ASHESI UNIVERSITY

KWANALYTICS: A GEOGRAPHIC INFORMATION SYSTEM FOR CROWDSOURCING AND AGGREGATING ROAD SURFACE QUALITY INFORMATION FROM SMARTPHONES

UNDERGRADUATE THESIS

B.Sc. Computer Science

Kevin Kwesi Kafui de Youngster

2020

ASHESI UNIVERSITY

KWANALYTICS: A GEOGRAPHIC INFORMATION SYSTEM FOR CROWDSOURCING AND AGGREGATING ROAD SURFACE QUALITY INFORMATION FROM SMARTPHONE

UNDERGRADUATE THESIS

Undergraduate Thesis submitted to the Department of Computer Science, Ashesi University in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Science.

Kevin Kwesi Kafui de Youngster

2020

DECLARATION

I hereby declare that this Undergraduate Thesis is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature……

Candidate's Name:……

Date:……..

I hereby declare that preparation and presentation of this Undergraduate Thesis were supervised in accordance with the guidelines on supervision of Undergraduate Thesis laid down by Ashesi University.

Supervisor's Signature: ……..

Supervisor's Name:……..

Date:……

Acknowledgements

Firstly, I express my deepest gratitude to my supervisor, Dr Ayorkor Korsah, who, despite having so many responsibilities (especially during this pandemic), supported and guided me throughout the entire project. Her keen insights and level-headedness when communicating with me helped in times of despair, and for that, I am eternally grateful.

I also thank my parents and younger siblings for providing physical and emotional support to help me finish this project as we were all staying in due to the COVID-19 pandemic.

I also give special shout-outs to my colleagues, who supported me by offering to listen and pretending to understand as I rambled on my many thoughts and theories—shout-outs to Oracking Amenreynolds, Marcus Nartey, Kuukua Bentil and others.

Finally, many thanks to the lecturers including Abdul Wasay who offered advice that helped me bring this project to fruition, and the Computer Science Department of Ashesi University, for providing a robust structured curriculum that supported my growth in the understanding the field that is Computer Science.

Abstract

Road surface quality information is critical to road users when navigating road networks, and road authorities when making decisions on road infrastructure. However, not many systems exist that readily provide this information. This work connects prior work and proposes a geographic information system for crowdsourcing and aggregating probabilistic road surface quality information collected from users' smartphones from various times and at different geographic locations.

Table of Contents

Chapter 1: Introduction

Drivers, riders, and pedestrians consider multiple factors when navigating road networks. Some of these factors are distance, traffic, and the quality of road surfaces [12]. There exist mapping services, such as Google Maps and Waze, that provide information on routing, distance, and traffic, to make navigation more convenient. However, not many exist that comprehensively show the surface quality of roads, which is useful to users during navigation [34].

Research suggests a strong correlation between the surface qualities of roads and road accidents, traffic, and travel times [14,24]. It is, therefore, crucial for governments and road authorities to obtain relevant road surface quality information to make data-driven decisions on road infrastructure. This task of acquiring such information at scale is, however, challenging, costly and labour-intensive because it currently requires the use of specialized equipment [30].

With the rapid increase in usage and ownership of smartphones, especially in emerging countries [27], crowdsourcing techniques [23] can be used to obtain road surface quality data from people's smartphones. By utilizing readings from smartphone sensors such as the linear accelerometer, gyroscope, and GPS, it is possible to classify the surface quality of road segments [13]. The road surface quality information from classification can then be displayed on a map interface to be used by drivers when navigating, and by institutions such as governments and road authorities when planning transport infrastructure maintenance.

In this work, I propose Kwanalytics, a system for crowdsourcing, aggregating, storing and visualizing road surface quality information obtained from smartphone sensors. To effectively create this system, the following research questions are explored:

1. How to aggregate classification information (in the form of probability distributions) over the dimensions of time and geographical space?
2. How to store and retrieve data over segments of roads?

1.1 Prior Work

This work builds upon some prior work at Ashesi University that focus on methods of classifying road surface quality information from sensor readings, user research to determine the usefulness of road surface quality information, and approaches to collect sensor information in the first place. They are as follows:

Vorgbe [36] implemented a classifier (using logistic regression) that took input from smartphone sensors such as linear accelerometer and gyroscope readings to classify road surfaces with labels {good, fair,bad}. The classifier was able to distinguish between good and bad roads with a true positive rate of 92%, good and fair roads with 83% accuracy but was unable to differentiate between fair and bad roads.

Doku [12] conducted user experience research on whether the embedding of road surface quality information into a map interface such as Google Maps would be useful to users and how best to visualize such information. It was concluded that embedding surface quality was helpful and that a colour-coded visual might be sufficient.

Abeo [1] built up on issues raised in Vorgbe's [36] work by evaluating five classification algorithms to determine which would best classify accelerometer and

gyroscope readings. It was concluded that for the data tested on, the best performing classifier was the decision-tree based classifier with an overall accuracy rate of 92%.

Boohene [9] then implemented an application to collect sensor readings from smartphones to a backend data store and prototyped a visualization of the collected data on a map interface to aid road users in navigation.

The main objective of this research is to build upon Boohene's work [9] by proposing an alternative approach to store collected road surface quality information and filling in the gaps not addressed by his work (namely crowdsourcing and aggregation). This work also connects prior work to form an end-to-end system for crowdsourcing road and visualizing road surface quality information.

In summary, the main contributions of this work are as follows:

1. A pipeline architecture design for crowdsourcing, aggregating and storing road surface quality information.
2. A method to aggregate probability distributions over the dimension of time.
3. A geospatial grid-based approach to aggregating information over road segments and geometries.

Chapter 2: Background and Related Work

2.1 Classifying Road Surface Quality from Smartphone Sensor Readings

This project works on the assumption that road surface quality information can be obtained from smartphone sensor readings using a classifier [13]. Many research papers have explored approaches to classifying sensor data from smartphones to obtain road surface quality information. Some used threshold-based classifiers [31], some used machine learning methods such as K-means clustering [1] and Support Vector Machines [7,25] others used signal processing techniques [3]. The various approaches have strengths and weaknesses depending on factors such as frequency of recording, the speed of the vehicle, and others, as outlined in Sattar et al.'s survey [30]. Most of these projects focus on testing the feasibility of their proposed classification approach and not necessarily how readings from multiple users can be crowdsourced.

2.2 Crowdsourcing Road Surface Quality Information

Crowdsourcing involves obtaining information from large numbers of people using technology. By leveraging sensors in people's smartphones, it is possible to obtain sensor information on a large-scale at lesser costs compared to traditional approaches (for example, using sensor networks). According to Kanhere [23], this sensing can be categorized as people-centric (collecting data about the user) or environment-centric (collecting data about the user's surroundings). This project falls in the second category, where road surface quality information is obtained from people's smartphones. To the best of my knowledge, there is only one work that crowdsources road surface quality information using environment-centric sensing: *SmartRoadSense* by Alessandroni et al. [3].

Alessandroni et al. [3] proposed a Geographic Information System called 'SmartRoadSense' that crowdsources road surface quality information from smartphones. They collected accelerometer readings from smartphones, classified them with a mathematical model [18] to obtain an ordinal 'roughness index' describing the quality of the road surface. They then stored the information in a spatial database (PostgreSQL with PostGIS extension) and visualized the information using a mapping service. In their approach, they aggregated the collected road surface quality information by computing the arithmetic average of the roughness indices for each segment of a road.

Freschi et al. [26] built upon Alessandroni et al. [3]'s system to enable it to scale effectively. They acknowledged that such a system, collecting massive amounts of data, may have storage overhead as many data points have to be processed. They addressed this issue by aggregating the collected sensor data spatially and temporally. To aggregate sensor readings spatially, they sampled 'centroid points' for each road segment and averaged all roughness indices within a specific radius of the centroid points. To temporally aggregate the roughness indices, they used a weighted average function to give more weight to recent observations.

Sattar et al. [30] in reviewing the various road surface quality work claimed that "[the] best approach to crowdsourcing road surface anomalies from multiple sources would be a probabilistic and spatiotemporal-based approach that would overcome both the uncertainty and variability in road surface anomalies". State-of-the-art approaches acknowledged that indeed variance in the types of devices and vehicles used affected the accuracy of their classifiers. This paper attempt to address the issue Sattar et al. [30] raised by proposing a system that works with probabilistic classification information (obtained from a multi-class classifier) and aggregates it with spatio-temporal aggregation methods.

2.3 Data Aggregation

Since crowdsourcing requires collecting massive amounts of data, there is a need for aggregation techniques to summarize such data to reduce the storage and computation overhead in a way that preserves information. This system involves summarizing road surface quality information (in the form of probability distributions) over the dimensions of time and space. The following subsections give an overview of probability, spatial and temporal aggregation methods.

2.3.1 Probability Aggregation

The problem of combining ('pooling') probabilistic information ('opinions') from different individuals is defined by Dietrich et al. [11] as the *opinion pooling problem.* It involves applying some function ('pooling method') to a collection of probability distributions to obtain a single aggregate distribution.

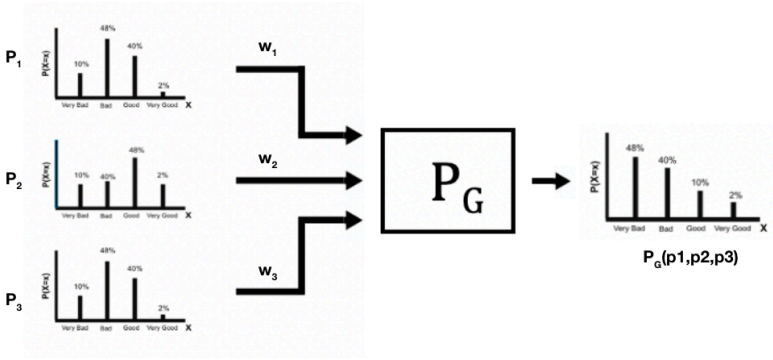

Figure 2.1: The Opinion Pooling Problem. P_1, P_2, and P_3 are combined by a pooling method P_G to obtain an aggregate distribution PG (P_1, P_2, P_3)

Pooling Methods

A pooling method is a function that combines multiple probabilities to obtain an approximation of their 'true' combination. In their reviews, Allard et al. [4] and Genest et al. [17] showed some pooling methods and explained that the choice of method depended on its application and desired properties. They also categorized most pooling methods based on how information was combined: additive or multiplicative.

Additive Methods

Additive methods express the aggregate of probabilities as the disjunction (union) of the constituent probabilities using addition. Methods include the most commonly used linear pooling (weighted arithmetic average) [5] and the beta-transformed arithmetic average [28].

$$P_G(E) = \sum_{i=1}^{n} w_i P_i(E)$$

Linear Pooling

$$P_G(E) = H_{\alpha,\beta}\left(\sum_{i=1}^{n} w_i P_i(E)\right)$$

Beta-transformed Linear Pooling

Figure 2.2: Additive Pooling Methods

Multiplicative Methods

Multiplicative methods, on the other hand, express the aggregate of probabilities as the conjunction (intersection) of these probabilities using multiplication. These methods require normalizing the aggregate probability with a constant *c*, to ensure the output is a discrete probability distribution (also called Probability Mass Function – PMF). Methods

include geometric pooling (normalized weighted geometric average) [17], multiplicative pooling [11], and conflation (normalized weighted product) [21].

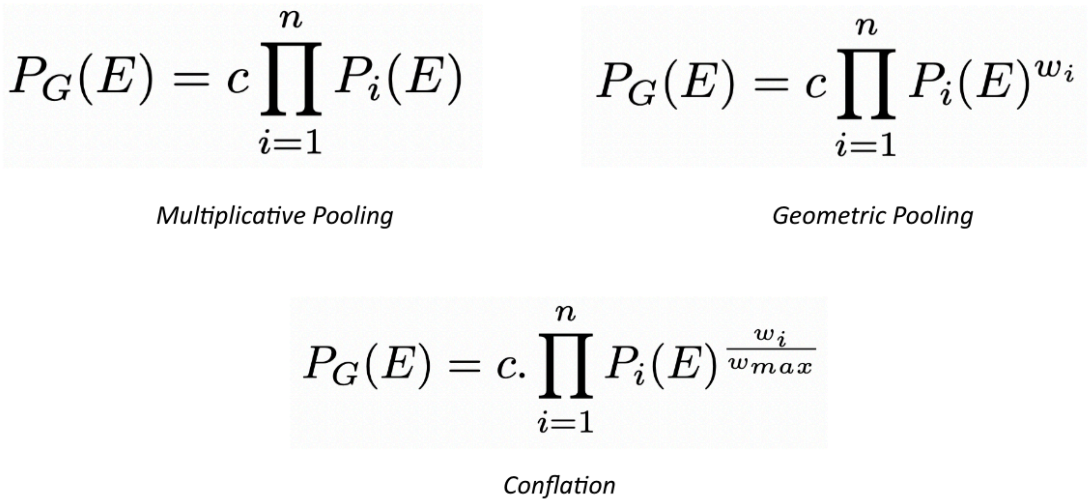

Figure 2.3: Multiplicative pooling methods

2.3.2 Temporal Aggregation

Temporal aggregation involves partitioning information into groups by a time granularity (for example, daily, monthly, or yearly), and applying a function on each group to obtain aggregates [16]. Systems that work with streaming information (sequences of continuously recorded data) often use the sliding window approach to summarize data [33]. It involves computing over only the N-most recent elements to answer queries where N is defined as the window size.

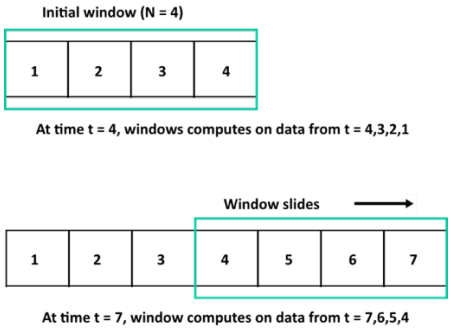

Figure 2.4: Illustration of Sliding Window

2.3.3 Spatial Aggregation

I have identified two main approaches to aggregating geospatial information regarding roads: aggregation based on road geometry and aggregation based on a grid index.

Aggregation based on Road Geometry

This involves aggregating information across road segments based on their geometry (shapes and coordinates). This approach requires prior information about road geometries and the connections between roads in a network. Roads are divided into segments by placing 'landmark points' on each road and computing aggregates for each landmark point.

Freschi et al. [15] used this approach to aggregate information for roads. They created landmark points (centroids) along a road geometry to divide it into segments. All observations that fell within a given radius of each centroid were aggregated and associated with that centroid.

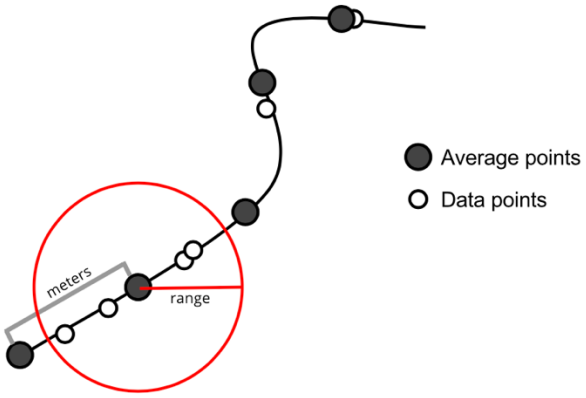

Figure 2.5: Freschi et al. [15]'s approach to spatial aggregation: placing centroids (average points) on a road segment

The limitations of this approach are that (i) it requires pre-processing a road network to determine where to place landmark points and (ii) it becomes more complicated when landmark points' locations are computed dynamically.

Aggregation based on a Grid Index

Another approach to aggregating spatial information over roads is to use a grid-based model (index) of the Earth, mapping portions of road segments to given cells and storing data for each cell. Grid indexes divide the Earth's surface into uniformly shaped cells to enable efficient aggregation of information. They can be classified into two forms: graticular and geodesic grid indexes.

Graticular Grid Indexes

These grid systems use the longitude and latitude lines (graticules) as a mesh around the Earth's surface to divide it into evenly spaced cells. They then use a geocoding

algorithm, such as GeoHash [32], to map GPS coordinates (latitude-longitude pairs) to the various cells in the grid.

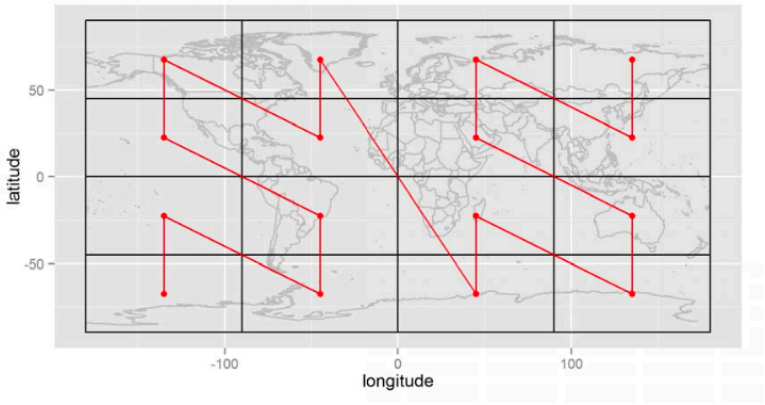

Figure 2.6: GeoHash-based approach divides the Earth with longitude and latitude lines into rectangular cells [32]. The red line indicates space-filling curve mapping each 2-D cell to a 1-D index.

A limitation of graticular grid indexes is the precision error in aggregation because cells are not uniformly shaped (due to the curvature of the Earth around the poles).

Geodesic Grid Indexes

Geodesic grid indexes [29], like graticular grid indexes, divide the Earth's surface into uniformly spaced cells. However, instead of using graticules to divide the Earth's surface, they project points on the Earth's surface unto a polyhedron and partition each face of the polyhedron into uniform grids. The use of projection overcomes the precision error from using graticules and results in uniformly shaped, uniformly sized and easily indexable grid cells. The shapes of the cells in a geodesic grid index may vary depending on the application: triangular, square, or hexagonal.

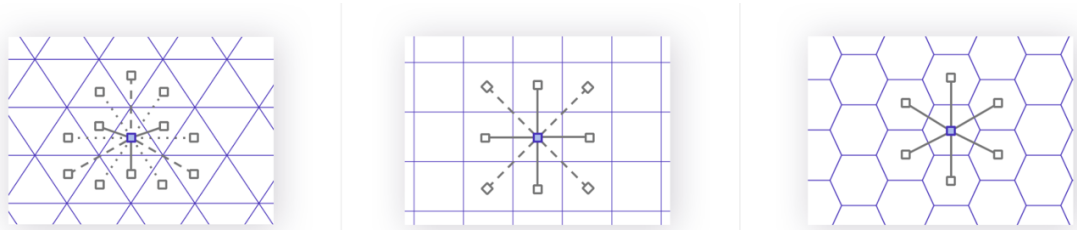

Figure 2.7: Possible cell shapes in a geodesic grid index [29]

Chapter 3: Methodology

This section introduces the architecture design of the proposed system and further overviews its various stages, showing the implementation of the stages to which this paper contributes.

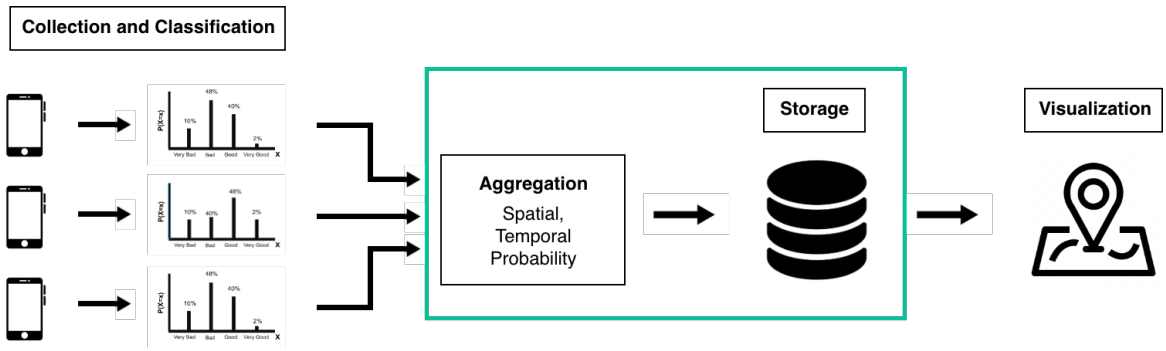

Figure 3.1: High-Level pipeline architecture of Kwanalytics, showing how contributions of this paper (highlighted in green) fit with the previous work done [1,9, 12, 36].

The proposed geographic information system (shown in figure 3.1) uses a pipeline architecture to tackle the problem of crowdsourcing road surface quality information because the processes involved occur in connected stages.

In summary, the full process of the pipeline is described as follows:

i. As vehicle navigates a road, collect sensor readings and GPS trail (**Collection**)

ii. Classify sensor readings to obtain surface quality information of the road segments on which the vehicle travels (**Classification**)

iii. Map the recorded GPS trail from (i) onto corresponding grid cells (**spatial aggregation**) and aggregate the classification information for each cell with that already associated with it (**probability and temporal aggregation**)

iv. Store the newly computed aggregate in the data store for all cells from (iii) (**Storage**)

v. Retrieve surface quality information (the aggregate) and visualize on map service (**visualization**)

3.1 Data Collection and Classification

Sensor readings and GPS trails are collected and validated from Android devices with an application built by Vorgbe [36]. The collected readings are then passed into a classifier [1, 36] which accepts a time series of sensor readings over a given segment and produces road surface quality information in the form of probability mass functions (PMFs). This output PMF gives the probability that a given road segment belongs to a given class *X* of road surface quality from a set of labels *{very bad, bad, good, very good}*.

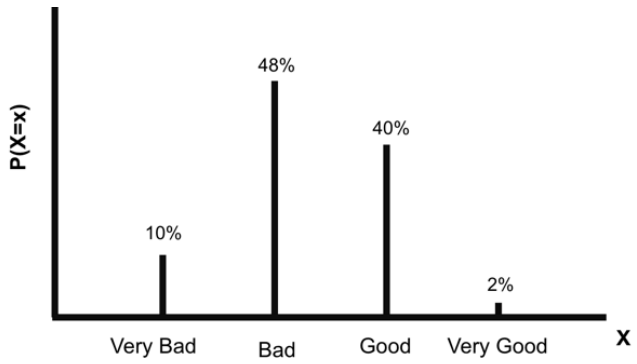

Figure 3.2: Sample output of classifier: A Probability Mass Function

x	Very Bad	Bad	Very Good	Good
P(X=x)	0.10	0.48	0.40	0.02

Figure 3.3: Hash table Representation of Probability Mass Function

The outputs from these stages are a GPS trail of a road segment (represented by a collection of latitude-longitude GPS coordinates), the corresponding surface quality information of that road segment (represented by a hash table with labels as keys and probabilities as values), and timestamp information (time of observation).

3.2 Spatial Aggregation

At this stage, the GPS trail representing a road segment (Figure 3.4) is divided into parts (Figure 3.5), and the surface quality information of the segment is associated with each part for further aggregation. Of the two approaches for spatial aggregation mentioned earlier in Chapter 2, the grid-based approach was chosen because it did not require pre-processing a road network. This makes it well suited for geographic regions where road networks undergo development and deterioration.

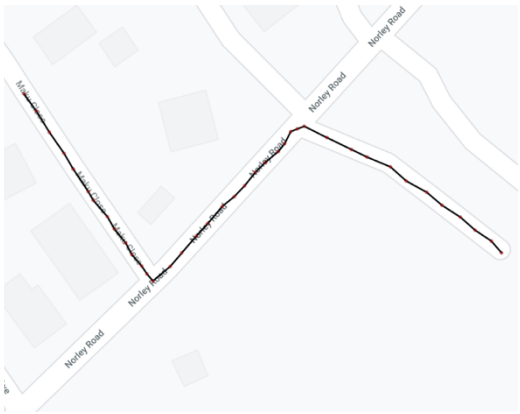

Figure 3.4: A trail of GPS coordinates T (black outline)

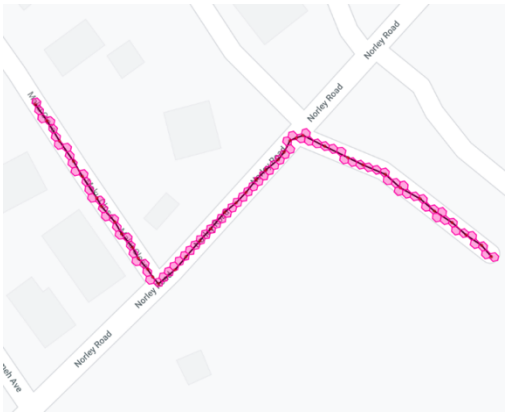

Figure 3.5: Spatially aggregated T into grid cells (pink)

Using a grid index raised two further questions or design choices:

i. What should the shape a unit grid cell be?

ii. What should the size of each unit grid cell be?

Shape of a unit Grid Cell

A hexagon-based grid system (Uber's H3 [10]) was chosen for aggregating information over road segments. Birch et al. [8] explained that the hexagonal and quadrilateral cell shapes were the most adequate for spatial aggregation and concluded that the choice of which to use depended on its application. For instance, the quadrilateral-

shaped grid cells can be recursively divided into smaller grid cells but have two types of neighbours (adjacent and diagonally separated cells). In contrast, the hexagonal-shaped grid cells are more compact and have one type of neighbour, making them more adequate for analysis involving movement across cells. Their stark differences, however, were irrelevant to the requirement of this project (cell indexing). Both were equally valid; hence the decision on which shape to use was based on the comparative performances of two available production-ready grid systems (the hexagonal grid system H3 [9] and the quadrilateral grid system S2 [19]). Experiment 4 in Chapter 4 provides more information on their performances.

Size of a unit Grid Cell

The size of a unit grid cell is crucial to the performance of the system. Using too large a cell might result in multiple roads covered by one grid cell (Figure 3.6) and using too small a cell might result in gaps and uncovered regions of a road (Figure 3.7). The choice size of a unit grid cell should ideally depend on the size of a road, but the various road standards make this problematic. Sizes and standards of roads vary by country [6] hence there may not be a one-size-fits-all.

Figure 3.6: Large grid cell size result in incorrectly representing road segments. In this scenario, one grid cell covers n > 2 different road segments

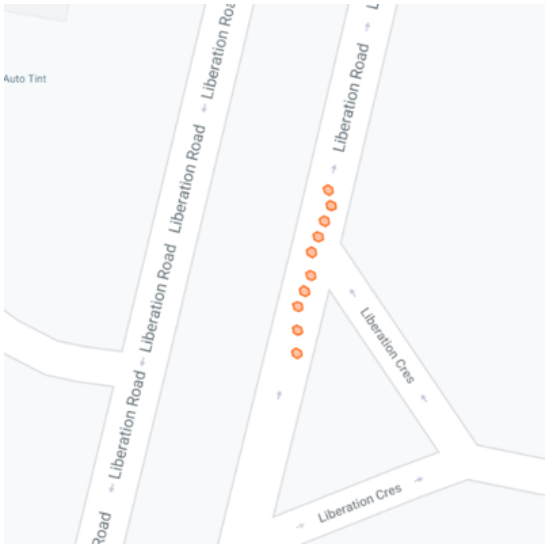

Figure 3.7: Small grid cell sizes result in large 'uncovered' regions on road segments. In this scenario, grid cells barely cover the one road segment.

The revised decision, therefore, was to pick a 'good enough' size to minimize the errors shown above. After consulting a few highway standards reports [2, 35], the chosen cell size (cell edge length) was 3.65 m (the minimum lane width of roads).

Though this choice prevents the scenario of a grid cell covering multiple road segments, it is still susceptible to leaving uncovered regions of road segments. Map matching [20], mapping 'raw' GPS trails to a road network (in this case Google Maps), was used to deal with this problem. Experiment 5 in Chapter 4 demonstrates the efficacy of this approach.

In summary, the spatial aggregation process, mapping road segments to corresponding grid cells, is as follows:

i. Obtain GPS trail over road segment

ii. Map match the trail to obtain a consistent polyline (using Google Maps API)

iii. Map the polyline to grid cells (using H3 grid system and an interpolation algorithm)

For (iii), this paper introduces an algorithm that maps polylines to grid cells by stepping incrementally along the input line segment over fixed intervals and maps each new point to a grid cell. Figure 3.8 and Figure 3.9 below visualize and outline the algorithm for this process.

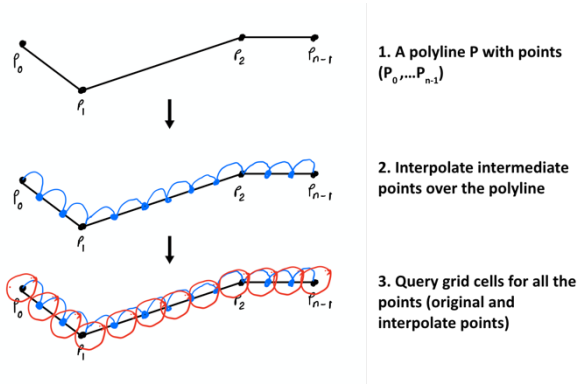

Figure 3.8: Mapping a polyline into grid cells. Grid cells are represented as red circles for easy illustration

Algorithm 1: Mapping a polyline to a corresponding set of grid cells

```
function PolylineToGridCells(P, d);
  Input  : Array P of GPS coordinates representing a polyline
           Integer d representing the step distance
  Output: A set S of grid cells
begin
    S ← set();
    l ← length of the polyline P (in units);
    current_distance ← d;
    first_cell ← query grid cell for point P0;
    S.add(first_cell);
    while current_distance < l do
        next_point ← find point that is current_distance along the
          polyline P;
        next_grid_cell ← query grid cell for next_point;
        S.add(next_grid_cell);
        current_distance ← current_distance + d;
    end
    last_cell ← query grid cell for point Pn−1;
    S.add(last_cell);
    return S
end
```

Figure 3.9: Algorithm for mapping road segments (represented as polylines) to grid cells

Runtime Complexity

The runtime complexity of the algorithm in Figure 3.9 is $O(\|P\|)$ where $\|P\|$ is the distance (in units) of the polyline P. It is more specifically $O(\|P\|/d)$ where d is the step distance.

3.3 Probability Aggregation

After spatial aggregation, the road surface quality information for each grid cell is aggregated with existing information. This section details how the chosen pooling method was selected based on its properties and the system's requirements. It further details how the method is implemented to suit the system.

The appropriate pooling method should have the following properties:

- **Weighted**: It should support weights to enable weighting observations differently.
- **Has no hyperparameters**: It should not require tuning or calibration to be useful as there is no available training data for aggregation.
- **Epistemically Valid**: It should produce approximations close to the true combination of probabilities. According to Dietrich et al. [11], an epistemically valid pooling method should depend 'primarily on the opinions of the more competent observations' as opposed to giving equal weight to each observation.
- **Commutative:** The output aggregate PMF of the method should not depend on the order in which PMFs are pooled [22]. That is, P_G (p1, p2, p3) = P_G (p2, p1, p3) [28].
- **Works with asymmetric information**: It should work with input PMFs based off different information [11] (in the system's case, smartphone sensor readings are assumed to be variant and dependent on various factors such as the type of vehicle and the quality of sensors on the phone).

- **Iterative**: Because aggregation in the system is done on a 'rolling' basis (only one aggregate is stored, and inputs are discarded), the pooling method must be iterative to support updating an aggregate with new information in a consistent manner. That is $P_G(p1, p2, p3) = P_G(P_G(p1, p2), p3)$.

Table 3.1: Comparison of pooling methods with associated relevant properties

	Linear	**Beta-Transformed Linear**	**Geometric**	**Multiplicative**	**Conflation**
Weighted	✓	✓	✓		✓
Epistemically Valid			✓	✓	✓
No Hyperparameters	✓		✓	✓	✓
Commutative	✓	✓	✓	✓	✓
Work with Asymmetric Information				✓	✓
Iterative					✓

Various literature [4,11,21–23] was consulted to obtain the properties of various pooling methods mentioned in Chapter 2. These pooling methods were then compared based on their properties (as shown in Table 3.1). Based on the system requirements, conflation, a normalized weighted product of the PMFs, was chosen. In addition to having the properties detailed in (Table 3.1), it also minimizes the loss of Shannon information and does not require weights to sum up to one since it uses relative weights [21].

Mathematical Formula for Conflation

The aggregate probability, $PG(E)$ of each outcome $E \in \{E_1, E_2 \ldots E_k\}$ across all n PMFs: $[P_1, P_2 \ldots P_n]$ with associated weights in W: $[w_1, w_2 \ldots w_n]$ is defined as:

$$P_G(E) = c. \prod_{i=1}^{n} P_i(E)^{\frac{w_i}{w_{max}}}$$

Figure 3.10: Formula for Conflation

Where *c*, the normalization constant, is the inverse of the sum of values of each outcome post-aggregation.

$$c = \frac{1}{\sum_{j=1}^{k} P_G(E_j)}$$

Figure 3.11: Normalization constant

Algorithm for Aggregating PMF (in Records) using the Conflation Method

The conflation method was implemented in Python 3. It accepts input PMFs (represented as an array of hash tables) and their corresponding weights (represented as an array of floats) and outputs an aggregate PMF.

Algorithm 2: Conflation: For finding the aggregate of weighted PMFs

```
function conflate (P, W);
  Input  : Array P of PMFs [p_0, p_1...p_{n-1}]
           Array W of weights [w_0, w_1...w_{n-1}]
  Output: Aggregate PMF P_G
begin
    P_G ← PMF();
    W_max ← max(W);
    E ← p_0.keys();
    foreach outcome e_j ∈ E do
        product ← 1;
        foreach pmf p_i ∈ P do
            product ← product × p_i[e_j]^(w_i / W_max);
        end
        P_G[e_j] ← product;
    end
    c ← 1/sum(P_G.values()) ;
    foreach outcome e_j ∈ E do
        P_G[e_j] ← c × P_G[e_j]
    end
    return P_G
end
```

Figure 3.12 Algorithm for aggregating road surface quality information (PMFs)

Runtime Complexity

The runtime complexity of the algorithm in Figure 3.12 is *O(n)*. It is more specifically *O(n.j)* where *n* is the number of PMFs to be aggregated, and *j* is the number of outcomes across all PMFs (a constant, 4 labels for each class of road surface quality).

Dealing with Zero-values.

Since conflation is based on the multiplication of probabilities, if one PMF has a zero value for a given outcome, the aggregate will continue to be zero. This property is defined by Allard et al. [4] as '0/1 enforcing' and was overcome by working within a range [0.001, 0.999].

3.4 Temporal Aggregation

For later temporal analysis of road surface quality information, the system uses a time granularity of a day. For each road segment corresponding to a grid cell, the system keeps daily aggregates (aggregate of all PMFs observed during a day).

Further, the sliding window approach is used to temporally aggregate observations (PMFs) because this system works with streaming information. A window size of 365 days was chosen to connote "considering observations made over the past year (365 days), what is the surface quality of a road segment?". Therefore, for each grid cell representing a portion of a road segment, daily aggregates as well as global aggregates are kept and regularly updated. Global aggregates represent the surface quality information of a road segment at any time t, and daily aggregates represent the surface quality of a road segment at any day over the past 365 days.

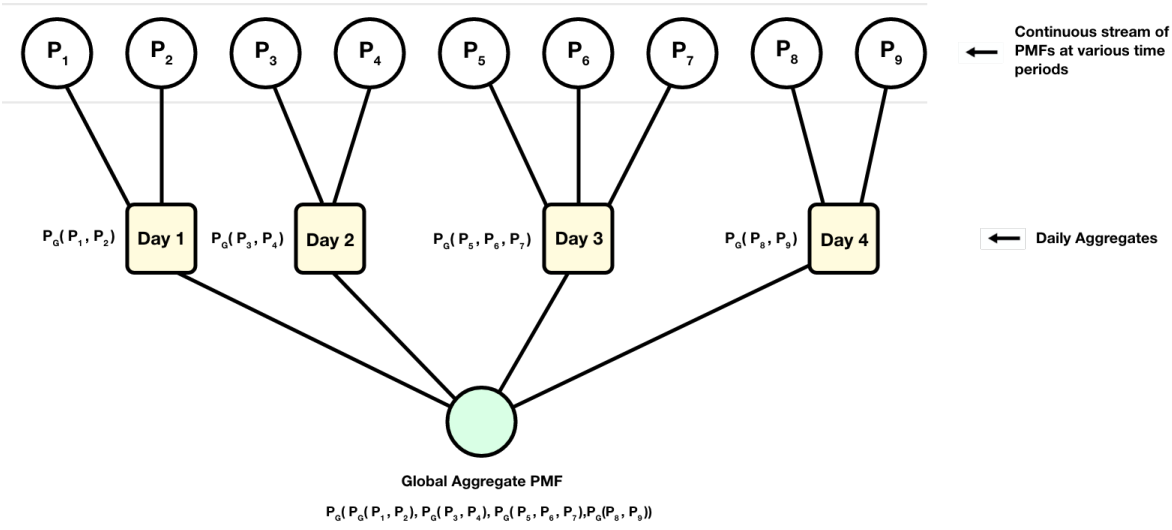

Figure 3.13: Temporal aggregation: for each road segment, the continuous data are aggregated by day, and each day's aggregate are further aggregated to obtain a global aggregate that represents the surface quality of that road segment.

Weighting Observations by Time

A requirement for this system is to consider recent observations (PMFs) more than older observations. This was done by weighting observations according to how long ago they were observed. An exponential decay function was used to weigh observations because of its horizontal asymptote. As an observation gets older, its corresponding weight approaches zero. The weighting function defines the weight *w* of a PMF observed at time t_i aggregated at time t_A as follows:

$$w(\Delta t) = \exp(\neg\frac{\Delta t}{T}), where\, \Delta t = t_A - t_i$$

Figure 3.14: Weighting function to weight observations according to their recency

Where *T,* the time constant, is the value of Δt that gives a weight of *0.368.* Since we are working with a sliding window of length 1 year (365 days), the value of T = 134 days (0.365 x 365 days).

To illustrate how weights are assigned to observations, consider an example. Suppose observations have been made on Day 1, Day 10 and Day 100 and aggregation is performed on Day 100, the weights of each observation are calculated as follows:

Table 3.2: Showing how weights are assigned to each daily aggregate when computing the global aggregate

	Day 10	Day 30	Day 100
t_i	10	30	100
t_A	100	100	100
Δt	90	70	0
w(Δt)	**0.51086915**	**0.59310249**	**1**

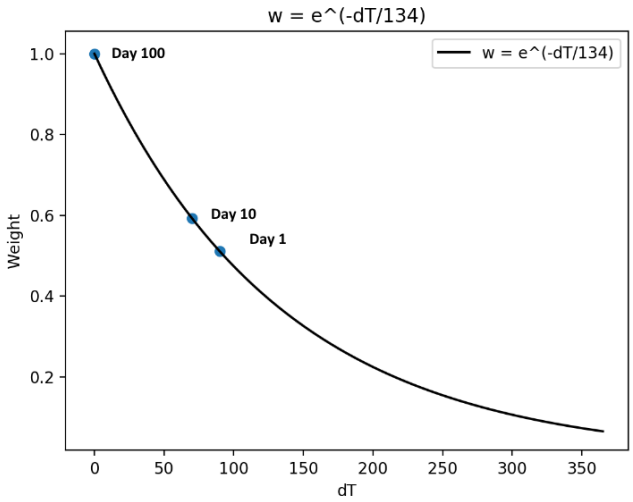

Figure 3.15: Graph representing w(t) the weight function and the corresponding weights of the three observations

Note: Observations made on the same day are aggregated with a uniform weight of 1 since Δt is 0 for those observations.

End-to-end Aggregation: Putting it all together

In the previous subsections, the various aspects of aggregation (spatial, probability and temporal) were detailed. This subsection attempts to put them all together in one procedure.

Given a GPS trail of coordinates, T representing a road segment, taken at DateTime t, with surface quality information (PMF) P, the aggregation process is as follows:

Algorithm 4: Aggregation

```
function aggregate (T, P, t)
Input  : Array T of GPS coordinates representing a trail
         PMF P representing the surface quality of the trail
         t timestamp of when the readings were recorded
Output:
begin
    Perform spatial aggregation:
    trail ← mapmatch(T)
    cells ← polylineToGridCells(line)
    foreach cell c ∈ cells do
        Update the daily aggregate with new observation P:
        d ← get daily aggregate for cell c based on time t
        W ← create array of weights for d and P
        d_new ← conflate([d, P], W)

        Recompute the global aggregate P_G:
        D ← get array of all daily aggregates for cell c
        W_D ← create array of weights for each daily aggregate in D
        P_G ← conflate(D, W_D)

        Update the road surface quality label for c to:
        label ← arg max_E P_G(E)
    end
end
```

Figure 3.16: Algorithm of the entire aggregation process

Runtime Complexity

The asymptotic runtime complexity of the aggregation process is *O(c)*, where *c* is the number of grid cells obtained from the spatial aggregation of the GPS trail.

3.5 Data Storage

A document-based datastore, MongoDB, is used to store surface quality information over road segments (grid cells in the grid index). Each document represents associated information of each grid cell and consists:

i. The grid cell's ID defined by the grid index system used for querying

ii. The global aggregate PMF

iii. A timestamp for the global aggregate PMF

iv. Label for the surface quality information

v. A collection of PMF-Timestamp pairs representing daily aggregates and the times they were recorded

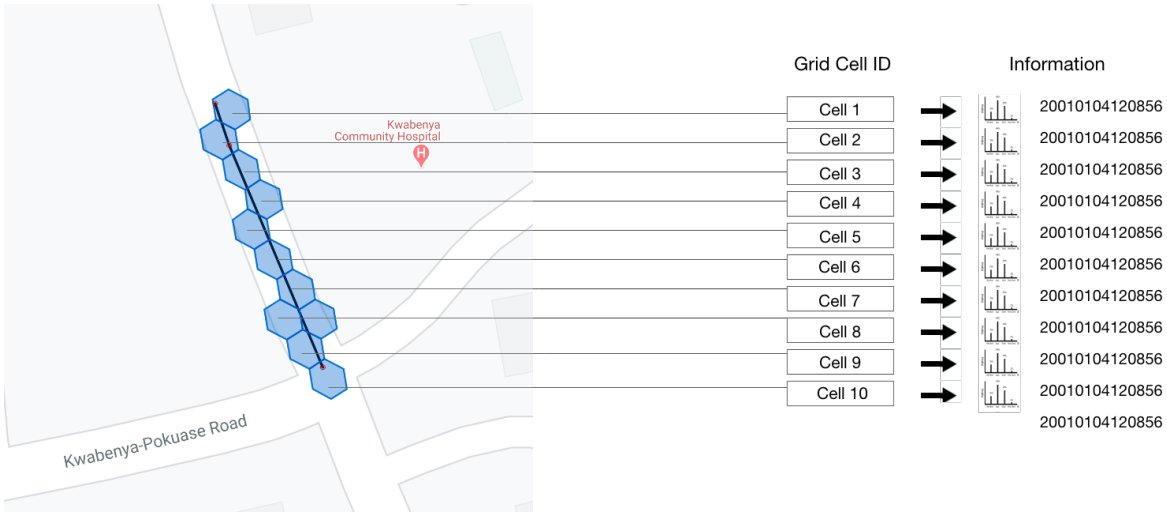

Figure 3.17: Information is stored for each grid cell

3.6 Data Retrieval and Visualization

Given an arbitrary route (road segment), the system performs the same mapping done in spatial aggregation to map the road to associated grid cells. The surface quality label of the road segment is retrieved from the datastore and used to visualize the road segment on Google Maps.

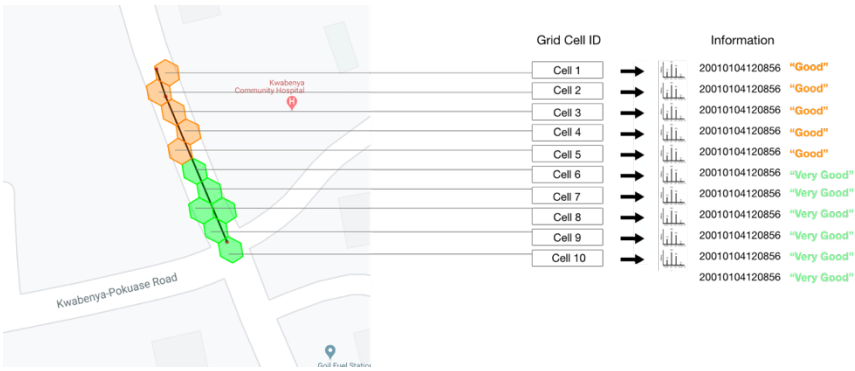

Figure 3.18: Retrieved label information for each grid cell is used to colour the cell

Chapter 4: Experiments and Results

This section describes the various experiments and tests ran to verify and demonstrate critical aspects of the stages of the system's pipeline architecture to which this paper contributes. The experiments and testing aimed to answer the following questions:

1. Is conflation as a probability aggregation method commutative and iterative?
2. What effect do outlier probabilities have on probability aggregates?
3. Does temporally aggregating daily aggregates maintain the iterative property?
4. Which has better performance at querying grid cells, H3 or S2?
5. Is map matching effective in ensuring consistent polylines for spatial aggregation?

All experiments were carried on a 2.3 quad-core 8th generation Intel Core i5 processor, 8GB RAM, running OS X 10.15.4.

Experiment 1: Verifying Critical Properties of Conflation as an Aggregation Method

This experiment verifies the commutative and iterative properties of the chosen probability methods. Experiments were done in Microsoft Excel. A random sample of five PMFs was generated (biased towards one outcome) and aggregates were computed with uniform and non-uniform weights.

Commutative Property: Is P_G (p1, p2) = P_G (p2, p1) and P_G (p1, p2, p3, p4, p5) = P_G (p2, p5, p3, p4, p1)? [28].

Table 4.1: Results from aggregating five PMFs

Input: 5 random PMFs (with a bias for very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1	0.02	0.9	0.02	0.06
PMF 2	0.3	0.5	0.01	0.19
PMF 3	0.1	0.6	0.2	0.1
PMF 4	0.9	0.02	0.03	0.05
PMF 5	0.87	0.05	0.03	0.05
P_G (p1, p2)	**0.01283**	**0.96236**	**0.00043**	**0.02438**
P_G (p2, p1)	**0.01283**	**0.96236**	**0.00043**	**0.02438**
P_G (p1, p2, p3, p4, p5)	**0.63257**	**0.36355**	**0.00004**	**0.00384**
P_G (p2, p5, p3, p4, p1)	**0.63257**	**0.36355**	**0.00004**	**0.00384**

It can be observed from Table 4.1 that the aggregate values for **P_G (p1, p2)** and **P_G (p2, p1)** are equal, likewise **P_G (p1, p2, p3, p4, p5)** = **P_G (p2, p5, p3, p4, p1)**. Therefore, the commutative property of the aggregation method is verified.

Iterative Property: Is $P_G(p1, p2, p3) = P_G(P_G(p1, p2), p3)$?

The iterative property is the most relevant because it enables the system to store one 'rolling' aggregate value, thereby removing the need for keeping all observations. Table 4.2 shows the results from computing rolling aggregates (incrementally updating aggregates with new information - $P_G(P_G(p1, p2), p3)$) and computing standard aggregates (computing aggregates of all the information - $P_G(p1, p2, p3)$)

Table 1.2: Results from computing rolling and standard aggregates of five PMFs

Input: 5 random PMFs (with a bias for very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1	0.02	0.9	0.02	0.06
PMF 2	0.3	0.5	0.01	0.19
PMF 3	0.1	0.6	0.2	0.1
PMF 4	0.9	0.02	0.03	0.05
PMF 5	0.87	0.05	0.03	0.05
P_G (p1, p2, p3, p4, p5)	**0.63257**	**0.36355**	**0.00005**	**0.00384**
P_G (p1, p2)	0.01283	0.96236	0.00043	0.02438
P_G (P_G (p1, p2), p3)	0.00221	0.99345	0.00015	0.00419
P_G (P_G (P_G (p1, p2), p3), p4)	0.09003	0.90027	0.00020	0.00950
P_G (P_G (P_G (P_G (p1, p2), p3), p4), p5)	**0.63257**	**0.363545**	**0.00005**	**0.00384**

It can be observed that the final aggregated from either computing aggregating rolling aggregates or computing standard aggregate is the same (in bold). This result verifies the iterative property of the aggregation method.

Experiment 2: What Effect do Outlier Probabilities have on Probability Aggregates?

This experiment investigates how the probability aggregation method performs with outliers (extreme probability values that deviate from other observations). The case scenario is described as follows:

Suppose we have five observed PMFs from five different smartphones for a given road segment. Four of the five observed PMFs are the same (reasonably inclined towards 'very bad' with probability 0.6) and one, an outlier (somewhat inclined towards 'good' with probability 0.6999 and extremely against 'very bad' with probability 0.0001).

Control: Assuming all observations are the same (each with uniform weight 0.2), the aggregate is shown in Table 4.3 below.

Table 4.3: Results from aggregating five unanimous PMFs (each with weight 0.2)

Input: 5 random PMFs (with a bias for a very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 2 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 3 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 4 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 5 (w = 0.2)	0.6	0.2	0.1	0.1
P_G (p1, p2, p3, p4, p5)	**0.99564**	**0.00410**	**0.00013**	**0.00013**

Without any outlier, and with unanimous observations, the aggregate surface quality of the road segment was 'very bad' with a probability ~0.99.

Now, assuming an outlier observation is introduced (highlighted in red) with extreme probability ~0.001 for 'very bad' and all observations are combined uniformly, the results are shown in Table 4.4 below.

Table 4.4: Results from aggregating one outlier PMF highlighted in red and four unanimous PMFs (each with weight 0.2)

Input: 5 random PMFs (with bias with a very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1 (w = 0.2)	0.001	0.2	0.1	0.699
PMF 2 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 3 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 4 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 5 (w = 0.2)	0.6	0.2	0.1	0.1
P_G (p1, p2, p3, p4, p5)	**0.24476**	**0.60434**	**0.01889**	**0.13201**

When an outlier was introduced, the aggregate probabilities and road surface quality information changed from 'very bad' to 'bad' (highlighted in green). This suggests that outliers do impact the result of aggregation and therefore, must be filtered out during aggregation.

A solution to this outlier phenomenon would be to give less weight to 'unreliable' or outlier observations. Table 4.5 shows the results from the scenario but with non-uniform weights (the outlier receives less weight than others).

Table 4.5: Results from aggregating four unanimous PMFs with uniform weight (~0.2) and one outlier with less weight (~0.02)

Input: 5 random PMFs (with a bias for very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1 (w = 0.02439)	0.001	0.2	0.1	0.699
PMF 2 (w = 0.2439)	0.6	0.2	0.1	0.1
PMF 3 (w = 0.2439)	0.6	0.2	0.1	0.1
PMF 4 (w = 0.2439)	0.6	0.2	0.1	0.1
PMF 5 (w = 0.2439)	0.6	0.2	0.1	0.1
P_G (p1, p2, p3, p4, p5)	**0.97687**	**0.02049**	**0.00119**	**0.00145**

When the outlier observation was weighted less than the rest of the observations, it barely affected the aggregate. The final aggregate surface quality ('very bad') highlighted in green coincided with that of the control group.

Experiment 3: Does Temporally Aggregating Daily Aggregates Maintain the Iterative Property?

Chapter 3 Section 4 explained the process of temporal aggregation. For each road segment, all PMFs observed in a day are aggregated to obtain daily aggregates which are then aggregated to obtain a global aggregate. This experiment simulates the temporal

aggregation of a random sample of PMFs observed on different days and verifies if the proposed aggregation method maintains the iterative property required by the system. The experiment was conducted with a Python 3 implementation of the desired algorithm and a case scenario as follows:

Suppose we have five PMFs observed for a given road segment and the first two, observed on Day 1, were aggregated separately from the remaining three, observed on Day 10. Will the global aggregate at Day 10 ($t_A = 10$) be the same if it were calculated as the aggregate of Day 1 and Day 10 aggregates as if it were calculated as an aggregate of all observed PMFs?

Control: Aggregate all PMFs assuming all observations are available. The PMFs observed on Day 1 are given lesser weight than more recent PMFs observed on Day 10. Table 4.6 below shows the results.

Table 4.6: Results from the temporal aggregation of five PMFs

Input: 5 random PMFs (with a bias for very bad)					
	w(Δt)	Very Bad	Bad	Very Good	Good
PMF 1 (t = 1)	0.94	0.2	0.5	0.2	0.1
PMF 2 (t = 1)	0.94	0.2	0.3	0.4	0.1
PMF 3 (t = 10)	1	0.3	0.5	0.1	0.1
PMF 4 (t = 10)	1	0.8	0.05	0.05	0.1
PMF 5 (t = 10)	1	0.1	0.1	0.1	0.7
P_G (p1, p2, p3, p4, p5)		**0.67654**	**0.24253**	**0.02695**	**0.05398**

Aggregating daily aggregates: Aggregate the daily aggregates for Day 1 and Day 10.

Table 4.7: Computing the Day 1 aggregate. Day 1 = PG (p1, p2)

Input: 5 random PMFs (with a bias for very bad)					
	w(Δt)	Very Bad	Bad	Very Good	Good
PMF 1 (t = 1)	1	0.2	0.5	0.2	0.1
PMF 2 (t = 1)	1	0.2	0.3	0.4	0.1
PG (p1, p2)		**0.14286**	**0.53571**	**0.28571**	**0.03571**

Table 4.8: Computing the Day 10 aggregate. Day 10 = PG (p3, p4, p5)

Input: 5 random PMFs (with a bias for very bad)					
	w(Δt)	Very Bad	Bad	Very Good	Good
PMF 3 (t = 10)	1	0.3	0.5	0.1	0.1
PMF 4 (t = 10)	1	0.8	0.05	0.05	0.1
PMF 5 (t = 10)	1	0.1	0.1	0.1	0.7
PG (p3, p4, p5)		**0.70588**	**0.07353**	**0.01470**	**0.20588**

Table 4.9: Results from aggregating Day 1 and Day 10 aggregates

Input: 5 random PMFs (with a bias for very bad)					
	w(Δt)	Very Bad	Bad	Very Good	Good
PG (p1, p2)	0.94	0.14286	0.53571	0.28571	0.03571
P_G (p3, p4, p5)	1	0.70588	0.07353	0.01470	0.20588
P_G (P_G (p1, p2), P_G (p3, p4, p5))		**0.67654**	**0.24253**	**0.02695**	**0.05398**

The aggregate of daily aggregates for day 1 and day 10 aggregates, P_G *(P_G (p1, p2),* P_G *(p3, p4, p5))* was the same as the aggregate of observations altogether, P_G *(p1, p2, p3, p4, p5).* This verifies that the aggregation method remains iterative when aggregating temporal (daily) aggregates.

Experiment 4: Comparing the Performance of two Candidate Grid Systems (a Hexagonal (Uber's H3) and a Quadrilateral (Google's S2) Grid System.

This experiment compares the performance of two grid index systems (hexagon-based Uber's H3 and quadrilateral-based Google's S2) in cell querying (finding the corresponding grid cell given a GPS coordinate) at similar resolutions (cell sizes).

Setup

Workloads of uniformly distributed random GPS coordinates in increasing quantities were each run 1000 times on both grid systems (implemented in Python 3) and the response times were recorded using the Python 3's native timeit module.

Results

Table 4.10: Response times (seconds) of the two systems across various workloads sizes

	Workload Size (number of queries)						
	1	**10**	**100**	**1000**	**10000**	**100000**	**1000000**
H3 (res = 12)	0.00001	0.00005	0.00048	0.00495	0.05234	0.52507	5.03337
S2 (res = 20)	0.00003	0.00025	0.00270	0.02520	0.26324	2.57054	24.64204

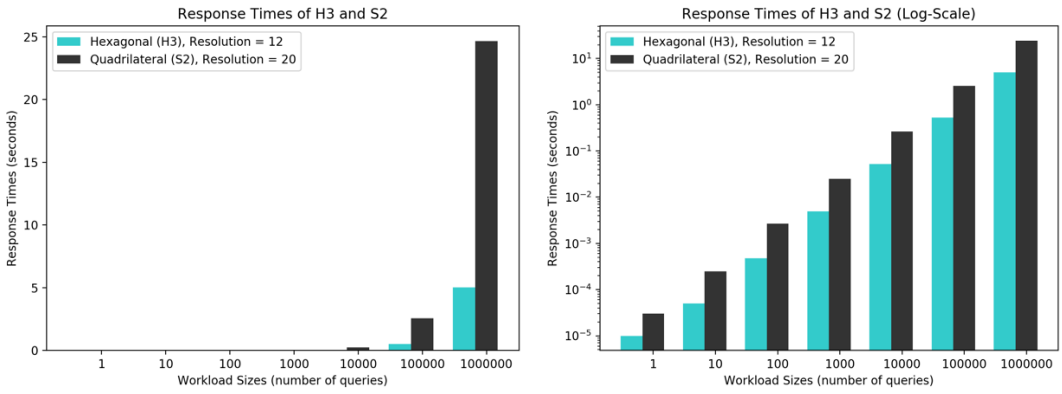

Figure 4.1: Graphs of response times against workload sizes of both systems. Left uses a linear scale and right uses a logarithmic scale.

The hexagonal-based H3 system was at least twice as fast as the quadrilateral-based S2 system at querying GPS coordinates across all workloads.

Experiment 5: Verifying the Efficacy of Map Matching in Spatial Aggregation

This simulation experiment verifies the efficacy of map matching to tackle the issue of GPS trails of vehicles travelling on uncovered sides of a road segment. Google Maps' map matching and map API was used to perform map matching and visualization.

Consider two vehicles ride on both banks of the road segment and generate parallel GPS trails T_1 and T_2. Spatial aggregation on both trails produces the associated sets of grid cells G_1 and G_2. For successful spatial aggregation, there must be no difference between G_1 and G_2. More formally, $G_1 \Delta G_2 = \emptyset$. The symmetric difference between G_1 and G_2 were compared when T_1 and T_2 were spatially aggregated with and without map-matching.

Figure 4.2: Results from spatial aggregation of T1 (red) and T2 (blue) without map matching. Only 1 grid cell was shared. |G1 Δ G2 |= 68

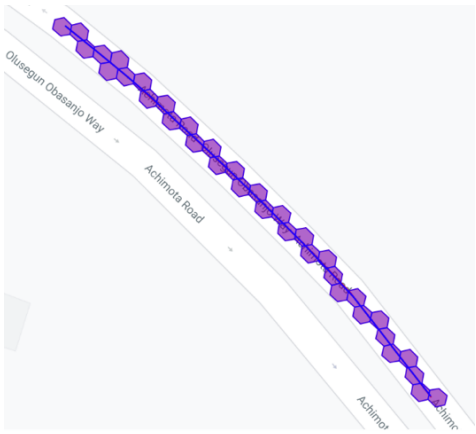

Figure: 4.3 Results from spatial aggregation of T1 (red) and T2 (blue) with map matching. All grid cells were shared. |G1 Δ G2| = 0

As depicted in Figure 4.2, without map matching, the resultant sets of grid cells from the spatial aggregation were different, but with map matching (Figure 4.3), the

resultant grid cells are equal, sharing all grid cells (shown as purple). This implies that map matching is a suitable technique to handle variant trails on the same road segment. It does not matter whether the GPS trail travels along uncovered regions as they would always be mapping to the same polyline.

Chapter 5: Conclusions and Recommendations

5.1 Summary

This paper; connects prior work [1,9,12,36] into one proposed pipeline system architecture for crowdsourcing and aggregating probabilistic road surface quality information over time and space, verifies conflation as a method for aggregating weighted probability mass function and introduces an approach to aggregating information over road segments with geospatial grid indexes. This is one more step towards making road surface quality information available to road users and administrators.

5.2 Limitations

This paper verifies the efficacy of chosen methods for the overall system theoretically but is yet to conduct a 'real-world' test. The lack of a 'ground truth' dataset of surface quality information for existing road segments makes it difficult to test how well the system performs completely.

Another limitation discovered in experiment 3 of Chapter 4 is that outlier observations of road surface quality limit the output of aggregation. The lack of a filtering process on input information makes the system susceptible to the effect of outlier information.

Lastly, the proposed system works with the assumption that a working multi-class classifier produces probabilistic surface quality information for any road segment travelled in a 10-second time window. Changes to the classifier may affect how aggregation is done

5. 3 Future Work

First, and foremost, a real-world case over a given geographic region with ground truth data available would be required to test the effectiveness and performance of this system thoroughly.

Another area to explore would be how to filter out unreliable and outlier information that may skew output aggregates. Experiment 3 hints that weighting outlier information with a much lesser value significantly reduces its effect on the aggregate.

More work could be done on how to visualize the information. Doku [12] verified that colour-coded visuals are useful in communicating road surface quality information; however, existing map services often use colours to show live traffic information. Alternative visuals can be explored, especially those considering the temporal nature of the information gathered (there should be some distinction to more recent information).

On the user perspective, an interface could be created to enable road administrators (government authorities and road authorities) to view and collect the road surface quality information gathered over long periods for further temporal analysis.

References

[1] Anthony Anabila Abeo. 2018. Evaluating and choosing a machine learning algorithm for classifying road surface quality data. Thesis. Ashesi University.

[2] African Union. 2011. *Basic guidelines for road classification and standards on trans-african highways*. African Union.

[3] Giacomo Alessandroni, Lorenz Cuno Klopfenstein, Saverio Delpriori, Matteo Dromedari, Gioele Luchetti, Brendan Paolini, Andrea Seraghiti, Emanuele Lattanzi, Valerio Freschi, Alberto Carini, and Alessandro Bogliolo. 2014. SmartRoadSense: Collaborative Road Surface Condition Monitoring.

[4] D. Allard, A. Comunian, and P. Renard. 2012. Probability Aggregation Methods in Geoscience. *Math Geosci* 44, 5 (July 2012), 545–581.

[5] Michael Bacharach. 1979. Normal Bayesian Dialogues. *Journal of the American Statistical Association* 74, 368 (1979), 837–846.

[6] Robert Bartlett. 2016. Road design standards 6.1. (2016), 11.

[7] Ravi Bhoraskar, Nagamanoj Vankadhara, Bhaskaran Raman, and Purushottam Kulkarni. 2012. Wolverine: Traffic and road condition estimation using smartphone sensors. In *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)*, 1–6.

[8] Colin P.D. Birch, Sander P. Oom, and Jonathan A. Beecham. 2007. Rectangular and hexagonal grids used for observation, experiment and simulation in ecology. *Ecological Modelling* 206, 3–4 (August 2007), 347–359.

[9] Kwabena Boohene. 2017. Automated Collection and Visualization of Road Quality Data to Aid Driver Navigation. Thesis. Ashesi University.

[10] Isaac Brodsky. 2018. H3: Uber's Hexagonal Hierarchical Spatial Index. *Uber Engineering Blog*. Retrieved December 17, 2019 from https://eng.uber.com/h3/

[11] Franz Dietrich and Christian List. 2016. Probabilistic Opinion Pooling. *The Oxford Handbook of Probability and Philosophy*.

[12] Antoinette Doku. 2014. Embedding information about road surface quality into Google Maps to improve navigation. Thesis. Ashesi University.

[13] Viengnam Douangphachanh and Hiroyuki Oneyama. 2013. Estimation of road roughness condition from smartphones under realistic settings. In *2013 13th International Conference on ITS Telecommunications (ITST)*, 433–439.

[14] Ahmed Elghriany, Ping Yi, Peng Liu, and Quan Yu. 2016. Investigation of the effect of pavement roughness on crash rates for rigid pavement. *Journal of Transportation Safety & Security* 8, 2 (April 2016), 164–176.

[15] V. Freschi, S. Delpriori, L. C. Klopfenstein, E. Lattanzi, G. Luchetti, and A. Bogliolo. 2014. Geospatial data aggregation and reduction in vehicular sensing applications: The case of road surface monitoring. In *2014 International Conference on Connected Vehicles and Expo (ICCVE)*, 711–716.

[16] Johann Gamper, Michael Böhlen, and Christian S. Jensen. 2009. Temporal Aggregation. *Encyclopedia of Database Systems* (2009), 2924–2929.

[17] Christian Genest and James V. Zidek. 1986. Combining Probability Distributions: A Critique and an Annotated Bibliography. *Statist. Sci.* 1, 1 (February 1986), 114–135.

[18] Thomas D. Gillespie. 1992. *Fundamentals of Vehicle Dynamics*. SAE International, Warrendale, PA.

[19] Google. *S2 Geometry*. Retrieved December 18, 2019 from http://s2geometry.io

[20] Mahdi Hashemi and Hassan A. Karimi. 2014. A critical review of real-time map-matching algorithms: Current issues and future directions. *Computers, Environment and Urban Systems* 48, (November 2014), 153–165.

[21] Theodore Hill. 2008. Conflations of Probability Distributions. *Transactions of the American Mathematical Society* 363, (August 2008).

[22] Theodore P. Hill and Jack Miller. 2011. How to combine independent data sets for the same quantity. *Chaos* 21, 3 (July 2011), 033102.

[23] Salil S. Kanhere. 2013. Participatory Sensing: Crowdsourcing Data from Mobile Smartphones in Urban Spaces. In *Distributed Computing and Internet Technology* (Lecture Notes in Computer Science), Springer Berlin Heidelberg, 19–26.

[24] Ted R. Miller and Eduard Zaloshnja. 2009. On a Crash Course: The Dangers and Health Costs of Deficient Roadways.

[25] Mikko Perttunen, Oleksiy Mazhelis, Fengyu Cong, Mikko Kauppila, Teemu Leppänen, Jouni Kantola, Jussi Collin, Susanna Pirttikangas, Janne Haverinen, Tapani Ristaniemi, and Jukka Riekki. 2011. Distributed Road Surface Condition Monitoring Using Mobile Phones. In *Ubiquitous Intelligence and Computing*, Ching-Hsien Hsu, Laurence T. Yang, Jianhua Ma and Chunsheng Zhu (eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 64–78.

[26] Richard Pettigrew. 2019. Aggregating incoherent agents who disagree. *Synthese* 196, 7 (July 2019), 2737–2776.

[27] Jacob Poushter. 2016. *Smartphone Ownership and Internet Usage Continues to Climb in Emerging Economies*. Pew Research Center, Washington, DC. Retrieved from https://www.pewresearch.org/global/2016/02/22/smartphone-ownership-and-internet-usage-continues-to-climb-in-emerging-economies/

[28] Roopesh Ranjan and Tilmann Gneiting. 2010. Combining probability forecasts. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72, 1 (2010), 71–91.

[29] Kevin Sahr, Denis White, and A. Jon Kimerling. 2003. Geodesic Discrete Global Grid Systems. *Cartography and Geographic Information Science* 30, 2 (January 2003), 121–134.

[30] Shahram Sattar, Songnian Li, and Michael Chapman. 2018. Road Surface Monitoring Using Smartphone Sensors:A Review. *Sensors* 18, (November 2018), 3845.

[31] Girisha D De Silva, Ravin S Perera, and Nayanajith M Laxman. Automated Pothole Detection System. 5.

[32] Iping Supriana, Dody Dharma, Dicky Satya, Dessi Satya, and Lestari. 2015. Geohash Index Based Spatial Data Model for Corporate.

[33] Kanat Tangwongsan, Martin Hirzel, Scott Schneider, and Kun-Lung Wu. 2015. General incremental sliding-window aggregation. *Proc. VLDB Endow.* 8, 7 (February 2015), 702–713.

[34] Transport Focus. 2017. *Road surface quality: what road users want from Highways England*. Transport Focus. Retrieved December 6, 2019 from https://www.transportfocus.org.uk/research-publications/publications/road-surface-quality-road-users-want-highways-england/

[35] United Nations Economic and Social Commission for Asia and the Pacific. 1993. *Asian highway classification and design standards*. United Nations Economic and Social Commission for Asia and the Pacific.

[36] Francis Delali Vorgbe. 2014. Classification of road surface quality using Android smartphone devices. Thesis. Ashesi University.

ASHESI UNIVERSITY

KWANALYTICS: A GEOGRAPHIC INFORMATION SYSTEM FOR CROWDSOURCING AND AGGREGATING ROAD SURFACE QUALITY INFORMATION FROM SMARTPHONES

UNDERGRADUATE THESIS

B.Sc. Computer Science

Kevin Kwesi Kafui de Youngster

2020

ASHESI UNIVERSITY

KWANALYTICS: A GEOGRAPHIC INFORMATION SYSTEM FOR CROWDSOURCING AND AGGREGATING ROAD SURFACE QUALITY INFORMATION FROM SMARTPHONE

UNDERGRADUATE THESIS

Undergraduate Thesis submitted to the Department of Computer Science, Ashesi University in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Science.

Kevin Kwesi Kafui de Youngster

2020

DECLARATION

I hereby declare that this Undergraduate Thesis is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature……

Candidate's Name:……

Date:……..

I hereby declare that preparation and presentation of this Undergraduate Thesis were supervised in accordance with the guidelines on supervision of Undergraduate Thesis laid down by Ashesi University.

Supervisor's Signature: ……..

Supervisor's Name:……..

Date:……

Acknowledgements

Firstly, I express my deepest gratitude to my supervisor, Dr Ayorkor Korsah, who, despite having so many responsibilities (especially during this pandemic), supported and guided me throughout the entire project. Her keen insights and level-headedness when communicating with me helped in times of despair, and for that, I am eternally grateful.

I also thank my parents and younger siblings for providing physical and emotional support to help me finish this project as we were all staying in due to the COVID-19 pandemic.

I also give special shout-outs to my colleagues, who supported me by offering to listen and pretending to understand as I rambled on my many thoughts and theories—shout-outs to Oracking Amenreynolds, Marcus Nartey, Kuukua Bentil and others.

Finally, many thanks to the lecturers including Abdul Wasay who offered advice that helped me bring this project to fruition, and the Computer Science Department of Ashesi University, for providing a robust structured curriculum that supported my growth in the understanding the field that is Computer Science.

Abstract

Road surface quality information is critical to road users when navigating road networks, and road authorities when making decisions on road infrastructure. However, not many systems exist that readily provide this information. This work connects prior work and proposes a geographic information system for crowdsourcing and aggregating probabilistic road surface quality information collected from users' smartphones from various times and at different geographic locations.

Table of Contents

Chapter 1: Introduction

Drivers, riders, and pedestrians consider multiple factors when navigating road networks. Some of these factors are distance, traffic, and the quality of road surfaces [12]. There exist mapping services, such as Google Maps and Waze, that provide information on routing, distance, and traffic, to make navigation more convenient. However, not many exist that comprehensively show the surface quality of roads, which is useful to users during navigation [34].

Research suggests a strong correlation between the surface qualities of roads and road accidents, traffic, and travel times [14,24]. It is, therefore, crucial for governments and road authorities to obtain relevant road surface quality information to make data-driven decisions on road infrastructure. This task of acquiring such information at scale is, however, challenging, costly and labour-intensive because it currently requires the use of specialized equipment [30].

With the rapid increase in usage and ownership of smartphones, especially in emerging countries [27], crowdsourcing techniques [23] can be used to obtain road surface quality data from people's smartphones. By utilizing readings from smartphone sensors such as the linear accelerometer, gyroscope, and GPS, it is possible to classify the surface quality of road segments [13]. The road surface quality information from classification can then be displayed on a map interface to be used by drivers when navigating, and by institutions such as governments and road authorities when planning transport infrastructure maintenance.

In this work, I propose Kwanalytics, a system for crowdsourcing, aggregating, storing and visualizing road surface quality information obtained from smartphone sensors. To effectively create this system, the following research questions are explored:

1. How to aggregate classification information (in the form of probability distributions) over the dimensions of time and geographical space?
2. How to store and retrieve data over segments of roads?

1.1 Prior Work

This work builds upon some prior work at Ashesi University that focus on methods of classifying road surface quality information from sensor readings, user research to determine the usefulness of road surface quality information, and approaches to collect sensor information in the first place. They are as follows:

Vorgbe [36] implemented a classifier (using logistic regression) that took input from smartphone sensors such as linear accelerometer and gyroscope readings to classify road surfaces with labels {good, fair,bad}. The classifier was able to distinguish between good and bad roads with a true positive rate of 92%, good and fair roads with 83% accuracy but was unable to differentiate between fair and bad roads.

Doku [12] conducted user experience research on whether the embedding of road surface quality information into a map interface such as Google Maps would be useful to users and how best to visualize such information. It was concluded that embedding surface quality was helpful and that a colour-coded visual might be sufficient.

Abeo [1] built up on issues raised in Vorgbe's [36] work by evaluating five classification algorithms to determine which would best classify accelerometer and

gyroscope readings. It was concluded that for the data tested on, the best performing classifier was the decision-tree based classifier with an overall accuracy rate of 92%.

Boohene [9] then implemented an application to collect sensor readings from smartphones to a backend data store and prototyped a visualization of the collected data on a map interface to aid road users in navigation.

The main objective of this research is to build upon Boohene's work [9] by proposing an alternative approach to store collected road surface quality information and filling in the gaps not addressed by his work (namely crowdsourcing and aggregation). This work also connects prior work to form an end-to-end system for crowdsourcing road and visualizing road surface quality information.

In summary, the main contributions of this work are as follows:

1. A pipeline architecture design for crowdsourcing, aggregating and storing road surface quality information.
2. A method to aggregate probability distributions over the dimension of time.
3. A geospatial grid-based approach to aggregating information over road segments and geometries.

Chapter 2: Background and Related Work

2.1 Classifying Road Surface Quality from Smartphone Sensor Readings

This project works on the assumption that road surface quality information can be obtained from smartphone sensor readings using a classifier [13]. Many research papers have explored approaches to classifying sensor data from smartphones to obtain road surface quality information. Some used threshold-based classifiers [31], some used machine learning methods such as K-means clustering [1] and Support Vector Machines [7,25] others used signal processing techniques [3]. The various approaches have strengths and weaknesses depending on factors such as frequency of recording, the speed of the vehicle, and others, as outlined in Sattar et al.'s survey [30]. Most of these projects focus on testing the feasibility of their proposed classification approach and not necessarily how readings from multiple users can be crowdsourced.

2.2 Crowdsourcing Road Surface Quality Information

Crowdsourcing involves obtaining information from large numbers of people using technology. By leveraging sensors in people's smartphones, it is possible to obtain sensor information on a large-scale at lesser costs compared to traditional approaches (for example, using sensor networks). According to Kanhere [23], this sensing can be categorized as people-centric (collecting data about the user) or environment-centric (collecting data about the user's surroundings). This project falls in the second category, where road surface quality information is obtained from people's smartphones. To the best of my knowledge, there is only one work that crowdsources road surface quality information using environment-centric sensing: *SmartRoadSense* by Alessandroni et al. [3].

Alessandroni et al. [3] proposed a Geographic Information System called 'SmartRoadSense' that crowdsources road surface quality information from smartphones. They collected accelerometer readings from smartphones, classified them with a mathematical model [18] to obtain an ordinal 'roughness index' describing the quality of the road surface. They then stored the information in a spatial database (PostgreSQL with PostGIS extension) and visualized the information using a mapping service. In their approach, they aggregated the collected road surface quality information by computing the arithmetic average of the roughness indices for each segment of a road.

Freschi et al. [26] built upon Alessandroni et al. [3]'s system to enable it to scale effectively. They acknowledged that such a system, collecting massive amounts of data, may have storage overhead as many data points have to be processed. They addressed this issue by aggregating the collected sensor data spatially and temporally. To aggregate sensor readings spatially, they sampled 'centroid points' for each road segment and averaged all roughness indices within a specific radius of the centroid points. To temporally aggregate the roughness indices, they used a weighted average function to give more weight to recent observations.

Sattar et al. [30] in reviewing the various road surface quality work claimed that "[the] best approach to crowdsourcing road surface anomalies from multiple sources would be a probabilistic and spatiotemporal-based approach that would overcome both the uncertainty and variability in road surface anomalies". State-of-the-art approaches acknowledged that indeed variance in the types of devices and vehicles used affected the accuracy of their classifiers. This paper attempt to address the issue Sattar et al. [30] raised by proposing a system that works with probabilistic classification information (obtained from a multi-class classifier) and aggregates it with spatio-temporal aggregation methods.

2.3 Data Aggregation

Since crowdsourcing requires collecting massive amounts of data, there is a need for aggregation techniques to summarize such data to reduce the storage and computation overhead in a way that preserves information. This system involves summarizing road surface quality information (in the form of probability distributions) over the dimensions of time and space. The following subsections give an overview of probability, spatial and temporal aggregation methods.

2.3.1 Probability Aggregation

The problem of combining ('pooling') probabilistic information ('opinions') from different individuals is defined by Dietrich et al. [11] as the *opinion pooling problem.* It involves applying some function ('pooling method') to a collection of probability distributions to obtain a single aggregate distribution.

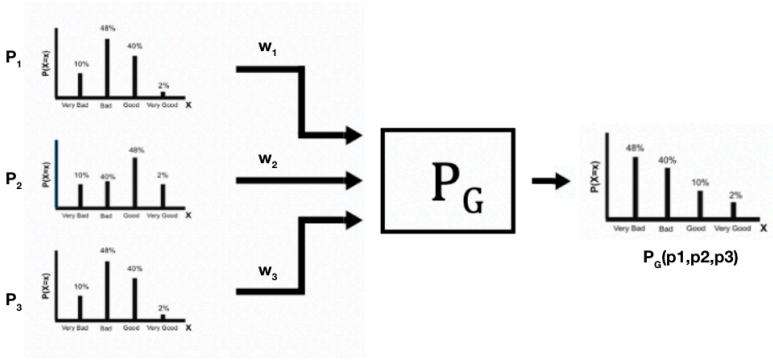

Figure 2.1: The Opinion Pooling Problem. P_1, P_2, and P_3 are combined by a pooling method P_G to obtain an aggregate distribution PG (P_1, P_2, P_3)

Pooling Methods

A pooling method is a function that combines multiple probabilities to obtain an approximation of their 'true' combination. In their reviews, Allard et al. [4] and Genest et al. [17] showed some pooling methods and explained that the choice of method depended on its application and desired properties. They also categorized most pooling methods based on how information was combined: additive or multiplicative.

Additive Methods

Additive methods express the aggregate of probabilities as the disjunction (union) of the constituent probabilities using addition. Methods include the most commonly used linear pooling (weighted arithmetic average) [5] and the beta-transformed arithmetic average [28].

$$P_G(E) = \sum_{i=1}^{n} w_i P_i(E)$$

Linear Pooling

$$P_G(E) = H_{\alpha,\beta}\left(\sum_{i=1}^{n} w_i P_i(E)\right)$$

Beta-transformed Linear Pooling

Figure 2.2: Additive Pooling Methods

Multiplicative Methods

Multiplicative methods, on the other hand, express the aggregate of probabilities as the conjunction (intersection) of these probabilities using multiplication. These methods require normalizing the aggregate probability with a constant *c*, to ensure the output is a discrete probability distribution (also called Probability Mass Function – PMF). Methods

include geometric pooling (normalized weighted geometric average) [17], multiplicative pooling [11], and conflation (normalized weighted product) [21].

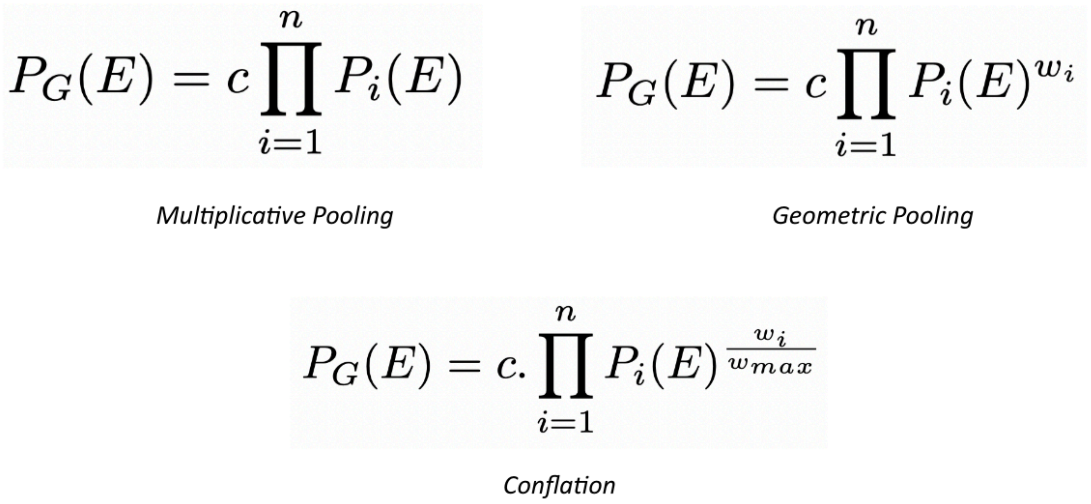

Figure 2.3: Multiplicative pooling methods

2.3.2 Temporal Aggregation

Temporal aggregation involves partitioning information into groups by a time granularity (for example, daily, monthly, or yearly), and applying a function on each group to obtain aggregates [16]. Systems that work with streaming information (sequences of continuously recorded data) often use the sliding window approach to summarize data [33]. It involves computing over only the N-most recent elements to answer queries where N is defined as the window size.

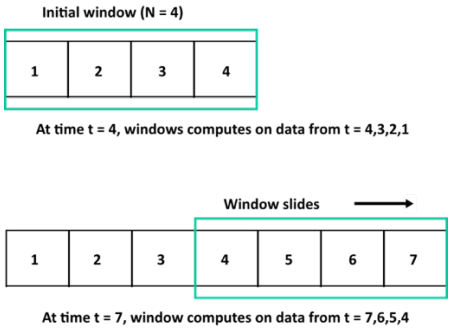

Figure 2.4: Illustration of Sliding Window

2.3.3 Spatial Aggregation

I have identified two main approaches to aggregating geospatial information regarding roads: aggregation based on road geometry and aggregation based on a grid index.

Aggregation based on Road Geometry

This involves aggregating information across road segments based on their geometry (shapes and coordinates). This approach requires prior information about road geometries and the connections between roads in a network. Roads are divided into segments by placing 'landmark points' on each road and computing aggregates for each landmark point.

Freschi et al. [15] used this approach to aggregate information for roads. They created landmark points (centroids) along a road geometry to divide it into segments. All observations that fell within a given radius of each centroid were aggregated and associated with that centroid.

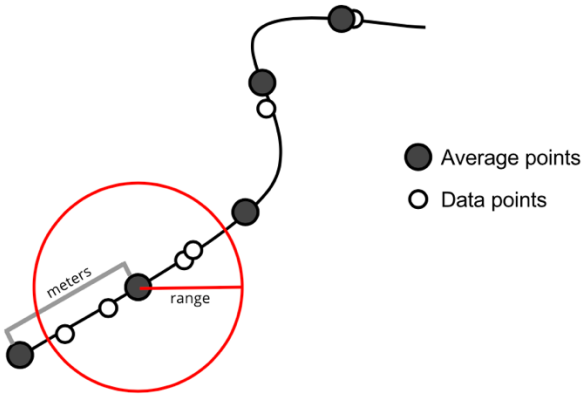

Figure 2.5: Freschi et al. [15]'s approach to spatial aggregation: placing centroids (average points) on a road segment

The limitations of this approach are that (i) it requires pre-processing a road network to determine where to place landmark points and (ii) it becomes more complicated when landmark points' locations are computed dynamically.

Aggregation based on a Grid Index

Another approach to aggregating spatial information over roads is to use a grid-based model (index) of the Earth, mapping portions of road segments to given cells and storing data for each cell. Grid indexes divide the Earth's surface into uniformly shaped cells to enable efficient aggregation of information. They can be classified into two forms: graticular and geodesic grid indexes.

Graticular Grid Indexes

These grid systems use the longitude and latitude lines (graticules) as a mesh around the Earth's surface to divide it into evenly spaced cells. They then use a geocoding

algorithm, such as GeoHash [32], to map GPS coordinates (latitude-longitude pairs) to the various cells in the grid.

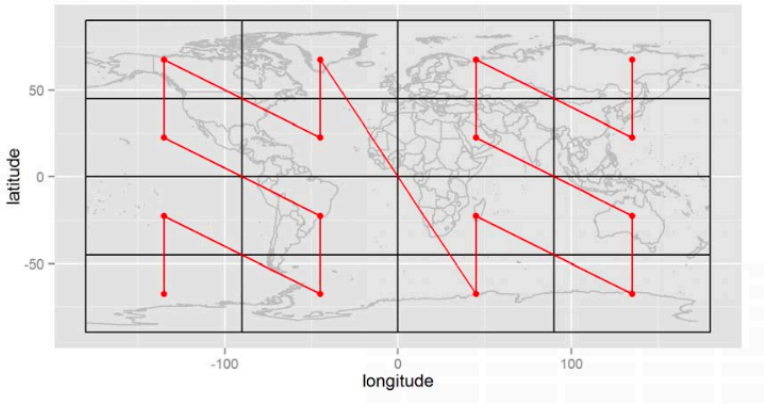

Figure 2.6: GeoHash-based approach divides the Earth with longitude and latitude lines into rectangular cells [32]. The red line indicates space-filling curve mapping each 2-D cell to a 1-D index.

A limitation of graticular grid indexes is the precision error in aggregation because cells are not uniformly shaped (due to the curvature of the Earth around the poles).

Geodesic Grid Indexes

Geodesic grid indexes [29], like graticular grid indexes, divide the Earth's surface into uniformly spaced cells. However, instead of using graticules to divide the Earth's surface, they project points on the Earth's surface unto a polyhedron and partition each face of the polyhedron into uniform grids. The use of projection overcomes the precision error from using graticules and results in uniformly shaped, uniformly sized and easily indexable grid cells. The shapes of the cells in a geodesic grid index may vary depending on the application: triangular, square, or hexagonal.

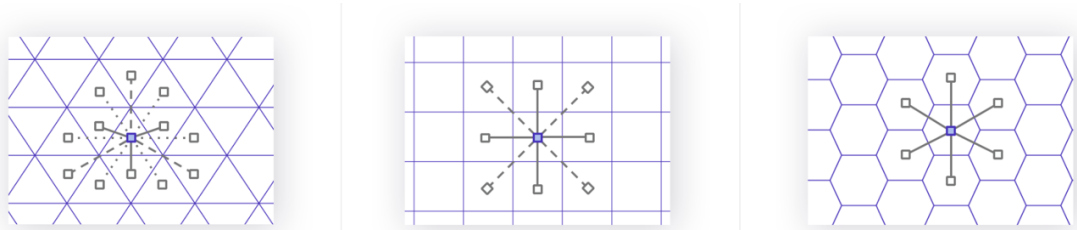

Figure 2.7: Possible cell shapes in a geodesic grid index [29]

Chapter 3: Methodology

This section introduces the architecture design of the proposed system and further overviews its various stages, showing the implementation of the stages to which this paper contributes.

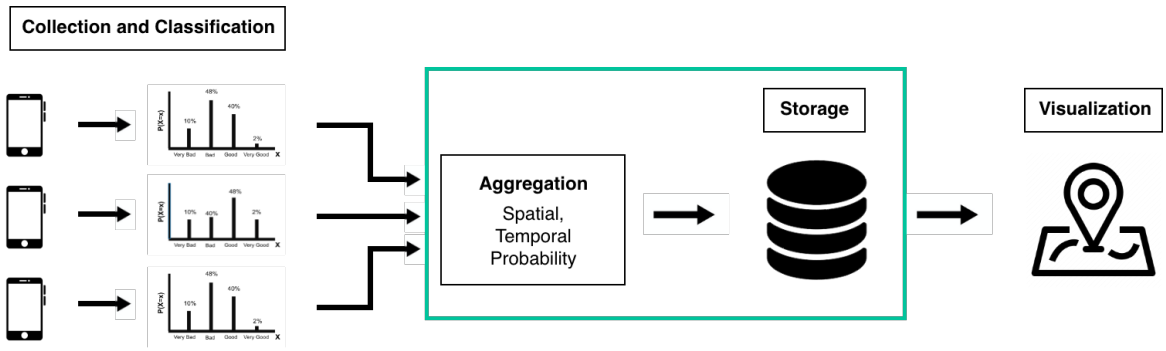

Figure 3.1: High-Level pipeline architecture of Kwanalytics, showing how contributions of this paper (highlighted in green) fit with the previous work done [1,9, 12, 36].

The proposed geographic information system (shown in figure 3.1) uses a pipeline architecture to tackle the problem of crowdsourcing road surface quality information because the processes involved occur in connected stages.

In summary, the full process of the pipeline is described as follows:

i. As vehicle navigates a road, collect sensor readings and GPS trail (**Collection**)

ii. Classify sensor readings to obtain surface quality information of the road segments on which the vehicle travels (**Classification**)

iii. Map the recorded GPS trail from (i) onto corresponding grid cells (**spatial aggregation**) and aggregate the classification information for each cell with that already associated with it (**probability and temporal aggregation**)

iv. Store the newly computed aggregate in the data store for all cells from (iii) (**Storage**)

v. Retrieve surface quality information (the aggregate) and visualize on map service (**visualization**)

3.1 Data Collection and Classification

Sensor readings and GPS trails are collected and validated from Android devices with an application built by Vorgbe [36]. The collected readings are then passed into a classifier [1, 36] which accepts a time series of sensor readings over a given segment and produces road surface quality information in the form of probability mass functions (PMFs). This output PMF gives the probability that a given road segment belongs to a given class *X* of road surface quality from a set of labels *{very bad, bad, good, very good}*.

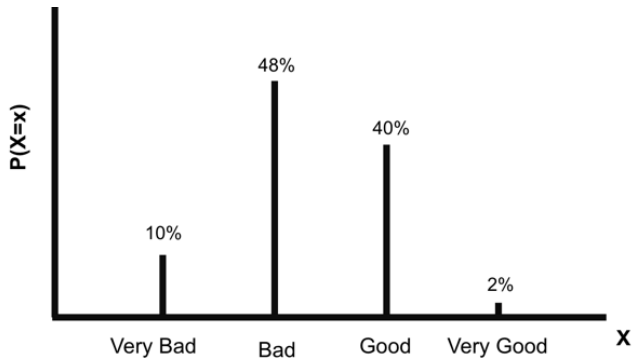

Figure 3.2: Sample output of classifier: A Probability Mass Function

x	Very Bad	Bad	Very Good	Good
P(X=x)	0.10	0.48	0.40	0.02

Figure 3.3: Hash table Representation of Probability Mass Function

The outputs from these stages are a GPS trail of a road segment (represented by a collection of latitude-longitude GPS coordinates), the corresponding surface quality information of that road segment (represented by a hash table with labels as keys and probabilities as values), and timestamp information (time of observation).

3.2 Spatial Aggregation

At this stage, the GPS trail representing a road segment (Figure 3.4) is divided into parts (Figure 3.5), and the surface quality information of the segment is associated with each part for further aggregation. Of the two approaches for spatial aggregation mentioned earlier in Chapter 2, the grid-based approach was chosen because it did not require pre-processing a road network. This makes it well suited for geographic regions where road networks undergo development and deterioration.

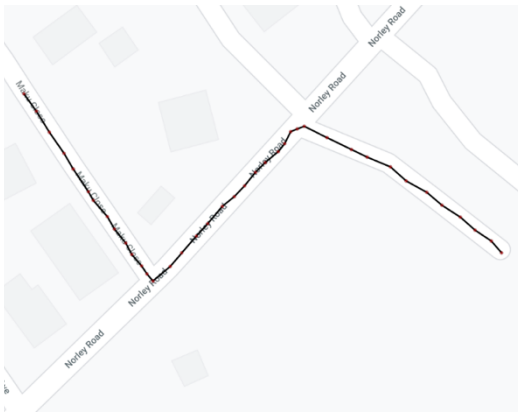

Figure 3.4: A trail of GPS coordinates T (black outline)

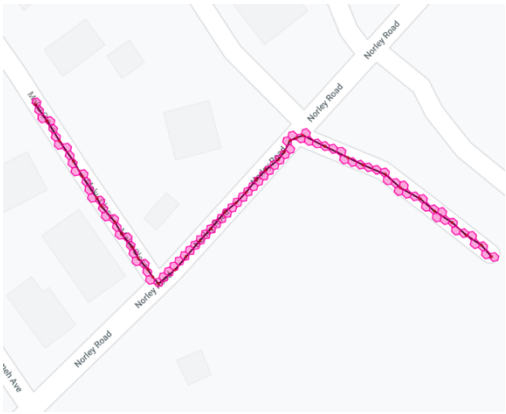

Figure 3.5: Spatially aggregated T into grid cells (pink)

Using a grid index raised two further questions or design choices:

i. What should the shape a unit grid cell be?

ii. What should the size of each unit grid cell be?

Shape of a unit Grid Cell

A hexagon-based grid system (Uber's H3 [10]) was chosen for aggregating information over road segments. Birch et al. [8] explained that the hexagonal and quadrilateral cell shapes were the most adequate for spatial aggregation and concluded that the choice of which to use depended on its application. For instance, the quadrilateral-

shaped grid cells can be recursively divided into smaller grid cells but have two types of neighbours (adjacent and diagonally separated cells). In contrast, the hexagonal-shaped grid cells are more compact and have one type of neighbour, making them more adequate for analysis involving movement across cells. Their stark differences, however, were irrelevant to the requirement of this project (cell indexing). Both were equally valid; hence the decision on which shape to use was based on the comparative performances of two available production-ready grid systems (the hexagonal grid system H3 [9] and the quadrilateral grid system S2 [19]). Experiment 4 in Chapter 4 provides more information on their performances.

Size of a unit Grid Cell

The size of a unit grid cell is crucial to the performance of the system. Using too large a cell might result in multiple roads covered by one grid cell (Figure 3.6) and using too small a cell might result in gaps and uncovered regions of a road (Figure 3.7). The choice size of a unit grid cell should ideally depend on the size of a road, but the various road standards make this problematic. Sizes and standards of roads vary by country [6] hence there may not be a one-size-fits-all.

Figure 3.6: Large grid cell size result in incorrectly representing road segments. In this scenario, one grid cell covers n > 2 different road segments

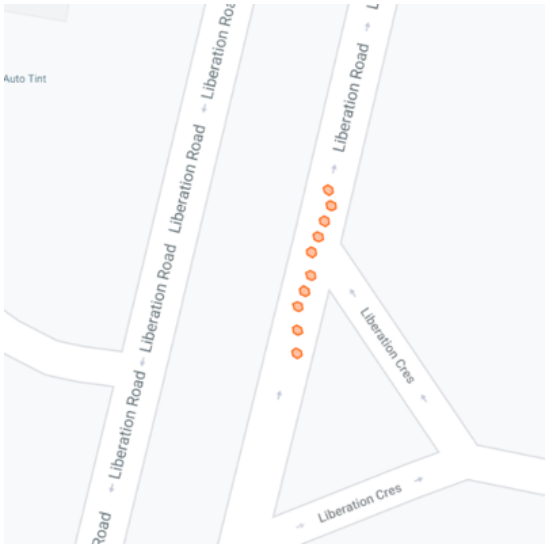

Figure 3.7: Small grid cell sizes result in large 'uncovered' regions on road segments. In this scenario, grid cells barely cover the one road segment.

The revised decision, therefore, was to pick a 'good enough' size to minimize the errors shown above. After consulting a few highway standards reports [2, 35], the chosen cell size (cell edge length) was 3.65 m (the minimum lane width of roads).

Though this choice prevents the scenario of a grid cell covering multiple road segments, it is still susceptible to leaving uncovered regions of road segments. Map matching [20], mapping 'raw' GPS trails to a road network (in this case Google Maps), was used to deal with this problem. Experiment 5 in Chapter 4 demonstrates the efficacy of this approach.

In summary, the spatial aggregation process, mapping road segments to corresponding grid cells, is as follows:

i. Obtain GPS trail over road segment

ii. Map match the trail to obtain a consistent polyline (using Google Maps API)

iii. Map the polyline to grid cells (using H3 grid system and an interpolation algorithm)

For (iii), this paper introduces an algorithm that maps polylines to grid cells by stepping incrementally along the input line segment over fixed intervals and maps each new point to a grid cell. Figure 3.8 and Figure 3.9 below visualize and outline the algorithm for this process.

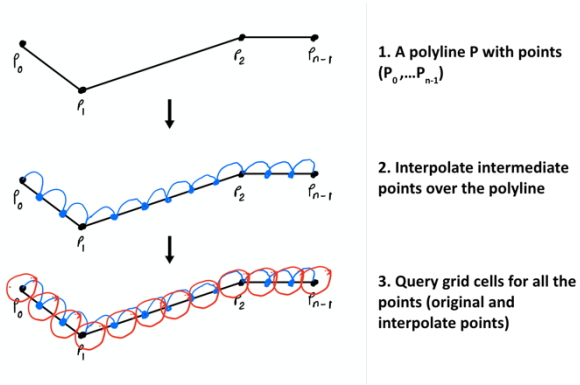

Figure 3.8: Mapping a polyline into grid cells. Grid cells are represented as red circles for easy illustration

Algorithm 1: Mapping a polyline to a corresponding set of grid cells

```
function PolylineToGridCells(P, d);
  Input  : Array P of GPS coordinates representing a polyline
           Integer d representing the step distance
  Output: A set S of grid cells
begin
    S ← set();
    l ← length of the polyline P (in units);
    current_distance ← d;
    first_cell ← query grid cell for point P0;
    S.add(first_cell);
    while current_distance < l do
        next_point ← find point that is current_distance along the
          polyline P;
        next_grid_cell ← query grid cell for next_point;
        S.add(next_grid_cell);
        current_distance ← current_distance + d;
    end
    last_cell ← query grid cell for point Pn−1;
    S.add(last_cell);
    return S
end
```

Figure 3.9: Algorithm for mapping road segments (represented as polylines) to grid cells

Runtime Complexity

The runtime complexity of the algorithm in Figure 3.9 is $O(\|P\|)$ where $\|P\|$ is the distance (in units) of the polyline P. It is more specifically $O(\|P\|/d)$ where d is the step distance.

3.3 Probability Aggregation

After spatial aggregation, the road surface quality information for each grid cell is aggregated with existing information. This section details how the chosen pooling method was selected based on its properties and the system's requirements. It further details how the method is implemented to suit the system.

The appropriate pooling method should have the following properties:

- **Weighted**: It should support weights to enable weighting observations differently.
- **Has no hyperparameters**: It should not require tuning or calibration to be useful as there is no available training data for aggregation.
- **Epistemically Valid**: It should produce approximations close to the true combination of probabilities. According to Dietrich et al. [11], an epistemically valid pooling method should depend 'primarily on the opinions of the more competent observations' as opposed to giving equal weight to each observation.
- **Commutative:** The output aggregate PMF of the method should not depend on the order in which PMFs are pooled [22]. That is, P_G (p1, p2, p3) = P_G (p2, p1, p3) [28].
- **Works with asymmetric information**: It should work with input PMFs based off different information [11] (in the system's case, smartphone sensor readings are assumed to be variant and dependent on various factors such as the type of vehicle and the quality of sensors on the phone).

- **Iterative**: Because aggregation in the system is done on a 'rolling' basis (only one aggregate is stored, and inputs are discarded), the pooling method must be iterative to support updating an aggregate with new information in a consistent manner. That is $P_G(p1, p2, p3) = P_G(P_G(p1, p2), p3)$.

Table 3.1: Comparison of pooling methods with associated relevant properties

	Linear	**Beta-Transformed Linear**	**Geometric**	**Multiplicative**	**Conflation**
Weighted	✓	✓	✓		✓
Epistemically Valid			✓	✓	✓
No Hyperparameters	✓		✓	✓	✓
Commutative	✓	✓	✓	✓	✓
Work with Asymmetric Information				✓	✓
Iterative					✓

Various literature [4,11,21–23] was consulted to obtain the properties of various pooling methods mentioned in Chapter 2. These pooling methods were then compared based on their properties (as shown in Table 3.1). Based on the system requirements, conflation, a normalized weighted product of the PMFs, was chosen. In addition to having the properties detailed in (Table 3.1), it also minimizes the loss of Shannon information and does not require weights to sum up to one since it uses relative weights [21].

Mathematical Formula for Conflation

The aggregate probability, $PG(E)$ of each outcome $E \in \{E_1, E_2 \ldots E_k\}$ across all n PMFs: $[P_1, P_2 \ldots P_n]$ with associated weights in W: $[w_1, w_2 \ldots w_n]$ is defined as:

$$P_G(E) = c. \prod_{i=1}^{n} P_i(E)^{\frac{w_i}{w_{max}}}$$

Figure 3.10: Formula for Conflation

Where *c*, the normalization constant, is the inverse of the sum of values of each outcome post-aggregation.

$$c = \frac{1}{\sum_{j=1}^{k} P_G(E_j)}$$

Figure 3.11: Normalization constant

Algorithm for Aggregating PMF (in Records) using the Conflation Method

The conflation method was implemented in Python 3. It accepts input PMFs (represented as an array of hash tables) and their corresponding weights (represented as an array of floats) and outputs an aggregate PMF.

Algorithm 2: Conflation: For finding the aggregate of weighted PMFs

```
function conflate (P, W);
  Input  : Array P of PMFs [p_0, p_1...p_{n-1}]
           Array W of weights [w_0, w_1...w_{n-1}]
  Output: Aggregate PMF P_G
begin
    P_G ← PMF();
    W_max ← max(W);
    E ← p_0.keys();
    foreach outcome e_j ∈ E do
        product ← 1;
        foreach pmf p_i ∈ P do
            product ← product × p_i[e_j]^(w_i / W_max);
        end
        P_G[e_j] ← product;
    end
    c ← 1/sum(P_G.values()) ;
    foreach outcome e_j ∈ E do
        P_G[e_j] ← c × P_G[e_j]
    end
    return P_G
end
```

Figure 3.12 Algorithm for aggregating road surface quality information (PMFs)

Runtime Complexity

The runtime complexity of the algorithm in Figure 3.12 is *O(n)*. It is more specifically *O(n.j)* where *n* is the number of PMFs to be aggregated, and *j* is the number of outcomes across all PMFs (a constant, 4 labels for each class of road surface quality).

Dealing with Zero-values.

Since conflation is based on the multiplication of probabilities, if one PMF has a zero value for a given outcome, the aggregate will continue to be zero. This property is defined by Allard et al. [4] as '0/1 enforcing' and was overcome by working within a range [0.001, 0.999].

3.4 Temporal Aggregation

For later temporal analysis of road surface quality information, the system uses a time granularity of a day. For each road segment corresponding to a grid cell, the system keeps daily aggregates (aggregate of all PMFs observed during a day).

Further, the sliding window approach is used to temporally aggregate observations (PMFs) because this system works with streaming information. A window size of 365 days was chosen to connote "considering observations made over the past year (365 days), what is the surface quality of a road segment?". Therefore, for each grid cell representing a portion of a road segment, daily aggregates as well as global aggregates are kept and regularly updated. Global aggregates represent the surface quality information of a road segment at any time t, and daily aggregates represent the surface quality of a road segment at any day over the past 365 days.

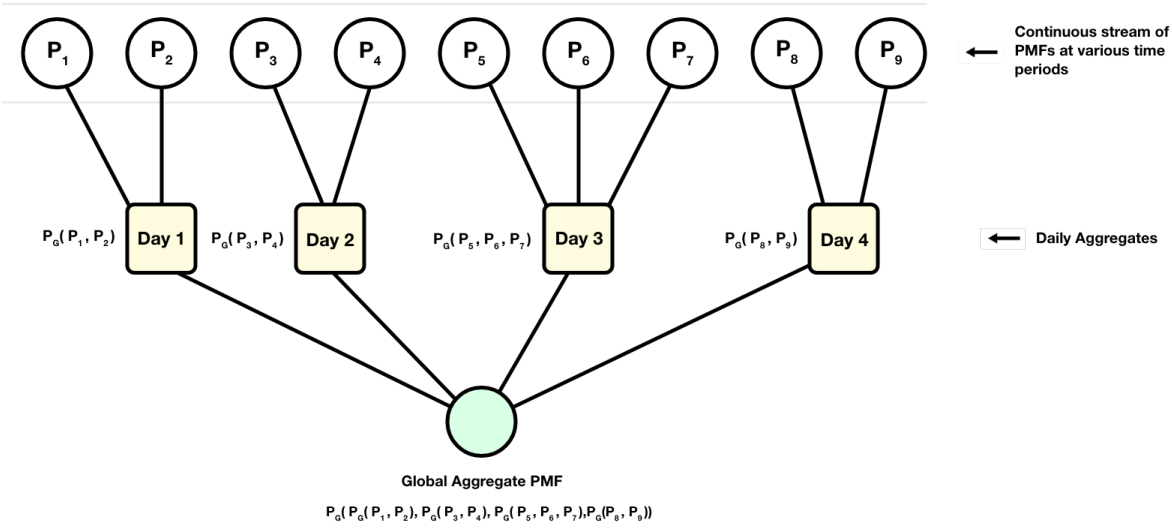

Figure 3.13: Temporal aggregation: for each road segment, the continuous data are aggregated by day, and each day's aggregate are further aggregated to obtain a global aggregate that represents the surface quality of that road segment.

Weighting Observations by Time

A requirement for this system is to consider recent observations (PMFs) more than older observations. This was done by weighting observations according to how long ago they were observed. An exponential decay function was used to weigh observations because of its horizontal asymptote. As an observation gets older, its corresponding weight approaches zero. The weighting function defines the weight *w* of a PMF observed at time t_i aggregated at time t_A as follows:

$$w(\Delta t) = \exp(\neg\frac{\Delta t}{T}), where\, \Delta t = t_A - t_i$$

Figure 3.14: Weighting function to weight observations according to their recency

Where *T,* the time constant, is the value of Δt that gives a weight of *0.368.* Since we are working with a sliding window of length 1 year (365 days), the value of T = 134 days (0.365 x 365 days).

To illustrate how weights are assigned to observations, consider an example. Suppose observations have been made on Day 1, Day 10 and Day 100 and aggregation is performed on Day 100, the weights of each observation are calculated as follows:

Table 3.2: Showing how weights are assigned to each daily aggregate when computing the global aggregate

	Day 10	Day 30	Day 100
t_i	10	30	100
t_A	100	100	100
Δt	90	70	0
w(Δt)	**0.51086915**	**0.59310249**	**1**

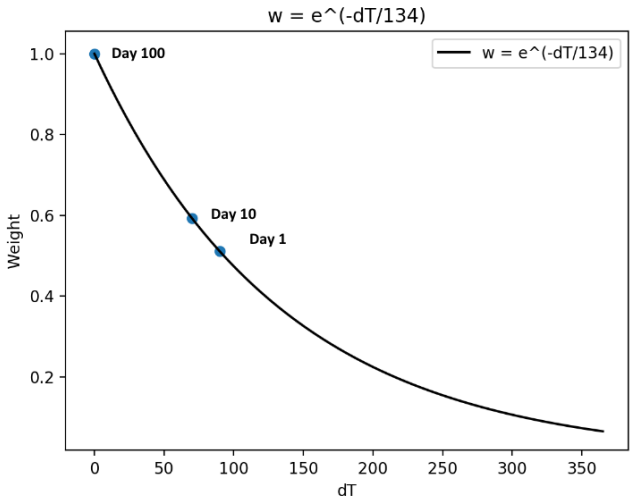

Figure 3.15: Graph representing w(t) the weight function and the corresponding weights of the three observations

Note: Observations made on the same day are aggregated with a uniform weight of 1 since Δt is 0 for those observations.

End-to-end Aggregation: Putting it all together

In the previous subsections, the various aspects of aggregation (spatial, probability and temporal) were detailed. This subsection attempts to put them all together in one procedure.

Given a GPS trail of coordinates, T representing a road segment, taken at DateTime t, with surface quality information (PMF) P, the aggregation process is as follows:

Algorithm 4: Aggregation

```
function aggregate (T, P, t)
Input  : Array T of GPS coordinates representing a trail
         PMF P representing the surface quality of the trail
         t timestamp of when the readings were recorded
Output:
begin
    Perform spatial aggregation:
    trail ← mapmatch(T)
    cells ← polylineToGridCells(line)
    foreach cell c ∈ cells do
        Update the daily aggregate with new observation P:
        d ← get daily aggregate for cell c based on time t
        W ← create array of weights for d and P
        d_new ← conflate([d, P], W)

        Recompute the global aggregate P_G:
        D ← get array of all daily aggregates for cell c
        W_D ← create array of weights for each daily aggregate in D
        P_G ← conflate(D, W_D)

        Update the road surface quality label for c to:
        label ← arg max_E P_G(E)
    end
end
```

Figure 3.16: Algorithm of the entire aggregation process

Runtime Complexity

The asymptotic runtime complexity of the aggregation process is *O(c)*, where *c* is the number of grid cells obtained from the spatial aggregation of the GPS trail.

3.5 Data Storage

A document-based datastore, MongoDB, is used to store surface quality information over road segments (grid cells in the grid index). Each document represents associated information of each grid cell and consists:

i. The grid cell's ID defined by the grid index system used for querying

ii. The global aggregate PMF

iii. A timestamp for the global aggregate PMF

iv. Label for the surface quality information

v. A collection of PMF-Timestamp pairs representing daily aggregates and the times they were recorded

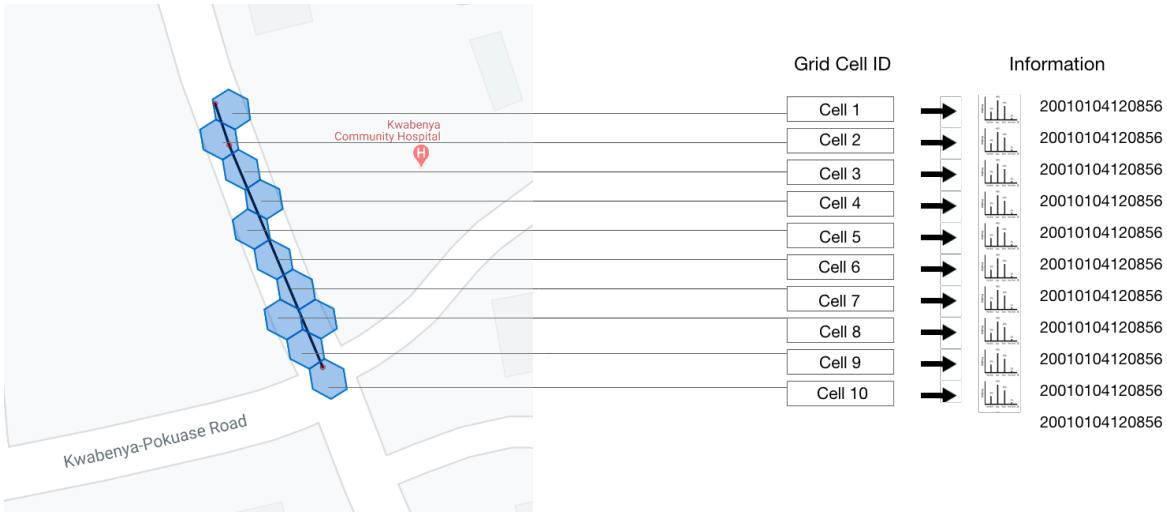

Figure 3.17: Information is stored for each grid cell

3.6 Data Retrieval and Visualization

Given an arbitrary route (road segment), the system performs the same mapping done in spatial aggregation to map the road to associated grid cells. The surface quality label of the road segment is retrieved from the datastore and used to visualize the road segment on Google Maps.

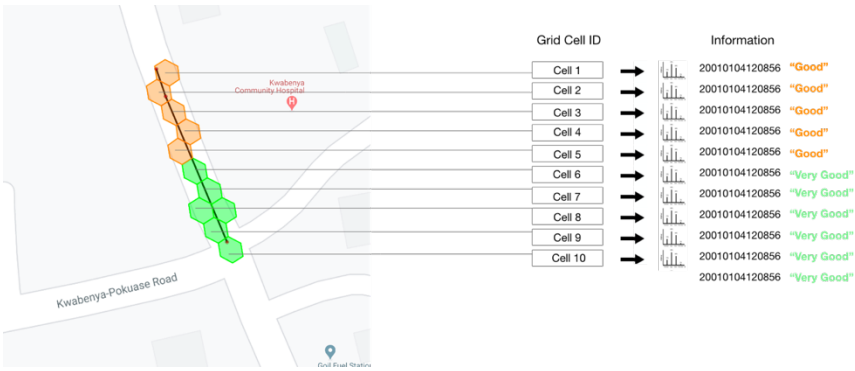

Figure 3.18: Retrieved label information for each grid cell is used to colour the cell

Chapter 4: Experiments and Results

This section describes the various experiments and tests ran to verify and demonstrate critical aspects of the stages of the system's pipeline architecture to which this paper contributes. The experiments and testing aimed to answer the following questions:

1. Is conflation as a probability aggregation method commutative and iterative?
2. What effect do outlier probabilities have on probability aggregates?
3. Does temporally aggregating daily aggregates maintain the iterative property?
4. Which has better performance at querying grid cells, H3 or S2?
5. Is map matching effective in ensuring consistent polylines for spatial aggregation?

All experiments were carried on a 2.3 quad-core 8th generation Intel Core i5 processor, 8GB RAM, running OS X 10.15.4.

Experiment 1: Verifying Critical Properties of Conflation as an Aggregation Method

This experiment verifies the commutative and iterative properties of the chosen probability methods. Experiments were done in Microsoft Excel. A random sample of five PMFs was generated (biased towards one outcome) and aggregates were computed with uniform and non-uniform weights.

Commutative Property: Is P_G (p1, p2) = P_G (p2, p1) and P_G (p1, p2, p3, p4, p5) = P_G (p2, p5, p3, p4, p1)? [28].

Table 4.1: Results from aggregating five PMFs

Input: 5 random PMFs (with a bias for very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1	0.02	0.9	0.02	0.06
PMF 2	0.3	0.5	0.01	0.19
PMF 3	0.1	0.6	0.2	0.1
PMF 4	0.9	0.02	0.03	0.05
PMF 5	0.87	0.05	0.03	0.05
P_G (p1, p2)	**0.01283**	**0.96236**	**0.00043**	**0.02438**
P_G (p2, p1)	**0.01283**	**0.96236**	**0.00043**	**0.02438**
P_G (p1, p2, p3, p4, p5)	**0.63257**	**0.36355**	**0.00004**	**0.00384**
P_G (p2, p5, p3, p4, p1)	**0.63257**	**0.36355**	**0.00004**	**0.00384**

It can be observed from Table 4.1 that the aggregate values for **P_G (p1, p2)** and **P_G (p2, p1)** are equal, likewise **P_G (p1, p2, p3, p4, p5)** = **P_G (p2, p5, p3, p4, p1)**. Therefore, the commutative property of the aggregation method is verified.

Iterative Property: Is $P_G(p1, p2, p3) = P_G(P_G(p1, p2), p3)$?

The iterative property is the most relevant because it enables the system to store one 'rolling' aggregate value, thereby removing the need for keeping all observations. Table 4.2 shows the results from computing rolling aggregates (incrementally updating aggregates with new information - $P_G(P_G(p1, p2), p3)$) and computing standard aggregates (computing aggregates of all the information - $P_G(p1, p2, p3)$)

Table 1.2: Results from computing rolling and standard aggregates of five PMFs

Input: 5 random PMFs (with a bias for very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1	0.02	0.9	0.02	0.06
PMF 2	0.3	0.5	0.01	0.19
PMF 3	0.1	0.6	0.2	0.1
PMF 4	0.9	0.02	0.03	0.05
PMF 5	0.87	0.05	0.03	0.05
P_G (p1, p2, p3, p4, p5)	**0.63257**	**0.36355**	**0.00005**	**0.00384**
P_G (p1, p2)	0.01283	0.96236	0.00043	0.02438
P_G (P_G (p1, p2), p3)	0.00221	0.99345	0.00015	0.00419
P_G (P_G (P_G (p1, p2), p3), p4)	0.09003	0.90027	0.00020	0.00950
P_G (P_G (P_G (P_G (p1, p2), p3), p4), p5)	**0.63257**	**0.363545**	**0.00005**	**0.00384**

It can be observed that the final aggregated from either computing aggregating rolling aggregates or computing standard aggregate is the same (in bold). This result verifies the iterative property of the aggregation method.

Experiment 2: What Effect do Outlier Probabilities have on Probability Aggregates?

This experiment investigates how the probability aggregation method performs with outliers (extreme probability values that deviate from other observations). The case scenario is described as follows:

Suppose we have five observed PMFs from five different smartphones for a given road segment. Four of the five observed PMFs are the same (reasonably inclined towards 'very bad' with probability 0.6) and one, an outlier (somewhat inclined towards 'good' with probability 0.6999 and extremely against 'very bad' with probability 0.0001).

Control: Assuming all observations are the same (each with uniform weight 0.2), the aggregate is shown in Table 4.3 below.

Table 4.3: Results from aggregating five unanimous PMFs (each with weight 0.2)

Input: 5 random PMFs (with a bias for a very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 2 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 3 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 4 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 5 (w = 0.2)	0.6	0.2	0.1	0.1
P_G (p1, p2, p3, p4, p5)	**0.99564**	**0.00410**	**0.00013**	**0.00013**

Without any outlier, and with unanimous observations, the aggregate surface quality of the road segment was 'very bad' with a probability ~0.99.

Now, assuming an outlier observation is introduced (highlighted in red) with extreme probability ~0.001 for 'very bad' and all observations are combined uniformly, the results are shown in Table 4.4 below.

Table 4.4: Results from aggregating one outlier PMF highlighted in red and four unanimous PMFs (each with weight 0.2)

Input: 5 random PMFs (with bias with a very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1 (w = 0.2)	0.001	0.2	0.1	0.699
PMF 2 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 3 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 4 (w = 0.2)	0.6	0.2	0.1	0.1
PMF 5 (w = 0.2)	0.6	0.2	0.1	0.1
P_G (p1, p2, p3, p4, p5)	**0.24476**	**0.60434**	**0.01889**	**0.13201**

When an outlier was introduced, the aggregate probabilities and road surface quality information changed from 'very bad' to 'bad' (highlighted in green). This suggests that outliers do impact the result of aggregation and therefore, must be filtered out during aggregation.

A solution to this outlier phenomenon would be to give less weight to 'unreliable' or outlier observations. Table 4.5 shows the results from the scenario but with non-uniform weights (the outlier receives less weight than others).

Table 4.5: Results from aggregating four unanimous PMFs with uniform weight (~0.2) and one outlier with less weight (~0.02)

Input: 5 random PMFs (with a bias for very bad)				
	Very Bad	Bad	Very Good	Good
PMF 1 (w = 0.02439)	0.001	0.2	0.1	0.699
PMF 2 (w = 0.2439)	0.6	0.2	0.1	0.1
PMF 3 (w = 0.2439)	0.6	0.2	0.1	0.1
PMF 4 (w = 0.2439)	0.6	0.2	0.1	0.1
PMF 5 (w = 0.2439)	0.6	0.2	0.1	0.1
P_G (p1, p2, p3, p4, p5)	**0.97687**	**0.02049**	**0.00119**	**0.00145**

When the outlier observation was weighted less than the rest of the observations, it barely affected the aggregate. The final aggregate surface quality ('very bad') highlighted in green coincided with that of the control group.

Experiment 3: Does Temporally Aggregating Daily Aggregates Maintain the Iterative Property?

Chapter 3 Section 4 explained the process of temporal aggregation. For each road segment, all PMFs observed in a day are aggregated to obtain daily aggregates which are then aggregated to obtain a global aggregate. This experiment simulates the temporal

aggregation of a random sample of PMFs observed on different days and verifies if the proposed aggregation method maintains the iterative property required by the system. The experiment was conducted with a Python 3 implementation of the desired algorithm and a case scenario as follows:

Suppose we have five PMFs observed for a given road segment and the first two, observed on Day 1, were aggregated separately from the remaining three, observed on Day 10. Will the global aggregate at Day 10 ($t_A = 10$) be the same if it were calculated as the aggregate of Day 1 and Day 10 aggregates as if it were calculated as an aggregate of all observed PMFs?

Control: Aggregate all PMFs assuming all observations are available. The PMFs observed on Day 1 are given lesser weight than more recent PMFs observed on Day 10. Table 4.6 below shows the results.

Table 4.6: Results from the temporal aggregation of five PMFs

Input: 5 random PMFs (with a bias for very bad)					
	w(Δt)	Very Bad	Bad	Very Good	Good
PMF 1 (t = 1)	0.94	0.2	0.5	0.2	0.1
PMF 2 (t = 1)	0.94	0.2	0.3	0.4	0.1
PMF 3 (t = 10)	1	0.3	0.5	0.1	0.1
PMF 4 (t = 10)	1	0.8	0.05	0.05	0.1
PMF 5 (t = 10)	1	0.1	0.1	0.1	0.7
P_G (p1, p2, p3, p4, p5)		**0.67654**	**0.24253**	**0.02695**	**0.05398**

Aggregating daily aggregates: Aggregate the daily aggregates for Day 1 and Day 10.

Table 4.7: Computing the Day 1 aggregate. Day 1 = PG (p1, p2)

Input: 5 random PMFs (with a bias for very bad)					
	w(Δt)	Very Bad	Bad	Very Good	Good
PMF 1 (t = 1)	1	0.2	0.5	0.2	0.1
PMF 2 (t = 1)	1	0.2	0.3	0.4	0.1
PG (p1, p2)		**0.14286**	**0.53571**	**0.28571**	**0.03571**

Table 4.8: Computing the Day 10 aggregate. Day 10 = PG (p3, p4, p5)

Input: 5 random PMFs (with a bias for very bad)					
	w(Δt)	Very Bad	Bad	Very Good	Good
PMF 3 (t = 10)	1	0.3	0.5	0.1	0.1
PMF 4 (t = 10)	1	0.8	0.05	0.05	0.1
PMF 5 (t = 10)	1	0.1	0.1	0.1	0.7
PG (p3, p4, p5)		**0.70588**	**0.07353**	**0.01470**	**0.20588**

Table 4.9: Results from aggregating Day 1 and Day 10 aggregates

Input: 5 random PMFs (with a bias for very bad)					
	w(Δt)	Very Bad	Bad	Very Good	Good
PG (p1, p2)	0.94	0.14286	0.53571	0.28571	0.03571
P_G (p3, p4, p5)	1	0.70588	0.07353	0.01470	0.20588
P_G (P_G (p1, p2), P_G (p3, p4, p5))		**0.67654**	**0.24253**	**0.02695**	**0.05398**

The aggregate of daily aggregates for day 1 and day 10 aggregates, P_G *(P_G (p1, p2),* P_G *(p3, p4, p5))* was the same as the aggregate of observations altogether, P_G *(p1, p2, p3, p4, p5).* This verifies that the aggregation method remains iterative when aggregating temporal (daily) aggregates.

Experiment 4: Comparing the Performance of two Candidate Grid Systems (a Hexagonal (Uber's H3) and a Quadrilateral (Google's S2) Grid System.

This experiment compares the performance of two grid index systems (hexagon-based Uber's H3 and quadrilateral-based Google's S2) in cell querying (finding the corresponding grid cell given a GPS coordinate) at similar resolutions (cell sizes).

Setup

Workloads of uniformly distributed random GPS coordinates in increasing quantities were each run 1000 times on both grid systems (implemented in Python 3) and the response times were recorded using the Python 3's native timeit module.

Results

Table 4.10: Response times (seconds) of the two systems across various workloads sizes

	Workload Size (number of queries)						
	1	**10**	**100**	**1000**	**10000**	**100000**	**1000000**
H3 (res = 12)	0.00001	0.00005	0.00048	0.00495	0.05234	0.52507	5.03337
S2 (res = 20)	0.00003	0.00025	0.00270	0.02520	0.26324	2.57054	24.64204

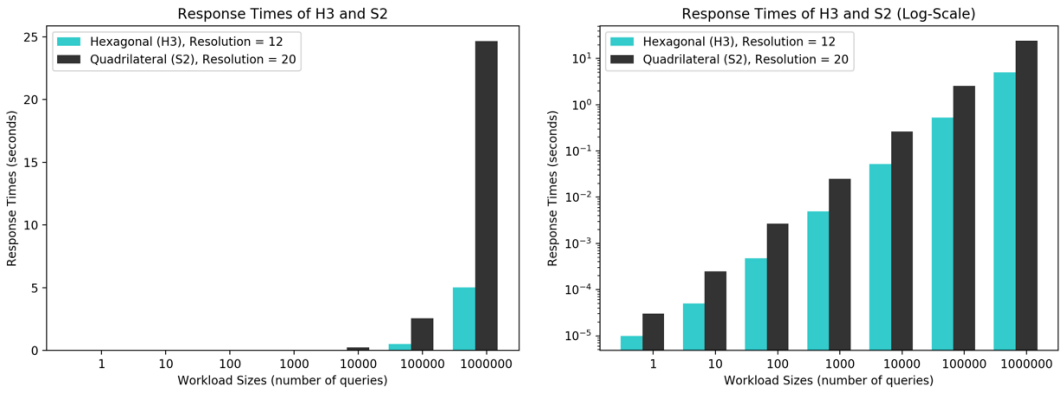

Figure 4.1: Graphs of response times against workload sizes of both systems. Left uses a linear scale and right uses a logarithmic scale.

The hexagonal-based H3 system was at least twice as fast as the quadrilateral-based S2 system at querying GPS coordinates across all workloads.

Experiment 5: Verifying the Efficacy of Map Matching in Spatial Aggregation

This simulation experiment verifies the efficacy of map matching to tackle the issue of GPS trails of vehicles travelling on uncovered sides of a road segment. Google Maps' map matching and map API was used to perform map matching and visualization.

Consider two vehicles ride on both banks of the road segment and generate parallel GPS trails T_1 and T_2. Spatial aggregation on both trails produces the associated sets of grid cells G_1 and G_2. For successful spatial aggregation, there must be no difference between G_1 and G_2. More formally, $G_1 \Delta G_2 = \emptyset$. The symmetric difference between G_1 and G_2 were compared when T_1 and T_2 were spatially aggregated with and without map-matching.

Figure 4.2: Results from spatial aggregation of T1 (red) and T2 (blue) without map matching. Only 1 grid cell was shared. |G1 Δ G2 |= 68

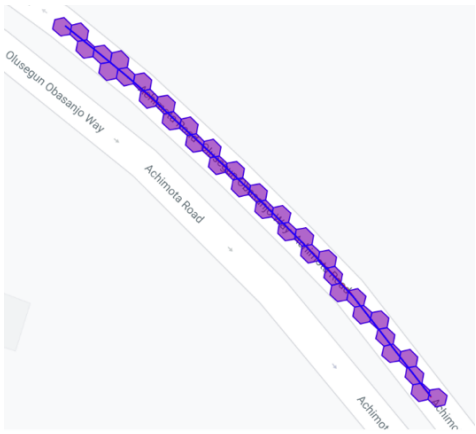

Figure: 4.3 Results from spatial aggregation of T1 (red) and T2 (blue) with map matching. All grid cells were shared. |G1 Δ G2| = 0

As depicted in Figure 4.2, without map matching, the resultant sets of grid cells from the spatial aggregation were different, but with map matching (Figure 4.3), the

resultant grid cells are equal, sharing all grid cells (shown as purple). This implies that map matching is a suitable technique to handle variant trails on the same road segment. It does not matter whether the GPS trail travels along uncovered regions as they would always be mapping to the same polyline.

Chapter 5: Conclusions and Recommendations

5.1 Summary

This paper; connects prior work [1,9,12,36] into one proposed pipeline system architecture for crowdsourcing and aggregating probabilistic road surface quality information over time and space, verifies conflation as a method for aggregating weighted probability mass function and introduces an approach to aggregating information over road segments with geospatial grid indexes. This is one more step towards making road surface quality information available to road users and administrators.

5.2 Limitations

This paper verifies the efficacy of chosen methods for the overall system theoretically but is yet to conduct a 'real-world' test. The lack of a 'ground truth' dataset of surface quality information for existing road segments makes it difficult to test how well the system performs completely.

Another limitation discovered in experiment 3 of Chapter 4 is that outlier observations of road surface quality limit the output of aggregation. The lack of a filtering process on input information makes the system susceptible to the effect of outlier information.

Lastly, the proposed system works with the assumption that a working multi-class classifier produces probabilistic surface quality information for any road segment travelled in a 10-second time window. Changes to the classifier may affect how aggregation is done

5. 3 Future Work

First, and foremost, a real-world case over a given geographic region with ground truth data available would be required to test the effectiveness and performance of this system thoroughly.

Another area to explore would be how to filter out unreliable and outlier information that may skew output aggregates. Experiment 3 hints that weighting outlier information with a much lesser value significantly reduces its effect on the aggregate.

More work could be done on how to visualize the information. Doku [12] verified that colour-coded visuals are useful in communicating road surface quality information; however, existing map services often use colours to show live traffic information. Alternative visuals can be explored, especially those considering the temporal nature of the information gathered (there should be some distinction to more recent information).

On the user perspective, an interface could be created to enable road administrators (government authorities and road authorities) to view and collect the road surface quality information gathered over long periods for further temporal analysis.

References

[1] Anthony Anabila Abeo. 2018. Evaluating and choosing a machine learning algorithm for classifying road surface quality data. Thesis. Ashesi University.

[2] African Union. 2011. *Basic guidelines for road classification and standards on trans-african highways*. African Union.

[3] Giacomo Alessandroni, Lorenz Cuno Klopfenstein, Saverio Delpriori, Matteo Dromedari, Gioele Luchetti, Brendan Paolini, Andrea Seraghiti, Emanuele Lattanzi, Valerio Freschi, Alberto Carini, and Alessandro Bogliolo. 2014. SmartRoadSense: Collaborative Road Surface Condition Monitoring.

[4] D. Allard, A. Comunian, and P. Renard. 2012. Probability Aggregation Methods in Geoscience. *Math Geosci* 44, 5 (July 2012), 545–581.

[5] Michael Bacharach. 1979. Normal Bayesian Dialogues. *Journal of the American Statistical Association* 74, 368 (1979), 837–846.

[6] Robert Bartlett. 2016. Road design standards 6.1. (2016), 11.

[7] Ravi Bhoraskar, Nagamanoj Vankadhara, Bhaskaran Raman, and Purushottam Kulkarni. 2012. Wolverine: Traffic and road condition estimation using smartphone sensors. In *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)*, 1–6.

[8] Colin P.D. Birch, Sander P. Oom, and Jonathan A. Beecham. 2007. Rectangular and hexagonal grids used for observation, experiment and simulation in ecology. *Ecological Modelling* 206, 3–4 (August 2007), 347–359.

[9] Kwabena Boohene. 2017. Automated Collection and Visualization of Road Quality Data to Aid Driver Navigation. Thesis. Ashesi University.

[10] Isaac Brodsky. 2018. H3: Uber's Hexagonal Hierarchical Spatial Index. *Uber Engineering Blog*. Retrieved December 17, 2019 from https://eng.uber.com/h3/

[11] Franz Dietrich and Christian List. 2016. Probabilistic Opinion Pooling. *The Oxford Handbook of Probability and Philosophy*.

[12] Antoinette Doku. 2014. Embedding information about road surface quality into Google Maps to improve navigation. Thesis. Ashesi University.

[13] Viengnam Douangphachanh and Hiroyuki Oneyama. 2013. Estimation of road roughness condition from smartphones under realistic settings. In *2013 13th International Conference on ITS Telecommunications (ITST)*, 433–439.

[14] Ahmed Elghriany, Ping Yi, Peng Liu, and Quan Yu. 2016. Investigation of the effect of pavement roughness on crash rates for rigid pavement. *Journal of Transportation Safety & Security* 8, 2 (April 2016), 164–176.

[15] V. Freschi, S. Delpriori, L. C. Klopfenstein, E. Lattanzi, G. Luchetti, and A. Bogliolo. 2014. Geospatial data aggregation and reduction in vehicular sensing applications: The case of road surface monitoring. In *2014 International Conference on Connected Vehicles and Expo (ICCVE)*, 711–716.

[16] Johann Gamper, Michael Böhlen, and Christian S. Jensen. 2009. Temporal Aggregation. *Encyclopedia of Database Systems* (2009), 2924–2929.

[17] Christian Genest and James V. Zidek. 1986. Combining Probability Distributions: A Critique and an Annotated Bibliography. *Statist. Sci.* 1, 1 (February 1986), 114–135.

[18] Thomas D. Gillespie. 1992. *Fundamentals of Vehicle Dynamics*. SAE International, Warrendale, PA.

[19] Google. *S2 Geometry*. Retrieved December 18, 2019 from http://s2geometry.io

[20] Mahdi Hashemi and Hassan A. Karimi. 2014. A critical review of real-time map-matching algorithms: Current issues and future directions. *Computers, Environment and Urban Systems* 48, (November 2014), 153–165.

[21] Theodore Hill. 2008. Conflations of Probability Distributions. *Transactions of the American Mathematical Society* 363, (August 2008).

[22] Theodore P. Hill and Jack Miller. 2011. How to combine independent data sets for the same quantity. *Chaos* 21, 3 (July 2011), 033102.

[23] Salil S. Kanhere. 2013. Participatory Sensing: Crowdsourcing Data from Mobile Smartphones in Urban Spaces. In *Distributed Computing and Internet Technology* (Lecture Notes in Computer Science), Springer Berlin Heidelberg, 19–26.

[24] Ted R. Miller and Eduard Zaloshnja. 2009. On a Crash Course: The Dangers and Health Costs of Deficient Roadways.

[25] Mikko Perttunen, Oleksiy Mazhelis, Fengyu Cong, Mikko Kauppila, Teemu Leppänen, Jouni Kantola, Jussi Collin, Susanna Pirttikangas, Janne Haverinen, Tapani Ristaniemi, and Jukka Riekki. 2011. Distributed Road Surface Condition Monitoring Using Mobile Phones. In *Ubiquitous Intelligence and Computing*, Ching-Hsien Hsu, Laurence T. Yang, Jianhua Ma and Chunsheng Zhu (eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 64–78.

[26] Richard Pettigrew. 2019. Aggregating incoherent agents who disagree. *Synthese* 196, 7 (July 2019), 2737–2776.

[27] Jacob Poushter. 2016. *Smartphone Ownership and Internet Usage Continues to Climb in Emerging Economies*. Pew Research Center, Washington, DC. Retrieved from https://www.pewresearch.org/global/2016/02/22/smartphone-ownership-and-internet-usage-continues-to-climb-in-emerging-economies/

[28] Roopesh Ranjan and Tilmann Gneiting. 2010. Combining probability forecasts. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72, 1 (2010), 71–91.

[29] Kevin Sahr, Denis White, and A. Jon Kimerling. 2003. Geodesic Discrete Global Grid Systems. *Cartography and Geographic Information Science* 30, 2 (January 2003), 121–134.

[30] Shahram Sattar, Songnian Li, and Michael Chapman. 2018. Road Surface Monitoring Using Smartphone Sensors:A Review. *Sensors* 18, (November 2018), 3845.

[31] Girisha D De Silva, Ravin S Perera, and Nayanajith M Laxman. Automated Pothole Detection System. 5.

[32] Iping Supriana, Dody Dharma, Dicky Satya, Dessi Satya, and Lestari. 2015. Geohash Index Based Spatial Data Model for Corporate.

[33] Kanat Tangwongsan, Martin Hirzel, Scott Schneider, and Kun-Lung Wu. 2015. General incremental sliding-window aggregation. *Proc. VLDB Endow.* 8, 7 (February 2015), 702–713.

[34] Transport Focus. 2017. *Road surface quality: what road users want from Highways England*. Transport Focus. Retrieved December 6, 2019 from https://www.transportfocus.org.uk/research-publications/publications/road-surface-quality-road-users-want-highways-england/

[35] United Nations Economic and Social Commission for Asia and the Pacific. 1993. *Asian highway classification and design standards*. United Nations Economic and Social Commission for Asia and the Pacific.

[36] Francis Delali Vorgbe. 2014. Classification of road surface quality using Android smartphone devices. Thesis. Ashesi University.