**Creating Cycle Routes on Strava Segments**

Wichpong Kao-ian1, Intiraporn Mulasastra2

Department of Computer Engineering

Faculty of Engineering, Kasetsart University

Bangkok, Thailand

1 k.wichpong@gmail.com, 2int@ku.ac.th

**Abstract**

**A vibrant cycling culture is growing in many countries, including Thailand, for both general transportation and recreation. However, few countries provide separate bicycle lanes for cyclists; hence, tools for suggesting cycle routes are essential for safety and a pleasant cycling experience. The aim of this study is to develop a web-based application for creating cycle routes according to user preferences: location and ride distance. Using the Strava segments in the area indicated by a user, we apply the Floyd-Warshall shortest path algorithm and a trade-off ratio to find a connected cycling route and show it on Google Maps, on an android device. The created route could be exported as a GPX file to a bike computer. Using geographic coordinates, our application connects distributed riding segments returned from calling Strava APIs, which Google navigation cannot do. In addition to improving the friendliness of user interfaces, future work should be done on improving the efficient calling of the Google application interfaces to better calculate the shortest distance among segments.**

Keywords: Floyd-Warshall algorithm; Bike computer; Android application; GPX file; Strava segments; Google navigator.

**I. Introduction**

A vibrant cycling culture is growing in many countries, including Thailand, for both general transportation and recreation. Cyclists receive both financial and health benefits when they bike. With today’s advanced technology, cyclists can use an application such as Strava as a source for cycle routes, and transfer a selected route to use as a navigator on a bike computer.

Strava is an application that can be used on mobile devices and via a website [1]. It is a social network connecting millions of athletes from around the world. The application can also be used to track other athletic activities, such as running, via GPS[[1]](#footnote-1). Strava provides users the features for recording cycle routes in its database and searching the database for available routes [2]. In addition, Strava can import and export GPX (GPS Exchange Format) files. The GPX file is an XML data format for the interchange of GPS data (waypoints, routes, and tracks) between applications and Web services on the Internet [3].

Strava datasets can be shared among its users, and users can vote on their most popular routes. Datasets gathered by Strava are also available to other services. However, the routes suggested in Strava are not always satisfactory to the cycling community due to being short or disconnected, especially in areas where there are only a few suggested routes.

This study presents the development of a web-based application for creating cycle routes based on two factors according to user preference: location and riding distance. Our application will select riding segments in the location chosen by the user from Strava; the segments returned from calling Strava APIs may be disconnected.

We apply the Floyd-Warshall shortest path algorithm and a trade-off ratio to find the most suitable cycle route. The results can be shown on Google Maps on any Android device, and can also be exported as a GPX file to a bike computer. Using geographic coordinates, our application can link disconnected riding segments, which navigation via Google Maps cannot do. The background knowledge, application development, and results are explained in the following sections.



Fig. 1 Strava route segments [1].

**II. Related work**

*A. GPS Exchange Format*

In interchanging GPS data (waypoints, tracks and routes) between programs, GPX (an XML schema designed as a common GPS data format) is used [5, 6]. GPX files contain a description of what's inside them, allowing anyone to create a program that can read the data within [5]. Sample descriptions of waypoints, tracks, and routes are shown in Figure 2, with explanations in Table 1.



Fig. 2 GPS Exchange Format [6].

TABLE I. GPS Data

|  |  |
| --- | --- |
| *GPS Data* | *Description* |
| Waypoint | An individual waypoint consists of the *GPS coordinates* of a point and other information (e.g., timestamp). |
| Track | An ordered list of points describing a path. Tracks are a record of where a person has been.  |
| Route | An ordered list of routepoints leading to a destination. A route is a suggestion about where to go in the future. |

*Source: Wikipedia[6]*

*B. Requesting GPS Data from a Server*

1) HTTP Request

An HTTP request message from a client to a server includes the method to be applied to the resource, the identifier of the resource, and the protocol version in use [7]. The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ends with Carriage Return and Line Feed (CRLF). The Method token indicates the method to be performed on the resource identified by the Request-URI, e.g., GET, POST, and CONNECT.

2) Java Script Object Notation (JSON)

JSON is text used for exchanging data between a browser and server. Any JavaScript object can be converted into JSON and sent to a server. Also, any JSON received from a server can be converted into JavaScript objects.

JavaScript has a built-in function, JSON.parse(), to convert a string, written in JSON format, into native JavaScript objects. We can convert a JavaScript object into a string with JSON.stringify() [8].

3) Encoded Polyline Algorithm

Google Maps applications define the notion of a polyline as a list of points, where line segments are drawn between consecutive points [9]. Polyline encoding is a compression algorithm for converting a series of coordinates (latitude and longitude points) into a single string for data interchange in text format [10].

*C. Floyd-Warshall Algorithm*

The *Floyd-Warshall* algorithm can be applied to find the length of the shortest path from point *i* to point *j* of a weighted graph with positive or negative edge weights (but with no negative cycles) [11,12]. We apply this algorithm to find the shortest paths between all pairs of nodes in our graph. Each node is a starting location of each segment returned from a Strava API call.

begin

integer i, j, k;

real infinity, s;

infinity = 1010;

for i = 1 to n do

 for j = 1 to n do

 if m[j, i] < infinity then

 for k = 1 to n do

 if m[i, k] < infinity then

 begin

 s = m[j, i] + m[i, k];

 if s < m[j, k] then m[j, k] = s

 end

end shortest path

Fig.3 Floyd-Warshall Algorithm [11]

**III. Our Proposed System for Creating Cycle Routes on Strava Segments**

We develop a web-based application for creating cycle routes, based on two criteria according to user preference: location and riding distance. Strava APIs are called to find the most popular riding segments, in the location chosen by the user, which may be unconnected. Our application will connect these segments and find the shortest path by applying the Floyd-Warshall algorithm as explained in Section II, and a trade-off ratio. A suitable cycling route suggested by our system in GPX format can be shown on Google Maps on any android device.

This section presents our overall system consisting of three major tasks: acquiring route segments from Strava datasets, connecting segments and finding the shortest path, and exporting a cycle route in GPX format. The following figure presents the overall system.

 

Fig. 2 Our overall system

*A. Acquiring Route Segments from Strava Datasets*

Two main classes that we use to acquire route segments from Strava are the GetLocation and JSONTask classes. Our system gets a location on Google Maps (for Android) from a user and then uses the GetLocation class to get the GPS location. We assign the specified location as a center for creating four grids on the map for finding Strava segments.

Then, the JSON Task.execute command is invoked to execute the segment Explore command, a Strava API. The command returns up to 10 popular segment objects, each of which consists of ID, name, average grade, the latitude and longitude coordinates of a start and end point of each segment.

The Explore command returns a maximum of 10 segments only, no matter how large the area specified. In this case, if a specified area is large, ten segments will be too few to find an optimum route. Hence, if a user-specified area is greater than four square kilometers, we will divide that specified area into four grids (Figure 4). Then, we call the Explore command four times (once per each grid) to get the total of 40 segments.



Fig.4 Four grids for finding Strava segments

Our program assumes that the diameter of a bicycle wheel is one meter. Hence, the distance traveled by a bicycle wheel in one complete turn is equal to 0.00314 kilometers. Pseudo codes of the procedure are shown in Figure 5.

* Repeat

Get the GPS location of a location specified by a user

If can\_get\_location Then

 If specified\_area > 4

 Calculate locations of 4 grids (user location is center)

 (top right and left, bottom right and left.)

 Else

 There will be only one grid

 For each grid

 Execute http requests to fetch JSON segments

 from Strava API

* Until can\_get\_location

Fig.5 Acquiring route segments

*B. Finding the Optimum Path*

We adapt the Floyd-Warshall Algorithm to find the optimum route from acquired Strava segments, as stated in the previous section. This method calculates the shortest paths among all nodes. Starting from a specified location (current location), an optimum route is calculated not only by considering the shortest distance between nodes, but the length of each segment.

To choose the next segment, the proportional ratio (length of a segment**/**shortest distance between a current location and that segment) is calculated. A segment with the maximum ratio will be selected as the next location of a route.

In our algorithm, the approximate distance between two segments is calculated from the geographical locations of their starting points, instead of calling the Google navigation API. The reason for this is that calling the API returns many alternative directions, resulting in a lot of requests required for finding the distances. Furthermore, Google limits the number of free API requests to only 1,000 per day, but the number of requests required by our system may be more than that. The pseudo codes of the algorithm is shown in Figure 6.

* Current node = a specified location
* Calculate shortest paths from a current location by

using Floyd-Warshall algorithm

* path\_list = empty
* While not out of nodes
* Find max ratio of paths from current node to other nodes
	+ Add the next node having max ratio to path\_list
* current node = next node with max ratio
* Return path\_list

Fig. 6 Finding the optimum path

*C. Showing the Resulted Route*

This method shows the resulted route on Google Maps. Starting at the user specified location, this method calls the method for finding the optimum path (Section B) to navigate through segments in the specified area. The Akexorcist library, the Google Direction API for Google Maps (Android), is used [13]. GetStartLocation() and GetEndLocation() methods are called to retrieve a position (latitude, longitude) on a plane of each segment. The detailed algorithm is shown in Figure 7.

* Create a header file (GPX file)
* Initialize dTest and sTest as a two dimensional array
* path\_list = finding the optimum path (dTest, sTest, user\_distance)
* total\_distance = 0
* If path\_list is not NULL then
	+ While total\_distance < user\_distance do
	+ Get distance between a pair of start and end points from Path\_list by Google navigation
	+ total\_distance += navigated distance+ segment length
	+ Write latitude and longitude to GPX file
	+ Show path on Google Maps

Fig.7 Showing the resulted route

**IV. Discussion and Conclusion**

This application makes use of available data on social media for the profit of the cyclist community. It can be easily used to find riding segments in a user specified area on Android mobile phones (Figure 8). In addition, if the segments found in a given area are not connected, our application can suggest connected routes by using Google navigation.

The suggested route can be displayed on Google Maps as well as exported for use on a bike computer. A starting segment is displayed with a thinner line, while the following segments are displayed by thicker lines in different colors.

  

Fig. 8 Our application on a mobile phone

We faced some problems in computing distance between segments due to the number of API required requests exceeding the daily limit allowed by Google. Hence, we solve this problem by computing distances from the geographical locations of the segments. However, when showing the optimum route, we use the Google Direction API to find the suitable connected paths.

Additionally, writing GPX files during route calculation was not successful. After finishing the calculation, we need to record route segments temporarily before writing all the data on a GPX file.

To use this application successfully, users need to be where there is a strong GPS signal. Further work should be conducted on improving the graphical interface and providing more options on selecting route segment preferences.

**Acknowledgments**

The authors would like to acknowledge all referenced researchers for their very helpful work. In addition, for his useful advice on exporting data to a bike computer, we would like to thank Dr. Chaiporn Jaikaew, Assistant Professor at the Department of Computer Engineering, Faculty of Engineering, Kasetsart University.

**References**

[1] Strava | Run and Cycling Tracking on the Social Network for Athletes", *Strava.com*, 2017. [Online]. Available: https://www.strava.com. [Accessed: 05- Sep- 2017].

[2] "Strava", *En.wikipedia.org*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Strava. [Accessed: 05- Sep- 2017].

[3] "GPX: the GPS Exchange Format", *Topografix.com*, 2017. [Online]. Available: http://www.topografix.com/gpx.asp. [Accessed: 05- Sep- 2017]..

[4] USAF, "Global Positioning System Standard Positioning Service Signal Specification", *US Government*, 1995.

[5] "The GPS Exchange Format", *Topografix.com*, 2017. [Online]. Available: http://www.topografix.com/gpx\_for\_users.asp. [Accessed: 05- Sep- 2017].

[6] "GPS Exchange Format", *Wikipedia.org*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/GPS\_Exchange\_Format. [Accessed: 17- Mar- 2017].

[7] "HTTP/1.1: Request", *W3.org*, 2017. [Online]. Available: https://www.w3.org/Protocols/rfc2616/rfc2616-sec5.html#sec5. [Accessed: 15- Mar- 2017].

[8] "JSON Introduction", *W3schools.com*, 2017. [Online]. Available: https://www.w3schools.com/js/js\_json\_intro.asp. [Accessed: 05- Sep- 2017].

[9] "Polyline", *Google APIs for Android*, 2017. [Online]. Available: https://developers.google.com/android/reference/com/google/android/gms/maps/model/Polyline. [Accessed: 14- Feb- 2017].

[10] "Encoded Polyline Algorithm Format", *Google Maps APIs*, 2017. [Online]. Available: https://developers.google.com/maps/documentation/utilities/ polylinealgorithm. [Accessed: 08- Mar- 2017].

[11] S. Warshall, "A Theorem on Boolean Matrices", *Journal of the ACM*, vol. 9, no. 1, pp. 11-12, 1962.

[12]"Floyd–Warshall algorithm", *Wikipedia.org*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Floyd%E2%80%93Warshall\_algorithm. [Accessed: 13- Apr- 2017].

[13] "Google Direction Library", *Akexorcist.com*, 2017. [Online]. Available: http://www.akexorcist.com/2015/12/google-direction-library-for-android-th.html. [Accessed: 12- May- 2017].

1. The Global Positioning System (GPS) is “a space-based radionavigation system which is managed for the Government of the United States by the U.S. Air Force (USAF), the system

operator” [4]. [↑](#footnote-ref-1)