Chapter 6

**System Development of the Simulation**

**6.1 Introduction**

The simulation takes shape during the development process. In this chapter we will take a look into the methodology and the approach to building the simulation.

**6.2 Platform Overview**

The main intention was to build the system as light and simple possible. Technologies that are used as a base upon which “The Traffic Symphony” the simulation is developed are given bellow –

**6.2.1 WebGL**

WebGL (Web Graphics Library) is a [JavaScript](https://en.wikipedia.org/wiki/JavaScript) [API](https://en.wikipedia.org/wiki/Application_programming_interface) for rendering interactive 2D and 3D graphics within any compatible [web browser](https://en.wikipedia.org/wiki/Web_browser) without the use of [plug-ins](https://en.wikipedia.org/wiki/Plug-in_(computing)). WebGL is fully integrated with other [web standards](https://en.wikipedia.org/wiki/Web_API), allowing GPU-accelerated usage of physics and image processing and effects as part of the web page canvas. WebGL elements can be mixed with other HTML elements and composited with other parts of the page or page background. WebGL programs consist of control code written in JavaScript and [shader](https://en.wikipedia.org/wiki/Shader" \o "Shader) code that is written in [OpenGL ES Shading Language](https://en.wikipedia.org/wiki/OpenGL_Shading_Language) (ESSL), a language similar to [C](https://en.wikipedia.org/wiki/C_(programming_language)) or [C++](https://en.wikipedia.org/wiki/C%2B%2B), and is executed on a computer's [graphics processing unit](https://en.wikipedia.org/wiki/Graphics_processing_unit) (GPU).

**6.2.2 three.js**

Three.js allows the creation of [Graphical Processing Unit](https://en.wikipedia.org/wiki/Graphical_Processing_Unit) (GPU)-accelerated 3D animations using the [JavaScript](https://en.wikipedia.org/wiki/JavaScript) language as part of a [website](https://en.wikipedia.org/wiki/Website) without relying on proprietary [browser plugins](https://en.wikipedia.org/wiki/Browser_plugin). This is possible due to the advent of [WebGL](https://en.wikipedia.org/wiki/WebGL" \o "WebGL).

High-level libraries such as Three.js or [GLGE](https://en.wikipedia.org/wiki/GLGE_(programming_library)), SceneJS, PhiloGL or a number of other libraries make it possible to author complex 3D computer animations that display in the browser without the effort required for a traditional standalone application or a plugin.

**6.2.3 P5.js**

p5.js is a JavaScript library that starts with the original goal of [Processing](http://processing.org/)—to make coding accessible for artists, designers, educators, and beginners—and reinterprets this for today's web.

Using the original metaphor of a software sketchbook, p5.js has a full set of drawing functionality. However, you're not limited to your drawing canvas, you can think of your whole browser page as your sketch! For this, p5.js has [addon libraries](http://p5js.org/libraries/) that make it easy to interact with other HTML5 objects, including text, input, video, webcam, and sound.

p5.js is a new interpretation, not an emulation or port, and it is in active development. You can also read how this [differs from processing.js](https://github.com/processing/p5.js/wiki/Frequently-Asked-Questions#how-is-this-different-than-processingjs), or about how this [relates to Processing](https://github.com/processing/p5.js/wiki/Processing-transition).

**6.2.4 python**

Threading in python is used to run multiple threads (tasks, function calls) at the same time. Note that this does not mean that they are executed on different CPUs. Python threads will NOT make your program faster if it already uses 100 % CPU time. In that case, you probably want to look into parallel programming. If you are interested in parallel programming with python, please see [here](https://wiki.python.org/moin/ParallelProcessing).

Python threads are used in cases where the execution of a task involves some waiting. One example would be interaction with a service hosted on another computer, such as a webserver. Threading allows python to execute other code while waiting; this is easily simulated with the sleep function.

**6.2.5 ML5.js**

ml5.js is an open source, friendly high level interface to TensorFlow.js, a library for handling GPU-accelerated mathematical operations and memory management for machine learning algorithms. ml5.js provides immediate access in the browser to pre-trained models for detecting human poses, generating text, styling an image with another, composing music, pitch detection, and common English language word relationships, and much more.

**6.2.6 Blender Model (Low Poly City)**

I have used a Low poly city model to demonstrate real life traffic situation. The model is built on blender. Made it interactive and the model is called by the webGL functions. The cars are running down the road and generating signals to the database to process on.

**6.3 Building and loading the Model**

Rather Than using objLoader, I have used gltfloader. Because that’s how I can get rid of heavy blender obj model.

**GLTF loader**

* {
* const gltfLoader = new GLTFLoader();
* const url = 'resources/models/cartoon\_lowpoly\_small\_city\_free\_pack/scene.gltf';
* gltfLoader.load(url, (gltf) => {
* const root = gltf.scene;
* scene.add(root);
* ...
* });

Then the 3d model preview looked like this



But still this is just a non interactive model. I wrote some code to dump put the scenegraph to the [JavaScript console](https://threejsfundamentals.org/threejs/lessons/threejs-debugging-javascript.html).

Here's the code to print out the scenegraph.

* function dumpObject(obj, lines = [], isLast = true, prefix = '') {
* const localPrefix = isLast ? '└─' : '├─';
* lines.push(`${prefix}${prefix ? localPrefix : ''}${obj.name || '\*no-name\*'} [${obj.type}]`);
* const newPrefix = prefix + (isLast ? ' ' : '│ ');
* const lastNdx = obj.children.length - 1;
* obj.children.forEach((child, ndx) => {
* const isLast = ndx === lastNdx;
* dumpObject(child, lines, isLast, newPrefix);
* });
* return lines;
* }

[Running that](https://threejsfundamentals.org/threejs/threejs-load-gltf-dump-scenegraph.html) I got this listing

* OSG\_Scene [Scene]
* └─RootNode\_(gltf\_orientation\_matrix) [Object3D]
* └─RootNode\_(model\_correction\_matrix) [Object3D]
* └─4d4100bcb1c640e69699a87140df79d7fbx [Object3D]
* └─RootNode [Object3D]
* │ ...
* ├─Cars [Object3D]
* │ ├─CAR\_03\_1 [Object3D]
* │ │ └─CAR\_03\_1\_World\_ap\_0 [Mesh]
* │ ├─CAR\_03 [Object3D]
* │ │ └─CAR\_03\_World\_ap\_0 [Mesh]
* │ ├─Car\_04 [Object3D]
* │ │ └─Car\_04\_World\_ap\_0 [Mesh]
* │ ├─CAR\_03\_2 [Object3D]
* │ │ └─CAR\_03\_2\_World\_ap\_0 [Mesh]
* │ ├─Car\_04\_1 [Object3D]
* │ │ └─Car\_04\_1\_World\_ap\_0 [Mesh]
* │ ├─Car\_04\_2 [Object3D]
* │ │ └─Car\_04\_2\_World\_ap\_0 [Mesh]
* │ ├─Car\_04\_3 [Object3D]
* │ │ └─Car\_04\_3\_World\_ap\_0 [Mesh]

So as a simple test I thought I would just try rotating all the children of the "Cars" node around their Y axis.

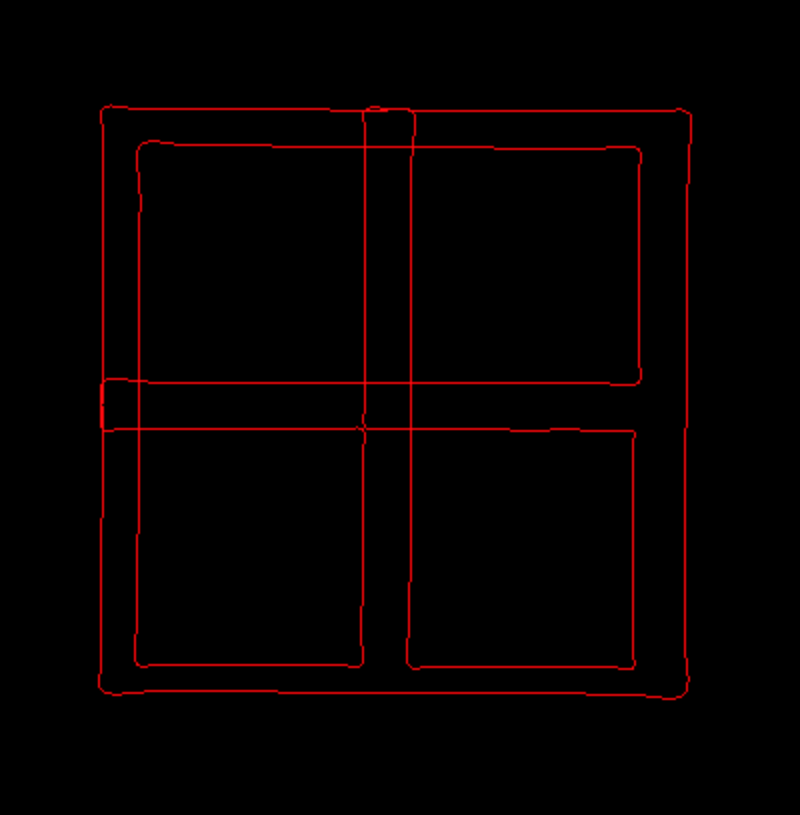
I looked up the "Cars" node after loading the scene and saved the result.

And my model car started doing crazy spins.

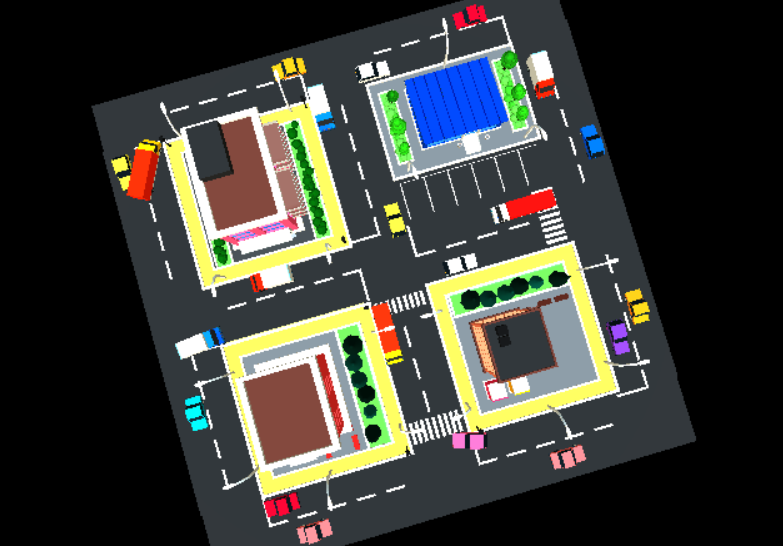


THREE.js has some curve classes. The [CatmullRomCurve3](https://threejs.org/docs/#api/en/extras/curves/CatmullRomCurve3) seemed like it might work. The thing about that kind of curve is it tries to make a smooth curve going through the points.

After adjusting the code, this gave me curve like this



After adjusting the path the car was moving correctly.



Short of.

Then I added some light and rendered some shadow to visualize better.

* {
* const color = 0xFFFFFF;
* const intensity = 1;
* const light = new THREE.DirectionalLight(color, intensity);
* ……….
* scene.add(light);
* scene.add(light.target)
* ………….



So far so good. I found a 3d model which is running on a browser. The car was moving on given direction.

**6.4 The Hyperledger**

I loaded python to my project. I have used a custom API to load current position, status, and destination of the car processed it back to the database. There it got compared with the other car’s status and got verification. And updated the ledger. I am going to discuss it to the next “Testing” Chapter.

**6.5 Summary**

Till now this is just a 3d model where cars are interactive. To build the Hyperledger more operative we have to test it with data, which I am going to demonstrate in the next chapter.

Chapter 6

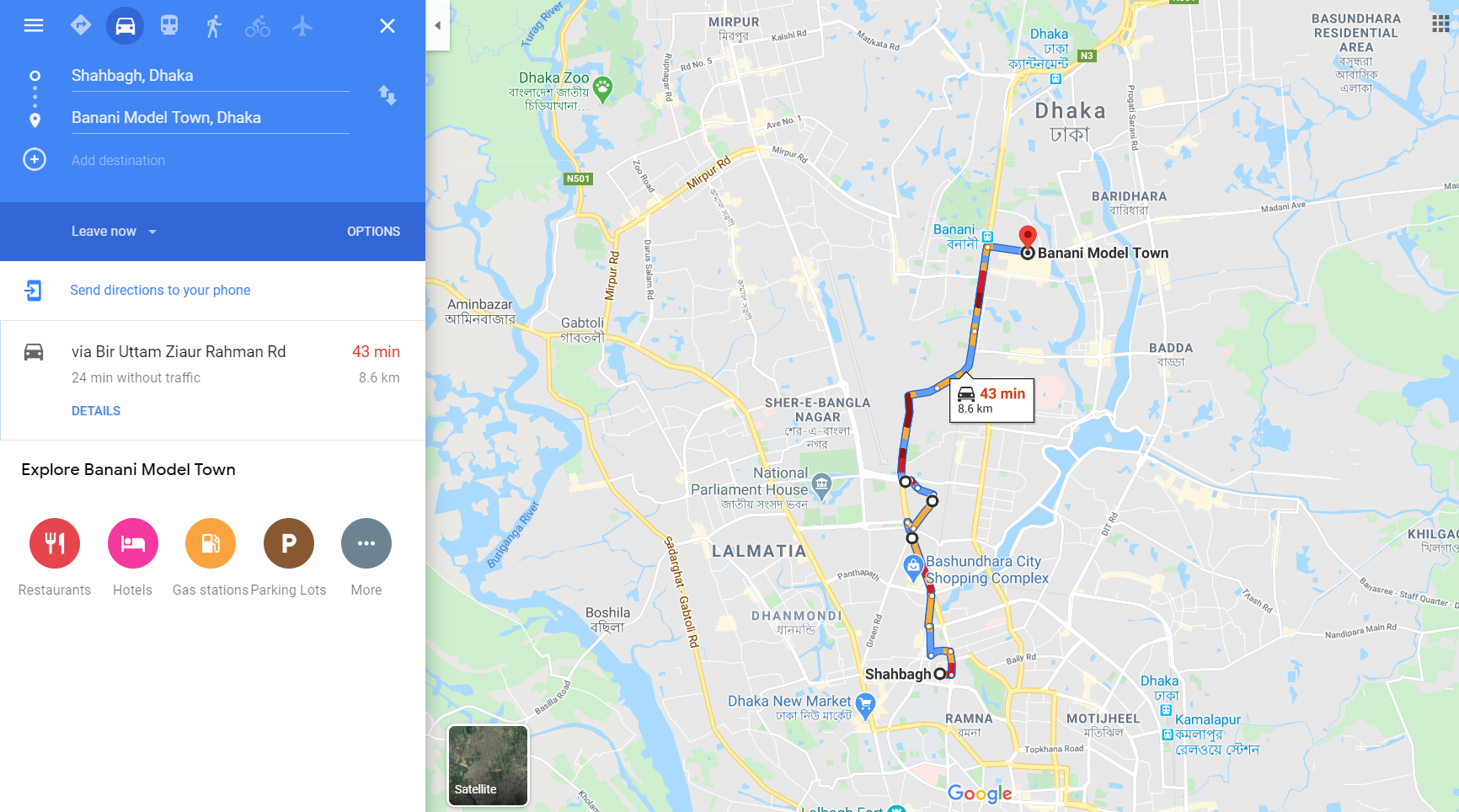
**System Testing**

**6.1 Introduction**

The testing process goes on parallel to the real life data. With the help of a web crawler I gathered some data of road situation in Dhaka. In this chapter I have demonstrated how I tested and generated result from the simulation.

**6.2 Principle of Testing**

To test the simulation I gathered some Google traffic data which shows me the road condition of Dhaka. I have classified the road traffic density by the time given, which is the time to reach the destination. In the picture you can see that it takes 43 minutes to reach Banani from shahaBag at 9:41AM



Where most of the car will take the rout through Farmgate. Cause it is a common practice to drive straight, or the path which comes first. It makes that particular rout busy for everyone in the rush hour. The vehicles from Asad Gate, Manik Miah Avenue, the vehicles towards shahaBag gets stuck at either Farmgate or Bijoy Sharani. But in the reality if the vehicles towers Banani took the Sat-Rasta rout could go faster.

I have generated a same situation like this. Putting a particular rout busy and as a result the vehicles towards a particular destination avoided the rout automatically. Same happed when a fire emergency created.

Chapter 7

**Conclusion and Future Work**

**7.1 Conclusion**

The new infrastructure opens a new window for the future. The real question is “Are we ready”? To implement the infrastructure, A lot of changes in constitutional and business code is needed. A mutual agreement of government and car manufacturers is needed. New Laws should be applied. The whole system should be monitored by a good hand either this could be devastating in the wrong hand.

**7.2 Future Work**

Again the real question is “Are we ready”? The system can be more useful and have more feature like biometric entry, auto transaction, dedicated time subscription where you don’t have to call a cab to go to office, where the car will be at your door when you are about to leave.