A1\_Ball=theta\_A1\_Ball-theta\_A1+pi;
43. end
44. if delta\_x\_A2\_Ball>=0
45. delta\_theta\_A2\_Ball=theta\_A2\_Ball-theta\_A2;
46. else
47. delta\_theta\_A2\_Ball=theta\_A2\_Ball-theta\_A2+pi;
48. end
49. for i=0:2
50. run pitch.m;
51. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A1\_Ball/2;
52. theta\_mid\_A2=theta\_A2+i\*delta\_theta\_A2\_Ball/2;
53. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
54. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
55. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','m','LineWidth',1);
56. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','m','LineWidth',1);
57. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
58. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_mid\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_mid\_A2+pi/3)],'Color','m','LineWidth',1);
59. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_mid\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_mid\_A2-pi/3)],'Color','m','LineWidth',1);
60. pause(0.5)
61. if i<2
62. clf;
63. else
64. theta\_A1=theta\_mid\_A1;
65. theta\_A2=theta\_mid\_A2;
66. end
67. end
68. end
69. % draw A1 runs to ball
70. for i=0:num\_A1
71. run pitch.m
72. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
73. x=A1\_pos(1)+i\*delta\_x\_A1\_Ball/num\_A1;
74. y=A1\_pos(2)+i\*delta\_y\_A1\_Ball/num\_A1;
75. viscircles([x y],0.2145,'Color','r','LineWidth',0.75);
76. line([x x+0.2145\*cos(theta\_A1+pi/3)],[y y+0.2145\*sin(theta\_A1+pi/3)],'Color','m','LineWidth',1);
77. line([x x+0.2145\*cos(theta\_A1-pi/3)],[y y+0.2145\*sin(theta\_A1-pi/3)],'Color','m','LineWidth',1);
78. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
79. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+pi/3)],'Color','m','LineWidth',1);
80. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2-pi/3)],'Color','m','LineWidth',1);
81. pause(0.2);
82. if i<num\_A1
83. clf;
84. else
85. A1\_pos(1)=x;
86. A1\_pos(2)=y;
87. end
88. end
89. % if the ball distance is out of the range that allow passing ball, A2 will
90. % turn its direction to the ball first
91. % when A1 gets the ball and it will find A2 to pass and A1 will turn its
92. % angle toward A2
93. if dis\_Ball<dis\_A2\_Ball
94. if delta\_x\_A2\_Ball>=0
95. delta\_theta\_A2\_Ball=theta\_A2\_Ball-theta\_A1+pi;
96. else
97. delta\_theta\_A2\_Ball=theta\_A2\_Ball-theta\_A1;
98. end
99. delta\_dis=dis\_A2\_Ball-dis\_Ball;
100. if A2\_pos(1)>A1\_pos(1)
101. delta\_x\_dis=delta\_dis\*cos(theta\_A2\_Ball);
102. delta\_y\_dis=delta\_dis\*sin(theta\_A2\_Ball);
103. else
104. delta\_x\_dis=-delta\_dis\*cos(theta\_A2\_Ball);
105. delta\_y\_dis=-delta\_dis\*sin(theta\_A2\_Ball);
106. end
107. t\_delta=delta\_dis/v\_A1;
108. num\_delta=ceil(t\_delta);
109. if theta\_A1\_A2~=theta\_A1
110. for i=0:2
111. run pitch.m;
112. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A2\_Ball/2;
113. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
114. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
115. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','m','LineWidth',1);
116. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','m','LineWidth',1);
117. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
118. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+pi/3)],'Color','m','LineWidth',1);
119. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2-pi/3)],'Color','m','LineWidth',1);
120. if i<2
121. clf;
122. else
123. theta\_A1=theta\_mid\_A1;
124. end
125. end
126. end
127. for i=0:num\_delta
128. run pitch.m
129. x=A1\_pos(1)+i\*delta\_x\_dis/num\_delta;
130. y=A1\_pos(2)+i\*delta\_y\_dis/num\_delta;
131. Ball\_pos(1)=x;
132. Ball\_pos(2)=y;
133. delta\_x\_A2\_Ball=Ball\_pos(1)-A2\_pos(1);
134. delta\_y\_A2\_Ball=Ball\_pos(2)-A2\_pos(2);
135. theta\_A2\_Ball=atan(delta\_y\_A2\_Ball/delta\_x\_A2\_Ball);
136. if delta\_x\_A2\_Ball>=0
137. delta\_theta\_A2\_Ball=theta\_A2\_Ball-theta\_A2;
138. else
139. delta\_theta\_A2\_Ball=theta\_A2\_Ball-theta\_A2+pi;
140. end
141. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
142. viscircles([x y],0.2145,'Color','r','LineWidth',0.75);
143. line([x x+0.2145\*cos(theta\_A1+pi/3)],[y y+0.2145\*sin(theta\_A1+pi/3)],'Color','m','LineWidth',1);
144. line([x x+0.2145\*cos(theta\_A1-pi/3)],[y y+0.2145\*sin(theta\_A1-pi/3)],'Color','m','LineWidth',1);
145. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
146. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+delta\_theta\_A2\_Ball+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+delta\_theta\_A2\_Ball+pi/3)],'Color','m','LineWidth',1);
147. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+delta\_theta\_A2\_Ball-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+delta\_theta\_A2\_Ball-pi/3)],'Color','m','LineWidth',1);
148. pause(0.2);
149. if i<num\_delta
150. clf;
151. else
152. A1\_pos(1)=x;
153. A1\_pos(2)=y;
154. Ball\_pos(1)=A1\_pos(1);
155. Ball\_pos(2)=A1\_pos(2);
156. end
157. end
158. theta\_Ballmove=theta\_A1;
159. delta\_x\_Ball=dis\_Ball\*cos(theta\_Ballmove);
160. delta\_y\_Ball=dis\_Ball\*sin(theta\_Ballmove);
161. t\_equal=dis\_Ball/(4\*0.2145);
162. num\_Ball=ceil(t\_equal);
163. for i=0:num\_Ball
164. run pitch.m
165. x=Ball\_pos(1)+i\*delta\_x\_Ball/num\_Ball;
166. y=Ball\_pos(2)+i\*delta\_y\_Ball/num\_Ball;
167. viscircles([x y],0.143/2,'Color','black','LineWidth',0.75);
168. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
169. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','m','LineWidth',1);
170. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','m','LineWidth',1);
171. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
172. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+pi/3)],'Color','m','LineWidth',1);
173. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2-pi/3)],'Color','m','LineWidth',1);
174. pause(0.2);
175. if i<num\_Ball
176. clf;
177. else
178. Ball\_pos(1)=A2\_pos(1);
179. end
180. end
181. end
182. % when A1 dribbles into the range that it can pass ball to A2
183. if dis\_Ball>=dis\_A2\_Ball
184. if delta\_x\_A2\_Ball>=0
185. delta\_theta\_A2\_Ball=theta\_A2\_Ball-theta\_A1+pi;
186. else
187. delta\_theta\_A2\_Ball=theta\_A2\_Ball-theta\_A1;
188. end
189. delta\_dis=dis\_A2\_Ball-dis\_Ball;
190. if A2\_pos(1)>A1\_pos(1)
191. delta\_x\_dis=delta\_dis\*cos(theta\_A2\_Ball);
192. delta\_y\_dis=delta\_dis\*sin(theta\_A2\_Ball);
193. else
194. delta\_x\_dis=-delta\_dis\*cos(theta\_A2\_Ball);
195. delta\_y\_dis=-delta\_dis\*sin(theta\_A2\_Ball);
196. end
197. t\_delta=delta\_dis/v\_A1;
198. num\_delta=ceil(t\_delta);
199. if theta\_A1\_A2~=theta\_A1
200. for i=0:2
201. run pitch.m;
202. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A2\_Ball/2;
203. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
204. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
205. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','m','LineWidth',1);
206. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','m','LineWidth',1);
207. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
208. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+pi/3)],'Color','m','LineWidth',1);
209. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2-pi/3)],'Color','m','LineWidth',1);
210. if i<2
211. clf;
212. else
213. theta\_A1=theta\_mid\_A1;
214. end
215. end
216. end
217. t\_equal=dis\_A2\_Ball/(4\*0.2145);
218. num\_Ball=ceil(t\_equal);
219. theta\_Ballmove=theta\_A1;
220. delta\_x\_Ball=dis\_A2\_Ball\*cos(theta\_Ballmove);
221. delta\_y\_Ball=dis\_A2\_Ball\*sin(theta\_Ballmove);
222. for i=0:num\_Ball
223. run pitch.m
224. x=Ball\_pos(1)+i\*delta\_x\_Ball/num\_Ball;
225. y=Ball\_pos(2)+i\*delta\_y\_Ball/num\_Ball;
226. viscircles([x y],0.143/2,'Color','black','LineWidth',0.75);
227. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
228. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','m','LineWidth',1);
229. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','m','LineWidth',1);
230. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
231. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+pi/3)],'Color','m','LineWidth',1);
232. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2-pi/3)],'Color','m','LineWidth',1);
233. pause(0.2);
234. if i<num\_Ball
235. clf;
236. else
237. Ball\_pos(1)=A2\_pos(1);
238. Ball\_pos(2)=A2\_pos(2);
239. end
240. end
241. end
242. % when A2 gets the ball, it will turn its direction toward goal
243. if theta\_A2\_Goal~=theta\_A2
244. delta\_theta\_A2\_Goal=theta\_A2\_Goal-theta\_A2;
245. for i=0:2
246. run pitch.m;
247. theta\_mid\_A2=theta\_A2+i\*delta\_theta\_A2\_Goal/2;
248. viscircles([A2\_pos(1) A2\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
249. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
250. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','m','LineWidth',1);
251. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','m','LineWidth',1);
252. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
253. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_mid\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_mid\_A2+pi/3)],'Color','m','LineWidth',1);
254. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_mid\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_mid\_A2-pi/3)],'Color','m','LineWidth',1);
255. pause(0.2);
256. if i<2
257. clf;
258. else
259. theta\_A2=theta\_A2\_Goal;
260. Ball\_pos(1)=A2\_pos(1);
261. Ball\_pos(2)=A2\_pos(2);
262. end
263. end
264. end
265. % after turning direction toward goal, A2 will dribble in front of goal
266. if dis\_A2\_goal>dis\_Ball
267. delta\_dis=dis\_A2\_goal-dis\_Ball;
268. delta\_x\_dis=delta\_dis\*cos(theta\_A2);
269. delta\_y\_dis=delta\_dis\*sin(theta\_A2);
270. t\_delta=delta\_dis/v\_A2;
271. num\_delta=ceil(t\_delta);
272. for i=0:num\_delta
273. run pitch.m
274. x=A2\_pos(1)+i\*delta\_x\_dis/num\_delta;
275. y=A2\_pos(2)+i\*delta\_y\_dis/num\_delta;
276. Ball\_pos(1)=x;
277. Ball\_pos(2)=y;
278. delta\_x\_A1\_Ball=Ball\_pos(1)-A1\_pos(1);
279. delta\_y\_A1\_Ball=Ball\_pos(2)-A1\_pos(2);
280. theta\_A1\_Ball=atan(delta\_y\_A1\_Ball/delta\_x\_A1\_Ball);
281. if delta\_x\_A1\_Ball>=0
282. delta\_theta\_A1\_Ball=theta\_A1\_Ball-theta\_A1;
283. else
284. delta\_theta\_A1\_Ball=theta\_A1\_Ball-theta\_A1+pi;
285. end
286. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
287. viscircles([x y],0.2145,'Color','r','LineWidth',0.75);
288. line([x x+0.2145\*cos(theta\_A2+pi/3)],[y y+0.2145\*sin(theta\_A2+pi/3)],'Color','m','LineWidth',1);
289. line([x x+0.2145\*cos(theta\_A2-pi/3)],[y y+0.2145\*sin(theta\_A2-pi/3)],'Color','m','LineWidth',1);
290. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
291. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+delta\_theta\_A1\_Ball+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+delta\_theta\_A1\_Ball+pi/3)],'Color','m','LineWidth',1);
292. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+delta\_theta\_A1\_Ball-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+delta\_theta\_A1\_Ball-pi/3)],'Color','m','LineWidth',1);
293. pause(0.2);
294. if i<num\_delta
295. clf;
296. else
297. A2\_pos(1)=x;
298. A2\_pos(2)=y;
299. Ball\_pos(1)=A2\_pos(1);
300. Ball\_pos(2)=A2\_pos(2);
301. end
302. end
303. t\_equal=dis\_Ball/(4\*0.2145);
304. num\_Ball=ceil(t\_equal);
305. theta\_Ballmove=theta\_A2;
306. delta\_x\_Ball=dis\_Ball\*cos(theta\_Ballmove);
307. delta\_y\_Ball=dis\_Ball\*sin(theta\_Ballmove);
308. for i=0:num\_Ball
309. run pitch.m
310. x=Ball\_pos(1)+i\*delta\_x\_Ball/num\_Ball;
311. y=Ball\_pos(2)+i\*delta\_y\_Ball/num\_Ball;
312. delta\_x\_A1\_Ball=x-A1\_pos(1);
313. delta\_y\_A1\_Ball=y-A1\_pos(2);
314. theta\_A1\_Ball=atan(delta\_y\_A1\_Ball/delta\_x\_A1\_Ball);
315. if delta\_x\_A1\_Ball>=0
316. delta\_theta\_A1\_Ball=theta\_A1\_Ball-theta\_A1;
317. else
318. delta\_theta\_A1\_Ball=theta\_A1\_Ball-theta\_A1+pi;
319. end
320. viscircles([x y],0.143/2,'Color','black','LineWidth',0.75);
321. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
322. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+delta\_theta\_A1\_Ball+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+delta\_theta\_A1\_Ball+pi/3)],'Color','m','LineWidth',1);
323. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+delta\_theta\_A1\_Ball-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+delta\_theta\_A1\_Ball-pi/3)],'Color','m','LineWidth',1);
324. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
325. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+pi/3)],'Color','m','LineWidth',1);
326. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2-pi/3)],'Color','m','LineWidth',1);
327. pause(0.2);
328. if i<num\_Ball
329. clf;
330. end
331. end
332. end
333. % when A2 dribble in front of the goal and into the shooting range, A2
334. % shoot the ball to goal
335. if dis\_A2\_goal<=dis\_Ball
336. t\_equal=dis\_Ball/(4\*0.2145);
337. num\_Ball=ceil(t\_equal);
338. theta\_Ballmove=theta\_A2;
339. delta\_x\_Ball=dis\_Ball\*cos(theta\_Ballmove);
340. delta\_y\_Ball=dis\_Ball\*sin(theta\_Ballmove);
341. for i=0:num\_Ball
342. run pitch.m
343. x=Ball\_pos(1)+i\*delta\_x\_Ball/num\_Ball;
344. y=Ball\_pos(2)+i\*delta\_y\_Ball/num\_Ball;
345. delta\_x\_A1\_Ball=x-A1\_pos(1);
346. delta\_y\_A1\_Ball=y-A1\_pos(2);
347. theta\_A1\_Ball=atan(delta\_y\_A1\_Ball/delta\_x\_A1\_Ball);
348. if delta\_x\_A1\_Ball>=0
349. delta\_theta\_A1\_Ball=theta\_A1\_Ball-theta\_A1;
350. else
351. delta\_theta\_A1\_Ball=theta\_A1\_Ball-theta\_A1+pi;
352. end
353. viscircles([x y],0.143/2,'Color','black','LineWidth',0.75);
354. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
355. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+delta\_theta\_A1\_Ball+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+delta\_theta\_A1\_Ball+pi/3)],'Color','m','LineWidth',1);
356. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+delta\_theta\_A1\_Ball-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+delta\_theta\_A1\_Ball-pi/3)],'Color','m','LineWidth',1);
357. viscircles([A2\_pos(1) A2\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
358. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2+pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2+pi/3)],'Color','m','LineWidth',1);
359. line([A2\_pos(1) A2\_pos(1)+0.2145\*cos(theta\_A2-pi/3)],[A2\_pos(2) A2\_pos(2)+0.2145\*sin(theta\_A2-pi/3)],'Color','m','LineWidth',1);
360. pause(0.2);
361. if i<num\_Ball
362. clf;
363. end
364. end
365. end

### ***KickBall\_f.m***

1. function KickBall\_f(A1pos,Ballpos,A1theta)
2. % A1pos = [1 1];
3. % Ballpos = [7.5 5];
4. % A1theta = 0;
5. %results=1; %---
6. A1\_pos=A1pos; %Initial Position of Player 1
7. Ball\_pos=Ballpos; %Initial Position of Ball
8. A1\_aim=Ball\_pos; %Final Position of Player 1
9. theta\_A1=A1theta; %Initial Orientation of Player 1
10. dis\_A1=sqrt((A1\_aim(1)-A1\_pos(1))^2+(A1\_aim(2)-A1\_pos(2))^2); %Distance of Player 1 from Initial to Final Position
11. v\_A1=2\*0.2145; %Velocity of Player 1
12. t\_A1=dis\_A1/v\_A1; %Time to travel for Player 1
13. num\_A1=ceil(t\_A1); %Round up Travel Time for Player 1 to higher
14. delta\_x\_A1=A1\_aim(1)-A1\_pos(1); %Change in X coordinate (Dist in X) of Player 1 from Initial Position to Final Position
15. delta\_y\_A1=A1\_aim(2)-A1\_pos(2); %Change in Y coordinate (Dist in Y) of Player 1 from Initial Position to Final Position
16. theta\_A1move=atan(delta\_y\_A1/delta\_x\_A1); %Change in Orientation of Player 1 required to reach Final Position
17. %Loop for Player 1 to change orientation in direction of movement towards ball in two steps
18. if theta\_A1~=theta\_A1move %Check if orientation of Player 1 is same as direction of movement or not
19. delta\_theta=theta\_A1move-theta\_A1; %If not equal to direction of movement, then get 'delta\_theta'
20. for i=0:2 %Two Steps of Simulation
21. run pitch.m %Plot Pitch
22. theta\_mid=theta\_A1+i\*delta\_theta/2; %Half movement calculation (Half movement in each step)
23. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75); %Plot Ball
24. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75); %Plot Player
25. %Plot Player Face (Angles of Pac-Man)
26. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid+pi/3)],'Color','m','LineWidth',1);
27. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid-pi/3)],'Color','m','LineWidth',1);
28. pause(0.5)
29. if i<2
30. clf; %Clear Figure after every step
31. end
32. end
33. end
34. %Loop for Player 1 to travel to Ball
35. for i=0:num\_A1 %Steps of Simulation as per travel time reqd for Player 1 to reach ball
36. run pitch.m
37. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75); %Plot Ball
38. x=A1\_pos(1)+i\*delta\_x\_A1/num\_A1; %X Coordinate calc for Player 1 when moving towards ball for each simulation step
39. y=A1\_pos(2)+i\*delta\_y\_A1/num\_A1; %Y Coordinate calc for Player 1 when moving towards ball for each simulation step
40. %Plot Player and Face Angles for each Simulation step with new X & Y Coordinates
41. viscircles([x y],0.2145,'Color','r','LineWidth',0.75);
42. line([x x+0.2145\*cos(theta\_A1move+pi/3)],[y y+0.2145\*sin(theta\_A1move+pi/3)],'Color','m','LineWidth',1);
43. line([x x+0.2145\*cos(theta\_A1move-pi/3)],[y y+0.2145\*sin(theta\_A1move-pi/3)],'Color','m','LineWidth',1);
44. pause(0.2);
45. if i<num\_A1
46. clf; %Clear Figure after every step
47. end
48. end
49. %Loop for Player 1 to change orientation with the Ball in direction of Centre of Away-Team Goal in two steps
50. Goal\_A\_aim=[10 4]; %Coordinates of Centre of Away-Team Goal
51. dis\_x\_Goal=Goal\_A\_aim(1)-Ball\_pos(1); %Change in X coordinate of Ball from Initial Position to Centre of Away-Team Goal
52. dis\_y\_Goal=Goal\_A\_aim(2)-Ball\_pos(2); %Change in Y coordinate of Ball from Initial Position to Centre of Away-Team Goal
53. theta\_goal=atan(dis\_y\_Goal/dis\_x\_Goal); %Orientation Req to face Centre of Away-Team Goal from Initial Position of Ball
54. if theta\_A1move~=theta\_goal %Check if orientation of Player 1 is same as direction required to face Centre of Away-Team Goal
55. delta\_theta=theta\_goal-theta\_A1move; %If not equal to direction required, then get 'delta\_theta'
56. for i=0:2 %Two Steps of Simulation
57. run pitch.m %Plot Pitch
58. theta\_mid=theta\_A1move+i\*delta\_theta/2; %Half movement calculation (Half movement in each step)
59. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75); %Plot Ball
60. viscircles([A1\_aim(1) A1\_aim(2)],0.2145,'Color','r','LineWidth',0.75); %Plot Player
61. %Plot Player Face (Angles of Pac-Man)
62. line([A1\_aim(1) A1\_aim(1)+0.2145\*cos(theta\_mid+pi/3)],[A1\_aim(2) A1\_aim(2)+0.2145\*sin(theta\_mid+pi/3)],'Color','m','LineWidth',1);
63. line([A1\_aim(1) A1\_aim(1)+0.2145\*cos(theta\_mid-pi/3)],[A1\_aim(2) A1\_aim(2)+0.2145\*sin(theta\_mid-pi/3)],'Color','m','LineWidth',1);
64. pause(0.5)
65. if i<2
66. clf; %Clear Figure after every step
67. end
68. end
69. end
70. %Loop for Ball Movement when Kicked by Player 1 to Centre of Away-Team Goal
71. theta\_Ballmove=theta\_goal; %Angle Req for movement of Ball when Kicked to Centre of Away-Team Goal
72. v\_Ball=2; %Velocity of Ball when Kicked
73. t\_ball=3; %Time for Ball to reach Goal (Will make Dependent on Kick Strength)
74. dis\_Ball=1/2\*v\_Ball\*t\_ball; %Half of the distance req for Ball to reach Centre of Away-Team Goal
75. delta\_x\_Ball=dis\_Ball\*cos(theta\_Ballmove); %Change in X coordinate (Dist in X) of Ball from Initial Position to Centre of Away-Team Goal
76. delta\_y\_Ball=dis\_Ball\*sin(theta\_Ballmove); %Change in Y coordinate (Dist in Y) of Ball from Initial Position to Centre of Away-Team Goal
77. Ball\_aim=[Ball\_pos(1)+delta\_x\_Ball Ball\_pos(2)+delta\_y\_Ball]; %Calc of X and Y Coordiantes of Final Position of Ball after Kicked
78. t\_equal=dis\_Ball/(2\*0.2145); %Calc of Time for Ball to go half of req distance with each step movement of equal to Player 1 Diameter (Control Speed of Ball Movement .i.e. to be made dependent on Kick Strength)
79. num\_Ball=ceil(t\_equal); %Round up Half Travel Time for Ball to higher
80. for i=0:num\_Ball %Steps of Simulation as per travel time reqd for Ball to reach Away-Team Goal Centre
81. run pitch.m %Plot Pitch
82. x=Ball\_pos(1)+i\*delta\_x\_Ball/num\_Ball; %X Coordinate calc for Ball when moving towards Away-Team Goal for each simulation step
83. y=Ball\_pos(2)+i\*delta\_y\_Ball/num\_Ball; %Y Coordinate calc for Ball when moving towards Away-Team Goal for each simulation step
84. %Plot Player and Face Angles for each Simulation step with same X & Y Coordinates from where he kicked the ball
85. viscircles([A1\_aim(1) A1\_aim(2)],0.2145,'Color','r','LineWidth',0.75);
86. line([A1\_aim(1) A1\_aim(1)+0.2145\*cos(theta\_goal+pi/3)],[A1\_aim(2) A1\_aim(2)+0.2145\*sin(theta\_goal+pi/3)],'Color','m','LineWidth',1);
87. line([A1\_aim(1) A1\_aim(1)+0.2145\*cos(theta\_goal-pi/3)],[A1\_aim(2) A1\_aim(2)+0.2145\*sin(theta\_goal-pi/3)],'Color','m','LineWidth',1);
88. %Plot Ball for each Simulation step with new X & Y Coordinates
89. viscircles([x y],0.143/2,'Color','black','LineWidth',0.75);
90. pause(0.2);
91. if i<num\_Ball
92. clf; %Clear Figure after every step
93. end
94. end

### ***test.m***

1. %MoveInPitch\_f([1,1],[7.5 5],0);
2. %BallFree\_f([5.5 4],0);
3. %GoToBall\_f([1 1],[7.5 5],0);
4. KickBall\_f([1 1],[7.5 5],0);

### ***A1\_B1.m***

1. %Initialize position coordinates and angles
2. A1\_pos=[3 1];
3. B1\_pos=[6 1];
4. Ball\_pos=[7 5];
5. Goal1\_pos=[10 4];
6. Goal2\_pos=[1 4];
7. Ball\_right=0;
8. theta\_A1=0;
9. theta\_B1=pi;
10. %Measure the distance and time between the player and the ball
11. A1\_aim=Ball\_pos;
12. B1\_aim=Ball\_pos;
13. dis\_A1\_Ball=sqrt((A1\_aim(1)-A1\_pos(1))^2+(A1\_aim(2)-A1\_pos(2))^2);
14. dis\_B1\_Ball=sqrt((B1\_aim(1)-B1\_pos(1))^2+(B1\_aim(2)-B1\_pos(2))^2);
15. v\_A1=2\*0.2145;
16. t\_A1=dis\_A1\_Ball/v\_A1;
17. num\_A1\_Ball=ceil(t\_A1);
18. v\_B1=2\*0.2145;
19. t\_B1=dis\_B1\_Ball/v\_B1;
20. num\_B1\_Ball=ceil(t\_B1);
21. %Measure the Angle between the player and the ball, and then move towards the ball at that Angle
22. delta\_x\_A1\_Ball=A1\_aim(1)-A1\_pos(1);
23. delta\_y\_A1\_Ball=A1\_aim(2)-A1\_pos(2);
24. theta\_A1\_Ball=atan(delta\_y\_A1\_Ball/delta\_x\_A1\_Ball);
25. delta\_x\_B1\_Ball=B1\_aim(1)-B1\_pos(1);
26. delta\_y\_B1\_Ball=B1\_aim(2)-B1\_pos(2);
27. theta\_B1\_Ball=atan(delta\_y\_B1\_Ball/delta\_x\_B1\_Ball);
28. %If player A1 can arrive faster, depending on the time that player A1 arrives at the ball, each player moves towards the ball until Player A1 arrives at the ball.
29. if num\_A1\_Ball<num\_B1\_Ball
30. %Adjust the current player's movement Angle to the dire
31. if theta\_A1~=theta\_A1\_Ball||theta\_B1~=theta\_B1\_Ball
32. delta\_theta\_A1=theta\_A1\_Ball-theta\_A1;
33. delta\_theta\_B1=theta\_B1\_Ball-theta\_B1;
34. for i=0:2
36. %Calculate the perspective of the players' faces
37. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A1/2;
38. theta\_mid\_B1=theta\_B1+i\*delta\_theta\_B1/2-pi;
39. if B1\_pos(1)<=Ball\_pos(1)
40. theta\_mid\_B1=theta\_mid\_B1+pi;
41. end
42. if A1\_pos(1)>Ball\_pos(1)
43. theta\_mid\_A1=theta\_mid\_A1+pi;
44. end
45. %Draw the pitch, ball, and the players after adjusting the direction
46. run pitch.m
47. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
48. %Draw the player A1 with perspective
49. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
50. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','r','LineWidth',1);
51. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','r','LineWidth',1);
52. %Draw the player B1 with perspective
53. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
54. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1+pi/3)],'Color','b','LineWidth',1);
55. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1-pi/3)],'Color','b','LineWidth',1);
56. pause(0.2)
57. if i<2
58. clf;
59. end
60. end
61. theta\_A1=theta\_mid\_A1;
62. theta\_B1=theta\_mid\_B1;
63. end
64. %Depending on the time that player A1 arrives at the ball,show the dynamic match.
65. for i=1:num\_A1\_Ball
66. run pitch.m;
67. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
68. x\_A1=A1\_pos(1)+i\*delta\_x\_A1\_Ball/num\_A1\_Ball;
69. y\_A1=A1\_pos(2)+i\*delta\_y\_A1\_Ball/num\_A1\_Ball;
70. x\_B1=B1\_pos(1)+i\*delta\_x\_B1\_Ball/num\_B1\_Ball;
71. y\_B1=B1\_pos(2)+i\*delta\_y\_B1\_Ball/num\_B1\_Ball;
72. viscircles([x\_A1 y\_A1],0.2145,'Color','r','LineWidth',0.75);
73. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1+pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
74. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1-pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
75. viscircles([x\_B1 y\_B1],0.2145,'Color','b','LineWidth',0.75);
76. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1+pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
77. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1-pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
78. pause(0.2);
79. if i<num\_A1\_Ball
80. clf;
81. end
82. end
83. % A1 arrives the ball
84. Ball\_right=1;
85. A1\_pos(1)=x\_A1;
86. A1\_pos(2)=y\_A1;
87. B1\_pos(1)=x\_B1;
88. B1\_pos(2)=y\_B1;
89. end
90. %If player B1 can arrive faster, depending on the time that player B1 arrives at the ball, each player moves towards the ball until Player B1 arrives at the ball.
91. if num\_A1\_Ball>num\_B1\_Ball
92. if theta\_A1~=theta\_A1\_Ball||theta\_B1~=theta\_B1\_Ball
93. delta\_theta\_A1=theta\_A1\_Ball-theta\_A1;
94. delta\_theta\_B1=theta\_B1\_Ball-theta\_B1;
95. for i=0:2
96. run pitch.m
97. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A1/2;
98. theta\_mid\_B1=theta\_B1+i\*delta\_theta\_B1/2-pi;
99. if B1\_pos(1)<=Ball\_pos(1)
100. theta\_mid\_B1=theta\_mid\_B1+pi;
101. end
102. if A1\_pos(1)>Ball\_pos(1)
103. theta\_mid\_A1=theta\_mid\_A1+pi;
104. end
105. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
106. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
107. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','r','LineWidth',1);
108. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','r','LineWidth',1);
109. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
110. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1+pi/3)],'Color','b','LineWidth',1);
111. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1-pi/3)],'Color','b','LineWidth',1);
112. pause(0.2)
113. if i<2
114. clf;
115. end
116. end
117. theta\_A1=theta\_mid\_A1;
118. theta\_B1=theta\_mid\_B1;
119. end
120. for i=1:num\_B1\_Ball
121. run pitch.m;
122. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
123. x\_A1=A1\_pos(1)+i\*delta\_x\_A1\_Ball/num\_A1\_Ball;
124. y\_A1=A1\_pos(2)+i\*delta\_y\_A1\_Ball/num\_A1\_Ball;
125. x\_B1=B1\_pos(1)+i\*delta\_x\_B1\_Ball/num\_B1\_Ball;
126. y\_B1=B1\_pos(2)+i\*delta\_y\_B1\_Ball/num\_B1\_Ball;
127. viscircles([x\_A1 y\_A1],0.2145,'Color','r','LineWidth',0.75);
128. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1+pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
129. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1-pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
130. viscircles([x\_B1 y\_B1],0.2145,'Color','b','LineWidth',0.75);
131. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1+pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
132. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1-pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
133. pause(0.2);
134. if i<num\_B1\_Ball
135. clf;
136. end
137. end
138. % A1 arrives the ball
139. Ball\_right=2;
140. A1\_pos(1)=x\_A1;
141. A1\_pos(2)=y\_A1;
142. B1\_pos(1)=x\_B1;
143. B1\_pos(2)=y\_B1;
144. end
145. %If they arrive at the same time, depending on the rand function to choose the winner who can arrive at the ball first.
146. if num\_A1\_Ball==num\_B1\_Ball
147. if theta\_A1~=theta\_A1\_Ball||theta\_B1~=theta\_B1\_Ball
148. delta\_theta\_A1=theta\_A1\_Ball-theta\_A1;
149. delta\_theta\_B1=theta\_B1\_Ball-theta\_B1;
150. for i=0:2
151. run pitch.m
152. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A1/2;
153. theta\_mid\_B1=theta\_B1+i\*delta\_theta\_B1/2-pi;
154. if B1\_pos(1)<=Ball\_pos(1)
155. theta\_mid\_B1=theta\_mid\_B1+pi;
156. end
157. if A1\_pos(1)>Ball\_pos(1)
158. theta\_mid\_A1=theta\_mid\_A1+pi;
159. end
160. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
161. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
162. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','r','LineWidth',1);
163. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','r','LineWidth',1);
164. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
165. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1+pi/3)],'Color','b','LineWidth',1);
166. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1-pi/3)],'Color','b','LineWidth',1);
167. pause(0.2)
168. if i<2
169. clf;
170. end
171. end
172. theta\_A1=theta\_mid\_A1;
173. theta\_B1=theta\_mid\_B1;
174. end
175. for i=1:num\_B1\_Ball-1
176. run pitch.m;
177. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
178. x\_A1=A1\_pos(1)+i\*delta\_x\_A1\_Ball/num\_A1\_Ball;
179. y\_A1=A1\_pos(2)+i\*delta\_y\_A1\_Ball/num\_A1\_Ball;
180. x\_B1=B1\_pos(1)+i\*delta\_x\_B1\_Ball/num\_B1\_Ball;
181. y\_B1=B1\_pos(2)+i\*delta\_y\_B1\_Ball/num\_B1\_Ball;
182. viscircles([x\_A1 y\_A1],0.2145,'Color','r','LineWidth',0.75);
183. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1+pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
184. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1-pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
185. viscircles([x\_B1 y\_B1],0.2145,'Color','b','LineWidth',0.75);
186. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1+pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
187. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1-pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
188. pause(0.2);
189. end
190. if rand(1)>=0.5
191. Ball\_right=1;
192. A1\_pos(1)=Ball\_pos(1);
193. A1\_pos(2)=Ball\_pos(2);
194. B1\_pos(1)=x\_B1;
195. B1\_pos(2)=y\_B1;
196. theta\_A1=theta\_B1-pi/2;
197. run pitch.m;
198. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
199. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
200. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
201. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
202. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
203. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
204. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
205. pause(0.2);
206. else
207. Ball\_right=2;
208. A1\_pos(1)=x\_A1;
209. A1\_pos(2)=y\_A1;
210. B1\_pos(1)=Ball\_pos(1);
211. B1\_pos(2)=Ball\_pos(2);
212. theta\_B1=theta\_A1+pi/2;
213. run pitch.m;
214. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
215. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
216. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
217. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
218. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
219. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
220. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
221. pause(0.2);
222. end
223. end
224. delta\_x\_A1\_B1=B1\_pos(1)-A1\_pos(1);
225. delta\_y\_A1\_B1=B1\_pos(2)-A1\_pos(2);
226. dis\_A1\_B1=sqrt((delta\_x\_A1\_B1)^2+(delta\_y\_A1\_B1)^2);
227. theta\_A1\_B1=atan(delta\_y\_A1\_B1/delta\_x\_A1\_B1);
228. %%If player A1 hode the ball,A1 with the ball moves to the Goal1
229. if Ball\_right==1
230. %A1 Angle Adjustment: Find the orientation Angle of Team B and adjust +-45 degrees in this direction. Choose the 45 degrees closest to the goal.
231. theta\_A1\_next1=theta\_B1+pi/2;
232. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
233. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
234. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
235. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
236. theta\_A1\_next2=theta\_B1-pi/2;
237. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
238. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
239. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
240. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
241. dis\_A1\_next1=sqrt((Goal1\_pos(1)-A1\_posnew\_1(1))^2+(Goal1\_pos(2)-A1\_posnew\_1(2))^2);
242. dis\_next2=sqrt((Goal1\_pos(1)-A1\_posnew\_2(1))^2+(Goal1\_pos(2)-A1\_posnew\_2(2))^2);
243. if dis\_A1\_next1<=dis\_next2
244. theta\_A1=theta\_A1\_next1;
245. else
246. theta\_A1=theta\_A1\_next2;
247. end
248. %Determine the new A1 according to the updated orientation Angle
249. delta\_x\_A1=v\_A1\*cos(theta\_A1);
250. delta\_y\_A1=v\_A1\*sin(theta\_A1);
251. A1\_posnew(1)=A1\_pos(1)+delta\_x\_A1;
252. A1\_posnew(2)=A1\_pos(2)+delta\_y\_A1;
253. %B1 Angle Adjustment:  The original direction stays the same, or it's reversed 90 degrees. Choose the degree closest to the A1.
254. theta\_B1\_next1=theta\_B1;
255. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
256. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
257. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
258. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
259. dis\_B1\_next1=sqrt((A1\_posnew(1)-B1\_posnew\_1(1))^2+(A1\_posnew(2)-B1\_posnew\_1(2))^2);
260. theta\_B1\_next2=theta\_B1-pi;
261. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
262. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
263. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
264. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
265. dis\_B1\_next2=sqrt((A1\_posnew(1)-B1\_posnew\_2(1))^2+(A1\_posnew(2)-B1\_posnew\_2(2))^2);
266. if dis\_B1\_next1<=dis\_B1\_next2
267. theta\_B1=theta\_B1\_next1;
268. else
269. theta\_B1=theta\_B1\_next2;
270. end
271. %Detects whether A1 is positioned beyond the court boundary: If it is,assign the court's maximum Y value or minimum X value .
272. if A1\_posnew(2)>=7
273. A1\_pos(1)=A1\_posnew(1);
274. A1\_pos(2)=7;
275. else
276. if A1\_posnew(2)<=1
277. A1\_pos(1)=A1\_posnew(1);
278. A1\_pos(2)=1;
279. else
280. A1\_pos(1)=A1\_posnew(1);
281. A1\_pos(2)=A1\_posnew(2);
282. end
283. end
284. %If the distance between B1 and A1 with the ball is within its unit velocity, then B1 and ball A1 will coincide, and B1 will obtain the ball.
285. if dis\_A1\_B1<=v\_B1
286. B1\_pos(1)=Ball\_pos(1);
287. B1\_pos(2)=Ball\_pos(2);
288. %Otherwise it will continue to move towards the A1.
289. else
290. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
291. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
292. end
293. Ball\_pos(1)=A1\_pos(1);
294. Ball\_pos(2)=A1\_pos(2);
295. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
296. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
297. run pitch.m;
298. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
299. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
300. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
301. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
302. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
303. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
304. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
305. pause(0.2);
306. end
307. %%If player B1 hode the ball,B1 with the ball moves to the Goal1
308. if Ball\_right==2
309. %B1 Angle Adjustment: Find the orientation Angle of Team A and adjust +-45 degrees in this direction. Choose the 45 degrees closest to the goal.
310. theta\_B1\_next1=theta\_A1+pi/2;
311. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
312. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
313. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
314. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
315. theta\_B1\_next2=theta\_B1-pi/2;
316. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
317. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
318. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
319. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
320. dis\_B1\_next1=sqrt((Goal2\_pos(1)-B1\_posnew\_1(1))^2+(Goal2\_pos(2)-B1\_posnew\_1(2))^2);
321. dis\_B1\_next2=sqrt((Goal2\_pos(1)-B1\_posnew\_2(1))^2+(Goal2\_pos(2)-B1\_posnew\_2(2))^2);
322. if dis\_B1\_next1<=dis\_B1\_next2
323. theta\_B1=theta\_B1\_next1;
324. else
325. theta\_B1=theta\_B1\_next2;
326. end
327. %Determine the new B1 according to the updated orientation Angle
328. delta\_x\_B1=v\_B1\*cos(theta\_B1);
329. delta\_y\_B1=v\_B1\*sin(theta\_B1);
330. B1\_posnew(1)=B1\_pos(1)+delta\_x\_B1;
331. B1\_posnew(2)=B1\_pos(2)+delta\_y\_B1;
332. theta\_B1\_next1=theta\_B1;
333. %A1 Angle Adjustment:  The original direction stays the same, or it's reversed 90 degrees. Choose the degree closest to the ball.
334. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
335. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
336. theta\_A1\_next1=theta\_A1;
337. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
338. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
339. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
340. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
341. dis\_A1\_next1=sqrt((B1\_posnew(1)-A1\_posnew\_1(1))^2+(B1\_posnew(2)-A1\_posnew\_1(2))^2);
342. theta\_A1\_next2=theta\_A1-pi;
343. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
344. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
345. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
346. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
347. dis\_A1\_next2=sqrt((B1\_posnew(1)-A1\_posnew\_2(1))^2+(B1\_posnew(2)-A1\_posnew\_2(2))^2);
348. if dis\_A1\_next1<=dis\_A1\_next2
349. theta\_A1=theta\_A1\_next1;
350. else
351. theta\_A1=theta\_A1\_next2;
352. end
353. %Detects whether B1 is positioned beyond the court boundary: If it is,assign the court's maximum Y value or minimum X value .
354. if B1\_posnew(2)>=7
355. B1\_pos(1)=B1\_posnew(1);
356. B1\_pos(2)=7;
357. else
358. if B1\_posnew(2)<=1
359. B1\_pos(1)=B1\_posnew(1);
360. B1\_pos(2)=1;
361. else
362. B1\_pos(1)=B1\_posnew(1);
363. B1\_pos(2)=B1\_posnew(2);
364. end
365. end
366. %If the distance between A1 and B1 with the ball is within its unit velocity, then B1 and ball A1 will coincide, and A1 will obtain the ball.
367. if dis\_A1\_B1<=v\_A1
368. A1\_pos(1)=Ball\_pos(1);
369. A1\_pos(2)=Ball\_pos(2);
370. %Otherwise it will continue to move towards the goal.
371. else
372. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
373. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
374. end
375. Ball\_pos(1)=B1\_pos(1);
376. Ball\_pos(2)=B1\_pos(2);
377. run pitch.m;
378. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
379. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
380. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
381. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
382. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
383. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
384. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
385. pause(0.2);
386. end
387. %Calculate the distance and Angle between players and their respective goals
388. dis\_A1\_Goal1=sqrt((Goal1\_pos(1)-A1\_pos(1))^2+(Goal1\_pos(2)-A1\_pos(2))^2);
389. delta\_x\_A1\_Goal1=Goal1\_pos(1)-A1\_pos(1);
390. delta\_y\_A1\_Goal1=Goal1\_pos(2)-A1\_pos(2);
391. theta\_A1\_Goal1=atan(delta\_y\_A1\_Goal1/delta\_x\_A1\_Goal1);
392. dis\_B1\_Goal2=sqrt((Goal2\_pos(1)-B1\_pos(1))^2+(Goal2\_pos(2)-B1\_pos(2))^2);
393. delta\_x\_B1\_Goal2=Goal2\_pos(1)-B1\_pos(1);
394. delta\_y\_B1\_Goal2=Goal2\_pos(2)-B1\_pos(2);
395. theta\_B1\_Goal2=atan(delta\_y\_B1\_Goal2/delta\_x\_B1\_Goal2);
396. % When Player A1 holds the ball and is more than 3 meters away from goal and cannot score directly
397. if Ball\_right==1
398. delta\_dis=dis\_A1\_Goal1-3;
399. t\_A1=delta\_dis/v\_A1;
400. num\_A1=ceil(t\_A1);
401. %Player A1 holds the ball and continues towards the goal until the distance between the player and the goal is 3.
402. for i=1:num\_A1
403. if Ball\_right==1
404. run pitch.m;
405. delta\_x\_A1=v\_A1\*cos(theta\_A1);
406. delta\_y\_A1=v\_A1\*sin(theta\_A1);
407. A1\_pos(1)=A1\_pos(1)+delta\_x\_A1;
408. A1\_pos(2)=A1\_pos(2)+delta\_y\_A1;
409. theta\_A1=theta\_A1\_Goal1;
410. Ball\_pos(1)=A1\_pos(1);
411. Ball\_pos(2)=A1\_pos(2);
412. %Player B1 moves towards A1 with the ball.
413. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
414. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
415. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
416. if B1\_pos(1)>A1\_pos(1)
417. theta\_B1=theta\_B1-pi;
418. end
419. %If Player B1 can reach the ball in unit time, then B1 obtains it,otherwise,keep moving towards the ball.
420. if (B1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<B1\_pos(2)+v\_B1\*sin(theta\_B1))
421. B1\_pos(1)=Ball\_pos(1);
422. B1\_pos(2)=Ball\_pos(2);
423. else
424. if (B1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>B1\_pos(2)+v\_B1\*sin(theta\_B1))
425. B1\_pos(1)=Ball\_pos(1);
426. B1\_pos(2)=Ball\_pos(2);
427. else
428. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
429. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
430. end
431. end
432. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
433. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
434. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
435. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
436. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
437. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
438. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
439. pause(0.2);
440. %If the positions of player B1 and the ball coincide after the time is up, the random function determines whether to transfer the ball to B1
441. if B1\_pos(1)==Ball\_pos(1)&&B1\_pos(2)==Ball\_pos(2)
442. if rand(1)>=0.3
443. Ball\_right=2;
444. end
445. end
446. end
447. end
448. %When Player A1 holds the ball and is less than 3 meters away from goal and can score directly
449. delta\_x\_A1\_Goal1=Goal1\_pos(1)-A1\_pos(1);
450. delta\_y\_A1\_Goal1=Goal1\_pos(2)-A1\_pos(2);
451. theta\_A1\_Goal1=atan(delta\_y\_A1\_Goal1/delta\_x\_A1\_Goal1);
452. %The ball flys towards the goal
453. theta\_Ball\_Goal1=theta\_A1\_Goal1;
454. theta\_A1=theta\_Ball\_Goal1;
455. v\_Ball=2;
456. t\_ball=3;
457. dis\_Ball=1/2\*v\_Ball\*t\_ball;
458. delta\_x\_Ball=4\*0.2145\*cos(theta\_Ball\_Goal1);
459. delta\_y\_Ball=4\*0.2145\*sin(theta\_Ball\_Goal1);
460. t\_equal=dis\_Ball/(4\*0.2145);
461. num\_Ball=ceil(t\_equal);
462. if Ball\_right==1
463. for i=1:num\_Ball
464. run pitch.m
465. Ball\_pos(1)=Ball\_pos(1)+delta\_x\_Ball;
466. Ball\_pos(2)=Ball\_pos(2)+delta\_y\_Ball;
467. %Player B moves towards the ball
468. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
469. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
470. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
471. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
472. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
473. if Ball\_pos(1)>Goal1\_pos(1)
474. Ball\_pos(1)=Goal1\_pos(1);
475. Ball\_pos(2)=Goal1\_pos(2);
476. end
477. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
478. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
479. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
480. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
481. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
482. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
483. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
484. pause(0.2);
485. %After the goal, no player holds the ball.
486. Ball\_right=0;
487. end
488. end
490. if Ball\_right==2
491. %B1 Angle Adjustment: Find the orientation Angle of Team A and adjust +-45 degrees in this direction. Choose the 45 degrees closest to the goal.
492. theta\_B1\_next1=theta\_A1+pi/2;
493. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
494. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
495. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
496. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
497. theta\_B1\_next2=theta\_B1-pi/2;
498. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
499. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
500. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
501. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
502. dis\_B1\_next1=sqrt((Goal2\_pos(1)-B1\_posnew\_1(1))^2+(Goal2\_pos(2)-B1\_posnew\_1(2))^2);
503. dis\_B1\_next2=sqrt((Goal2\_pos(1)-B1\_posnew\_2(1))^2+(Goal2\_pos(2)-B1\_posnew\_2(2))^2);
504. if dis\_B1\_next1<=dis\_B1\_next2
505. theta\_B1=theta\_B1\_next1;
506. else
507. theta\_B1=theta\_B1\_next2;
508. end
509. %Determine the new B1 according to the updated orientation Angle
510. delta\_x\_B1=v\_B1\*cos(theta\_B1);
511. delta\_y\_B1=v\_B1\*sin(theta\_B1);
512. B1\_posnew(1)=B1\_pos(1)+delta\_x\_B1;
513. B1\_posnew(2)=B1\_pos(2)+delta\_y\_B1;
514. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
515. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
516. %A1 Angle Adjustment:The original direction stays the same, or it's reversed 90 degrees. Choose the degree closest to the ball.
517. theta\_A1\_next1=theta\_A1;
518. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
519. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
520. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
521. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
522. dis\_A1\_next1=sqrt((B1\_posnew(1)-A1\_posnew\_1(1))^2+(B1\_posnew(2)-A1\_posnew\_1(2))^2);
524. theta\_A1\_next2=theta\_A1-pi;
525. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
526. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
527. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
528. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
529. dis\_A1\_next2=sqrt((B1\_posnew(1)-A1\_posnew\_2(1))^2+(B1\_posnew(2)-A1\_posnew\_2(2))^2);
530. if dis\_A1\_next1<=dis\_A1\_next2
531. theta\_A1=theta\_A1\_next1;
532. else
533. theta\_A1=theta\_A1\_next2;
534. end
535. %Detects whether B1 is positioned beyond the court boundary: If it is,assign the court's maximum Y value or minimum X value .
536. if B1\_posnew(2)>=7
537. B1\_pos(1)=B1\_posnew(1);
538. B1\_pos(2)=7;
539. else
540. if B1\_posnew(2)<=1
541. B1\_pos(1)=B1\_posnew(1);
542. B1\_pos(2)=1;
543. else
544. B1\_pos(1)=B1\_posnew(1);
545. B1\_pos(2)=B1\_posnew(2);
546. end
547. end
548. %If the distance between B1 and A1 with the ball is within A1's unit velocity, then B1 and ball A1 will coincide, and A1 will obtain the ball.
549. if dis\_A1\_B1<=v\_A1
550. A1\_pos(1)=Ball\_pos(1);
551. A1\_pos(2)=Ball\_pos(2);
552. %Otherwise it will continue to move towards the goal.
553. else
554. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
555. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
556. end
557. Ball\_pos(1)=B1\_pos(1);
558. Ball\_pos(2)=B1\_pos(2);
559. run pitch.m;
560. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
561. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
562. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
563. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
564. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
565. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
566. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
567. pause(0.2);
568. end
570. end
571. %%When Player B1 holds the ball and is more than 3 meters away from goal and cannot score directly
572. if Ball\_right==2
573. dis\_B1\_Goal2=sqrt((Goal2\_pos(1)-B1\_pos(1))^2+(Goal2\_pos(2)-B1\_pos(2))^2);
574. delta\_dis=dis\_B1\_Goal2-3;
575. t\_B1=delta\_dis/v\_B1;
576. num\_B1=ceil(t\_B1);
577. %Player B1 holds the ball and continues towards the goal until the distance between the player and the goal is 3.
578. for i=1:num\_B1
579. if Ball\_right==2
580. run pitch.m;
581. theta\_B1=theta\_B1\_Goal2-pi;
582. delta\_x\_B1=v\_B1\*cos(theta\_B1);
583. delta\_y\_B1=v\_B1\*sin(theta\_B1);
584. B1\_pos(1)=B1\_pos(1)+delta\_x\_B1;
585. B1\_pos(2)=B1\_pos(2)+delta\_y\_B1;
586. Ball\_pos(1)=B1\_pos(1);
587. Ball\_pos(2)=B1\_pos(2);
588. %Player A1 moves towards B1 with the ball.
589. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
590. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
591. theta\_A1=atan(delta\_y\_A1/delta\_x\_A1);
592. if B1\_pos(1)<A1\_pos(1)
593. theta\_A1=theta\_A1-pi;
594. end
595. %If Player A1 can reach the ball in unit time, then A1 obtains it,otherwise,keep moving towards the ball.
596. if (A1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<A1\_pos(1)+v\_A1\*cos(theta\_A1)) && (A1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<A1\_pos(2)+v\_A1\*sin(theta\_A1))
597. A1\_pos(1)=Ball\_pos(1);
598. A1\_pos(2)=Ball\_pos(2);
599. else
600. if (A1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>A1\_pos(1)+v\_A1\*cos(theta\_A1)) && (A1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>A1\_pos(2)+v\_B1\*sin(theta\_A1))
601. A1\_pos(1)=Ball\_pos(1);
602. A1\_pos(2)=Ball\_pos(2);
603. else
604. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
605. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
606. end
607. end
608. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
609. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
610. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
611. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
612. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
613. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
614. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
615. if A1\_pos(1)==Ball\_pos(1)&&A1\_pos(2)==Ball\_pos(2)
616. if rand(1)>=0.3
617. Ball\_right=1;
618. end
619. end
620. pause(0.2);
621. end
622. end
623. %When Player B1 holds the ball and is less than 3 meters away from goal and can score directly
624. delta\_x\_B1\_Goal2=Goal2\_pos(1)-B1\_pos(1);
625. delta\_y\_B1\_Goal2=Goal2\_pos(2)-B1\_pos(2);
626. theta\_B1\_Goal2=atan(delta\_y\_B1\_Goal2/delta\_x\_B1\_Goal2);
627. theta\_Ball\_Goal2=theta\_B1\_Goal2;
628. v\_Ball=2;
629. t\_ball=3;
630. dis\_Ball=1/2\*v\_Ball\*t\_ball;
631. delta\_x\_Ball=-4\*0.2145\*cos(theta\_Ball\_Goal2);
632. delta\_y\_Ball=-4\*0.2145\*sin(theta\_Ball\_Goal2);
633. t\_equal=dis\_Ball/(4\*0.2145);
634. num\_Ball=ceil(t\_equal);
635. theta\_B1=theta\_B1\_Goal2-pi;
636. if Ball\_right==2
637. %The ball flies towards the goal
638. for i=1:num\_Ball
639. run pitch.m
640. Ball\_pos(1)=Ball\_pos(1)+delta\_x\_Ball;
641. Ball\_pos(2)=Ball\_pos(2)+delta\_y\_Ball;
642. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
643. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
644. theta\_A1=atan(delta\_y\_A1/delta\_x\_A1);
645. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
646. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
647. if Ball\_pos(1)<Goal2\_pos(1)
648. Ball\_pos(1)=Goal2\_pos(1);
649. Ball\_pos(2)=Goal2\_pos(2);
650. end
651. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
652. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
653. line([A1\_pos(1) A1\_pos(1)-0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)-0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
654. line([A1\_pos(1) A1\_pos(1)-0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)-0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
655. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
656. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
657. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
658. pause(0.2);
659. end
660. end
662. if Ball\_right==1
663. %A1 Angle Adjustment: Find the orientation Angle of Team B and adjust +-45 degrees in this direction. Choose the 45 degrees closest to the goal.
664. theta\_A1\_next1=theta\_B1+pi/2;
665. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
666. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
667. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
668. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
669. theta\_A1\_next2=theta\_B1-pi/2;
670. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
671. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
672. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
673. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
674. dis\_A1\_next1=sqrt((Goal1\_pos(1)-A1\_posnew\_1(1))^2+(Goal1\_pos(2)-A1\_posnew\_1(2))^2);
675. dis\_next2=sqrt((Goal1\_pos(1)-A1\_posnew\_2(1))^2+(Goal1\_pos(2)-A1\_posnew\_2(2))^2);
676. if dis\_A1\_next1<=dis\_next2
677. theta\_A1=theta\_A1\_next1;
678. else
679. theta\_A1=theta\_A1\_next2;
680. end
681. %Determine the new A1 according to the updated orientation Angle
682. delta\_x\_A1=v\_A1\*cos(theta\_A1);
683. delta\_y\_A1=v\_A1\*sin(theta\_A1);
684. A1\_posnew(1)=A1\_pos(1)+delta\_x\_A1;
685. A1\_posnew(2)=A1\_pos(2)+delta\_y\_A1;
686. %B1 Angle Adjustment:  The original direction stays the same, or it's reversed 90 degrees. Choose the degree closest to the ball.
687. theta\_B1\_next1=theta\_B1;
688. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
689. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
690. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
691. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
692. dis\_B1\_next1=sqrt((A1\_posnew(1)-B1\_posnew\_1(1))^2+(A1\_posnew(2)-B1\_posnew\_1(2))^2);
693. theta\_B1\_next2=theta\_B1-pi;
694. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
695. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
696. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
697. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
698. dis\_B1\_next2=sqrt((A1\_posnew(1)-B1\_posnew\_2(1))^2+(A1\_posnew(2)-B1\_posnew\_2(2))^2);
699. if dis\_B1\_next1<=dis\_B1\_next2
700. theta\_B1=theta\_B1\_next1;
701. else
702. theta\_B1=theta\_B1\_next2;
703. end
704. %Detects whether A1 is positioned beyond the court boundary: If it is,assign the court's maximum Y value or minimum X value .
705. if A1\_posnew(2)>=7
706. A1\_pos(1)=A1\_posnew(1);
707. A1\_pos(2)=7;
708. else
709. if A1\_posnew(2)<=1
710. A1\_pos(1)=A1\_posnew(1);
711. A1\_pos(2)=1;
712. else
713. A1\_pos(1)=A1\_posnew(1);
714. A1\_pos(2)=A1\_posnew(2);
715. end
716. end
717. %If the distance between B1 and A1 with the ball is within its unit velocity, then B1 and ball A1 will coincide, and B1 will obtain the ball.
718. if dis\_A1\_B1<=v\_B1
719. B1\_pos(1)=Ball\_pos(1);
720. B1\_pos(2)=Ball\_pos(2);
721. %Otherwise it will continue to move towards the goal
722. else
723. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
724. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
725. end
726. Ball\_pos(1)=A1\_pos(1);
727. Ball\_pos(2)=A1\_pos(2);
728. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
729. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
730. run pitch.m;
731. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
732. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
733. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
734. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
735. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
736. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
737. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
738. pause(0.2);
739. end
741. if Ball\_right==1
742. %When Player A1 holds the ball and is more than 3 meters away from goal and cannot score directly
743. delta\_dis=dis\_A1\_Goal1-3;
744. t\_A1=delta\_dis/v\_A1;
745. num\_A1=ceil(t\_A1);
746. %Player A1 holds the ball and continues towards the goal until the distance between the player and the goal is 3.
747. for i=1:num\_A1
748. if Ball\_right==1
749. run pitch.m;
750. delta\_x\_A1=v\_A1\*cos(theta\_A1);
751. delta\_y\_A1=v\_A1\*sin(theta\_A1);
752. A1\_pos(1)=A1\_pos(1)+delta\_x\_A1;
753. A1\_pos(2)=A1\_pos(2)+delta\_y\_A1;
754. theta\_A1=theta\_A1\_Goal1;
755. Ball\_pos(1)=A1\_pos(1);
756. Ball\_pos(2)=A1\_pos(2);
757. %%Player B1 moves towards A1 with the ball.
758. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
759. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
760. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
761. if B1\_pos(1)>A1\_pos(1)
762. theta\_B1=theta\_B1-pi;
763. end
764. %If Player B1 can reach the ball in unit time, then B1 obtains it,otherwise,keep moving towards the ball.
765. if (B1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<B1\_pos(2)+v\_B1\*sin(theta\_B1))
766. B1\_pos(1)=Ball\_pos(1);
767. B1\_pos(2)=Ball\_pos(2);
768. else
769. if (B1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>B1\_pos(2)+v\_B1\*sin(theta\_B1))
770. B1\_pos(1)=Ball\_pos(1);
771. B1\_pos(2)=Ball\_pos(2);
772. else
773. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
774. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
775. end
776. end
777. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
778. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
779. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
780. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
781. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
782. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
783. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
784. pause(0.2);
785. if B1\_pos(1)==Ball\_pos(1)&&B1\_pos(2)==Ball\_pos(2)
786. if rand(1)>=0.3
787. Ball\_right=2;
788. end
789. end
790. end
791. end
792. %When Player A1 holds the ball and is less than 3 meters away from goal and can score directly
793. delta\_x\_A1\_Goal1=Goal1\_pos(1)-A1\_pos(1);
794. delta\_y\_A1\_Goal1=Goal1\_pos(2)-A1\_pos(2);
795. theta\_A1\_Goal1=atan(delta\_y\_A1\_Goal1/delta\_x\_A1\_Goal1);
796. theta\_Ball\_Goal1=theta\_A1\_Goal1;
797. theta\_A1=theta\_Ball\_Goal1;
798. v\_Ball=2;
799. t\_ball=3;
800. dis\_Ball=1/2\*v\_Ball\*t\_ball;
801. delta\_x\_Ball=4\*0.2145\*cos(theta\_Ball\_Goal1);
802. delta\_y\_Ball=4\*0.2145\*sin(theta\_Ball\_Goal1);
803. t\_equal=dis\_Ball/(4\*0.2145);
804. num\_Ball=ceil(t\_equal);
805. if Ball\_right==1
806. %The ball flies towards the goal
807. for i=1:num\_Ball
808. run pitch.m
809. Ball\_pos(1)=Ball\_pos(1)+delta\_x\_Ball;
810. Ball\_pos(2)=Ball\_pos(2)+delta\_y\_Ball;
811. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
812. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
813. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
814. %Player B moves towards the ball
815. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
816. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
817. if Ball\_pos(1)>Goal1\_pos(1)
818. Ball\_pos(1)=Goal1\_pos(1);
819. Ball\_pos(2)=Goal1\_pos(2);
820. end
821. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
822. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
823. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
824. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
825. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
826. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
827. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
828. pause(0.2);
829. %After the goal, no player holds the ball.
830. Ball\_right=0;
831. end
832. end
834. if Ball\_right==2
835. %B1 Angle Adjustment: Find the orientation Angle of Team A and adjust +-45 degrees in this direction. Choose the 45 degrees closest to the goal.
836. theta\_B1\_next1=theta\_A1+pi/2;
837. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
838. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
839. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
840. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
841. theta\_B1\_next2=theta\_B1-pi/2;
842. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
843. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
844. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
845. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
846. dis\_B1\_next1=sqrt((Goal2\_pos(1)-B1\_posnew\_1(1))^2+(Goal2\_pos(2)-B1\_posnew\_1(2))^2);
847. dis\_B1\_next2=sqrt((Goal2\_pos(1)-B1\_posnew\_2(1))^2+(Goal2\_pos(2)-B1\_posnew\_2(2))^2);
848. if dis\_B1\_next1<=dis\_B1\_next2
849. theta\_B1=theta\_B1\_next1;
850. else
851. theta\_B1=theta\_B1\_next2;
852. end
853. %Determine the new B1 according to the updated orientation Angle
854. delta\_x\_B1=v\_B1\*cos(theta\_B1);
855. delta\_y\_B1=v\_B1\*sin(theta\_B1);
856. B1\_posnew(1)=B1\_pos(1)+delta\_x\_B1;
857. B1\_posnew(2)=B1\_pos(2)+delta\_y\_B1;
858. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
859. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
860. %A1 Angle Adjustment:  The original direction stays the same, or it's reversed 90 degrees. Choose the degree closest to the ball.
861. theta\_A1\_next1=theta\_A1;
862. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
863. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
864. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
865. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
866. dis\_A1\_next1=sqrt((B1\_posnew(1)-A1\_posnew\_1(1))^2+(B1\_posnew(2)-A1\_posnew\_1(2))^2);
867. theta\_A1\_next2=theta\_A1-pi;
868. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
869. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
870. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
871. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
872. dis\_A1\_next2=sqrt((B1\_posnew(1)-A1\_posnew\_2(1))^2+(B1\_posnew(2)-A1\_posnew\_2(2))^2);
873. if dis\_A1\_next1<=dis\_A1\_next2
874. theta\_A1=theta\_A1\_next1;
875. else
876. theta\_A1=theta\_A1\_next2;
877. end
878. %Detects whether B1 is positioned beyond the court boundary: If it is,assign the court's maximum Y value or minimum X value .
879. if B1\_posnew(2)>=7
880. B1\_pos(1)=B1\_posnew(1);
881. B1\_pos(2)=7;
882. else
883. if B1\_posnew(2)<=1
884. B1\_pos(1)=B1\_posnew(1);
885. B1\_pos(2)=1;
886. else
887. B1\_pos(1)=B1\_posnew(1);
888. B1\_pos(2)=B1\_posnew(2);
889. end
890. end
891. if dis\_A1\_B1<=v\_A1
892. A1\_pos(1)=Ball\_pos(1);
893. A1\_pos(2)=Ball\_pos(2);
894. else
895. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
896. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
897. end
898. Ball\_pos(1)=B1\_pos(1);
899. Ball\_pos(2)=B1\_pos(2);
900. run pitch.m;
901. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
902. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
903. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
904. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
905. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
906. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
907. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
908. pause(0.2);
909. end
910. end
911. end

### ***A2\_B2.m***

1. A1\_pos=[3 3];
2. A2\_pos=[1 4];
3. B1\_pos=[6 6];
4. B2\_pos=[10 4];
5. Ball\_pos=[7 5];
6. Goal1\_pos=[10 4];
7. Goal1\_t\_pos=[10 5.3];
8. Goal1\_b\_pos=[10 2.7];
9. Goal2\_pos=[1 4];
10. Goal2\_t\_pos=[1 5.3];
11. Goal2\_b\_pos=[1 2.7];
12. Ball\_right=0;
13. theta\_A1=0;
14. theta\_A2=pi;
15. theta\_B1=pi;
16. theta\_B2=0;
17. A1\_aim=Ball\_pos;
18. B1\_aim=Ball\_pos;
19. dis\_A1\_Ball=sqrt((A1\_aim(1)-A1\_pos(1))^2+(A1\_aim(2)-A1\_pos(2))^2);
20. dis\_B1\_Ball=sqrt((B1\_aim(1)-B1\_pos(1))^2+(B1\_aim(2)-B1\_pos(2))^2);
21. v\_A1=2\*0.2145;
22. t\_A1=dis\_A1\_Ball/v\_A1;
23. num\_A1\_Ball=ceil(t\_A1);
24. v\_A2=2\*0.18;
25. v\_B1=2\*0.2145;
26. t\_B1=dis\_B1\_Ball/v\_B1;
27. num\_B1\_Ball=ceil(t\_B1);
28. v\_B2=2\*0.18;
29. delta\_x\_A1\_Ball=A1\_aim(1)-A1\_pos(1);
30. delta\_y\_A1\_Ball=A1\_aim(2)-A1\_pos(2);
31. theta\_A1\_Ball=atan(delta\_y\_A1\_Ball/delta\_x\_A1\_Ball);
32. delta\_x\_B1\_Ball=B1\_aim(1)-B1\_pos(1);
33. delta\_y\_B1\_Ball=B1\_aim(2)-B1\_pos(2);
34. theta\_B1\_Ball=atan(delta\_y\_B1\_Ball/delta\_x\_B1\_Ball);
35. frame=3;
36. if num\_A1\_Ball<num\_B1\_Ball
37. if theta\_A1~=theta\_A1\_Ball||theta\_B1~=theta\_B1\_Ball
38. delta\_theta\_A1=theta\_A1\_Ball-theta\_A1;
39. delta\_theta\_B1=theta\_B1\_Ball-theta\_B1;
40. for i=0:2
41. run pitch.m
42. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A1/2;
43. theta\_mid\_B1=theta\_B1+i\*delta\_theta\_B1/2-pi;
44. if B1\_pos(1)<=Ball\_pos(1)
45. theta\_mid\_B1=theta\_mid\_B1+pi;
46. end
47. if A1\_pos(1)>Ball\_pos(1)
48. theta\_mid\_A1=theta\_mid\_A1+pi;
49. end
50. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
51. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
52. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','r','LineWidth',1);
53. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','r','LineWidth',1);
54. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
55. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1+pi/3)],'Color','b','LineWidth',1);
56. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1-pi/3)],'Color','b','LineWidth',1);
57. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
58. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
59. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
60. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
61. pause(0.2)
62. if i<2
63. clf;
64. end
65. end
66. theta\_A1=theta\_mid\_A1;
67. theta\_B1=theta\_mid\_B1;
68. end
69. for i=1:num\_A1\_Ball
70. run pitch.m;
71. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
72. x\_A1=A1\_pos(1)+i\*delta\_x\_A1\_Ball/num\_A1\_Ball;
73. y\_A1=A1\_pos(2)+i\*delta\_y\_A1\_Ball/num\_A1\_Ball;
74. x\_B1=B1\_pos(1)+i\*delta\_x\_B1\_Ball/num\_B1\_Ball;
75. y\_B1=B1\_pos(2)+i\*delta\_y\_B1\_Ball/num\_B1\_Ball;
76. viscircles([x\_A1 y\_A1],0.2145,'Color','r','LineWidth',0.75);
77. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1+pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
78. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1-pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
79. viscircles([x\_B1 y\_B1],0.2145,'Color','b','LineWidth',0.75);
80. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1+pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
81. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1-pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
82. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
83. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
84. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
85. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
86. pause(0.2);
87. if i<num\_A1\_Ball
88. clf;
89. end
90. end
91. Ball\_right=1;
92. A1\_pos(1)=x\_A1;
93. A1\_pos(2)=y\_A1;
94. B1\_pos(1)=x\_B1;
95. B1\_pos(2)=y\_B1;
96. end
97. if num\_A1\_Ball>num\_B1\_Ball
98. if theta\_A1~=theta\_A1\_Ball||theta\_B1~=theta\_B1\_Ball
99. delta\_theta\_A1=theta\_A1\_Ball-theta\_A1;
100. delta\_theta\_B1=theta\_B1\_Ball-theta\_B1;
101. for i=0:2
102. run pitch.m
103. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A1/2;
104. theta\_mid\_B1=theta\_B1+i\*delta\_theta\_B1/2-pi;
105. if B1\_pos(1)<=Ball\_pos(1)
106. theta\_mid\_B1=theta\_mid\_B1+pi;
107. end
108. if A1\_pos(1)>Ball\_pos(1)
109. theta\_mid\_A1=theta\_mid\_A1+pi;
110. end
111. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
112. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
113. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','r','LineWidth',1);
114. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','r','LineWidth',1);
115. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
116. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1+pi/3)],'Color','b','LineWidth',1);
117. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1-pi/3)],'Color','b','LineWidth',1);
118. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
119. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
120. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
121. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
122. pause(0.2)
123. if i<2
124. clf;
125. end
126. end
127. theta\_A1=theta\_mid\_A1;
128. theta\_B1=theta\_mid\_B1;
129. end
130. for i=1:num\_B1\_Ball
131. run pitch.m;
132. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
133. x\_A1=A1\_pos(1)+i\*delta\_x\_A1\_Ball/num\_A1\_Ball;
134. y\_A1=A1\_pos(2)+i\*delta\_y\_A1\_Ball/num\_A1\_Ball;
135. x\_B1=B1\_pos(1)+i\*delta\_x\_B1\_Ball/num\_B1\_Ball;
136. y\_B1=B1\_pos(2)+i\*delta\_y\_B1\_Ball/num\_B1\_Ball;
137. viscircles([x\_A1 y\_A1],0.2145,'Color','r','LineWidth',0.75);
138. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1+pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
139. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1-pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
140. viscircles([x\_B1 y\_B1],0.2145,'Color','b','LineWidth',0.75);
141. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1+pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
142. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1-pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
143. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
144. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
145. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
146. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
147. pause(0.2);
148. if i<num\_B1\_Ball
149. clf;
150. end
151. end
152. Ball\_right=2;
153. A1\_pos(1)=x\_A1;
154. A1\_pos(2)=y\_A1;
155. B1\_pos(1)=x\_B1;
156. B1\_pos(2)=y\_B1;
157. end
158. if num\_A1\_Ball==num\_B1\_Ball
159. if theta\_A1~=theta\_A1\_Ball||theta\_B1~=theta\_B1\_Ball
160. delta\_theta\_A1=theta\_A1\_Ball-theta\_A1;
161. delta\_theta\_B1=theta\_B1\_Ball-theta\_B1;
162. for i=0:2
163. run pitch.m
164. theta\_mid\_A1=theta\_A1+i\*delta\_theta\_A1/2;
165. theta\_mid\_B1=theta\_B1+i\*delta\_theta\_B1/2-pi;
166. if B1\_pos(1)<=Ball\_pos(1)
167. theta\_mid\_B1=theta\_mid\_B1+pi;
168. end
169. if A1\_pos(1)>Ball\_pos(1)
170. theta\_mid\_A1=theta\_mid\_A1+pi;
171. end
172. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
173. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
174. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1+pi/3)],'Color','r','LineWidth',1);
175. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_mid\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_mid\_A1-pi/3)],'Color','r','LineWidth',1);
176. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
177. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1+pi/3)],'Color','b','LineWidth',1);
178. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_mid\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_mid\_B1-pi/3)],'Color','b','LineWidth',1);
179. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
180. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
181. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
182. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
183. pause(0.2)
184. if i<2
185. clf;
186. end
187. end
188. theta\_A1=theta\_mid\_A1;
189. theta\_B1=theta\_mid\_B1;
190. end
191. for i=1:num\_B1\_Ball-1
192. run pitch.m;
193. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
194. x\_A1=A1\_pos(1)+i\*delta\_x\_A1\_Ball/num\_A1\_Ball;
195. y\_A1=A1\_pos(2)+i\*delta\_y\_A1\_Ball/num\_A1\_Ball;
196. x\_B1=B1\_pos(1)+i\*delta\_x\_B1\_Ball/num\_B1\_Ball;
197. y\_B1=B1\_pos(2)+i\*delta\_y\_B1\_Ball/num\_B1\_Ball;
198. viscircles([x\_A1 y\_A1],0.2145,'Color','r','LineWidth',0.75);
199. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1+pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
200. line([x\_A1 x\_A1+0.2145\*cos(theta\_A1-pi/3)],[y\_A1 y\_A1+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
201. viscircles([x\_B1 y\_B1],0.2145,'Color','b','LineWidth',0.75);
202. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1+pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
203. line([x\_B1 x\_B1+0.2145\*cos(theta\_B1-pi/3)],[y\_B1 y\_B1+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
204. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
205. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
206. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
207. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
208. pause(0.2);
209. end
210. if rand(1)>=0.5
211. Ball\_right=1;
212. A1\_pos(1)=Ball\_pos(1);
213. A1\_pos(2)=Ball\_pos(2);
214. B1\_pos(1)=x\_B1;
215. B1\_pos(2)=y\_B1;
216. theta\_A1=theta\_B1-pi/2;
217. run pitch.m;
218. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
219. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
220. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
221. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
222. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
223. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
224. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
225. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
226. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
227. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
228. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
229. pause(0.2);
230. else
231. Ball\_right=2;
232. A1\_pos(1)=x\_A1;
233. A1\_pos(2)=y\_A1;
234. B1\_pos(1)=Ball\_pos(1);
235. B1\_pos(2)=Ball\_pos(2);
236. theta\_B1=theta\_A1+pi/2;
237. run pitch.m;
238. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
239. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
240. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
241. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
242. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
243. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
244. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
245. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
246. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
247. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
248. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
249. pause(0.2);
250. end
251. end
252. delta\_x\_A1\_B1=B1\_pos(1)-A1\_pos(1);
253. delta\_y\_A1\_B1=B1\_pos(2)-A1\_pos(2);
254. dis\_A1\_B1=sqrt((delta\_x\_A1\_B1)^2+(delta\_y\_A1\_B1)^2);
255. theta\_A1\_B1=atan(delta\_y\_A1\_B1/delta\_x\_A1\_B1);
256. if Ball\_right==1
257. theta\_A1\_next1=theta\_B1+pi/2;
258. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
259. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
260. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
261. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
262. theta\_A1\_next2=theta\_B1-pi/2;
263. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
264. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
265. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
266. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
267. dis\_A1\_next1=sqrt((Goal1\_pos(1)-A1\_posnew\_1(1))^2+(Goal1\_pos(2)-A1\_posnew\_1(2))^2);
268. dis\_next2=sqrt((Goal1\_pos(1)-A1\_posnew\_2(1))^2+(Goal1\_pos(2)-A1\_posnew\_2(2))^2);
269. if dis\_A1\_next1<=dis\_next2
270. theta\_A1=theta\_A1\_next1;
271. else
272. theta\_A1=theta\_A1\_next2;
273. end
274. delta\_x\_A1=v\_A1\*cos(theta\_A1);
275. delta\_y\_A1=v\_A1\*sin(theta\_A1);
276. A1\_posnew(1)=A1\_pos(1)+delta\_x\_A1;
277. A1\_posnew(2)=A1\_pos(2)+delta\_y\_A1;
278. theta\_B1\_next1=theta\_B1;
279. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
280. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
281. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
282. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
283. dis\_B1\_next1=sqrt((A1\_posnew(1)-B1\_posnew\_1(1))^2+(A1\_posnew(2)-B1\_posnew\_1(2))^2);
284. theta\_B1\_next2=theta\_B1-pi;
285. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
286. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
287. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
288. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
289. dis\_B1\_next2=sqrt((A1\_posnew(1)-B1\_posnew\_2(1))^2+(A1\_posnew(2)-B1\_posnew\_2(2))^2);
290. if dis\_B1\_next1<=dis\_B1\_next2
291. theta\_B1=theta\_B1\_next1;
292. else
293. theta\_B1=theta\_B1\_next2;
294. end
296. if A1\_posnew(2)>=7
297. A1\_pos(1)=A1\_posnew(1);
298. A1\_pos(2)=7;
299. else
300. if A1\_posnew(2)<=1
301. A1\_pos(1)=A1\_posnew(1);
302. A1\_pos(2)=1;
303. else
304. A1\_pos(1)=A1\_posnew(1);
305. A1\_pos(2)=A1\_posnew(2);
306. end
307. end
308. if dis\_A1\_B1<=v\_B1
309. B1\_pos(1)=Ball\_pos(1);
310. B1\_pos(2)=Ball\_pos(2);
311. else
312. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
313. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
314. end
315. Ball\_pos(1)=A1\_pos(1);
316. Ball\_pos(2)=A1\_pos(2);
317. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
318. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
319. run pitch.m;
320. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
321. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
322. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
323. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
324. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
325. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
326. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
327. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
328. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
329. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
330. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
331. pause(0.2);
332. end
333. if Ball\_right==2
334. theta\_B1\_next1=theta\_A1+pi/2;
335. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
336. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
337. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
338. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
339. theta\_B1\_next2=theta\_B1-pi/2;
340. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
341. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
342. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
343. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
344. dis\_B1\_next1=sqrt((Goal2\_pos(1)-B1\_posnew\_1(1))^2+(Goal2\_pos(2)-B1\_posnew\_1(2))^2);
345. dis\_B1\_next2=sqrt((Goal2\_pos(1)-B1\_posnew\_2(1))^2+(Goal2\_pos(2)-B1\_posnew\_2(2))^2);
346. if dis\_B1\_next1<=dis\_B1\_next2
347. theta\_B1=theta\_B1\_next1;
348. else
349. theta\_B1=theta\_B1\_next2;
350. end
351. delta\_x\_B1=v\_B1\*cos(theta\_B1);
352. delta\_y\_B1=v\_B1\*sin(theta\_B1);
353. B1\_posnew(1)=B1\_pos(1)+delta\_x\_B1;
354. B1\_posnew(2)=B1\_pos(2)+delta\_y\_B1;
355. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
356. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
357. theta\_A1\_next1=theta\_A1;
358. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
359. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
360. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
361. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
362. dis\_A1\_next1=sqrt((B1\_posnew(1)-A1\_posnew\_1(1))^2+(B1\_posnew(2)-A1\_posnew\_1(2))^2);
363. theta\_A1\_next2=theta\_A1-pi;
364. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
365. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
366. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
367. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
368. dis\_A1\_next2=sqrt((B1\_posnew(1)-A1\_posnew\_2(1))^2+(B1\_posnew(2)-A1\_posnew\_2(2))^2);
369. if dis\_A1\_next1<=dis\_A1\_next2
370. theta\_A1=theta\_A1\_next1;
371. else
372. theta\_A1=theta\_A1\_next2;
373. end
375. if B1\_posnew(2)>=7
376. B1\_pos(1)=B1\_posnew(1);
377. B1\_pos(2)=7;
378. else
379. if B1\_posnew(2)<=1
380. B1\_pos(1)=B1\_posnew(1);
381. B1\_pos(2)=1;
382. else
383. B1\_pos(1)=B1\_posnew(1);
384. B1\_pos(2)=B1\_posnew(2);
385. end
386. end
387. if dis\_A1\_B1<=v\_A1
388. A1\_pos(1)=Ball\_pos(1);
389. A1\_pos(2)=Ball\_pos(2);
390. else
391. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
392. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
393. end
394. Ball\_pos(1)=B1\_pos(1);
395. Ball\_pos(2)=B1\_pos(2);
396. run pitch.m;
397. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
398. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
399. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
400. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
401. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
402. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
403. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
404. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
405. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
406. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
407. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
408. pause(0.2);
409. end
410. dis\_A1\_Goal1=sqrt((Goal1\_pos(1)-A1\_pos(1))^2+(Goal1\_pos(2)-A1\_pos(2))^2);
411. dis\_A1\_Goal1\_t=sqrt((Goal1\_t\_pos(1)-A1\_pos(1))^2+(Goal1\_t\_pos(2)-A1\_pos(2))^2);
412. dis\_A1\_Goal1\_b=sqrt((Goal1\_b\_pos(1)-A1\_pos(1))^2+(Goal1\_b\_pos(2)-A1\_pos(2))^2);
413. dis\_B1\_Goal2=sqrt((Goal2\_pos(1)-B1\_pos(1))^2+(Goal2\_pos(2)-B1\_pos(2))^2);
414. dis\_B1\_Goal2\_t=sqrt((Goal2\_t\_pos(1)-B1\_pos(1))^2+(Goal2\_t\_pos(2)-B1\_pos(2))^2);
415. dis\_B1\_Goal2\_b=sqrt((Goal2\_b\_pos(1)-B1\_pos(1))^2+(Goal2\_b\_pos(2)-B1\_pos(2))^2);
416. delta\_x\_A1\_Goal1=Goal1\_pos(1)-A1\_pos(1);
417. delta\_y\_A1\_Goal1=Goal1\_pos(2)-A1\_pos(2);
418. delta\_x\_A1\_Goal1\_t=Goal1\_t\_pos(1)-A1\_pos(1);
419. delta\_y\_A1\_Goal1\_t=Goal1\_t\_pos(2)-A1\_pos(2);
420. delta\_x\_A1\_Goal1\_b=Goal1\_b\_pos(1)-A1\_pos(1);
421. delta\_y\_A1\_Goal1\_b=Goal1\_b\_pos(2)-A1\_pos(2);
422. delta\_x\_B1\_Goal2=Goal2\_pos(1)-B1\_pos(1);
423. delta\_y\_B1\_Goal2=Goal2\_pos(2)-B1\_pos(2);
424. delta\_x\_B1\_Goal2\_t=Goal2\_t\_pos(1)-B1\_pos(1);
425. delta\_y\_B1\_Goal2\_t=Goal2\_t\_pos(2)-B1\_pos(2);
426. delta\_x\_B1\_Goal2\_b=Goal2\_b\_pos(1)-B1\_pos(1);
427. delta\_y\_B1\_Goal2\_b=Goal2\_b\_pos(2)-B1\_pos(2);
428. theta\_A1\_Goal1=atan(delta\_y\_A1\_Goal1/delta\_x\_A1\_Goal1);
429. theta\_B1\_Goal2=atan(delta\_y\_B1\_Goal2/delta\_x\_B1\_Goal2);
430. theta\_A1\_Goal1\_t=atan(delta\_y\_A1\_Goal1\_t/delta\_x\_A1\_Goal1\_t);
431. theta\_B1\_Goal2\_t=atan(delta\_y\_B1\_Goal2\_t/delta\_x\_B1\_Goal2\_t);
432. theta\_A1\_Goal1\_b=atan(delta\_y\_A1\_Goal1\_b/delta\_x\_A1\_Goal1\_b);
433. theta\_B1\_Goal2\_b=atan(delta\_y\_B1\_Goal2\_b/delta\_x\_B1\_Goal2\_b);
434. if Ball\_right==1
435. t\_A1\_t=dis\_A1\_Goal1\_t/v\_A1;
436. num\_A1\_t=ceil(t\_A1\_t);
437. t\_A1\_b=dis\_A1\_Goal1\_b/v\_A1;
438. num\_A1\_b=ceil(t\_A1\_b);
439. if num\_A1\_t< num\_A1\_b
440. Goal1\_pos=Goal1\_t\_pos;
441. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
442. dis\_A1\_Goal1=dis\_A1\_Goal1\_t;
443. end
444. if num\_A1\_b< num\_A1\_t
445. Goal1\_pos=Goal1\_b\_pos;
446. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
447. dis\_A1\_Goal1=dis\_A1\_Goal1\_b;
448. end
449. if num\_A1\_t==num\_A1\_b
450. if rand(1)>=0.5
451. Goal1\_pos=Goal1\_t\_pos;
452. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
453. dis\_A1\_Goal1=dis\_A1\_Goal1\_t;
454. else
455. Goal1\_pos=Goal1\_b\_pos;
456. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
457. dis\_A1\_Goal1=dis\_A1\_Goal1\_b;
458. end
459. end

462. delta\_dis=dis\_A1\_Goal1-3;
463. t\_A1=delta\_dis/v\_A1;
464. delta\_dis\_t=dis\_A1\_Goal1\_t-3;
465. t\_A1\_t=delta\_dis\_t/v\_A1;
466. delta\_dis\_b=dis\_A1\_Goal1\_b-3;
467. t\_A1\_b=delta\_dis\_b/v\_A1;
468. num\_A1=ceil(t\_A1);
469. num\_A1\_t=ceil(t\_A1\_t);
470. num\_A1\_b=ceil(t\_A1\_b);
472. if num\_A1>0
473. for i=1:num\_A1
474. if Ball\_right==1
475. run pitch.m;
476. theta\_A1=theta\_A1\_Goal1;
477. delta\_x\_A1=v\_A1\*cos(theta\_A1);
478. delta\_y\_A1=v\_A1\*sin(theta\_A1);
479. A1\_pos(1)=A1\_pos(1)+delta\_x\_A1;
480. if abs(A1\_pos(2)+delta\_y\_A1-Goal1\_pos(2))<=abs(A1\_pos(2)-delta\_y\_A1-Goal1\_pos(2))
481. A1\_pos(2)=A1\_pos(2)+delta\_y\_A1;
482. else
483. A1\_pos(2)=A1\_pos(2)-delta\_y\_A1;
484. end
485. Ball\_pos(1)=A1\_pos(1);
486. Ball\_pos(2)=A1\_pos(2);
487. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
488. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
489. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
490. if B1\_pos(1)>A1\_pos(1)
491. theta\_B1=theta\_B1-pi;
492. end
493. if (B1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<B1\_pos(2)+v\_B1\*sin(theta\_B1))
494. B1\_pos(1)=Ball\_pos(1);
495. B1\_pos(2)=Ball\_pos(2);
496. else
497. if (B1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>B1\_pos(2)+v\_B1\*sin(theta\_B1))
498. B1\_pos(1)=Ball\_pos(1);
499. B1\_pos(2)=Ball\_pos(2);
500. else
501. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
502. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
503. end
504. end
505. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
506. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
507. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
508. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
509. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
510. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
511. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
512. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
513. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
514. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
515. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
516. pause(0.2);
517. if B1\_pos(1)==Ball\_pos(1)&&B1\_pos(2)==Ball\_pos(2)
518. if rand(1)>=0.3
519. Ball\_right=2;
520. end
521. end
522. end
523. end
524. end
525. r=-1;
526. dis\_Ball=sqrt((Goal1\_pos(1)-Ball\_pos(1))^2+(Goal1\_pos(2)-Ball\_pos(2))^2);
527. dis\_A1\_B1=sqrt((B1\_pos(1)-A1\_pos(1))^2+(B1\_pos(2)-A1\_pos(2))^2);
528. delta\_dis=dis\_Ball-frame\*4\*0.2145;
529. t\_dis=delta\_dis/v\_A1;
530. num\_dis=ceil(t\_dis);
532. for i=1:num\_dis
533. if dis\_Ball >= frame\*4\*0.2145
534. if dis\_A1\_B1 >= v\_B1
535. delta\_x\_A1=v\_A1\*cos(theta\_A1);
536. delta\_y\_A1=v\_A1\*sin(theta\_A1);
537. A1\_pos(1)=A1\_pos(1)+delta\_x\_A1;
538. A1\_pos(2)=A1\_pos(2)+delta\_y\_A1;
539. theta\_A1=theta\_A1\_Goal1;
540. Ball\_pos(1)=A1\_pos(1);
541. Ball\_pos(2)=A1\_pos(2);
542. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
543. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
544. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
545. if B1\_pos(1)>A1\_pos(1)
546. theta\_B1=theta\_B1-pi;
547. end
548. if (B1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<B1\_pos(2)+v\_B1\*sin(theta\_B1))
549. B1\_pos(1)=Ball\_pos(1);
550. B1\_pos(2)=Ball\_pos(2);
551. else
552. if (B1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>B1\_pos(2)+v\_B1\*sin(theta\_B1))
553. B1\_pos(1)=Ball\_pos(1);
554. B1\_pos(2)=Ball\_pos(2);
555. else
556. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
557. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
558. end
559. end
560. run pitch.m;
561. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
562. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
563. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
564. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
565. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
566. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
567. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
568. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
569. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
570. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
571. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
572. pause(0.2);
573. if B1\_pos(1)==Ball\_pos(1)&&B1\_pos(2)==Ball\_pos(2)
574. if rand(1)>=0.3
575. Ball\_right=2;
576. end
577. end
578. end
579. end
580. end

583. delta\_x\_A1\_Goal1=Goal1\_pos(1)-A1\_pos(1);
584. delta\_y\_A1\_Goal1=Goal1\_pos(2)-A1\_pos(2);
585. theta\_A1\_Goal1=atan(delta\_y\_A1\_Goal1/delta\_x\_A1\_Goal1);
586. theta\_Ball\_Goal1=theta\_A1\_Goal1;
587. theta\_A1=theta\_Ball\_Goal1;
588. v\_Ball=2;
589. t\_ball=3;
590. dis\_Ball=sqrt((Goal1\_pos(1)-Ball\_pos(1))^2+(Goal1\_pos(2)-Ball\_pos(2))^2);
591. delta\_x\_Ball=4\*0.2145\*cos(theta\_Ball\_Goal1);
592. delta\_y\_Ball=4\*0.2145\*sin(theta\_Ball\_Goal1);
593. t\_equal=dis\_Ball/(4\*0.2145);
594. num\_Ball=ceil(t\_equal);
595. if Ball\_right==1
596. for i=1:num\_Ball
597. run pitch.m
598. Ball\_pos(1)=Ball\_pos(1)+delta\_x\_Ball;
599. Ball\_pos(2)=Ball\_pos(2)+delta\_y\_Ball;
600. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
601. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
602. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
603. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
604. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
605. if Goal1\_pos(2)==Goal1\_t\_pos(2)
606. a=1;
607. else
608. a=-1;
609. end
610. B2\_pos(2)=B2\_pos(2)+a\*(i-1)\*v\_B2;
611. if B2\_pos(2) >= Goal1\_t\_pos(2)
612. B2\_pos(2)=Goal1\_t\_pos(2);
613. end
614. if B2\_pos(2) <= Goal1\_b\_pos(2)
615. B2\_pos(2)=Goal1\_b\_pos(2);
616. end
617. if Ball\_pos(1)>Goal1\_pos(1)
618. Ball\_pos(1)=Goal1\_pos(1);
619. Ball\_pos(2)=Goal1\_pos(2);
620. end
621. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
622. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
623. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
624. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
625. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
626. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
627. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
628. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
629. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
630. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
631. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
632. pause(0.2);
633. Ball\_right=0;
634. end
635. end
636. Goal1\_pos=[10 4];
637. if Ball\_right==2
638. theta\_B1\_next1=theta\_A1+pi/2;
639. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
640. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
641. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
642. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
643. theta\_B1\_next2=theta\_B1-pi/2;
644. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
645. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
646. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
647. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
648. dis\_B1\_next1=sqrt((Goal2\_pos(1)-B1\_posnew\_1(1))^2+(Goal2\_pos(2)-B1\_posnew\_1(2))^2);
649. dis\_B1\_next2=sqrt((Goal2\_pos(1)-B1\_posnew\_2(1))^2+(Goal2\_pos(2)-B1\_posnew\_2(2))^2);
650. if dis\_B1\_next1<=dis\_B1\_next2
651. theta\_B1=theta\_B1\_next1;
652. else
653. theta\_B1=theta\_B1\_next2;
654. end
655. delta\_x\_B1=v\_B1\*cos(theta\_B1);
656. delta\_y\_B1=v\_B1\*sin(theta\_B1);
657. B1\_posnew(1)=B1\_pos(1)+delta\_x\_B1;
658. B1\_posnew(2)=B1\_pos(2)+delta\_y\_B1;
659. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
660. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
661. theta\_A1\_next1=theta\_A1;
662. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
663. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
664. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
665. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
666. dis\_A1\_next1=sqrt((B1\_posnew(1)-A1\_posnew\_1(1))^2+(B1\_posnew(2)-A1\_posnew\_1(2))^2);
667. theta\_A1\_next2=theta\_A1-pi;
668. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
669. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
670. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
671. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
672. dis\_A1\_next2=sqrt((B1\_posnew(1)-A1\_posnew\_2(1))^2+(B1\_posnew(2)-A1\_posnew\_2(2))^2);
673. if dis\_A1\_next1<=dis\_A1\_next2
674. theta\_A1=theta\_A1\_next1;
675. else
676. theta\_A1=theta\_A1\_next2;
677. end
679. if B1\_posnew(2)>=7
680. B1\_pos(1)=B1\_posnew(1);
681. B1\_pos(2)=7;
682. else
683. if B1\_posnew(2)<=1
684. B1\_pos(1)=B1\_posnew(1);
685. B1\_pos(2)=1;
686. else
687. B1\_pos(1)=B1\_posnew(1);
688. B1\_pos(2)=B1\_posnew(2);
689. end
690. end
692. if dis\_A1\_B1<=v\_A1
693. A1\_pos(1)=Ball\_pos(1);
694. A1\_pos(2)=Ball\_pos(2);
695. else
696. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
697. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
698. end
699. Ball\_pos(1)=B1\_pos(1);
700. Ball\_pos(2)=B1\_pos(2);
701. run pitch.m;
702. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
703. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
704. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
705. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
706. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
707. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
708. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
709. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
710. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
711. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
712. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
713. pause(0.2);
714. end
715. end
716. dis\_A1\_Goal1=sqrt((Goal1\_pos(1)-A1\_pos(1))^2+(Goal1\_pos(2)-A1\_pos(2))^2);
717. dis\_A1\_Goal1\_t=sqrt((Goal1\_t\_pos(1)-A1\_pos(1))^2+(Goal1\_t\_pos(2)-A1\_pos(2))^2);
718. dis\_A1\_Goal1\_b=sqrt((Goal1\_b\_pos(1)-A1\_pos(1))^2+(Goal1\_b\_pos(2)-A1\_pos(2))^2);
719. dis\_B1\_Goal2=sqrt((Goal2\_pos(1)-B1\_pos(1))^2+(Goal2\_pos(2)-B1\_pos(2))^2);
720. dis\_B1\_Goal2\_t=sqrt((Goal2\_t\_pos(1)-B1\_pos(1))^2+(Goal2\_t\_pos(2)-B1\_pos(2))^2);
721. dis\_B1\_Goal2\_b=sqrt((Goal2\_b\_pos(1)-B1\_pos(1))^2+(Goal2\_b\_pos(2)-B1\_pos(2))^2);
722. delta\_x\_A1\_Goal1=Goal1\_pos(1)-A1\_pos(1);
723. delta\_y\_A1\_Goal1=Goal1\_pos(2)-A1\_pos(2);
724. delta\_x\_A1\_Goal1\_t=Goal1\_t\_pos(1)-A1\_pos(1);
725. delta\_y\_A1\_Goal1\_t=Goal1\_t\_pos(2)-A1\_pos(2);
726. delta\_x\_A1\_Goal1\_b=Goal1\_b\_pos(1)-A1\_pos(1);
727. delta\_y\_A1\_Goal1\_b=Goal1\_b\_pos(2)-A1\_pos(2);
728. delta\_x\_B1\_Goal2=Goal2\_pos(1)-B1\_pos(1);
729. delta\_y\_B1\_Goal2=Goal2\_pos(2)-B1\_pos(2);
730. delta\_x\_B1\_Goal2\_t=Goal2\_t\_pos(1)-B1\_pos(1);
731. delta\_y\_B1\_Goal2\_t=Goal2\_t\_pos(2)-B1\_pos(2);
732. delta\_x\_B1\_Goal2\_b=Goal2\_b\_pos(1)-B1\_pos(1);
733. delta\_y\_B1\_Goal2\_b=Goal2\_b\_pos(2)-B1\_pos(2);
734. theta\_A1\_Goal1=atan(delta\_y\_A1\_Goal1/delta\_x\_A1\_Goal1);
735. theta\_B1\_Goal2=atan(delta\_y\_B1\_Goal2/delta\_x\_B1\_Goal2);
736. theta\_A1\_Goal1\_t=atan(delta\_y\_A1\_Goal1\_t/delta\_x\_A1\_Goal1\_t);
737. theta\_B1\_Goal2\_t=atan(delta\_y\_B1\_Goal2\_t/delta\_x\_B1\_Goal2\_t);
738. theta\_A1\_Goal1\_b=atan(delta\_y\_A1\_Goal1\_b/delta\_x\_A1\_Goal1\_b);
739. theta\_B1\_Goal2\_b=atan(delta\_y\_B1\_Goal2\_b/delta\_x\_B1\_Goal2\_b);
741. if Ball\_right==2
743. t\_B1\_t=dis\_B1\_Goal2\_t/v\_B1;
744. num\_B1\_t=ceil(t\_B1\_t);
745. t\_B1\_b=dis\_B1\_Goal2\_b/v\_B1;
746. num\_B1\_b=ceil(t\_B1\_b);
747. if num\_B1\_t< num\_B1\_b
748. Goal2\_pos=Goal2\_t\_pos;
749. theta\_B1= getTheta(B1\_pos(1),B1\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
750. dis\_B1\_Goal2=dis\_B1\_Goal2\_t;
751. end
752. if num\_B1\_b< num\_B1\_t
753. Goal2\_pos=Goal2\_b\_pos;
754. theta\_B1= getTheta(B1\_pos(1),B1\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
755. dis\_B1\_Goal2=dis\_B1\_Goal2\_b;
756. end
757. if num\_B1\_t==num\_B1\_b
758. if rand(1)>=0.5
759. Goal2\_pos=Goal2\_t\_pos;
760. theta\_B1= getTheta(B1\_pos(1),B1\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
761. dis\_B1\_Goal2=dis\_B1\_Goal2\_t;
762. else
763. Goal2\_pos=Goal2\_b\_pos;
764. theta\_B1= getTheta(B1\_pos(1),B1\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
765. dis\_B1\_Goal2=dis\_B1\_Goal2\_b;
766. end
767. end
769. delta\_dis=dis\_B1\_Goal2-3;
770. t\_B1=delta\_dis/v\_B1;
771. delta\_dis\_t=dis\_B1\_Goal2\_t-3;
772. t\_B1\_t=delta\_dis\_t/v\_B1;
773. delta\_dis\_b=dis\_B1\_Goal2\_b-3;
774. t\_B1\_b=delta\_dis\_b/v\_B1;
775. num\_B1=ceil(t\_B1);
776. num\_B1\_t=ceil(t\_B1\_t);
777. num\_B1\_b=ceil(t\_B1\_b);
779. if num\_B1>0
780. for i=1:num\_B1
781. if Ball\_right==2
782. run pitch.m;
783. theta\_B1=theta\_B1\_Goal2-pi;%不清楚为什么这样
784. delta\_x\_B1=v\_B1\*cos(theta\_B1);
785. delta\_y\_B1=v\_B1\*sin(theta\_B1);
786. B1\_pos(1)=B1\_pos(1)+delta\_x\_B1;
787. if abs(B1\_pos(2)+delta\_y\_B1-Goal2\_pos(2))<=abs(B1\_pos(2)-delta\_y\_B1-Goal2\_pos(2))
788. B1\_pos(2)=B1\_pos(2)+delta\_y\_B1;
789. else
790. B1\_pos(2)=B1\_pos(2)-delta\_y\_B1;
791. end
792. B1\_pos(2)=B1\_pos(2)+delta\_y\_B1;
793. Ball\_pos(1)=B1\_pos(1);
794. Ball\_pos(2)=B1\_pos(2);
795. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
796. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
797. theta\_A1=atan(delta\_y\_A1/delta\_x\_A1);
798. if B1\_pos(1)<A1\_pos(1)
799. theta\_A1=theta\_A1-pi;
800. end
801. if (A1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<A1\_pos(1)+v\_A1\*cos(theta\_A1)) && (A1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>A1\_pos(2)+v\_A1\*sin(theta\_A1))
802. A1\_pos(1)=Ball\_pos(1);
803. A1\_pos(2)=Ball\_pos(2);
804. else
805. if (A1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>A1\_pos(1)+v\_A1\*cos(theta\_A1)) && (A1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<A1\_pos(2)+v\_B1\*sin(theta\_A1))
806. A1\_pos(1)=Ball\_pos(1);
807. A1\_pos(2)=Ball\_pos(2);
808. else
809. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
810. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
811. end
812. end
813. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
814. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
815. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
816. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
817. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
818. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
819. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
820. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
821. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
822. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
823. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
824. if A1\_pos(1)==Ball\_pos(1)&&A1\_pos(2)==Ball\_pos(2)
825. if rand(1)>=0.3
826. Ball\_right=1;
827. end
828. end
829. pause(0.2);
830. end
831. end
832. end
834. dis\_Ball=sqrt((Goal2\_pos(1)-Ball\_pos(1))^2+(Goal2\_pos(2)-Ball\_pos(2))^2);
835. dis\_A1\_B1=sqrt((B1\_pos(1)-A1\_pos(1))^2+(B1\_pos(2)-A1\_pos(2))^2);
836. delta\_dis=dis\_Ball-frame\*4\*0.2145;
837. t\_dis=delta\_dis/v\_B1;
838. num\_dis=ceil(t\_dis);
840. for i=1:num\_dis
841. if dis\_Ball >= frame\*4\*0.2145
842. if dis\_A1\_B1 >= v\_A1
843. delta\_x\_B1=v\_B1\*cos(theta\_B1);
844. delta\_y\_B1=v\_B1\*sin(theta\_B1);
845. B1\_pos(1)=B1\_pos(1)+delta\_x\_B1;
846. B1\_pos(2)=B1\_pos(2)+delta\_y\_B1;
847. Ball\_pos(1)=B1\_pos(1);
848. Ball\_pos(2)=B1\_pos(2);
849. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
850. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
851. theta\_A1=atan(delta\_y\_A1/delta\_x\_A1);
852. if B1\_pos(1)<A1\_pos(1)
853. theta\_A1=theta\_A1-pi;
854. end
855. if (A1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<A1\_pos(1)+v\_A1\*cos(theta\_A1)) && (A1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<A1\_pos(2)+v\_A1\*sin(theta\_A1))
856. A1\_pos(1)=Ball\_pos(1);
857. A1\_pos(2)=Ball\_pos(2);
858. else
859. if (A1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>A1\_pos(1)+v\_A1\*cos(theta\_A1)) && (A1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>A1\_pos(2)+v\_B1\*sin(theta\_A1))
860. A1\_pos(1)=Ball\_pos(1);
861. A1\_pos(2)=Ball\_pos(2);
862. else
863. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
864. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
865. end
866. end
867. run pitch.m;
868. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
869. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
870. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
871. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
872. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
873. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
874. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
875. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
876. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
877. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
878. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
879. pause(0.2);
880. if A1\_pos(1)==Ball\_pos(1)&&A1\_pos(2)==Ball\_pos(2)
881. if rand(1)>=0.3
882. Ball\_right=1;
883. end
884. end
885. end
886. end
887. end
889. delta\_x\_B1\_Goal2=Goal2\_pos(1)-B1\_pos(1);
890. delta\_y\_B1\_Goal2=Goal2\_pos(2)-B1\_pos(2);
891. theta\_B1\_Goal2=atan(delta\_y\_B1\_Goal2/delta\_x\_B1\_Goal2);
892. theta\_Ball\_Goal2=theta\_B1\_Goal2;
893. v\_Ball=2;
894. t\_ball=3;
895. dis\_Ball=sqrt((Goal2\_pos(1)-Ball\_pos(1))^2+(Goal2\_pos(2)-Ball\_pos(2))^2);
896. delta\_x\_Ball=-4\*0.2145\*cos(theta\_Ball\_Goal2);
897. delta\_y\_Ball=-4\*0.2145\*sin(theta\_Ball\_Goal2);
898. t\_equal=dis\_Ball/(4\*0.2145);
899. num\_Ball=ceil(t\_equal);
900. theta\_B1=theta\_B1\_Goal2-pi;%看删不删
902. if Ball\_right==2
903. for i=1:num\_Ball
904. run pitch.m
905. Ball\_pos(1)=Ball\_pos(1)+delta\_x\_Ball;
906. Ball\_pos(2)=Ball\_pos(2)+delta\_y\_Ball;
907. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
908. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
909. theta\_A1=atan(delta\_y\_A1/delta\_x\_A1);
910. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
911. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
912. if Goal2\_pos(2)==Goal2\_t\_pos(2)
913. a=1;
914. else
915. a=-1;
916. end
917. A2\_pos(2)=A2\_pos(2)+a\*(i-1)\*v\_A2;
918. if A2\_pos(2) >= Goal2\_t\_pos(2)
919. A2\_pos(2)=Goal2\_t\_pos(2);
920. end
921. if A2\_pos(2) <= Goal2\_b\_pos(2)
922. A2\_pos(2)=Goal2\_b\_pos(2);
923. end
924. if Ball\_pos(1)<Goal2\_pos(1)
925. Ball\_pos(1)=Goal2\_pos(1);
926. Ball\_pos(2)=Goal2\_pos(2);
927. end
928. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
929. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
930. line([A1\_pos(1) A1\_pos(1)-0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)-0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
931. line([A1\_pos(1) A1\_pos(1)-0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)-0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
932. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
933. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
934. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
935. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
936. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
937. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
938. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
939. pause(0.2);
940. end
941. end
943. if Ball\_right==1
944. theta\_A1\_next1=theta\_B1+pi/2;
945. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
946. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
947. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
948. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
949. theta\_A1\_next2=theta\_B1-pi/2;
950. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
951. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
952. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
953. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
954. dis\_A1\_next1=sqrt((Goal1\_pos(1)-A1\_posnew\_1(1))^2+(Goal1\_pos(2)-A1\_posnew\_1(2))^2);
955. dis\_next2=sqrt((Goal1\_pos(1)-A1\_posnew\_2(1))^2+(Goal1\_pos(2)-A1\_posnew\_2(2))^2);
956. if dis\_A1\_next1<=dis\_next2
957. theta\_A1=theta\_A1\_next1;
958. else
959. theta\_A1=theta\_A1\_next2;
960. end
961. delta\_x\_A1=v\_A1\*cos(theta\_A1);
962. delta\_y\_A1=v\_A1\*sin(theta\_A1);
963. A1\_posnew(1)=A1\_pos(1)+delta\_x\_A1;
964. A1\_posnew(2)=A1\_pos(2)+delta\_y\_A1;
965. theta\_B1\_next1=theta\_B1;
966. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
967. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
968. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
969. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
970. dis\_B1\_next1=sqrt((A1\_posnew(1)-B1\_posnew\_1(1))^2+(A1\_posnew(2)-B1\_posnew\_1(2))^2);
971. theta\_B1\_next2=theta\_B1-pi;
972. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
973. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
974. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
975. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
976. dis\_B1\_next2=sqrt((A1\_posnew(1)-B1\_posnew\_2(1))^2+(A1\_posnew(2)-B1\_posnew\_2(2))^2);
977. if dis\_B1\_next1<=dis\_B1\_next2
978. theta\_B1=theta\_B1\_next1;
979. else
980. theta\_B1=theta\_B1\_next2;
981. end
983. if A1\_posnew(2)>=7
984. A1\_pos(1)=A1\_posnew(1);
985. A1\_pos(2)=7;
986. else
987. if A1\_posnew(2)<=1
988. A1\_pos(1)=A1\_posnew(1);
989. A1\_pos(2)=1;
990. else
991. A1\_pos(1)=A1\_posnew(1);
992. A1\_pos(2)=A1\_posnew(2);
993. end
994. end
995. if dis\_A1\_B1<=v\_B1
996. B1\_pos(1)=Ball\_pos(1);
997. B1\_pos(2)=Ball\_pos(2);
998. else
999. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
1000. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
1001. end
1002. Ball\_pos(1)=A1\_pos(1);
1003. Ball\_pos(2)=A1\_pos(2);
1004. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
1005. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
1006. run pitch.m;
1007. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
1008. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
1009. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
1010. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
1011. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
1012. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
1013. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
1014. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1015. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1016. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
1017. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
1018. pause(0.2);
1019. end
1021. dis\_A1\_Goal1=sqrt((Goal1\_pos(1)-A1\_pos(1))^2+(Goal1\_pos(2)-A1\_pos(2))^2);
1022. dis\_A1\_Goal1\_t=sqrt((Goal1\_t\_pos(1)-A1\_pos(1))^2+(Goal1\_t\_pos(2)-A1\_pos(2))^2);
1023. dis\_A1\_Goal1\_b=sqrt((Goal1\_b\_pos(1)-A1\_pos(1))^2+(Goal1\_b\_pos(2)-A1\_pos(2))^2);
1024. dis\_B1\_Goal2=sqrt((Goal2\_pos(1)-B1\_pos(1))^2+(Goal2\_pos(2)-B1\_pos(2))^2);
1025. dis\_B1\_Goal2\_t=sqrt((Goal2\_t\_pos(1)-B1\_pos(1))^2+(Goal2\_t\_pos(2)-B1\_pos(2))^2);
1026. dis\_B1\_Goal2\_b=sqrt((Goal2\_b\_pos(1)-B1\_pos(1))^2+(Goal2\_b\_pos(2)-B1\_pos(2))^2);
1027. delta\_x\_A1\_Goal1=Goal1\_pos(1)-A1\_pos(1);
1028. delta\_y\_A1\_Goal1=Goal1\_pos(2)-A1\_pos(2);
1029. delta\_x\_A1\_Goal1\_t=Goal1\_t\_pos(1)-A1\_pos(1);
1030. delta\_y\_A1\_Goal1\_t=Goal1\_t\_pos(2)-A1\_pos(2);
1031. delta\_x\_A1\_Goal1\_b=Goal1\_b\_pos(1)-A1\_pos(1);
1032. delta\_y\_A1\_Goal1\_b=Goal1\_b\_pos(2)-A1\_pos(2);
1033. delta\_x\_B1\_Goal2=Goal2\_pos(1)-B1\_pos(1);
1034. delta\_y\_B1\_Goal2=Goal2\_pos(2)-B1\_pos(2);
1035. delta\_x\_B1\_Goal2\_t=Goal2\_t\_pos(1)-B1\_pos(1);
1036. delta\_y\_B1\_Goal2\_t=Goal2\_t\_pos(2)-B1\_pos(2);
1037. delta\_x\_B1\_Goal2\_b=Goal2\_b\_pos(1)-B1\_pos(1);
1038. delta\_y\_B1\_Goal2\_b=Goal2\_b\_pos(2)-B1\_pos(2);
1039. theta\_A1\_Goal1=atan(delta\_y\_A1\_Goal1/delta\_x\_A1\_Goal1);
1040. theta\_B1\_Goal2=atan(delta\_y\_B1\_Goal2/delta\_x\_B1\_Goal2);
1041. theta\_A1\_Goal1\_t=atan(delta\_y\_A1\_Goal1\_t/delta\_x\_A1\_Goal1\_t);
1042. theta\_B1\_Goal2\_t=atan(delta\_y\_B1\_Goal2\_t/delta\_x\_B1\_Goal2\_t);
1043. theta\_A1\_Goal1\_b=atan(delta\_y\_A1\_Goal1\_b/delta\_x\_A1\_Goal1\_b);
1044. theta\_B1\_Goal2\_b=atan(delta\_y\_B1\_Goal2\_b/delta\_x\_B1\_Goal2\_b);
1046. if Ball\_right==1
1047. dis\_A1\_Goal1\_t=sqrt((Goal1\_t\_pos(1)-A1\_pos(1))^2+(Goal1\_t\_pos(2)-A1\_pos(2))^2);
1048. dis\_A1\_Goal1\_b=sqrt((Goal1\_b\_pos(1)-A1\_pos(1))^2+(Goal1\_b\_pos(2)-A1\_pos(2))^2);
1049. t\_A1\_t=dis\_A1\_Goal1\_t/v\_A1;
1050. num\_A1\_t=ceil(t\_A1\_t);
1051. t\_A1\_b=dis\_A1\_Goal1\_b/v\_A1;
1052. num\_A1\_b=ceil(t\_A1\_b);
1053. if num\_A1\_t< num\_A1\_b
1054. Goal1\_pos=Goal1\_t\_pos;
1055. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
1056. dis\_A1\_Goal1=dis\_A1\_Goal1\_t;
1057. end
1058. if num\_A1\_b< num\_A1\_t
1059. Goal1\_pos=Goal1\_b\_pos;
1060. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
1061. dis\_A1\_Goal1=dis\_A1\_Goal1\_b;
1062. end
1063. if num\_A1\_t==num\_A1\_b
1064. if rand(1)>=0.5
1065. Goal1\_pos=Goal1\_t\_pos;
1066. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
1067. dis\_A1\_Goal1=dis\_A1\_Goal1\_t;
1068. else
1069. Goal1\_pos=Goal1\_b\_pos;
1070. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
1071. dis\_A1\_Goal1=dis\_A1\_Goal1\_b;
1072. end
1073. end

1076. delta\_dis=dis\_A1\_Goal1-3;
1077. t\_A1=delta\_dis/v\_A1;
1078. delta\_dis\_t=dis\_A1\_Goal1\_t-3;
1079. t\_A1\_t=delta\_dis\_t/v\_A1;
1080. delta\_dis\_b=dis\_A1\_Goal1\_b-3;
1081. t\_A1\_b=delta\_dis\_b/v\_A1;
1082. num\_A1=ceil(t\_A1);
1083. num\_A1\_t=ceil(t\_A1\_t);
1084. num\_A1\_b=ceil(t\_A1\_b);
1086. if num\_A1>0
1087. for i=1:num\_A1
1088. if Ball\_right==1
1089. run pitch.m;
1090. theta\_A1=theta\_A1\_Goal1;
1091. delta\_x\_A1=v\_A1\*cos(theta\_A1);
1092. delta\_y\_A1=v\_A1\*sin(theta\_A1);
1093. A1\_pos(1)=A1\_pos(1)+delta\_x\_A1;
1094. if abs(A1\_pos(2)+delta\_y\_A1-Goal1\_pos(2))<=abs(A1\_pos(2)-delta\_y\_A1-Goal1\_pos(2))
1095. A1\_pos(2)=A1\_pos(2)+delta\_y\_A1;
1096. else
1097. A1\_pos(2)=A1\_pos(2)-delta\_y\_A1;
1098. end
1099. Ball\_pos(1)=A1\_pos(1);
1100. Ball\_pos(2)=A1\_pos(2);
1101. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
1102. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
1103. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
1104. if B1\_pos(1)>A1\_pos(1)
1105. theta\_B1=theta\_B1-pi;
1106. end
1107. if (B1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<B1\_pos(2)+v\_B1\*sin(theta\_B1))
1108. B1\_pos(1)=Ball\_pos(1);
1109. B1\_pos(2)=Ball\_pos(2);
1110. else
1111. if (B1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>B1\_pos(2)+v\_B1\*sin(theta\_B1))
1112. B1\_pos(1)=Ball\_pos(1);
1113. B1\_pos(2)=Ball\_pos(2);
1114. else
1115. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
1116. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
1117. end
1118. end
1119. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
1120. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
1121. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
1122. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
1123. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
1124. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
1125. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
1126. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1127. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1128. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
1129. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
1130. pause(0.2);
1131. if B1\_pos(1)==Ball\_pos(1)&&B1\_pos(2)==Ball\_pos(2)
1132. if rand(1)>=0.3
1133. Ball\_right=2;
1134. end
1135. end
1136. end
1137. end
1138. end
1140. dis\_Ball=sqrt((Goal1\_pos(1)-Ball\_pos(1))^2+(Goal1\_pos(2)-Ball\_pos(2))^2);
1141. dis\_A1\_B1=sqrt((B1\_pos(1)-A1\_pos(1))^2+(B1\_pos(2)-A1\_pos(2))^2);
1142. delta\_dis=dis\_Ball-frame\*4\*0.2145;
1143. t\_dis=delta\_dis/v\_A1;
1144. num\_dis=ceil(t\_dis);
1146. for i=1:num\_dis
1147. if dis\_Ball >= frame\*4\*0.2145
1148. if dis\_A1\_B1 >= v\_B1
1149. delta\_x\_A1=v\_A1\*cos(theta\_A1);
1150. delta\_y\_A1=v\_A1\*sin(theta\_A1);
1151. A1\_pos(1)=A1\_pos(1)+delta\_x\_A1;
1152. A1\_pos(2)=A1\_pos(2)+delta\_y\_A1;
1153. theta\_A1=theta\_A1\_Goal1;
1154. Ball\_pos(1)=A1\_pos(1);
1155. Ball\_pos(2)=A1\_pos(2);
1156. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
1157. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
1158. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
1159. if B1\_pos(1)>A1\_pos(1)
1160. theta\_B1=theta\_B1-pi;
1161. end
1162. if (B1\_pos(1)<Ball\_pos(1))&&(Ball\_pos(1)<B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)<Ball\_pos(2))&&(Ball\_pos(2)<B1\_pos(2)+v\_B1\*sin(theta\_B1))
1163. B1\_pos(1)=Ball\_pos(1);
1164. B1\_pos(2)=Ball\_pos(2);
1165. else
1166. if (B1\_pos(1)>Ball\_pos(1))&&(Ball\_pos(1)>B1\_pos(1)+v\_B1\*cos(theta\_B1)) && (B1\_pos(2)>Ball\_pos(2))&&(Ball\_pos(2)>B1\_pos(2)+v\_B1\*sin(theta\_B1))
1167. B1\_pos(1)=Ball\_pos(1);
1168. B1\_pos(2)=Ball\_pos(2);
1169. else
1170. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
1171. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
1172. end
1173. end
1174. run pitch.m;
1175. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
1176. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
1177. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
1178. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
1179. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
1180. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
1181. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
1182. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1183. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1184. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
1185. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
1186. pause(0.2);
1187. if B1\_pos(1)==Ball\_pos(1)&&B1\_pos(2)==Ball\_pos(2)
1188. if rand(1)>=0.3
1189. Ball\_right=2;
1190. end
1191. end
1192. end
1193. end
1194. end

1197. delta\_x\_A1\_Goal1=Goal1\_pos(1)-A1\_pos(1);
1198. delta\_y\_A1\_Goal1=Goal1\_pos(2)-A1\_pos(2);
1199. theta\_A1\_Goal1=atan(delta\_y\_A1\_Goal1/delta\_x\_A1\_Goal1);
1200. theta\_Ball\_Goal1=theta\_A1\_Goal1;
1201. theta\_A1=theta\_Ball\_Goal1;
1202. v\_Ball=2;
1203. t\_ball=3;
1204. dis\_Ball=sqrt((Goal1\_pos(1)-Ball\_pos(1))^2+(Goal1\_pos(2)-Ball\_pos(2))^2);
1205. delta\_x\_Ball=4\*0.2145\*cos(theta\_Ball\_Goal1);
1206. delta\_y\_Ball=4\*0.2145\*sin(theta\_Ball\_Goal1);
1207. t\_equal=dis\_Ball/(4\*0.2145);
1208. num\_Ball=ceil(t\_equal);
1209. if Ball\_right==1
1210. for i=1:num\_Ball
1211. run pitch.m
1212. Ball\_pos(1)=Ball\_pos(1)+delta\_x\_Ball;
1213. Ball\_pos(2)=Ball\_pos(2)+delta\_y\_Ball;
1214. delta\_x\_B1=Ball\_pos(1)-B1\_pos(1);
1215. delta\_y\_B1=Ball\_pos(2)-B1\_pos(2);
1216. theta\_B1=atan(delta\_y\_B1/delta\_x\_B1);
1217. B1\_pos(1)=B1\_pos(1)+v\_B1\*cos(theta\_B1);
1218. B1\_pos(2)=B1\_pos(2)+v\_B1\*sin(theta\_B1);
1219. if Goal1\_pos(2)==Goal1\_t\_pos(2)
1220. a=1;
1221. else
1222. a=-1;
1223. end
1224. B2\_pos(2)=B2\_pos(2)+a\*(i-1)\*v\_B2;
1225. if B2\_pos(2) >= Goal1\_t\_pos(2)
1226. B2\_pos(2)=Goal1\_t\_pos(2);
1227. end
1228. if B2\_pos(2) <= Goal1\_b\_pos(2)
1229. B2\_pos(2)=Goal1\_b\_pos(2);
1230. end
1231. if Ball\_pos(1)>Goal1\_pos(1)
1232. Ball\_pos(1)=Goal1\_pos(1);
1233. Ball\_pos(2)=Goal1\_pos(2);
1234. end
1235. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
1236. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
1237. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
1238. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
1239. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
1240. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
1241. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
1242. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1243. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1244. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
1245. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
1246. pause(0.2);
1247. Ball\_right=0;
1248. end
1249. end
1250. Goal1\_pos=[10 4];
1251. if Ball\_right==2
1252. theta\_B1\_next1=theta\_A1+pi/2;
1253. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
1254. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
1255. B1\_posnew\_1(1)=B1\_pos(1)+delta\_x\_B1\_next1;
1256. B1\_posnew\_1(2)=B1\_pos(2)+delta\_y\_B1\_next1;
1257. theta\_B1\_next2=theta\_B1-pi/2;
1258. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
1259. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
1260. B1\_posnew\_2(1)=B1\_pos(1)+delta\_x\_B1\_next2;
1261. B1\_posnew\_2(2)=B1\_pos(2)+delta\_y\_B1\_next2;
1262. dis\_B1\_next1=sqrt((Goal2\_pos(1)-B1\_posnew\_1(1))^2+(Goal2\_pos(2)-B1\_posnew\_1(2))^2);
1263. dis\_B1\_next2=sqrt((Goal2\_pos(1)-B1\_posnew\_2(1))^2+(Goal2\_pos(2)-B1\_posnew\_2(2))^2);
1264. if dis\_B1\_next1<=dis\_B1\_next2
1265. theta\_B1=theta\_B1\_next1;
1266. else
1267. theta\_B1=theta\_B1\_next2;
1268. end
1269. delta\_x\_B1=v\_B1\*cos(theta\_B1);
1270. delta\_y\_B1=v\_B1\*sin(theta\_B1);
1271. B1\_posnew(1)=B1\_pos(1)+delta\_x\_B1;
1272. B1\_posnew(2)=B1\_pos(2)+delta\_y\_B1;
1273. delta\_x\_A1=Ball\_pos(1)-A1\_pos(1);
1274. delta\_y\_A1=Ball\_pos(2)-A1\_pos(2);
1275. theta\_A1\_next1=theta\_A1;
1276. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
1277. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
1278. A1\_posnew\_1(1)=A1\_pos(1)+delta\_x\_A1\_next1;
1279. A1\_posnew\_1(2)=A1\_pos(2)+delta\_y\_A1\_next1;
1280. dis\_A1\_next1=sqrt((B1\_posnew(1)-A1\_posnew\_1(1))^2+(B1\_posnew(2)-A1\_posnew\_1(2))^2);
1281. theta\_A1\_next2=theta\_A1-pi;
1282. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
1283. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
1284. A1\_posnew\_2(1)=A1\_pos(1)+delta\_x\_A1\_next2;
1285. A1\_posnew\_2(2)=A1\_pos(2)+delta\_y\_A1\_next2;
1286. dis\_A1\_next2=sqrt((B1\_posnew(1)-A1\_posnew\_2(1))^2+(B1\_posnew(2)-A1\_posnew\_2(2))^2);
1287. if dis\_A1\_next1<=dis\_A1\_next2
1288. theta\_A1=theta\_A1\_next1;
1289. else
1290. theta\_A1=theta\_A1\_next2;
1291. end
1293. if B1\_posnew(2)>=7
1294. B1\_pos(1)=B1\_posnew(1);
1295. B1\_pos(2)=7;
1296. else
1297. if B1\_posnew(2)<=1
1298. B1\_pos(1)=B1\_posnew(1);
1299. B1\_pos(2)=1;
1300. else
1301. B1\_pos(1)=B1\_posnew(1);
1302. B1\_pos(2)=B1\_posnew(2);
1303. end
1304. end
1305. if dis\_A1\_B1<=v\_A1
1306. A1\_pos(1)=Ball\_pos(1);
1307. A1\_pos(2)=Ball\_pos(2);
1308. else
1309. A1\_pos(1)=A1\_pos(1)+v\_A1\*cos(theta\_A1);
1310. A1\_pos(2)=A1\_pos(2)+v\_A1\*sin(theta\_A1);
1311. end
1312. Ball\_pos(1)=B1\_pos(1);
1313. Ball\_pos(2)=B1\_pos(2);
1314. run pitch.m;
1315. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
1316. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
1317. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
1318. line([A1\_pos(1) A1\_pos(1)+0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)+0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
1319. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
1320. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
1321. line([B1\_pos(1) B1\_pos(1)+0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)+0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
1322. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1323. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1324. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
1325. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
1326. pause(0.2);
1327. end
1328. end
1329. end

### ***pitch\_f.m***

1. function pitch\_f
2. PitchLength=9;
3. PitchWidth=6;
4. GoalDepth=0.6;
5. GoalWidth=2.6;
6. GoalHeight=1.2;
7. GoalAreaLength=1;
8. GoalAreaWidth=3;
9. PenaltyMarkDistance=1.5;
10. CentreCircleDiameter=1.5;
11. BorderStripWidth=1;
12. PenaltyAreaLength=2;
13. PenaltyAreaWidth=5;
14. A=PitchLength;
15. B=PitchWidth;
16. C=GoalDepth;
17. D=GoalWidth;
18. E=GoalAreaLength;
19. F=GoalAreaWidth;
20. G=PenaltyMarkDistance;
21. H=CentreCircleDiameter;
22. I=BorderStripWidth;
23. J=PenaltyAreaLength;
24. K=PenaltyAreaWidth;
25. axis([0 A+2\*I 0 B+2\*I]);
26. rectangle('Position',[0 0 2\*I+A 2\*I+B],'FaceColor','g','EdgeColor','w');
27. rectangle('Position',[I I A B],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
28. rectangle('Position',[I-C B/2+I-D/2 C D],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
29. rectangle('Position',[I+A B/2+I-D/2 C D],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
30. rectangle('Position',[I B/2+I-F/2 E F],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
31. rectangle('Position',[I+A-E B/2+I-F/2 E F],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
32. rectangle('Position',[I B/2+I-K/2 J K],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
33. rectangle('Position',[I+A-J B/2+I-K/2 J K],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
34. viscircles([I+A/2 I+B/2],H/2,'Color','w','LineWidth',0.75);
35. hold on;
36. plot(I+G,I+B/2,'w+');
37. plot(I+A/2,I+B/2,'w+');
38. plot(I+A-G,I+B/2,'w+');
39. line([I+A/2 I+A/2],[I+B I],'Color','w','LineWidth',1);

### getTheta***.m***

1. function [m]=getTheta(x1,y1,x2,y2)
2. delta\_x=x2-x1;
3. delta\_y=y2-y1;
4. theta=atan(delta\_y/delta\_x);
5. if delta\_x<0
6. m=theta;
7. else
8. m=theta+pi;
9. end
10. if delta\_x==0&&delta\_y==0
11. if x1>5.5
12. m=pi;
13. else
14. m=0;
15. end
16. if x2==10
17. m=0;
18. else
19. if x2==1
20. m=pi;
21. end
22. end
23. end

### ***getDis.m***

1. function [m]=getDis(x1,y1,x2,y2)
2. delta\_x=x2-x1;
3. delta\_y=y2-y1;
4. m=sqrt(delta\_x^2+delta\_y^2);

### ***draw\_A.m***

1. function draw\_A(x,y,angle)
2. A1\_pos(1)=x;
3. A1\_pos(2)=y;
4. theta\_A1=angle;
5. viscircles([A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
6. line([A1\_pos(1) A1\_pos(1)-0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)-0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
7. line([A1\_pos(1) A1\_pos(1)-0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)-0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);

### ***draw\_B.m***

1. function draw\_B(x,y,angle)
2. B1\_pos(1)=x;
3. B1\_pos(2)=y;
4. theta\_B1=angle;
5. viscircles([B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
6. line([B1\_pos(1) B1\_pos(1)-0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)-0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
7. line([B1\_pos(1) B1\_pos(1)-0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)-0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);

### ***draw\_whole.m***

1. function draw\_whole(Ball,A1,A1\_a,A2,A2\_a,A3,A3\_a,A4,A4\_a,B1,B1\_a,B2,B2\_a,B3,B3\_a,B4,B4\_a)
2. run pitch.m;
3. Ball\_pos(1)=Ball(1);
4. Ball\_pos(2)=Ball(2);
5. A1\_pos(1)=A1(1);
6. A1\_pos(2)=A1(2);
7. A2\_pos(1)=A2(1);
8. A2\_pos(2)=A2(2);
9. A3\_pos(1)=A3(1);
10. A3\_pos(2)=A3(2);
11. A4\_pos(1)=A4(1);
12. A4\_pos(2)=A4(2);
13. B1\_pos(1)=B1(1);
14. B1\_pos(2)=B1(2);
15. B2\_pos(1)=B2(1);
16. B2\_pos(2)=B2(2);
17. B3\_pos(1)=B3(1);
18. B3\_pos(2)=B3(2);
19. B4\_pos(1)=B4(1);
20. B4\_pos(2)=B4(2);
21. theta\_A1=A1\_a;
22. theta\_A2=A2\_a;
23. theta\_A3=A3\_a;
24. theta\_A4=A4\_a;
25. theta\_B1=B1\_a;
26. theta\_B2=B2\_a;
27. theta\_B3=B3\_a;
28. theta\_B4=B4\_a;
29. viscircles([Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
30. draw\_A(A1\_pos(1),A1\_pos(2),theta\_A1);
31. draw\_A(A2\_pos(1),A2\_pos(2),theta\_A2);
32. draw\_A(A3\_pos(1),A3\_pos(2),theta\_A3);
33. draw\_A(A4\_pos(1),A4\_pos(2),theta\_A4);
34. draw\_B(B1\_pos(1),B1\_pos(2),theta\_B1);
35. draw\_B(B2\_pos(1),B2\_pos(2),theta\_B2);
36. draw\_B(B3\_pos(1),B3\_pos(2),theta\_B3);
37. draw\_B(B4\_pos(1),B4\_pos(2),theta\_B4);

### ***A4\_B4.m***

1. %Define and initialize variables for players, ball, goals and initial conditions for the game
2. A1\_pos=[4 4];
3. theta\_A1=pi;
4. v\_A1=2\*0.2145;
5. A2\_pos=[4 6];
6. theta\_A2=pi;
7. v\_A2=2\*0.2145;
8. A3\_pos=[4 2];
9. theta\_A3=pi;
10. v\_A3=2\*0.2145;
11. A4\_pos=[1 4];
12. theta\_A4=pi;
13. v\_A4=2\*0.18;
14. B1\_pos=[7 4];
15. theta\_B1=0;
16. v\_B1=2\*0.2145;
17. B2\_pos=[7 6];
18. theta\_B2=0;
19. v\_B2=2\*0.2145;
20. B3\_pos=[7 2];
21. theta\_B3=0;
22. v\_B3=2\*0.2145;
23. B4\_pos=[10 4];
24. theta\_B4=0;
25. v\_B4=2\*0.18;
26. Ball\_pos=[5.5 4];
27. v\_Ball=4\*0.2145;
28. Goal1\_pos\_abs=[10 4];
29. Goal1\_t\_pos=[10 5.3];
30. Goal1\_b\_pos=[10 2.7];
31. Goal2\_pos\_abs=[1 4];
32. Goal2\_t\_pos=[1 5.3];
33. Goal2\_b\_pos=[1 2.7];
34. Ball\_right=0;
35. defind\_pos\_1=[8 4.65];
36. defind\_pos\_2=[8 3.35];
37. defind\_pos\_3=[3 4.65];
38. defind\_pos\_4=[3 3.35];
39. frame=3;
40. score\_A=0;
41. score\_B=0;
42. %When the score haven't reach 3
43. while (score\_A<3 && score\_B<3)
44. %calculate the distance between each player and the ball, and then calculates the time it would take each player to reach the ball based on their speed.
45. while (Ball\_right==0)
46. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
47. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
48. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
49. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
50. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
51. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
52. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
53. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
54. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
55. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
56. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
57. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
59. %The player closest to the ball is then selected and moved closer to the ball in a loop that runs for the amount of time it would take them to reach the ball.
60. %------------A to Ball----------------%
61. if dis\_A1\_Ball<=dis\_A2\_Ball && dis\_A1\_Ball<=dis\_A3\_Ball
62. dis\_A\_Ball=dis\_A1\_Ball;
63. t\_A\_Ball=dis\_A\_Ball/v\_A1;
64. end
65. if dis\_A2\_Ball<dis\_A1\_Ball && dis\_A2\_Ball<=dis\_A3\_Ball
66. dis\_A\_Ball=dis\_A2\_Ball;
67. t\_A\_Ball=dis\_A\_Ball/v\_A2;
68. end
69. if dis\_A3\_Ball<dis\_A1\_Ball && dis\_A3\_Ball<dis\_A2\_Ball
70. dis\_A\_Ball=dis\_A3\_Ball;
71. t\_A\_Ball=dis\_A\_Ball/v\_A3;
72. end
73. num\_A\_Ball=ceil(t\_A\_Ball);
74. %--------------------------------------%
76. %------------B to Ball----------------%
77. if dis\_B1\_Ball<=dis\_B2\_Ball && dis\_B1\_Ball<=dis\_B3\_Ball
78. dis\_B\_Ball=dis\_B1\_Ball;
79. t\_B\_Ball=dis\_B\_Ball/v\_B1;
80. end
81. if dis\_B2\_Ball<dis\_B1\_Ball && dis\_B2\_Ball<=dis\_B3\_Ball
82. dis\_B\_Ball=dis\_B2\_Ball;
83. t\_B\_Ball=dis\_B\_Ball/v\_B2;
84. end
85. if dis\_B3\_Ball<dis\_B1\_Ball && dis\_B3\_Ball<dis\_B2\_Ball
86. dis\_B\_Ball=dis\_B3\_Ball;
87. t\_B\_Ball=dis\_B\_Ball/v\_B3;
88. end
89. num\_B\_Ball=ceil(t\_B\_Ball);
90. %--------------------------------------%
91. %The code then repeats the calculation of the player's distance to the ball and the time it would take them to reach the ball until the ball is hit by a player from team A or team B to score a point.
92. if num\_A\_Ball<num\_B\_Ball
93. for i=1:num\_A\_Ball
94. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
95. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
96. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
97. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
98. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
99. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
100. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
101. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
103. %-----A1 to Ball-----%
104. if dis\_A\_Ball==dis\_A1\_Ball
105. if i<num\_A\_Ball
106. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
107. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
108. else
109. A1\_pos(1)=Ball\_pos(1);
110. A1\_pos(2)=Ball\_pos(2);
111. end
112. end
113. %----------------------%
115. %-----A2 to Ball-----%
116. if dis\_A\_Ball==dis\_A2\_Ball
117. if i<num\_A\_Ball
118. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
119. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
120. else
121. A2\_pos(1)=Ball\_pos(1);
122. A2\_pos(2)=Ball\_pos(2);
123. end
124. end
125. %----------------------%
126. %-----A3 to Ball-----%
127. if dis\_A\_Ball==dis\_A3\_Ball
128. if i<num\_A\_Ball
129. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
130. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
131. else
132. A3\_pos(1)=Ball\_pos(1);
133. A3\_pos(2)=Ball\_pos(2);
134. end
135. end
136. %----------------------%
138. %-----B1 to Ball-----%
139. if dis\_B\_Ball==dis\_B1\_Ball
140. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
141. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
142. end
143. %----------------------%
144. %-----B2 to Ball-----%
145. if dis\_B\_Ball==dis\_B2\_Ball
146. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
147. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
148. end
149. %----------------------%
151. %-----B3 to Ball-----%
152. if dis\_B\_Ball==dis\_B3\_Ball
153. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
154. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
155. end
156. %----------------------%                draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
157. pause(0.2);
158. end
159. end
161. %Repeat the code for different situations
162. if num\_B\_Ball<num\_A\_Ball
163. for i=1:num\_B\_Ball
164. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
165. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
166. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
167. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
168. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
169. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
170. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
171. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
172. %-----B1 to Ball-----%
173. if dis\_B\_Ball==dis\_B1\_Ball
174. if i<num\_B\_Ball
175. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
176. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
177. else
178. B1\_pos(1)=Ball\_pos(1);
179. B1\_pos(2)=Ball\_pos(2);
180. end
181. end
182. %----------------------%
184. %-----B2 to Ball-----%
185. if dis\_B\_Ball==dis\_B2\_Ball
186. if i<num\_B\_Ball
187. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
188. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
189. else
190. B2\_pos(1)=Ball\_pos(1);
191. B2\_pos(2)=Ball\_pos(2);
192. end
193. end
194. %----------------------%
195. %-----B3 to Ball-----%
196. if dis\_B\_Ball==dis\_B3\_Ball
197. if i<num\_B\_Ball
198. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
199. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
200. else
201. B3\_pos(1)=Ball\_pos(1);
202. B3\_pos(2)=Ball\_pos(2);
203. end
204. end
205. %----------------------%
207. %-----A1 to Ball-----%
208. if dis\_A\_Ball==dis\_A1\_Ball
209. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
210. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
211. end
212. %----------------------%
213. %-----A2 to Ball-----%
214. if dis\_A\_Ball==dis\_A2\_Ball
215. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
216. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
217. end
218. %----------------------%
220. %-----A3 to Ball-----%
221. if dis\_A\_Ball==dis\_A3\_Ball
222. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
223. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
224. end
225. %----------------------%
226. % Call a function to draw the current positions of all objects and pauses for a short time.
227. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,
228. A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,
229. B4\_pos,theta\_B4);
230. pause(0.2);
231. end
232. end
234. if num\_A\_Ball==num\_B\_Ball
235. for i=1:num\_A\_Ball-1
236. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
237. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
238. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
239. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
240. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
241. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
242. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
243. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
245. %-----A1 to Ball-----%
246. if dis\_A\_Ball==dis\_A1\_Ball
247. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
248. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
249. end
250. %----------------------%
251. %-----A2 to Ball-----%
252. if dis\_A\_Ball==dis\_A2\_Ball
253. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
254. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
255. end
256. %----------------------%
258. %-----A3 to Ball-----%
259. if dis\_A\_Ball==dis\_A3\_Ball
260. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
261. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
262. end
263. %----------------------%
264. %-----B1 to Ball-----%
265. if dis\_B\_Ball==dis\_B1\_Ball
266. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
267. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
268. end
269. %----------------------%
270. %-----B2 to Ball-----%
271. if dis\_B\_Ball==dis\_B2\_Ball
272. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
273. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
274. end
275. %----------------------%
277. %-----B3 to Ball-----%
278. if dis\_B\_Ball==dis\_B3\_Ball
279. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
280. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
281. end
282. %----------------------%
283. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,
284. A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,
285. B4\_pos,theta\_B4);
286. pause(0.2);
287. end
289. %determining which player should receive the ball next.
290. %------Ball to who---------%
291. if rand(1)>=0.5
292. if dis\_A\_Ball==dis\_A1\_Ball
293. A1\_pos(1)=Ball\_pos(1);
294. A1\_pos(2)=Ball\_pos(2);
295. end
296. if dis\_A\_Ball==dis\_A2\_Ball
297. A2\_pos(1)=Ball\_pos(1);
298. A2\_pos(2)=Ball\_pos(2);
299. end
300. if dis\_A\_Ball==dis\_A3\_Ball
301. A3\_pos(1)=Ball\_pos(1);
302. A3\_pos(2)=Ball\_pos(2);
303. end
304. else
305. if dis\_B\_Ball==dis\_B1\_Ball
306. B1\_pos(1)=Ball\_pos(1);
307. B1\_pos(2)=Ball\_pos(2);
308. end
309. if dis\_B\_Ball==dis\_B2\_Ball
310. B2\_pos(1)=Ball\_pos(1);
311. B2\_pos(2)=Ball\_pos(2);
312. end
313. if dis\_B\_Ball==dis\_B3\_Ball
314. B3\_pos(1)=Ball\_pos(1);
315. B3\_pos(2)=Ball\_pos(2);
316. end
317. end
318. %-------------------------%
319. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,A4\_pos,
320. theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,
321. theta\_B4);
322. pause(0.2);
323. end
325. %Check which player is closest to the ball and updating the variable Ball\_right accordingly.
326. %--------Go to Ball\_righit=?----------%
327. if A1\_pos(1)==Ball\_pos(1) && A1\_pos(2)==Ball\_pos(2)
328. Ball\_right=1;
329. end
330. if A2\_pos(1)==Ball\_pos(1) && A2\_pos(2)==Ball\_pos(2)
331. Ball\_right=2;
332. end
333. if A3\_pos(1)==Ball\_pos(1) && A3\_pos(2)==Ball\_pos(2)
334. Ball\_right=3;
335. end
336. if A4\_pos(1)==Ball\_pos(1) && A4\_pos(2)==Ball\_pos(2)
337. Ball\_right=4;
338. end
339. if B1\_pos(1)==Ball\_pos(1) && B1\_pos(2)==Ball\_pos(2)
340. Ball\_right=5;
341. end
342. if B2\_pos(1)==Ball\_pos(1) && B2\_pos(2)==Ball\_pos(2)
343. Ball\_right=6;
344. end
345. if B3\_pos(1)==Ball\_pos(1) && B3\_pos(2)==Ball\_pos(2)
346. Ball\_right=7;
347. end
348. if B4\_pos(1)==Ball\_pos(1) && B4\_pos(2)==Ball\_pos(2)
349. Ball\_right=8;
350. end
351. %---------------------------------%
353. end
354. %Calculate the distance between the ball and each of the players (A1, A2, A3, B1, B2, B3) and calculate the angle (theta) between each player and the ball.
355. while (Ball\_right==1)
356. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
357. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
358. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
359. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
360. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
361. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
362. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
363. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
364. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
365. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
366. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
367. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
369. %compare the distances between the ball and each player
370. if dis\_B1\_Ball <= dis\_B2\_Ball && dis\_B1\_Ball <= dis\_B3\_Ball
371. theta\_B=theta\_B1;
372. end
373. if dis\_B2\_Ball < dis\_B1\_Ball && dis\_B2\_Ball <= dis\_B3\_Ball
374. theta\_B=theta\_B2;
375. end
376. if dis\_B3\_Ball < dis\_B1\_Ball && dis\_B3\_Ball < dis\_B2\_Ball
377. theta\_B=theta\_B3;
378. end
380. %updates the position and orientation of player A1 in a soccer game simulation. The code computes two potential new positions for A1 based on the current position of the ball and the position of player B, and chooses the one that is closer to the goal.
381. theta\_A1\_next1=theta\_B+pi/2;
382. delta\_x\_A1\_next1=v\_A1\*cos(theta\_A1\_next1);
383. delta\_y\_A1\_next1=v\_A1\*sin(theta\_A1\_next1);
384. A1\_posnew\_1(1)=A1\_pos(1)-delta\_x\_A1\_next1;
385. A1\_posnew\_1(2)=A1\_pos(2)-delta\_y\_A1\_next1;
386. theta\_A1\_next2=theta\_B-pi/2;
387. delta\_x\_A1\_next2=v\_A1\*cos(theta\_A1\_next2);
388. delta\_y\_A1\_next2=v\_A1\*sin(theta\_A1\_next2);
389. A1\_posnew\_2(1)=A1\_pos(1)-delta\_x\_A1\_next2;
390. A1\_posnew\_2(2)=A1\_pos(2)-delta\_y\_A1\_next2;
391. %calculate the distance from goal\_1 and goal\_2
392. dis\_A1\_next1=getDis(A1\_posnew\_1(1),A1\_posnew\_1(2),Goal1\_pos\_abs(1),Goal1\_pos\_abs(2));
393. dis\_A1\_next2=getDis(A1\_posnew\_2(1),A1\_posnew\_2(2),Goal1\_pos\_abs(1),Goal1\_pos\_abs(2));
394. if dis\_A1\_next1<dis\_A1\_next2
395. theta\_A1=theta\_A1\_next1;
396. end
397. if dis\_A1\_next1>dis\_A1\_next2
398. theta\_A1=theta\_A1\_next2;
399. end
400. if dis\_A1\_next1==dis\_A1\_next2
401. if rand(1)>=0.5
402. theta\_A1=theta\_A1\_next1;
403. else
404. theta\_A1=theta\_A1\_next2;
405. end
406. end
407. delta\_x\_A1=v\_A1\*cos(theta\_A1);
408. delta\_y\_A1=v\_A1\*sin(theta\_A1);
409. A1\_posnew(1)=A1\_pos(1)-delta\_x\_A1;
410. A1\_posnew(2)=A1\_pos(2)-delta\_y\_A1;
412. if A1\_posnew(2)>=7
413. A1\_pos(1)=A1\_posnew(1);
414. A1\_pos(2)=7;
415. else
416. if A1\_posnew(2)<=1
417. A1\_pos(1)=A1\_posnew(1);
418. A1\_pos(2)=1;
419. else
420. A1\_pos(1)=A1\_posnew(1);
421. A1\_pos(2)=A1\_posnew(2);
422. end
423. end
424. Ball\_pos(1)=A1\_pos(1);
425. Ball\_pos(2)=A1\_pos(2);
426. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,A4\_pos,theta\_A4,
427. B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
428. pause(0.2);
430. %calculate the distance between A1 and two sides of the goal (top and bottom)
431. %if the distance A1 to top of Goal\_1 is shorter than A1 to bottom to Goal\_1, then the player will face to the top side of the goal.
432. %-----A1 to t or b-----%
433. dis\_A1\_Goal1\_t=getDis(A1\_pos(1),A1\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
434. t\_A1\_t=dis\_A1\_Goal1\_t/v\_A1;
435. num\_A1\_t=ceil(t\_A1\_t);
436. dis\_A1\_Goal1\_b=getDis(A1\_pos(1),A1\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
437. t\_A1\_b=dis\_A1\_Goal1\_b/v\_A1;
438. num\_A1\_b=ceil(t\_A1\_b);
439. if num\_A1\_t< num\_A1\_b
440. %if the distance A1 to top of Goal\_1 is shorter than A1 to bottom to Goal\_1, then the player will choose to the top side of the goal.
441. Goal1\_pos=Goal1\_t\_pos;
442. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
443. dis\_A1\_Goal1=dis\_A1\_Goal1\_t;
444. end
445. if num\_A1\_b< num\_A1\_t
446. Goal1\_pos=Goal1\_b\_pos;
447. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
448. dis\_A1\_Goal1=dis\_A1\_Goal1\_b;
449. end
450. %if the player is in the middle, then it will randomly choose one side (facing top or bottom) and turn.
451. if num\_A1\_t==num\_A1\_b
452. if rand(1)>=0.5
453. Goal1\_pos=Goal1\_t\_pos;
454. theta\_A1= getTheta(A1\_pos(1),A1\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
455. dis\_A1\_Goal1=dis\_A1\_Goal1\_t;
456. else
457. Goal1\_pos=Goal1\_b\_pos;
458. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
459. dis\_A1\_Goal1=dis\_A1\_Goal1\_b;
460. end
461. end
462. %----------------------%
464. %When A1 gets the ball, it will calculate the distance between A1 and Goal1, and the shooting range for the players is 3. A1/A2/A3 becomes attackers and the rest of the teammates can get the distance from ball.The B team players also get the distance from ball.
465. delta\_dis=dis\_A1\_Goal1-3;
466. if delta\_dis>0
467. t\_A1=delta\_dis/v\_A1;
468. num\_A1=ceil(t\_A1);
470. %-----Ball to B1/B2/B3-----%
471. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
472. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
473. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
474. %------------------%
476. for i=1:num\_A1
477. if Ball\_right==1
478. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
479. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
480. Ball\_pos(1)=A1\_pos(1);
481. Ball\_pos(2)=A1\_pos(2);
483. %-----A2/A3-----%
484. if Goal1\_pos(2)==Goal1\_t\_pos(2)
485. A\_Attack\_pos=[A1\_pos(1),A1\_pos(2)-2];
486. end
487. if Goal1\_pos(2)==Goal1\_b\_pos(2)
488. A\_Attack\_pos=[A1\_pos(1),A1\_pos(2)+2];
489. end
490. dis\_A2\_Attack=getDis(A2\_pos(1),A2\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
491. dis\_A3\_Attack=getDis(A3\_pos(1),A3\_pos(2),A\_Attack\_pos(1),
492. A\_Attack\_pos(2));
493. %------------------------------------------%
495. %-----A2 attack-----%
496. %Compare the distance between ball and A2 and the distance from ball to A3, then update the positions and facing angles of A2 and A3.
497. if dis\_A2\_Attack<=dis\_A3\_Attack
498. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
499. A3\_x=A1\_pos(1)-sqrt(3^2-(A1\_pos(2)-4)^2);
500. dis\_A3\_Attack\_new=getDis(A3\_pos(1),A3\_pos(2),A3\_x,4);
501. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A3\_x,4);
502. %update the A2 position. Using the velocity and position to update the positon of A2 in every fram until A2 goes to the A\_Attack\_pos.
503. if dis\_A2\_Attack>v\_A2
504. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
505. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
506. else
507. A2\_pos(1)=A\_Attack\_pos(1);
508. A2\_pos(2)=A\_Attack\_pos(2);
509. end
511. if dis\_A3\_Attack\_new>v\_A3
512. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
513. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
514. else
515. A3\_pos(1)=A3\_x;
516. A3\_pos(2)=4;
517. end
518. end
519. %-------------------%
520. %When A3 is closer to the A\_Attack\_pos, update the A3 position. Using the velocity and position to update the positon of A2 in every frame until A3 goes to the A\_Attack\_pos.
521. %-----A3 attack-----%
522. if dis\_A3\_Attack<dis\_A2\_Attack
523. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
524. A2\_x=A1\_pos(1)-sqrt(3^2-(A1\_pos(2)-4)^2);
525. dis\_A2\_Attack\_new=getDis(A2\_pos(1),A2\_pos(2),A2\_x,4);
526. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A2\_x,4);
528. if dis\_A3\_Attack>v\_A3
529. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
530. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
531. else
532. A3\_pos(1)=A\_Attack\_pos(1);
533. A3\_pos(2)=A\_Attack\_pos(2);
534. end
536. if dis\_A2\_Attack\_new>v\_A2
537. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
538. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
539. else
540. A2\_pos(1)=A2\_x;
541. A2\_pos(2)=4;
542. end
543. end
544. %-------------------%
545. %Here is the situation when B1 is defensing and it is the closest one to the ball, update the distance and angle to the ball. Also update those variables of B2 and B3. When team A gets the ball, team B will defend and B1 is set to chase the ball while B2 and B3 will go back to the defind\_pos first which is already set.
546. %------B1 to Ball-----%
547. if dis\_B1\_Ball<=dis\_B2\_Ball && dis\_B1\_Ball<=dis\_B3\_Ball
548. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
549. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
550. dis\_B2\_Defend=getDis(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),
551. defind\_pos\_1(2));
552. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
553. dis\_B3\_Defend=getDis(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
554. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
555. %Update the positions of B2 and B3, showing them are running until they are in the specific positions and then turn to face the ball.
556. if dis\_B2\_Defend>v\_B2
557. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
558. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
559. else
560. B2\_pos(1)=defind\_pos\_1(1);
561. B2\_pos(2)=defind\_pos\_1(2);
562. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
563. end
565. if dis\_B3\_Defend>v\_B3
566. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
567. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
568. else
569. B3\_pos(1)=defind\_pos\_2(1);
570. B3\_pos(2)=defind\_pos\_2(2);
571. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
572. end
574. if dis\_B1\_Ball>v\_B1
575. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
576. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
577. else
578. B1\_pos(1)=Ball\_pos(1);
579. B1\_pos(2)=Ball\_pos(2);
580. theta\_B1=theta\_A1;
581. if rand(1)>=0.3
582. Ball\_right=5;
583. end
584. end
585. end
586. %--------------------%
588. %When B2 is the closest defender to the ball, B2 will goes to stop the attacker. B1 and B3 will goes to the defind\_pos and the turn their direction fo the ball (the defend positions which are already set at the beginning). The code shown below are similar to the code shown above.
589. %-----B2 to Ball-----%
590. if dis\_B2\_Ball<dis\_B1\_Ball && dis\_B2\_Ball<=dis\_B3\_Ball
591. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
592. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
593. dis\_B1\_Defend=getDis(B1\_pos(1),B1\_pos(2),defind\_pos\_1(1),
594. defind\_pos\_1(2));
595. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
596. dis\_B3\_Defend=getDis(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
597. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
599. if dis\_B1\_Defend>v\_B1
600. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
601. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
602. else
603. B1\_pos(1)=defind\_pos\_1(1);
604. B1\_pos(2)=defind\_pos\_1(2);
605. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
606. end
608. if dis\_B3\_Defend>v\_B3
609. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
610. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
611. else
612. B3\_pos(1)=defind\_pos\_2(1);
613. B3\_pos(2)=defind\_pos\_2(2);
614. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
615. end
617. if dis\_B2\_Ball>v\_B2
618. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
619. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
620. else
621. B2\_pos(1)=Ball\_pos(1);
622. B2\_pos(2)=Ball\_pos(2);
623. theta\_B2=theta\_A1;
624. if rand(1)>=0.3
625. Ball\_right=6;
626. end
627. end
628. end
629. %--------------------%
631. %When B3 is the defender who is the closest one to the ball, B3 will chase the ball. B1 and B2 will do the same action, goes to the defind\_pos\_1 and defind\_pos\_2 and then turn to the ball.
632. %-----B3 to Ball-----%
633. if dis\_B3\_Ball<dis\_B1\_Ball && dis\_B3\_Ball<dis\_B2\_Ball
634. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
635. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
636. dis\_B1\_Defend=getDis(B1\_pos(1),B1\_pos(2),defind\_pos\_2(1),
637. defind\_pos\_2(2));
638. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
639. dis\_B2\_Defend=getDis(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
640. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
641. if dis\_B1\_Defend>v\_B1
642. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
643. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
644. else
645. B1\_pos(1)=defind\_pos\_2(1);
646. B1\_pos(2)=defind\_pos\_2(2);
647. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
648. end
650. if dis\_B2\_Defend>v\_B2
651. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
652. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
653. else
654. B2\_pos(1)=defind\_pos\_1(1);
655. B2\_pos(2)=defind\_pos\_1(2);
656. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
657. end
659. if dis\_B3\_Ball>v\_B3
660. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
661. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
662. else
663. B3\_pos(1)=Ball\_pos(1);
664. B3\_pos(2)=Ball\_pos(2);
665. theta\_B3=theta\_A1;
666. if rand(1)>=0.3
667. Ball\_right=7;
668. end
669. end
670. end
671. %--------------------%
672. end
673. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,A4\_pos,
674. theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
675. pause(0.2);
676. end
677. end
678. %When A team is attacking and A1 is dribbling, but A1 is facing the defenders, A1 will pass the ball to A2 or A3 depending the different situations. So the distances from A1 to every players on the court are important.
679. if Ball\_right==1
680. dis\_A1\_B1=getDis(A1\_pos(1),A1\_pos(2),B1\_pos(1),B1\_pos(2));
681. dis\_A1\_B2=getDis(A1\_pos(1),A1\_pos(2),B2\_pos(1),B2\_pos(2));
682. dis\_A1\_B3=getDis(A1\_pos(1),A1\_pos(2),B3\_pos(1),B3\_pos(2));
683. dis\_A1\_A2=getDis(A1\_pos(1),A1\_pos(2),A2\_pos(1),A2\_pos(2));
684. dis\_A1\_A3=getDis(A1\_pos(1),A1\_pos(2),A3\_pos(1),A3\_pos(2));
685. dis\_A1\_Goal1=getDis(A1\_pos(1),A1\_pos(2),Goal1\_pos(1),Goal1\_pos(2));
686. t\_A1\_Goal1=dis\_A1\_Goal1/v\_Ball;
687. num\_A1\_Goal1=ceil(t\_A1\_Goal1);
689. %When A1 is out of the range and facing the defender who is close enough, first we calculate the distance from A2 and A3 respectively. If A2 is closer to A1, A1 will pass the ball to A2, otherwise A1 will pass the ball to A3.
690. if num\_A1\_Goal1 > frame
691. if dis\_A1\_B1<=v\_B1 || dis\_A1\_B2<=v\_B2 || dis\_A1\_B3<=v\_B3
692. if A2\_pos(1)>=A3\_pos(1)
693. dis\_A=dis\_A1\_A2;
694. else
695. dis\_A=dis\_A1\_A3;
696. end
698. %-----Pass Ball-----%
699. if dis\_A<=3
700. if dis\_A==dis\_A1\_A2
701. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A2\_pos(1),A2\_pos(2));
702. A\_pos(1)=A2\_pos(1);
703. A\_pos(2)=A2\_pos(2);
704. else
705. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A3\_pos(1),A3\_pos(2));
706. A\_pos(1)=A3\_pos(1);
707. A\_pos(2)=A3\_pos(2);
708. end
710. t\_A=dis\_A/v\_Ball;
711. num\_A=ceil(t\_A);
713. %Update the ball position and the defenders positions while the ball is passing. If one of the players gets the ball, the state of Ball\_right changes.
714. for i=1:num\_A
715. if Ball\_right==1
716. if i<num\_A
717. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A1);
718. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A1);
719. else
720. Ball\_pos(1)=A\_pos(1);
721. Ball\_pos(2)=A\_pos(2);
722. end
724. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
725. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
726. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
727. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
728. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
729. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
731. if dis\_B1\_Ball>v\_B1
732. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
733. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
734. else
735. B1\_pos(1)=Ball\_pos(1);
736. B1\_pos(2)=Ball\_pos(2);
737. Ball\_right=5;
738. end
740. if dis\_B2\_Ball>v\_B2
741. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
742. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
743. else
744. B2\_pos(1)=Ball\_pos(1);
745. B2\_pos(2)=Ball\_pos(2);
746. Ball\_right=6;
747. end
749. if dis\_B3\_Ball>v\_B3
750. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
751. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
752. else
753. B3\_pos(1)=Ball\_pos(1);
754. B3\_pos(2)=Ball\_pos(2);
755. Ball\_right=7;
756. end
757. %Update the distance and angle from A2 and A3 to the ball.
758. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
759. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
760. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
761. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
763. if dis\_A2\_Ball==0
764. Ball\_right=2;
765. end
767. if dis\_A3\_Ball==0
768. Ball\_right=3;
769. end
770. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,
771. A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,
772. B4\_pos,theta\_B4);
773. pause(0.2);
774. end
775. end
776. end
777. %-------------------%
779. %-----Kick Ball-----%
780. %When A1 kick the ball to the goal, update positions of every defender. If one of the defender stop the ball then Ball\_right will change.
781. if Ball\_right==1
782. t\_A1\_Goal1=dis\_A1\_Goal1/v\_Ball;
783. num\_A1\_Goal1=ceil(t\_A1\_Goal1);
785. for i=1:num\_A1\_Goal1
786. if Ball\_right==1
787. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A1);
788. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A1);
789. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
790. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
791. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
792. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
793. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
794. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
795. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
796. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
797. if dis\_B1\_Ball>v\_B1
798. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
799. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
800. else
801. B1\_pos(1)=Ball\_pos(1);
802. B1\_pos(2)=Ball\_pos(2);
803. Ball\_right=5;
804. end
806. if dis\_B2\_Ball>v\_B2
807. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
808. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
809. else
810. B2\_pos(1)=Ball\_pos(1);
811. B2\_pos(2)=Ball\_pos(2);
812. Ball\_right=6;
813. end
815. if dis\_B3\_Ball>v\_B3
816. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
817. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
818. else
819. B3\_pos(1)=Ball\_pos(1);
820. B3\_pos(2)=Ball\_pos(2);
821. Ball\_right=7;
822. end
824. %Goalkeeper in B team will do actions to defend, if attacker shoot to the top side of the goal and goalkeeper will move to the to top side and try to save the ball. If the goalkeeper save the ball, the Ball\_right changes.
825. if Goal1\_pos(2)==Goal1\_t\_pos(2)
826. a=1;
827. else
828. a=-1;
829. end
831. B4\_pos(1)=B4\_pos(1);
832. B4\_pos(2)=B4\_pos(2)+a\*v\_B4;
834. if B4\_pos(2) >= Goal1\_t\_pos(2)
835. B4\_pos(2)=Goal1\_t\_pos(2);
836. end
838. if B4\_pos(2) <= Goal1\_b\_pos(2)
839. B4\_pos(2)=Goal1\_b\_pos(2);
840. end
842. if Ball\_pos(1)>Goal1\_pos(1)
843. Ball\_pos(1)=Goal1\_pos(1);
844. Ball\_pos(2)=Goal1\_pos(2);
845. end
847. if Ball\_pos(1)==B4\_pos(1) && Ball\_pos(2)==B4\_pos(2)
848. Ball\_right=8;
849. end
851. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
852. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
853. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,
854. B3\_pos,theta\_B3,B4\_pos,theta\_B4);
855. pause(0.2);
856. end
857. end
858. pause(1);
859. end
860. %-------------------%
862. end
864. %When A1 is dribbling, A2 and A3 will move to the specific spot.
865. if Ball\_right==1
866. det\_dis\_A1\_Goal1=dis\_A1\_Goal1-frame\*v\_Ball;
867. t\_det=det\_dis\_A1\_Goal1/v\_A1;
868. num\_det=ceil(t\_det);
870. %-----move with ball-----%
871. for i=1:num\_det
872. if Ball\_right==1
873. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
874. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
875. Ball\_pos(1)=A1\_pos(1);
876. Ball\_pos(2)=A1\_pos(2);
878. %-----A2/A3-----%
879. if Goal1\_pos(2)==Goal1\_t\_pos(2)
880. A\_Attack\_pos=[A1\_pos(1),A1\_pos(2)-2];
881. end
882. if Goal1\_pos(2)==Goal1\_b\_pos(2)
883. A\_Attack\_pos=[A1\_pos(1),A1\_pos(2)+2];
884. end
885. dis\_A2\_Attack=getDis(A2\_pos(1),A2\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
886. dis\_A3\_Attack=getDis(A3\_pos(1),A3\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
887. %---------------%
889. %-----A2 attack-----%
890. if dis\_A2\_Attack<=dis\_A3\_Attack
891. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
892. A3\_x=A1\_pos(1)-sqrt(3^2-(A1\_pos(2)-4)^2);
893. dis\_A3\_Attack\_new=getDis(A3\_pos(1),A3\_pos(2),A3\_x,4);
894. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A3\_x,4);
896. if dis\_A2\_Attack>v\_A2
897. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
898. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
899. else
900. A2\_pos(1)=A\_Attack\_pos(1);
901. A2\_pos(2)=A\_Attack\_pos(2);
902. end
904. if dis\_A3\_Attack\_new>v\_A3
905. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
906. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
907. else
908. A3\_pos(1)=A3\_x;
909. A3\_pos(2)=4;
910. end
912. end
913. %-------------------%
915. %-----A3 attack-----%
916. if dis\_A3\_Attack<dis\_A2\_Attack
917. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
918. A2\_x=A1\_pos(1)-sqrt(3^2-(A1\_pos(2)-4)^2);
919. dis\_A2\_Attack\_new=getDis(A2\_pos(1),A2\_pos(2),A2\_x,4);
920. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A2\_x,4);
922. if dis\_A3\_Attack>v\_A3
923. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
924. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
925. else
926. A3\_pos(1)=A\_Attack\_pos(1);
927. A3\_pos(2)=A\_Attack\_pos(2);
928. end
930. if dis\_A2\_Attack\_new>v\_A2
931. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
932. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
933. else
934. A2\_pos(1)=A2\_x;
935. A2\_pos(2)=4;
936. end
938. end
939. %-------------------%
941. %Update the defenders positions and angles. When the defenders try to take the ball, there is a random decision (or chance) to decide if the defenders can take the ball to make a sussessful defense. If defenders make the successful defense and then the Ball\_right will change.
942. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
943. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
944. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
945. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
946. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
947. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
949. if dis\_B1\_Ball>v\_B1
950. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
951. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
952. else
953. B1\_pos(1)=Ball\_pos(1);
954. B1\_pos(2)=Ball\_pos(2);
955. theta\_B1=theta\_A1;
956. if rand(1)>=0.3
957. Ball\_right=5;
958. end
959. end
961. if dis\_B2\_Ball>v\_B2
962. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
963. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
964. else
965. B2\_pos(1)=Ball\_pos(1);
966. B2\_pos(2)=Ball\_pos(2);
967. theta\_B2=theta\_A1;
968. if rand(1)>=0.3
969. Ball\_right=6;
970. end
971. end
973. if dis\_B3\_Ball>v\_B3
974. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
975. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
976. else
977. B3\_pos(1)=Ball\_pos(1);
978. B3\_pos(2)=Ball\_pos(2);
979. theta\_B3=theta\_A1;
980. if rand(1)>=0.3
981. Ball\_right=7;
982. end
983. end
984. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
985. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,
986. theta\_B3,B4\_pos,theta\_B4);
987. pause(0.2);
988. end
989. end
990. %------------------------%
992. end
993. end
995. if Ball\_right==1
996. dis\_A1\_Goal1=getDis(A1\_pos(1),A1\_pos(2),Goal1\_pos(1),Goal1\_pos(2));
997. t\_A1\_Goal1=dis\_A1\_Goal1/v\_Ball;
998. num\_A1\_Goal1=ceil(t\_A1\_Goal1);
1000. %-----Kick Ball-----%
1001. for i=1:num\_A1\_Goal1
1002. if Ball\_right==1
1003. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A1);
1004. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A1);
1005. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1006. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1008. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1009. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1010. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1011. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1012. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1013. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1015. if dis\_B1\_Ball>v\_B1
1016. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1017. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1018. else
1019. B1\_pos(1)=Ball\_pos(1);
1020. B1\_pos(2)=Ball\_pos(2);
1021. Ball\_right=5;
1022. end
1024. if dis\_B2\_Ball>v\_B2
1025. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1026. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1027. else
1028. B2\_pos(1)=Ball\_pos(1);
1029. B2\_pos(2)=Ball\_pos(2);
1030. Ball\_right=6;
1031. end
1033. if dis\_B3\_Ball>v\_B3
1034. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1035. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1036. else
1037. B3\_pos(1)=Ball\_pos(1);
1038. B3\_pos(2)=Ball\_pos(2);
1039. Ball\_right=7;
1040. end
1042. if Goal1\_pos(2)==Goal1\_t\_pos(2)
1043. a=1;
1044. else
1045. a=-1;
1046. end
1048. B4\_pos(1)=B4\_pos(1);
1049. B4\_pos(2)=B4\_pos(2)+a\*v\_B4;
1051. if Ball\_pos(1)>Goal1\_pos(1)
1052. Ball\_pos(1)=Goal1\_pos(1);
1053. Ball\_pos(2)=Goal1\_pos(2);
1054. end
1056. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
1057. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
1058. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,
1059. B3\_pos,theta\_B3,B4\_pos,theta\_B4);
1060. pause(0.2);
1061. end
1062. end
1063. %-------------------%
1065. %If goalkeeper save the ball successfully, then Ball\_right turns to 8. If goalkeeper cannot catch the ball, A team scores.
1066. Ball\_right=8;
1067. if Ball\_pos(2)~=B4\_pos(2)
1068. score\_A=score\_A+1;
1069. end
1070. pause(1);
1071. end
1072. end
1073. end
1075. %The code below show the different situations when A2 gets the ball. A team will be the attackers and most of this part is similar to the part that A1 gets the ball and A team attacks.
1076. while (Ball\_right==2)
1077. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1078. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1079. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1080. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1081. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1082. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1083. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1084. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1085. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1086. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1087. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1088. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1089. if dis\_B1\_Ball <= dis\_B2\_Ball && dis\_B1\_Ball <= dis\_B3\_Ball
1090. theta\_B=theta\_B1;
1091. end
1092. if dis\_B2\_Ball < dis\_B1\_Ball && dis\_B2\_Ball <= dis\_B3\_Ball
1093. theta\_B=theta\_B2;
1094. end
1095. if dis\_B3\_Ball < dis\_B1\_Ball && dis\_B3\_Ball < dis\_B2\_Ball
1096. theta\_B=theta\_B3;
1097. end
1098. theta\_A2\_next1=theta\_B+pi/2;
1099. delta\_x\_A2\_next1=v\_A2\*cos(theta\_A2\_next1);
1100. delta\_y\_A2\_next1=v\_A2\*sin(theta\_A2\_next1);
1101. A2\_posnew\_1(1)=A2\_pos(1)-delta\_x\_A2\_next1;
1102. A2\_posnew\_1(2)=A2\_pos(2)-delta\_y\_A2\_next1;
1103. theta\_A2\_next2=theta\_B-pi/2;
1104. delta\_x\_A2\_next2=v\_A2\*cos(theta\_A2\_next2);
1105. delta\_y\_A2\_next2=v\_A2\*sin(theta\_A2\_next2);
1106. A2\_posnew\_2(1)=A2\_pos(1)-delta\_x\_A2\_next2;
1107. A2\_posnew\_2(2)=A2\_pos(2)-delta\_y\_A2\_next2;
1108. dis\_A2\_next1=getDis(A2\_posnew\_1(1),A2\_posnew\_1(2),Goal1\_pos\_abs(1),Goal1\_pos\_abs(2));
1109. dis\_A2\_next2=getDis(A2\_posnew\_2(1),A2\_posnew\_2(2),Goal1\_pos\_abs(1),Goal1\_pos\_abs(2));
1110. if dis\_A2\_next1<dis\_A2\_next2
1111. theta\_A2=theta\_A2\_next1;
1112. end
1113. if dis\_A2\_next1>dis\_A2\_next2
1114. theta\_A2=theta\_A2\_next2;
1115. end
1116. if dis\_A2\_next1==dis\_A2\_next2
1117. if rand(1)>=0.5
1118. theta\_A2=theta\_A2\_next1;
1119. else
1120. theta\_A2=theta\_A2\_next2;
1121. end
1122. end
1123. delta\_x\_A2=v\_A2\*cos(theta\_A2);
1124. delta\_y\_A2=v\_A2\*sin(theta\_A2);
1125. A2\_posnew(1)=A2\_pos(1)-delta\_x\_A2;
1126. A2\_posnew(2)=A2\_pos(2)-delta\_y\_A2;
1127. if A2\_posnew(2)>=7
1128. A2\_pos(1)=A2\_posnew(1);
1129. A2\_pos(2)=7;
1130. else
1131. if A2\_posnew(2)<=1
1132. A2\_pos(1)=A2\_posnew(1);
1133. A2\_pos(2)=1;
1134. else
1135. A2\_pos(1)=A2\_posnew(1);
1136. A2\_pos(2)=A2\_posnew(2);
1137. end
1138. end
1139. Ball\_pos(1)=A2\_pos(1);
1140. Ball\_pos(2)=A2\_pos(2);
1141. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,
1142. A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,
1143. theta\_B4);
1144. pause(0.2);
1145. %calculates the time it will take for a robot called A2 to reach two possible goal positions
1146. %It uses the distance formula to calculate the distance between A2 and each of the two goal positions. It then divides the distance by A2's speed, v\_A2, to get the time it will take to reach each position.
1147. %-----A2 to t or b-----%
1148. dis\_A2\_Goal1\_t=getDis(A2\_pos(1),A2\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
1149. t\_A2\_t=dis\_A2\_Goal1\_t/v\_A2;
1150. num\_A2\_t=ceil(t\_A2\_t);
1151. dis\_A2\_Goal1\_b=getDis(A2\_pos(1),A2\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
1152. t\_A2\_b=dis\_A2\_Goal1\_b/v\_A2;
1153. num\_A2\_b=ceil(t\_A2\_b);
1154. if num\_A2\_t < num\_A2\_b
1155. Goal1\_pos=Goal1\_t\_pos;
1156. theta\_A2= getTheta(A2\_pos(1),A2\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
1157. dis\_A2\_Goal1=dis\_A2\_Goal1\_t;
1158. end
1159. if num\_A2\_t > num\_A2\_b
1160. Goal1\_pos=Goal1\_b\_pos;
1161. theta\_A2= getTheta(A2\_pos(1),A2\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
1162. dis\_A2\_Goal1=dis\_A2\_Goal1\_b;
1163. end
1164. if num\_A2\_t==num\_A2\_b
1165. if rand(1)>=0.5
1166. Goal1\_pos=Goal1\_t\_pos;
1167. theta\_A2= getTheta(A2\_pos(1),A2\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
1168. dis\_A2\_Goal1=dis\_A2\_Goal1\_t;
1169. else
1170. Goal1\_pos=Goal1\_b\_pos;
1171. theta\_A2= getTheta(A2\_pos(1),A2\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
1172. dis\_A2\_Goal1=dis\_A2\_Goal1\_b;
1173. end
1174. end
1175. %----------------------%
1176. % calculates the distance between player A2's position and the target goal position (Goal1)
1177. delta\_dis=dis\_A2\_Goal1-3;
1178. % If the distance is greater than zero, it calculates the time required to reach the goal and the number of steps required to reach the goal
1179. if delta\_dis>0
1180. t\_A2=delta\_dis/v\_A2;
1181. num\_A2=ceil(t\_A2);
1183. %-----B1/B2/B3-----%
1184. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1185. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1186. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1187. %------------------%
1189. % execute the movement of the ball and the players (A1, A2, A3) until the ball reaches the goal or the loop ends
1190. % calculate the distance between each of the players B1, B2, and B3 and the ball. If the ball is on the right side, player A2 moves towards the ball's position, and the code checks if player A1 or player A3 is in a better position to attack the goal. The code then calculates the position and movement of player A1 or player A3 accordingly.
1191. for i=1:num\_A2
1192. if Ball\_right==2
1193. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
1194. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
1195. Ball\_pos(1)=A2\_pos(1);
1196. Ball\_pos(2)=A2\_pos(2);
1198. %-----A1/A3-----%
1199. if Goal1\_pos(2)==Goal1\_t\_pos(2)
1200. A\_Attack\_pos=[A2\_pos(1),A2\_pos(2)-2];
1201. end
1202. if Goal1\_pos(2)==Goal1\_b\_pos(2)
1203. A\_Attack\_pos=[A2\_pos(1),A2\_pos(2)+2];
1204. end
1205. dis\_A1\_Attack=getDis(A1\_pos(1),A1\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1206. dis\_A3\_Attack=getDis(A3\_pos(1),A3\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1207. %---------------%
1209. %-----A1 attack-----%
1210. if dis\_A1\_Attack<=dis\_A3\_Attack
1211. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1212. A3\_x=A2\_pos(1)-sqrt(3^2-(A2\_pos(2)-4)^2);
1213. dis\_A3\_Attack\_new=getDis(A3\_pos(1),A3\_pos(2),A3\_x,4);
1214. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A3\_x,4);
1216. if dis\_A1\_Attack>v\_A1
1217. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
1218. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
1219. else
1220. A1\_pos(1)=A\_Attack\_pos(1);
1221. A1\_pos(2)=A\_Attack\_pos(2);
1222. end
1224. if dis\_A3\_Attack\_new>v\_A3
1225. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
1226. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
1227. else
1228. A3\_pos(1)=A3\_x;
1229. A3\_pos(2)=4;
1230. end
1232. end
1233. %-------------------%
1235. %-----A3 attack-----%
1236. if dis\_A3\_Attack<dis\_A1\_Attack
1237. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1238. A1\_x=A2\_pos(1)-sqrt(3^2-(A2\_pos(2)-4)^2);
1239. dis\_A1\_Attack\_new=getDis(A1\_pos(1),A1\_pos(2),A1\_x,4);
1240. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A1\_x,4);
1242. if dis\_A3\_Attack>v\_A3
1243. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
1244. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
1245. else
1246. A3\_pos(1)=A\_Attack\_pos(1);
1247. A3\_pos(2)=A\_Attack\_pos(2);
1248. end
1250. if dis\_A1\_Attack\_new>v\_A1
1251. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
1252. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
1253. else
1254. A1\_pos(1)=A1\_x;
1255. A1\_pos(2)=4;
1256. end
1258. end
1259. %-------------------%
1260. %handles the movement of the ball and the three players on the team B. The code also includes a random chance for the ball to be passed to another player on the defending team (either B1 or B2) if they have possession of the ball, indicated by the "Ball\_right" variable.
1261. %------B1 to Ball-----%
1262. if dis\_B1\_Ball<=dis\_B2\_Ball && dis\_B1\_Ball<=dis\_B3\_Ball
1263. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1264. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1265. dis\_B2\_Defend=getDis(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
1266. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
1267. dis\_B3\_Defend=getDis(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
1268. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
1269. %The code starts with the defending player B1 moving towards the ball, as long as their distance to the ball is greater than the speed at which they can move (v\_B1). The other two defending players (B2 and B3) will also move towards the ball if their distances to the ball are greater than their respective speeds.
1270. if dis\_B2\_Defend>v\_B2
1271. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1272. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1273. else
1274. B2\_pos(1)=defind\_pos\_1(1);
1275. B2\_pos(2)=defind\_pos\_1(2);
1276. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1277. end
1278. %If a team member reaches the ball, they will stop moving and take possession of the ball. The other two players will then move towards their designated positions to block the attacking team.
1279. if dis\_B3\_Defend>v\_B3
1280. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1281. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1282. else
1283. B3\_pos(1)=defind\_pos\_2(1);
1284. B3\_pos(2)=defind\_pos\_2(2);
1285. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1286. end
1288. if dis\_B1\_Ball>v\_B1
1289. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1290. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1291. else
1292. B1\_pos(1)=Ball\_pos(1);
1293. B1\_pos(2)=Ball\_pos(2);
1294. theta\_B1=theta\_A2;
1295. if rand(1)>=0.3
1296. Ball\_right=5;
1297. end
1298. end
1299. end
1300. %--------------------%
1301. %-----B2 to Ball-----%
1302. if dis\_B2\_Ball<dis\_B1\_Ball && dis\_B2\_Ball<=dis\_B3\_Ball
1303. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1304. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1305. dis\_B1\_Defend=getDis(B1\_pos(1),B1\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
1306. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
1307. dis\_B3\_Defend=getDis(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
1308. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
1310. if dis\_B1\_Defend>v\_B1
1311. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1312. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1313. else
1314. B1\_pos(1)=defind\_pos\_1(1);
1315. B1\_pos(2)=defind\_pos\_1(2);
1316. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1317. end
1319. if dis\_B3\_Defend>v\_B3
1320. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1321. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1322. else
1323. B3\_pos(1)=defind\_pos\_2(1);
1324. B3\_pos(2)=defind\_pos\_2(2);
1325. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1326. end
1328. if dis\_B2\_Ball>v\_B2
1329. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1330. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1331. else
1332. B2\_pos(1)=Ball\_pos(1);
1333. B2\_pos(2)=Ball\_pos(2);
1334. theta\_B2=theta\_A2;
1335. if rand(1)>=0.3
1336. Ball\_right=6;
1337. end
1338. end
1340. end
1341. %--------------------%
1343. %-----B3 to Ball-----%
1344. if dis\_B3\_Ball<dis\_B1\_Ball && dis\_B3\_Ball<dis\_B2\_Ball
1345. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1346. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1347. dis\_B1\_Defend=getDis(B1\_pos(1),B1\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
1348. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
1349. dis\_B2\_Defend=getDis(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
1350. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
1351. if dis\_B1\_Defend>v\_B1
1352. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1353. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1354. else
1355. B1\_pos(1)=defind\_pos\_2(1);
1356. B1\_pos(2)=defind\_pos\_2(2);
1357. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1358. end
1360. if dis\_B2\_Defend>v\_B2
1361. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1362. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1363. else
1364. B2\_pos(1)=defind\_pos\_1(1);
1365. B2\_pos(2)=defind\_pos\_1(2);
1366. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1367. end
1369. if dis\_B3\_Ball>v\_B3
1370. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1371. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1372. else
1373. B3\_pos(1)=Ball\_pos(1);
1374. B3\_pos(2)=Ball\_pos(2);
1375. theta\_B3=theta\_A2;
1376. if rand(1)>=0.3
1377. Ball\_right=7;
1378. end
1379. end
1381. end
1382. %--------------------%
1383. end
1384. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
1385. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
1386. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
1387. pause(0.2);
1388. end
1389. end
1390. %check if A2 has the ball right. It calculates the distance between A2 and each of the other players and the distance between A2 and a goal. If any of these players are close enough, the program checks which of the two closest players from A1 and A3 is closer to A2 and passes the ball to that player.
1391. if Ball\_right==2
1392. dis\_A2\_B1=getDis(A2\_pos(1),A2\_pos(2),B1\_pos(1),B1\_pos(2));
1393. dis\_A2\_B2=getDis(A2\_pos(1),A2\_pos(2),B2\_pos(1),B2\_pos(2));
1394. dis\_A2\_B3=getDis(A2\_pos(1),A2\_pos(2),B3\_pos(1),B3\_pos(2));
1395. dis\_A2\_A1=getDis(A2\_pos(1),A2\_pos(2),A1\_pos(1),A1\_pos(2));
1396. dis\_A2\_A3=getDis(A2\_pos(1),A2\_pos(2),A3\_pos(1),A3\_pos(2));
1397. dis\_A2\_Goal1=getDis(A2\_pos(1),A2\_pos(2),Goal1\_pos(1),Goal1\_pos(2));
1398. t\_A2\_Goal1=dis\_A2\_Goal1/v\_Ball;
1399. num\_A2\_Goal1=ceil(t\_A2\_Goal1);
1401. if num\_A2\_Goal1 > frame
1402. if dis\_A2\_B1<=v\_B1 || dis\_A2\_B2<=v\_B2 || dis\_A2\_B3<=v\_B3
1404. if A1\_pos(1)>=A3\_pos(1)
1405. dis\_A=dis\_A2\_A1;
1406. else
1407. dis\_A=dis\_A2\_A3;
1408. end
1409. %If the distance between A2 and the chosen player is less than or equal to 3, the program simulates the passing of the ball from A2 to the chosen player by calculating the distance and time it would take for the ball to travel that distance, and simulating the movement of the ball and the players over that time period.
1410. %-----Pass Ball-----%
1411. if dis\_A<=3
1413. if dis\_A==dis\_A2\_A1
1414. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A1\_pos(1),A1\_pos(2));
1415. A\_pos(1)=A1\_pos(1);
1416. A\_pos(2)=A1\_pos(2);
1417. else
1418. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A3\_pos(1),A3\_pos(2));
1419. A\_pos(1)=A3\_pos(1);
1420. A\_pos(2)=A3\_pos(2);
1421. end
1423. t\_A=dis\_A/v\_Ball;
1424. num\_A=ceil(t\_A);
1426. for i=1:num\_A
1427. if Ball\_right==2
1428. if i<num\_A
1429. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A2);
1430. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A2);
1431. else
1432. Ball\_pos(1)=A\_pos(1);
1433. Ball\_pos(2)=A\_pos(2);
1434. end
1436. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1437. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1438. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1439. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1440. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1441. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1443. if dis\_B1\_Ball>v\_B1
1444. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1445. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1446. else
1447. B1\_pos(1)=Ball\_pos(1);
1448. B1\_pos(2)=Ball\_pos(2);
1449. Ball\_right=5;
1450. end
1452. if dis\_B2\_Ball>v\_B2
1453. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1454. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1455. else
1456. B2\_pos(1)=Ball\_pos(1);
1457. B2\_pos(2)=Ball\_pos(2);
1458. Ball\_right=6;
1459. end
1461. if dis\_B3\_Ball>v\_B3
1462. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1463. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1464. else
1465. B3\_pos(1)=Ball\_pos(1);
1466. B3\_pos(2)=Ball\_pos(2);
1467. Ball\_right=7;
1468. end
1470. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1471. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1472. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1473. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1474. % If the distance between the ball and A1 or A3 is zero, that means A1 or A3 is dribbling, then the program updates the Ball\_right value to reflect that the ball is with the corresponding player.
1475. if dis\_A1\_Ball==0
1476. Ball\_right=1;
1477. end
1479. if dis\_A3\_Ball==0
1480. Ball\_right=3;
1481. end
1483. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
1484. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
1485. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
1486. pause(0.2);
1487. end
1488. end
1489. end
1490. %-------------------%
1491. %Simulates the movement of the ball and players for each step, including updating the positions of the ball and players based on their current positions, velocities, and angles. It also checks for collisions between the ball and players and adjusts their positions accordingly.
1492. %-----Kick Ball-----%
1493. if Ball\_right==2
1494. t\_A2\_Goal1=dis\_A2\_Goal1/v\_Ball;
1495. num\_A2\_Goal1=ceil(t\_A2\_Goal1);
1497. for i=1:num\_A2\_Goal1
1498. if Ball\_right==2
1499. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A2);
1500. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A2);
1501. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1502. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1504. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1505. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1506. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1507. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1508. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1509. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1511. %Check if the defenders can get the ball, once one defender get the ball, the Ball\_right value would be changed.
1512. if dis\_B1\_Ball>v\_B1
1513. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1514. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1515. else
1516. B1\_pos(1)=Ball\_pos(1);
1517. B1\_pos(2)=Ball\_pos(2);
1518. Ball\_right=5;
1519. end
1521. if dis\_B2\_Ball>v\_B2
1522. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1523. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1524. else
1525. B2\_pos(1)=Ball\_pos(1);
1526. B2\_pos(2)=Ball\_pos(2);
1527. Ball\_right=6;
1528. end
1530. if dis\_B3\_Ball>v\_B3
1531. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1532. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1533. else
1534. B3\_pos(1)=Ball\_pos(1);
1535. B3\_pos(2)=Ball\_pos(2);
1536. Ball\_right=7;
1537. end
1538. %checks if the ball has reached the goal position and if so, updates the score. If goalkeeper’s position equal the ball position, then it means goalkeeper make a successful defense and the Ball\_right value changes.
1539. if Goal1\_pos(2)==Goal1\_t\_pos(2)
1540. a=1;
1541. else
1542. a=-1;
1543. end
1545. B4\_pos(1)=B4\_pos(1);
1546. B4\_pos(2)=B4\_pos(2)+a\*v\_B4;
1548. if B4\_pos(2) >= Goal1\_t\_pos(2)
1549. B4\_pos(2)=Goal1\_t\_pos(2);
1550. end
1552. if B4\_pos(2) <= Goal1\_b\_pos(2)
1553. B4\_pos(2)=Goal1\_b\_pos(2);
1554. end
1556. if Ball\_pos(1)>Goal1\_pos(1)
1557. Ball\_pos(1)=Goal1\_pos(1);
1558. Ball\_pos(2)=Goal1\_pos(2);
1559. end
1561. if Ball\_pos(1)==B4\_pos(1) && Ball\_pos(2)==B4\_pos(2)
1562. Ball\_right=8;
1563. end
1565. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
1566. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
1567. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
1568. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
1569. pause(0.2);
1570. end
1571. end
1572. pause(1);
1573. end
1574. %-------------------%
1576. end
1578. if Ball\_right==2
1579. det\_dis\_A2\_Goal1=dis\_A2\_Goal1-frame\*v\_Ball;
1580. t\_det=det\_dis\_A2\_Goal1/v\_A2;
1581. num\_det=ceil(t\_det);
1583. %-----move with ball-----%
1584. for i=1:num\_det
1585. if Ball\_right==2
1586. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
1587. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
1588. Ball\_pos(1)=A2\_pos(1);
1589. Ball\_pos(2)=A2\_pos(2);
1591. %-----A1/A3-----%
1592. if Goal1\_pos(2)==Goal1\_t\_pos(2)
1593. A\_Attack\_pos=[A2\_pos(1),A2\_pos(2)-2];
1594. end
1595. if Goal1\_pos(2)==Goal1\_b\_pos(2)
1596. A\_Attack\_pos=[A2\_pos(1),A2\_pos(2)+2];
1597. end
1598. dis\_A1\_Attack=getDis(A1\_pos(1),A1\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1599. dis\_A3\_Attack=getDis(A3\_pos(1),A3\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1600. %---------------%
1602. %-----A1 attack-----%
1603. if dis\_A1\_Attack<=dis\_A3\_Attack
1604. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1605. A3\_x=A2\_pos(1)-sqrt(3^2-(A2\_pos(2)-4)^2);
1606. dis\_A3\_Attack\_new=getDis(A3\_pos(1),A3\_pos(2),A3\_x,4);
1607. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A3\_x,4);
1609. if dis\_A1\_Attack>v\_A1
1610. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
1611. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
1612. else
1613. A1\_pos(1)=A\_Attack\_pos(1);
1614. A1\_pos(2)=A\_Attack\_pos(2);
1615. end
1617. if dis\_A3\_Attack\_new>v\_A3
1618. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
1619. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
1620. else
1621. A3\_pos(1)=A3\_x;
1622. A3\_pos(2)=4;
1623. end
1624. end
1625. %-------------------%
1627. %-----A3 attack-----%
1628. if dis\_A3\_Attack<dis\_A1\_Attack
1629. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1630. A1\_x=A2\_pos(1)-sqrt(3^2-(A2\_pos(2)-4)^2);
1631. dis\_A1\_Attack\_new=getDis(A1\_pos(1),A1\_pos(2),A1\_x,4);
1632. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A1\_x,4);
1634. if dis\_A3\_Attack>v\_A3
1635. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
1636. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
1637. else
1638. A3\_pos(1)=A\_Attack\_pos(1);
1639. A3\_pos(2)=A\_Attack\_pos(2);
1640. end
1642. if dis\_A1\_Attack\_new>v\_A1
1643. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
1644. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
1645. else
1646. A1\_pos(1)=A1\_x;
1647. A1\_pos(2)=4;
1648. end
1650. end
1651. %-------------------%
1653. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1654. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1655. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1656. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1657. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1658. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1660. if dis\_B1\_Ball>v\_B1
1661. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1662. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1663. else
1664. B1\_pos(1)=Ball\_pos(1);
1665. B1\_pos(2)=Ball\_pos(2);
1666. theta\_B1=theta\_A2;
1667. if rand(1)>=0.3
1668. Ball\_right=5;
1669. end
1670. end
1672. if dis\_B2\_Ball>v\_B2
1673. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1674. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1675. else
1676. B2\_pos(1)=Ball\_pos(1);
1677. B2\_pos(2)=Ball\_pos(2);
1678. theta\_B2=theta\_A2;
1679. if rand(1)>=0.3
1680. Ball\_right=6;
1681. end
1682. end
1684. if dis\_B3\_Ball>v\_B3
1685. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1686. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1687. else
1688. B3\_pos(1)=Ball\_pos(1);
1689. B3\_pos(2)=Ball\_pos(2);
1690. theta\_B3=theta\_A2;
1691. if rand(1)>=0.3
1692. Ball\_right=7;
1693. end
1694. end
1696. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
1697. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,
1698. theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
1699. pause(0.2);
1701. end
1702. end
1703. %------------------------%
1704. end
1705. end
1707. if Ball\_right==2
1708. dis\_A2\_Goal1=getDis(A2\_pos(1),A2\_pos(2),Goal1\_pos(1),Goal1\_pos(2));
1709. t\_A2\_Goal1=dis\_A2\_Goal1/v\_Ball;
1710. num\_A2\_Goal1=ceil(t\_A2\_Goal1);
1712. %-----Kick Ball-----%
1713. for i=1:num\_A2\_Goal1
1714. if Ball\_right==2
1715. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A2);
1716. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A2);
1717. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1718. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1719. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1720. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1721. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1722. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1723. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1724. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1726. if dis\_B1\_Ball>v\_B1
1727. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1728. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1729. else
1730. B1\_pos(1)=Ball\_pos(1);
1731. B1\_pos(2)=Ball\_pos(2);
1732. Ball\_right=5;
1733. end
1735. if dis\_B2\_Ball>v\_B2
1736. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1737. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1738. else
1739. B2\_pos(1)=Ball\_pos(1);
1740. B2\_pos(2)=Ball\_pos(2);
1741. Ball\_right=6;
1742. end
1744. if dis\_B3\_Ball>v\_B3
1745. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1746. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1747. else
1748. B3\_pos(1)=Ball\_pos(1);
1749. B3\_pos(2)=Ball\_pos(2);
1750. Ball\_right=7;
1751. end
1753. if Goal1\_pos(2)==Goal1\_t\_pos(2)
1754. a=1;
1755. else
1756. a=-1;
1757. end
1759. B4\_pos(1)=B4\_pos(1);
1760. B4\_pos(2)=B4\_pos(2)+a\*v\_B4;
1762. if Ball\_pos(1)>Goal1\_pos(1)
1763. Ball\_pos(1)=Goal1\_pos(1);
1764. Ball\_pos(2)=Goal1\_pos(2);
1765. end
1767. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
1768. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
1769. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,
1770. B3\_pos,theta\_B3,B4\_pos,theta\_B4);
1771. pause(0.2);
1772. end
1773. end
1774. %-------------------%
1776. Ball\_right=8;
1777. if Ball\_pos(2)~=B4\_pos(2)
1778. score\_A=score\_A+1;
1779. end
1780. pause(1);
1781. end
1782. end
1783. end
1784. %When A3 gets the ball and A team will be attacking, A3 will do the similar way to deal with the ball. When A3 is dribbling and facing the defenders, it will choose one teammate who is the closest and pass the ball. If A3 is in in the shooting range, it will choose to kick ball. If defender or goalkeeper from B team gets the ball, the Ball\_right value will be changed and the B team begin to attack.
1785. while (Ball\_right==3)
1786. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1787. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1788. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1789. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1790. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1791. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1792. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1793. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1794. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1795. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1796. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1797. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1798. %Calculate the angle of one defender who is the closest to the ball, and then let A3’s angle turn 90 degrees from the closest defender to make A3 can get rid of the defense easily. Considering the attacker can turn right or turn left to get rid of the defender, the attacker will choose one way which is closer to the goal after it run away from the defender.
1799. if dis\_B1\_Ball <= dis\_B2\_Ball && dis\_B1\_Ball <= dis\_B3\_Ball
1800. theta\_B=theta\_B1;
1801. end
1802. if dis\_B2\_Ball < dis\_B1\_Ball && dis\_B2\_Ball <= dis\_B3\_Ball
1803. theta\_B=theta\_B2;
1804. end
1805. if dis\_B3\_Ball < dis\_B1\_Ball && dis\_B3\_Ball < dis\_B2\_Ball
1806. theta\_B=theta\_B3;
1807. end
1808. %Make A3/attacker turn and get rid of defender and choose a better way to the goal. If the distances of two ways to goal are equal, then there are 50% chance for the attack to choose one way to go.
1809. theta\_A3\_next1=theta\_B+pi/2;
1810. delta\_x\_A3\_next1=v\_A3\*cos(theta\_A3\_next1);
1811. delta\_y\_A3\_next1=v\_A3\*sin(theta\_A3\_next1);
1812. A3\_posnew\_1(1)=A3\_pos(1)-delta\_x\_A3\_next1;
1813. A3\_posnew\_1(2)=A3\_pos(2)-delta\_y\_A3\_next1;
1814. theta\_A3\_next2=theta\_B-pi/2;
1815. delta\_x\_A3\_next2=v\_A3\*cos(theta\_A3\_next2);
1816. delta\_y\_A3\_next2=v\_A3\*sin(theta\_A3\_next2);
1817. A3\_posnew\_2(1)=A3\_pos(1)-delta\_x\_A3\_next2;
1818. A3\_posnew\_2(2)=A3\_pos(2)-delta\_y\_A3\_next2;
1819. dis\_A3\_next1=getDis(A3\_posnew\_1(1),A3\_posnew\_1(2),Goal1\_pos\_abs(1),Goal1\_pos\_abs(2));
1820. dis\_A3\_next2=getDis(A3\_posnew\_2(1),A3\_posnew\_2(2),Goal1\_pos\_abs(1),Goal1\_pos\_abs(2));
1821. if dis\_A3\_next1<dis\_A3\_next2
1822. theta\_A3=theta\_A3\_next1;
1823. end
1824. if dis\_A3\_next1>dis\_A3\_next2
1825. theta\_A3=theta\_A3\_next2;
1826. end
1827. if dis\_A3\_next1==dis\_A3\_next2
1828. if rand(1)>=0.5
1829. theta\_A3=theta\_A3\_next1;
1830. else
1831. theta\_A3=theta\_A3\_next2;
1832. end
1833. end
1834. delta\_x\_A3=v\_A3\*cos(theta\_A3);
1835. delta\_y\_A3=v\_A3\*sin(theta\_A3);
1836. A3\_posnew(1)=A3\_pos(1)-delta\_x\_A3;
1837. A3\_posnew(2)=A3\_pos(2)-delta\_y\_A3;
1838. if A3\_posnew(2)>=7
1839. A3\_pos(1)=A3\_posnew(1);
1840. A3\_pos(2)=7;
1841. else
1842. if A3\_posnew(2)<=1
1843. A3\_pos(1)=A3\_posnew(1);
1844. A3\_pos(2)=1;
1845. else
1846. A3\_pos(1)=A3\_posnew(1);
1847. A3\_pos(2)=A3\_posnew(2);
1848. end
1849. end
1850. Ball\_pos(1)=A3\_pos(1);
1851. Ball\_pos(2)=A3\_pos(2);
1852. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,
1853. A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,
1854. B4\_pos,theta\_B4);
1855. pause(0.2);
1856. %When A3 is attacking, it will go to the goal depend which side of the goal is closer (to the top or to the bottom). If two way distances is equal, there will be a 50% chance for A3 to go.
1857. %-----A3 to t or b-----%
1858. dis\_A3\_Goal1\_t=getDis(A3\_pos(1),A3\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
1859. t\_A3\_t=dis\_A3\_Goal1\_t/v\_A3;
1860. num\_A3\_t=ceil(t\_A3\_t);
1861. dis\_A3\_Goal1\_b=getDis(A3\_pos(1),A3\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
1862. t\_A3\_b=dis\_A3\_Goal1\_b/v\_A3;
1863. num\_A3\_b=ceil(t\_A3\_b);
1864. if num\_A3\_t < num\_A3\_b
1865. Goal1\_pos=Goal1\_t\_pos;
1866. theta\_A3= getTheta(A3\_pos(1),A3\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
1867. dis\_A3\_Goal1=dis\_A3\_Goal1\_t;
1868. end
1869. if num\_A3\_t > num\_A3\_b
1870. Goal1\_pos=Goal1\_b\_pos;
1871. theta\_A3= getTheta(A3\_pos(1),A3\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
1872. dis\_A3\_Goal1=dis\_A3\_Goal1\_b;
1873. end
1874. if num\_A3\_t==num\_A3\_b
1875. if rand(1)>=0.5
1876. Goal1\_pos=Goal1\_t\_pos;
1877. theta\_A3= getTheta(A3\_pos(1),A3\_pos(2),Goal1\_t\_pos(1),Goal1\_t\_pos(2));
1878. dis\_A3\_Goal1=dis\_A3\_Goal1\_t;
1879. else
1880. Goal1\_pos=Goal1\_b\_pos;
1881. theta\_A3= getTheta(A3\_pos(1),A3\_pos(2),Goal1\_b\_pos(1),Goal1\_b\_pos(2));
1882. dis\_A3\_Goal1=dis\_A3\_Goal1\_b;
1883. end
1884. end
1885. %----------------------%
1886. delta\_dis=dis\_A3\_Goal1-3;
1888. if delta\_dis>0
1889. t\_A3=delta\_dis/v\_A3;
1890. num\_A3=ceil(t\_A3);
1892. %-----B1/B2/B3-----%
1893. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1894. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1895. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1896. %------------------%
1898. for i=1:num\_A3
1899. if Ball\_right==3
1900. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
1901. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
1902. Ball\_pos(1)=A3\_pos(1);
1903. Ball\_pos(2)=A3\_pos(2);
1905. %-----A1/A2-----%
1906. if Goal1\_pos(2)==Goal1\_t\_pos(2)
1907. A\_Attack\_pos=[A3\_pos(1),A3\_pos(2)-2];
1908. end
1909. if Goal1\_pos(2)==Goal1\_b\_pos(2)
1910. A\_Attack\_pos=[A3\_pos(1),A3\_pos(2)+2];
1911. end
1912. dis\_A1\_Attack=getDis(A1\_pos(1),A1\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1913. dis\_A2\_Attack=getDis(A2\_pos(1),A2\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1914. %---------------%
1916. %-----A1 attack-----%
1917. if dis\_A1\_Attack<=dis\_A2\_Attack
1918. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1919. A2\_x=A3\_pos(1)-sqrt(3^2-(A3\_pos(2)-4)^2);
1920. dis\_A2\_Attack\_new=getDis(A2\_pos(1),A2\_pos(2),A2\_x,4);
1921. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A2\_x,4);
1923. if dis\_A1\_Attack>v\_A1
1924. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
1925. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
1926. else
1927. A1\_pos(1)=A\_Attack\_pos(1);
1928. A1\_pos(2)=A\_Attack\_pos(2);
1929. end
1931. if dis\_A2\_Attack\_new>v\_A2
1932. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
1933. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
1934. else
1935. A2\_pos(1)=A2\_x;
1936. A2\_pos(2)=4;
1937. end
1938. end
1939. %-------------------%
1941. %-----A2 attack-----%
1942. if dis\_A2\_Attack<dis\_A1\_Attack
1943. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
1944. A1\_x=A3\_pos(1)-sqrt(3^2-(A3\_pos(2)-4)^2);
1945. dis\_A1\_Attack\_new=getDis(A1\_pos(1),A1\_pos(2),A1\_x,4);
1946. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A1\_x,4);
1948. if dis\_A2\_Attack>v\_A2
1949. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
1950. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
1951. else
1952. A2\_pos(1)=A\_Attack\_pos(1);
1953. A2\_pos(2)=A\_Attack\_pos(2);
1954. end
1956. if dis\_A1\_Attack\_new>v\_A1
1957. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
1958. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
1959. else
1960. A1\_pos(1)=A1\_x;
1961. A1\_pos(2)=4;
1962. end
1964. end
1965. %-------------------%
1967. %For the defenders in Team B, we calculate every distance between ball and defenders. Defenders are chasing the ball and once one of them reach the ball, there will be 70% chance for them to get the ball and change the Ball\_right value.
1968. %------B1 to Ball-----%
1969. if dis\_B1\_Ball<=dis\_B2\_Ball && dis\_B1\_Ball<=dis\_B3\_Ball
1970. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1971. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
1972. dis\_B2\_Defend=getDis(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
1973. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
1974. dis\_B3\_Defend=getDis(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
1975. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
1977. if dis\_B2\_Defend>v\_B2
1978. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
1979. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
1980. else
1981. B2\_pos(1)=defind\_pos\_1(1);
1982. B2\_pos(2)=defind\_pos\_1(2);
1983. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
1984. end
1986. if dis\_B3\_Defend>v\_B3
1987. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
1988. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
1989. else
1990. B3\_pos(1)=defind\_pos\_2(1);
1991. B3\_pos(2)=defind\_pos\_2(2);
1992. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
1993. end
1995. if dis\_B1\_Ball>v\_B1
1996. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
1997. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
1998. else
1999. B1\_pos(1)=Ball\_pos(1);
2000. B1\_pos(2)=Ball\_pos(2);
2001. theta\_B1=theta\_A3;
2002. if rand(1)>=0.3
2003. Ball\_right=5;
2004. end
2005. end
2006. end
2007. %--------------------%
2008. %Same action like B1 does when B2 is the defender who are the closest one to the ball.
2009. %-----B2 to Ball-----%
2010. if dis\_B2\_Ball<dis\_B1\_Ball && dis\_B2\_Ball<=dis\_B3\_Ball
2011. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2012. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2013. dis\_B1\_Defend=getDis(B1\_pos(1),B1\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
2014. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
2015. dis\_B3\_Defend=getDis(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
2016. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
2018. if dis\_B1\_Defend>v\_B1
2019. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
2020. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
2021. else
2022. B1\_pos(1)=defind\_pos\_1(1);
2023. B1\_pos(2)=defind\_pos\_1(2);
2024. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2025. end
2027. if dis\_B3\_Defend>v\_B3
2028. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2029. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2030. else
2031. B3\_pos(1)=defind\_pos\_2(1);
2032. B3\_pos(2)=defind\_pos\_2(2);
2033. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2034. end
2036. if dis\_B2\_Ball>v\_B2
2037. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2038. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2039. else
2040. B2\_pos(1)=Ball\_pos(1);
2041. B2\_pos(2)=Ball\_pos(2);
2042. theta\_B2=theta\_A3;
2043. if rand(1)>=0.3
2044. Ball\_right=6;
2045. end
2046. end
2047. end
2048. %--------------------%
2050. %Same action like B1 does when B3 is the defender who are the closest one to the ball.
2051. %-----B3 to Ball-----%
2052. if dis\_B3\_Ball<dis\_B1\_Ball && dis\_B3\_Ball<dis\_B2\_Ball
2053. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2054. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2055. dis\_B1\_Defend=getDis(B1\_pos(1),B1\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
2056. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),defind\_pos\_2(1),defind\_pos\_2(2));
2057. dis\_B2\_Defend=getDis(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
2058. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),defind\_pos\_1(1),defind\_pos\_1(2));
2059. if dis\_B1\_Defend>v\_B1
2060. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
2061. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
2062. else
2063. B1\_pos(1)=defind\_pos\_2(1);
2064. B1\_pos(2)=defind\_pos\_2(2);
2065. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2066. end
2068. if dis\_B2\_Defend>v\_B2
2069. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2070. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2071. else
2072. B2\_pos(1)=defind\_pos\_1(1);
2073. B2\_pos(2)=defind\_pos\_1(2);
2074. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2075. end
2077. if dis\_B3\_Ball>v\_B3
2078. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2079. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2080. else
2081. B3\_pos(1)=Ball\_pos(1);
2082. B3\_pos(2)=Ball\_pos(2);
2083. theta\_B3=theta\_A3;
2084. if rand(1)>=0.3
2085. Ball\_right=7;
2086. end
2087. end
2088. end
2089. %--------------------%
2090. end
2091. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
2092. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
2093. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
2094. pause(0.2);
2095. end
2096. end
2098. %If A3 finds that defender is close to it enough and it will decide to pass the ball. It will find a teammate who is closer to him and then pass the ball. If there are some defenders can stop the ball while the ball is moving, then Ball\_right is changed.
2099. if Ball\_right==3
2100. dis\_A3\_B1=getDis(A3\_pos(1),A3\_pos(2),B1\_pos(1),B1\_pos(2));
2101. dis\_A3\_B2=getDis(A3\_pos(1),A3\_pos(2),B2\_pos(1),B2\_pos(2));
2102. dis\_A3\_B3=getDis(A3\_pos(1),A3\_pos(2),B3\_pos(1),B3\_pos(2));
2103. dis\_A3\_A1=getDis(A3\_pos(1),A3\_pos(2),A1\_pos(1),A1\_pos(2));
2104. dis\_A3\_A2=getDis(A3\_pos(1),A3\_pos(2),A2\_pos(1),A2\_pos(2));
2105. dis\_A3\_Goal1=getDis(A3\_pos(1),A3\_pos(2),Goal1\_pos(1),Goal1\_pos(2));
2106. t\_A3\_Goal1=dis\_A3\_Goal1/v\_Ball;
2107. num\_A3\_Goal1=ceil(t\_A3\_Goal1);
2109. if num\_A3\_Goal1 > frame
2110. if dis\_A3\_B1<=v\_B1 || dis\_A3\_B2<=v\_B2 || dis\_A3\_B3<=v\_B3
2112. if A1\_pos(1)>=A2\_pos(1)
2113. dis\_A=dis\_A3\_A1;
2114. else
2115. dis\_A=dis\_A3\_A2;
2116. end
2118. %-----Pass Ball-----%
2119. if dis\_A<=3
2120. if dis\_A==dis\_A3\_A1
2121. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A1\_pos(1),A1\_pos(2));
2122. A\_pos(1)=A1\_pos(1);
2123. A\_pos(2)=A1\_pos(2);
2124. else
2125. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),A2\_pos(1),A2\_pos(2));
2126. A\_pos(1)=A2\_pos(1);
2127. A\_pos(2)=A2\_pos(2);
2128. end
2130. t\_A=dis\_A/v\_Ball;
2131. num\_A=ceil(t\_A);
2133. for i=1:num\_A
2134. if Ball\_right==3
2135. if i<num\_A
2136. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A3);
2137. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A3);
2138. else
2139. Ball\_pos(1)=A\_pos(1);
2140. Ball\_pos(2)=A\_pos(2);
2141. end
2143. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2144. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2145. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2146. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2147. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2148. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2150. if dis\_B1\_Ball>v\_B1
2151. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
2152. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
2153. else
2154. B1\_pos(1)=Ball\_pos(1);
2155. B1\_pos(2)=Ball\_pos(2);
2156. Ball\_right=5;
2157. end
2159. if dis\_B2\_Ball>v\_B2
2160. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2161. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2162. else
2163. B2\_pos(1)=Ball\_pos(1);
2164. B2\_pos(2)=Ball\_pos(2);
2165. Ball\_right=6;
2166. end
2168. if dis\_B3\_Ball>v\_B3
2169. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2170. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2171. else
2172. B3\_pos(1)=Ball\_pos(1);
2173. B3\_pos(2)=Ball\_pos(2);
2174. Ball\_right=7;
2175. end
2177. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2178. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2179. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2180. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2182. if dis\_A1\_Ball==0
2183. Ball\_right=1;
2184. end
2186. if dis\_A2\_Ball==0
2187. Ball\_right=2;
2188. end
2189. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
2190. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
2191. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
2192. pause(0.2);
2193. end
2194. end
2195. end
2196. %-------------------%
2198. %-----Kick Ball-----%
2199. if Ball\_right==3
2200. t\_A3\_Goal1=dis\_A3\_Goal1/v\_Ball;
2201. num\_A3\_Goal1=ceil(t\_A3\_Goal1);
2203. for i=1:num\_A3\_Goal1
2204. if Ball\_right==3
2205. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A3);
2206. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A3);
2207. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2208. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2210. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2211. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2212. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2213. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2214. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2215. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2217. if dis\_B1\_Ball>v\_B1
2218. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
2219. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
2220. else
2221. B1\_pos(1)=Ball\_pos(1);
2222. B1\_pos(2)=Ball\_pos(2);
2223. Ball\_right=5;
2224. end
2226. if dis\_B2\_Ball>v\_B2
2227. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2228. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2229. else
2230. B2\_pos(1)=Ball\_pos(1);
2231. B2\_pos(2)=Ball\_pos(2);
2232. Ball\_right=6;
2233. end
2235. if dis\_B3\_Ball>v\_B3
2236. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2237. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2238. else
2239. B3\_pos(1)=Ball\_pos(1);
2240. B3\_pos(2)=Ball\_pos(2);
2241. Ball\_right=7;
2242. end
2244. if Goal1\_pos(2)==Goal1\_t\_pos(2)
2245. a=1;
2246. else
2247. a=-1;
2248. end
2250. B4\_pos(1)=B4\_pos(1);
2251. B4\_pos(2)=B4\_pos(2)+a\*v\_B4;
2253. if B4\_pos(2) >= Goal1\_t\_pos(2)
2254. B4\_pos(2)=Goal1\_t\_pos(2);
2255. end
2257. if B4\_pos(2) <= Goal1\_b\_pos(2)
2258. B4\_pos(2)=Goal1\_b\_pos(2);
2259. end
2261. if Ball\_pos(1)>Goal1\_pos(1)
2262. Ball\_pos(1)=Goal1\_pos(1);
2263. Ball\_pos(2)=Goal1\_pos(2);
2264. end
2266. if Ball\_pos(1)==B4\_pos(1) && Ball\_pos(2)==B4\_pos(2)
2267. Ball\_right=8;
2268. end
2270. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
2271. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
2272. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
2273. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
2274. pause(0.2);
2275. end
2276. end
2277. pause(1);
2278. end
2279. %-------------------%
2280. end
2282. if Ball\_right==3
2283. det\_dis\_A3\_Goal1=dis\_A3\_Goal1-frame\*v\_Ball;
2284. t\_det=det\_dis\_A3\_Goal1/v\_A3;
2285. num\_det=ceil(t\_det);
2287. %-----move with ball-----%
2288. for i=1:num\_det
2289. if Ball\_right==3
2290. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
2291. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
2292. Ball\_pos(1)=A3\_pos(1);
2293. Ball\_pos(2)=A3\_pos(2);
2295. %-----A1/A2-----%
2296. if Goal1\_pos(2)==Goal1\_t\_pos(2)
2297. A\_Attack\_pos=[A3\_pos(1),A3\_pos(2)-2];
2298. end
2299. if Goal1\_pos(2)==Goal1\_b\_pos(2)
2300. A\_Attack\_pos=[A3\_pos(1),A3\_pos(2)+2];
2301. end
2302. dis\_A1\_Attack=getDis(A1\_pos(1),A1\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
2303. dis\_A2\_Attack=getDis(A2\_pos(1),A2\_pos(2),A\_Attack\_pos(1),A\_Attack\_pos(2));
2304. %---------------%
2306. %-----A1 attack-----%
2307. if dis\_A1\_Attack<=dis\_A3\_Attack
2308. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A\_Attack\_pos(1),
2309. A\_Attack\_pos(2));
2310. A2\_x=A3\_pos(1)-sqrt(3^2-(A3\_pos(2)-4)^2);
2311. dis\_A2\_Attack\_new=getDis(A2\_pos(1),A2\_pos(2),A2\_x,4);
2312. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A2\_x,4);
2314. if dis\_A1\_Attack>v\_A1
2315. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
2316. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
2317. else
2318. A1\_pos(1)=A\_Attack\_pos(1);
2319. A1\_pos(2)=A\_Attack\_pos(2);
2320. end
2322. if dis\_A2\_Attack\_new>v\_A2
2323. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
2324. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
2325. else
2326. A2\_pos(1)=A2\_x;
2327. A2\_pos(2)=4;
2328. end
2329. end
2330. %-------------------%
2332. %-----A2 attack-----%
2333. if dis\_A2\_Attack<dis\_A1\_Attack
2334. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),A\_Attack\_pos(1),
2335. A\_Attack\_pos(2));
2336. A1\_x=A3\_pos(1)-sqrt(3^2-(A3\_pos(2)-4)^2);
2337. dis\_A1\_Attack\_new=getDis(A1\_pos(1),A1\_pos(2),A1\_x,4);
2338. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),A1\_x,4);
2340. if dis\_A2\_Attack>v\_A2
2341. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
2342. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
2343. else
2344. A2\_pos(1)=A\_Attack\_pos(1);
2345. A2\_pos(2)=A\_Attack\_pos(2);
2346. end
2348. if dis\_A1\_Attack\_new>v\_A1
2349. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
2350. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
2351. else
2352. A1\_pos(1)=A1\_x;
2353. A1\_pos(2)=4;
2354. end
2356. end
2357. %-------------------%
2359. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2360. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2361. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2362. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2363. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2364. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2366. if dis\_B1\_Ball>v\_B1
2367. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
2368. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
2369. else
2370. B1\_pos(1)=Ball\_pos(1);
2371. B1\_pos(2)=Ball\_pos(2);
2372. theta\_B1=theta\_A3;
2373. if rand(1)>=0.3
2374. Ball\_right=5;
2375. end
2376. end
2378. if dis\_B2\_Ball>v\_B2
2379. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2380. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2381. else
2382. B2\_pos(1)=Ball\_pos(1);
2383. B2\_pos(2)=Ball\_pos(2);
2384. theta\_B2=theta\_A3;
2385. if rand(1)>=0.3
2386. Ball\_right=6;
2387. end
2388. end
2390. if dis\_B3\_Ball>v\_B3
2391. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2392. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2393. else
2394. B3\_pos(1)=Ball\_pos(1);
2395. B3\_pos(2)=Ball\_pos(2);
2396. theta\_B3=theta\_A3;
2397. if rand(1)>=0.3
2398. Ball\_right=7;
2399. end
2400. end
2401. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
2402. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
2403. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
2404. pause(0.2);
2406. end
2407. end
2408. %------------------------%
2409. end
2410. end
2412. if Ball\_right==3
2413. dis\_A3\_Goal1=getDis(A3\_pos(1),A3\_pos(2),Goal1\_pos(1),Goal1\_pos(2));
2414. t\_A3\_Goal1=dis\_A3\_Goal1/v\_Ball;
2415. num\_A3\_Goal1=ceil(t\_A3\_Goal1);
2417. %-----Kick Ball-----%
2418. for i=1:num\_A3\_Goal1
2419. if Ball\_right==3
2420. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_A3);
2421. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_A3);
2422. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2423. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2425. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2426. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2427. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2428. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2429. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2430. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2432. if dis\_B1\_Ball>v\_B1
2433. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
2434. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
2435. else
2436. B1\_pos(1)=Ball\_pos(1);
2437. B1\_pos(2)=Ball\_pos(2);
2438. Ball\_right=5;
2439. end
2441. if dis\_B2\_Ball>v\_B2
2442. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2443. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2444. else
2445. B2\_pos(1)=Ball\_pos(1);
2446. B2\_pos(2)=Ball\_pos(2);
2447. Ball\_right=6;
2448. end
2450. if dis\_B3\_Ball>v\_B3
2451. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2452. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2453. else
2454. B3\_pos(1)=Ball\_pos(1);
2455. B3\_pos(2)=Ball\_pos(2);
2456. Ball\_right=7;
2457. end
2459. if Goal1\_pos(2)==Goal1\_t\_pos(2)
2460. a=1;
2461. else
2462. a=-1;
2463. end
2465. B4\_pos(1)=B4\_pos(1);
2466. B4\_pos(2)=B4\_pos(2)+a\*v\_B4;
2468. if Ball\_pos(1)>Goal1\_pos(1)
2469. Ball\_pos(1)=Goal1\_pos(1);
2470. Ball\_pos(2)=Goal1\_pos(2);
2471. End
2472. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
2473. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
2474. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
2475. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
2476. pause(0.2);
2477. end
2478. end
2479. %-------------------%
2481. Ball\_right=8;
2482. if Ball\_pos(2)~=B4\_pos(2)
2483. score\_A=score\_A+1;
2484. end
2485. pause(1);
2486. end
2487. end
2488. end
2490. %When A team goalkeeper is holding the ball, A1/A2/A3 will be attackers and there will be three different line-ups for A team to change. And every line-up has equal chance for A team to choose.
2491. while (Ball\_right==4 && score\_A<3 && score\_B<3)
2492. B1\_pos=[7 4];
2493. B2\_pos=[7 6];
2494. B3\_pos=[7 2];
2495. B4\_pos=[10 4];
2496. A4\_pos=[1 4];
2497. Ball\_pos=[1 4];
2498. a=rand(1);
2500. if a<=0.33
2501. A1\_pos=[4 4];
2502. A2\_pos=[5 6];
2503. A3\_pos=[5 2];
2504. end
2506. if a>0.33 && a<0.66
2507. A1\_pos=[4 4];
2508. A2\_pos=[4 6];
2509. A3\_pos=[4 2];
2510. end
2512. if a>=0.66 && a<=1
2513. A1\_pos=[5 4];
2514. A2\_pos=[4 6];
2515. A3\_pos=[4 2];
2516. end
2518. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2519. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2520. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2521. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
2522. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2523. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2524. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2525. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
2526. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
2527. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,
2528. theta\_B3,B4\_pos,theta\_B4);
2529. pause(0.2);
2531. b=rand(1);
2532. if b<=0.33
2533. Aim\_pos=A1\_pos;
2534. end
2536. if b>0.33 && b<0.66
2537. Aim\_pos=A2\_pos;
2538. end
2540. if b>=0.66 && b<=1
2541. Aim\_pos=A3\_pos;
2542. end
2544. dis\_A4\_Aim=getDis(A4\_pos(1),A4\_pos(2),Aim\_pos(1),Aim\_pos(2));
2545. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Aim\_pos(1),Aim\_pos(2));
2546. v\_Aim=dis\_A4\_Aim/frame;
2547. dis\_B1\_Aim=getDis(B1\_pos(1),B1\_pos(2),Aim\_pos(1),Aim\_pos(2));
2548. dis\_B2\_Aim=getDis(B2\_pos(1),B2\_pos(2),Aim\_pos(1),Aim\_pos(2));
2549. dis\_B3\_Aim=getDis(B3\_pos(1),B3\_pos(2),Aim\_pos(1),Aim\_pos(2));
2551. for i=1:frame
2552. if Ball\_right==4
2553. Ball\_pos(1)=Ball\_pos(1)-v\_Aim\*cos(theta\_A4);
2554. Ball\_pos(2)=Ball\_pos(2)-v\_Aim\*sin(theta\_A4);
2555. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2556. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2557. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2558. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2559. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2560. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2561. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
2562. if i==frame
2563. if dis\_B1\_Aim<=dis\_B2\_Aim && dis\_B1\_Aim<=dis\_B3\_Aim
2564. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
2565. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
2566. end
2567. if dis\_B2\_Aim<dis\_B1\_Aim && dis\_B2\_Aim<=dis\_B3\_Aim
2568. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2569. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2570. end
2571. if dis\_B3\_Aim<dis\_B1\_Aim && dis\_B3\_Aim<dis\_B2\_Aim
2572. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2573. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2574. end
2575. end
2576. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
2577. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,
2578. theta\_B3,B4\_pos,theta\_B4);
2579. pause(0.2);
2580. end
2581. end
2583. %Goalkeeper in A team will pass the ball to the attackers, once the attackers get the ball and Ball\_right value will be changed.
2584. if Aim\_pos(2)== A1\_pos(2)
2585. Ball\_pos(1)=A1\_pos(1);
2586. Ball\_pos(2)=A1\_pos(2);
2587. Ball\_right=1;
2588. end
2589. if Aim\_pos(2)== A2\_pos(2)
2590. Ball\_pos(1)=A2\_pos(1);
2591. Ball\_pos(2)=A2\_pos(2);
2592. Ball\_right=2;
2593. end
2594. if Aim\_pos(2)== A3\_pos(2)
2595. Ball\_pos(1)=A3\_pos(1);
2596. Ball\_pos(2)=A3\_pos(2);
2597. Ball\_right=3;
2598. end
2599. end
2601. %When B1 gets the ball, B team will begin to attack and A team will become the defenders and goalkeeper. Similar to the situation when A team attacks to the B team’s goal.
2602. while (Ball\_right==5)
2603. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2604. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2605. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2606. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2607. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2608. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2609. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2610. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2611. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2612. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2613. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2614. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2616. if dis\_A1\_Ball <= dis\_A2\_Ball && dis\_A1\_Ball <= dis\_A3\_Ball
2617. theta\_A=theta\_A1;
2618. end
2619. if dis\_A2\_Ball < dis\_A1\_Ball && dis\_A2\_Ball <= dis\_A3\_Ball
2620. theta\_A=theta\_A2;
2621. end
2622. if dis\_A3\_Ball < dis\_A1\_Ball && dis\_A3\_Ball < dis\_A2\_Ball
2623. theta\_A=theta\_A3;
2624. end
2626. %Same way for the B team attacker to dribble and try to get rid of the defender while A team is trying to get the ball.
2627. theta\_B1\_next1=theta\_A+pi/2;
2628. delta\_x\_B1\_next1=v\_B1\*cos(theta\_B1\_next1);
2629. delta\_y\_B1\_next1=v\_B1\*sin(theta\_B1\_next1);
2630. B1\_posnew\_1(1)=B1\_pos(1)-delta\_x\_B1\_next1;
2631. B1\_posnew\_1(2)=B1\_pos(2)-delta\_y\_B1\_next1;
2632. theta\_B1\_next2=theta\_A-pi/2;
2633. delta\_x\_B1\_next2=v\_B1\*cos(theta\_B1\_next2);
2634. delta\_y\_B1\_next2=v\_B1\*sin(theta\_B1\_next2);
2635. B1\_posnew\_2(1)=B1\_pos(1)-delta\_x\_B1\_next2;
2636. B1\_posnew\_2(2)=B1\_pos(2)-delta\_y\_B1\_next2;
2637. dis\_B1\_next1=getDis(B1\_posnew\_1(1),B1\_posnew\_1(2),Goal2\_pos\_abs(1),Goal2\_pos\_abs(2));
2638. dis\_B1\_next2=getDis(B1\_posnew\_2(1),B1\_posnew\_2(2),Goal2\_pos\_abs(1),Goal2\_pos\_abs(2));
2639. if dis\_B1\_next1<dis\_B1\_next2
2640. theta\_B1=theta\_B1\_next1;
2641. end
2642. if dis\_B1\_next1>dis\_B1\_next2
2643. theta\_B1=theta\_B1\_next2;
2644. end
2645. if dis\_B1\_next1==dis\_B1\_next2
2646. if rand(1)>=0.5
2647. theta\_B1=theta\_B1\_next1;
2648. else
2649. theta\_B1=theta\_B1\_next2;
2650. end
2651. end
2652. delta\_x\_B1=v\_B1\*cos(theta\_B1);
2653. delta\_y\_B1=v\_B1\*sin(theta\_B1);
2654. B1\_posnew(1)=B1\_pos(1)-delta\_x\_B1;
2655. B1\_posnew(2)=B1\_pos(2)-delta\_y\_B1;
2657. if B1\_posnew(2)>=7
2658. B1\_pos(1)=B1\_posnew(1);
2659. B1\_pos(2)=7;
2660. else
2661. if B1\_posnew(2)<=1
2662. B1\_pos(1)=B1\_posnew(1);
2663. B1\_pos(2)=1;
2664. else
2665. B1\_pos(1)=B1\_posnew(1);
2666. B1\_pos(2)=B1\_posnew(2);
2667. end
2668. end
2669. Ball\_pos(1)=B1\_pos(1);
2670. Ball\_pos(2)=B1\_pos(2);
2671. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
2672. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,
2673. theta\_B3,B4\_pos,theta\_B4);
2674. pause(0.2);
2676. %Calculate the distances from top of the goal and from bottom of the goal to the B1, and choose the shorter way to go.
2677. %-----B1 to t or b-----%
2678. dis\_B1\_Goal2\_t=getDis(B1\_pos(1),B1\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
2679. t\_B1\_t=dis\_B1\_Goal2\_t/v\_B1;
2680. num\_B1\_t=ceil(t\_B1\_t);
2681. dis\_B1\_Goal2\_b=getDis(B1\_pos(1),B1\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
2682. t\_B1\_b=dis\_B1\_Goal2\_b/v\_B1;
2683. num\_B1\_b=ceil(t\_B1\_b);
2684. if num\_B1\_t < num\_B1\_b
2685. Goal2\_pos=Goal2\_t\_pos;
2686. theta\_B1= getTheta(B1\_pos(1),B1\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
2687. dis\_B1\_Goal2=dis\_B1\_Goal2\_t;
2688. end
2689. if num\_B1\_b < num\_B1\_t
2690. Goal2\_pos=Goal2\_b\_pos;
2691. theta\_B1= getTheta(B1\_pos(1),B1\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
2692. dis\_B1\_Goal2=dis\_B1\_Goal2\_b;
2693. end
2694. if num\_B1\_t == num\_B1\_b
2695. if rand(1)>=0.5
2696. Goal2\_pos=Goal2\_t\_pos;
2697. theta\_B1= getTheta(B1\_pos(1),B1\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
2698. dis\_B1\_Goal2=dis\_B1\_Goal2\_t;
2699. else
2700. Goal2\_pos=Goal2\_b\_pos;
2701. theta\_B1= getTheta(B1\_pos(1),B1\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
2702. dis\_B1\_Goal2=dis\_B1\_Goal2\_b;
2703. end
2704. end
2705. %----------------------%
2706. %The code first calculates the distance (delta\_dis) between B1 and Goal2 and if it is positive, calculates the time (t\_B1) B1 will take to reach Goal2, and the number of movements (num\_B1) it will take to reach there.
2707. delta\_dis=dis\_B1\_Goal2-3;
2709. if delta\_dis>0
2710. t\_B1=delta\_dis/v\_B1;
2711. num\_B1=ceil(t\_B1);
2713. %-----A1/A2/A3-----%
2714. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2715. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2716. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2717. %------------------%
2719. for i=1:num\_B1
2720. if Ball\_right==5
2721. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
2722. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
2723. Ball\_pos(1)=B1\_pos(1);
2724. Ball\_pos(2)=B1\_pos(2);
2726. %-----B2/B3-----%
2727. if Goal2\_pos(2)==Goal2\_t\_pos(2)
2728. B\_Attack\_pos=[B1\_pos(1),B1\_pos(2)-2];
2729. end
2730. if Goal2\_pos(2)==Goal2\_b\_pos(2)
2731. B\_Attack\_pos=[B1\_pos(1),B1\_pos(2)+2];
2732. end
2733. dis\_B2\_Attack=getDis(B2\_pos(1),B2\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
2734. dis\_B3\_Attack=getDis(B3\_pos(1),B3\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
2735. %---------------%
2736. % consider two possible attack strategies for B2 and B3 depending on their distance from the attacking position and the ball's current position.
2737. %-----B2 attack-----%
2738. if dis\_B2\_Attack<=dis\_B3\_Attack
2739. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
2740. B3\_x=B1\_pos(1)+sqrt(3^2-(B1\_pos(2)-4)^2);
2741. dis\_B3\_Attack\_new=getDis(B3\_pos(1),B3\_pos(2),B3\_x,4);
2742. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B3\_x,4);
2744. if dis\_B2\_Attack>v\_B2
2745. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2746. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2747. else
2748. B2\_pos(1)=B\_Attack\_pos(1);
2749. B2\_pos(2)=B\_Attack\_pos(2);
2750. end
2752. if dis\_B3\_Attack\_new>v\_B3
2753. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2754. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2755. else
2756. B3\_pos(1)=B3\_x;
2757. B3\_pos(2)=4;
2758. end
2759. end
2760. %-------------------%
2762. %-----B3 attack-----%
2763. if dis\_B3\_Attack<dis\_B2\_Attack
2764. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
2765. B2\_x=B1\_pos(1)+sqrt(3^2-(B1\_pos(2)-4)^2);
2766. dis\_B2\_Attack\_new=getDis(B2\_pos(1),B2\_pos(2),B2\_x,4);
2767. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B2\_x,4);
2769. if dis\_B3\_Attack>v\_B3
2770. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
2771. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
2772. else
2773. B3\_pos(1)=B\_Attack\_pos(1);
2774. B3\_pos(2)=B\_Attack\_pos(2);
2775. end
2777. if dis\_B2\_Attack\_new>v\_B2
2778. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
2779. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
2780. else
2781. B2\_pos(1)=B2\_x;
2782. B2\_pos(2)=4;
2783. end
2784. end
2785. %-------------------%
2787. % check which player is closest to the ball and which player should defend against the ball carrier.
2788. %------A1 to Ball-----%
2789. if dis\_A1\_Ball<=dis\_A2\_Ball && dis\_A1\_Ball<=dis\_A3\_Ball
2790. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2791. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2792. dis\_A2\_Defend=getDis(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
2793. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
2794. dis\_A3\_Defend=getDis(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
2795. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
2797. if dis\_A2\_Defend>v\_A2
2798. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
2799. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
2800. else
2801. A2\_pos(1)=defind\_pos\_3(1);
2802. A2\_pos(2)=defind\_pos\_3(2);
2803. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2804. end
2806. if dis\_A3\_Defend>v\_A3
2807. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
2808. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
2809. else
2810. A3\_pos(1)=defind\_pos\_4(1);
2811. A3\_pos(2)=defind\_pos\_4(2);
2812. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),
2813. Ball\_pos(2));
2814. end
2816. if dis\_A1\_Ball>v\_A1
2817. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
2818. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
2819. else
2820. A1\_pos(1)=Ball\_pos(1);
2821. A1\_pos(2)=Ball\_pos(2);
2822. theta\_A1=theta\_B1;
2823. if rand(1)>=0.3
2824. Ball\_right=1;
2825. end
2826. end
2827. end
2828. %--------------------%
2830. %-----A2 to Ball-----%
2831. if dis\_A2\_Ball<dis\_A1\_Ball && dis\_A2\_Ball<=dis\_A3\_Ball
2832. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2833. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2834. dis\_A1\_Defend=getDis(A1\_pos(1),A1\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
2835. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
2836. dis\_A3\_Defend=getDis(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
2837. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
2839. if dis\_A1\_Defend>v\_A1
2840. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
2841. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
2842. else
2843. A1\_pos(1)=defind\_pos\_3(1);
2844. A1\_pos(2)=defind\_pos\_3(2);
2845. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2846. end
2848. if dis\_A3\_Defend>v\_A3
2849. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
2850. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
2851. else
2852. A3\_pos(1)=defind\_pos\_4(1);
2853. A3\_pos(2)=defind\_pos\_4(2);
2854. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2855. end
2857. if dis\_A2\_Ball>v\_A2
2858. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
2859. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
2860. else
2861. A2\_pos(1)=Ball\_pos(1);
2862. A2\_pos(2)=Ball\_pos(2);
2863. theta\_A2=theta\_B1;
2864. if rand(1)>=0.3
2865. Ball\_right=2;
2866. end
2867. end
2868. end
2869. %--------------------%
2871. %-----A3 to Ball-----%
2872. if dis\_A3\_Ball<dis\_A1\_Ball && dis\_A3\_Ball<dis\_A2\_Ball
2873. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2874. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2875. dis\_A1\_Defend=getDis(A1\_pos(1),A1\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
2876. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
2877. dis\_A2\_Defend=getDis(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
2878. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
2880. if dis\_A1\_Defend>v\_A1
2881. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
2882. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
2883. else
2884. A1\_pos(1)=defind\_pos\_4(1);
2885. A1\_pos(2)=defind\_pos\_4(2);
2886. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2887. end
2889. if dis\_A2\_Defend>v\_A2
2890. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
2891. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
2892. else
2893. A2\_pos(1)=defind\_pos\_3(1);
2894. A2\_pos(2)=defind\_pos\_3(2);
2895. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2896. end
2898. if dis\_A3\_Ball>v\_A3
2899. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
2900. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
2901. else
2902. A3\_pos(1)=Ball\_pos(1);
2903. A3\_pos(2)=Ball\_pos(2);
2904. theta\_A3=theta\_B1;
2905. if rand(1)>=0.3
2906. Ball\_right=3;
2907. end
2908. end
2909. end
2910. %--------------------%
2911. end
2912. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
2913. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,
2914. theta\_B3,B4\_pos,theta\_B4);
2915. pause(0.2);
2916. end
2917. end
2919. if Ball\_right==5
2920. % calculates the distance between the ball (B1\_pos) and various players and goals
2921. dis\_B1\_A1=getDis(B1\_pos(1),B1\_pos(2),A1\_pos(1),A1\_pos(2));
2922. dis\_B1\_A2=getDis(B1\_pos(1),B1\_pos(2),A2\_pos(1),A2\_pos(2));
2923. dis\_B1\_A3=getDis(B1\_pos(1),B1\_pos(2),A3\_pos(1),A3\_pos(2));
2924. dis\_B1\_B2=getDis(B1\_pos(1),B1\_pos(2),B2\_pos(1),B2\_pos(2));
2925. dis\_B1\_B3=getDis(B1\_pos(1),B1\_pos(2),B3\_pos(1),B3\_pos(2));
2926. dis\_B1\_Goal2=getDis(B1\_pos(1),B1\_pos(2),Goal2\_pos(1),Goal2\_pos(2));
2927. t\_B1\_Goal2=dis\_B1\_Goal2/v\_Ball;
2928. num\_B1\_Goal2=ceil(t\_B1\_Goal2);
2930. %If the time to reach Goal2\_pos (t\_B1\_Goal2) is greater than the frame number (frame), it checks if any of the opposing players A1\_pos, A2\_pos, A3\_pos are within v\_A1, v\_A2, and v\_A3 distance from the ball. If so, it determines which of the two opposing players B2\_pos and B3\_pos are closer to B1\_pos, and if they are within a distance of 3 units, it passes the ball to the closer player.
2931. if num\_B1\_Goal2 > frame
2932. if dis\_B1\_A1<=v\_A1 || dis\_B1\_A2<=v\_A2 || dis\_B1\_A3<=v\_A3
2933. if B2\_pos(1)<=B3\_pos(1)
2934. dis\_B=dis\_B1\_B2;
2935. else
2936. dis\_B=dis\_B1\_B3;
2937. end
2939. %-----Pass Ball-----%
2940. if dis\_B<=3
2941. if dis\_B==dis\_B1\_B2
2942. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),B2\_pos(1),B2\_pos(2));
2943. B\_pos(1)=B2\_pos(1);
2944. B\_pos(2)=B2\_pos(2);
2945. else
2946. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),B3\_pos(1),B3\_pos(2));
2947. B\_pos(1)=B3\_pos(1);
2948. B\_pos(2)=B3\_pos(2);
2949. end
2951. t\_B=dis\_B/v\_Ball;
2952. num\_B=ceil(t\_B);
2954. for i=1:num\_B
2955. if Ball\_right==5
2956. if i<num\_B
2957. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B1);
2958. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B1);
2959. else
2960. Ball\_pos(1)=B\_pos(1);
2961. Ball\_pos(2)=B\_pos(2);
2962. end
2964. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2965. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
2966. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2967. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2968. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2969. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
2971. if dis\_A1\_Ball>v\_A1
2972. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
2973. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
2974. else
2975. A1\_pos(1)=Ball\_pos(1);
2976. A1\_pos(2)=Ball\_pos(2);
2977. Ball\_right=1;
2978. end
2980. if dis\_A2\_Ball>v\_A2
2981. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
2982. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
2983. else
2984. A2\_pos(1)=Ball\_pos(1);
2985. A2\_pos(2)=Ball\_pos(2);
2986. Ball\_right=2;
2987. end
2989. if dis\_A3\_Ball>v\_A3
2990. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
2991. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
2992. else
2993. A3\_pos(1)=Ball\_pos(1);
2994. A3\_pos(2)=Ball\_pos(2);
2995. Ball\_right=3;
2996. end
2998. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
2999. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3000. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3001. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3002. % it checks if the ball has reached B2\_pos or B3\_pos and updates Ball\_right accordingly.
3003. if dis\_B2\_Ball==0
3004. Ball\_right=6;
3005. end
3007. if dis\_B3\_Ball==0
3008. Ball\_right=7;
3009. end
3010. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
3011. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
3012. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
3013. pause(0.2);
3014. end
3015. end
3016. end
3017. %-------------------%
3019. %-----Kick Ball-----%
3020. % it calculates the time t\_B1\_Goal2 it takes for the ball to reach Goal2 and the number of steps num\_B1\_Goal2 it will take to get there.
3021. if Ball\_right==5
3022. t\_B1\_Goal2=dis\_B1\_Goal2/v\_Ball;
3023. num\_B1\_Goal2=ceil(t\_B1\_Goal2);
3024. % Within the loop, it updates the positions of A1, A2, A3, and A4, based on their current positions, velocities, and angles with respect to the ball. The loop also updates the position of the ball based on its current position, velocity, and angle.
3025. for i=1:num\_B1\_Goal2
3026. if Ball\_right==5
3027. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B1);
3028. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B1);
3029. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3030. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3032. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3033. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3034. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3035. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3036. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3037. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3039. if dis\_A1\_Ball>v\_A1
3040. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3041. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3042. else
3043. A1\_pos(1)=Ball\_pos(1);
3044. A1\_pos(2)=Ball\_pos(2);
3045. Ball\_right=1;
3046. end
3048. if dis\_A2\_Ball>v\_A2
3049. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3050. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3051. else
3052. A2\_pos(1)=Ball\_pos(1);
3053. A2\_pos(2)=Ball\_pos(2);
3054. Ball\_right=2;
3055. end
3057. if dis\_A3\_Ball>v\_A3
3058. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3059. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3060. else
3061. A3\_pos(1)=Ball\_pos(1);
3062. A3\_pos(2)=Ball\_pos(2);
3063. Ball\_right=3;
3064. end
3066. if Goal2\_pos(2)==Goal2\_t\_pos(2)
3067. a=1;
3068. else
3069. a=-1;
3070. end
3072. A4\_pos(1)=A4\_pos(1);
3073. A4\_pos(2)=A4\_pos(2)+a\*v\_A4;
3075. if A4\_pos(2) >= Goal2\_t\_pos(2)
3076. A4\_pos(2)=Goal2\_t\_pos(2);
3077. end
3079. if A4\_pos(2) <= Goal2\_b\_pos(2)
3080. A4\_pos(2)=Goal2\_b\_pos(2);
3081. end
3083. if Ball\_pos(1)<Goal2\_pos(1)
3084. Ball\_pos(1)=Goal2\_pos(1);
3085. Ball\_pos(2)=Goal2\_pos(2);
3086. end
3088. if Ball\_pos(1)==A4\_pos(1) && Ball\_pos(2)==A4\_pos(2)
3089. Ball\_right=4;
3090. end
3092. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),
3093. Ball\_pos(2));
3094. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,
3095. theta\_A2,A3\_pos,theta\_A3,A4\_pos,theta\_A4,
3096. B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,
3097. B4\_pos,theta\_B4);
3098. pause(0.2);
3099. end
3100. end
3101. pause(1);
3102. end
3103. %-------------------%
3105. end
3107. if Ball\_right==5
3108. det\_dis\_B1\_Goal2=dis\_B1\_Goal2-frame\*v\_Ball;
3109. t\_det=det\_dis\_B1\_Goal2/v\_B1;
3110. num\_det=ceil(t\_det);
3112. %-----move with ball-----%
3113. % the position of B1 and the Ball are updated
3114. for i=1:num\_det
3115. if Ball\_right==5
3116. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
3117. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
3118. Ball\_pos(1)=B1\_pos(1);
3119. Ball\_pos(2)=B1\_pos(2);
3121. %-----B2/B3-----%
3122. if Goal2\_pos(2)==Goal2\_t\_pos(2)
3123. B\_Attack\_pos=[B1\_pos(1),B1\_pos(2)-2];
3124. end
3125. if Goal2\_pos(2)==Goal2\_b\_pos(2)
3126. B\_Attack\_pos=[B1\_pos(1),B1\_pos(2)+2];
3127. end
3128. dis\_B2\_Attack=getDis(B2\_pos(1),B2\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3129. dis\_B3\_Attack=getDis(B3\_pos(1),B3\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3130. %---------------%
3131. % The position of B2 and B3 are updated based on their distances to an attacking position (B\_Attack\_pos), where B2 moves towards the attacking position while B3 moves towards a position above the attacking position. If the distance between B2 and the attacking position is greater than the distance between B3 and the attacking position, B2 moves towards the attacking position, otherwise, B3 moves towards the above position.
3132. %-----B2 attack-----%
3133. if dis\_B2\_Attack<=dis\_B3\_Attack
3134. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B\_Attack\_pos(1),
3135. B\_Attack\_pos(2));
3136. B3\_x=B1\_pos(1)+sqrt(3^2-(B1\_pos(2)-4)^2);
3137. dis\_B3\_Attack\_new=getDis(B3\_pos(1),B3\_pos(2),B3\_x,4);
3138. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B3\_x,4);
3140. if dis\_B2\_Attack>v\_B2
3141. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
3142. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
3143. else
3144. B2\_pos(1)=B\_Attack\_pos(1);
3145. B2\_pos(2)=B\_Attack\_pos(2);
3146. end
3148. if dis\_B3\_Attack\_new>v\_B3
3149. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
3150. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
3151. else
3152. B3\_pos(1)=B3\_x;
3153. B3\_pos(2)=4;
3154. end
3155. end
3156. %-------------------%
3158. %-----B3 attack-----%
3159. if dis\_B3\_Attack<dis\_B2\_Attack
3160. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B\_Attack\_pos(1),
3161. B\_Attack\_pos(2));
3162. B2\_x=B1\_pos(1)+sqrt(3^2-(B1\_pos(2)-4)^2);
3163. dis\_B2\_Attack\_new=getDis(B2\_pos(1),B2\_pos(2),B2\_x,4);
3164. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B2\_x,4);
3166. if dis\_B3\_Attack>v\_B3
3167. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
3168. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
3169. else
3170. B3\_pos(1)=B\_Attack\_pos(1);
3171. B3\_pos(2)=B\_Attack\_pos(2);
3172. end
3174. if dis\_B2\_Attack\_new>v\_B2
3175. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
3176. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
3177. else
3178. B2\_pos(1)=B2\_x;
3179. B2\_pos(2)=4;
3180. end
3181. end
3182. %-------------------%
3183. %Update the positions and angles of team A and the ball.
3184. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3185. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3186. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3187. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3188. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3189. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3190. %Let the player move towards the ball or get the ball right. Depends on it’s position
3191. if dis\_A1\_Ball>v\_A1
3192. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3193. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3194. else
3195. A1\_pos(1)=Ball\_pos(1);
3196. A1\_pos(2)=Ball\_pos(2);
3197. theta\_A1=theta\_B1;
3198. if rand(1)>=0.3
3199. Ball\_right=1;
3200. end
3201. end
3203. if dis\_B2\_Ball>v\_B2
3204. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3205. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3206. else
3207. A2\_pos(1)=Ball\_pos(1);
3208. A2\_pos(2)=Ball\_pos(2);
3209. theta\_A2=theta\_B1;
3210. if rand(1)>=0.3
3211. Ball\_right=2;
3212. end
3213. end
3215. if dis\_A3\_Ball>v\_A3
3216. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3217. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3218. else
3219. A3\_pos(1)=Ball\_pos(1);
3220. A3\_pos(2)=Ball\_pos(2);
3221. theta\_A3=theta\_B1;
3222. if rand(1)>=0.3
3223. Ball\_right=3;
3224. end
3225. end
3226. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,
3227. theta\_A2,A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,
3228. theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
3229. pause(0.2);
3230. end
3231. end
3232. %------------------------%
3233. end
3234. end
3236. if Ball\_right==5
3237. dis\_B1\_Goal2=getDis(B1\_pos(1),B1\_pos(2),Goal2\_pos(1),Goal2\_pos(2));
3238. t\_B1\_Goal2=dis\_B1\_Goal2/v\_Ball;
3239. num\_B1\_Goal2=ceil(t\_B1\_Goal2);
3241. %-----Kick Ball-----%
3242. % The code updates the positions of each player and the ball based on their velocities, distances, and angles, and changes the state of the ball (Ball\_right) depending on its position in relation to the players and goals.
3243. for i=1:num\_B1\_Goal2
3244. if Ball\_right==5
3245. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B1);
3246. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B1);
3247. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3248. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3250. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3251. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3252. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3253. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3254. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3255. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3257. if dis\_A1\_Ball>v\_A1
3258. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3259. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3260. else
3261. A1\_pos(1)=Ball\_pos(1);
3262. A1\_pos(2)=Ball\_pos(2);
3263. Ball\_right=1;
3264. end
3266. if dis\_A2\_Ball>v\_A2
3267. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3268. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3269. else
3270. A2\_pos(1)=Ball\_pos(1);
3271. A2\_pos(2)=Ball\_pos(2);
3272. Ball\_right=2;
3273. end
3275. if dis\_A3\_Ball>v\_A3
3276. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3277. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3278. else
3279. A3\_pos(1)=Ball\_pos(1);
3280. A3\_pos(2)=Ball\_pos(2);
3281. Ball\_right=3;
3282. end
3284. if Goal2\_pos(2)==Goal2\_t\_pos(2)
3285. a=1;
3286. else
3287. a=-1;
3288. end
3290. A4\_pos(1)=A4\_pos(1);
3291. A4\_pos(2)=A4\_pos(2)+a\*v\_A4;
3292. %Check if the ball has reached the goal or if it has been intercepted by a player
3293. if A4\_pos(2) >= Goal2\_t\_pos(2)
3294. A4\_pos(2)=Goal2\_t\_pos(2);
3295. end
3297. if A4\_pos(2) <= Goal2\_b\_pos(2)
3298. A4\_pos(2)=Goal2\_b\_pos(2);
3299. end
3301. if Ball\_pos(1)<Goal2\_pos(1)
3302. Ball\_pos(1)=Goal2\_pos(1);
3303. Ball\_pos(2)=Goal2\_pos(2);
3304. end
3306. if Ball\_pos(1)==A4\_pos(1) && Ball\_pos(2)==A4\_pos(2)
3307. Ball\_right=4;
3308. end
3310. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
3311. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
3312. the ta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,
3313. B3\_pos,theta\_B3,B4\_pos,theta\_B4);
3314. pause(0.2);
3315. end
3316. end
3317. %-------------------%
3318. %Team B get the point
3319. Ball\_right=4;
3320. if Ball\_pos(2)~=A4\_pos(2)
3321. score\_B=score\_B+1;
3322. end
3323. pause(1);
3324. end
3325. end
3326. end
3328. % The code continuously updates the positions of the ball and the players based on their distances and angles to the ball.
3329. while (Ball\_right==6)
3330. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3331. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3332. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3333. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3334. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3335. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3336. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3337. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3338. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3339. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3340. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3341. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3343. if dis\_A1\_Ball <= dis\_A2\_Ball && dis\_A1\_Ball <= dis\_A3\_Ball
3344. theta\_A=theta\_A1;
3345. end
3346. if dis\_A2\_Ball < dis\_A1\_Ball && dis\_A2\_Ball <= dis\_A3\_Ball
3347. theta\_A=theta\_A2;
3348. end
3349. if dis\_A3\_Ball < dis\_A1\_Ball && dis\_A3\_Ball < dis\_A2\_Ball
3350. theta\_A=theta\_A3;
3351. end
3352. %calculates the next position for player B2 based on its current position and the position of the ball.
3353. theta\_B2\_next1=theta\_A+pi/2;
3354. delta\_x\_B2\_next1=v\_B2\*cos(theta\_B2\_next1);
3355. delta\_y\_B2\_next1=v\_B2\*sin(theta\_B2\_next1);
3356. B2\_posnew\_1(1)=B2\_pos(1)-delta\_x\_B2\_next1;
3357. B2\_posnew\_1(2)=B2\_pos(2)-delta\_y\_B2\_next1;
3358. theta\_B2\_next2=theta\_A-pi/2;
3359. delta\_x\_B2\_next2=v\_B2\*cos(theta\_B2\_next2);
3360. delta\_y\_B2\_next2=v\_B2\*sin(theta\_B2\_next2);
3361. B2\_posnew\_2(1)=B2\_pos(1)-delta\_x\_B2\_next2;
3362. B2\_posnew\_2(2)=B2\_pos(2)-delta\_y\_B2\_next2;
3363. dis\_B2\_next1=getDis(B2\_posnew\_1(1),B2\_posnew\_1(2),Goal2\_pos\_abs(1),Goal2\_pos\_abs(2));
3364. dis\_B2\_next2=getDis(B2\_posnew\_2(1),B2\_posnew\_2(2),Goal2\_pos\_abs(1),Goal2\_pos\_abs(2));
3365. % Depending on the distances from the calculated positions of player B2 to the goal, the code determines the direction for player B1 to move towards the goal.
3366. if dis\_B2\_next1<dis\_B2\_next2
3367. theta\_B1=theta\_B2\_next1;
3368. end
3369. if dis\_B2\_next1>dis\_B2\_next2
3370. theta\_B1=theta\_B2\_next2;
3371. end
3372. if dis\_B2\_next1==dis\_B2\_next2
3373. if rand(1)>=0.5
3374. theta\_B2=theta\_B2\_next1;
3375. else
3376. theta\_B2=theta\_B2\_next2;
3377. end
3378. end
3379. delta\_x\_B2=v\_B2\*cos(theta\_B2);
3380. delta\_y\_B2=v\_B2\*sin(theta\_B2);
3381. B2\_posnew(1)=B2\_pos(1)-delta\_x\_B2;
3382. B2\_posnew(2)=B2\_pos(2)-delta\_y\_B2;
3383. % ensures it does not go out of bounds, and updates the position of the ball.
3384. if B2\_posnew(2)>=7
3385. B2\_pos(1)=B2\_posnew(1);
3386. B2\_pos(2)=7;
3387. else
3388. if B2\_posnew(2)<=1
3389. B2\_pos(1)=B2\_posnew(1);
3390. B2\_pos(2)=2;
3391. else
3392. B2\_pos(1)=B2\_posnew(1);
3393. B2\_pos(2)=B2\_posnew(2);
3394. end
3395. end
3396. Ball\_pos(1)=B2\_pos(1);
3397. Ball\_pos(2)=B2\_pos(2);
3398. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
3399. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
3400. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
3401. pause(0.2);
3403. %-----B2 to t or b-----%
3404. %This code block calculates the distance and time required for an object B2 to reach one of two goals (Goal2\_t or Goal2\_b) and selects the closer goal.
3405. dis\_B2\_Goal2\_t=getDis(B2\_pos(1),B2\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
3406. t\_B2\_t=dis\_B2\_Goal2\_t/v\_B2;
3407. num\_B2\_t=ceil(t\_B2\_t);
3408. dis\_B2\_Goal2\_b=getDis(B2\_pos(1),B2\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
3409. t\_B2\_b=dis\_B2\_Goal2\_b/v\_B2;
3410. num\_B2\_b=ceil(t\_B2\_b);
3411. if num\_B2\_t < num\_B2\_b
3412. Goal2\_pos=Goal2\_t\_pos;
3413. theta\_B2= getTheta(B2\_pos(1),B2\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
3414. dis\_B2\_Goal2=dis\_B2\_Goal2\_t;
3415. end
3416. if num\_B2\_b < num\_B2\_t
3417. Goal2\_pos=Goal2\_b\_pos;
3418. theta\_B2= getTheta(B2\_pos(1),B2\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
3419. dis\_B2\_Goal2=dis\_B2\_Goal2\_b;
3420. end
3421. % If both goals require the same amount of time, a random selection is made.
3422. if num\_B2\_t == num\_B2\_b
3423. if rand(1)>=0.5
3424. Goal2\_pos=Goal2\_t\_pos;
3425. theta\_B2= getTheta(B2\_pos(1),B2\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
3426. dis\_B2\_Goal2=dis\_B2\_Goal2\_t;
3427. else
3428. Goal2\_pos=Goal2\_b\_pos;
3429. theta\_B2= getTheta(B2\_pos(1),B2\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
3430. dis\_B2\_Goal2=dis\_B2\_Goal2\_b;
3431. end
3432. end
3433. %----------------------%
3434. %If the remaining distance to the selected goal is greater than 3, the time required to travel the remaining distance (delta\_dis) is calculated, and the number of time steps required to travel that distance is determined (num\_B2)
3435. delta\_dis=dis\_B2\_Goal2-3;
3437. if delta\_dis>0
3438. t\_B2=delta\_dis/v\_B2;
3439. num\_B2=ceil(t\_B2);
3441. %-----A1/A2/A3-----%
3442. %the distances between the positions of the soccer ball and the three players A1, A2, and A3.
3443. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3444. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3445. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3446. %------------------%
3447. %Inside the loop, the position of player B2 and the soccer ball are updated. The positions of players B1 and B3 are also updated based on the position of B2, and their distances from the attacking position are calculated.
3448. for i=1:num\_B2
3449. if Ball\_right==6
3450. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
3451. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
3452. Ball\_pos(1)=B2\_pos(1);
3453. Ball\_pos(2)=B2\_pos(2);
3455. %-----B1/B3-----%
3456. if Goal2\_pos(2)==Goal2\_t\_pos(2)
3457. B\_Attack\_pos=[B2\_pos(1),B2\_pos(2)-2];
3458. end
3459. if Goal2\_pos(2)==Goal2\_b\_pos(2)
3460. B\_Attack\_pos=[B2\_pos(1),B2\_pos(2)+2];
3461. end
3462. dis\_B1\_Attack=getDis(B1\_pos(1),B1\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3463. dis\_B3\_Attack=getDis(B3\_pos(1),B3\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3464. %---------------%
3465. %If the distance of B1 from the attacking position is less than or equal to the distance of B3 from the attacking position, then the positions of B1 and B3 are updated according to their respective distances from the attacking position.
3466. %-----B1 attack-----%
3467. if dis\_B1\_Attack<=dis\_B3\_Attack
3468. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3469. B3\_x=B2\_pos(1)+sqrt(3^2-(B2\_pos(2)-4)^2);
3470. dis\_B3\_Attack\_new=getDis(B3\_pos(1),B3\_pos(2),B3\_x,4);
3471. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B3\_x,4);
3473. if dis\_B1\_Attack>v\_B1
3474. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
3475. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
3476. else
3477. B1\_pos(1)=B\_Attack\_pos(1);
3478. B1\_pos(2)=B\_Attack\_pos(2);
3479. end
3481. if dis\_B3\_Attack\_new>v\_B3
3482. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
3483. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
3484. else
3485. B3\_pos(1)=B3\_x;
3486. B3\_pos(2)=4;
3487. end
3489. end
3490. %-------------------%
3492. %-----B3 attack-----%
3493. if dis\_B3\_Attack<dis\_B2\_Attack
3494. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3495. B1\_x=B2\_pos(1)+sqrt(3^2-(B2\_pos(2)-4)^2);
3496. dis\_B1\_Attack\_new=getDis(B1\_pos(1),B1\_pos(2),B1\_x,4);
3497. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),B1\_x,4);
3499. if dis\_B3\_Attack>v\_B3
3500. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
3501. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
3502. else
3503. B3\_pos(1)=B\_Attack\_pos(1);
3504. B3\_pos(2)=B\_Attack\_pos(2);
3505. end
3507. if dis\_B1\_Attack\_new>v\_B1
3508. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
3509. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
3510. else
3511. B1\_pos(1)=B1\_x;
3512. B1\_pos(2)=4;
3513. end
3515. end
3516. %-------------------%
3518. %------A1 to Ball-----%
3519. %calculates the distances and angles between the players and the ball and between the second and third players and defenders.
3520. if dis\_A1\_Ball<=dis\_A2\_Ball && dis\_A1\_Ball<=dis\_A3\_Ball
3521. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3522. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3523. dis\_A2\_Defend=getDis(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
3524. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
3525. dis\_A3\_Defend=getDis(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
3526. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
3527. %The code then checks if the distance between each defender and its target position is larger than its speed. If it is, then it updates the defender's position accordingly. Otherwise, it sets the defender's position to its target position and recalculates the angle between the defender and the ball.
3528. if dis\_A2\_Defend>v\_A2
3529. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3530. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3531. else
3532. A2\_pos(1)=defind\_pos\_3(1);
3533. A2\_pos(2)=defind\_pos\_3(2);
3534. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3535. end
3537. if dis\_A3\_Defend>v\_A3
3538. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3539. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3540. else
3541. A3\_pos(1)=defind\_pos\_4(1);
3542. A3\_pos(2)=defind\_pos\_4(2);
3543. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3544. end
3546. if dis\_A1\_Ball>v\_A1
3547. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3548. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3549. else
3550. A1\_pos(1)=Ball\_pos(1);
3551. A1\_pos(2)=Ball\_pos(2);
3552. theta\_A1=theta\_B2;
3553. if rand(1)>=0.3
3554. Ball\_right=1;
3555. end
3556. end
3557. end
3558. %--------------------%
3559. %Similar with A1 to Ball
3560. %-----A2 to Ball-----%
3561. if dis\_A2\_Ball<dis\_A1\_Ball && dis\_A2\_Ball<=dis\_A3\_Ball
3562. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3563. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3564. dis\_A1\_Defend=getDis(A1\_pos(1),A1\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
3565. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
3566. dis\_A3\_Defend=getDis(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
3567. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
3569. if dis\_A1\_Defend>v\_A1
3570. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3571. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3572. else
3573. A1\_pos(1)=defind\_pos\_3(1);
3574. A1\_pos(2)=defind\_pos\_3(2);
3575. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3576. end
3578. if dis\_A3\_Defend>v\_A3
3579. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3580. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3581. else
3582. A3\_pos(1)=defind\_pos\_4(1);
3583. A3\_pos(2)=defind\_pos\_4(2);
3584. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3585. end
3587. if dis\_A2\_Ball>v\_A2
3588. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3589. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3590. else
3591. A2\_pos(1)=Ball\_pos(1);
3592. A2\_pos(2)=Ball\_pos(2);
3593. theta\_A2=theta\_B2;
3594. if rand(1)>=0.3
3595. Ball\_right=2;
3596. end
3597. end
3598. end
3599. %--------------------%
3600. %Similar with A1 to Ball
3601. %-----A3 to Ball-----%
3602. if dis\_A3\_Ball<dis\_A1\_Ball && dis\_A3\_Ball<dis\_A2\_Ball
3603. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3604. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3605. dis\_A1\_Defend=getDis(A1\_pos(1),A1\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
3606. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
3607. dis\_A2\_Defend=getDis(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
3608. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
3610. if dis\_A1\_Defend>v\_A1
3611. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3612. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3613. else
3614. A1\_pos(1)=defind\_pos\_4(1);
3615. A1\_pos(2)=defind\_pos\_4(2);
3616. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3617. end
3619. if dis\_A2\_Defend>v\_A2
3620. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3621. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3622. else
3623. A2\_pos(1)=defind\_pos\_3(1);
3624. A2\_pos(2)=defind\_pos\_3(2);
3625. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3626. end
3628. if dis\_A3\_Ball>v\_A3
3629. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3630. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3631. else
3632. A3\_pos(1)=Ball\_pos(1);
3633. A3\_pos(2)=Ball\_pos(2);
3634. theta\_A3=theta\_B2;
3635. if rand(1)>=0.3
3636. Ball\_right=3;
3637. end
3638. end
3640. end
3641. %--------------------%
3642. end
3643. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
3644. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
3645. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
3646. pause(0.2);
3647. end
3648. end
3650. % The code checks if Ball\_right equals 6, then calculates distances between different positions, calculates time for the ball to reach Goal2\_pos, and determines if the time is greater than frame.
3651. if Ball\_right==6
3652. dis\_B2\_A1=getDis(B2\_pos(1),B2\_pos(2),A1\_pos(1),A1\_pos(2));
3653. dis\_B2\_A2=getDis(B2\_pos(1),B2\_pos(2),A2\_pos(1),A2\_pos(2));
3654. dis\_B2\_A3=getDis(B2\_pos(1),B2\_pos(2),A3\_pos(1),A3\_pos(2));
3655. dis\_B2\_B1=getDis(B2\_pos(1),B2\_pos(2),B1\_pos(1),B1\_pos(2));
3656. dis\_B2\_B3=getDis(B2\_pos(1),B2\_pos(2),B3\_pos(1),B3\_pos(2));
3657. dis\_B2\_Goal2=getDis(B2\_pos(1),B2\_pos(2),Goal2\_pos(1),Goal2\_pos(2));
3658. t\_B2\_Goal2=dis\_B2\_Goal2/v\_Ball;
3659. num\_B2\_Goal2=ceil(t\_B2\_Goal2);
3660. % the code checks the position of B1\_pos relative to B3\_pos and sets dis\_B accordingly. If dis\_B is less than or equal to 3, the code passes the ball to either B1\_pos or B3\_pos based on the value of dis\_B.
3661. if num\_B2\_Goal2 > frame
3662. if dis\_B2\_A1<=v\_A1 || dis\_B2\_A2<=v\_A2 || dis\_B2\_A3<=v\_A3
3663. if B1\_pos(1)<=B3\_pos(1)
3664. dis\_B=dis\_B2\_B1;
3665. else
3666. dis\_B=dis\_B2\_B3;
3667. end
3669. %-----Pass Ball-----%
3670. if dis\_B<=3
3671. if dis\_B==dis\_B2\_B1
3672. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B1\_pos(1),B1\_pos(2));
3673. B\_pos(1)=B1\_pos(1);
3674. B\_pos(2)=B1\_pos(2);
3675. else
3676. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B3\_pos(1),B3\_pos(2));
3677. B\_pos(1)=B3\_pos(1);
3678. B\_pos(2)=B3\_pos(2);
3679. end
3681. t\_B=dis\_B/v\_Ball;
3682. num\_B=ceil(t\_B);
3683. % for each step between B\_pos and Ball\_pos, the code updates the position of the ball and the positions of the players A1\_pos, A2\_pos, A3\_pos, B1\_pos, and B3\_pos based on their respective distances and angles relative to the ball's position.
3684. for i=1:num\_B
3685. if Ball\_right==6
3686. if i<num\_B
3687. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B2);
3688. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B2);
3689. else
3690. Ball\_pos(1)=B\_pos(1);
3691. Ball\_pos(2)=B\_pos(2);
3692. end
3694. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3695. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3696. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3697. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3698. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3699. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3701. if dis\_A1\_Ball>v\_A1
3702. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3703. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3704. else
3705. A1\_pos(1)=Ball\_pos(1);
3706. A1\_pos(2)=Ball\_pos(2);
3707. Ball\_right=1;
3708. end
3710. if dis\_A2\_Ball>v\_A2
3711. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3712. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3713. else
3714. A2\_pos(1)=Ball\_pos(1);
3715. A2\_pos(2)=Ball\_pos(2);
3716. Ball\_right=2;
3717. end
3719. if dis\_A3\_Ball>v\_A3
3720. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3721. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3722. else
3723. A3\_pos(1)=Ball\_pos(1);
3724. A3\_pos(2)=Ball\_pos(2);
3725. Ball\_right=3;
3726. end
3728. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3729. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3730. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3731. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3732. %checks if the ball has reached B1\_pos or B3\_pos and updates the value of Ball\_right accordingly
3733. if dis\_B1\_Ball==0
3734. Ball\_right=5;
3735. end
3737. if dis\_B3\_Ball==0
3738. Ball\_right=7;
3739. end
3740. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
3741. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
3742. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
3743. pause(0.2);
3744. end
3745. end
3746. end
3747. %-------------------%
3748. %the code calculates the time it would take for the ball to reach the opponent's goal, and then uses a for loop to simulate the ball's movement towards the goal for that duration.
3749. %-----Kick Ball-----%
3750. if Ball\_right==6
3751. t\_B2\_Goal2=dis\_B2\_Goal2/v\_Ball;
3752. num\_B2\_Goal2=ceil(t\_B2\_Goal2);
3753. %During each iteration of the loop, the code updates the positions of the various players and the ball based on their current positions and velocities. The code also checks for collisions between the ball and the players and updates the ball's possession accordingly.
3754. for i=1:num\_B2\_Goal2
3755. if Ball\_right==6
3756. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B2);
3757. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B2);
3758. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3759. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3761. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3762. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3763. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3764. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3765. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3766. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3768. if dis\_A1\_Ball>v\_A1
3769. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3770. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3771. else
3772. A1\_pos(1)=Ball\_pos(1);
3773. A1\_pos(2)=Ball\_pos(2);
3774. Ball\_right=1;
3775. end
3777. if dis\_A2\_Ball>v\_A2
3778. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3779. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3780. else
3781. A2\_pos(1)=Ball\_pos(1);
3782. A2\_pos(2)=Ball\_pos(2);
3783. Ball\_right=2;
3784. end
3786. if dis\_A3\_Ball>v\_A3
3787. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3788. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3789. else
3790. A3\_pos(1)=Ball\_pos(1);
3791. A3\_pos(2)=Ball\_pos(2);
3792. Ball\_right=3;
3793. end
3795. if Goal2\_pos(2)==Goal2\_t\_pos(2)
3796. a=1;
3797. else
3798. a=-1;
3799. end
3801. A4\_pos(1)=A4\_pos(1);
3802. A4\_pos(2)=A4\_pos(2)+a\*v\_A4;
3804. if A4\_pos(2) >= Goal2\_t\_pos(2)
3805. A4\_pos(2)=Goal2\_t\_pos(2);
3806. end
3808. if A4\_pos(2) <= Goal2\_b\_pos(2)
3809. A4\_pos(2)=Goal2\_b\_pos(2);
3810. end
3812. if Ball\_pos(1)<Goal2\_pos(1)
3813. Ball\_pos(1)=Goal2\_pos(1);
3814. Ball\_pos(2)=Goal2\_pos(2);
3815. end
3817. if Ball\_pos(1)==A4\_pos(1) && Ball\_pos(2)==A4\_pos(2)
3818. Ball\_right=4;
3819. end
3821. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
3822. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
3823. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,
3824. B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
3825. pause(0.2);
3826. end
3827. end
3828. pause(1);
3829. end
3830. %-------------------%
3831. end
3832. % it calculates the time required for player B2 to reach the goal and moves the players accordingly using a for loop.
3833. if Ball\_right==6
3834. det\_dis\_B2\_Goal2=dis\_B2\_Goal2-frame\*v\_Ball;
3835. t\_det=det\_dis\_B2\_Goal2/v\_B1;
3836. num\_det=ceil(t\_det);
3838. %-----move with ball-----%
3839. for i=1:num\_det
3840. if Ball\_right==6
3841. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
3842. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
3843. Ball\_pos(1)=B2\_pos(1);
3844. Ball\_pos(2)=B2\_pos(2);
3846. %-----B1/B3-----%
3847. if Goal2\_pos(2)==Goal2\_t\_pos(2)
3848. B\_Attack\_pos=[B2\_pos(1),B2\_pos(2)-2];
3849. end
3850. if Goal2\_pos(2)==Goal2\_b\_pos(2)
3851. B\_Attack\_pos=[B2\_pos(1),B2\_pos(2)+2];
3852. end
3853. dis\_B1\_Attack=getDis(B1\_pos(1),B1\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3854. dis\_B3\_Attack=getDis(B3\_pos(1),B3\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3855. %---------------%
3856. % There are two different scenarios for the attack, depending on which player is closer to the attack position. The code calculates the distances between the players and the attack position, and moves them accordingly.
3857. %-----B1 attack-----%
3858. if dis\_B1\_Attack<=dis\_B3\_Attack
3859. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
3860. B3\_x=B2\_pos(1)+sqrt(3^2-(B2\_pos(2)-4)^2);
3861. dis\_B3\_Attack\_new=getDis(B3\_pos(1),B3\_pos(2),B3\_x,4);
3862. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B3\_x,4);
3864. if dis\_B1\_Attack>v\_B1
3865. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
3866. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
3867. else
3868. B1\_pos(1)=B\_Attack\_pos(1);
3869. B1\_pos(2)=B\_Attack\_pos(2);
3870. end
3872. if dis\_B3\_Attack\_new>v\_B3
3873. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
3874. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
3875. else
3876. B3\_pos(1)=B3\_x;
3877. B3\_pos(2)=4;
3878. end
3880. end
3881. %-------------------%
3883. %-----B3 attack-----%
3884. if dis\_B3\_Attack<dis\_B1\_Attack
3885. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B\_Attack\_pos(1),
3886. B\_Attack\_pos(2));
3887. B1\_x=B2\_pos(1)+sqrt(3^2-(B2\_pos(2)-4)^2);
3888. dis\_B1\_Attack\_new=getDis(B1\_pos(1),B1\_pos(2),B1\_x,4);
3889. theta\_B2=getTheta(B1\_pos(1),B1\_pos(2),B1\_x,4);
3891. if dis\_B3\_Attack>v\_B3
3892. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
3893. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
3894. else
3895. B3\_pos(1)=B\_Attack\_pos(1);
3896. B3\_pos(2)=B\_Attack\_pos(2);
3897. end
3899. if dis\_B1\_Attack\_new>v\_B1
3900. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
3901. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
3902. else
3903. B1\_pos(1)=B1\_x;
3904. B1\_pos(2)=4;
3905. end
3907. end
3908. %-------------------%
3909. %The code calculates and updates the distances and angles between three positions (A1, A2, A3) and a ball position. A random decision is made regarding the ball's movement.
3910. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3911. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3912. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3913. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3914. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3915. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3917. if dis\_A1\_Ball>v\_A1
3918. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3919. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3920. else
3921. A1\_pos(1)=Ball\_pos(1);
3922. A1\_pos(2)=Ball\_pos(2);
3923. theta\_A1=theta\_B2;
3924. if rand(1)>=0.3
3925. Ball\_right=1;
3926. end
3927. end
3929. if dis\_B2\_Ball>v\_B2
3930. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3931. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3932. else
3933. A2\_pos(1)=Ball\_pos(1);
3934. A2\_pos(2)=Ball\_pos(2);
3935. theta\_A2=theta\_B2;
3936. if rand(1)>=0.3
3937. Ball\_right=2;
3938. end
3939. end
3941. if dis\_A3\_Ball>v\_A3
3942. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3943. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
3944. else
3945. A3\_pos(1)=Ball\_pos(1);
3946. A3\_pos(2)=Ball\_pos(2);
3947. theta\_A3=theta\_B2;
3948. if rand(1)>=0.3
3949. Ball\_right=3;
3950. end
3951. end                     draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
3952. pause(0.2);
3953. end
3954. end
3955. %------------------------%
3956. end
3957. end
3958. %Calculate the time, distance and number of iterations when ball right is 6
3959. if Ball\_right==6
3960. dis\_B2\_Goal2=getDis(B2\_pos(1),B2\_pos(2),Goal2\_pos(1),Goal2\_pos(2));
3961. t\_B2\_Goal2=dis\_B2\_Goal2/v\_Ball;
3962. num\_B2\_Goal2=ceil(t\_B2\_Goal2);
3964. %-----Kick Ball-----%
3965. for i=1:num\_B2\_Goal2
3966. if Ball\_right==6
3967. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B2);
3968. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B2);
3969. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3970. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3972. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3973. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
3974. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3975. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
3976. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3977. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
3979. if dis\_A1\_Ball>v\_A1
3980. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
3981. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
3982. else
3983. A1\_pos(1)=Ball\_pos(1);
3984. A1\_pos(2)=Ball\_pos(2);
3985. Ball\_right=1;
3986. end
3988. if dis\_A2\_Ball>v\_A2
3989. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
3990. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
3991. else
3992. A2\_pos(1)=Ball\_pos(1);
3993. A2\_pos(2)=Ball\_pos(2);
3994. Ball\_right=2;
3995. end
3997. if dis\_A3\_Ball>v\_A3
3998. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
3999. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4000. else
4001. A3\_pos(1)=Ball\_pos(1);
4002. A3\_pos(2)=Ball\_pos(2);
4003. Ball\_right=3;
4004. end
4006. if Goal2\_pos(2)==Goal2\_t\_pos(2)
4007. a=1;
4008. else
4009. a=-1;
4010. end
4012. A4\_pos(1)=A4\_pos(1);
4013. A4\_pos(2)=A4\_pos(2)+a\*v\_A4;
4015. if A4\_pos(2) >= Goal2\_t\_pos(2)
4016. A4\_pos(2)=Goal2\_t\_pos(2);
4017. end
4019. if A4\_pos(2) <= Goal2\_b\_pos(2)
4020. A4\_pos(2)=Goal2\_b\_pos(2);
4021. end
4023. if Ball\_pos(1)<Goal2\_pos(1)
4024. Ball\_pos(1)=Goal2\_pos(1);
4025. Ball\_pos(2)=Goal2\_pos(2);
4026. end
4028. if Ball\_pos(1)==A4\_pos(1) && Ball\_pos(2)==A4\_pos(2)
4029. Ball\_right=4;
4030. end
4032. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
4033. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
4034. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,
4035. B3\_pos,theta\_B3,B4\_pos,theta\_B4);
4036. pause(0.2);
4037. end
4038. end
4039. %-------------------%
4041. Ball\_right=4;
4042. if Ball\_pos(2)~=A4\_pos(2)
4043. score\_B=score\_B+1;
4044. end
4045. pause(1);
4046. end
4047. end
4048. end
4049. %The code calculates the distance of each player to the ball and determines the angle that the player must move to in order to reach the ball. The position of player B3 is then updated based on the determined angle and the velocity of the player. If the player reaches the upper or lower boundary, their position is adjusted accordingly. Finally, the position of the ball is updated to the new position of player B3. This loop will continue until the ball is in position 7(the boundary).
4050. while (Ball\_right==7)
4051. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4052. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4053. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4054. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4055. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4056. dis\_B3\_Ball=getDis(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4057. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4058. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4059. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4060. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4061. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4062. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4064. if dis\_A1\_Ball <= dis\_A2\_Ball && dis\_A1\_Ball <= dis\_A3\_Ball
4065. theta\_A=theta\_A1;
4066. end
4067. if dis\_A2\_Ball < dis\_A1\_Ball && dis\_A2\_Ball <= dis\_A3\_Ball
4068. theta\_A=theta\_A2;
4069. end
4070. if dis\_A3\_Ball < dis\_A1\_Ball && dis\_A3\_Ball < dis\_A2\_Ball
4071. theta\_A=theta\_A3;
4072. end
4074. theta\_B3\_next1=theta\_A+pi/2;
4075. delta\_x\_B3\_next1=v\_B3\*cos(theta\_B3\_next1);
4076. delta\_y\_B3\_next1=v\_B3\*sin(theta\_B3\_next1);
4077. B3\_posnew\_1(1)=B3\_pos(1)-delta\_x\_B3\_next1;
4078. B3\_posnew\_1(2)=B3\_pos(2)-delta\_y\_B3\_next1;
4079. theta\_B3\_next2=theta\_A-pi/2;
4080. delta\_x\_B3\_next2=v\_B3\*cos(theta\_B3\_next2);
4081. delta\_y\_B3\_next2=v\_B3\*sin(theta\_B3\_next2);
4082. B3\_posnew\_2(1)=B3\_pos(1)-delta\_x\_B3\_next2;
4083. B3\_posnew\_2(2)=B3\_pos(2)-delta\_y\_B3\_next2;
4084. dis\_B3\_next1=getDis(B3\_posnew\_1(1),B3\_posnew\_1(2),Goal2\_pos\_abs(1),Goal2\_pos\_abs(2));
4085. dis\_B3\_next2=getDis(B3\_posnew\_2(1),B3\_posnew\_2(2),Goal2\_pos\_abs(1),Goal2\_pos\_abs(2));
4086. if dis\_B3\_next1<dis\_B3\_next2
4087. theta\_B3=theta\_B3\_next1;
4088. end
4089. if dis\_B3\_next1>dis\_B3\_next2
4090. theta\_B3=theta\_B3\_next2;
4091. end
4092. if dis\_B3\_next1==dis\_B3\_next2
4093. if rand(1)>=0.5
4094. theta\_B3=theta\_B3\_next1;
4095. else
4096. theta\_B3=theta\_B3\_next2;
4097. end
4098. end
4099. delta\_x\_B3=v\_B3\*cos(theta\_B3);
4100. delta\_y\_B3=v\_B3\*sin(theta\_B3);
4101. B3\_posnew(1)=B3\_pos(1)-delta\_x\_B3;
4102. B3\_posnew(2)=B3\_pos(2)-delta\_y\_B3;
4104. if B3\_posnew(2)>=7
4105. B3\_pos(1)=B3\_posnew(1);
4106. B3\_pos(2)=7;
4107. else
4108. if B3\_posnew(2)<=1
4109. B3\_pos(1)=B3\_posnew(1);
4110. B3\_pos(2)=1;
4111. else
4112. B3\_pos(1)=B3\_posnew(1);
4113. B3\_pos(2)=B3\_posnew(2);
4114. end
4115. end
4116. Ball\_pos(1)=B3\_pos(1);
4117. Ball\_pos(2)=B3\_pos(2);
4118. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,A4\_pos,
4119. theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
4120. pause(0.2);
4121. %The code calculates the distance and time required for object B3 to reach two goal positions (Goal2\_t and Goal2\_b) based on its current position and speed.
4122. %-----B3 to t or b-----%
4123. dis\_B3\_Goal2\_t=getDis(B3\_pos(1),B3\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
4124. t\_B3\_t=dis\_B3\_Goal2\_t/v\_B3;
4125. num\_B3\_t=ceil(t\_B3\_t);
4126. dis\_B3\_Goal2\_b=getDis(B3\_pos(1),B3\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
4127. t\_B3\_b=dis\_B3\_Goal2\_b/v\_B3;
4128. num\_B3\_b=ceil(t\_B3\_b);
4129. if num\_B3\_t < num\_B3\_b
4130. Goal2\_pos=Goal2\_t\_pos;
4131. theta\_B3= getTheta(B3\_pos(1),B3\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
4132. dis\_B3\_Goal2=dis\_B3\_Goal2\_t;
4133. end
4134. if num\_B3\_b < num\_B3\_t
4135. Goal2\_pos=Goal2\_b\_pos;
4136. theta\_B3= getTheta(B3\_pos(1),B3\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
4137. dis\_B3\_Goal2=dis\_B3\_Goal2\_b;
4138. end
4139. %If the distances to the two goal positions are equal, it randomly chooses one of them as the goal position
4140. if num\_B3\_t == num\_B3\_b
4141. if rand(1)>=0.5
4142. Goal2\_pos=Goal2\_t\_pos;
4143. theta\_B3= getTheta(B3\_pos(1),B3\_pos(2),Goal2\_t\_pos(1),Goal2\_t\_pos(2));
4144. dis\_B3\_Goal2=dis\_B3\_Goal2\_t;
4145. else
4146. Goal2\_pos=Goal2\_b\_pos;
4147. theta\_B3= getTheta(B3\_pos(1),B3\_pos(2),Goal2\_b\_pos(1),Goal2\_b\_pos(2));
4148. dis\_B3\_Goal2=dis\_B3\_Goal2\_b;
4149. end
4150. end
4151. %----------------------%
4152. %The code calculates the distance between the player(B3) and the goal, and if that distance is greater than zero, it calculates the time required for B3 to move to that position and sets a loop to iterate through that time period.
4153. delta\_dis=dis\_B3\_Goal2-3;
4155. if delta\_dis>0
4156. t\_B3=delta\_dis/v\_B3;
4157. num\_B3=ceil(t\_B3);
4159. %-----A1/A2/A3-----%
4160. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4161. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4162. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4163. %------------------%
4164. %It also calculates the distance between the ball and players, and checks which player is closest to the ball. Depending on which player is closest, the code calculates the angles and velocities required for attacking with two other players, and updates their positions accordingly.
4165. for i=1:num\_B3
4166. if Ball\_right==7
4167. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
4168. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
4169. Ball\_pos(1)=B3\_pos(1);
4170. Ball\_pos(2)=B3\_pos(2);
4172. %-----B1/B2-----%
4173. if Goal2\_pos(2)==Goal2\_t\_pos(2)
4174. B\_Attack\_pos=[B3\_pos(1),B3\_pos(2)-2];
4175. end
4176. if Goal2\_pos(2)==Goal2\_b\_pos(2)
4177. B\_Attack\_pos=[B3\_pos(1),B3\_pos(2)+2];
4178. end
4179. dis\_B1\_Attack=getDis(B1\_pos(1),B1\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
4180. dis\_B2\_Attack=getDis(B2\_pos(1),B2\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
4181. %---------------%
4183. %-----B1 attack-----%
4184. if dis\_B1\_Attack<=dis\_B2\_Attack
4185. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
4186. B2\_x=B3\_pos(1)+sqrt(3^2-(B2\_pos(2)-4)^2);
4187. dis\_B2\_Attack\_new=getDis(B2\_pos(1),B2\_pos(2),B2\_x,4);
4188. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B2\_x,4);
4190. if dis\_B1\_Attack>v\_B1
4191. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
4192. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
4193. else
4194. B1\_pos(1)=B\_Attack\_pos(1);
4195. B1\_pos(2)=B\_Attack\_pos(2);
4196. end
4198. if dis\_B2\_Attack\_new>v\_B2
4199. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
4200. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
4201. else
4202. B2\_pos(1)=B2\_x;
4203. B2\_pos(2)=4;
4204. end
4205. end
4206. %-------------------%
4208. %-----B2 attack-----%
4209. if dis\_B2\_Attack<dis\_B1\_Attack
4210. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
4211. B1\_x=B3\_pos(1)+sqrt(3^2-(B3\_pos(2)-4)^2);
4212. dis\_B1\_Attack\_new=getDis(B1\_pos(1),B1\_pos(2),B1\_x,4);
4213. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),B1\_x,4);
4215. if dis\_B2\_Attack>v\_B2
4216. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
4217. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
4218. else
4219. B2\_pos(1)=B\_Attack\_pos(1);
4220. B2\_pos(2)=B\_Attack\_pos(2);
4221. end
4223. if dis\_B1\_Attack\_new>v\_B1
4224. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
4225. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
4226. else
4227. B1\_pos(1)=B1\_x;
4228. B1\_pos(2)=4;
4229. end
4230. end
4231. %-------------------%
4232. %three players A1, A2, and A3 are moving towards the ball while defending players in team B positioned at position 4.
4233. %------A1 to Ball-----%
4234. %In the first block, if the distance from A1 to ball is less than the distances from A2 and A3 to ball, then A1 moves towards the ball, A2 moves towards defind\_pos\_3, and A3 moves towards defind\_pos\_4. The code updates the positions of these players and changes their directions if they reach the corresponding position.
4235. %The second and the third block is similar to the first block, but it handles the case where A2 or A3 is the closest player to the ball.
4236. if dis\_A1\_Ball<=dis\_A2\_Ball && dis\_A1\_Ball<=dis\_A3\_Ball
4237. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4238. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4239. dis\_A2\_Defend=getDis(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
4240. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
4241. dis\_A3\_Defend=getDis(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
4242. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
4244. if dis\_A2\_Defend>v\_A2
4245. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
4246. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
4247. else
4248. A2\_pos(1)=defind\_pos\_3(1);
4249. A2\_pos(2)=defind\_pos\_3(2);
4250. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4251. end
4253. if dis\_A3\_Defend>v\_A3
4254. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
4255. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4256. else
4257. A3\_pos(1)=defind\_pos\_4(1);
4258. A3\_pos(2)=defind\_pos\_4(2);
4259. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4260. end
4262. if dis\_A1\_Ball>v\_A1
4263. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
4264. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
4265. else
4266. A1\_pos(1)=Ball\_pos(1);
4267. A1\_pos(2)=Ball\_pos(2);
4268. theta\_A1=theta\_B3;
4269. if rand(1)>=0.3
4270. Ball\_right=1;
4271. end
4272. end
4273. end
4274. %--------------------%
4276. %-----A2 to Ball-----%
4277. if dis\_A2\_Ball<dis\_A1\_Ball && dis\_A2\_Ball<=dis\_A3\_Ball
4278. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4279. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4280. dis\_A1\_Defend=getDis(A1\_pos(1),A1\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
4281. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
4282. dis\_A3\_Defend=getDis(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
4283. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
4285. if dis\_A1\_Defend>v\_A1
4286. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
4287. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
4288. else
4289. A1\_pos(1)=defind\_pos\_3(1);
4290. A1\_pos(2)=defind\_pos\_3(2);
4291. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4292. end
4294. if dis\_A3\_Defend>v\_A3
4295. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
4296. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4297. else
4298. A3\_pos(1)=defind\_pos\_4(1);
4299. A3\_pos(2)=defind\_pos\_4(2);
4300. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4301. end
4303. if dis\_A2\_Ball>v\_A2
4304. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
4305. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
4306. else
4307. A2\_pos(1)=Ball\_pos(1);
4308. A2\_pos(2)=Ball\_pos(2);
4309. theta\_A2=theta\_B3;
4310. if rand(1)>=0.3
4311. Ball\_right=2;
4312. end
4313. end
4314. end
4315. %--------------------%
4317. %-----A3 to Ball-----%
4318. if dis\_A3\_Ball<dis\_A1\_Ball && dis\_A3\_Ball<dis\_A2\_Ball
4319. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4320. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4321. dis\_A1\_Defend=getDis(A1\_pos(1),A1\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
4322. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),defind\_pos\_4(1),defind\_pos\_4(2));
4323. dis\_A2\_Defend=getDis(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
4324. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),defind\_pos\_3(1),defind\_pos\_3(2));
4326. if dis\_A1\_Defend>v\_A1
4327. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
4328. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
4329. else
4330. A1\_pos(1)=defind\_pos\_4(1);
4331. A1\_pos(2)=defind\_pos\_4(2);
4332. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4333. end
4335. if dis\_A2\_Defend>v\_A2
4336. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
4337. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
4338. else
4339. A2\_pos(1)=defind\_pos\_3(1);
4340. A2\_pos(2)=defind\_pos\_3(2);
4341. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4342. end
4344. if dis\_A3\_Ball>v\_A3
4345. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
4346. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4347. else
4348. A3\_pos(1)=Ball\_pos(1);
4349. A3\_pos(2)=Ball\_pos(2);
4350. theta\_A3=theta\_B3;
4351. if rand(1)>=0.3
4352. Ball\_right=3;
4353. end
4354. end
4355. end
4356. %--------------------%
4357. end
4358. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
4359. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,
4360. theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
4361. pause(0.2);
4362. end
4363. end
4364. % it calculates the distances between the positions of different objects (B3, A1, A2, A3, B1, B2, and Goal2) and assigns them to respective variables. It then calculates the time it would take for B3 to reach Goal2 and assigns it to "t\_B3\_Goal2"
4365. if Ball\_right==7
4366. dis\_B3\_A1=getDis(B3\_pos(1),B3\_pos(2),A1\_pos(1),A1\_pos(2));
4367. dis\_B3\_A2=getDis(B3\_pos(1),B3\_pos(2),A2\_pos(1),A2\_pos(2));
4368. dis\_B3\_A3=getDis(B3\_pos(1),B3\_pos(2),A3\_pos(1),A3\_pos(2));
4369. dis\_B3\_B2=getDis(B3\_pos(1),B3\_pos(2),B2\_pos(1),B2\_pos(2));
4370. dis\_B3\_B1=getDis(B3\_pos(1),B3\_pos(2),B1\_pos(1),B1\_pos(2));
4371. dis\_B3\_Goal2=getDis(B3\_pos(1),B3\_pos(2),Goal2\_pos(1),Goal2\_pos(2));
4372. % calculates the time it would take for B3 to reach Goal2 and assigns it to "t\_B3\_Goal2".
4373. t\_B3\_Goal2=dis\_B3\_Goal2/v\_Ball;
4374. num\_B3\_Goal2=ceil(t\_B3\_Goal2);
4376. if num\_B3\_Goal2 > frame
4377. % checks if the distances between the positions of A1, A2, or A3 and B3 are less than or equal to their respective velocities
4378. if dis\_B3\_A1<=v\_A1 || dis\_B3\_A2<=v\_A2 || dis\_B3\_A3<=v\_A3
4379. if B1\_pos(1)<=B2\_pos(1)
4380. dis\_B=dis\_B3\_B1;
4381. else
4382. dis\_B=dis\_B3\_B2;
4383. end
4384. %Check which player is closer and decide who to pass the ball
4385. %-----Pass Ball-----%
4386. if dis\_B<=3
4387. if dis\_B==dis\_B3\_B1
4388. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B1\_pos(1),B1\_pos(2));
4389. B\_pos(1)=B1\_pos(1);
4390. B\_pos(2)=B1\_pos(2);
4391. else
4392. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),B2\_pos(1),B2\_pos(2));
4393. B\_pos(1)=B2\_pos(1);
4394. B\_pos(2)=B2\_pos(2);
4395. end
4397. t\_B=dis\_B/v\_Ball;
4398. num\_B=ceil(t\_B);
4400. for i=1:num\_B
4401. if Ball\_right==7
4402. if i<num\_B
4403. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B3);
4404. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B3);
4405. else
4406. Ball\_pos(1)=B\_pos(1);
4407. Ball\_pos(2)=B\_pos(2);
4408. end
4410. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4411. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4412. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4413. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4414. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4415. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4416. % If the distances are less than or equal to their respective velocities, it assigns the position of the ball to the position of the respective player and updates the value of "Ball\_right".
4417. if dis\_A1\_Ball>v\_A1
4418. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
4419. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
4420. else
4421. A1\_pos(1)=Ball\_pos(1);
4422. A1\_pos(2)=Ball\_pos(2);
4423. Ball\_right=1;
4424. end
4426. if dis\_A2\_Ball>v\_A2
4427. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
4428. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
4429. else
4430. A2\_pos(1)=Ball\_pos(1);
4431. A2\_pos(2)=Ball\_pos(2);
4432. Ball\_right=2;
4433. end
4435. if dis\_A3\_Ball>v\_A3
4436. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
4437. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4438. else
4439. A3\_pos(1)=Ball\_pos(1);
4440. A3\_pos(2)=Ball\_pos(2);
4441. Ball\_right=3;
4442. end
4444. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4445. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4446. dis\_B1\_Ball=getDis(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4447. dis\_B2\_Ball=getDis(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4449. if dis\_B1\_Ball==0
4450. Ball\_right=5;
4451. end
4453. if dis\_B2\_Ball==0
4454. Ball\_right=6;
4455. end
4456. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,
4457. theta\_A2,A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,
4458. theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,
4459. B4\_pos,theta\_B4);
4460. pause(0.2);
4461. end
4462. end
4463. end
4464. %-------------------%
4465. %the ball is kicked towards Goal2. The code calculates the time required for the ball to reach Goal2, based on its initial position and velocity. It then updates the positions of the ball and players A1, A2, A3, and A4, based on their velocities and angles with respect to the ball.
4466. %-----Kick Ball-----%
4467. if Ball\_right==7
4468. t\_B3\_Goal2=dis\_B3\_Goal2/v\_Ball;
4469. num\_B3\_Goal2=ceil(t\_B3\_Goal2);
4471. for i=1:num\_B3\_Goal2
4472. if Ball\_right==7
4473. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B3);
4474. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B3);
4475. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4476. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4478. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4479. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4480. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4481. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4482. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4483. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4484. %If any player gets closer to the ball than their maximum distance, they take possession of the ball, and the ball\_right variable is updated accordingly.
4485. if dis\_A1\_Ball>v\_A1
4486. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
4487. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
4488. else
4489. A1\_pos(1)=Ball\_pos(1);
4490. A1\_pos(2)=Ball\_pos(2);
4491. Ball\_right=1;
4492. end
4494. if dis\_A2\_Ball>v\_A2
4495. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
4496. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
4497. else
4498. A2\_pos(1)=Ball\_pos(1);
4499. A2\_pos(2)=Ball\_pos(2);
4500. Ball\_right=2;
4501. end
4503. if dis\_A3\_Ball>v\_A3
4504. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
4505. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4506. else
4507. A3\_pos(1)=Ball\_pos(1);
4508. A3\_pos(2)=Ball\_pos(2);
4509. Ball\_right=3;
4510. end
4512. if Goal2\_pos(2)==Goal2\_t\_pos(2)
4513. a=1;
4514. else
4515. a=-1;
4516. end
4518. A4\_pos(1)=A4\_pos(1);
4519. A4\_pos(2)=A4\_pos(2)+a\*v\_A4;
4521. if A4\_pos(2) >= Goal2\_t\_pos(2)
4522. A4\_pos(2)=Goal2\_t\_pos(2);
4523. end
4525. if A4\_pos(2) <= Goal2\_b\_pos(2)
4526. A4\_pos(2)=Goal2\_b\_pos(2);
4527. end
4529. if Ball\_pos(1)<Goal2\_pos(1)
4530. Ball\_pos(1)=Goal2\_pos(1);
4531. Ball\_pos(2)=Goal2\_pos(2);
4532. end
4534. if Ball\_pos(1)==A4\_pos(1) && Ball\_pos(2)==A4\_pos(2)
4535. Ball\_right=4;
4536. end
4538. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
4539. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,
4540. theta\_A2,A3\_pos,theta\_A3,A4\_pos,theta\_A4,
4541. B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,
4542. B4\_pos,theta\_B4);
4543. pause(0.2);
4544. end
4545. end
4546. pause(1);
4547. end
4548. %-------------------%
4549. end
4550. %the code calculates the time it would take for the ball to reach a particular point (goal2) and moves players and the ball accordingly.
4551. if Ball\_right==7
4552. det\_dis\_B3\_Goal2=dis\_B3\_Goal2-frame\*v\_Ball;
4553. t\_det=det\_dis\_B3\_Goal2/v\_B3;
4554. num\_det=ceil(t\_det);
4556. %-----move with ball-----%
4557. for i=1:num\_det
4558. if Ball\_right==7
4559. B3\_pos(1)=B3\_pos(1)-v\_B3\*cos(theta\_B3);
4560. B3\_pos(2)=B3\_pos(2)-v\_B3\*sin(theta\_B3);
4561. Ball\_pos(1)=B3\_pos(1);
4562. Ball\_pos(2)=B3\_pos(2);
4563. % If the ball is with player B3, then players B1 and B2 move to attack the opposing team's goal. The code calculates the distance between the attackers and the goal and updates their positions accordingly.
4564. %-----B1/B2-----%
4565. if Goal2\_pos(2)==Goal2\_t\_pos(2)
4566. B\_Attack\_pos=[B3\_pos(1),B3\_pos(2)-2];
4567. end
4568. if Goal2\_pos(2)==Goal2\_b\_pos(2)
4569. B\_Attack\_pos=[B3\_pos(1),B3\_pos(2)+2];
4570. end
4571. dis\_B2\_Attack=getDis(B2\_pos(1),B2\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
4572. dis\_B1\_Attack=getDis(B1\_pos(1),B1\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
4573. %---------------%
4575. %-----B2 attack-----%
4576. if dis\_B2\_Attack<=dis\_B3\_Attack
4577. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B\_Attack\_pos(1),B\_Attack\_pos(2));
4578. B1\_x=B3\_pos(1)+sqrt(3^2-(B3\_pos(2)-4)^2);
4579. dis\_B1\_Attack\_new=getDis(B1\_pos(1),B1\_pos(2),B1\_x,4);
4580. theta\_B3=getTheta(B1\_pos(1),B1\_pos(2),B1\_x,4);
4582. if dis\_B2\_Attack>v\_B2
4583. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
4584. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
4585. else
4586. B2\_pos(1)=B\_Attack\_pos(1);
4587. B2\_pos(2)=B\_Attack\_pos(2);
4588. end
4590. if dis\_B1\_Attack\_new>v\_B1
4591. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
4592. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
4593. else
4594. B1\_pos(1)=B1\_x;
4595. B1\_pos(2)=4;
4596. end
4597. end
4598. %-------------------%
4600. %-----B1 attack-----%
4601. if dis\_B1\_Attack<dis\_B2\_Attack
4602. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),B\_Attack\_pos(1),
4603. B\_Attack\_pos(2));
4604. B2\_x=B3\_pos(1)+sqrt(3^2-(B3\_pos(2)-4)^2);
4605. dis\_B2\_Attack\_new=getDis(B2\_pos(1),B2\_pos(2),B2\_x,4);
4606. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),B2\_x,4);
4608. if dis\_B1\_Attack>v\_B1
4609. B1\_pos(1)=B1\_pos(1)-v\_B1\*cos(theta\_B1);
4610. B1\_pos(2)=B1\_pos(2)-v\_B1\*sin(theta\_B1);
4611. else
4612. B1\_pos(1)=B\_Attack\_pos(1);
4613. B1\_pos(2)=B\_Attack\_pos(2);
4614. end
4616. if dis\_B2\_Attack\_new>v\_B2
4617. B2\_pos(1)=B2\_pos(1)-v\_B2\*cos(theta\_B2);
4618. B2\_pos(2)=B2\_pos(2)-v\_B2\*sin(theta\_B2);
4619. else
4620. B2\_pos(1)=B2\_x;
4621. B2\_pos(2)=4;
4622. end
4624. end
4625. %-------------------%
4626. % calculates the distance between the ball and the other team's defenders (A1, A2, A3) and moves them towards the ball.
4627. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_po s(2));
4628. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4629. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4630. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4631. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4632. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4634. if dis\_A1\_Ball>v\_A1
4635. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
4636. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
4637. else
4638. A1\_pos(1)=Ball\_pos(1);
4639. A1\_pos(2)=Ball\_pos(2);
4640. theta\_A1=theta\_B3;
4641. if rand(1)>=0.3
4642. Ball\_right=1;
4643. end
4644. end
4646. if dis\_B2\_Ball>v\_B2
4647. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
4648. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
4649. else
4650. A2\_pos(1)=Ball\_pos(1);
4651. A2\_pos(2)=Ball\_pos(2);
4652. theta\_A2=theta\_B3;
4653. if rand(1)>=0.3
4654. Ball\_right=2;
4655. end
4656. end
4658. if dis\_A3\_Ball>v\_A3
4659. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
4660. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4661. else
4662. A3\_pos(1)=Ball\_pos(1);
4663. A3\_pos(2)=Ball\_pos(2);
4664. theta\_A3=theta\_B3;
4665. if rand(1)>=0.3
4666. Ball\_right=3;
4667. end
4668. end
4669. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
4670. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,
4671. theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
4672. pause(0.2);
4674. end
4675. end
4676. %------------------------%
4677. end
4678. end
4679. % The code calculates the time and distance required for the ball to reach the goal and updates the positions of all players and the ball accordingly.
4680. if Ball\_right==7
4681. dis\_B3\_Goal2=getDis(B3\_pos(1),B3\_pos(2),Goal2\_pos(1),Goal2\_pos(2));
4682. t\_B3\_Goal2=dis\_B3\_Goal2/v\_Ball;
4683. num\_B3\_Goal2=ceil(t\_B3\_Goal2);
4685. %-----Kick Ball-----%
4686. for i=1:num\_B3\_Goal2
4687. if Ball\_right==7
4688. Ball\_pos(1)=Ball\_pos(1)-v\_Ball\*cos(theta\_B3);
4689. Ball\_pos(2)=Ball\_pos(2)-v\_Ball\*sin(theta\_B3);
4690. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4691. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4693. dis\_A1\_Ball=getDis(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4694. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4695. dis\_A2\_Ball=getDis(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4696. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4697. dis\_A3\_Ball=getDis(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4698. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4699. % decides which team should get the ball next.
4700. if dis\_A1\_Ball>v\_A1
4701. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
4702. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
4703. else
4704. A1\_pos(1)=Ball\_pos(1);
4705. A1\_pos(2)=Ball\_pos(2);
4706. Ball\_right=1;
4707. end
4709. if dis\_A2\_Ball>v\_A2
4710. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
4711. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
4712. else
4713. A2\_pos(1)=Ball\_pos(1);
4714. A2\_pos(2)=Ball\_pos(2);
4715. Ball\_right=2;
4716. end
4718. if dis\_A3\_Ball>v\_A3
4719. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
4720. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4721. else
4722. A3\_pos(1)=Ball\_pos(1);
4723. A3\_pos(2)=Ball\_pos(2);
4724. Ball\_right=3;
4725. end
4727. if Goal2\_pos(2)==Goal2\_t\_pos(2)
4728. a=1;
4729. else
4730. a=-1;
4731. end
4733. A4\_pos(1)=A4\_pos(1);
4734. A4\_pos(2)=A4\_pos(2)+a\*v\_A4;
4736. if A4\_pos(2) >= Goal2\_t\_pos(2)
4737. A4\_pos(2)=Goal2\_t\_pos(2);
4738. end
4740. if A4\_pos(2) <= Goal2\_b\_pos(2)
4741. A4\_pos(2)=Goal2\_b\_pos(2);
4742. end
4744. if Ball\_pos(1)<Goal2\_pos(1)
4745. Ball\_pos(1)=Goal2\_pos(1);
4746. Ball\_pos(2)=Goal2\_pos(2);
4747. end
4749. if Ball\_pos(1)==A4\_pos(1) && Ball\_pos(2)==A4\_pos(2)
4750. Ball\_right=4;
4751. end
4752. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),
4753. Ball\_pos(2));
4754. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,
4755. A3\_pos,theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,
4756. theta\_B2,B3\_pos,theta\_B3,B4\_pos,theta\_B4);
4757. pause(0.2);
4758. end
4759. end
4760. %-------------------%
4762. Ball\_right=4;
4763. if Ball\_pos(2)~=A4\_pos(2)
4764. score\_B=score\_B+1;
4765. end
4766. pause(1);
4767. end
4768. end
4769. end
4771. while (Ball\_right==8 && score\_A<3 && score\_B<3)
4772. % The code initializes the positions of the players and the ball.
4773. A1\_pos=[4 4];
4774. A2\_pos=[4 6];
4775. A3\_pos=[4 2];
4776. A4\_pos=[1 4];
4778. B4\_pos=[10 4];
4779. Ball\_pos=[10 4];
4780. % The variable "a" is set to a random value between 0 and 1. Depending on the value of "a", the positions of the players on team B are set differently.
4781. a=rand(1);
4783. if a<=0.33
4784. B1\_pos=[7 4];
4785. B2\_pos=[6 6];
4786. B3\_pos=[6 2];
4787. end
4789. if a>0.33 && a<0.66
4790. B1\_pos=[7 4];
4791. B2\_pos=[7 6];
4792. B3\_pos=[7 2];
4793. end
4795. if a>=0.66 && a<=1
4796. B1\_pos=[6 4];
4797. B2\_pos=[7 6];
4798. B3\_pos=[7 2];
4799. end
4801. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4802. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4803. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4804. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
4805. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4806. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4807. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4808. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Ball\_pos(1),Ball\_pos(2));
4809. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,
4810. theta\_A3,A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,
4811. theta\_B3,B4\_pos,theta\_B4);
4812. pause(0.2);
4813. % It starts by randomly selecting a target position (Aim\_pos) from three possible positions (B1\_pos, B2\_pos, and B3\_pos) based on a random value b.
4814. b=rand(1);
4816. if b<=0.33
4817. Aim\_pos=B1\_pos;
4818. end
4820. if b>0.33 && b<0.66
4821. Aim\_pos=B2\_pos;
4822. end
4824. if b>=0.66 && b<=1
4825. Aim\_pos=B3\_pos;
4826. end
4828. dis\_B4\_Aim=getDis(B4\_pos(1),B4\_pos(2),Aim\_pos(1),Aim\_pos(2));
4829. theta\_B4=getTheta(B4\_pos(1),B4\_pos(2),Aim\_pos(1),Aim\_pos(2));
4830. v\_Aim=dis\_B4\_Aim/frame;
4831. dis\_A1\_Aim=getDis(A1\_pos(1),A1\_pos(2),Aim\_pos(1),Aim\_pos(2));
4832. dis\_A2\_Aim=getDis(A2\_pos(1),A2\_pos(2),Aim\_pos(1),Aim\_pos(2));
4833. dis\_A3\_Aim=getDis(A3\_pos(1),A3\_pos(2),Aim\_pos(1),Aim\_pos(2));
4834. %In a loop that runs for the specified number of frames, the code updates the position of the ball based on its speed and angle.
4835. for i=1:frame
4836. if Ball\_right==8
4837. Ball\_pos(1)=Ball\_pos(1)-v\_Aim\*cos(theta\_B4);
4838. Ball\_pos(2)=Ball\_pos(2)-v\_Aim\*sin(theta\_B4);
4839. theta\_A1=getTheta(A1\_pos(1),A1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4840. theta\_A2=getTheta(A2\_pos(1),A2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4841. theta\_A3=getTheta(A3\_pos(1),A3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4842. theta\_A4=getTheta(A4\_pos(1),A4\_pos(2),Ball\_pos(1),Ball\_pos(2));
4843. theta\_B1=getTheta(B1\_pos(1),B1\_pos(2),Ball\_pos(1),Ball\_pos(2));
4844. theta\_B2=getTheta(B2\_pos(1),B2\_pos(2),Ball\_pos(1),Ball\_pos(2));
4845. theta\_B3=getTheta(B3\_pos(1),B3\_pos(2),Ball\_pos(1),Ball\_pos(2));
4847. if i==frame
4848. if dis\_A1\_Aim<=dis\_A2\_Aim && dis\_A1\_Aim<=dis\_A3\_Aim
4849. A1\_pos(1)=A1\_pos(1)-v\_A1\*cos(theta\_A1);
4850. A1\_pos(2)=A1\_pos(2)-v\_A1\*sin(theta\_A1);
4851. end
4852. if dis\_A2\_Aim<dis\_A1\_Aim && dis\_A2\_Aim<=dis\_A3\_Aim
4853. A2\_pos(1)=A2\_pos(1)-v\_A2\*cos(theta\_A2);
4854. A2\_pos(2)=A2\_pos(2)-v\_A2\*sin(theta\_A2);
4855. end
4856. if dis\_A3\_Aim<dis\_A1\_Aim && dis\_A3\_Aim<dis\_A2\_Aim
4857. A3\_pos(1)=A3\_pos(1)-v\_A3\*cos(theta\_A3);
4858. A3\_pos(2)=A3\_pos(2)-v\_A3\*sin(theta\_A3);
4859. end
4860. end
4861. draw\_whole(Ball\_pos,A1\_pos,theta\_A1,A2\_pos,theta\_A2,A3\_pos,theta\_A3,
4862. A4\_pos,theta\_A4,B1\_pos,theta\_B1,B2\_pos,theta\_B2,B3\_pos,theta\_B3,
4863. B4\_pos,theta\_B4);
4864. pause(0.2);
4865. end
4866. end
4867. % After the loop ends, the code checks whether the ball has reached one of the target positions. If so, it sets the ball's position to that target position and sets a variable (Ball\_right) to a corresponding value (5, 6, or 7).
4868. if Aim\_pos(2)== B1\_pos(2)
4869. Ball\_pos(1)=B1\_pos(1);
4870. Ball\_pos(2)=B1\_pos(2);
4871. Ball\_right=5;
4872. end
4873. if Aim\_pos(2)== B2\_pos(2)
4874. Ball\_pos(1)=B2\_pos(1);
4875. Ball\_pos(2)=B2\_pos(2);
4876. Ball\_right=6;
4877. end
4878. if Aim\_pos(2)== B3\_pos(2)
4879. Ball\_pos(1)=B3\_pos(1);
4880. Ball\_pos(2)=B3\_pos(2);
4881. Ball\_right=7;
4882. end
4883. end
4884. end

### MATLAB App Designer written by Yujie Liu

1. classdef app1 < matlab.apps.AppBase
2. % Properties that correspond to app components
3. properties (Access = public)
4. UIFigure          matlab.ui.Figure
5. Title             matlab.ui.control.Label
6. point             matlab.ui.control.EditField
7. timer1            matlab.ui.control.EditField
8. Ball              matlab.ui.control.EditField
9. BallLabel         matlab.ui.control.Label
10. B4                matlab.ui.control.EditField
11. B4Label           matlab.ui.control.Label
12. B3                matlab.ui.control.EditField
13. B3Label           matlab.ui.control.Label
14. B2                matlab.ui.control.EditField
15. B2Label           matlab.ui.control.Label
16. B1                matlab.ui.control.EditField
17. B1Label           matlab.ui.control.Label
18. timer1\_           matlab.ui.control.NumericEditField
19. datetime          matlab.ui.control.EditField
20. A4                matlab.ui.control.EditField
21. A4EditFieldLabel  matlab.ui.control.Label
22. A3                matlab.ui.control.EditField
23. A3EditFieldLabel  matlab.ui.control.Label
24. A2                matlab.ui.control.EditField
25. A2EditFieldLabel  matlab.ui.control.Label
26. A1                matlab.ui.control.EditField
27. A1EditFieldLabel  matlab.ui.control.Label
28. ResetButton       matlab.ui.control.Button
29. StopButton        matlab.ui.control.Button
30. ScoreLogo         matlab.ui.control.Image
31. TimerLogo         matlab.ui.control.Image
32. FootballLogo      matlab.ui.control.Image
33. PlayButton        matlab.ui.control.Button
34. NAORed            matlab.ui.control.Image
35. NAOBlue           matlab.ui.control.Image
36. Image             matlab.ui.control.Image
37. Image2            matlab.ui.control.Image
38. Image3            matlab.ui.control.Image
39. Image4            matlab.ui.control.Image
40. game              matlab.ui.control.UIAxes
41. end
43. properties (Access = public)
44. Timer\_id1;%Used for real-time synchronization of score display
45. Timer\_id2;%used for real-time display of match animations
46. Timer\_id3;%used for displaying date and time
47. Timer\_id4;%used to assist in displaying a timer
48. Timer\_id5;%used for displaying a timer
49. end
51. methods (Access = public)
53. function timer\_init1(app)
54. app.Timer\_id1 = timer;
55. app.Timer\_id1.StartDelay = 0.01; % delay time for the start to take effect
56. app.Timer\_id1.Period = 0.01;% Period
57. app.Timer\_id1.ExecutionMode = 'fixedSpacing';
58. app.Timer\_id1.TimerFcn = @(~, ~) updatepoint(app);
59. end
60. function updatepoint(app)
61. app.point.Value=strcat(num2str(evalin('base', 'score\_A')),":",num2str(evalin('base', 'score\_B')));
62. %app.point.Value=strcat(num2str(evalin('base', 'score\_A')),"    :    ",num2str(evalin('base', 'score\_B')));
63. end


67. function timer\_init2(app)
68. app.Timer\_id2 = timer;
69. app.Timer\_id2.StartDelay = 0.01; % delay time for the start to take effect
70. app.Timer\_id2.Period = 0.01;% Period
71. app.Timer\_id2.ExecutionMode = 'fixedSpacing';
72. app.Timer\_id2.TimerFcn = @(~, ~) updatepitch(app);
73. end
74. function updatepitch(app)
75. draw\_whole(app,evalin('base', 'g\_Ball\_pos'),evalin('base', 'g\_A1\_pos'),evalin('base', 'g\_theta\_A1'),evalin('base', 'g\_A2\_pos'),evalin('base', 'g\_theta\_A2'),evalin('base', 'g\_A3\_pos'),evalin('base', 'g\_theta\_A3'),evalin('base', 'g\_A4\_pos'),evalin('base', 'g\_theta\_A4'),evalin('base', 'g\_B1\_pos'),evalin('base', 'g\_theta\_B1'),evalin('base', 'g\_B2\_pos'),evalin('base', 'g\_theta\_B2'),evalin('base', 'g\_B3\_pos'),evalin('base', 'g\_theta\_B3'),evalin('base', 'g\_B4\_pos'),evalin('base', 'g\_theta\_B4'));
76. %draw\_whole(app,[5.5 4],[4.5 3],pi,[4.5 4],pi,[4.5 5],pi,[0.5 4],pi,[6.5 3],0,[6.5 4],0,[6.5 5],0,[10.5 4],0);
77. end
78. function results1 = pitch(app)
79. PitchLength=9;
80. PitchWidth=6;
81. GoalDepth=0.6;
82. GoalWidth=2.6;
83. GoalHeight=1.2;
84. GoalAreaLength=1;
85. GoalAreaWidth=3;
86. PenaltyMarkDistance=1.5;
87. CentreCircleDiameter=1.5;
88. BorderStripWidth=1;
89. PenaltyAreaLength=2;
90. PenaltyAreaWidth=5;
92. A=PitchLength;
93. B=PitchWidth;
94. C=GoalDepth;
95. D=GoalWidth;
96. E=GoalAreaLength;
97. F=GoalAreaWidth;
98. G=PenaltyMarkDistance;
99. H=CentreCircleDiameter;
100. I=BorderStripWidth;
101. J=PenaltyAreaLength;
102. K=PenaltyAreaWidth;
104. axis([0 A+2\*I 0 B+2\*I]);
105. hold(app.game,'on')
106. rectangle(app.game,'Position',[0 0 2\*I+A 2\*I+B],'FaceColor','g','EdgeColor','w');
107. hold(app.game,'on')
108. rectangle(app.game,'Position',[I I A B],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
109. hold(app.game,'on')
110. rectangle(app.game,'Position',[I-C B/2+I-D/2 C D],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
111. hold(app.game,'on')
112. rectangle(app.game,'Position',[I+A B/2+I-D/2 C D],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
113. hold(app.game,'on')
114. rectangle(app.game,'Position',[I B/2+I-F/2 E F],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
115. hold(app.game,'on')
116. rectangle(app.game,'Position',[I+A-E B/2+I-F/2 E F],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
117. hold(app.game,'on')
118. rectangle(app.game,'Position',[I B/2+I-K/2 J K],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
119. hold(app.game,'on')
120. rectangle(app.game,'Position',[I+A-J B/2+I-K/2 J K],'FaceColor','g','EdgeColor','w','LineWidth',1.5);
121. hold(app.game,'on')
122. viscircles(app.game,[I+A/2 I+B/2],H/2,'Color','w','LineWidth',0.75);
123. hold(app.game,'on')
124. plot(app.game,I+G,I+B/2,'w+');
125. hold(app.game,'on')
126. plot(app.game,I+A/2,I+B/2,'w+');
127. hold(app.game,'on')
128. plot(app.game,I+A-G,I+B/2,'w+');
129. hold(app.game,'on')
130. line(app.game,[I+A/2 I+A/2],[I+B I],'Color','w','LineWidth',1);
131. end
132. function results2 = draw\_whole(app,Ball,A1,A1\_a,A2,A2\_a,A3,A3\_a,A4,A4\_a,B1,B1\_a,B2,B2\_a,B3,B3\_a,B4,B4\_a)
133. Ball\_pos(1)=Ball(1);
134. Ball\_pos(2)=Ball(2);
135. A1\_pos(1)=A1(1);
136. A1\_pos(2)=A1(2);
137. A2\_pos(1)=A2(1);
138. A2\_pos(2)=A2(2);
139. A3\_pos(1)=A3(1);
140. A3\_pos(2)=A3(2);
141. A4\_pos(1)=A4(1);
142. A4\_pos(2)=A4(2);
143. B1\_pos(1)=B1(1);
144. B1\_pos(2)=B1(2);
145. B2\_pos(1)=B2(1);
146. B2\_pos(2)=B2(2);
147. B3\_pos(1)=B3(1);
148. B3\_pos(2)=B3(2);
149. B4\_pos(1)=B4(1);
150. B4\_pos(2)=B4(2);
151. theta\_A1=A1\_a;
152. theta\_A2=A2\_a;
153. theta\_A3=A3\_a;
154. theta\_A4=A4\_a;
155. theta\_B1=B1\_a;
156. theta\_B2=B2\_a;
157. theta\_B3=B3\_a;
158. theta\_B4=B4\_a;
159. pitch(app);
160. hold(app.game,'on')
161. draw\_Ball(app,Ball\_pos(1),Ball\_pos(2));
162. draw\_A(app,A1\_pos(1),A1\_pos(2),theta\_A1);
163. draw\_A(app,A2\_pos(1),A2\_pos(2),theta\_A2);
164. draw\_A(app,A3\_pos(1),A3\_pos(2),theta\_A3);
165. draw\_A(app,A4\_pos(1),A4\_pos(2),theta\_A4);
166. draw\_B(app,B1\_pos(1),B1\_pos(2),theta\_B1);
167. draw\_B(app,B2\_pos(1),B2\_pos(2),theta\_B2);
168. draw\_B(app,B3\_pos(1),B3\_pos(2),theta\_B3);
169. draw\_B(app,B4\_pos(1),B4\_pos(2),theta\_B4);
170. end
171. function results2 =draw\_Ball(app,x,y)
172. Ball\_pos(1)=x;
173. Ball\_pos(2)=y;
174. viscircles(app.game,[Ball\_pos(1) Ball\_pos(2)],0.143/2,'Color','black','LineWidth',0.75);
175. hold(app.game,'on')
176. end
177. function results4 =draw\_A(app,x,y,angle)
178. A1\_pos(1)=x;
179. A1\_pos(2)=y;
180. theta\_A1=angle;
181. viscircles(app.game,[A1\_pos(1) A1\_pos(2)],0.2145,'Color','r','LineWidth',0.75);
182. hold(app.game,'on')
183. line(app.game,[A1\_pos(1) A1\_pos(1)-0.2145\*cos(theta\_A1+pi/3)],[A1\_pos(2) A1\_pos(2)-0.2145\*sin(theta\_A1+pi/3)],'Color','r','LineWidth',1);
184. hold(app.game,'on')
185. line(app.game,[A1\_pos(1) A1\_pos(1)-0.2145\*cos(theta\_A1-pi/3)],[A1\_pos(2) A1\_pos(2)-0.2145\*sin(theta\_A1-pi/3)],'Color','r','LineWidth',1);
186. end
187. function results5 =draw\_B(app,x,y,angle)
188. B1\_pos(1)=x;
189. B1\_pos(2)=y;
190. theta\_B1=angle;
191. viscircles(app.game,[B1\_pos(1) B1\_pos(2)],0.2145,'Color','b','LineWidth',0.75);
192. hold(app.game,'on')
193. line(app.game,[B1\_pos(1) B1\_pos(1)-0.2145\*cos(theta\_B1+pi/3)],[B1\_pos(2) B1\_pos(2)-0.2145\*sin(theta\_B1+pi/3)],'Color','b','LineWidth',1);
194. hold(app.game,'on')
195. line(app.game,[B1\_pos(1) B1\_pos(1)-0.2145\*cos(theta\_B1-pi/3)],[B1\_pos(2) B1\_pos(2)-0.2145\*sin(theta\_B1-pi/3)],'Color','b','LineWidth',1);
196. hold(app.game,'on')
197. end

200. function timer\_init3(app)
201. app.Timer\_id3 = timer;
202. app.Timer\_id3.StartDelay = 0.01; % Delay time for the start to take effect
203. app.Timer\_id3.Period = 1.0;% Period
204. app.Timer\_id3.ExecutionMode = 'fixedSpacing';
205. app.Timer\_id3.TimerFcn = @(~, ~) updateTime(app);
206. end
207. function updateTime(app)
208. time = datestr(now,"mmmm dd, yyyy HH:MM:SS AM");
209. app.datetime.Value = time;
210. end
212. function timer\_init4(app)
213. app.Timer\_id4 = timer;
214. app.Timer\_id4.StartDelay = 0.01;
215. app.Timer\_id4.Period = 1.0;
216. app.Timer\_id4.ExecutionMode = 'fixedSpacing';
217. app.Timer\_id4.TimerFcn = @(~, ~) timershow1\_(app);
218. end
219. function timershow1\_(app)
220. %Assisting in executing tasks for the timer
221. app.timer1\_.Value = app.timer1\_.Value +1;
222. end
223. function timer\_init5(app)
224. app.Timer\_id5 = timer;
225. app.Timer\_id5.StartDelay = 0.01;
226. app.Timer\_id5.Period = 1.0;
227. app.Timer\_id5.ExecutionMode = 'fixedSpacing';
228. app.Timer\_id5.TimerFcn = @(~, ~) timershow1(app);
229. end
230. function timershow1(app)
231. %Executing tasks for the timer
232. app.timer1.Value=second\_change(app,app.timer1\_.Value);
233. end
234. function Output = second\_change(app,num) % Reference: https://blog.csdn.net/fair\_li/article/details/118331110
235. hour = floor(num/3600);              % floor: Flooring or rounding down
236. minute = floor(mod(num,3600)/60);  % mod： Calculating the remainder
237. second = num - 3600\*hour - 60\*minute;
238. if hour < 10
239. hour = ['0',mat2str(hour)];      % mat2str：Converting a double to a string
240. else
241. hour = mat2str(hour);
242. end
244. if minute < 10
245. minute = ['0',mat2str(minute)];
246. else
247. minute = mat2str(minute);
248. end
250. if second < 10
251. second = ['0',mat2str(second)];
252. else
253. second = mat2str(second);
254. end
256. Output = [hour,':',minute,':',second];
258. end
259. end
261. % Callbacks that handle component events
262. methods (Access = private)
263. % Code that executes after component creation
264. function startupFcn(app)
265. timer\_init1(app);
266. %app.point.Value="00    :    00";
267. app.point.Value="00:00";
269. timer\_init2(app);
270. draw\_whole(app,[5.5 4],[4.5 3],pi,[4.5 4],pi,[4.5 5],pi,[0.5 4],pi,[6.5 3],0,[6.5 4],0,[6.5 5],0,[10.5 4],0);
272. timer\_init3(app);
273. time =datestr(now,"mmmm dd, yyyy HH:MM:SS AM");
274. app.datetime.Value = time;
276. timer\_init4(app);
277. app.timer1\_.Value=0;
279. timer\_init5(app);
280. app.timer1.Value=datestr(0,'HH:MM:SS');
281. app.Ball.Value="5.5 4";
282. app.A1.Value="4 3";
283. app.A2.Value="4.5 4";
284. app.A3.Value="4.5 5";
285. app.A4.Value="0.5 4";
286. app.B1.Value="6.5 3";
287. app.B2.Value="6.5 4";
288. app.B3.Value="6.5 5";
289. app.B4.Value="10.5 4";
291. end
292. % Button pushed function: PlayButton
293. function PlayButtonPushed(app, event)
294. global g\_A1\_pos;
295. global g\_A2\_pos;
296. global g\_A3\_pos;
297. global g\_A4\_pos;
298. global g\_B1\_pos;
299. global g\_B2\_pos;
300. global g\_B3\_pos;
301. global g\_B4\_pos;
302. global g\_Ball\_pos;
303. g\_Ball\_pos=str2num(app.Ball.Value);
304. g\_A1\_pos=str2num(app.A1.Value);
305. g\_A2\_pos=str2num(app.A2.Value);
306. g\_A3\_pos=str2num(app.A3.Value);
307. g\_A4\_pos=str2num(app.A4.Value);
309. g\_B1\_pos=str2num(app.B1.Value);
310. g\_B2\_pos=str2num(app.B2.Value);
311. g\_B3\_pos=str2num(app.B3.Value);
312. g\_B4\_pos=str2num(app.B4.Value);
314. start(app.Timer\_id1);
315. start(app.Timer\_id2);
316. start(app.Timer\_id3);
317. start(app.Timer\_id4);
318. start(app.Timer\_id5);
319. end
320. % Button pushed function: StopButton
321. function StopButtonPushed(app, event)
322. stop(app.Timer\_id1);
323. stop(app.Timer\_id2);
324. stop(app.Timer\_id3);
325. stop(app.Timer\_id4);
326. stop(app.Timer\_id5);
327. delete(app.Timer\_id1);
328. delete(app.Timer\_id2);
329. delete(app.Timer\_id3);
330. delete(app.Timer\_id4);
331. delete(app.Timer\_id5);
332. end
333. % Close request function: UIFigure
334. function UIFigureCloseRequest(app, event)
335. %             stop(app.Timer\_id1);
336. %             stop(app.Timer\_id2);
337. %             stop(app.Timer\_id3);
338. %             stop(app.Timer\_id4);
339. %             stop(app.Timer\_id5);
340. %             delete(app.Timer\_id1);
341. %             delete(app.Timer\_id2);
342. %             delete(app.Timer\_id3);
343. %             delete(app.Timer\_id4);
344. %             delete(app.Timer\_id5);
345. delete(app);
346. %system('taskkill /F /IM matlab.exe');
347. end
348. % Button pushed function: ResetButton
349. function ResetButtonPushed(app, event)
350. global g\_A1\_pos;
351. global g\_A2\_pos;
352. global g\_A3\_pos;
353. global g\_A4\_pos;
354. global g\_B1\_pos;
355. global g\_B2\_pos;
356. global g\_B3\_pos;
357. global g\_B4\_pos;
358. global g\_Ball\_pos;
359. g\_Ball\_pos=str2num(app.Ball.Value);
360. g\_A1\_pos=str2num(app.A1.Value);
361. g\_A2\_pos=str2num(app.A2.Value);
362. g\_A3\_pos=str2num(app.A3.Value);
363. g\_A4\_pos=str2num(app.A4.Value);
365. g\_B1\_pos=str2num(app.B1.Value);
366. g\_B2\_pos=str2num(app.B2.Value);
367. g\_B3\_pos=str2num(app.B3.Value);
368. g\_B4\_pos=str2num(app.B4.Value);
369. end
370. end
371. % Component initialization
372. methods (Access = private)
373. % Create UIFigure and components
374. function createComponents(app)
375. % Get the file path for locating images
376. pathToMLAPP = fileparts(mfilename('fullpath'));
377. % Create UIFigure and hide until all components are created
378. app.UIFigure = uifigure('Visible', 'off');
379. app.UIFigure.Color = [0.9412 0.9412 0.9412];
380. app.UIFigure.Position = [100 100 726 633];
381. app.UIFigure.Name = 'MATLAB App';
382. app.UIFigure.CloseRequestFcn = createCallbackFcn(app, @UIFigureCloseRequest, true);
383. app.UIFigure.Scrollable = 'on';
384. app.UIFigure.HandleVisibility = 'on';
385. % Create game
386. app.game = uiaxes(app.UIFigure);
387. app.game.XLim = [0 11];
388. app.game.YLim = [0 8];
389. app.game.XTick = [0 1 2 3 4 5 6 7 8 9 10 11];
390. app.game.XTickLabel = {'0'; '1'; '2'; '3'; '4'; '5'; '6'; '7'; '8'; '9'; '10'; '11'};
391. app.game.YTick = [0 1 2 3 4 5 6 7 8];
392. app.game.YTickLabel = {'0'; '1'; '2'; '3'; '4'; '5'; '6'; '7'; '8'};
393. app.game.Position = [87 148 547 382];
394. % Create Image4
395. app.Image4 = uiimage(app.UIFigure);
396. app.Image4.ScaleMethod = 'stretch';
397. app.Image4.Position = [627 158 100 373];
398. app.Image4.ImageSource = fullfile(pathToMLAPP, 'R1.png');
399. % Create Image3
400. app.Image3 = uiimage(app.UIFigure);
401. app.Image3.ScaleMethod = 'stretch';
402. app.Image3.Position = [0 136 101 394];
403. app.Image3.ImageSource = fullfile(pathToMLAPP, 'L.png');
404. % Create Image2
405. app.Image2 = uiimage(app.UIFigure);
406. app.Image2.ScaleMethod = 'stretch';
407. app.Image2.Position = [1 2 726 168];
408. app.Image2.ImageSource = fullfile(pathToMLAPP, 'A2.png');
409. % Create Image
410. app.Image = uiimage(app.UIFigure);
411. app.Image.ScaleMethod = 'stretch';
412. app.Image.Position = [0 529 727 108];
413. app.Image.ImageSource = fullfile(pathToMLAPP, 'B2.png');
414. % Create NAOBlue
415. app.NAOBlue = uiimage(app.UIFigure);
416. app.NAOBlue.Position = [627 215 100 202];
417. app.NAOBlue.ImageSource = 'blueNAORobot.png';
418. % Create NAORed
419. app.NAORed = uiimage(app.UIFigure);
420. app.NAORed.Position = [1 204 100 226];
421. app.NAORed.ImageSource = 'redNAORobot.png';
422. % Create PlayButton
423. app.PlayButton = uibutton(app.UIFigure, 'push');
424. app.PlayButton.ButtonPushedFcn = createCallbackFcn(app, @PlayButtonPushed, true);
425. app.PlayButton.Icon = 'run.png';
426. app.PlayButton.BackgroundColor = [1 1 1];
427. app.PlayButton.Position = [257 544 100 22];
428. app.PlayButton.Text = 'Play';
429. % Create FootballLogo
430. app.FootballLogo = uiimage(app.UIFigure);
431. app.FootballLogo.Position = [337 109 47 40];
432. app.FootballLogo.ImageSource = 'footballBall.png';
433. % Create TimerLogo
434. app.TimerLogo = uiimage(app.UIFigure);
435. app.TimerLogo.Position = [64 535 50 40];
436. app.TimerLogo.ImageSource = 'time.png';
437. % Create ScoreLogo
438. app.ScoreLogo = uiimage(app.UIFigure);
439. app.ScoreLogo.Position = [518 535 50 40];
440. app.ScoreLogo.ImageSource = 'scores.png';
441. % Create StopButton
442. app.StopButton = uibutton(app.UIFigure, 'push');
443. app.StopButton.ButtonPushedFcn = createCallbackFcn(app, @StopButtonPushed, true);
444. app.StopButton.Icon = 'Stop Button.png';
445. app.StopButton.BackgroundColor = [1 1 1];
446. app.StopButton.Position = [396 544 100 22];
447. app.StopButton.Text = 'Stop';
448. % Create ResetButton
449. app.ResetButton = uibutton(app.UIFigure, 'push');
450. app.ResetButton.ButtonPushedFcn = createCallbackFcn(app, @ResetButtonPushed, true);
451. app.ResetButton.Icon = fullfile(pathToMLAPP, 'Reset-Settings.png');
452. app.ResetButton.BackgroundColor = [1 1 1];
453. app.ResetButton.Position = [157 114 100 23];
454. app.ResetButton.Text = 'Reset';
455. % Create A1EditFieldLabel
456. app.A1EditFieldLabel = uilabel(app.UIFigure);
457. app.A1EditFieldLabel.HorizontalAlignment = 'right';
458. app.A1EditFieldLabel.Position = [134 76 25 22];
459. app.A1EditFieldLabel.Text = 'A1';
460. % Create A1
461. app.A1 = uieditfield(app.UIFigure, 'text');
462. app.A1.Tag = 'A1';
463. app.A1.HorizontalAlignment = 'center';
464. app.A1.BackgroundColor = [0.9412 0.9412 0.9412];
465. app.A1.Position = [174 76 60 22];
466. % Create A2EditFieldLabel
467. app.A2EditFieldLabel = uilabel(app.UIFigure);
468. app.A2EditFieldLabel.HorizontalAlignment = 'right';
469. app.A2EditFieldLabel.Position = [251 76 25 22];
470. app.A2EditFieldLabel.Text = 'A2';
471. % Create A2
472. app.A2 = uieditfield(app.UIFigure, 'text');
473. app.A2.Tag = 'A2';
474. app.A2.HorizontalAlignment = 'center';
475. app.A2.BackgroundColor = [0.9412 0.9412 0.9412];
476. app.A2.Position = [291 76 60 22];
477. % Create A3EditFieldLabel
478. app.A3EditFieldLabel = uilabel(app.UIFigure);
479. app.A3EditFieldLabel.HorizontalAlignment = 'right';
480. app.A3EditFieldLabel.Position = [363 76 25 22];
481. app.A3EditFieldLabel.Text = 'A3';
482. % Create A3
483. app.A3 = uieditfield(app.UIFigure, 'text');
484. app.A3.Tag = 'A3';
485. app.A3.HorizontalAlignment = 'center';
486. app.A3.BackgroundColor = [0.9412 0.9412 0.9412];
487. app.A3.Position = [403 76 60 22];
488. % Create A4EditFieldLabel
489. app.A4EditFieldLabel = uilabel(app.UIFigure);
490. app.A4EditFieldLabel.HorizontalAlignment = 'right';
491. app.A4EditFieldLabel.Position = [478 76 25 22];
492. app.A4EditFieldLabel.Text = 'A4';
493. % Create A4
494. app.A4 = uieditfield(app.UIFigure, 'text');
495. app.A4.Tag = 'A4';
496. app.A4.HorizontalAlignment = 'center';
497. app.A4.BackgroundColor = [0.9412 0.9412 0.9412];
498. app.A4.Position = [518 76 60 22];
499. % Create datetime
500. app.datetime = uieditfield(app.UIFigure, 'text');
501. app.datetime.BackgroundColor = [0.9412 0.9412 0.9412];
502. app.datetime.Position = [547 2 170 22];
503. % Create timer1\_
504. app.timer1\_ = uieditfield(app.UIFigure, 'numeric');
505. app.timer1\_.HorizontalAlignment = 'center';
506. app.timer1\_.Visible = 'off';
507. app.timer1\_.Position = [113 494 100 22];
508. % Create B1Label
509. app.B1Label = uilabel(app.UIFigure);
510. app.B1Label.HorizontalAlignment = 'right';
511. app.B1Label.Position = [134 44 25 22];
512. app.B1Label.Text = 'B1';
513. % Create B1
514. app.B1 = uieditfield(app.UIFigure, 'text');
515. app.B1.Tag = 'B1';
516. app.B1.HorizontalAlignment = 'center';
517. app.B1.BackgroundColor = [0.9412 0.9412 0.9412];
518. app.B1.Position = [174 44 60 22];
519. % Create B2Label
520. app.B2Label = uilabel(app.UIFigure);
521. app.B2Label.HorizontalAlignment = 'right';
522. app.B2Label.Position = [251 44 25 22];
523. app.B2Label.Text = 'B2';
524. % Create B2
525. app.B2 = uieditfield(app.UIFigure, 'text');
526. app.B2.Tag = 'B2';
527. app.B2.HorizontalAlignment = 'center';
528. app.B2.BackgroundColor = [0.9412 0.9412 0.9412];
529. app.B2.Position = [291 44 60 22];
530. % Create B3Label
531. app.B3Label = uilabel(app.UIFigure);
532. app.B3Label.HorizontalAlignment = 'right';
533. app.B3Label.Position = [363 44 25 22];
534. app.B3Label.Text = 'B3';
535. % Create B3
536. app.B3 = uieditfield(app.UIFigure, 'text');
537. app.B3.Tag = 'B3';
538. app.B3.HorizontalAlignment = 'center';
539. app.B3.BackgroundColor = [0.9412 0.9412 0.9412];
540. app.B3.Position = [403 44 60 22];
541. % Create B4Label
542. app.B4Label = uilabel(app.UIFigure);
543. app.B4Label.HorizontalAlignment = 'right';
544. app.B4Label.Position = [478 44 25 22];
545. app.B4Label.Text = 'B4';
546. % Create B4
547. app.B4 = uieditfield(app.UIFigure, 'text');
548. app.B4.Tag = 'B4';
549. app.B4.HorizontalAlignment = 'center';
550. app.B4.BackgroundColor = [0.9412 0.9412 0.9412];
551. app.B4.Position = [518 44 60 22];
552. % Create BallLabel
553. app.BallLabel = uilabel(app.UIFigure);
554. app.BallLabel.HorizontalAlignment = 'right';
555. app.BallLabel.Position = [479 115 25 22];
556. app.BallLabel.Text = 'Ball';
557. % Create Ball
558. app.Ball = uieditfield(app.UIFigure, 'text');
559. app.Ball.Tag = 'Ball';
560. app.Ball.HorizontalAlignment = 'center';
561. app.Ball.BackgroundColor = [0.9412 0.9412 0.9412];
562. app.Ball.Position = [519 115 60 22];
563. % Create timer1
564. app.timer1 = uieditfield(app.UIFigure, 'text');
565. app.timer1.HorizontalAlignment = 'center';
566. app.timer1.BackgroundColor = [0.9412 0.9412 0.9412];
567. app.timer1.Position = [113 544 100 22];
568. % Create point
569. app.point = uieditfield(app.UIFigure, 'text');
570. app.point.HorizontalAlignment = 'center';
571. app.point.BackgroundColor = [0.9412 0.9412 0.9412];
572. app.point.Position = [567 544 100 22];
573. % Create Title
574. app.Title = uilabel(app.UIFigure);
575. app.Title.HorizontalAlignment = 'center';
576. app.Title.FontName = 'Showcard Gothic';
577. app.Title.FontSize = 24;
578. app.Title.FontAngle = 'italic';
579. app.Title.FontColor = [1 1 1];
580. app.Title.Position = [66 592 615 31];
581. app.Title.Text = 'RoboCup - Robot Football Simulation - Team 2';
582. % Show the figure after all components are created
583. app.UIFigure.Visible = 'on';
584. end
585. end
586. % App creation and deletion
587. methods (Access = public)
588. % Construct app
589. function app = app1
590. % Create UIFigure and components
591. createComponents(app)
592. % Register the app with App Designer
593. registerApp(app, app.UIFigure)
594. % Execute the startup function
595. runStartupFcn(app, @startupFcn)
596. if nargout == 0
597. clear app
598. end
599. end
600. % Code that executes before app deletion
601. function delete(app)
602. % Delete UIFigure when app is deleted
603. delete(app.UIFigure)
604. end
605. end
606. end