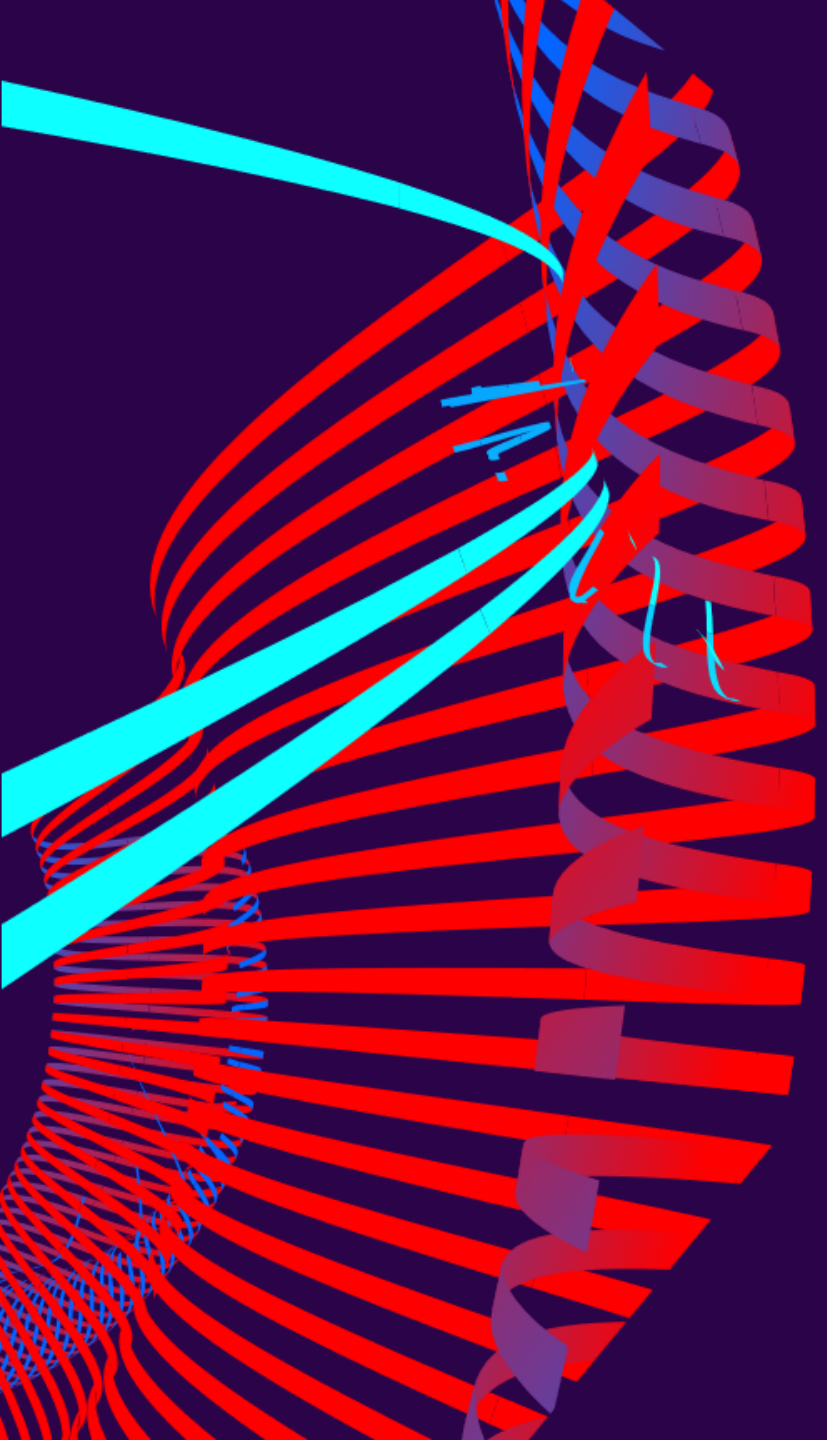
CONCURRENT TRAFFIC SIMULATION

Multi Threading

CONTENTS

OVERVIEW



Traffic simulation in which vehicles are moving along streets and are crossing intersections. However, with increasing traffic in the city, traffic lights are needed for road safety. Each intersection will therefore be equipped with a traffic light. In this project, i build a suitable and thread-safe communication protocol between vehicles and intersections to complete the simulation. Use The knowledge of concurrent programming (such as mutexes, locks and message queues) to implement the traffic lights and integrate them properly in the code base.

PHASES OVERVIEW

Every Phase Either Add New Features Or Fixes Problems in the Previous Phase

Phase 1

Put in The Fundamental Base

Describes The Purpose of every class

Phase 2

Communications Between Vehicles And Intersections

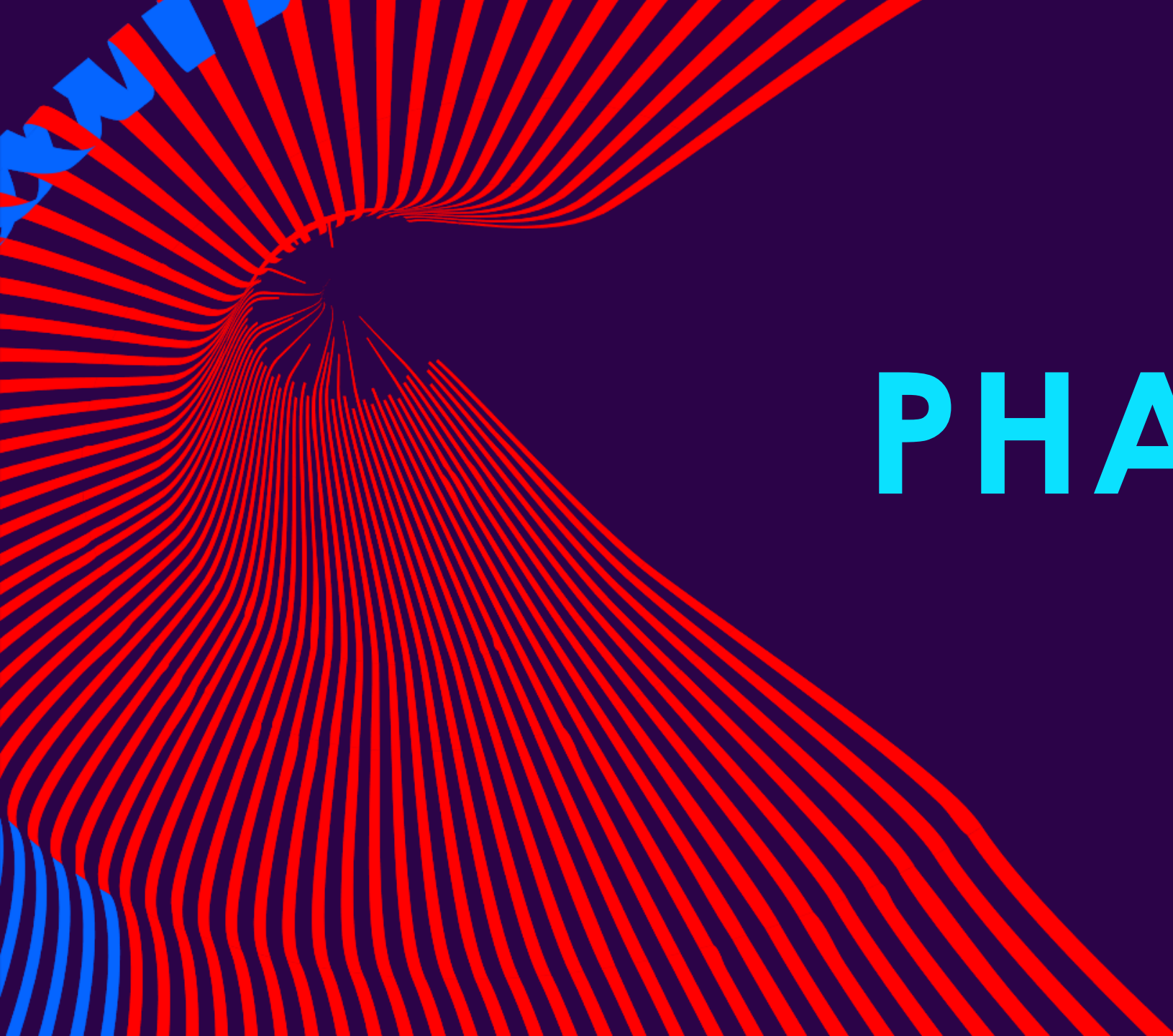
To increase Safety

Phase 3

Reduce expenses for replacement products

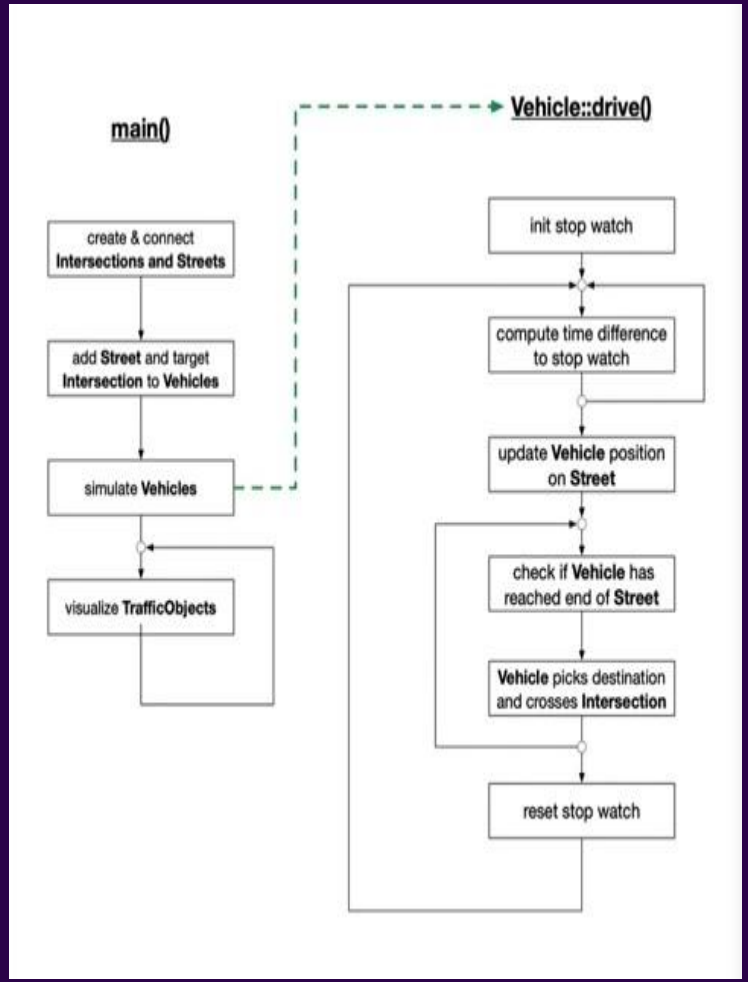
Final phase

Simple design that gives customers the targeted information they need



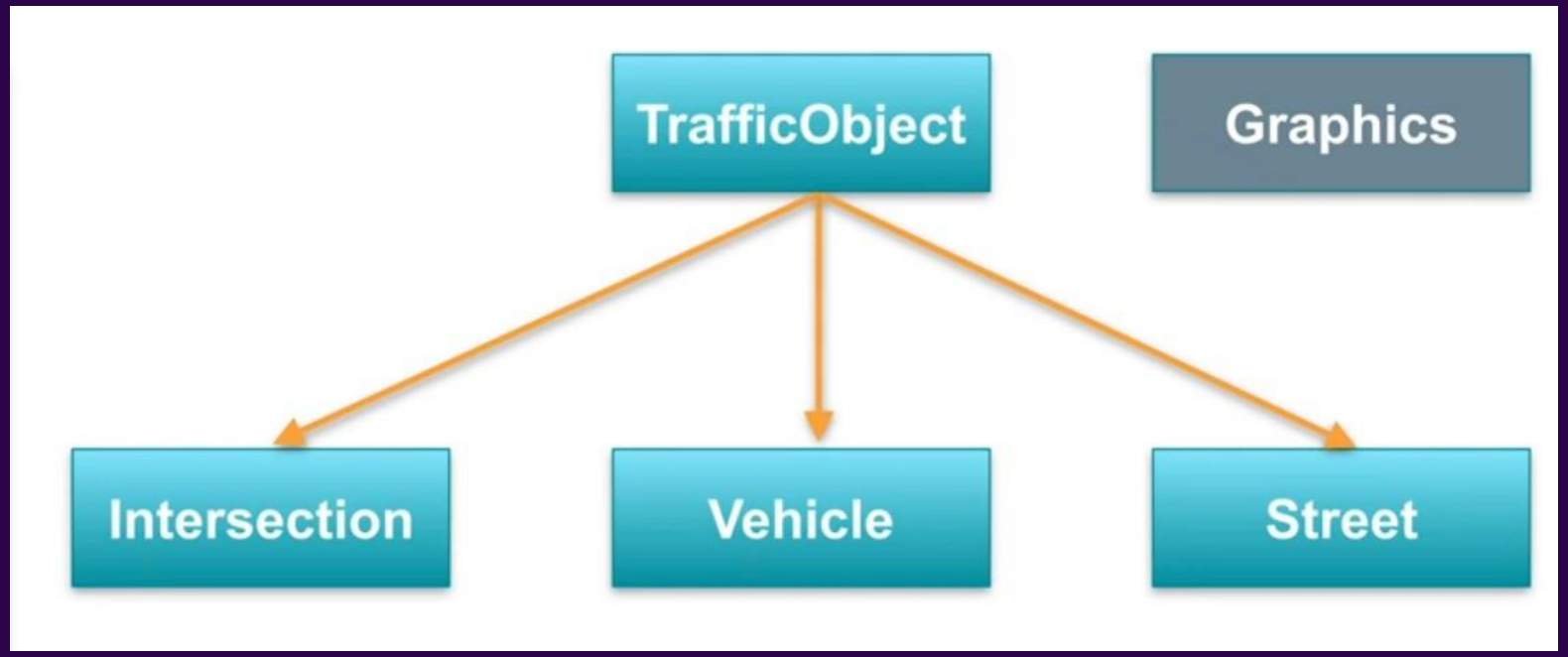
PHASE 1

Main Flow of Phase 1



PROGRAM STRUCTURE

- Graphics.cpp
- Graphics.h
- Intersection.cpp
- Intersection.h
- nyc.jpg
- Street.cpp
- Street.h
- TrafficObject.cpp
- TrafficObject.h
- TrafficSimulator-L1.cpp
- Vehicle.cpp
- Vehicle.h



PHASE 1 SUMMARY

TrafficObject(Parent Class)

-I first created A enum that describes all possibilities of object type (No Object , Vehicle , Intersection , Street)

-And Created Three Attributes That describes The Object (Type of Object , ID , Position (x & y) , Vector of threads to contain all threads that have been launched by this Thread)

-And I add additional attribute (idcnt) , To count id of object So every object has its unique ID

-And At the End I created the destructor To make a thread Barrier To make sure that all the threads Joined successfully .

Intersection(Child Class)

-I first created a Attribute Name (_Streets) to Detect all the streets connected to this intersection

-Then I create Constructor To Define The object type (intersection Type) , And two methods

1-AddStreet : To add street to vector _Streets

2-queryStreets : To return the (outgoing) Streets

Vehicle(Child Class)

- create Constructor To Define The object type (Vehicle Type)
- Method SetCurrent Street/Destination() : To determine The current Street And Destination
- Method Simulate() : It’s the Method Which is responsiple To send The threads To Drive Method
- Method Drive() : It’s the Main Method here And The Majority of the work is done here Which Do :

1- initialize Stop Watch

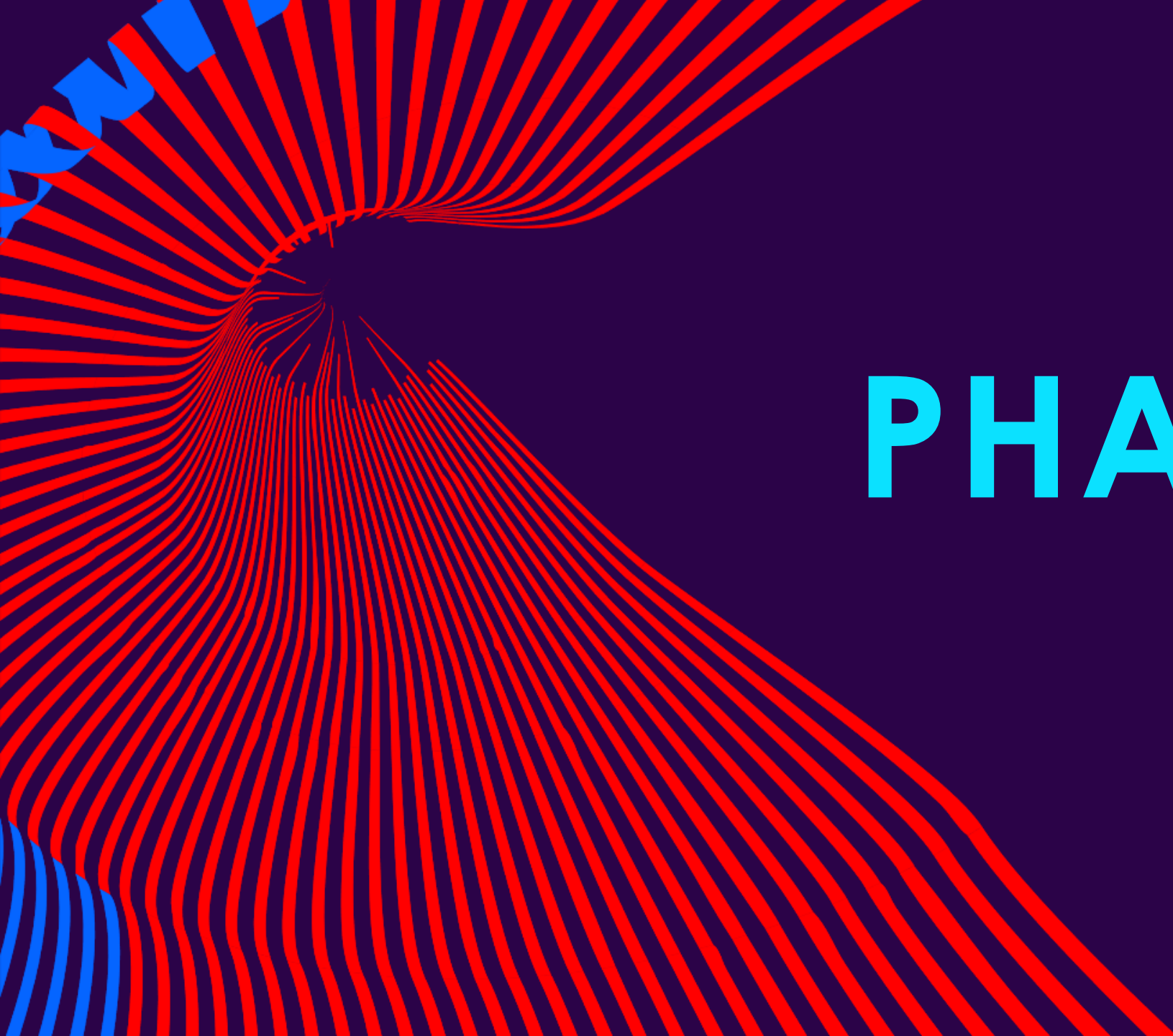
2-enter infinite Loop

3-Update Position

4-Slow Down Speed when when we reach 90% of destination

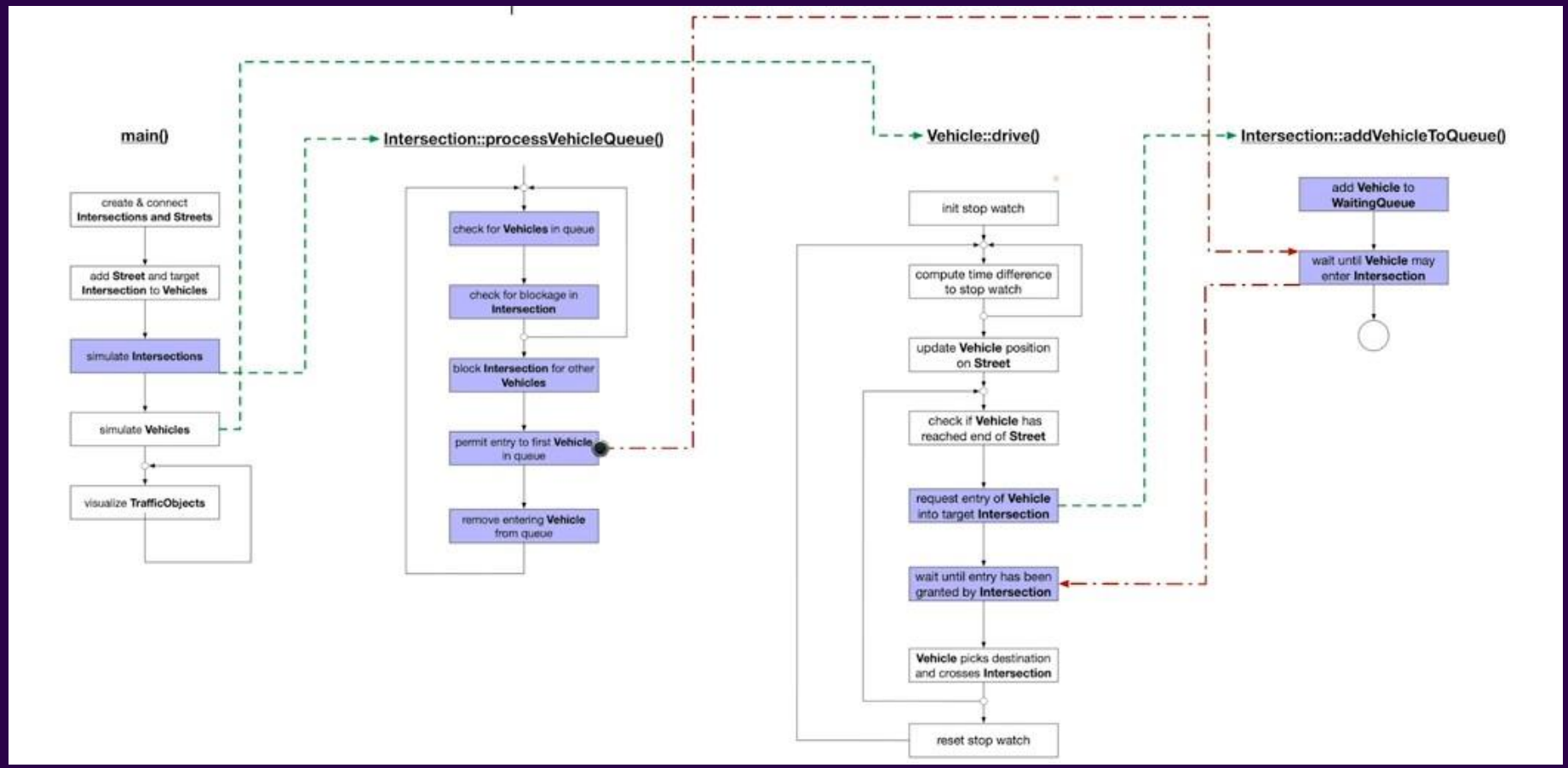
5-if we reached the destination we will randomize the next street

6-Reset The speed And reset Stop Watch



PHASE 2

Main Flow Of Phase 2



THE NEW FEATURES THAT THE SECOND PHASE ADDED

We solved a dangerous problem we faced in the previous Phase and its that (More than one car can enter the intersection in the same Time , So in this Phase I added new feature (Vehicles Waiting Queue) , And permitted to Only one car to Enter the intersection By creating new Function intersection Simulate To send A thread that process That queue And it will be discussed in the next couple of pages .

FILES MODIFICATIONS

1-TrafficObject : There is no modifications Since I did not add any New Object

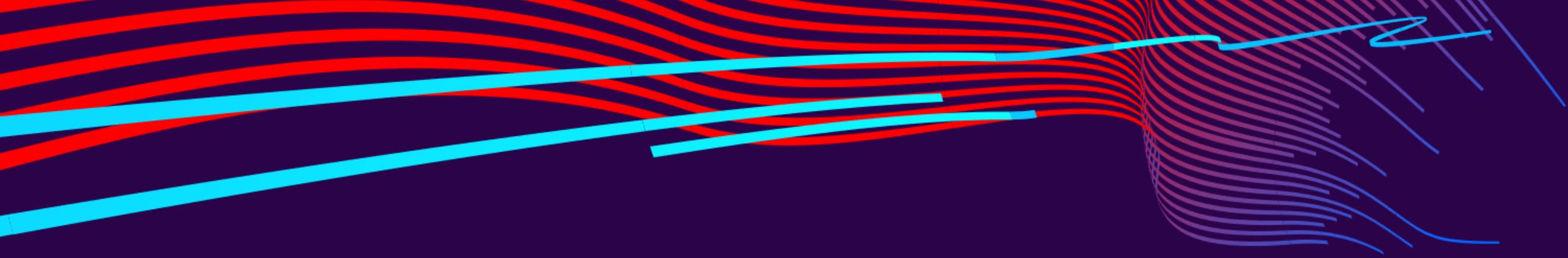
2-Vehicle : When The vehicle About to enter The intersection I added new Feature , I make A task

To addVehicleToQueue Function And Waited to make Sure the vehicle can enter the intersection

3-Street : There is no modifications

4-intersection : (Main modifications is done here)

First I added new Class (WaitingVehicles) And it represents The Queue Which Has :



- **Two Attributes :**

  -(_Vehicles) Which represents the Waiting Car to enter the Queue

  -(_Promises) Which represents list of associated promises

- **Three Method :**

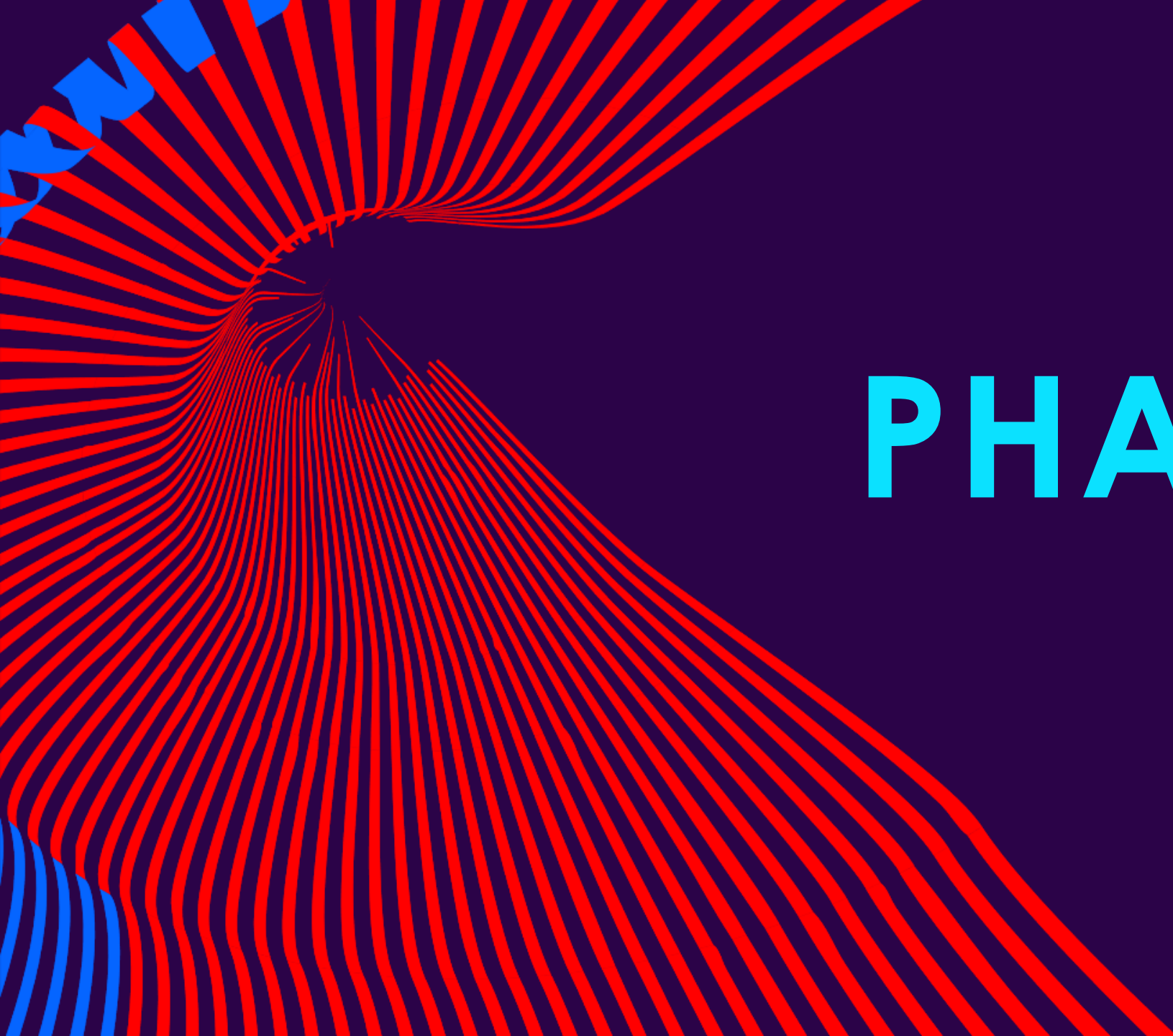
  -(getSize()) Which Return the Number of Vehicles in the queue

  -(permitEntryToFirstInQueue()) Which Permit the first Vehicle To enter intersection

  - (pushback()) Which Add new Vehicle To the end of the Queue

And in The intersection Class I added :

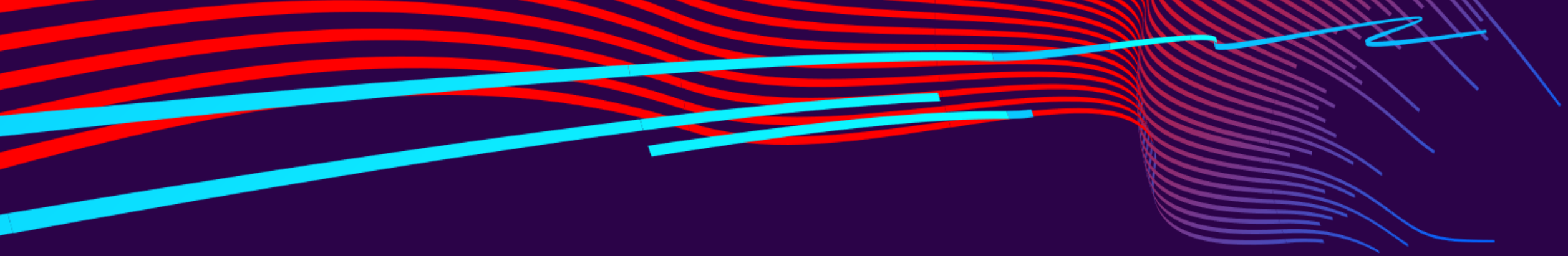
- **New Attribute (isBlocked) :** Which represent The intersection is blocked Or not
- **New Four Methods :**
  - **setIsBlocked() :** To change the Stat of the intersection (BLOCKED / FREE)
  - **addVehicleToQueue() :** To Push Back The Vehicle in Queue And its associated promise
  - **simulate() :** virtual function which is executed in a thread
  - **vehicleHasLeft() :** Used To Convert The Stat of intersection to FREE



PHASE 3

THE NEW FEATURES THAT THE SECOND PHASE ADDED

At this Point The Traffic Simulation Seems To be Working Nicely , How ever , There Is a data race hidden in the code : The Access to the shared Vectors (_Vehicles) And (_Promises) in class WaitingVehicles is not protected , this bug will be corrected with this Phase . Also , Access to the console shall be protected so that Printed Strings are Not mixed up .So we will See The modification To the Files in the next couple Pages .



First in (WaitingVehicles Class) i applied Locking Mechanism to Private members (_Vehicles & _Promises) To Avoid Deadlocks.

Then i Created new Static Mutex (called _mtxcout) To use it in the next Step .

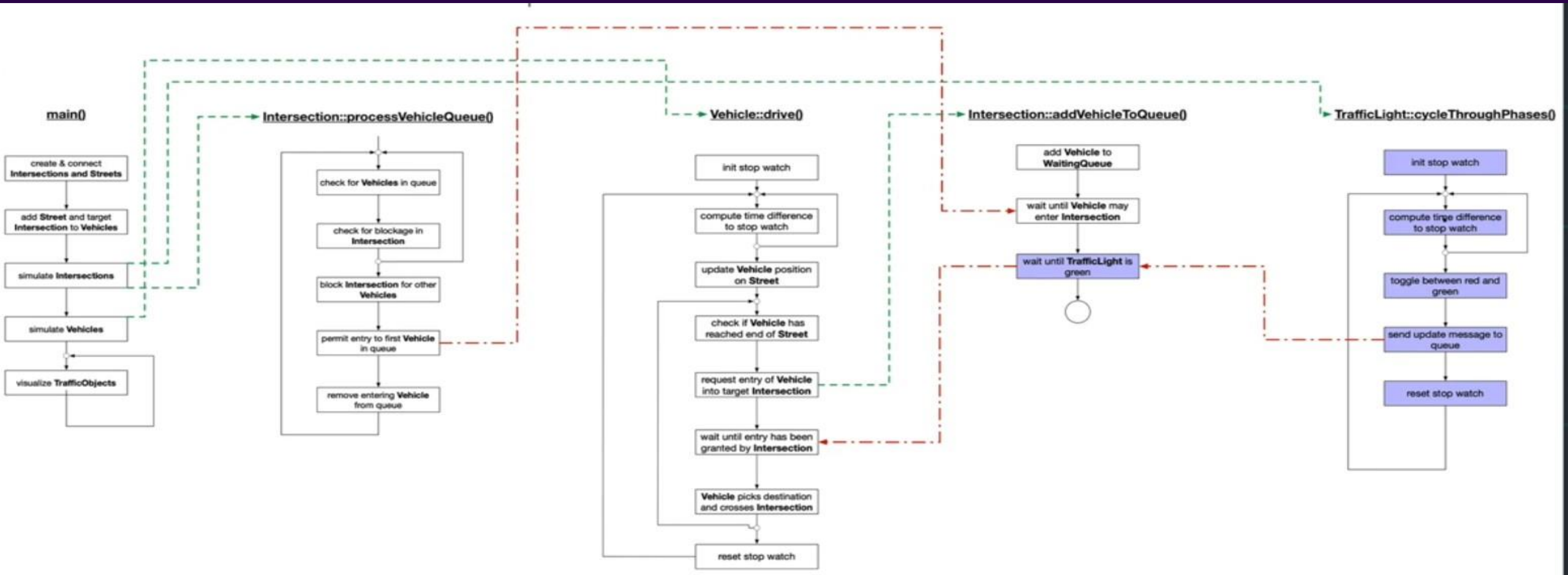
Finally in (addVehicleToQueue) I ensured that the text ouput locks the console as a shared resource , Using The mutex (_mtxcout)

FINAL PHASE

THE NEW FEATURES THAT THE SECOND PHASE ADDED

The Traffic simulation in its current state is working fine And vehicles are moving along streets and are crossing intersections . However, with increasing traffic in the city , traffic lights are needed for road user safety . Each intersection will therefore be equipped with traffic light . In the project , a suitable and thread-safe communication protocol between vehicles , intersections and traffic lights needs to be established.

MAIN FLOW OF FINAL PHASE



NEW CLASS(TRAFFIC LIGHT)

Added New class Traffic Light , To organize The traffic , The intersection switches between RED and GREEN , Green means vehicle can pass , Red means the vehicle cant pass , And I created Message Queue to communicate between vehicles , And I random The intersection every 4 to 6 seconds it changes its state from red to green . Through the simulate function .

THANK YOU

Abdelrahman Tarek Mahmoud

Code :

https://github.com/AbdelrahmanTarekMahmoud/CppND-Program-a-Concurrent-Traffic-Simulation

Youtube :

https://www.youtube.com/watch?v=RkE2Qnw3BkU