Assignment 1

Data Quality Issues for Data-Driven Simulation

EPA133a Advanced Simulation

Group 16
Rachel Delvin Sutiono, No. 6284736
Celia Martínez Sillero, No. 6222102
Daniela Ríos Mora, No. 6275486
Thunchanok Phutthaphaiboon, No. 6141153
Yao Wang, No. 6157513

Outline

1. Introduction

High-quality data is essential for developing accurate simulation models. This report evaluates data quality issues in Bangladesh's transport infrastructure dataset, which contains information on the locations of roads and bridges—key elements for assessing bridge infrastructure criticality, vulnerability, and investment priorities. However, errors and missing details in the data can compromise simulation outcomes. To address this, we identify the main issues, propose strategies for resolution, prioritize the most critical problems, and implement selected solutions. The objective is to enhance data quality and ensure the dataset is ready for future simulation tasks.

2. Conceptual analysis of data quality issues and solutions

To understand the quality of the transport data for Bangladesh, we first explored the overall statistics of the data, as shown in Table 2.1 and Table 2.2. Alongside this, we visualized the data using the provided Java simulation tool to inspect how the roads and bridges appeared on the map. This helped us identify several mistakes in the data, which could lead to problems in future simulations.

Table 2.1: Data quality of 'BMMS_overview.xlsx'

	missing_values	data_types	duplicates	unique_values
road	0	object	0	741
km	1	float64	0	14483
type	0	object	0	14
LRPName	0	object	0	1502
name	347	object	0	9874
length	9	float64	0	2814
condition	0	object	0	4
structureNr	0	int64	0	21407
roadName	1	object	0	748
chainage	0	object	0	14484
width	3117	float64	0	956
constructionYear	3118	float64	0	63
spans	3117	float64	0	24
zone	1	object	0	10
circle	1	object	0	21
division	1	object	0	65
sub-division	1	object	0	126
lat	94	float64	0	18523
lon	94	float64	0	18079
EstimatedLoc	0	object	0	6

Table 2.2: Overall statistics of data from 'BMMS_overview.xlsx'

	km	length	structureNr	width	constructionYear	spans	lat	lon
count	21406.000000	21398.00000	21407.000000	18290.000000	18289.000000	18290.000000	21313.000000	21313.000000
mean	41.217405	16.62539	110961.360676	7.813633	1990.171961	1.465500	23.828505	90.227328
std	78.810955	43.85946	6319.200562	6.193465	10.123705	1.179609	2.247895	3.364169
min	0.000000	0.20000	100001.000000	1.380000	1942.000000	1.000000	0.000000	0.000000
25%	6.207250	2.68000	105466.500000	5.000000	1985.000000	1.000000	22.942527	89.377806
50%	15.147000	5.60000	110978.000000	7.300000	1992.000000	1.000000	23.823743	90.289194
75%	35.012250	15.80000	116441.500000	9.700000	1998.000000	1.000000	24.720939	91.288417
max	522.718000	1786.00000	121862.000000	702.000000	2013.000000	31.000000	91.544194	93.298416

In the following sections, the identified data issues are classified based on the framework provided by Huang (2013). Additionally, each section specified the dataset associated with the error and the reasons why these are critical.

2.1 Misplaced coordinates of location reference points (LRPs)

Dataset: _roads.tsv
Criticality: High

Description: The coordinates (Latitude and longitude) of some of the Location Reference Points (LRPs) are incorrect, causing spikes in roads (see Figure 2.1). This can lead to inaccurate routing in future simulations, resulting in poor agent-based modelling outcomes and wrong transportation infrastructure assessment.

Data Quality Category: The data quality category is classified as **semantic accuracy** because it concerns the meaningful correctness of geographic coordinates rather than their syntactic validity or completeness. Although the dataset includes latitude and longitude values, their inaccuracy in representing real-world locations compromises the intended meaning of the data.

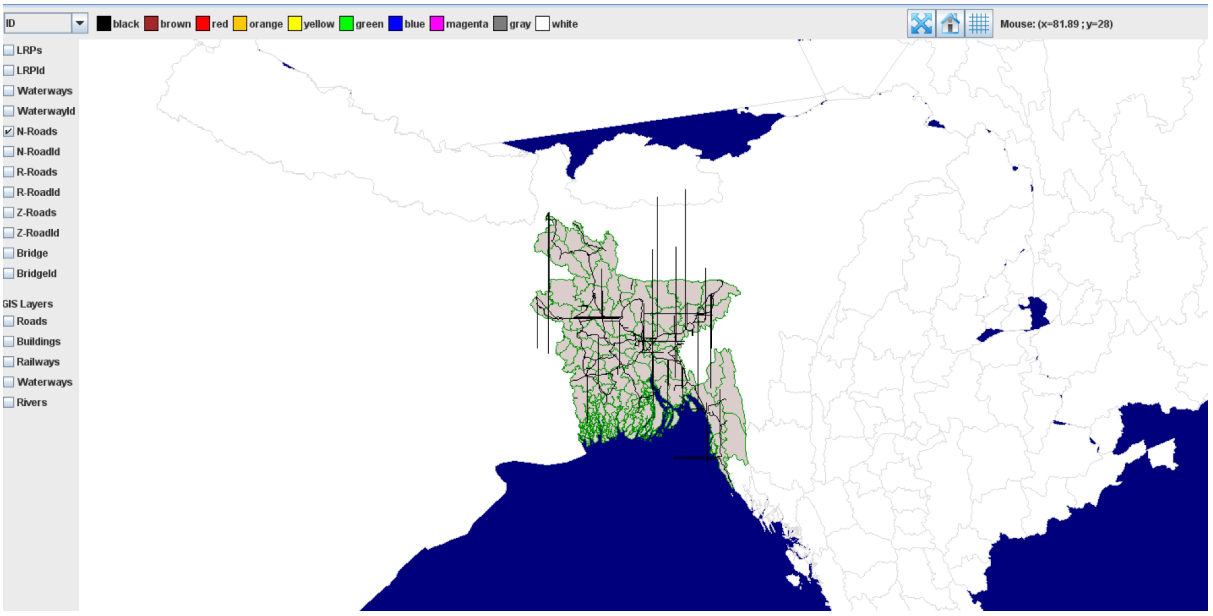

Figure 2.1: Mistakes in LRP's coordinates of the national roads

Solution: Calculate the haversine distance between consecutive LRPs for each road. If these distances exceed a specific threshold based on the quantile of all the distances in the road under study, they are classified as “offroads” and fixed accordingly. Since there are several types of cases, we have opted to create a sequential process in which, even though the structure remains similar, each iteration addresses a different offroad type. Notably, we tried to make the code functions as reusable as possible so that some parts of the iterations are identical by using the same functions. In Section 4.1, the program’s algorithm will be explained further.

2.2 Missing coordinates for bridges and duplicate entries.

Dataset: BMMS_overview.xlsx
Criticality: High

Description: Some bridges in the dataset lack geographic coordinates (latitude and longitude), making it impossible to accurately position them on the map (see Table 2.1). Without this information, these bridges cannot be connected to roads, disrupting the structure of the transport network. Additionally, there are duplicate entries for some bridges, which leads to inconsistencies within the dataset. These issues may result in disconnected routes, unrealistic agent behaviour in simulations, and inaccurate travel time calculations.

Data Quality Category: The data quality issues in the dataset fall under semantic completeness and mapping consistency. The absence of geographic coordinates means that crucial values required to accurately position bridges on the map are missing. Since this omission prevents the dataset from fulfilling its intended purpose, it aligns with the definition of semantic completeness. Additionally, duplicate bridge entries create inconsistencies, as multiple records may refer to the same real-world instance with different values. This lack of uniformity in key values violates mapping consistency, further compromising the dataset’s reliability.

Solutions: For the missing bridge location data and duplicate entries, the initial approach is to remove these problematic bridges from *BMMS_overview.xlsx*. Since accurate positioning requires integration with road data, the bridge locations will be determined later in the process, when both datasets (road and bridge data) are combined. This ensures that incorrect or incomplete data does not interfere with early-stage processing, allowing for more precise and reliable placement of bridges during the data file integration phase.

2.3 Bridges located outside the country boundaries

Dataset: BMMS_overview.xlsx
Criticality: High
Description: According to the visualization in Figure 2.2, some bridges have coordinates placing them outside Bangladesh’s borders. These errors suggest incorrect location data, which may lead to missing or disconnected road-bridge links in the simulation, affecting network connectivity and route calculations.

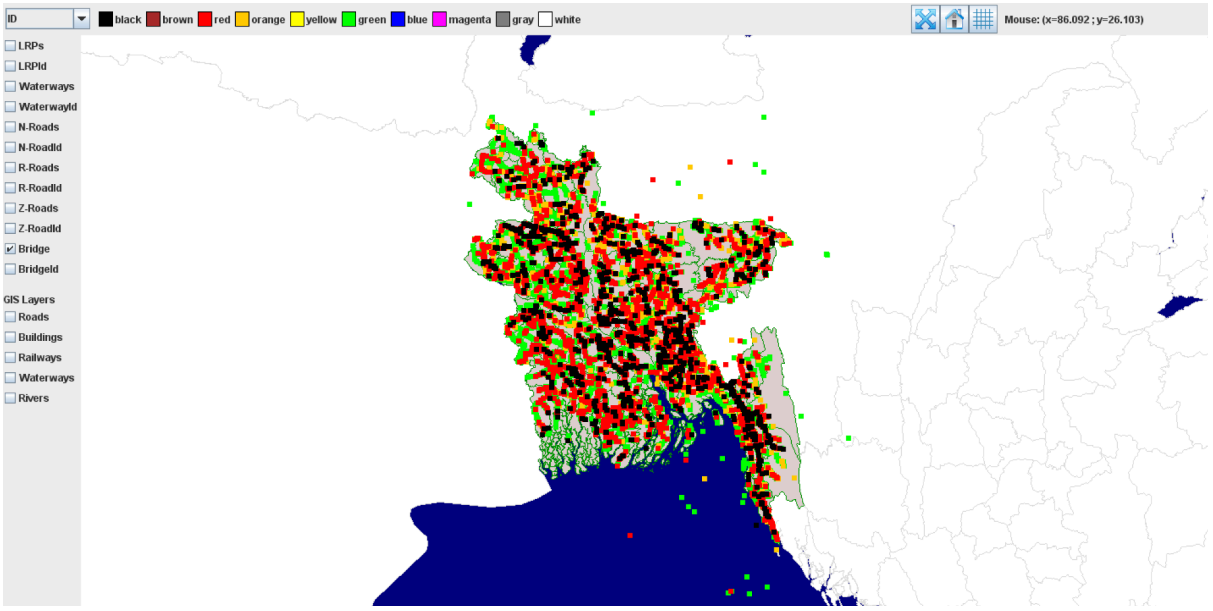

Figure 2.2: Mistakes in the coordinates of some bridges that are outside the boundaries of the country

Bangladesh is located between 20°34'N and 26°38'N latitude and 88°01'E and 92°41'E longitude (Bangladesh High Commission in Canberra, n.d). In addition, the overall statistics of data in Table 2 show that minimum and maximum coordinates fall beyond these boundaries (see Table 1). These errors suggest serious inaccuracies in location data.

Data Quality Category: The issue falls under **semantic accuracy** because it concerns the correctness of geographic coordinates in relation to real-world locations. Some bridges have latitude and longitude values that place them outside Bangladesh's borders, suggesting errors in location data. These inaccuracies could stem from simple misplacements, such as swapped latitude and longitude values or incorrect coordinates.

Solution: For the issue of bridges being located outside the national boundaries in the BMMS_overview.xlsx dataset, we assume the primary cause of this issue appears to be the misplacement of latitude and longitude values, which leads to incorrect bridge locations in the visualization. Given that Bangladesh is situated within the approximate coordinate range of 20°N to 27°N latitude and 88°E to 93°E longitude, any values significantly deviating from this range likely indicate a coordinate swap error. Our approach involves identifying bridges with out-of-bounds coordinates and attempting a latitude-longitude swap to reposition them correctly. B n This adjustment is expected to restore accurate bridge positioning, ensuring proper road-bridge connectivity in the simulation, which is crucial for maintaining a reliable transportation network model.

2.4 A group of bridges with no connecting roads and missing road data

Dataset: Combination of BMMS_overview.xlsx and _road.tsv

Criticality: High

Description: A cluster of bridges appears to be aligned in a connected sequence on the map, creating the illusion of a continuous path. However, no actual roads link them, and the corresponding road data is missing (see Figure 2.3). This misrepresentation of connectivity results in the isolation of these bridges. The lack of proper road links disrupts network continuity, leading to unrealistic routing behaviours in future simulations and adversely affecting the overall transport flow analysis.

Data Quality Category: This issue falls under semantic completeness because it involves missing road data essential for the dataset's intended purpose. While the dataset contains bridge locations, the absence of connecting roads means it fails to provide a complete and functionally accurate representation of the network.

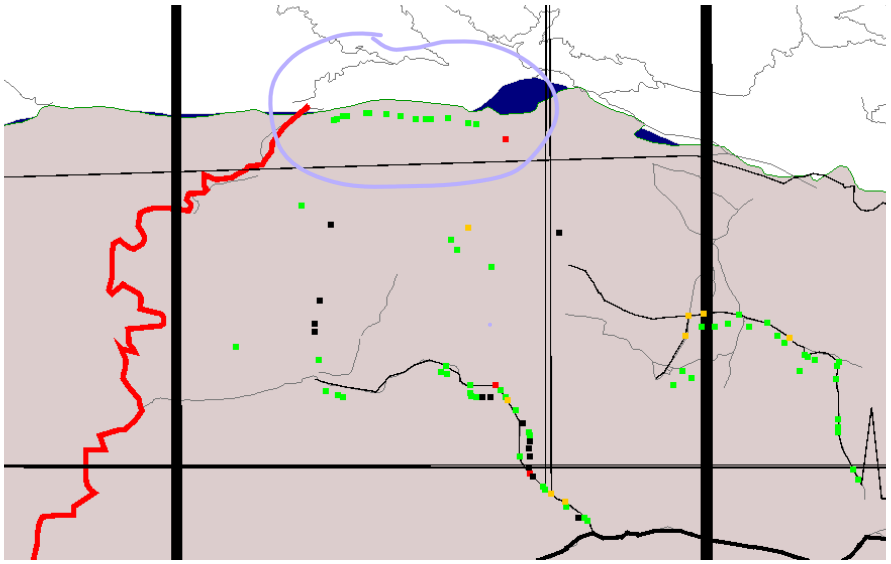

Figure 2.3: Mistakes in a group of bridges with no connecting roads

Solution: To address the issue of bridges appearing connected but lacking actual roadways, we conducted a manual verification using Google Maps to compare the dataset with real-world locations. Upon review, we discovered that these bridge clusters were incorrectly positioned and did not correspond to any existing road infrastructure. Since their locations are erroneous and no valid road data is available to link them appropriately, we decided to remove these misplaced bridges from the dataset. This action ensures a more accurate and realistic transportation network, preventing simulation errors caused by non-existent road connections.

2.5 Unusually large bridge width

Dataset: BMMS_overview.xlsx
Criticality: Medium

Description: The maximum recorded bridge width is 702 meters, while the average width is only 7.8 meters with a standard deviation of 6.19 meters (see Table 1). This extreme value is likely a data entry error and may lead to incorrect simulation outputs, particularly when bridge width may influence traffic capacity or structural assessments.

Solution: A solution based on quantiles can be implemented to address the unusually large bridge width issue. By calculating the 25th, 50th, and 75th percentiles for bridge width across the dataset, a threshold can be defined to identify typical bridge widths. Any values that fall significantly outside this range can be flagged as outliers. These outliers can then be reviewed for potential data entry errors and corrected accordingly.

3. Prioritization of data quality issues

Ensuring high-quality data is essential for reliable simulation modelling. The transportation data for Bangladesh presents several data quality challenges that affect road and bridge connectivity, routing accuracy, and the integrity of simulations. To systematically prioritize data cleaning tasks, we categorize the identified issues based on the three main dimensions of data quality: Syntactics, Semantics, and Pragmatics. The previous description of data quality issues provided a general overview, but now we dive deeper into a step-by-step approach. We establish a prioritization strategy that follows a logical sequence, addressing high-impact errors first before tackling less critical refinements.

3.1 Ensuring structural integrity – Highest Priority

Errors in data format, duplicate records, and inconsistencies directly affect network connectivity and simulation feasibility.

Step 1: Identify and correct misplaced LRP points in roads

- **Category:** *Syntactic Accuracy*
- **Issue:** Some road LRPs are misaligned, causing unrealistic deviations in the network.
- **Action:** Adjust LRP coordinates to ensure continuous road connectivity.

Step 2: Remove invalid bridges (Missing or zero geographic data)

- **Category:** *Semantic Completeness*
- **Issue:** Bridges with null or zero latitude and longitude cannot be mapped.
- **Action:** Remove incomplete entries to prevent disconnected links.

Step 3: Detect and remove duplicate bridge entries

- **Category:** *Syntactic Consistency*
- **Issue:** Multiple records exist for the same bridge, causing conflicting data points.
- **Action:** Keep only the most complete and updated bridge records.

3.2 Ensuring real-world accuracy – High Priority

After fixing structural errors, we address data misrepresentation issues that affect real-world alignment.

Step 4: Correct bridges placed outside Bangladesh

- **Category:** *Semantic Accuracy*
- **Issue:** Some bridges have incorrect coordinates outside valid geographic boundaries.
- **Action:** Apply latitude-longitude swapping and remove invalid records.

Step 5: Remove bridges without associated roads

- **Category:** *Semantic Completeness*
- **Issue**: Some groups of bridges appear in isolation and are not linked to any road.
- **Action:** Identify bridges beyond the last LRP names of the roads and remove them.

Step 6: Ensure consistent LRP coordinates between bridges and Roads

- **Category:** *Mapping Consistency*
- **Issue:** Bridges from different dataframes reference the same LRPs but have different coordinates.
- **Action:** Replace bridge LRP coordinates from the bridge file with value from the road file for consistency.

4. Implemented solutions and results

To implement planned solutions according to Chapter 3, we divide the tasks into two Jupyter notebook files: CleanRoads.ipynb and CleanBridges.ipynb. We provide a summary of how the program works in the README file.

4.1 CleanRoads.ipynb

This section describes the process of cleaning the road LRP (Location Reference Points) data, identifying outliers, and interpolating their locations to create a smooth and continuous road.

First, referring to Step 1 in Chapter 3, the original _roads.tsv file is transposed and reformatted into a long-format table, following pandas syntax standards. We export this new file as road_transposed.csv which will be an input for CleanBridges.py. The algorithm's output is saved as "infrastructure\cleaned\roads_cleaned.xlsx."

Next, we proceed with the data cleaning algorithm, involving:

1. Calculating the distance (km) between each LRP and its neighbor.
2. Identifying distances that surpass a set threshold and marking them as "off-roads" (outliers).
3. Addressing the outliers by adjusting their latitude and longitude using linear interpolation from the SciPy library.

This algorithm runs in two iterations: the first uses the 85th quantile as the threshold, and the second uses the 95th quantile to catch any remaining outliers.

4.1.1 The first iteration: Identifying and Addressing Offroads Beyond the 85th Quantile

The first iteration focuses on detecting LRPs whose distances to their neighbors exceed the 85th quantile of all the LRP distances for each road. The steps involved are as follows:

1. For each road in road_transposed.csv, calculate the distance between each LRP and its previous neighbor using the Haversine formula.
2. Determine the threshold as the 85th quantile of all LRP distances for the given road, then check if any LRP distance exceeds this threshold.
3. For each identified offroad LRP, calculate its new latitude and longitude using linear interpolation to find the average location between the previous and next neighboring LRP.
4. Replace the latitude and longitude of the offroad LRPs with the interpolated values.
5. Save the corrected dataset as df.

4.1.2 The Second Iteration: Refining Outliers Beyond the 95th Quantile

The second iteration follows the exact same steps as the first iteration, but with the threshold set at the 95th quantile of all the LRP distances per road. Since the first iteration removes most of the significant outliers, the higher threshold in the second iteration allows for capturing more subtle outliers that may have been missed initially, ensuring a cleaner final dataset.

Result and analysis

On the first glance, the outliers have been "pulled" to the road, hence corrected. For example, in Figure 4.1, the interpolated road shows no outliers left. It might seem like the roads were successfully corrected. However, when we visualize it with Java, we could see that the shape of the road, while having no extreme outliers, still misaligned with the real road (see Figure 4.2). This emphasizes that checking the result for each road by plotting them on the python notebook and visualizing them on Java is important to determine whether the road is completely corrected or not.

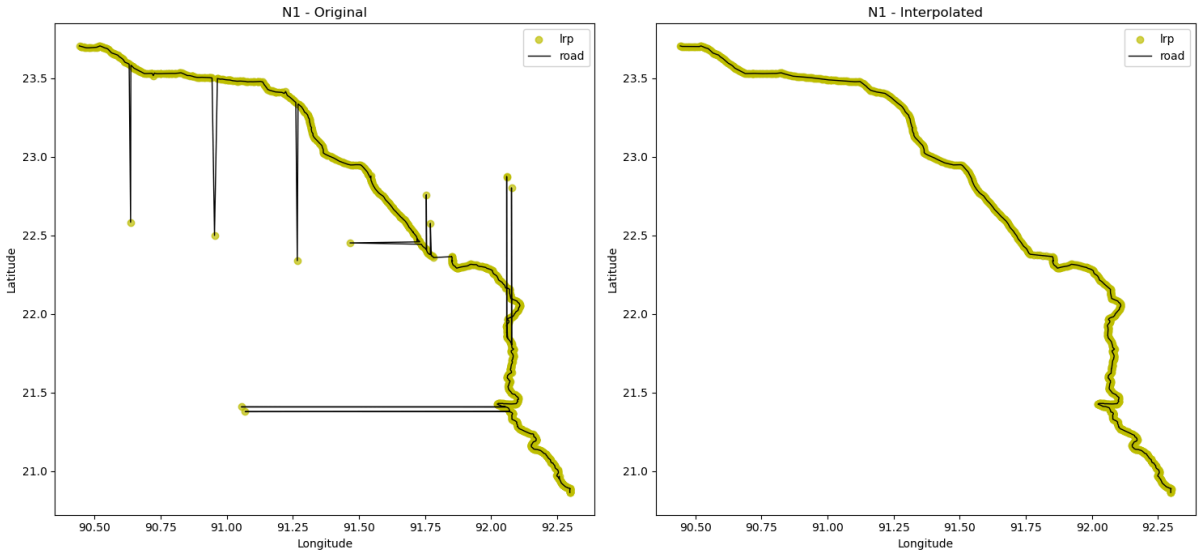

Figure 4.1: Road N1 before and after the second interpolation iteration

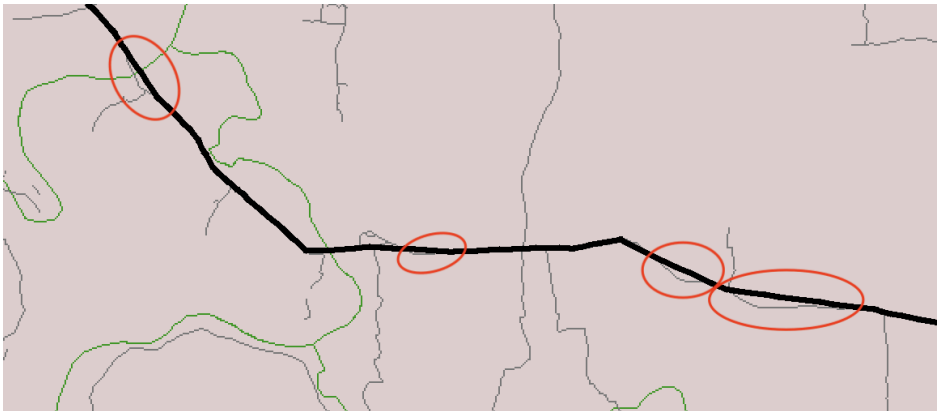

Figure 4.2: Corrected Road N1 (thick black line) and the real road (thin black line) visualized in Java. Red circles highlight misalignments.

In the figures below we could see that the result is sensitive to the amount of iteration. In the case of road Z4016, we could observe how running the algorithm two times with a different threshold helps addressing different outliers and creating a more fixed road. The first iteration cleans the larger offroads, while the second addresses the smaller outliers. For instance, we can see how the outlier that creates spikes on the original data plot is effectively removed by the first iteration. However, the milder outlier on the first iterated road—bottom right part of the road—is only removed after the second iteration. Although, we could see that not all outliers have been removed. It might need more iteration to pull in all the outliers in place.

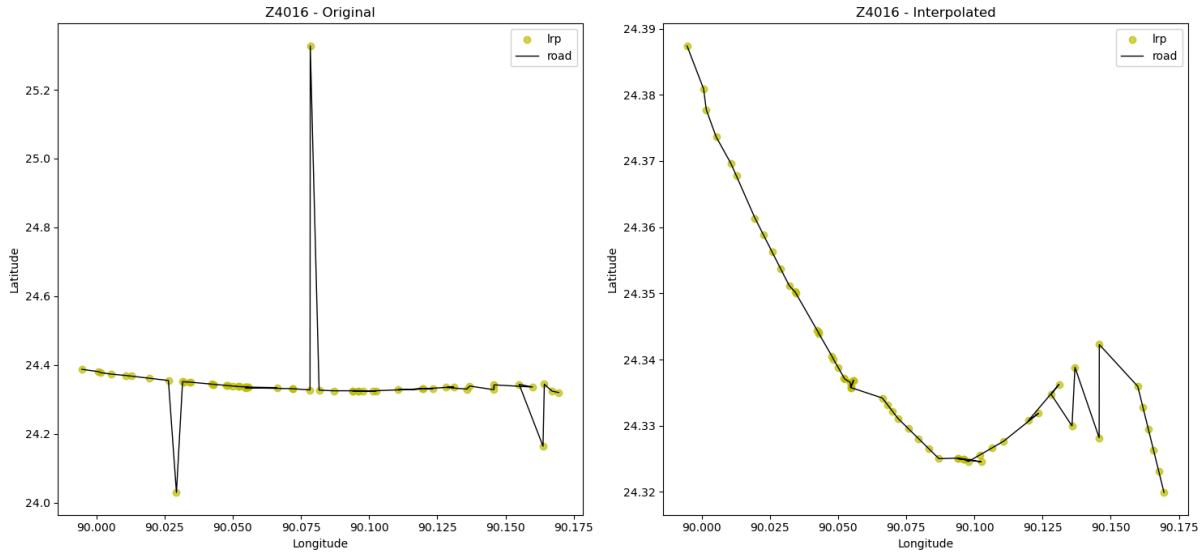

Figure 4.3: Road N806 before and after the first iteration (85th quantile)

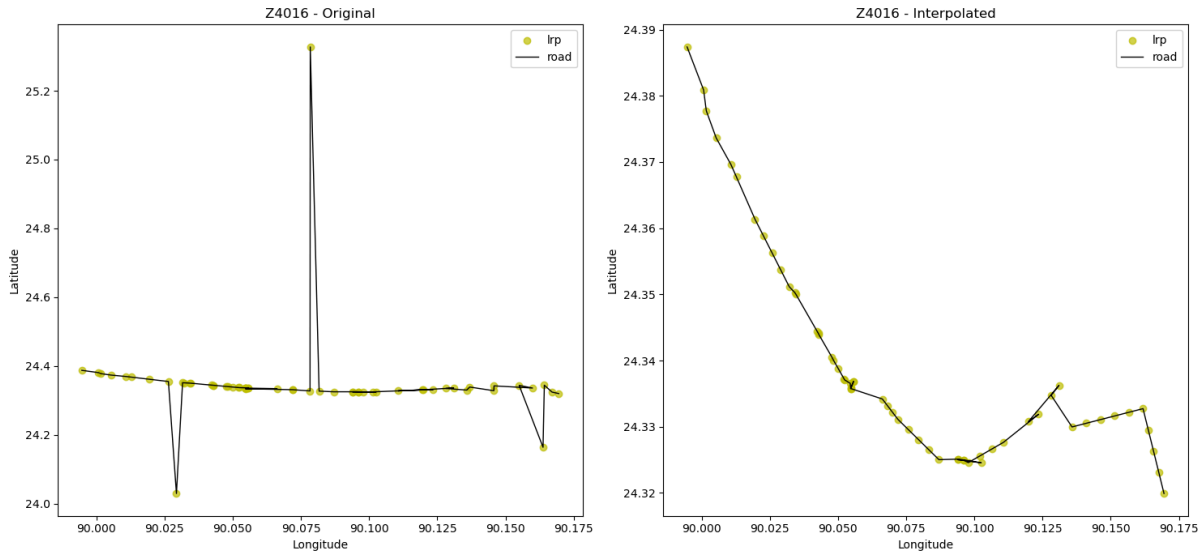

Figure 4.4: Road N806 original and Road N806 after the second iteration (95th quantile)

This algorithm presents two main limitations. First, we observed that LRPS and LRPE that are offroads have not been effectively addressed (see Figure 4.5). This is due to these edge outliers only having one neighbour point, but our interpolation algorithm requires two sandwiching data points to work. This should be resolved before running simulation in the next assignments. Second, while the two-iteration process is generally effective, in some cases, it is an overkill. When all LRP outliers are corrected in the first iteration, the second iteration unnecessarily alters the road shape, straightening sections (see Figures 4.6 and 4.7). To avoid this, the algorithm should ensure roads which outliers are addressed are not altered again by future runs.

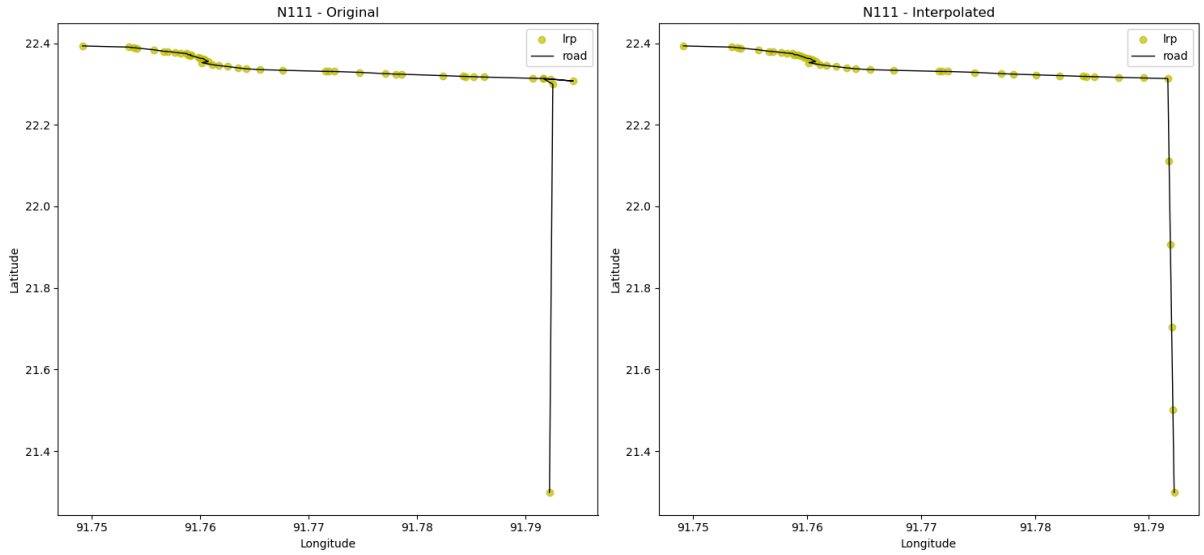

Figure 4.5: Road N111 before and after the second interpolation iteration

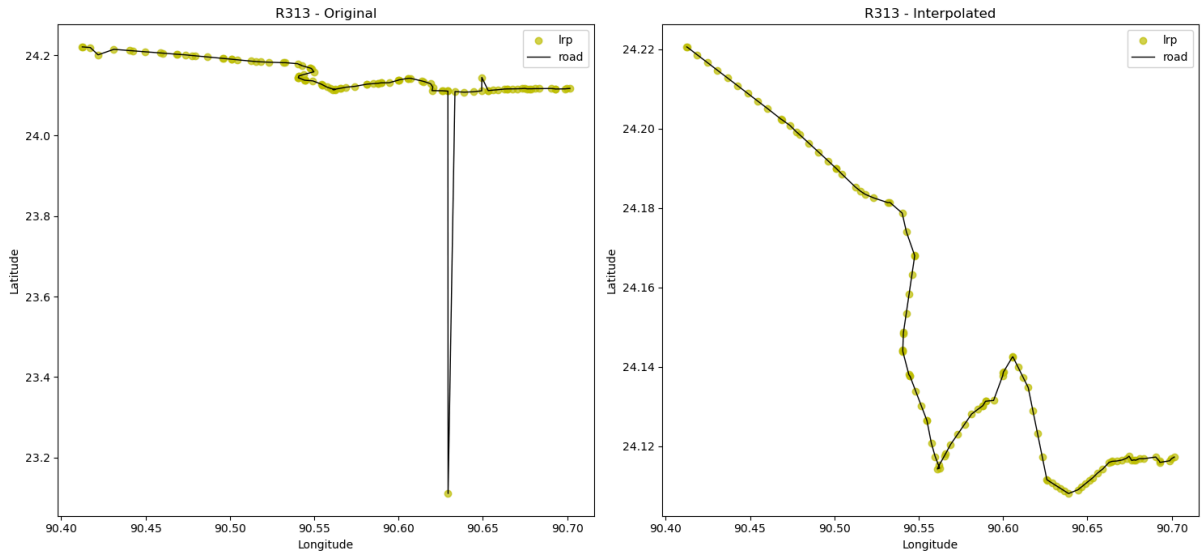

Figure 4.6: Road 313 before and after the first iteration (85th quantile)

These were some examples of cases that urged us to reassess our implementation. Currently, some roads have not been completely corrected. A more complete list of roads requiring additional iterations or different processing methods is provided in the Annex. Table 1 also lists the anomaly outliers.

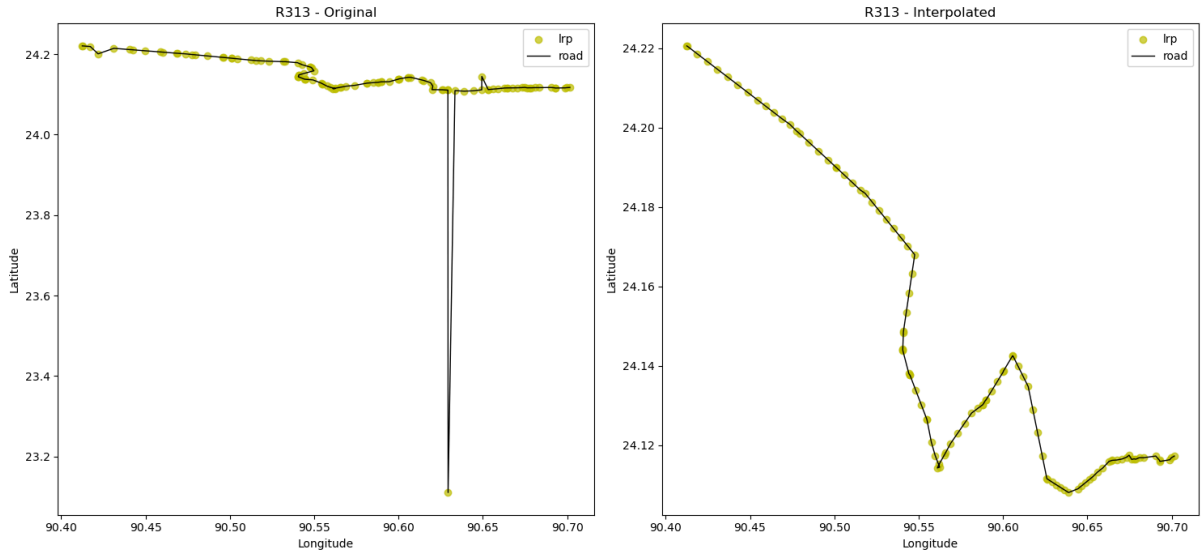

Figure 4.7: Road 313 before and after the second iteration (95th quantile)

4.2 CleanBridges.ipynb

4.2.1 Clean Bridge Data

This part of the code is to manage missing bridge location data, duplicate entries and bridge locations outside the country boundaries according to the Steps 2-4 in Chapter 3. The output of this algorithm is “infrastructure\cleaned\BMMS_overview_cleaned_prelim.xlsx”.

Algorithms:

Missing geographic data prevents the proper placement of bridges on the transport network. Duplicate bridge entries introduce inconsistencies that can lead to misconnections or redundant calculations in the simulation. Bridges located outside Bangladesh’s boundaries will be corrected. To resolve this:

- Load and scan the dataset to identify bridges with missing latitude or longitude values.
- Remove all bridges with missing coordinates, ensuring they do not interfere with the simulation.
- Detect the duplicate entries based on LRP names, keeping only the most complete and reliable record by checking the number of null values for columns for each row.
- Define the valid geographic range, and set the latitude range (20°N to 27°N) and longitude range (88°E to 93°E) based on Bangladesh’s boundaries.
- Detect the misplaced bridges by checking each bridge’s coordinates. Apply coordinate swapping and judge the results. If swapping the latitude and longitude places the bridge inside the valid range, update the dataset with the corrected values.

Result and analysis:

After cleaning the bridge data in the BMMS_overview.xlsx file, we observed several insights from the visualization results in Figure 4.3. The visualization of the cleaned bridge dataset shows an improved distribution of bridge conditions, with a higher proportion of green bridges, indicating better overall structural quality. However, it is challenging to accurately assess whether duplicated bridges are in good or poor condition, as the data was scraped from different web pages (bcs1 or bcs3). When filtering out duplicates, the method prioritized rows with fewer null values, which may

have inadvertently removed more bridges in poorer condition instead of preserving them. We should consider retaining bridges with the best available condition ratings, rather than solely focusing on those with fewer null values.

Despite the improvements made to the cleaned dataset, some misplaced bridges remain, as evidenced by points still appearing in the ocean. This suggests that the coordinate swap was insufficient to correct all errors. The inaccuracies may not always stem from a simple latitude-longitude swap. Some bridges may have fundamentally incorrect coordinates. We will work on enhancing the longitude and latitude in the next steps.

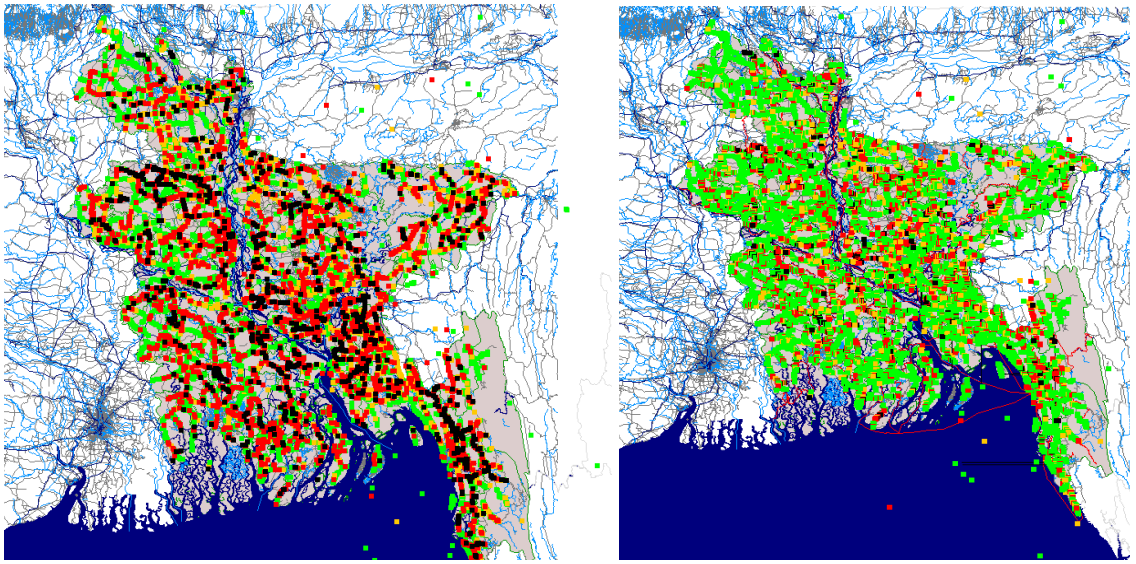

Figure 4.3: Original Visualization (left) vs. Clean Bridge Visualization (right)

4.2.2 Remove Bridges after LRPE

This part of the code is to remove all LRPs that come after LRPE as we assume that, in step 5 Chapter 3, these LRPs are not associated with any roads, so we want to remove them. The output of this algorithm is “infrastructure\cleaned\BMMS_overview_cleaned_bridges_after_LPRE_removed.xlsx”. And the list of bridges that are removed by this algorithm is in “infrastructure\cleaned\BMMS_overview_removed_lrps.xlsx”

Algorithms:

To systematically identify and remove these misplaced bridges, we implemented a data-driven filtering process based on Location Reference Points (LRPs).

- Compare the LRP names of bridges from the BMMS_overview.xlsx file with the LRP names of road points from the _roads.tsv file.
- If a bridge's LRP sequence exceeded the total number of LRP points for the corresponding road, it was assumed to be outside the valid boundary of that road.
- Bridges that failed this validation were removed to ensure they did not contribute to erroneous road connections in the simulation.

Result and analysis:

The process of removing bridges based on Location Reference Points (LRPs) was implemented to ensure that bridges located beyond the last valid road point (LRPE) were eliminated, as they were assumed to be misplaced and not connected to the road network. However, after

analyzing the results, two key issues were identified. First, as illustrated in Figure 4.4, some correctly placed bridges were mistakenly removed despite being visibly situated on the road. This suggests that some bridges may have had incorrect LRP names while retaining accurate latitude and longitude values, leading to their erroneous removal. Second, upon zooming into one affected area, as shown in Figure 4.5, we compared the dataset with Google Maps and discovered that some roads existed in reality but were absent from the dataset. Since the bridge removal algorithm relies on road data, the absence of a road from the dataset may result in the erroneous removal of associated bridges. These findings indicate that the current bridge removal logic may require adjustments to prevent the deletion of correctly placed bridges due to inaccuracies in road data or mislabeling of LRP names.

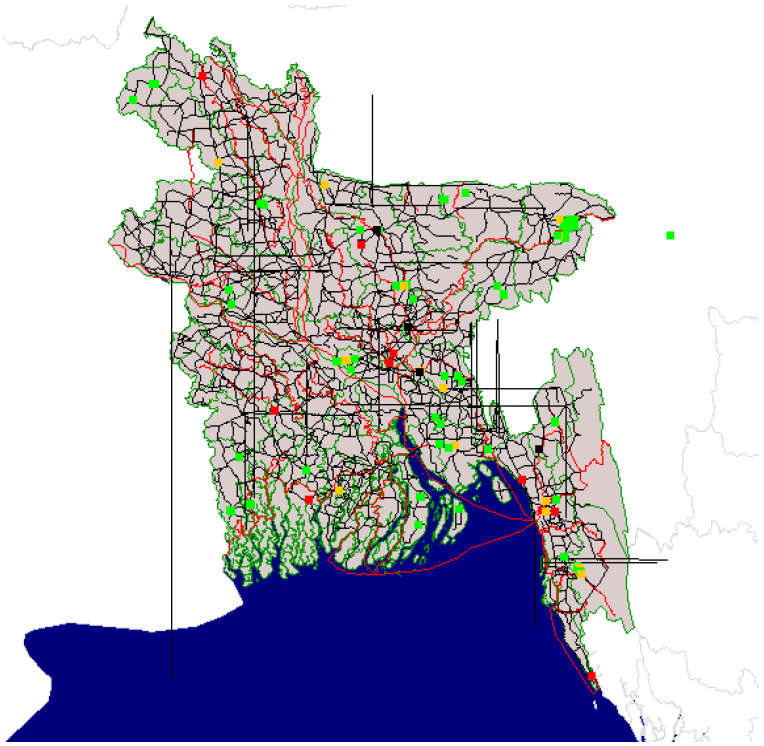

Figure 4.4: 230 bridges that are removed after applying the algorithm.

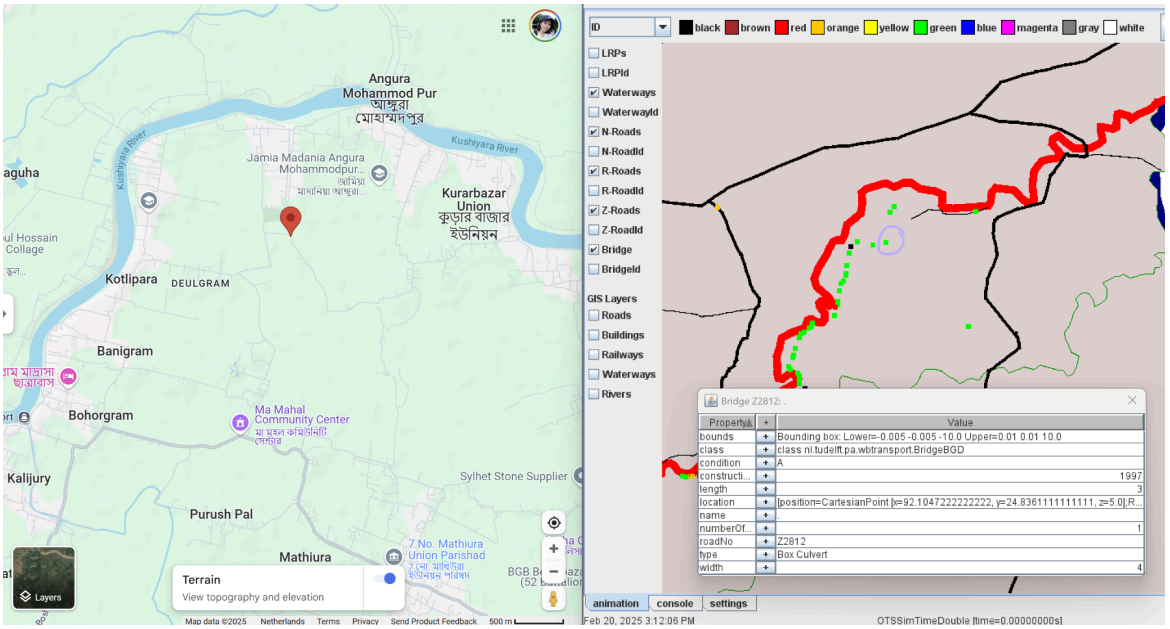

Figure 4.5: Zooming in, the algorithm could remove some non-road bridges near a waterway

4.2.3 Adjust Bridge Coordinates

This part of the code is to adjust the bridge coordinates as a consequence of interpolated road's LRPs we implemented in CleanRoads.ipynb. We want to ensure that all remaining bridges are on the roads if they should. The output of this algorithm is "infrastructure\cleaned\ **BMMS_overview.xlsx**" which is the one we use in the given Java visualization.

Algorithms:

- For each unique road in the bridge data:
  - Find all the rows in the road data that belong to the current road.
  - Create a list of coordinates (longitude and latitude) for each point on the road.
  - If there are no points for the current road, move on to the next road.
  - Create a structure (KDTree) to efficiently find the nearest points on the road.
  - For each bridge on the current road:
    - Get the coordinates (longitude and latitude) of the bridge.
    - Use the KDTree to find the closest point on the road to the bridge.
    - Get the coordinates of this closest point.
    - Update the bridge's coordinates in the adjusted bridge data to match the closest point on the road.

Result and analysis:

Figures 4.6 and 4.7 show the result after adjusting the bridge locations. We noticed that all bridges were pulled towards the nearest road points (see Figure 4.7).

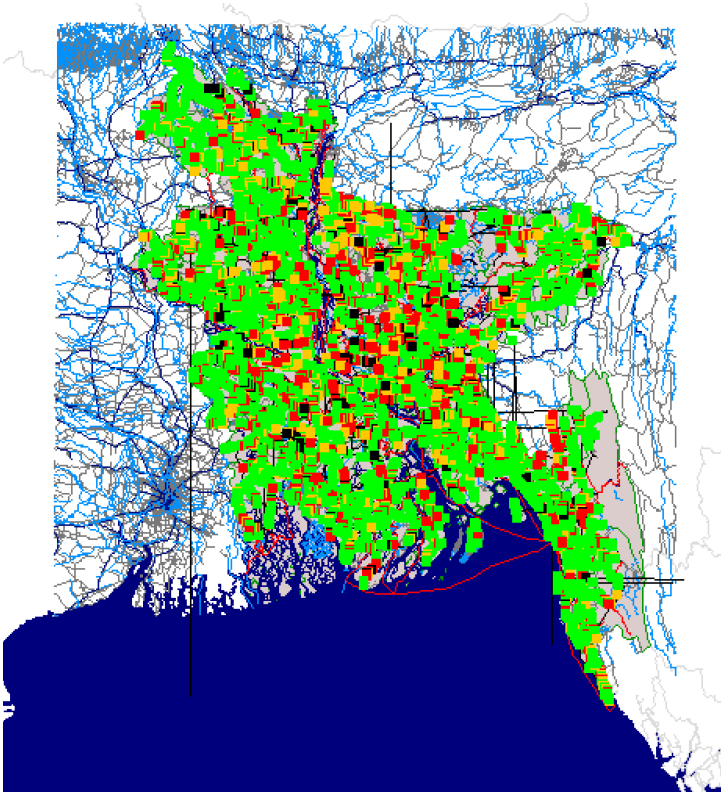

Figure 4.6: The final result of the bridge cleaning process

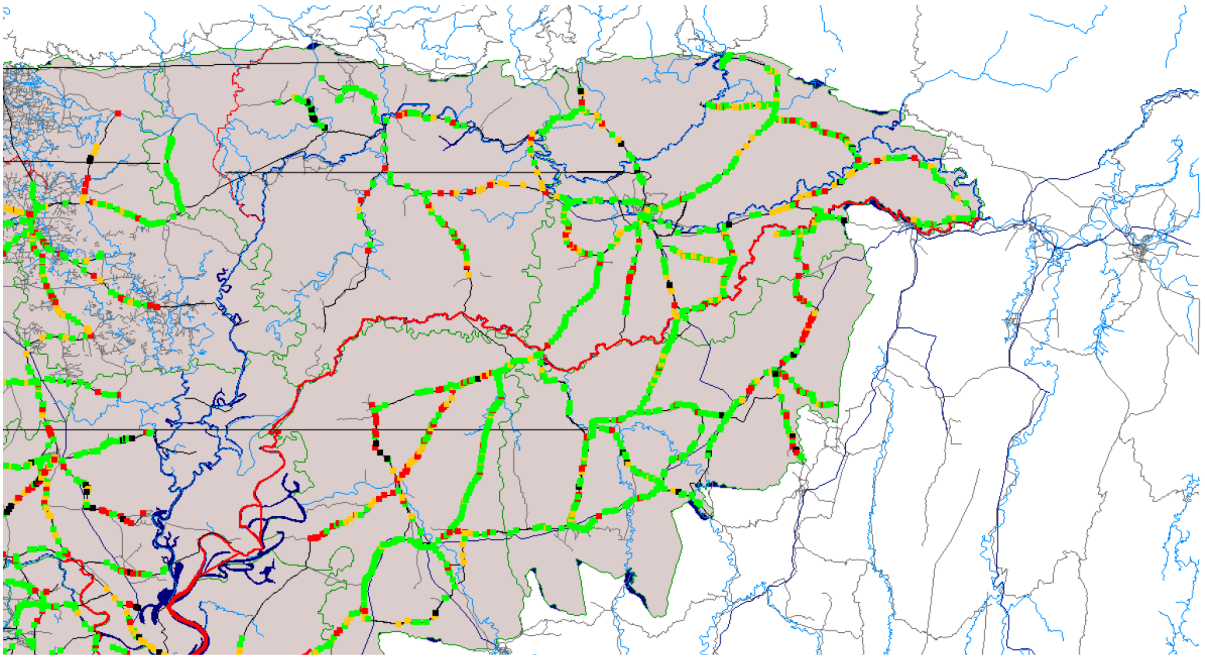

Figure 4.7: Zooming in, all bridges are put into the roads.

5. Reflection & Suggestions

One of the main challenges we faced was not collecting the dataset ourselves. This meant that before we could start working on data cleaning, we had to take time to fully understand how the data was structured and what each parameter represented. Without prior knowledge of how the data had been gathered or labelled, we had to rely on guidance from the professors or infer meanings from patterns within the dataset. This initial learning curve required significant effort before we could make informed decisions about data cleaning.

Time constraints added another layer of difficulty. Unlike researchers who may spend months working with a dataset, we had only two weeks to analyze, clean, and prepare the data for further use. A portion of this time had to be dedicated to understanding the real-world problem at hand and determining which files were most relevant to our goals. Given the size and complexity of the dataset, this meant making strategic choices about which aspects of the data required the most attention within the limited timeframe.

Collaborating in a large group naturally comes with its challenges. In the beginning, version management was difficult, as not everyone was familiar with it. Additionally, with five team members from diverse backgrounds, each with their own coding style and workflow, effective communication and coordination required effort. However, this diversity brought valuable perspectives, and once we found common ground, the experience became a great opportunity to strengthen our teamwork and adaptability.

Our current approach for the road data is sensitive to the quantile threshold, which may not consistently identify true geographic outliers or accurately gauge the required number of interpolation iterations. To improve our implementation, we could automate the iterative cleaning process by dynamically determining the number of necessary runs based on the density and distribution of errors within a given road segment. Additionally, incorporating spatial boundary delineation to define road edges before interpolating middle LRPs could resolve LRPS and LRPE outlier issues.

Besides statistical thresholds, we could also detect outliers by integrating machine learning techniques, such as clustering algorithms, to better classify different types of LRP anomalies. An unsupervised learning model could help distinguish between minor deviations and critical misplacements, allowing for more fitting corrections. However, black-box methods should be used carefully, as they can obscure the process, undermine our understanding, and reduce the ability to replicate the cleaning for other contexts, such as in a different country.

Using the haversine formula to calculate distances in kilometers also helps us think about outliers more logically than using Euclidean distance, as it reflects real-world spatial relationships better. Furthermore, after inspecting each cleaned road result, if there are extreme cases—such as more than 10 consecutive LRP outliers that can't be addressed with the current method (see Figure 5.1)—these cases should be isolated and dealt with individually to avoid distorting the rest road geometry.

Our approach to removing bridges with missing coordinates may have resulted in the loss of important data, particularly for assessing critical infrastructure investments. Many of the bridges that disappeared during cleaning were black, the most critical condition type, which could have been essential for the World Bank's evaluations. We could have considered the condition column when handling duplicates to improve this process, allowing us to flag high-priority bridges for further review instead of immediate removal. Additionally, instead of discarding entries with missing coordinates, we could have explored alternative methods, such as manually searching for location data or using other sources to estimate missing values.

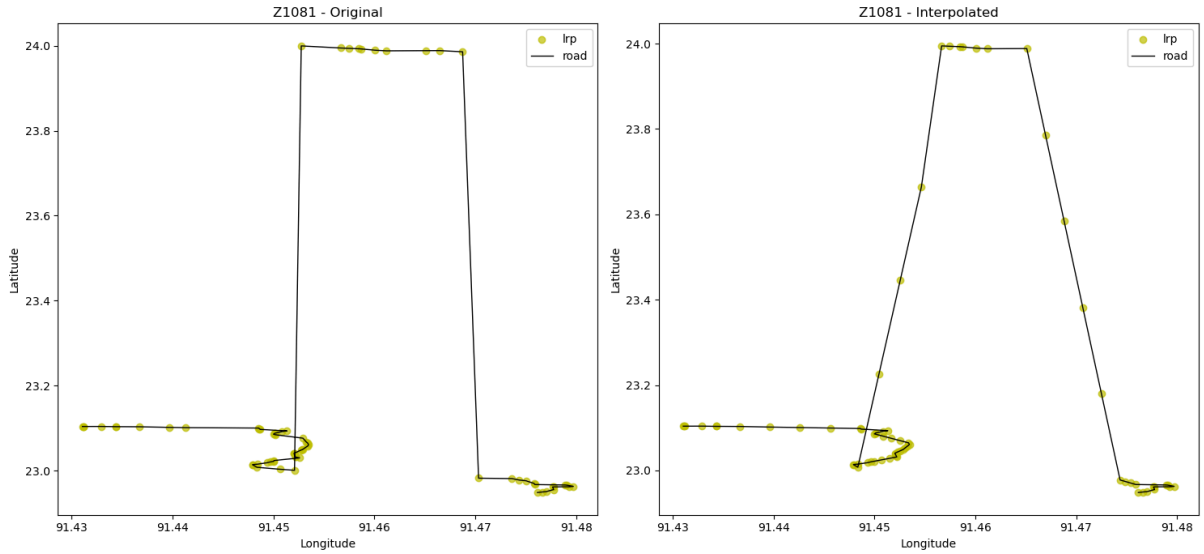

Figure 5.1: Road Z1081 has more than 10 consecutive outliers

References

High Commission of Bangladesh in Canberra. (n.d.). *About Bangladesh*. Government of the People's Republic of Bangladesh. https://canberra.mofa.gov.bd/bn/site/page/Bangladesh

Huang, Y.,(2013) Automated Simulation Model Generation, Delft University of Technology, pp.44-52

Annex

We tried to check each cleaned road and identify the remaining issues. This was done only once, so the list may not be thorough—some issues might be overlooked. Most of the remaining outliers are road edges (LRPS or LRPE).

Table 1: Outliers that have not been addressed by our method

Road Name	Issue to Consider
N101	Outlier LRPS
N111	Outlier LRPE
N211	Few data point stretched distance between LRPs
N406	Outlier LRPE
N6	Unusual road shape
N602	Unusual road shape
N707	Outlier LRPE
N806	Sensitive to the amount of iteration
R203	Outlier LRPS
R316	Only have one LRP (unusual outlier case)
R548	Sensitive to the amount of iteration
R822	Few data point stretched distance between LRPs
Z1005	Unusual road shape
Z1013	Sensitive to the amount of iteration
Z1030	Outlier LRPS
Z1038	Sensitive to the amount of iteration
Z1039	Sensitive to the amount of iteration
Z1047	Outlier LRPS
Z1081	More than 10 consecutive outliers (unusual outlier case)
Z1098	Some LRPs distances are stretched while the others are not
Z1124	Outlier LRPE
Z1129	Outlier LRPE

Road Name	Issue to Consider
Z1210	Outlier LRPS
Z1401	Unusual road shape
Z1421	Outlier LRPE
Z1437	Outlier LRPE
Z1439	Sensitive to the amount of iteration
Z1447	Outlier LRPE
Z1605	Outlier LRPS
Z1610	Some LRPs distances are stretched while the others are not
Z1611	Unusual road shape
Z1804	Some parts overlap (unusual outlier case)
Z1811	Some LRPs distances are stretched while the others are not
Z2403	Unusual road shape
Z2808	Unusual road shape
Z2834	Some LRPs distances are stretched while the others are not
Z3711	Outlier LRPS
Z4016	Sensitive to the amount of iteration
Z4606	Outlier LRPS
Z5019	Outlier LRPS
Z5074	Outlier LRPS
Z5210	Outlier LRPE
Z5210	Outlier LRPE
Z6813	Outlier LRPE
Z7452	Outlier LRPE
Z7706	Outlier LRPE
Z7717	Outlier LRPS
Z8604	Outlier LRPS

Acknowledgement

The use of AI

Use of AI in Code Development: In this assignment, AI tools such as GitHub Copilot and ChatGPT were used to improve code quality and generate ideas for implementation. These tools assist in structuring code, suggesting optimizations, and debugging issues. However, AI's role was strictly supportive—we conducted final decisions, implementations, and refinements to ensure accuracy and adherence to project requirements.

Use of AI in Writing and Editing: ChatGPT and Grammarly were used for text improvement, helping refine clarity, coherence, and grammatical correctness. It assisted in brainstorming alternative phrasings and ensuring correct writing standards. However, all AI-generated suggestions were critically reviewed, modified when necessary, and integrated into our work only after careful validation. This approach ensured that AI served as a tool rather than a replacement for our original analysis and insights.

Contribution of each member

Member	Contribution
Rachel Delvin Sutiono	- Ideas for cleaning strategies - Implementation of algorithms - Report writing
Celia Martínez Sillero	- Ideas for cleaning strategies - Implementation of algorithms - Report writing
Daniela Ríos Mora	- Ideas for cleaning strategies - Report writing - Git management
Thunchanok Phutthaphaiboon	- Ideas for cleaning strategies - Implementation of algorithms - Report writing
Yao Wang	- Ideas for cleaning strategies - Implementation of algorithms - Report writing