**PROJECT REPORT**

**ON**

**Traffic management system**

****

REPORT SUBMITTED

TO

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

FOR THE PBL OF **PYTHON FOR ENGINEERS**

IN

**ENGINEERING AND APPLIED SCIENCE DEPARTMENT**

By

**Mihir Patwardhan- 268/22111346**

**Aditya Adinath Patil - 269/ 22110016**

**Krishna Gupta- 270/22110152**

**Triambak Kapoor – 271/22110153**

**Class: F.Y BTECH Division: B Batch: B3**

**Batch teacher:- Aparna Barbadekar**

**Index**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Contents** | **Page no.** |
| **1** | **Abstract** | **03** |
| **2** | **Introduction and theory** | **04-09** |
| **3** | **Flow of program** | **10** |
| **4** | **List of the topics/ concept which are covered from the syllabus** | **11** |
| **5** | **Conclusion** | **12** |
| **6** | **References** | **13** |

**Abstract**

We are building a 4-way traffic stimulation in which we have used Pygame to run the stimulation. The stimulation is based on an 4-way traffic intersection with traffic signals controlling the flow of traffic in each direction. We have set a timer for each signal as the time ends the light of the signal gets switched from the respective red, green, yellow, vehicles which we have used in our stimulation are cars, bikes, buses, and trucks their movement is controlled according to the signals and vehicles around them. By using this stimulation, we can further send it to data analytics to analyze the applications of it. We can also draw a table where we can see how much time can be saved by using this stimulation and why it is very useful in our practical life. We have used PyCharm to perform our coding applications and to code it.

**Introduction and theory**

1. Before diving up into the code we first found out the images of vehicles like as we have told we have included bike, car, truck as prime vehicles so we took their pics from all angles so that the camera posted on the stimulation site can analyze it well. After that we imported traffic signals in which we know there are three which are red, yellow, green. After that we found out a four-way intersection image by which we would place our vehicles as well as traffic signals. After that we created a folder in which we included the all images with file names as {*down, up, left, right*} where *down* contains the images of vehicles facing down, *up* contains the images of vehicles facing up, *left* contains the images of vehicles facing left, *right* contains the images of vehicles facing right, signals contain the images of traffic signals.
2. Installing the most important library used for our stimulation which is Pygame we downloaded Pygame extension from PyCharm itself.
3. **Importing the libraries-:** while we know that we created a called as traffic stimulation.py and in which we imported total 5 libraries.
4. **Random**-: We imported random library because in case of when the stimulation runs the vehicles are generated randomly.to be more precise to set the. Vehicle movement random we imported random library.
5. **Time**-: We imported time library because as we know we have signals in our program and to set the timing of the signal we imported the time library so that whenever a vehicle traverses through the signal the signals should have definite time to switch colors. the time library which we have imported depends upon the no of vehicles (vehicle density.)
6. **Threading-:** The library threading is used in our program to perform multiple tasks as we know that in our program many conditionings and looping in the vehicles is going on simultaneously, so we need to thread each task to make it perform efficiently.
7. **Pygame-:** Pygame is a very important aspect of our stimulation because our stimulation is running on Pygame only. We installed Pygame with the PyCharm itself.
8. **Sys-:** We all know about how important sys module is because it provides us information about the constants, functions and methods of the python interpreter.
9. Defining the constants where we defined the constants that will be used in the movement of vehicles in the stimulation as well as in control of traffic signal timers and we set their default values of timers. We also defined about the speed of the vehicles to be very precise we have declared the average speed of the vehicles which we are having.
10. Next, we defined about the co-ordinates of vehicles with the co-ordinates of signal image, timer, vehicle count and the co-ordinates of stop lines. We also gave that what should be the stopping gap between the vehicles and moving gap between them.
11. The next and most important step of our traffic stimulation was initializing Pygame by pygame.init() method.
12. Next, we defined some classes whose objects will be generated in the stimulation
13. **traffic** **class**-: the first class we build was traffic class. Now as our stimulation is a 4-way stimulation we need 4-traffic signals in our stimulation. So, we build a traffic signal class which has the following attributes-:
14. **red**-: value of red signal timer.
15. **yellow**-: value of yellow signal timer.
16. **green**-: value of green signal timer.
17. **signaltext**-:value of timer to display.
18. ***Vehicle class***-: this is a class that represents objects of vehicles that we will be generating in the stimulation. The vehicle class we included has the following attributes and methods-:

* **vehicleclass**: represents the class of the vehicle such as car, bus, truck, or bike
* **Speed**: represents the speed of the vehicle according to its class
* **Direction\_number**: represents the direction — 0 for right, 1 for down, 2 for left, and 3 for up
* **Direction**: represents the direction in text format
* **X**: represents the current x-coordinate of the vehicle
* **Y**: represents the current y-coordinate of the vehicle
* **Crossed**: represents whether the vehicle has crossed the signal or not
* **Index**: represents the relative position of the vehicle among the vehicles moving in the same direction and the same lane
* **Image**: represents the image to be rendered
* **Render** **( )**: to display the image on screen
* **Move ( )**: to control the movement of the vehicle according to the traffic light and the vehicles ahead

1. Now let’s talk about the most important function included in our program that is **move ( )** function which is one of the most important pieces of code in our stimulation. For each direction we checked that the vehicle has crossed the intersection or not. This is important because if the vehicle has already crossed the lane, it can keep moving regardless of the signal being green or red .so when the vehicle crossed the intersection, we set the value of crossed =1. Next, we decide when the vehicle moves and when it stops
   * if it has not reached its stop point before the intersection.
   * if it has already crossed the intersection.
   * if the traffic signal controlling the direction in which the vehicle is moving is green.

Only in these 3 cases, the co-ordinate of the vehicle is updated by incrementing/decrementing them by the speed of the vehicle, depending on their direction of motion. However, we need to consider one more possibility that there is a vehicle ahead moving in the same direction and lane. In this case, the vehicle can move only if there is a sufficient gap to the vehicle ahead, and this is decided by taking into consideration the coordinate and the width/height of the vehicle ahead of it, as well as the *moving gap*.

1. **Initializing the signals-:** Next, we initialize 4 traffic signal objects, from top left to bottom left in a clockwise direction with default values original signal timer. the red signal timer of ts2 is set equal to the sum of yellow and green signal timer of ts1.
2. **Repeat ( ) -:**
3. The function repeat () that is called at the end of the initialize () function above is a recursive function that runs our entire simulation. This is the driving force of our simulation.
4. The repeat () function first calls the updatevalues() function every second to update the signal timers until the *green* timer of the *currentgreen* signal reaches 0. It then sets that signal to yellow and resets the *stop* value of all vehicles moving in the direction controlled by the *currentgreen* signal. It then calls the updatevalues() function again after every second until the *yellow* timer of the *currentgreen* signal reaches 0. The *currentyellow* value is now set to 0 as this signal will turn red now. Lastly, the values of the *currentgreen* signal are restored to the default values, the value of *currentgreen* and *nextgreen* is updated to point to the next signals in the cycle, and the value of *nextgreen* signal’s *red* timer is updated according to *yellow* and *green* of the updated *currentgreen* signal. The repeat () function then calls itself, and the process is repeated with the updated *currentgreen* signal.
5. **updatevalues ()-:** the function updatevalues() updates the timers of all the signals after every second .
6. **generatevehicles ()-:** the generatevehicles () function is used to generate the vehicles. The type of vehicle (car, bus, truck, or bike), the lane number (1 or 2) as well as the direction the vehicle moves towards is decided by using random numbers. The variable *dist*represents the cumulative distribution of vehicles in percentage. So, a distribution of [25,50,75,100] means that there is an equal distribution (25% each) of vehicles across all 4 directions. Some other distributions can be [20,50,70,100], [10,20,30,100], and so on. A new vehicle is added to the simulation after every 1 second.
7. **main class ()-:** let us understand the main () function by breaking it down into smaller pieces. We start by creating a separate thread for initialize () method, which instantiates the 4 trafficsignal objects. Then we define 2 colors, white and black, that we will be using in our display. Next, we define the screen width and screen size, as well as the background and caption to be displayed in the simulation window. We then load the images of the 3 signals, i.e., Red, yellow, and green. Now we create another thread for generatevehicles (). Next, we run an infinite loop that updates our simulation screen continuously. Within the loop, we first define the exit criteria. In the next section, we render the appropriate signal image and signal timer for each of the 4 traffic signals. Finally, we render the vehicles on the screen and call the function move () on each vehicle. This function causes the vehicles to move in the next update *the***blit()** function is used to render the objects on the screen. It takes 2 arguments, the image to render and the coordinates. The coordinates point to the top-left of the image.

**Flow of the program**

Text, letter

Description automatically generated

**List of topics which are covered from the syllabus**

* Dictionaries, lists, tuples.
* Functions in python
* If-elif ladder
* Classes and objects
* Constructors in python
* While loop
* For loop
* Nested if-else

**Conclusion**

According to stimulation results, the system shows about 23% improvement over the current system in terms of no of vehicles crossing the intersection which is a significant improvement. This system can be integrated with the cctv cameras in major cities in order to facilitate better management of traffic. This will lower the unwanted delays and reduce congestion and waiting time which in turn reduces the fuel consumption and pollution. We learnt a lot from this PBL. Especially the stimulation part, the Pygame module which we imported were very interesting to showcase .we also proposed a program which is not just a program it’s an implementation in real life where we told with the help of coding skills how to manipulate the traffic and control it and how it saves a lot amount of time.by virtue of this presentation we also enhanced our presentation as well as speaking skills .it also taught us how to use leadership as a principal and implement it .this stimulation project was not just a project it was a spark of understanding and application of intelligence in a suitable manner in form of coding with python.