QT 大作业报告

张章 2200012902 丁宇翔 2200012904 吴青阳 2200012905

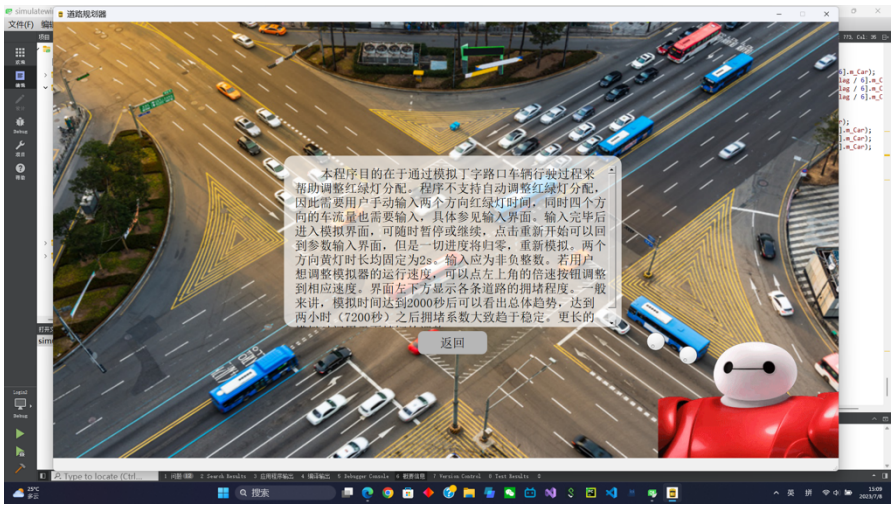
一． 程序功能介绍

本项目主题为道路规划器，功能为在用户输入一个丁字路口各个车道的红绿灯时长和车流量时，模拟出对应的交通情况，并对拥堵情况做出相应的评估。

二． 项目模块

用户运行程序时，首先可以看到三个按钮，分别是“帮助”输入“和”退出“，每个界面左上角有显示图标是一辆车，而界面名称就叫”道路规划器“第一个界面背景是一辆车。点击”退出“则显然程序就退出运行了，而点击”帮助“则可以看到道路规划器的简介说明，背景是一个十字路口，而右下角有一个可爱的大白动图一直在眨眼。此时我们可以点击”返回“按钮回到第一个界面，并开始输入参数。

初始界面

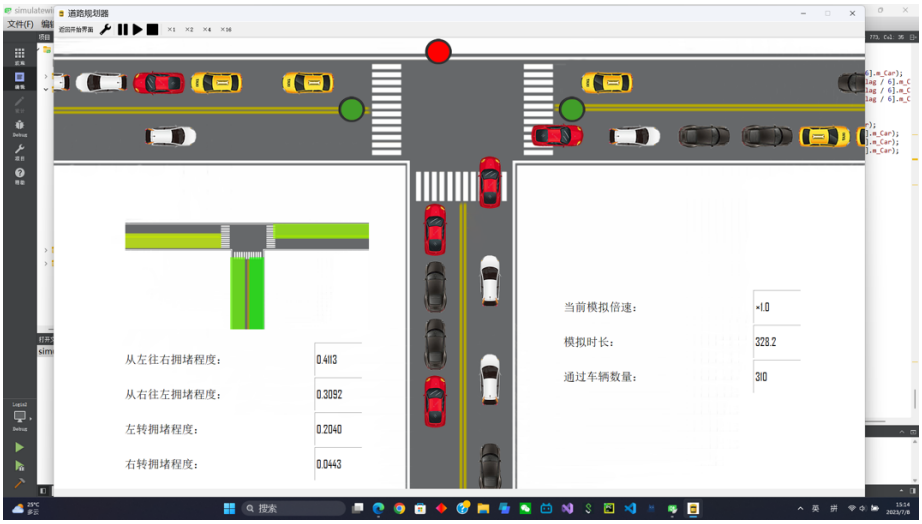


帮助界面

输入参数的界面的背景是另一辆车，并且有六个参数可输入，分别是直行绿灯时间，左转绿灯时间，向左直行车辆，向右直行车辆，左转车辆，右转车辆。当然用户要注意相对应的单位，以免参数不符合预期。两侧绿灯结束后会有固定两秒黄灯时间，此时另一个灯仍为红灯。如果没有输入所有数字，那么将弹出提示，直到输入所有数字为止。这样用户就可以进入模拟界面了。

输入界面

模拟界面会按照用户输入的数据来运行。每条道路有四种颜色的车，每次生成一辆的时候均为随机，左上角可以选择调运行的速度，最慢为*0.5，最快为*10，当然红绿灯时长也会相应变化。点击扳手图案可以回到输入参数界修改参数，点击左上角可以回到主页重新开始。而每条道路有一个实时拥堵系数，计算方式为该道路上拥堵时长与总时长之比，精度为 0.0001，显然拥堵系数越高越拥堵。左下角有一个每条道路的实时缩略图，绿色代表畅通，黄色代表较拥堵，红色代表拥堵。



s

三． 类设计细节

1. 界面组织（丁宇翔、吴青阳）

（1） 界面之间的结构

主场景界面、帮助界面和设置参数界面分别对应一个类，再设置一个主界面类以容纳这些界面，通过指针将它们联系起来。需要展示某一界面之时，主界面通过 takecentralwidget（）与 setcentralwidget（）两个函数来实现中心部件的切换。

（2） 界面之间的切换

得益于 Qt 信号与槽的机制和成熟的 QPushbutton 类，我们只需构建按钮，再将点击按钮这一信号与实现界面切换的代码进行绑定即可。

在 mainwindow.cpp 中 setwindowtitle 来存放各个界面的图标和名称，名称存放在 config.h。

ui 设计：用 frame 创建文字框，在里面填充帮助中的介绍，用 photoshop 将网上一个 gif 动图的背景扣掉，放在帮助界面上，背景图片可以作为 border-image，可以自适应尺寸，且每个界面有 pushbutton 用来界面切换。

在输入数据中会进行识别，没全输入会弹出相对应的 warning，六个全输入才会 accept()进入下一个界面。

2. 主场景部分（张章）

主场景有一个类 ，类名为 SimulateWindow，其中包含主场景的全部内容。

以下是基本的介绍：

（1） 车辆构造：车辆单独成一类为 Car，包含在 car 文件中，存放车辆编号、车辆位置（x、y 坐标）、车辆速度、车辆种类以及判断其上一步是否在行驶的变量。车辆按用户输出的车流量以相应概率构造，构造位置为屏幕外不可见的位置。如果相应位置存在车辆，则构造的时候只是将程序中的相应变量++（是一个 int 类型变量，分别是 left_to_right_num 等等），之后有位置构造的时候再进行构造，再将这个变量--，从而实现了存放待构造车辆的功能。待构造车辆也算在后续计算车辆数量中。

（2） 场景的加速：画面运行方式是利用更新频率不断更新实现的，所以画面加速只需要加快更新频率即可。

（3） 记时的实现：记时利用的是从程序运行开始到现在的更新次数除以相应频率得到的，这样可以让计时器和程序的加速同步。

（4） 红绿灯的实现：红绿灯单独成一类，包含在 light 文件中。红绿灯利用计时器和用户输入的红绿灯时长计算现在的状态，其中两侧的黄灯均安排为两秒，目的是留出充足时间给车辆完成转弯。

（5） 车辆类以及转弯方法：本程序中有四种车辆但是最后设计了六个类，这里也是项目的一个缺陷。多出来的两个类分别是右转完毕在直行车道上的车以及左转完毕在直行车道上的车。之所以如此设计是因为转弯车辆采用的是用 9 张图片离散地显示（这是因为当时设计的时候没有搜索到一个比较方便的将图片旋转的方式，但是后来觉得或许不断更新车辆对象中存放的图片来实现似乎是一个不错的选择，但是也需要不段更新车辆的 x 方向与 y 方向速度，感觉对于这样一个小工程的项目而言不是十分划算）。每次一旦开启转弯便删除转弯前的车辆对象，显示和这个车辆颜色相同的转弯图像，转弯完成之后马上构造一个颜色相同的在直行车道上的车。这里其实应该把这辆车构造在本来直行车辆的 vector 里的，这样计算车辆距离会更加方便，对于更大的项目似乎更优，所以这里也是这个项目的一个小缺陷。四种车辆颜色的实现就是单纯的复制粘贴代码。四种颜色随机生成。

（6） 车辆组织：首先是如何让同一车道的车不会相撞，这里采用的方式是计算同一车道上前面的车和这辆车距离是否近，如果近那么这辆车下一步不移动，否则按自身速度进行移动。其次是如何保证不同车道的车不会相撞。左转和直行车辆不相撞利用红绿灯来保障，唯一需要处理的是向右直行的车如何与右转的车不相撞。这里采用的是定义一个新的变量记录现在是否有车在右转，以及是否有车在直行。如果有车辆在右转则直行车辆停止在斑马线之前；如果有车辆在直行，则右转车辆也停在相应位置。这样也能保证左转车

辆和向右直行车辆不会相撞（如果向右直行车辆停止在马路中央，那么将会和左转车辆相撞）。

（7） 拥堵系数以及小地图：拥堵系数的计算方式就是当一条道路上的车辆数量超过一定数目以后就记录一次，用这个数值除以总的更新次数就得到了一个拥堵系数。可以看出这个数一定是 0 到 1 中的一个实数，代表了拥堵时间占总时间的比例。小地图中的四条车道的颜色显示了四条车道的拥堵系数，其中 0 是绿色，1 是红色，中间是相应的过渡。

（8） 评估系统的设计：评估系统分为五个等级，分为 S、A、B、C、D，评估方法是计算四个拥堵系数的方差，方差越小评级越高。我们认为，当车流量固定之后，红绿灯好坏不应该看绝对拥堵程度，更重要的是相对拥堵程度，当四条道路拥堵程度比较平均的时候是一个比较好的红绿灯设计。同时如果四条道路拥堵系数扩了 t 倍，最终方差扩 t 平方倍，所以它也是考虑绝对拥堵系数在内的。

以下是 simulatewindow.cpp 中各函数的功能：

（1） SimulateWindow：构造函数了，初始化所有信息

（2） ~SimulateWindow：析构函数，删除对象及 ui

（3） initScene：初始化场景，用来初始化定时器以及窗口大小、窗口小标题、图案等内容。

（4） destruct_turn_right_cars：删除等待右转的车（右转开始的时候调用）。

（5） destruct_turn_left_cars：删除等待左转的车（左转开始的时候调用）。

（6） destruct_right_to_left_cars :删除向左直行的车（向左直行的车走到头的时候调用）。

（7） destruct_left_to_right_cars :删除向右直行的车（向右直行的车走到头的时候调用）。

（8） destruct_turned_left_cars :删除左转完毕的车（左转车走到头的时候调用）。

（9） destruct_turned_right_cars :删除右转完毕的车（右转车走到头的时候调用）。

（10） playGame：完成定时器设置、监听定时器、计算红绿灯、构造车辆、计算拥堵系数、更新坐标、绘图以及连接其它场景的作用。

（11） build_turn_right_Cars :构造等待右转的车辆，需要判断构造位置是否可以构造车（如果有太近的车辆那么就不可以构造车），如果不可以构造，需要存储“欠构造”的车辆，之后构造。“欠”的车辆在后续计算拥堵系数的时候也要算在内。

（12） buid_turn_left_Cars：同上

（13） build_left_to_right_Cars：同上

（14） build_right_to_left_Cars：同上

（15） updatePosition：分别更新六个 vector 中的所有车辆（等待右转的、等待左转的、右转完毕直行的、左转完毕直行的、以及两列直行的）。更新方法大同小异，检查每辆车移动到下一个位置后是否会发生车辆冲突（撞车），如果不会撞车那么按照车辆速度来更新车辆位置。值得注意的是右转弯车辆与向右直行车辆互相影响，因此设计了相应部分防止二者撞车（设计了 heading 变量）

（16） paintEvent：绘制状态，同时在这里构造右转完毕以及左转完毕的直行车，更新当前转弯状态（转弯是由 9 张图片构成的，在这里要更新目前应该显示的图片）

四． 项目总结与反思

本项目合作分工合理，页面精美，类和对象设计的过程也很巧妙，成功设计出了一个较为简约的道路交通系统。可改善之处有二：1.代码量将近 2000 行，可以考虑优化代码写得更漂亮；2.只有一个丁字路口的车，现实生活中十字路口更多而且会有更多的路口需要模拟。

五． 小组分工：

张章：主界面主场景设计（车辆显示、车辆安排构造等）

丁宇翔：主界面小地图设计、主界面工具栏设计、退出界面设计、界面切换设计

吴青阳：输入界面设计、帮助界面设计、初始界面设计、帮助界面设计