Bike Smart

A redefined Philadelphia bike route recommendation app

https://kristenzhao.github.io/BikeRouting/

University of Pennsylvania

April 28th 2017

Jianting Zhao

A. Abstract

My interests in conducting bike score evaluation and route recommendation for Philadelphia stem from my concern about bikers' safety on Philadelphia streets. I introduced the concept "Low stress street biking" (Furth & Mekuria, 2013), which about reducing bikers' stress level and improve biking safety.

In this project, my first goal is to use this concept as a guideline to create a rating system that reflects the bikeability of each street in Philadelphia. More specifically, I will use the street information from Opendataphilly.org and conduct a weighted overlay analysis. And my second goal is to create a useful App that recommends bikers the optimal bike routes that minimizes stress level. I would accomplish this goal by using a my bike score outcome and a handful of APIs and programming languages including Google Map Javascript API.

To summarize, I hope that the bike scores can be used as a decision-making tool for the city planners to prioritize bike infrastructure improvements, and that the application can be useful for bikers.

B. Introduction

As car-dependent urban planning raise traffic and health related issues, many cities are thriving to promote walking and biking. Philadelphia is known for its walkability, and people think that it is highly bikeable as well. As I found out from the Internet, many media rank Philadelphia high in bikeability (Figure 1 to 4). However, through my personal biking experience in Center City and University City,

Figure 1. Neighborhood Ranking from movoto.com (Left-Bottom)
Figure 2. News from phillymag.com (Right-Top)
Figure 3. Bike Score Ranking from forbes.com (Right-Middle)
Figure 4. Bikeable Streets Ranking from Redfin.com (Right-Bottom)

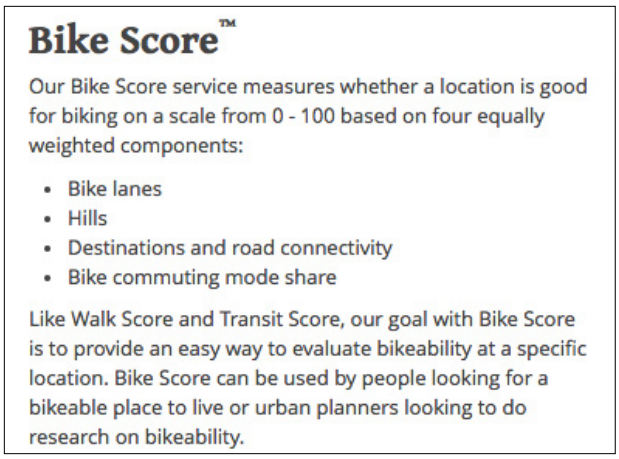
1. University City
2. Kensington/Port Richmond
3. Fishtown
4. Cedar Park (tie)
4. Manayunk (tie)
6. Mantua
7. Roxborough
8. Chestnut Hill
9. Brewerytown (tie)
9. Mill Creek (tie)

Clearly, Chestnut Hill wasn't the only bikeable part of the city. We came up with this list of 10 by looking at Philadelphia's neighborhoods in terms of these bike-friendly criteria:

- Bike stores per capita
- Parks per capita
- Bike Score (the higher, the better)

Bike Score Ranking of Large U.S. Cities

Rank	City	Bike Score
1	Minneapolis, MN	81.3
2	San Francisco, CA	75.1
3	Portland, OR	72.0
4	Denver, CO	71.3
5	Boston, MA	70.3
6	Chicago, IL	70.2
7	Washington, D.C.	69.5
8	Sacramento, CA	68.9
9	Tucson, AZ	67.9
10	Philadelphia, PA	67.5



Bike Score™

Our Bike Score service measures whether a location is good for biking on a scale from 0 - 100 based on four equally weighted components:

- Bike lanes
- Hills
- Destinations and road connectivity
- Bike commuting mode share

Like Walk Score and Transit Score, our goal with Bike Score is to provide an easy way to evaluate bikeability at a specific location. Bike Score can be used by people looking for a bikeable place to live or urban planners looking to do research on bikeability.

Figure 5. Methodology from Walkscore.com

I always felt frightened by fast moving cars being so close to me and did not enjoy the city biking experience. Therefore, I am curious how they came up with the rating. A prevalent website that provides bike scores, walkscore.com, only count limited factors into the rating system and applies with equal weights (Figure 5). This rating metric does not reflect real user experience. I hope to create a redefined rating system that reflects user experience better. After intense research, I locked down the "low stress street biking" concept (Furth & Mekuria, 2013).

This concept is a measure of how stressful a street is for bikers and how to appoint the best route for people with different stress level tolerance. Based on this concept, I gathered street lane type, topography, crash density and intersection control density information. Due to the absence of real-time traffic speed and volume data, stressfulness is reflected in lane type and crash density.

C. Literature Review

In the Low-Stress Bicycling and Network Connectivity article, Furth and Mekuria study how bikers' street condition demand change based on different stress tolerance levels (2013). The low-stress mainly refers to the traffic speed and amount. They propose a new classification scheme for roads with four levels of traffic stress (LTS). The LTS 1 is defined as a road that is "presenting little traffic stress and demanding little attention from cyclists, and attractive enough for a relaxing bike ride... Suitable for adults and children with all skill levels." LTS 2 is suitable for adults but not children. LTS 3 has more traffic stress than LTS 2, but usually offering cyclists a protected bike lane, making it suitable for most adults. And lastly, LTS 4 is anything beyond LTS 3. Although there are four levels of stressfulness, Furth and Mekuria show that less than 8% of Portland residents are confident to bike; in contrast, the rest interviewees either concern about it or simply refuse to do so. This finding reinforces my desire to create a bike rating that reflect the stress level to people who do not tolerate much traffic stress.

In addition, Furth and Mekuria state a list of factors that are not traffic but also add stress. They include steep hills, pavement quality and crime danger. Due to the data availability, I incorporated topography as part of my street stress level, i.e. bike score, measurement.

How steep is a hill not bikeable? According to Bicycle Networks (2017), an Australian organization, for 5% slope people can climb around 800ft, and anything beyond 10% slope is generally non-bikeable. This concept help me create a scoring guideline for slopes.

D. Data Gathering, Cleaning and Score definition

I gathered Philadelphia region DEM data from USGS, and street network, 2011-2014 crash counts and intersection controls shapefiles from Opendataphilly.org, and bike trail shapefile from PASDA.PSU.edu.

D.1 Slope Score

For the Philadelphia region DEM, I used "mosaic to new raster" and "extract by mask" tools in ArcGIS to create a continuous DEM for only Philadelphia County. Then I used "slope" function to obtain the slope for each pixel. Lastly, to get the slope for each street segments, I applied "zonal statistics

	Rowid	SEG_ID *	COUNT	AREA	MEAN
▸	1	1	1387	1387	5.94742
	2	2	1366	1366	9.703965
	3	3	872	872	4.625508
	4	4	984	984	7.924664
	5	5	524	524	7.72427
	6	6	1682	1682	2.774942
	7	7	761	761	5.274012
	8	8	617	617	4.200835

Figure 6. Average Slope per Segment

as table" on the slope raster, and obtained the average slope on each street segment in the street network shapefile (figure 6). Once I get the slopes, I set up a 0-100 scale to score the slopes. Below is my conversion:

Slope (%)	Score
0-1	100
1.1-2	90
2.1-3	80
3.1-4	70
4.1-5	60
5.1-6	50
6.1-7	40
7.1-8	30
8.1-9	20
9.1-10	10
> 10.1	0

D.2 Street Lane Type Score

As the base layer, street network shapefile contains all types of streets for the entire City of Philadelphia, ranging from small roads to highways. I cleaned it up by removing the “high speed ramp”, “highway”, “regional” and “boarder” street types, due to the fact that they are not bikeable by their nature. This still leaves most of the street on map. However, this shapefile does not include bike trails. Therefore I joined the bike trail shapefile with street network by SEGID field.

In these streets, the bike lane types range from the most unsafe one “none” to the safest “trail”. I defined a 0-100 range score for each type according to their separateness from mainstream traffic.

Type	Score
Trail	100
Buffered with Conventional	90
Buffered	80
Contraflow with Conventional	70
Conventional with Sharrow	60
Conventional	50
Sharrow	40
NULL	30

D.3 Crash Density

Bicycle crash counts can be an indication of how safe a street is. Therefore, I incorporated it in the bike score evaluation. Based on the crash count data from 2011- 2014, I first filtered out the records that was involved with bicycles, and then removed the cases with incomplete information. I spatially

joined the crashes to street network, and used "summarize" function to get the number of crash on each street segment. At last, the crash density is calculated by crash count/segment length. In order to normalize the data to the 0-100 scale, I divided the density range by standard deviation.

Crash Density	Score
= 0	100
<=0.0007	90
<=0.0016	80
<=0.0025	70
<=0.0034	60
<=0.0042	50
<=0.0051	40
> 0.0051	30

D4. Street Segment Connectivity

Street connectivity is an important measure for bikeability. It is defined by the number of junctions. For instance a street segment as shown in blue in figure 7 has 6 connected roads. To calculate the connectivity, I used the "line and junction connectivity tool" created by Ibeale and spatially joined it with the street segments to calculate the number of lines that each segment connects to. However,

Figure 7. Connectivity Demonstration

this tool misinterpreted junction locations and reported 0 connectivity for most bike trails, which leads to an underestimation. Finally, I normalized the level of connectivity to the 0-100 range:

Number of junctions	Score
>= 10	100
9	90
8	80
7	70
6	60
5	50
4	40
3	30
2	20
1	10
0	0

D5. Intersection Control Density

Intersection control density refers to how often you have to stop. Biking on Chestnut St, for example, is very painful because you have to stop frequently and you just cannot enjoy the smooth biking experience. Therefore, I take it into consideration. Similar to calculating crash count density, I first spatially join intersection controls to street segments, then summarize the number of controls per

segment, and finally calculate the density by street segment length / number of signal controls in that segment. The score is calculated by standard deviation divisions. Since there is no signal controls for bike trails, they get score 100.

Ctrl Density	Score
Trails	100
>0.38	80
0.32-0.38	70
0.27-0.32	60
0.21-0.27	50
0.15-0.21	40
0.09-0.15	30
0.03-0.09	20
<= 0.03	10

D6. Final Score

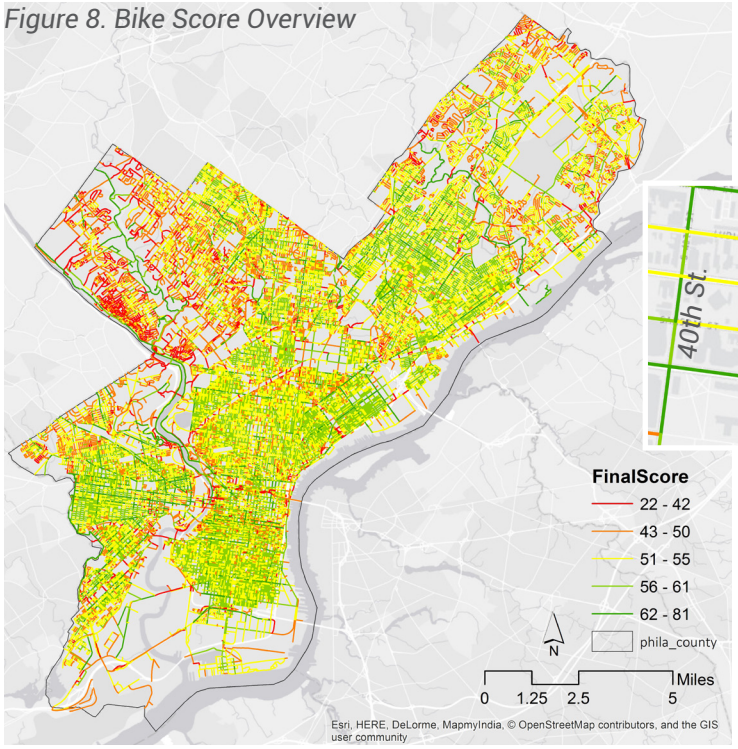
Final score is calculated as a weighted average of all other scores. The Lane Type Score is considered to be the most important, because it directly relates to biking experience and safety. Therefore I applied 40% weight on it. Following is slope score, which is also deterministic towards bikeability. Lastly the intersection control density, crash density and connectivity are evenly weighted, 10% each.

1. Bike Score = Lane Type Score * 40% + Slope Score * 30% + Intersection Control Density * 10% + Crash Density * 10% + Connectivity * 10%

E. Bike Score Result

By overlaying the five criteria with their assigned weights, I get the bike score for each street segment (figure 8). Figure 8 shows that the bike score is high in central urban area, but low at the Manayunk Neighborhood area. Figure 9 shows a zoomed in view on the streets. It's worth pointing out that the even on the same street, the scores on different segments are very disconnected.

Figure 8. Bike Score Overview



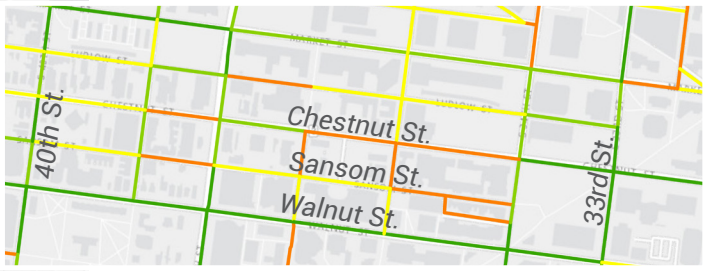


Figure 9. Zoomed-in View

F. Building the Bike Smart Application

Not only do I want to study the bike score in Philadelphia, but I also want to create a routing recommendation using the study result to help bikers find their best route. Having searched about the routing algorithms such as the network analyst and cost distance analysis in ArcGIS and the Dijkstra's algorithm, I found them to be either too slow to process or too challenging for me to learn in just a few weeks. Fortunately, Google Maps provides application program interface (API) that I can request route planning information from. Even though I would not be able to customize their routing algorithm, I can at least build my application on top of theirs. Therefore, my application would be able to retrieve route alternatives provided by Google and to calculate the bike score for each route. Users can choose the routes based on the score, distance and time.

Bike Smart is composed of three components: the core, the spatial query and the web framework.

F.1 Core

The core is the APIs that I use to retrieve the routing data. More specifically, I used the Google Maps Javascript API, Directions API and Places API Web Services as the supporting algorithm. With these APIs, I am able to create a map that I can insert an origin and destination pair and return with 3 alternative bike routes.

However, the routes are encoded in polyline encoding, something like this "_p~iF~ps|U_ulLnnqC_mqNvxq`@". In order to decode them, I used Ismaels' decoding function to get the coordinates of the polyline. Afterwards, I converted the coordinates into Well-known text polyline, which has the format like this: 'LINESTRING(-75.14777 39.9473, -75.14913 39.94747, -75.1495 39.94751, -75.14944 39.94785, -75.14937 39.94823, -75.1492 39.94896, -75.14919 39.94904, -75.14922 39.94915, -75.14922 39.94922, -75.14921 39.9493, -75.14892 39.95052)'. The route polylines are ready to use for SQL queries.

F.2 Carto SQL API

In order to conduct spatial analysis on the web, I utilize Carto SQL API. Carto is a web platform to store databases, and its SQL API allows users to query data in JavaScript.

I first stored my bike score shapefile in a database, and then used ST_DWithin, ST_GeomFromText and ST_Centroid functions to conduct the query. Below is my query structure:

```
SELECT *
FROM allscore_length AS L
WHERE ST_DWithin(ST_GeomFromText('LINESTRING(-75.19503 39.95318,
-75.19621 39.95332, -75.19677 39.95339, -75.19661 39.95415, -75.19646
39.95492)')::geography,st_centroid(L.the_geom)::geography,5)
```

The query asks for the street segments that their centroid is within 5 meters from the route polyline. Using this method, I can retrieve all the street segments that are involved in a route and calculate the route score. More specifically, the route score is calculated as the weighted average of the street segment length and segment score. As shown in figure 10, shorter segments weighs less and longer segments weighs heavier.

Figure 10. Score Calculation Algorithm

F3. Web Framework

I used html, css and javascript to construct the framework. The libraries that I used are Material Design Lite, jQuery and underscore.js. Figure 11 is the initial stage of the application; it is a simple and clean interface. When you click on the ☰ button, the input boxes will appear , as shown in figure 12, waiting for user interaction. Once you enter the addresses, the server will return several panels with trip information (figure 13), depending on how many trips were recommended by Google Maps API. In this example, there are three routes recommended.

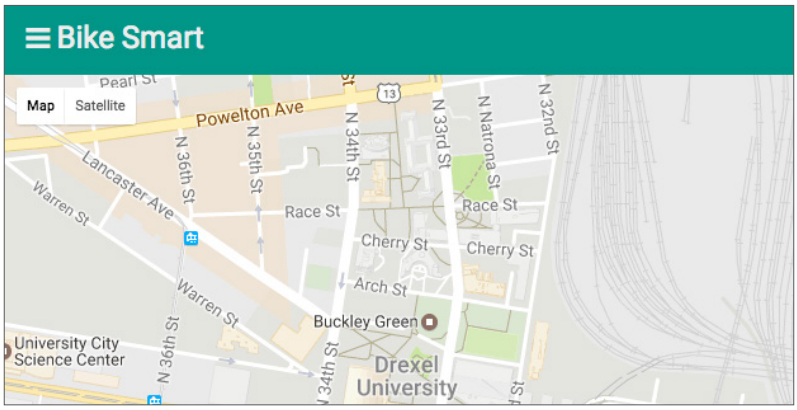


Figure 11. Homepage Interface

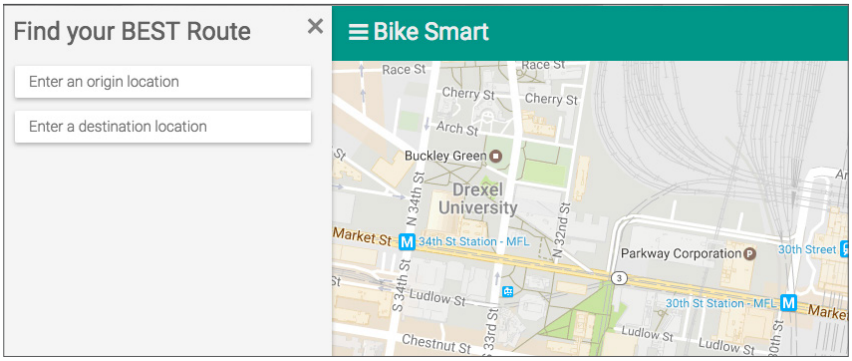


Figure 12. User Input Interface

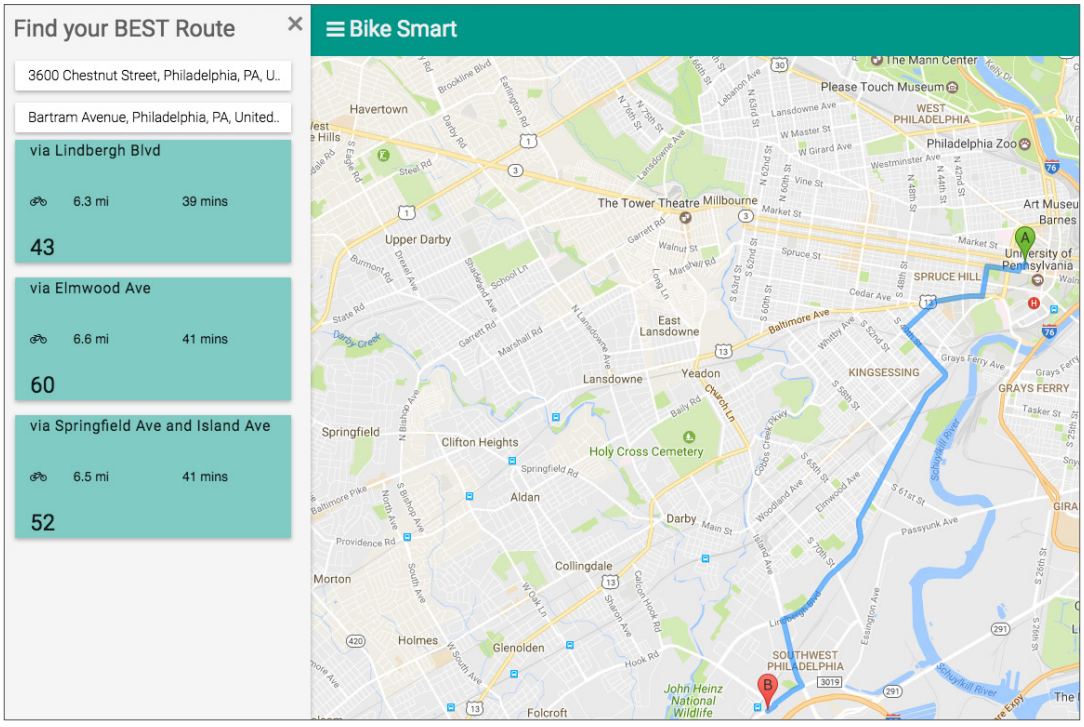


Figure 13. Result Page

G. Application Functionality Discussion

G.1 Algorithm Limitations

When using SQL query, it queries the street segments without differentiating the direction. Therefore, it selected several polylines that are irrelevant to the route. For example, figure 14 shows a route that I am trying to query, figure 15 is when I query using ST_DWithin for any polyline within a 20-meter search distance, and figure 16 is when I minimized the search distance to 5-meter. Minimizing the search distance helps reduce selection error, but the error could be avoided if there was a smarter query function that would allow me to filter the directions as well.

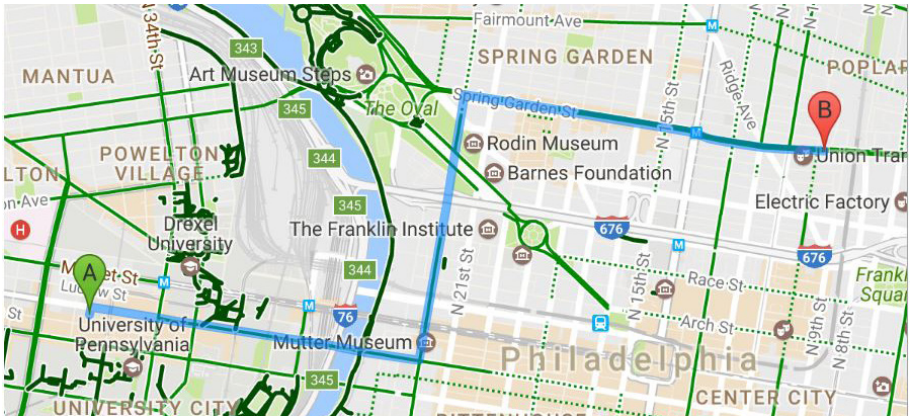


Figure 14. Targeted Route

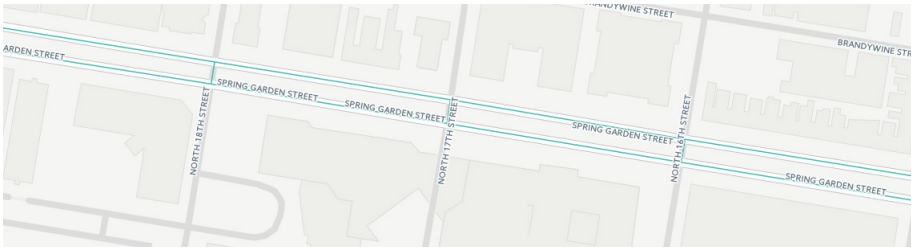


Figure 15. Query with distance = 20 meters

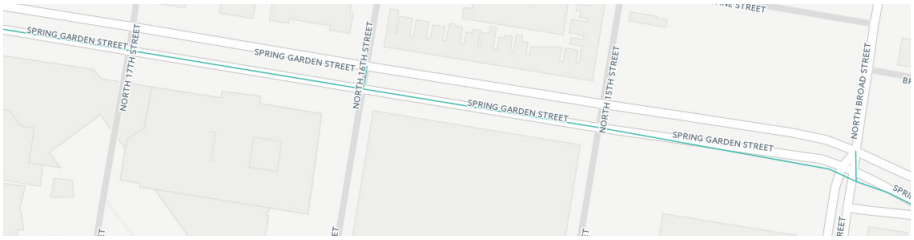


Figure 16. Query with distance = 5 meters

G.2 Application Live Testing

During the final presentation, I was inspired by the audience feedback that I should test my application in person. I agree that this is a feasible way to get first hand feedback; therefore, I mounted a GoPro, hopped on a bike and did it. I went from “3600 Chestnut St.” to “Philadelphia Municipal Services Building". As my application shows, the two recommended routes have score 51 and 59 respectively (figure 17). The blue is the default route, mainly uses Chestnut St, and the pink is the alternative route which runs through Chestnut and Race St. Since both route only start to deviate on the East side of the Schuylkill River, I focus on comparing the that part.

After the test, I felt significant difference in biking on these two routes. The Chestnut St. route is full of intersection controls and as a biker, I was stuck behind buses all the time. Even though it is a shorter

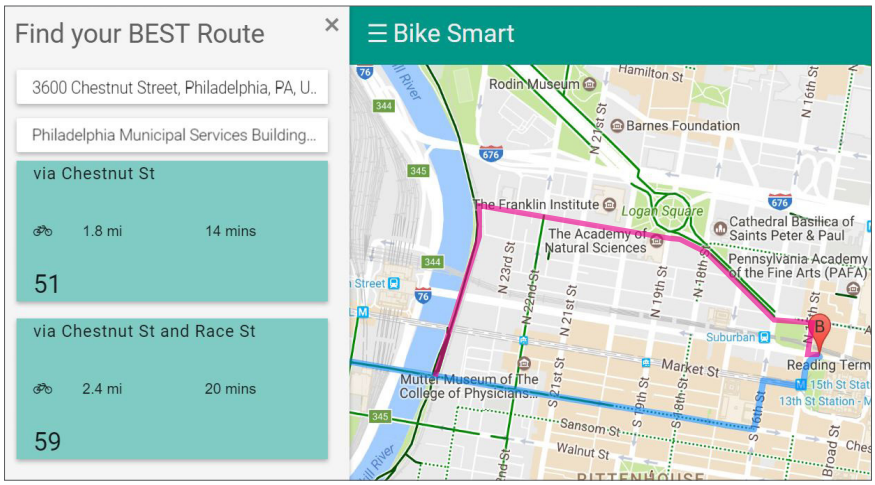


Figure 17. Test Ride

distance, it actually took more than 30 minutes get to destination due to the traffic. As figure 18 shows, a portion of Chestnut St. has separated bike lane. However, figure 19 shows a car is occupying the bike lane. I had to stop and walk around the car. Figure 20 shows how close a car can get to bikers, which is intimidating to stress-intolerant bikers like me. Figure 21 shows how a bike can get stuck by buses forever on a bus/bike shared lane. In addition to the traffic, the vehicle exhaust, noise and heat were also devastating to my biking experience on Chestnut Street.

Figure 18

Figure 19

Figure 20

Figure 21

In contrast, the other route provided me a much better experience. Biking on Schuylkill River Trail, there is much less barrier, so I was able to bike with a normal speed (figure 22). The view and sound were also exceptional. Even when I turned into Race Street, there was little traffic and very quiet (figure 23). Later when I get onto Benjamin Franklin Parkway, there was clearly indicated bike lane which

Figure 22

Figure 23

Figure 24

Figure 25

made me feel safe (figure 24). However, there was also presence of vehicle that occupies bike lane (figure 25). But overall, this route has much less barrier and the riding experience was much better. In terms of scores, an 8-score difference in this case means a heaven and a hell. Subjectively, I would rate Chestnut Street route 0 and Race Street Route 100. I wish my rating system can take the pavement, noise, traffic volume and view into account, in order to rate routes more accurately.

G.3 Future Possibilities

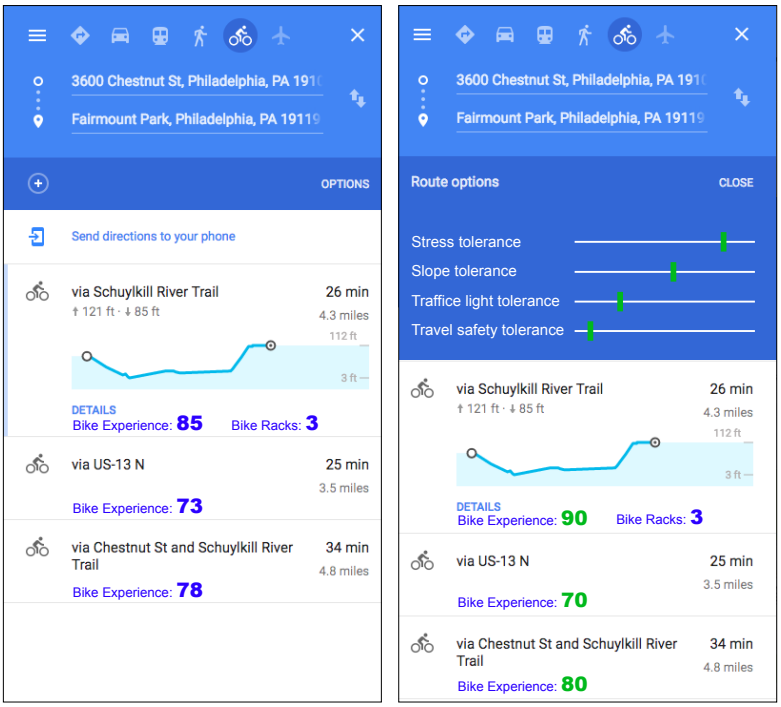


Figure 26. Improved Google Map Interface with Bike Score Comparison

For the future development of this app, I hope to first add some customization widgets, so that users can adjust the weight of each criterion and get the score suites best for their needs. Secondly, I want to add a user rating function, and be able to train the bike routes with more accurate scores. Lastly, I hope that I can pitch my scoring system to Google Maps, such that when you open the Google Map and use the bicycle function, you will be able to see the bike score I provided.

H. Conclusion

The bike score is a visualization of Philadelphia's bike infrastructure quality. It not only provides as a base of my application, but would also help planners prioritize infrastructure improvement tasks. On the other hand, as average people, we have no ability to improve the infrastructure, so the Bike Smart application would be a tool for bikers to find the best route based on current infrastructure.

I. Reference and Thank you Notes

1. Furth, Peter G., Maaza C. Mekuria, and Hilary Nixon. "Network Connectivity For Low-Stress Bicycling". Transportation Research Record: Journal of the Transportation Research Board 2587 (2016): 41-49. Web.

2. Mcatlas.org. N.p., 2017. Web. 2 Mar. 2017.

3. "Story Map Journal". Drx.maps.arcgis.com. N.p., 2017. Web. 2 Mar. 2017.

4. opendataphilly.org

5. Bliss, L. http://www.citylab.com/commute/2016/05/mapping-how-stressful-streets-can-be-for-cyclists/482469/

6. https://www.bicyclenetwork.com.au/general/for-government-and-business/2864/

7. http://www.arcgis.com/home/item.html?id=3fa41b1f8b764879be8f21b4e7ffbabd

8. https://gist.github.com/ismaels/6636986

9. Thank Susan Dannenberg, Gustave Scheerbaum and Gregory Krykewycz for their information.

10. Thank Nathan Zimmerman, Jeff Frankl, Theodore Lim and Simon Kassel for their technical support.