EC601 YongxinJiang Automated refueling planning process

1.program statement

The program is meticulously designed to ensure that drivers can automatically plan their refueling paths, making the entire driving experience more convenient and efficient.

2.User Story

As a driver, your expectation from this program is twofold. For longer journeys, you want the navigation to be intelligent enough to arrange refueling stops strategically along your path. It should consider the total distance to be covered and the vehicle’s fuel efficiency to pinpoint the optimal locations for refueling. The goal is to eliminate the worry of running out of gasoline mid-journey and ensuring that each stop is well-planned and doesn’t lead to unnecessary detours.

In the case of shorter trips, the program should aim at identifying a refueling station near the destination point. This ensures that upon arrival, the vehicle is refueled and ready for the next journey. It’s about instilling confidence in the driver that they won’t be left stranded due to an empty fuel tank and that the next trip can be started without any trouble.

3.MVPS

For the MVP, the process begins with the program retrieving the path between the designated starting point and the end point. The initial step is to calculate the total distance of the journey. If this distance falls below a certain threshold, the program jumps into action, retrieving all the available gas stations in proximity to the end point. It then selects the closest station, integrates it into the planned path, and re-calculates the journey ensuring the added stop is efficient and doesn’t lead to significant deviations.

For journeys that exceed the predetermined distance threshold, the program adopts a more complex approach. It doesn’t just look at the end point but also keeps an eye on the entire path. At regular intervals, based on the vehicle’s fuel consumption rate and the total distance to be covered, it identifies potential refueling points. It retrieves the gas stations available surrounding these points, evaluates them based on proximity and convenience, and then seamlessly integrates the selected stations into the path. The re-planned path ensures that the driver can cover the long distance with confidence, knowing well that refueling stops have been systematically arranged.

In essence, this program is about blending technology with foresight. It’s about ensuring that every journey, short or long, is covered with the assurance that refueling won’t be a trouble but a well-orchestrated stop that is woven seamlessly into the journey.

1. Code

import googlemaps

import folium

from datetime import datetime

import polyline

import geopy.distance

API\_KEY = 'AIzaSyBrhDCbzeWtINOlos2Abc4O6lJFGV-haN8'

gmaps = googlemaps.Client(key=API\_KEY)

origin = input("Enter the starting point: ")

destination = input("Enter the destination: ")

directions\_result = gmaps.directions(origin**,** destination**,** departure\_time=datetime.now())

total\_distance = directions\_result[**0**]['legs'][**0**]['distance']['value'] / **1000**

route\_map = folium.Map(location=[**35.8617, 104.1954**]**,** zoom\_start=**5**)

path\_coordinates = polyline.decode(directions\_result[**0**]['overview\_polyline']['points'])

if total\_distance < **300**:

end\_location = (directions\_result[**0**]['legs'][**0**]['end\_location']['lat']**,** directions\_result[**0**]['legs'][**0**]['end\_location']['lng'])

places\_result = gmaps.places\_nearby(location=end\_location**,** radius=**50000,** type='gas\_station')

if places\_result['results']:

stations = places\_result['results']

distances = [geopy.distance.distance(end\_location**,** (station['geometry']['location']['lat']**,** station['geometry']['location']['lng'])).km for station in stations]

nearest\_station\_index = distances.index(min(distances))

nearest\_station = stations[nearest\_station\_index]

folium.PolyLine(path\_coordinates**,** color='blue').add\_to(route\_map)

folium.Marker(

location=(nearest\_station['geometry']['location']['lat']**,** nearest\_station['geometry']['location']['lng'])**,**

popup="Gas Station"**,**

icon=folium.Icon(color='red'**,** icon='info-sign')

).add\_to(route\_map)

else:

print("No gas stations found.")

else:

travel\_distance = **0**

last\_station\_coord = path\_coordinates[**0**]

for coord in path\_coordinates[**1**:]:

segment\_distance = geopy.distance.distance(last\_station\_coord**,** coord).km

travel\_distance += segment\_distance

if travel\_distance >= **300**:

places\_result = gmaps.places\_nearby(location=coord**,** radius=**50000,** type='gas\_station')

if places\_result['results']:

station = places\_result['results'][**0**]

folium.Marker(

location=(station['geometry']['location']['lat']**,** station['geometry']['location']['lng'])**,**

popup="Gas Station"**,**

icon=folium.Icon(color='red'**,** icon='info-sign')

).add\_to(route\_map)

travel\_distance = **0**

else:

print(f"No gas stations found near {coord}.")

last\_station\_coord = coord

folium.PolyLine(path\_coordinates**,** color='blue').add\_to(route\_map)

route\_map.save("route\_map.html")

print("The map has been saved as route\_map.html. Open it with a web browser.")

Here are two test maps，From Boston college to Boston University central and from Florida University to Boston University



