程序说明

参赛者主要修改的函数为 imageProcess 函数和 run 函数
其中 imageProcess 函数为图像处理，run 为机器人的控制代码。

1、imageProcess

```
37 void UniRobot::imageProcess()
38 {
39   unsigned char *rgb = getRGBImage();  //get raw data, format: RGB
40   Mat rgbMat = getRGBMat(); //get rgb data to cv::Mat
41
42   showImage(rgb);
43   if(mode == MODE_BALL)
44   {
45     //TODO Write down your code
46     //update the resInfo
47     resInfo.ball_found = false;
48     resInfo.ball_x = 0.0;
49     resInfo.ball_y = 0.0;
50   }
51   else if(mode == MODE_LINE)
52   {
53     //TODO Write down your code
54     //update the resInfo
55   }
56   rgbMat.release();
57   delete []rgb;
58 }
```

给的代码的默认如上图所示，其中，函数 getRGBImage 可以获取机器人摄像头当前的数据，返回的结果为 RGB 图像，上面代码的指针 rgb 中就存了每一点的 RGB 值，其中图像的宽带为 image_w，高度为 image_h，可以在代码中直接使用，无需另外获取。

程序还提供了基础的 opencv 支持，参赛者可以使用 getRGBMat 函数来获取 cv::Mat 类型的数据，方便 opencv 的使用。

代码有两个模式: MODE_BALL, MODE_LINE
MODE_BALL 为踢球模式， MODE_LINE 为循迹模式。
关于踢球的图像处理代码，请写在 mode == MODE_BALL 中，关于循迹的图像处理代码，请写在 mode == MODE_LINE 中。图像处理获取的结果可以放入 resInfo 这个结构体中，结构体定义在 UniRobot.hpp 头文件中，如下所示：

```
struct ResultsInfo
{
  bool ball_found;
  double ball_x, ball_y;
};
ResultsInfo resInfo;
```

参赛者可以根据需要自行定义结构体中的变量。

2、run

```
else //*************************************************************//
{
  imageProcess();
  //TODO control the robot according to the resInfo you updated in imageProce
  //demo
  //kick ball
  if(mode == MODE_BALL) // mode ball
  {
    if(resInfo.ball_found)
    {
      if (resInfo.ball_y > 0.35)
      {
        mGaitManager->stop();
        wait(500);
        if (resInfo.ball_x<0.0)
          mMotionManager->playPage(13); // left kick
        else
          mMotionManager->playPage(12); // right kick
        mMotionManager->playPage(9); // walkready position
        mGaitManager->start();
      }
    }
    //walk control
    mGaitManager->setXAmplitude(0.0); //x -1.0 ~ 1.0
    mGaitManager->setYAmplitude(0.0); //y -1.0 ~ 1.0
    mGaitManager->setAAmplitude(0.0); //dir -1.0 ~ 1.0
    mGaitManager->step(mTimeStep);
    //head control
    neckPosition = clamp(0.0, minMotorPositions[18], maxMotorPositions[18]);
    headPosition = clamp(0.5, minMotorPositions[19], maxMotorPositions[19]);
    mMotors[18]->setPosition(neckPosition);
    mMotors[19]->setPosition(headPosition);
  }
  else if(mode == MODE_LINE) //mode line
  {
    //walk control
    mGaitManager->setXAmplitude(0.0); //x -1.0 ~ 1.0
    mGaitManager->setYAmplitude(0.0); //y -1.0 ~ 1.0
    mGaitManager->setAAmplitude(0.0); //dir -1.0 ~ 1.0
    mGaitManager->step(mTimeStep);
    //head control
    neckPosition = clamp(0.0, minMotorPositions[18], maxMotorPositions[18]);
    headPosition = clamp(0.0, minMotorPositions[19], maxMotorPositions[19]);
    mMotors[18]->setPosition(neckPosition);
    mMotors[19]->setPosition(headPosition);
  }
}
```

参赛者只能修改 run 函数中如上图所示的部分。
控制代码同样分为 MODE_BALL 和 MODE_LINE 两个部分，请将控制代码写在对应的地方。
这里可以利用图像处理获得的 resInfo，对机器人进行控制。
上面的代码给出了行走，踢球，转动脑袋等基本操作，参赛者请根据需要进行调用。