— SCORE project —

GPXCleaner: GPS Path Editing and Simplification

Michal Young

Version 1.0, 2008.01.15

Abstract

There are many commercial and non-commercial systems for displaying GPS paths on a map and/or profile graph. They are typically used by runners and cyclists to record and analyze their excursions. An important and poorly supported function in these systems is reducing a GPS path (typically represented in the open GPX XML format) into a simpler path with fewer points, which is particularly important when producing a "mash-up" application with a service like Google Maps. The user should be able to vary the degree of simplification to balance accuracy with performance. Additional editing functions (clipping to a subset of the path, combining paths, inserting waypoint data) could also be supported.

1 Introduction

Global positioning system (GPS) units are increasingly popular for outdoor recreation, such as cycling, hiking, and running. A growing variety of services allow recreational enthusiasts to share records of their adventures with others. Many of these are "mash-ups" that use a separate mapping service (e.g., Google maps or Yahoo maps) to display a path over a map. Often the map image is combined in some way with a separate "profile" display, showing some attribute of interest (e.g., altitude or speed) as a function of time or distance.

GPS records as they are obtained directly from a GPS device is often poorly suited for direct use with one of the data sharing services. A record of a relatively lengthy excursion (a long bicycle ride or a hike) may contain thousands of points, while a much smaller number of points is sufficient for accurate rendering, and

rendering more than a few hundred points in a browser can be frustratingly slow. Also, data from the GPS devices may be "dirty" in several ways: It may include travel before and after the excursion of interest, the data may have gaps because the GPS device intermittently lost contact with satellites, and it may contain some inaccurate points from poor satellite communication. There is a need for software to help clean up these problems before publishing the data to a sharing service.

The key requirement of this project is to allow the user of a GPS device to reduce the number of points in a GPS trip record (henceforth called a "track") before submission to a data sharing service. Many other features are possible, but not strictly required. A successful project will provide a very good implementation of the main required function, plus some set of optional features, which may be selected from some suggestions here or devised by the development team. Developers must choose carefully among the many possible combinations of optional features so that the resulting product is useful, coherent, dependable, and pleasant to use.

2 Application domain and scenarios

2.1 Sharing recreational GPS data

The target user of this application uses a GPS device to capture records of recreational excursions. Although hikers, mountain bike riders, motorcyclists, runners, and many more outdoor enthusiasts use GPS devices for this purpose, to narrow our focus we will assume the user is a road bike cyclists. We will further assume that the excursion the rider wishes to share is a typical ride in a single day, i.e., between 5km and 300km traveled in a period of at least 30 minutes but less than 24 hours.

A record of an excursion is called a *track*. A track has a name and may be composed of one or more track *segment*s and a set of *waypoints*. A track segment is a sequence of *trackpoints*. A trackpoint consists of at least a latitude and longitude, and may in addition include other attributes, typically time (which in turn consists of a date and a time of day) and elevation.

A track may be recorded in one of several formats, the most common being the GPX interchange format (XML schema `http://www.topografix.com/GPX/1/1/gpx.xsd`; see `http://www.topografix.com/gpx.asp` for further information) and the Keyhole Markup Language (KML), of which the current dialect is XML schema `http://earth.google.com/kml/2.2`. For the required functionality of this project we will assume the GPX format, but handling KML input and output is an optional feature. A small excerpt of a typical GPX file is illustrated in Figure 1.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<gpx
  xmlns="http://www.topografix.com/GPX/1/1"
  ... >
<trk>
<name>ACTIVE LOG</name>
<trkseg>
<trkpt lat="44.02272038" lon="-123.1245574">
          <ele>196.726929</ele>
          <time>2007-12-08T15:07:07Z</time>
        </trkpt>
<trkpt lat="44.02273019" lon="-123.1245815">
           <ele>202.014282</ele>
           <time>2007-12-08T15:07:17Z</time>
        </trkpt>
        ...
<trkpt lat="44.02243431" lon="-123.1245071">
           <ele>283.726196</ele>
           <time>2007-12-08T23:44:14Z</time>
        </trkpt>
</trkseg>
<trkseg>
<trkpt lat="44.02243255" lon="-123.1245022">
           <ele>275.555054</ele>
           <time>2007-12-09T01:13:07Z</time>
        </trkpt>
</trkseg>
</trk>
</gpx>
```

Figure 1: Excerpts from a typical GPS track in the GPX XML format. "..." indicates where text has been elided. Elevation and time information are not always present. Many other attributes can legally be included in GPX; see `http://www.topografix.com/gpx/1/1/` for a complete definition.

2.2 Scenarios

The first scenario below describes the basic functionality required of GPXCleaner, although details of interfaces may vary (e.g., a graphic slider is not the only way one could control the number of points in the reduced track). Following the basic scenario, several variations below describe how optional features could be used. It is not expected that any single GPXCleaner project will implement all optional features described here. GPXCleaner teams are also encouraged to devise features that are not described here, as long as they fit coherently in a useful product.

2.2.1 Basic use scenario

Addie Marx is a recreational road cyclist who keeps an online journal of her rides. She mounts a Garmin Etrex GPS on the handlebars of her bicycle, clears all tracks and waypoints from its memory, and starts recording as she sets out on her Saturday morning ride. She rides for about 8.5 hours, about 140km (100 miles) and returns in late afternoon. Along the way she makes several stops for food, but leaves the GPS unit running, finally shutting it off only when she returns home.

After eating and showering, Addie connects the GPS unit to her computer with a USB cable and uses the freeware program LoadMyTracks[1] to download a record of her trip. This produces an GPX XML file called Track-2007-12-08.gpx.

Addie intends to publish a record of her ride on `EveryTrail.com`, a site for sharing records of excursions.

Addie opens Track-2007-12-08.gpx in GPXCleaner. The initial GPXCleaner display indicates that Track-2007-12-08.gpx contains one track called `Active Log`, containing two track segments. The first track segment contains 2435 track points, from 15:07:07 on 8 December until 23:44:14 on 8 December. The second segment has just one track point, recorded at 01:13:07 on 9 December. Addie changes the time display to her local time zone (Pacific Standard Time) and recognizes that the first track segment runs starts at 7:07am and ends at 3:44pm, while the point in the second track segment was recorded at 6:13pm local time, just before she downloaded the GPX file. Recognizing that the second segment was recorded when she turned on the unit to download data, she indicates that only the first segment should be included in the output.

As Addie moves a slider between “2435 points” and “10 points”, GPXCleaner displays an estimate of the accuracy that would be maintained if the track were reduced to different numbers of points. She finds that, with only 150 points, the reduced track would never deviate from the original track by more than 0.1km, which she judges a good balance of size and accuracy.

[1] `http://www.cluetrust.com/LoadMyTracks.html`

Addie also renames the track from "Active Log" to "December 8 Brownsville Loop", and then saves the reduced GPX file as "Track-2007-12-08.gpx". GPX-Cleaner renames the original, unreduced file to "Track-2007-12-08-bak.gpx" before replacing it with the reduced data.

Addie then uses the GPSVisualizer web service[2] to produce a time-distance profile of her trip, a time-elevation profile, and a Google maps display.

2.2.2 Variations: Usage with map display

Addie produces a GPX file as described above. While using GPXCleaner, she periodically clicks the "display" button to see how her reduced GPX file would look in Google Maps.

Variation — Other map services: Substitute any other interactive mapping service, such as Yahoo Maps, Google Earth, Mapquest, or Google maps.

Variation — Browser-based: GPXCleaner runs in a web browser, along with the browser-based web application.

Variation — Own display: GPXCleaner does not run in a web browser window, but maintains its own graphic display of the track, showing the original unreduced track and the reduced track together to graphically illustrate the degree of accuracy maintained in the reduced track.

Variation — Own map display: GPXCleaner maintains its own graphic display of the original and reduced track with a background map.

2.2.3 Variation: Cutting wild points

Note: This variation requires an interactive map display, which is also an optional feature.

Addie prepares a GPX file as described above, but when she displays the map, she finds that there is some apparently "wild" data near the Crawfordsville store (see Figure 2).

Addie locates the wild point and marks it for deletion.

Note: Deleting wild points should take place *before* reducing the number of points in a trace, because wild points tend to be preserved in trace reduction and may affect which other points are preserved.

[2] `www.gpsvisualizer.com`

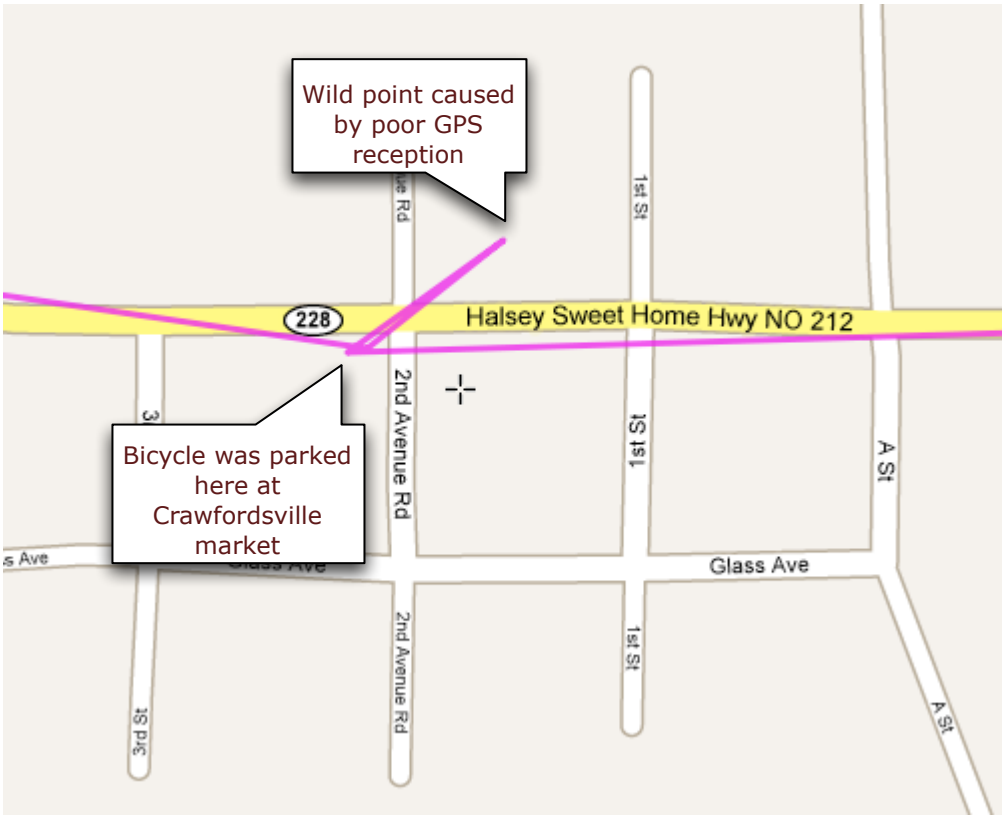

Figure 2: An example of inaccurate data in a GPS track. The accuracy of a GPS unit depends partly on how many satellites it is able to track, and "wild" data are typically inserted in a track when the GPS unit loses contact with some satellites, or while it is acquiring satellites near the beginning of recording. In this example, a bicycle with the GPS unit was beside a building (the Crawfordsville Market), blocking the signal from one or more of the satellites, and causing an inaccurate reading that seems to indicate a quick detour across Highway 212 toward 1st Street. Wild data may be recognizable from implausible speeds (e.g., a speed over 200kph recorded for a cyclist), but sometimes only a human user can recognize incorrect deviations from a path.

2.2.4 Variation: Segment joining

Addie produces a GPX file as described above, but she does not leave her GPS unit running during the whole time of her ride. When she stops to eat and rest, she temporarily turns the GPS unit off, then turns it back on again before continuing on her ride. Instead of seeing two segments in GPXCleaner, she sees six segments, of which five are really part of her ride and one is a junk segment accidentally produced when she turned on the GPS unit to download data. Addie indicates that the sixth segment should be discarded, and the first five should be joined into a single segment in the display.

2.2.5 Variation: Segment transposition

Addie turns the GPS unit off at her rest stops, as described above. Unfortunately, she forgets to turn it back on after her stop in Crawfordsville, noticing only when she reaches her next stop in Sweet Home. Fortunately, though, she has ridden this same route many times in the past, and has GPX tracks of those rides on record. She loads a second track, Track-2007-07-15.gpx into GPXCleaner, and indicates that she is looking for track points that connect the end of segment 2 with the beginning of segment 3. GPXCleaner finds the nearest track points and creates a new track segment 3 (making the old segment 3 into segment 4, the old 4 into 5, etc).

Since the July 15 ride was taken later in the day than the December 8 ride, the time associated with track points are not good estimates of the times Addie would actually have passed those points on her December 8 ride. Moreover, Addie was riding a good deal faster on the July ride than on her December ride, so a simple transposition of times by a fixed amount is not sufficient to adapt them. Splicing the segment into the existing track establishes the beginning and ending time of that segment, and GPXCleaner adjusts each time point between, interpolating between the beginning and ending time in proportion to the relative time of each point in the July 15 ride (making the new trackpoints somewhat farther apart in time).

2.2.6 Variation: Route Construction

Note: This variation requires an interactive map display, which is also an optional feature.

Over a period of months, Addie has taken many rides along the route of her December 8 ride, with several variations (including a detour to ride over Wendling covered bridge or skipping it; choosing different roads between Crawfordsville and Sweet Home; etc). She decides to compile these into a single map to share with other riders in the area. For this she loads several tracks (in their reduced or

original form) into GPXCleaner. Using the graphical display, Addie displays tracks from several rides and picks out parts to combine in a map of possible routes. The resulting collection of track segments includes options for paths of many different lengths passing through some of the same points, so it no longer makes sense to associate a single time with each point. Addie instructs GPXCleaner to simply omit time information from tracks.

3 Project goals

3.1 Required functionality

There are only two functional requirements.

[I/O formats] Accept and produce GPS tracks in GPX format

The product shall be capable of consuming and emitting data in the GPX interchange format for GPS records, in accordance with XML schema `http://www.topografix.com/GPX/1/1/gpx.xsd`. (See `http://www.topografix.com/gpx.asp` for further information).

[Simplify] Simplify GPS path

The product shall allow the user to specify the number of points to be retained from an input GPS track, and shall emit an output GPS track (in the GPX format) with no more than that number of track points. The output track shall follow the input track as closely as practical with the limited number of points.

Note: Although "as close as practical" is not a precise specification, we make the following observations about "good" and "bad" approximations: To minimize deviations from a path, more points are needed along portions of the path containing curves and turns than along portions that are straight. Many GPS units provide a built-in simplification algorithm that simply records a new point when a certain distance from the previously recorded point. This algorithm is simple to implement, but wastes points on straight sequences that could be better used near corners and on curves. Much better approximations can be obtained quickly enough for interactive use.

3.2 Optional functionality

Development teams are encouraged to implement additional, optional functionality based on the scenarios above or other reasonable scenarios of use. The features

chosen for implementation should form a coherent whole. *Choosing an attractive, coherent, useful set of features is as important as implementing those features well.*

4 Process and Deliverables

Development teams are not required to use any particular software process. They are strongly advised to *design* a process that is suitable for their team and for the project. Whether they choose a standard, well-known process or create their own process, a rationale for choosing that process is required.

Although a team may choose any development process and methodology, they must deliver a milestone release of a working prototype together with a plan for project completion. The prototype and plan must together make a convincing case that the project is likely to be successful if pursued to completion. It will be very hard to make a convincing case if the prototype does not demonstrate some reasonable portion of the planned product functionality.

4.1 Milestone Delivery 1

The first milestone delivery shall consist of at least the following:

Executive summary: The executive summary briefly summarizes the project plan, what has been accomplished so far, and what is yet to be accomplished. It summarizes the resource expended so far (including calendar time) and the resource yet to be extended. It includes a schedule with minor milestones leading to final delivery.

Software system architecture: The software system architecture document describes the overall organization of the product, and especially the principles for dividing it into modules. It may be expressed in text, in diagrams, or in any combination of text and diagrams. Standard notations must be identified, and non-standard notations must be explained. The software system architecture must include a rationale for key architectural design decisions. It need not be elaborate, so long as a reader can understand from it how and why the software has been organized as it has. It should be concise: A reader familiar with the notations used in the architecture description should be able to read and fully understand the architecture in an hour or less.

Prototype software product: The software product includes not only source and executable forms of the software, but also documentation for developers and for users, test plans and test data, and everything else required to continue

development up to and beyond delivery. A prototype of the product may include some portions of the final product and/or some artifacts constructed in the course of development but not intended to form part of the final product. There is no fixed rule for what should be included in the prototype, but there are two important requirements arising from the two purposes of the prototype. First, the prototype will be an important factor in deciding whether it is worthwhile to continue the project, so it should provide strong evidence one way or the other of the likelihood that continuing will produce a useful, dependable, and pleasant product within an acceptable expenditure of resource (especially calendar time). Second, the prototype should provide as much useful information as possible to the development team, regarding matters that the team has judged to be the highest risks.

Plan and status: The project plan document describes the overall plan for completing the software project, including what has already been accomplished as well as what is yet to be accomplished. It may be simple or elaborate, depending on the nature of the development process selected by the team, and it should include at least a brief description of that process. It should also make the case for (or against) continuation of the process, based on the prototype software product. It must therefore include at least a brief rationale for the choices made in the prototype, relative to an explicit assessment of risks.

These milestone deliverables are intended to be useful to the development team. A team should produce nothing that it is a waste of its time. If the development team decides that any of the deliverables above is a waste of time, they should not produce it and should instead produce a brief statement indicating how the purposes of that deliverable are met in another way.

4.2 Product delivery

Executive summary: The executive summary briefly summarizes the project plan, what was accomplished, and what was deferred or abandoned. It summarizes the resource expended so far (including calendar time) and the resource yet to be extended. It includes a brief summary of expected evolution of the product after delivery (e.g., how it might be adopted and expended by another group of developers).

Software system architecture: The software system architecture document is as described above for the milestone delivery, but reflecting any changes and

elaborations of the software organization since the time of the milestone delivery.

The software system architecture description can be judged by its suitabilty for orienting a new developer to the system. With a good architecture description (of a good architectural design), a skilled developer who is unfamiliar with the system should be able, in less than an hour, assess the feasibility and approximate effort required to add or change a system feature that was not anticipated by the original developers, and also determine which modules of the system would require changing or replacement.

Software product: The software product includes everything mentioned above for the prototype software product, substantially complete. Moreover, the software product must be appropriately packaged for both delivery to end users (e.g., through a web site) and adoption by future developers. For example, it could include a full source "tarball" on SourceForge.net, Collab.net, or another open source project service. A product that cannot be adopted and extended by another team of developers is not complete. (Note that this means the "product" as provided to other open source developers includes all the other artifacts, such as the executive summary and architecture documentation.)

Plan and status: The project plan document is retained as a record and a basis for planning future versions of the product. It should particularly include a record of mistakes that should not be repeated (e.g., feature X turned out to be much harder or less useful than anticipated, for reason Y).

5 Tools and standards

The key standard relevant to the GPXCleaner project is the GPX exchange format for GPS data. GPX is the *only* standard that is directly required for this project, but development teams are encouraged to consider also supporting the Keyhole Markup Language (KML).

Teams are encouraged (but not required) to use tools that are widely and cheaply available, and preferably tools that are available on multiple platforms. However, portability and tool standardization may be sacrificed for good reason, e.g., to accelerate development. Teams are also encouraged (but not required) to use standard interface libraries as appropriate. For example, using a standard library to interact with interactive web functions (so-called "AJAX") can make it easier for future developers to continue development or reuse modules in other projects.

6 Interaction with Stakeholders

The author of this project description is willing to serve as a representative of potential users of the application. Communications regarding the problem will be made public to all contestants, through the blog at `http://gpxcleaner.blogspot.com/`. Sample GPX data is linked from the blog. Contestants may make inquiries to the problem author at `michal.young@gmail.com` with "SCORE" in the subject line. Answers along with questions will be posted to the blog, so that all contestants have access to the same information. Contestants may also make comments directly in the blog. Development teams can expect a reply to a query within one week (usually sooner). On occasion, when the author is traveling, a blog post will make a note of his absence and expected return.

At their option, development teams might choose instead or in addition to find other representatives of the target user group. Interaction and requirements elicitation with other representatives of the target user group should be documented to explain how those interactions lead particular choices in the product design.

In addition to discussion of requirements, the problem author is willing to discuss technical issues, including path simplification algorithms, to a limited degree and also in the blog area.