MYWAY

PERSONALIZED CAMPUS ROUTING

NATALIE CHYBA
PATRICK LERCHI
ISABEL VIEGAS DE LIMA
JOSH ZARRABI

CEE186: CYBER-PHYSICAL SYSTEMS

Moving around campus is always a challenge: which route will get you to Econ on time? Which route will you be least likely to run into your GSI? Which route will keep you most fresh for your interview? Navigating around campus is not always as simple as getting from point A to point B the fastest. As students, we spend a lot of time deciding which path to take and which to avoid based on our personal preferences. MyWay speaks to this and streamlines the decision making process, allowing users to customize their preferences and thus providing a unique and optimized route to meet their needs.

TABLE OF CONTENTS

INTRODUCTION

MOTIVATION AND BACKGROUND

Within the last decade, the surge in navigational apps that provide optimal routing services have redefined how we travel and revolutionized the transportation industry. While these services have become commonplace when navigating cities or countries around the world, there has not been a successful implementation of this technology for areas of a small scale. Most services can route you through a college campus or national park; however, they often fail to recognize that users' priorities in these spaces are not always centered upon minimizing travel time and instead often depend on the experience along the way. When traversing an airport, getting from gate to gate the quickest may be less imperative than guaranteeing you'll pass a Starbucks on your way. When navigating Disneyland, your path from ride to ride may be defined by the likelihood of a route providing your five-year-old with a Buzz Lightyear photo-op rather than being the first in line. Likewise, there are a number of factors that students, faculty, and visitors might take into consideration while navigating a campus. We hope to fill the void in this niche market and provide a standard for navigating areas with a unique set of characteristics and user preferences. To prototype this concept, we have created MyWay; a personalized routing service for UC Berkeley designed to meet students' needs.

RELEVANT LITERATURE

"We're about to undergo a wholesale redefinition of what urban mobility is, and how it is provided."

—Adam Greenfield

London School of Economics

CityLab

In the past decade, transit providers and public agencies have made their real-time data public. With this data, tools have been developed so user can make more informed decisions. Decisions range from changing driving routes to changing departure times or even travel mode. In the early 2000s, transit agencies started providing real-time information on their services. As smartphones became more prominent, apps for turn-by-turn routing based on user information were developed. Today, multi-modal trip advising and planning apps are the norm for navigating big cities around the world. As people become more dependent on their phones for trip planning, apps are starting to offer incentives to reduce spacial and temporal saturations of transportation networks.

With the wide reach of smart phones, mobile applications have been emerging to provide real-time trip advice. Waze, for example, is an application through desktop and mobile platforms, such as Android, iPhone, and Blackberry. The app gives users turn-by-turn routing for autos based on information from other mobile users and user-reported events, such as accidents (https://www.waze.com/).

A study was conducted by researchers at Pennsylvania State University on the effectiveness of turn-by-turn advice. The study used a travel simulation to understand what affects a driver's likelihood to accept real-time advice. The mixed model created to understand compliance found that, while "freeway advice, turning advice, congestion occurrence, incident occurrence" played a large role, "subjects' spatial experience, subjects' temporal experience, and subjects' education level" also affected their likelihood to comply (Chen and Jovanis, 2014). Thus, their drivers will not always comply with the instructions given by trip advisors.

More recently, multi-modal models have been developed to make travel more efficient. Overall, an Advanced Traveler Advisory Tool (ATAT) is used to advise and guide users on multimodal trips with both path and mode choices. While trip planners tend to make suggestions based on real-time route information, trip advisors also allow the user to tailor the advice to their preferences.

According to Nuzzolo et al. (2014), there are three major types of trip planners and advisors:

1. rule-based, i.e. they refer to a selective approach in which a set of filters is applied to reduce the choice set of all feasible paths and remove generally unaccepted paths (e.g. those exceeding a maximum walk time or distance, number of changes, transfer time); such rules can be defined by the transport agency and/or by the user;

2. weighted time-based, i.e. they refer to paths individuated through a function of weighted time components (such as access, waiting, transfer on-board, and so on), with weights that can be defined by the transport agency and/or by the user;

3. utility -based, i.e. they refer to the path “cost” on the basis of the utility theory, with a utility function of path attributes associated to each alternative. The parameters, which should be calibrated, can be average values applied to all users or can be individual parameters tailored on the basis of personal user preferences (personal traveler advisory tools) (Nuzzolo et al., 2014)

While a number of multi-modal apps currently exist on the market, some come to mind for their presentation and coverage. RideScout started in November 2013 and is now available on desktop and for Android and iPhone. Covering many major cities throughout the United States, the app provides route options that list different modes, approximate cost, calories burnt, departure and arrival times, and trip duration.

Another trip planner is Citymapper, available for desktop, Android, and iPhone. It consolidates real-time information for practically all modes in the cities it covers. As of June 2015, it is available in London, Washington DC, San Francisco, México DF, Philadelphia, Vancouver, New York, Madrid, Chicago, Manchester, São Paulo, Montreal, Paris, Boston, Milan, Hamburg, Singapore, Berlin, Barcelona, Rome, Los Angeles, and Toronto. Other than allowing the user to set arrival and departure times, the app gives suggestions based on travel time and cost, mode choices, and calories burned. The app integrates transit, ride sharing, car sharing, auto, bike sharing, etc (https://citymapper.com/).

TripGo, on the other hand, is a trip advisor available for Android and iPhone. The app allows the user to set their relative priorities between saving money, saving time, the environment, and convenience. It then uses utility theory to make route suggestions. Suggestions tell you arrival time, trip duration, approximate cost and CO2 emissions. The app also allows you to select what modes you are willing to take. The app integrates transit, ride sharing, car sharing, auto, bike sharing, etc. Furthermore, TripGo also allows users to create agendas for their days. The app then creates routes and schedules to make sure you arrive on time.

Certain applications have also developed incentive systems to reduce congestion. Metropia, which is available for desktop and on Android and iPhone, currently works in Austin, TX and Tucson, AZ. Metropia essentially provides routes for commuting, but then offers incentives for people to take alternative routes and depart at different times to reduce saturating certain routes of the network. Awards include music online, gift cards to local and online shops, etc. The app also tracks how many pounds of CO2 you save and, through a partnership with American Forests, they plant trees based on your savings. According to internal data, 74% of Metropia users report saving time and 65% of users are willing to change their regular departure time. Users experience, on average, 20% reduction in travel times.

Since this type of technology is fairly new and evolving very rapidly, formal studies are lacking on the effects of these apps. However, it is believed that "[almost] all movement in a major city now begins with a phone" (Goldwyn, 2014). People depend on technology to get around in major cities where multi-modal apps provide an easily navigable approach towards the various choices presented in routing.

FOCUS OF THIS STUDY

Our primary objective is to develop a cyber-physical system that takes input from the users in regards to set preferences, and provides them an optimal route to reach their destination. Our trip advisor will be utility-based, tailored to the individual needs of the users.

TECHNICAL DESCRIPTION

CPS ARCHITECTURE

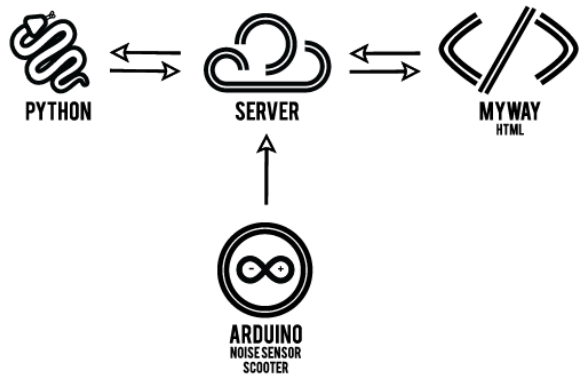

Figure 1: MyWay Cyber-Physical System Architecture

MyWay has four parts; hardware, software, a user interface, and a server. Together, they define the cyber-physical system needed for accurate and optimal trip advice. The hardware, software, and user interface each pull or push information onto the server, making the server the center of the architecture. MyWay's user interface pushes user preferences and origin-destination information, while sensors push data. The software, developed using Python, pulls this information and uses it to populate a utility function. It then pushes the optimal route onto the server, where MyWay's user interface retrieves it. The individual parts of this system will be further explained in the remainder of this report.

DEFINING ATTRIBUTES

MyWay considers four attributes in route choice that we feel take into consideration user preferences when traversing campus. They are defined as follows:

ATTRIBUTE	DEFINITION
Time	Time a route will take based on mode of travel
Scenery	Scenic value of a route
Required Energy	Required energy output of a route
Sociability	Congestion of a Route

Each attribute is weighed by users' preferences using the following utility model:

$$U(T,S,E,P) = T\begin{bmatrix} t_1 \\ t_2 \\ t_{\dots} \\ t_n \end{bmatrix} + S\begin{bmatrix} s_1 \\ s_2 \\ s_{\dots} \\ s_n \end{bmatrix} + E\begin{bmatrix} e_1 \\ e_2 \\ e_{\dots} \\ e_n \end{bmatrix} + P\begin{bmatrix} p_1 \\ p_2 \\ p_{\dots} \\ p_n \end{bmatrix}$$

where:

T	Importance of time, inputted by user on a scale of 0-10
S	Importance of scenery, inputted by user on a scale of 0-10
E	Importance of energy required, inputted by user on a scale of 0-10
P	Importance of sociability, inputted by user on a scale of 0-10
t_x	The time duration of route x
s_x	"Scenic Value" of route x, a normalized summation of each link's value
e_x	"Energy Required" by route x, a normalized summation of each link's value
p_x	"Sociability" of route x, a normalized summation of each link's value

The following section will discuss the derivation of links' values.

DETERMINATION OF ROUTES' VALUES

The attribute values of each route is defined by the summation of its individual links' values and then normalized. Normalizing the values guarantees each attribute is weighed the same in the utility function.

TIME

The "Time" attribute is the most straightforward. MyWay uses the time provided by Google Maps and Open Trip Planner when they are called for a specific origin-destination pair inputted by the user. They are then normalized using the following equation:

$$t_x = \frac{m_x}{\sum_{i=1}^{n} m_i}$$

where:

t_x	Time value of route x
m	time duration of specified route

SCENERY

"Scenery" is defined by MyWay by analyzing the foliage and architecture of campus. The campus is divided into 32 spaces, as shown in figure 2, with each space given a ranking from 1-5, 5 being the most beautiful.

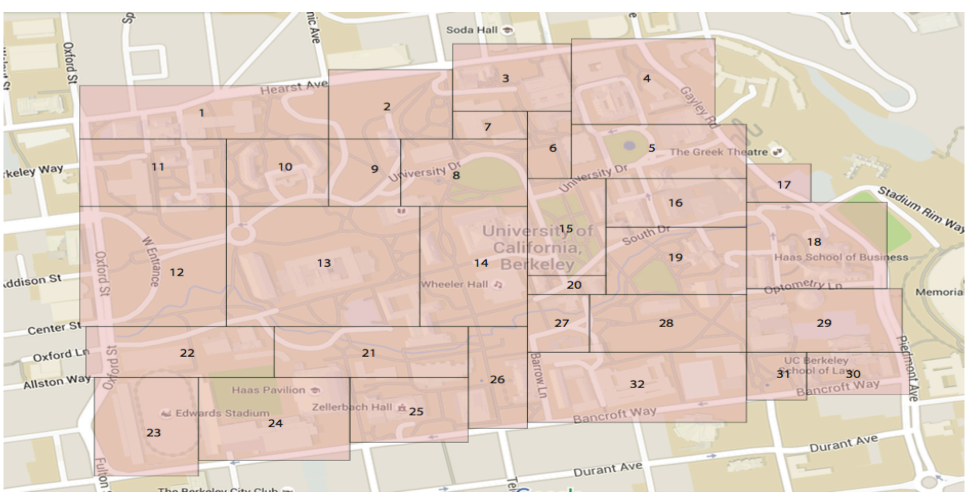

Figure 2: Map of defined regions for "Scenery" attribute

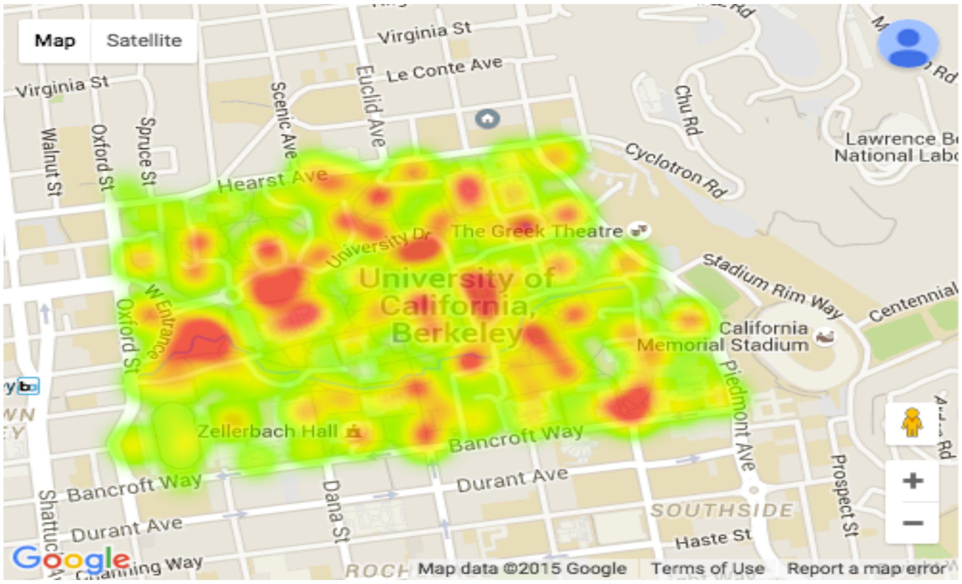

Figure 3: Heat Map of "Scenic" values around UC Berkeley's campus

The ranking process is by nature subjective, however; we try to minimize this by ranking regions with dense foliage, groomed landscape, or buildings with notable architecture the highest. The heat map in figure 3 visualizes flagged points on campus. We sum the scenic values of each region a route crosses, and normalize the data as follows:

$$s_x = \frac{\sum_{i=1}^{n} r_i}{5n}$$

where:

r_i	The "scenic value" of a region traversed in a given route, scale of 1-5
n	Number of regions traversed in a route

Rankings per region can be found in Appendix A.

ENERGY REQUIRED

The eScooter was utilized when quantifying the "Required Energy" per route. We connected an Arduino equipped with an SD card reader to the eScooter to collect voltage and current information as we traversed routes on campus. The routes traversed are shown in figure 4, and were chosen to show variations in steep, mild, and negligible slopes. The Arduino code used to collect data can be found in Appendix B. The routes were travelled three times, with the average values used in calculations.

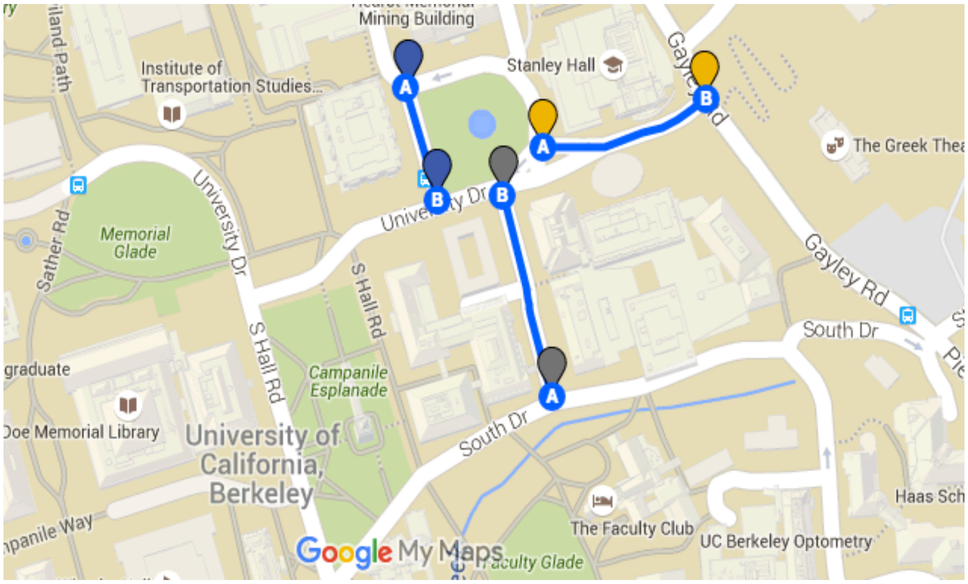

Figure 4: Map of routes traversed on eScooter

To make the data useful, a relationship between power and elevation gain must be found. The following equation was used to convert collected voltage and current data to power:

$$Power = Current * Voltage$$

Next, we integrated power by meters travelled for each of the routes to get $power - meters$ using trapezoidal numerical integration with MATLAB. The value found was divided by total distance travelled per route. This gave us three values that represent power required for a steep, mild, and negligible sloped link. These new values were then plotted against the individual routes' elevation gain, producing the graph shown below.

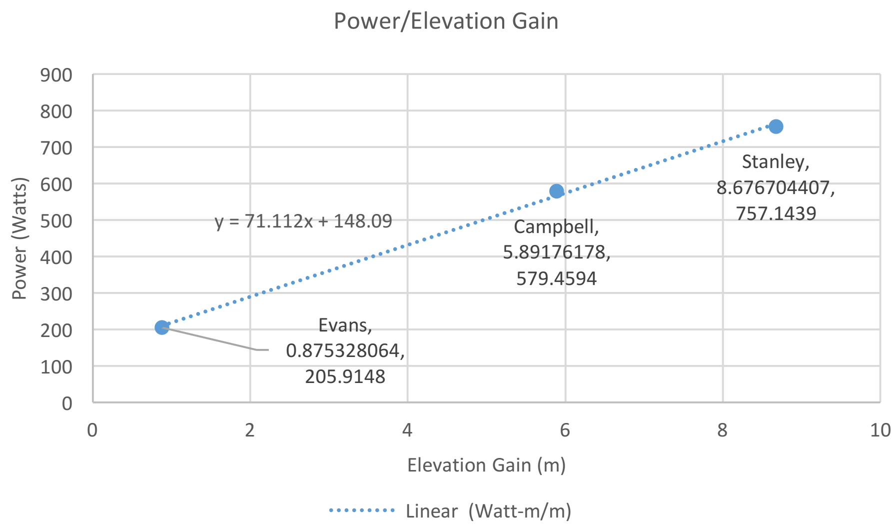

Figure 5: Power/Elevation Gain for the three routes traversed

The equation found by linear regression is used by MyWay to populate the "Energy Required" values by inputting the elevation gain of each link within the route, averaging the values along the route, and finally, normalizing them.

$$\overline{P_x} = \frac{\sum_{i=1}^{n}(71.112x_i + 148.09)}{n}$$

$$e_x = \frac{\overline{P_x}}{\sum_{i=1}^{k}\overline{P_i}}$$

where:

$\overline{P_x}$	Average power along route x
x_i	Elevation gain in link I along route x
n	Number of links in route x
k	Number of routes being compared

The MATLAB code used for integration can be found in Appendix C, and original data collected found in Appendix D.

SOCIABILITY

"Sociability" is the measure of the congestion of a route. MyWay considers two factors when determining the "Sociability" value. Firstly, we use current routing information. If we have populated a route, it lessens the likelihood of it being assigned again. Secondly, we use noise sensor data. We designed sound boxes using Arduino Uno boards to be placed around campus to measure noise levels; locations shown in figure 6. The microphones used can be seen in "Bill of Materials" found in Appendix F.

Figure 6: Map of sensor locations for noise collection

The noise levels were mapped from 0 to 20 for ease of comparison. The Arduino code can be found in Appendix E. Data was collected every 30 seconds from 6:00am-10:00pm. The graph in figure 7 depicts the ten minute averages for each region.

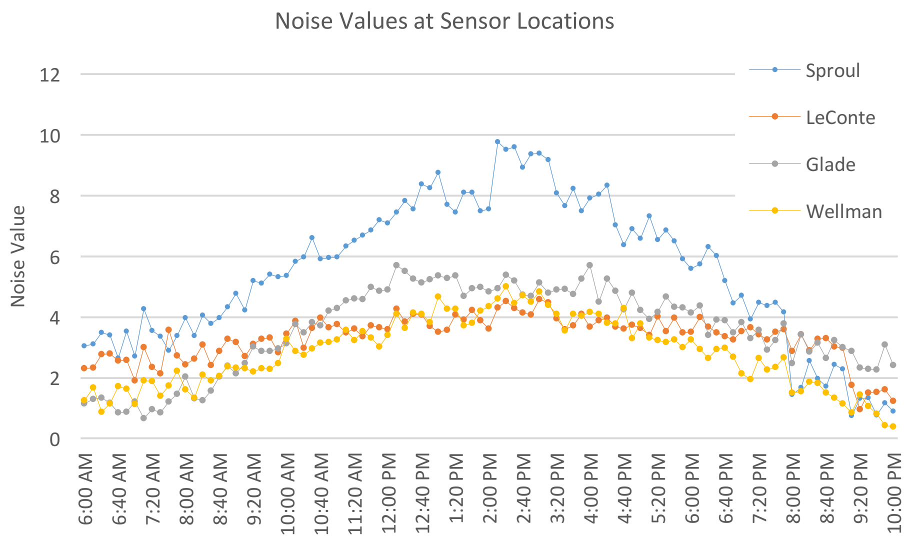

Figure 7: Plot of Noise Values for each Sensor Location

A correlation can be seen between a spike in noise levels and time between classes, 10 minutes till and 10 minutes past each hour. "Sociability" value is quantified using the following:

$$p_x = \left[\frac{\sum_{i=1}^{n} N_i}{20n} + \frac{u_x}{\sum_{i=1}^{k} u_i}\right] * \frac{1}{2}$$

where:

p_x	"Sociability" Factor of route x
N	"Real-time" noise value at each link
n	Number of links in route x
u	Number of users on a route
k	Number of routes being compared

The "real-time" noise values are highly simplified for proof of concept. Areas without a noise sensor are given the average value of the areas that do. Further, noise values are set to 0 between 10:00pm and 6:00am. Sensing data was only taken on a single day, with that data being used to mimic "real-time" data. However, to make the system dynamic in the future, sensors can be deployed full time for real-time data.

PYTHON – DATA ANALYSIS

The backend of MyWay was developed using Python. After receiving origin-destination and mode of travel data from the user, it calls Google and Open Trip Planner's API for various route choices. MyWay then picks a random way-point along the route and calls the APIs once again for the sake of variation, as depending on preferences a route that is not originally displayed may have a higher utility. Once the route options are called, formulating the utility function is a matter of calculating the various attributes discussed above. Links are determined by latitudes and longitudes provided by the API along the routes. The code determines optimal route choice and pushes it back to the server. The python code can be found on github.[1]

VISUALIZATION

MyWay's web platform is a straight-forward user interface. It has standard user inputs: mode of travel, origin, destination, time constraints, and preference settings. After it receives your inputs, it pushes them to the server, then retrieves your optimal route. The optimal route is presented on a google map, along with a table showing the utility of each of the compared routes. Noise data, elevations, and a heat map of scenery are displayed as well. The figures below are examples of MyWay's interface.

---

[1] https://github.com/joshzarrabi/e-mission-server/tree/ce186/emission/ce186

Figure 8: MyWay User Inputs

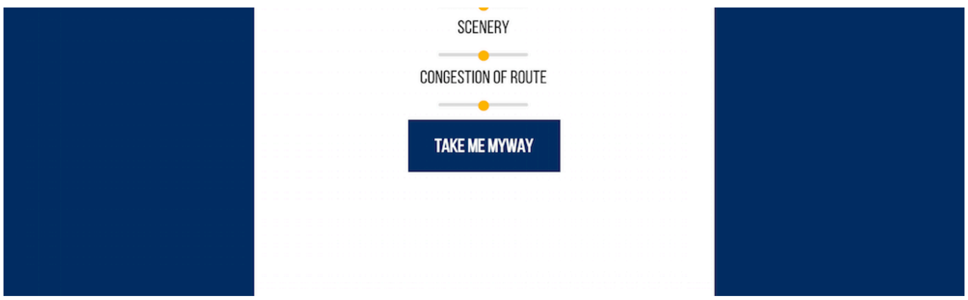

Figure 9: "Take Me MyWay"

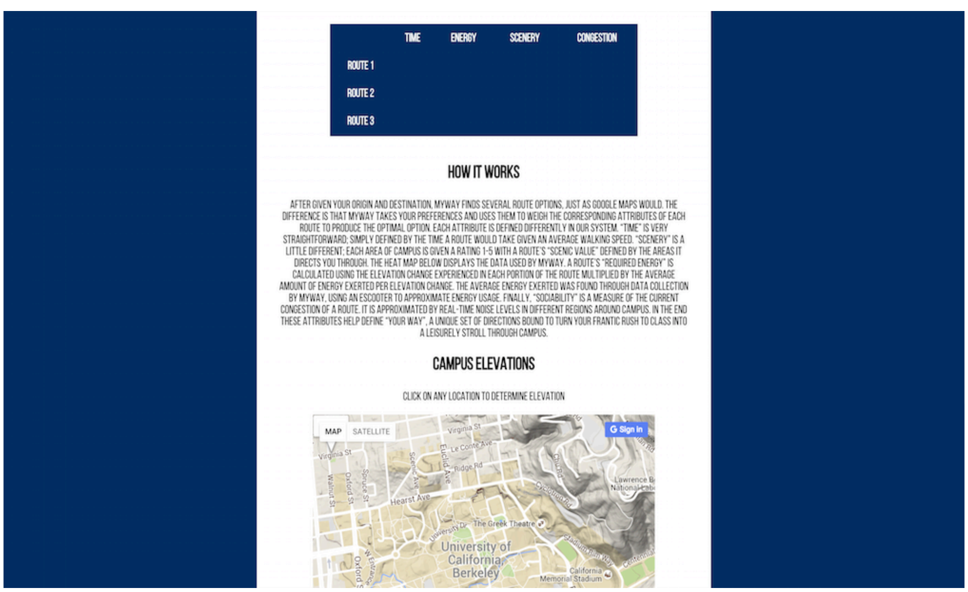

Figure 10: Table of Various Routes' Utilities and "How It Works"

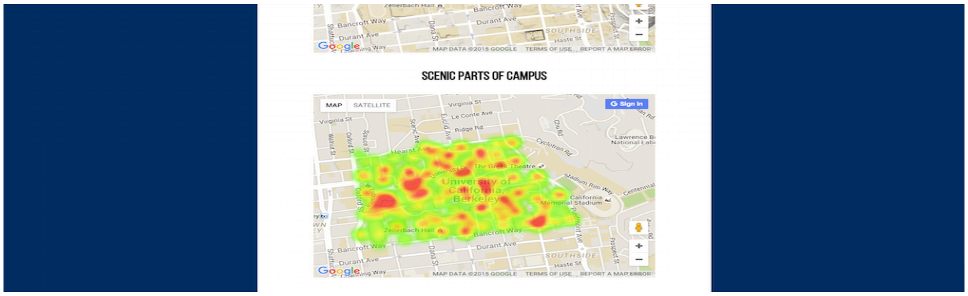

Figure 11: Heat Map of UC Berkeley's Scenery

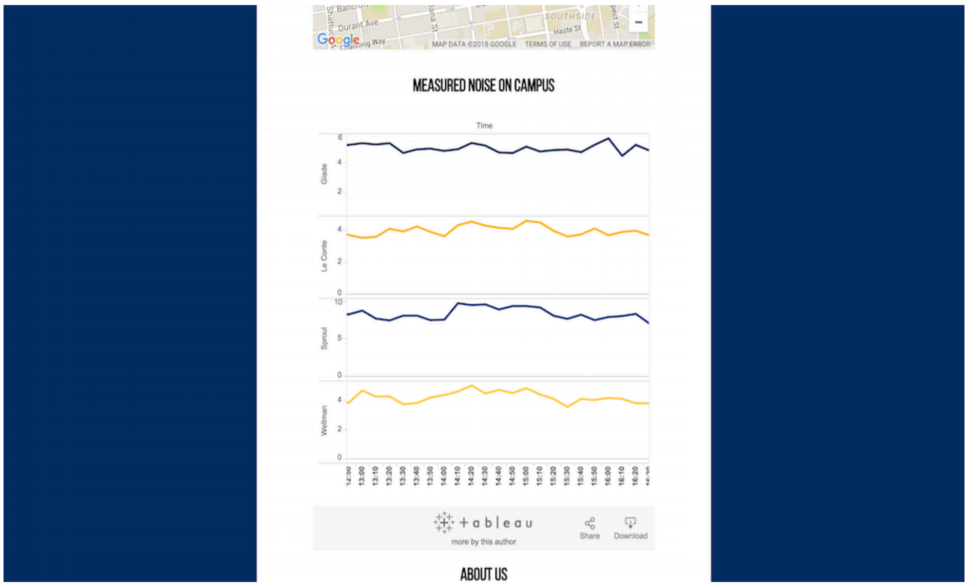

Figure 12: Visualization of Noise Data at Sensor Locations on Campus

DISCUSSION

MyWay is an attempt at addressing the lack of trip advisors for areas of a small scale. The void in this niche market can likely be attributed to the difficulties in determining users' desires and quantifying attributes within regions. Tackling these challenges for UC Berkeley was simplified by our experiences on campus and knowledge of the region. We were able to quickly define users' desires because we have experienced the qualms of making it to an interview on time, sweat free, and relaxed. However, in regions we are not familiar with traversing, such as international airports or national parks, this becomes much more of a challenge. Additionally, we spent many hours quantifying scenery through campus, collecting noise data, and developing a relationship between energy output and elevation gain. To apply this across several spaces would be tedious and inaccurate.

The issues with scalability are enough to keep MyWay merely a proof of concept. However, after spending time quantifying attributes on campus, it became clear that much of the work can be automated. For example, it would be possible to quantify scenery by making a heat map around buildings marked as historical landmarks or built by notable architects and foliage of a certain density. This could be validated by user input within the application. Further, sociability and congestion can be measured without the noise data once there are enough users on the system. Finally, required energy of a route is greatly simplified now that a linear relationship between elevation gain and power has been developed. Defining users' desires would require a case study for each region, but could be applied to all spaces of a similar definition, as airport travelers' needs are the same whether they are in LAX or SFO. In conclusion, with the help of automation, MyWay is scalable and can provide unique optimal routing suggestions in a variety of spaces.

SUMMARY

MyWay is a cyber-physical system that redefines students' relationship with campus by providing the optimal route choice given their origin, destination, mode choice, current needs, and preferences. We determined four quantifiable attributes on campus that we feel define students' needs: time, scenery, required energy, and sociability. By defining attributes of regions across campus and calling routes through Google and Open Trip Planner's APIs, utility theory can be employed to optimize routes. The system is made dynamic by bringing people into the loop. As preferences change, route suggestions change as well. Further, MyWay can manage congestion by noting current users' route assignments when optimizing a route, adding to its dynamic qualities. While MyWay was developed for UC Berkeley, its design is fully modular, allowing for easy implementation across spaces with the same set of attributes. MyWay serves as a proof-of-concept for trip advisors in regions of a small scale and proves there is feasibility in scalability across various spaces.

APPENDIX

APPENDIX A – SCENIC VALUES OF CAMPUS

SCENIC ATTRIBUTE DATA

RegionID	Top Left Lat	Top Left Long	Top Right Lat	Top Right Long	Bottom Right Lat	Bottom Right Long	Bottom Left Lat	Bottom Left Long	Scenic Value
1	37.875	-122.266	37.875	-122.262	37.875	-122.262	37.875	-122.266	2
2	37.875	-122.262	37.875	-122.260	37.875	-122.260	37.875	-122.262	5
3	37.875	-122.260	37.875	-122.258	37.874	-122.258	37.874	-122.260	1
4	37.876	-122.258	37.876	-122.256	37.874	-122.256	37.874	-122.258	2
5	37.874	-122.258	37.874	-122.255	37.873	-122.255	37.873	-122.258	4
6	37.874	-122.259	37.874	-122.258	37.873	-122.258	37.873	-122.259	1
7	37.873	-122.259	37.873	-122.259	37.875	-122.259	37.875	-122.259	5
8	37.875	-122.261	37.875	-122.259	37.873	-122.259	37.873	-122.261	3
9	37.875	-122.262	37.875	-122.261	37.873	-122.261	37.873	-122.262	2
10	37.875	-122.264	37.875	-122.262	37.873	-122.262	37.873	-122.264	2
11	37.875	-122.266	37.875	-122.264	37.873	-122.264	37.873	-122.266	5
12	37.873	-122.266	37.873	-122.264	37.871	-122.264	37.871	-122.266	4
13	37.873	-122.264	37.873	-122.260	37.871	-122.260	37.871	-122.264	3
14	37.873	-122.260	37.873	-122.259	37.871	-122.259	37.871	-122.260	5
15	37.873	-122.259	37.873	-122.257	37.871	-122.257	37.871	-122.259	1
16	37.873	-122.257	37.873	-122.255	37.872	-122.255	37.872	-122.257	2
17	37.873	-122.255	37.873	-122.254	37.873	-122.254	37.873	-122.255	2
18	37.873	-122.255	37.873	-122.253	37.871	-122.253	37.871	-122.255	3
19	37.872	-122.257	37.872	-122.255	37.871	-122.255	37.871	-122.257	4
20	37.871	-122.259	37.871	-122.257	37.871	-122.257	37.871	-122.259	1
21	37.871	-122.263	37.871	-122.260	37.870	-122.260	37.870	-122.263	5
22	37.871	-122.266	37.871	-122.263	37.870	-122.263	37.870	-122.266	3
23	37.870	-122.266	37.870	-122.264	37.868	-122.264	37.868	-122.266	2
24	37.870	-122.264	37.870	-122.262	37.868	-122.262	37.868	-122.264	2
25	37.870	-122.262	37.870	-122.260	37.869	-122.260	37.869	-122.262	3
26	37.871	-122.260	37.871	-122.259	37.869	-122.259	37.869	-122.260	3
27	37.871	-122.259	37.871	-122.258	37.870	-122.258	37.870	-122.259	2
28	37.871	-122.258	37.871	-122.255	37.870	-122.255	37.870	-122.258	3
29	37.871	-122.255	37.871	-122.252	37.870	-122.252	37.870	-122.255	3
30	37.870	-122.254	37.870	-122.252	37.869	-122.252	37.869	-122.254	2
31	37.870	-122.255	37.870	-122.254	37.869	-122.254	37.869	-122.255	3
32	37.870	-122.259	37.870	-122.255	37.869	-122.255	37.869	-122.259	2

APPENDIX B – ARDUINO CODE FOR ESCOOTER DATA COLLECTION

eScooter.ino

```
#include <SoftwareSerial.h>
#include <SPI.h>
#include <SD.h>

File eScooter;
//Call eScooter's Serial Port
SoftwareSerial radioSerial(2, 3); // RX=>DOUT, TX=>DIN

// Constant Variables
const int switchPin = 7;
const int chipSelect = 10; //Adafruit SD shields and modules: pin 10

//Changing Variables
int switchstate = 0;


void setup() {
  pinMode(switchPin, INPUT);
  radioSerial.begin(9600);
  Serial.begin(9600);
  Serial.println("radioSerial Initialized");
  //Ask eScooter for all values
  radioSerial.print("r;");
  //Ask eScooter for readings every 1 second
  radioSerial.print("t1;");
  pinMode(chipSelect, OUTPUT);
  if(!SD.begin(chipSelect)) {
      Serial.println("initialization failed!");
      return;
```

```
    }
  Serial.println("Initialization done.");
  delay(500);
  //Verify proper opening of SD file
  eScooter = SD.open("ES.txt", FILE_WRITE);
    if(SD.exists("ES.txt")){
      Serial.println("ES.txt exists.");
      eScooter.println("new");
    }else{
        Serial.println("ES.txt does not exist.");}
  eScooter.close();
}

void loop() {
  switchstate = digitalRead(switchPin);
  //Check eScooter Serial for available data
  if (radioSerial.available()>0){
    // If switch is on - check eScooter Serial Port
    if (switchstate == HIGH) {
      eScooter = SD.open("ES.txt", FILE_WRITE);
      if(SD.exists("ES.txt")){
        while (radioSerial.available()>0){
          char i = radioSerial.read();
          Serial.print(i);
          eScooter.print(i);
        }


      }
      eScooter.close();
    }else{
      while (radioSerial.available()>0){
        char i = radioSerial.read();
```

```
        //Serial.print(i);
      }
    }


  }
  delay(1);
}
```

APPENDIX C – MATLAB CODE FOR ENERGY INTEGRATION

eScooter.m

```
%% Scooter

%%T1 = Evans flat average power output
T1 = [948.7912667
735.9049667
610.22
440.2547
406.4888667
246.8261333
220.6164333
225.9810333
245.0238
199.4365
91.00373333
1.235566667
2.347333333
2.223966667
2.352733333
1.6129
1.6791
1.735066667
1.9864
2.111466667]; %Watts

X1 = 0:45.4207424942/(length(T1)-1):45.4207424942; %m -- length of route provided by Google API
PM1 = trapz(X1,T1); %% Power-meter for Route 1
PM1_M = PM1/45.4207424942; %%power-meter/meter

%%T2 = Campbell uphill average power output
T2 = [869.387
988.8548
981.2432333
911.2304
733.8487667
```

```
639.9328667
614.1795
619.6322667
643.8648333
673.7056
698.2148333
706.4883667
722.5094333
692.2248667
604.2137
585.9493667
536.5135667
424.9014667
446.4665667
450.4204333
244.7841333
231.4812667
178.9743333
110.5109333
64.37376667]; %%Watts

X2 = 0:72.5655658424/(length(T2)-1):72.5655658424; %m -- length of route
provided by Google API
PM2 = trapz(X2,T2); %Power-meter for Route 2
PM2_M = PM2/72.5655658424; %power-meter/meter

%%T3 = Stanley uphill average power output
T3 = [941.2789667
978.6522333
983.7965667
978.4417667
934.5746333
880.7101667
800.9847333
751.8444
725.4330667
704.6720333
667.3675667
```

```
639.6788667
646.8751
656.3771
691.6823
704.2643667
734.2714
767.7138667
816.6138667
839.6193667
844.3856
877.1343333]; %%Watts

X3 = 1:114.177218445/length(T3):114.177218445;%m -- length of route provided
by Google API
PM3 = trapz(X3,T3); %%Power-meter for Route 3
PM3_M = PM3/114.177218445; %power-meter/meter

ELEVGAIN = [0.8753280639648011; 5.891761779785199; 8.67670440673831]; %%
elevation gain for each trip [evan campbell stanley]
```

APPENDIX D – ORIGINAL DATA COLLECTED FROM ESCOOTER

TRIP 1 - EVANS (NEGLIGIBLE INCLINE)

Trial 1			Trial 2			Trial 3			
Voltage (Volts)	Current (Amps)	Power (Watts	Voltage (Volts)	Current (Amps)	Power (Watts	Voltage (Volts)	Current (Amps)	Power (Watts	Avg Power (Watts)
33.84	29.03	982.38	33.30	30.00	999.00	33.71	25.66	865.00	948.79
33.49	28.35	949.44	33.08	29.71	982.81	35.59	7.74	275.47	735.90
34.24	20.40	698.50	33.27	27.76	923.58	35.84	5.82	208.59	610.22
35.61	8.91	317.29	34.32	17.83	611.93	35.18	11.13	391.55	440.25
35.75	8.07	288.50	34.98	12.76	446.34	34.69	13.97	484.62	406.49
35.76	8.38	299.67	35.21	10.64	374.63	36.36	1.82	66.18	246.83
36.00	6.72	241.92	35.30	10.15	358.30	36.47	1.69	61.63	220.62
35.99	6.75	242.93	35.38	9.68	342.48	36.43	2.54	92.53	225.98
35.98	6.81	245.02	36.69	0.35	12.84	33.91	21.97	745.00	245.02
36.46	3.31	120.68	36.88	0.10	3.69	34.67	13.67	473.94	199.44
36.95	0.85	31.41	36.96	0.06	2.22	35.57	6.73	239.39	91.00
37.12	0.04	1.48	37.09	0.05	1.85	36.74	0.01	0.37	1.24
37.24	0.05	1.86	37.09	0.08	2.97	36.88	0.06	2.21	2.35
37.25	0.05	1.86	37.10	0.05	1.86	36.93	0.08	2.95	2.22
37.31	0.06	2.24	37.23	0.06	2.23	36.94	0.07	2.59	2.35
37.39	0.03	1.12	37.24	0.05	1.86	37.10	0.05	1.86	1.61
37.40	0.06	2.24	36.19	7.75	280.47	37.14	0.03	1.11	1.68
37.40	0.04	1.50	36.96	0.04	1.48	37.18	0.06	2.23	1.74
37.39	0.05	1.87	37.09	0.04	1.48	37.23	0.07	2.61	1.99
37.40	0.06	2.24	37.09	0.04	1.48	37.24	0.07	2.61	2.11

TRIP 2 – CAMPBELL HALL (MILD INCLINE)

Trial 1			Trial 2			Trial 3			
Voltage (Volts)	**Current (Amps)**	**Power (Watts**	**Voltage (Volts)**	**Current (Amps)**	**Power (Watts**	**Voltage (Volts)**	**Current (Amps)**	**Power (Watts**	**Avg Power (Watts)**
33.40	29.38	981.29	33.36	28.65	955.76	33.98	19.75	671.11	869.39
33.20	30.14	1000.65	33.07	29.64	980.19	32.77	30.08	985.72	988.85
33.19	30.09	998.69	32.98	29.99	989.07	32.84	29.11	955.97	981.24
33.33	28.54	951.24	33.03	29.30	967.78	33.32	24.45	814.67	911.23
34.01	21.97	747.20	34.02	20.03	681.42	33.46	23.10	772.93	733.85
34.46	17.96	618.90	34.19	18.62	636.62	33.84	19.63	664.28	639.93
34.59	16.89	584.23	34.19	18.58	635.25	34.01	18.32	623.06	614.18
34.41	18.44	634.52	34.13	19.18	654.61	34.22	16.65	569.76	619.63
34.35	18.85	647.50	34.11	19.20	654.91	34.01	18.50	629.19	643.86
34.32	19.25	660.66	33.96	20.48	695.50	33.84	19.65	664.96	673.71
34.17	20.52	701.17	33.99	19.99	679.46	33.68	21.20	714.02	698.21
34.13	20.61	703.42	33.90	20.95	710.21	33.74	20.92	705.84	706.49
34.12	20.78	709.01	33.67	23.01	774.75	33.65	20.32	683.77	722.51
33.95	22.31	757.42	34.16	18.53	632.98	33.69	20.37	686.27	692.22
34.41	18.11	623.17	34.38	16.64	572.08	33.96	18.18	617.39	604.21
34.50	17.46	602.37	34.34	17.20	590.65	34.17	16.53	564.83	585.95
34.50	17.41	600.65	34.71	13.99	485.59	34.36	15.23	523.30	536.51
34.82	14.71	512.20	35.64	6.55	233.44	34.31	15.42	529.06	424.90
34.84	14.73	513.19	35.55	9.25	328.84	34.42	14.45	497.37	446.47
35.43	9.64	341.55	34.68	14.92	517.43	34.45	14.29	492.29	450.42
36.85	0.53	19.53	35.68	6.65	237.27	34.53	13.83	477.55	244.78
35.48	10.67	378.57	35.54	7.94	282.19	36.22	0.93	33.68	231.48
35.71	8.14	290.68	35.68	6.84	244.05	36.54	0.06	2.19	178.97
35.75	7.75	277.06	36.55	1.40	51.17	36.67	0.09	3.30	110.51
36.30	3.92	142.30	36.56	1.36	49.72	36.79	0.03	1.10	64.37

TRIP 3 - STANLEY (STEEP INCLINE)

Trial 1			Trial 2			Trial 3			
Voltage (Volts)	Current (Amps)	Power (Watts	Voltage (Volts)	Current (Amps)	Power (Watts	Voltage (Volts)	Current (Amps)	Power (Watts	Avg Power (Watts)
33.75	26.03	878.51	33.10	29.18	965.86	32.78	29.88	979.47	941.28
33.21	29.64	984.34	32.96	29.70	978.91	32.63	29.81	972.70	978.65
33.11	29.87	989.00	32.89	30.01	987.03	32.61	29.91	975.37	983.80
33.11	29.83	987.67	32.90	29.68	976.47	32.59	29.80	971.18	978.44
33.28	28.10	935.17	32.99	28.78	949.45	32.79	28.03	919.10	934.57
33.40	26.99	901.47	33.28	26.10	868.61	32.97	26.45	872.06	880.71
33.70	23.97	807.79	33.46	24.34	814.42	33.28	23.46	780.75	800.98
33.85	22.65	766.70	33.71	22.17	747.35	33.40	22.20	741.48	751.84
33.92	21.73	737.08	33.69	21.92	738.48	33.56	20.88	700.73	725.43
34.02	21.08	717.14	33.85	20.65	699.00	33.60	20.77	697.87	704.67
34.15	19.70	672.76	33.93	19.77	670.80	33.72	19.53	658.55	667.37
34.31	18.39	630.96	34.07	18.79	640.18	33.78	19.18	647.90	639.68
34.26	18.99	650.60	34.06	18.84	641.69	33.75	19.21	648.34	646.88
34.20	19.43	664.51	34.05	18.98	646.27	33.71	19.53	658.36	656.38
34.02	20.85	709.32	33.93	20.03	679.62	33.60	20.42	686.11	691.68
34.01	20.93	711.83	33.88	20.26	686.41	33.50	21.33	714.56	704.26
33.89	21.98	744.90	33.78	21.10	712.76	33.40	22.31	745.15	734.27
33.74	23.23	783.78	33.60	22.57	758.35	33.29	22.86	761.01	767.71
33.64	24.44	822.16	33.45	24.08	805.48	33.10	24.84	822.20	816.61
33.56	25.12	843.03	33.33	25.21	840.25	33.04	25.29	835.58	839.62
33.55	25.11	842.44	33.29	25.47	847.90	33.00	25.54	842.82	844.39
33.41	26.27	877.68	33.19	26.29	872.57	32.83	26.84	881.16	877.13

APPENDIX E – ARDUINO CODE FOR NOISE DATA COLLECTION

eNoise.ino

```
#include <SoftwareSerial.h>
#include <SPI.h>
#include <SD.h>

//Create File
File eNoise;

// Constant Variables
const int NoiseSensor = A0;
const int chipSelect = 10; //Adafruit SD shields and modules: pin 10

//Changing Variables
float Noise = 0;
int Count = 0;
float MapNoise = 0;

void setup() {
  pinMode(NoiseSensor, INPUT);
  Serial.begin(9600);
  Serial.print("Initializing SD card...");
  pinMode(chipSelect, OUTPUT);
  if(!SD.begin(chipSelect)) {
    Serial.println("initialization failed!");
    return;
  }
  Serial.println("Initialization done.");
  delay(500);
  eNoise = SD.open("eN.txt", FILE_WRITE);
  if(SD.exists("eN.txt")){
```

```
    Serial.println("eN.txt exists.");
    eNoise.println("new");
  }else{
      Serial.println("eN.txt does not exist.");}
  eNoise.close();
  }

void loop() {


  Serial.println("Writing to eN.txt...");
  //Collect every 30 seconds for 12 hours
  if(Count<1440) {
  Noise = analogRead(NoiseSensor);
  //Map Noise Values to scale of 0-20
  MapNoise = Noise*20/1023;
  Serial.println(MapNoise);
  eNoise = SD.open("eN.txt",FILE_WRITE);
    if(SD.exists("eN.txt")){
      eNoise.println(MapNoise);
      delay(30000); //Collect data every thirty seconds
      eNoise.close();
    }else {
    }
  Count = Count + 1;
  Serial.println("Done.");
  }
  delay(1);
}
```

APPENDIX F – BILL OF MATERIALS

ITEM	QTY	COST
eScooter equipped with Arduino	1	CEE 186 Equipped
Data Logger Shield	4	CEE 186 Equipped
SparkFun Electret Microphone Breakouts	4	$36.02
MicroSD Card	5	CEE 186 Equipped
Battery Packs	4	CEE 186 Equipped
MicroSD Card Reader	1	CEE 186 Equipped
Arduino Uno Boards	5	CEE 186 Equipped

WORKS CITED

Chen, W., & Jovanis, P. (2014), *Driver En Route Guidance Compliance and Driver Learning with Advanced Traveler Information Systems: Analysis with Travel Simulation Experiment*. Transportation Research Board. < http://trrjournalonline.trb.org/doi/abs/10.3141/1843-10>

Goldwyn, E. (2014). The Most Important Transportation Innovation of the Decade Is the Smartphone. CityLab. <http://www.citylab.com/commute/2014/09/the-most-important-transportation-innovation-of-this-decade-is-the-smartphone/379525/>

Nuzzolo, A., Comi, A., Crizalli, U., & Rosati, L. (2014). *A new Advanced Traveler Advisory Tool based on personal user preferences.* 2014 IEEE 17th International Conference on Intelligent Transportation Systems (ITSC). <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6957915>