Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.084300

ARTICLE

A Knowledge Graph Construction Method for UAV Flight Behaviour Understanding

Tian Liu[1], **Xichao Wang**[1,*], **Jun Wang**[2], **Song Gao**[2], **Hang Gao**[1], **Yuxi Liu**[1] **and Yitao Zhuang**[1]

[1]School of Electrical and Energy Engineering, Shanghai Dianji University, Shanghai, China
[2]School of Information Engineering, Henan University of Science and Technology, Luoyang, China

*Corresponding Author: Xichao Wang. Email: wangxc@sdju.edu.cn

Received: 20 April 2026; Accepted: 08 July 2026; Published: 13 August 2026

ABSTRACT: In unmanned aerial vehicle (UAV) flight behaviour understanding, the lack of a unified semantic representation for continuous multi-source temporal data makes it difficult to model flight events, behavioural relationships, and composite behaviours in an interpretable manner. To address this issue, this paper proposes a knowledge graph construction method for UAV flight behaviour understanding. The proposed method integrates atomic event detection, temporal knowledge modelling, and composite behaviour reasoning, thereby enabling the automatic transformation of continuous flight logs into structured behavioural knowledge. First, local statistical features, including smoothed velocity and robust climb rate, are extracted from multi-source flight logs. Adaptive dynamic decision boundaries based on local means and standard deviations are then used to detect atomic flight events such as Takeoff, Turn, Hover, and Descent. Next, a temporal knowledge graph is constructed with atomic events as the core semantic units to link UAV platforms, mission instances, and geographical regions. Finally, domain ontology and SWRL rules are introduced to support composite behaviour reasoning and semantic querying. Experiments conducted on a hybrid dataset derived from MUN-FRL and EuRoC MAV show that the proposed method achieves a macro-average F1 score of 0.83 across four typical event categories, with per-event F1 improvements ranging from 4 to 17 percentage points over the compared baselines under the same event-level evaluation protocol. Bootstrap-based validation provides additional evidence for the stability of the observed improvement under the current test setting. The constructed temporal flight knowledge graph contains more than 5600 entities. Typical semantic query response times remain within 30 ms under the current graph size and query setting. The proposed method provides an interpretable and structured framework for UAV flight behaviour analysis and understanding.

KEYWORDS: Unmanned aerial vehicle; flight behaviour understanding; atomic event detection; temporal knowledge graph; semantic reasoning; SWRL

1 Introduction

In recent years, unmanned aerial vehicles (UAVs) have been widely applied in surveying, disaster monitoring, logistics, and other fields [1]. During flight, UAV platforms continuously generate large volumes of multi-source logs, including altitude, speed, heading, and inertial measurements [2]. Although these logs record the evolution of flight states in detail, they remain continuous sensor-level time series and cannot directly represent behaviour-level semantics such as Takeoff, Turn, Hover, and Descent. They also cannot explicitly describe the temporal organisation of such behaviours within mission contexts [3].

This semantic gap limits the utility of flight logs in behaviour analysis, risk identification, and intelligent monitoring [4]. In practice, raw sensor values and simple trajectory visualisations are insufficient for

interpreting the UAV's operational behaviour during a specific period [5]. They are also inadequate for revealing how behaviour evolved over time in a semantically structured manner [6]. Potentially risky manoeuvres are likewise difficult to identify directly from low-level logs alone [7].

Existing methods mainly suffer from three limitations. First, rule-based event detection methods usually rely on fixed thresholds [8]. Such threshold settings are often sensitive to UAV type, mission configuration, and environmental noise [9]. Second, data-driven temporal models can classify windows or detect anomalies from flight logs [10]. Some methods further rely on multimodal sensor fusion to improve anomaly detection performance [11]. Deep learning has also been introduced for swarm-mission anomaly analysis [12]. Memory-based strategies have additionally been explored for anomaly detection with limited sensor variables [13]. Autoencoder-based methods have likewise been proposed for unsupervised UAV anomaly detection [14]. However, these models are often difficult to interpret [15]. They also cannot explicitly represent behaviour chains with temporal logic [5]. Third, existing flight-log-based studies have mainly used UAV logs as data sources for anomaly detection and behaviour assessment [16,17], while UAV knowledge-graph studies usually focus on relatively static domain knowledge, such as platform capability and mission/task planning. Therefore, high-frequency flight logs are still rarely transformed into event-centred semantic processes.

To address these issues, this paper proposes a knowledge graph construction method for UAV flight behaviour understanding. First, atomic flight events are detected from continuous flight logs by using local statistical features and adaptive dynamic boundaries [18]. Next, these events are organised into a temporal knowledge graph that links UAV platforms, mission instances, and geographical regions [19]. Finally, composite behaviours are inferred by integrating domain ontology with rule-based reasoning, thereby enabling semantic querying and structured behaviour analysis [20].

The main contributions of this paper are summarised as follows:

(1) An event-centred log-to-graph framework is proposed for UAV flight behaviour understanding. By using atomic flight events as intermediate semantic units, the framework connects continuous flight logs with temporal knowledge graph construction, distinguishing it from semantic trajectory annotation and static UAV knowledge graph modelling.
(2) An interpretable atomic event extraction method is developed by combining local statistical features, adaptive decision boundaries, state constraints, and interval-level post-processing. The method distinguishes change-sensitive manoeuvres from sustained-state behaviours.
(3) A temporal flight behaviour knowledge graph is constructed to represent UAVs, missions, spatial regions, atomic events, event chains, and composite behaviours in a unified semantic structure.
(4) Ontology-based rule reasoning and semantic querying are introduced to support behaviour-chain analysis, spatial behaviour retrieval, and risk-related behaviour identification.

2 Related Work

2.1 Flight Log Analysis and Anomaly Detection

UAV flight control systems continuously generate logs containing position, attitude, velocity, and system status information [2]. Early studies mainly focused on log parsing, key-field extraction, and trajectory reconstruction for accident investigation and mission review [1]. However, these methods generally remained at the level of data restoration and visual inspection [7]. They did not explicitly support high-level flight behaviour understanding [21].

With the development of machine learning and deep learning, research gradually shifted toward anomaly detection and health monitoring. Classical data-driven studies explored anomaly detection and

isolation for groups of UAVs [16]. PCA-based techniques were also introduced to assess UAV behavioural abnormality [17]. Prediction-based deep learning methods were later used for UAV sensor anomaly detection [22]. Spatiotemporal correlation modelling was further applied to anomaly detection and recovery prediction [23]. More recent work studied unsupervised anomaly detection and recovery from UAV flight data [4]. Multivariate regression was also introduced for data-driven anomaly detection and recovery [24]. LSTM-based models further addressed multirate flight data anomaly detection [10]. Hybrid multimodal neural networks were proposed to combine heterogeneous sensor information [11]. Research on swarm missions extended anomaly detection to coordinated multi-UAV scenarios [12]. Other work used memorisation of normality to handle anomaly detection with only a few sensor values [13]. Autoencoder-based frameworks were also proposed for unsupervised UAV anomaly detection [14]. Recent studies further explored flight anomaly localisation under random channel masking [25]. Attention-based multi-instance learning was also introduced to explain anomalous flight events [15].

Despite these advances, most existing studies still focus primarily on anomaly detection rather than behaviour understanding. Their outputs are often limited to anomaly labels or window-level categories. As a result, it remains difficult to represent behaviour chains, temporal logic, and mission context in an interpretable way.

2.2 Time-Series Event Detection Techniques

Time-series event detection methods can generally be divided into rule-driven and data-driven approaches [26]. A broad body of work has examined change-point detection as a core mechanism for segmenting temporal signals [8]. Direct density-ratio estimation has been used as an effective way to detect changes in time-series distributions [9]. These studies provide important foundations for segmenting continuous UAV logs into candidate behavioural intervals.

Rule-driven methods identify events by applying physical thresholds or logical conditions to variables such as speed, altitude, climb rate, and turn rate [27]. These methods are easy to implement and usually maintain good interpretability [17]. However, their thresholds often depend strongly on specific platforms and operating environments.

Data-driven methods formulate event detection as a sequence labelling or subsequence classification task. More recent studies have also developed parameter-free segmentation techniques for time series [28]. Streaming time-series segmentation has likewise been investigated for online scenarios [29]. Earlier work addressed time-series segmentation for context recognition in mobile devices [30]. Online segmentation algorithms were also proposed for continuous time-series streams. Although these approaches improve automation, they are often less interpretable in engineering applications. They also do not naturally encode explicit semantic relations between adjacent events. To balance interpretability and robustness, this paper adopts an adaptive local statistical boundary strategy and further organises the detected events into structured behavioural knowledge.

2.3 Knowledge Engineering in the UAV Domain

Knowledge graphs have been widely used for structured knowledge representation and reasoning. In the UAV domain, attentive semantic representation has been used for intelligent UAV knowledge graph construction [31]. Multi-domain fusion has also been applied to cargo UAV fault diagnosis knowledge graph construction [32]. Ontology-based autonomous task planning further demonstrated the role of structured knowledge in unmanned systems [18]. Mission-planning-oriented knowledge graphs for UAV systems have also been reported [19]. These studies show the value of knowledge engineering in UAV applications, but they mainly focus on static domain knowledge rather than dynamic flight processes [31].

Related studies on semantic trajectories have explored how movement data can be enriched with semantic geographical information [21]. Ontology-based trajectory modelling and reasoning were subsequently investigated in conceptual modelling research. General semantic trajectory modelling frameworks further formalised movement analysis from a semantic perspective. SeMiTri provided a framework for semantic annotation of heterogeneous trajectories. Recent UAV-oriented work studied semantic trajectories as knowledge graphs in application scenarios such as cultural heritage documentation [33]. Some studies explicitly discussed the engineering of UAVs' semantic trajectories as knowledge graphs [34]. Other work considered semantic modelling and reconstruction of UAVs' trajectories [35]. The datAcron ontology further provided a formal ontology for the specification of semantic trajectories [36]. More recent ontology research has examined the representation and querying of semantic trajectories [37]. Semantic trajectory patterns have also been mined from geo-tagged data [38].

However, these approaches are still not deeply integrated with high-frequency UAV flight logs and behaviour-oriented rule reasoning [33]. In particular, the transformation from continuous low-level flight logs to atomic flight events remains insufficiently connected with temporal knowledge graph construction [34]. Different from these studies, this paper focuses on transforming continuous flight logs into atomic events and modelling them within a temporal knowledge graph, so that the flight process itself can be represented as a semantic object for reasoning and querying [35].

In summary, existing studies provide useful foundations for UAV anomaly detection, time-series segmentation, semantic trajectory modelling, and UAV knowledge graph construction. However, these research directions are still not sufficiently integrated for UAV flight behaviour understanding. Time-series detection methods usually output window-level labels or abnormality scores, but they do not explicitly represent the temporal and semantic relations among flight behaviours. Semantic trajectory models enrich movement data with spatial or contextual semantics, but they usually focus on trajectory-level annotation rather than extracting atomic flight behaviours from high-frequency UAV logs. Existing UAV knowledge graphs mainly describe relatively static knowledge, such as platforms, missions, faults, or planning constraints, while the dynamic flight process is often not modelled as a first-class semantic object.

Different from these studies, this paper uses atomic flight events as the bridge between continuous logs and graph-based semantic reasoning. The proposed method first detects behaviour-level event intervals from low-level flight logs, then organises them into temporal event chains, and finally infers composite behaviours through ontology rules. This event-centred design is the main distinction between the proposed method and existing semantic trajectory or UAV knowledge graph studies.

3 Methodology

3.1 Overall Architectural Framework

The proposed framework is designed around atomic flight events rather than raw trajectory points or isolated sensor windows. Atomic events serve as intermediate semantic units that connect low-level flight-log signals with high-level graph reasoning. Therefore, the framework is not a simple sequential combination of signal processing, ontology modelling, and graph storage. Instead, each module is organised around the same event-centred representation: local statistical features are used to extract atomic events, the ontology defines event-centred semantic relations, the temporal knowledge graph stores event chains, and SWRL rules infer composite behaviours from event sequences.

Based on this event-centred design, the proposed method consists of three main stages: atomic flight event detection, temporal knowledge graph construction, and composite behaviour inference with semantic querying. Before these stages, preprocessing operations, including data cleaning, temporal alignment, and

feature extraction, are performed to support subsequent event detection and graph construction. The overall framework is illustrated in Fig. 1.

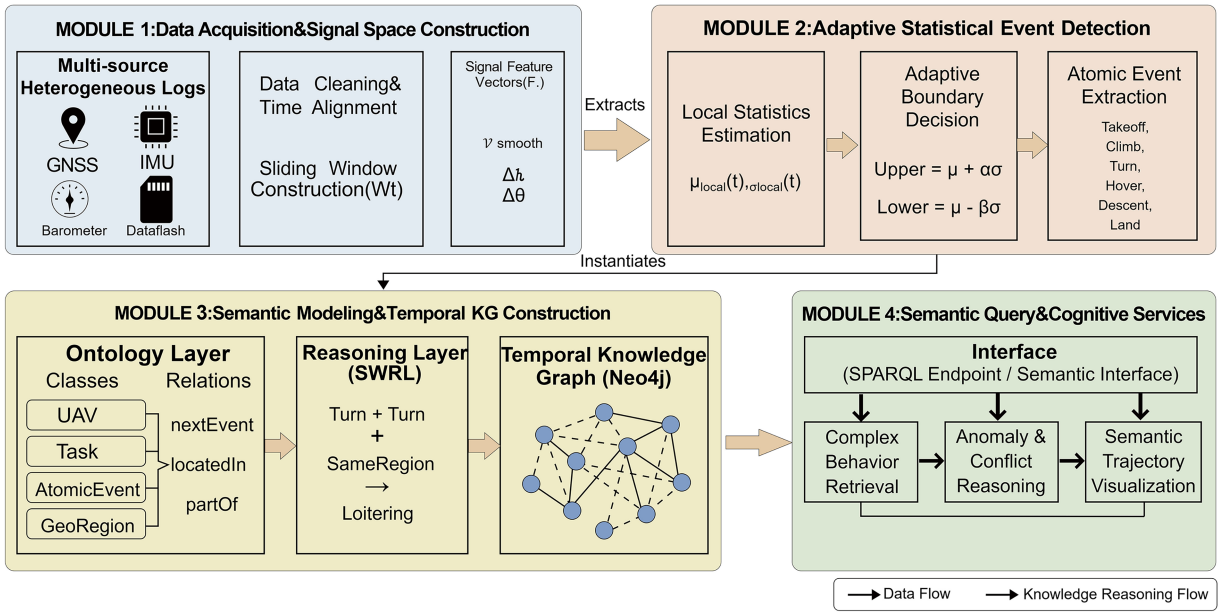

Figure 1: Overall technical approach.

3.2 Feature Construction for Flight Logs and Detection of Atomic Events

3.2.1 Construction of the Signal Space and Feature Extraction

Let the original flight log L be a sequence of observations strictly ordered by timestamp, as defined in Eq. (1).

$$L = \langle x_1, x_2, \ldots, x_T \rangle \tag{1}$$

here, $x_t \in R^d$ denotes the state vector at sample index t, comprising GPS coordinates, barometric altitude h_t, ground speed v_t, heading angle θ_t and other sensor readings.

For each timestamp t, a local sliding window is constructed to describe the temporal context around the current observation, as shown in Eq. (2):

$$W_t = \{x_i \mid max(1, t - k_l) \le i \le min(T, t + k_r)\} \tag{2}$$

where K denotes the total window length, T denotes the total number of observations in the flight log, and x_i denotes the flight-log observation at sample index i. Here, $k_l = [(K-1)/2]$ and $k_r = K - 1 - k_l$. This definition ensures that each local window contains approximately K samples, except near the beginning or end of the sequence where boundary truncation is required. In the experiments, K was set to 40, which is consistent with the input sequence length used by the supervised baseline models.

To illustrate more clearly the process of constructing local statistical features from flight logs, Fig. 2 shows the overall workflow, from the input of raw flight logs and the construction of symmetric sliding windows, through the extraction of v_{smooth}, robust climb rate and rate of turn, to the final formation of the feature vector Ft.

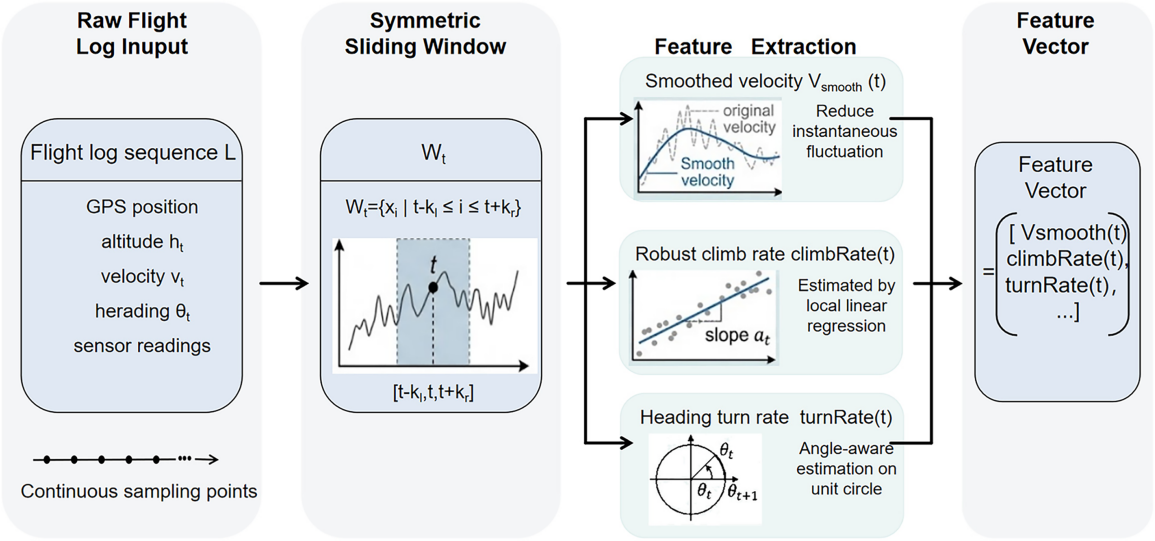

Figure 2: Framework for extracting local statistical features from flight logs.

Based on the local window W_t, the v_{smooth} is calculated using the median of the velocity sequence, as shown in Eq. (3):

$$v_{\text{smooth}}(t) = \text{median}\{v_i \mid x_i \in W_t\} \tag{3}$$

The median operation is adopted to suppress transient speed jitter and isolated abnormal sensor readings.

To estimate the vertical motion trend, a simple linear regression model is fitted to the altitude sequence within W_t. The slope of the fitted line is used as the robust climb rate, as shown in Eq. (4):

$$climbRate(t) = \frac{\sum_{x_i \in W_t} (t_i - t_{\text{mean}})(h_i - h_{\text{mean}})}{\sum_{x_i \in W_t} (t_i - t_{\text{mean}})^2} \tag{4}$$

where t_i and h_i denote the timestamp and altitude of observation x_i, respectively, and t_{mean} and h_{mean} are the mean timestamp and mean altitude within the local window. Compared with adjacent-point differencing, this regression-based estimation reduces the influence of barometric noise and short-term altitude fluctuations.

For heading variation, the periodicity of the heading angle must be considered. Therefore, the heading change rate is calculated after angular wrap-around correction, as shown in Eq. (5):

$$turnRate(t) = \frac{atan2\left(\sin\left(heading(t) - heading(t-1)\right), \cos\left(heading(t) - heading(t-1)\right)\right)}{\Delta t} \tag{5}$$

where Δt is the sampling interval and $heading(t)$ is measured in radians. If the original heading angle is recorded in degrees, it is first converted into radians before applying the sine and cosine functions. This operation maps the angular difference to the principal angle range and avoids false discontinuities caused by the wrap-around between 0° and 360°.

Finally, the feature vector used for adaptive event detection is defined in Eq. (6):

$$F_t = [v_{smooth}(t), climbRate(t), \text{turnRate}(t)] \tag{6}$$

to provide input for subsequent adaptive event detection.

3.2.2 Event Detection Based on Adaptive Statistical Characteristics

Conventional event detection typically employs a fixed global threshold τ (such as using a fixed climb-rate or heading-change threshold), which struggles to adapt to different mission phases, aircraft types and environmental noise. To address this, this paper proposes an adaptive detection algorithm based on local statistical moments.

For any given time t, the local mean and standard deviation of the target feature are calculated within the corresponding local feature window W_t^F, which follows the same K-sample window definition as W_t in Section 3.2.1, and are denoted by $\mu_{local}(t)$ and $\sigma_{local}(t)$, respectively.

Based on these statistics, the upper and lower adaptive decision boundaries are constructed as given in Eqs. (7) and (8).

$$Upper(t) = \mu_{local}(t) + \alpha \cdot \sigma_{local}(t) + \varepsilon \tag{7}$$

$$Lower(t) = \mu_{local}(t) - \beta \cdot \sigma_{local}(t) - \varepsilon \tag{8}$$

here, α and β are sensitivity coefficients, and $\varepsilon \geq 0$ is the baseline significance offset, which is used to control the conservativeness of the detection.

For a given feature ft (e.g., $v_{smooth}(t)$, climbRate(t), turnRate(t)), the point-level detection function for UAV events is defined in Eq. (9).

$$\mathbb{I}_{event}(t) = \begin{cases} 1, & f_t \geq Upper(t) \ or f_t \leq \text{Lower}(t) \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

After point-level boundary judgment, the detected points are further assigned to specific event categories according to the physical meaning of the dominant flight feature. In the experimental evaluation, four representative event categories are considered, namely Takeoff, Turn, Hover, and Descent. Their labels are assigned based on altitude variation, smoothed velocity, heading change rate, and duration constraints.

For each timestamp t, the candidate event label is determined as follows. If climbRate(t) exceeds the adaptive upper boundary of the climb-rate feature, the altitude shows a continuous increasing trend within the local window, and the event starts from a low-altitude or ground-near state, the timestamp is labelled as Takeoff. If climbRate(t) is lower than the adaptive lower boundary and the altitude shows a continuous decreasing trend, the timestamp is labelled as Descent. If the absolute value of turnRate(t) exceeds the adaptive upper boundary of the heading-change feature and $v_{smooth}(t)$ is higher than the movement threshold, the timestamp is labelled as Turn. If $v_{smooth}(t)$, |climbRate(t)|, and |turnRate(t)| remain below predefined low-motion thresholds for a minimum duration, the timestamp is labelled as Hover. Otherwise, the timestamp is labelled as Normal.

After candidate labels are obtained, consecutive timestamps with the same non-Normal label are merged into candidate event intervals. Candidate intervals shorter than the minimum duration threshold are removed to suppress transient false detections. Adjacent intervals with the same label are further merged when their temporal gap is shorter than the predefined merging threshold. For each retained event interval, the event type, start time, end time, duration, and representative spatial location are recorded.

It should be noted that the adaptive boundary mechanism is mainly used for change-sensitive events, such as Takeoff, Turn, and Descent. Hover is a sustained low-motion state rather than an abrupt change. Therefore, Hover is identified using state constraints, including low smoothed velocity, near-zero climb rate, limited heading change rate, and a minimum duration requirement. This design avoids missing long stable hovering segments that may become part of the local statistical baseline.

Building upon the point-level candidate labels, the minimum-duration constraint and adjacent-interval merging strategy are applied to generate the final ordered atomic event set, as shown in Eq. (10).

$$E = \{e_1, e_2, \ldots, e_n\} \tag{10}$$

Each event ei possesses attributes such as type, start and end times, duration, and representative spatial location.

This adaptive detection strategy utilises $\sigma_{\text{local}}(t)$ to automatically detect local noise and manoeuvrability: during steady flight phases (when $\sigma_{\text{local}}(t)$ is small), it is more sensitive to minute changes, facilitating the detection of subtle manoeuvres; during periods of vigorous manoeuvring (where $\sigma_{\text{local}}(t)$ is large), it automatically raises the detection threshold to help suppress false events caused by significant fluctuations, thereby enhancing the overall robustness of the detection.

Before introducing the temporal knowledge graph construction, the computational cost of the atomic event detection process is briefly analysed. Let T denote the number of observations in a flight log and K denote the total sliding-window length. The direct computation of local statistical features within each sliding window has a time complexity of O(TK). When rolling-window updates or incremental statistics are adopted, this computation can be reduced to O(T). The point-level event labelling step requires a single scan of the sequence and therefore has O(T) complexity. The subsequent interval merging and short-interval filtering steps also require O(T) time. If M atomic event intervals are generated, entity mapping and temporal relation construction require O(M) time. Therefore, when rolling-window statistics are used, the overall event detection and graph construction process is approximately linear with respect to the number of flight-log observations.

3.3 Modelling of Temporal Knowledge Graphs

3.3.1 Construction of Domain Ontologies

To support UAV flight behaviour understanding, this paper constructs a domain ontology centred on the flight process. The ontology describes the semantic chain from UAV platforms and mission instances to atomic flight events, composite behaviours, and spatial regions, so that flight behaviours can be represented in a unified and machine-interpretable manner.

Formally, the ontology is defined as O = (C, P, R), where C, P, and R denote the class set, property set, and relation set, respectively. The class set includes six core concepts: UAV, Task, GeoRegion, AtomicEvent, CandidateBehavior, and ComplexBehavior. In the implemented framework evaluated in this study, AtomicEvent contains four subclasses: Takeoff, Descent, Turn, and Hover. Other flight-event classes, such as Climb and Landing, can be incorporated in future extensions of the ontology, but they were not included in the current detection, annotation, or experimental evaluation. GeoFence is defined as a subclass of GeoRegion and represents a region subject to operational or safety constraints. CandidateBehavior represents an intermediate behaviour instance generated from a sequence of supporting atomic events before rule-based classification. ComplexBehavior represents higher-level behaviour classes inferred by the SWRL rules, including Loitering, DangerousApproach, RepeatedHovering, RepeatedTurning, TakeoffAndHover, and DescentAfterHover.

The property set consists of object properties and data properties. The principal object properties include performedBy, partOf, locatedIn, nextEvent, and consistsOf. The relation performedBy(AtomicEvent, UAV) indicates that an atomic event is performed by a UAV; partOf(AtomicEvent, Task) indicates that an event belongs to a mission task; locatedIn(AtomicEvent, GeoRegion) specifies the geographical region in which an event occurs; and nextEvent(AtomicEvent, AtomicEvent) represents immediate temporal

succession between two adjacent events. The relation consistsOf(CandidateBehavior, AtomicEvent) links a candidate composite behaviour to the atomic events supporting its inference. A candidate behaviour can also be associated with a mission and a geographical region through partOf(CandidateBehavior, Task) and locatedIn(CandidateBehavior, GeoRegion), respectively.

The data properties of AtomicEvent include startTime, endTime, hasDuration, hasAltitude, and eventType. The properties startTime and endTime record the temporal boundaries of an event, hasDuration represents its duration in seconds, hasAltitude represents its representative altitude in metres, and eventType records its event category. The data properties of CandidateBehavior include hasTimeSpan, hasDisplacement, and hasTimeGap. hasTimeSpan denotes the total temporal span of the supporting atomic-event sequence, hasDisplacement denotes the spatial displacement associated with the sequence, and hasTimeGap denotes the temporal gap between adjacent supporting events. The units of hasTimeSpan and hasTimeGap are seconds, while the unit of hasDisplacement is metres.

During reasoning, candidate behaviour instances are first generated and connected to their supporting atomic events through consistsOf. The SWRL rules subsequently classify these instances into the corresponding subclasses of ComplexBehavior according to event type, temporal order, spatial relation, altitude, displacement, and time constraints. Fig. 3 illustrates the revised class hierarchy and the object and data properties of the proposed ontology.

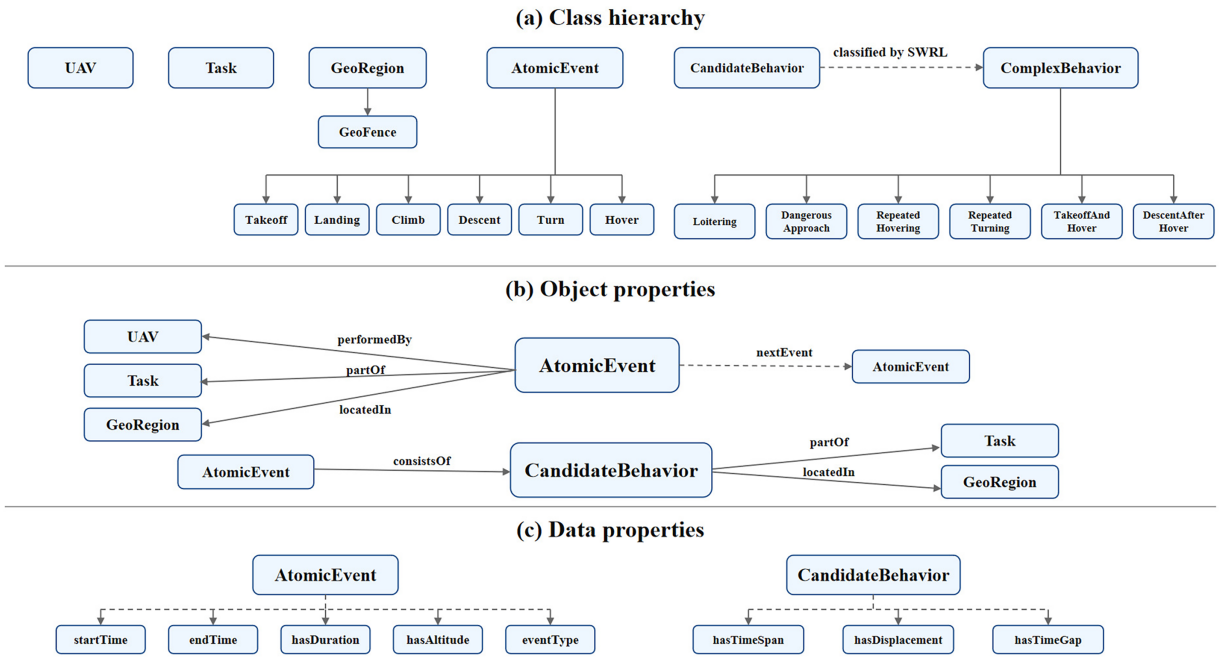

Figure 3: Class hierarchy and object/data properties of the UAV flight behaviour ontology.

3.3.2 Process for Constructing a Temporal Knowledge Graph

Following the definition of the domain ontology, this paper proceeds to construct a temporal knowledge graph based on the atomic flight events identified in the preceding section, following the workflow of "atomic event generation—entity mapping—relationship generation—graph storage". Fig. 4 illustrates the overall construction process from continuous flight logs to the temporal knowledge graph. This process uses atomic events as the intermediate semantic units and, through steps such as entity mapping, relationship generation and graph storage, ultimately forms a structured representation of behavioural knowledge.

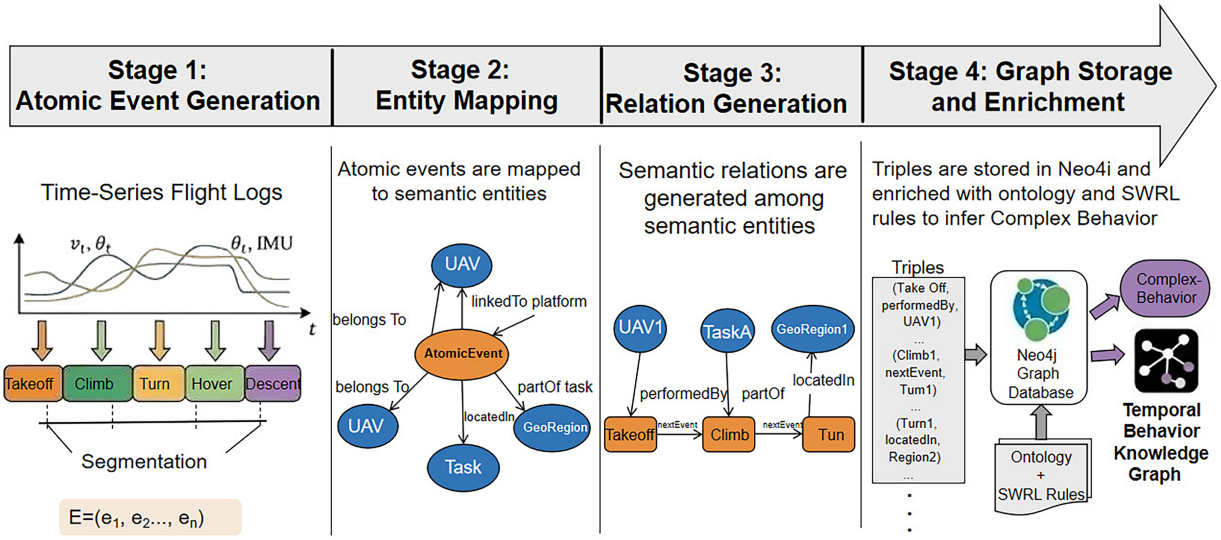

Figure 4: Process for constructing a temporal knowledge graph.

(1) Generation of Atomic Events

First, the raw flight logs are processed using the event detection method proposed earlier, which is based on local statistical features and adaptive dynamic boundaries, to identify atomic events such as Takeoff, Descent, Turn, and Hover. For each detected atomic event, attributes such as event type, start time, end time, duration and corresponding representative spatial location are recorded, thereby forming a chronologically ordered set of atomic events, as expressed in Eq. (10).

In particular, any elementary event e_i can be expressed as in Eq. (11).

$$e_i = (type_i, start_i, end_i, duration_i, loc_i) \tag{11}$$

This step converts a continuous sequence of numerical values into discrete behavioural units, providing the foundational input for subsequent knowledge graph construction.

(2) Entity Mapping

Once the set of atomic events has been obtained, the detection results are further mapped to entity nodes within the graph. Specifically, each atomic event e_i is mapped to an AtomicEvent class entity and, based on the platform identifier, task number and spatial location information in the log, is associated with the corresponding UAV, Task and GeoRegion entities, respectively.

Here, the UAV entity represents the flying entity, the Task entity represents the current flight mission or mission segment, and the GeoRegion entity represents the geographical area where the event occurred. Through this mapping process, numerical fragments in the raw logs are converted into computable, organised semantic entities, thereby enabling key behavioural units during flight to be represented within a unified semantic framework.

(3) Relationship Generation

First, subject, task, and spatial relations are generated for each atomic event, including performedBy (AtomicEvent, UAV), indicating that a specific UAV performs a given atomic event; partOf(AtomicEvent, Task), indicating that the event belongs to a specific task instance; and locatedIn(AtomicEvent, GeoRegion), indicating that the event occurs within a specific geographical region.

Second, to describe the dynamic evolution of the flight process, temporal relations are constructed according to the chronological order of atomic events. For adjacent events within the same task, the relation nextEvent(*ei*, *ej*) is established to indicate that event *ej* occurs immediately after event *ei*. When necessary, more general temporal relations, such as Before and Follow, can also be introduced to preserve the continuity of the event sequence. Through these relations, originally discrete atomic events are organised into behaviour chains with explicit temporal logic and spatial context, thereby providing a structural basis for subsequent composite behaviour inference.

(4) Graph Storage

After entity and relation generation, the resulting data are converted into triples and stored in a graph database. Specifically, entity instances, attribute information, and object relations are mapped to graph nodes, node attributes, and edges, respectively, and Neo4j is adopted for graph storage and management.

During storage, atomic event entities and their associations with UAVs, Tasks, and GeoRegions are written into the graph database while preserving key attributes such as startTime, endTime, and hasDuration. For event chains satisfying predefined temporal and spatial conditions, CandidateBehavior instances are first generated during graph construction and linked to their supporting atomic events through the consistsOf relation. SWRL reasoning is subsequently used to classify these candidate instances into specific subclasses of ComplexBehavior, such as Loitering, DangerousApproach, and RepeatedHovering. The resulting temporal knowledge graph not only preserves the flight events themselves, but also explicitly represents the temporal order, spatial constraints, and mission context among events.

Through the above process, continuous flight logs are transformed into a structured behavioural knowledge graph, enabling raw temporal data to be organised in an interpretable, inferable, and queryable form. This provides a unified data foundation for subsequent composite behaviour inference and semantic querying.

3.4 Composite Behaviour Inference and Semantic Querying

After the atomic event graph is constructed, composite behaviours are further inferred through ontology-based rules, and semantic queries are used to evaluate graph-level application capability. The reasoning module is designed on top of the event-centred representation. Atomic events are first detected from flight logs and mapped into graph entities. Candidate composite behaviour instances are then generated from temporally adjacent atomic-event sequences and linked to their supporting atomic events through the relation consistsOf. SWRL rules are used to classify these candidate behaviours according to event type, temporal order, spatial region, altitude constraint, and duration constraint.

It should be noted that SWRL rules are used here for behaviour classification and consistency checking rather than directly creating new individuals. In other words, the complex behaviour instance b is first generated as a candidate behaviour entity during graph construction, and the rule engine then determines whether b can be classified as Loitering, DangerousApproach, or other composite behaviour types. This avoids the ambiguity of introducing Loitering(b) without specifying how b is connected to the supporting atomic events.

To make the graph construction process explicit, Table 1 lists representative RDF-style triples generated from a detected atomic event and its associated candidate composite behaviour. For example, suppose that a Turn event is detected in Task_03, performed by UAV_01, and located in Region_A. Additional triples are then generated to connect the candidate composite behaviour with its supporting atomic events.

Table 1: Example triples for atomic event and candidate composite behaviour representation.

Graph Component	Representative Triples
Entity definition	UAV_01 rdf:type UAV Task_03 rdf:type Task Region_A rdf:type GeoRegion
Atomic event	Event_015 rdf:type Turn Event_015 performedBy UAV_01 Event_015 partOf Task_03 Event_015 locatedIn Region_A
Temporal relation	Event_015 startTime “35.2” Event_015 nextEvent Event_016 Event_015 endTime “42.6” Event_015 hasDuration “7.4”
Candidate behaviour	Behaviour_002 rdf:type CandidateBehavior Behaviour_002 consistsOf Event_015 Behaviour_002 consistsOf Event_016 Behaviour_002 partOf Task_03 Behaviour_002 locatedIn Region_A

Table 2 further summarises representative composite behaviour reasoning rules defined on the basis of candidate behaviours and their supporting atomic events.

Table 2: Representative composite behaviour reasoning rules.

Rule ID	Composite Behaviour	Supporting Atomic Events	Main Constraints
R1	Loitering	Turn–Turn–Turn	Same region, short time gap, limited displacement
R2	DangerousApproach	Descent	Low altitude, sensitive region
R3	RepeatedHovering	Hover–Hover	Same region, repeated occurrence
R4	RepeatedTurning	Turn-Turn	Consecutive turns, short temporal gap
R5	TakeoffAndHover	Takeoff → Hover	Early mission phase, short time gap
R6	DescentAfterHover	Hover → Descent	Temporal succession, same mission

The threshold parameters used in the SWRL-style rules are defined as follows. θ_t denotes the maximum temporal span within which a group of atomic events can be combined into a candidate composite behaviour; θ_d denotes the maximum spatial displacement allowed among the supporting events of a Loitering behaviour; θ_{safe} denotes the safety-altitude threshold used to identify a potential DangerousApproach behaviour; and θ_{gap} denotes the maximum temporal gap allowed between two consecutive atomic events.

The candidate ranges of these parameters were determined according to the distributions observed in the training data and the relevant temporal, spatial, and operational constraints. The final parameter settings were selected on the validation set according to composite-behaviour reasoning accuracy and were fixed before evaluation on the test set. No threshold parameter was adjusted using the test-set results. The complete parameter configuration is available from the corresponding author upon reasonable request.

Representative SWRL-style rules are given as follows.

Rule 1: Loitering behaviour.

If a candidate behaviour contains consecutive turning events within a short time window, and these events occur in the same geographical region with limited displacement, the candidate behaviour is classified as Loitering:

$$\begin{aligned}&\text{CandidateBehavior}(b) \wedge \text{consistsOf}(b, e_1) \wedge \text{consistsOf}(b, e_2) \wedge \text{consistsOf}(b, e_3)\\ &\wedge \text{Turn}(e_1) \wedge \text{Turn}(e_2) \wedge \text{Turn}(e_3)\\ &\wedge \text{nextEvent}(e1, e2) \wedge \text{nextEvent}(e2, e3)\\ &\wedge \text{locatedIn}(e_1, r) \wedge \text{locatedIn}(e_2, r) \wedge \text{locatedIn}(e_3, r)\\ &\wedge \text{hasTimeSpan}(b, t) \wedge \text{swrlb:lessThan}(t, \theta_t)\\ &\wedge \text{hasDisplacement}(b, d) \wedge \text{swrlb:lessThan}(d, \theta_d)\\ &\rightarrow \text{Loitering}(b)\end{aligned}$$

Rule 2: Dangerous low-altitude approach.

If a candidate behaviour contains a descent event in a sensitive region and the altitude is below a predefined safety threshold, the candidate behaviour is classified as DangerousApproach:

$$\begin{aligned}&\text{CandidateBehavior}(b) \wedge \text{consistsOf}(b, e)\\ &\wedge \text{Descent}(e)\\ &\wedge \text{locatedIn}(e, r) \wedge \text{GeoFence}(r)\\ &\wedge \text{hasAltitude}(e, h) \wedge \text{swrlb:lessThan}\left(h, \theta_{safe}\right)\\ &\rightarrow \text{DangerousApproach}(b)\end{aligned}$$

Rule 3: Repeated hovering.

If a candidate behaviour contains repeated hovering events in the same region within a short time interval, the candidate behaviour is classified as RepeatedHovering:

$$\begin{aligned}&\text{CandidateBehavior}(b) \wedge \text{consistsOf}(b, e_1) \wedge \text{consistsOf}(b, e_2)\\ &\wedge \text{Hover}(e_1) \wedge \text{Hover}(e_2)\\ &\wedge \text{nextEvent}(e_1, e_2)\\ &\wedge \text{locatedIn}(e_1, r) \wedge \text{locatedIn}(e_2, r)\\ &\wedge \text{hasTimeGap}(b, g) \wedge \text{swrlb:lessThan}\left(g, \theta_{gap}\right)\\ &\rightarrow \text{RepeatedHovering}(b)\end{aligned}$$

Rule 4: Repeated turning.

If two or more turn events occur consecutively within the same mission and their temporal gap is smaller than a predefined threshold, the candidate behaviour is classified as RepeatedTurning:

$$\begin{aligned}&\text{CandidateBehavior}(b) \wedge \text{consistsOf}(b, e_1) \wedge \text{consistsOf}(b, e_2)\\&\wedge \text{Turn}(e_1) \wedge \text{Turn}(e_2)\\&\wedge \text{nextEvent}(e_1, e_2)\\&\wedge \text{partOf}(e_1, m) \wedge \text{partOf}(e_2, m)\\&\wedge \text{hasTimeGap}(b, g) \wedge \text{swrlb:lessThan}\left(g, \theta_{gap}\right)\\&\rightarrow \text{RepeatedTurning}(b)\end{aligned}$$

Here, b denotes a candidate behaviour instance; e, e_1, e_2, and e_3 denote atomic-event instances; r denotes a geographical region; m denotes a mission instance; and g, t, d, and h denote the temporal gap, temporal span, spatial displacement, and altitude, respectively. The predicate consistsOf links a candidate behaviour to its supporting atomic events, nextEvent represents immediate temporal succession, and partOf and locatedIn represent the mission and spatial contexts of an event, respectively.

Due to space limitations, only representative SWRL-style rules are shown in the main text, while the implemented rule set includes the six behaviour rules listed in Table 2.

In the implementation, Protégé was used to define and validate the ontology schema, including classes, object properties, and data properties. Detected atomic events and candidate composite behaviours were then materialised according to this ontology schema. SWRL rules were used to classify candidate behaviours based on event type, temporal order, spatial relation, altitude threshold, and duration constraint. The HermiT reasoner was adopted to check ontology consistency and infer class memberships of candidate behaviours under the predefined rules. After reasoning, the inferred behaviour types and semantic relations were exported together with the event entities and written into Neo4j for graph storage, Cypher-based querying, and visualisation. In this workflow, Protégé and HermiT are mainly used for ontology validation and rule-based reasoning, while Neo4j is used for graph storage, retrieval, and visual analysis.

4 Experiments and Results

4.1 Experimental Setup

4.1.1 Dataset Construction

Experiments were conducted on two publicly available UAV flight-log datasets, namely MUN-FRL and EuRoC MAV. MUN-FRL contains outdoor flight sequences collected from aerial platforms such as a DJI M600 hexacopter and an NRC Bell 412 ASRA helicopter, while EuRoC MAV provides visual-inertial flight sequences mainly acquired from multirotor platforms. Since the two datasets differ in platform type, flight envelope, and mission scenario, they were processed both separately and jointly in this study.

For each dataset, position, altitude, ground speed, heading angle, and IMU-related variables were extracted when available. All flight logs were temporally aligned and resampled to a unified sampling interval. Missing values, abnormal timestamps, and incomplete records were removed before feature construction. To reduce the influence of platform-specific scale differences, local statistical features were calculated within each mission sequence rather than across the entire hybrid dataset.

The hybrid dataset was constructed only after the event definitions and preprocessing procedures were unified. In the experiments, MUN-FRL and EuRoC MAV were first evaluated independently, and the hybrid dataset was then used as an additional cross-scenario evaluation setting. This design avoids relying solely on mixed-data results and allows the platform-specific performance of the proposed method to be analysed.

4.1.2 Annotation Protocol

The event boundaries and event categories were manually annotated according to a predefined annotation guideline. The guideline specified the semantic definition, minimum-duration requirement, and temporal-boundary criteria for each event category. Four representative event categories were considered in this study: Takeoff, Turn, Hover, and Descent.

Takeoff was defined as the initial continuous upward transition from a ground-near or low-altitude state. Turn was defined as a non-stationary flight interval characterised by continuous and significant heading variation. Hover was defined as a sustained low-motion interval with low ground speed, a near-zero climb rate, limited heading variation, and a minimum-duration requirement. Descent was defined as a continuous downward altitude-transition interval.

The initial annotations were produced by one researcher with experience in UAV flight-data analysis. During annotation, the synchronised altitude, ground-speed, heading, climb-rate, and trajectory profiles were jointly inspected. Each annotated event was assigned an event type, start time, end time, mission identifier, and representative spatial region.

After the initial annotation, all event labels and temporal boundaries were manually reviewed by two additional researchers. The review focused on whether the assigned event type was consistent with the predefined annotation guideline and whether the start and end boundaries corresponded to observable changes in the synchronised sensor profiles. When a questionable label or boundary was identified, the relevant flight segment was re-examined, and the final annotation was determined through discussion based on the altitude, velocity, heading, climb-rate, and trajectory information.

After manual checking and conflict resolution, the final annotation set contained 433 Takeoff events, 1333 Turn events, 820 Hover events, and 394 Descent events, for a total of 2980 annotated events. Specifically, the MUN-FRL dataset contained 253 Takeoff, 840 Turn, 520 Hover, and 247 Descent events, whereas the EuRoC MAV dataset contained 180 Takeoff, 493 Turn, 300 Hover, and 147 Descent events.

Because the annotations were not independently produced by multiple annotators, a formal inter-annotator agreement statistic, such as Cohen's kappa, was not available. This limitation has been explicitly acknowledged in the revised manuscript. Future work will adopt an independent multi-annotator protocol and report formal agreement statistics to further assess annotation reliability.

To avoid temporal leakage caused by overlapping sliding windows, the data were split at the mission level rather than at the individual-window level. The mission-level split followed a 70%/15%/15% ratio for training, validation, and testing, respectively. All windows extracted from the same mission were assigned to the same subset, and the same split was used for all compared methods.

4.1.3 Baseline Configuration and Fair Comparison Protocol

To ensure reproducibility and fair comparison, the implementation details of the baseline methods are specified in this section. Three representative time-series classification baselines were used, namely LSTM-Cls, MiniRocket, and Transformer-Cls. These methods represent recurrent, kernel-based, and self-attention-based sequence modelling paradigms, respectively.

All baseline models used the same preprocessed input features, mission-level data split, and evaluation metrics as those used by the proposed method. The input sequence length was set to 40. Under the window definition in Section 3.2.1, this corresponds to the same sliding-window length K = 40 used by the proposed method, ensuring that the supervised baselines and the proposed method operate on comparable local temporal contexts.

LSTM-Cls was implemented as a two-layer LSTM classifier with a hidden size of 64 and a dropout rate of 0.3. The final hidden state was fed into a fully connected layer for event classification.

Transformer-Cls consisted of two Transformer encoder layers with four attention heads, a model dimension of 64, a feed-forward dimension of 128, and a dropout rate of 0.1. A fully connected classification head was used to predict the window-level event label.

MiniRocket used 10,000 convolutional features generated from a fixed set of predefined convolutional kernels to transform each input sequence into a fixed-dimensional representation, followed by a ridge classifier.

For the neural network baselines, the Adam optimiser was adopted with a learning rate of 1e−3. The batch size was set to 128, and the maximum number of training epochs was set to 50. The validation set was used for hyperparameter selection and model monitoring.

Since LSTM-Cls, MiniRocket, and Transformer-Cls generate window-level predictions, their outputs were converted into event intervals before event-level evaluation. Consecutive windows with the same predicted non-Normal label were merged into candidate event intervals. Candidate intervals shorter than the minimum duration threshold were removed, and adjacent intervals of the same type were merged when their temporal gap was shorter than the predefined merging threshold. The same post-processing strategy was applied to all methods to ensure fair interval-level comparison.

4.1.4 Evaluation Metrics

Precision, recall, and F1 score were used to evaluate event detection performance. A predicted event was considered correctly matched with a ground-truth event only when they belonged to the same event category and their temporal intersection-over-union was not less than 0.5. Unmatched predicted events were counted as false positives, and unmatched ground-truth events were counted as false negatives.

Boundary error (*BE*) was further used to evaluate the temporal localisation accuracy of detected event intervals. For a matched predicted event and ground-truth event, *BE* was defined as the average absolute deviation between their start and end timestamps, as shown in Eq. (12):

$$BE = \frac{\left|t_{\text{start,pred}} - t_{\text{start,true}}\right| + \left|t_{\text{end,pred}} - t_{\text{end,true}}\right|}{2} \tag{12}$$

where $t_{\text{start,pred}}$ and $t_{\text{end,pred}}$ denote the predicted start and end times of an event, while $t_{\text{start,true}}$ and $t_{\text{end,true}}$ denote the corresponding manually annotated start and end times. A smaller *BE* indicates more accurate temporal localisation of the detected event boundary. *BE* was calculated only for correctly matched event pairs and was reported in seconds. Macro-average precision, recall, F1 score, and *BE* were calculated over the four evaluated event categories.

4.2 Validation of Atomic Flight Event Detection Performance

This section evaluates the effectiveness of the atomic flight event extraction module from five aspects: comparison with baseline methods, separate dataset evaluation, statistical validation, ablation studies, and parameter sensitivity analysis.

4.2.1 Comparison with Baseline Methods

To evaluate the effectiveness of the proposed method, it was compared with three representative time-series baselines, namely LSTM-Cls, MiniRocket, and Transformer-Cls. These baselines represent recurrent,

kernel-based, and self-attention-based sequence modelling paradigms, respectively. Based on the fair comparison protocol described in Section 4.1.3, all methods used the same preprocessed input features, window length, mission-level data split, and event-level evaluation metrics.

Since LSTM-Cls, MiniRocket, and Transformer-Cls produce window-level classification results, their outputs were converted into event intervals before evaluation. Specifically, consecutive windows with the same predicted non-Normal label were merged into candidate event intervals, short intervals were removed using the same minimum-duration constraint, and adjacent intervals of the same type were further merged when their temporal gap was shorter than the predefined merging threshold. Therefore, all methods were evaluated under the same interval-level event detection protocol.

Different from the supervised sequence classification baselines, the proposed method detects atomic flight events through local statistical features, adaptive dynamic boundaries, and event-specific state constraints. This design preserves the interpretability of rule-based detection while improving robustness to local noise and scene variation.

As shown in Table 3, the proposed method achieves better F1 scores than LSTM-Cls, MiniRocket, and Transformer-Cls across the four evaluated event categories, namely Takeoff, Turn, Hover, and Descent. For Takeoff and Descent, the proposed method achieves F1 scores of 0.89 and 0.86, respectively, indicating that altitude-transition events can be detected reliably using the regression-based climb-rate feature. For Turn events, the F1 score reaches 0.80, which is higher than those of LSTM-Cls, MiniRocket, and Transformer-Cls. This improvement is mainly attributed to the heading wrap-around correction and the use of heading-change-rate features. For Hover events, the F1 score increases from 0.61 for LSTM-Cls to 0.78 for the proposed method, showing the effectiveness of the state constraints for sustained low-motion behaviour.

Table 3: Detection performance comparison of different methods on typical flight events.

Event Type	Method	Precision	Recall	F1 Score	BE/s
Takeoff	LSTM-Cls	0.82	0.75	0.78	1.20
	MiniRocket	0.84	0.80	0.82	0.95
	Transformer-Cls	0.86	0.83	0.84	0.90
	Ours	0.87	0.91	0.89	0.65
Turn	LSTM-Cls	0.70	0.62	0.66	2.10
	MiniRocket	0.73	0.68	0.70	1.85
	Transformer-Cls	0.78	0.72	0.75	1.60
	Ours	0.81	0.79	0.80	1.10
Hover	LSTM-Cls	0.65	0.58	0.61	3.00
	MiniRocket	0.68	0.62	0.65	2.60
	Transformer-Cls	0.72	0.66	0.69	2.30
	Ours	0.80	0.76	0.78	1.70
Descent	LSTM-Cls	0.79	0.73	0.76	1.50
	MiniRocket	0.81	0.77	0.79	1.30
	Transformer-Cls	0.84	0.80	0.82	1.10
	Ours	0.88	0.85	0.86	0.80

Overall, the proposed method achieves a macro-average F1 score of approximately 0.83 over the four evaluated event categories. These results indicate that the proposed method is suitable for behaviour-oriented

event extraction from continuous UAV flight logs, especially when both interpretability and event-boundary localisation are required.

4.2.2 Separate Evaluation on MUN-FRL and EuRoC MAV

To further address the distributional differences between MUN-FRL and EuRoC MAV, the proposed method was evaluated separately on the two datasets in addition to the hybrid dataset. The same event definitions, preprocessing procedure, mission-level split, and event-level evaluation protocol were used for all datasets. The separate evaluation results are reported in Table 4.

Table 4: Separate evaluation on MUN-FRL, EuRoC MAV, and Hybrid.

Dataset	Event	Precision	Recall	F1	BE/s
MUN-FRL	Takeoff	0.88	0.92	0.90	0.62
MUN-FRL	Turn	0.83	0.81	0.82	1.05
MUN-FRL	Hover	0.81	0.77	0.79	1.62
MUN-FRL	Descent	0.89	0.86	0.87	0.78
MUN-FRL	Macro-average	0.85	0.84	0.85	1.02
EuRoC MAV	Takeoff	0.86	0.89	0.87	1.02
EuRoC MAV	Turn	0.79	0.77	0.78	1.24
EuRoC MAV	Hover	0.77	0.73	0.75	1.88
EuRoC MAV	Descent	0.86	0.82	0.84	0.93
EuRoC MAV	Macro-average	0.82	0.80	0.81	1.27
Hybrid	Macro-average	0.84	0.83	0.83	1.06

The results show that the proposed method maintains stable performance on both datasets. The performance on MUN-FRL is slightly higher than that on EuRoC MAV, which may be related to the clearer altitude and velocity transitions in several outdoor flight segments. Hover obtains relatively lower F1 scores and larger boundary errors because its boundaries are less distinct than those of Takeoff and Descent.

4.2.3 Statistical Validation and Significance Analysis

To further evaluate the statistical reliability of the observed performance improvement, a paired mission-level bootstrap analysis was conducted on the test set. In each bootstrap iteration, test missions were sampled with replacement, and the same resampled missions were used to evaluate both the proposed method and Transformer-Cls, which achieved the strongest overall performance among the baseline models. The macro-average F1 score and macro-average boundary error were then recalculated according to the event-level matching criterion described in Section 4.1.4. The bootstrap procedure was repeated 1000 times.

For each method, the mean, standard deviation, and 95% confidence interval were estimated from the corresponding bootstrap distribution. In addition, the paired performance differences were calculated according to Eqs. (13) and (14):

$$\Delta F1_{macro} = F1_{macro}^{Ours} - F1_{macro}^{Transformer} \tag{13}$$

$$\Delta BE_{macro} = BE_{macro}^{Ours} - BE_{macro}^{Transformer} \tag{14}$$

The 95% confidence intervals of the paired differences were estimated using the 2.5th and 97.5th percentiles of their bootstrap distributions. Two-sided bootstrap p-values were also calculated to determine whether the paired differences were significantly different from zero.

As shown in Table 5, Transformer-Cls achieved an $F1_{macro}$ of 0.78 ± 0.02, with a 95% confidence interval of [0.73, 0.82], whereas the proposed method achieved 0.83 ± 0.02, with a 95% confidence interval of [0.79, 0.86]. The paired improvement in $F1_{macro}$ was 0.05, with a 95% confidence interval of [0.02, 0.08] and a two-sided bootstrap p-value of 0.018. Because the confidence interval of the paired difference was entirely above zero, the improvement in event-detection performance was statistically significant at the 0.05 level.

Table 5: Bootstrap statistical validation on the test missions.

Method or Comparison	Mean	Bootstrap SD	95% Confidence Interval	p-Value
Transformer-Cls ($F1_{macro}$)	0.78	0.02	[0.73, 0.82]	
Ours ($F1_{macro}$)	0.83	0.02	[0.79, 0.86]	
$\Delta F1_{macro}$	0.05	0.02	[0.02, 0.08]	0.018
Transformer-Cls (BE_{macro})	1.48	0.15	[1.22, 1.82]	
Ours (BE_{macro})	1.06	0.15	[0.86, 1.32]	
ΔBE_{macro}	−0.42	0.13	[−0.65, −0.18]	0.012

For event-boundary localisation, Transformer-Cls achieved a BE_{macro} of 1.48 ± 0.15 s, whereas the proposed method achieved 1.06 ± 0.15 s. The paired difference was −0.42 s, with a 95% confidence interval of [−0.65, −0.18] s and a two-sided bootstrap p-value of 0.012. The negative difference indicates that the proposed method significantly reduced temporal-boundary error compared with the strongest baseline.

Overall, the proposed method consistently achieved a higher macro-average F1 score and a lower macro-average boundary error under mission-level bootstrap resampling. The paired confidence intervals for both performance differences excluded zero, providing statistical evidence that the observed improvements were not solely caused by the particular composition of the current test set. Nevertheless, the statistical conclusions remain limited by the number and diversity of the available test missions, and further validation on larger cross-platform datasets is still required.

4.2.4 Ablation Studies

To analyse the effects of key design components, namely the adaptive boundary mechanism and feature smoothing, on detection performance, two ablation variants were constructed:

Ours-w/o-Adapt: the adaptive boundary mechanism is removed and a fixed $k\sigma$ boundary is used for classification;

Ours-w/o-Smooth: the adaptive boundary is applied directly to the raw features without smoothing.

As shown in Table 6, removing feature smoothing (Ours-w/o-Smooth) reduces the F1 score from 0.83 to 0.76, while increasing the boundary error to 2.10 s. This result indicates that high-frequency noise significantly affects the stability of event boundaries. When adaptive boundaries are removed (Ours-w/o-Adapt), although the precision of some events improves slightly, the inability to adjust thresholds dynamically across different manoeuvring phases and complex scenarios leads to more false positives and false negatives, thereby reducing the overall F1 score and increasing the boundary error. By contrast, the full model (Ours) achieves a better balance between precision and recall, demonstrating the effectiveness of both adaptive boundaries and feature smoothing in improving event detection performance.

Table 6: Ablation results of different variants.

Method	Precision_macro	Recall_macro	F1_macro	BE_macro/s
Ours-w/o-Adapt	0.80	0.77	0.78	1.90
Ours-w/o-Smooth	0.78	0.74	0.76	2.10
Ours	0.84	0.83	0.83	1.06

Although the ablation results indicate that feature smoothing and adaptive decision boundaries help reduce sensitivity to fluctuations in the evaluated flight logs, controlled robustness experiments involving synthetic Gaussian, altitude, or heading perturbations were not conducted. Therefore, the present results should be interpreted as evidence of stability under the naturally occurring noise and variation in the evaluated datasets, rather than general robustness to arbitrary sensor-noise conditions.

4.2.5 Parameter Sensitivity Analysis

To examine whether the proposed method is overly dependent on specific parameter settings, a parameter sensitivity analysis was conducted. The tested parameters included the sliding-window length K and the adaptive boundary coefficients α and β. When one parameter was varied, the remaining parameters were fixed at their default values. The default setting was a sliding-window length of K = 40 samples, step size = 10 samples, $\alpha = 0.7$, $\beta = 0.3$, and a minimum duration threshold of 1.0 s. Table 7 presents the sensitivity analysis results for different values of the sliding-window length K.

Table 7: Sensitivity analysis of window length K.

K	Precision_macro	Recall_macro	F1_macro	BE_macro/s
20	0.80	0.76	0.77	1.72
40	0.84	0.83	0.83	1.06
60	0.83	0.81	0.82	1.24
80	0.81	0.77	0.80	1.55

When K is too small, the local statistical features become sensitive to short-term noise, resulting in unstable event boundaries. When K is too large, short-duration manoeuvres may be smoothed out, leading to missed detections and increased boundary error. The results show that the method obtains stable performance around K = 40. Fig. 5 shows the sensitivity of $F1_{macro}$ to different combinations of the adaptive boundary coefficients α and β.

The highest $F1_{macro}$ value is obtained when $\alpha = 0.7$ and $\beta = 0.3$. In most tested parameter combinations, the $F1_{macro}$ value remains within a relatively narrow range, indicating that the proposed method is not dependent on a single parameter setting.

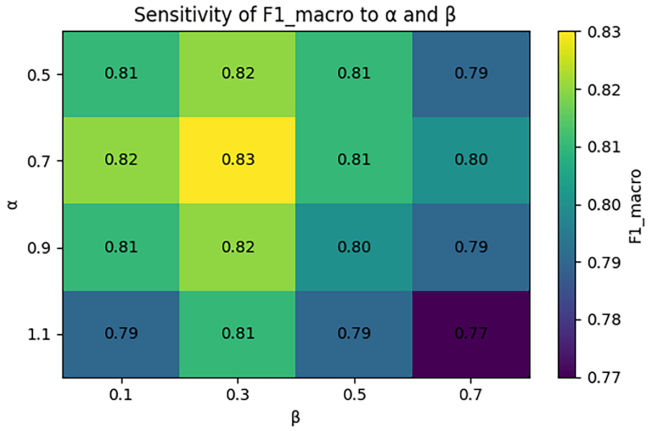

Figure 5: Sensitivity of F1_macro to α and β.

For the adaptive boundary coefficients, smaller α and β values make the detector more sensitive and may improve recall, but they may also introduce more false positives. Larger values make the detector more conservative, which may improve precision but reduce recall. Overall, the proposed method remains stable within a moderate parameter range. It should be noted that α and β are not assumed to be universal constants; they were selected on the validation set in this study.

4.3 Construction and Application Validation of the Temporal Knowledge Graph

4.3.1 Analysis of Graph Scale and Structure

Table 8 summarises the composition of the constructed temporal knowledge graph. AtomicEvent and ComplexBehavior entities account for the largest proportion, indicating that the graph is centred on event-level behavioural representation and reasoning. Temporal relations such as NextEvent and Follow further connect atomic events into ordered behavioural chains, while task- and region-related relations provide mission and spatial context. A representative local subgraph is shown in Fig. 6.

Table 8: Entity distribution in the temporal knowledge graph.

Entity Type	Count	Proportion
UAV	3	0.05%
Task	6	0.11%
AtomicEvent	3800	67.3%
ComplexBehavior	1200	21.2%
GeoRegion	80	1.4%
GeoFence	60	1.1%
Other entities	500	8.9%
Total entities	5649	

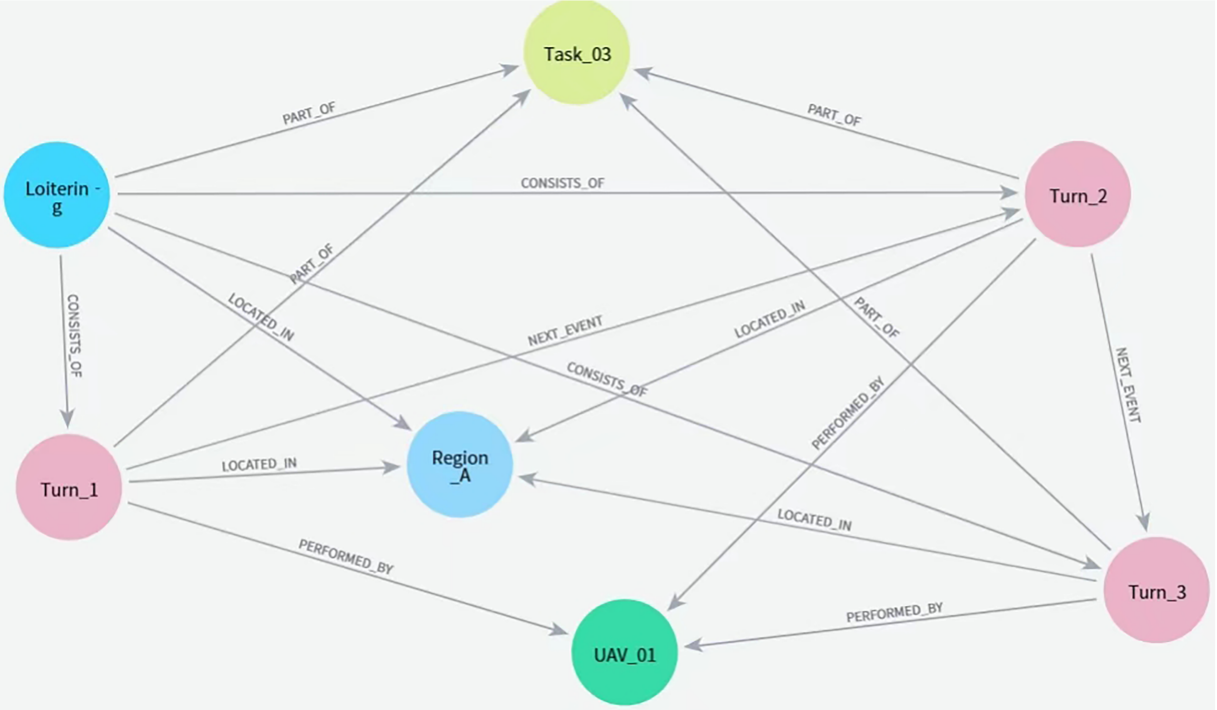

Figure 6: Partial visualisation of the temporal knowledge graph.

The 2980 events reported in Section 4.1.2 refer to the manually annotated ground-truth events used for event-detection evaluation. In contrast, the 3800 AtomicEvent entities in Table 8 include all atomic-event instances generated and stored during knowledge graph construction; therefore, the two quantities represent different data scopes.

4.3.2 Validation of Typical Behaviour Analysis and Semantic Understanding Capabilities

To evaluate the performance of the temporal knowledge graph in semantic query scenarios, three typical queries were designed:

Q1: Analyse the spatial distribution characteristics of UAV loitering behaviour in a specified mission, and retrieve relevant loitering instances and their associated locations.

Q2: Analyse the phased behavioural evolution of UAV flight processes within a given time interval, and retrieve the "Takeoff–Hover–Descent" event sequence.

Q3: Analyse potential dangerous low-altitude approach behaviours during UAV flight, and retrieve instances of such behaviours identified through inference along with their triggering conditions.

Queries Q1–Q3 were executed on the Neo4j graph database. Each query was run 50 times, and the average response time was calculated. Query accuracy was further evaluated by comparing the returned results with manually checked query answers. The results are shown in Table 9.

Table 9: Semantic query performance.

Query ID	Query Description	Query Accuracy	Average Response Time/ms
Q1	Spatial distribution analysis of loitering behaviour in a specified mission	0.97	8.5
Q2	Analysis of staged behavioural evolution during the flight process	0.93	18.7
Q3	Analysis of potential dangerous low-altitude approach behaviour	0.90	27.3

The experimental results show that, for single-behaviour queries and scenarios with simple spatial constraints (Q1), the response time remains within 8.5 ms. For event-chain retrieval and behavioural pattern retrieval involving multiple path matches (Q2 and Q3), the response time also remains at the tens-of-milliseconds level. The query accuracy values of 0.97, 0.93, and 0.90 further indicate that the graph-based retrieval results are largely consistent with the manually checked query answers. Considering the temporal granularity of the flight logs and the fact that most application scenarios involve offline analysis and near-real-time auditing, this query performance provides initial evidence for the feasibility of graph-based semantic retrieval in offline analysis and near-real-time auditing scenarios.

5 Conclusions and Future Work

This paper proposes an event-centred knowledge graph construction framework for UAV flight behaviour understanding. The framework integrates interpretable atomic-event detection, temporal knowledge modelling, and ontology-based composite-behaviour reasoning, enabling continuous multi-source flight logs to be transformed into structured, queryable, and semantically interpretable behavioural knowledge. In contrast to approaches that represent flight logs only as raw time series, trajectory points, or window-level labels, the proposed method uses atomic flight events as intermediate semantic units to connect low-level sensor observations with high-level behaviour representation and reasoning.

For atomic-event detection, local statistical features, including smoothed velocity, robust climb rate, and heading-change rate, are extracted from symmetric sliding windows. Adaptive statistical decision boundaries and event-specific state constraints are then applied to identify four representative event categories: Takeoff, Turn, Hover, and Descent. Under the same input features, mission-level data split, post-processing procedure, and event-level evaluation protocol, the proposed method achieves a macro-average F1 score of 0.83. The reported F1 scores are higher than those of the compared LSTM-Cls, MiniRocket, and Transformer-Cls baselines for all four event categories. Separate evaluations on the MUN-FRL and EuRoC MAV datasets further indicate that the proposed method maintains relatively stable performance across the two experimental settings. In addition, mission-level bootstrap validation provides further evidence for the stability of the observed improvement under the current test setting, although the results are not intended as a universal statistical guarantee.

The detected atomic events are further organised into a temporal flight behaviour knowledge graph that represents UAV platforms, mission instances, geographical regions, temporal event chains, candidate behaviours, and inferred composite behaviours within a unified semantic structure. Candidate behaviour

instances are connected to their supporting atomic events through the consistsOf relation and are subsequently classified using SWRL-style rules based on event type, temporal order, spatial relation, altitude, displacement, and time-gap constraints. The constructed graph contains 5649 entities. For the three representative semantic-query scenarios evaluated in this study, the query accuracy ranges from 0.90 to 0.97, while the average response time ranges from 8.5 to 27.3 ms. These results provide preliminary evidence that the proposed event-centred representation can support the considered semantic-query tasks efficiently under the current graph size and experimental configuration.

Several limitations remain. First, although two public datasets are used, the experimental validation is still limited in terms of data volume, UAV platform diversity, mission coverage, environmental conditions, and graph instances. The event definitions and adaptive parameters adopted in this study are not universal constants. When the framework is deployed on a new UAV platform, the detector parameters should be recalibrated using a small platform-specific validation set. For multirotor and helicopter platforms, behaviours such as Hover remain applicable, although differences in velocity range, vibration characteristics, and altitude fluctuation require separate parameter calibration. For fixed-wing UAVs, Hover is generally not applicable and should be replaced by platform-relevant behaviours such as holding, circling, or loitering.

Second, the initial event annotations were produced by one researcher and subsequently manually reviewed according to predefined event definitions and boundary criteria. Because the labels were not independently generated by multiple annotators, a formal inter-annotator agreement statistic, such as Cohen's kappa, was not available. Although questionable labels and temporal boundaries were re-examined using synchronised altitude, ground-speed, heading, climb-rate, and trajectory profiles, an independent multi-annotator protocol would provide a stronger assessment of annotation reliability.

Third, the robustness of the event detector was examined indirectly through separate dataset evaluations, ablation studies, and parameter-sensitivity analysis. Systematic controlled experiments involving synthetic sensor perturbations were not conducted in the current study. Therefore, the present results do not fully characterise the effects of Gaussian noise, sensor drift, temporally correlated disturbances, packet loss, or sensor failures. The conclusions regarding robustness should consequently be understood within the scope of the current datasets and experimental conditions.

Fourth, the reasoning module mainly relies on manually designed deterministic SWRL rules. This design provides explicit semantics, interpretable conditions, and traceable reasoning results, but it also requires domain expertise and may become difficult to maintain as the number of behaviour types and constraints increases. Deterministic rules also have limited ability to represent uncertainty arising from noisy sensor observations, ambiguous event boundaries, incomplete information, or previously unseen behaviour patterns. In addition, the current scalability evaluation is based on a graph containing 5600 entities and three predefined query scenarios. Larger graph instances and more diverse query workloads are required before the framework can be considered suitable for large-scale operational deployment.

Future work will collect additional real-world flight logs from multirotor, fixed-wing, and helicopter platforms and investigate platform-specific parameter calibration and event-vocabulary adaptation. An independent multi-annotator protocol will be adopted to report formal agreement statistics and improve the reliability of the ground-truth labels. Controlled robustness experiments will also be conducted under different types and intensities of sensor perturbations. Larger knowledge graphs will be evaluated in terms of storage consumption, graph-construction time, reasoning overhead, memory usage, and percentile query latency. Furthermore, automatic rule learning, probabilistic reasoning, fuzzy rule matching, uncertainty-aware knowledge graphs, graph representation learning, and data-driven rule mining will be investigated to improve the adaptability, automation, and practical applicability of UAV flight behaviour understanding.

Acknowledgement: The authors would like to thank all the researchers who helped to improve the quality of the manuscript.

Funding Statement: This research was funded by the Artificial Intelligence Promotes Paradigm Reform in Scientific Research and Empowers Discipline Advancement Plan, grant number 25AZ015.

Author Contributions: The authors confirm contribution to the paper as follows: conceptualization, Tian Liu and Xichao Wang; methodology, Tian Liu; software, Tian Liu; validation, Tian Liu, Xichao Wang, Jun Wang, Song Gao, Hang Gao, Yuxi Liu and Yitao Zhuang; formal analysis, Tian Liu; investigation, Tian Liu; resources, Xichao Wang, Jun Wang, Song Gao, Hang Gao, Yuxi Liu and Yitao Zhuang; data curation, Tian Liu; writing—original draft preparation, Tian Liu; writing—review and editing, Tian Liu, Xichao Wang, Jun Wang, Song Gao, Hang Gao, Yuxi Liu and Yitao Zhuang; visualization, Tian Liu; supervision, Xichao Wang; project administration, Xichao Wang; funding acquisition, Xichao Wang. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The MUN-FRL and EuRoC MAV datasets used in this study are publicly available from their original sources. The ontology schema, event definitions, annotation guidelines, representative SWRL rules, parameter settings, and query templates are available from the corresponding author upon reasonable request. The complete source code is not currently available in a public repository because the implementation is still being consolidated, documented, and separated from project-specific configurations. The authors will provide the core implementation materials required to reproduce the reported experiments upon reasonable request.

Ethics Approval: Not applicable.

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

References

1. Keipour A, Mousaei M, Scherer S. ALFA: a dataset for UAV fault and anomaly detection. Int J Robot Res. 2021;40 (2–3):515–20. doi:10.1177/0278364920966642.
2. Yang L, Li S, Li C, Zhang A, Zhang X. A survey of unmanned aerial vehicle flight data anomaly detection: technologies, applications, and future directions. Sci China Technol Sci. 2023;66(4):901–19. doi:10.1007/s11431-022-2213-8.
3. Parent C, Spaccapietra S, Renso C, Andrienko G, Andrienko N, Bogorny V, et al. Semantic trajectories modeling and analysis. ACM Comput Surv. 2013;45(4):1–32. doi:10.1145/2501654.2501656.
4. Yang L, Li S, Li C, Zhu C, Zhang A, Liang G. Data-driven unsupervised anomaly detection and recovery of unmanned aerial vehicle flight data based on spatiotemporal correlation. Sci China Technol Sci. 2023;66(5):1304–16. doi:10.1007/s11431-022-2312-8.
5. Baglioni M, Macedo J, Renso C, Wachowicz M. An ontology-based approach for the semantic modelling and reasoning on trajectories. In: Advances in conceptual modeling—challenges and opportunities. Berlin/Heidelberg, Germany: Springer; 2008. p. 344–53.
6. Yan Z, Chakraborty D, Parent C, Spaccapietra S, Aberer K. SeMiTri: a framework for semantic annotation of heterogeneous trajectories. In: Proceedings of the 14th International Conference on Extending Database Technology; 2011 Mar 21–24; Uppsala, Sweden. p. 259–70.
7. Ghasemi A, Ghaffari A, Derakhshanfard N, Ibrahimoğlu N, Pakmehr A. Anomaly detection in unmanned aerial vehicles flight data: a survey. Ad Hoc Netw. 2025;178:103989. doi:10.1016/j.adhoc.2025.103989.
8. Aminikhanghahi S, Cook DJ. A survey of methods for time series change point detection. Knowl Inf Syst. 2017;51(2):339–67. doi:10.1007/s10115-016-0987-z.
9. Kawahara Y, Sugiyama M. Change-point detection in time-series data by direct density-ratio estimation. In: Proceedings of the 2009 SIAM International Conference on Data Mining; 2009 Apr 30–May 2; Sparks, Nevada. p. 389–400.

10. Lu H, Wang Z, Shi Y. Unmanned aerial vehicle flight data anomaly detection based on multirate-aware LSTM. IEEE Trans Instrum Meas. 2024;73:3526713. doi:10.1109/TIM.2024.3440379.
11. Deng H, Lu Y, Yang T, Liu Z, Chen J. Unmanned aerial vehicles anomaly detection model based on sensor information fusion and hybrid multimodal neural network. Eng Appl Artif Intell. 2024;132:107961. doi:10.1016/j.engappai.2024.107961.
12. Ahn H, Chung S. Deep learning-based anomaly detection for individual drone vehicles performing swarm missions. Expert Syst Appl. 2024;244:122869. doi:10.1016/j.eswa.2023.122869.
13. Do Yoo J, Kim GM, Song MG, Kim HK. MeNU: memorizing normality for UAV anomaly detection with a few sensor values. Comput Secur. 2025;150:104248. doi:10.1016/j.cose.2024.104248.
14. Li J, Zhou Y, Dong Z. TSAE-UAV: a time-sense autoencoder for unsupervised anomaly detection of UAV based on flight data. Aerosp Sci Technol. 2026;169:111431. doi:10.1016/j.ast.2025.111431.
15. Yang J, Tang D, Yu J, Zhang J, Liu H. Explaining anomalous events in flight data of UAV with deep attention-based multi-instance learning. IEEE Trans Veh Technol. 2024;73(1):107–19. doi:10.1109/TVT.2023.3301678.
16. Wang Y, Wang D, Wang J. A data driven approach for detection and isolation of anomalies in a group of UAVs. Chin J Aeronaut. 2015;28(1):206–13. doi:10.1016/j.cja.2014.12.003.
17. Alos AM, Dahrouj Z, Dakkak M. A novel technique to assess UAV behavior using PCA-based anomaly detection algorithm. Int J Mech Eng Robot Res. 2020;9(5):721–6. doi:10.18178/ijmerr.9.5.721-726.
18. Pang W, Gu W, Li H. Ontology-based task planning for autonomous unmanned system: framework and principle. J Phys Conf Ser. 2022;2253(1):012018. doi:10.1088/1742-6596/2253/1/012018.
19. Duan X, Chai R, Zhang S, Liang C. A knowledge graph for UAV mission planning systems. In: Communications and networking. Cham, Switzerland: Springer; 2024. p. 97–111.
20. Ge Y, Zhang S, Cai Y, Lu T, Wang H, Hui X, et al. Ontology based autonomous robot task processing framework. Front Neurorobot. 2024;18:1401075. doi:10.3389/fnbot.2024.1401075.
21. Alvares LO, Bogorny V, Kuijpers B, de Macedo JAF, Moelans B, Vaisman A. A model for enriching trajectories with semantic geographical information. In: Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems; 2007 Nov 7–9; Seattle, WA, USA.
22. Wang B, Wang Z, Liu L, Liu D, Peng X. Data-driven anomaly detection for UAV sensor data based on deep learning prediction model. In: Proceedings of the 2019 Prognostics and System Health Management Conference (PHM-Paris); 2019 May 2–5; Paris, France. p. 286–90.
23. Zhong J, Zhang Y, Wang J, Luo C, Miao Q. Unmanned aerial vehicle flight data anomaly detection and recovery prediction based on spatio-temporal correlation. IEEE Trans Reliab. 2022;71(1):457–68. doi:10.1109/TR.2021.3134369.
24. Yang L, Li S, Li C, Zhu C. Data-driven multivariate regression-based anomaly detection and recovery of unmanned aerial vehicle flight data. J Comput Des Eng. 2024;11(2):176–93. doi:10.1093/jcde/qwae023.
25. Zhong J, Zhang H, Miao Q. Flight anomaly detection and localization based on flight data fusion and random channel masking. Appl Soft Comput. 2025;174:113023. doi:10.1016/j.asoc.2025.113023.
26. Guralnik V, Srivastava J. Event detection from time series data. In: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 1999 Aug 15–18; San Diego, CA, USA. p. 33–42.
27. Yang T, Chen J, Deng H, Lu Y. UAV abnormal state detection model based on timestamp slice and multi-separable CNN. Electronics. 2023;12(6):1299. doi:10.3390/electronics12061299.
28. Ermshaus A, Schäfer P, Leser U. ClaSP: parameter-free time series segmentation. Data Min Knowl Discov. 2023;37(3):1262–300. doi:10.1007/s10618-023-00923-x.
29. Ermshaus A, Schäfer P, Leser U. Raising the ClaSS of streaming time series segmentation. Proc VLDB Endow. 2024;17(8):1953–66. doi:10.14778/3659437.3659450.
30. Himberg J, Korpiaho K, Mannila H, Tikanmaki J, Toivonen HTT. Time series segmentation for context recognition in mobile devices. In: Proceedings 2001 IEEE International Conference on Data Mining; 2001 Nov 29–Dec 2; San Jose, CA, USA. p. 203–10.

31. Fan Y, Mi B, Sun Y, Yin L. Research on the intelligent construction of UAV knowledge graph based on attentive semantic representation. Drones. 2023;7(6):360. doi:10.3390/drones7060360.
32. Xiao A, Yan W, Zhang X, Liu Y, Zhang H, Liu Q. Multi-domain fusion for cargo UAV fault diagnosis knowledge graph construction. Auton Intell Syst. 2024;4(1):10. doi:10.1007/s43684-024-00072-y.
33. Kotis K, Angelis S, Moraitou E, Kopsachilis V, Papadopoulou EE, Soulakellis N, et al. A KG-based integrated UAV approach for engineering semantic trajectories in the cultural heritage documentation domain. Remote Sens. 2023;15(3):821. doi:10.3390/rs15030821.
34. Moraitou E, Angelis S, Kotis K, Caridakis G, Papadopoulou EE, Soulakellis N. Towards engineering drones' semantic trajectories as knowledge graphs. In: Proceedings of the 5th International Workshop on Geospatial Linked Data (GeoLD 2022); 2022 May 30; Hersonissos, Greece. p. 31–6.
35. Soularidis A, Kotis K. Semantic modeling and reconstruction of drones' trajectories. In: The semantic web: ESWC 2022 satellite events. Cham, Switzerland: Springer; 2022. p. 158–62.
36. Vouros GA, Santipantakis GM, Doulkeridis C, Vlachou A, Andrienko G, Andrienko N, et al. The datAcron ontology for the specification of semantic trajectories. J Data Semant. 2019;8(4):235–62. doi:10.1007/s13740-019-00108-0.
37. Santipantakis MG, Doulkeridis C, Vouros AG. An ontology for representing and querying semantic trajectories in the maritime domain. In: Advances in databases and information systems. Cham, Switzerland: Springer; 2023. p. 224–37.
38. Cai G, Lee K, Lee I. Mining semantic trajectory patterns from geo-tagged data. J Comput Sci Technol. 2018;33(4):849–62. doi:10.1007/s11390-018-1860-1.