Web Application For Pollution-Aware Route Planning

Joud Asfari

***Abstract*—Air pollution is linked to adverse effects on respiratory health, cardiovascular health, and other chronic diseases. One way to reduce exposure is by planning better travel routes for individuals at risk. To address this, a web application for route planning using air pollution data has been developed. The web application has three main components; a database server, a service logic server that contains the search algorithm and pre-processing scripts, and a user interface that communicates with the service logic server. Air pollution readings were taken in the Greater Wellington region and the resulting spatiotemporal data mapped onto real geographic locations. User experience testing was conducted on participants studying engineering and computer science at Victoria University of Wellington. The paths found by the application are consistent and reproducible and allow users the ability to fine-tune searches to fit their needs and gain insights about air pollution in Wellington roads.**

***Index Terms*—Air pollution, Route-planning tools, Web applications, Lung health & wellness**

I. Introduction

AIR pollution has been linked to detrimental long-term impacts on respiratory health, cardiovascular health, and other chronic diseases, particularly in heavily polluted regions. For instance, in China, long-term exposure to $PM_{2.5}$ was predicted to account for 30.8 million deaths over 17 years [1]. Emission control technologies in China reduced air pollution-related mortality by 71%, avoiding 870,000 deaths [2]. In contrast, air quality in New Zealand is among the best internationally and is below dangerous levels, consistently being categorised as having “good” air quality daily in several international air indexes [3]. Despite air quality in New Zealand being significantly better than more polluted countries, 13,155 hospitalisations and an estimated 3,317 premature deaths in 2016 were linked to exposure to air pollutants [4]. The importance of reducing exposure to air pollution is growing as more reports demonstrate that atmospheric measurements of air are increasingly more polluted [5]. Awareness of air pollution is increasing, and researchers are discovering that there may be no safe threshold [6]. Improving the air quality of urban areas is essential to improving population health.

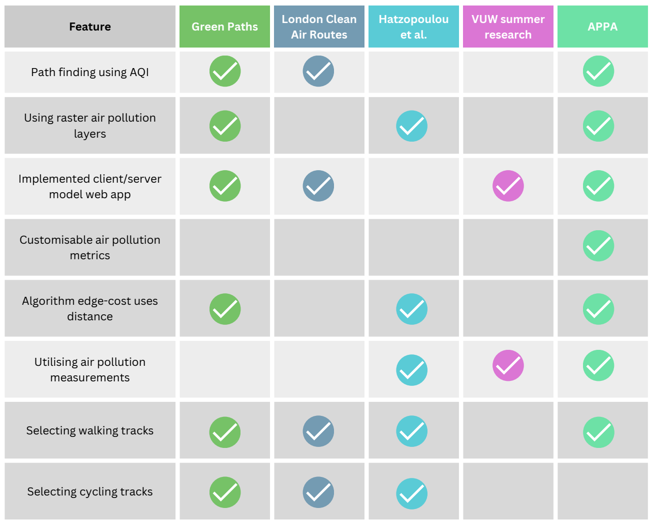
Urban areas are susceptible to pollution from construction, busy traffic, and other sources. People biking or walking in those areas would be exposed to air pollution at close proximity. People that are sensitive to air pollution might want to avoid areas where they may breathe in more irritants such as $PM_{2.5}$. Existing air pollution route planning tools are constrained, e.g., by regional limitations or limited data views [7] [8]. Due to the limited usage of existing route planning tools (with or without air pollution metrics), there is a gap between user needs and application functions. People with health issues are more restricted with the routes they can or cannot take, but that is not accounted for in most route planning applications. Air pollution data can be used to plan better paths for cycling, walking, and driving. Air pollution measurements can be effectively integrated into least-cost graph problems, as demonstrated by studies developing route planning methods to reduce car carbon footprints, and a few air pollution-based route planning tools developed in recent years that will be discussed later in this report.

This project builds upon work accomplished in 2022, which involved the development of a device for crowd-sourced air quality monitoring, enabling more precise air pollution measurements [9]. This technology was put to use during a summer research project, where pollution data was projected onto real roads in New Zealand, facilitating route planning based on air quality readings [10]. The present endeavour extends and enhances these previous projects. By introducing customisable search options, we are increasing individual-level route planning capabilities, ultimately fostering the well-being choices of individuals. Additionally, this extension addresses the system’s requirements for storing and accessing spatiotemporal information, contributing to improved overall system performance and efficiency. This project works towards the good health and well-being sustainable development goal, specifically target 3.9 which is focused on reducing mortality and illness rates caused by air pollution among others [11]. The development of this application will help communities with making better decisions for their health. It gives people in the public access to information about air quality which affects them in their daily lives and may indirectly help make cities more sustainable by raising awareness of air pollution near populated areas [12].

The outcome of this project is a scalable web application titled “Air Pollution Path-finding Application” (APPA), which allows users to search for customised routes using reliable air pollution data and visualising this information in an easily digestible way. Users of the application will be able to reduce their daily exposure to air pollution, thereby improving their long-term health. Deliverables include a user-friendly web interface, a spatiotemporal database, and an improved path-finding algorithm. The comprehensive evaluation of

This project was supervised by Professor Winston Seah, Dr. Jyoti Sahni, and Professor Colin Simpson from the Health Faculty at Victoria University of Wellington.

the application has demonstrated the effectiveness of the air pollution algorithm and its ability to provide valuable insights. The System Usability Scale (SUS) assessment, with a median score of 82.5, reflects the application's high level of user-friendliness. The project's choice of technology stack, comprising Node.js, OpenStreetMap, Leaflet, Nominatim API, PostgreSQL, PostGIS, and QGIS, played a pivotal role in facilitating development. The decision to leverage open-source technologies brought a multitude of benefits to the project. Notably, the abundance of learning resources and a thriving developer community provided a rich and dynamic environment for skill development and problem-solving. Furthermore, the inclusion of specialized software and tools for Geographic Information Systems (GIS) proved instrumental in handling geographical data efficiently. These resources streamlined the management and analysis of complex geospatial information, reinforcing the project's ability to process and present data accurately.



Feature	Green Paths	London Clean Air Routes	Hatzopoulou et al.	VUW summer research	APPA
Path finding using AQI	✓	✓			✓
Using raster air pollution layers	✓		✓		✓
Implemented client/server model web app	✓	✓		✓	✓
Customisable air pollution metrics					✓
Algorithm edge-cost uses distance	✓		✓		✓
Utilising air pollution measurements			✓	✓	✓
Selecting walking tracks	✓	✓	✓		✓
Selecting cycling tracks	✓	✓	✓		

Fig. 1. Comparison between this project (APPA) and related work.

A. Terminology

- **Geocoding:** Providing geographical coordinates associated with a location. Similarly, reverse geocoding is providing a location associated with geographical coordinates.
- **Geographical information systems:** A spatial system that creates, manages, analyzes, and maps all types of data, as defined by Esri [13].
- **Geospatial:** Information that is associated with a particular geographic location.
- **Raster:** As opposed to vector images or data, rasters are represented by individual pixels. We also use the definition used by QGIS which is that rasters are made up of a matrix of pixels containing values that represents the conditions for the area covered by each pixel [14].
- **Route:** Used interchangeably with path. The way or path that is found.
- **Spatiotemporal:** Data that changes over space and time. In this context we mean geospatial and time-series data.

II. Related work

Studies have demonstrated that alternative route planning methods can be developed using air pollution data in least-cost graph problems. One such study was done by Helle et al. from Helsinki University [8]. A web application was developed that allows users to choose one of three environment variables to use for finding an optimal path. The study uses noise, greenery, and air pollution data from the Helsinki Metropolitan area provided by the Finnish Meteorological Institute and the Helsinki Region Environmental Services Authority. A similar tool to the Helsinki web application is the London Clean Air Routes web application, which finds the paths with the least air pollution in London [15]. Another study is an ArcGIS-based route planning tool made to reduce cyclists' exposure to NO_2 in Montreal [7].

The Helsinki web app uses raster images of geospatial air quality data, which is a pixel grid filled with air quality data. The air quality information uses a pre-processed air quality index (AQI) and does not contain any raw pollution information. The AQI covers NO_2, SO_2, O_3, $PM_{2.5}$ and PM_{10}. The Montreal study also uses a raster image, but the layer contains information about average ambient concentrations of NO_2 per road segment. The NO_2 concentrations were calculated from a land use regression model (LUR) that was based on a series of data samplings in 133 locations in Montreal. Both studies use a distance-based least-cost search and build on it to add sensitivity to air pollution.

$$C_e = C_t + C_t * c_e * s \tag{1}$$

$$C_e = C_L * c_e \tag{2}$$

Equation 1 from the Helsinki study calculates the edge cost using the travel time, and multiples this base cost with the environmental cost and an arbitrary sensitivity coefficient. The environmental cost depends on the environmental variable that was chosen by the user for the path-finding search. As shown in equation 2 from the Montreal study, the edge cost is calculated as the product of the edge segment's length and the environmental cost. The environmental cost in the Montreal study is the average concentration of NO_2 in the segment.

As shown in figure 1, our system aims to allow an increased amount of customisation in the paths found- users can view raw pollution data (not just a calculated AQI) and choose which metrics affect them the most and prioritise paths based on those metrics. The path-finding search is also done based on multiple environmental variables in one search, which is not something implemented in previous solutions. This can be achieved by normalising pollution data to apply the cost of environmental variables equally in the total edge cost equation. Path customisation options and the path-finding algorithm are further discussed in the design and implementation sections.

III. Design

A. Requirements

The project is subject to a set of requirements encompassing application functions in terms of capabilities, conditions, and constraints. The capabilities of the application necessitate the ability to find the optimal path between two specified points, utilising metrics defined by the user, provide insights on air pollution based on user input, and present the results in a comprehensible format. In line with the specified conditions, the application must rely on collected or generated air pollution geospatial data to inform insights, and it must be developed as a web application, ensuring accessibility and usability across a diverse range of platforms.

It is vital to acknowledge the constraints associated with open-source geospatial data, which is continually evolving and may contain gaps, and the time limitations imposed on collecting real air pollution data for comprehensive testing, which may require the generation of data to meet the project's requirements. The scarcity of air pollution data was expected to be a major issue in the development and testing of the software. The software was thus required to handle situations where geospatial areas are missing data and respond in a fail-safe manner.

In addition to the application's functional requirements are its non-functional requirements. The application must be scalable in terms of its performance and presentation to larger road networks, larger sets of air pollution metrics, and a larger number of users. It should be accessible to users with basic digital literacy, allowing anyone in society with technological familiarity to use the application for improved health outcomes, even if they have no prior knowledge of air pollution.

B. Architecture

The web application has three main components; a database server, a service logic server that contains the search algorithm and pre-processing scripts, and a user interface that communicates with the service logic server. Figure 2 shows the full high-level system architecture. The database stores all pollution and road data. The system uses specialised open source GIS (geographic information systems) software for the pre-processing of data as well as scripts written specifically for this web application. The system also uses map APIs at the user interface to get map visualisation and interaction functionalities, map information such as raster layers, and geocoding which maps geographic coordinates to real-life addressing systems.

C. Data storage and usage

The air pollution data collected is spatiotemporal. Collected data are discrete, with each pollution reading representing a point where data was collected. The scarcity of air pollution data was expected to be a major issue in the development and testing of the software. The software should be able to handle situations where geospatial areas are missing data and respond in a fail-safe manner, or ideally, have fewer scenarios to fail. Spatial air pollution data is not constant as areas may have different levels of air pollution over time and external factors may completely change the results in an area. As such, for any use of the system after project completion, the database will need to be maintained with new air pollution data, whether static or real-time. To address these issues, a pre-processing method was developed to assist with any future database updates that uses previous database scripts and new scripts utilising APIs made available by specialised open source GIS software. This is discussed further in the implementation section.

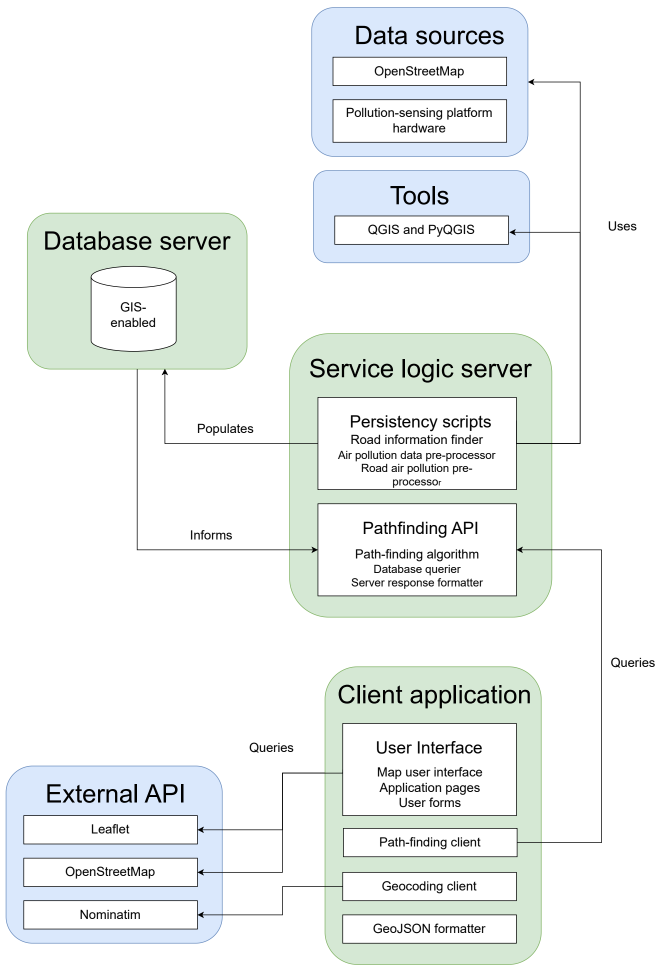
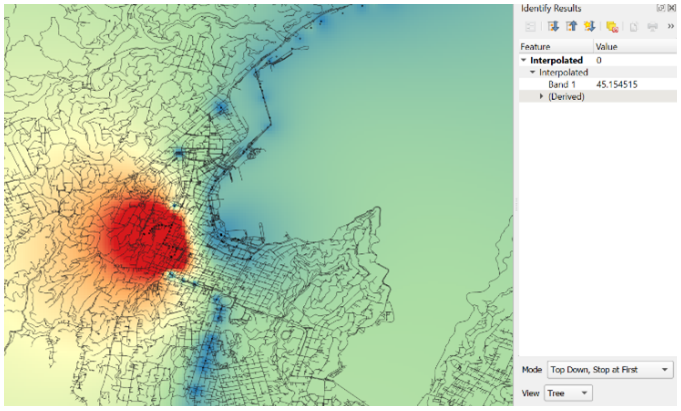


Fig. 2. System diagram for the application.

Air pollution data that has been collected is sparse and spread out unevenly around the Wellington region. Therefore, a spatial analysis method was required to fill in missing data points for effective path finding. The spatial analysis method chosen was inverse distance weighed (IDW) interpolation due to it being an interpolation technique proven usable for aquiring air pollution information at unsampled locations [16].


Identify Results

Feature	Value
Interpolated	0
Interpolated	
Band 1	45.154515
(Derived)	

Mode: Top Down, Stop at First

View: Tree

Fig. 3. An example of a raster layer of interpolated air pollution values. This image uses test nitrogen dioxide data.

Every air pollution metric was used individually to create a raster layer using IDW interpolation. Every air pollution metric was interpolated to create a raster layer, an example is shown in figure 3. Other interpolation techniques could have been used such as Kriging, but would have required further in-depth spatial analysis.

One drawback of the interpolation method is that real raw air pollution measurements are lost in processing due to the interpolation resolution being too low to allow exact geographical coordinates to have their own measurements. For example, a measurement at (174.826583, -41.169559) longitude and latitude could be 14.7 micrograms per cubic meter, but after interpolation the value found from the raster may be 13.7 micrograms per cubic meter. Improving the accuracy of interpolated values requires more spread-out measurement of air pollution and a higher interpolation resolution. The latter is not ideal as it may negatively affect the performance of database queries if a raster image is used.

Geospatial databases use geographic shapes and objects embedded in the system. This approach mitigates scalability limitations exhibited in rudimentary implementations of geospatial databases that negatively impact both query performance and database indexing as more entries are added to the database. A geographic information system (GIS) extension can simplify the DB design and also give the DB extended geospatial operation capabilities that other DBMSs do not have [17]. As such, the database was designed with a GIS extension in mind. This is shown in figure 4. The database contains a table for processed air pollution data, and the location column is geometric. The edges table contains road data information. Using this table we are able to visualise the road network topology in figure 5 using GIS software such as ArcGIS or QGIS.

D. Path-finding algorithm

A suitable algorithm for this project would consider the distance vector as well as air quality and time. Finding the best solution in such cases requires considering various factors simultaneously. Multi-criteria decision-making algorithms

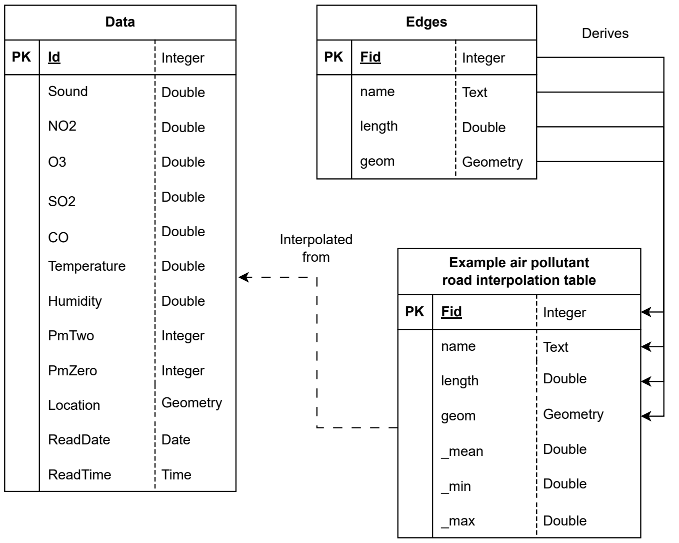


Fig. 4. System database with example air pollution table. The air pollutant tables have identical columns but different values.

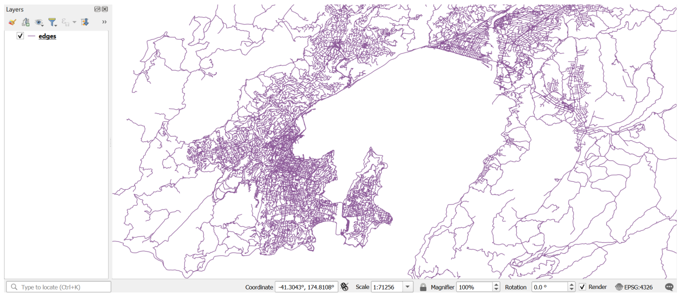


Fig. 5. Road network topology extracted from the Edges table of the geospatial database and into QGIS.

are particularly useful when a compromise solution rather than an optimal one is sought. Figures 6 and 7 illustrate that air pollution lacks a direct correlation. Consequently, any solution reached would represent a compromise, known as a Pareto-optimal solution [18]. Moreover, the inclusion of the distance vector and timestamps further complicates the search for an optimal solution. Various path-finding algorithms, such as multi-criteria decision algorithms, or Dijkstra's algorithm with optimizations for Pareto-optimal searches, can be applied [19] [20].

Multi-criteria algorithms have higher memory and time complexity needs. A generalised approach to the path-finding algorithm fits the project requirements, as long as a solution is found that has a minimal total cost. Therefore, an extended implementation of Dijkstra's algorithm can be used with air pollution coefficients and weights and a distance vector per edge cost, similar to the path-finding algorithm in the Helsinki application [8].

The path-finding algorithm builds on an implementation of Dijkstra's algorithm which uses air pollution data calculated per route as an edge cost. The algorithm can also be built on A*, which is an optimisation of Dijkstra's

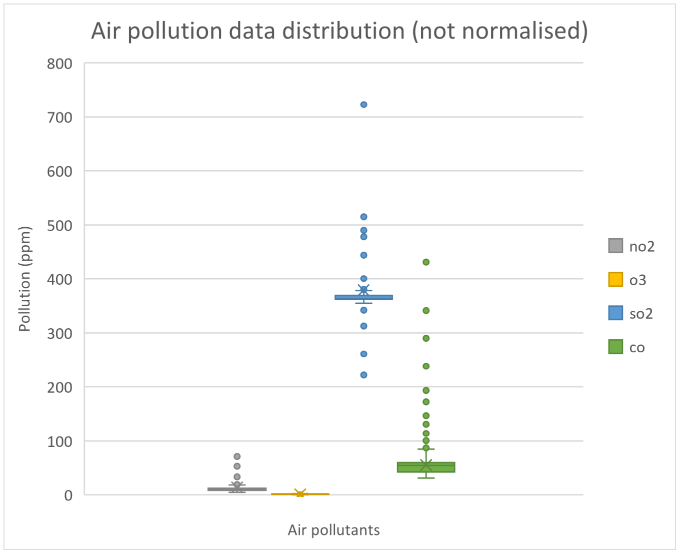


Fig. 6. Air pollution measurements before normalisation across different spatiotemporal points.

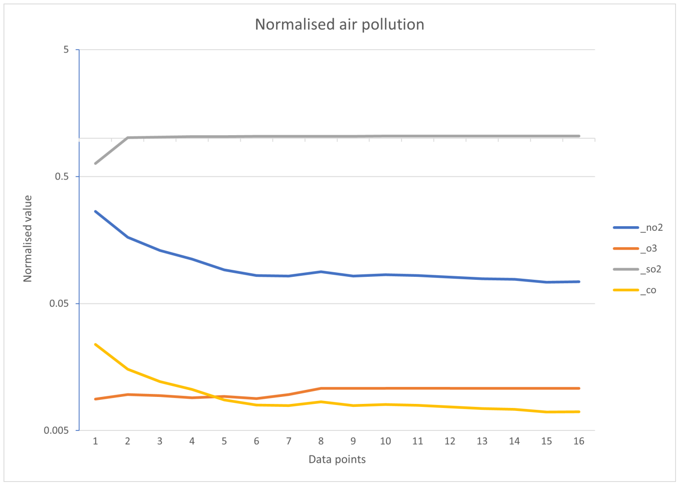


Fig. 7. Air pollution measurements after normalisation with a smaller set of spatiotemporal points.

algorithm, however the simplicity of Dijkstra's algorithm is sufficient to prove the efficacy of the air pollution cost estimation function. The search is a graph problem, where the graph is made up of nodes and edges. Nodes on the graph represent the intersections or ends of road segments on the real-life road network. Edges represent the road segments and retain information such as the road name, connections, and pre-processed air pollution data. However, the edge information saved on the database does not include distance information as the original goal was to find the least-cost path where the cost is the air pollution in an area.

The improved algorithm includes using user-defined weights for different air pollution metrics and using the time stamps of the data. The algorithm also uses distance in the edge costs to find the shortest path when there is no variation in air pollution levels, and to prevent the search from finding paths too far away from the goal- which in a real-life scenario might mean the software suggests a path going through a far away rural area instead of a short 2-minute walk through a polluted area. The calculation of air pollution for each edge was designed to be pre-computed in the database before the server was started to avoid unnecessary volumes of calculations at every run of the algorithm.

This was achieved by adding weight costs to the original cost estimation computation.

$$C_p = \sum_{i=1}^{n} x_i * w_i \tag{3}$$

$$C_e = d * s_1 + C_p * s_2 \tag{4}$$

Where w_i is the air pollution's user-defined weighting at i of the air pollution metrics. C_p is the air pollution cost for the edge, and C_e is the total edge cost where d is the distance vector and s is the cost's sensitivity to distance (s_1) and air pollution (s_2). Doing this ensures that in the case of the user choosing to search for an optimal path in an area with no variation in pollution levels, the algorithm would still find a solution which would be based on the distance.

Air quality in New Zealand is based on NES-AQ the 2004 legislative act that specifies legally binding air pollution limits that local councils must abide by [21]. There are also other standards which can be followed such as Australian standards, EPA, and WHO standards. To calculate air quality, the air pollution measurement is divided by the national standard and multiplied by 100. This calculation normalises the data and allows comparison between the values of different pollutants which may otherwise be too varied in expected and unhealthy ranges to be used in a heuristic. The normalisation ensures that every pollution metric affects the cost equally if all pollution weights (w_i) are equal to 1. To determine the band of air quality for the spatiotemporal point, several techniques can be followed depending on the country's approach. For example, in Australia the air metric with the highest value is the one that decides the band of air quality for the spatiotemporal point [22]. Different approaches to air quality measurement worldwide would make it more difficult down the line to scale the solution internationally.

IV. Implementation

A. Database

The database employed is a relational database, and PostgreSQL was selected for our purposes over other databases such as SQL Server. That is because PostgreSQL is open-source and provides a variety of features at no cost. The database uses the PostGIS extension which is a geospatial extension for PostgreSQL that enhances its capabilities for managing and analyzing geographic data. PostGIS enables the database to store, query, and manipulate geospatial information, making it a valuable asset for projects involving geographic data. Its benefits include support for various geographic data types, advanced geospatial functions, and the ability to perform spatial queries and analysis. In

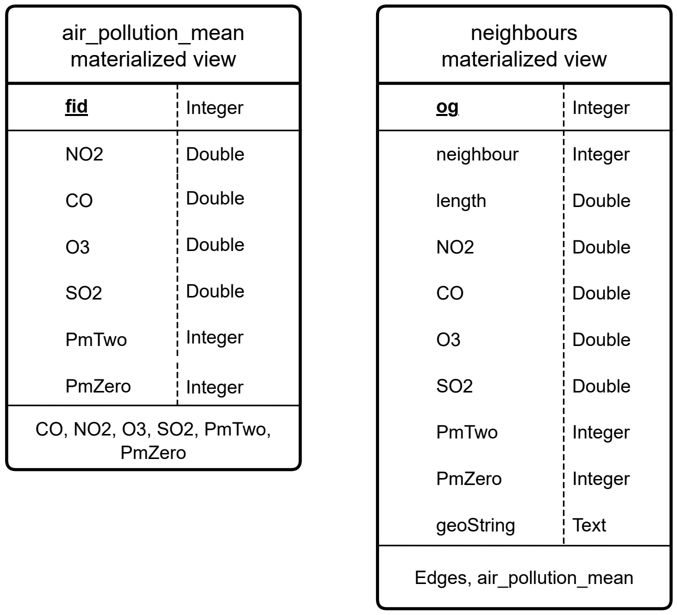


Fig. 8. Materialized views were used to optimise expensive SQL queries.

our context, PostGIS is an indispensable solution as it allows us to handle the geographic aspects of our data effectively, particularly when dealing with air pollution and geographical point references. It is also supported by many GIS software, some of which are open-source, such as QGIS, and have thus been used in the course of this project. The relationship between the entities in the database are dependent on the geographic information.

The "data" table shown in figure 4 within the database encompasses the initial air pollution readings recorded at specific geographic points, with each data point assigned a unique identifier. To maintain data integrity and prevent redundancy, a unique constraint is imposed upon the attributes pertaining to location, date, and time. The "location" attribute pertains to the geospatial point denoted by a singular set of latitude and longitude coordinates at which the air pollution measurement was acquired. Meanwhile, the "readdate" and "readtime" attributes correspond to the respective date and time of the air pollution reading. The attributes "no2" "so2" "co" "pm2" "pm10" "temp" "humidity" and "sound" each represent distinct measurements of air pollution measured by the device used for data collection during this project. The "edges" table is generated through the execution of the $generategpkg.py$ script, which serves the purpose of importing geospatial data from OpenStreetMap. The script extracts road-related information from OpenStreetMap and packages it into a geopackage format. This format can be easily incorporated into the geospatial database, thereby simplifying the integration of essential road network details within the database.

The database is populated with road network information only once, and otherwise manually for maintaining relevance for the network. Other than that, the database is populated with air pollution data by database administrator users and maintained by the persistency scripts in the service logic server of the application.

B. Service logic server

The logic server can be accessed by the frontend client via a Node.js API and receives geographical information as well as user-defined search options. The options received from the client are:

- Distance sensitivity: Refers to the coefficients that should be used in the search algorithm's heuristic for distance and air pollution metrics, respectively.
- Pollution settings: Refers to the coefficients used for each individual air pollution metric.
- Source latitude: Is the latitude of the search's geographical starting point.
- Source longitude: Is the longitude of the search's geographical starting point.
- Destination latitude: Is the latitude of the search's geographical ending point.
- Destination longitude: Is the longitude of the search's geographical ending point.

The service logic can split into two parts: the path-finding API, and the persistency scripts. The path-finding API is a RESTful service that is exposed to client applications. The persistency scripts contain logic required to keep data up-to-date. As the air pollution data used in this project are geographic points with air pollution metrics associated with them, the persistency scripts run through a process of interpolating and associating the data with the roads in the database in order to be usable for the path-finding search.

1) Path-finding API: The path-finding algorithm is included in the path-finding API and is called when the server processes a client request. The API is built with Node.jS which starts an http server. When the service receives a client request, an asynchronous function is called. Asynchronous functions in the Node event loop are non-blocking, so in principle the API can accept and process multiple requests. This level of concurrency is essential to allow the program to continue while I/O functions (such as querying a database) are run. It is also a stepping stone to multi-threading solutions.

The server processes the request and calls the search algorithm with the relevant information (such as the search's geographical coordinates, search mode, and user-defined air pollution constants). The search begins by finding the closest roads from the road network to the start and end points defined in the request. A query to the database is made for every coordinate that is searched. After that, the program searches for all of the "neighbours" or connected roads to the closest road that was found to the starting point. At every point, the air pollution for each of those roads is also queried from the database. Due to this information being contained in different tables in the database, SQL joins are needed as well as other expensive operations in PostGIS such as *st_touches* which finds the geometric intersection between two geometries. Once the base information for the search is found, a recursive function is called to run Dijkstra's algorithm with the path-finding heuristics we have defined. At every run, the program will query the database to find the "visited" (currently being searched) road's neighbours. This is done to avoid loading the entire road topology into the program, which would take up large amounts of memory especially as information required per road increases as the number of air pollution values monitored increases.

When the search finishes, the service packages the information of the roads, their coordinates, air pollution, calculated score, and health category into a JSON. The JSON is then sent as a response to the client request and the database connection is closed until the next request.

Due to the algorithm querying the database at a high volume and with queries that have high planning and execution costs, alternative SQL queries had to be explored. Two ways that we have chosen to optimise our queries were to create materialized views in the database to pre-compute results that are needed for a search. For example, as seen in figure 8, we have utilised a materialized view for joining interpolated and analysed air pollution data into one accessible place due to the query containing 5 join operations. The other materialized view used holds the results of neighbouring roads and air pollution data for every road in the network. Another optimisation is using an index on the "geom" column of the edges table in the database. The index used is a general index structure (GIST) which can be used for indexing custom types of data such as geometry defined in PostGIS [23]. The spatial index allows faster queries to occur by avoiding sequential scans in the query tree, thereby speeding up individual queries which adds up to a faster path search.

2) Persistency scripts: The utilisation of raster images in the context of air pollution analysis introduces a degree of complexity. The selected Database Management System (DBMS) for this project is PostgreSQL extended with PostGIS. PostGIS is a GIS extension and one of its features allows it to facilitate the retrieval of data values from raster images that have been integrated into the database. It should be noted that the same approach is not feasible when dealing with geographical lines, such as in our road network. As a consequence, the need arises to decide on specific coordinates along the road network to serve as representative indicators of air quality. It is imperative that these chosen points remain consistent across multiple runs to ensure the reproducibility of results, particularly when the air pollution data in the database remains unaltered.

For this purpose, the raster images generated through the interpolation of air pollution metrics were subjected to a zonal statistics analysis within QGIS. Zonal statistics analysis, in this context, refers to the process of extracting statistical information from areas or zones defined by polygons. In this case, the road layer was initially transformed into polygon representations where each line representing a road was buffered with a 100m radius to produce the polygon. Zonal statistics analysis was performed on each air pollution raster image using these road polygons as reference zones.

The outcome of this analysis yields a polygon layer in which each road segment has its own mean, minimum, and maximum air pollution values, providing the ability to query and obtain results that are consistent across multiple runs. It is important to acknowledge that this method does not distinguish between short and long road segments, and is wholly dependent on the road network topology being reliable. Consequently, longer roads may exhibit a broader range of air pollution values which may not entirely represent the air quality as a mean within the surrounding area.

C. Frontend

The frontend of the application is client-facing and displays information to allow users to interact with the software application. A Leaflet interactive raster map was employed to display interactive maps on the web interface, and the raster layers were retrieved from OpenStreetMap [24]. The user interface was made using React MUI, which is a React library containing user-friendly reactive components using the Material-UI which are Google's user interface design guidelines. Actions in the program are event-based, meaning if a user clicks something on the UI then an event is called and information updates are made. Leaflet itself has user events which were utilised in this project for a seamless user experience. When a user clicks on the interactive map, a latitude and longitude are retrieved. The geographical coordinates can be converted into street addresses using reverse-geocoding, and an API was used to do that in this application. React contexts were used to keep track of program states that were required in several parts of the application.

One of the features of Leaflet that were utilised in the implementation of the user interface is its layered system. Everything that moves in the map view of Leaflet is considered a layer, including the raster tiles making up the map images themselves. Leaflet layers were used to add and manipulate map markers as well as draw the resulting paths received from the service logic server. Leaflet can draw geometry on the map given a GeoJSON, which is a JSON format to represent geographical features. The information received from the server contains geographical information as well as information required to display air pollution data, categories, and other information that was used in the path-finding algorithm which resulted in the found path to be chosen as the optimal path. Consequently, handling the received format and converting it into a displayable format was the responsibility of the frontend application.

The GeoJSON Leaflet layers could have been implemented in a few ways. One way is to create a GeoJSON for every complete path, using MultiLinestring geometry, and adding that as one layer to Leaflet to show every section of the path as part of one whole rather than individual pieces. In a MultiLinestring, a path is represented as a single, continuous entity composed of multiple line segments. This would mean that any click events added to the layer would work on the path as a whole rather than any individual sections of it. The other way is to create a GeoJSON for every section of a complete path, using Linestring geomtry which would display not only the path but also the different sections of the route. In the case of Linestring geometry, each line represents a single line segment between two points. This would allow every section added as a Leaflet layer to have its own click events and thus be selected individually to perform actions on. Because of this, a GeoJSON object is made for every section of the route so that in future iterations of the application users will be able to click on sections of the route and get information about them.

V. Evaluation

To evaluate the application, we chose to test the usability of the frontend, and the resulting paths found by the path-finding algorithm. The usability testing was designed to help analyse how the system adheres to user experience (UX) design principles and how real users would use the system, and if there are any improvements that can be made which make it more accessible. Heuristic evaluations typically involve usability experts or evaluators who assess the user interface against a set of predefined heuristics or guidelines. While heuristic evaluations can provide valuable insights, they are not always representative of how the system will be used by real users. The decision to skip a heuristic evaluation, in favor of usability testing with actual users, is based on the recognition that the primary target users of the web application are the general public, and these users may not possess technical or design expertise.

A. Usability testing

For the usability testing, participants were asked to complete questionnaires with Likert scale questions for a System Usability Scale (SUS) along with some open and closed-ended questions. SUS was chosen because of its ease of use and standardisation in user experience testing [25]. Questions were posed to users to gauge how well the current application meets the needs of potential users. The user testing was undergone on-premises and performed on 10 participants. The testing was moderated to record observations of user behavioural responses to usability tasks.

Figure 10 illustrates how the application was scored according to SUS. Most users rated the application above 68, which is the SUS score studies use to indicate whether a system is above or below average [26]. There is one outlier which gave responses that yielded a score of 65, otherwise responses were overwhelmingly positive. Some students commented on UI features that could be improved, such as the application sidebar, marker popups, and the behaviour of markers when they are clicked. A few application bugs were noted to be fixed in the future.

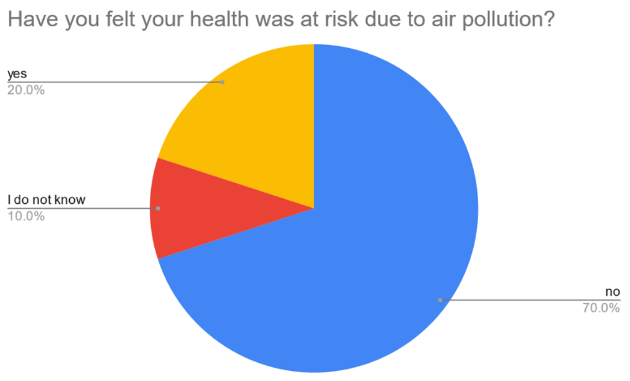


Fig. 9. Percentage of user testing participants who felt their health was at risk due to air pollution.

From the questionnaire provided and while answering the usability tasks, most students said that they would normally not care about air pollution and would rather take shorter or faster paths to get to their destination. Some justified that by saying that their life activities may require prioritising time-efficiency. This aligns with the results from figure 9, where 70% of participants indicate that they do not feel that air pollution has put their health at risk. A few students pointed out that a path with less air pollution is not guaranteed to be a healthier option, and being exposed to air pollution for longer may be more problematic than going through a highly polluted area for a shorter period of time. Overall, users were receptive and seemed willing to use the application, with some users even continuing to play around with the application even after the user testing was over.

Participants were also asked to fill out general survey questions to gauge the demographic that is being worked with. Among the questions asked were what devices they normally use to access maps, and if they have felt their health was at risk due to air pollution. Results in figure 11 show that all of the participants use phones to access maps, and only 1 participant uses a computer or other device. This is an important finding because the current web application is designed to work on wide-screen displays such as in desktop browsers. In order to continue addressing user needs, the web application would need to be modified to also work with phone displays.

B. Path-finding results

To evaluate the effectiveness of the path-finding results, a comparison is made across the three available search modes within the application. This comparative analysis serves the purpose of showcasing the operational logic and user interface outcomes, as well as highlighting the distinctions between a pollution-aware search and one that solely considers distance. In the distance-only search mode,

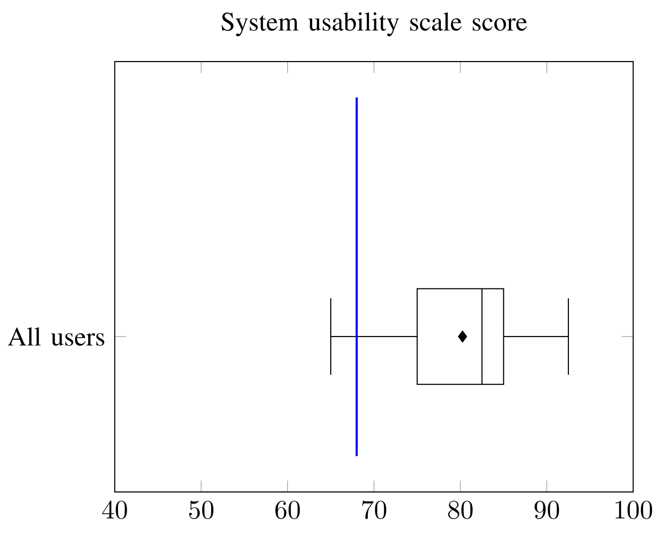


Fig. 10. System usability score calculated from user testing participant responses.

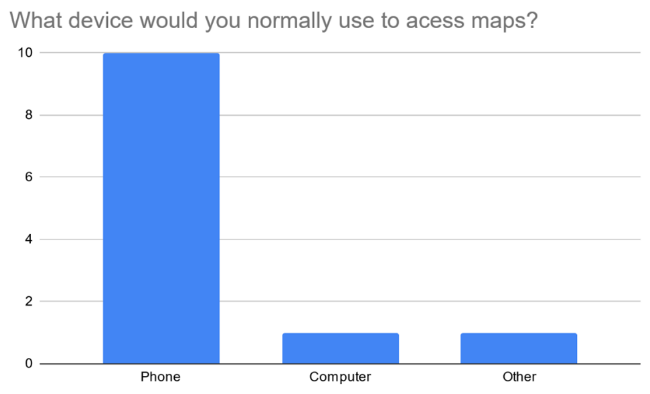


Fig. 11. Diagram showing user testing participant chosen devices to access maps.

the air pollution's contribution to the edge cost in equation 4 is given a coefficient of 0, while the distance is weighted with a coefficient of 1. On the other hand, the distance-and-pollution mode assigns a coefficient of 1 to both components of the sum in equation 4. In contrast, the pollution-only search attributes a coefficient of 0 to the distance, effectively ignoring it, while placing sole emphasis on the air pollution cost with a coefficient of 1. The duration required for a search to reach completion is measured by the service logic server. At each incoming request, the server records the console time, which subsequently forms the basis for comparison of search speeds. Most importantly, the air pollution on paths found is compared using the application user interface, which displays air pollution results on every path using information curated by the service logic server. This is to evaluate whether the search algorithm presented is an appropriate solution for pollution-aware route-planning.

1) Martin Square to 1A Wallace Street: The first path searched is using the distance-only mode and is from Martin Square to 1A Wallace St. The starting point is denoted with a red marker, and the destination is represented by a white circle with a black outline. This is one of the source and destination combinations found during testing that resulted in interesting outcomes. The shortest path found by the algorithm between the two points is shown in figure 12. The server took around 6 seconds to return the result, but if we swap the start and destination points the server takes around 1 second to return. This is interesting because while the result appears to be the same, it indicates that the algorithm can go through more options depending on where the search starts. This aligns with our expectations because no part of the road network is uniform, but it does pose a question of whether the search speed and not just the result need to be consistent.

The second path searched is from the same starting and destination points as the first one, but instead uses the distance-and-pollution search mode. When Martin Square is used as a starting point, we can get the result shown in figure 13 in around 2 seconds, which is about a second longer than the distance-only search. Swapping the start and destination for this search mode, however, does not show the same result. Instead of identifying a consistent and replicable path analogous to the one depicted in Figure 13, the server's response corresponds to a path similar to that shown in Figure 12. This path represents the shortest route shown in figure 12 and takes around 4 seconds to be returned from the server. This inconsistency contingent on the starting point is attributed to the nuances of the edge costs we have assigned in equation 4 of the design section. These edge costs account for both distance and pollution factors and drive the variability in path selection based on the chosen origin, introducing an element of path preference in the search outcomes.

The air pollution for the first path is as shown in figure 16 poor and contains road sections which are categorised as "very poor". Sections are only categorised by the service logic server as "very poor" if they exceed national limits. The air pollution for the second path shown in figure 17 is similarly categorised as poor and contains sections which are "very poor" despite taking air pollution into consideration in the search. However, the level of pollution is lower, as can be seen from the lower maximum pollution. If we consider the number representing the maximum air pollution as an air quality index for the application, both paths found come close to being categorised as "very poor" due to the extent of the air pollution on the paths but one is better by 336.35 which is a significant difference.

2) Victoria University of Wellington to 46 Devon Street: The third path searched is using a distance-and-pollution search and is from Victoria University of Wellington to 46 Devon St. The server takes around 600ms to return the result. As depicted in Figure 14, it is noteworthy that the path remains consistent even when inverting the start and destination points. The path is the same when the search mode is distance-only, demonstrating that the edge costs for the air pollution for this path did not introduce a significant change to the path chosen as the optimal one.

The fourth path uses a pollution-only search, and is one of few routes than can be found for this search mode for reasons that will be discussed below. The server surprisingly takes only 400ms to find the path shown in figure 15 despite it finding more road sections. More good categories counted in figure 19 means path consists of more road sections, in total 50 for this one. In an ideal scenario we can estimate 51 queries to the database were made; first two to find the start and end edges, then 49 requests to find neighbouring roads. We can tell that the pollution-only search went through more road sections than the distance-and-pollution search on the same coordinates because figure 18 shows only 7 sections being categorised. The pollution-only search is potentially a healthier alternative to the distance-and-pollution search by a very small margin, as it has a lower maximum air pollution on the route by 0.85.

The reason this pollution-only search is one of few that can be found is due to it being a mode of search that is far less limited than the others in that it's only ending condition is finding a path that has less air pollution. In traditional path-finding algorithms, the search process generally halts once the shortest or most efficient route in terms of physical distance is identified. However, the pollution-only search is unbound by this conventional constraint, enabling it to explore and evaluate alternative routes that may be longer in distance but substantially reduce exposure to air pollution. That is the goal of the pollution-only search, but it is also its biggest drawback, as the search can continue indefinitely searching in the road network. It's important to acknowledge that real-world road networks are vast and, while not technically boundless, they are extraordinarily extensive. This means that the server continues the search until it hits a resource limit and then has to stop. A resource limit most of the time is a database connection timeout, but it can also manifest in the form of program memory consumption exceeding capacity.

VI. Future work

In the context of improving the current system, there are some clear directions that can explored in the future to make the application work better and address important issues discussed in the evaluation of the project.

The application is currently reliant on data that is collected manually via a mobile pollution measurement device. One of the limitations discussed and mitigated in the design and development of the software was dealing with the limitations of using data from this device. Due to the data being collected manually, the measurements are taken irregularly in small areas and time intervals that would not allow for accurate air pollution analysis. The mitigation was to use the data and interpolate it, which covered unknown areas with estimated measurements. However, this method in itself is limited, as interpolations require data that is sparse to be able to fill in

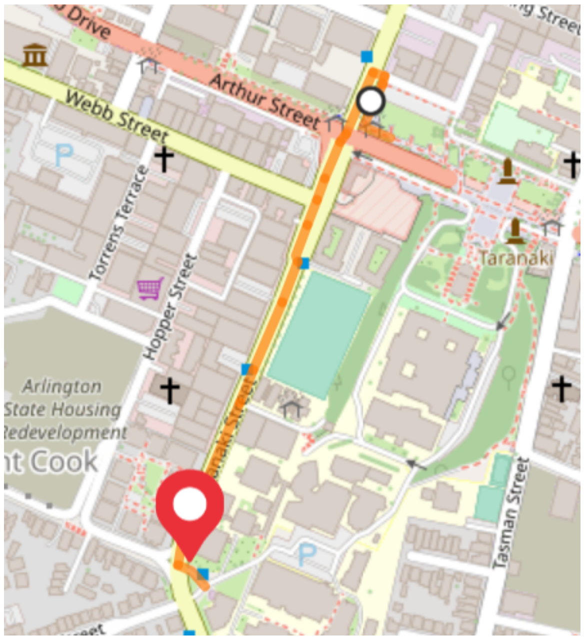


Fig. 12. Result of a distance-only search.

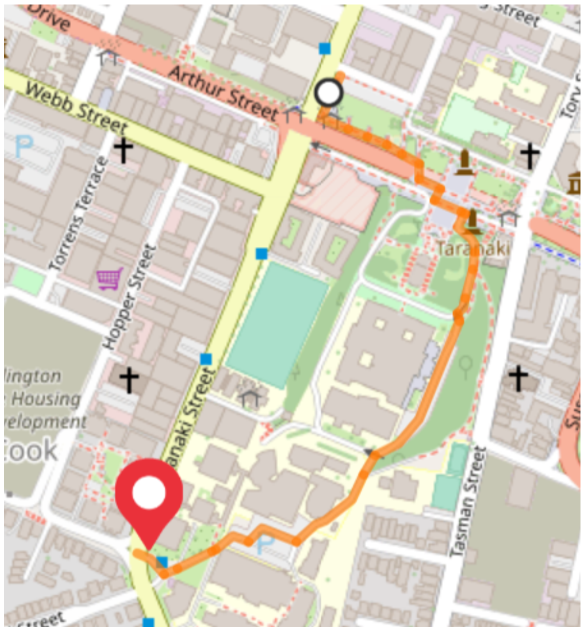


Fig. 13. Result of a search considering distance and air pollution.

unknowns more accurately without over-fitting to the existing data as can be seen in figure 3.

To address this, alternative data collection methods can be explored such as through using atmospheric remote sensors such as the Copernicus ESA Sentinel-5p sattelite which monitors various air measurements globally [27]. There are also NASA satellites which provide some open-source data for air pollution measurements [28]. It should be noted that for any remote sensing that is used, the the data should be taken frequently and have a granular enough resolution to allow for accurate interpolations. Finding a

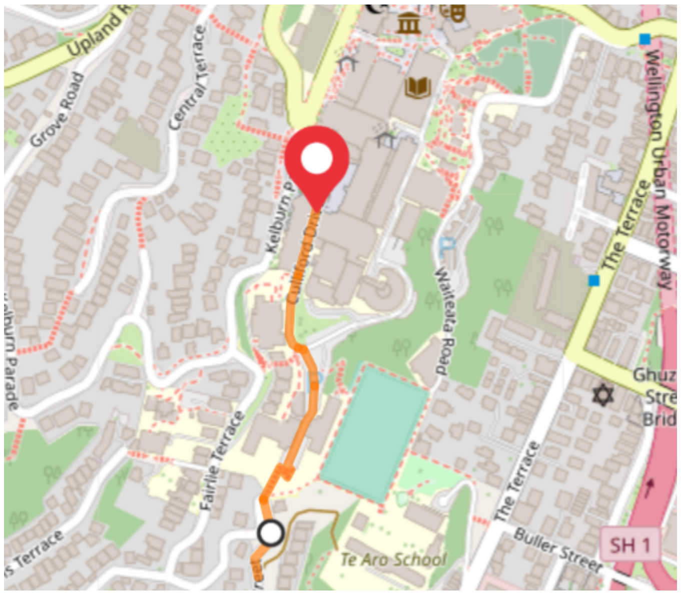


Fig. 14. Result of a distance and air pollution search, which is identical to the result of a distance-only search. Search completes in 580.33ms.

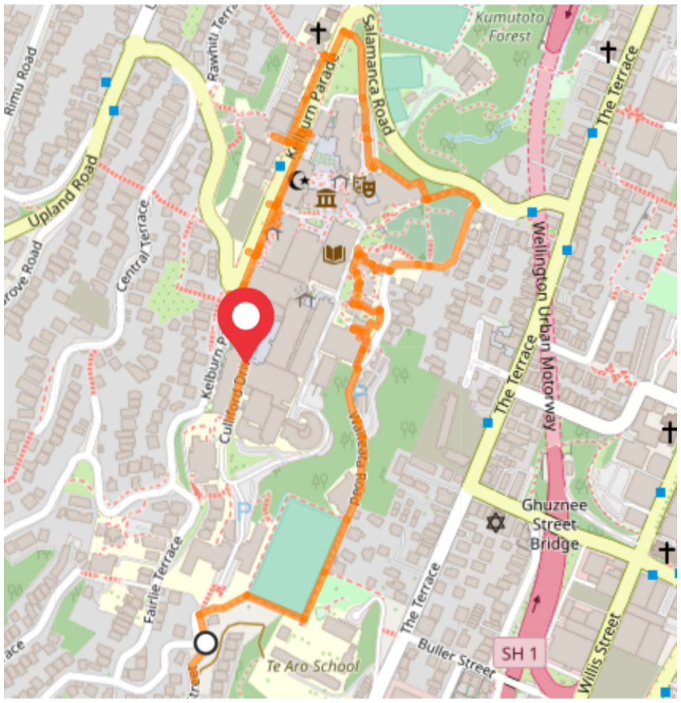
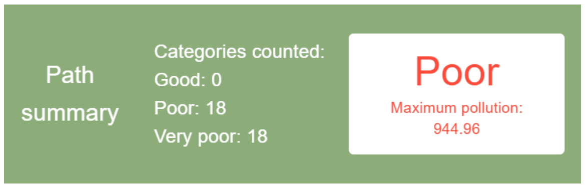


Fig. 15. Result of a pollution-only search. Search completes in 395.88ms.

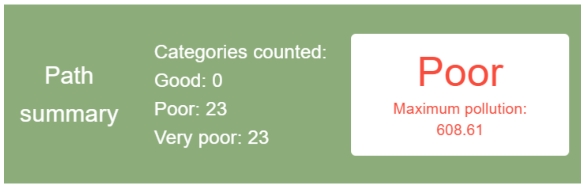

Path summary

Categories counted:
Good: 0
Poor: 18
Very poor: 18

Poor
Maximum pollution:
944.96

Fig. 16. The recorded air quality of the path found in figure 12.

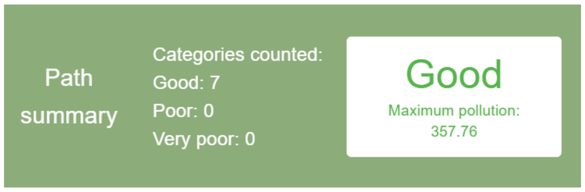
New Zealand-specific dataset would also be essential to maintain this project in a New Zealand context. Notably, the


Path summary

Categories counted:
Good: 0
Poor: 23
Very poor: 23

Poor
Maximum pollution:
608.61

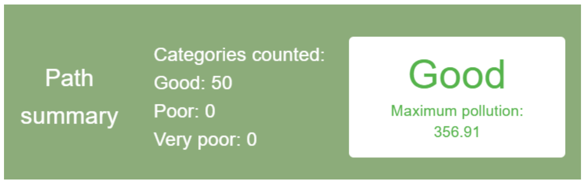
Fig. 17. The recorded air quality of the path found in figure 13.


Path summary

Categories counted:
Good: 7
Poor: 0
Very poor: 0

Good
Maximum pollution:
357.76

Fig. 18. The recorded air quality of the path found in figure 14.


Path summary

Categories counted:
Good: 50
Poor: 0
Very poor: 0

Good
Maximum pollution:
356.91

Fig. 19. The recorded air quality of the path found in figure 15.

search for open data from the National Institute of Water and Atmospheric Research (NIWA) is of paramount importance, given the organization's involvement in climate, atmospheric, and hydrological research. This pursuit holds the potential to bolster the accuracy and applicability of air pollution data within New Zealand.

The current system predominantly relies on a combination of air pollution data and distance metrics for path selection. However, the existing method does not provide a comprehensive understanding of users' potential exposure to air pollutants, which is a critical factor in assessing health risks associated with air pollution. The goal is to motivate users to make informed route selections. To achieve this objective, it is imperative to improve the underlying algorithm, going beyond mere measurements to encompass a more holistic representation of air pollution exposure. This enhancement does not only require a thorough analysis of the algorithm but also a strategic approach that factors in duration, intensity, and other variables contributing to cumulative exposure.

To further extend the efficiency of the path-finding algorithm, the implementation of optimizations is highly advisable. Strategies such as informed search techniques, for instance, the A* algorithm, present the potential to speed up route calculations by directing the exploration process towards the solution. Similarly, the integration of bi-directional searches can significantly reduce the search space, thereby enhancing efficiency. Partial graph loading presents another approach for optimization, beyond the confines of the immediate neighboring nodes. This approach can substantially diminish the number of database calls and, in turn, facilitate the expeditious retrieval of routing information.

Additionally, using time in the search method is something to consider. Adding time in the search would essentially mean adding another variable to filter data by. This would require further aggregations of pollution layers and more complex pre-processing in the service logic server. It would also mean more complex queries. Therefore due to the added complexity a temporal search would cause, such changes should only happen if the search is made more efficient so that it can handle more expensive database queries.

References

[1] F. Liang, Q. Xiao, K. Huang, X. Yang, F. Liu, J. Li, X. Lu, Y. Liu, and D. Gu, "The 17-y spatiotemporal trend of pm2.5 and its mortality burden in china," *Proceedings of the National Academy of Sciences*, vol. 117, no. 41, pp. 25 601–25 608, 9 2020.

[2] G. Guannan, Z. Yixuan, Q. Zhang, X. Tao, H. Zhao, D. Tong, . Davis, and S, "Drivers of pm2.5 air pollution deaths in china," *Nature Geoscience*, vol. 14, no. 9, pp. 645–650, 2021.

[3] "Air quality in wellington city." [Online]. Available: https://www.iqair.com/new-zealand/wellington/wellington-city

[4] "Health impacts of exposure to human-made air pollution." [Online]. Available: https://www.stats.govt.nz/news/health-impacts-of-exposure-to-human-made-air-pollution/

[5] "Daily co2 measurements from niwa's atmospheric monitoring station at baring head," Apr 2022. [Online]. Available: https://niwa.co.nz/climate/research-projects/carbonwatchnz/dailyco2measurements

[6] M. Daalder, "The invisible killer: New zealand's air pollution crisis," Oct 2023. [Online]. Available: https://www.newsroom.co.nz/the-invisible-killer-new-zealands-air-pollution-crisis

[7] M. Hatzopoulou, S. Weichenthal, G. Barreau, M. Goldberg, W. Farrell, D. Crouse, and N. Ross, "A web-based route planning tool to reduce cyclists' exposures to traffic pollution: A case study in montreal, canada," *Environmental Research*, vol. 123, pp. 58–61, 5 2013.

[8] J. Helle, A. Poom, E. Willberg, and T. Toivonen, "The green paths route planning software for exposure-optimised travel."

[9] L. A. Hawinkels, "Mounting air quality sensors to a uav to create a crowd-sourced network," 2022.

[10] J. Asfari, "Pollution-aware route planning using drone-based sensors," 2022.

[11] "Ensure healthy lives and promote well-being for all at all ages." [Online]. Available: https://sdgs.un.org/goals/goal3

[12] "Make cities and human settlements inclusive, safe, resilient and sustainable." [Online]. Available: https://sdgs.un.org/goals/goal11

[13] [Online]. Available: https://www.esri.com/en-us/what-is-gis/overview

[14] Jun 2020. [Online]. Available: https://docs.qgis.org/3.0/en/docs/gentle_gis_introduction/raster_data.html

[15] A. Grieve, "Find a clean air route," Jun. 2017. [Online]. Available: https://cleanairroutes.london/#plan-a-route

[16] G. P. M and J. S, "Evaluation of interpolation techniques for air quality monitoring using statistical error metrics a review," Apr 2018. [Online]. Available: https://www.ijert.org/evaluation-of-interpolation-techniques-for-air-quality-monitoring-using-statistical-error-metrics-a-review

[17] P. Ramsey and M. Leslie, "2. introduction - introduction to postgis." [Online]. Available: https://postgis.net/workshops/postgis-intro/introduction.html

[18] A. S. Arora, *Editor(s): Jasbir Singh Arora, Introduction to Optimum Design*. Academic Press, 2017, pp. 771–794.

[19] Y. Disser, M. Muller–Hannemann, , and M. Schnee, "Multi-criteria shortest paths in time-dependenttrain networks," *McGeoch, C.C. (eds) Experimental Algorithms*, vol. 5038, 2008.

[20] S. Majumder and S. Kar, "Multi-criteria shortest path for rough graph," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 6, pp. 1835–1859, 2018.

[21] "Resource management (national environmental standards for air quality) regulations 2004." [Online]. Available: https://www.legislation.govt.nz/regulation/public/2004/0309/latest/DLM287036.html

[22] [Online]. Available: https://www.health.act.gov.au/about-our-health-system/population-health/environmental-monitoring/air-quality/measuring-air

[23] [Online]. Available: http://postgis.net/workshops/postgis-intro/indexing.html

[24] V. Agafonkin, "An open-source javascript library for interactive maps." [Online]. Available: https://leafletjs.com/

[25] J. Brooke, "Sus: A quick and dirty usability scale," *Usability Eval. Ind.*, vol. 189, 11 1995.

[26] J. Sauro, "Measuring usability with the system usability scale (sus)," Feb 2011. [Online]. Available: https://measuringu.com/sus/

[27] [Online]. Available: https://www.copernicus.eu/en/accessing-data-where-and-how/copernicus-services-catalogue?combine=&amp;cc_source_service_target_id%5B2770%5D=2770

[28] N. Earth Science Data Systems, "Air quality data pathfinder - find data," Jul 2023. [Online]. Available: https://www.earthdata.nasa.gov/learn/pathfinders/air-quality-data-pathfinder/find-data#trace-gas-data

VII. Acknowledgements

Thanks to my supervisors Professor Winston Seah, Professor Colin Simpson, and Dr Jyoti Sahni who made this project possible and provided continuous support. Also thanks to the Victoria University of Wellington Wireless Networks research group whose members provided invaluable insights.