A Method for ATCO Intent Assessment

Deliverable ID:	**D2.16**
Project acronym:	**AWARE**
Grant:	**101167442**
Call:	**HORIZON-SESAR-2023-DES-ER-02**
Topic:	**HORIZON-SESAR-2023-DES-ER2-WA2-4**
Consortium coordinator:	**FTTS**
Edition date:	**16 February 2026**
Edition:	**01.02**
Status:	**Official**
Classification:	**PU**

Abstract

This deliverable presents the development of a data-driven method for Air Traffic Controller intent assessment within AWARE. The proposed approach combines eye-tracking, interaction with the Human machine Interface, and contextual simulator data to infer controller intent through expert-labelled training and hierarchical machine learning models. The results are promising and demonstrate that multimodal behavioural patterns can indicate operational intent to some extent. However, they also highlight the need of further development and especially an improved controller intent dataset to form a solid foundation for integration into the Artificial Situational Awareness system and future adaptive human–machine collaboration in air traffic management.

Co-funded by the European Union

Authoring & approval

Author(s) of the document

Organisation name	Date
Timothé Krauth/ZHAW	27.10.2025
Daniel Regado/ZHAW	22.01.2026
Celina Vetter/ZHAW	29.10.2025

Reviewed by

Organisation name	Date
Ruth Häusler/ZHAW	13 Feb 2026
Tomislav Radišić/FTTS	16 Feb 2026

Approved for submission to the SESAR 3 JU by

Organisation name	Date
Tomislav Radišić/FTTS	16 Feb 2026

Rejected by[1]

Organisation name	Date

Document history

Edition	Date	Status	Company Author	Justification
00.01	27.10.2025	Initial Draft	T. Krauth	New Document
00.02	29.10.2025	Draft	C. Vetter	Rephrasing, template adaptions
00.03	16.12.2025	Draft	T. Krauth	Document modification
01.01	12.01.2026	First version	T. Krauth	Integrating new results
01.02	22.01.2026	Final version	T. Krauth	Reviewing

[1] Representatives of the beneficiaries involved in the project.

AWARE

ACHIEVING HUMAN-MACHINE COLLABORATION WITH ARTIFICIAL SITUATIONALAWARENESS

AWARE

This document is part of a project that has received funding from the SESAR 3 Joint Undertaking under grant agreement No 101167442 under European Union's Horizon Europe research and innovation programme.

1 Table of Contents

List of Figures

List of Tables

2 Introduction

Air Traffic Management (ATM) operations rely heavily on the expertise, anticipation, and situational awareness of Air Traffic Controllers (ATCOs). Their ability to interpret evolving traffic situations, predict future aircraft trajectories, and take proactive decisions is central to maintaining safety and efficiency in increasingly complex airspaces. Understanding and assessing ATCO intent – that is, the cognitive and operational goals underlying their observed actions – is therefore a key enabler for next-generation support systems and human–machine teaming concepts.

In contemporary ATM environments, increasing traffic density, emerging aircraft types, and growing system complexity place substantial cognitive demands on ATCOs. To maintain safety and performance under such conditions, future ATM systems must not only understand the traffic situation but also the intentions and goals of the human operator. This human-aware capability is essential for adaptive automation and mutual situational understanding between human and AI teammates.

The Artificial Situational Awareness (ASA) system lies at the heart of AWARE's approach. It semantically represents the traffic situation using a knowledge-graph-based reasoning engine, allowing the system to "understand" the current context. However, situational understanding alone is insufficient for meaningful collaboration: the system must also infer what the controller is trying to achieve. Recognizing ATCO intent, the mental and operational goal guiding current actions, is therefore a cornerstone for achieving genuine human–AI teamwork.

AWARE contributes to this vision by developing a data-driven method for ATCO Intent Assessment based on multimodal behavioural data. Specifically, this work builds upon synchronised eye-tracking, interaction with the Human Machine Interaction (HMI) logs, and simulator context data to infer the controller's focus, activity, and underlying intent with respect to the traffic situation. The proposed method integrates expertise from human factors specialists, operational ATCOs, and data scientists to establish a bridge between human behaviour interpretation and computational modelling.

This necessity led to Activity 2.2: Attention Tracking and Intent Analysis, which focuses on developing a comprehensive method for assessing ATCO intent based on multimodal behavioural data. Within this activity, Task 2.2.3 – Development of a Method for ATCO Intent Assessment specifically targets the design of algorithms and analytical frameworks that link eye-tracking and Human-Machine Interaction (HMI) data to the cognitive and operational intent of ATCOs during simulated air traffic control tasks

The method leverages both expert knowledge and data-driven techniques. Subject-matter experts (ATCOs) analyse recorded simulation sessions to label observed intent and strategies. These expert annotations provide the ground truth required for model development. Researchers then extract gaze and HMI features, identify behavioural patterns, and employ both rule-based and machine learning methods to predict ATCO intent. The integration of these approaches ensures robustness, interpretability, and adaptability to complex and dynamic air traffic situations.

By contextualizing attention and HMI data within the semantic representation of the traffic situation provided by the ASA system, the project enables a new level of machine understanding of human actions. The resulting framework supports retrospective analysis of controller strategies and also lays the foundation for future real-time intent recognition modules capable of enhancing adaptive automation, decision support, and safety monitoring systems.

2.1 Objectives

The main objective of this deliverable is to present a validated method for ATCO intent assessment, addressing the following specific aims:

1. **Link observed behaviour to cognitive intent**: Define a taxonomy of ATCO intents based on operational goals and validate it through expert labelling of simulation recordings. This creates the foundation for supervised learning and rule derivation.
2. **Identify patterns and predictive features**: Determine and compute features based on eye-tracking data and HMI data that describe behavioural patterns, such as such as gaze allocation, interaction frequency, and spatial focus. Combine expert-driven insights with data-driven discovery to enhance interpretability.
3. **Develop and evaluate machine learning models**: Implement and test machine learning methods capable of inferring ATCO intent from the computed predictive features. Evaluate models against expert labels to assess their accuracy, reliability, and operational plausibility.
4. **Enable intent-based human–machine collaboration**: Provide a technical and conceptual basis for integrating intent assessment into the Artificial Situational Awareness System. This will enable future systems to dynamically adapt support levels, detect degraded situational awareness, and maintain ATCOs in the operational loop.

2.2 Expected Outcome

By fulfilling the objectives, this work will deliver:

- A methodology and proof-of-concept for intent detection based on eye-tracking and HMI data.
- Validated mappings between gaze/HMI behaviour and operational intent.
- A foundation for real-time integration of intent recognition into future AI Assistant Applications.

Ultimately, this deliverable will contribute to the evolution of adaptive, human-aware ATM systems, ensuring that automation complements rather than replaces the controller's expertise and judgment.

3 Conceptual Framework

Understanding ATCO's intent is central to enabling human-aware automation and effective human–machine collaboration. In the context of this work, intent is defined on two complementary levels:

- **Operational intent** refers to the controller's explicit goals and planned actions within the air traffic situation, such as resolving a conflict, sequencing arrivals, or coordinating a handover. It represents what the ATCO aims to achieve in managing traffic. The ATCO's task list is described in D2.14 – AWARE Validation Scenarios.
- **Cognitive intent** captures the underlying mental processes that guide perception, attention, and decision-making, why and how the controller chooses a specific course of action. This includes prioritisation, anticipation, and situation assessment, reflecting the controller's internal representation of the problem space. Cognitive intent also reflects an ATCO's awareness of their workload and self-regulatory strategies to maintain workload at an acceptable level.

From a HMI perspective, ATCO intent manifests through measurable behavioural indicators. Gaze behaviour reveals information acquisition and prioritisation patterns, indicating which elements of the traffic situation are currently being monitored or assessed. HMI interactions (e.g. mouse movements, measurement tools, pop-up windows, etc...) reflect the transition from perception to action, showing how the controller translates cognitive goals into operational steps. All together, these multimodal signals provide a trace of intent.

The design of the ATCO intent assessment method follows two guiding principles:

1. **Transparency**: ensuring that both human experts and machine components can understand how inferred intent relates to observed behaviour and situational data.
2. **Traceability**: maintaining clear links between raw input data, intermediate computed features, and the resulting intent classification to support validation and operational acceptance.

This conceptual framework serves as the theoretical foundation for translating observable multimodal behaviour into meaningful indicators of ATCO intent, forming the basis for the data-driven methodology described in the following sections.

4 Methodology for ATCO Intent Assessment

In this deliverable, intent is operationalised as the ATCO's current task type (taxonomy in **Error! Reference source not found.**) and, when applicable, the associated aircraft instance. The ATCO intent assessment is carried out through two sequential steps.

First, the system infers the task type based on gaze data and HMI interactions observed over the previous few seconds. A machine-learning model computes and assigns a probability to each possible task type. This is developed in Section 4.4.

Second, once the task type is determined, the system identifies the task instance, that is, the specific aircraft to which the ATCO is applying that task. This instance identification is performed through a rule-based method that links gaze fixation points to the aircraft currently displayed on the screen. This is further developed in Section **Error! Reference source not found.**.

In the remainder of this section, we describe how the training data were collected and how these data were processed before being used by the machine-learning model. We then detail the feature-engineering phase, which transforms eye-tracking and HMI data into relevant indicators for inferring task type intent. Subsequently, we present the machine-learning methodology used to infer the task type, followed by the rule-based procedure used to deduce the task instance. Finally, we outline the limitations of the current model and discuss potential avenues for improvement.

4.1 Training Dataset Collection

4.1.1 Collection methodology

Ground truth data for training the ATCO intent recognition algorithm were collected during a four-week simulation campaign in September 2025. The sessions involved 23 Ukrainian ATCOs performing three distinct en-route control scenarios, each designed to expose participants to a range of operational contexts.

The scenarios included varying combinations of tasks like aircraft conflicts, pilot and feeder requests, quality-of-service optimisations, and non-conformance events. Each scenario was carefully structured to elicit authentic ATCO decision-making and interaction patterns under realistic workload and traffic conditions.

During the simulations, a continuous multimodal data stream was recorded for each participant, combining:

- Eye-tracking data (gaze coordinates),
- HMI interaction data (mouse movements, pop-up windows, measurement tools, clearance and transfer actions, etc...)
- Simulator context data (aircraft positions and label windows).

These data formed synchronised time series representing the ATCO's visual attention and interaction behaviour over time while managing evolving traffic situations.

Following the simulations, each ATCO participated in a self-annotation session. They reviewed their own session recordings, showing the radar display and gaze overlays, and were instructed to mark the time periods corresponding to specific tasks they performed during the simulation. Each label corresponds to a continuous time interval during which the ATCO pursued a particular operational goal.

Table 1: Task names, ids, and counts

Task name	Task id	Total observations count
Idle	Task -1	-
Aircraft requests	Task 0	313
Assume	Task 1	1766
Conflict resolution	Task 2	307
Entry conditions	Task 3	63
Entry conflict resolution	Task 4	26
Entry coordination	Task 5	300
Exit conditions	Task 6	133
Exit conflict resolution	Task 7	16
Exit coordination	Task 8	212
Non-conformance resolution	Task 9	50
Quality of Service (QoS)	Task 10	343
Return to route	Task 11	512
Transfer	Task 12	1736
Zone conflict	Task 13	751

The labelling process was guided by a predefined task taxonomy given in **Error! Reference source not found.**, derived from scenario procedures and the ASA task framework. Inter-rater reliability was evaluated to ensure annotation consistency across participants. Disagreements were discussed and resolved collectively during expert review sessions.

The resulting dataset thus represents a synchronized, expert-labelled multimodal time series, linking gaze and HMI interactions to corresponding operational intents. This labelled data serves as the ground truth for training and validating the machine learning models, where each task label constitutes the target class the system is ultimately designed to predict.

4.1.2 Dataset limitations

The collected dataset, which combines eye-tracking data, ATCO simulator data, and detailed task labels, is of great value. However, we have identified four major shortcomings that introduce significant challenges when using it to predict ATCO tasks.

First, the data labelling is manually performed. Although the ATCOs responsible for labelling are domain experts, human labelling inevitably introduces noise, making some labels unreliable. A key issue is that the definitions of task start and end points may vary across ATCOs, meaning that the same task can encompass slightly different events depending on the individual. In addition, the labelling workload is substantial, and human errors are frequent. For example, across all collected scenarios, we identified more than 300 start or end flags without a corresponding counterpart, preventing us from identifying the task in those portions of the data stream, even when a task actually occurred. Another indicator of the limited reliability of the labels is that participants do not always have the same number of tasks, despite having executed the same scenario. Overall, this label-induced noise makes task prediction particularly challenging for machine-learning algorithms.

Second, limitations arise from the way scenarios were constructed. As in real operational environments, some controller tasks occur far more frequently than others. As a result, the dataset exhibits an extreme imbalance: the most frequent tasks (e.g., Transfer or Assume) appear more than 1,700 times, while the rarest tasks (e.g., Exit conflict resolution) appear only 16 times. Although class imbalance is a common issue in machine learning, a ratio exceeding 100:1 in a multi-class classification setting makes accurate identification of very rare tasks nearly impossible. Combined with noisy labels, we expect particularly low detection rates for these rare categories.

Third, the ATCOs participating in the data-collection campaign were not familiar with the Polaris simulation tool. In addition to inter-controller variability in task execution, unfamiliarity with the tool led to inconsistent use of its built-in features (distance measurement tools, speed vectors, clearance annotations, etc.). Consequently, HMI interactions were not uniformly recorded across scenarios, and some HMI modalities are significantly under-represented. This uneven representation introduces additional noise and further complicates task prediction.

Finally, the eye-tracking and HMI data were collected on different computers, creating difficulties when merging them into a unified reference frame. Eye-tracking data were recorded in screen coordinates for an HD resolution, whereas HMI data were recorded in a 4K resolution and only referenced to the Polaris window, not the full screen. This mismatch introduced offsets that had to be manually identified. Furthermore, the clocks of the eye-tracking and Polaris systems were not synchronised, and a temporal drift was observed. This issue is particularly problematic because accurate alignment of eye-tracking and HMI data (down to the millisecond) is essential for deriving meaningful features.

While some of these shortcomings can be mitigated through extensive data cleaning and preprocessing, the high noise level, task imbalance, and dependence on manual labels make it very difficult to reliably train a machine-learning model. The resulting limitations of the model will be detailed later in the document.

4.2 Data Sources and Preprocessing

The method development relied on multimodal data recorded during ATCO-in-the-loop simulations conducted in AWARE. Each session captured synchronised eye-tracking, HMI interaction, and simulator context data within a controlled en-route ATC environment.

The input data for ATCO intent assessment combine eye-tracking measurements, using the Tobii Fusion Bar [2], with HMI interaction logs from Polaris recorded during simulation sessions in September 2025. The goal is to generate synchronised, analysis-ready data streams that reflect both visual attention and interface interaction.

Based on the gaze, mouse, and HMI streams, we construct three observation windows (short-, medium-, and long-term) representing the interaction history preceding each prediction time. The ground truth associated with each prediction time is the task that the ATCO is performing *at that exact moment*. These windows form the input to the machine-learning model.

To obtain these windows, we developed a standardized preprocessing pipeline designed to produce synchronised, analysis-ready multimodal data that capture both visual attention and interface interaction.

- **HMI tracking** recorded all user interactions with the radar interface, including mouse clicks, menu selections, cursor movements, and interaction with interface widgets. Each event is time-stamped and linked to the corresponding object in the interface.
- **Simulator data** provided dynamic contextual information, such as aircraft positions and identifiers.

The preprocessing pipeline consists of the following steps:

- **Data extraction**: Eye-tracking data are retrieved from tabular exports, whereas mouse and HMI interactions are loaded from structured Polaris databases. This step also associates each data stream with the correct participant and scenario.
- **Data cleaning**: Cleaning routines attempt to mitigate imperfections in the raw data, such as missing entries, inconsistent records, or invalid gaze samples.
- **Time alignment**: All timestamps are converted to a common UTC-millisecond reference. This alignment is essential to synchronise the streams precisely and tries to compensate for the **clock drift** between the machine collecting eye-tracking data and the computer running Polaris. Without this correction, cross-modal temporal relationships would be unreliable.
- **Multimodal integration**: Once time-aligned, gaze samples, mouse movements, and HMI events are merged into a single coherent multimodal sequence for each participant. This sequence represents everything the controller perceived and interacted with at each instant.
- **Construction of observation windows**: For every chosen prediction time, we extract three temporal windows representing recent ATCO activity:
    - **Short-term window**: captures the most immediate gaze patterns and interface actions,
    - **Medium-term window**: captures ongoing interactions and visual focus,

- **Long-term window**: captures broader behavioural context.

These windows form the model inputs, while the current task at the prediction time serves as the label.

The final output of this pipeline is a dataset containing, for each participant and scenario, multiple prediction times. Each prediction sample consists of three synchronized observation windows paired with the corresponding ground-truth task.

4.2.1 Temporal Synchronization

A major challenge in constructing the dataset was the temporal alignment of data from two independent hardware systems: the "Recorder" machine (running the eye-tracking software and TeamViewer for gaze overlay) and the "Participant" machine (running Polaris). We observed a significant clock drift between the two systems. Over the course of the recording campaign, the baseline offset between the internal clocks grew from approximately 17 seconds in the first session to 66 seconds in the final session. Furthermore, we also observed that this offset changed over the course of a single session, often in a non-linear manner.

Internal computer clocks are subject to inherent inaccuracies caused by variations in quartz crystal oscillator frequencies, temperature fluctuations, and component aging. These factors cause the system time to drift over time. Standard systems mitigate this by synchronizing with a master clock via the Network Time Protocol (NTP). In this setup, we hypothesize that one machine remained isolated from the network, accumulating drift, while the other performed periodic NTP updates. This unilateral correction is the likely cause of the observed non-linear drift patterns between the two systems.

Figure 1 illustrates an example of clock drift between the Polaris and Recorder computers. The drift is neither constant nor strictly linear over time, which makes it particularly challenging to estimate and correct. Moreover, the magnitude of the drift is substantial, making explicit compensation unavoidable. Without correction, this drift would render the synchronisation between eye-tracking and HMI data streams effectively unreliable, severely degrading any multimodal analysis.

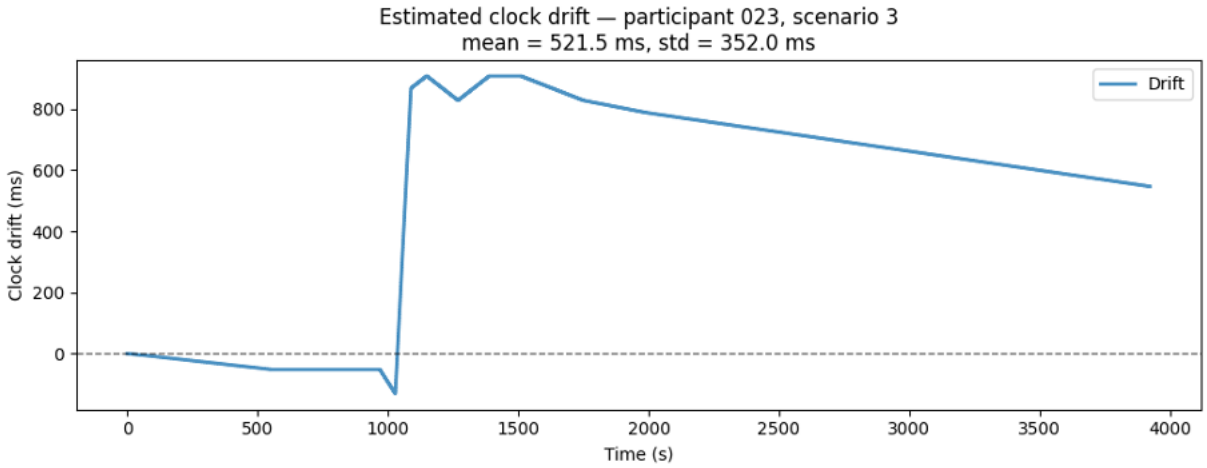

Figure 1: Example of estimated clock drift between Recorder and Participant systems

Quantitatively, we observed that the average clock drift per scenario ranges from −2,794 ms to +749 ms, with a standard deviation of 484 ms across scenarios. Such variability highlights the absence of a stable temporal relationship between the two acquisition systems. Given that eye and mouse movement dynamics must be resolved at the millisecond scale to accurately capture rapid gaze shifts, saccades, or brief interaction bursts, this level of drift is unacceptable. Without correction, the temporal misalignment would obscure fine-grained behavioural patterns and substantially compromise the validity of any multimodal feature extraction. Consequently, explicit drift estimation and compensation are a prerequisite for reliable gaze–HMI integration.

To resolve this, we developed a custom visualisation tool that utilises the TeamViewer screen recording. This screen recording captured the system clocks of both machines simultaneously in HH:MM format, providing the only common temporal reference frame. We treated the Participant clock as the ground truth and calculated the necessary time shift to apply to the eye-tracking data.

4.2.1.1 Clock offset calculation algorithm

The tool operates by identifying "synchronization pairs", a specific moment in the video recording where the system clock advances by one minute, both for Recorder and Participant independently. By extracting the Presentation Timestamp (PTS) of the video frame where these transitions occur, we calculate the exact offset required to align the eye-tracking data with the Participant system. The offset Δt for a specific synchronization pair is calculated as:

$$\Delta t = \left(T_{Recorder} - T_{Participant}\right) - \left(PTS_{Recorder} - PTS_{Participant}\right)$$

$T_{Recorder}$ and $T_{Participant}$ represents the respective wall-clock times displayed in the screen. $PTS_{Recorder}$ and $PTS_{Participant}$ are the video presentation timestamps (relative to the start of the video) where that specific minute change occurred.

However, calculating this offset only at the start and end of a recording is insufficient due to the non-linear nature of the drift. To address this, a recursive piecewise interpolation method was applied:

1. **Initialization**: Offsets are calculated for the start and end of the recording.
2. **Linear Prediction**: A linear interpolation is assumed between the two known points to predict the offset at the temporal midpoint.
3. **Verification**: The predicted midpoint frame is visually validated and corrected, if required.
4. **Recursion**: If the difference between the predicted and actual frame for the midpoint exceeds the uncertainty threshold (defined by the video frame rate), the segment is split, and the process is repeated recursively for the new sub-segments.
5. **Termination**: The recursion stops when the predicted deviation is negligible or when the resolution saturation is reached (i.e., the distance between points is less than one minute, rendering further visual extraction impossible).

4.2.1.2 Limitations and error sources

Despite the robustness of this method, several sources of uncertainty remain:

- **Clock Granularity:** The clocks only display time in minutes, so if a non-linear drift event occurs, it contaminates the entire one-minute window, as it cannot be further resolved.
- **Discrete Time Steps:** The recordings were captured at 25 frames per second (fps), introducing a quantization error of 40 ms for every timestamp extraction.
- **TeamViewer Latency:** The screen sharing stream introduces a variable latency that affects when the Polaris screen appears in the video relative to the Recorder's overlay, dependent on network stability. This latency cannot be measured dynamically and is treated as inherent noise in the synchronization.

4.2.1.3 Results of the synchronization process

The synchronization algorithm successfully aligned the data streams for 56 recordings, detailed in Table 2. To ensure robustness, a minimum of 5 synchronization pairs were collected for every recording (Start, End, Midpoint, and two quartiles). Only one scenario did not have TeamViewer recording.

Table 2: Distribution of non-linear clock drift events across recordings

Drift Behaviour	Non-linear drift events	Count
Linear / Stable	0	22 (39%)
Single Jump	1	30 (54%)
Complex / Unstable	2+	4 (7%)
Total	-	56

Only 22 recordings (39%) exhibited a linear drift that could be accurately modelled by a constant rate of change. The most prevalent behaviour, observed in 30 recordings (54%), was the "Single Jump" pattern. In these cases, the clock drift remained linear for most of the session but was interrupted by exactly one discrete non-linear event, likely corresponding to a scheduled system clock update (NTP). Finally, a small subset of 4 recordings (7%) presented complex instability involving multiple changes in drift rate, requiring the algorithm to saturate the minute-level precision limit multiple times to maintain alignment. But even in the most extreme cases, the recursive approach isolates these abnormal drift events to the one-minute window in which they occur, maintaining correct synchronization in most of the recording.

These findings underscore the necessity of the recursive piecewise approach. Had a standard linear regression been applied, over 60% of the recordings would have suffered from significant local misalignment. The applied correction minimises these errors, ensuring that gaze fixations and HMI events are correlated with the highest achievable precision for the subsequent feature engineering phase.

4.2.2 Data Cleaning

To mitigate part of the shortcomings identified in the collected dataset, an extensive data-cleaning phase is required. The initial cleaning steps can be divided into four categories: time-related cleaning, task-related cleaning, handling of missing values, and the spatial matching between eye-tracking and HMI data.

4.2.2.1 Time-related cleaning

Time-related cleaning addresses the temporal misalignment between the eye-tracking data and the Polaris HMI data, namely inconsistent timestamp formats. Eye-tracking data are time-stamped in milliseconds relative to the start of the eye-tracking acquisition, whereas HMI events are time-stamped in Unix Epoch milliseconds. The relative timestamps in the eye-tracking stream are unreliable because the first minutes of the recording correspond to calibration procedures rather than actual simulation activity. For this reason, we convert all timestamps to a common reference (Unix Epoch time) to allow temporal merging of the two modalities.

4.2.2.2 Task-related cleaning

Task-related cleaning addresses the noise present in the task labelling. Because task annotation is performed manually (ATCOs rewatch their own session and place start and end flags marking intervals they associate with a specific task type) errors inevitably occur and degrade the reliability of the ground truth used by the algorithm.

Across all data acquisitions, we identified 340 flags with no matching start or end counterpart. Although this represents only about 0.5% of the 6,458 labelled tasks, it still confirms that the labelling process is imperfect and introduces noise into the dataset. Intervals where a task is missing a start or end flag cannot be retrieved and are therefore labelled as IDLE, even though a task was actually being performed. These mislabelled segments will negatively affect algorithm performance.

To mitigate this source of noise, all recordings were subsequently reviewed by an external Swiss ATCO. During this review, task annotations containing missing start or end event markers were checked. Incomplete or inconsistent task labels were corrected by adding or adjusting event markers based on the video recordings. The identified missing markers dropped down from 340 to 39 only.

In addition, we observed high variability in task durations, with some tasks being implausibly short or excessively long. Such inconsistencies also indicate unreliability in the human labelling process. While very short tasks are not necessarily problematic for model training, abnormally long ones likely correspond to labelling mistakes and must be removed.

To mitigate these issues, we apply duration-based filtering. For each task type, we compute the interquartile range (IQR[2] = $Q_{75} - Q_{25}$) of task durations and discard tasks whose duration exceeds $Q_{75} + 3 \times IQR$, as these outliers are likely erroneous. We also discard all tasks shorter than 0.5 s, which we consider below realistic human reaction time. In total, 86 tasks were removed by this filtering and relabelled as IDLE.

Table 3 provides the distribution of task durations in seconds before and after applying these cleaning operations.

[2] IQR: Interquartile Range

Table 3: Task duration [s] distribution before and after removing unusually long task labels

	Before cleaning				After cleaning			
Task id	Mean	Median	Min	Max	Mean	Median	Min	Max
0	15.279	12.438	0.821	94.872	14.861	12.330	0.821	53.370
1	8.107	7.221	0.371	76.658	7.783	7.178	0.715	26.759
2	12.035	9.702	2.059	61.685	11.203	9.566	2.059	30.020
3	18.903	18.034	2.226	62.712	18.197	17.534	2.226	47.518
4	21.825	19.598	1.582	69.566	19.915	19.255	1.582	62.712
5	18.668	17.400	1.021	71.876	18.207	17.340	1.021	47.518
6	12.297	9.218	1.232	70.116	11.367	9.120	1.232	37.135
7	13.763	7.894	1.870	53.879	11.089	7.773	1.870	32.910
8	18.339	16.161	1.705	76.243	17.806	16.000	1.705	52.108
9	15.843	13.396	4.493	40.152	15.843	13.396	4.493	40.152
10	8.356	7.460	0.540	32.903	8.221	7.408	0.540	26.931
11	9.368	8.367	0.011	44.322	9.104	8.356	0.787	30.014
12	10.165	7.862	0.004	3284.167	8.126	7.821	0.785	25.344
13	11.791	9.938	1.011	302.949	10.451	9.771	1.011	33.280

4.2.2.3 Missing values

Handling missing values requires particular attention because the different data streams exhibit different behaviours regarding missingness. Mouse and HMI events from the Polaris software are event-driven: data are transmitted only when an interaction occurs. Consequently, the absence of data does not necessarily correspond to missing values, but simply to the absence of user interaction (no mouse movement or no triggered HMI event).

In contrast, gaze data are recorded at fixed sampling intervals, and a substantial proportion of samples are missing when the ATCO's gaze cannot be reliably mapped onto the screen. Importantly, these missing gaze samples carry meaningful information. They may correspond to a blink, a loss of visual attention, or a sensor acquisition issue. We therefore use missingness patterns to derive additional features:

- **Blink detection**: Consecutive missing samples shorter than 100 ms are classified as blinks. A high blink rate is often associated with increased workload and may indicate that the ATCO is engaged in a demanding task.

- **Loss of attention**: Missing value runs longer than 400 ms are interpreted as periods where the ATCO looked away from the screen. These intervals typically correspond to lower workload phases or off-screen monitoring behaviour.

- **Acquisition issues**: If more than 80% of the samples in any observation window are missing, we assume a technical acquisition problem rather than a behavioural pattern. Such windows are discarded. In total, 4,781 windows out of the 67,117 constructed were removed for this reason.

While missing gaze data can provide valuable behavioural cues, it is not always possible to perfectly distinguish genuine physiological events (blinks or off-screen glances) from acquisition artefacts. For example, we observed windows in which the detected blink rate exceeded 40 blinks per second, whereas typical human blink rates are around 20 blinks per minute. Although these sequences are categorised as blinks by our rule-based criteria, they are almost certainly contaminated by sensor noise. To the best of our knowledge, no further reliable cleaning is possible given the available data.

4.2.2.4 Spatial matching

Spatial matching between eye-tracking coordinates and Polaris HMI data is also required. The eye-tracking sensor is calibrated to map gaze positions over the entire physical screen, with the origin in the top-left corner and a resolution of 1920 × 1080 pixels (HD). In contrast, Polaris reports HMI coordinates only within the application window, not the full screen, and uses a 4K coordinate system (3840 × 2160 pixels) with the same origin convention.

The first step is therefore to convert eye-tracking coordinates from the HD reference frame to the Polaris 4K reference frame. In addition, because the upper part of the screen contains a system taskbar that is visible to the eye-tracking sensor but not included in the Polaris window, we identified a 27-pixel vertical offset between the two coordinate systems. This offset must be subtracted when projecting gaze coordinates onto the Polaris interface.

During analysis, we also observed that both eye-tracking and Polaris HMI coordinates occasionally fall outside their nominal ranges (e.g., negative values or values exceeding the declared resolution). This occurs when the ATCO looks slightly outside the physical screen boundaries, or when Polaris internally tracks interface elements that extend beyond the visible window. In such cases, the Polaris software records the coordinates even though the corresponding elements are not displayed.

Rather than discarding these out-of-range values, we retain them. Gaze or HMI events occurring outside the visible screen may carry meaningful behavioural information, for example, indicating that the ATCO is looking away from the radar display, or interacting with interface elements that temporarily appear outside the main viewport.

4.2.3 Chunking and Labelling Strategies

With the data streams cleaned and synchronised, the next step is to construct the training dataset that will allow us to simulate real-time prediction of the task the ATCO is performing. To achieve this, we

divide each multimodal data stream into consecutive prediction times, spaced 3 seconds apart. For each prediction time, the algorithm must infer the task (or IDLE) being performed at that moment, using only the preceding observations.

Construction of observation windows

For every prediction timestamp, we extract three observation windows that capture different temporal scales of ATCO behaviour:

- Short-term window (3–5 s): Captures the most immediate gaze shifts, mouse actions, and HMI interactions relevant to the ongoing task.
- Medium-term window (10–20 s): Reflects the broader sequence of actions and visual focus patterns that provide context for the emerging task.
- Long-term window (20–45 s): Represents the high-level operational context and strategic behaviours leading up to the prediction point.

The target label for each training sample is the task the ATCO is performing at the end of these windows. A small temporal margin is added to avoid having to predict a task that has just begun, which may not yet be reflected in the observed interaction patterns.

Parameters used for dataset construction

In our experiments, we adopt the following parameters:

- Interval between prediction times: 3000 ms
- Short-term window length: 5000 ms
- Medium-term window length: 10,000 ms
- Long-term window length: 25,000 ms
- Margin between task start and prediction time: 2000 ms

Output of the chunking workflow

This chunking procedure yields a uniform, temporally aligned dataset where each sample consists of three synchronised behavioural windows (short, medium, and long term) paired with a ground-truth task label. The objective of the machine-learning model is to learn how to map the last 25 seconds of multimodal gaze, mouse, and HMI activity onto the ATCO's operational intent, as represented by the task label. A visual representation of one training sample is displayed on Figure 2.

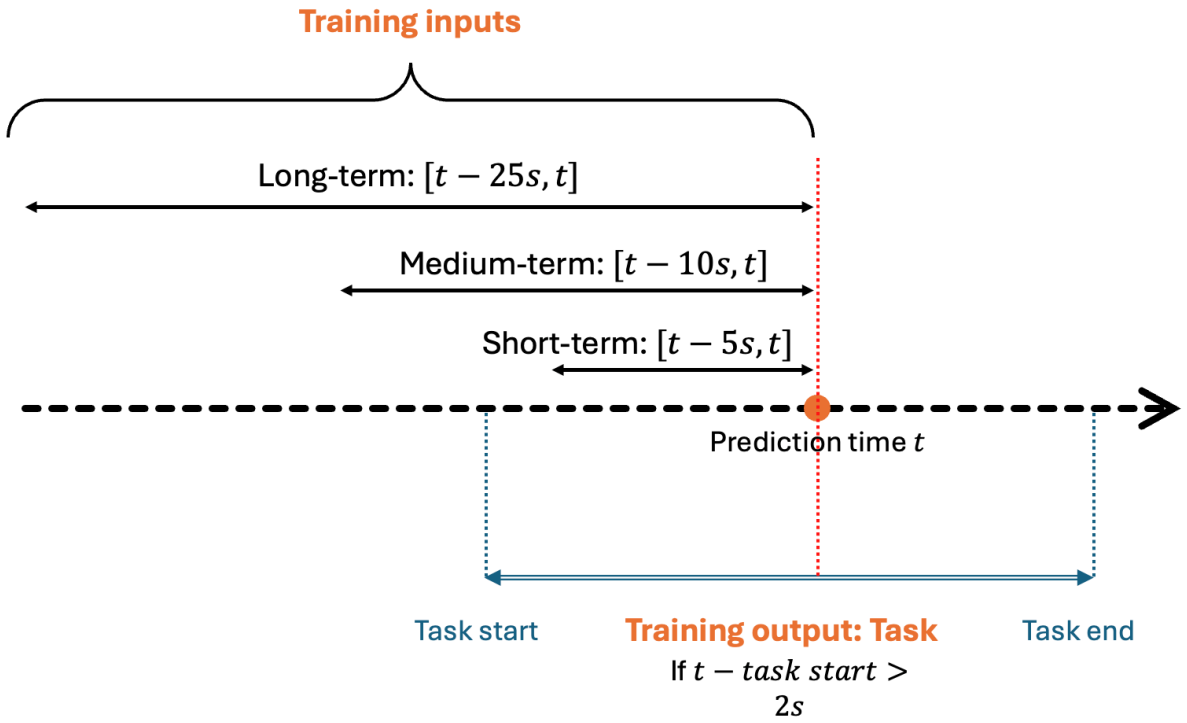

Figure 2: Representation of one training sample at a given prediction time

4.3 Feature Engineering

The feature engineering stage transforms the pre-processed observation windows into a structured set of behavioural indicators suitable for predicting the ATCO's task from gaze, mouse, and HMI information. This stage combines manual feature construction, quality-control procedures, and automated time-series feature extraction to capture both visual attention patterns and interaction dynamics that are relevant to ATCO intent. For clarity and methodological consistency, we divide the feature-engineering process into three components: gaze-derived features, mouse-derived features and HMI-derived features. Thus, for one prediction time defined earlier, we have the scalar metrics computed for all the short-, medium- and long-term observations windows.

4.3.1 Gaze-derived features

Blinks and loss-of-attention periods are not directly provided by the eye-tracking device and must therefore be manually inferred, as they contain crucial information for workload estimation. As detailed in the data-cleaning section, we identify short missing-data intervals as blinks and longer gaps as loss-of-attention runs. Short blinks are common physiological events and carry meaningful workload cues, whereas extended gaps may reflect temporary disengagement or sensor-tracking failures. Distinguishing between these cases enhances interpretability and improves the robustness of downstream modelling by separating physiological signals from technical artefacts.

Once missing-value segments have been categorised, each valid analysis window is transformed into a compact set of gaze-behaviour indicators describing how and where the ATCO allocated visual

attention. We first compute a set of expert-defined gaze metrics designed to characterise core aspects of human visual behaviour:

Fixations

Fixations correspond to periods during which the gaze remains spatially stable, indicating focused attention. We define a fixation as any interval during which gaze coordinates remain within a 50-pixel radius for at least 100 ms. For each observation window, we compute:

- Fixation count
- Total fixation time [s]
- Mean fixation duration [s]

Saccades

Saccades represent rapid transitions between fixations. They reflect visual search, comparison between interface elements, or scanning behaviour associated with specific ATCO tasks. We define a saccade as a displacement of more than 50 pixels in a single frame (i.e., within 0.008 s given the 120 Hz sampling rate), corresponding to a large one-frame jump. For each window, we compute:

- Saccade count
- Average saccade amplitude [px]
- Average saccade velocity [px/s]

Global gaze-motion characteristics

These metrics capture the overall tempo and fluidity of the gaze trajectory:

- Mean gaze velocity [px/s]
- Mean gaze acceleration [px/s^2]

Blink and dispersion metrics

- Blink rate, computed as the number of blinks per second based on the inferred blink flags
- Gaze dispersion, defined as the area of the bounding box enclosing all gaze samples in the window (px^2), which quantifies spatial spread

Together, these expert features quantify focus, search strategy, and the dynamics of visual exploration, all of which are strong predictors of cognitive state and task engagement.

In addition to expert-based features, we compute general time-series descriptors using the TSFresh library's MinimalFCParameters set. For each observation window and for both the X and Y gaze coordinate time series, we extract statistical descriptors including:

- absolute energy

- mean
- sample count
- standard deviation
- variance
- skewness
- kurtosis
- maximum and minimum
- mean absolute change
- mean change
- median
- root mean square
- large standard-deviation test (Boolean)
- duplicate-value indicators
- sample entropy

These features provide an automated, domain-independent characterisation of the gaze trajectory and complement the expert-defined metrics. We apply a feature-significance hypothesis test to select the most informative features for predicting the target task, discarding irrelevant or redundant descriptors. This step ensures that only meaningful gaze-behaviour indicators are provided to the machine-learning model.

4.3.2 Mouse-derived features

Mouse positions are transmitted through the HMI event recorder of the Polaris software. Unlike the eye-tracking data, mouse positions are not sampled at a fixed rate but are recorded only when the mouse is moving. Consequently, long intervals without mouse-position updates indicate that the mouse remained stationary. Across acquisitions, we observe that the average time difference between two consecutive mouse-position events ranges from 100 ms to 316 ms, confirming the irregular and sparse nature of the mouse data stream. When the mouse is in movement, the data are sent with a 10Hz frequency (a position every 100 ms).

This event-driven recording mechanism makes it inappropriate to synchronise gaze and mouse trajectories onto a single unified timestamp grid. Such an alignment would artificially introduce large numbers of missing values into the mouse time series or create temporal artefacts depending on the interpolation strategy used. For this reason, we treat gaze and mouse streams independently, each within its own native timestamp domain.

A direct consequence is that we cannot model gaze and mouse jointly as a multivariate time series, nor can we reliably compute synchronous correlations, such as gaze–mouse distance over time or temporal coupling measures. This is unfortunate, as gaze–mouse correlation often provides valuable behavioural insights. Instead, we incorporate this relationship indirectly through higher-level features associated with specific HMI interactions, described in the next section about HMI-derived features.

The computed features describing mouse movements are:

- Velocity and acceleration: average cursor displacement and rate of change (px/s, px/s²)
- Movement frequency & idle time: frequency of motion and cumulative stillness duration
- Path direction changes: number of substantial movement-angle shifts, indicating search or corrective behaviour
- Distance and stops: total travel distance and number of low-speed pauses
- Movement bursts & stillness: number of high-speed bursts and near-still periods

These indicators summarize pace, precision, and effort in HMI interaction, complementing gaze data to reflect the perceptual–motor loop of ATCO activity. Clicks are not explicitly collected by the Polaris software as they are grasped by the interaction with other HMI objects.

4.3.3 HMI-derived features

While gaze and mouse movements form the primary behavioural channels, additional HMI-interaction features provided by the Polaris software are essential for understanding what the ATCO is looking at, what information is present on the screen, and which controlling tools are being used. Beyond mouse-position updates, Polaris transmits a range of event-driven HMI records whenever specific interface interactions occur:

- Track screen positions: records the on-screen coordinates of all aircraft in the radar display. Aircraft may be either visible (within the window) or outside the visible area but still tracked. These updates are automatically generated and do not require user interaction.

- Track label positions: records the positions and bounding boxes of aircraft labels displayed on the screen. Like track positions, these events are automatically produced by the simulator.

- Pop-up events: report the opening or closing of pop-up windows, including their type and screen location.

- Transfer actions: capture controller-initiated aircraft transfers, such as assuming control of an aircraft entering the sector or handing it over to the next sector.

- Clearances: record altitude, speed, or route clearances provided by the controller to specific aircraft. These entries must be manually registered by the ATCO and are not automatically generated when a verbal clearance is issued.

- Separation tool usage: logs the use of the built-in measurement tool that allows the ATCO to check real-time distances between two objects (e.g., aircraft-to-aircraft, aircraft-to-waypoint).

Other HMI features, such as route interactions, heading or speed-vector manipulations, and keyboard shortcuts, are theoretically available but were excluded from analysis because they were unreliably recorded in several acquisition runs.

Working with these HMI features presents several challenges. ATCOs may choose to use some tools selectively or not at all, depending on personal working style. Furthermore, because the participating ATCOs were unfamiliar with the Polaris interface, some tools were rarely or inconsistently used. For example, we observed several cases in which clearances were not registered in the system even though they were issued verbally. Finally, we detected substantial variability in the number of HMI events recorded for the same ATCO across different scenarios, suggesting the possibility of acquisition inconsistencies in certain runs. The following HMI features are computed for each observation window:

Generic features

- Total number of HMI events
- Number of unique events
- Average number of events per ms
- Average number of events per timestamps
- Entropy of event types
- One-hot encoding of event types

Track and label positions features

- Number of visible aircraft positions
- Ratio between visible aircraft and tracked aircraft
- Statistical descriptors of aircraft positions (mean, std, min, max)
- Number of aircraft appearing, disappearing, persisting and transient
- Number of labels visible, hovered, selected and on picture-in-picture (pip)
- Ratios of labels hovered, selected and on pip
- Statistical descriptors or aircraft label positions (mean, std, min, max)
- Mean label width, height and area
- Total label area

Transfer features

- Transfer type count

- Transfer type presence (Boolean)

Popup features

- Number of popups open
- Number of popups closed
- Popup presence (Boolean)
- Number of popups overlapping
- Popup dwell time (total, mean, max)
- Number of popups attached to aircraft (total, mean, max)
- Popups revisit count
- Inter-popup opening time (mean, median, std)

Clearance features

- Clearance count
- Clearance presence (Boolean)
- Number of flights that received a clearance
- Maximum number of clearances given to a same flight
- One-hot encoding of clearance types
- Inter-clearance time (mean, median, std)

Cross-modal gaze / mouse VS label / track features

- Fraction of gaze outside the Polaris window
- Total gaze time on labels
- Fraction of gaze time on labels
- Number of labels looked at
- Mean distance between gaze and labels
- Mean distance between gaze and aircraft positions
- Ratio of gaze within a label, a selected label, a hovered label
- Ratio of mouse inside a label
- Total time when the gaze is near an aircraft position

- Total time when the mouse is near an aircraft position

4.4 Machine Learning Methodology for Task Type Detection

4.4.1 Hierarchical model presentation

The machine-learning pipeline is designed to detect and classify the ATCO task types (cf. **Error! Reference source not found.**) at the prediction times defined in Section 4.2.3, using the behavioural features computed in Section 4.3. Formally, the model can be expressed as:

$$y_t = \hat{f}(X_t^{short}, X_t^{medium}, X_t^{long}),$$

where t denotes the prediction time, y_t is the vector of probabilities associated with each possible task type (including IDLE), $\hat{f}$ is the machine learning model, and $(X_t^{short}, X_t^{medium}, X_t^{long})$ represent the gaze-, mouse-, and HMI-based scalar features extracted from the short-, medium-, and long-term observation windows preceding time t.

Because synchronising gaze, mouse, and HMI time series onto a unified temporal grid would introduce unwanted artefacts, we cannot exploit multivariate sequence models such as Long Short-Term Memory (LSTMs) or Transformer-based architectures. Instead, we rely on classical machine-learning techniques that operate directly on the scalar feature vectors described in Section 4.3, which are designed to encode the essential characteristics of the underlying time series without requiring explicit temporal alignment.

Using the chunking strategy described in Section 4.2.3 and after applying the data-cleaning procedures, Table 4 reports the number of available prediction samples for each task class.

Table 4: Number of training samples for each task after chunking and cleaning

Task name	Task id	Total windows count
Idle	Task -1	43389
Aircraft requests	Task 0	1396
Assume	Task 1	4205
Conflict resolution	Task 2	1001
Entry conditions	Task 3	180
Entry conflict resolution	Task 4	136
Entry coordination	Task 5	1429
Exit conditions	Task 6	316
Exit conflict resolution	Task 7	47

Exit coordination	Task 8	1047
Non-conformance resolution	Task 9	241
QoS	Task 10	851
Return to route	Task 11	1436
Transfer	Task 12	4232
Zone conflict	Task 13	2430

Table 4 clearly highlights a strong imbalance between IDLE windows and the task-related windows. At the extreme, *Exit Conflict Resolution* appears in only 47 prediction samples, whereas IDLE occurs 43,389 times. Such a disparity makes reliable identification of the rarest tasks virtually impossible within a standard multi-class classification framework.

To mitigate this issue, we adopt a hierarchical two-stage XGBoost approach, which effectively reduces the imbalance seen at the task-class level:

- **Stage A – Activity Detection**: A binary classifier distinguishes between windows associated with active control behaviour and those corresponding to IDLE monitoring.
- **Stage B – Task Type Classification**: A multi-class classifier is applied only to the samples identified as active by Stage A, in order to predict the specific operational task being performed.

This decomposition substantially reduces the imbalance problem by isolating the overwhelming majority class (IDLE) and allowing the model in Stage B to focus exclusively on discriminating between task types within a more balanced subset of the data.

4.4.2 Data split and participant-based cross validation

The dataset is split into training and test sets (80/20). Model selections and threshold calibration are performed only on the training set. To avoid data leakage across controllers, cross-validation is participant-disjoint: folds are created using the participant identifier as a grouping variable, ensuring that samples from the same ATCO never appear in both train and validation splits. This choice is critical to assess the model's ability to generalise to unseen controllers rather than exploiting participant-specific behavioural patterns.

For Stage A, we use a group-based cross-validation scheme with participants as groups. For Stage B, which operates only on active windows, we employ a Stratified Group K-Fold strategy to preserve both participant separation and class proportions across folds.

Hyperparameter optimisation for both stages is performed using randomised search with 100 iterations, evaluated through cross-validation on the training data only.

4.4.3 Stage A: Activity Detection

The first stage of the algorithm aims to distinguish windows containing active control behaviour from those corresponding to idle monitoring. Idle windows account for approximately 70% of the training dataset, resulting in a highly imbalanced classification problem. The remaining 30% of the data correspond to active control windows but are further split across 14 different task types, which exacerbates the imbalance. For example, the *Exit Conflict Resolution* task represents only 0.07% of the training samples, making direct identification in a single-stage multi-class setting virtually impossible.

To mitigate this imbalance, Stage A focuses on filtering out the majority class (IDLE) so that subsequent models can concentrate on task identification only when activity is detected. This hierarchical design introduces a trade-off: errors made in the stage A may propagate to stage B, potentially degrading overall performance. However, empirical evaluation showed that this two-stage approach consistently outperforms a single-stage classification strategy.

Stage A is implemented as a binary gradient-boosted decision tree model (XGBoost), with the binary cross-entropy as training objective, and the soft probabilities (probability associated to each output) as output. The input features are those described in Section 4.3, and the target label is defined as 1 when the task ID is different from −1 (i.e., any operational task) and 0 otherwise. Stage A evaluates the following probability:

$$P(active \mid X_t^{short}, X_t^{medium}, X_t^{long}),$$

where $active$ is the binary output, and $(X_t^{short}, X_t^{medium}, X_t^{long})$ the training inputs at time t.

Class imbalance is addressed through sample weighting. A per-sample weight vector linearly upweights the minority class (active windows), with a maximum weight capped at 6. This strategy improves recall for active windows while avoiding excessive penalisation of the majority class.

The optimal model is selected using cross-validation across participants, ensuring participant-disjoint folds to prevent data leakage. Hyperparameters are tuned via a random search over 100 iterations, with the following search space:

- n_estimators (number of trees): 300–900
- max_depth (tree depth): 3–9
- learning_rate: 0.03–0.2
- subsample (row sampling per tree): 0.5–0.8
- colsample_bytree (feature sampling per tree): 0.5–0.8

Because the dataset is strongly imbalanced and we care about identifying the minority (active) class, the model selection metric for the random search is Average Precision (area under the Precision–Recall curve), which is better suited than accuracy under heavy imbalance.

After selecting the best hyperparameter configuration, we compute for the positive class:

- For each cross-validation fold for the training set, the model is trained on fold-train and evaluated on out-of-fold (OOF).

- The OOF output is $\hat{p}_A = P(\text{active} \mid X_t^{short}, X_t^{medium}, X_t^{long})$ for every training sample, obtained without using that sample in training.

A decision threshold is then selected using only OOF probabilities: we search a grid of thresholds in $[0.01, 0.99]$ and choose the threshold that maximises the F1-score (active class). This yields:

- a calibrated threshold τ_A,
- and an OOF F1-score that reflects realistic generalisation behaviour.

OOF predictions are thresholded as:

$$\hat{y}_A = \mathbb{1}\{\hat{p}_A \geq \tau_A\},$$

where $\hat{y}_A$ is 1 if active, and 0 else.

4.4.4 Stage B: Task Type Classification:

Stage B aims to predict the specific operational task performed by the ATCO within windows that have been identified as active by Stage A. To this end, the model is trained exclusively on windows associated with operational tasks, with all IDLE windows removed from the training set. By discarding the dominant IDLE class, this stage operates on a more balanced dataset and has a substantially improved ability to discriminate between individual task types, even those that are rare in the original data distribution.

Stage B is implemented as a multi-class XGBoost classifier with 14 task classes, using the same feature set as Stage A. The training objective is the negative log-likelihood, corresponding to a multi-class generalisation of binary cross-entropy, and the model outputs soft probability estimates for each task. Target labels are encoded as integers from 0 to 13, corresponding to the task identifiers listed in Table 4. Stage B evaluates:

$$P(task = c \mid active, X^{(B)}),$$

where $c \in [\![0,13]\!]$ representing the task id, and $X^{(B)} = \{(X_t^{short}, X_t^{medium}, X_t^{long}) : active = 1\}$.

As in Stage B, class imbalance is also handled through sample weighting, but with a more aggressive strategy. Class weights are computed using an exponential scaling with an exponent of 1.2, which strongly upweights rare tasks relative to frequent ones. To prevent instability during training, the maximum weight is clipped to 8.

Model selection is performed via participant-disjoint cross-validation, ensuring that no participant appears simultaneously in training and validation folds. Hyperparameters are optimised using a random search over the following space:

- n_estimators (number of trees): 500–1200
- max_depth (tree depth): 3–9
- learning_rate: 0.03–0.2

- subsample (row sampling per tree): 0.5–0.8
- colsample_bytree (feature sampling per tree): 0.5–0.8
- gamma: 0.0-0.4

4.4.5 End-to-end hierarchical model (combined model)

The final task-probability map is obtained by combining the outputs of Stage A and Stage B according to the following formulation. For each CV fold:

1. Fit Stage A on fold-train and compute $P(\text{active} \mid X_t)$ on fold-validation.
2. Fit Stage B on fold-train active samples only and compute conditional probabilities $P(\text{task} = c \mid \text{active}, X_t)$ on fold-validation (computed for all validation samples for consistency, including IDLE).
3. Combine into the final task probability vector:

$$P(task = c \mid X_t) = \begin{cases} 1 - P(active \mid X_t), & if\ c = IDLE \\ P(task = c \mid active, X_t) \times P(active \mid X_t), & otherwise \end{cases},$$

where $IDLE = -1$ denotes the IDLE class, $X_t = (X_t^{short}, X_t^{medium}, X_t^{long})$ represents the feature vector extracted from the observation windows, $P(\text{active} \mid X_t)$ is produced by Stage A, and $P(\text{task} = c \mid \text{active}, X_t)$ is produced by Stage B.

To implement this formulation, Stages A and B are evaluated in parallel rather than sequentially. For a given batch of samples, we first compute the probability of each window being active using Stage A. In parallel, Stage B computes a full task-probability vector for all samples in the batch, including those that may ultimately be classified as idle.

The final task-probability map is then obtained by taking the element-wise product of the two probability vectors. Intuitively:

- If Stage A assigns a low probability to the active state, then the resulting probabilities for all operational tasks are close to zero, while the IDLE probability (computed as the complement of the active probability) remains high.
- Conversely, if Stage A assigns a high probability to the active state, the IDLE probability becomes low, and the remaining probability mass is distributed across the operational task classes according to the output of Stage B.

This soft probabilistic fusion is more stable and less prone to error propagation than a strictly sequential approach in which Stage A produces hard decisions (active vs. idle) and Stage B is executed only for samples classified as active. By avoiding hard gating, uncertainty from Stage A is explicitly preserved and propagated to the final prediction, leading to smoother probability estimates and improved robustness.

4.4.6 Postprocessing – Temporal Smoothing

To stabilise temporal outputs and reduce flickering between consecutive predictions, we apply a temporal smoothing step to the Stage B task-probability maps. Specifically, we use an Exponential Moving Average (EMA) to smooth the probability trajectories over time by incorporating information from the previous prediction step. This approach exploits the strong temporal coherence of ATCO behaviour: abrupt transitions between unrelated tasks are rare, monitoring periods (IDLE) typically span long intervals, and operational tasks usually persist across several consecutive prediction times.

Combined with the multi-scale observation windows (short, medium, and long term), EMA further reinforces the modelling of longer-term dynamics while maintaining responsiveness to genuine task transitions. The EMA update rule is defined as:

$$\mathrm{P}_{\mathrm{t}}^{\mathrm{smooth}} = \alpha \mathrm{P}_{\mathrm{t}} + (1-\alpha)\mathrm{P}_{\mathrm{t-1}}^{\mathrm{smooth}},$$

where $\mathbf{P}_t$ denotes the task-probability vector at prediction time t, $\mathbf{P}_t^{\mathrm{smooth}}$ the smoothed probability vector, and $\alpha = 0.6$ in our configuration.

This post-processing step improves temporal coherence and leads to higher overall macro-F1 scores, while preserving sufficient reactivity for near–real-time applications.

4.5 Validation and Performance Evaluation

4.5.1 Evaluation metrics

The evaluation of the proposed hierarchical intent-recognition model is designed to (i) provide a reliable estimate of generalisation performance, (ii) account for strong class imbalance, and (iii) avoid data leakage across participants. The strategy follows the two-stage structure of the model and evaluates each stage independently before assessing the combined end-to-end system. Both out-of-fold (OOF) and test performance metrics are estimated.

4.5.2 Results

All components of the pipeline are evaluated using the same two-step procedure: out-of-fold (OOF) evaluation during cross-validation on the training set, followed by evaluation on a held-out test set. This provides (i) a reliable validation estimate during model development and hyperparameter tuning, and (ii) an unbiased estimate of final performance on unseen participants.

OOF Evaluation: During cross-validation, the model is trained on a subset of folds and evaluated on the fold that was not used for training. By concatenating predictions across all validation folds, each training sample receives a prediction from a model that never saw it during training. OOF performance is therefore a strong proxy for validation performance during training while limiting optimistic bias.

Test evaluation: After model selection, the final models are trained on the full training set and evaluated on a separate test set composed of participants not used for training. This test performance reflects the expected generalisation capability of the method on unseen ATCOs.

The same evaluation procedure is applied to each component of the overall system:

- **Stage A:** binary classification of IDLE vs Active windows

- **Stage B:** multi-class classification of the task type, restricted to the windows that are active
- **Combined:** probabilistic fusion of Stage A and Stage B outputs (as described in Section 4.4.5), without further post-processing.
- **Combined (smoothed):** same combined model with temporal post-processing (EMA smoothing on Stage B probabilities).

4.5.2.1 Metrics reported

Because the task distribution is highly imbalanced, we report multiple complementary metrics. These metrics are computed in a consistent way across stages, with minor adaptations between binary and multi-class settings.

Accuracy

Accuracy is the proportion of correctly classified samples:

$$\text{Accuracy} = \frac{\#\{\text{correct predictions}\}}{\#\{\text{samples}\}}.$$

Although easy to interpret, accuracy can be misleading under strong imbalance (e.g., predicting IDLE everywhere may yield high accuracy while failing to detect rare tasks).

Precision, recall, and F1-score (per class)

For each class c:

- Precision measures how reliable predictions of class c are:

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c}.$$

It answers: among what the model predicted as class c, how much is correct?

- Recall measures how well the model recovers class c:

$$\text{Recall}_c = \frac{TP_c}{TP_c + FN_c}.$$

It answers: among true instances of class c, how many did the model detect?

- F1-score is the harmonic mean of precision and recall:

$$\text{F1}_c = \frac{2 \cdot \text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c}.$$

It provides a single per-class summary that penalises models that perform well on only precision or only recall.

These per-class metrics are reported in the classification report to diagnose which tasks are well recognised and which ones are missed or confused.

Macro-F1

Macro-F1 is the unweighted average of the per-class F1-scores:

$$\text{Macro-F1} = \frac{1}{C}\sum_{c=1}^{C} \text{F1}_c \,,$$

where C is the number of classes. Macro-F1 treats all classes equally, regardless of their frequency, and is therefore particularly informative in imbalanced multi-class settings because it reflects performance on rare classes as much as on frequent ones.

N.B.: In the binary case (Stage A), the macro F1-score is equivalent to the F1-score of the positive class (active), which is the class of interest.

Weighted-average F1 (support-weighted)

The weighted-average F1-score is the mean of per-class F1-scores weighted by the number of true samples (support) in each class:

$$\text{Weighted-F1} = \sum_{c=1}^{C} \left(\frac{n_c}{N}\right) \text{F1}_c,$$

where n_c is the support of class c and N is the total number of samples. This metric reflects overall performance while accounting for the dataset distribution. It is more influenced by frequent classes than Macro-F1.

Confusion matrix

The confusion matrix provides a detailed view of misclassifications by counting, for each true class, how predictions are distributed across predicted classes. It highlights:

- which classes are systematically confused,
- which classes are over-predicted or under-predicted,
- and whether errors follow consistent patterns (e.g., rare tasks being absorbed into frequent ones).

This qualitative diagnostic is essential to identify bottlenecks at each stage of the pipeline.

4.5.2.2 Stage A

The decision threshold for classifying a window as active was selected using out-of-fold (OOF) predictions on the training set. The threshold that maximised the F1-score of the active class was $\tau_A = 0.32$, yielding an OOF F1-score of 0.606. This relatively low threshold reflects the prioritisation of recall over precision for active windows, which is desirable in this context to avoid missing operational tasks. This threshold was then fixed and reused unchanged for evaluation on the held-out test set.

On OOF data, the model achieves an accuracy of 0.697 and an active-class F1-score of 0.606.

- Recall for the active class (0.777) is relatively high, indicating that most active windows are successfully detected.
- Precision for the active class (0.497) is lower, meaning that a non-negligible fraction of windows predicted as active are in fact idle.

This behaviour is intentional and aligned with the design choice of Stage A: it acts as a high-recall activity detector, favouring the detection of potential task-related windows at the cost of some false positives. These false positives are subsequently handled by Stage B. The confusion matrix confirms this behaviour: a substantial number of IDLE windows are misclassified as active, while relatively few active windows are missed.

On the held-out test set (participants unseen during training), the model shows stable and slightly improved performance:

- Accuracy: 0.713
- Active-class F1-score: 0.658

Notably:

- Active recall increases to 0.854, indicating strong generalisation in detecting active control behaviour.
- Active precision improves to 0.536, suggesting better discrimination between idle and active windows on unseen participants.

The consistency between OOF and test results indicates that the model does not overfit the training participants and generalises well across controllers.

Overall, Stage A fulfils its intended role in the hierarchical pipeline: (i) it reliably filters out a large proportion of idle windows, (ii) it maintains high recall on active windows, ensuring that operational tasks are rarely missed, and (iii) the observed false positives (idle windows classified as active) are acceptable, as they are further processed by Stage B.

This performance profile validates the choice of Average Precision–based model selection, OOF-based threshold calibration, and F1-oriented evaluation, and confirms that Stage A provides a robust and conservative activity detection layer for the subsequent task classification stage.

4.5.2.3 Stage B

Stage B addresses a multi-class classification problem: identifying the specific operational task performed by the ATCO, conditional on the window being active. The model is trained and evaluated only on active windows, which removes the dominant IDLE class but leaves a highly imbalanced task distribution, with several rare tasks represented by only a few dozen samples. Performance is therefore primarily assessed using macro-averaged F1-score, which gives equal importance to all task classes and provides a realistic view of performance on rare tasks.

On out-of-fold data, Stage B achieves:

- Accuracy: 0.521
- Macro-F1: 0.270

These results highlight the intrinsic difficulty of the task. While overall accuracy greatly exceeds chance level (1/14), macro-F1 remains low, reflecting the model's limited ability to correctly identify very rare task types.

The per-class analysis reveals a clear performance stratification:

- **Well-recognised tasks**: Tasks with high support (e.g. classes 1, 11, 12, and 13) achieve comparatively strong F1-scores (up to 0.73 for class 12). These tasks exhibit consistent gaze and HMI patterns that are easier for the model to capture.
- **Moderately recognised tasks:** Tasks such as classes 0, 2, 5, 8, and 10 achieve modest F1-scores (≈0.15–0.35), indicating partial discrimination but frequent confusion with similar tasks.
- **Rare and poorly recognised tasks:** Several classes (e.g. 3, 4, 7) show zero recall and zero F1-score. These tasks have extremely low support and are effectively absorbed by more frequent classes, despite the use of class weighting.

The confusion matrix confirms that misclassifications are not random: rare tasks are predominantly misclassified as semantically or operationally related frequent tasks, rather than being spread uniformly across classes.

On the held-out test set, Stage B shows slightly improved performance:

- Accuracy: 0.557
- Macro-F1: 0.315

The increase in macro-F1 (+0.045 compared to OOF) suggests that the model generalises reasonably well and does not overfit the training participants. Moreover, we observe stable or improved performance for frequent tasks, with F1-scores remaining high for classes such as 1, 11, 12, and 13. There are also marginal gains for some mid-frequency tasks, indicating that learned patterns transfer across participants. Persistent failure on extremely rare tasks, which remain undetected despite aggressive class weighting.

Stage B illustrates the core challenge of fine-grained ATCO task recognition:

- Even after removing IDLE windows, the task distribution remains extremely imbalanced.
- Tasks with consistent and distinctive interaction patterns are recognised reliably.
- Tasks with very few examples or weak behavioural signatures remain difficult to detect.

These results justify the architectural choices made earlier in the pipeline:

- The hierarchical decomposition (Stage A → Stage B) is necessary but not sufficient on its own to solve rare-task detection, and the model might benefit from an improved training dataset.
- Reporting macro-F1 is essential, as weighted metrics would largely hide the poor performance on rare but operationally important tasks.
- The confusion patterns motivate the use of probabilistic fusion and temporal smoothing in the combined model, which aim to stabilise predictions over time rather than relying on single-window decisions.

Overall, Stage B provides meaningful discrimination among frequent operational tasks and a reasonable baseline for task recognition, while clearly exposing the limitations imposed by label scarcity and noise. Limitations that are further addressed in the combined and temporally smoothed stages.

4.5.2.4 Hierarchical Model

This evaluation assesses the end-to-end hierarchical pipeline, combining Stage A and Stage B, without temporal smoothing. Predictions are obtained by probabilistic fusion of both stages, and performance is evaluated using both OOF predictions and the test set. Because this setting reintroduces the highly dominant IDLE class and includes all operational tasks, performance is reported using Macro-F1 over all classes, and Macro-F1 over active tasks only, which isolates task-recognition performance independently of idle detection.

On OOF data, the combined unsmoothed model achieves:

- Accuracy: 0.699
- Macro-F1 (all classes): 0.192
- Macro-F1 (active only): 0.162

The high accuracy is largely driven by the strong performance on the IDLE class, which dominates the dataset. In contrast, the relatively low macro-F1 values reflect the difficulty of simultaneously detecting activity and correctly identifying the task type, particularly for rare tasks.

Per-class behaviour:

- IDLE detection remains robust, with an F1-score of 0.83 and a recall close to 0.89, confirming that Stage A retains its effectiveness when integrated into the full pipeline.

- Frequent operational tasks (e.g. classes 1, 11, 12, 13) achieve moderate F1-scores (≈0.29–0.50), indicating that correct task recognition is possible when sufficient training data are available.
- Rare tasks (e.g. classes 3, 4, 6, 7) exhibit zero or near-zero recall and F1-scores. These classes are typically misclassified as IDLE or absorbed into semantically related frequent tasks.

The confusion matrix highlights that most errors occur in two forms:

- Active windows misclassified as IDLE, reflecting residual false negatives from Stage A
- Confusions among operational tasks, particularly where behavioural signatures overlap.

On the held-out test set, the combined unsmoothed model yields:

- Accuracy: 0.711
- Macro-F1 (all classes): 0.220
- Macro-F1 (active only): 0.181

Compared to OOF results, both macro-F1 metrics improve slightly, suggesting that the model generalises well to unseen participants and does not suffer from severe overfitting. Moreover, there is (i) stable and strong IDLE performance (F1 ≈ 0.84), confirming consistent activity detection, (ii) an improved recognition of frequent tasks, notably class 12, which maintains an F1-score close to 0.60, and (iii) a persistent difficulty with rare tasks, which remain largely undetected even in the combined setting.

The combined unsmoothed results highlight several important points:

- The hierarchical strategy successfully integrates activity detection and task recognition, preserving the strengths of Stage A while enabling task-level predictions.
- Overall accuracy remains high but is not a completely reliable indicator of task recognition quality due to extreme class imbalance.
- Macro-F1 (active only) provides a more meaningful measure of operational task recognition and confirms that performance is still limited by rare-task scarcity and label noise.
- The absence of temporal smoothing leads to instability in per-window predictions, particularly for short or ambiguous task segments.

These findings motivate the final step of the pipeline: temporal smoothing of Stage B probabilities, which aims to stabilise predictions across time and reduce spurious task switches, results that are analysed in the following section.

4.5.2.5 Impact of Temporal Smoothing on the Combined Model

Applying temporal smoothing to the combined hierarchical model leads to consistent but moderate improvements over the unsmoothed variant, both in OOF validation and on test set. While smoothing

does not fundamentally solve the rare-task recognition problem, it improves temporal coherence, stabilises predictions, and yields measurable gains in aggregate metrics.

- Compared to the unsmoothed model:
- Accuracy increases from 0.699 → 0.711 (OOF) and 0.711 → 0.726 (test).
- Macro-F1 (all classes) improves from 0.192 → 0.197 (OOF) and 0.220 → 0.230 (test).
- Macro-F1 on active tasks only increases from 0.162 → 0.165 (OOF) and 0.181 → 0.188 (test).

Although these numerical gains are modest, they are systematic and consistent across validation and test sets, indicating that smoothing improves generalisation rather than overfitting temporal patterns in the training data.

Temporal smoothing primarily benefits frequent and mid-frequency tasks:

- Tasks such as classes 1, 5, 8, 11, 12, and 13 show increased F1-scores and more stable recall.
- The IDLE class benefits from increased recall (up to 0.93 on the test set), reinforcing long monitoring phases and reducing spurious activations.
- Extremely rare tasks (e.g. classes 3, 4, 6, 7, 9) remain largely undetected. Smoothing cannot compensate for the lack of discriminative evidence or sufficient training examples and therefore does not introduce false positives for these classes.

The confusion matrices show fewer rapid oscillations between operational tasks and IDLE, and fewer single-window misclassifications where a task is briefly predicted and immediately disappears.

From a modelling perspective, the improvements brought by smoothing are structural rather than statistical:

- ATCO tasks are temporally extended phenomena: once a task starts, it typically spans multiple consecutive prediction windows.
- The unsmoothed model operates on independent windows and is therefore sensitive to local noise, especially for short-term gaze and HMI fluctuations.
- EMA smoothing enforces a soft temporal prior: it discourages abrupt task switches while preserving responsiveness to genuine transitions.

Crucially, smoothing is applied only to Stage B probabilities, while Stage A (idle vs active detection) remains unsmoothed. This design choice prevents the suppression of short active bursts and avoids biasing the system toward excessive IDLE predictions.

4.5.2.6 Comparison with the previous algorithmic version

Compared to the previous version of the algorithm, the current approach represents a substantial methodological and performance improvement, both in terms of activity detection robustness and task-level discrimination, despite operating on a more realistic and challenging evaluation setup. Improvements were made on:

- The introduction of short/mid/long-term input windows instead of a single long one
- Windows are labelled according to the task performed at a certain point, and not based on a majority vote
- Eye-tracking and HMI streams are now temporally aligned, eliminating feature misalignment artifacts present in the earlier approach
- task labels have been reviewed by a third-party ATCO, reducing annotation noise.

First, Stage A shows a clear gain. The previous model achieved an overall accuracy of 0.62 and a macro-F1 of 0.44, whereas the updated version reaches 0.71 accuracy and an active-class F1-score around 0.65 on the test set. This improvement is particularly significant given that the new model operates on shorter, overlapping windows and uses a probability-calibrated threshold selected via out-of-fold evaluation. The improved recall of active windows confirms that finer temporal resolution, better feature engineering, and improved data cleaning significantly enhance the reliability of activity detection.

Second, Stage B improves from a macro-F1 of ≈0.16 in the previous version to ≈0.27 (OOF) and ≈0.32 (test) in the current implementation. While task-level recognition remains difficult, especially for rare tasks, the gain is notable given that the new model operates on noisier but more realistic short-term windows.

Frequent operational tasks (e.g. Assume, Entry Coordination, Transfer, Zone Conflict) now exhibit more stable and higher F1-scores, indicating that the richer multi-scale temporal context (short-, medium-, and long-term windows) better captures task-specific behavioural signatures.

Finally, at the system level, the combined hierarchical model shows a marked improvement in both accuracy and temporal coherence. The previous end-to-end pipeline achieved an overall accuracy of 0.61 and a weighted F1 of 0.61, whereas the current version reaches 0.73 accuracy and improved macro-F1 after temporal smoothing. Beyond raw metrics, the new system produces far more stable intent timelines, with reduced flickering, fewer spurious task switches, and clearer separation between monitoring and active control phases.

Overall, these improvements stem not from a single modelling change, but from a cumulative refinement of the entire pipeline: cleaner and better-aligned data, more precise task labelling, multi-scale temporal modelling, probabilistic hierarchical fusion, and principled temporal smoothing. While rare-task recognition remains limited by data scarcity and behavioural overlap, the current algorithm constitutes a significant step forward in both performance and operational plausibility, providing a much stronger foundation for future work.

4.6 Task instance association

While the machine learning models described in Section 4.4 identify what type of task the ATCO is performing, they do not inherently identify which aircraft is the target of that action. To enable full situational awareness, the system must link the inferred task intent to a specific aircraft instance within the airspace.

Unlike the task-type classification, where expert-annotated labels were available for training, the specific target aircraft was not explicitly labelled. Consequently, supervised learning approaches are not applicable for this sub-problem. Instead, we propose a probabilistic **Evidence Accumulation Model** based on cognitive principles of working memory and visual attention. This method functions as a state estimation engine, aggregating multimodal signals (gaze, mouse, and ASD events) to maintain a dynamic "Attention Score" for every visible aircraft. The final validation of the system (task type + task instance detection) will be conducted in the validation exercise performed in April 2026.

4.6.1 The Evidence Accumulation Framework

Human attention is inherently temporal rather than instantaneous. When an ATCO interacts with an aircraft, cognitive focus accumulates over time (for example, while reading a label) and persists briefly in short-term memory even after looking away to scan other areas. To model this, the proposed method treats attention as a continuous signal that rises and decays based on observed behaviour.

The core mechanism is a target-seeking state estimator. Rather than simply summing data points, the system determines a target confidence level based on the intensity of the current interaction. The score for an aircraft moves smoothly toward this target, incorporating temporal dynamics:

- **Accumulation**: When evidence (such as a gaze fixation or mouse hover) is present, the score rises, reflecting the acquisition of information.
- **Decay**: When evidence ceases, the score decays gradually rather than dropping instantly. This simulates the persistence of working memory, ensuring that an aircraft remains the probable target even during brief blinks, saccades, or transitions between sub-tasks.

This approach provides robustness against sensor noise, variable sampling rates, and interruptions in the data stream, ensuring a stable prediction of the target aircraft.

4.6.2 Hierarchical Behavioural Logic

The target confidence for each aircraft is determined by a hierarchy of behavioural evidence, ranging from broad visual scanning to specific physical manipulation. The method integrates these inputs to distinguish between passing attention and active operational intent.

4.6.2.1 Active manipulation

Definitive physical actions on the interface provide the strongest evidence of intent.

- **Direct Interaction:** Actions such as dragging a waypoint, modifying a vector, or opening a specific context menu imply a focused intent on the associated aircraft.
- **Tool Usage:** The activation of specific analysis tools, such as separation halos or distance measurement lines, indicates an active monitoring or problem-solving process regarding the target.

4.6.2.2 HMI interaction

Mouse dynamics provide strong indications of focus, particularly when correlated with specific interface elements.

- **Cursor Dwell:** The system tracks the mouse cursor relative to aircraft labels. An active hover is treated as a strong signal of interest.

- **Activity Filtering:** To avoid false positives caused by common operational habits, such as "parking" the mouse cursor on a label while visually scanning elsewhere, the system differentiates between an actively moving mouse and a stationary, idle cursor.

4.6.2.3 Visual attention

Gaze data provides the broadest signal but requires careful filtering to determine intent.

- **Dual-Entity Association:** The system accounts for the two visual components of an aircraft: the track label (containing textual flight data) and the track position (the radar dot). Attention to the track label typically implies information gathering, while attention to the track position implies spatial monitoring.

- **Stability Filtering:** The model distinguishes between fixations and saccades. Only stable fixations contribute to the evidence score, filtering out noise caused by the eye sweeping across the screen.

4.6.3 Contextual Memory

Finally, the system accounts for the persistent state of the HMI to model longer-term intent. If an aircraft is explicitly Selected (hooked) or Marked by the ATCO, the system applies a baseline confidence floor. This ensures that the aircraft retains a minimum level of importance in the system's estimation, preventing it from being "forgotten" even if the ATCO momentarily shifts their visual attention to other parts of the sector.

4.7 Limitations

4.7.1 Task-type recognition

While the proposed hierarchical intent-recognition framework demonstrates clear improvements over previous versions and yields encouraging results, several limitations remain. These limitations are primarily related to data acquisition conditions, annotation processes, and task distribution, rather than shortcomings of the modelling approach itself. Identifying them is essential both to contextualise the current results and to guide future research directions.

4.7.1.1 Nature of data collection

A key limitation arises from the nature of the data collection. ATCOs operated within realistic, end-to-end simulation scenarios rather than tightly scripted experimental tasks. While this greatly enhances realism, it also introduces substantial variability in interaction strategies, tool usage, and task execution timing. Importantly, these variations reflect legitimate differences in working styles among qualified professionals, not inconsistencies in expertise.

In particular, some HMI tools available in the Polaris system (e.g., clearance annotation, separation tools) were not used consistently across participants or scenarios. This is largely attributable to limited prior exposure to the specific simulation interface, rather than to operational habits. As a result, some

HMI interaction features are under-represented or inconsistently recorded, reducing their discriminative value for task recognition.

4.7.1.2 Noisy and subjective task labelling

Task labels were generated through retrospective self-annotation, with ATCOs reviewing their own recorded sessions and marking task start and end times. While this approach leverages expert knowledge, it inevitably introduces subjective interpretation and temporal uncertainty, particularly for tasks that overlap, interleave, or evolve gradually.

Despite careful post-processing and third-party review, the resulting ground truth remains noisy by nature, which places an upper bound on achievable model performance. In this context, misclassifications do not always reflect modelling errors but may instead correspond to ambiguous or debatable task boundaries.

4.7.1.3 Severe class imbalance and rare-task scarcity

The dataset exhibits extreme task imbalance, with monitoring (IDLE) dominating the timeline and several operational tasks appearing only a handful of times. Although hierarchical modelling, class weighting, and macro-F1 evaluation partially mitigate this issue, rare-task detection remains fundamentally limited by data scarcity.

This imbalance is exacerbated by the fact that rare tasks might occur under high workload and time pressure, leading to shorter and noisier behavioural signatures that are difficult to distinguish reliably from more frequent tasks.

4.7.1.4 Sensor and synchronisation constraints

Although substantial effort was invested in aligning eye-tracking and HMI data streams, they were collected on separate systems with different timestamping schemes and sampling characteristics. Residual misalignment and acquisition artefacts cannot be entirely ruled out, especially at fine temporal scales. These constraints precluded the use of fully sequence-based models and limited the exploitation of direct gaze–mouse temporal correlations.

4.7.1.5 Recommendations for future work

Controlled task-oriented acquisition scenarios

Results from our earlier proof-of-concept study using Microsoft Office tasks provide an important reference point. In that setting:

- Tasks were explicitly instructed and well defined,
- The task distribution was balanced,
- Labelling was less subjective,
- Eye-tracking and mouse data were collected within a single, synchronised system,
- No IDLE state was involved, as participants were continuously engaged.

Despite tasks being short and sometimes behaviourally similar, the system achieved over 85% accuracy, demonstrating that the modelling approach itself is sound when data conditions are favourable.

This strongly suggests that ATCO task recognition could be significantly improved by designing dedicated acquisition protocols, in which:

- ATCOs are asked to perform specific, isolated tasks in controlled mini-scenarios,
- Each task is repeated multiple times to ensure sufficient representation,
- HMI tool usage is either standardised or explicitly documented,
- Task boundaries are defined a priori rather than reconstructed retrospectively.

Such scenarios would complement full-scenario recordings rather than replace them, allowing a trade-off between realism and experimental control.

Improved training and familiarisation with simulation tools

Providing ATCOs with additional familiarisation sessions on the Polaris interface prior to data collection could improve the consistency and richness of HMI interactions. This would increase the informational value of HMI-derived features without constraining operational freedom.

Alternative task representations and grouping strategies

Given the difficulty of identifying very fine-grained and rare tasks, future work could explore:

- Hierarchical task grouping (e.g., Entry vs Exit vs Monitoring),
- Multi-label formulations for overlapping tasks within a window,

These representations may better match the granularity at which intent can be inferred reliably from behavioural data.

5 References

[1] D2.14 – AWARE Validation Scenarios

[2] Tobii Pro Fusion - Tobii

[3] Tsfresh - Documentation

6 List of acronyms

Acronym	Description
AI	Artificial Intelligence
ASA	Artificial Situational Awareness
ASD	Air Situation Display
ATCO	Air Traffic Controller
ATM	Air Traffic Management
CV	Cross-validation
EMA	Exponential Moving Average
HD	High Definition
HMI	Human-Machine Interaction
Hz	Hertz
IQR	Interquartile Range
LSTM	Long Short-Term Memory
NTP	Network Time Protocol
OOF	Out-Of-Fold
QoS	Quality of Service
UTC	Coordinated Universal Time
ms	milliseconds
px	pixels
s	seconds