applied sciences

MDPI

Article

ReconTraj4Drones: A Framework for the Reconstruction and Semantic Modeling of UAVs' Trajectories on Moving Pandas

Konstantinos Kotis * and Andreas Soularidis

Intelligent Systems Lab, Department of Cultural Technology and Communication, University of the Aegean, University Hill, 81100 Mytilene, Greece
* Correspondence: kotis@aegean.gr

Abstract: Unmanned aerial vehicles (UAVs), also known as drones, are important for several application domains, such as the military, agriculture, cultural heritage documentation, surveillance, and the delivery of goods/products/services. A drone's trajectory can be enriched with external and heterogeneous data beyond latitude, longitude, and timestamp to create its semantic trajectory, providing meaningful and contextual information on its movement data, enabling decision makers to acquire meaningful and enriched contextual information about the current situation in the field of its operation and eventually supporting simulations and predictions of high-level critical events. In this paper, we present an ontology-based, tool-supported framework for the reconstruction, modeling, and enrichment of drones' semantic trajectories. This framework extends MovingPandas, a widely used and open-source trajectory analytics and visualization tool. The presented research extends our preliminary work on drones' semantic trajectories by contributing (a) an updated methodology for the reconstruction of drones' trajectories from geo-tagged photos taken by drones during their flights in cases in which flight plans and/or real-time movement data have been lost or corrupted; (b) an enrichment of the reconstructed trajectories with external data; (c) the semantic annotation of the enriched trajectories based on a related ontology; and (d) the use of SPARQL queries to analyze and retrieve knowledge related to the flight of a drone and the field of operations (context). An evaluation of the presented framework, namely, ReconTraj4Drones, was conducted against several criteria, using real and open datasets.

Keywords: UAV; semantic trajectory; ontology; geo-tagging; MovingPandas

Citation: Kotis, K.; Soularidis, A. ReconTraj4Drones: A Framework for the Reconstruction and Semantic Modeling of UAVs' Trajectories on MovingPandas. *Appl. Sci.* **2023**, *13*, 670. https://doi.org/10.3390/app13010670

Academic Editor: Mauro Lo Brutto

Received: 20 December 2022
Revised: 29 December 2022
Accepted: 31 December 2022
Published: 3 January 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Technological advances in the field of aerospace have led to the excessive use of UAVs (drones) in a wide range of domains. UAVs are able to operate manually, autonomously [1], or semi-autonomously [2], alone or in a swarm, providing a high degree of flexibility in the assigned mission. The market for UAVs has been steadily growing [3], from the delivery of goods/services and precision agriculture to surveillance and military operations. The effective modeling and analysis of UAVs' trajectories enable decision makers to acquire meaningful and enriched information/knowledge about the current situation in the field of operations, eventually supporting tool-based automated or semi-automated simulations for making predictions of high-level critical events.

A semantic trajectory of a swarm of drones [4] is a synthesis of semantic trajectories [5] of multiple units moving (flying) in a specified formation, sharing common origin–destination points, having a common mission, enriched with semantic annotations at different levels of detail, and having one or more complementary segmentations, where each segmentation consists of a list of annotated episodes.

A drone's trajectory is a sequence of points (traces) that specify the position of the moving entity in space and time. A segment is a part of the trajectory that contains a list

of episodes. Each episode has a starting and ending timestamp, a segmentation criterion (annotation type), and an episode annotation. For example, an annotation type can be the "weather conditions", and an episode annotation can be "a storm", "heavy rain", "extremely high waves", etc.

The utilization of drones in the cultural heritage and archaeology domain has increased and plays an increasingly important role [6,7], since drones allow for an easy, quick, and low-cost solution for documentation by collecting aerial images and videos at different levels of altitude and angles. This is highly important, especially concerning the mapping, documentation, and detection of subsurface archeological sites [8–11]. The effectiveness of UAV operations requires an efficient representation of their movement data, along with additional information about the context of the mission [5]. A semantic trajectory of a swarm of drones requires the recording of raw movement data, such as the latitude, longitude, and timestamp collected from each unit, along with external heterogenous data, such as mission details, points/regions of interest (POIs/ROIs), and weather data. However, unpredicted events (e.g., unit malfunction, weather conditions, security violations) and known operational restrictions of drones can cause incorrect, invalid, or even missing movement data. Thus, there is a real need for the reconstruction of trajectories using other data recorded during a mission. In the present work, the utilization of geo-tagged photos taken by drones' carrying documentation equipment during their flights (or obtained from other sources) is proposed, since these provide exploitable metadata for the process of trajectory reconstruction.

For instance, in a potential scenario, a group of archeologists and geologists are interested in taking aerial photos of a petrified forest to document and obtain an initial mapping about the morphology of a geopark. Because the forest extends over a large and steep area, they decide to use a properly equipped drone to take photos of the landscape. The research team plan and upload the mission path in the drone's pilot software. Following the predefined path, the drone approaches the area of interest, taking photos during the planned mission (Figure 1). After landing, the research team analyzes the mission's photos and pilot data to acquire the trajectory followed by the drone, to validate the initial (planned) path. These data, along with data regarding weather conditions (ground temperature, humidity, etc.) from nearby weather stations, are necessary for the research team to obtain a general view of the area of interest. The team realizes that the pilot data for a specific part of the flight are missing due to a unit malfunction. Thus, it is not possible to reconstruct the whole trajectory of the drone to cross-validate flight details, nor to create its semantic trajectory to acquire a more comprehensive view about the area of interest and its context.

Figure 1. Drone's flight area of interest in the presented scenario, taking photos (blue pins) during a cultural heritage documentation mission (green and red pins represent the start and the end of the flight).

The evolution of systems for the real-time processing of UAV data for wireless networking and other related technologies, such as GPS and high-resolution cameras, allows for the constant and remote sharing of the movement of swarms of UAVs. These raw spatiotemporal data are the basis for the construction of UAVs' trajectories. However, analyses of these trajectories often produce poor and restricted results. To achieve a more efficient analysis of trajectories, contextual information e.g., on the environment, weather, and mission details, are required [12]. Most times, such data are heterogeneous, gathered from disparate sources, making their semantic enrichment and integration a difficult task. Ontologies have been adopted in various domains, offering a powerful tool for the semantic representation and integration of data, reducing the structural and semantic heterogeneity among the various data sources, specifying semantics in a clear, explicit, and formal way [13].

The aim of this paper is to present our latest efforts (a) to implement an ontology-based, tool-supported framework for the reconstruction of drones' trajectories using geo-tagged photos; (b) to enrich the reconstructed trajectories with external data; (c) to semantically annotate the enriched trajectories using an ontological approach; and (d) to utilize semantic queries in SPARQL to support simple analytics tasks. The goal is to implement the above modules as extensions on the well-known and widely used MovingPandas platform, delivering a novel framework, namely, the ReconTraj4Drones framework. We have evaluated the implemented solution using real datasets (photosets) collected during cultural heritage documentation missions, as well as open datasets.

ReconTraj4Drones is currently the only free and open-source trajectory management tool that supports (a) trajectory reconstruction using geo-tagged photos and (b) the annotation and analytics of the reconstructed and semantically enriched trajectories. By reviewing related works, it is concluded that these emphasize the trajectory reconstruction process based on (i) photos found in sources such as Flickr, (ii) videos, and (iii) users' posts on social media. These approaches do not provide a freely available and open-source tool for testing and evaluation. In this paper, we compare ReconTraj4Drones with two related tools, and we evaluate its performance against several criteria using four different datasets.

This paper is an extension of our recently published work [4]. This extension (more than 50%) mainly concerns (a) the implementation of the proposed framework, namely, ReconTraj4Drones, as an extension of the MovingPandas platform and (b) an evaluation of the implemented framework by conducting experiments with real datasets (inhouse and open ones).

The structure of the paper is as follows. Section 2 presents related work regarding semantic trajectories, trajectory reconstruction from geo-tagged photos, and trajectory analytics and visualization tools. Section 3 presents the implemented framework that extends the functionality of MovingPandas. Section 4 presents the datasets used for the evaluation of our solution and the produced results. Section 5 critically discusses related work and the developed solutions, comparing these based on specific requirements, the present experimental results, and future work. Finally, Section 6 concludes the paper.

2. Related Work

Santipantakis et al. [14] proposed the datAcron ontology to represent semantic trajectories at varying levels of spatiotemporal analysis and to demonstrate the procedure of data transformation via enhanced queries in SPARQL to support analytics tasks. Mobility analytics tasks are based on the volume and variety of data/information sources that need to be integrated. The proposed ontology, as a generic conceptual framework, tackles this challenging problem. The experimental results (http://www.datacron-project.eu/,

accessed 27 December 2022) in the domain of Air Traffic Management (ATM) demonstrate that the proposed ontology supports the representation of trajectories at multiple and interlinked levels of analysis.

Cai et al. [15] focused on extracting semantic trajectory patterns from geo-tagged data. They proposed a semantic trajectory pattern mining the framework from geo-tagged data taken from social media to create raw geographic trajectories. These raw trajectories are enriched with contextual/semantic annotations, using a ROI as stop to illustrate places of interest. The authors further enriched these places with multiple semantic annotations. Thus, each semantic trajectory is a sequence of ROIs with a set of multiple additional semantics. The algorithm returns basic and multi-dimensional semantic trajectory patterns.

To successfully carry out a mission, high-precision tracking of a drone's trajectory is vital. Even if a quadrotor drone has great advantages, such as a lightweight structure, vertical takeoff, and landing capability, etc., it is also vulnerable to disturbance factors, such as aerodynamic effects, gyroscopic moments, wind gusts, etc. Thus, the implementation of an efficient controller for a quadrotor UAV is a challenging task. Towards this end, various approaches have been developed based on observer techniques [16], sliding mode control (SMC) [17,18], neural networks (NN) [19], etc. However, these approaches are beyond the scope of this paper.

Concerning trajectory analytics and visualization tools, Grasier proposed a general-purpose Python library for the analysis and visualization of trajectory data called MovingPandas [20]. The proposed library composes a combination of Pandas [21] and GeoPandas [22]. In MovingPandas, the trajectory is the core object, which is modeled as a time-ordered series of geometries, stored as GeoDataFrame, and integrated with coordinate reference system information. Depending on the domain and the purpose of the analysis, a trajectory object in MovingPandas can represent its data either as point-based or as line-based. Meanwhile, the analysis process and the visualization are executed in two-dimensional space. The proposed library can be used as a stand-alone Python script, as well as within the desktop GIS application QGIS as a plugin called Trajetools.

Reyes et al. [23] proposed a software library called Yupi. The main goals of the proposed library were (a) to be abstract enough to be used across several domains and (b) to be simple enough for research with limited programming or computer vision knowledge. Yupi's modules contain special tools that allow for the creation of trajectories from video sources, artificial trajectory creation, visualization, and the statistical analysis of trajectories. To provide compatibility between the existing tools, the proposed software enables a two-way data convention among Yupi and third-party software through the Yupiwrap package. The authors validated their software and reproduced the results in selected research papers (with a few lines of code), demonstrating the effectiveness of the proposed software.

Pappalardo et al. [24] proposed a Python library called scikit-mobility to provide a unified environment for the generation, analysis, visualization, and privacy risk assessment of human mobility data. Even if the proposed library is oriented to human mobility analysis, its features can be applied to other types of mobility data. Scikit-mobility extends Pandas' library of Python for data analysis. There are two main data structures: the TrajDataFrame and FlowDataFrame, for the representation of trajectories and flows, respectively. A flow is the objects' aggregated movements between a set of locations. Both structures inherit the functionality from the DataFrame structure of Pandas, allowing for compatibility with other Python libraries and machine learning tools. The proposed library is characterized for its efficiency and ease of use.

The presented related work (tools) is discussed and compared to our presented approach in Section 4.

3. Materials and Methods

3.1. Framework and System Architecture

In the previous sections, the importance of UAV trajectory reconstruction using geo-tagged photos taken during missions (flights) was discussed. To the best of our knowledge, a free and open-source framework that implements the reconstruction, semantic modeling, and enrichment of UAV trajectories does not exist. To fill this gap, we have extended the functionality of MovingPandas, a free, open-source, and widely used trajectory analytics and visualization tool, developing a new framework called Recon-Traj4Drones. Our framework extends MovingPandas with additional modules regarding trajectory reconstruction using geo-tagged photos, trajectory enrichment with weather and POI data, trajectory segmentation based on specific criteria (altitude variations, time intervals, distance), trajectory interactive visualization using OpenStreetMap, trajectory semantic annotation using our Onto4drone ontology (https://github.com/KotisK/onto4drone, accessed on 30 December 2022), and finally, trajectory semantic analytics using SPARQL queries.

The workflow of the proposed processes (Figure 2), from geo-tagged images to semantic trajectories, does not follow a predefined sequence of modules' execution. Starting from the reconstruction of the raw trajectory from geo-tagged images and their enrichment with external data, the enriched trajectory can be segmented based on specific criteria, visualized, or semantically annotated using the Onto4drone ontology. The latter can also be executed after the Interactive Visualization and Segmentation modules. Moreover, after the Enrichment process, the stops of the UAV can be detected in the trajectory using the corresponding functionality of MovingPandas. Finally, after the Semantic Annotation module, the drone's semantic trajectory is created (in RDF), and the Semantic Analytics can be executed using SPARQL queries. Figure 2 depicts the overall workflow.

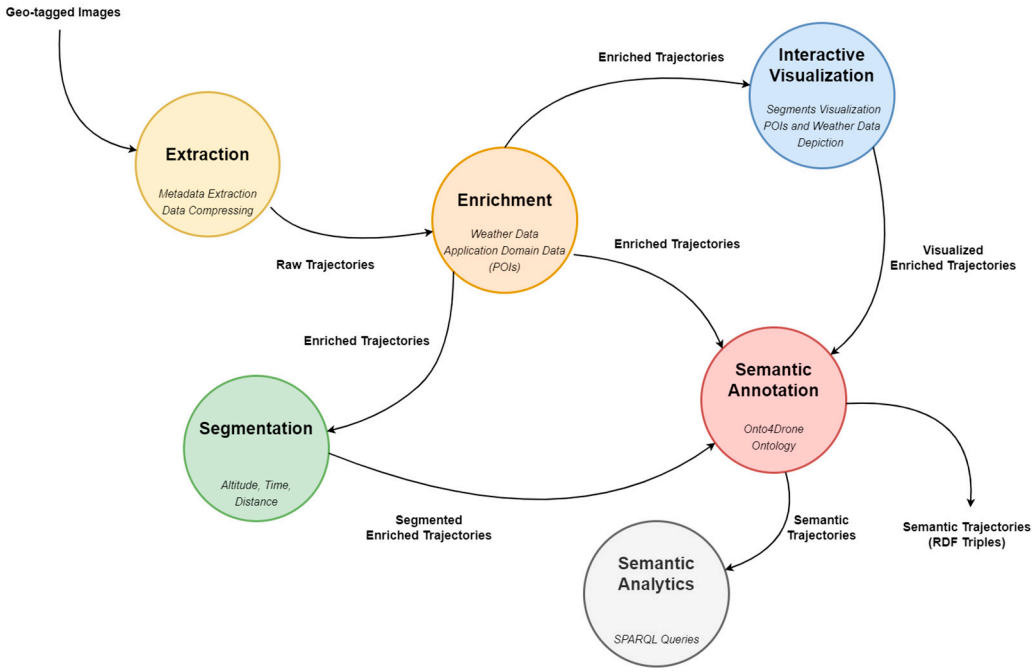


Figure 2. Workflow of the trajectory reconstruction, semantic annotation, and analytics.

The extracted metadata from geo-tagged photos are used as input in the trajectory construction module of MovingPandas to create the raw trajectory. The provided functionality of the original framework, such as visualization, stop detection, trajectory smoothing, trajectory splitting, and trajectory generalization, can be applied in the reconstructed trajectory. Then, the reconstructed raw trajectory can be enriched with external data (weather data, POIs) using the Trajectory Enrichment module. The Trajectory Segmentation module segments the enriched trajectory based on a flight's altitude variations, time intervals, or the distance between two consecutive recording points (geo-tagged

images). The Trajectory Stop Detection module extends the corresponding functionality of MovingPandas, not only to detect stops in a trajectory, but also to detect the number of recording points (geo-tagged photos) included in each stop. The Interactive Trajectory Visualization module is used to interactively visualize not only the drone's trajectory in a simple line, but also to depict additional information that the trajectory contains. Such information includes starting and ending points, segments, stops, weather, and POI data. The Semantic Annotation module is used for the annotation of the enriched trajectories using the Onto4drone ontology to ensure a high-level abstraction of the represented data, creating the drone's semantic trajectory, which can eventually be exported in RDF. The Semantic Trajectory Analytics module is used for the analytics of the reconstructed semantic trajectories using SPARQL queries. Figure 3 depicts the high-level architectural design of the modules of ReconTraj4Drones, integrated in MovingPandas. Geo-tagged photos, external data (weather data, POIs), and the Onto4drone ontology constitute the input data of the framework. On the other hand, CSV files, both for raw and enriched movement data, RDF triples of the reconstructed semantic trajectory, along with interactive visualization, trajectory segmentation, trajectory stops, and trajectory analytics results, constitute the output data of the extended framework.

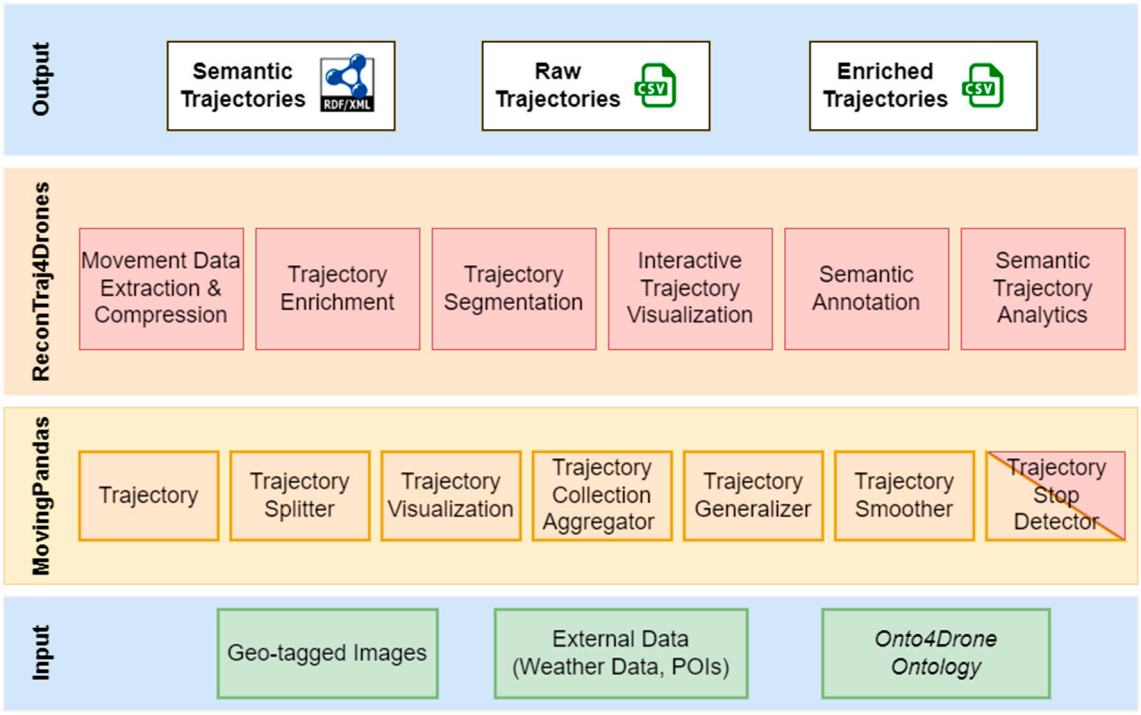


Figure 3. High-level architectural design of ReconTraj4Drones framework.

3.2. Trajectory Reconstruction

To create the trajectory of a moving object, it is necessary to have access to the movement data of that object. Typically, the movement data is related to GPS points that are recorded during the movement of the object. In our solution, we reconstructed raw trajectories using geo-tagged photos taken by drones during their mission (flight), as they provide us with suitable data/metadata for the reconstruction process. The trajectory reconstruction process consists of two parts. In the first part, the related metadata is extracted from geo-tagged photos, and in the second part, the photos are resized to make them easier to manage in later stages of analysis and visualization.

The ReconTraj4Drones framework uses a set of geo-tagged photos as input. Each photo is characterized as a (recording) point of the trajectory, providing information about the latitude, longitude, and timestamp of that point (position). Subsequently, putting these photos in chronological order, the drone's trajectory can be reconstructed. The number and

the time interval between these photos define the accuracy of the reconstructed trajectory. Typically, a geo-tagged photo not only provides spatiotemporal data (latitude, longitude, timestamp), but may also provide further valuable information about the altitude of the drone, the maker and the model of the carried camera, its focal length, etc. To acquire the drone's movement data, we extracted the spatiotemporal data that are necessary for the reconstruction of raw trajectories. We also extracted data about the altitude at each point (photo), the file itself, such as its title, storage path, size, and format, and finally, the data about the drone's camera, such as the maker and the model. The latter gave us more information about the flight and were used in the next stages of trajectory analytics and visualization. Because MovingPandas handles trajectories as DataFrames, these were stored as a DataFrame as well. Figure 4 depicts the form of a DataFrame.

	t	geometry	alt	trajectory_id	title	storage_path	size	format	camera_model
0	2022-01-31 12:42:20	POINT (25.88159 39.22564)	309.241	1	DJI_20220131124220_0001.JPG	C:\Users\andre\Andreas\Ευφυή Συστήματα Πληροφο...	20.96	JPG	DJI_ZenmuseP1
1	2022-01-31 12:42:22	POINT (25.88154 39.22561)	309.280	1	DJI_20220131124222_0002.JPG	C:\Users\andre\Andreas\Ευφυή Συστήματα Πληροφο...	20.88	JPG	DJI_ZenmuseP1
2	2022-01-31 12:42:25	POINT (25.88147 39.22558)	309.274	1	DJI_20220131124225_0003.JPG	C:\Users\andre\Andreas\Ευφυή Συστήματα Πληροφο...	22.24	JPG	DJI_ZenmuseP1
3	2022-01-31 12:42:27	POINT (25.88140 39.22555)	309.272	1	DJI_20220131124227_0004.JPG	C:\Users\andre\Andreas\Ευφυή Συστήματα Πληροφο...	20.65	JPG	DJI_ZenmuseP1
4	2022-01-31 12:42:30	POINT (25.88133 39.22552)	309.248	1	DJI_20220131124230_0005.JPG	C:\Users\andre\Andreas\Ευφυή Συστήματα Πληροφο...	20.75	JPG	DJI_ZenmuseP1

Figure 4. The form of a DataFrame after the trajectory reconstruction process.

Part of the reconstruction phase concerns the compression of the input data (photos). Typically, drones are equipped with high-resolution cameras that take high-quality photos during their flights. As a result, the size of the photos is quite large, making them difficult to use in the trajectory visualization phase. Moreover, the use of such large photos makes the visualization unreadable for the end users. To solve this problem, we resized the input photos to dimensions equal to 250 px × 250 px. Algorithm 1 outlines the logic of the trajectory reconstruction module.

Having extracted the necessary movement data from the photos, it is possible to create the raw trajectory using the functionality provided by MovingPandas. Then, it is possible to use the additional functionalities of MovingPandas, such as visualization, stop detection, trajectory smoother, etc. Last, but not least, it is possible to export the extracted metadata in the form of a CSV file. Thus, taking advantage of the corresponding functionality of MovingPandas, in the future, we can create the same trajectory using the CSV file without repeating the whole reconstruction process.

Algorithm 1. Trajectory Reconstruction

Input: a set (folder) of geo-tagged images and a selection of data exported to csv
Output: a (raw) trajectory and a csv file

```
traj_data = []
images = get_images(folder_name)
for each image ∈ images do
    point = extract the metadata of the image
    transform the coordinates in decimal degree format
    append the point into traj_data;
end for
resize_images(images)
if extract_to_csv == True then
    create_csv(traj_data)
```

11: **end if**
12: *df* = create a *DataFrame(traj_data)* using Pandas
13: *geo_df* = create a *GeoDataFrame(df)* using geoPandas
14: *trajectory* = create a *Trajectory(geo_df, 1)* using MovingPandas
15: **return** *trajectory*

3.3. Trajectory Enrichment

A raw trajectory cannot provide detailed information about the field of operation the drone has flown. In contrast, an enriched trajectory can provide further (contextual) information about the mission and the visited ROI/POIs. A trajectory can be enriched with external (to flight) data, beyond the latitude, longitude, and timestamp. This data depends on the mission and on the domain. In the ReconTraj4Drones framework, a module for the enrichment of the reconstructed trajectory is developed, acquiring data about the weather conditions and POIs of a particular flight. Thus, for each point (latitude, longitude, timestamp) of the reconstructed trajectory, the aim is to retrieve the weather conditions (temperature, humidity, pressure, etc.) present at the specified time (timestamp). Moreover, the geographic region (boundaries) in which the drone has flown is computed. Towards this aim, data about the particular area that include POIs, roads, buildings, etc. are retrieved. Finally, the data extracted from the geo-tagged images beyond the latitude, longitude, and timestamp are used to further enrich the reconstructed trajectories. Such data include altitude and camera details.

In the trajectory enrichment module, the main goal is to interrelate each point of the reconstructed trajectory with the weather conditions present at that point. Weather conditions concern data such as the dew point, temperature, pressure, wind direction, and wind speed. In our approach, since the reconstruction of a drone's trajectory does not take place in real time (i.e., during a flight), the approach utilizes historical weather data. The acquisition of this data is carried out using suitable Web services, namely, the OpenWeather (https://openweathermap.org/api, accessed on 30 December 2022) and the Meteostat (https://meteostat.net/en/, accessed on 30 December 2022) services. The developed framework fetches the historical weather data by implementing the corresponding application programming interfaces (APIs). OpenWeather provides historical weather data for a given latitude, longitude, and timestamp. For each point of the reconstructed drone's trajectory (the latitude, longitude, and timestamp), the historical weather data are fetched. The data are returned in the JSON format in the following form:


```
{
   'lat': 39.2256,
   'lon': 25.8815,
   'timezone': 'Europe/Athens',
   'timezone_offset': 7200,
   'data': [
      {
         'dt': 1643632920,
         'sunrise': 1643606671,
         'sunset': 1643643307,
         'temp': 12.16,
         'feels_like': 10.68,
         'pressure': 1012,
         'humidity': 48,
         'dew_point': 1.49,
         'clouds': 50,
         'wind_speed': 2.69,
         'wind_deg': 218,
```

```
      'weather': [
        {
          'id': 802,
          'main': 'Clouds',
          'description': scattered clouds',
          'icon': '03d'
        }
      ]
    }
  ]
}
```


The Meteostat Web service provides historical weather data for a given point as well. However, unlike OpenWeather, it takes the altitude as input (in addition to latitude, longitude, timestamp) and returns the historical weather data in a pandas DataFrame form. Consequently, the framework can acquire historic weather data for a trajectory regarding either the ground or the drone's flight altitude. In the latter case, if there are no historic weather data available for the given altitude, the historical weather data of the ground are returned. Figure 5 depicts the returned data from the Meteostat Web service for a given latitude, longitude, timestamp, and altitude.

```
2021-11-07 17:47:05
                     temp  dwpt  rhum  prcp  snow   wdir  wspd  wpgt    pres  tsun  coco
time
2021-11-07 17:00:00  14.2  13.1  93.0   0.0   NaN  352.0   9.0   NaN  1019.0   NaN   2.0
<class 'pandas.core.frame.DataFrame'>
```

Figure 5. Returned data from Meteostat for a given latitude, longitude, timestamp, and altitude.

To further enrich the trajectories with information related to a ROI/POI, the OpenStreetMap (https://www.openstreetmap.org/, accessed on 30 December 2022) was used. More specifically, ReconTraj4Drones acquires information about the name, address, type, place id, and the bounding box. Thus, for each point of the reconstructed trajectory, the spatial data (latitude, longitude) are provided as input, and the abovementioned information is retrieved as output. The choice of OpenStreetMap is based on the fact that it is a free open-source Web service that meets the requirements of the ReconTraj4Drones framework. Moreover, the returned data are provided in JSON format, facilitating easy management and reuse. The following is an example of the returned data for a given point.


```
{
  "place_id": 188523385,
  "licence": "Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright",
  "osm_type": "way",
  "osm_id": 366465550,
  "lat": "39.22527421360034",
  "lon": "25.88153577687583",
  "display_name": "ΕΠ19, Municipality of Western Lesvos, Lesbos Regional Unit, Northern Aegean, Aegean, 81103, Greece",
  "address":
    {
      "road": "ΕΠ19",
      "municipality": "Municipality of Western Lesvos",
      "county": "Lesbos Regional Unit",
```

```
        "state_district": "Northern Aegean",
        "state": "Aegean",
        "postcode": "81103",
        "country": "Greece",
        "country_code": "gr"
      },
      "boundingbox": ["39.218668", "39.2343256", "25.8715699", "25.9620231"]
}
```


3.4. Trajectory Segmentation

Trajectory segmentation is the process of dividing the trajectory into a number of parts (segments), such that each part satisfies a given criterion (discussed below). In ReconTraj4Drones, we provide three different ways to segment a trajectory into parts. As the reconstructed trajectory consists of a set of recording points, where each of these is a geo-tagged photo, the trajectory is split/segmented based on the time interval between two consecutive geo-tagged photos. The time interval is defined by the user, and the module splits the reconstructed trajectory into parts whenever the time interval between two consecutive photos (recording points) exceeds a given threshold.

Similarly, the module also considers, as an additional splitting criterion, the distance between two consecutive geo-tagged photos. The haversine distance between every two consecutive points is calculated. Whenever this distance is greater than a given threshold, the trajectory is segmented. As in the first case, the threshold is defined by the users according to the domain and their specific needs.

In addition to the time interval and haversine distance segmentation criteria, a third case is the segmentation of the reconstructed trajectory based on the criterion of altitude variations of a drone during its flight. The drone's flight altitude is recorded in geo-tagged photos, and this information is extracted during the trajectory reconstruction phase. Thus, the altitude variations are computed between every two consecutive photos. Whenever these variations exceed the threshold (provided by users), the trajectory is segmented.

In all three above cases (segmentation criteria), the produced trajectory parts are returned as a set of (i, j), where i is the i-th (recording) point defining the beginning of that segment, and j is the j-th (recording) point defining the end of that segment. In cases in which the segmentation criterion is not satisfied for any point on the trajectory, the framework returns the entire trajectory as one segment, and the points (i, j) correspond to the start and end points of the trajectory. Finally, if the trajectory segmentation process is executed before the interactive visualization process, then the visualization of these segments is available.

3.5. Trajectory Stop Detection

Stop detection constitutes one of the basic functionalities in a trajectory analysis pipeline. Taking advantage of the corresponding functionality provided by MovingPandas, ReconTraj4Drones can efficiently detect stops in a given enriched trajectory. A stop is detected if a drone stays within a specific area for a specific time duration. Both parameters (area and time) are set by users (duration in seconds and max diameter in meters). Based on this functionality, ReconTraj4Drones computes the number of recording points included in each stop. Finally, the framework returns details about each detected stop, such as its ID and coordinates, the start and end time, the duration in seconds, and the number of recording points within it. The generated data can be visualized if the stop detector module is executed before the visualization process.

3.6. Trajectory Visualization

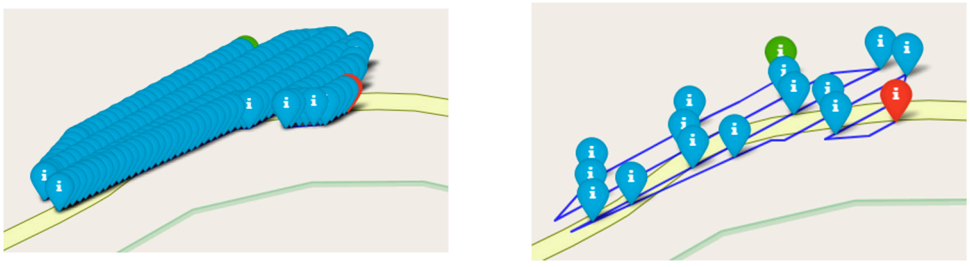
Visualization is of high importance in the management of trajectories. Even though MovingPandas provides a functionality for trajectory visualization, ReconTraj4Drones

implements an additional, self-contained module for this purpose. The aim is to visualize the reconstructed trajectory with all the additional information that is used for its enrichment and its segmentation. As already stated, the additional information concerns data beyond latitude, longitude, and timestamp that uniquely define each point of the trajectory. Such data concern the drone's mission and flight details, the type of drone, the flight altitude, the camera model and maker, and finally, the geo-tagged image itself that is related to a particular point. Moreover, if the enrichment of a trajectory has been carried out using weather and/or POI data, then this data can also be visualized. Subsequently, the aim is to visualize not only the raw trajectory, but also all the external integrated information that is obtained for each point of this trajectory.

To implement the abovementioned requirement, ReconTraj4Drones uses the Folium library (https://python-visualization.github.io/folium/, accessed on 30 December 2022), an open-source and free library that utilizes the OpenStreetMap to visualize trajectories. This library also enables the creation of interactive visualization maps. Interactive in this context means that not only is each point of the trajectory displayed with a special symbol (pin), but also that these symbols are clickable, hiding all the information available for that point. In addition, the starting and ending point of the trajectory are assigned with special symbols of different colors, i.e., green for the starting point and red for the ending point.

Having said that, because the reconstruction of the trajectory is based on the number of geo-tagged photos and the time interval between them, it is observed that, in cases of short time intervals between geo-tagged photos, the number of points is very large. Consequently, visually capturing all these points makes the trajectory unreadable (Figure 6a). To solve this problem, a percentage of the total amount of points is visualized only. This percentage is defined by the user (Figure 6b).

Last, but not least, in ReconTraj4Drones, it is possible to visualize the different trajectory segments, as well as the detected stops, if the trajectory segmentation and/or trajectory stop detector modules have already been executed. In the first case, each trajectory segment is visualized with a unique color to distinguish between them. In addition, the starting and the ending point of each segment are indicated with green and red color pins, respectively. On the other hand, if the segmentation process has not been executed, then the entire trajectory is defined as one segment and is visualized with the same color. Regarding stops' visualization, each trajectory stop is visualized as a red circle. Each circle is a clickable item displaying all the information about that stop. The user determines whether segments and/or stops will be displayed, even if both modules have already been executed.



(**a**) (**b**)

Figure 6. Trajectory interactive visualization. (**a**) Visualization of the total number of points that make up the trajectory; (**b**) visualization of a percentage of points set by the user.

3.7. Trajectory Semantic Annotation

After the execution of the reconstruction and the enrichment modules, the resulting trajectory integrates external heterogenous information beyond latitude, longitude, and timestamp. Thus, the need arises for a high-level abstract formalism that will semantically annotate the seamlessly integrated data and expose it as linked data (for third-party services' consumption). Having said that, the use of the developed ontology is proposed for

the actual semantic data integration, i.e., for seamlessly mapping the heterogeneous internal and external data to shared, explicit, and formal semantics.

Towards this aim, ReconTraj4Drones integrates a drone-related ontology, namely, Onto4drone, which was recently developed in our laboratory in the context of related research. Figure 7 depicts a high-level design of the core semantics of the developed ontology. Currently, the ontology version is 1.0.2, and it is available online in OWL (https://github.com/KotisK/onto4drone, accessed on 30 December 2022). The Onto4drone ontology is based directly on datAcron [25] ontology and indirectly on DUL [26], SKOS [27], SOSA/SSN [28], SF [29], GML [30], and GeoSparql [31] ontologies. The ontological approach ensures the utilization of a high-level abstract formalism for the representation of semantic trajectories, as various disparate and heterogenous data are seamlessly integrated to semantically enrich the reconstructed trajectory. A detailed description of the ontology is out of the scope of this paper.

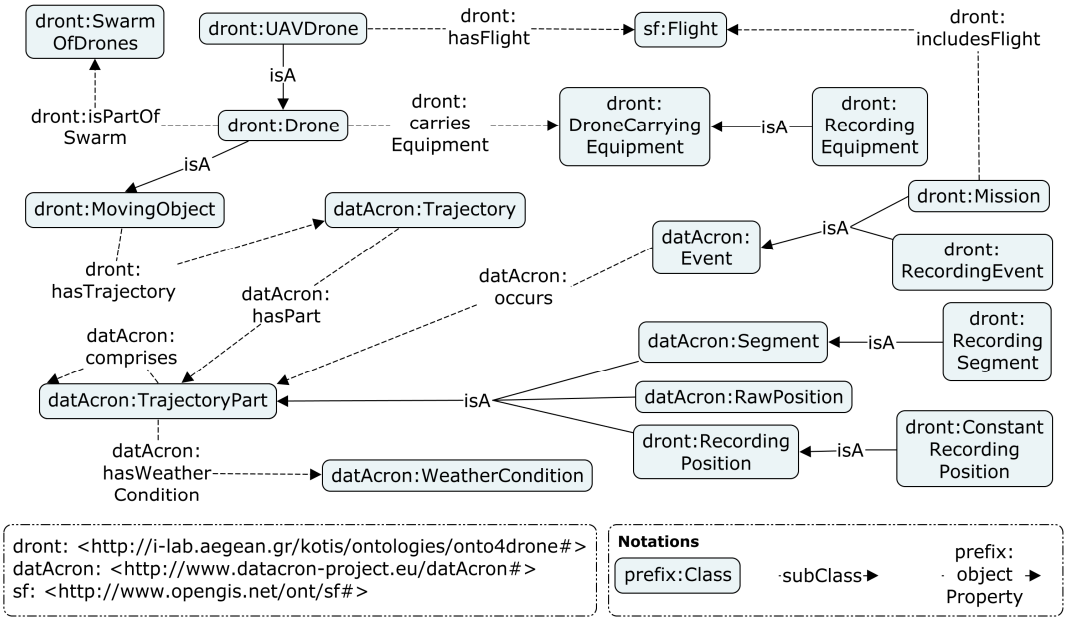


Figure 7. Basic concepts and relations of the Onto4drone ontology.

To incorporate ontologies and work with RDF models [32] in MovingPandas, the Owlready2 library [33] was used to allow for transparent access to OWL ontologies. Owlready2 is a software package for ontology-oriented programming in Python. It allows for the use of ontologies as Python objects, modifying and saving them, performing reasoning via the HermiT reasoner [34], and executing queries in the SPARQL language [35]. In the semantic annotation phase, the aim is to use the ontology for the semantic annotation of the heterogenous data that the trajectory integrates, allowing (eventually) for the analysis of the reconstructed semantic trajectories via reasoning and queries in SPARQL. Finally, Owlready2 is used to export semantically annotated data in RDF for further semantic analysis in third-party semantic applications that are able to consume semantically annotated spatiotemporal movement data.

The first step is to load the Onto4drone ontology into the ReconTraj4Drones. The framework is expected to import all the referenced ontologies. The current version of Owlready2 can load ontologies in RDF/XML, OWL/XML, and N-triples syntax/encoding. After importing Onto4drone and the associated ontologies (e.g., datAcron), the individual entities (instances of ontological classes) are created from the data that represent the trajectory. As already stated, such data are either obtained from the external Web services, such as the weather conditions and POIs; or they are extracted from the geo-tagged photos, such as the camera model and maker; or they are inserted by the user, such as the drone type, mission, and/or flight details; or they are even created during the

segmentation process (different segments). For each entity, the corresponding individual is created by specifying its label, its resource identifier (IRI), and a set of data/object properties, according to the data collected and the available ontology. The user can then proceed to the semantic analysis of the semantically enriched trajectory. Finally, the data of the semantically annotated trajectory can be exported either in RDF/XML or in N-triples encoding.

3.8. Semantic Trajectory Analytics

The Semantic Trajectory analytics module provides an analysis of the trajectories resulting from the Semantic Annotation module. To support this functionality in ReconTraj4Drones, the Owlready2 library was used. ReconTraj4Drones integrates the native SPARQL engine instead of the RDFlib (https://rdflib.readthedocs.io/en/stable/, accessed on 30 December 2022). The first is used since it is 60 times faster than the latter. Moreover, it automatically creates prefixes from the last part of the ontology's IRI. For example, the ontology (http://myontology.org/onto.owl) will be automatically associated with the "onto:" prefix. The following prefixes are automatically pre-defined, so there is no need for them to be defined by the user:

- rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
- rdfs: http://www.w3.org/2000/01/rdf-schema#
- owl: http://www.w3.org/2002/07/owl#
- xsd: http://www.w3.org/2001/XMLSchema#
- obo: http://purl.obolibrary.org/obo/

The execution of SPARQL queries is feasible only after the execution of the semantic annotation process. Indicative examples of SPARQL queries concern the number of recording points and the number of POIs in a single trajectory. The corresponding queries are the following:

Query 1:

```
list(default_world.sparql("""
      SELECT (COUNT(?recording_points) AS ?rp) WHERE {
                  ?tr Onto4drone_v1.0.1:encloses ?rs .
                  ?rs TopDatAcronOnto_SSN_FM:comprises ?recording_points .
      }""")[0][0].
```

Query 2:

```
list(default_world.sparql("""
      SELECT ?point_of_interest WHERE {
                  ?tr Onto4drone_v1.0.1:encloses ?rs .
                  ?rs TopDatAcronOnto_SSN_FM:comprises ?rp .
                  ?rp Onto4drone_v1.0.1:hasOccurredEvent ?pse .
                  ?pse Onto4drone_v1.0.1:records ?point_of_interest .
      } """)).
```

The *sparql()* method is used to execute a SPARQL query and obtain the returned results. Because this method returns a generator, the *list()* function is used to present the results. Finally, even if the implemented framework contains a couple of predefined SPARQL queries, users can execute their own in the Jupyter notebook interface.

4. Results

4.1. Datasets

The goal is to test and evaluate the ReconTraj4Drones framework for its efficiency (time and space complexity) and scalability, using real-world datasets. Towards this direction, real datasets, both inhouse and open ones, were used. The inhouse datasets were acquired during documentation missions/flights, using two types of drones: (a) Phantom 4 Pro and (b) Inspire. The flights took place in different places (presented in detail below) and on different dates, producing differently sized datasets of geo-tagged images. The open dataset is from Konya University and was acquired using a Phantom 4 Pro drone. ReconTraj4Drones was tested and evaluated using datasets in various sizes. Thus, small datasets (less than 50 photos), medium-range datasets (up to 250 photos), and large datasets (more than 250 photos) were used. Finally, as ReconTraj4Drones reconstructs drones' trajectories using geo-tagged photos, other data regarding the flights, such as flight plans and log files, were omitted.

4.1.1. Petrified Forest

The petrified forest of Lesvos is located on the western part of Lesbos Island and was formed from the fossilized remains of trees and plants. It constitutes an area of high geological importance, and thus, its mapping at frequent time intervals is of high importance for conservation and protection purposes. Thus, a documentation mission was organized using the Phantom 4 Pro drone. The flight took place on 31 January 2022 in the petrified forest area of Lesvos Geopark, and the drone flew at an altitude of 25m above the sea level. The produced dataset consists of 208 files (geo-tagged photos) of 4.8 GB.

4.1.2. Vrisa Village

Vrisa is a village in the southern part of Lesbos Island, a traditional settlement protected by the Greek government. On 12 June 2017, the village was hit by an earthquake of 6.3 mw, significantly affecting the settlements of the village. Many buildings collapsed, and others were heavily damaged. Three years later, a mission was organized to record the condition of the village. The mission took place on 17 May 2020, using a Phantom 4 Pro drone. The produced dataset consists of 431 files (geo-tagged photos) of 2.91 GB.

4.1.3. University Hill

Two flights over the buildings of the University of the Aegean in Mytilene, Lesvos were performed. Both flights took place on 8 December 2021, using an Inspire 2 and a Phantom 4 Pro drone. In the first flight, the Inspire 2 drone covered the perimeter of the geography building. The maximum altitude approached by the drone was 50 m. The dataset produced from the first flight consists of 26 files (geo-tagged images) of 230 MB. In the second flight, the Phantom 4 Pro approached a maximum altitude of 60 m. The dataset produced by the second flight consists of 15 files of 104 MB.

4.1.4. Open Dataset

Part of the dataset (Group 6) [36] of Konya University, Turkey, was also used for evaluation purposes. This dataset was created by the University of Konya in Turkey using a DJI Phantom 4 Pro drone. The flight took place on 29 June 2018, capturing the campus of the university. The produced dataset consists of 440 files (geo-tagged images), has a total size of 3.59 GB, and constitutes the largest dataset used in our experiments. The size and the area of interest of this dataset are the main reasons for its selection.

4.2. Experiments

4.2.1. Trajectory Reconstruction

The trajectory reconstruction process was executed first. The extracted metadata from the geo-tagged images were transformed into a trajectory. All the trajectory-related data were stored in DataFrames. Figure 8 depicts a part of the extracted data from the University Hill dataset using the Phantom 4 Pro drone and the drone's corresponding reconstructed trajectory.

	t	geometry	alt	trajectory_id
0	2021-12-08 13:11:58	POINT (26.56950 39.08495)	99.07	1
1	2021-12-08 13:12:00	POINT (26.56950 39.08495)	99.07	1
2	2021-12-08 13:12:13	POINT (26.56937 39.08501)	98.97	1
3	2021-12-08 13:12:22	POINT (26.56936 39.08512)	98.97	1
4	2021-12-08 13:12:31	POINT (26.56915 39.08507)	99.07	1

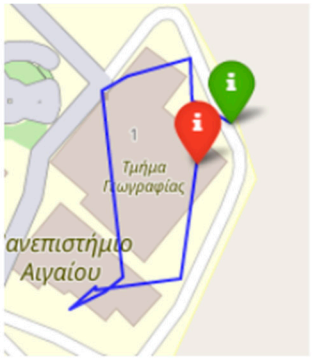


Figure 8. A trajectory reconstructed from extracted data/metadata.

4.2.2. Trajectory Enrichment

The reconstructed trajectory was enriched using the third-party open Web services for weather and POI data. To retrieve this data, the corresponding Web services were called using their APIs. The returned data were stored as DataFrames as well. Each row of the DataFrame represents a point of the trajectory and contains the corresponding external data. Figures 9 and 10 illustrate a portion of the weather and POI data, respectively, for the reconstructed enriched trajectory of the University Hill mission acquired by the Phantom 4 Pro drone.

```
t                                2021-12-08 13:11:58
alt                                            99.07
geometry          POINT (26.56949625 39.08494838888889)
dew_point                                        9.0
max_temperature                                 14.0
min_temperature                                 14.0
pressure                                      1010.0
wind_direction_max                             230.0
wind_direction_min                             230.0
wind_speed_max                                  14.8
wind_speed_min                                  14.8
```

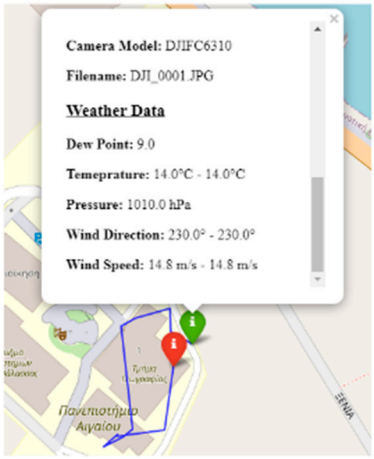


Figure 9. Weather data obtained using open Web services.

```
t                                          2021-12-08 13:11:58
alt                                                     99.07
geometry                  POINT (26.56949625 39.08494838888889)
place_id                                           162166559
osm_type                                                 way
display_name    University of Aegean, Πανεπιστημίου, Λόφος Ξεν...
boundingbox       [39.082758, 39.0865758, 26.5674222, 26.570065]
address         {'amenity': 'University of Aegean', 'road': 'Π...
```

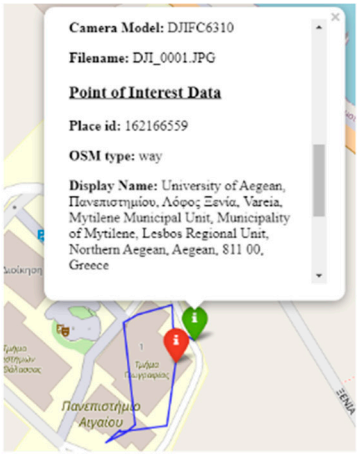


Figure 10. Point of interest data obtained using open Web services.

4.2.3. Trajectory Segmentation

The enriched trajectory was segmented into smaller parts (segments) based on specific criteria and thresholds defined by the user. Segmentation criteria constituted (a) the altitude variations of the drone and (b) the time interval and distance between two consecutive points (geo-tagged photos). Figure 11 illustrates the results of the segmentation using the segmentation criterion of distance with a threshold equal to 50 m. The segmentation process was implemented in the reconstructed trajectory of the Konya University dataset using the Phantom 4 drone.

```
semantic_traj.trajectory_segmentation(threshold=5, attr="altitude")
```

```
3 segments were created based on altitude with a threshold of 5 meters
The created segments are the following:
(point-0 - point-250)
(point-251 - point-353)
(point-354 - point-438)
```

Figure 11. Trajectory segments based on distance.

4.2.4. Stop Detection

The functionality of MovingPandas was used to detect stops in the evaluated trajectories. The corresponding module calculated and returned the number of recording points included in each stop. The user determined the duration of the stop and the maximum diameter, and the framework returned the data related to the detected stops. This data was visualized in the interactive visualization module. Figure 12 depicts the detected stops when the value for duration was 15, and the maximum diameter value was 20.

```
2 stops were found using as paremeters seconds = 15 and max diameter = 20
The stops found are the following:
Stop id: 1_2021-12-08 13:11:58
Trajectory id: 1
coordinates: (26.56949625, 39.08494838888889)
Start time: 2021-12-08 13:11:58, End time: 2021-12-08 13:12:13, Duration: 15.000000000000002 sec
Number of Recording Points: 3
------------------------------
Stop id: 1_2021-12-08 13:13:02
Trajectory id: 1
coordinates: (26.569136277777776, 39.08455188888889)
Start time: 2021-12-08 13:13:02, End time: 2021-12-08 13:13:39, Duration: 37.0 sec
Number of Recording Points: 5
------------------------------
```

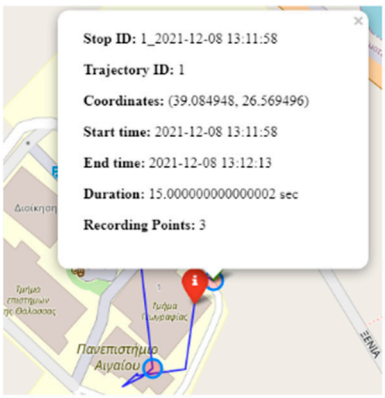


Figure 12. Detected stops of evaluated trajectories.

4.2.5. Trajectory Visualization

The interactive visualization module was used for the depiction of the reconstructed trajectory as a line with informative points. The percentage of the points to be displayed was defined by the user. The segmentation and stop detector modules were executed first, and then the produced segments and stops were visualized. For each trajectory part, the starting and ending point of the segment were displayed with a green and a red pin, respectively, along with a percentage of the recorded points. On the other hand, each trajectory stop was displayed as a circle, with a diameter determined by the users. This area was made clickable for displaying all the information related to a stop. The users were in control of the interactive visualization module, since they determined not only the type of map displayed (Stamen Toner, CartoDB positron, Cartodb dark_matter, etc.), but also the percentage of points displayed, the display (or not) of the segments, and the display (or not) of the stops. Figure 13 depicts the reconstructed trajectory of the Konya University dataset along with the abovementioned segments.

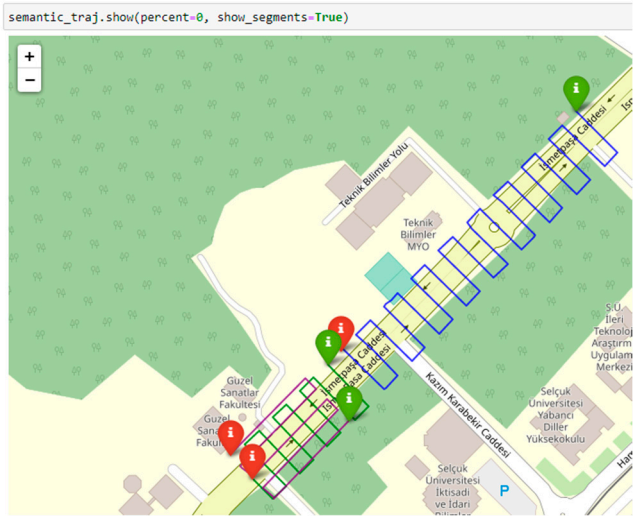
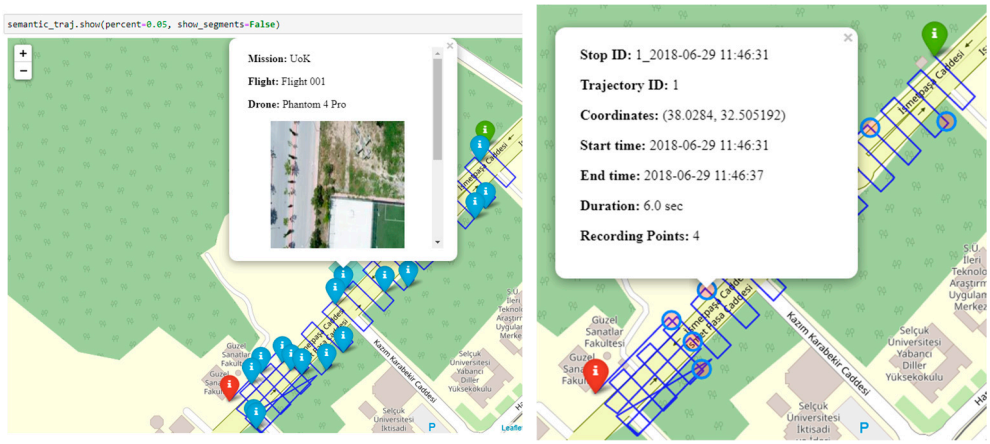


Figure 13. Visualization of a segmented trajectory.

The user was also able to click to each point and/or stop area and see all the information related to the particular point/area. Figure 14 illustrates the evaluated scenario.



(**a**) (**b**)

Figure 14. Interactive visualization of a trajectory. (**a**) Visualization of information contained in each point of the trajectory; (**b**) visualization of the data contained in a stop area of the trajectory.

4.2.6. Semantic Annotation and RDFization

Semantic annotation ensures a high-level formalism for the seamless integration of reconstructed trajectories and external heterogeneous data. This was achieved with an ontological approach using the Onto4Drone ontology. The semantically annotated data were exported to RDF (RDFization task). Figure 15 depicts a part of the Petrified Forest dataset exported in RDF, encoded in RDF/XML serialization.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:datA="http://www.datacron-project.eu/datAcron#">
<datA:Geometry rdf:about="#point_001">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
  <datA:hasTemporalFeature rdf:resource="#instant_001"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">point 001</rdfs:label>
  <datA:hasAltitude rdf:datatype="http://www.w3.org/2001/XMLSchema#string">99.07</datA:hasAltitude>
  <datA:hasLattitude rdf:datatype="http://www.w3.org/2001/XMLSchema#string">26.56949661111111</datA:hasLattitude>
  <datA:hasLongitude rdf:datatype="http://www.w3.org/2001/XMLSchema#string">39.08495016666667</datA:hasLongitude>
</datA:Geometry>
<datA:WeatherCondition rdf:about="#weather_condition_001">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">weather condition 001</rdfs:label>
  <datA:reportedMaxTemperature rdf:datatype="http://www.w3.org/2001/XMLSchema#string">14.0</datA:reportedMaxTemperature>
  <datA:reportedMinTemperature rdf:datatype="http://www.w3.org/2001/XMLSchema#string">14.0</datA:reportedMinTemperature>
  <datA:reportedDewPoint rdf:datatype="http://www.w3.org/2001/XMLSchema#string">9.0</datA:reportedDewPoint>
  <datA:reportedPressure rdf:datatype="http://www.w3.org/2001/XMLSchema#string">1010.0</datA:reportedPressure>
  <datA:windDirectionMax rdf:datatype="http://www.w3.org/2001/XMLSchema#string">230.0</datA:windDirectionMax>
  <datA:windDirectionMin rdf:datatype="http://www.w3.org/2001/XMLSchema#string">230.0</datA:windDirectionMin>
  <datA:windSpeedMax rdf:datatype="http://www.w3.org/2001/XMLSchema#string">14.8</datA:windSpeedMax>
  <datA:windSpeedMin rdf:datatype="http://www.w3.org/2001/XMLSchema#string">14.8</datA:windSpeedMin>
</datA:WeatherCondition>
```

Figure 15. A point in the evaluated trajectory, along with weather data, in RDF.

4.2.7. Trajectory Analytics

Semantic analytics was the final process of the evaluated approach. During this process, enriched and meaningful information was retrieved from the semantically integrated (RDF) data using SPARQL queries. Figure 16 depicts part of the result of a predefined query regarding the number of recording POIs of the trajectory, written in the Jupyter notebook interface.

```
list(default_world.sparql("""
            SELECT (COUNT(?recording_points) AS ?rp) WHERE {
                    ?tr dront:encloses ?rs .
                    ?rs TopDatAcronOnto_SSN_FM:comprises ?recording_points .
                }
      """))
```

```
[[15]]
```

Figure 16. An example SPARQL query defined by user in Jupyter notebook environment.

The source code and the testing environment of the ReconTraj4Drones system is provided at GitHub (https://github.com/KotisK/ReconTraj4Drones, accessed on 30 December 2022).

5. Discussion

To the best of our knowledge, ReconTraj4Drones is currently the only free and open-source trajectory management tool that implements trajectory reconstruction using geotagged photos, semantic annotation, and analytics in reconstructed and enriched semantic trajectories. In Table 1, an evaluation of related works based on specific criteria is presented. More specifically, the evaluation concerns the capability of related works with respect to trajectory reconstruction using other data sources beyond GPS data, the support of tasks such as segmentation and stop detection, the enrichment of trajectories with external data, their interactive visualization, and finally, the semantic annotation and

analytics of the reconstructed semantic trajectories. The columns correspond to the functionalities/tasks, and the rows correspond to the related studies.

Table 1. Comparison of available tools for trajectory reconstruction.

Tools	Reconstruction			Stop Detection	Segmentation	Enrichment	Semantic Annotation/ Analytics	Interactive Visualization
	Geo-Tagged Images	Video	Social Media Posts					
Yupi	-	√	-	-	-	-	-	-
scikit-mobility	-	-	√	√	√	-	-	√
ReconTraj4Drones	√	-	-	√	√	√	√	√

By reviewing the related works concerning the reconstruction of trajectories, it can be concluded that they emphasize the trajectory reconstruction process based on (i) geo-tagged photos in sources such as Flickr, (ii) videos, and (iii) users' posts on social media. However, these approaches do not provide a freely available tool for testing and evaluation. Our aim was to compare the ReconTraj4Drones framework with other implemented frameworks related to trajectory reconstruction using geo-tagged images and to measure its efficiency in terms of time, space complexity, and scalability; however, this was not possible due to the aforementioned reason.

We have evaluated the performance of ReconTraj4Drones using the datasets described in Section 3.1. The experiments were conducted using a laptop computer with an Intel Core(TM) i7-2670QM 2.2 GHz CPU with 8 GB RAM and a 50 Mbps internet connection. The framework's evaluation (Table 2) was based on the following criteria:

a. Memory used for the reconstruction of raw trajectory (MB).
b. Memory used for the enrichment process (MB).
c. Execution time for the reconstruction of raw trajectory (seconds).
d. Execution time for the enrichment process (seconds).
e. CPU usage for the reconstruction of raw trajectory (percentage).
f. CPU usage for the enrichment of raw trajectory (percentage).
g. RDFization capacity (number of RDF triples).

Table 2. Experimental results of ReconTraj4Drones evaluation.

Dataset	Dataset Size (Image Files)	Trajectory Length (m)	a (MB)	b (MB)	c (s)	d (s)	e (%)	f (%)	g (Triples)
Petrified Forest	206	1297.4	417	360	117	103	14.6	2.4	10,462
Vrisa Village	431	5366.7	402	388	67	235	8.6	1.4	21,262
University Hill (Phantom 4 Pro)	15	193.3	368	363	4	8.5	7.5	3.8	1294
University Hill (Inspire 2)	26	193	373	364	7.5	14.6	5.2	8.8	1822
University of Konya	439	2804.2	380	367	119	240	13.4	2.9	22,646

Despite the successful reconstruction, enrichment, and semantic annotation of drones' trajectories using ReconTraj4Drones, some limitations and restrictions were identified. A restriction concerns the integration of historic weather data. Particularly, the accuracy of the weather data is at the level of hours (not in minutes). As a result, all the points recorded during the same hour correspond to the same weather data. Regarding the Meteostat Web service, the returned weather data based on altitude were null in most of the cases. Thus, the weather data of the ground were mostly used. Finally, regarding the semantic annotation process, although the Owlready2 library provides an easy and efficient way to load and manipulate ontologies, annotating and semantically analyzing the reconstructed semantic trajectories, the library's maintenance has been discontinued since 2019.

Concerning the ReconTraj4Drones framework, although the initial requirements were implemented and successfully evaluated, the following limitations are identified and left for future work. First, the framework is able to manage only one trajectory at a time, making it incomplete in terms of the reconstruction of trajectories in a swarm of drones. Second, the implemented framework does not cover advanced trajectory analytics, such as trajectory clustering and prediction of the drone's movement.

6. Conclusions

The market for UAV applications has been steadily growing, pointing to UAVs' great potential in a variety of domains, such as the military, precision agriculture, documentation, surveillance, and the delivery of goods/products/services. The management and analysis of drones' movement data are becoming more and more popular, especially in the era of emerging technologies (IoT, robotics, AI, and big data analytics). Moreover, to derive meaningful information from these data, their semantic annotation and enrichment with external heterogenous data is required. Factors such as unexpected UAV malfunction, extreme weather conditions, or hostile actions (e.g., hacking) can cause incorrect, invalid, or even lost movement/trajectory data, making the trajectory reconstruction from other data (e.g., images, video) a demanding task. To respond to this problem, we have implemented a novel framework that reconstructs semantic trajectories of drones from geo-tagged photos collected during their flights. Moreover, the proposed framework integrates tasks that support the enrichment, segmentation, visualization, semantic annotation, and analysis of the reconstructed (semantic) trajectories. Such functionality has been implemented as an extension to the well-known and widely used MovingPandas platform, as a free and open-source analytic and visualization system of drone's semantic trajectories called ReconTraj4Drones. The evaluation of the implemented system with real datasets demonstrates that the reconstruction and modeling of drones' semantic trajectories using geo-tagged images can be efficiently supported by the proposed framework.

Author Contributions: Conceptualization, K.K.; methodology, K.K.; software, A.S.; validation, A.S., K.K.; formal analysis, A.S., K.K.; investigation, A.S., K.K.; resources, A.S., K.K.; data curation, A.S., K.K.; writing—original draft preparation, A.S.; writing—review and editing, K.K.; visualization, A.S.; supervision, K.K.; project administration, K.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors acknowledge the contribution to this work made by members of the research team of the "Research e-Infrastructure [e-Aegean R&D Network], Action 1.2 e-Aegean Geospatial data services" project, code number MIS 5046494, implemented within the framework of the "Regional Excellence" Action of the Operational Program "Competitiveness, Entrepreneurship and Innovation".

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wikipedia Contributors. *Unmanned Aerial Vehicle*; Wikimedia Foundatio: Saint Petersburg, FL, USA, 1964.
2. Perez-Grau, F.J.; Ragel, R.; Caballero, F.; Viguria, A.; Ollero, A. Semi-autonomous teleoperation of UAVs in search and rescue scenarios. In Proceedings of the 2017 International Conference on Unmanned Aircraft Systems (ICUAS), Miami, FL, USA, 13–16 June 2017; pp. 1066–1074. https://doi.org/10.1109/ICUAS.2017.7991349.
3. Shakhatreh, H.; Sawalmeh, A.H.; Al-Fuqaha, A.; Dou, Z.; Almaita, E.; Khalil, I.; Othman, N.S.; Khreishah, A.; Guizani, M. Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges. *IEEE Access* **2019**, *7*, 48572–48634. https://doi.org/10.1109/ACCESS.2019.2909530.
4. Soularidis, A.; Kotis, K. Semantic Modeling and Reconstruction of Drones' Trajectories. *Lect. Notes Comput. Sci.* **2022**, *13384*, 158–162. https://doi.org/10.1007/978-3-031-11609-4_30.
5. Parent, C.; Pelekis, N.; Theodoridis, Y.; Yan, Z.; Spaccapietra, S.; Renso, C.; Andrienko, G.; Andrienko, N.; Bogorny, V.; Damiani, M.L.; et al. Semantic trajectories modeling and analysis. *ACM Comput. Surv.* **2013**, *45*, 1–32. https://doi.org/10.1145/2501654.2501656.
6. Stek, T.D. Drones over Mediterranean landscapes. The potential of small UAV's (drones) for site detection and heritage management in archaeological survey projects: A case study from Le Pianelle in the Tappino Valley, Molise (Italy). *J. Cult. Herit.* **2016**, *22*, 1066–1071. https://doi.org/10.1016/J.CULHER.2016.06.006.
7. Themistocleous, K. *The Use of UAVs for Cultural Heritage and Archaeology*; Springer: Cham, Switzerland, 2020; pp. 241–269. https://doi.org/10.1007/978-3-030-10979-0_14.
8. Verhoeven, G.J.J. Providing an archaeological bird's-eye view—An overall picture of ground-based means to execute low-altitude aerial photography (LAAP) in Archaeology. *Archaeol. Prospect.* **2009**, *16*, 233–249. https://doi.org/10.1002/ARP.354.
9. Chiabrando, F.; Nex, F.; Piatti, D.; Rinaudo, F. UAV and RPV systems for photogrammetric surveys in archaelogical areas: Two tests in the Piedmont region (Italy). *J. Archaeol. Sci.* **2011**, *38*, 697–710. https://doi.org/10.1016/J.JAS.2010.10.022.
10. Colomina, I.; Molina, P. Unmanned aerial systems for photogrammetry and remote sensing: A review. *ISPRS J. Photogramm. Remote Sens.* **2014**, *92*, 79–97. https://doi.org/10.1016/j.isprsjprs.2014.02.013.
11. Nex, F.; Remondino, F. UAV for 3D mapping applications: A review. *Appl. Geomatics* **2014**, *6*, 1–15. https://doi.org/10.1007/S12518-013-0120-X/FIGURES/11.
12. Karatzoglou, A.; Koehler, D.; Beigl, M. Purpose-of-Visit-Driven Semantic Similarity Analysis on Semantic Trajectories for Enhancing the Future Location Prediction. In Proceedings of the 2018 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops, Athens, Greece, 19–23 March 2018; pp. 100–106. https://doi.org/10.1109/PERCOMW.2018.8480230.
13. Manaa, M.; Akaichi, J. Ontology-based modeling and querying of trajectory data. *Data Knowl. Eng.* **2017**, *111*, 58–72. https://doi.org/10.1016/j.datak.2017.06.005.
14. Vouros, G.A.; Santipantakis, G.M.; Doulkeridis, C.; Vlachou, A.; Andrienko, G.; Andrienko, N.; Fuchs, G.; Cordero Garcia, J.M.; Martinez, M.G. The datAcron Ontology for the Specification of Semantic Trajectories: Specification of Semantic Trajectories for Data Transformations Supporting Visual Analytics. *J. Data Semant.* **2019**, *8*, 235–262. https://doi.org/10.1007/s13740-019-00108-0.
15. Cai, G.; Lee, K.; Lee, I. Mining Semantic Trajectory Patterns from Geo-Tagged Data. *J. Comput. Sci. Technol.* **2018**, *33*, 849–862. https://doi.org/10.1007/s11390-018-1860-1.
16. Liu, K.; Wang, R.; Zheng, S.; Dong, S.; Sun, G. Fixed-time disturbance observer-based robust fault-tolerant tracking control for uncertain quadrotor UAV subject to input delay. *Nonlinear Dyn.* **2022**, *107*, 2363–2390. https://doi.org/10.1007/S11071-021-07080-0/TABLES/4.
17. Liu, K.; Wang, R. Antisaturation Adaptive Fixed-Time Sliding Mode Controller Design to Achieve Faster Convergence Rate and Its Application. *IEEE Trans. Circuits Syst. II Express Briefs* **2022**, *69*, 3555–3559. https://doi.org/10.1109/TCSII.2022.3167532.
18. Liu, K.; Wang, Y.; Ji, H.; Wang, S. Adaptive saturated tracking control for spacecraft proximity operations via integral terminal sliding mode technique. *Int. J. Robust Nonlinear Control* **2021**, *31*, 9372–9396. https://doi.org/10.1002/RNC.5774.
19. Liu, K.; Wang, R.; Wang, X.; Wang, X. Anti-saturation adaptive finite-time neural network based fault-tolerant tracking control for a quadrotor UAV with external disturbances. *Aerosp. Sci. Technol.* **2021**, *115*, 106790. https://doi.org/10.1016/J.AST.2021.106790.
20. Graser, A. MovingPandas: Efficient structures for movement data in Python. *GI_Forum* **2019**, *7*, 54–68. https://doi.org/10.1553/GISCIENCE2019_01_S54.
21. Package Overview—Pandas 1.3.5 Documentation. 2022. Available online: https://pandas.pydata.org/pandas-docs/stable/getting_started/overview.html (accessed on 17 January 2022).
22. GeoPandas 0.10.2.dev+28—GeoPandas 0.10.2+28.gd01d99b.dirty Documentation. 2022. Available online: https://geopandas.org/en/latest/ (accessed on 30 December 2022).
23. Reyes, A.; Viera-López, G.; Morgado-Vega, J.J.; Altshuler, E. yupi: Generation, Tracking and Analysis of Trajectory Data in Python. *arXiv* **2021**, arXiv:2108.06340.
24. Pappalardo, L.; Simini, F.; Barlacchi, G.; Pellungrini, R. Scikit-Mobility: A Python Library for the Analysis, Generation and Risk Assessment of Mobility Data. *arXiv* **2019**, arXiv:1907.07062.

25. datAcron Ontology. 2022. Available online: http://ai-group.ds.unipi.gr/datacron_ontology/ (accessed on 13 December 2022).
26. Ontology:DOLCE+DnS Ultralite—Odp. 2022. Available online: http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite (accessed on 18 December 2022).
27. SKOS Simple Knowledge Organization System Reference. 2022. Available online: https://www.w3.org/TR/skos-reference/ (accessed on 18 December 2022).
28. Haller, A.; Janowicz, K.; Cox, S.J.D.; Le Phuoc, D.; Taylor, K.; Lefrancois, M. Semantic Sensor Network Ontology. W3C Recommendation. 2017. Available online: https://www.w3.org/TR/2017/REC-vocab-ssn-20171019/ (accessed on 10 December 2022).
29. Available online: http://www.opengis.net/ont/sf (accessed on 18 December 2022).
30. GML Geometries. 2022. Available online: https://schemas.opengis.net/gml/3.2.1/gml_32_geometries.rdf (accessed on 18 December 2022).
31. OGC GeoSPARQL 1.0. 2022. Available online: https://schemas.opengis.net/geosparql/1.0/geosparql_vocab_all.rdf (accessed on 18 December 2022).
32. RDF - Semantic Web Standards. 2022. Available online: https://www.w3.org/RDF/ (accessed on 3 March 2022).
33. Lamy, J.-B. Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. *Artif. Intell. Med.* **2017**, *80*, 11–28. https://doi.org/10.1016/j.artmed.2017.07.002.
34. HermiT Reasoner: Home. 2022. Available online: http://www.hermit-reasoner.com/ (accessed on 18 December 2022).
35. SPARQL Query Language for RDF. 2022. https://www.w3.org/TR/rdf-sparql-query/ (accessed on 18 December 2022).
36. Makineci, H.; Karabörk, H. UAV images at the campus area II, Konya, Turkiye. *Mendeley Data V1* **2021**. https://doi.org/10.17632/3M8JYJDTC3.1.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.