Université Tunis El Manar
École Nationale d'Ingénieurs de Tunis

Département des Technologies de l'Information et de la Communication

Tunisia Geospatial Analysis

Made by : **SLIM Hassen**
Supervised by : **M. Frihida Ali**

2024 - 2025

Evaluating Map Projections for Tunisia: A Professional Analysis

SLIM Hassen

May 28, 2025

Abstract

This report examines the effects of three map projections—Conformal (Mercator), Equidistant Conic, and Equal-area (Albers)—on multiple Tunisia geospatial datasets. By projecting administrative boundaries, waterways, highways, coastline, points of interest, and defined locations, and visualizing them in QGIS, we analyze distortions in shape, distance, and area to inform projection choice for various cartographic tasks.

1 Introduction

Map projections inevitably distort aspects of the Earth's surface when rendered on a flat plane. Depending on the intended use—navigation, distance measurement, or area-based thematic mapping—one projection may be preferable over others. Here, we evaluate how three widely used projections impact Tunisia-specific layers, guiding best practices in projection selection.

2 Data and Tools

- **Datasets:** Shapefiles for Tunisia's *administrative boundaries*, *water features*, *highways*, *coastline*, *points of interest (POI)*, and selected *locations*.
- **Software:** Google Colab: Python (GeoPandas, Cartopy, Matplotlib) and QGIS.
- **Projections:**
  1. Mercator (Conformal)
  2. Equidistant Conic (Equidistant)
  3. Albers Equal Area (Equal-area)

3 Methodology

1. **Import and Inspect:** Loaded each shapefile into GeoPandas and verified original CRS (EPSG:4326).
2. **Define Target CRSs:** Configured Cartopy CRS objects:
   - Mercator
   - Lambert Conformal Conic (equidistant) with central longitude 9°E and standard parallels 30°N, 37°N
   - Albers Equal Area with the same parameters as above
3. **Reprojection:** Applied `GeoDataFrame.to_crs()` to transform each layer.
4. **Export:** Saved reprojected layers as shapefiles with descriptive filenames.
5. **Visualization:** In QGIS, loaded all six layers per projection using consistent symbology: red outlines for boundaries, blue for water, orange for highways, green for coastline, purple dots for POI, yellow dots for locations.
6. **Analysis:** Assessed distortions in shape, distance fidelity, and area preservation qualitatively.

4 Results

4.1 Visual Observations

Each projection reshaped Tunisia's geography in characteristic ways:

- **Mercator:** Preserves local shapes but inflates area toward southern regions. Highway segments near the Sahara appear stretched compared to northern highways. Distance measurement is unreliable.
- **Equidistant Conic:** Maintains accurate distances from the central meridian and between the chosen parallels. POI and location distributions align properly along great-circle paths. Shapes exhibit slight curvature near edges.
- **Albers Equal Area:** Areas of water bodies and administrative regions remain true to scale. Coastal outlines and highways appear slightly compressed but proportions are consistent, making area comparisons straightforward.

4.2 Comparison Summary

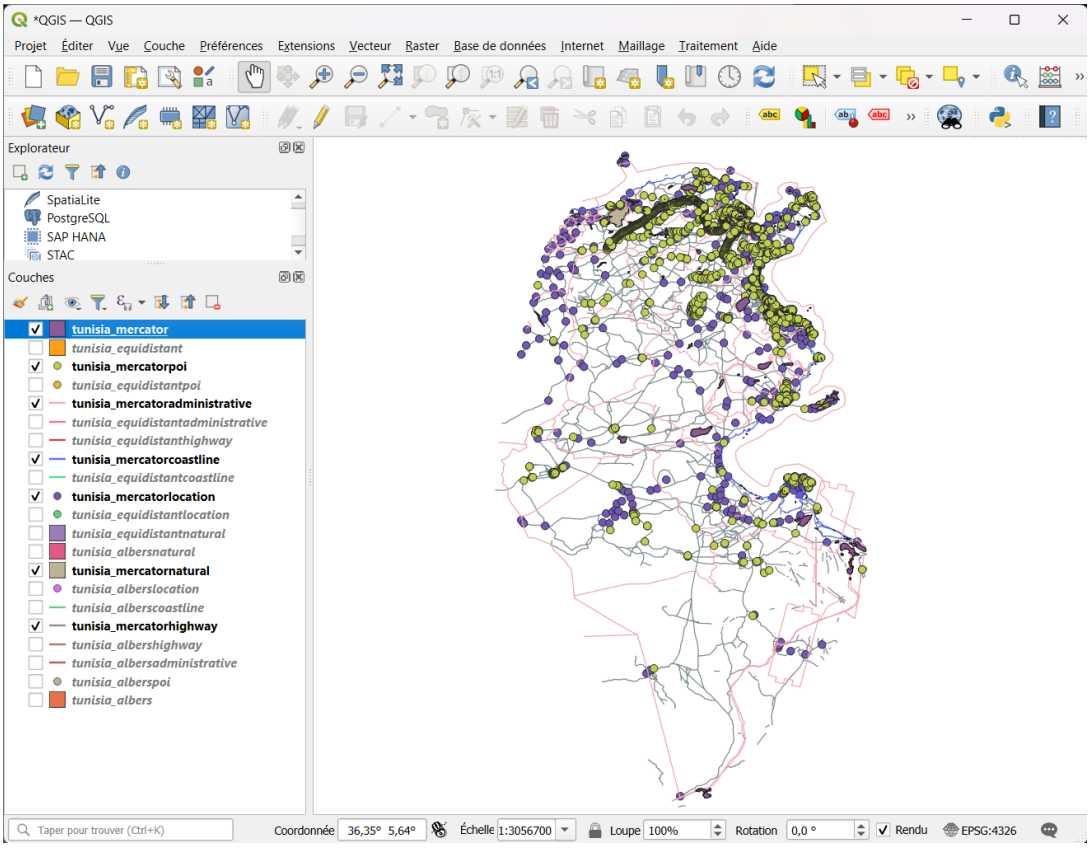

Figure 1: Tunisia layers in Mercator projection

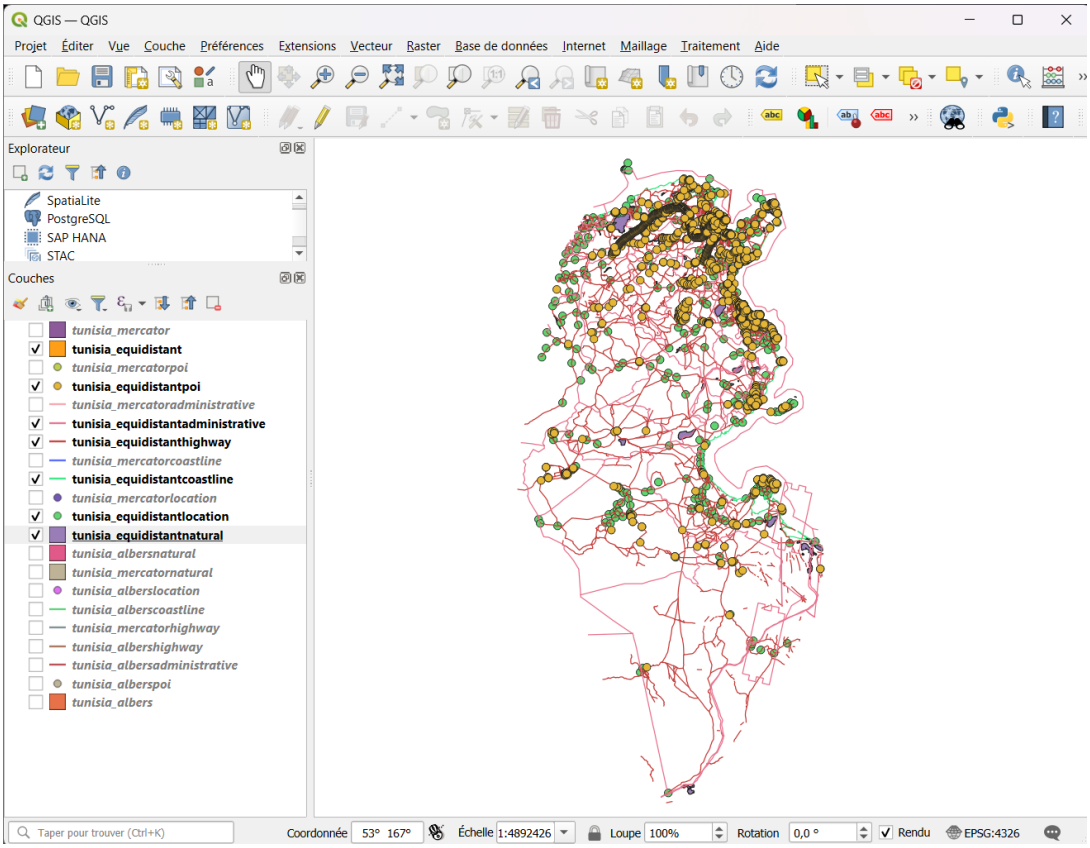

Figure 2: Tunisia layers in Equidistant Conic projection

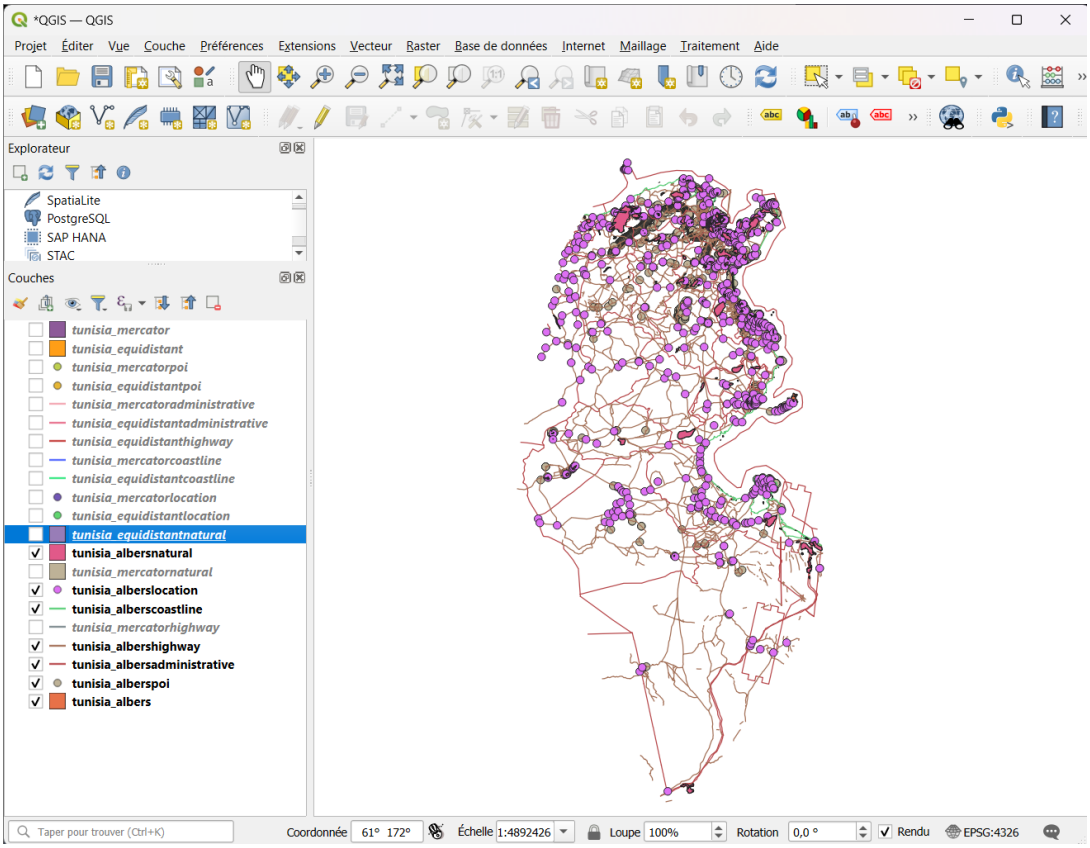

Figure 3: Tunisia layers in Albers Equal Area projection

Criterion	Mercator	Equidistant Conic	Albers Equal Area
Shape fidelity	High	Moderate	Moderate
Distance accuracy	Low	High	Moderate
Area preservation	Low	Low	High
Best for navigation	Excellent	Fair	Poor
Best for distance measurement	Poor	Excellent	Fair
Best for thematic mapping	Fair	Good	Excellent

Table 1: Performance of each projection across key map-making criteria.

5 Discussion

Choosing the right projection depends on map purpose:

- **Mercator** excels in preserving local shape and straight-line (rhumb) navigation but distorts area significantly, making it unsuitable for quantitative area analysis.
- **Equidistant Conic** provides reliable distance measurements within Tunisia's latitudinal range, favorable for logistical planning and route optimization.
- **Albers Equal Area** ensures accurate area representation, critical for resource allocation studies such as water coverage or regional land use comparisons.

6 Conclusion

By systematically reprojection and visual inspection of six diverse Tunisia layers, we illustrated how Mercator, Equidistant Conic, and Albers Equal Area projections each impact shape, distance, and area. This comparative framework supports informed projection selection tailored to specific cartographic objectives.

For an interactive demonstration, I am available to present a live GeoNotebook workflow showcasing reprojection steps and dynamic layer overlays. Feel free to reach out for a hands-on session.

A Code

```
%matplotlib inline
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature
import geopandas as gpd
from google.colab import files

# Cette ligne va te permettre de téléverser tes fichiers
uploaded = files.upload()

gdf = gpd.read_file("tunisia_natural.shp")


fig, axes = plt.subplots(1, 3, figsize=(20, 8))

#
projections = [
    ccrs.Mercator(),  # Conforme
    ccrs.EquidistantConic(central_longitude=10, central_latitude=34),  # Équidistante
    ccrs.AlbersEqualArea(central_longitude=10, central_latitude=34)  # Équivalente
]

titles = [
    "Conformal: Mercator",
    "Equidistant: Equidistant Conic",
    "Equivalent: Albers Equal Area"
]

#Maps
for i, ax in enumerate(axes):
    # Set up the projection
    ax = plt.subplot(1, 3, i+1, projection=projections[i])

    # Add basic map features
```

```
    ax.add_feature(cfeature.LAND, facecolor='lightgray')
    ax.add_feature(cfeature.OCEAN, facecolor='lightblue')
    ax.add_feature(cfeature.COASTLINE)

    # Plot your shapefile data
    gdf.plot(ax=ax, transform=ccrs.PlateCarree(),
             edgecolor='red', facecolor='none', linewidth=1)

    # Set title and adjust view
    ax.set_title(titles[i], pad=15)
    ax.set_global()  # Or use set_extent() to zoom

    # Add gridlines
    gl = ax.gridlines(draw_labels=True, linewidth=0.5, color='gray', alpha=0.5)
    gl.top_labels = False
    gl.right_labels = False

plt.tight_layout()

# Enregistrer la figure en image png
plt.savefig('tunisia_natural.png', dpi=300, bbox_inches='tight')

plt.show()

---------------

import geopandas as gpd
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
from google.colab import files


# Upload all related shapefiles (.shp, .shx, .dbf, .prj) via the upload panel
gdf = gpd.read_file("tunisia_natural.shp")

# 2. Define projections
proj_mercator = "EPSG:3395"  # Mercator (Conformal)
proj_equidist = "+proj=eqdc +lon_0=10 +lat_0=34 +lat_1=33 +lat_2=36"  # Equidistant Conic
proj_albers = "+proj=aea +lat_1=33 +lat_2=36 +lat_0=34 +lon_0=10"  # Albers Equal Area (Equivalent)

# 3. Project the GeoDataFrame
gdf_mercator = gdf.to_crs(proj_mercator)
gdf_equidist = gdf.to_crs(proj_equidist)
gdf_albers = gdf.to_crs(proj_albers)

# 4. Save projected files as GeoJSON
gdf_mercator.to_file("tunisia_mercatornatural.geojson", driver="GeoJSON")
gdf_equidist.to_file("tunisia_equidistantnatural.geojson", driver="GeoJSON")
gdf_albers.to_file("tunisia_albersnatural.geojson", driver="GeoJSON")

# 5. Download them
files.download("tunisia_mercatornatural.geojson")
files.download("tunisia_equidistantnatural.geojson")
files.download("tunisia_albersnatural.geojson")

# 6. Plot the three projections
fig, axs = plt.subplots(1, 3, figsize=(18, 6))
```

```
# Plot Mercator
axs[0].set_title("Conformal: Mercator")
gdf_mercator.plot(ax=axs[0], color='skyblue', edgecolor='black')
axs[0].axis('off')

# Plot Equidistant
axs[1].set_title("Equidistant: Equidistant Conic")
gdf_equidist.plot(ax=axs[1], color='lightgreen', edgecolor='black')
axs[1].axis('off')

# Plot Equivalent
axs[2].set_title("Equivalent: Albers Equal Area")
gdf_albers.plot(ax=axs[2], color='salmon', edgecolor='black')
axs[2].axis('off')

plt.tight_layout()
plt.show()
```

Extracting and Analyzing Building Data in Ksar Hellal, Monastir Using OSMnx

SLIM Hassen

May 28, 2025

Abstract

This report presents a geospatial workflow for retrieving and analyzing building data from OpenStreetMap (OSM) using Python libraries such as OSMnx and GeoPandas. Focusing on the town of Ksar Hellal in the Monastir Governorate of Tunisia, we construct a bounding box, extract building geometries, and explore their spatial distribution. This foundational spatial analysis provides a basis for further urban studies or cartographic representations.

1 Introduction

Urban planning, disaster management, and infrastructure development rely heavily on accurate geospatial data. With the rise of open data platforms like OpenStreetMap, tools like OSMnx offer powerful capabilities to programmatically extract and analyze such data. This project investigates the urban structure of Ksar Hellal by retrieving all available building features within a defined bounding box.

2 Data and Tools

- **Datasets**: OpenStreetMap (OSM) features tagged as `building` within specified coordinates.
- **Software & Libraries**:
  - Google Colab : Python (Jupyter Notebook)
  - `OSMnx`, `GeoPandas`, `Shapely`, `PyProj`
  - `Matplotlib` for visualization (optional)

3 Methodology

1. **Environment Setup**: Installed required packages (`osmnx`, `pyproj`) and set up the PROJ library path to avoid errors with coordinate transformations.
2. **Bounding Box Definition**: Defined geographic bounds around Ksar Hellal using:

   `[10.75, 35.75, 10.85, 35.80]`

3. **Data Extraction**: Queried OSM for all features tagged as `building=True` using `ox.features_from_polygon`
4. **Data Inspection**: Displayed the number of buildings retrieved and previewed the entries.

4 Results

4.1 Visual Observations

- The bounding box successfully captured a dense cluster of buildings corresponding to urban and peri-urban areas in Ksar Hellal.
- The dataset includes both geometry and building metadata.

4.2 Summary Table

Item	Result
Bounding box area	Approx. 0.005° x 0.05°
Number of buildings retrieved	499
Data type	GeoDataFrame
Geometry type	Polygon / MultiPolygon

5 Administrative Boundaries Visualization

The following figure displays the administrative boundaries of **Ksar Hellal** in the Monastir Governorate, extracted from OpenStreetMap using the `OSMnx` library with `admin_level = 6`.

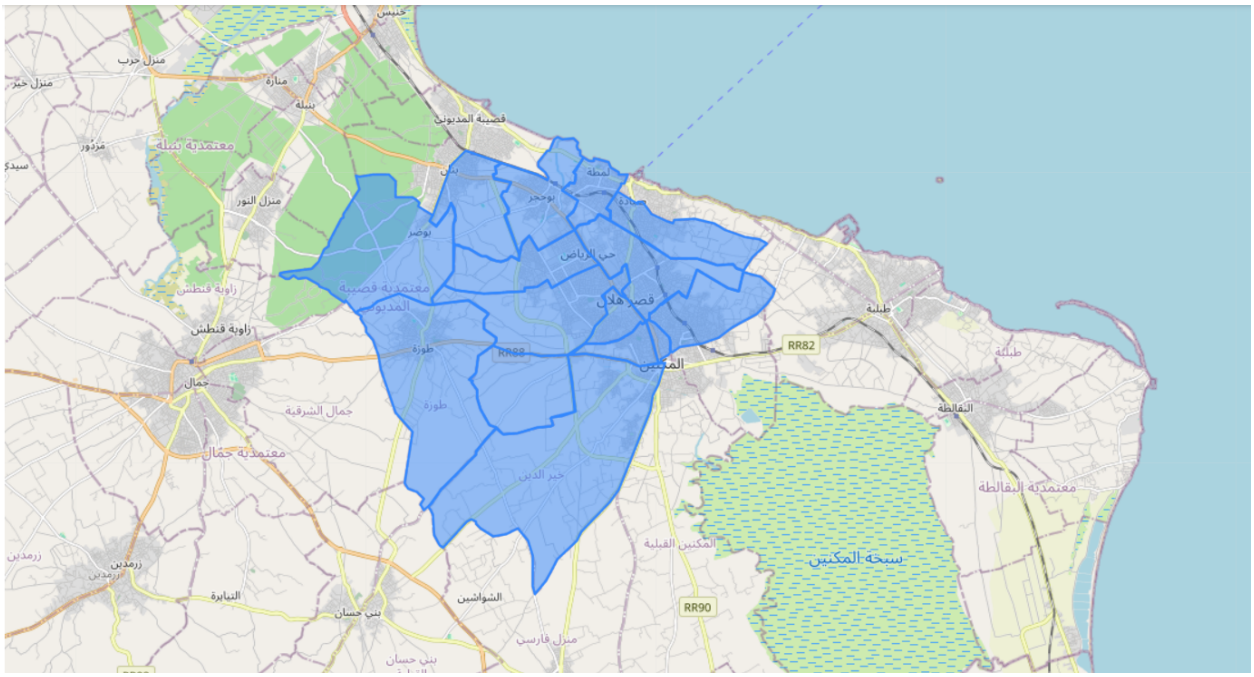

Figure 1: Administrative boundaries of Ksar Hellal (admin_level=6)

6 Discussion

The method efficiently retrieves building-level data with minimal configuration. OSMnx's bounding box and tag-based querying system makes it ideal for localized studies like this one. However, data

completeness is limited by OSM contributions in the region. For further analysis, incorporating additional tags or buffer zones could enrich urban analysis, especially for public facilities, road networks, or land use.

7 Conclusion

This study demonstrated a streamlined approach to acquiring and visualizing urban structure data using OSMnx and GeoPandas. It sets the foundation for more comprehensive geospatial analysis, such as density heatmaps, building classification, or urban growth monitoring in the Monastir region.

Appendix A: Building Extraction Code

```
!pip install osmnx

# Ensure pyproj is set up
import pyproj
print(pyproj.datadir.get_data_dir())

# Main imports
import osmnx as ox
import os
os.environ["PROJ_LIB"] = "D:/anaconda3/envs/autogis/Library/share/proj"
import geopandas as gpd
from shapely.geometry import box

# Bounding box for Monastir, Tunisia
bounds = [10.75, 35.75, 10.85, 35.80]
bbox = box(*bounds)

# Extract buildings
buildings = ox.features_from_polygon(bbox, tags={"building": True})

# Preview and count
print(buildings.head())
print(f"Number␣of␣buildings:␣{len(buildings)}")
```

Appendix B: Boundary Extraction Code

```
import osmnx as ox

# Define the area of interest
query = "Ksar␣Hellal,␣Monastir,␣Tunisia"

# Extract administrative boundaries
borders = ox.features_from_place(query, tags={"admin_level": "6"})

# Display the map interactively (in a notebook or browser)
borders.explore()
```

Street Network Analysis in Mahdia Using OSMnx and NetworkX

SLIM Hassen

May 28, 2025

Abstract

This report presents a method to extract and analyze the street network of Mahdia, Tunisia using the OSMnx and NetworkX Python libraries. By querying OpenStreetMap and generating a drivable network graph, we explore topological properties and spatial patterns. This approach offers insights useful for urban planning and transportation analysis.

1 Introduction

Street network analysis is critical for understanding urban structure, transportation planning, and accessibility. With the advent of open geospatial data and specialized libraries like OSMnx, it is now feasible to extract and analyze street networks from OpenStreetMap for any location worldwide. This report investigates the street infrastructure of Mahdia, Tunisia using a graph-based approach.

2 Data and Tools

- **Datasets**: OpenStreetMap street network data for Mahdia, Tunisia.
- **Software and Libraries**:
  - `Google Colab : Python`
  - `OSMnx`, `NetworkX`, `GeoPandas`, `Matplotlib`

3 Methodology

1. Queried OpenStreetMap for drivable streets in Mahdia using OSMnx.
2. Constructed a directed graph of the street network.
3. Extracted nodes and edges as GeoDataFrames.
4. Explored attributes such as `highway`, `length`, and `maxspeed`.
5. Visualized the street network.

4 Results

4.1 Graph Summary

- The graph contains multiple nodes and edges representing intersections and road segments.
- Key attributes include:
  - Road type (`highway`)
  - Speed limits (`maxspeed`)
  - Length
- The graph is well-formed and suitable for further analysis.

4.2 Example Edge Attributes

Sample of street segment attributes:

- `highway`: residential
- `maxspeed`: 50 km/h
- `length`: 120.3 meters

5 Shortest Route Visualization

This section presents the shortest driving route between **Ksour Essef** and **Chebba**, two towns in the Mahdia Governorate, Tunisia. The route is computed using the OSMnx library and visualized using Matplotlib.

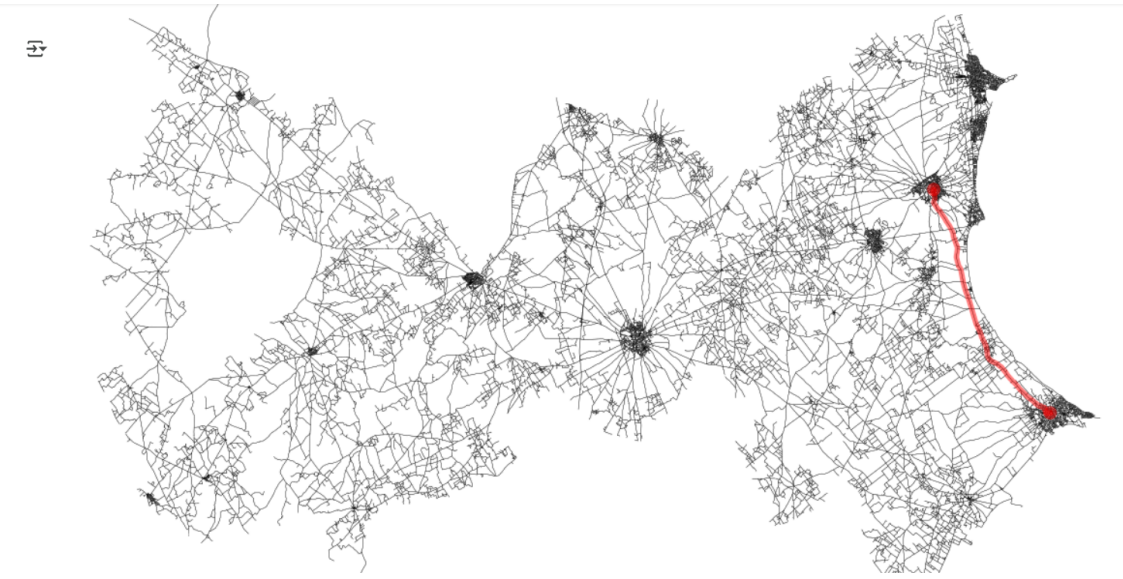

Figure 1: Shortest route from Ksour Essef to Chebba

Appendix B: Shortest Route Code

```
# Algorithme de plus court chemin (Dijkstra)

import osmnx as ox

# Adresse d'origine    Mahdia
orig_address = "Chebba, Mahdia, Tunisia"
orig_y, orig_x = ox.geocode(orig_address)

# Adresse de destination    Mahdia
dest_address = "Ksour Essef, Mahdia, Tunisia"
dest_y, dest_x = ox.geocode(dest_address)

print("Coordonn es de d part :", orig_x, orig_y)
print("Coordonn es de destination :", dest_x, dest_y)
# Find the closest nodes for origin and destination
orig_node_id, dist_to_orig = ox.distance.nearest_nodes(G, X=orig_x, Y=
    ↪ orig_y, return_dist=True)
dest_node_id, dist_to_dest = ox.distance.nearest_nodes(G, X=dest_x, Y=
    ↪ dest_y, return_dist=True)

print("Origin node-id:", orig_node_id, "and distance:", dist_to_orig, "
    ↪ meters.")
print("Destination node-id:", dest_node_id, "and distance:", dist_to_dest,
    ↪  "meters.")# Calculate the paths
metric_path = nx.dijkstra_path(G, source=orig_node_id, target=dest_node_id
    ↪ , weight='length')
time_path = nx.dijkstra_path(G, source=orig_node_id, target=dest_node_id,
    ↪ weight='travel_time_seconds')

# Get also the actual travel times (summarize)
travel_length = nx.dijkstra_path_length(G, source=orig_node_id, target=
    ↪ dest_node_id, weight='length')
travel_time = nx.dijkstra_path_length(G, source=orig_node_id, target=
    ↪ dest_node_id, weight='travel_time_seconds')metric_path == time_path
# Shortest path based on distance
fig, ax = ox.plot_graph_route(G, metric_path,
                              edge_linewidth=0.2, node_size=0, bgcolor="
                                 ↪ white", edge_color="black", figsize
                                 ↪ =(14,10))

# Print some useful information as well
print(f"Shortest path distance {travel_length: .1f} meters.")
```

6 Discussion

This street network analysis lays the groundwork for more advanced studies such as route optimization, centrality analysis, and accessibility metrics. OSMnx simplifies the acquisition of detailed network data and integrates well with other spatial libraries like GeoPandas.

7 Conclusion

Using OSMnx and NetworkX, we successfully extracted and examined the street network of Mahdia. This framework supports scalable and repeatable urban analysis across Tunisian cities and beyond.

Appendix A: Code

```
!pip install osmnx

import osmnx as ox
import geopandas as gpd
import pandas as pd
import networkx as nx

# Define area of interest
query = "Mahdia, Tunisia"

# Download graph for drivable roads
G = ox.graph_from_place(query, network_type="drive", simplify=False)

# Convert graph to GeoDataFrames
nodes, edges = ox.graph_to_gdfs(G)

# Display edge attributes
print(edges.head())
```

Lab Report
Satellite Image Classification with ENVI

SLIM Hassen

Contents

1 Introduction

This report presents the various steps of satellite image classification using the ENVI software. Through image exploration, supervised and unsupervised classification, and multiple experiments, the goal is to understand the methods used for analyzing satellite data.

2 Software Installation

The ENVI software was installed from the provided folder. The update patch was applied, and the license file was loaded.

3 Image Loading and Exploration

3.1 Opening the Image

Used file: `can_tmr.img`. The image includes the following bands:

Band	Wavelength (µm)	Resolution (m)
1 (Blue)	0.45–0.52	30
2 (Green)	0.52–0.60	30
3 (Red)	0.63–0.69	30
4 (Near Infrared)	0.76–0.90	30
5 (SWIR 1)	1.55–1.75	30
6 (Thermal)	10.40–12.50	120 (resampled to 30)
7 (SWIR 2)	2.08–2.35	30

3.2 Creating a False Color Composite

- Band assignment: R = Band 4, G = Band 3, B = Band 2 - The image highlights vegetation areas in bright red.

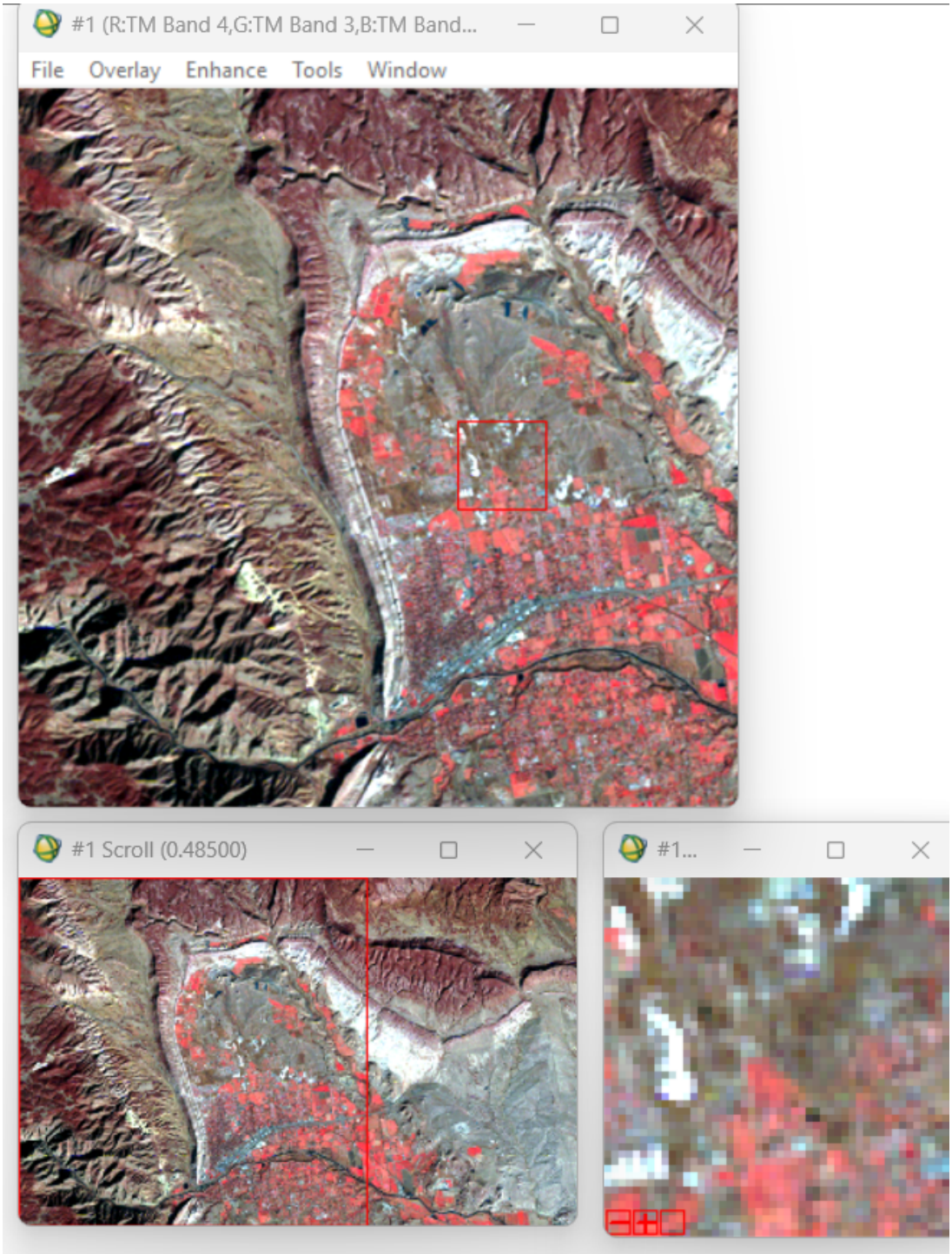

Figure 1: False Color Composite Image

3.3 Exploring Pixel Values

Tools used: `Cursor Location/Value` and `Z Profile (Spectrum)`. These allow the user to observe spectral values pixel by pixel.

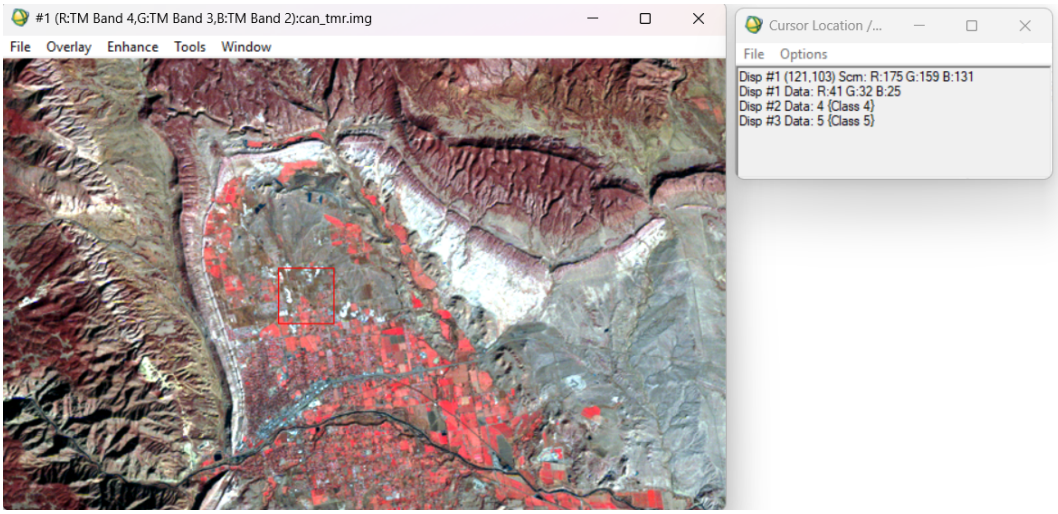

Figure 2: Cursor Location/Value Tool in ENVI

4 Unsupervised Classification

4.1 K-Means Method

- Image used: `can_tmr.img` - Default parameters accepted - Result: pixel clustering based on spectral similarity

K-Means Experiments

- Change in number of classes - Modification of thresholds (max distance, standard deviation)

Comments:

- Fewer classes simplify the classification but reduce detail.
- Larger thresholds give more flexibility but may result in mixed class regions.

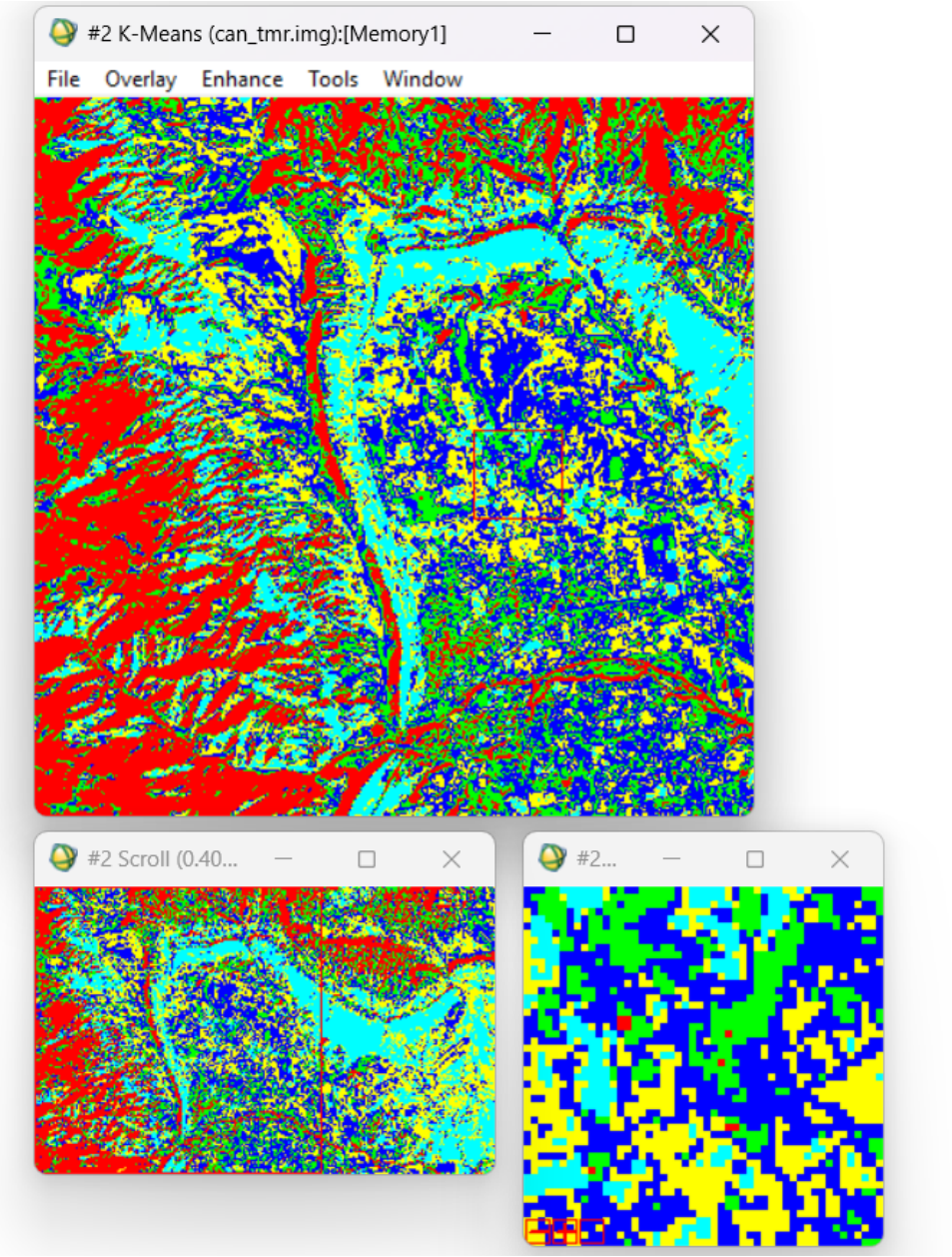

Figure 3: K-Means Classification Result

4.2 ISODATA Method

- Image used: `can_tmr.img` - Default parameters at first

ISODATA Experiments

Two experiments:

- Change in number of iterations
- Change in threshold percentage

Comments:

- More iterations allow better convergence.

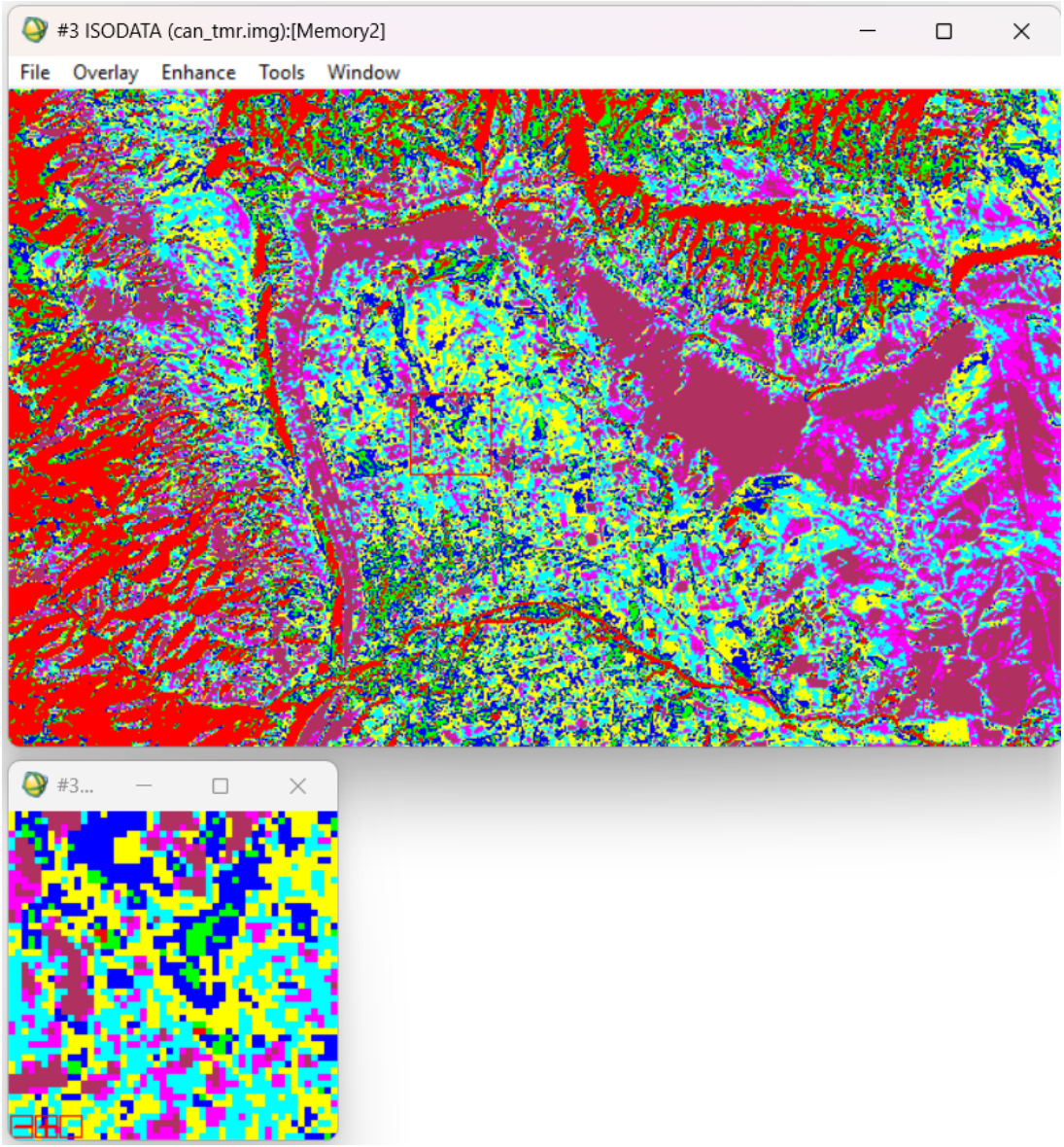

Figure 4: ISODATA Classification Result

- Too low a threshold over-segments the image.

5 Supervised Classification

5.1 Loading ROI

File: `classes.roi` – used as training data for classification.

5.2 Parallelepiped Method

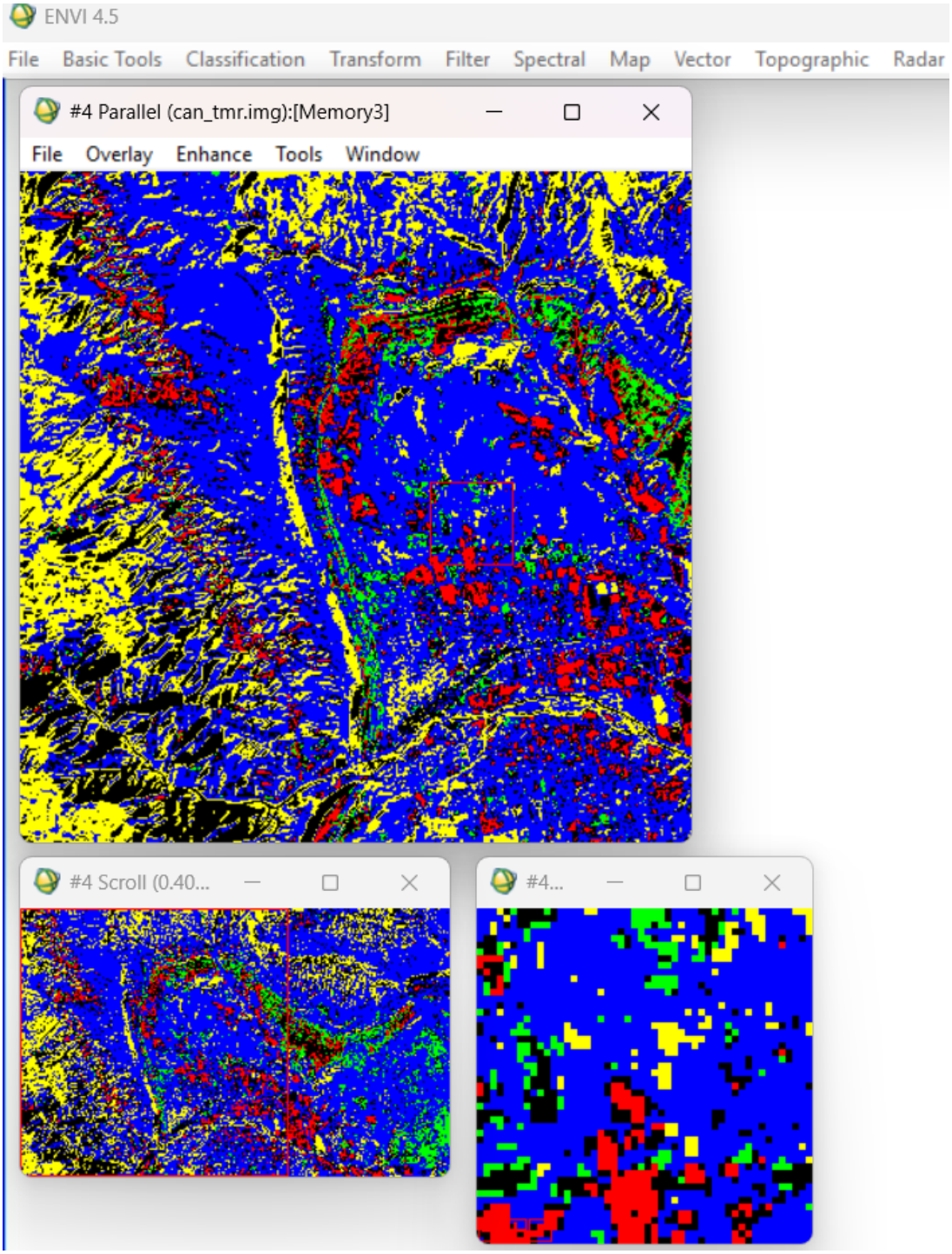

Figure 5: Parallelepiped Classification Result

5.3 Classification Comparison

Linked image display via `Tools->Link Display`. **Comments:**

- Supervised classification using ROI is more accurate than K-Means.
- ISODATA provides better spatial distribution of classes.

6 Advanced Supervised Methods (with Arabic Explanation)

6.1 Maximum Likelihood

Classification Principle (in Arabic):

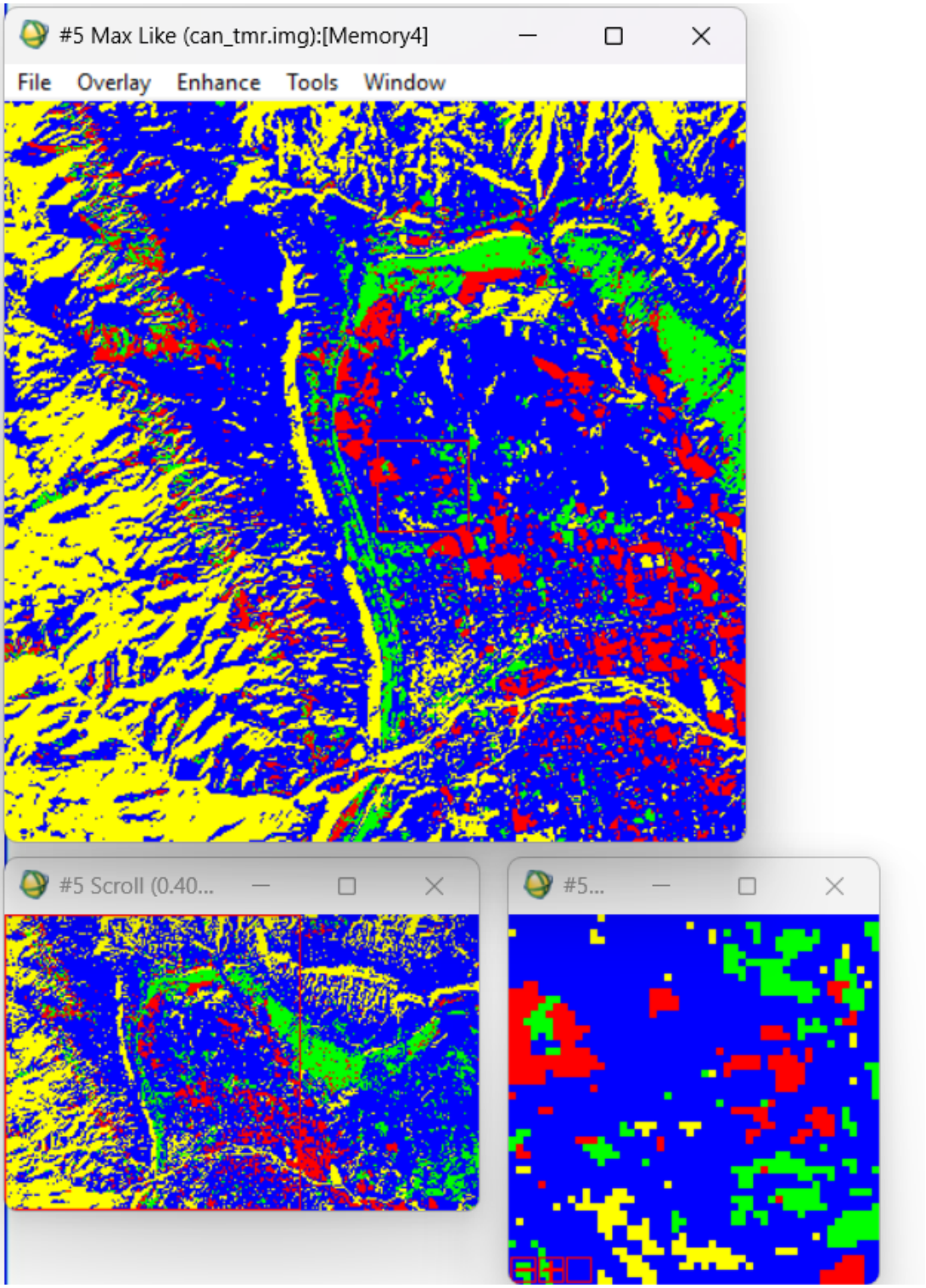

Figure 6: Maximum Likelihood Classification

6.2 Minimum Distance

Classification Principle (in Arabic):

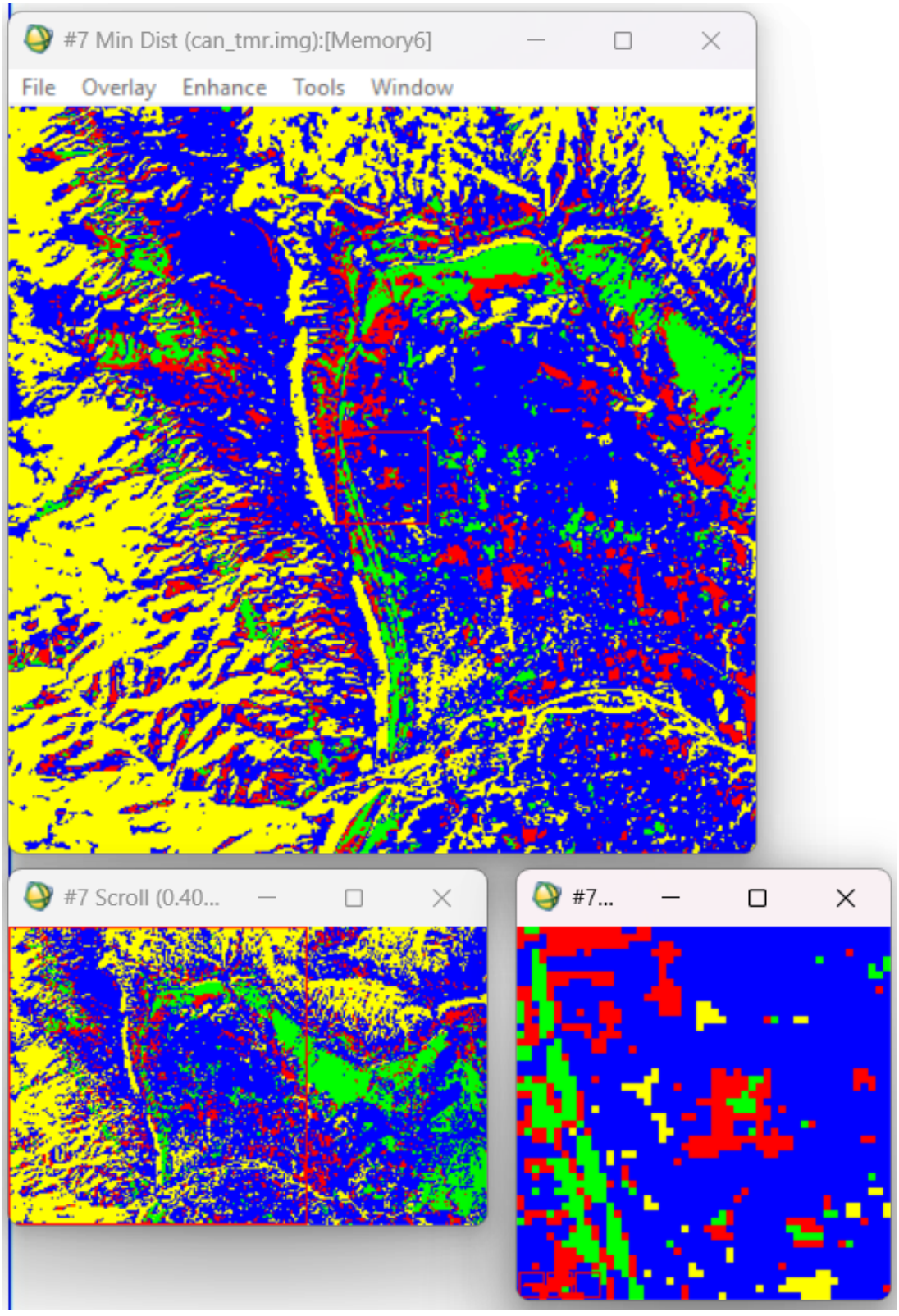

Figure 7: Minimum Distance Classification

6.3 Mahalanobis

Classification Principle (in Arabic): ” Minimum Distance.”

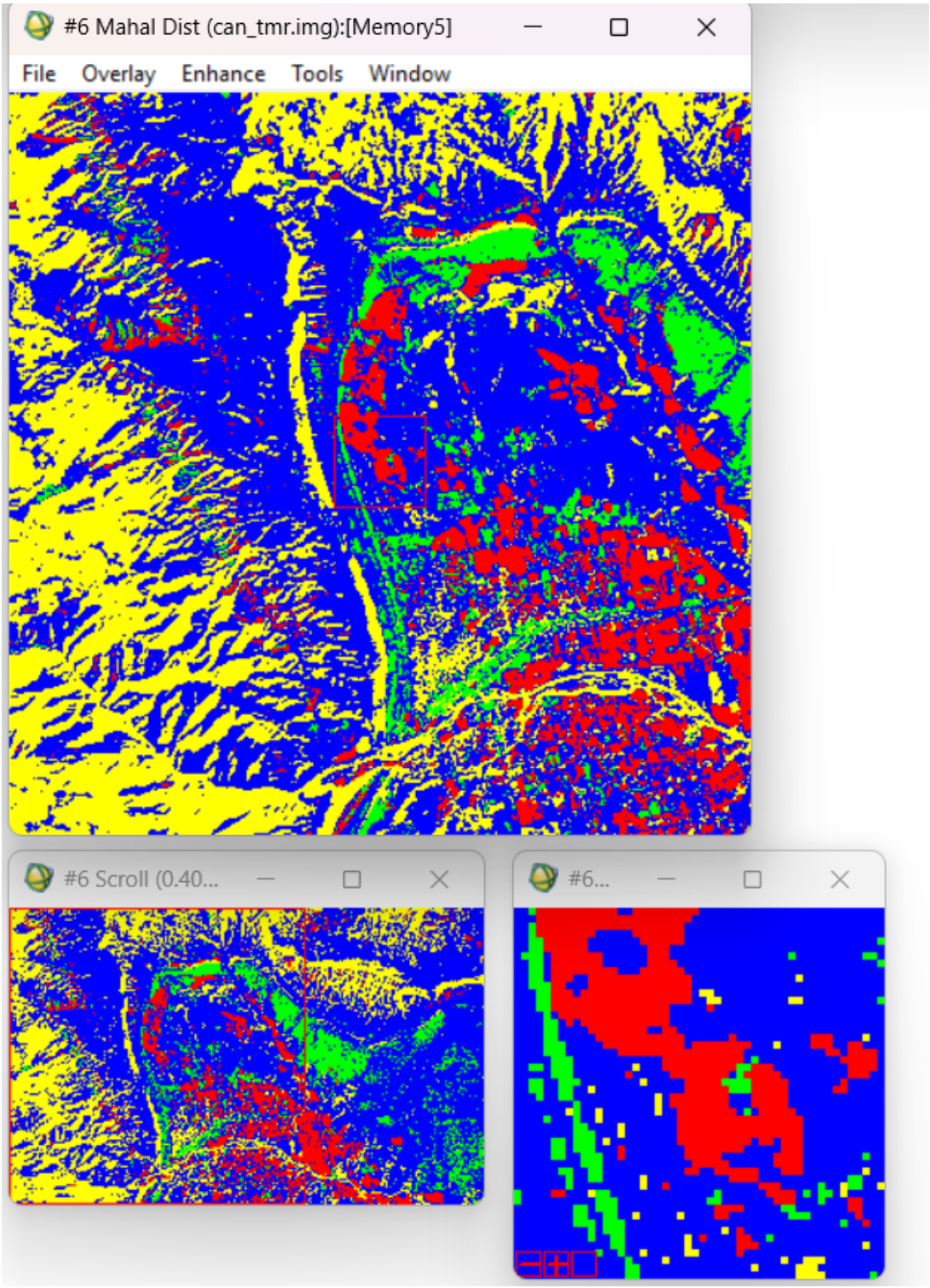

Figure 8: Mahalanobis Classification

Maximum Likelihood :
يعتمد على حساب احتمال انتماء كل بكسل إلى كل فئة، ويتم تعيين البكسل للفئة ذات الاحتمال الأعلى. يعتمد على التوزيع الغاوسي.

Minimum Distance:
يعتمد على المسافة الإقليدية بين البكسل ومركز كل فئة. يُعيَّن البكسل لأقرب فئة.

Mahalanobis:
"يأخذ بعين الاعتبار التباين داخل الفئة ويحسب المسافة بناءً على مصفوفة التغاير، وهو أكثر دقة من Minimum Distance."

Conclusion

This lab work allowed us to discover the ENVI environment, manipulate satellite images, and apply various classification techniques. Comparing the methods shows the advantage of supervised approaches when reliable ground truth is available.