To use reinforcement learning algorithms

1. Install Python package of OpenVO library from github repository.

2. Next step, create project in PyCharm, and create file \*.py.

3. Add imports:

from Open\_VO import (MapAnalysis, Detection, DetectorPy, CameraParams, PosAngle, Pos\_i2, Pos\_d3, Pos\_f2, SurfaceData, LocalData)

4. Initialize MapAnalysis class:

map\_analyzer\_full = MapAnalysis(  
 input\_file="input\_tmp2.txt",  
 img\_folder="./",  
 output\_file="output\_tmp2.txt",  
 labels\_file="labels.txt",  
 model\_path="best.torchscript", # Or .pt, .onnx etc.  
 cuda\_enabled=True,  
 img\_w=640, img\_h=640,  
 sc\_thres=0.45, nms\_thres=0.50, max\_d=100,  
 gl\_model\_path="best.torchscript",  
 gl\_cuda\_enabled=True,  
 gl\_img\_w=1280, gl\_img\_h=1280,  
 gl\_sc\_thres=0.3, gl\_nms\_thres=0.45, gl\_max\_d=300  
)

\* adjust parameters for your project

5. Process satellite map:

map\_analyzer\_full.calculate\_map\_objects()  
map\_analyzer\_full.calc\_map\_edges()

6. Find and process local object on frame (“frame” variable):

map\_analyzer\_full.calculate\_local\_objects(frame, identity\_delta=10,  
cam\_params=cam\_p, curr\_angles=angles,  
curr\_offset=offset, meter\_in\_pixel=m\_in\_pix)

7. Check if UV is inside the satellite map:

is\_inside = map\_analyzer\_full.location\_verification(curr\_x=10.0, curr\_y=10.0, fov\_x=5.0, fov\_y=5.0, delta\_fov=1.0)  
print(f"Is (10,10) inside map? {is\_inside}")

8. If it is inside, then try to match local objects and objects from satellite map, calculate deltas and update local objects position:

deltas\_found = map\_analyzer\_full.object\_matcher(  
 curr\_x=50.0, curr\_y=50.0, fov\_x=100.0, fov\_y=100.0, delta\_fov=10.0,  
 delta\_offset=20.0, match\_delta=15, conf\_overlap=2, obj\_per\_class\_thresh=1, scale=0.1  
)  
print(f"Object matcher found deltas: {deltas\_found}")

Knowing calculated deltas you can adjust your UV position.

(You can use this module without NN for satellite map, if it is already processed and saved into the “input\_file”).