

x5ahkotq3

January 26, 2025

Dr. Jesús Martí Gavilá. Departamento de Ingeniería Cartográfica, Geodesia y fotogrametría

PRÁCTICA 7

CURSO BÁSICO PYTHON: FICHERO GPX

Grado Tecnologías Interactivas | Tecnologías de la Información Geográfica

1 TAREA

Realizar un análisis de rendimiento a partir de un fichero GNSS con waypoints en formato GPX.

El resultado a obtener sera:

Un Dataframe en formato HTML con los siguientes campos:

‘Punto’, ‘Latitud’, ‘Longitud’, ‘Cota (m)’, ‘Hora’, ‘Distancia (m)’, ‘Dist_O (m)’, ‘Pendiente (%)’, ‘Velocidad (km/h)’, ‘Orientación (°)’, ‘Tiempo’

Gráfico con los valores de distancia en el eje X y valores de pendiente y velocidad en la Y

Mapa interactivo

2 Estructura GPX e importación de librerías

```
[ ]: """
Creado Viernes 14 Oct 18:22:39 2022, @autor/es: .....
↪.....
FORMATO EJEMPLO GPX
<wpt lat="38.968484" lon="-0.148688">
    <ele>-6.847729</ele>
    <time>2020-10-06T15:42:19Z</time>
    <name>001</name>
    <sym>Flag, Blue</sym>
</wpt>
"""

import pandas as pd
import xml.etree.cElementTree as et
import os
import math as m
from datetime import datetime
```

```
import matplotlib.pyplot as plt
```

3 Fichero de entrada y Dataframe de salida

```
[ ]: # Introducción de datos. Solicitar al usuario el nombre del fichero sin
      ↪ extensión.
```

```
.....
.....
```

```
# Este nombre también se utilizará para crear el fichero html
```

```
.....
.....
```

```
[ ]: # Accedemos al XML y creamos un Dataframe con los campos arriba indicados
```

```
docxml = et.parse(pathgpx)
```

```
raiz = docxml.getroot()
```

```
puntos = len(raiz)-1
```

```
columnas = ['Punto', 'Latitud', 'Longitud', 'Cota (m)', 'Hora', 'Distancia_
```

```
↪(m)', 'Dist_0 (m)', 'Pendiente (%)', 'Velocidad (km/h)', 'Orientación_
```

```
↪(°)', 'Tiempo']
```

```
df = pd.DataFrame(columns=columnas)
```

4 Funciones de cálculo

```
[ ]: # Cálculo de la distancia parcial . función distacias
```

```
def distancias .....
```

```
[ ]: # Cálculo del acimut. Verificar la posición del punto B. funcion orientación
```

```
def orientacion .....
```

```
[ ]: # Cálculo de la pendiente. En % y angulo. funcion pendiente
```

```
def pendiente .....
```

```
[ ]: # Cálculo de la velocidad en km/h. funcion velocidad
```

```
def velocidad .....
```

5 Obtención de los datos de entrada y llamada a funciones

```
[ ]: waypoints = range (1,puntos,1)
```

```
n = 0.00
```

```
for i in waypoints:
```

```
    # Datos del punto A
```

```
    wpt1 = raiz[i]
```

```
    lat1_g = float(wpt1.get("lat"))
```

```
    lon1_g = float(wpt1.get("lon"))
```

```

lat1 = m.radians (lat1_g)
lon1 = m.radians (lon1_g)
ele1 = round(float(wpt1[0].text),3)
time1 = wpt1[1].text
if i == 1:
    df_1 = pd.DataFrame([[i,lat1_g,lon1_g,ele1,time1,n,n,n,n,n,n]],columns=
↳ list(columnas))
    df = pd.concat([df if not df.empty else None,df_1],ignore_index=True)
↳ #controla si un dataframe esta vacio
# Incrementamos puntos
i = i+1

# Datos del punto B
wpt2 = raiz[i]
lat2_g = float(wpt2.get("lat"))
lon2_g = float(wpt2.get("lon"))
lat2 = m.radians(lat2_g)
lon2 = m.radians(lon2_g)
ele2 = round(float(wpt2[0].text),3)
time2 = wpt2[1].text

# Llamar a las diferentes funciones y añadir los datos resultantes al Dataframe
d, dr = distancias(.....) #Introducir los parámetros
↳ necesarios para realizar el cálculo
acimut = orientacion(.....) #Introducir los
↳ parámetros necesarios para realizar el cálculo
peng, pend, dg = pendiente(.....) #Introducir los
↳ parámetros necesarios para realizar el cálculo
tiempo, velkmh = velocidad(.....)#Introducir los
↳ parámetros necesarios para realizar el cálculo
df_1 = pd.DataFrame([[i,lat2_g,lon2_g,ele2,time2,dg,0.
↳ 00,pend,velkmh,acimut,tiempo]],columns = list(columnas))
df = pd.concat([df,df_1],ignore_index=True )

df['Dist_0 (m)'] = ..... # crear un comando
↳ para rellenar df['Dist_0 (m)']

```

6 Impresión y exportación de los obtenidos

```

[ ]: # Imprimir los valores en formato HTML
.....

```

```

[ ]: # Plotear el gráfico de rendimiento y guardar en JPG. Puedes ver ayuda en
↳ internet matplotlib.plot
# Empieza para plotear 2 ejes

```

```
fig, ax = plt.subplots()
plt.plot(df["Dist_0 (m)"], df['Pendiente (%)'], marker = "o", label="Pendiente_
↳%")

.....
↳.....
# indicar leyendas, titulo, etiquetas del eje X
.....
↳.....
# Exportar en JPG
.....
↳.....
```

7 Mapa interactivo con Folium. Código ejemplo

```
[ ]: #Instalar primeramente la libreria folium
.....
↳.....

#Código creación fichero de coordenadas
df.loc[:,['Latitud','Longitud']].to_csv('coordenadas.csv',header=True,
↳index=False) #obtaines los valores de latitud y longitud y creas un fichero_
↳de coordenadas
lista_geo = pd.read_csv('coordenadas.csv').to_numpy().tolist()

#Creación del mapa interactivo
import folium
from folium import plugins
m = folium.Map()
folium.plugins.AntPath(locations = lista_geo, popup = 'Ruta 20/11/2023',
↳reverse="True", dash_array=[10, 20], color='red', pulse_color = 'blue').
↳add_to(m)
m.fit_bounds(m.get_bounds())
m
```

[]: