Universitat de les Illes Balears

escola politècnica superior

Trabajo Fin de Grado

GRADO EN INGENIERÍA INFORMÁTICA

Generador pseudo-aleatorio de trayectorias

AHMED ANTONIO BOUTAOUR SANCHEZ

Tutor
Isaac Lera Castro

Escola Politècnica Superior
Universitat de les Illes Balears
Palma, 12 de abril de 2020

ÍNDICE GENERAL

ACRÓNIMOS

GPS Geospatial Position System

GIS Geospatial Information System

XML Extensive Markup Language

GML Geography Markup Language

KML Keyhole Markup Language

GPX GPS eXchange Format

OSM OpenStreetMap

HMM Hidden Markov Model

BSP Binary Space Partition

SQL-DB Structured Data Base

NO-SQL-DB Non-structured Data Base

CAPÍTULO 1

ABSTRACT

CAPÍTULO 2

SISTEMA DE POSICIONAMIENTO GLOBAL

El Sistemas de Posicionamiento Global, también llamado GPS por sus siglas en inglés es un sistema de navegación basado en satélites. Un mínimo de 24 satélites en órbita proporcionan información de posicionamiento y tiempo accesible para cualquier usuario. El funcionamiento de estos sistemas se basan el cálculo de la distancia de un punto de la tierra a tres satélites aplicando conocimientos de "Position resection". [1] Esta información tiene una precisión de 100 y 156 para la componente horizontal y vertical, respectivamente al 95porciento de probabilidad. [2] Las aplicaciones que permiten almacenar, manipular y presentar la información geográfica son llamadas Sistemas de Información Geográfica, Geospatial Information System (GIS) por sus siglas en inglés. [3]

Estas herramientas necesitan que los datos obtenidos por GPS sigan un formato compatible. Existen una gran diversidad de formatos que se dividen en ráster y Vectoriales. Mientras que los formatos ráster representan los datos en píxeles regulares de valor único (Esri Grid, GeoTIFF, JPEG2000), los formatos vectoriales se basan en la representación mediante puntos, líneas y polígonos. De este segundo grupo destacamos los siguientes:

Geography Markup Language (GML) Lenguaje procedente del esquema Extensive Markup Language (XML) que almacena información geográfica.[4]

Keyhole Markup Language (KML) Lenguaje basado en el esquema XML para la representar información geográfica en un navegador cartográfico como Google Earth o Google Maps. [5]

GPX Formato basado en el esquema XML para el intercambio de datos GPS. En especial para la descripción de points, tracks y routes. La principal ventaja de este formato es la capacidad de intercambio de datos entre diferentes programas en diferentes entornos (Windows, MacOS, Linux, Palm, PocketPC). Es este el formato seleccionado como entrada de datos GPS del desarrollo de este proyecto.

2.1 Estructura de los ficheros GPX

El formato GPX al ser basado en XML establece sus propias etiquetas y tipos con la información del fichero y del espacio geográfico que describe. La etiqueta principal gpx (gpxType) es la raíz del fichero XML. Presenta de forma obligatoria la versión y el creador del fichero, así como una cabecera de metadatos, dónde se describe la información del fichero, autor, así como restricciones de copyright de ser necesario. Los elementos esenciales para la descripción de la información geográfica son los siguientes:

***Waypoints* (wptType)** Representación de un punto de interés. Consta de una secuencia de elementos opcionales para añadir posición, descripción y precisión, así como los atributos obligatorios de latitud/longitud.

textitRoute (rteType) Lista ordenada de Waypoints. Representan una sucesión de puntos hacia un destino. Contiene una secuencia de elementos descriptivos de carácter opcional.

***Tracks* (trkType)** Lista ordenada de puntos que escriben un camino. Contiene el mismo tipo de secuencia de elementos descriptivos de carácter opcional.

La importación, tratamiento y edición de ficheros de formato GPX en el proyecto se realiza a partir de la librería Gpxpy. Esta librería aporta todas las herramientas necesarias para el parseo y manipulación de este formato en Python.

2.2 OSM

OSM es un mapa interactivo open-source que permite visualizar información geográfica. Las gran ventaja de OSM es la cantidad de herramientas aportadas por la comunidad que permiten importar, analizar, visualizar y editar la información. El formato de fichero GPX es el más utilizado dentro de esta herramienta.

La librería OSMnx de python aporta un nuevo método para adquirir,construir, analizar y visualizar las redes complejas de calles. Esta librería está presente en la continuación del documento. [6]

Esta sección se han introducido los conceptos básicos referentes al tipos de información, herramientas y formatos de datos geográficos con el que de trabaja en la continuación del documento.

CAPÍTULO 3

ARQUITECTURA DE LA APLICACIÓN TRACKSIMULATOR

En esta sección presentamos la aplicación TrackSimulator. Realizaremos una descripción de las funcionalidades que presenta y de los módulos de los que consta. Posteriormente realizaremos una descripción en detalle de cada uno de estos módulos, desgranando el funcionamiento completo de la aplicación.

3.1 Funcionalidades de *TrackSimulator*

TrackSimulator es una aplicación en Python para el análisis, simulación y exportación de trayectorias geoespaciales. La base de esta aplicación es la simulación de trayectorias geoposicionales dentro de un territorio delimitado, en función del análisis previo de trayectorias reales. El término *aplicación* se entiende como la suma de conjunto de implementaciones de código, ejecutable y del que se espera un resultado.

TrackSimulator permite:

Importación y análisis de archivos GPX La aplicación soportará la importación de un archivo GPX concreto, o bien la ruta de una carpeta con diversos archivos GPX. Estos archivos GPX deben contener trayectorias realizadas en el espacio delimitado. En nuestro caso las inmediaciones del Castillo de Bellver.

La aplicación, con los datos importados por los archivos, permitirá un análisis de trayectorias aplicando técnicas de *map-matching*. Del análisis se obtendrá información:

Distancia entre puntos Como la distancia entre las capturas de posición GPS en el fichero.

Distancia relativa entre punto de ruta y punto de trayectoria Como la distancia del punto GPS detectado de la trayectoria a la punto proyección dentro de la ruta.

Frecuencia de paso por segmento de ruta Como la frecuencia de paso por el segmento x_a, x_b relativo a todos los segmentos desde x_a.

Toda esta información quedará almacenada en una base de datos de forma que la información sea accesible para realizar la simulación de la ruta.

Muestra de resultados del análisis La aplicación permite la impresión por pantalla de información gráfica de los resultados del análisis.

Creación de una trayectoria a partir de parámetros Con los resultados de la información analizada se puede realizar una simulación de trayectorias dentro del espacio geográfico.

Exportación de trayectorias a fichero GPX La aplicación permite realizar una exportación de las trayectorias simuladas en formato GPX.

Visualización de la trayectoria La aplicación mostrará una representación gráfica de la trayectoria tanto analizada como simulada dentro del territorio geográfico.

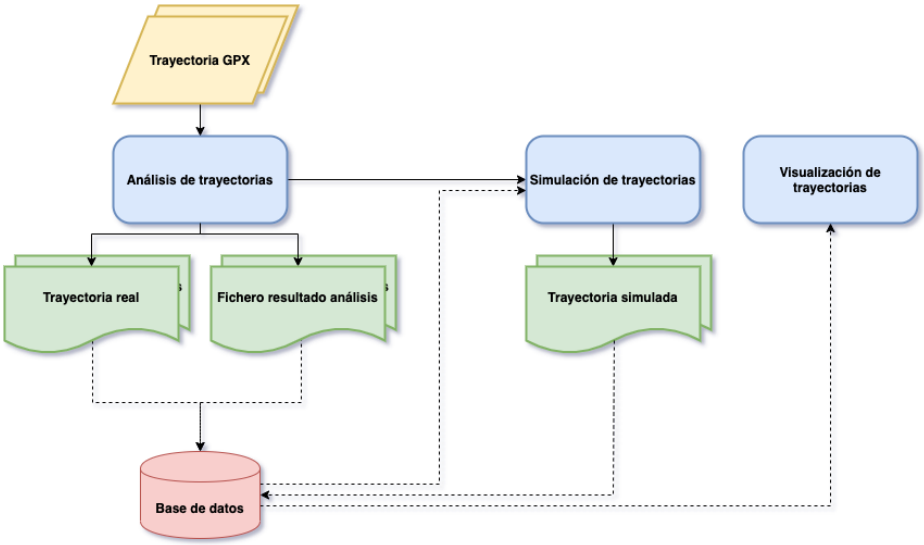

Figura 3.1: Diagrama de funcionamiento de TrackSimulator

3.2 Requerimientos de Track Analyzer

Explicadas las funcionalidades de la aplicación de Track Analyzer los requerimientos identificados son los siguientes:

Importación de datos La aplicación debe realizar una importación de los datos desde un fichero .GPX al modelo lógico elegido para la el posterior tratamiento.

Análisis de ruta La aplicación debe realizar un análisis de la ruta que proporcione por una parte indicadores y por otro valores medibles, cuantificables y utilizables para realizar una simulación lo más precisa posible.

Almacenamiento de las rutas reales en base de datos La aplicación debe realizar un almacenamiento de los datos analizados en una base de datos, con el objetivo de poder ser repetible sin necesidad de importar nuevamente los datos.

Almacenamiento del análisis en base de datos La aplicación debe realizar un almacenamiento de los resultados del análisis en una base de datos, con el objetivo de poder ser accesibles para la etapa de simulación.

Muestra de gráficas del análisis de los datos La aplicación debe poder realizar una muestra de la información resultante para su visualización por parte del usuario.

Lectura de la base de datos para el acceso a la información La aplicación debe poder realizar una lectura de la base de datos para acceder a la información necesaria para la muestra de resultados o simulación de trayectorias.

Obtención del camino más probable a partir de parámetros de entrada La aplicación debe poder generar el camino más probable dada un análisis previo y unos parámetros de entrada.

Simulación de puntos GPS a partir de resultado de análisis La aplicación debe generar los puntos de cada uno de los segmentos necesarios dado un camino.

Generación del fichero GPX correspondiente a la simulación de la trayectoria La aplicación realizar una exportación de los datos a formato .GPX.

Visualicación de trayectoria La aplicación debe poder mostrar al usuario una visualización de la trayectoria dado un fichero .GPX.

3.3 Librerías externas utilizadas

Para la realización de las funcionalidades comentadas en la parte anterior se ha hecho uso de las siguientes librerías externas:

gpxpy Librería para la manipulación de ficheros GPX. Mediante esta librería se puede realizar una importación del fichero para su posterior manipulación. De esta forma se obtiene la secuencia de puntos GPS que describe una trayectoria determinada.

pandas Librería para el análisis de los datos. Toda la información correspondiente a las diferentes trayectorias queda reflejada en un Dataframe para su posterior tratamiento y acceso.

osmnx Librería para la extracción, visualización y análisis de redes de calles. Mediante esta librería se puede obtener toda la información referente a nuestro espacio determinado del Castillo de Bellver e importar toda la información de sus caminos y carreteras transitables. De esta forma tenemos toda una estructura de datos

con información geográfica precisa del entorno. Por otra parte mediante esta librería se puede visualizar las trayectorias escogidas, así como la capacidad de almacenar imágenes de las trayectorias analizadas o simuladas.

matplotlib Librería por excelencia para la representación de información de forma visual. Mediante el uso de esta librería podemos mostrar diversas gráficas de la información obtenida y analizada por los diferentes algoritmos.

numpy Librería para el cálculo científico. Esta librería nos aporta diferentes estructuras de datos para el acceso y cálculo de forma eficiente a la información. Una de las principales ventajas de esta librería es la implementación de matrices de forma que el acceso a los datos incrementa su eficiencia de forma considerable.

geopy Librería para el tratamiento de coordenadas. Mediante esta librería se obtienen las herramientas necesarias para tratar al par de datos *(Lat,Long)* como un punto de coordenada GPS. Por otra parte se obtiene implementación del cálculo de la distancia entre dos coordenadas.

itertools Esta librería implementa diversos bloques de iteradores. El uso dentro de nuestro proyecto queda exclusivamente para la agrupación de elementos dentro de un set de datos.

sklearn Librería para el análisis de de datos. Esta librería nos aporta todas las herramientas necesarias para el análisis de los datos GPS. Con ella realizamos los cálculos de puntos próximos a partir de una búsqueda basada en Binary Space Partition (BSP) como veremos en la siguiente sección.

shapely Librería para el tratamiento de figuras geométricas. De esta forma podemos tratar elementos comos puntos GPS de forma abstracta. Por otra parte nos permite tratar las rutas del espacio a analizar como una linea de sucesivos puntos GPS.

pickle Librería para la codificación-decodificación de ficheros. Mediante el uso de esta librería se realiza la exportación de la información estadística y de la estructura de grafo generada y almacenada en los ficheros .txt/edgelist sucesivamente.

CAPÍTULO 4

PREPARACIÓN DEL ENTORNO DE DATOS

4.1 Importación de datos GPS a *TrackSimulator*

Para realizar un tratamiento de los datos a partir de un fichero GPX se realiza un uso de la librería *gpxpy*. El primer paso a realizar es el parseo del fichero. De esta forma encontramos un objeto track, que contiene segmentos que a su vez contienen los diferentes waypoints. De los que hemos hablado anteriormente. Los waypoints tienen la estructura que podemos ver en esta imagen:

```
<trk>
  <name>Trote </name>
  <type>trail_running</type>
  <trkseg>
    <trkpt lat="39.56738345324993133544921875" lon="2.623643483966588973999023437 5">
      <ele>30.200000762939453125</ele>
      <time>2019-01-06T11:12:47.000Z</time>
      <extensions>
        <ns3:TrackPointExtension>
          <ns3:atemp>26.0</ns3:atemp>
          <ns3:hr>85</ns3:hr>
          <ns3:cad>48</ns3:cad>
        </ns3:TrackPointExtension>
      </extensions>
    </trkpt>
```

Figura 4.1: Estructura de un *waypoint* dentro de un fichero .GPX

La información necesaria para la implentación de la solución propuesta en este documento es la posición en latitud y longitud de cada uno de los puntos. El resto de información no será usada para el desarrollo de la propuesta, queda una estructura en forma de lista del siguiente tipo de elemento.

Punto preprocesado		
Campo	**Tipo**	**Descripción**
longitud	Float	Posición del nodo respecto al eje horizontal.
latitud	Float	Posición del nodo respecto al eje vertical.
Point(longitud, latitud)	Point	Estructura de punto para tratamiento interno.

Cuadro 4.1: Estructura lista de puntos.

4.2 Importación de la red compleja de datos

Como apunta G. Boeing en su artículo [6] , para abordar correctamente la problemática del análisis de rutas dentro del espacio urbano se debe plantear con una solución con una red compleja. Esta red está modelada en forma de grafo. Con esta estructura de datos se puede contemplar un proceso de importación, análisis, y exportación de datos geográficos. Para realizar la importación de los datos se recurre a la librería OSMnx comentada en la sección anterior. Mediante el uso de un corto número de instrucciones se consigue importar toda la estructura de la zona del Castell de Bellver.

Con la importación de los datos se ha conseguido obtener la información de OpenStreetMap, no obstante se debe realizar un tratamiento de filtrado de los datos importados debido a que aparece información que en este proyecto no tienen relevancia. Esta información se trata de características concretas de calles, peatonales o no, como la dirección. Nuestro planteamiento del problema define que el la trayectoria a simular no tiene en cuenta el sentido de la carretera. Por otra parte, añadiremos información tanto a los nodos como a las aristas del grafo para almacenar información necesaria para la solución del problema. Por lo tanto al final del tratamiento de la red obtendremos un grafo dirigido, donde las intersecciones entre la infraestructura urbana (calle, camino, carretera) está definido como un nodo y la característica de esta como la arista. Por simplicidad de entendimiento para la implementación de soluciones se ha decididaplicar dos aristas por cada nodo en común, con la información geoespacial correspondiente. La red compleja queda de la siguiente forma una vez aplicado el procesado:

Nodo		
Campo	**Tipo**	**Descripción**
Identificador	Integer	Clave del nodo.
x	Float	Posición del nodo respecto al eje horizontal.
y	Float	Posición del nodo respecto al eje vertical.
osmid	Integer	Identificador de OSM

Cuadro 4.2: Estructura nodo.

4.3 Almacenamiento del dato

Tanto de trayectorias reales como de los datos generados por el análisis deben ser almacenados de forma que sean accessibles de forma rápida. Para el almacenamiento

Arista		
Campo	**Tipo**	**Descripción**
Identificador	(Integer, Integer, Integer)	Clave de la arista
osmid	Integer	Identificador de OSM
oneway	Boolean	Carretera de un único sentido.
name	String	Nombre de la carretera.
highway	String	Tipo de carretera.
length	Float	Longitud del camino.
geometry	LineString	Geometría de la carretera.
num of detections	Integer	Núm. de detecciones de puntos.
frequency	Float	Frecuencia de puntos detectados.

Cuadro 4.3: Estructura arista.

de datos, existen dos grandes posibilidades: El uso de Bases de datos relacionales, a partir de ahora descritas como Structured Data Base (SQL-DB), o el opuesto, las bases de datos no relacionales, Non-structured Data Base (NO-SQL-DB). Las ventajas de las NO-SQL-DB han hecho que con el paso del tiempo, los desarrollos se vean orientados al tipo de almacenamiento no relacional. Para la realización de la propuesta descrita en este documento se ha decidido realizar el almacenamiento de toda la información en *MongoDB*, una de las NO-SQL-DB más usadas y que detallamos a continuación.

4.3.1 MongoDB

MongoDB se trata de una NO-SQL-DB de código abierto y su funcionamiento es documental. Se almacenan colecciones de documentos, que son series de elementos JSON clave-valor.

MongoDB elimina las limitaciones de las bases de datos relacionales [7]. Permite almacenar de forma eficiente grandes cantidades de información, siendo a su vez flexible a modificaciones debido a que los documentos que se almacenan en las colecciones no tienen una taxonomía definida. Por lo que el desarrollo incremental de la aplicación y la aparición de nuevos campos dentro del modelo de datos no es un problema. Otro gran motivo para la elección de MongoDB como base de datos de este proyecto es la escalabilidad que ofrece, de forma que a grandes cantidades de trayectorias por almacenar, la aplicación respondería de forma eficiente y mucho más precisa.

Actualmente para el desarrollo de la propuesta no se ha contado con una gran cantidad de datos. No obstante el problema ha sido planteado con el objetivo de poder analizar grandes cantidades de datos y que puedan ser almacenados y accesible mediante el uso de esta base de datos.

CAPÍTULO 5

ANÁLISIS DE LOS DATOS GPS

5.1 Map-Matching

Uno de los grandes retos del análisis de trayectorias es la identificación y asignación de cada uno de los puntos GPS en su correspondiente trayectoria. Los puntos GPS se transmiten a partir de un dispositivo. Este dispositivo administra la serie de puntos GPS correspondientes a una trayectoria con un cierto error y desviación. Esta serie de puntos son puntos aislados y no tienen relación directa con el modelo lógico establecido.

Definiremos como ***Map-matching*** al proceso que nos permite asignar las coordenadas GPS a un modelo lógico establecido.

Existen diversos procedimientos para abordar el problema del *Map-matching*. Nuestra aproximación al problema la realizamos aplicando el modelo probabilístico de las cadenas de Markov, que explicamos en el siguiente apartado.

5.1.1 HMM

HMM sigue el modelo estadístico tradicional de Markov. [8]. En 1907, A. A. Markov comenzó el estudio de un proceso donde un experimento podía afectar a la salida del siguiente experimento. Este tipo de proceso se denominó Cadena de Markov.

Definiremos un conjunto de estados $S = s_1, s_2, ..., s_n$. El proceso empieza en un estado y se traslada de estado en estado. Esta transición se le denomina *step*. Los *steps* entre un estado S_i y S_j cumple una **probabilidad de transición**.

Por otro lado encontramos la **probabilidad de emisión**. Esta probabilidad responde a observar o en un estado S en un instante t determinado.

Markov define como suposición de primer orden que la probabilidad de ocurrencia de un evento en un tiempo t solo dependerá de $t-1$. De esta forma las observaciones $O-2, ..., O-n$ no afectaran directamente a t.

5.1.2 Aproximación HMM a Map-matching

Nuestra estructura del territorio está definida como un grafo formado por diferentes *nodos* y *aristas*. Los nodos corresponden a la intersección entre caminos, mientras que en las aristas se almacena la información referente a la constitución geográfica de estos.

Esta información está almacenada en un *Shape* de forma que se trata de una sucesión de puntos GPS tal y como se muestra en la siguiente imagen:

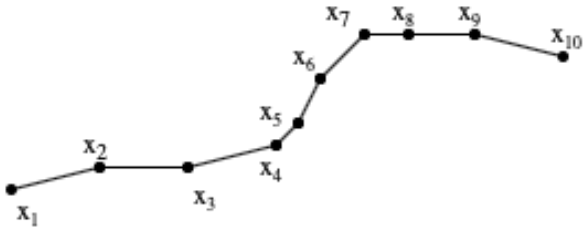

Figura 5.1: Ejemplo de estructura LineString

Cada uno de los posibles puntos de los subsegmentos identificados entre $x_1, ..., x_{10}$ son proyecciones validas a una observación determinada.

Por otra parte tenemos la información GPX. El fichero que almacena la información de trayectorias que se han realizado en las delimitaciones de nuestro territorio. Esta serie de puntos se identifican como **observaciones**.

Tenemos entonces los dos elementos principales: Una serie de observaciones y un modelo. Se plantea el **problema general de decodificación**: Dada una secuencia de observaciones, ¿qué secuencia de estados es la más probable para generarlos?

Este problema se resuelve mediante la implementación del ***Algoritmo de Viterbi***, que explicamos a continuación.

5.1.3 Algoritmo de Viterbi

El algoritmo de Viterbi se presenta como una técnica computacional con la que se obtiene la sucesión de nodos más probable para una cadena de Markov.

Para cada una de las proyecciones se obtiene una probabilidad, siendo la más alta la elegida para ser el nuevo elemento del camino de Viterbi.

La probabilidad tiene en cuenta dos factores:

Probabilidad de emisión Esta probabilidad es calculada por análisis espacial. Por convenio establecemos que cuanto más próxima sea la observación a la proyección más probable será que sea la elección correcta, partiendo de una distribución normal con media cero tal y como se muestra en la referencia [9].

$$P_{emission} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(d_i^j - \mu)^2}{2\sigma^2}} \tag{5.1}$$

Probabilidad de transición La probabilidad de llegar al estado S_i de una proyección viene determinado por el estado anterior S_{i-1} en una relación de dos aspectos.

Por una parte la distancia espacial entre el S_i y S_{i-1} identificado como $d(S_i, S_{i-1})$. Por otra parte encontramos el camino más corto entre estos dos estados. El camino más corto definido como $SP(S_i, S_{i-1})$ se entiende como la cantidad de nodos que se tienen que atravesar para llegar de S_i a S_{i-1}. Esta última se mantiene como una distribución exponencial de forma que a mayor número de nodos por atravesar en el camino más corto esta proyección es penalizada.

$$P_{transition} = \frac{distance(p_{i-1}, p_i)}{e^{SP(p_{i-1}, p_i)}} \tag{5.2}$$

La implementación del algoritmo se realiza de forma iterativa para todos los puntos GPS de la trayectoria. Por cada punto se buscarán las proyecciones en un radio determinado. De esta forma se realiza una primera criba de proyecciones que no serán probables. Por cada una de las proyecciones se calculará las probabilidades descritas anteriormente. La probabilidad de transición viene determinada por el estado anterior, tal y como se puede observar en la fórmula. Una vez tratadas todas las proyecciones. Se almacena aquella proyección con la mayor probabilidad total. Si el camino más corto entre el nodo correspondiente a la proyección p_i y p_{i-1} es mayor que uno se debe completar el camino más corto. En la propuesta se tiene en cuenta el posible desfase temporal de los diferentes dispositivos localizadores de GPS. Si el camino más corto tiene 8 o más nodos se descartará la trayectoria, al no ser valorable para el análisis.

Con la aplicación de esta información se obtiene una secuencia de puntos GPS relacionados con la estructura de datos y queda solucionado el problema del *map-matching*.

5.1.4 Ejemplo de aplicación del algoritmo de Viterbi

Con el siguiente ejemplo se pueden observar los beneficios y los inconvenientes de la aplicación del algoritmo implementado en esta propuesta.

En la figura 5.2 se observa en puntos rojos la trayectoria real de un individuo dentro del territorio limitado del Castell de Bellver. Los puntos verdes corresponden a la detección de las proyecciones más probables por parte del algoritmo de Map-matching.

El algoritmo realiza una detección punto a punto de su proyección dentro del camino. Pueden aparecer situaciones en las que por desviación de precisión en la detección GPS un punto parezca situarse cerca de un camino no correcto. No obstante, la probabilidad de transición descrita en el apartado anterior se encarga de ofrecer como opción más probable, aquella donde el *shortest path* del grafo sea menor. 5.3

No obstante pueden aparecer casos aislados donde la deteccion GPS es tan cercana a una proyección de un camino incorrecto que hace imposible su no elección respecto a la proyección correcta. 5.4

Figura 5.2: Ejemplo de aplicación de algortimo de Viterbi para detección de trayectorias.

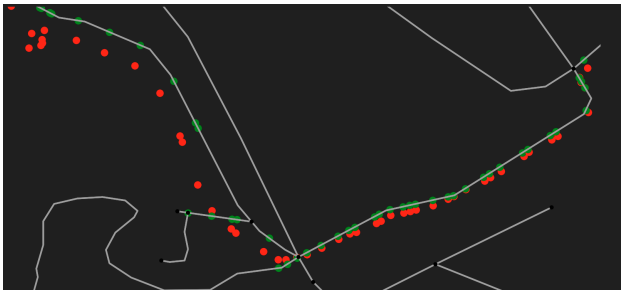

Figura 5.3: Ejemplo de aplicación de algortimo de Viterbi para detección de trayectorias.

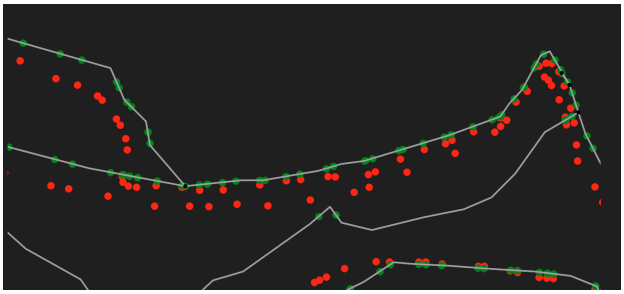

Figura 5.4: Ejemplo de aplicación de algortimo de Viterbi para detección de trayectorias.

5.2 Explotación del dato

Con la problemática descrita en los apartados anteriores resuelta. Tenemos la capacidad de poder inferir toda la información planteada para la propuesta de esta solución. Por una parte encontramos la relación $P_{real} - P_{proyectado}$ y $P_n - P_{n+1}$, podemos generar la distancia entre estos puntos para toda la trayectoria. La distancia que calculamos es la distancia más corta sobre la superficie terrestre, la formula elegida es la formula Haversine [10] [11]. Esta fórmula tiene en cuenta el radio de la esfera terrestre para realizar el cálculo de la siguiente forma:

$$a = \pi/2 - lat_1 \tag{5.3}$$

$$b = \cos(lat_1) * \cos(lat_2) * \cos(lon_2 - lon_1) \tag{5.4}$$

$$c = \arccos(a + b) \tag{5.5}$$

$$d = R * c \tag{5.6}$$

Con el cálculo de la distancia encontramos tanto la desviación entre la trayectoria real y la ruta real como la distancia entre las muestras de puntos GPS de la trayectoria. La detección de la proyección nos permite saber que caminos se han atravesado para generar la trayectoria. Con esta información podemos generar la frecuencia de paso por cada uno de los caminos. La frecuencia de paso será relativa a la frecuencia de paso de todos los caminos con el mismo nodo origen. De esta forma si hay 4 caminos $c_{1,...,4}$ para un nodo n_x la probabilidad del punto será la siguiente:

$$p_{c_i} = \frac{d_{c_i}}{\sum d_{n_x}} \tag{5.7}$$

siendo d el número de detecciones en en la arista.

Esta información será analizada y tratada de forma que el proceso de simulación genere una trayectoria basada en información real.

CAPÍTULO 6

SIMULACIÓN DE TRAYECTORIAS

6.1 Generación del camino más probable

En este apartado se pretende describir como se selecciona el camino más probable.

6.2 Generación de los segmentos

En este apartado se pretende describir como se realiza la simulación de los segmentos del camino elegido.

6.3 Generación del punto pseudo-aleatorio

En este apartado se pretende describir como se realiza la generación del punto pseudo-aleatorio a partir de la información analizada.

CAPÍTULO 7

EXPORTACIÓN DE TRAYECTORIA

7.1 Exportación de trayectoria a fichero .GPX

En esta sección se pretende describir la característica del aplicativo para realizar la exportación a GPX.

CAPÍTULO 8

EXPERIMENTACION

8.1 Experimentación sin importación de datos reales

En este apartatado se pretende realizar la muestra de resultados de la generación de 10 rutas creadas mediante la simulación sin datos amacenados.

8.2 Experimentación con importación de datos reales

En este apartado se pretende realizar la muestra de resultados de la generación de 10 rutas creadas mediante la simulación a partir de los datos almacenados.

CAPÍTULO 9

CONCLUSIONES

En esta sección se realizará una explicará la opinión sobre el desarrollo de la propuesta.

9.1 Posibles mejoras

En esta sección se detallaran las posibles mejoras que se podrían implementar en la propuesta.

BIBLIOGRAFÍA

[1] R. B. Langley, "The mathematics of gps," vol. 2, pp. 45 – 50, July/August 1991. 2

[2] A. El-Rabbany, *Introduction to GPS: The Global Positioning System*. Artech House Publishers, 2002. 2

[3] U. S. E. P. Agency, *Geographic information Systems Guidelines document*, 1988. 2

[4] O. G. Consortium, "Ogc® geography markup language (gml) — extended schemas and encoding rules," 2012. [Online]. Available: http://www.opengis.net/spec/GML/3.3 2

[5] ——, "Kml 2.1 reference – an ogc best practice," *Carl Reed on behalf of Google Earth staff*, 2007. 2

[6] G. Boeing, "Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks," *Computers Environment and Urban Systems*, pp. 126 –139, 2017. 2.2, 4.2

[7] "Advantatges of nosql." [Online]. Available: https://www.mongodb.com/scale/advantages-of-nosql 4.3.1

[8] H. S. Malvar, "Effective communication: Tips on technical writing," *IEEE Signal Processing Magazine*, vol. 25, no. 3, pp. 129–132, 2008. 5.1.1

[9] W.-J. H. Yu-Ling H., Ho-Chian C., "A hidden markov model-based map-matching approach for low-sampling-rate gps trajectories," *IEEE*, pp. 120 –142, November 2017. 5.1.3

[10] M. T. Ltd, "Gis faq q5.1: Great circle distance between 2 points," 2012. [Online]. Available: https://www.movable-type.co.uk/scripts/gis-faq-5.1.html 5.2

[11] R. Sinnott, "Virtues of the haversine," *Sky and Telescope*, p. p. 159, November 1984. 5.2