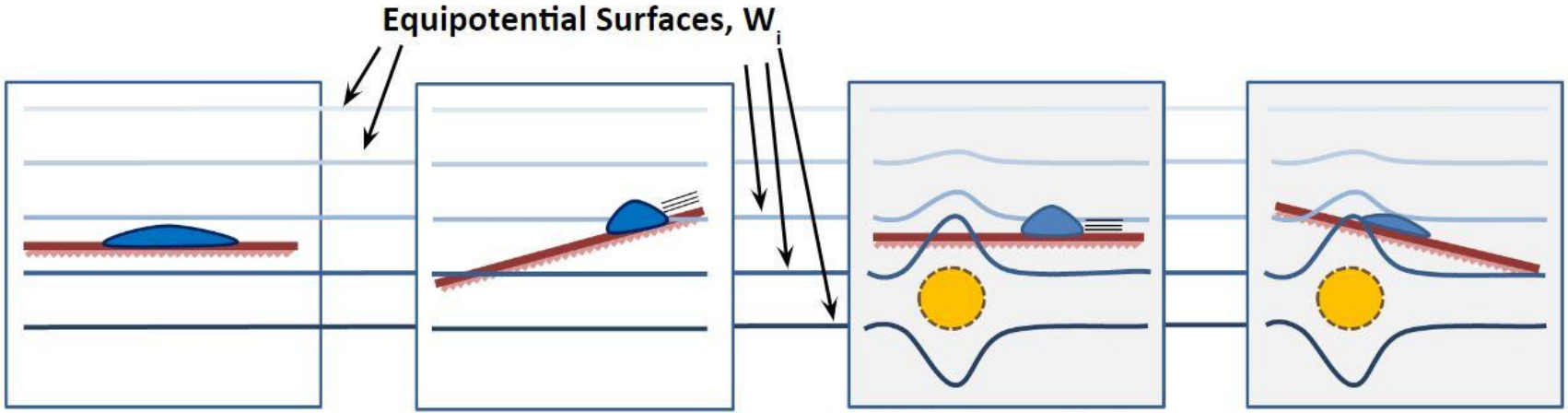
Google Developer Group
Gela

Why flat?
Why code?

Ing. Gianfranco Di Pietro
gianfrancodp.github.io

@gianfrancodp

Why Flat?



Imagine pouring water onto a dry lake bed. If everything is level it will simply pool up. It would "like" to get to lower potential (darker lines), but the lake bed is in the way.

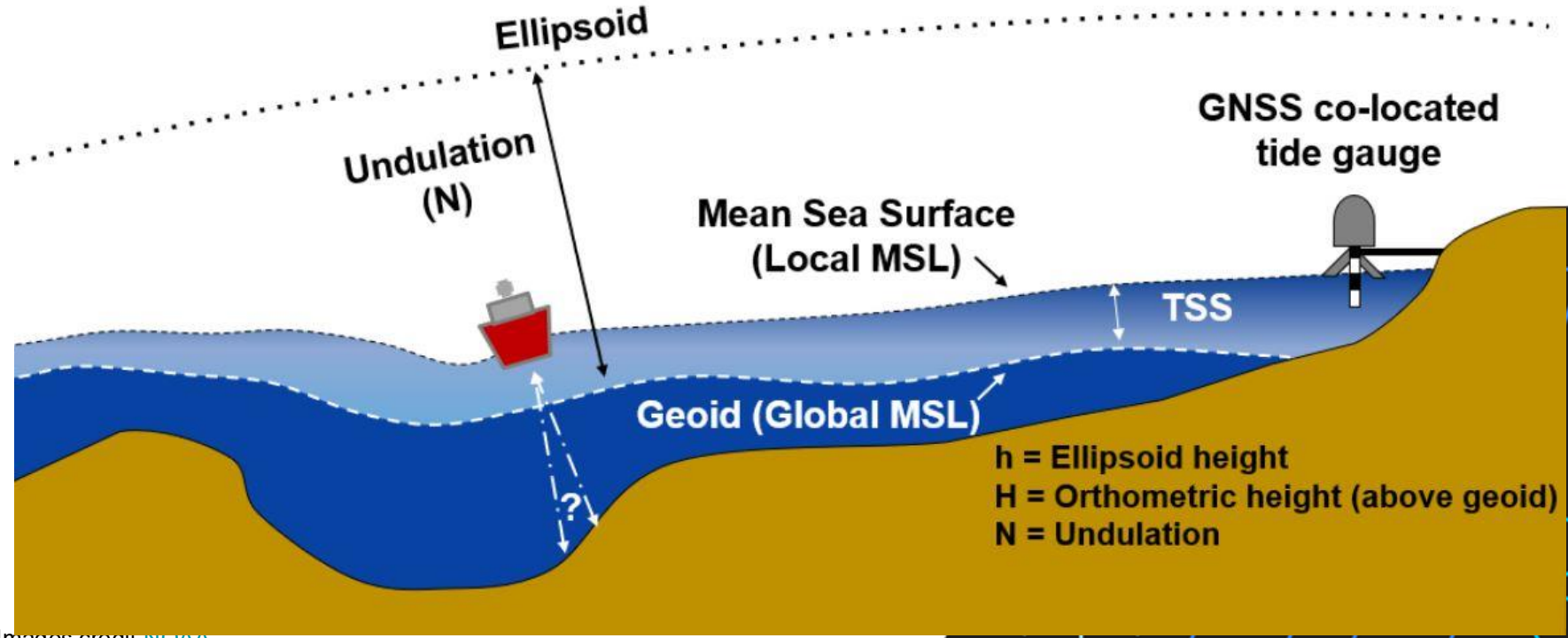
If the lake bed is tilted (it's now the side of a hill), the water can move towards lower potential (towards darker lines). Water appears to flow because of a change in height

If the lake bed appears flat, but there is a mass anomaly under one end, the resulting geopotential difference causes water to flow to lower potential (again, from light to dark lines). It appears to flow because of gravity.

Exaggerated (!), but you can imagine a scenario where the height difference and gravity attraction "conspire" to keep the water still on the side of a hill!

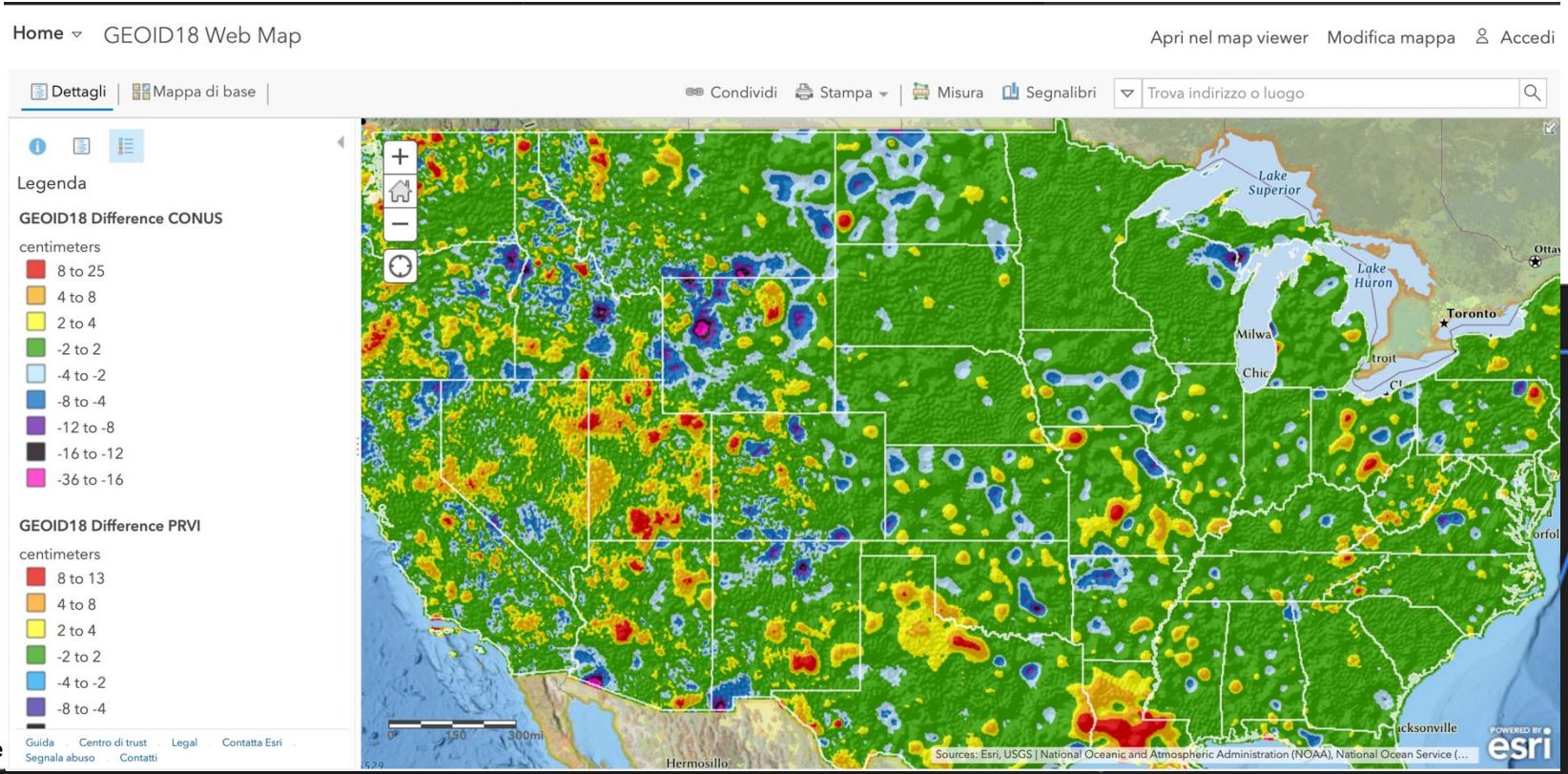
Images credit NOAA

Why Flat?



Images credit NOAA

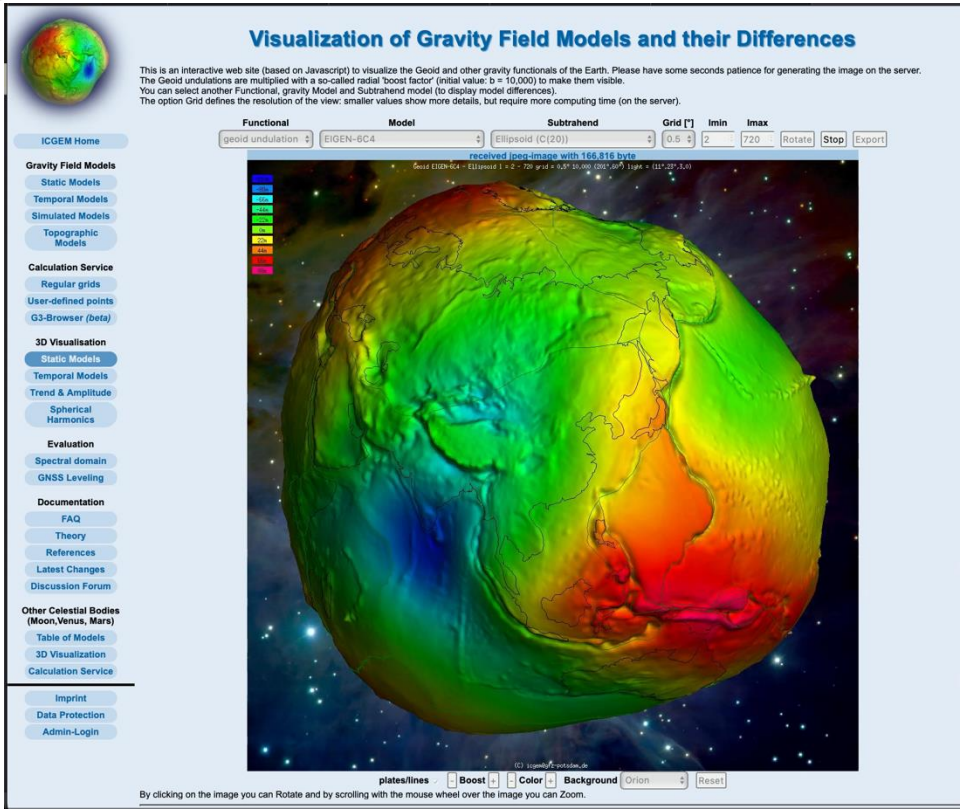
Why Flat?

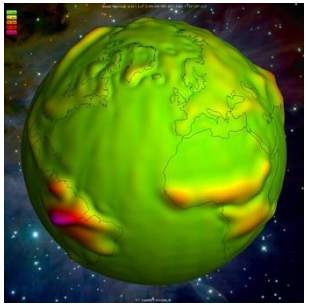


Image

Why Flat?

- Timeseries of Gravity anomalies
- Geoid data as anomalies from simplified geometries (ellipsoid)

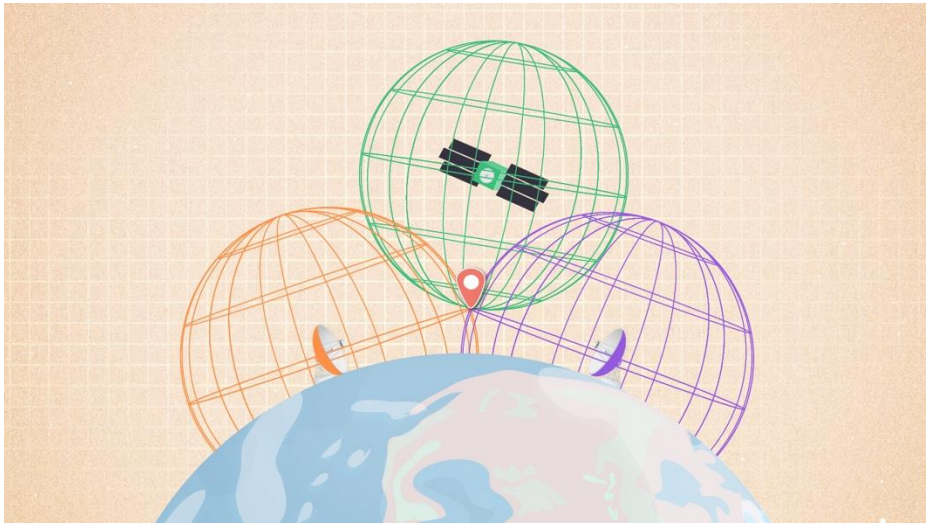




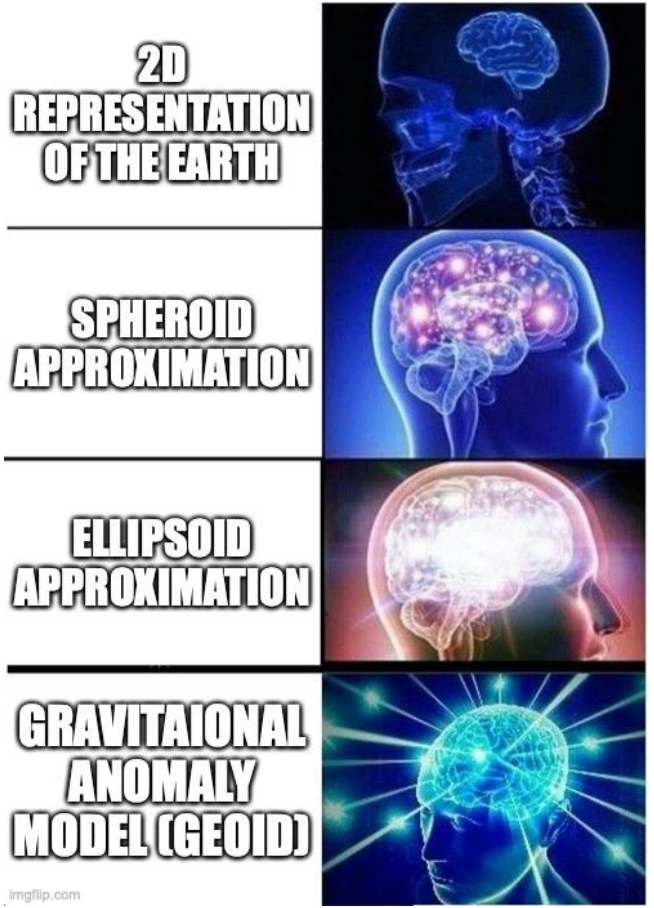
Images credit [International Centre for Global Earth Models (ICGEM)]

WGS84 and Global Poistion System

Lat/Lon value is angles not meters!

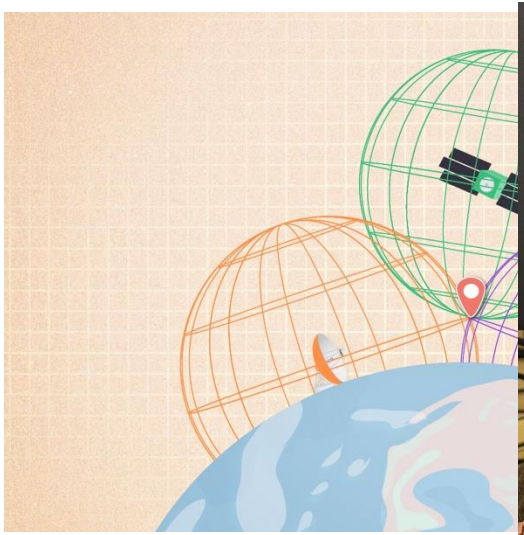


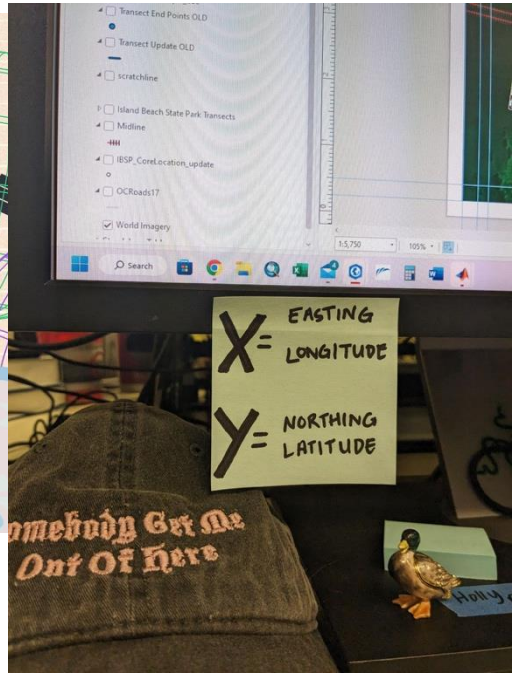
Credit: NASA/JPL-Caltech



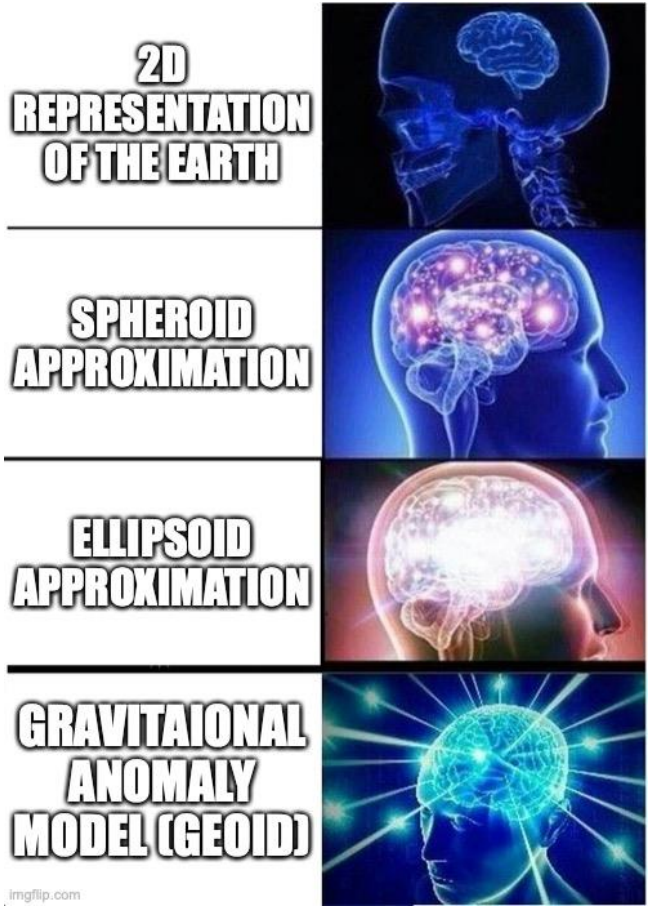
WGS84 and Global Poistion System

Lat/Lon value is angles not meters!





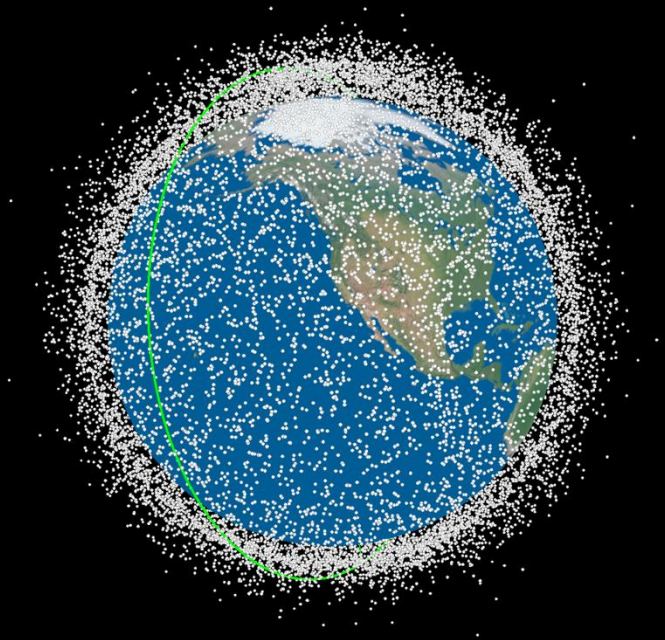
Credit: NASA/JPL-Caltech



Satellite Tracker 3D

Name or NORAD ID

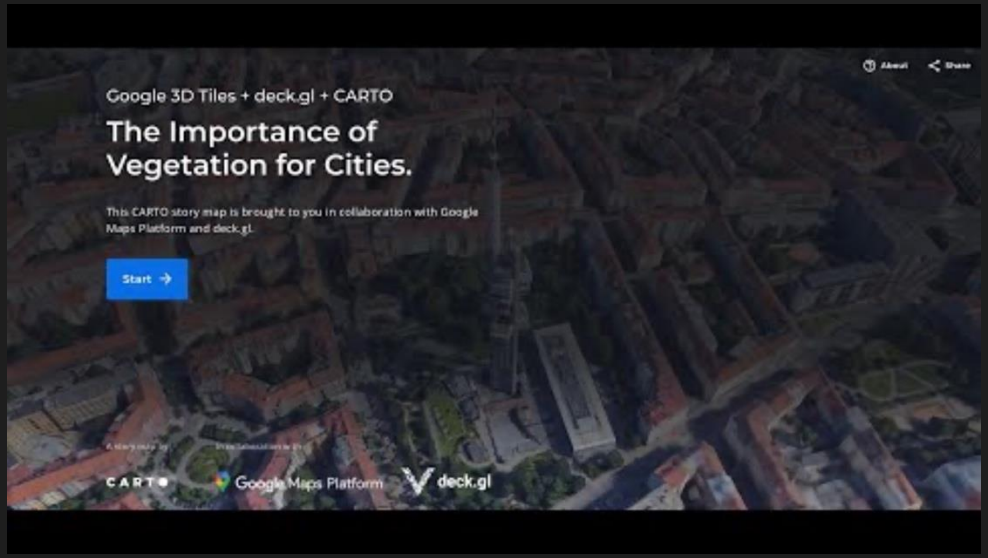
DOVE 3



SURCAL 159 #2872

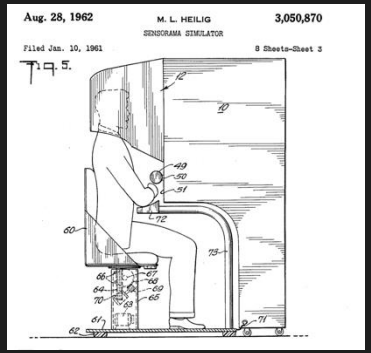
Speed: 26,612 km/h
Height: 919 km
Latitude: -62.44°
Longitude: -108.72°

Google Developer Group
Gela



I soldati dell'esercito americano si ammalano durante il test delle cuffie basate su HoloLens di Microsoft

3D in geospatial, una sfida aperta

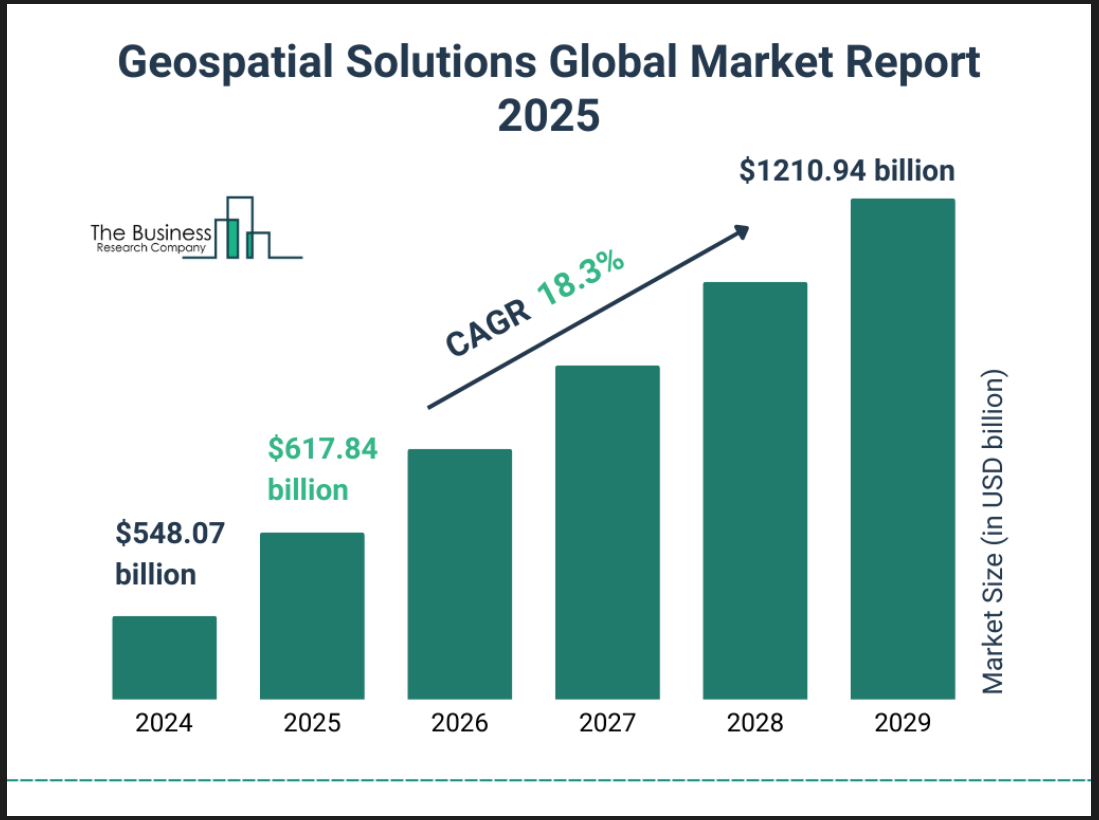


Google Developer Group
Gela

2025 compound annual growth rate

12.7%

Con una crescita stimata del 18.3 nei prossimi anni



source: https://www.thebusinessresearchcompany.com/

Google Developer Group
Gela

Why Code?

1. Scripting, analysis and report
2. Deploy services (back-end)
3. Data management
4. Data viz and sharing (front-end)

Why Code?

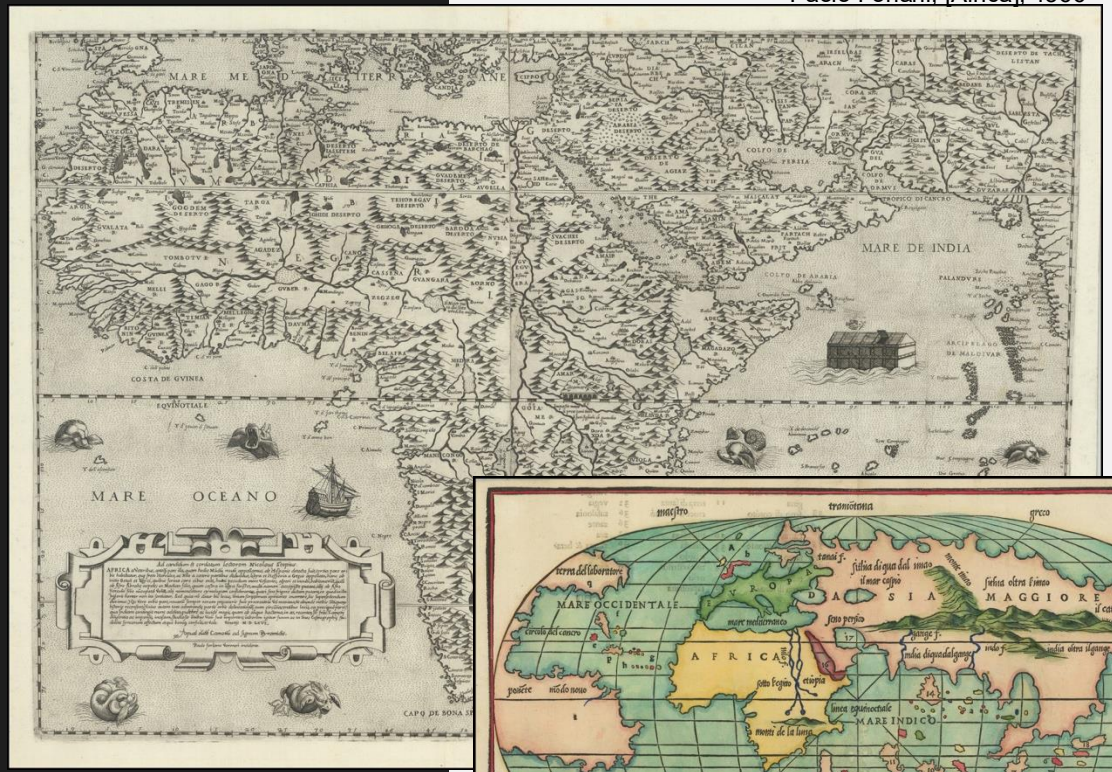
1. Computational thinking.
2. Data thinking
3. Data science thinking
4. Geospatial thinking

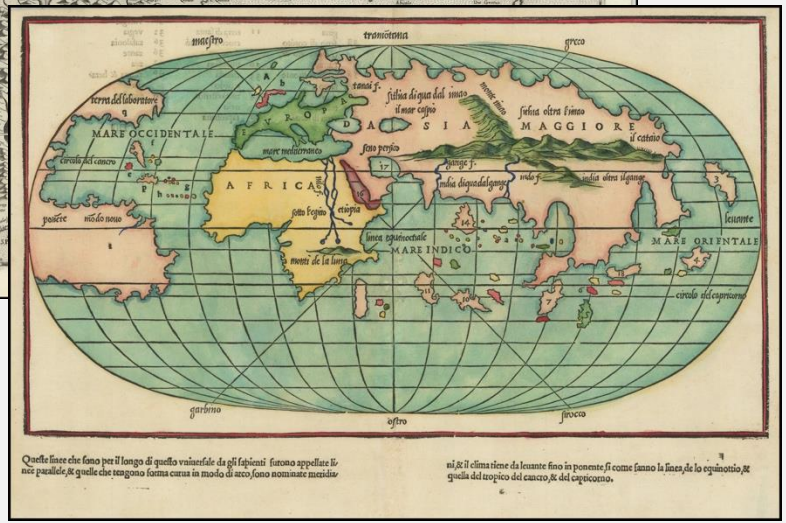
Geospatial technology is…
and is not…

@gianfrancodp

A little history…

Paolo Forlani, [Africa], 1566





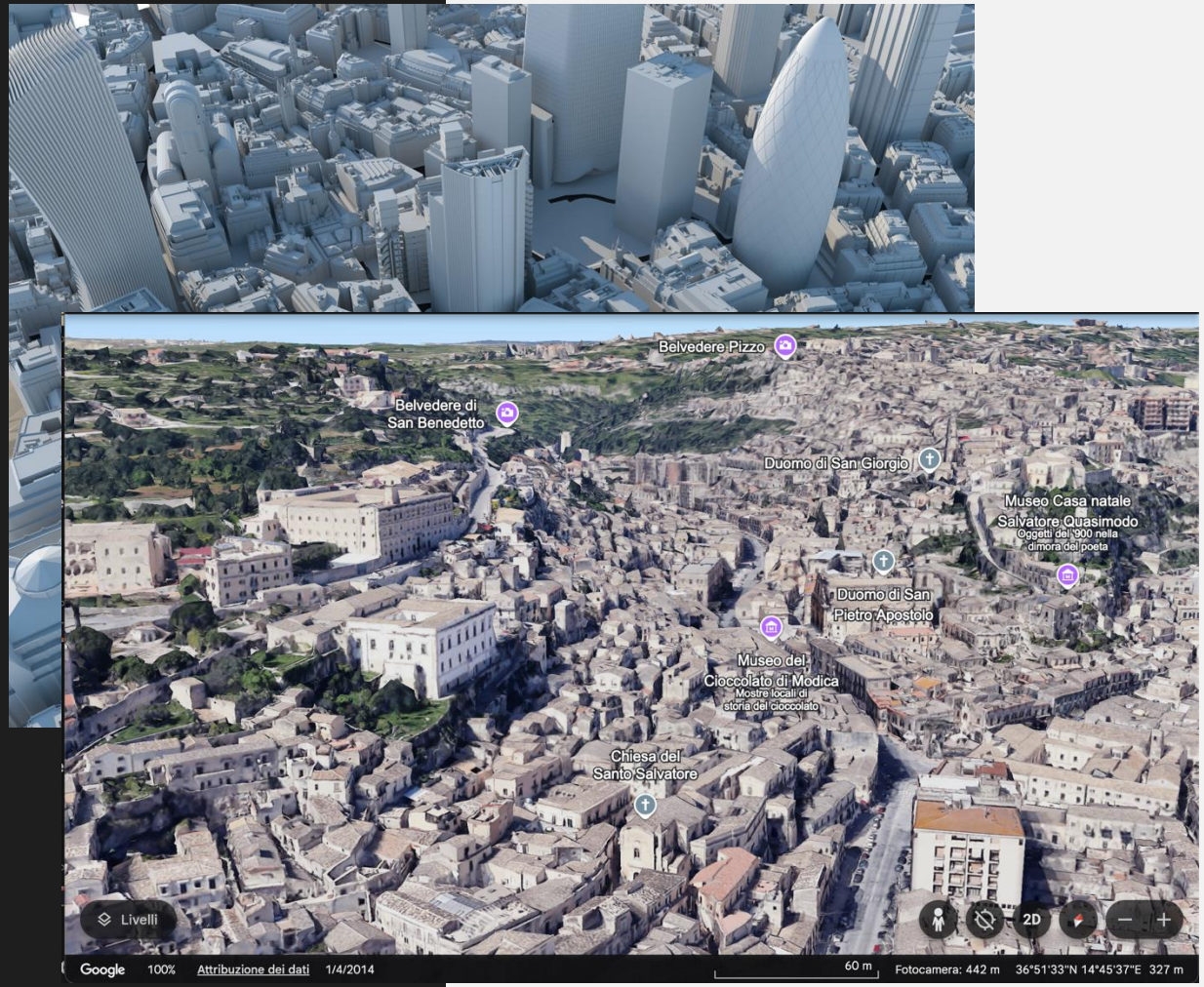
Benedetto Bordone, 1534

Google Developer Group
Gela

A little history…

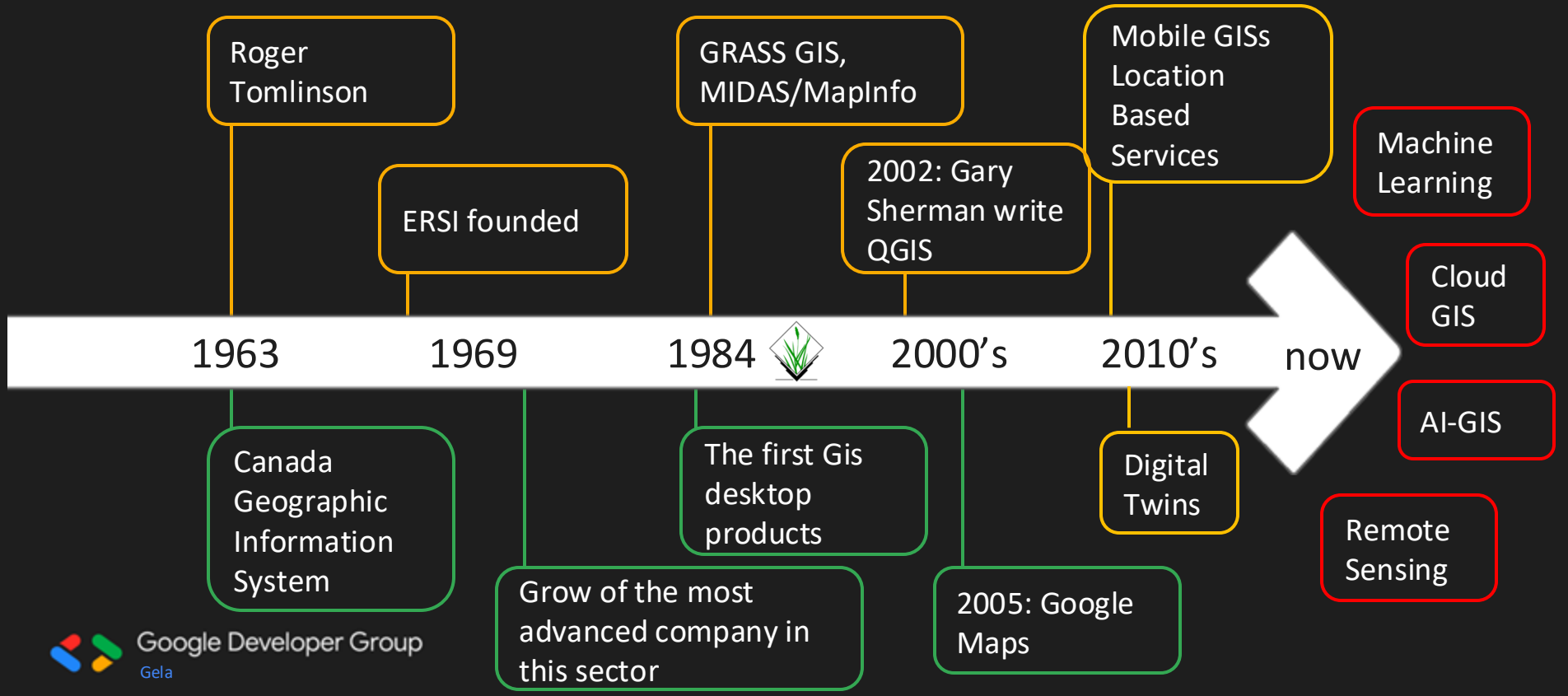
British Patè [Making Maps], 1961

A little history…

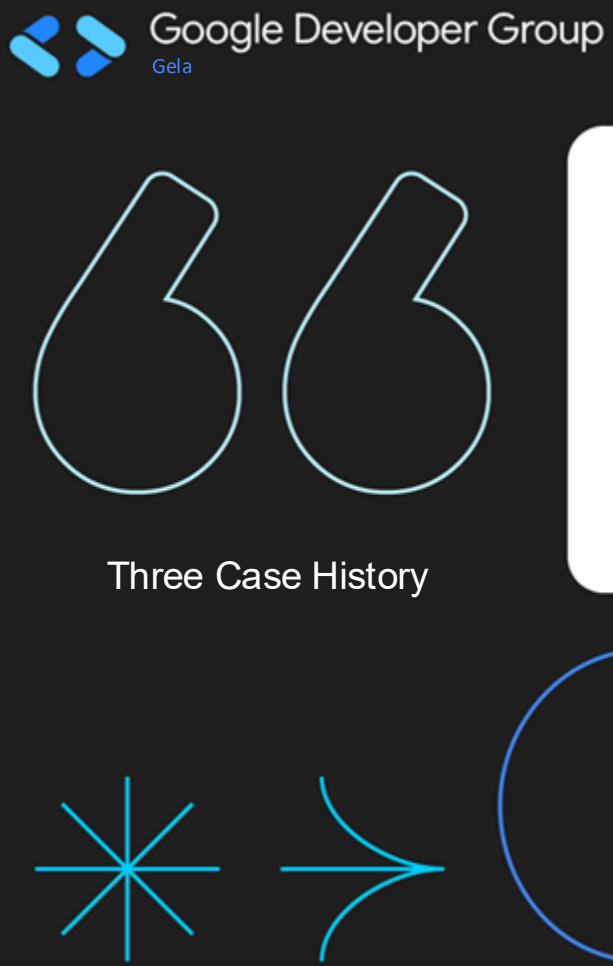


Google Developer Group
Gela

Milestone of GIS (Geographical Information Systems)



Google Developer Group
Gela

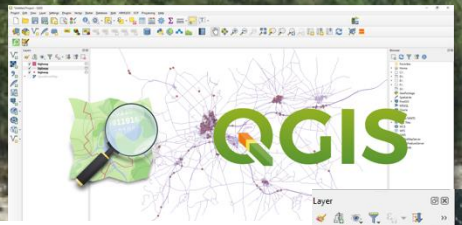

Google Developer Group
Gela
I wisely started with a map
J.R.R. Tolkien
Three Case History

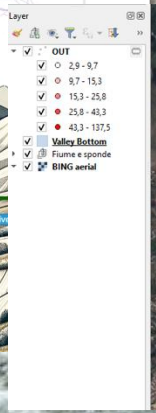
Case #1: NO-CODE approach

Qgis Riverbanks Tools

toolbox for river analysis

with Qgis and Python





https://github.com/gianfrancodp/qgis-riverbanks-tools

Case #1: NO-CODE approach

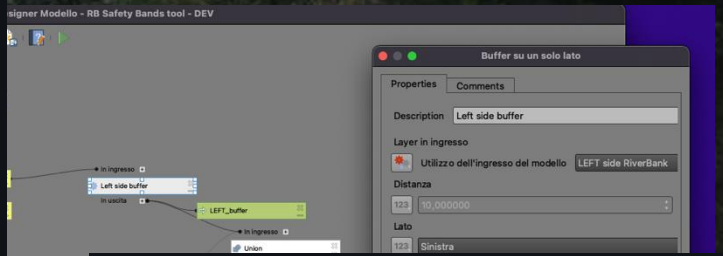
Qgis Riverbanks Tools

qgis-riverbanks-tools / Models / CVI / **Confined_Valley_index_v.1.1.py** ↑ Top

Code | Blame | Executable File · 607 lines (536 loc) · 32.2 KB | Raw

```
#Imports from qgis.core

from qgis.core import QgsProcessing
from qgis.core import QgsProcessingAlgorithm
from qgis.core import QgsProcessingMultiStepFeedback
from qgis.core import QgsProcessingParameterVectorLayer
from qgis.core import QgsProcessingParameterNumber
from qgis.core import QgsProcessingParameterBoolean
from qgis.core import QgsProcessingParameterFeatureSink
from qgis.core import QgsProcessingParameterDefinition
import processing


class ConfinedValleyIndex(QgsProcessingAlgorithm):

    #Definition of input parameters
    def initAlgorithm(self, config=None):
        self.addParameter(QgsProcessingParameterVectorLayer('left_river_bank', 'LEFT River Bank', types=[QgsProcessing.TypeVectorLine]
        self.addParameter(QgsProcessingParameterVectorLayer('right_river_bank', 'RIGHT River Bank', types=[QgsProcessing.TypeVectorLin
        self.addParameter(QgsProcessingParameterVectorLayer('river_line', 'River Line', types=[QgsProcessing.TypeVectorLine], defaultV
        param = QgsProcessingParameterNumber('sezid_number_lenght', 'SEZ-ID number lenght', type=QgsProcessingParameterNumber.Integer,
        param.setFlags(param.flags() | QgsProcessingParameterDefinition.FlagAdvanced)
        self.addParameter(param)
        param = QgsProcessingParameterBoolean('transects_id_assignment_revert', 'Transects ID assignment revert', defaultValue=False)
        param.setFlags(param.flags() | QgsProcessingParameterDefinition.FlagAdvanced)
        self.addParameter(param)
        self.addParameter(QgsProcessingParameterNumber('transects_step', 'Transects STEP', type=QgsProcessingParameterNumber.Integer,
        self.addParameter(QgsProcessingParameterNumber('transects_width_m', 'Transects WIDTH [m]', type=QgsProcessingParameterNumber.I
        self.addParameter(QgsProcessingParameterVectorLayer('valley_bottom', 'Valley Bottom', types=[QgsProcessing.TypeVectorPolygon],
        self.addParameter(QgsProcessingParameterFeatureSink('Out', 'OUT', type=QgsProcessing.TypeVectorAnyGeometry, createByDefault=Tr

    def processAlgorithm(self, parameters, context, model_feedback):
```

Exported .py file with class and scripts



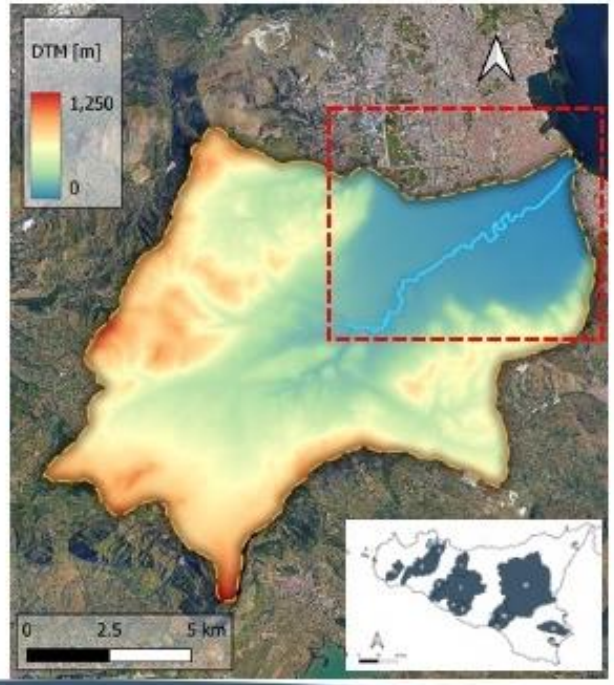
Code | Blame | 2385 lines (2385 loc) · 128 KB | Raw

```
<!DOCTYPE model>
<Option type="Map">
  <Option type="Map" name="children">
    <Option type="Map" name="native:fieldcalculator_1">
      <Option type="bool" name="active" value="true"/>
      <Option name="alg_config"/>
      <Option type="QString" name="alg_id" value="native:fieldcalculator"/>
      <Option type="QString" name="color" value=""/>
      <Option type="Map" name="comment">
        <Option type="QString" name="color" value=""/>
        <Option type="QString" name="component_description" value=""/>
        <Option type="double" name="component_height" value="116.00000000000001"/>
        <Option type="double" name="component_pos_x" value="1491"/>
        <Option type="double" name="component_pos_y" value="576"/>
        <Option type="double" name="component_width" value="149.99999999999997"/>
        <Option type="bool" name="outputs_collapsed" value="true"/>
        <Option type="bool" name="parameters_collapsed" value="true"/>
      </Option>
      <Option type="QString" name="component_description" value="Assign SEZ-ID to centerline points"/>
      <Option type="double" name="component_height" value="38.037166486819615"/>
      <Option type="double" name="component_pos_x" value="1265"/>
      <Option type="double" name="component_pos_y" value="524.0185832434098"/>
      <Option type="double" name="component_width" value="200"/>
      <Option name="dependencies"/>
      <Option type="QString" name="id" value="native:fieldcalculator_1"/>
      <Option name="outputs"/>
      <Option type="bool" name="outputs_collapsed" value="false"/>
```

Native .model3 file format (xml-like)

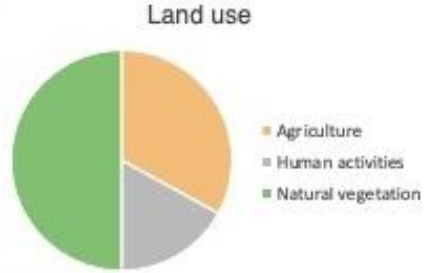
Case #1: NO-CODE approach

Qgis Riverbanks Tools



Watershed characterization

Area [km²]	129.7
Perimeter [km]	57.6
Maximum height [m a.m.s.l.]	1,250
Mean height [m a.m.s.l.]	466



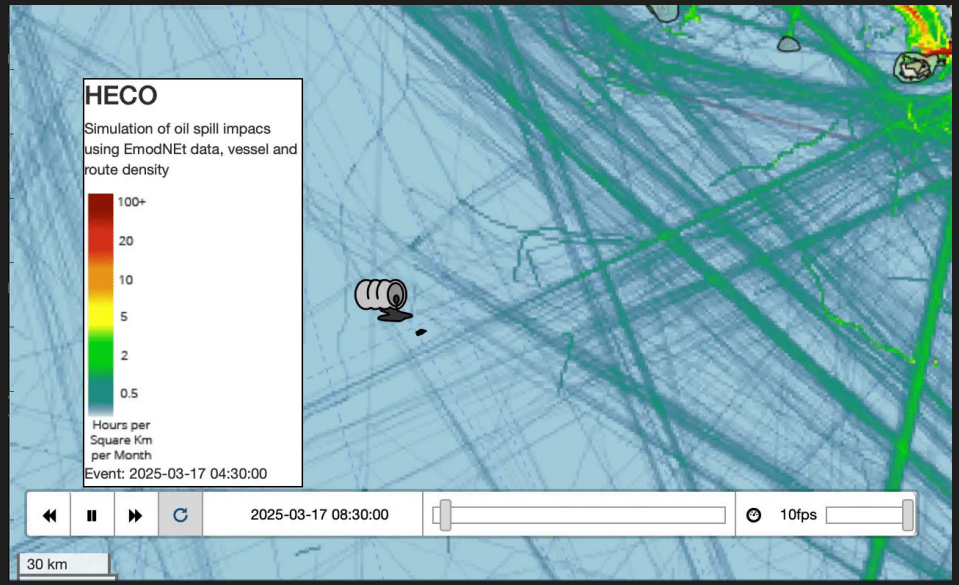
Studied river segment

- Length: 12 km
- Cross-section width:15÷50 m
- Mean slope:1.04%
- Hydraulic regime: intermittent

at (xml-like)

Case #2 – Hardcoded geoprocessing script

HECO: Here Comes the Oil! – proof of concept



Highlights

📌 A quick tool for simulating surface oil spill using 2D surface simplified LDPM model.

🛰️ EMODnet data and real-time sea current forecasting used as forcings

⚠️ Identifies at-risk marine areas and vessel routes

🗺️ Interactive web-GIS environment for instant visualization

🚢 Supports decision-making in emergency response

HECO: Here Comes the Oil! – Proof of concept

https://github.com/SeaQuestTeam/HECO

heco.py

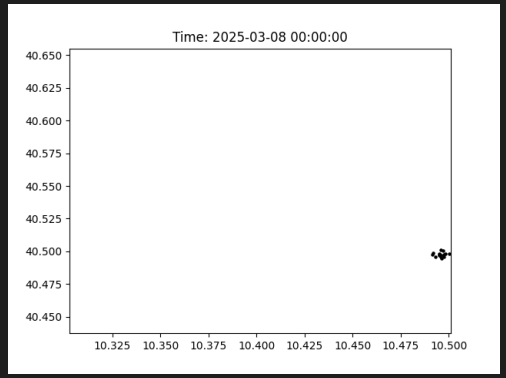
HECO / heco / heco.py

gianfrancodp updated function name

Code | Blame | 879 lines (681 loc) · 32.2 KB

```
'''
This module contains the functions develped for the HECO Proof of Concept
author: Gianfranco Di Pietro - PhD student at University of Catania
contributors: Martina Stagnitti, Massimiliano Marino, Elisa Castro, Sofia Nasca
supervisor: Rosaria Ester Musumeci
'''

import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import matplotlib.colorbar as mcb
from matplotlib.patches import FancyBboxPatch
from matplotlib.animation import FuncAnimation, PillowWriter
import pandas as pd
import numpy as np
import os
import cartopy.crs as ccrs
import cartopy.io.img_tiles as cimgt
import xarray as xr
import ssl
import ipywidgets as widgets
from IPython.display import display
import pyproj
from glob import glob
from geopandas import GeoDataFrame
from shapely.geometry import Point, LineString
import yaml
```

Lagrangian Dispersion Particle Model



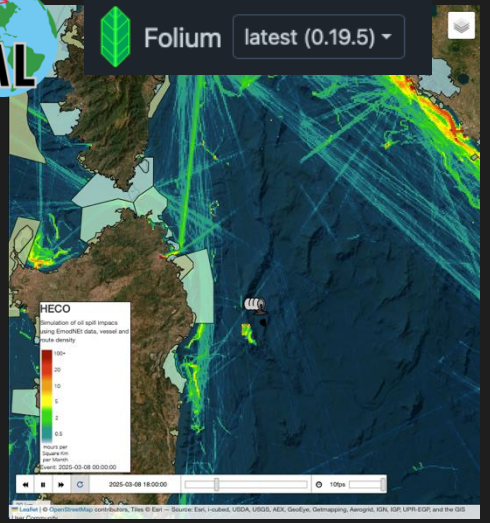

```
472         # Compute the new position of the oil spill
473         x_new = x + u * dt + np.random.normal(0,1)* np.sqrt(2 * D * dt)
474         y_new = y + v * dt + np.random.normal(0,1)* np.sqrt(2 * D * dt)
```

GeoPandas

Data web-gis visualization

GDAL

Folium latest (0.19.5)



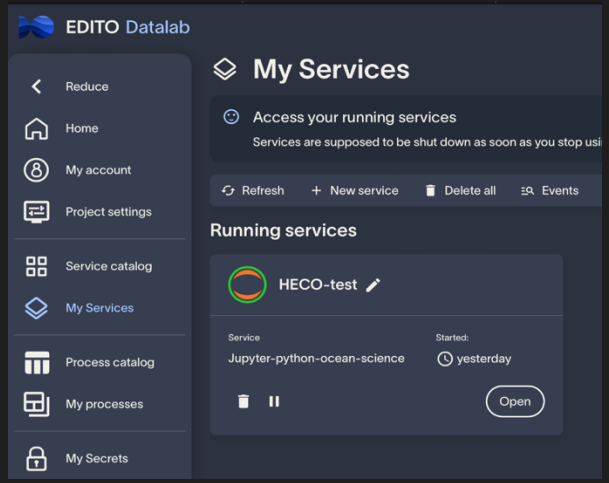
Google Developer Group
Gela

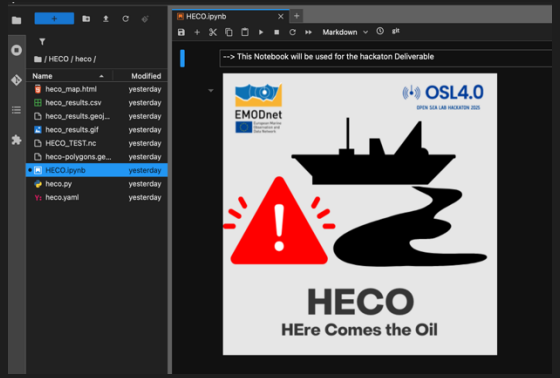
Case #2 – Hardcoded geoprocessing script

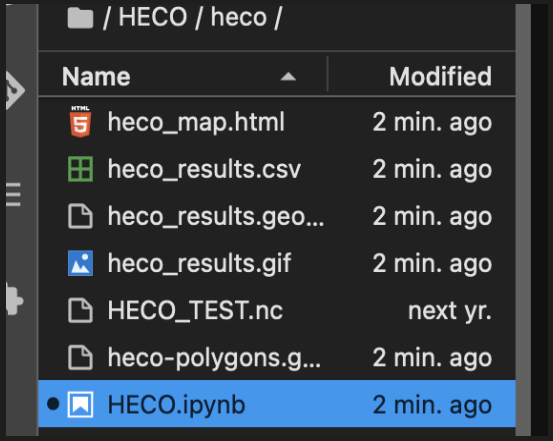
HECO: Here Comes the Oil! – Proof of concept

https://github.com/SeaQuestTeam/HECO

Deploy in EDITO's cloud containers

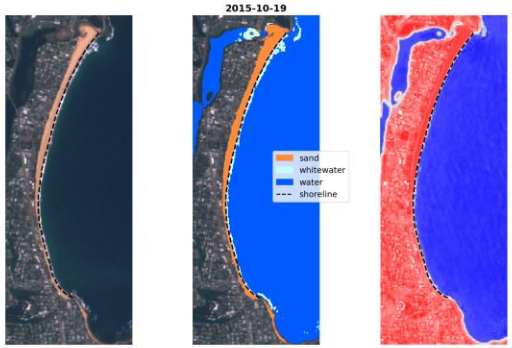


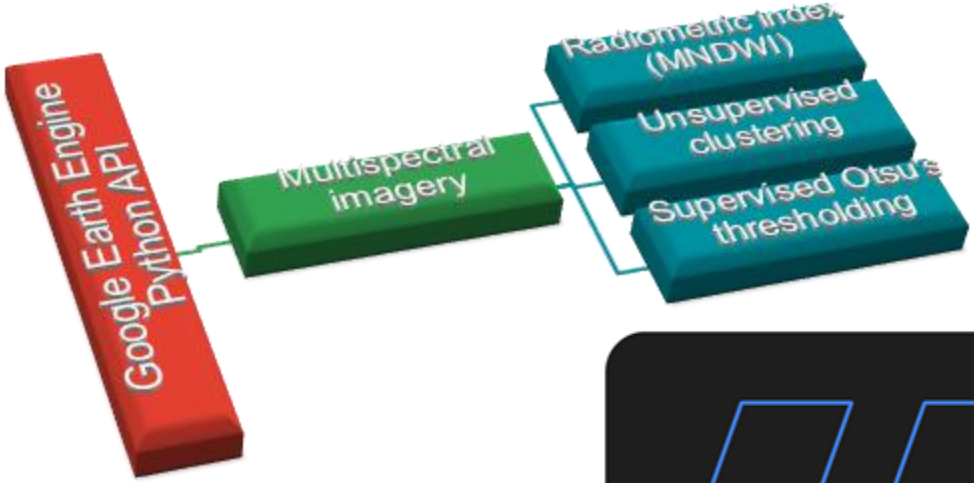




Google Developer Group
Gela

Case #2 – Hardcoded geoprocessing script





Satellite Derived Shoreline

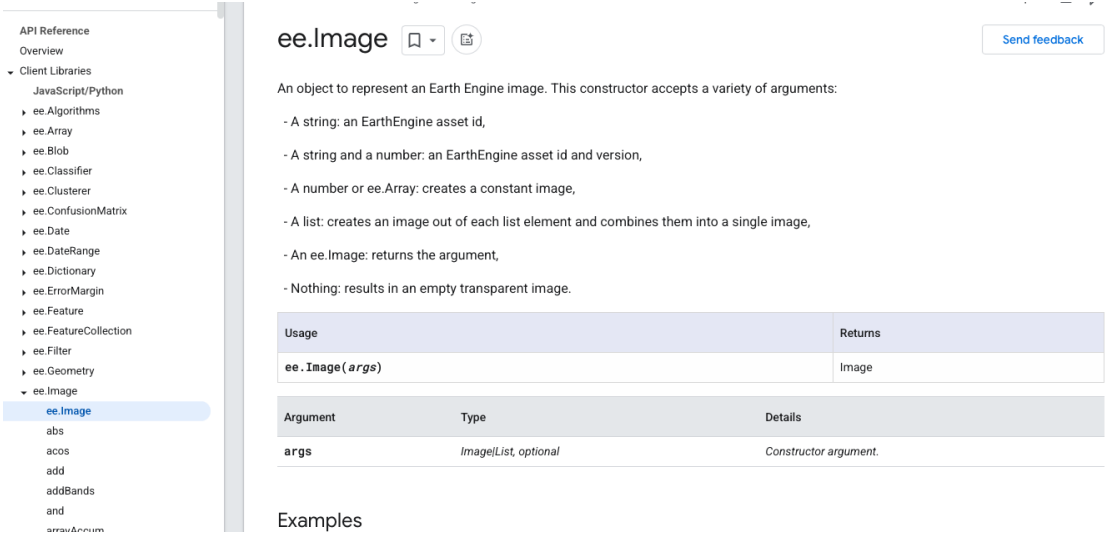
ML-Segmentation from Google Earth Engine datasets

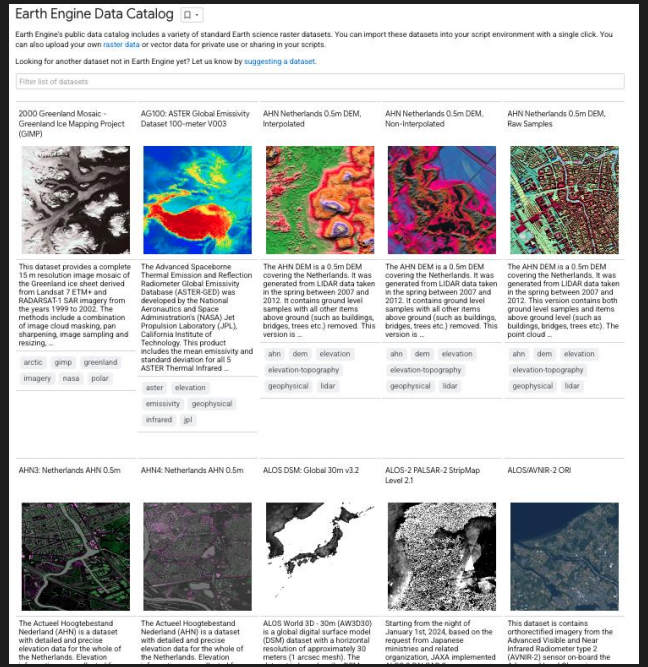
Google Developer Group
Gela

Google Earth Engine

EE contain more than 40 years of satellite and scientific data

1. Registrare un Progetto Google cloud
2. Abilitare Python API
3. Importare in python/javascript: ee (aka. Earth Engine)

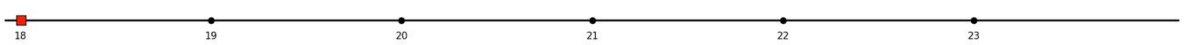


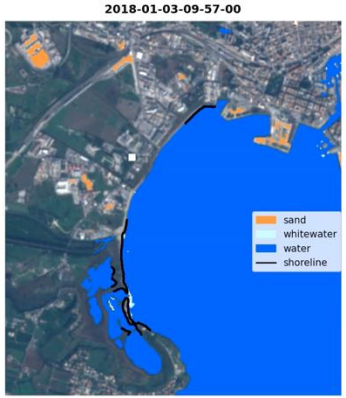


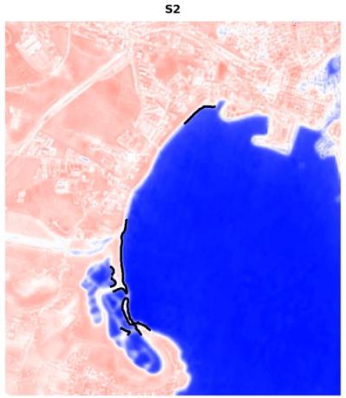
CASE #3 – Machine Learning for Remote Sensing

Machine learning in Python:
The power of scikitlearn and matplotlib

- Classification: Identifying which category an object belongs to
- Used pre-trained neural network pretrained classifier with *joblib* function

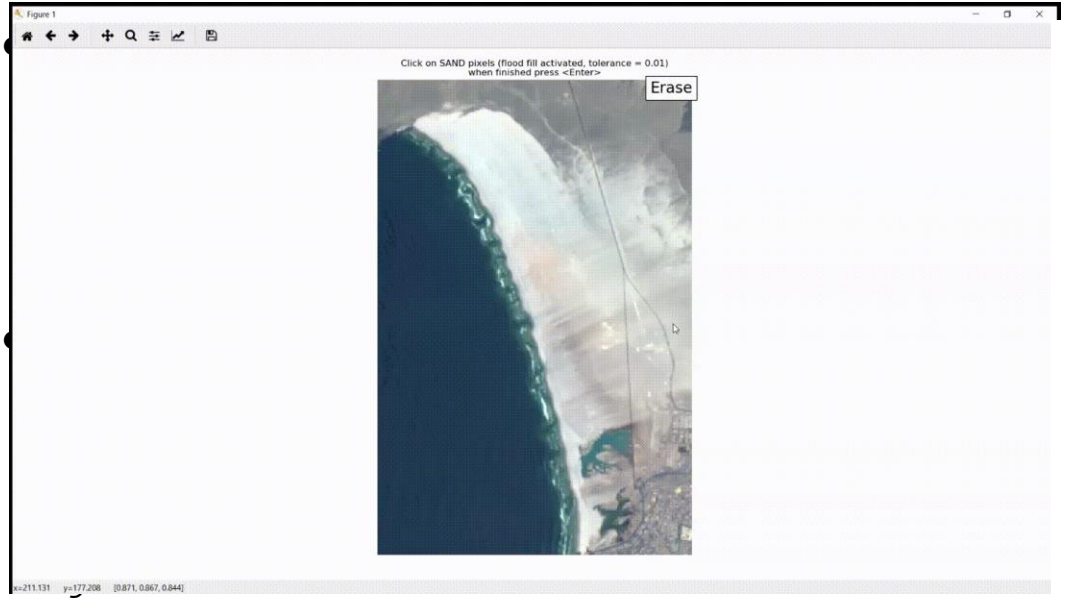






Google Developer Group
Gela

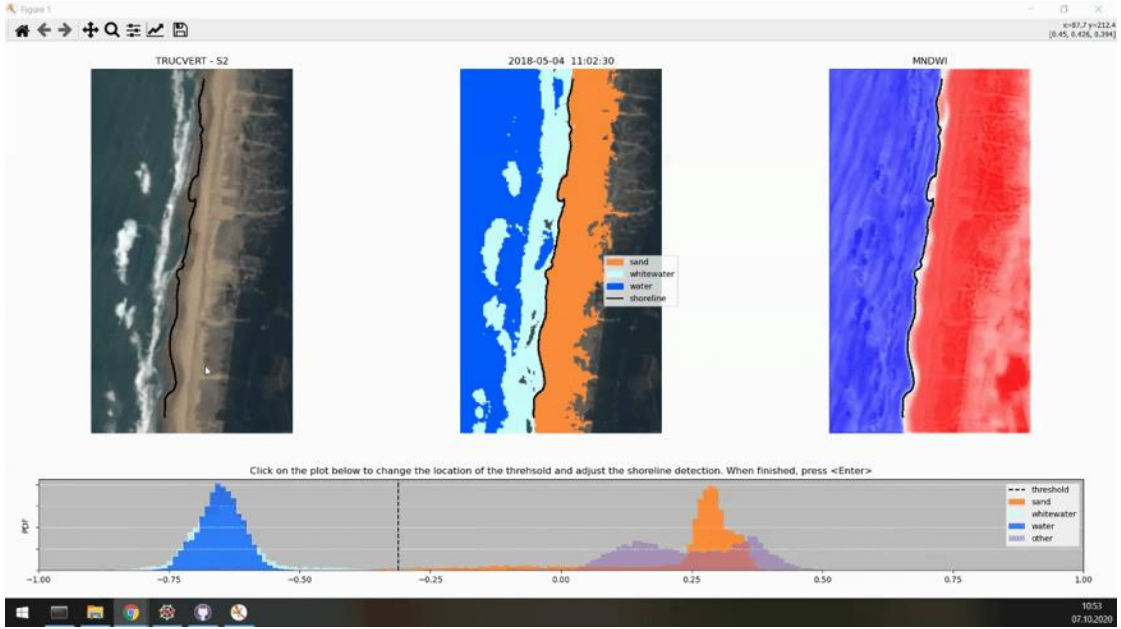
Machine learning in Python:
The power of scikitlearn and matplotlib



- Train new classifier with manual labeling and matplotlib inside a jupyter notebook.
- Supervised threshold adjusting for better segmentation

Google Developer Group
Gela

Machine learning in Python:
The power of scikitlearn and matplotlib



- Train new classifier with manual labeling and matplotlib inside a jupyter notebook.
- Supervised threshold adjusting for better segmentation

Google Developer Group
Gela

Google Developer Group
Gela

Grazie per l’attenzione

@gianfrancodp