移位左转配时策略研究



重庆大学SRTP项目开发报告

移位左转配时策略研究项目组

学生姓名：张鑫燃 龚宇晨 杨天昊

指导教师：郭 平 教 授

### 目 录

[第一章 问题背景 3](#_Toc7810679)

[第二章 算法依据 3](#_Toc7810680)

[第三章 技术路线 9](#_Toc7810681)

[第四章 可行性分析 10](#_Toc7810682)

[第五章 系统详细设计 14](#_Toc7810683)

[第六章 系统交互设计 21](#_Toc7810684)

[第七章 系统功能演示 22](#_Toc7810685)

[第八章 测试报告 26](#_Toc7810686)

[第九章 后续工作 29](#_Toc7810687)

[第十章 项目总结 29](#_Toc7810688)

# 第一章 问题背景

移位左转又称为连续流交叉口（CFI，Continuous Flow Intersection）是世界上最前沿的交通组织手段之一。移位左转通过将左转车道渠化转移设置，重组道路断面，来实现相对方向直行和左转同时放行，从而减少信号相位，提升整个路口通行效率的交通组织形式。2017 年 10 月 7 日，深圳交警在彩田-福华路口启动全国首个“移位左转”（CFI）交通组织路口。由此，国内逐步开展了移位左转交通组织方法的研究。

通过信号灯控制交叉路口的车辆通行是目前交叉路口最常见的交通控制方式。配时策略则是实施交叉路口信号灯控制的基础。配时策略由各车道绿灯时长（绿灯放行时间）和绿灯开始时间两部分组成。目前，配时策略主要有固定配时策略和自适应配时策略两种。固定配时策略依据不同时间段设置绿灯时长或者全天采用统一的绿灯时长，这种方法简单方便，但不能适应灵活多变的交通流。自适应配时策略依据监测的实时交通流来动态调整绿灯时长，这种方法实现较复杂，但能够适应交通流的变化，提升交叉路口的通行效率。

本项目提出一种城市交叉路口道路设计与红绿灯时长结合，根据实时车流量分配通行时长的智能算法，并开发出相应的仿真系统。自适应移位左转路口可以预分离左转的车辆进入左转车道，在路口正式通行时减少车辆拥堵，增加了通行效率，同时可以根据车辆通行状况预测未来的道路状况，从而起到调节车流的功能。

# 第二章 算法依据

## 2.1信号相位

由于不同路口的交通状况不同，因而在组织路口的运转时引入了相位的概念。将几个车道划归为一个相位，每个相位时间内保证每个车道都有通行的机会，同时在一个时间周期内执行一遍全部的相位。

如图为两相位的路口设置，第一相位时东西通行南北禁行，第二相位则相反。

第1相位

第2相位

东西方向

南北方向

南北方向

东西方向

图2.1.1 两相位配时方案

同时根据路口车流的状况不同，可以单独为左转车道分离出一个相位执行。下图就是分离出左转相位的三相位组合。

第1相位

第2相位

南北方向

南北方向

第3相位

南北方向

图2.1.2 含有专用左转相位的三相位配时

## 2.2相容车道

定义任意一个车道和车道通行不发生冲突，就称这两个车道互为为相容车道，一个车道的相容车道可能不止一个，比如L6的相容车道有：L2，L3，L5。 如图2.1所所示：

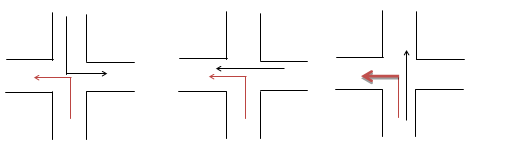


图2.2.1 相容车道

## 2.3信号周期算法

信号周期不宜过长过短，以保证车辆的顺利通行，设绿灯时长下限为Tg，信号周期的上限为Tmax。由此，信号灯控制策略为：

1. 每个信号周期的每个车道有至少一次通行权。
2. 每个车道的最短绿灯时长不短于
3. 每个信号周期的长度变化且不长于
4. 任意时刻仅允许两两相容车道组合的车辆通行

以上约束条件控制信号灯的时长配置，我们通过第i轮的数据来在i+1轮周期中计算并得到新的配时方案，并且在第i+2个周期中投入使用，使得i+2的通行效率达到改善。

## 2.4普通路口车道组合计算

设置一种车道组合方式，由两个车道序列构成如

（式2.4.1）

在Π中，与互为相容车道，分别属于车道序列一和车道序列二，且它们同时通行。

当所有车道都放行一遍之后会结束，算法会在没有执行过的车道中选择一个添加进队列作为主车道，并在主车道的相容车道中选择一个以组成车道组合片段，如果两个车道剩余的绿灯时长相同则共同执行，否则将后执行完的车道作为主车道，继续添加其另一条相容车道，直到所有车道都添加进队列。

这样选出来的相容车道有多种组合方式，系统会选择周期时间最短的车道组合，并和设置的最长周期时间作比较，若时长大于最大值，则选择一条合适的车道将其时间进行调整，之后再重新获得最优队列。

## 2.5 移位左转相位组合

移位左转路面组织方式不同于普通路口，将左转车辆和迎面而来的直行车辆的冲突提前调整至路中，左转车辆提前变道，在主交叉路口与对向直行车辆同时通行，减少了组成周期的信号相位，缩短等待时间，可以实现2个、3个、4个路口的同时放行

图2.5.1 移位左转车道编号

移位左转和普通路口有较大差别，不能采用相同的方式进行相位设计。我们将所有可以同时通行的车道组合成一个相位，共有如图2.5.2所示的7种相位情况。

（1）车道、、、同时通行

（2）车道、

同时通行

（3）车道、、、同时通行

（4）车道、、、同时通行

（5）车道、、、同时通行

（6）车道、、、同时通行

（7）车道、、、同时通行

图2.5.2 七种相位情况

对这七种相位进行组合，并满足每个周期每条车道至少拥有一次通行机会的原则，我们得到了三种周期类型(相位组合方式)用于不同的交通流情况。

图2.5.3 周期类型1

相位1，

主交叉路口：、

相位2，

主交叉路口：、

相位3，

主交叉路口：、、、

相位1，

主交叉路口：、、

相位2，

主交叉路口：、、

相位3，

主交叉路口：、、、

图2.5.4 周期类型2

图2.5.5 周期类型3

相位2，

主交叉路口：、、

相位3，

主交叉路口：、、

相位1，

主交叉路口：、

## 2.6移位左转自适应周期计算

在一轮放行结束之后，每个车道会得到对应的通行时间和通行车辆数，从而计算出通行效率。首先对各个车道的绿灯时长进行优化，然后按照不同的周期和相位组合对其下一轮通行车辆数进行预测。

相关参数：

相位时间：得到新的各个车道的预测时间后，我们可以算出某一个周期中的某个相位的主车道时间加和平均以得到这个相位的时间。

相位效率：一个相位的效率就是该相位中的各个车道效率加和。

相位车辆通过数：通过相位时间和相位效率乘机，我们可以得到一个周期中的每个相位通过车辆数。

通过比较每个周期的相位车辆预期通过数总和就可以判断出应该选择哪个周期来执行，选出最优周期后，将周期时长与预定的最大周期时长相比较，如果超过了最大周期时长则按照与普通自适应超时处理方法类似的方式缩小时间：

将三个相位与既定的效率标准线对比，如果存在相位低于标准线，则在效率低于标准线的相位中选择效率最低且可以削减时长的相位（指削减时长后仍然高于相位最低时长）；如果所有相位都高于标准线，则选择时间最长的相位进行实践削减。

总体算法流程如下：

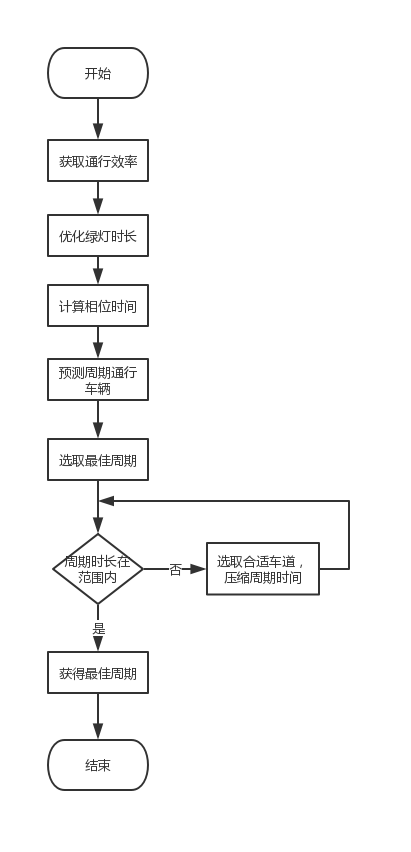


图2.6.1 移位左转自适应配时算法流程图

其中，绿灯时长优化算法对于普通路口和移位左转路口都是通用的：

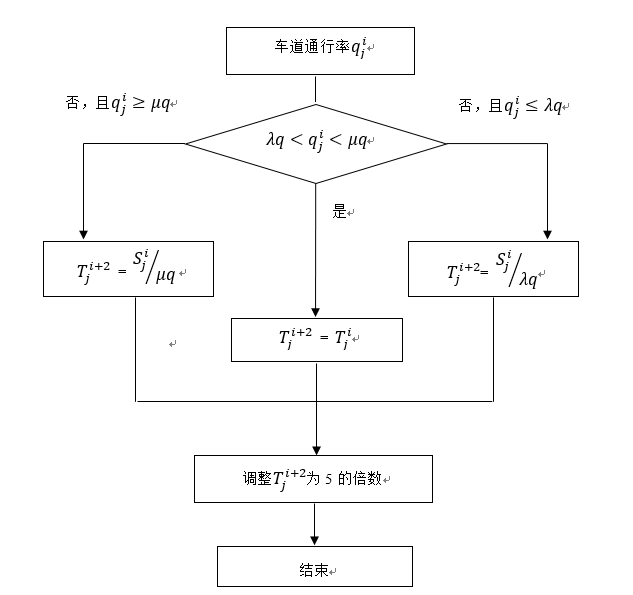


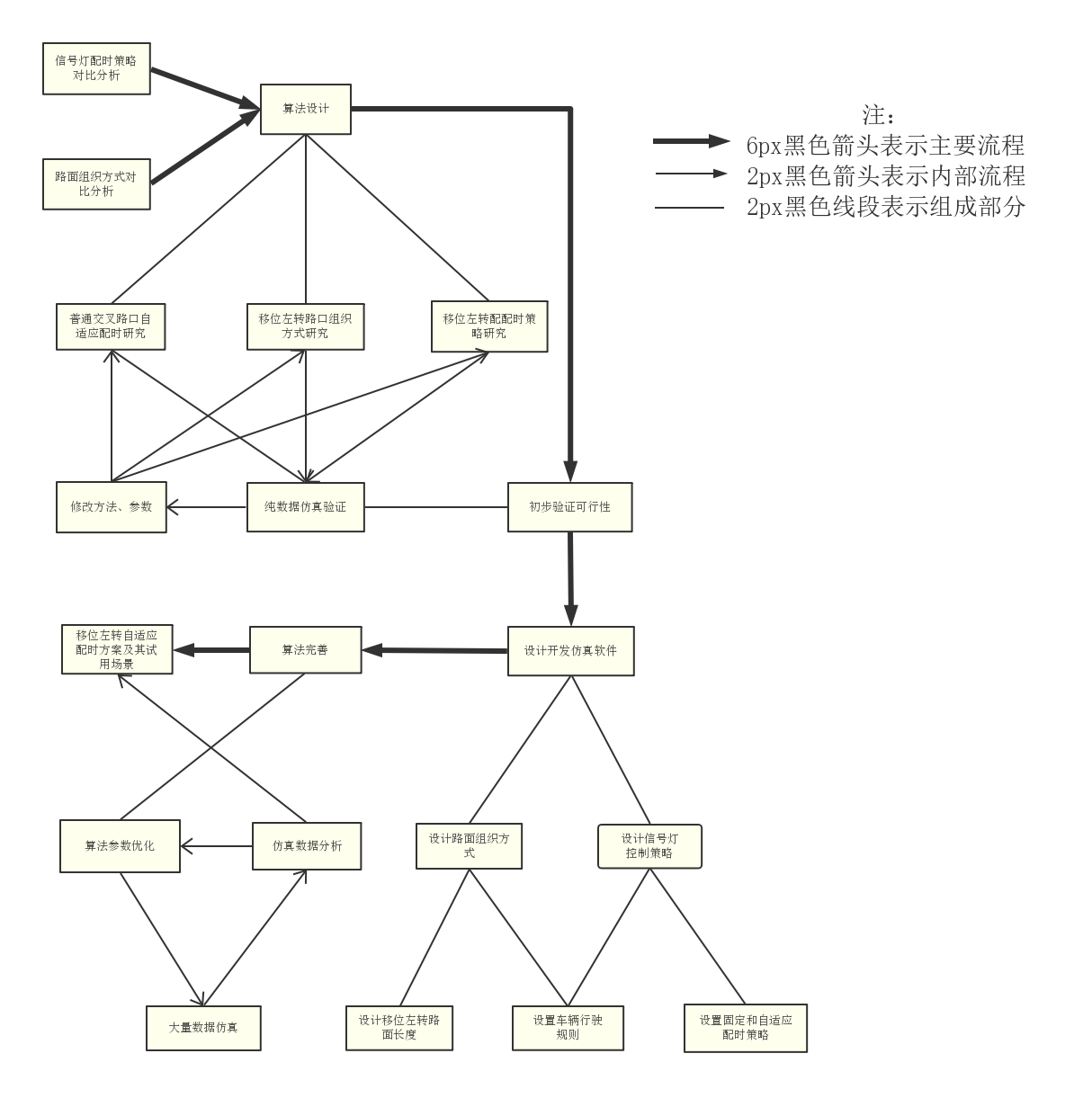
图2.6.2 绿灯时长优化算法流程图

# 技术路线

本项目的研究从交叉路口传统交通组织方式入手，通过“学习→研究→实践→提高”的循环完成本项目研究任务。具体的研究路线是：

1. 从交叉路口传统交通组织方式入手，研究配时策略相关算法，包括但不限于移位左转路口组织方式、自适应配时算法、车道相位组合设计
2. 研究移位左转交通组织方式中车道设计方法，确定交通控制相位组合相关参数和车道相关参数，研究移位左转配时策略，设计相位选择算法，车道绿灯时长优化算法，车道绿灯开始时间算法
3. 采用纯数据的仿真，利用随机产生的各车道车流量，测试传统交叉路口和移位左转路口配时策略的有效性，利用实测数据测试移位左转路口道路设计的合理性
4. 利用专业交通仿真软件（难以进行大量数据的仿真）以及自行开发的交通仿真软件进行大量数据的交通仿真，通过分析仿真结果对算法设计进行优化
5. 完成算法的设计，并分析得到算法的最优适用场景

技术路线图如下：



# 可行性分析

为了验证最终的方案设计是否具有可行性，我们在方案设计的基础上开发了一款应用于特定场景的可视化交通仿真软件，进行了大量的随机数据仿真。经过仿真结果验证，我们发现总体上，移位左转自适应算法相对于普通路口固定配时算法的通行效率有了巨大的提升。我们对其中29组数据进行具体的可行性分析。

具体的仿真输入和输出结果如下：





对数据进行处理分析：

普通自适应对比普通固定配时算法，通行效率有了显著地提升，最高提升了40%；移位左转自适应对比移位左转固定配时算法，通行效率也有明显提升，最高提升了52%，可以验证自适应算法对比固定配时算法有更高的通行效率，且提升较大。

通过对比可以发现，使用固定配时，在大部分情况下，移位左转路口的通行效率要比普通路口高得多，说明移位左转这种路面的组织方式确实要优于普通的路口，能够明显提升通行效率，具有很高的经济效益和研究价值，研究该路面组织方式是可行的。但是移位左转固定配时对于普通路口自适应配时的提升并不明显。经过分析发现，在设置移位左转车道的方向上左转车辆数较多、直行车辆数较少时移位左转路口的效率提升会受到影响，这也是我们对于该方案适用范围的分析成果。同时，只是设置移位左转或者只是采用自适应配时是具有一定的局限性的，因此我们将两者结合起来作为研究重点，能够克服单独采用某种方式的局限性。

通过将移位左转和自适应配时进行结合，对比普通固定配时有了显著地提升，最高提升76%，平均提升32%，可以验证我们的算法是可行的。

# 第五章 系统详细设计

## 5.1 移位左转路面设计

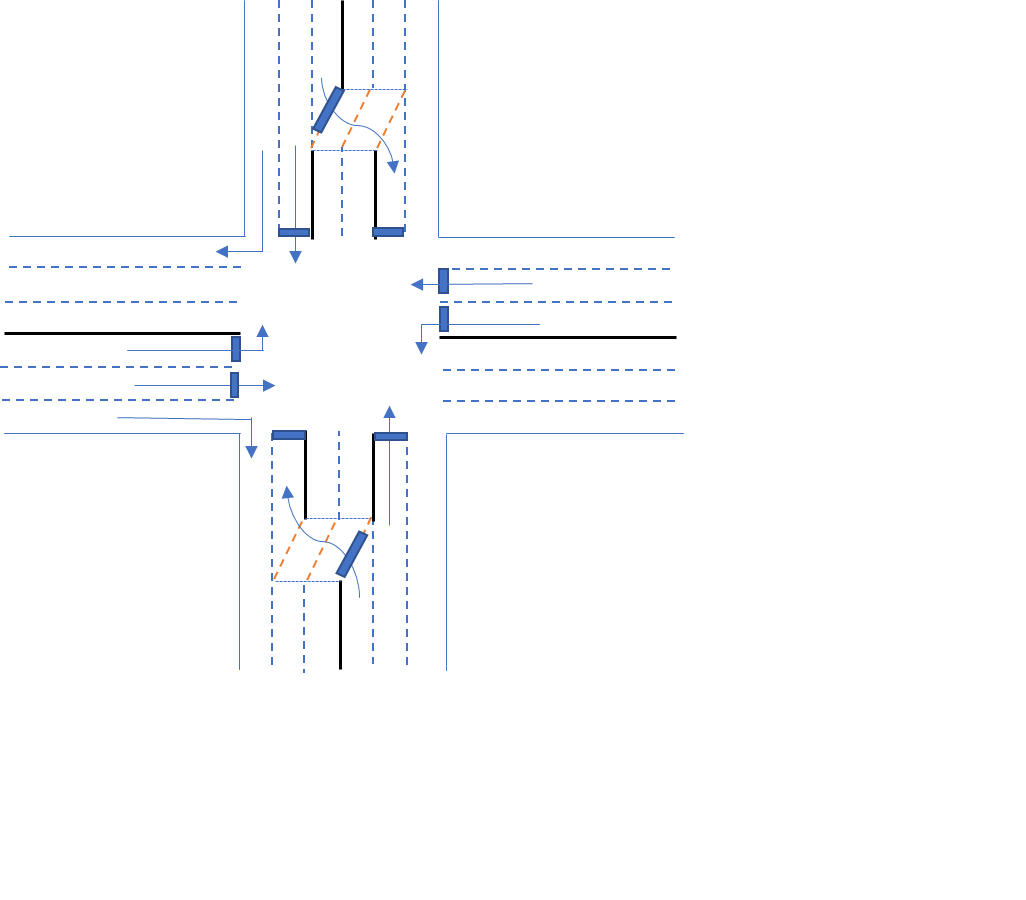
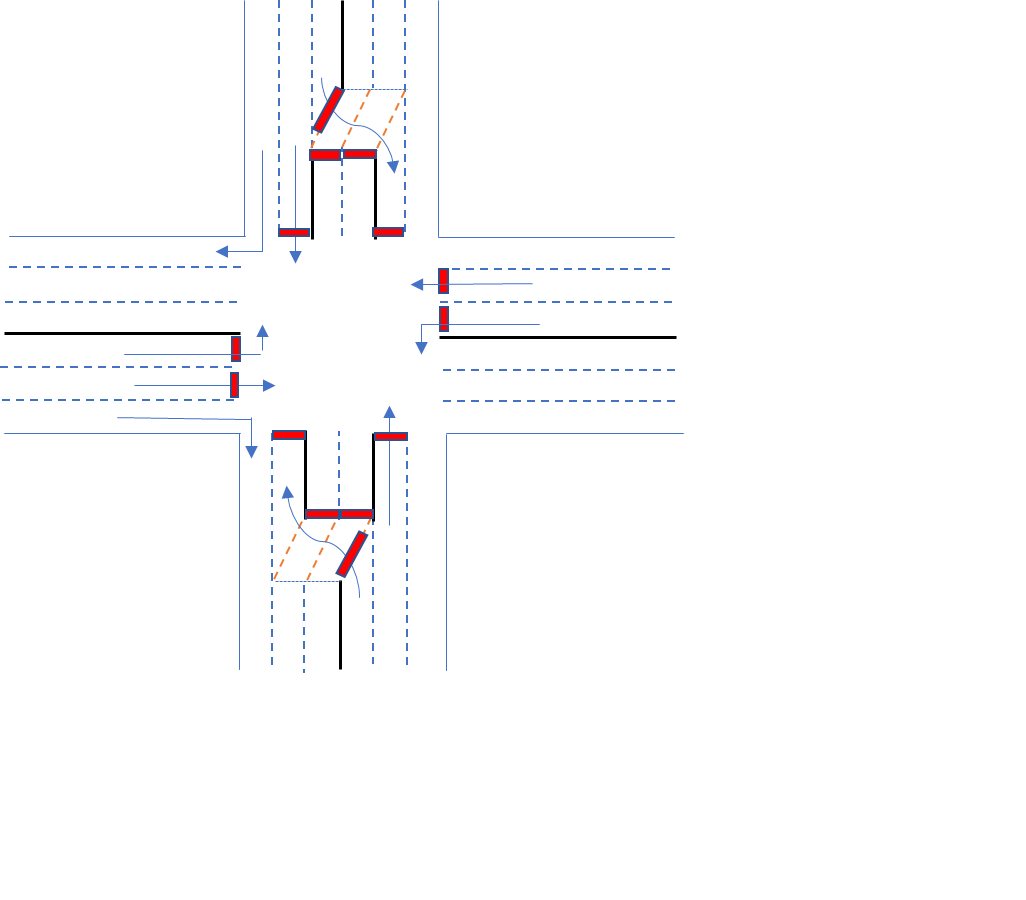


图5.1.1 车辆检测器部署图

对于车辆检测器的部署，我们设计其在各个车道通行处，通过车辆检测器检测到交通流数据，可以进行周期配时的优化，



60m

图5.1.2 交通信号灯部署图

如图为我们设计的红绿灯安置方案，可以通过各个信号灯之间的协调避免任何交通流冲突的发生，并在计算出信号周期的基础上对个信号灯进行亮灯状态控制，达到提升通行效率的目的。

经过了大量数据仿真，我们将图中移位左转车道长度设计为60米，这样能够在使用有限的路面资源时最大程度提高通行效率。

## 5.2 软件实现-主要模块简介

|  |  |  |
| --- | --- | --- |
| 模块 | 类 | 作用 |
| 界面 | TrafficSim | 主界面，包含菜单项、工具栏和主窗体 |
| Ordinary\_Vehicles\_Inputs | 普通路口车辆输入 |
| CFI\_Vehicles\_Inputs | 移位左转车辆输入 |
| Simulation\_Configure | 仿真参数设置 |
| SimResult | 仿真结果显示 |
| 仿真器 | simulation\_con | 仿真控制器，实现车辆、信号灯的生成、控制、回收；实现各种交通规则、控制车辆的行驶动作；实现车辆检测，与信号机交换数据 |
| vehicle | 存储车辆信息：外观、速度、位置、行驶路线等 |
| signal | 存储信号灯信息：外观、状态、位置等 |
| 信号机 | Ordinary | 实现了普通路口信号固定配时和自适应配时算法，对外提供接口，返回信号灯状态 |
| Left | 实现了移位左转信号固定配时和自适应配时算法，对外提供接口，返回信号灯状态 |
| 通用 | Com\_use | 定义了其他类会使用到的公共枚举变量、结构体等 |

## 5.3 软件实现-主要类说明

界面类只负责实现与用户的交互以及数据的输入输出等逻辑事务，不涉及到核心的算法，因此不在这里进行介绍

a. Com\_use

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| RoadType | enum | 定义了路口类型 |
| SignalControl | enum | 定义了配时方案 |
| SignalState | enum | 定义了信号灯状态 |
| ROAD\_conflict | Int \*\* | 定义了个车道之间的冲突关系，用于仿真时的冲突避免算法 |
| Ord\_Signals  CFI\_Signals | Int \*\* | 定义了信号灯放置的位置以及方向 |
| Road\_Line | class | 定义了不同场景下各车道的行驶路线 |

b. simulation\_con

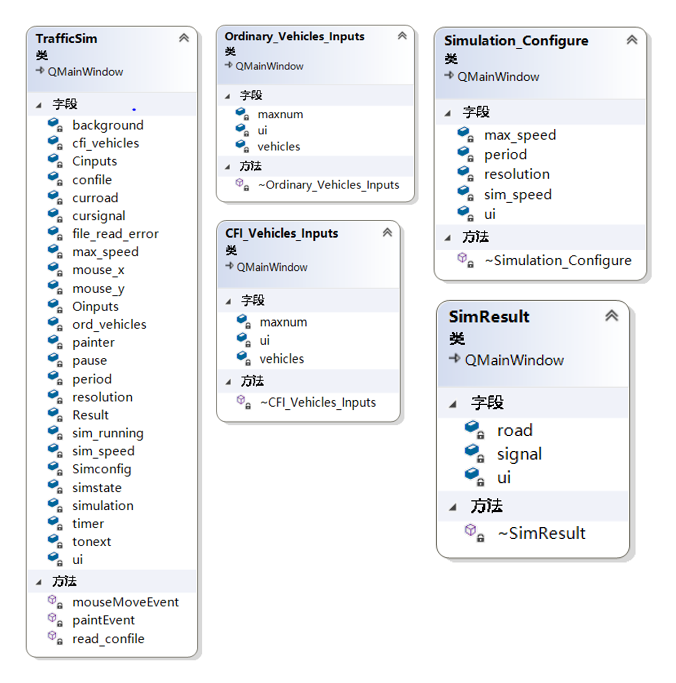
|  |  |  |
| --- | --- | --- |
| **主要字段** | **类型** | **说明** |
| sim\_running | bool | 表示仿真器是否正出于仿真状态 |
| probability | float\* | 表示按照泊松分布的车辆到达率，即仿真器生成车辆的概率 |
| vehicles\_inputs | QList<vehicle \*>\* | 存放仿真时生成的车辆 |
| ord\_signals  cfi\_signals | QVector<signal \*> | 存放仿真时放置的信号灯 |
| all\_time | int | 总仿真时长 |
| cur\_time | int | 已经经过的仿真时长 |
| **主要方法** | **类型** | **说明** |
| start\_sim() | void | 开始仿真是调用，进行数据的初始化 |
| single\_step\_sim() | void | 运行单步仿真，控制车辆的生成和移动、改变信号灯的状态、绘制车辆和信号灯 |
| stop\_sim() | void | 停止仿真，销毁生成的车辆和信号灯、回收分配的内存 |
| Ord\_check\_signal()  CFI\_check\_signal() | void | 控制车辆根据信号灯状态改变驾驶行为，实现信号灯控制交通流 |
| Ord\_check\_impact()  CFI\_check\_impact() | bool | 检测车辆是否可能发生碰撞，根据返回值来改变车辆的驾驶行为 |

c. Ordinary(Left类的字段和方法与Ordinary相似，只是方法的实现方式不同)

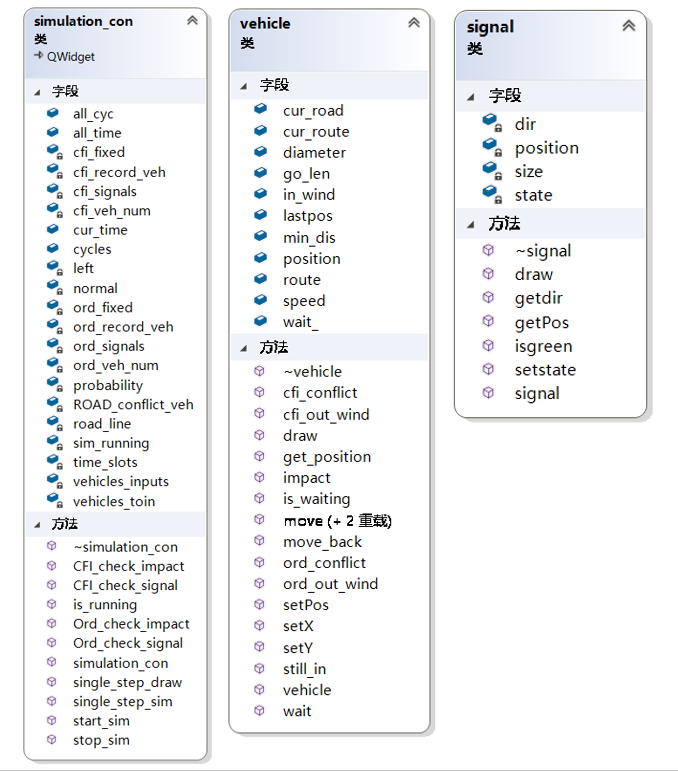
|  |  |  |
| --- | --- | --- |
| **主要字段** | **类型** | **说明** |
| Tmax  S\_Tmin | int | 算法参数，设置周期的时间和车道的最小通行时间 |
| combination | int \*\* | 算法参数，列出了各个车道的所有相容车道 |
| Q | float \* | 算法参数，设置了各个车道的最大通行效率 |
| road\_t | int \* | 分配给各个车道的通行时间 |
| start  end | int \* | 各车道放行的开始和结束时间点 |
| is\_green | bool \* | 信号灯状态，反应该车道是否放行 |
| **主要方法** | **类型** | **说明** |
| fixed\_next()  next() | void | 核心算法，固定配时、自适应配时根据车辆检测数据来获取下一个周期的配时方案 |
| judge() | bool \* | 提供的外部接口，返回信号灯状态 |

## 5.4 类视图-详细

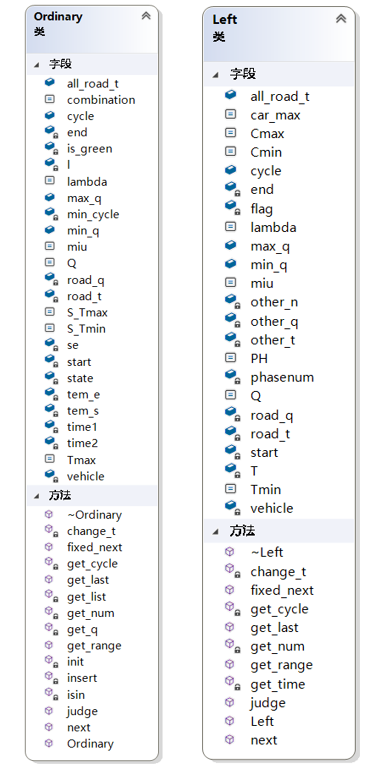
### 5.4.1 界面类



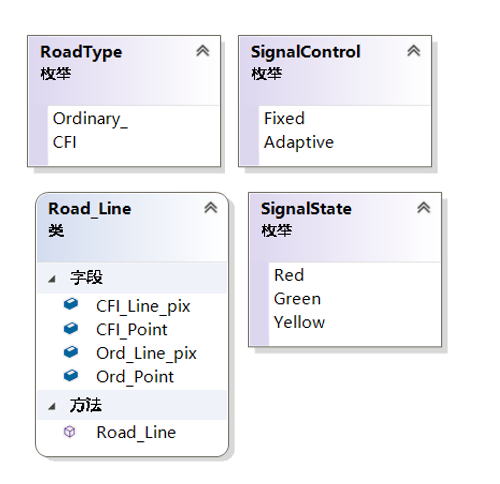
### 5.4.2 仿真器类



### 5.4.3 信号机



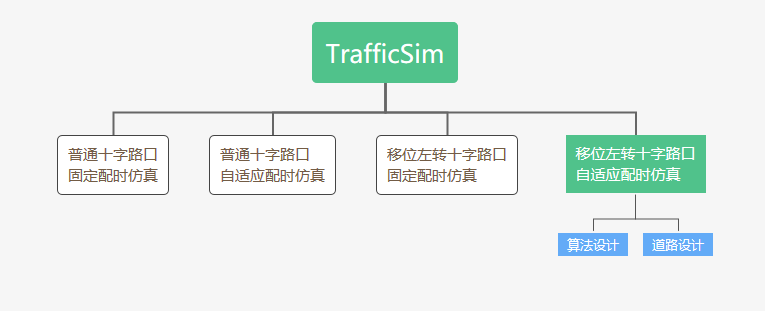
### 5.4.4 通用



# 第六章 系统交互设计

## 6.1 作品功能设计

为了验证算法的可行性，我们设计了专门的可视化交通仿真软件，实现了算法的可视化和软件与用户的交互，软件主要功能设计如下：



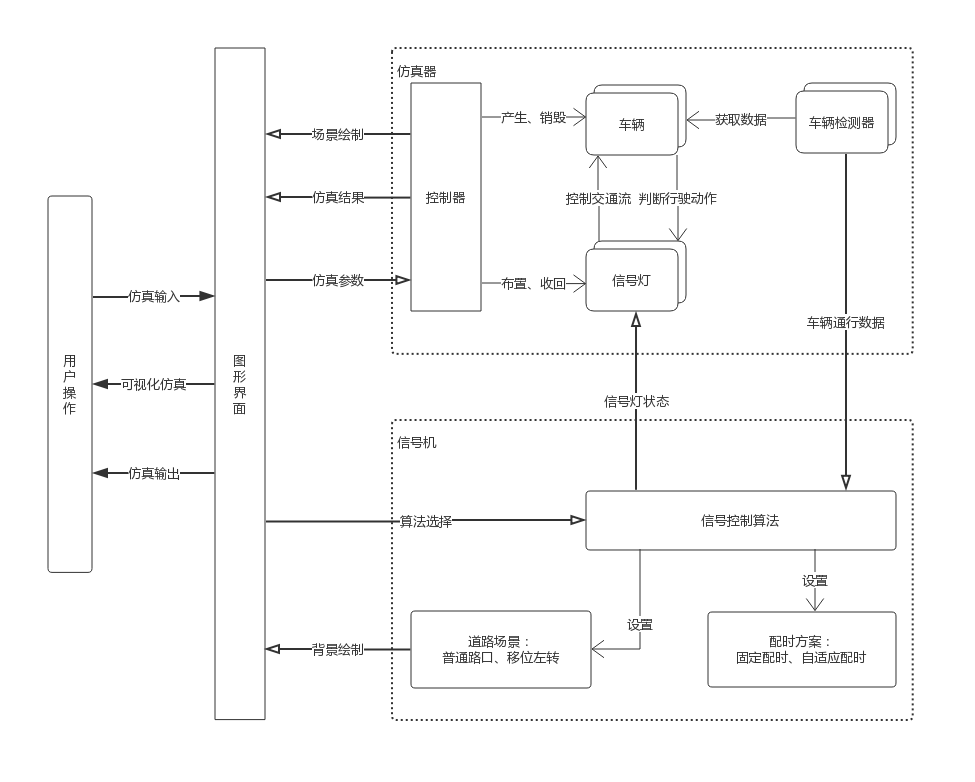
①普通十字路口固定配时仿真，主要模拟现阶段大多数交通路段的通行方式，目的是与移位左转自适应配时路口做出比较，对比车辆通行率；

②普通十字路口自适应配时仿真，将红绿灯的配时策略变为相对智能的自适应模式，目前已有少量城市路口使用此种策略；

③移位左转固定配时策略仿真，此种情况是移位左转路口结合自适应配时的基础，在固定配时策略下，设计移位左转路口，结合相对简单的固定配时，由成果可以看出此时相对于普通路口固定配时车辆通行效率有所改善；

④移位左转自适应配时策略下的交通仿真，此种算法结合了智能的自适应配时与相对较为先进新颖的移位左转路口，是作品的核心所在，通过设计相容车道、绿灯配时、车道选择方式等改善车辆通行率。

## 6.2 作品结构设计



软件整体结构分为三部分：图形界面、仿真器、信号机

图形界面：实现与用户的交互，获取输入数据、实现交通仿真可视化、将仿真结果输出给用户

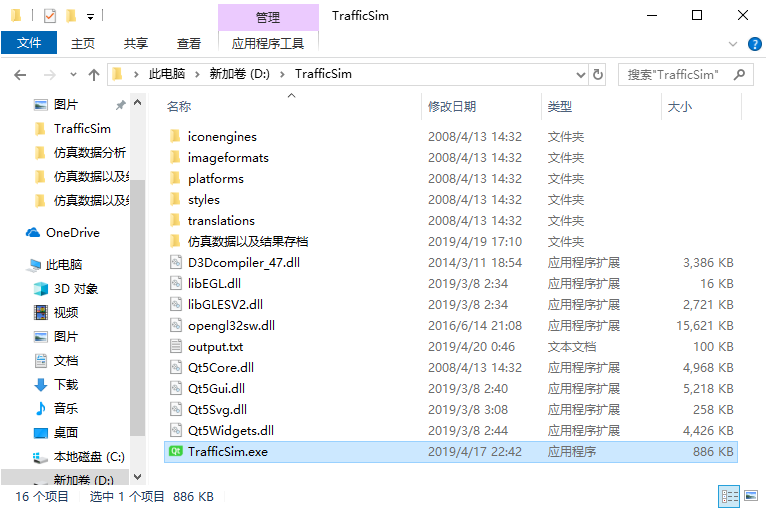
仿真器：实现车辆的移动和碰撞避免、交通规则的建立与实施等，实现车辆和交通信号灯的绘制

信号机：实现交通信号灯的配时以及状态改变，包括了两种场景、两种配时方案的四种组合算法

# 第七章 系统功能演示

## 7.1 安装与初始化

本软件已经将必要的资源文件、动态链接库以及主程序打包好，无需安装。

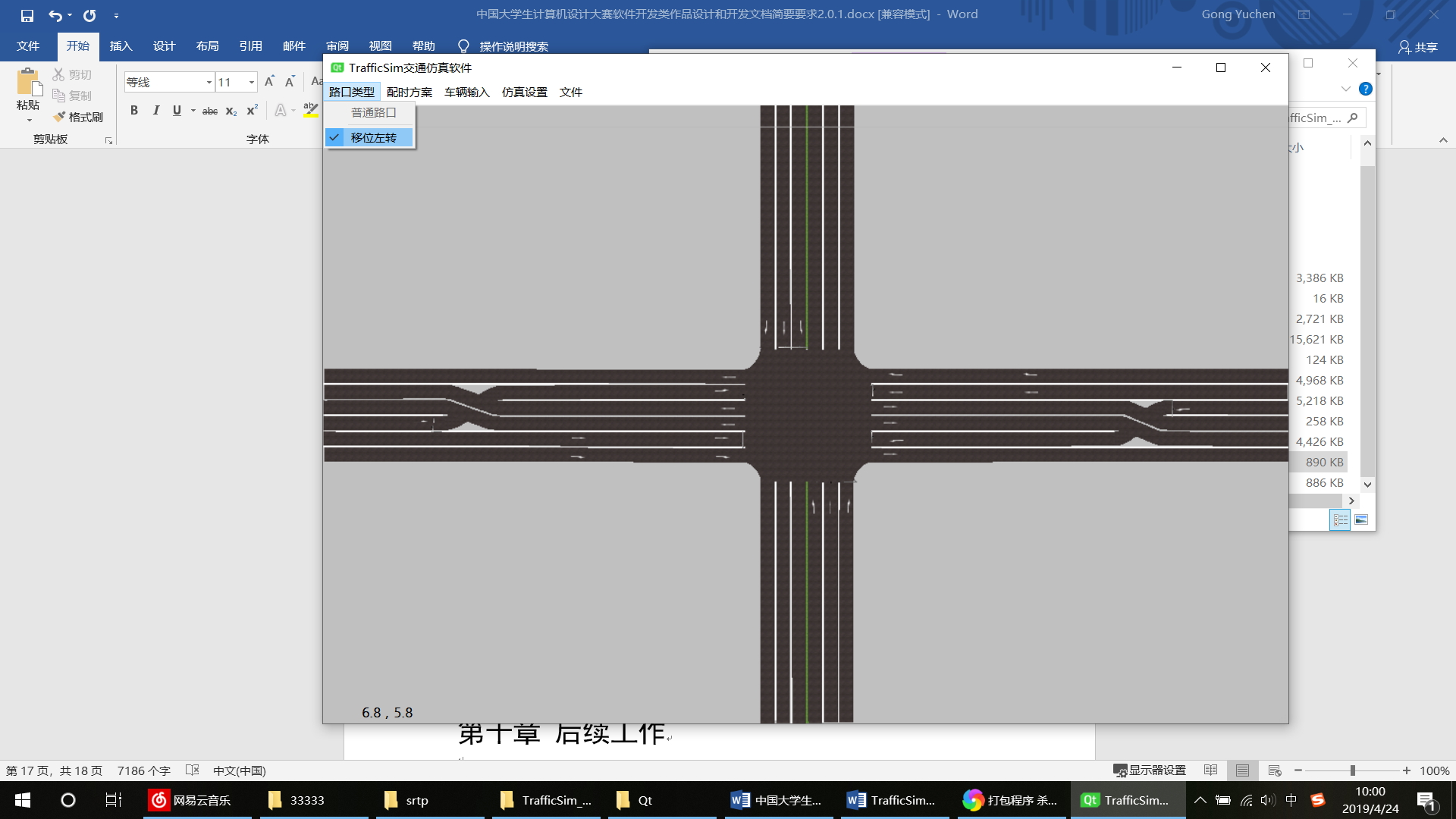
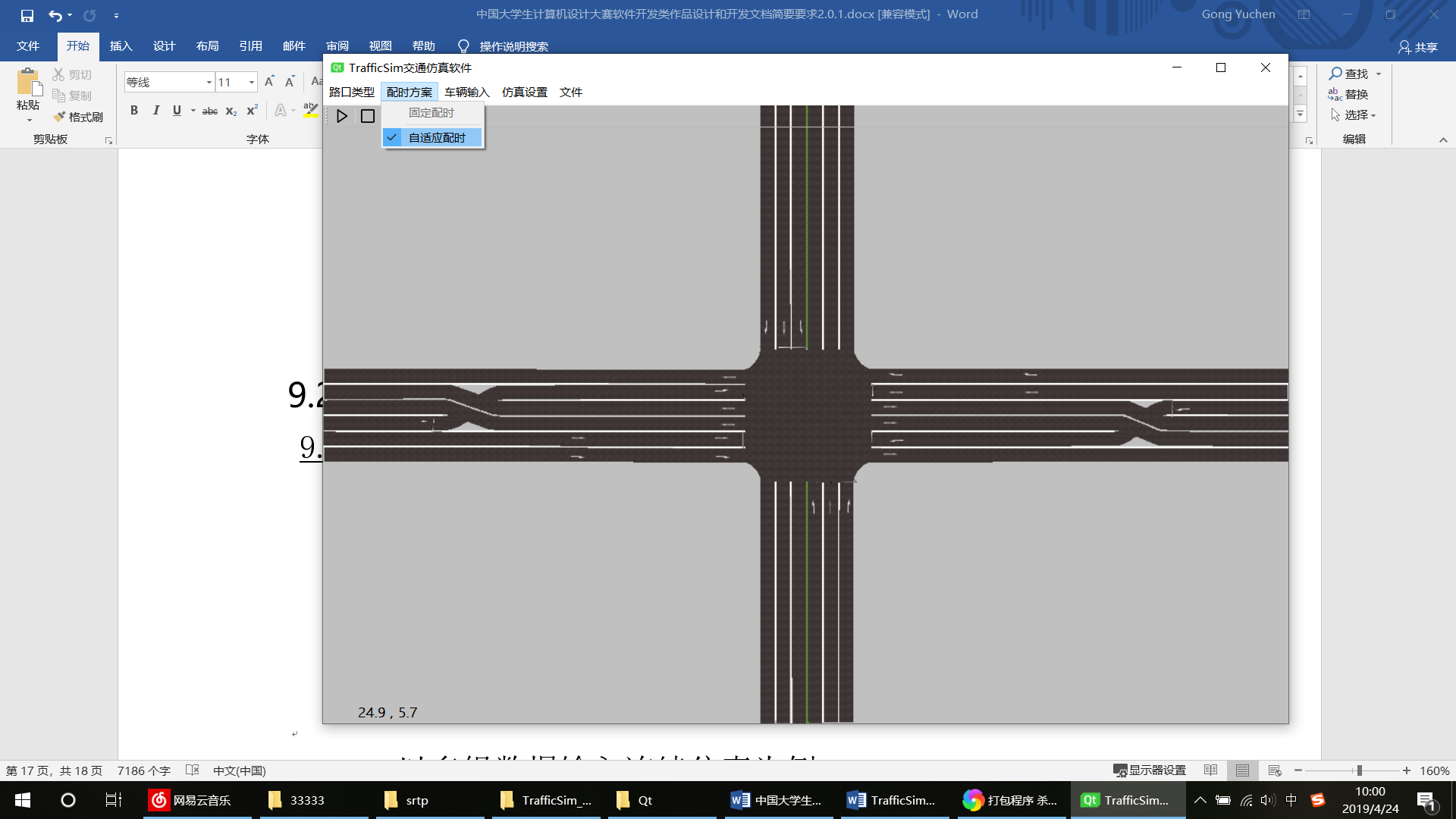


点击程序目录中的TrafficSim.exe文件即可运行程序

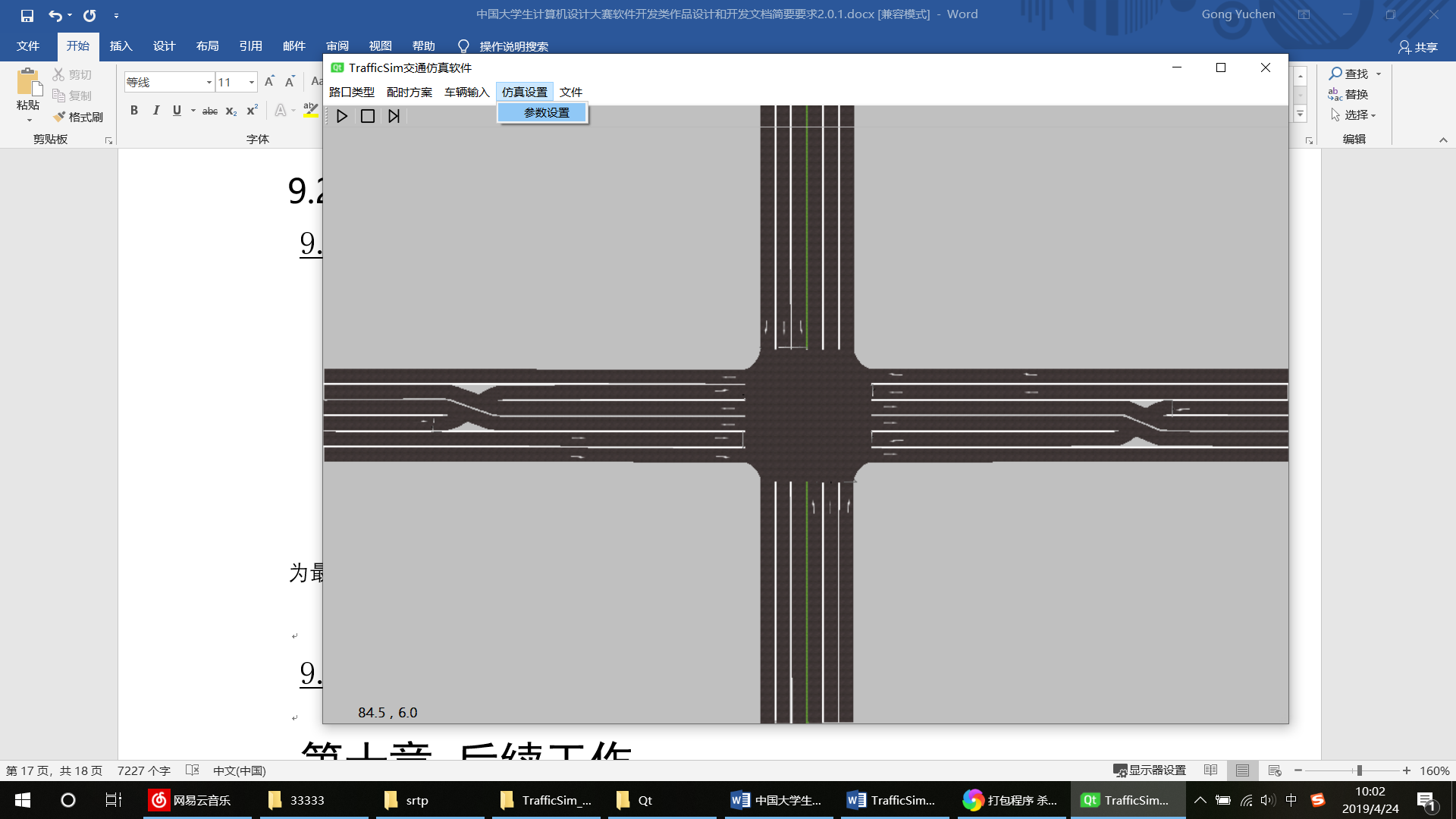
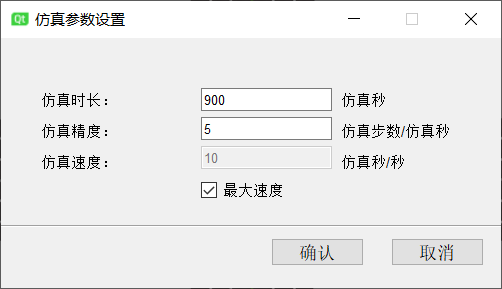
## 7.2 使用说明

### 7.2.1 以单次移位左转自适应仿真为例

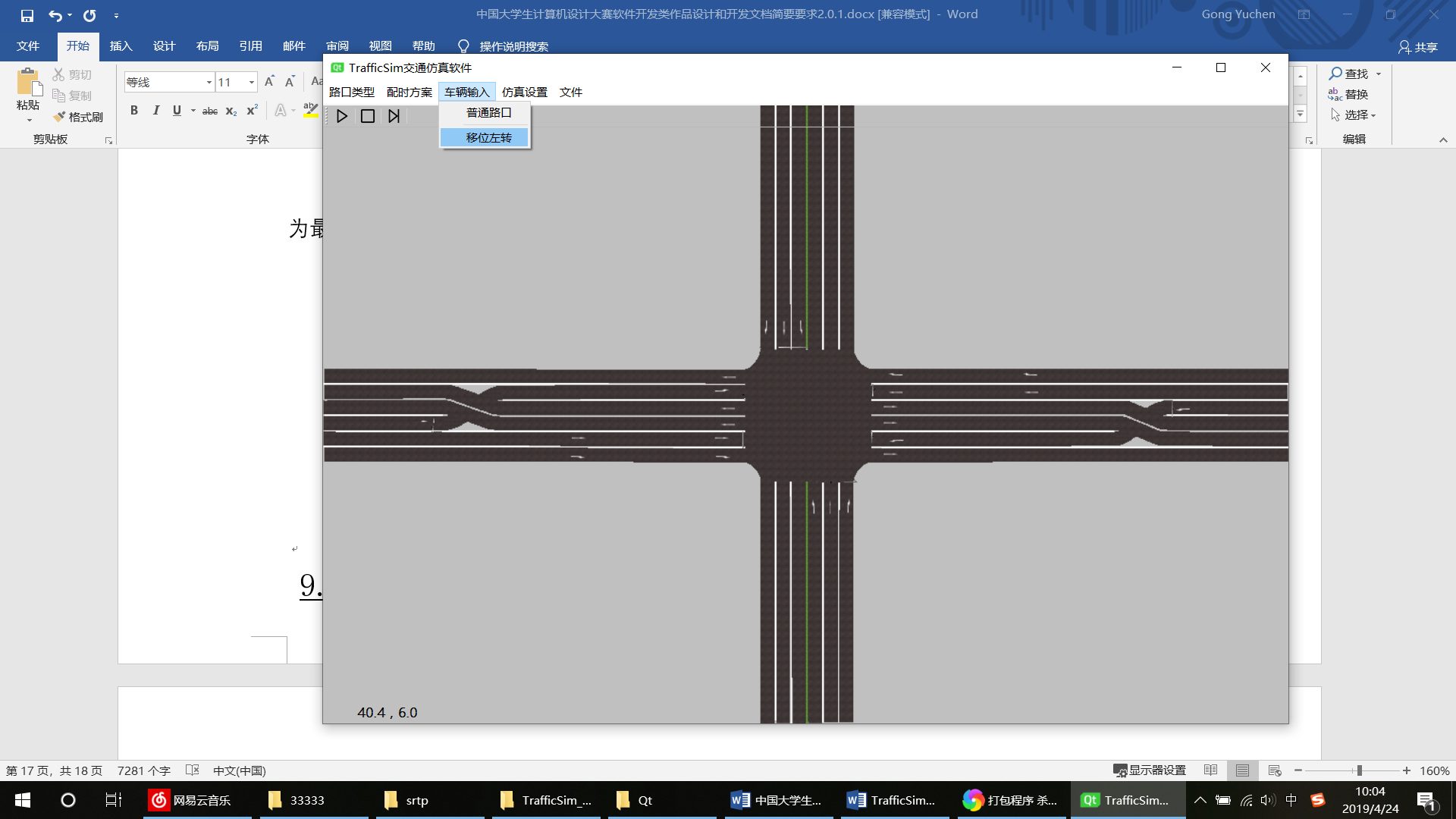
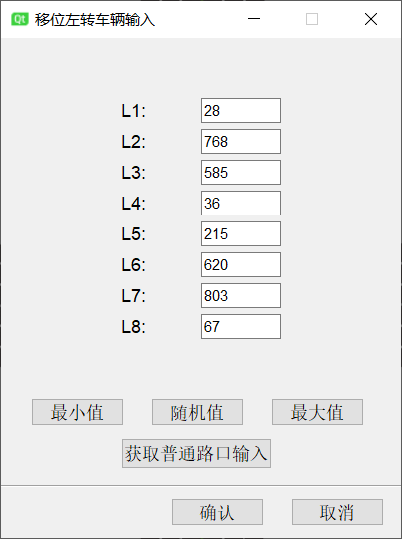
在菜单项中选择路口类型为移位左转，选择配时方案为自适应配时，请确认对应选项前面显示为已勾选

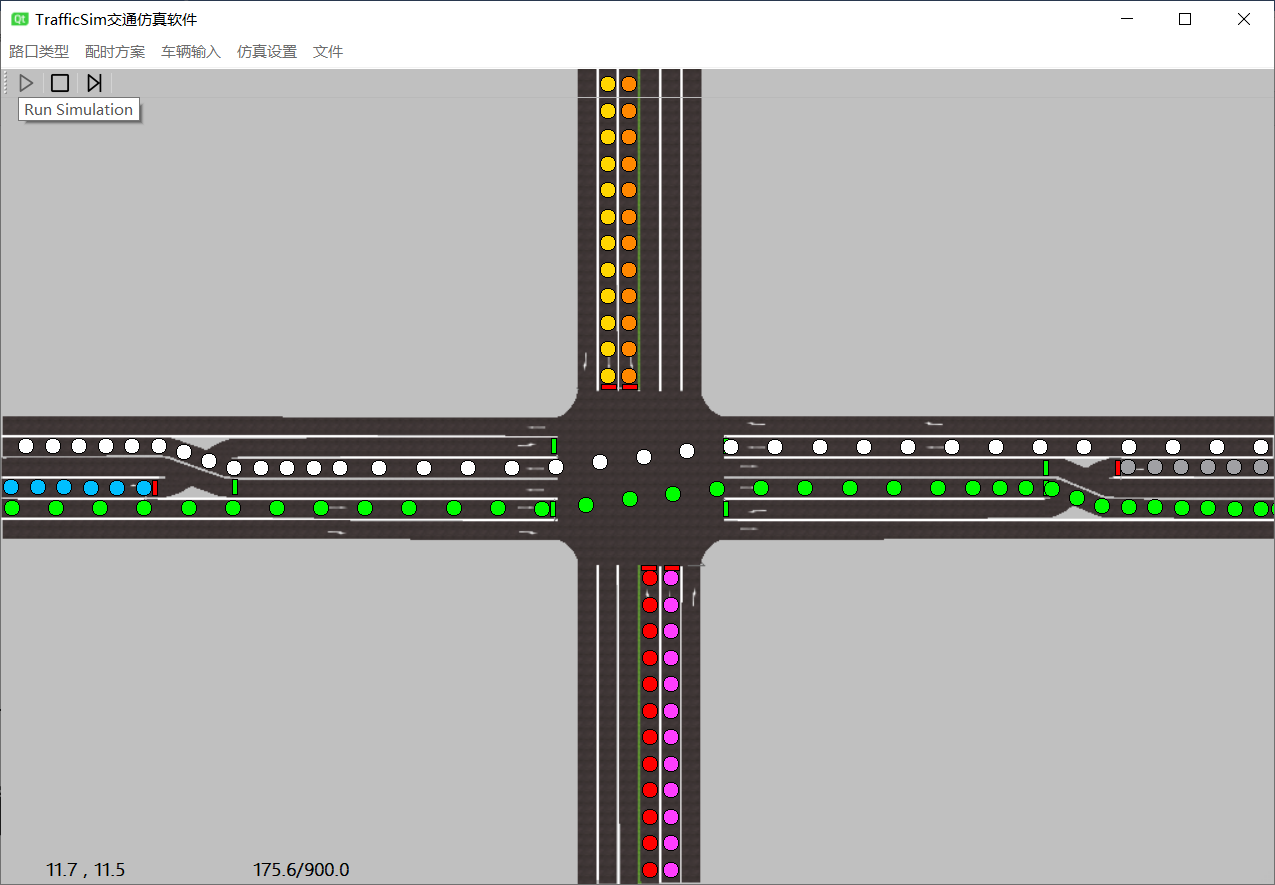
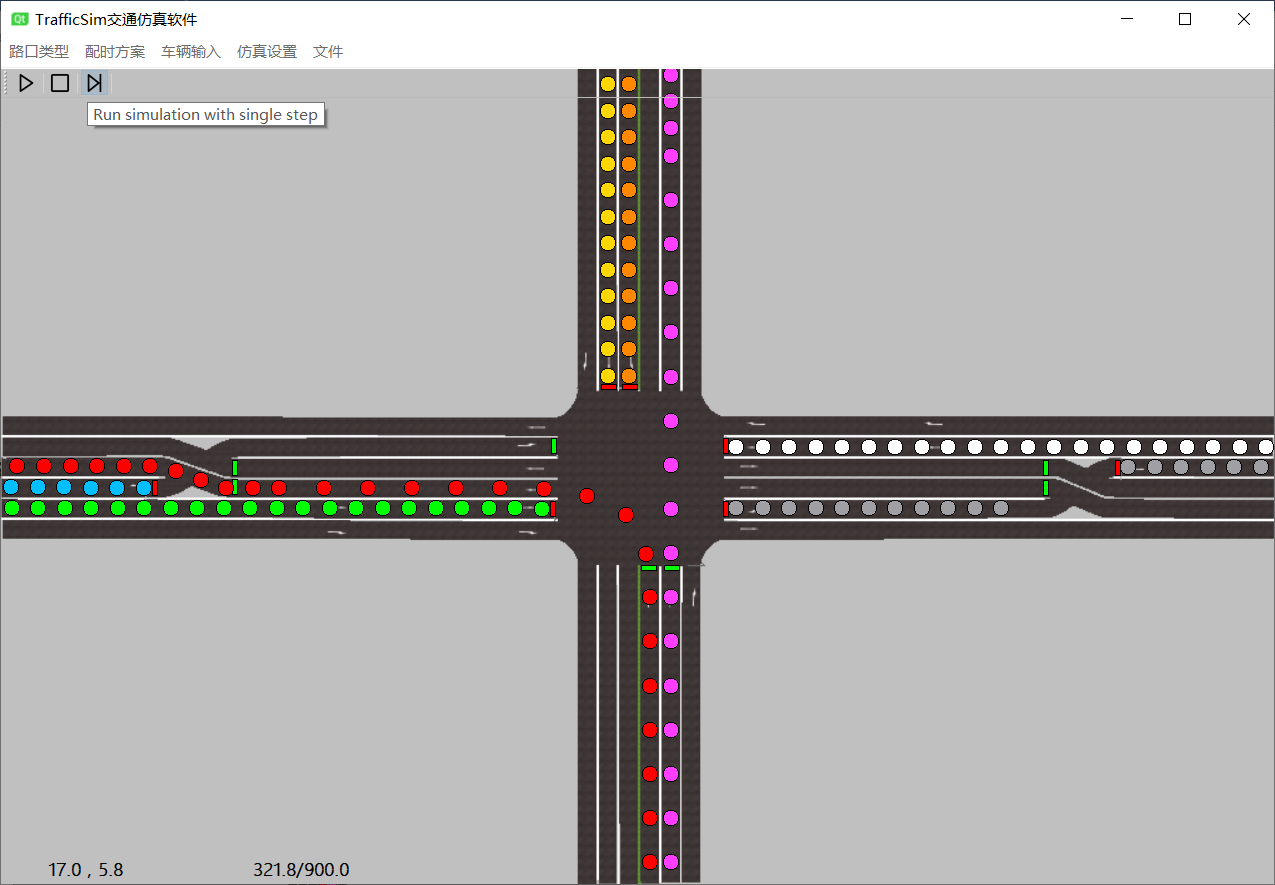
设置仿真参数，这里我们设置仿真时长为900仿真秒、仿真精度为5（推荐）、速度为最大速度，设置完后点击确认

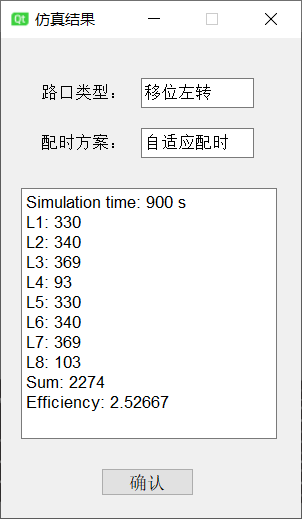
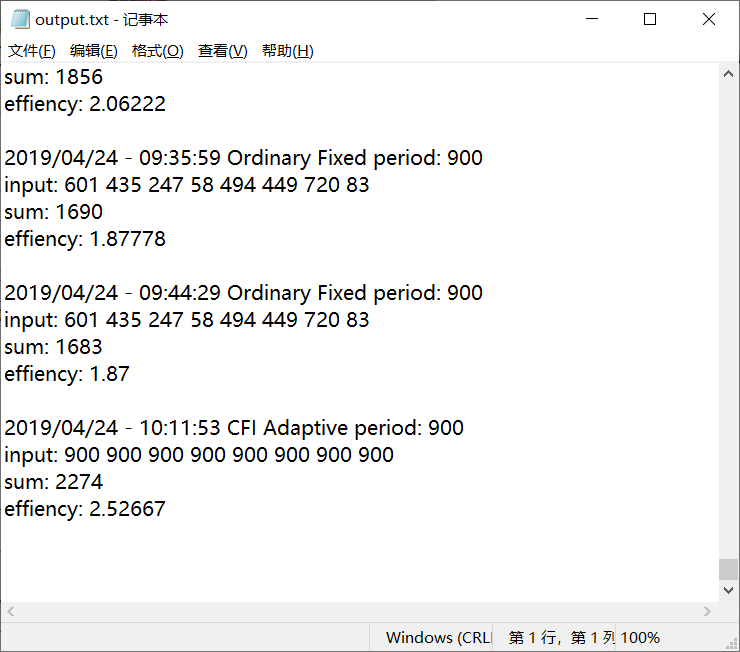
设置车辆输入，可以手动输入，也可以直接点击按钮进行快速设置，设置完成后注意点击确认，如果提示参数设置过大，请减小数据，或者点击最大值来获取最大范围

仿真数据和参数都准备完毕，可以点击仿真按钮开始仿真，仿真过程中可以点击单步仿真按钮，进行单步观察，也可以按下停止键停止仿真

仿真完整完成后，可以显示仿真结果，并将结果记录到文件根目录的output.txt文件中

### 7.2.2 以多组数据输入连续仿真为例

在使用文件进行输入时，只需要设置仿真精度即可

设置好精度过后，通过菜单项打开输入文件，输入文件的内容格式需要按照标准格式；进行文件输入仿真时，软件会自动对同一组数据进行四种不同情境下的仿真并记录结果，直到所有数据仿真完毕，不需要用户进行手动设置，仿真过程中也可以进行单步仿真或者直接停止仿真

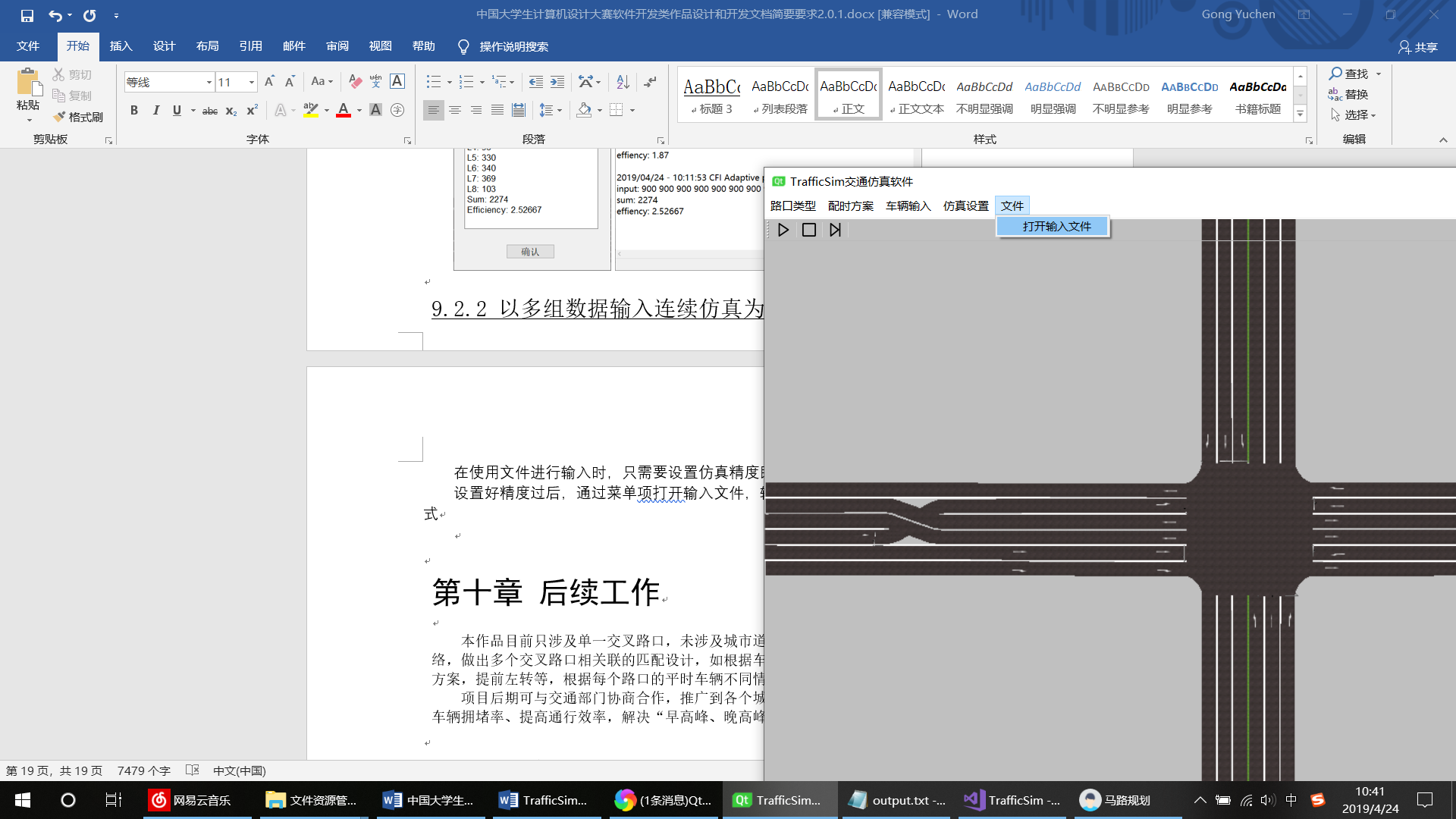
输入文件格式：Period L1 L2 L3 L4 L5 L6 L7 L8

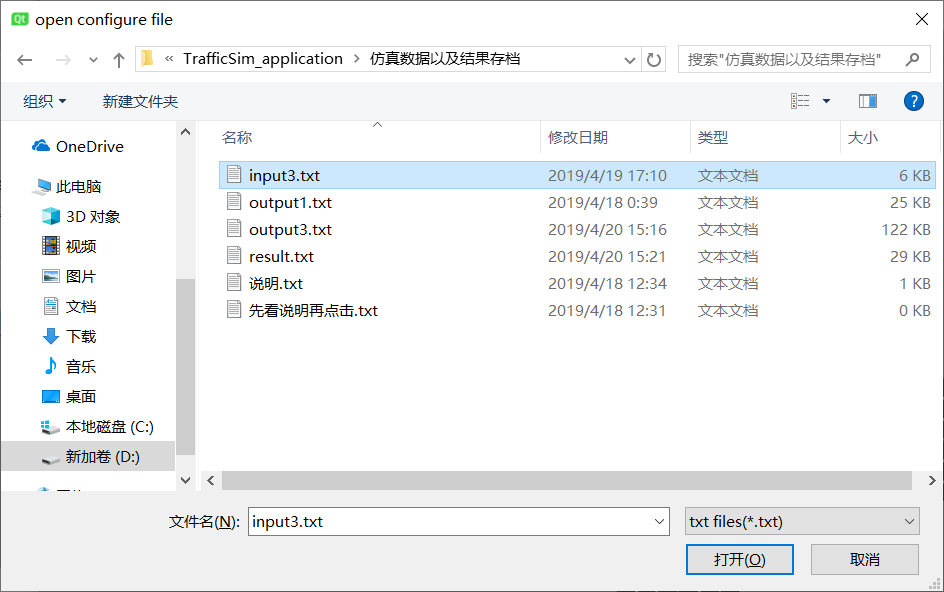
1）数据左对齐，开头结尾可以有空格

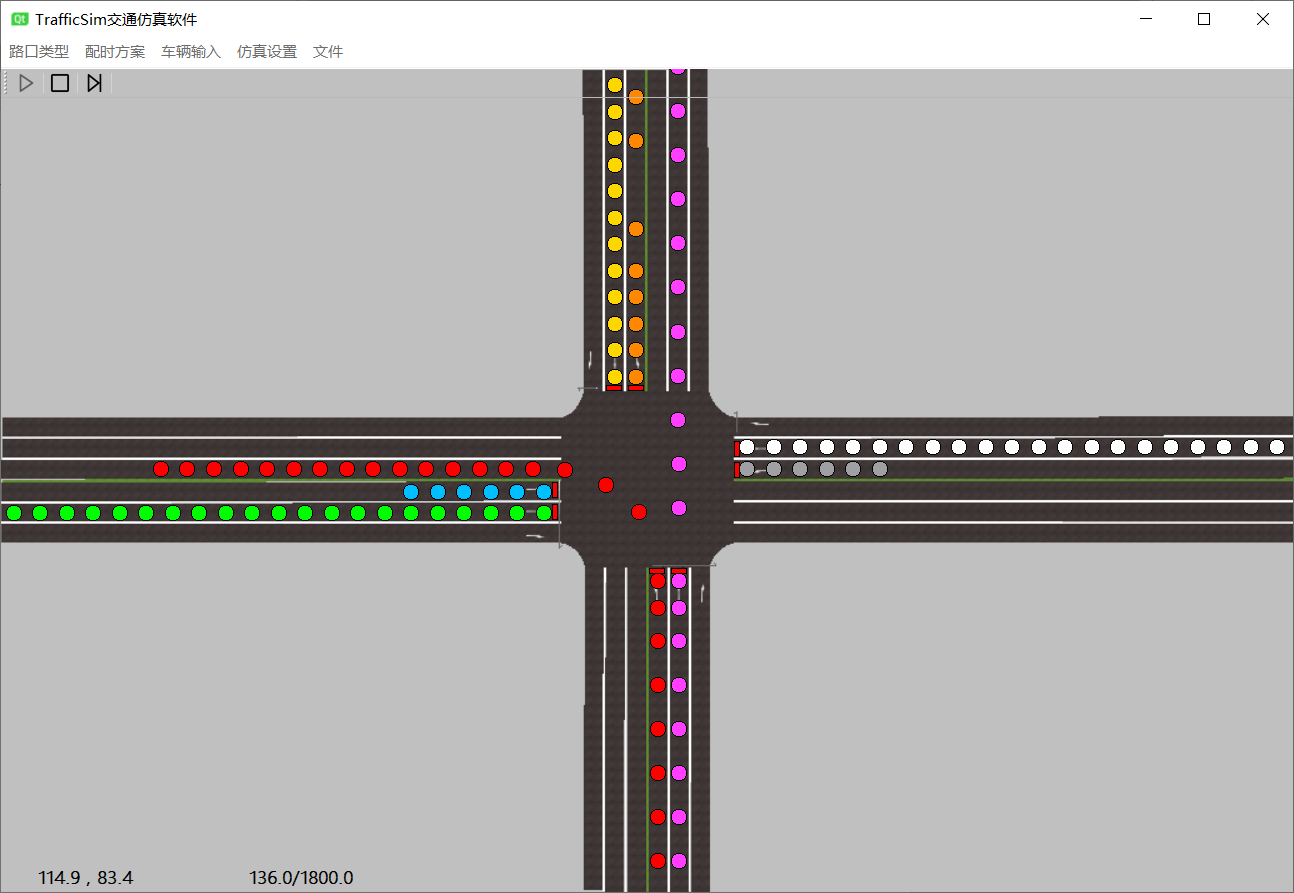
2） Period表示仿真时长，L1-L7分别表示8个车道的输入车辆数，如果超过最大限制，会在开始仿真时自动调整

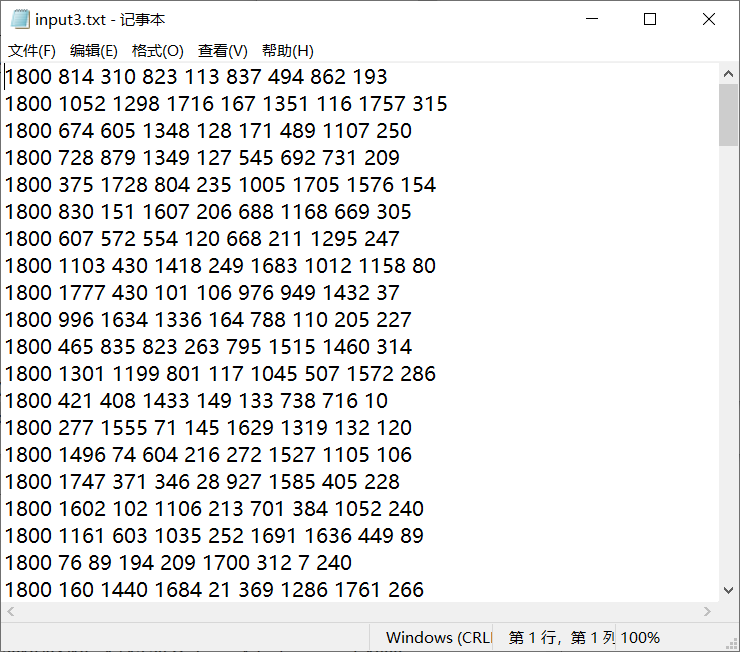
3）每一行为一组仿真数据，可以进行完整的四种情况下的连续仿真并记录结果，每组的单个数据间用空格分开，可以包含多个空格，各组数据间用回车隔开，不能包含空行

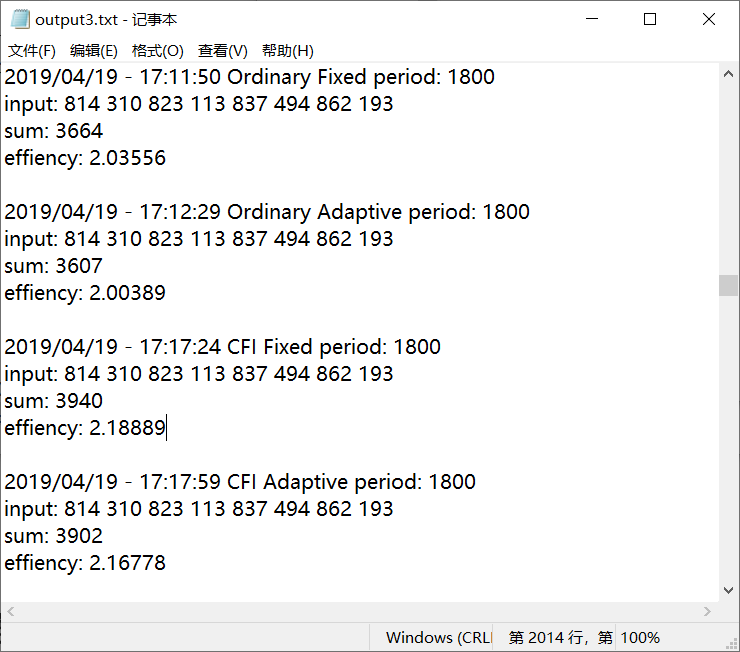
4）软件在读取到空行、格式错误的行、文件结尾会停止读取和仿真











# 第八章 测试报告

## 8.1 引言

### 8.1.1 项目背景

本测试报告为TrafficSim软件测试报告，本报告目的在于总结测试阶段的测试及测试结果分析，描述软件是否达到用户需求的目的。

本报告预期参考人员包括测试人员、软件开发人员。

### 8.1.2 定义

TrifficSim：软件名称，一款可视化的交通仿真软件

Sim.sec：仿真秒单位

Time Steps：仿真步长，一仿真步长进行一次模拟运算并绘制一次实时画面

Ordinary：指普通交叉路口，区别于移位左转

CFI：Continuous Flow Intersection，指移位左转，又称为连续流交叉口，是世界上最前沿的交通组织手段之一

Fixed：固定配时

Adaptive：自适应配时

Period：仿真时长，指在仿真中模拟的时间

Resolution：仿真精度，指每个仿真秒的仿真步长，单位为 Time Steps /Sim.sec

Speed：仿真速度，指每个实际秒进行仿真的仿真秒数

## 8.2 测试基本信息

### 8.2.1 测试范围

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 产品 | 模块 | 子模块 | 功能 | 测试点 | 优先级 |
| TrafficSim | 仿真器 | 交通流控制 | 控制车辆位置更新和绘制 | 1. 车辆行驶时不发生碰撞  2. 车辆行驶时按照交通规则在固定车道行驶  3. 车辆行驶时按照交通规则根据信号灯状态行驶 | 高 |
| 信号灯控制 | 控制信号灯状态改变 | 1. 信号灯显示当时的状态 | 高 |
| TrafficSim | 信号机 | 信号控制算法 | 根据检测到的数据进行之后的周期计算 | 1. 调整算法参数，对比效率提升效果  2. 调整算法方法，对比效率提升效果 | 高 |
| 信号灯控制 | 根据检测到的数据获取信号灯状态 | 1.信号灯返回的状态能正常控制交通流不发生异常状况 | 高 |

本次测试覆盖的范围包括：功能测试、合理性测试、算法优化测试

### 8.2.2 测试要求

a. 要求功能方面的测试点必须正确实现

b. 要求算法方面的测试点尽量有效果上的提升

### 8.2.3 测试用例设计

|  |  |
| --- | --- |
| 测试点 | 测试用例 |
| 1. 仿真器：车辆行驶时不发生碰撞 | 采用Period=900 Sim.sec  Resolution分别采用1、5、10  所有车道输入采用最大值  对两种路口、两种配时方案的四种组合分别进行测试，共测试12次  先采用Max Speed仿真速度观察大致情况，再采用单步仿真模式仔细检查碰撞问题 |
| 2. 仿真器：车辆行驶时按照交通规则在固定车道行驶 | 采用Period=900 Sim.sec  Resolution分别采用1、5、10  所有车道输入采用最大值  对两种路口、两种配时方案的四种组合分别进行测试，共测试12次  采用Max Speed仿真速度观察车辆是否在固定车道行驶（该测试可以再测试点1测试时共同进行） |
| 3. 仿真器：车辆行驶时按照交通规则根据信号灯状态行驶 | 与测试点1采用相同的测试方案 |
| 4. 仿真器：控制信号灯状态改变 | 采用Period=900 Sim.sec  Resolution任意  所有车道输入采用最大值  对两种路口的固定配时分别进行重复测试，每个测试5次  采用Speed=1 Sim.sec/s  观察信号灯是否在指定时刻显示对应的状态颜色 |
| 5. 信号机：调整算法参数，对比效率提升效果 | 对于多个可以设置的参数，分别采用随机生成的50组数据，使用txt文件输入的方法进行持续仿真，并对比采用不同参数时的综合仿真结果 |
| 6. 信号机：调整算法方法，对比效率提升效果 | 对于多种可以采用的方法，分别采用随机生成的50组数据，使用txt文件输入的方法进行持续仿真，并对比采用不同参数时的综合仿真结果 |
| 7. 信号机：根据检测到的数据获取信号灯状态 | 采用随机生成的5组数据，采用  Period=900 Sim.sec  Resolution=5  Speed=10 Sim.sec/s  对两种路口在两种配时方案的四种状况下分别进行仿真，观察是否发生不合理的交通灯状态 |

## 8.3 测试情况

### 8.3.1 测试时间

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 测试阶段 | 计划开始时间 | 计划结束时间 | 实际开始时间 | 实际结束时间 | 计划工作量 | 实际工作量 |
| 一. 测试基本功能 | 2019/4/10 | 2019/4/12 | 2019/4/11 | 2019/4/13 | 测试点1、2、3、4、7 | 测试点1、2、3、4 |
| 二. 测试算法参数并优化 | 2019/4/13 | 2019/4/18 | 2019/4/15 | 2019/4/20 | 测试点5 | 测试点5、6、7 |
| 三. 测试算法方法并优化 | 2019/4/19 | 2019/4/24 | 2019/4/20 | 未结束 | 测试点6 | 测试点5、6 |

### 8.3.2 测试用例统计

对于功能性的测试点，不出现任何不合理的情况视为成功

对于算法测试点，我们认为成功的条件要满足通行效率：普通路口固定配时≤普通路口自适应配时≤移位左转固定配时≤移位左转自适应配时

实际上，经过数据分析统计，采用不同的参数或者方法的实现都可以统计出仿真结果的平均效率满足以上条件

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试点 | 执行案例总数 | 成功个数 | 失败个数 | 案例成功率 |
| 1 | 12 | 3 | 9 | 25% |
| 2 | 12 | 12 | 0 | 100% |
| 3 | 12 | 12 | 0 | 100% |
| 4 | 10 | 10 | 0 | 100% |
| 5 | 50x4 | 20/17/28/25 | 30/33/32/25 | 40/34/56/50% |
| 6 | 50x3 | 20/23/20 | 30/27/30 | 40/46/40% |
| 7 | 20 | 20 | 0 | 100% |

## 8.4 问题统计

### 8.4.1 发现的问题

a. 仿真时存在大量不同车道的车辆在路口中间碰撞情况，已解决，新的测试结果：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试点 | 执行案例总数 | 成功个数 | 失败个数 | 案例成功率 |
| 1 | 20 | 20 | 0 | 100% |

b. 对参数设置为几个不同的值，得到的效果不同，已将参数设置为用例成功率最高的参数

c. 对算法中部分方法采取不同的实现方式，得到的仿真效果不同，已将方法采用微用例成功率最高的方法

### 8.4.2 未解决问题及问题分析

我们认为算法仍有提升的空间，参数还可以在范围内进行变动，也可以设计出其他的实现方法，因此将算法问题设为未来的主要问题，进行后期的改和深入研究

## 8.5 测试结论及建议

本项目根据项目组成员讨论出的功能需求以及其他测试人员的反馈意见，覆盖了所有的测试需求以及案例，均已测试完成，目前测试检验出的问题都已得到有效解决。

综上所述，TrifficSim软件需求已经达到标准，本次测试通过，可以进行验收测试

# 第九章 后续工作

本作品目前只涉及单一交叉路口，未涉及城市道路网络，项目进一步会结合城市道路网络，做出多个交叉路口相关联的匹配设计，如根据车辆通行方向以及车辆数目提前规划道路方案，提前左转等，根据每个路口的平时车辆不同情况设计多种产品。

项目后期可与交通部门协商合作，推广到各个城市的主要大流量交通路口，将明显降低车辆拥堵率、提高通行效率，解决“早高峰、晚高峰”问题。

# 第十章 项目总结

通过这次项目的开发，我们学到了很多，收获了很多。

首先是项目的讨论，任务的分工。我们在明确了课题之后，有了一个大致的思路和设计雏形，但是对于需求分析，实现过程和方法，具体实现的成品形象都没有太明确的概念。通过多次和教授的讨论，团队对于整体任务的算法和重难点大致有了印象，同时根据比赛的时间，将任务根据难度分块，并划分了几个时间段，预计在各个时间段完成相应内容。

第一阶段是完成普通路口的设计算法，相较于移位左转的复杂路口设计，普通路口更为简单以及具有参考性。同时根据相位的基本概念，讨论设计了车道组合，时间周期以及效率评比的相关概念。同时遇到了一些问题，有些是仅仅针对普通路口的，有的是共同的有关时间缩减比例的与之后算法设计的相关问题。对于当时可以处理的，我们尝试和教授多次的沟通和下来自己讨论解决了，对于一时间难以解决的，我们在后续的设计中继续考虑解决方案。

第二阶段是完成左转路口的设计算法，普通路口的设计成功以及算法代码的顺利执行给了我们很大的自信，但是左转路口复杂程度比之前的大得多。普通路口的自适应算法对于左转路口的自适应是有参考意义的，同时左转路口算法有它自己独特的难点，我们通过类比，将车道组合转化为相位组合，在周期内循环，同时设计了新的效率评比方式。

第三阶段是软件的编写，基于算法的基础上，重新设计一个交通道路软件，包括车辆的表现形式，道路的设计，以及车辆行走和停止的判断。同时在编写的时候也遇到各种头痛的问题包括红绿灯的表现形式，车辆的碰撞检测等等。都是在很多次长期不断的讨论中，尝试了多种方法后得出了处理方案。同时最后我们处理了之前遇到的残余问题。

这一过程中，我们真切了解到了讨论的重要性以及头脑风暴的强大之处，充分交换彼此的意见往往可以得到更多的意见。同时计划的按时执行和合理分配也是重要的一环。

其次是能力的提高，通过项目的开发，每个人都得到了不同程度能力的收获。像是对一个需求的理解和分析，对于项目边界的明确，对任务量概念的明确。同时，还收获了代码的编写能力，软件的编写能力，团队的协调配合能力，以及论文的撰写，查询资料和信息的能力。