Urban Planning using SNA Approach for Improved Road Transportation

Patel Het
het.p5@ahduni.edu.in
Ahmedabad University (SEAS)
Ahmedabad, Gujarat, India

Hingu Dhruv
dhruv.h@ahduni.edu.in
Ahmedabad University (SEAS)
Ahmedabad, Gujarat, India

Kayastha Vatsal
vatsal.k2@ahduni.edu.in
Ahmedabad University (SEAS)
Ahmedabad, Gujarat, India

Abstract

In our exploration of traffic congestion, we set out to understand how traffic behaves in urban areas during peak times, aiming to find solutions to ease congestion and improve city planning. To achieve this, we embarked on a journey of data collection and analysis, leveraging tools like Open Street Map and SUMO simulation software. By simulating realistic traffic scenarios and converting the data into a manageable CSV format, we obtained a comprehensive dataset ready for analysis. With this dataset, we are equipped to delve into traffic dynamics, identify congestion hotspots, and propose strategies for enhancing traffic flow. Our findings underscore the importance of using simulation tools and creative problem-solving to address complex challenges in urban planning and transportation management. Ultimately, our goal is to pave the way for more efficient and sustainable cities, where residents can navigate the streets with ease and comfort, free from the frustrations of traffic congestion.

Keywords: Traffic congestion, Urban planning, Social network analysis, Road transportation, SUMO simulation, OpenStreetMap, Dataset collection, Data preprocessing, Traffic dynamics, Congestion patterns, Infrastructure improvements, Simulation tools, Network analysis, Traffic management, Transportation systems

ACM Reference Format:
Patel Het, Hingu Dhruv, and Kayastha Vatsal. 2024. Urban Planning using SNA Approach for Improved Road Transportation. In *Proceedings of on April 13, 2024 (SNA Final Report).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Imagine you're stuck in a traffic jam during rush hour, and you're wondering if there's a better way to plan our cities to avoid these frustrating situations. That's the problem we're trying to solve: understanding how traffic flows across a big area at a specific time, usually when it's busiest. By using some smart analysis techniques, we can learn a lot about how traffic moves and use that information to make our cities work better for everyone.

Think of it like this: we have this giant map of roads and intersections in the form of nodes and edges, and we want to figure out where all the cars are and how they're moving around. We pick a time, like the busiest part of the day, and then we start digging into the data to see what's going on. We look at each road and intersection and see how much traffic is passing through. This helps us spot places where there's a lot of congestion, like those annoying spots where you always seem to get stuck.

Once we've gathered all this data, we can start looking for patterns and trends. For example, maybe we notice that one particular intersection is always super busy, causing long delays for everyone. By studying the data closely, we might come up with ideas for how to improve things. Maybe we could build a new road or a bypass to help cars avoid that congested area, saving people time, fuel, and eventually reducing pollution in the process. The key is to use all this information to make smarter decisions about how we plan and develop our cities. By understanding where the traffic hotspots are and why they happen, we can come up with creative solutions to make life easier for everyone who lives and travels in our cities. It's all about using data and analysis to build better, more efficient communities that work for everyone.

2 Related Works

To understand the current research on our problem, we found that studies are focusing on using centrality measures like Betweenness centrality, 2-step reach, and Eigenvector centrality to identify key points in road networks where traffic congestion occurs.[4] These studies highlight congested intersections but don't offer solutions on how to alleviate this congestion. This area of providing actionable solutions to reduce traffic congestion still needs more research. So, we are focusing our efforts on this gap to develop practical solutions.

SNA Final Report, April 13 2024, Ahmedabad University, Ahmedabad

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

3 Dataset Collection

3.1 Initial Finding of the DataSet

We wanted to study and improve traffic flow on busy roads by analyzing the road networks of a specific area. First, we looked for data on websites like Kaggle and various research papers to find real-world road network information where intersections are represented as nodes and roads as edges between these nodes. Ideally, this data would also include traffic details, like the number of vehicles on each road, which is crucial for understanding congestion.

Initially, we checked SNAP, a site known for its network analysis resources, but the datasets available only had basic information about nodes and edges and lacked detailed traffic data. Realizing this wasn't enough, we considered assigning random traffic values to the roads ourselves. However, we figured out this wouldn't accurately reflect actual traffic patterns, especially those affected by roads with a lot of intersections, so we abandoned this idea.

We also found a dataset from North America that had information about the average speed of vehicles on different roads. However, this data was more about speed limits for each lane and didn't focus on identifying specific road sections that cause traffic jams. This detail is crucial for us, mainly because the traffic in places like India is very complex. So, this dataset wasn't suitable for our needs either.

We thought about making up traffic data by assigning random numbers to each road intersection, but we realized this method wouldn't accurately reflect the real traffic situations, especially on roads with many intersections. This meant it wouldn't work well for our analysis. Therefore, we decided it was important to create a dataset based on real roads to ensure the accuracy and relevance of our findings.

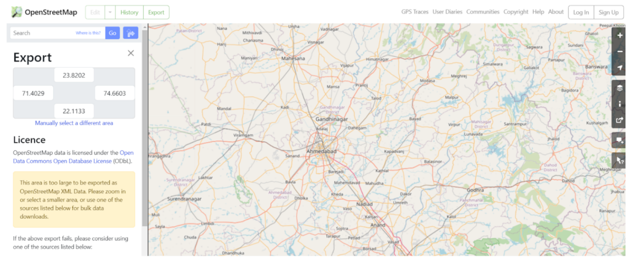
3.2 Creation of the Dataset

We then decided to create our Dataset for that we came across this tool known as SUMO (Simulation of Urban Mobility)[1] for which we first took a snapshot of a specific area on the Open Street Map [2] to create the dataset. By selecting the area and clicking on "export," we obtained a .osm file, including the area's road network information. Below is the view of the area from Open Street Map:

Here, we took a snapshot of an area in Ahmedabad with numerous intersections, known as Commerce Six Road, and ran our simulation on that road network.

3.3 Simulation Using SUMO

Next, use SUMO (Simulation of Urban Mobility), an open-source, highly portable traffic simulation package capable of handling large networks. It enables users to effectively



=

Figure 1. Open Street Map Web Page

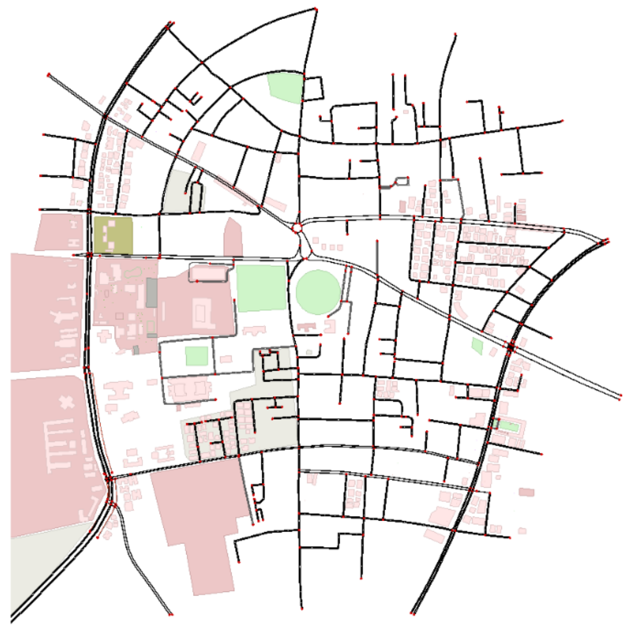


Figure 2. Commerce Six Road, Ahmedbad

model vehicular traffic behaviors and test traffic management strategies. After installing SUMO, use the netconvert option to convert the .osm file to an XML file that contains the road network information of the area you selected on Open Street Map. Then, employ the RandomTrip.py script within SUMO to generate random traffic patterns on this road network. Configure and run the simulation using sumo-config to visualize traffic flow on the network.[3]

Here the Yellow Arrows represents the vehicles on the road.

After setting up the traffic simulation with the necessary parameters, we ran it for about 30 seconds. This resulted in a network comprising '390 nodes' and '800 edges' around Commerce Six Road in Ahmedabad. We could visualize the simulation and gather an XML file containing detailed data such as vehicle IDs, timestamps, and edge IDs. This data

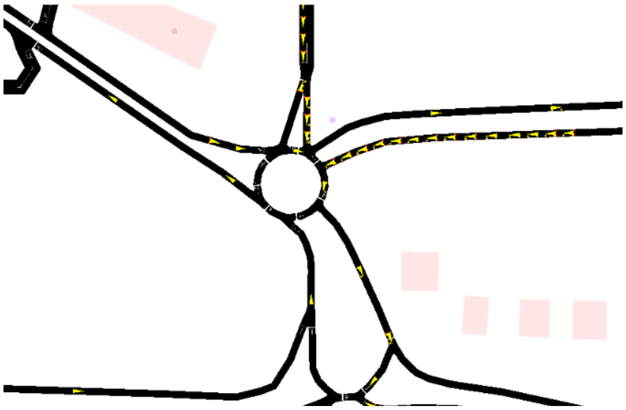


Figure 3. Simulation of Traffic

helps track the positions of vehicles over time.

3.4 Pre-processing on the Dataset

Since the data was in XML format and we needed it in a more accessible form for analysis, we converted it to CSV format. Given the dataset's size, with nearly 2500 data points, we used specific parsing techniques to manage and prepare the dataset for further analysis efficiently.

Here is an example of the dataset we obtained after running the simulation, now formatted as a CSV:

node_from	node_to	vehicle_count
9870050283	9870050277	83.0
9870050280	9870050283	9.0
9870050285	9870050280	-1.0
9870050281	9870050285	-1.0
9870050283	9870050282	13.0
9870050282	11723727602	7.0

Figure 4. Final Dataset

In conclusion, our dataset is now ready for detailed analysis. This will allow us to dive deep into traffic dynamics, understand congestion patterns, and make well-informed recommendations for infrastructure improvements. The comprehensive process of acquiring this dataset highlighted the critical role of simulation tools like SUMO and the necessity of creative problem-solving in overcoming data challenges within network analysis projects

Here are the plots and graphs to visualize the data more clearly.

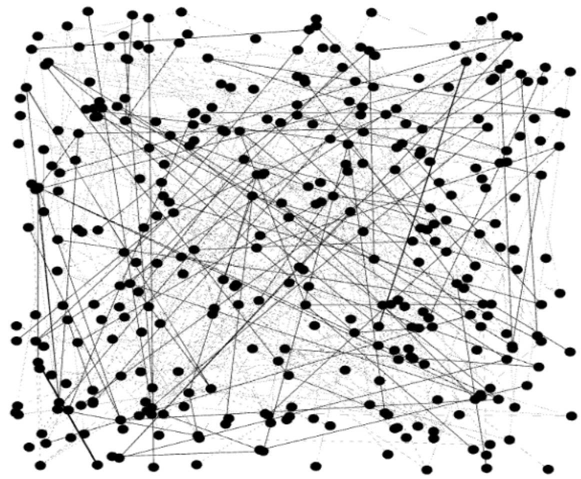


Figure 5. Graph Representation using Gephi

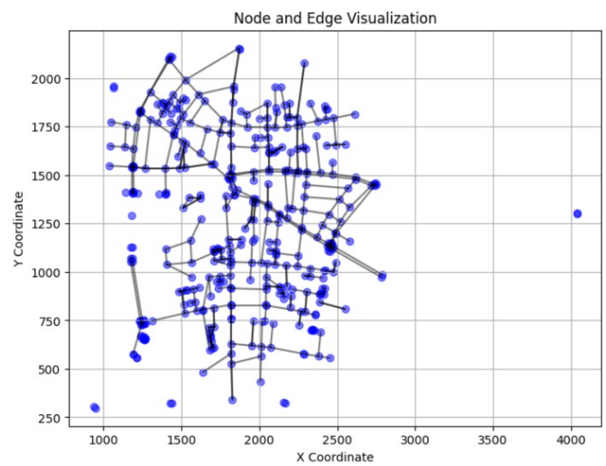


Figure 6. Data Plotting along with co-ordinates

4 Our Approach to the Problem

4.1 Pseudo Code and Explaination

1. **Load Dataset**:
   - Load traffic data from a CSV file.
2. **Get Timestamp**:
   - Prompt the user to input a timestamp to analyze traffic conditions.
3. **Calculate Congestion**:
   - For each edge in the traffic network, calculate the total congestion based on traffic data at the given timestamp.
4. **Plotting**:
   - Create two plots:
     - Plot 1: Nodes versus weights, showing the congestion levels.

  - Plot 2: Number of edges with the same congestion levels.
5. **Edge Selection**:
   - Determine the number of edges to consider using the formula:

     $$\text{num\_edges} = \min(\text{len}(\text{graph})//4, 10) + 1$$

   - Select the top X edges with the highest congestion levels.
6. **Find Congested Edge Pairs**:
   - Form pairs of all congested edges to analyze their relationships.
7. **Calculate Distance between Congested Edge Pairs**:
   - Calculate the distance between all possible pairs of congested edges and select the minimum distance between them.
8. **Find Adjacent Paths for Congested Edges**:
   - For each congested edge, find the first two paths with the minimum distance.
   - Calculate the distance and weights present in these alternate paths.
9. **Create Dictionaries**:
   - Create two dictionaries:
     - `iot_w`: Contains edge labels as keys and weights as values for multiple possible paths.
     - `iot_d`: Contains edge labels as keys and distances as values for multiple possible paths.
10. **Check for Area Congestion**:
    - Calculate the diameter of the whole graph.
    - Find the maximum distance between any two congested edges.
    - If the ratio of maximum distance to diameter is greater than 15%, opt for path development; otherwise, go for area development.
11. **Visualize Data**:
    - Plot `iot_w` and `iot_d` for better visualization of weights and distances.
12. **Identify Nodes with No Alternate Paths**:
    - Generate a list of nodes with no alternate paths available and with higher congestion levels.
13. **Check for Adjacency between Congested Edges**:
    - Check if any entry in the third column of the list, which gives the minimum distance between any two congested edges, is 0. If not, the edges are not adjacent to each other; otherwise, they are adjacent, indicating a specific case.
14. **Junction Development Suggestions**:
    - For junctions having two other nodes besides the central node, find the sum weight and suggest development based on this analysis.

5 Outputs and Results

These are the results for the time frame from 0 to 10.(Short Duration) This shows congestion on each edge here we have

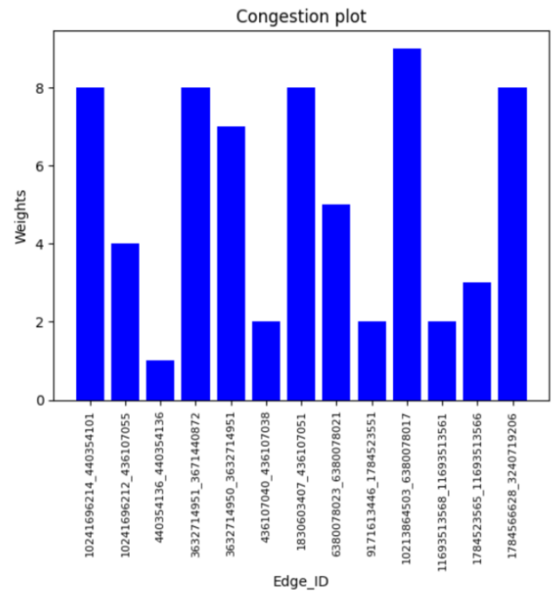


Figure 7. Congestion on Edges

considered edges for only small intervals of time so less edges. This shows a list of congested edges whose edge weight is

```
Edges with highest weights:
['10213864503', '6380078017', 9.0]
['10241696214', '440354101', 8.0]
['3632714951', '3671440872', 8.0]
['1830603407', '436107051', 8.0]
```

Start_Node	End_Node	Distance
6380078017	10241696214	17
6380078017	3632714951	12
6380078017	436107051	18
10241696214	3671440872	7
440354101	1830603407	6
3671440872	436107051	5

Figure 8. List of Congested Edges

greater than a certain threshold defined in the pseudo-code part. The Difference in edge weight between alternates path between two nodes to the original path. The Path Length Difference between alternates path between two nodes to the original path. This are the list of edges which has no alternate path available.

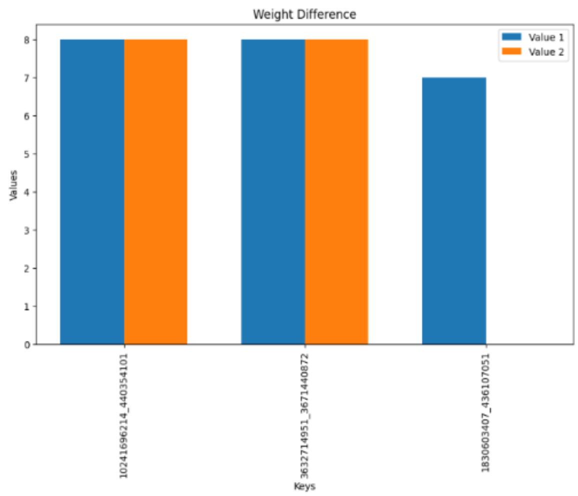


Figure 9. Weight Difference

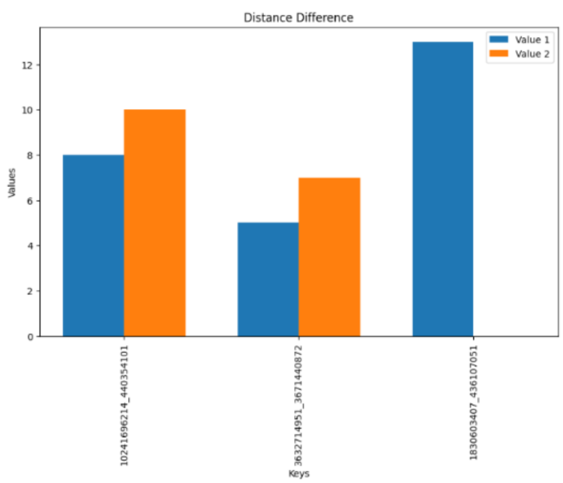


Figure 10. Distance Difference

Start_Node	End_Node	Weight
10213864503	6380078017	9

Figure 11. Edge with No Alternate Path

5.1 Results for Larger Time Frame

These are the results for the time frame from 0 to 999.(Large Duration)

This shows congestion on each edge here we have considered edges for large intervals of time so high edges. This shows a list of congested edges whose edge weight is greater than a certain threshold defined in the pseudo-code part. The

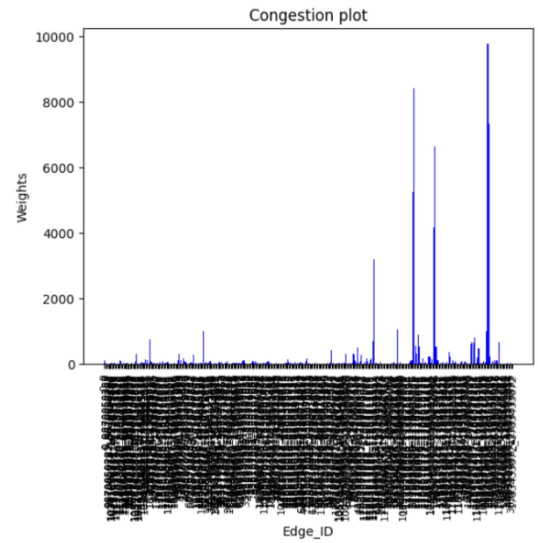


Figure 12. Congestion on Large # of Edges

```
Edges with highest weights:
['1784566628', '3240719206', 9774.0]
['1779085276', '1779085276', 8427.0]
['3240719206', '1784529288', 7350.0]
['3353748102', '1779260096', 6635.0]
['3776249830', '1779085276', 5243.0]
['436107022', '3353748102', 4175.0]
['11693513552', '11693513552', 3202.0]
['1827200290', '1788445421', 1066.0]
['3632714949', '1784566628', 1004.0]
['1830940444', '1830940444', 996.0]
['1779260094', '1779260096', 906.0]
```

Figure 13. Congested Edges for Large Data

Difference in edge weight between alternates path between two nodes to the original path.

The Path Length Difference between alternates path between two nodes to the original path. This shows junction congestion.

6 Future Possibilities

- Utilizing more efficient algorithms to perform the same operations with improved time complexities.
- Implementing dynamic algorithms capable of processing various datasets to provide tailored outputs.
- Developing dynamic clustering algorithms that can identify and recommend subsets of data based on proximity metrics, adapting to different scenarios.
- Analyzing weight distribution changes in graphs resulting from edge additions or removals to enhance model understanding.

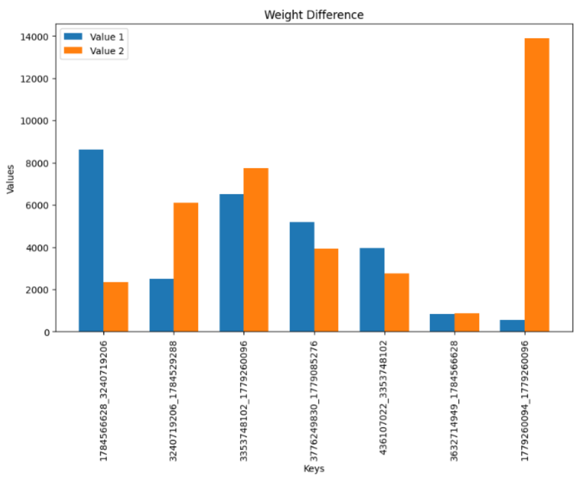


Figure 14. Weight Difference of all edges

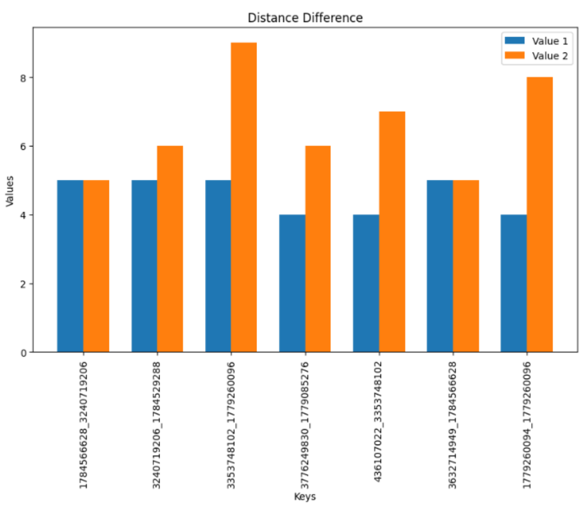


Figure 15. Distance Difference on large dataset

Start_Node	End_Node	Weight
1779085276	1779085276	8427
11693513552	11693513552	3202
1827200290	1788445421	1066
1830940444	1830940444	996

Figure 16. Edge with No Alternate Path

- Creating a simulator game where users invest money and make decisions based on traffic data, with the model providing efficiency reports and feedback.

Start_Node	End_Node	Weight
1784566628	1784529288	17124
3240719206	3632714949	10778
1779085276	3776249830	13670
1779260096	436107022	10810
3353748102	1779260094	7541

Figure 17. All Junction Congestion

7 Conclusion

In conclusion, our journey to tackle traffic congestion led us to create a detailed dataset capturing the dynamics of vehicular movement in a specific area. We started by exploring existing datasets but found them lacking in the necessary traffic details. To overcome this, we ventured into creating our dataset using Open Street Map and SUMO simulation software. This allowed us to simulate realistic traffic scenarios, generating valuable data for analysis. After running the simulation and converting the data into CSV format, we now have a comprehensive dataset ready for in-depth analysis.

With this dataset in hand, we are well-equipped to delve into traffic dynamics, identify congestion patterns, and propose effective solutions for improving traffic flow. By studying the movement of vehicles over time, we can gain insights into critical areas where traffic jams occur and understand the underlying factors contributing to congestion. Armed with this knowledge, we can make informed recommendations for infrastructure enhancements and traffic management strategies, ultimately striving to create more efficient and sustainable transportation systems for our cities.

Acknowledgments

We would like to extend our heartfelt gratitude to Professor Amit Navavati for his invaluable guidance and constant support throughout every phase of our project. His expertise, insightful feedback, and encouragement were instrumental in shaping our research and achieving our goals. We are also grateful to our dedicated classmates whose collaboration and constructive discussions enriched our learning experience and contributed to the success of our endeavor. Special thanks are due to Ph.D. scholar Kashish Shah for his assistance and expertise in navigating the intricacies of SUMO (Simulation of Urban Mobility) for road network simulation. His mentorship and technical insights significantly enhanced our understanding and proficiency in utilizing SUMO for our analysis. Our project benefited tremendously from the collective efforts and collaboration of these individuals, and we are deeply appreciative of their contributions.

References

[1] Eclipse SUMO - Simulation of Urban Mobility. https://eclipse.dev/sumo/. Accessed: Insert Date.

[2] Openstreetmap. https://www.openstreetmap.org/export#. Accessed: Insert Date.

[3] Adil Alsuhaim. SUMO - Creating Traffic Maps, Random Trips, and Exporting Trace for ns-3. YouTube video, September 22 2020. URL: https://www.youtube.com/watch?v=yl9yLch9pwU.

[4] Author(s) Name. Social network analysis approach for improved transportation planning. ResearchGate, 2016. DOI: Insert DOI or URL.