ZEN ROUTING

An Undergraduate Research Scholars Thesis

by

JUSTIN LEWIS and JOSE PABLO DOMINGUEZ

Submitted to the Undergraduate Research Scholars program at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by Research Advisor: Dr. Jean-Francois Chamberland

May 2017

Major: Electrical Engineering

TABLE OF CONTENTS

ABSTRACT

Zen Routing

Justin Lewis and Jose Pablo Dominguez
Department of Electrical Engineering
Texas A&M University

Research Advisor: Dr. Jean-Francois Chamberland
Department of Electrical Engineering
Texas A&M University

Driving induced stress is a problem inherent to contemporary living in urban areas. Traffic congestion, route unpredictability, and other factors cause undue stress to commuters daily. This project's purpose is to alleviate driving related stress by offering alternative "Zen" routes. This service will be provided in the form of a web application. Currently, navigational apps provide options based on shortest estimated time of arrival or shortest distance. The planned application will analyze a number of other factors to suggest routes that are comparable in time to the fastest route, but are less stressful. The ideal "Zen" route will be determined by applying Dijkstra's shortest path algorithm to a directed roadway graph. Stress related characteristics will be incorporated into this process by defining network edge weights as a scaled sum of roadway factors. The way in which each roadway factor contributes to the final route decision will be learned through user feedback. In the end, this method of routing will incentivize people to take alternative routes based on the benefits of stress reduction. The overall benefit to the user will hopefully take form in increased driving safety and overall well-being.

CHAPTER I

INTRODUCTION

A.) Motivations

Commuting and driving in general are a major cause of stress to many people. The extent to which this stress affects our health goes relatively unnoticed. New reports from several sources expound on the effects of driving related stress. One report from the U.K.'s Office of National Statistics shows that individuals who have long commutes report higher levels of anxiety and lower levels of life satisfaction as compared to short distance commuters [1]. The point being that the longer you spend driving under stressful conditions, the higher the cost. The long distance nature of an individual's commute cannot be alleviated by a routing application; however, the driving environment and path taken can be improved. The choice of a route can become a deciding factor in the driver's overall commute experience.

As pointed out in past studies, specific route factors affect drivers' stress. These factors are generalized into the concepts of personal control and commute predictability [2]. Situations such as traffic jams and congestion due to construction are examples which fall partially into both categories. Because these occurrences are somewhat unpredictable, drivers become frustrated when their commutes are lengthened or made more stressful due to an unforeseen change. Additionally, stress can become further elevated when drivers have limited options in these situations. They may become stuck on the highway in the middle of high congestion with no way to exit or change their current situation.

The proposed “Zen” approach to routing takes advantage of the stress-related factors described above. Routes can be analyzed for specific, measurable characteristics. Such characteristics include: high vehicle congestion, presence of construction, and stop-go traffic behavior. Routes with these characteristics will be avoided. Instead, the route chosen will exhibit inverse characteristics that should reduce driving stress. These characteristics may include: low traffic volume and route time predictability. This approach to traffic routing is currently unavailable via existing services, and it is progressive in its perspective on driving. The hope is that an application which focuses on driving stress reduction will lead to better driving experiences and will change individuals’ viewpoint on navigation.

B.) Design Framework

B.1) Primary Goals

In order to approach the project goal of stress-sensitive traffic routing, three primary components were identified: I) A framework for describing roadways and intersections. II) A mathematical model for evaluating the stress-related characteristics of a route. III) A method for selecting the optimal route. With these objectives in mind, simple yet powerful mathematical descriptions were chosen by the research team. For the first objective, a directed network graph structure was used in order to describe a city's traffic network. Within this scheme, the network graph *edges* represent the roadways while the graph *nodes* represent the intersections.

Additionally, each *edge* has a set of numerical *weights* which describes a road segment's features (distance is one example). For the evaluation of stress-related features, a mathematical metric was developed and termed as a route's Zen score. The form of this metric translates a road segment's stress-related features into a quantified number. For the final objective, a modified version of Dijkstra's shortest path algorithm was utilized. The modification made allows for multiple factors (e.g. "Zenness" and driving time) to be accounted for in finding the optimal route.

C.) Technical Background

Before diving into the project methods, there are several terms and concepts used in this thesis which may seem foreign to readers of various technical backgrounds. In order to mitigate this issue, they are introduced here for accessibility:

C.1) Network Graph

Mathematical tool used to model pairwise relations between objects. A network graph $\boldsymbol{G(\vec{N}, \vec{E})}$ consists of nodes $\boldsymbol{\vec{N}}$ and edges $\boldsymbol{\vec{E}}$. An edge $\boldsymbol{E(N_A, N_B)}$ can be seen as the connection between two unique nodes $\boldsymbol{N_A}$ and $\boldsymbol{N_B}$. Each edge is assigned a set of values or weights which model the system in question. In addition to edges and nodes, a network graph has a number of important characteristics including: connectedness and directivity. Connectedness describes the degree to which either node pairs or the network graph as a whole are connected. Directivity describes whether or not graph edges have an associated direction. In the context of the project presented, a city map is modeled as a *directed* network graph with streets modeled as edges and intersections modeled as nodes.

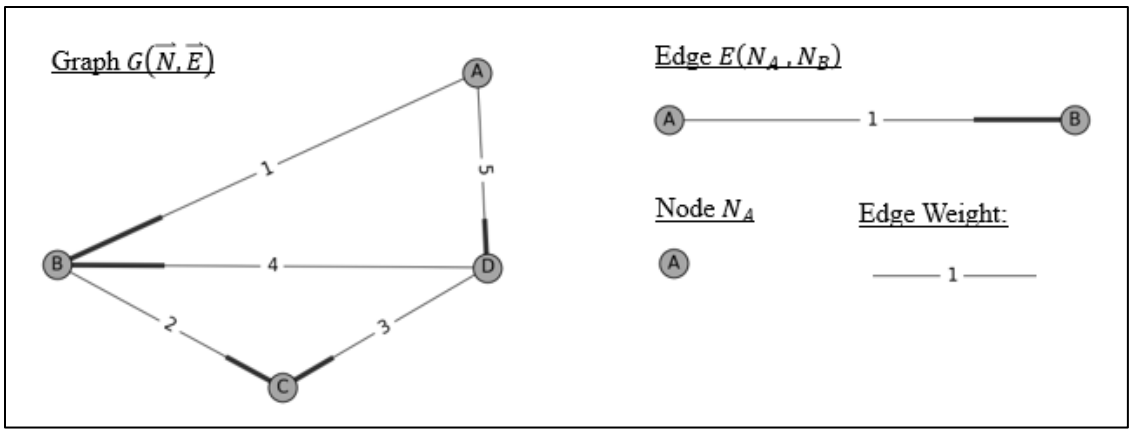

Figure 1) Network Graph Diagram

C.2) Network Path

Subnetwork component which is useful in the context of traffic routing. A network path between nodes N_A and N_B can be defined as a list of nodes such that each subsequent pair of nodes in the list is a well-defined edge of the given network graph.

Definition 1) Network Graph Path

Given $G(\vec{N}, \vec{E})$ and node choices $[c_1, c_2, \dots, c_n]$ $s.t.$ $c_1 = A$ and $c_n = B$

Path $P \triangleq [N_{c_1}, N_{c_2}, \dots, N_{c_n}]$ $s.t.$ $E(N_{c_i}, N_{c_{i+1}}) \in \vec{E}$ $\forall$ $i = 1,2, \dots, n-1$

C.3) Shortest Path Algorithm

Given a network graph $\boldsymbol{G(\vec{N}, \vec{E})}$ and two nodes of interest $\boldsymbol{N_O}$ and $\boldsymbol{N_D}$ a shortest path algorithm quickly determines the shortest path between origin and destination. The shortest path in this context is defined as the path that connects the two desired nodes and minimizes the sum of the edge values or weights along the path. In the context of the project presented, the shortest path algorithm is used to find the optimal Zen route.

C.4) GoogleMaps API Set

An Application Program Interface or API, is a set of routines which enables access to a built code library or data set. In the context of the project presented, the API sets are provided by Google. Furthermore, these API sets are utilized to access real-time traffic data such as road congestion and accident presence.

C.5) Zen-ness

This term was coined by the project researchers in order to describe the overall stress-related character of a route or roadway. A path which possesses a good Zen score is considered pleasant to the driver; in contrast, a path which possesses a bad Zen score is considered stressful. The way in which this abstract idea of Zen-ness was defined mathematically is detailed in the coming sections.

CHAPTER II

METHODS

A.) Shortest-Time Traffic Routing

A.1) Mathematical Background

Before describing how stress-factors or Zen-ness can be incorporated into a routing decision, it is instructive to explain how single-objective routing is accomplished. This problem is framed in the context of constrained optimization. Within this framework, an objective function $F(\vec{x})$ is minimized subject to a set of constraints. This is stated mathematically:

Problem 1) Single-Objective Optimization

Minimize	$F(\vec{x})$,	$\vec{x} = [x_1, x_2, \dots, x_n]$
Subject to	$g_i(\vec{x}) \leq 0$,	$i = 1,2, \dots, m$
	$h_j(\vec{x}) = 0$,	$j = 1,2, \dots, k$

The objective function $F(\vec{x})$ is a function of n independent decision variables (denoted by $\vec{x}$). The values which these variables can take is limited by two sets of constraints: inequalities $g_i(\vec{x})$ and equalities $h_j(\vec{x})$. Simply stated, the goal of the constrained optimization problem is to find the optimal values of $\vec{x}$ while satisfying the defined constraints.

A.2) Shortest-time Traffic Routing

Within this optimization framework, the end goal of shortest-time traffic routing is well articulated. Given a directed network graph $G(\vec{N}, \vec{E})$ defined by nodes $\vec{N}$ and directed edges $\vec{E}$, the optimal route is one which minimizes the time spent driving between nodes N_O and N_D. The objective function in this case is the time spent driving from origin to destination. This objective function is defined by only one decision variable: path choice. The path decision is constrained by the stipulation that it must connect origin and destination; otherwise, it cannot be accepted as a valid solution. Assuming there is one or more paths which connect nodes N_O and N_D, there is at least one path which is optimal (i.e. a path which minimizes the objective function). This discussion is summarized mathematically:

Problem 2) Shortest-Time Routing Problem

Given all paths $\vec{P}$, nodes N_O and N_D, and $t(P_i) =$ time along P_i		
Minimize	$F(P_i) = t(P_i)$,	$s.t.\ \ P_i \in \vec{P}$
Subject to	P_i connects N_O and N_D	

So how can the optimal path $P_{optimal}$ be obtained? The path space $\vec{P}$, even if it was limited to only non-looping routes, is very large. For this reason, it is impractical to search through every valid path and find the optimal route. There are a number of shortest path algorithms which solve the problem posed above in a more efficient manner. Of these options, Dijkstra's algorithm was chosen, because it is one of the most accessible methods. The Dijkstra is relatively fast and runs in time $O(|N|^2)$ where $|N|$ is the number of graph nodes. In other

words, the approximate computational time of the algorithm increases quadratically as $|N|$ increases. More specifically, this big O notation describes an upper bound on computation time as $|N|$ approaches infinity. The actual computational time required is highly dependent on the graph's connectivity. In order to be thorough, the algorithm is briefly presented here:

A.3) Dijkstra's Shortest Path Algorithm

As previously mentioned, the main objective of the Dijkstra's algorithm is to find the shortest path between nodes on a graph. The algorithm initially sets the distances from the starting node to every other node as infinity in order to indicate that the nodes have not been analyzed yet. The first node is always the starting point which by default has a set distance of zero. Then, the algorithm selects the "current node" for each of the following iterations with the "current node" described as the closest node to the starting point that has not been analyzed.

For each iteration, the tentative distance between the "current node" and an analyzed node around it is calculated. After, the tentative distance and the label of the adjacent node are added. This sum is then assigned as a label to the "current node". The process is repeated with every analyzed node around the "current node". If a sum that is smaller than the current labeled distance is found, the label is updated with the smaller value. Otherwise, the same value is kept.

Once every adjacent node to the "current node" has been considered, the "current node" is marked as analyzed. A new "current node" is then selected and the process is repeated. It is important to note that once a node has been analyzed, it will never be checked again and therefore cannot be the "current node" anymore. The algorithm continues the calculation and only stops if it marks the destination node as analyzed or if the smallest tentative distance between nodes is infinity. In the latter case, there are no paths between the initial and final nodes.

Figure 2 is a visual representation of the algorithm. The bolded lines represent the edges that have been analyzed, while the black dots represent all of the nodes in the graph. The arrow points to the starting node.

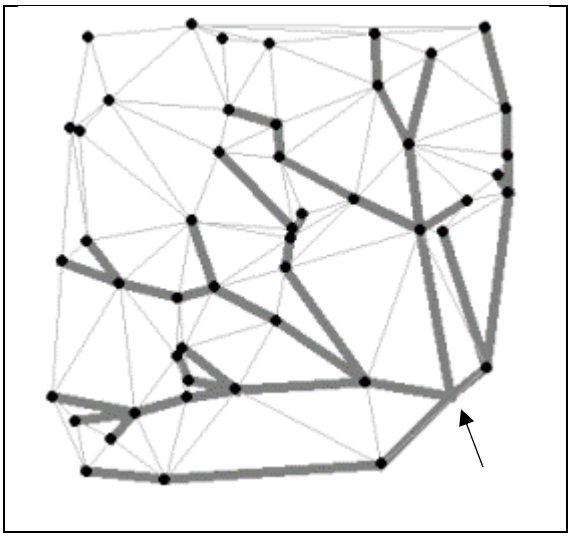

Figure 2) Example of Dijkstra's Algorithm calculating path

The algorithm can be followed in the following pseudocode. In this version, a graph, a starting node and a destination node are provided to the algorithm, which calculates the shortest path as previously described.

Code Snippet) Dijkstra's Algorithm

```
function Dijkstra(Graph, starting, destination):

    create node set N

    for each node n in Graph:                                   // Initialize
        distance[n] ← INFINITY                                  // Indicate nodes have not been analyzed
        previous[n] ← UNDEFINED                                 // There is no previous node at the start
        add n to N                                              // Add every node n to N (These nodes will be analyzed)
    distance[starting] ← 0                                      // Starting node

    while N is not empty:                                       // Once empty all the nodes will have been analyzed
        currentN ← node in N with min distance[currentN]        // Node with the least distance
        remove currentN from N                                  // Node has been analyzed

        for each neighbor n of currentN:                        // n still in N
            tent ← distance[currentN] + length(currentN, n)     // sum of distance from starting point and tentative distance
            if tent < distance[n]:                              // A shorter path to n has been found
                distance[n] ← tent
                previous[n] ← currentN
            if currentN == destination                          // Check to see if destination is reached
                return distance[], previous[]                   // If reached, return the distance and path
                break


    return "no paths available"                                 // If all adjacent nodes have been checked (N is empty)
```

B.) "Zen" Traffic Routing

B.1) Mathematical Background

So, given the basic structure for solving shortest path problems, how can stress-related features be incorporated into the method? As opposed to the single-objective optimization problem demonstrated in the previous section, "Zen" routing incorporates multiple, unique objectives. This new problem of optimizing more than one objective is described mathematically as follows:

Problem 3) Multiple-Objective Optimization

Minimize	$[F_1(\vec{x}), F_2(\vec{x}), \dots, F_N(\vec{x})]$	$\vec{x} = [x_1, x_2, \dots, x_n]$
Subject to	$g_i(\vec{x}) \leq 0,$	$i = 1,2, \dots, m$
	$h_j(\vec{x}) = 0,$	$j = 1,2, \dots, k$

This modified problem statement is termed within the literature as a multi-objective optimization problem. Similar to single-objective optimization, there are given sets of inequality and equality constraints. The modification is that now there are several, unique objectives to be optimized simultaneously. Solutions to multi-objective optimization problems are much more challenging and are generally classified as NP-hard or NP-complete. The multi-objective optimization problem is interesting in that there is no global solution which is considered "best" (withholding trivial cases). Instead, there is a set of "best" solutions which are termed in the literature as Pareto optimal. A solution is deemed Pareto optimal when there is no other possible solution which will provide a more beneficial value for one objective without detracting from any other objectives. In the language of the literature, no other solution dominates a solution which is Pareto optimal. A diagram is useful for explaining this concept:

Note: optimal solutions in this case minimize the objectives f_1 and f_2.

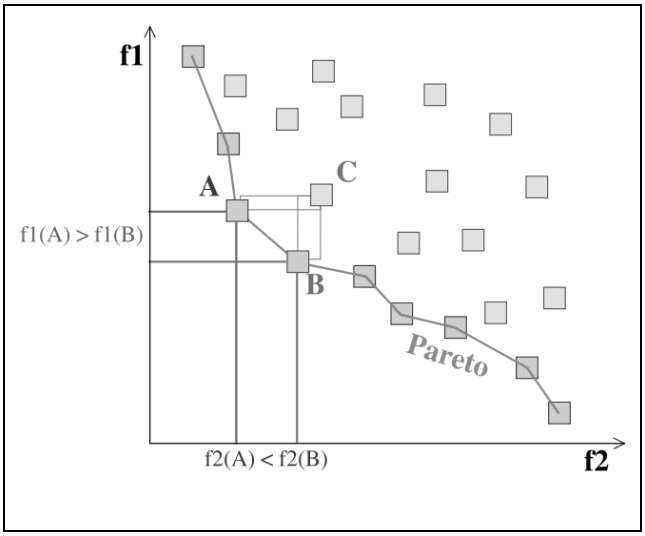

Figure 3) Example of Pareto Optimality

The line connecting the points at the border of the solution space is termed the Pareto frontier and is the set of Pareto optimal points. Assuming all possible solutions are present in the figure (each represented by a square box), it is easy to see how the connected points are Pareto optimal. If any new solution was found to exist on the left-hand side of the Pareto frontier, then it would take the place of one of the Pareto optimal solutions.

This concept of Pareto optimality is fundamental to the method of "Zen" routing. Within this routing scheme, more than one objective is considered in optimality. In addition to minimizing path distance, the scheme hopes to also minimize stress-related factors. For this reason, there is a set of Pareto optimal points rather than one best solution. The question which naturally arises is: which Pareto optimal solution is best? There are several methods to tackle this issue and the one which was chosen for the "Zen" routing scheme is described in the subsequent section.

B.2) Zen Routing Scheme

In order to achieve one unique solution among the set of Pareto optimal driving paths, the multi-objective problem is reduced to a single-objective problem. This reduction allows for the use of the shortest path algorithms developed for single-objective optimization such as Dijkstra's algorithm. So, how is this problem reduction conducted? The normalization technique utilized is straightforward. This normalization method is actualized through a modification of the single-objective function. In the shortest-time routing problem, the objective function accounts only for path time. In contrast, the "Zen" approach incorporates several path factors (i.e. traffic congestion, path distance, route predictability, and path time). In order to account for more than one factor, part of the objective function's magnitude must come from each factor. This idea is described

mathematically as a scaled sum of factors. The overall value of the objective function is now a sum of factors where each factor is scaled by its corresponding factor weight. This is described mathematically as follows:

Problem 4) Zen Routing Problem

Given paths $\vec{P}$, nodes N_O and N_D, path factors $\vec{f}(P_i)$, and factor weights $\vec{w}$

Minimize $F(P_i) = \sum_{all\ k} f_k(P_i) \cdot w_k,$ $s.t. \quad P_i \in \vec{P}, \quad f_k(P_i) \in \vec{f}(P_i)$, and $w_k \in \vec{w}$

Subject to $\sum_{all\ k} w_k = 1$

P_i connects N_O and N_D

Within this framework, there is a natural question as to how the factor weights $\vec{w}$ should be chosen. Ultimately, this choice is dependent on the individual driver. Each person has their own unique perspective on the tradeoffs between various path factors. By giving more or less weight to a specific factor, the more or less that factor dominates the resulting solution. Regardless of how the factor weights are chosen, the resulting path solution from Dijkstra's algorithm is Pareto optimal; therefore, the choice of which Pareto optimal path to take is individually determined. In later sections, methods for inferring a user's factor weights $\vec{w}$ are described. In order to gain a better understanding of how the factor weights affect the resulting "Zen" route, a simple example is presented:

B.3) Zen Routing Example

This simple example shows a network with only four nodes and five edges. The circles labeled with different letters represent the nodes. The bold segments next to the nodes in Figure 4 represent destination of the edge.

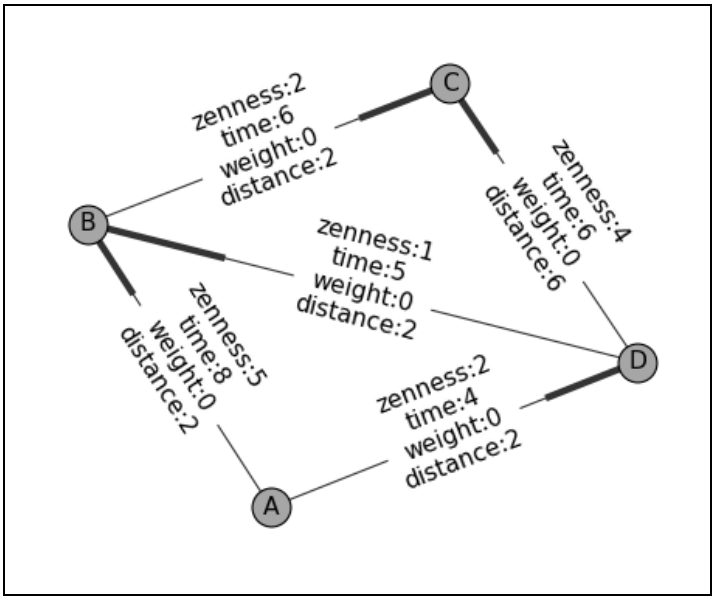

Figure 4) Example network

Each edge has been given specific values for Zenness, time, and distance to be able to test different weights which would normally be given by the user. These values can be observed in Table 1. No units were used for this example since the normalization of values is outside the scope of this simple example.

Table 1) Edge Values for Simple Example

Edge	Zenness	Time	Distance
A → B	5	8	2
A → D	2	4	2
D → B	1	5	2
B → C	2	6	2
D → C	4	6	6

After setting the values, different weights were tested to see if the algorithm would in fact choose the best route given certain parameters. The origin node was set as node A and the destination node was set to be node C. First, equal weights were passed to have a control route for comparison purposes. This route was found to be A→D→C. Figure 5 highlights the suggested route based on equal weights. Manual calculations confirm the selected route as the best option based on the weights.

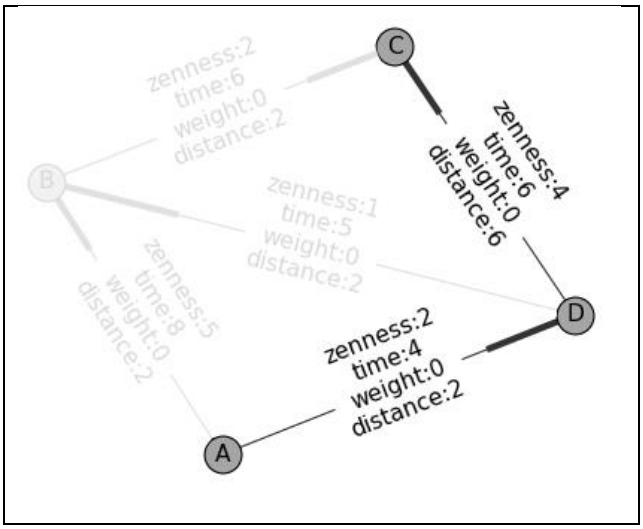

Figure 5) Route with equal weights

Next, the route for Zenness was calculated, giving it the highest weight. As a result, the algorithm found route A→D→B→C to be the best route for this mode. The route is highlighted in Figure 6.

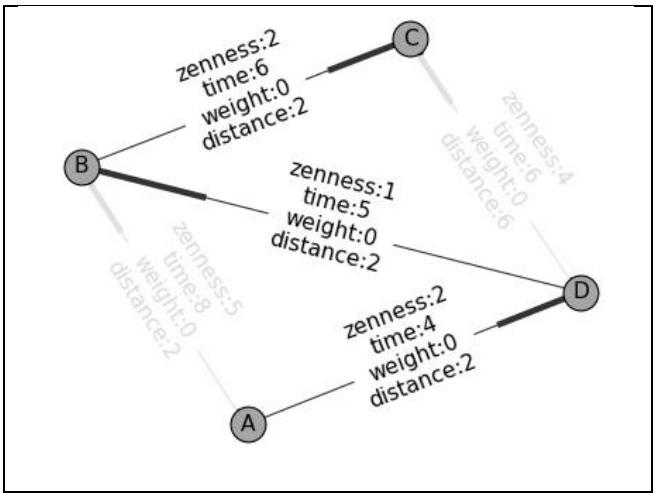

Figure 6) Zenness route

When weighting time the highest, the route to take was A→D→C which was the same as the control route. Lastly, the best route to take if distance traveled is weighted the highest is A→B→C. All the routes were manually calculated as well to prove the algorithm delivered the correct results. This simple example demonstrates the algorithm's capacity to take different inputs and correctly deliver the best path to the destination.

C.) Zen Scoring

Another major portion of the Zen routing scheme that has yet to be discussed is the measure of stress-related factors. To start off with, only traffic congestion is considered for simplicity. In order to measure the congestion of a roadway segment, a composite metric was devised and termed as the road's Zen score. The metric takes several values as input to generate a single calculated number. The metric is defined for each road segment as follows:

Equation 1):

$$Zen\ Score = \ln\left(1 + \frac{t_{current} - t_{base}}{t_{base}}\right) \cdot d_{segment} \qquad \forall \ t_{current} \geq t_{base} \geq 0$$

Within this formula, there are several quantities of interest: the *current-time*, *base-time*, and *segment distance*. The *current-time* is the expected time to drive along a segment *with* current roadway traffic levels accounted for. The *base-time* is the expected time to drive along a segment *without* traffic. Thus, the difference between these two values describes the added time due to traffic. This difference is divided by the *base-time* because the ratio of traffic time to *base-time* is more indicative than the traffic time alone. The logarithmic form was chosen due to the metric's tie to human nature. Internal human perception of various external stimuli has been found to

loosely follow a logarithmic form [5]. By mapping Zen scores in this way, the metric better fits human perception of traffic. The last input to the metric, *segment distance*, is multiplied to account for the distance over which the traffic congestion occurs.

D.) Data Extraction

A significant portion of the project methods were guided by the limitations of data availability. In order to extract real-time traffic information, standard Google Maps API services were utilized. Understandably, the amount of information provided by Google is limited in nature. The Directions API service allows a client to acquire the expected travel time between two nodes based on current or future traffic conditions. The number of these API requests is limited to a daily quota of 2500 free queries. This limitation led to several key project decisions. First, a reduction in the overall network graph structure was conducted (the details of this reduction are outlined in the next section). Second, scripts were written in order to utilize multiple API keys for testing. Access to the API services is tracked by Google through the use of an identifying developer key. Thus, by using multiple registered keys, access to API server queries was increased.

E.) Network Reduction

To accommodate the limited access to real-time traffic data, the analyzed network graph was reduced to its most essential edges. Two different methods were utilized to achieve this reduction. The first method used the pre-defined structure of the roadway data provided by OpenStreetMap (OSM). This open license map service provides geodata such as street coordinates which were used to generate the original network graph. Additionally, OSM provides identifiers for each road segment. As an example, highways are given a unique tag which is distinct from the tag assigned to residential sections. By removing road segments based on tag information, the network was reduced from approx. 5000 nodes to only 500, as demonstrated in the figure below.

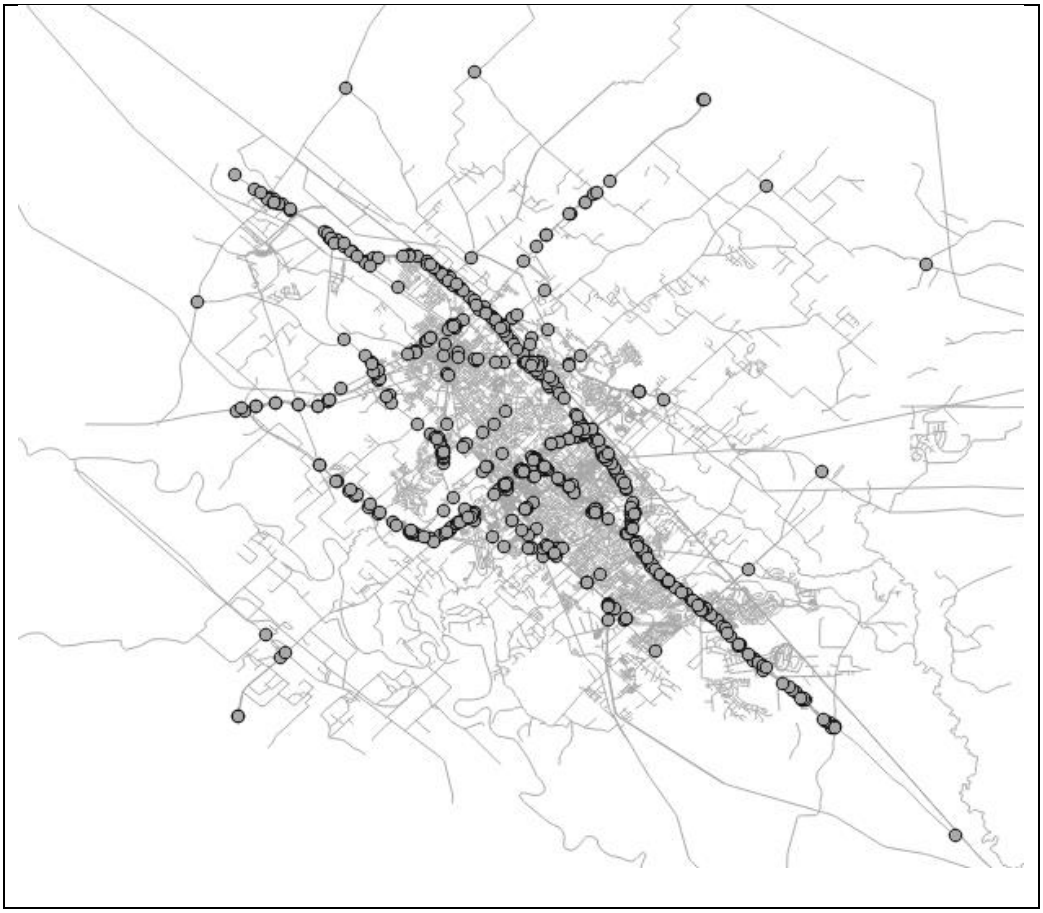

Figure 7) Reduced Network Nodes

CHAPTER III

RESULTS

A.) Zen Score Validation

To confirm the validity of the Zen Score, a visual representation of the data was generated for interpretation. For each edge within network G, its corresponding Zen Score was mapped to a point within a color gradient. Figure eight represents a subregion of the College Station/Bryan area with an overlay of the reduced network graph from section II.E).

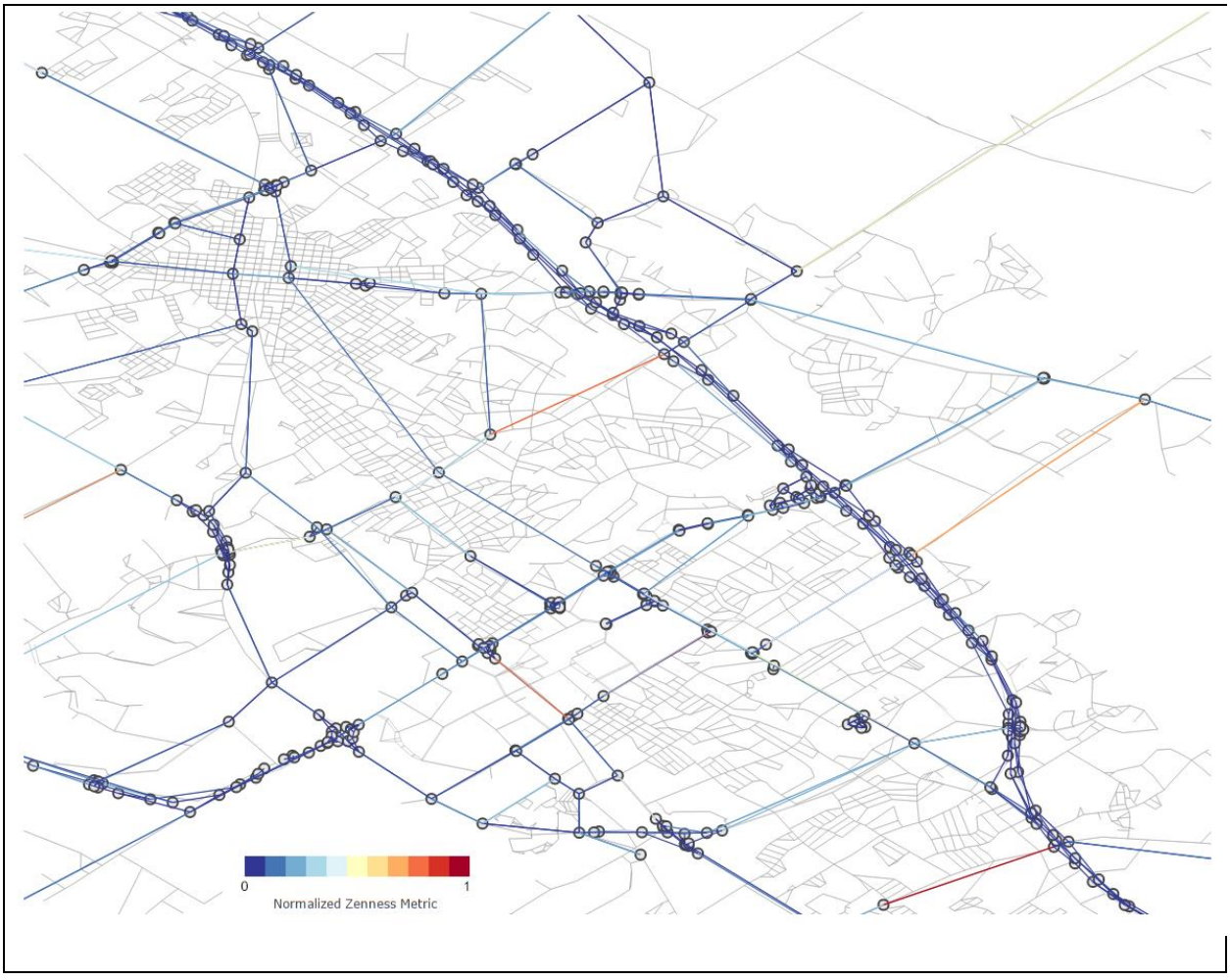

Figure 8) Network with Zen Scores:

Viewing figure eight, the dark red edges have a higher Zen Score value which means they possess elevated congestion levels. Such congestion leads to added stress to the driver's route experience. On the other hand, the blue edges of the network are streets with a low Zen Score, meaning they are less stressful. In order to provide some verification of the Zen Score's ability to approximate street congestion, a histogram of the network's Zen scores was generated. The histogram plot found in figure X2 demonstrates the naturally expected variation of Zen scores within the network. The plot has two primary components: one group of bars at the origin and another distinct group at higher Zen score levels. Both components are distributed in a way which is relatively smooth which is to be expected. In short, most edges are uncongested; however, there is a less dominate mode which bears elevated congestion levels.

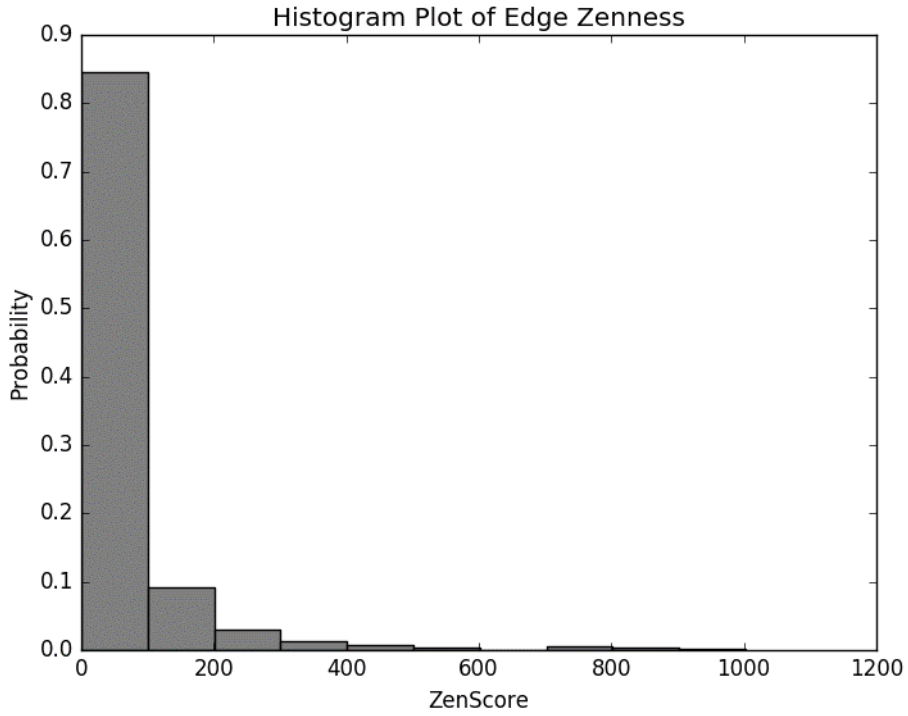

Figure 9) Zen Scores Histogram:

B.) Zen Route Example

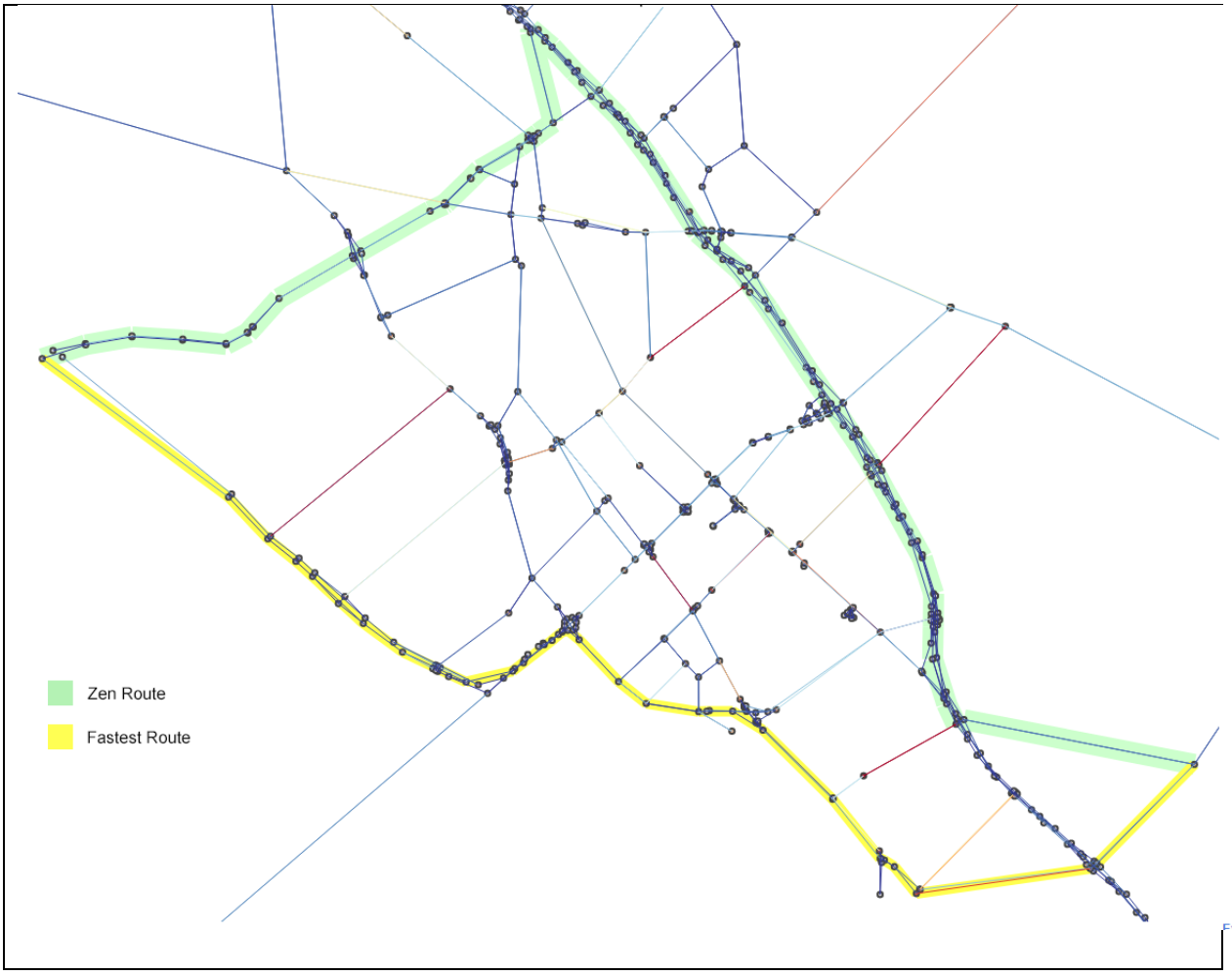

Figure 10) Difference in routes:

The yellow route displays the fastest route possible and the green route shows the Zen Route. The Zen Route is calculated using the shortest path algorithm and the Zen Scores attached to each edge. As previously described, the algorithm goes through every possible option to find the Zen Route which is the route with the lowest Zen Score.

Table 2) Edge Values for Simple Example

Route	Zenness	Time (min)
Zen Route	964.374	27.2000
Fastest Route	1499.053	27.1833
Difference	534.679	0.0167

As it can be observed in Table 2, the algorithm is able to find a route that differs in time by less than a minute when compared to the fastest route, but offers a significantly lower value of Zenness.

C.) Pareto Optimality

As stated previously, the Zen routing scheme utilizes the Dijkstra algorithm to generate routes which optimize a defined objective function. For a given realization of factor weights $\overrightarrow{\boldsymbol{w}}$, it can be further qualified that such routes are Pareto optimal solutions. For the simplified two factor scheme adopted (i.e path time and Zenness), a Pareto frontier can be generated by sweeping the values of the two corresponding factor weights $\boldsymbol{w}_{zen}$ and $\boldsymbol{w}_{time}$. The number of points along this frontier is dependent on the specific origin and destination chosen. An example from the traffic network was selected in order to illustrate the relevance of Pareto optimality:

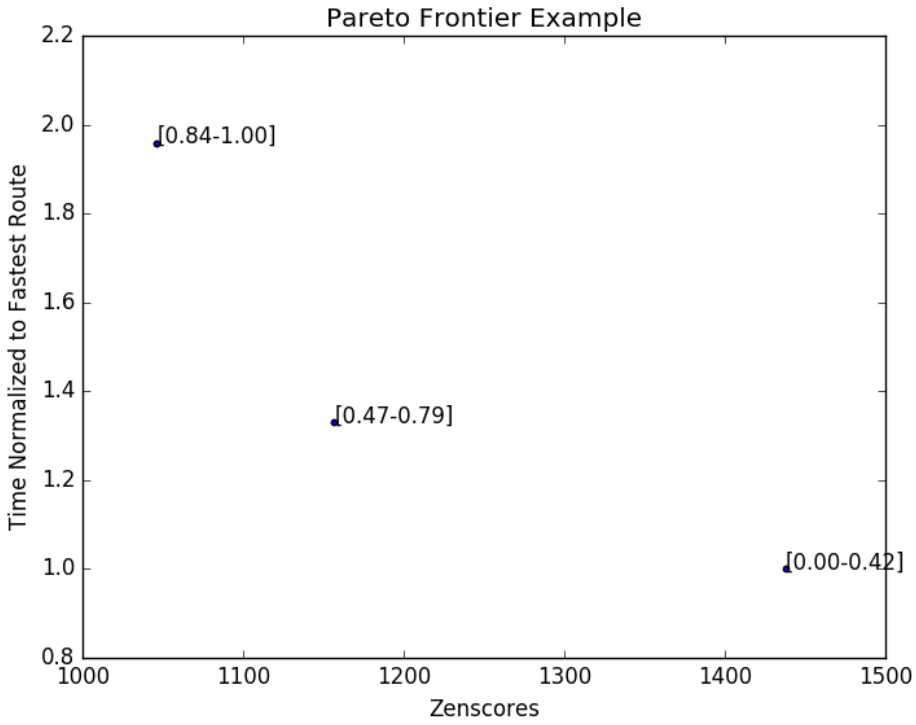

Figure 11) Pareto Frontier Example:

For the origin and destination chosen, three Pareto optimal routes were generated by the Dijkstra algorithm. With two factors considered, the Zenness factor weight was swept from zero to one while the path time factor weight was chosen to satisfy: $w_{zen} + w_{time} = 1$. For each route, a range of values for w_{zen} mapped to it. These ranges are annotated above their corresponding data points within Figure eleven. The plot shows that for a Zenness factor weight of zero, the fastest route is found. In contrast, for a weight value of one, the generated path possesses the lowest Zen score. This route may be optimal in terms of path congestion quality; however, the path time is twice that of the fastest route. For this reason, most individuals would likely be uninclined to take such a route. The third point finds a middle ground between congestion quality and time. Some time is added to the best possible path time, but a reduction in path congestion is gained as a result. These three different routes are presented below in Figure twelve.

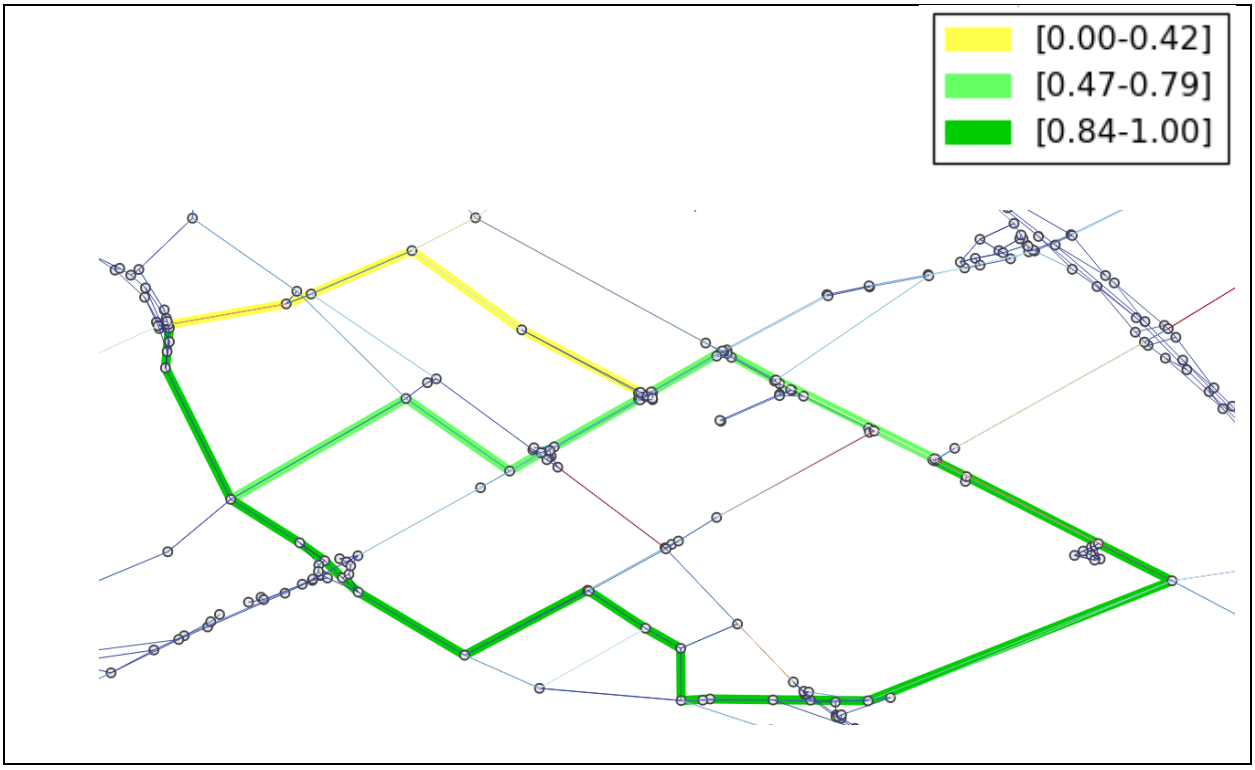

Figure 12) Pareto Optimal Routes:

C.) Estimating User Weights

D.) Zen Relevance

A specific question of interest to the research team was the following: under what network conditions are the benefits of Zen routing most relevant? For users of this routing scheme, this insight could be very useful. To answer the question, the Zen route is compared to the fastest route. By comparing the Zen scores of these two routes, trends can be found in the variation of network conditions. One intuitive network feature to analyze for Zen benefits is average congestion. The thought is that as the network becomes more congested on average, there is an increasing potential for reduction in stress. The following plot shows the experimental trend found between the Zen score reduction (between the fastest route and the Zen route) and average network congestion.

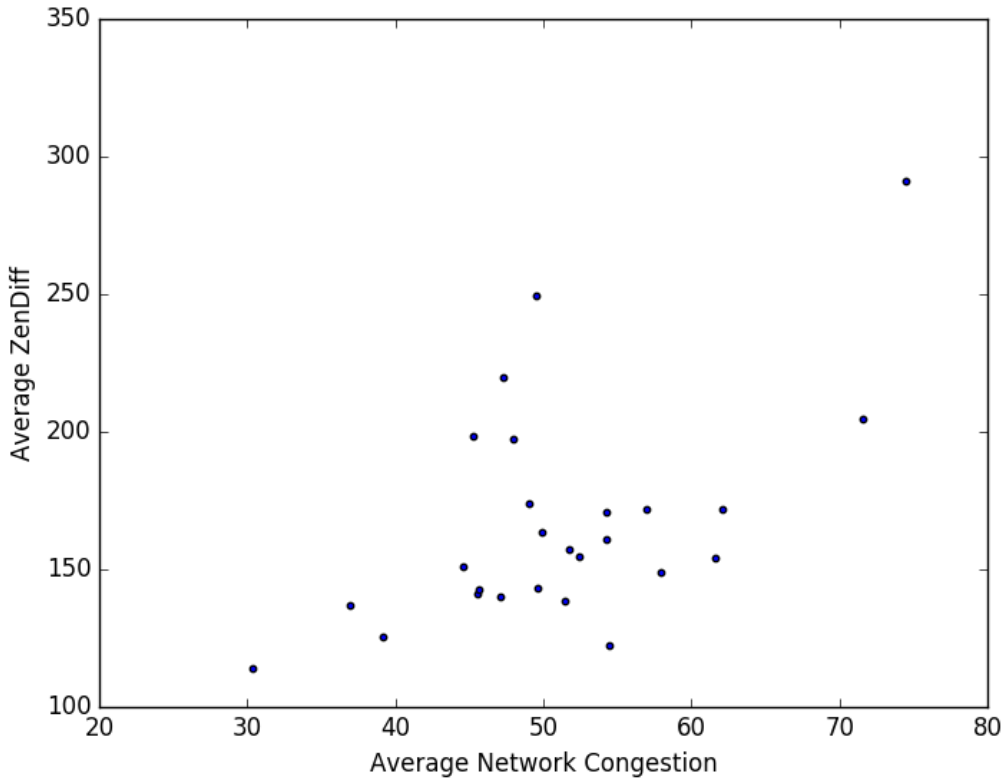

Figure X) Zen Difference vs. Network Congestion:

Figure X seems to validate the intuition behind stress reduction benefiting with increased congestion. For low values of average congestion, it is clear why the average Zen score difference is small. This is because with a lack of added time due to congestion, the Zen route converges to the fastest route. As average congestion increases the plot shows a slight trend in increasing Zen reduction by taking the alternative Zen route.

Although Zen score reduction is a useful measure of Zen relevance, it does not account for the resulting addition of path time. In order to achieve higher reductions in Zenness, time is naturally added. One metric that captures this inevitable tradeoff between Zenness and time is a ratio of Zen score reduction over time added. This metric is given by the following equation:

Equation 2):

$$tradeoff\ ratio = \frac{Zenness_{fastest} - Zenness_{Zen}}{time_{Zen} - time_{fastest}}$$

With another metric by which to view Zen relevance, further plots can be developed. Similar to the first metric of Zen score difference, it seems that the tradeoff ratio would be directly affected by average network congestion levels. The initial intuition developed was that the ratio might initially rise with average network congestion and then fall off. The reasoning behind this is that there is a fleeting set of network conditions by which Zen reduction can be traded off efficiently for time. As an example, as one road segment becomes increasingly congested, it is natural for other vehicles to take secondary or side routes. Whether or not the intuition provided is valid, when the average tradeoff ratio is found as a function of average congestion, figure Y is the result:

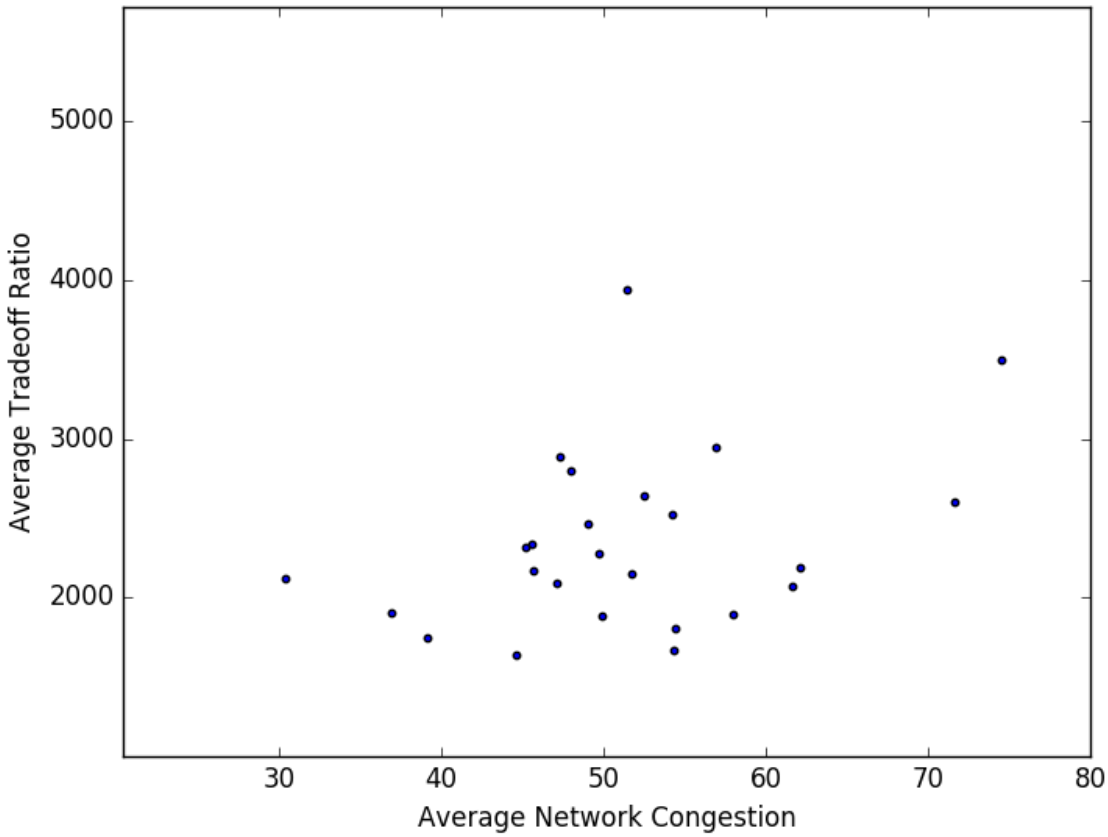

Figure Y) Tradeoff Ratio vs. Network Congestion:

This graph lacks the clearer trend found in the case of the Zen score difference metric; however, it does meet the intuition provided in some ways. There seems to be a concentration of high values of the tradeoff ratio at moderate network congestion. This may have some merit, but the portion of the graph at the lowest values for network congestion seems suspicious. In this region, the ratio should be at its lowest yet it is not. All of these things considered, there does not seem to be enough data or clear pattern to merit a strong claim about a trend between the tradeoff ratio and average network congestion.

CHAPTER IV

CONCLUSION

A.) Further Improvements

A.1) ZenScoring

The way in which stress due to congestion was modeled within the Zen routing scheme could be improved with further data access. The model was influenced heavily by the source of real-time traffic data available. For this reason, the presented Zen score metric is a rough approximation. By using added time due to traffic, the Zen score oversimplifies the dynamic behavior of congestion. If access to crowdsourced GPS data was available, a more complete metric could be devised to characterize congestion.

A.2) Potential Roadway Factors

The presented method of characterizing roadway stress was very simplistic; thus, there is great room for improvement. In addition to roadway congestion, other factors such as route predictability and comfortability could be considered. Metrics for route predictability could be used to characterize route time variation. This would provide drivers with a way to choose a route which is highly predictable when timing is essential. Route comfortability might be useful in avoiding routes which cause stress due to unusual driving environments. An example could be narrow, two lane highways.

A.3) User Classification

Because the Zen routing scheme requires inference of user weights, there is some cost in estimating these weights. As path factors are added, complete user characterization could take numerous driving iterations. In order to speed up the process of weight inference, user classification could be used to acquire data which is revealing of an individual's affinity for driving stress. As an example, driving habits may be indicative of the sort of routes a person might find stressful. By first classifying a driver by their driving ability, an improved scheme can create a better initial guess for the users stress factor weights. The benefit here is due to the relative ease of gathering driving habit data over gathering data for stress factor preference.

B.) Potential Applications

Although Zenness and stress was the main focus of the scheme presented, the application's main framework could be used to implement other features as well. Some of these examples are shown in the following sections:

B.1) Safe Route

With a similar mechanism as the Zen Route, a safe route application can be implemented where the main goal is to give the users directions to a destination the safest way possible. Navigation applications often only focus on the time it takes to get to a destination but are not as responsive to accidents, natural events, and safety in general. A safe route application could focus on updating the routes as safety information is acquired. This could include but not be limited to: frozen roads, snow, roads under water, car accidents, debris on the road, wildlife, and even long road segments without a gas station or access to help.

B.2) Scenic Route

Especially useful when vacationing and going on road trips, a scenic route feature could show the users what route to take to see areas of natural or cultural beauty. With the help of image processing and user feedback, a scenic route feature can use the Zen Route main framework to rate different routes and choose the one that bests balances the parameters input by the user. In addition, machine learning could help with improving the type of scenic routes each user enjoys.

B.3) Car Insurance

Car insurance rates are often not specific to each car driver. For instance, males under 25 pay more for car insurance than men above that age. Although the age brackets are based on statistics from other drivers in that age bracket, not every driver is the same. Some insurance companies currently try to account for this generalization by analyzing individuals' driving habits. This concept of user-specific rates could be furthered by the route analysis tools inherent to the multi-objective routing schemes. Routes could be characterized in terms of safety and users that take these routes might be given insurance rate reductions. By assessing an individual's choice of route and driving environment, companies can gain more information for reducing risk.

C.) Final Remarks

REFERENCES

[1] Commuting and Personal Well-being. Publication. London: Office for National Statistics, 2014. The National Archives. Web. 4 Sept. 2016.

[2] Novaco, Raymond W., and Oscar Gonzalez I. Commuting and Well-being. Rep. Irvine: University of California, N/A. Print.

[3] Goldberg, Andrew V., and Chris Harrelson. Computing the Shortest Path: A Search Meets Graph Theory. Tech. no. MSR-TR-2004-24. Redmond, WA: Microsoft Corporation, 2003. Print.

[4] Nuha A. S. Alwan, "Performance Analysis of Dijkstra-Based Weighted Sum Minimization Routing Algorithm for Wireless Mesh Networks," Modelling and Simulation in Engineering, vol. 2014, Article ID 658408

[5] Why do we perceive logarithmically? Significance, Vol. 10, No. 1. (1 February 2013), pp. 28-31, Lav R. Varshney, John Z. Sun

APPENDIX

A. Code Platform: Python / Networkx / OSM