**实验五 特殊比赛模式的设计**

**一、实验目的**

1.掌握 Robocup 仿真机器人足球比赛中特殊比赛模式发生的条件；

2.掌握 Robocup 仿真机器人足球比赛特殊比赛模式的规则要求；

3.了解 Robocup 仿真机器人足球比赛特殊比赛模式的战术设计思想；进一步熟悉 WorldModel 类。

**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**

**1.角球。最近的球员开球，次近的球员跑到接应点。**

角球是要求向场内踢。编写pointKickTo()返回在不同角时应该踢向的位置。

当球在自己半场时，将位置的X坐标调大，尽量向对方半场踢；而在对方半场时，X设置得比较小，为夺球攻门创造机会。

位置：basicplayer

VecPosition BasicPlayer::pointKickTo()//计算角球应该踢向的点

{

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;

}

位置：demeer5

else if( WM->isCornerKickUs())//我方角球

{

if(WM->isBallKickable() )

{

soc = kickTo(pointKickTo(), SS->getBallSize());

}

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(pointKickTo(), PS->getPlayerWhenToTurnAngle());

}

else

{

soc = moveToPos(WM->getStrategicPosition(), PS->getPlayerWhenToTurnAngle());

}

ACT->putCommandInQueue( soc ); // 放入命令队列

ACT->putCommandInQueue( turnNeckToObject( OBJECT\_BALL, soc ) );

}

**2.界外球。实验基本思路和内容和角球相同。编写程序，类似角球实现接应。**

界外球和角球类似，区别在于比赛中很少有机会在对方底线发界外球。所以大部分情况都将接应点设置得更加靠近敌方半场，增加进攻机会。

位置：demeer5

else if( WM->isKickInUs() )//我方界外球

{

if( WM->isBallKickable() )

{

soc = kickTo(pointKickToK(), SS->getBallSize());

}

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 ) );

}

位置：basicplayer

/\* 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

\*

\*/

VecPosition BasicPlayer::pointKickToK()//计算界外球应该踢向的点

{

VecPosition posBall, posKickTo;

double x, y;

posBall = WM->getBallPos();

x = posBall.getX();

y = posBall.getY();

posKickTo.setX( x + 12 );

posKickTo.setY(- y / fabs(y) \* 12 + y );

return posKickTo;

}

**3.定位球/任意球，设计思路和实验内容也类似角球。区别在于情况合适时可以选择直接射门。**

基本思路同界外球，都是尽可能向对方半场踢。但当条件合适（比如X比较大或者在禁区内），我们选择向较大的夹角直接射门。

else if( WM->isFreeKickUs() )//我方任意球

{

if( WM->isBallKickable() )

{

if(WM->getBallPos().getX() > 35 || WM->isInTheirPenaltyArea())//x大于35直接射门

{

soc = maxAngShoot( posAgent );

}

else

{

soc = kickTo(pointKickToK(), SS->getBallSize());

}

}

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 ) );

}

**4.让守门员沿最近敌方球员、自己、次近敌方球员三人构成夹角的角平分线开球。**

先求出最近敌方球员、次近敌方球员的位置，然后结合自己位置分别和他们连线，调用平分线函数求出平分点，往这个点开球。

SoccerCommand BasicPlayer::goalKickIntesec( VecPosition posAgent )

{

ObjectT o1, o2;

VecPosition pos1, pos2;

SoccerCommand soc;

o1 = WM->getClosestInSetTo( OBJECT\_SET\_OPPONENTS, posAgent );

o2 = WM->getSecondClosestInSetTo( OBJECT\_SET\_OPPONENTS, OBJECT\_TEAMMATE\_1 );

pos1 = WM->getGlobalPosition( o1 );

pos2 = WM->getGlobalPosition( o2 );

Line L1 = Line::makeLineFromTwoPoints( posAgent, pos1 );

Line L2 = Line::makeLineFromTwoPoints( posAgent, pos2 );

VecPosition posShoot = L1.getIntersection( L2 );

soc = kickTo( posShoot, SS->getBallSpeedMax() );

return soc;

}