

Design Studio #2

Part 1

Ankit Jain

Cory Matthew Cunanan

David Quang Bui

Michael C. Hernandez

Slavyana Naydenova Nedelcheva

TABLE OF CONTENTS

NO.	TOPIC	PAGE(S)
1.	Global Design Decisions	1 – 2
2.	Design Approach #1	2
3.	Design Approach #2	2 – 3
4.	Design Approach #3 – Main Approach	3
5.	Design Approach #3 – UML Diagram	4
6.	Design Approach #3 – Pseudo Code/Algorithm	5 - 9

Global Design Decisions

Audience

- Students
- Professor E

Secondary Stakeholders

- Professor E
- U.S. Regulations and traffic laws

Design Goals

- Create an educational traffic flow simulator program.
- Create a visual map of an area, laying out roads in a pattern
- The visual map should be displayed to the user in real-time as he/she emerges in the situation.
- Specify the interaction mechanism that will control the behavior/pattern of the traffic lights at each of the intersections.
 - A variety of sequences and timing schemes should be allowed.
- The time that people spend at the traffic light should be minimized by the interaction mechanism that applies to that traffic light.
- Create a traffic light pattern that will allow cars to flow through the intersection from each direction in a fluid manner to avoid any crashes
- The visual map should display the current state and of the intersection traffic lights and update the change.
- The simulator would have an option to change the incoming traffic density by controlling how many cars enter the simulation at each of the roads on the edge. There are:
 - Busy roads
 - Seldom used roads
 - And in between those two
- Display any problems with map's timing scheme, be able to change them and display the changes.
- The application aims to simply illustrate the impact that traffic signal timing has on traffic.
- Determine changing the time with a few second what effects has on the next traffic light.

Design Constraints

- Each simulation should consist of at least six separate intersections.
- Combinations of traffic light states that would result in car crashes on an intersection are not allowed.
- Every intersection on the map must have traffic lights.
- There are no stop signs, overpasses, or other variations of these kinds.
- All intersections must be four-way, and no "T" intersections and one-way roads.

- Only one simulation can run at a time.
- All cars advance at the same speed.
- Pedestrians, crossing roads, bridges, accidents, road closures are not considered for this simulator
- All roads must be horizontal or vertical and there are no diagonal or winding roads.

Design Assumptions

- We might assume that we can reuse an existing software package that provides relevant mathematical functionality such as statistical distribution, random number generators, and queuing theory.
- We are assuming that all vehicles slow down at a yellow light relative to the position of the car from the traffic light
- All the cars accelerate at the same time, but at different acceleration rates.
- Assume all the cars are the same size.

Design Approach #1

Design Description

- The cars do not turn
- Each position has object type:
- Vehicle contains position
- Position contains
 - x,y coordinates
 - Lanes, Intersection, Empty

Additional Design Assumptions

- All cars will be moving in the same direction-- there are no left, right, or u-turns.
- Cars will have the same acceleration
- Using a grid system map to store objects.

Design Approach #2

Design Description

- 3D interactive experience incorporating all aspects
- Students will be able to zoom in and out of the GUI, allowing them to see the impacts of traffic at both a micro and macro level
- A boolean set to true will allow the user to see the simulation from a bird's eye view, with the option to zoom in and out, and the same boolean set to false will allow for a street level view wherever the user chooses
 - Zooming into the simulation repeatedly will eventually change views and the state of the boolean
- All cars will be moving in the same direction the entirety of their time in the simulation
- A maximum of 6 intersections arranged in patterns of the user's choosing will be allowed

Additional Design Assumptions

- Cars move in the same direction without making U-turns
- We can track car position using x, y, and z coordinates
- Cars will be following US traffic laws

Design Approach #3 – Main Approach

Why this approach?

We choose the approach as compared to the other approaches, it is the one which offers the most relatability to the real world, as well as maintains the requested level of simplicity. It is also a known solution which seems to have worked in the past which makes it easier to generate for the design team as well.

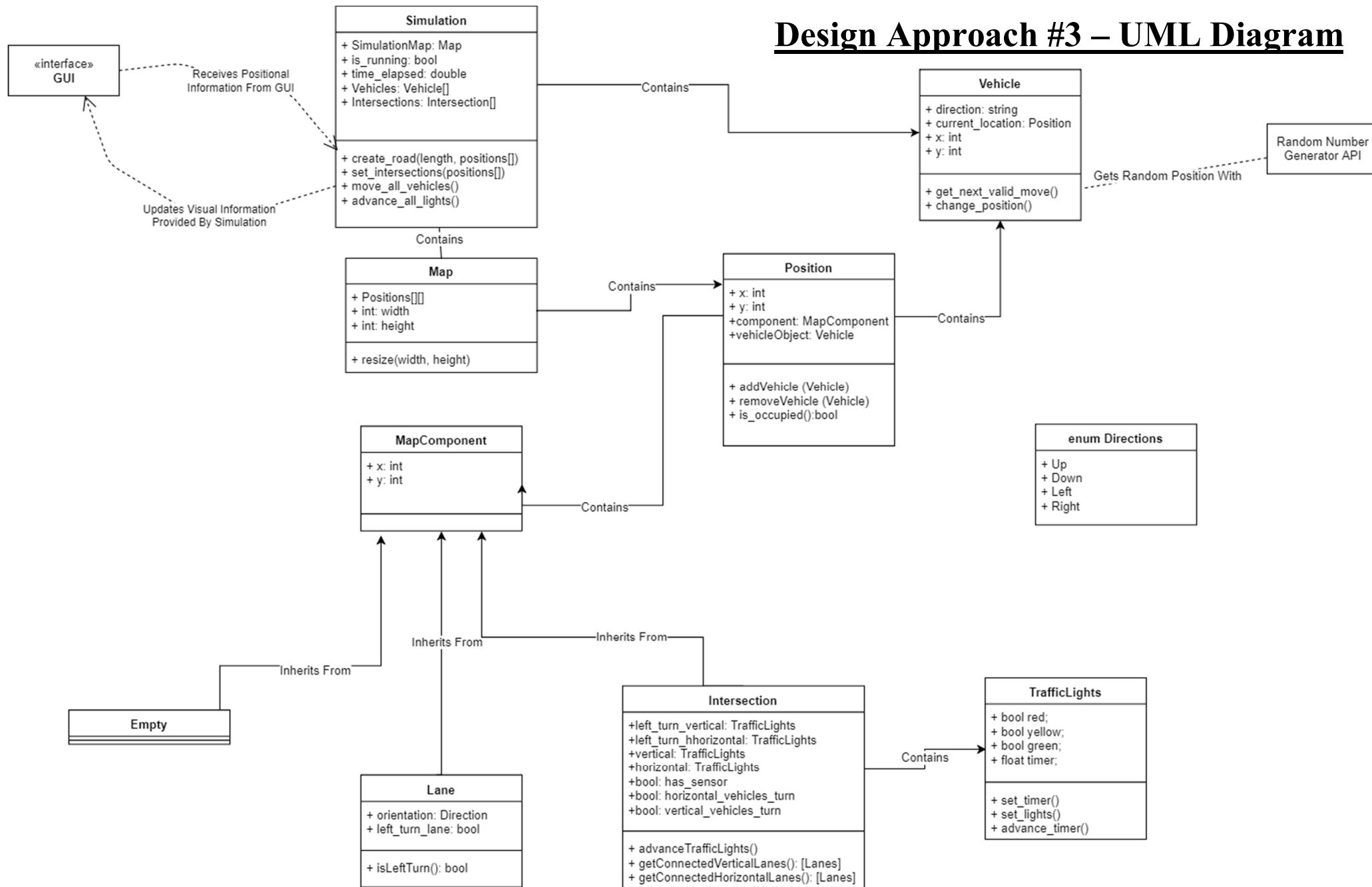
Design Description

- The cars can make left, right, and u-turns
- Map Components contain either a lane, an intersection, or an empty space
- Positions can contain vehicles and Map Components or just Map Components
- The system functions by constantly moving vehicles through positions in a 2D array

Additional Design Assumptions

- Assume all cars will be following US traffic laws, which means we are assuming the car can turn to the right on a red light.
- Using a grid system map to store objects.
Assume all roads will have two lanes

Design Approach #3 – UML Diagram



Design Approach #3 – Pseudo Code/Algorithm







Contains the pseudo code for our algorithm or running the simulation for important methods

MoveAllCars Method

For all vehicles










```
{  
    If (next position is not an intersection and next position is empty)  
    {  
        Move vehicle  
    }  
    Else  
    {
```

If (next position is intersection(I) & the current position include light=green & the current position is at the end of the lane (L))

	A	B	C	D	E	F
1	EMPTY	L	L	L	L	EMPTY
2	L	I	I	I	I	L
3	L	I	I	I	I	L
4	L 	I 	I 	I	I	L
5	L 	I 	I 	I	I	L
6	EMPTY	L	L	L	L	EMPTY

```
{  
    If ( next position is unoccupied)  
    {  
        then move the car forward in the next position to enter the  
        intersection  
    }  
    Else  
    {  
        The car remains in the same position until next position is  
        unoccupied  
    }  
}
```

```
}  
If (current position is at intersection)
```

	A	B	C	D	E	F
1	EMPTY	L	L	L	L	EMPTY
2	L	I	I	I	I	L
3	L	I 	I 	I	I	L
4	L 	I 	I 	I	I	L
5	L 	I 	I 	I	I	L
6	EMPTY	L 	L	L	L	EMPTY

```
{  
    If ( move forward)  
    {  
        If ( next position is unoccupied)
```

```

        {
            move one position forward until end of intersection
        }
    Else
        Stay at the current position
    }
    If (turn to the right)
    {
        If (the next position is unoccupied)
        {
            Move to the right and exit the intersection
        }
        else
            Stay at the current position
    }
    If (turn to the left)
    {
        If (the next position is unoccupied)
        {
            MoveUp diagonally two position
            And one position up until exit the intersection
        }
        else
            Stay at the current position
    }
    If ( make a u-turn)
    {
        If (the next position is unoccupied)
        {
            moveUp two position
            And one position left until exit the intersection
        }
        else
            Stay at the current position
    }
}

```

	A	B	C	D	E	F
1	EMPTY	L	L	L	L	EMPTY
2	L	I	I	I	I	L
3	L	I	I	I	I	L
4	L	I	I	I	I	L
5	L	I	I	I	I	L
6	EMPTY	L	L	L	L	EMPTY
7		L				

}


```
} end
```

AdvanceTrafficLights Method

For (All Intersections)

```
{
    If (isVerticalLanesTurn)
    {
        if(Vertical Left Lane Turn Traffic Light Timer > 0)
        {
            Decrease time for left lane turn traffic light timer one tick
        }
        Else if(Vertical Left Lane Turn Traffic Light Timer == 0)
        {
            if(has sensor)
            {
                Bool decrease timer = false
                For (all connected horizontal lanes)
                {
                    If (lane is occupied)
                    {
                        Decrease timer = true
                    }
                }
                if(decrease timer == true)
                {
                    Decrease time for vertical traffic light
                }
            }
            Else
            {
                pass
                //Dont decrease time for vertical light since no one is there
            }
        }
        Else
        {
            Decrease time for vertical light
        }
        If(the vertical light time == 0)
        {
            VerticalLaneTurn = false;
            HorizontalLaneTurn = true;
            Reset Horizontal Left Turn Traffic Light timer
            Reset Horizontal Traffic Light timer
        }
    }
}
```

```

    }
Else
{
    if(Horizontal Left Lane Turn Traffic Light Timer > 0)
    {
        Decrease time for left lane turn traffic light timer one tick
    }
Else if(Horizontal Left Lane Turn Traffic Light Timer == 0)
{
    if(has sensor)
    {
        Bool decrease timer = false
        For (all connected vertical lanes)
        {
            If (lane is occupied)
            {
                Decrease timer = true
            }
        }
        if(decrease timer == true)
        {
            Decrease time for horizontal traffic light
        }
    }
Else
    {
        Pass
        //Don't decrease time for horizontal light since no one is there
    }
}
Else
{
    Decrease time for horizontal light
}
If(the horizontal light time == 0)
{
    HorizontalLaneTurn = false;
    VeritcallLaneTurn = true;
    Reset Vertical Left Turn Traffic Light timer
    Reset Vertical Light timer
}
}
}
}

```

SimulationMain Method:
While(Simulation is Running)

```
{  
    moveAllCars()  
    AdvanceTrafficLights()  
    time_elapsed++  
}
```