**实验七 机器人足球队的完整设计**

**班级：**计算机科学与技术21-1班  **实验日期:**2022.11.11

**小组成员：**

2021218067 王帅杨

2021218070 冀国鹏

2021218071 韩浩天

**一、实验目的**

根据以上6个实验，大家应该可以设计出一支具有一定战术策略的机器人足球队，本实验就是要求大家按照小组的方法完成一支仿真足球队伍。

**二、实验设备**

硬件环境：PC机

软件环境：操作系统linux

**三、实验内容**

完成一支完整的仿真机器人足球队

**四、球队设计**

1. 为了完成球队代码的设计，我们小组设计了几个函数，在BasicPlayer.h文件中声明闪避带球、获取角球开球点、获取界外球开球点、向球门空隙较大一侧射门、按周期上下偏移射门函数。

|  |
| --- |
| SoccerCommand dribble\_Dodge(VecPosition tar, VecPosition posAgent);//闪避带球  VecPosition pointKickToC();//角球开球点  VecPosition pointKickToK();//界外球开球点  SoccerCommand maxAngShoot(VecPosition posAgent);//夹角判定射门  SoccerCommand maxAngShoot1(VecPosition posAgent);//普通射门 |

在BasicPlayer.cpp文件中完成上述函数的定义。代码如下：

//新增加的代码

/\*以下这个函数可以向空隙较大的一方射门，采用夹角比较

\* 参数：自己位置

\* 返回值：command型

\* 需要在BasicPlayer.h里添加内容

\*/

//朝较大的夹角射门

|  |
| --- |
| SoccerCommand BasicPlayer::maxAngShoot( VecPosition posAgent )  {  SoccerCommand soc;  VecPosition posGoalie;  AngDeg ang\_goalie, angup, angdown;    posGoalie = WM->getGlobalPosition(WM->getOppGoalieType());  ang\_goalie = (posGoalie - posAgent).getDirection();  angup = (VecPosition(52.5,6.0) - posAgent).getDirection();  angdown = (VecPosition(52.5,-6.0) - posAgent).getDirection();    if(fabs(angup - ang\_goalie) > fabs(angdown - ang\_goalie))  {  soc = kickTo(VecPosition(52.5, -6.5),2.7);  }  else  {  soc = kickTo(VecPosition(52.5, 6.5),2.7);  }    return soc;  } |

//普通射门

|  |
| --- |
| SoccerCommand BasicPlayer::maxAngShoot1( VecPosition posAgent )  {  SoccerCommand soc;  VecPosition posGoal(PITCH\_LENGTH/2.0,  (-1 + 2\*(WM->getCurrentCycle()%2)) \*  0.4 \* SS->getGoalWidth());  soc=kickTo( posGoal,SS->getBallSpeedMax());// 朝球门方向将球以最大力度  return soc;  } |

//闪避带球

|  |
| --- |
| SoccerCommand BasicPlayer::dribble\_Dodge(VecPosition tar, VecPosition posAgent)  {  SoccerCommand soc;  AngDeg ang = ( tar - posAgent ).getDirection();  soc = dribble(ang, DRIBBLE\_FAST );    if( WM->isOpponentAtAngleEx( ang - 45, ang , 7 ) )  {  ang += 45;  soc = dribble( ang , DRIBBLE\_FAST );  }  else if( WM->isOpponentAtAngleEx( ang, ang + 45 , 7 ) )  {  ang -= 45;  soc = dribble( ang , DRIBBLE\_FAST );  }  return soc;  } |

//计算角球应该踢向的点

|  |
| --- |
| VecPosition BasicPlayer::pointKickToC()  {  VecPosition posBall, posKickTo;  double x, y;  posBall = WM->getBallPos();  x = posBall.getX();  y = posBall.getY();    posKickTo.setX(- x / fabs(x) \* 5 + x );  posKickTo.setY(- y / fabs(y) \* 12 + y );    return posKickTo;  } |

/\*

\*此函数返回界外球应该踢向的点

\*策略是把球尽可能往敌方半场踢

\*/

//计算界外球应该踢向的点

|  |
| --- |
| VecPosition BasicPlayer::pointKickToK()  {  VecPosition posBall, posKickTo;  double x, y;  posBall = WM->getBallPos();  x = posBall.getX();  y = posBall.getY();    posKickTo.setX( x + 7 );  posKickTo.setY(- y / fabs(y) \* 12 + y );    return posKickTo;  } |

1. 为了使队伍中球员的带球、传球和射门技术更强，在WorldModel.h文件中声明了isOpponentAtAngleEx函数，函数接收三个参数，角度一、角度二和一个距离参数，判断俩个角度之间，距离参数范围内是否有对方球员，根据判断情况返回一个bool值。

bool isOpponentAtAngleEx( AngDeg angA, AngDeg angB, double dDist);

该函数的具体实现定义在WorldModel.cpp文件中，代码如下：

/\*这个函数判断在扇区内有没有敌方球员，若有则ture 否则false

\*

\*/

|  |
| --- |
| bool WorldModel::isOpponentAtAngleEx( AngDeg angA, AngDeg angB, double dDist )  {  VecPosition posAgent = getAgentGlobalPosition();  VecPosition posOpp;  AngDeg angOpp;  int iIndex;    for( ObjectT o = iterateObjectStart(iIndex, OBJECT\_SET\_OPPONENTS );  o != OBJECT\_ILLEGAL;  o = iterateObjectNext( iIndex, OBJECT\_SET\_OPPONENTS ))  {  posOpp = getGlobalPosition(o);  angOpp = ( posOpp - posAgent ).getDirection();  if ( angA <= angOpp && angOpp <= angB && posAgent.getDistanceTo( posOpp ) < dDist)  return true;  }  iterateObjectDone( iIndex ) ;  return false;  } |

1. 球队的核心决策代码在PlayerTeams.cpp文件中的deMeer5函数中。

首先需要在deMeer5函数中添加特殊比赛模式情况的应对策略。我方角球模式下，如果球可踢，则将球踢向角球接应点，此处调用了计算角球位置函数pointKickToC。距离球最近的队员跑到球的位置，距离球次近的球员跑向接应点。其他球员策略跑位。

我方界外球模式下，如果球可踢，将球踢向界外球接应点。距离球最近的球员跑向球的位置。离球次近的球员跑向界外球接应点。其他球员策略跑位。该过程调用了计算界外球接应点函数pointKickToK。

我方任意球模式下，如果球可踢，当球的x坐标大于35，调用朝球门空隙较大一侧射门函数进行射门，否则，将球踢向接应点。若我离球最近，我跑向球的位置。如果我离球次近，则跑向接应点。其他球员策略跑位。

//新增特殊模式代码

|  |
| --- |
| else if( WM->isCornerKickUs())//我方角球  {  if(WM->isBallKickable() )  {  soc = kickTo(pointKickToC(), 2.0);  }  else if(WM->getFastestInSetTo( OBJECT\_SET\_TEAMMATES, OBJECT\_BALL, &iTmp)  == WM->getAgentObjectType())  {  soc = moveToPos(posBall, PS->getPlayerWhenToTurnAngle());  }  else if(WM->getSecondClosestInSetTo(OBJECT\_SET\_TEAMMATES, OBJECT\_BALL)  == WM->getAgentObjectType())  {  soc = moveToPos(pointKickToC(), PS->getPlayerWhenToTurnAngle());  }  else  {  soc = moveToPos(WM->getStrategicPosition(), PS->getPlayerWhenToTurnAngle());  }    ACT->putCommandInQueue( soc ); // 放入命令队列  ACT->putCommandInQueue( turnNeckToObject( OBJECT\_BALL, soc ) );  }    else if( WM->isKickInUs() )//我方界外球  {  if( WM->isBallKickable() )  {  soc = kickTo(pointKickToK(), 2.0);  }  else if(WM->getFastestInSetTo( OBJECT\_SET\_TEAMMATES, OBJECT\_BALL, &iTmp)  == WM->getAgentObjectType())  {  soc = moveToPos(posBall, PS->getPlayerWhenToTurnAngle());  }  else if(WM->getSecondClosestInSetTo(OBJECT\_SET\_TEAMMATES, OBJECT\_BALL)  == WM->getAgentObjectType())  {  soc = moveToPos(pointKickToK(), PS->getPlayerWhenToTurnAngle());  }  else  {  soc = moveToPos(WM->getStrategicPosition(), PS->getPlayerWhenToTurnAngle());  }    ACT->putCommandInQueue( soc ); // 放入命令队列  ACT->putCommandInQueue( turnNeckToObject( OBJECT\_BALL, soc ) );  }    else if( WM->isFreeKickUs() )//我方任意球  {  if( WM->isBallKickable() )  {  if(WM->getBallPos().getX() > 35)//x大于35直接射门  {  soc = maxAngShoot( posAgent );  }  else  {  soc = kickTo(pointKickToK(), 2.0);  }  }  else if(WM->getFastestInSetTo( OBJECT\_SET\_TEAMMATES, OBJECT\_BALL, &iTmp)  == WM->getAgentObjectType())  {  soc = moveToPos(posBall, PS->getPlayerWhenToTurnAngle());  }  else if(WM->getSecondClosestInSetTo(OBJECT\_SET\_TEAMMATES, OBJECT\_BALL)  == WM->getAgentObjectType())  {  soc = moveToPos(pointKickToK(), PS->getPlayerWhenToTurnAngle());  }  else  {  soc = moveToPos(WM->getStrategicPosition(), PS->getPlayerWhenToTurnAngle());  }  ACT->putCommandInQueue( soc ); // 放入命令队列  ACT->putCommandInQueue( turnNeckToObject( OBJECT\_BALL, soc ) );  } |

1. 在正常比赛模式下，首先设置球队进攻整型为334，并初始化soc命令对象。判断球的位置信息可信度，如果球的信息可信度较低，则寻找球，并更新世界模型的信息。在球可踢的情况下，如果我方开球根据球门和我的角度，在一定范围内判断是否有对方球员，以选择踢球方向。当球在我方半场，X的距离小于-25时，2、5、6、7号球员将球踢向安全区，10号、11号将球传向（10，25）或（10，-25）位置。其他球员的话，根据纵坐标，将球传向（posAngent.getX()+20,25）或（posAngent.getX()+20,-25）位置。如果球的横坐标在-25到-5之间，2号、7号将球踢向（5，25）或（5，-25）的位置，6号、7号将球踢向（15，25）或（15，-25）的位置。10号球员捡球踢向（20，30）的位置，11号球员将球踢向（20，-30）的位置。其他纵坐标大于0的球员把球踢向（posAngent.getX()+20,25）位置，纵坐标小于0的将球踢向（posAngent.getX()+20,-25）位置。如果球的横坐标在-5到10之间， 10号球员将球踢向（35，30）的位置，11号球员将球踢向（35，-30）的位置。其他纵坐标大于0的球员把球传给10号球员，纵坐标小于0的球员将球传给11号球员。当球的横坐标位置在10到45之间，11号球员判断自己身体6米范围内是否有球员，如果身边多于两个人且自己不在对方的禁区内，此时判断8号和6号球员距离球门的远近，将球传给离球门近的队友。接到球的球员离球门不太远的时候，使用向球门最大空隙侧射门的函数进行射门。如果进入对方禁区，判断自己和对方球门90度方向内有无对方球员，如果有则传给最近的队友，否则按周期偏移射门。其他球员按照进攻阵营跑位。10号与11号球员对称，决策类似。接应手6号、7号球员和突击手8号9号球员根据自己的实际情况选择传球、带球或者射门。当球的横坐标位置大于45时，8、9、10、11号球员依据自生的实际情况选择传球、带球或射门。总之，决策的核心就是根据当前球在场上的横坐标位置，对球员进行不同的踢球策略。

|  |
| --- |
| else if ( WM->isBallKickable()) // if kickable // 如果球已知，而且当前球在我脚下(可踢)  {  if( WM->isFreeKickUs() )//带球绕人  {  double ang = (VecPosition(52.5,0)-posAgent).getDirection();  if ( WM->isOpponentAtAngleEx(ang-45, ang, 6) )  ang+=45;  else if ( WM->isOpponentAtAngleEx(ang,ang+45,6) )  ang-=45;  soc = kickTo ( ang , SS->getBallSpeedMax() );  }    else if ( WM->getBallPos().getX() < -25 )//在本方半场  {  if( WM->getPlayerNumber() == 2 || WM->getPlayerNumber() == 5 )  {  soc = clearBall( CLEAR\_BALL\_DEFENSIVE );  }    if( WM->getPlayerNumber() == 6 || WM->getPlayerNumber() == 7 )  {  soc = clearBall( CLEAR\_BALL\_DEFENSIVE );  }    if( WM->getPlayerNumber() == 10 || WM->getPlayerNumber() == 11 )  {  if( WM->getPlayerNumber() == 10 )  soc = directPass( VecPosition(10, 25), PASS\_FAST);  else  soc = directPass( VecPosition(10, -25), PASS\_FAST);  }    else  {  if( posAgent.getY() > 0)  soc = directPass( VecPosition( posAgent.getX() + 20, 25), PASS\_FAST);  else  soc = directPass( VecPosition( posAgent.getX() + 20, -25), PASS\_FAST);  }    }    else if ( WM->getBallPos().getX() > -25 && WM->getBallPos().getX() < -5 )//在本方半场  {  if( WM->getPlayerNumber() == 2 || WM->getPlayerNumber() == 5 )  {  if( WM->getPlayerNumber() == 2 )  soc = directPass( VecPosition(5, 25), PASS\_FAST);  else  soc = directPass( VecPosition(5, -25), PASS\_FAST);  soc = clearBall( CLEAR\_BALL\_DEFENSIVE );  }    if( WM->getPlayerNumber() == 6 || WM->getPlayerNumber() == 7 )  {  if( WM->getPlayerNumber() == 7 )  soc = directPass( VecPosition(15, 25), PASS\_FAST);  else  soc = directPass( VecPosition(15, -25), PASS\_FAST);  soc = clearBall( CLEAR\_BALL\_DEFENSIVE );  }    if( WM->getPlayerNumber() == 10 || WM->getPlayerNumber() == 11 )  {  if( WM->getPlayerNumber() == 10 )  soc = directPass( VecPosition(20, 30), PASS\_FAST);  else  soc = directPass( VecPosition(20, -30), PASS\_FAST);  }    else  {  if( posAgent.getY() > 0)  soc = directPass( VecPosition( posAgent.getX() + 20, 25), PASS\_FAST);  else  soc = directPass( VecPosition( posAgent.getX() + 20, -25), PASS\_FAST);  }    }    else if ( WM->getBallPos().getX() < 10 && WM->getBallPos().getX() >= -5 )//在本方半场  {  if( WM->getPlayerNumber() == 10 || WM->getPlayerNumber() == 11 )  {  if( WM->getPlayerNumber() == 10 )  soc = dribble\_Dodge( VecPosition(35, 30), posAgent);  else  soc = dribble\_Dodge( VecPosition(35, -30), posAgent);  }    else  {  if(WM->getBallPos().getY() < 0)  soc = leadingPass( OBJECT\_TEAMMATE\_11 , 1.5 );  else  soc = leadingPass( OBJECT\_TEAMMATE\_10 , 1.5 );  }    }    else if( WM->getBallPos().getX() >= 10 && WM->getBallPos().getX() <= 45)//进入后半场  {  if( WM->getPlayerNumber() == 11 )//边路选手11  {  Circle cir1( posAgent, 6 );  int num=WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);    if(num >= 2 && !WM->isInTheirPenaltyArea( WM->getBallPos()) )//有人阻拦且没到禁区  {  ObjectT temp = OBJECT\_TEAMMATE\_8;    if( WM->getGlobalPosition(temp).getX() >  WM->getGlobalPosition(OBJECT\_TEAMMATE\_6).getX() &&  WM->getGlobalPosition( temp ).getX() >  posAgent.getX() - 10 ) //8号比6靠前,且不远  {  soc = throughPass( temp , posGoal );  }    else if(posAgent.getDistanceTo( WM->getClosestInSetTo(OBJECT\_SET\_TEAMMATES, posAgent) )  < posAgent.getDistanceTo( WM->getGlobalPosition(OBJECT\_TEAMMATE\_6)) )  {  soc = leadingPass( WM->getClosestInSetTo(OBJECT\_SET\_TEAMMATES , posAgent) ,1 );  }    else if( WM->getGlobalPosition( OBJECT\_TEAMMATE\_6 ).getX() >  posAgent.getX() - 10 ) //6号离得butai远  {  soc = leadingPass( OBJECT\_TEAMMATE\_6 ,1 );  }    else  soc = maxAngShoot1(posAgent);    }    else if( WM->isInTheirPenaltyArea( WM->getBallPos()) && fabs( WM->getBallPos().getY()) < 13 )//进入禁区射门  {  double ang = (VecPosition(52.5,0)-posAgent).getDirection();  if( WM-> isOpponentAtAngleEx(ang-15, ang + 15, 15) )  {  soc = leadingPass( WM->getClosestInSetTo( OBJECT\_SET\_TEAMMATES , posAgent ), 1);  }  else  soc = maxAngShoot( posAgent );  }    else//往攻击阵位移动  {  VecPosition posGoal = WM->getPosOpponentGoal();  VecPosition atkPos( posGoal.getX() - 6, posGoal.getY() - 15 );//攻击阵位    if( WM->getPlayerNumber() == 10 )  {  atkPos.setY( posGoal.getY() + 15 );  }    soc = dribble\_Dodge( atkPos , posAgent );    }  }    else if( WM->getPlayerNumber() == 10 )//边路选手10  {  Circle cir1( posAgent, 6 );  int num=WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);    if(num >= 2 && !WM->isInTheirPenaltyArea( WM->getBallPos()) )//有人阻拦且没到禁区  {  ObjectT temp = OBJECT\_TEAMMATE\_9;    if( WM->getGlobalPosition(temp).getX() >  WM->getGlobalPosition(OBJECT\_TEAMMATE\_7).getX() &&  WM->getGlobalPosition( temp ).getX() >  posAgent.getX() - 10 ) //9号比7靠前,且不远  {  soc = throughPass( temp , posGoal );  }    else if(posAgent.getDistanceTo( WM->getClosestInSetTo(OBJECT\_SET\_TEAMMATES, posAgent) )  < posAgent.getDistanceTo( WM->getGlobalPosition(OBJECT\_TEAMMATE\_7)) )  {  soc = leadingPass( WM->getClosestInSetTo(OBJECT\_SET\_TEAMMATES, posAgent) ,1 );  }    else if( WM->getGlobalPosition( OBJECT\_TEAMMATE\_7 ).getX() >  posAgent.getX() - 10 ) //7号离得不太远  {  soc = leadingPass( OBJECT\_TEAMMATE\_7 ,1 );  }    else  soc = maxAngShoot1(posAgent);  }    else if( WM->isInTheirPenaltyArea( WM->getBallPos()) && fabs( WM->getBallPos().getY()) < 13 )//进入禁区射门  {  double ang = (VecPosition(52.5,0)-posAgent).getDirection();  if( WM-> isOpponentAtAngleEx(ang-15, ang + 15, 15) )  {  soc = leadingPass( WM->getClosestInSetTo( OBJECT\_SET\_TEAMMATES , posAgent ), 1);  }  else  soc = maxAngShoot( posAgent );  }  else//往攻击阵位移动  {  VecPosition posGoal = WM->getPosOpponentGoal();  VecPosition atkPos( posGoal.getX() - 6, posGoal.getY() - 15 );//攻击阵位    if( WM->getPlayerNumber() == 10 )  {  atkPos.setY( posGoal.getY() + 15 );  }    soc = dribble\_Dodge( atkPos , posAgent );  }  }    else if( WM->getPlayerNumber() == 6 )//接应选手6  {  Circle cir1( posAgent, 6 );  int num = WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);  double ang = (VecPosition(52.5,0)-posAgent).getDirection();      if(num >= 2 && WM-> isOpponentAtAngleEx(ang-20, ang + 20, 15))  {  ObjectT temp = OBJECT\_TEAMMATE\_9;    Circle cir1( WM->getGlobalPosition(temp), 6 );  int num1 = WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);  Circle cir2( WM->getGlobalPosition(OBJECT\_TEAMMATE\_8), 6 );  int num2 = WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);    if( num1 > num2 )  soc = leadingPass( OBJECT\_TEAMMATE\_8 , 1);  else if( num1 < num2)  soc = leadingPass( temp , 1);  else  soc = leadingPass( OBJECT\_TEAMMATE\_11 ,1);  }  else if( WM->getBallPos().getX() < 35 )  {  soc = dribble\_Dodge( VecPosition(35, -10) , posAgent );  }    else  soc = maxAngShoot( posAgent );//直接射门  }    else if( WM->getPlayerNumber() == 7 )//接应选手7  {  Circle cir1( posAgent, 6 );  int num = WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);  double ang = (VecPosition(52.5,0)-posAgent).getDirection();    if(num >= 2 && WM-> isOpponentAtAngleEx(ang-20, ang + 20, 15))  {  ObjectT temp = OBJECT\_TEAMMATE\_9;    Circle cir1( WM->getGlobalPosition(temp), 6 );  int num1 = WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);  Circle cir2( WM->getGlobalPosition(OBJECT\_TEAMMATE\_8), 6 );  int num2 = WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);    if( num1 > num2 )  soc = leadingPass( OBJECT\_TEAMMATE\_8 , 1);  else if( num1 < num2 )  soc = leadingPass( temp , 1);    else  soc = leadingPass( OBJECT\_TEAMMATE\_10 ,1);    }  else if( WM->getBallPos().getX() < 35 )  {  soc = dribble\_Dodge( VecPosition(35, 9) , posAgent );  }    else  soc = maxAngShoot( posAgent );//直接射门  }    else if( WM->getPlayerNumber() == 9 || WM->getPlayerNumber() == 8 )//突击手8,9  {  Circle cir1( posAgent, 6 );  int num = WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);    double ang = (VecPosition(52.5,0)-posAgent).getDirection();    if( WM->isOpponentAtAngleEx( ang-15, ang+15, 15) //用来应对球门球 5.14  && WM->getBallPos().getX() > 33  && fabs( WM->getBallPos().getY() ) < 16)  {  soc = maxAngShoot1( posAgent );  }    else if(num >= 3)  {  if(posAgent.getDistanceTo( WM->getGlobalPosition(OBJECT\_TEAMMATE\_6)) //有人抢，回传  < posAgent.getDistanceTo( WM->getGlobalPosition(OBJECT\_TEAMMATE\_7)) )  soc = leadingPass( OBJECT\_TEAMMATE\_6 ,1 );  else  soc = leadingPass( OBJECT\_TEAMMATE\_7 ,1 );  }    else if( WM->getBallPos().getX() < 30 )  soc = dribble\_Dodge( WM->getPosOpponentGoal() , posAgent );  else  soc = maxAngShoot( posAgent );//直接射门  }    else soc = maxAngShoot1( posAgent );  }  else if( WM->getBallPos().getX() >= 45 &&  WM->getBallPos().getX() < 52)//球距离球门比较近了  {  if( WM->getPlayerNumber() == 9 || WM->getPlayerNumber() == 8 )//主要突破选手  {  Circle cir1( posAgent, 6 );  int num=WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);  double ang = (VecPosition(52.5,0)-posAgent).getDirection();    if( WM->isOpponentAtAngleEx( ang-15, ang+15, 15))  {  soc = directPass( WM->getGlobalPosition(  WM->getClosestInSetTo(OBJECT\_SET\_TEAMMATES, posAgent) ), PASS\_FAST);  }  else soc = maxAngShoot( posAgent );  }    else if( WM->getPlayerNumber() == 10 )  {  Circle cir1( posAgent, 3 );  int num=WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);  double ang = (VecPosition(52.5,0)-posAgent).getDirection();  if( WM-> isOpponentAtAngleEx(ang-15, ang +15, 15) )  {  soc = leadingPass( OBJECT\_TEAMMATE\_8 , 1);  }  if(num >= 1)  {  if(WM->getGlobalPosition(OBJECT\_TEAMMATE\_7).getX()  < WM->getGlobalPosition(OBJECT\_TEAMMATE\_9).getX() )  soc = directPass( WM->getGlobalPosition( OBJECT\_TEAMMATE\_9 ), PASS\_FAST);  else  soc = directPass( WM->getGlobalPosition( OBJECT\_TEAMMATE\_7 ), PASS\_FAST);  }  else soc = maxAngShoot1( posAgent );  }    else if( WM->getPlayerNumber() == 11 )  {  Circle cir1( posAgent, 3 );  int num=WM->getNrInSetInCircle(OBJECT\_SET\_OPPONENTS,cir1);  double ang = (VecPosition(52.5,0)-posAgent).getDirection();    if( WM-> isOpponentAtAngleEx(ang-15, ang + 15, 15) )  {  soc = leadingPass( OBJECT\_TEAMMATE\_9 , 1);  }  else if(num >= 1)  {  if(WM->getGlobalPosition(OBJECT\_TEAMMATE\_6).getX()  < WM->getGlobalPosition(OBJECT\_TEAMMATE\_8).getX() )  soc = directPass( WM->getGlobalPosition( OBJECT\_TEAMMATE\_8 ), PASS\_FAST);  else  soc = directPass( WM->getGlobalPosition( OBJECT\_TEAMMATE\_6 ), PASS\_FAST);  }  else soc = maxAngShoot( posAgent );  }    else soc = maxAngShoot1( posAgent );  }  else//默认策略  {  VecPosition posGoal(PITCH\_LENGTH/2.0,  (-1 + 2\*(WM->getCurrentCycle()%2)) \*  0.4 \* SS->getGoalWidth());  soc=kickTo( posGoal,SS->getBallSpeedMax());// 朝球门方向将球以最大力度  }  ACT->putCommandInQueue( soc ); // 放入命令队列  ACT->putCommandInQueue( turnNeckToObject( OBJECT\_BALL, soc ) ); // 把脖子转向球，也就是一直看着球  Log.log( 100, "kick ball" );  } |

1. 我们小组还改善了守门员发球的位置以及将球发给谁的问题，代码如下：

|  |
| --- |
| static const VecPosition posLeftTop( -PITCH\_LENGTH/2.0 +  0.7\*PENALTY\_AREA\_LENGTH, -19.0 );  static const VecPosition posRightTop( -PITCH\_LENGTH/2.0 +  0.7\*PENALTY\_AREA\_LENGTH, +19.0 );  soc = kickTo( WM->getSecondClosestInSetTo(OBJECT\_SET\_TEAMMATES,OBJECT\_TEAMMATE\_1), 2.0 ); |

**五、实验心得和体会**

在这次实验中，我们组将课堂上的知识与实践相结合，在共同的努力下设计了这支球队的代码。

通过反复观看原型的比赛，我们根据每个球员的场上位置和他们的表现分析了原有球队中各个球员的优缺点，探讨了各个队员之间配合的多种可能性。在刚开始上手设计时，我们想到了借用现实中的一些技术，加大侧翼的进攻力度并通过传中组织攻击，但在分析后我们发现有一些战术并不能在2d足球比赛中展现出应有的效果，究其原因就是2d的赛场上没有高度一说，所有的传球只能在地面进行，这代表着球员间大多只能依靠短传，而不能起高球。相应的，我们的球队策略也要依据这一点进行更改。

经过几次的机器人足球实验以及课堂的知识，我们对机器人足球有了一定的了解，在大家的相互配合下最终完成了任务。我们对机器人足球代码设计了解的还可以但是并不够,仍然需要加强这方面知识。在设计球队代码时我们借鉴了老师所给的50道题来进行球队设计，在配合下完成了球队设计。设计过程中有些地方出现的错误找起来较为麻烦，但在经过不断改正后完成了这次球队设计实验，我们每个人都是十分高兴的。

在正式编程实现的过程中也遇到了不少的麻烦，也需要不断地做出改进，每修改一次代码，我都会重新运行一下足球队，进行一场比赛来检验这次更改代码有没有满足自己的预期，实现自己想要的功能以及效果。因为修改和改进的地方太多，我就举几个其中的例子来说明，不过多赘述。通过观察比赛，最让我头痛的就是守门员的发球，很多时候丢球都是因为守门员的发球不当而导致的，守门员很多时候都会把球发给对面，而且是连续几次发给对面，这无疑增加了对面的进攻机会，很多时候发出去的球，自己又守不住对面的攻击，就导致了丢球。这一点对于一场比赛的结果的影响还是很巨大的。因此这一点必须做出修改，守门员的修改花了我们不少的时间，我们一开始考虑让他把球发给最近的队员，但是做出判定后，他的发球是非常奇怪的他会把球发到左边界的位置（我们怀疑在他眼里最近的是coach），因此应该把球发给第二近的队员。其次是发球的位置，在未修改之前他发球的位置前方一般都是有敌人的，这很容易让球发到对方手中，因此我们变更了他发球的位置，让前方有人的概率减小，这也降低了发球失败的概率。总的来说编程的过程是十分不容易的，从开始编程到完成整个球队，我们做出了数不尽次数的修改，测试的比赛也差不多打了近百场，虽然辛苦，但是最后完成了足球队的设计，我们都是十分自豪的。

通过这次实验，提高了我们球队整体设计思维能力并且加深了我们对c++类和代码的理解。值得一提的是球队的编程实际上就是游戏中AI的行为编程，因而也对于我们以后一些可能性的工作打下了基础。通过组内相互间的合作，我们互相查漏补缺，取长补短，都收获了许多。

除了这些，我们也通过这次实验学会了更加熟练地使用linux操作系统，也知道了linux操作系统的各种指令。通过自己的实践，使我们将课上的理论知识与实践相结合，更进一步加深了我们对机器人足球的认识，巩固了我们的理论知识，也提高了实践能力。