Impact of Flooding Inundation on Traffic Flow in Virginia Beach, VA

CSE 6730 Final Project

Yuchen Wang
ywang4119@gatech.edu
April 25, 2023
Georgia Institute of Technology

Abstract

Flooding is a frequent disaster in urban areas. It disrupts the urban transportation network, blocks portions of the roads, and results in longer travel times for residents and service vehicles. The primary approach to predict the travel time in prior studies is to simulate individual vehicle movement on disrupted road networks based on the fastest routing analysis. However, this approach implicitly assumes free-flow transportation and does not effectively model the movement of other vehicles as well as the macro-level traffic flows. To create a more effective travel model for residential vehicles, this study creates an improved traffic simulation in Python. Firstly, a simulation model is built to mimic the movement of vehicles from 8 AM to 6 PM in the whole city, which covers both the morning and evening peaks. In the morning peak, the drivers are more likely to depart from residential areas and arrive at business centers in the city, and vice versa. The movement of vehicles is influenced by other vehicles regulated by the relationship between density and speed. Secondly, the inundation depth in a hurricane that struck the city several years ago is used to determine the blocked roads. The traffic simulation is run in both normal and flooding time to determine the increase in average travel time in Virginia Beach, VA. This project is expected to quantitatively evaluate the impact of inundation to the transportation in the city and support taking measure to mitigate the impact.

1. Project description

This study aims to model the impact of flooding on the roadway system. During flooding, the inundation in the city will cut off some road segments and impact the transportation. This section describes the major components of the system being studied.

The City of Virginia Beach (VB), Virginia, is used as the study system. As shown in Figures 1 and 2, Virginia Beach is located on the east side of the state of Virginia, with the Atlantic Ocean to the east and the Chesapeake Bay to the north. The city's road network data was obtained from the open data portal of VB (https://gis.data.vbgoc.com/). The dataset consists of over 20,000 road segments. It was split at all the intersections, and a polygon of the road surface shape represents each road segment.

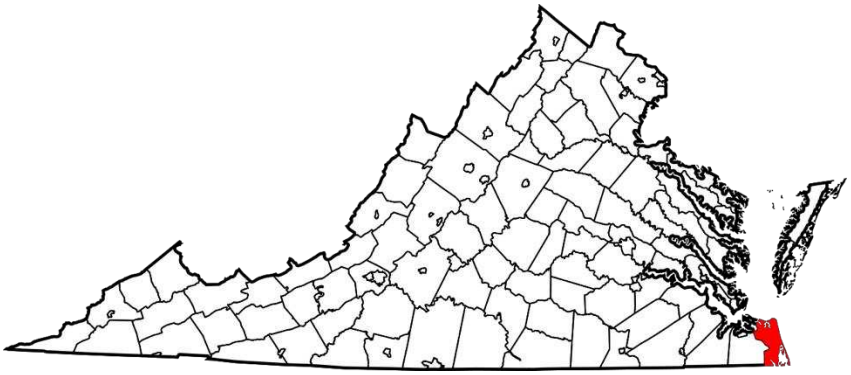

Figure 1: Map of the State of Virginia, USA, with Virginia Beach in red (right)

Figure 2: Map of Virginia Beach, Virginia, USA

Vehicles will depart from different locations in the city, which is determined by the residential and business density and the travel patterns (e.g., moving between the residential areas to business districts at peak times). During the flooding caused by hurricanes or coastal floods, the movement of vehicles will be influenced, because some roads are disrupted and even cut off. The inundation is obtained from the study by Loftis et al. (2018). Loftis et al. (2018) produced 24 inundation maps for each hour at 10/09/2016 UTC with the depth of water featuring the impact of Hurricane Matthew in VB. The inundation map for 10:00-11:00 UTC was used for the road disruption estimation in the present study because the impact of inundation reaches its maximum during this period.

We evaluate the variations of the road network system's performance under these phenomena with the various metrics: Average, maximum, minimum and standard deviation of

vehicle travel time; Number and percentage of cars unable to reach destination within simulation time (10 hours); Number and percentage of cars that blocked by the inundation. By building the simulation and conducting the corresponding analysis, this study aims to uncover new understandings of the system's behaviors during flooding, devise optimization insights, and help increase the roadway system's overall resilience.

2. Literature review

Some previous studies focused on simulating the movement of emergency service vehicles (e.g., emergency medical services, fire rescues, police dispatches) on the individual level to measure the impact of inundation on road networks and emergency services relying on transportation. Coles et al. (2017) study is one of the most typical in this direction. They obtained the flooding areas and the inundation depth from hydrodynamic models and set a threshold to identify the roads the flooding would disrupt. A micro-scale and individual-level traffic simulation was conducted with the disrupted road network and pre-defined network connectivity rules, including speed limit, turn restriction, and one-way traffic. A quickest routing analysis could be conducted based on the traveling speed in the simulation and suggest the traveling time increase due to the disruption. The following studies attempted to improve the rules defined in the simulation. Koch et al. (2020) developed an agent-based simulation model for ambulance moving with more accurate behaviors, where ambulances would respond to randomly generated calls and return to emergency medical service stations after delivering the patient to the hospital. Yin et al. (2020) emphasized the temporal variance of traffic status and emergency service demands and proposed a scenario-based approach. Four scenarios, including midnight, night, day, and

morning and evening peak, were designed to tune the parameters in the simulation. Yu et al. (2020) designed different scenarios for three groups of vulnerable populations in the U.K., including the elderly, young children, and people with terrible health, who have different demographical distributions and levels of demands.

Other studies have investigated how the road network for the general population is impacted after a disaster. He et al. (2012) created a discrete-time model of the traffic disruption of the collapse of the I-35W Mississippi River Bridge in Minneapolis, Minnesota. They proposed the prediction-correction model, which is crucial for disruption simulations as drivers react differently under irregular circumstances. A minimization optimizing constraint was set upon the computation of the drivers' predictions that will be corrected through travel cost iteratively after each time unit as the disruption takes place. This allows the modeling of driving behavior to consider both past experiences and anticipations of traffic conditions. Their comparison results between simulations and empirical data showed greater capability from the prediction-correction model to capture recovery characteristics of unexpected disruptions.

The transportation disruption simulation during flood incidents has also been extensively studied. Li et al. (2018) developed a model to capture the traffic disruption in urban areas during pluvial flash floods (PFFs), which are low-frequency but high-disruption floods. Their road network model consisted of roads, links, and crosses with link direction constraints that will be blocked once the flooding exceeds 30cm. The BPR model, which describes the relationship between volume and travel time, and the A* algorithm, which is a heuristic optimization of the Dijkstra algorithm, were used to assign trip paths. To augment the actual travel data, the researchers used a random distribution to complement trips in their simulation. Improvements

can be made upon practical calibration of the BPR model's parameters and increasing its scale for better travel data validation across the entire city. Shahdani et al. (2022) employed the mesoscopic simulation technique to identify traffic disruptions during flood events and used the Santarém, Portugal flood as a case study. The researchers evaluated the functionality of the transportation network before and after the infrastructure failed from flooding using static and dynamic traffic models. The model incorporated the road network geometry, trip paths, road closures, and speed limitations to evaluate the redistribution of travel time and vehicle volume during the flood. The impact of flooding on transportation networks was assessed by analyzing the changes in traffic data (i.e., travel time, travel distance, and street speeds) under normal and flooded situations, Costa et al. (2020) used SUMO to model traffic through a network of partially blocked roads after an earthquake to determine the increased travel time and distance for drivers. In their model, the roads are represented by nodes and edges with different capacities and directions. They determine the capacity of each edge, or road, by calculating how much debris would fall on the road after the earthquake. This work is novel because they study the impact of partially closed and fully closed roads on the transportation network, while most studies only study the impact of fully closed roads after a disaster. Hou et al. (2022) also studied how partially closing roads in a transportation network impacts mobility for the general population. Because current models do not consider the decreased capacity of partially closed roads, the calculated updated travel time is inaccurate and does not represent traffic flow after a disaster. They use a cellular automata model to model this behavior and determine an accurate travel time function.

In addition to understanding the transportation disruption behavior after disasters, engineers must optimize the transportation strategies after disasters for rapid and safe evacuation. Accordingly, Escribano-Macias et al. (2020) developed a pre-planning model that is based on a hybrid simulation-optimization method to optimize evacuation response strategies by demand staging and signal phasing. The evacuation policies were evaluated by a dynamic traffic assignment model to incorporate congestion, queuing, and vehicle spillback. The study employed derivative-free optimization algorithms to identify optimal evacuation strategies based on a benchmark dataset and examined the impacts of different network conditions on evacuation efficiency by varying the number of activated paths and frequency of departure. The simulation results indicated that combining departure time scheduling with signal phasing is an effective approach to enhance evacuation efficiency.

3. Conceptual model

Assessing and predicting the impact of the flooding inundation on transportation with respect to the population density distribution induced traffic flow as well as the cascading effects is critical for taking mitigation measures. A system is developed step by step in this section to predict and measure the level of impact in advance of flooding incidents.

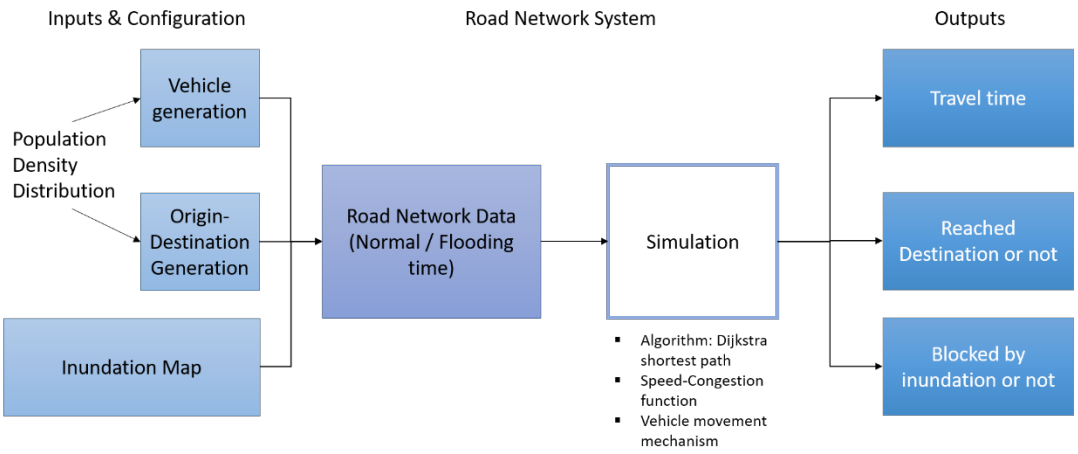

Figure 3: Graphic of the traffic simulation conceptual model

3.1. Inputs & Configuration

The Input dataset includes the road network data (road node form), population density distribution and the inundation map of the City of Virginia Beach. The generation of vehicles together with their origins and destinations are subsequently generated according to the population density distribution. The road network data has two versions after incorporating the inundation map, one for normal time and one for flooding time.

3.2. Road Network System & Inundation analysis

To obtain the direct impact of inundation on road network, the inundation is mapped onto the road network to determine the road disruption. The inundation area was overlaid on the road segments and the maximum inundation depth was used at the degree of inundation for the road segment. Figure 4 illustrates the inundation depth on the road segments. To eliminate the influence of outliers, only inundation below 6 ft was plotted. The darkness of the color suggests the depth of inundation, and the darkest color indicates an inundation of 6 ft. The roads in the

city's central area were inundated to varying degrees. Roads are assumed to be impassable if the inundation exceeded 25cm (0.82ft), according to a study by Yu et al. (2020). Impassable roads were taken off the whole road network. The updated road networks are then used in the macroscopic traffic analysis.

Figure 4: Inundation depth on the road network, City of Virginia Beach, VA. The darkest color indicates an inundation depth of 6 ft.

3.3. Simulation

The simulation process contains route calculation (Dijkstra's algorithm), travel speed function (variates with congestion), travel time computation and vehicle movement computation.

Each car will be assigned its origin and destination at its generation, then have its route computed with consideration to inundations. Then the car enters its first road node and has its speed and time left on this route computed. All the cars will share the same simulation time

steps. After each timestep, the car object will check if it has spent the full duration on the road that it is currently on and decide whether it will move on to the next road node. This process will be performed iteratively in the full simulation time span. Eventually, the vehicle will be removed from the network as it leaves the last road node on its route and reaches its destination.

3.4. Outputs

The simulation will report back: as the following table.

Table 1. Table of reported outputs

Simulation Logistics	Total cars generated, Simulation time
Metrics of Interest	Number of cars left simulation (reach destination) Number of cars in simulation (didn't reach destination) Number of cars couldn't find route (blocked by inundation) Average travel time Standard deviation of travel time Max. & Min. of travel time

4. Simulation Model

This section introduces the rules in the simulation model in detail. It first introduces the format of the input data, including the road graphs and the pre-defined rule of generating vehicles as well as their origins and destinations (OD). The generated vehicles with different ODs can move in the road graph. The rule of the interaction among vehicles is defined by building a function between the density of vehicles on a road segment and their theoretical speed.

4.1 Road network

The road network in normal times and flooding times are different. In the normal time, the road network data, which is originally geographical data in shapefile format, is converted into a graph in json format. After importing into the computation environment, the dict of the json contains the number of roads as keys, and the neighbors of the road as values. Each value is a list of tuples, whose first element is the number of the neighbor, and the second element is the distance traveling from the road in key to the number in the neighbor. In the flooding time, the road network data in the graph format has the same structure as that in normal time. However, to indicate the impact of flooding and inundation, part of distances traveling from one road to another is set as infinity, which means the link between these two roads is disrupted and no vehicle can move in this direction.

4.2 Vehicle generation

The vehicle generation module is trying to answer two questions, the first of which is "How many vehicles should be generated per minute?" According to a report published by the Virginia Beach Department of Transportation the annual average daily traffic in the city is 180,000. Assume 60% of the traffic volume occurs at the morning and evening peaks, and each peak time is 2.5 hours long. Therefore, there would be about 300 vehicles generated per minute in the city. We would use the vehicle generation rate in the peak in the whole simulation period (8 AM to 6PM) to avoid any underestimation of the traffic flow, as the purpose of the study is to evaluate the worst case of the traffic.

The second question the vehicle generation module must answer is "What's the spatial distribution of the vehicle generation?" We assume the spatial distribution of the vehicle

generation is primarily determined by the commuting of people for daily work. In the morning, most of the vehicles would depart from the residential areas in the city and move to the business centers in the city, and vice versa. We searched “apartments” and “corporate office” in Google Map to observe the residential and business centers in the city. As shown in Figure 5, the orange points are the residential centers, and purple points are the business centers.

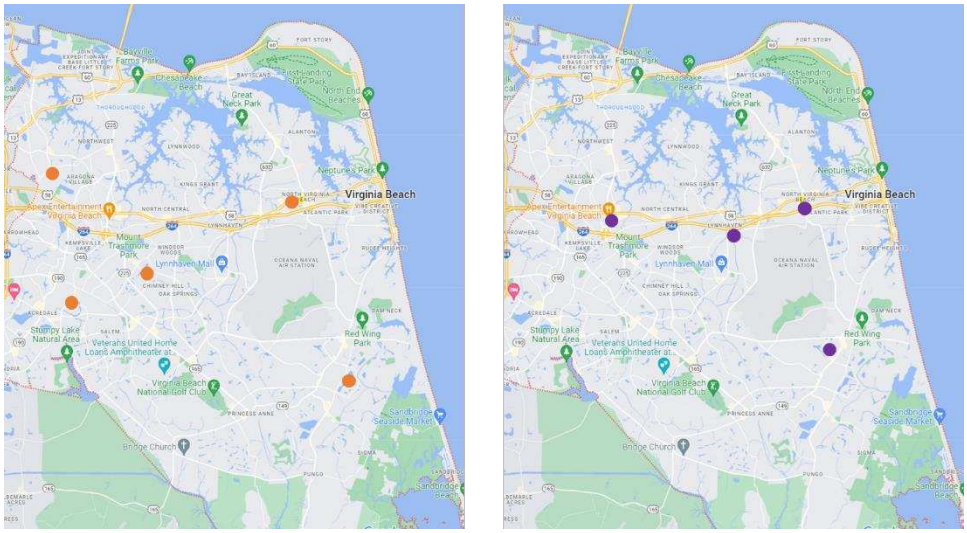

Figure 5: Residential and business centers in the VB. Orange points are the residential centers and the purple points are the business points.

We located several centers of business areas and residential areas, then assume higher population density going from the business centers to residential areas during daytime hours and vice versa during off work hours. If a vehicle departs from residential areas, the conditional probability that a vehicle is generated from a specific road segment is determined by the distance of this road to the nearest residential center (i.e., the longer the distance, the lower the probability), and the probability that a vehicle is going to a specific road segment is determined by the distance of this road to the nearest business center (i.e., the longer the distance, the lower

the probability). The rule is also applied for generating vehicles departing from business areas and moving to residential areas.

Given the conditional probabilities, the probability a vehicle is departing from residential places and business places should still be determined. A sequence of numbers is generated using the exponential function to represent the weight of “departing from residential places” from 8 AM to 6 PM. Naturally, the value of the number should decrease from 8 AM to 6 PM, as people depart from their homes in the morning and go back home in the evening. Similarly, another sequence of exponential numbers with an ascending trend is used to represent the weight of “departing from business places. The weights are normalized time point wise to generate probabilities of these two incidents.

```python
import random
import numpy as np

def originDestination_generation(p_res, p_biz, p_res_roads, p_biz_roads):
    # Args
    #   p_res: the probability a car is generated by residential effect
    #   p_biz: the probability a car is generated by business effect
    #   p_res_roads: given residential effect, the p of generating a car at each road
    #   p_biz_roads: given business effect, the p of generating a car at each road

    p_res_roads = {k: v * p_res for k, v in p_res_roads.items()}
    p_biz_roads = {k: v * p_biz for k, v in p_biz_roads.items()}
    p_generation = {key: p_res_roads[key] + p_biz_roads[key] for key in p_res_roads}

    roads = list(p_generation.keys())
    p = list(p_generation.values())

    # replace non-finite values with the maximum finite value
    # max_finite_value = np.nanmax([w for w in p if math.isfinite(w)])
    # finite_p = [w if math.isfinite(w) else max_finite_value for w in p]
    # p = finite_p

    return random.choices(roads, p, k = 1)[0]

def numbers_exponential(total_timeStep):
    numbers_exp = np.exp(-0.3 * np.arange(total_timeStep))
    numbers_exp /= np.sum(numbers_exp)
    return numbers_exp

def p_residential_business(numbers_exp):
    numbers_exp_flip = np.flip(numbers_exp)
    numbers_exp_sum = numbers_exp_flip + numbers_exp
    p_residential = numbers_exp / numbers_exp_sum
    p_business = numbers_exp_flip / numbers_exp_sum
    return p_residential.tolist(), p_business.tolist()
```

Figure 6: Codes for generating vehicles

Along with the conditional probabilities, the vehicle generation distribution could be determined. The start and end points of the vehicles are determined by sampling from the distribution. Dijkstra's algorithm is used to find the shortest paths between the start point and end point. This path becomes the route for the car.

```
def dijkstra(start, end):
    distances = {node: float('infinity') for node in road_graph}
    distances[start] = 0
    previous_nodes = {node: None for node in road_graph}

    priority_queue = [(0, start)]
    while priority_queue:
        current_distance, current_node = heapq.heappop(priority_queue)

        if current_distance > distances[current_node]:
            continue

        tied_neighbors = []
        min_distance = float('infinity')

        #print(road_graph)
        for neighbor, weight in road_graph[current_node]:
            # if neighbor not in inundation_roads:  # Avoid inundation roads
            if neighbor not in inundation_roads[current_node]:  # Avoid inundation roads
                distance = current_distance + weight

                if distance < distances[neighbor]:
                    distances[neighbor] = distance
                    previous_nodes[neighbor] = current_node
                    heapq.heappush(priority_queue, (distance, neighbor))

                # Check for ties
                if distance < min_distance:
                    min_distance = distance
                    tied_neighbors = [(neighbor, current_node)]
                elif distance == min_distance:
                    tied_neighbors.append((neighbor, current_node))

        # If there's a tie, randomly pick one of the tied neighbors
        if tied_neighbors:
            chosen_neighbor, chosen_previous_node = random.choice(tied_neighbors)
            if distances[chosen_neighbor] == min_distance:
                distances[chosen_neighbor] = min_distance
                previous_nodes[chosen_neighbor] = chosen_previous_node
                heapq.heappush(priority_queue, (min_distance, chosen_neighbor))
    return distances[end], distances, previous_nodes
```

Figure 7: Code for Dijkstra's Algorithm

4.3 Vehicle movement and activity log

The vehicles are generated every 6o time steps in the simulation. For each discrete time step in the simulation, the generated cars are moved to the next index along their7 path based on the number of cars on the road and the length of the road. Once a car reaches its endpoint, it is removed from the simulation. For each time step, the number of cars generated, the number of cars in the simulation, the number of cars that left the simulation, and the number of cars that could not find a route is outputted as text.

```
def compute_time_left_at_this_road(car):
    # Calculate time_left_at_this_road of a car based on the number of cars on its road and the length of the edge
    current_road = car.at
    next_road_index = car.route.index(current_road) + 1
    next_road = car.route[next_road_index] if next_road_index < len(car.route) else None

    if next_road is None:
        return 0

    edge_length = None
    for neighbor, weight in road_graph[current_road]:
        if neighbor == next_road:
            edge_length = weight
            break

    if edge_length is None:
        return 0

    current_road_node = road_nodes_list[current_road]

    ## edited by dak
    # speed function w.r.t car flow
    current_road_width = road_width[current_road]
    # max speed 80km/h
    # edited by Zefang, V_max in m/s
    V_max = 8000/6/60
    # 3.6576m assumes 12ft lanes
    lane_num = math.floor(current_road_width/3.6576)
    # assume 20ft of space needed per car
    max_cars_on_the_road = math.floor(edge_length/6.096)*lane_num
    # setting speed to 10 mph if at full copacity
    if len(current_road_node.cars_on_the_road) >= max_cars_on_the_road:
        # edited by Zefang, time in second
        return edge_length/1600*6*60
    else:
        reduction_rate = len(current_road_node.cars_on_the_road)/max_cars_on_the_road
        return edge_length/(V_max*(1-reduction_rate))
```

Figure 8: Code for speed & time left on road node computation

It should be noted that there are interactions among cars. The number of vehicles on a road segment indicates the level of congestion, which determines the speed of vehicles. When a

car enters a new road node on its route, the function above will compute an average speed of this car on this road node related to its congestion condition. Subsequently, the duration of this car on this road node can be obtained. We set the speed-congestion function to be as follows, more cars on a road node would linearly decrease the maximum speed(50mph) of the vehicles entering it, the minimum speed is set at 10 mph.

n: number of cars on this road node;
n_{max}: maximum number of cars on this road node(lane max.)
ρ: speed reduction rate;
L: road node length;
V_{max}: maximum speed on road node, manually set at 50mph

$$\rho = \frac{n}{n_{max}}$$

$$V = \frac{L}{V_{max}(1-\rho)} (\forall \rho < 1)$$

$$V = 10 (\rho = 1)$$

Figure 9: Speed-Congestion Function

4.4 Verification

The simulation has been verified through unit testing, code traces, and run-time assertions to conclude that the simulation can capture real-world behavior. Because our system has many roads and intersections, the simulation was first tested on a simple road network with a lower car generation rate.

Every time during the car generation, we printed out the starting point, ending point, traveling routes for all the generated cars. We manually checked the routes to make sure that they were valid. During every time step, we printed out the number of total cars generated, the

number of cars left simulation, the number of cars currently in simulation, and the number of cars that couldn't find a route till the current timestep. We summed up the number of cars left simulation, the number of cars currently in simulation, and the number of cars that couldn't find a route to check if it was equal to that of the total number of cars generated. At the end of the simulation, we print out the length of the array that stored the travel time of all the cars that left the simulation to check if it was equal to the number of cars that left the generation. We also printed out the average time per car that left the simulation and checked if it was reasonable. Since the codes passed all the tests that we mentioned above, we can safely assume that the codes could simulate the system well.

5. Experimental results and validation

5.1 Experimental Procedure

Peak Period Analysis: For this experiment, the traffic in the City of Virginia Beach is modeled during the peak morning and peak evening travel periods, combined, for an inundated and normal scenario. We chose to model traffic during the peak periods to see how flooding impacts the road network at the busiest times. During the peak morning period, most vehicles begin at their residence and end at their workplace. During the peak evening period, most vehicles are expected to travel in the opposite direction. These periods are merged together during the analysis.

Inundated vs. Intact Scenarios: To determine the inundated road network, we overlaid the flooding map onto the map of Virginia Beach. Roads are assumed to be impassable if the

inundation exceeded 25cm (0.82ft), according to a study by Yu et al (2020). Impassable roads were taken off the road network. These two road networks, the inundated and intact, were used in the simulation model to determine and compare the average travel times.

Vehicle Generation and Destinations: During the morning peak period, most vehicles are expected to begin at their residence and travel to their workplace. To model this relationship, vehicles are generated based on the population density of Virginia Beach during the morning peak period and end at the city center, which is assumed to be the location of most workplaces. During the evening peak period, when most vehicles are expected to begin at their workplace and end at their residence, vehicles' origin and destination are generated in the opposite manner.

Steady State Analysis: Rather than evaluating the steady state of the simulation numerically, we used a timestep value determined from the literature that was already confirmed to be steady state, 36000 seconds (10 hours).

5.2 Experimental Assumptions:

In this work, we model traffic throughout the entire City of Virginia Beach during the peak morning and peak evening periods during a flooded and intact scenario. Due to the complex nature of traffic modeling, several assumptions had to be made.

- *Vehicle generation:* For the morning peak period, the vehicles are generated based on the population density of Virginia Beach, and the final destination is the city center of Virginia Beach. This is reasonable because most trips in the morning peak period are commuters traveling from their residence to their workplace. For the evening peak period, the

vehicles are generated in the city center and their final destination is based on the population density or the inverse of the morning peak period. This is reasonable because most trips in the evening peak are commuters traveling to their residence from their workplace.

- *Vehicle speed:* The speed of the vehicles on a road segment is based on the density of cars on that road segment. This is consistent with traffic theory and used in other studies as well.
- *Pedestrian interaction and unusual events:* Pedestrian movements are not modeled in this simulation and do not have an impact on the traffic flow. This is reasonable because the traffic network is large and has a high throughput, so there are few pedestrian attractions in the area. Emergency situations, such as ambulances or firetrucks, are also not modeled in this simulation and do not have an impact on the traffic flow. This is reasonable because the amount of emergency situations is negligible compared to the amount of regular traffic.
- *Exiting the simulation:* Vehicles may only exit the simulation when they have reached their final destination. There are no exit lanes or roads that would permit a vehicle to leave the simulation if it has not reached its final destination. Once the vehicle reaches its destination, it exits the simulation automatically. This is reasonable because the traffic network is large and we can assume that all vehicles are traveling within the City of Virginia Beach.
- *Vehicle attributes:* In this simulation, all vehicles are modeled as identical and modeled as a typical passenger vehicle. There are no trucks or buses in the simulation because

according to real-world data, only 5% of the vehicles in Virginia Beach are multi-axial and therefore assumed to be negligible for the simulation.

5.3 Analysis:

The traffic simulation was performed for both the non-flooded and flooded scenarios. The road network for the non-flooded scenario had no segments removed, while the flooded scenarios have disconnected segments due to inundation blockage. For both scenarios, the simulation ran for 36000 timesteps (one timestep equals one second), which corresponded to 600 minutes (10 hours). In the simulation, 300 cars were generated randomly every minute in the whole system based on the population density, which corresponded to 180300 cars in total, and the cars in the simulation moved once a second.

Normal (intact) scenario: As shown in Figure 10, before the end of the simulation, most (97.3%) of the generated cars left the simulation (i.e., arrived at their destination); only 2.3% of the cars remained in the simulation; and only 0.4% cannot find a path from their origin to their destination because of a disconnect route.

Flooded (inundated) scenario: For the flooded scenario, in contrast, only 69.8% of the generated cars left the simulation, which is 28.2% lower than that of the normal scenario; 4.4% of the cars remained in the simulation, which is 87.1% higher than that of the normal scenario; and 25.8% cannot find a path, which is 64.1 times that of the normal scenario. The simulation results indicated that the flood negatively impacted the traffic dramatically.

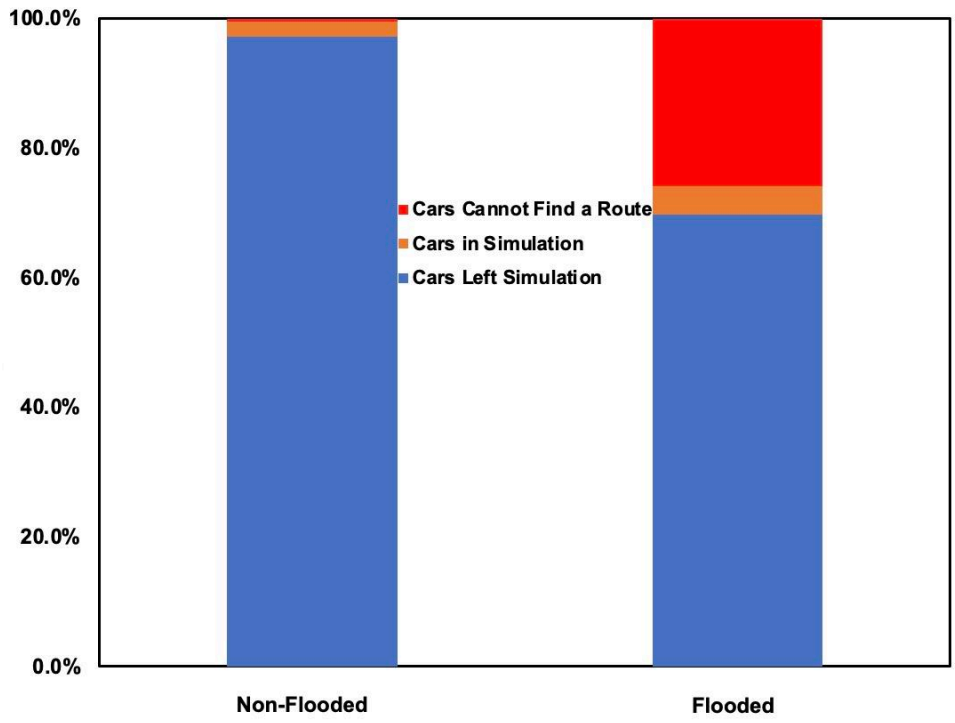

Figure 10: Results of traffic simulation for 10 hours in normal and flooded conditions with 300 cars generated per minute

Evaluation: As shown in Table 2, the average travel time during a flooded scenario is greater than the average travel time during normal conditions. These results are consistent with the real-world scenario. In the normal conditions, the average travel time was 695.7 seconds (about 11 and a half minutes), while in the flooded conditions, it was 1776.3 seconds (about 29 and a half minutes). Additionally, the standard deviation of the travel times increased from 371.6 seconds (about 6 minutes) to 1572.1 seconds (about 26 minutes), which shows that the travel times varied more in the flooded scenario than the normal scenario. This result is also consistent with expectations based on the real-world scenario.

Table 2. Summary of Simulation Results; Intact and Inundated

	Intact Road Network	Inundated Road Network
Average travel time (s)	695.7	1776.3
Standard deviation (s)	371.6	1572.1
Minimum travel time (s)	0.8	1.4
Maximum travel time (s)	4340.1	9524.4
Lower Bound for 95% Confidence Level	694.0	1767.6
Upper Bound for 95% Confidence Level	697.5	1785.0
Lower Bound for 99% Confidence Level	693.5	1764.9
Upper Bound for 99% Confidence Level	698.0	1787.7

5.4 Validation

The first method we used to validate our code was face validation. Face validation assesses the visual appearance of a simulation against the real-world system by comparing the output of the model to the expectation in the real world. In the real world, we expect the average travel time to increase when there is flooding on the roads compared to when there is no flooding, so we validated the model by making sure it reflected this phenomenon. As shown in the analysis section above, this is true for our simulation.

The second method we used to validate the model was behavior validation. Behavior validation compares the output of the model to the expected behavior of the phenomenon it represents, as well as to theories and empirical data. This involves assessing how the model's

behavior changes are certain variables are altered. In this simulation, we doubled the number of cars in the simulation and expected the travel time to increase. This is according to theory, which states that vehicular speed decreases as road density increases. The average travel time increased when we doubled the number of cars, which is consistent with theory.

6. Conclusion

This study presents a description of a simulation code developed to model travel times through Virginia Beach when some of the roads are inundated and results obtained from the simulation. The simulation was able to model real-world phenomena, even with several modeling assumptions. By comparing the travel time for the inundated road network and the unaffected road network, we observed that travel time increased by 1080.6 seconds (about 18minutes) during the peak periods.

For future work, many of our simplifying assumptions should be eliminated to represent reality more accurately. One assumption in our simulation was constant vehicle type and size, however, there are multiple types of vehicles that could be on the Virginia Beach road network. Different types of vehicles are affected by road blockages differently, so allowing for different vehicle sizes would change the output of the model. Additionally, it would be interesting to calculate the increase for each type of vehicle in the simulation, rather than merge them into one metric. Another potential for future work would be to analyze how travel time varies across the transportation network rather than looking at the entire network. This analysis would help city planners and engineers determine where flooding impacts the population the most and should consider adding flooding mitigation techniques.

7. References

- Alam, M., & Habib, M. (2020). Modeling Traffic Disruptions during Mass Evacuation. Procedia Computer Science. 170, 506-513. https://doi.org/10.1016/j.procs.2020.03.115
- Coles, D., Yu, D., Wilby, R. L., Green, D., & Herring, Z. (2017). Beyond 'flood hotspots': Modelling emergency service accessibility during flooding in York, UK. Journal of Hydrology, 546, 419–436. https://doi.org/10.1016/j.jhydrol.2016.12.013
- Costa, C., Figeuiredo, R., Silva, V., & Bazzurro, O. (2020). Application of open tools and datasets to probabilistic modeling of road traffic disruptions due to earthquake damage. Earthquake Engineering & Structural Dynamics. 49(12), 1236-1255. https://doi.org/10.1002/eqe.3288
- Escribano-Macias, J., Angeloudis, P., & Han, K. (2020). Optimal design of Rapid evacuation strategies in constrained urban transport networks. Transportmetrica A: Transport Science, 16(3), 1079-1110. https://doi.org/10.1080/23249935.2020.1725179
- He, X., Liu, H. X. (2012). Modeling the day-to-day traffic evolution process after an unexpected network disruption. Transportation Research Part B: Methodological. Volume 46, Issue 1 https://doi.org/10.1016/j.trb.2011.07.012
- Hou, G., Chen, S., & Bao, Y. (2022). Development of travel time functions for disrupted urban arterials with microscopic traffic simulation. Physica A: Statistical Mechanics and its Applications. 593, 126961. https://doi.org/10.1016/j.physa.2022.126961

- Koch, Z., Yuan, M., & Bristow, E. (2020). Emergency Response after Disaster Strikes: Agent-Based Simulation of Ambulances in New Windsor, NY. Journal of Infrastructure Systems, 26(3), 06020001. https://doi.org/10.1061/(asce)is.1943-555x.0000565
- Li, M., Huang, Q., Wang, L., Yin, J., & Wang, J. (2018). Modeling the traffic disruption caused by Pluvial Flash Flood on the intra-urban road network. Transactions in GIS. Volume 22, Issue 1 https://doi.org/10.1111/tgis.12311
- Shahdani, F., Santamaria-Ariza, M., Sousa, H., Coelho, M., & Matos, J. (2022). Assessing Flood Indirect Impacts on Road Transport Networks Applying Mesoscopic Traffic Modelling: The Case Study of Santarém, Portugal. Applied Sciences, 12(6), 3076. https://doi.org/10.3390/app12063076
- Yin, J., Yu, D., & Liao, B. (2020). A city-scale assessment of emergency response accessibility to vulnerable populations and facilities under normal and pluvial flood conditions for Shanghai, China. Environment and Planning B: Urban Analytics and City Science, 0(0), 239980832097130. https://doi.org/10.1177/2399808320971304
- Yu, D., Yin, J., Wilby, R. L., Lane, S. N., Aerts, J. C. J. H., Lin, N., Liu, M., Yuan, H., Chen, J., Prudhomme, C., Guan, M., Baruch, A., Johnson, C. W. D., Tang, X., Yu, L., & Xu, S. (2020). Disruption of emergency response to vulnerable populations during floods. Nature Sustainability, 3(9), 728–736. https://doi.org/10.1038/s41893-020-0516-7