**Video File Naming Convention**

8/17/25

# **Suggested Video File Names for S3 Uploading**

video/<vehicle id>/<equipment id>/<start date>-<start time>-<start city name>-<start street name>-<starting compass heading>.mp4

# **How to Obtain City and Street Names**

To obtain an address from coordinates in Python, reverse geocoding is required. This process converts geographic coordinates (latitude and longitude) into a human-readable address. The Google Maps Geocoding API offers a reverse geocoding service. This method requires a Google Cloud project and an API key (see my active API key below).

This is a paid service with some initial free portion. I believe the free portion will be enough for our development and research project. In order to scale up we will need to developed our own service based on Open Street Maps.

Here is a Python code to get address by coordinates

import requests

api\_key = "API\_KEY"

def get\_address\_from\_coordinates(api\_key, latitude, longitude):

"""

Performs reverse geocoding to get an address from coordinates.

Args:

api\_key (str): Your Google Maps API key.

latitude (float): The latitude coordinate.

longitude (float): The longitude coordinate.

Returns:

str: The formatted address, or None if an error occurs.

"""

url = "https://maps.googleapis.com/maps/api/geocode/json"

params = {

"latlng": f"{latitude},{longitude}",

"key": api\_key

}

response = requests.get(url, params=params)

if response.status\_code == 200:

data = response.json()

if data["status"] == "OK" and data["results"]:

return data["results"][0]["formatted\_address"]

else:

print(f"Error: {data.get('error\_message', 'No results found')}")

return None

else:

print(f"HTTP Error: {response.status\_code}")

return None

# Example usage:

lat = 37.7749 # Example latitude

lon = -122.4194 # Example longitude

address = get\_address\_from\_coordinates(api\_key, lat, lon)

print(f"Address: {address}")

# **How to Calculate Compass Direction**

from geographiclib.geodesic import Geodesic

def get\_compass\_direction\_geographiclib(lat1, lon1, lat2, lon2):

"""

Calculates the initial azimuth (compass direction) in degrees

from point (lat1, lon1) to point (lat2, lon2) using geographiclib.

"""

geod = Geodesic.WGS84 # Use the WGS84 ellipsoid model

# Inverse calculation to find the geodesic properties between two points

result = geod.Inverse(lat1, lon1, lat2, lon2)

# azi1 is the initial azimuth (bearing) from point 1 to point 2

return result['azi1']

# Example usage:

lat\_a, lon\_a = 41.4107628, 2.1745004

lat\_b, lon\_b = 41.4126728, 2.1704725

bearing\_geographiclib = get\_compass\_direction\_geographiclib(lat\_a, lon\_a, lat\_b, lon\_b)

print(f"The compass bearing from point A to point B (geographiclib): {bearing\_geographiclib:.2f} degrees")

**Note:** The azi1 value returned by geographiclib is the initial azimuth, measured clockwise from true North, with values typically in the range of -180 to 180 degrees.

In the Google Street View Static API, the heading parameter, which indicates the compass heading of the camera, accepts values from 0 to 360. Both 0 and 360 degrees represent North, with 90 degrees indicating East, and 180 degrees indicating South. In order to match the Google Earth representation, you can normalize the output of the above routine using (azi1 + 360) % 360.