浙江大学

**科研和工程中的C++编程开发报告**

**毛晨炀 分报告**

目录

[解决方案 1](#__RefHeading__3019_173396680)

[设计思路 2](#__RefHeading__3041_173396680)

[运行效果 2](#__RefHeading__3041_173396680)

[所遇困难 2](#__RefHeading__3023_173396680)

[开发总结 2](#__RefHeading__3025_173396680)

[心得体会 2](#__RefHeading__3027_173396680)

[课程建议 3](#__RefHeading__3029_173396680)

# 解决方案

在本次开发中，我承担的任务是对图片的交通标志的标识和截图。

## 设计思路

要求利用视频单次传递的图片，标识出图片中交通标志，并将交通标志的图片截图，保存到文件夹中。

Rect模板

|  |
| --- |
| class Rect{public: …………  private:  //矩形左上角坐标和矩形的宽和高来确定一个矩形区域 T m\_xMin;//左上方点的X坐标 T m\_yMin;//左上方点的Y坐标 T m\_width;//矩形宽度 T m\_height;//矩形高度};typedef Rect<int> IntRect;  typedef Rect<float> FloatRect; |

Tracker类

|  |
| --- |
| class Tracker{public: Tracker(const Config& conf);//配置文件导入 ~Tracker(); …… inline const FloatRect& GetBB() const { return m\_bb; }//得到目标图片中的交通标识区域 private: const Config& m\_config; bool m\_initialised; std::vector<Features\*> m\_features; std::vector<Kernel\*> m\_kernels; LaRank\* m\_pLearner; FloatRect m\_bb; cv::Mat m\_debugImage; bool m\_needsIntegralImage; bool m\_needsIntegralHist; void UpdateLearner(const ImageRep& image); void UpdateDebugImage(const std::vector<FloatRect>& samples, const FloatRect& centre, const std::vector<double>& scores);  }; |

rectangle函数

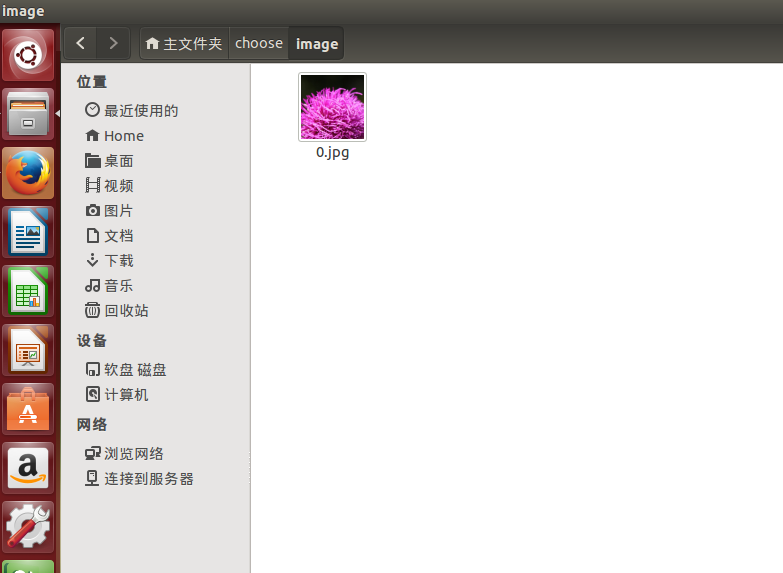
|  |
| --- |
| 函数功能： 通过对角线上的两个顶点绘制简单、指定粗细或者带填充的矩形  函数原型：void Cvrectangle( CvArr\* img, CvPoint pt1, CvPoint pt2, CvScalar color,int thickness=1, int line\_type=8, int shift=0 );  参数介绍：  img -- 图像.  pt1 -- 矩形的一个顶点。  pt2 -- 矩形对角线上的另一个顶点  color -- 线条颜色 (RGB) 或亮度（灰度图像 ）(grayscale image）。  thickness -- 组成矩形的线条的粗细程度。取负值时（如 CV\_FILLED）函数绘制填充了色彩的矩形。  line\_type -- 线条的类型。见cvLine的描述  shift -- 坐标点的小数点位数。 |

TrackSign函数

|  |
| --- |
| if (true) //(outFile)  {  const IntRect& bb = tracker.GetBB();  Mat image(result, cv::Rect(bb.XMin(), bb.YMin(), bb.Width(), bb.Height()));  char outImage[128];  sprintf(outImage, "image/%d.jpg", frameInd/SPEED);  //cout << "Write to: " << outImage << endl;  imwrite(outImage, image);  SetMark(image);  }  rectangle(result, tracker.GetBB(), CV\_RGB(0, 255, 0)); |

## 运行效果





## 所遇困难

刚开始装Ubuntu的操作系统和Boost,openCV的库装了好几遍，极大的浪费了开发时间。

在探讨整体架构的时候，一开始没有理解MVVM的模式。

开始的开发思路也有问题，部分自己原来可以实现的东西，整合到全组中出现了错误，最后推倒，重新写了部分代码。深刻感到多次迭代的优点。

# 开发总结

## 心得体会

这次短学期中学到了许多在科研和工程中C++的写法，感觉在程序开始写之前的架构设计是整个工程的重点，迭代设计的安排将整个设计耦合度降低，保证了每次得到的迭代成果都是可以运行的，大大保证了整个工程的编写效率，这种设计方法很让人受益。组长在这次短学期的工作中起到了关键性的作用，带领团队将大程完成，我从团队成员中学到很多，感谢每一个成员的付出。

课程建议

感觉资料给的有点少，许多参考资料找了半天，特别是理解MVVM模式的时候，缺少标准的源代码，理解起来有点困难。

2014年7月23日