Multimodal RAG for Context Aware UAV Navigation

Rithanya M
22011101090
Vasudharini VJ
22011101120

Dr. Rudrashis Majumder
Shiv Nadar University, Chennai

SHIV NADAR
UNIVERSITY
CHENNAI

Introduction

- Urban drone navigation requires perception (visual + possibly LiDAR), scene understanding (semantics, affordances), long-horizon planning, and robust low-level control. Recent advances in large-scale vision-language models (VLMs) and LLMs enable language-level task specification, commonsense reasoning, and generation of plans or action sequences, while multimodal VLMs provide visual grounding that drones can exploit for perception in complex urban scenes. Surveys and recent systems show the promise of combining these capabilities for more flexible, explainable UAV autonomy.

Literature Survey

Paper	Approach	Limitations
LLM-Land (2025)	Integrated BLIP for image captioning + LLM-RAG to generate safe drone landing zones, combined with real-time MPC trajectory planning	Only uses vision; ignores structured telemetry (e.g., GPS, battery), limiting real-time UAV state awareness
Scenario-Driven UAV Decision System (2025)	Uses telemetry logs as RAG context for LLM-based mission command generation in centralized IoD setup	No vision input; semantic decisions rely only on structured logs, reducing situational richness
FlightGPT (May 2025)	Employs a vision-language navigation model with Chain-of-Thought prompting and reinforcement learning for UAV goal following	No telemetry or RAG integration; not generalizable for broader UAV decision tasks (e.g., landing, rerouting)
AI-Driven UAV & IoT Traffic Optimization (2025)	Combines UAV surveillance + IoT sensors with LLM (Gemini 2.0 Flash) to generate adaptive traffic commands for congestion and emissions reduction	Focused on traffic control; UAVs are passive observers, not active agents; lacks semantic visual grounding or direct UAV reasoning
Aero-LLM (2025)	Distributed LLM framework across UAV, edge, and cloud for mission planning, anomaly detection, and secure comms	Does not fuse vision and telemetry; focuses more on infrastructure, not semantic fusion or real-time decision making

Problem Definition and Research Gap

Problem

- UAV perception mostly vision-based (RGB only).
- Telemetry cues (altitude, yaw, roll, pitch, speed) ignored.
- Lacks **context-awareness** under poor visibility or ambiguous scenes.
- Results in limited perception robustness and **decision reliability**.

Research Gap

- Few works fuse vision + telemetry at representation level.
- Existing fusion = static / heuristic (e.g., simple concatenation).
- Missing **adaptive, learnable** fusion mechanism.
- Lack of interpretability on modality contributions.
- Need for gated fusion framework enabling dynamic weighting and better UAV understanding

Proposed Methodology

- To extract visual embeddings from UAV images using pretrained models (BLIP-2, SegFormer).
- To generate telemetry embeddings using a lightweight MLP encoder on normalized flight parameters.
- To design a gated fusion mechanism that adaptively learns the contribution of each modality.
- To evaluate the fused embeddings using:
    1. t-SNE / UMAP for latent space analysis
    2. Heat Maps for feature saliency
    2. Gate Trends for modality contribution insights
- To compare fused vs. single-modality representations for separability and contextual consistency.

Dataset Preparation

- **Dataset Used: CARLA-based Multi-View UAV Dataset** (RGB + telemetry).

- **Modalities used:**

  1. RGB Front-view images for visual perception. (640 × 480 px, rgb .png format, ~1000 paired samples)
  2. Telemetry JSONs containing altitude, pitch, roll, yaw, velocity, and battery. (5-7 features per frame.)

- Sample pairing matched by frame ID (e.g., 000123.png ↔ 000123.json).

Semantic segmentation:

1. Generated custom semantic maps from RGB using SegFormer
2. Produces per-pixel class maps for contextual understanding.

Additional preprocessing:

1. RGB → resized and normalized before BLIP encoder.
2. Telemetry → normalized numerical tensor for MLP encoder.

System Architechture

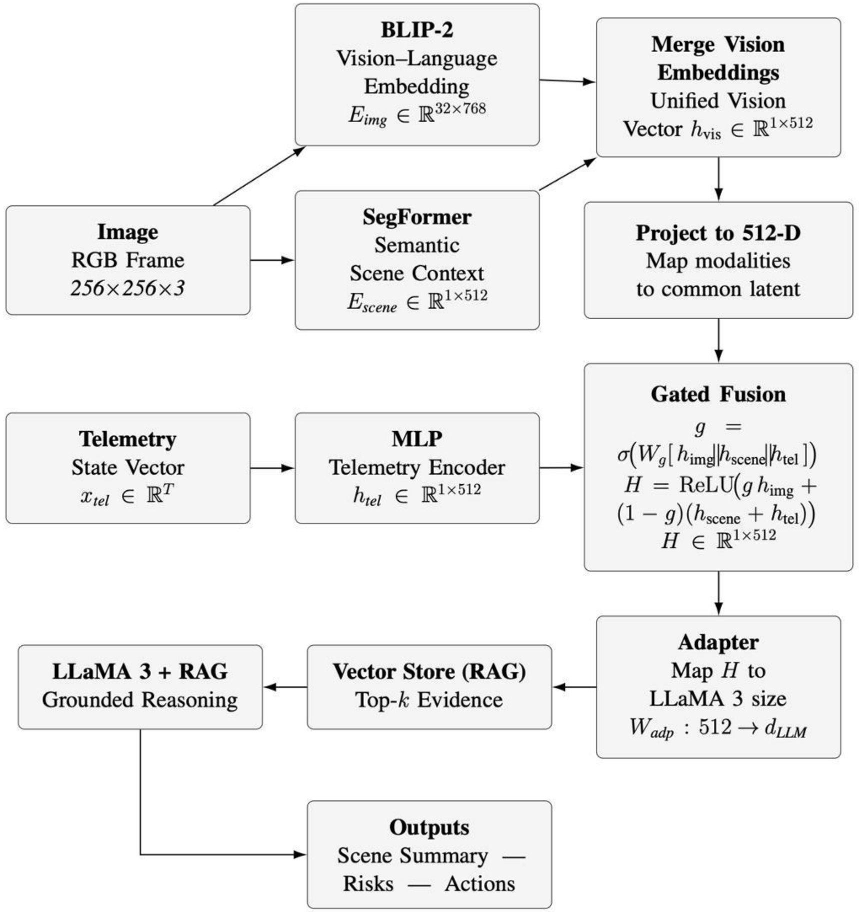


Image
RGB Frame
$256 \times 256 \times 3$

BLIP-2
Vision–Language
Embedding
$E_{img} \in \mathbb{R}^{32\times768}$

SegFormer
Semantic
Scene Context
$E_{scene} \in \mathbb{R}^{1\times512}$

Merge Vision Embeddings
Unified Vision
Vector $h_{\text{vis}} \in \mathbb{R}^{1\times512}$

Project to 512-D
Map modalities
to common latent

Telemetry
State Vector
$x_{tel} \in \mathbb{R}^{T}$

MLP
Telemetry Encoder
$h_{tel} \in \mathbb{R}^{1\times512}$

Gated Fusion
$g =$
$\sigma(W_g[\, h_{\text{img}} \| h_{\text{scene}} \| h_{\text{tel}}\,])$
$H = \text{ReLU}(g\, h_{\text{img}} +$
$(1-g)(h_{\text{scene}} + h_{\text{tel}}))$
$H \in \mathbb{R}^{1\times512}$

Adapter
Map H to
LLaMA 3 size
$W_{adp} : 512 \to d_{LLM}$

Vector Store (RAG)
Top-k Evidence

LLaMA 3 + RAG
Grounded Reasoning

Outputs
Scene Summary —
Risks — Actions

Figure 3.1: Multimodal pipeline

This architecture enables interpretable multimodal understanding suitable for UAV mission support, real-time situational awareness, and operator-assistive decision-making.

SHIV NADAR
UNIVERSITY
CHENNAI

Vision Encoder

Aspect	Vision Embedding (BLIP-2)	Scene Layout / Segmentation (SegFormer)
What it captures	Objects and semantics	Spatial structure and free/obstructed space
Focus	“What is there?”	“Where can I go?”
Output Type	High-level meaning features	Geometry + region boundaries
Examples	“I see a building, some trees, and a road.”	“These pixels are road, these are obstacles.”

Telemetry Encoder

- Telemetry vector per frame:

  Roll, Pitch, Yaw, Velocity, Battery, Altitude → **6 values**

•All values normalized (0–1 range). (to make a dense, learnable embedding)

Architecture:

Input Layer: 6 neurons (one per telemetry feature)
Hidden Layer 1: 64 neurons, ReLU activation
Hidden Layer 2: 128 neurons, ReLU activation + dropout
Output Layer: 256-dimensional feature vector

Output:

•Dense telemetry embedding → **t′ (256-D)**
•Represents UAV state context (orientation, motion, power, altitude)

Flow :

Raw telemetry (6D) → MLP → latent embedding (256D)
→ passed to **Gated Fusion Module** along with vision embedding (256D).

Outcome:

•Numeric flight state -> compact, high-level representation.

SHIV NADAR
UNIVERSITY
CHENNAI

Working of Gated Fusion

- **Adaptive trust** across modalities
- **Clear vision:** rely on visual features
- **Low visibility:** shift to scene layout + telemetry
- Gate adjusts weights per **512-D** feature channel

Data used	Situation	What you trust
BLIP-2 embeddings	Clear visibility	You trust your eyes
Segformer masks	Very dark	You trust your sense of touch + memory
Telemetry data	Slippery floor	You trust body balance signals

How Gated Fusion works?

- **Fusion Formula**

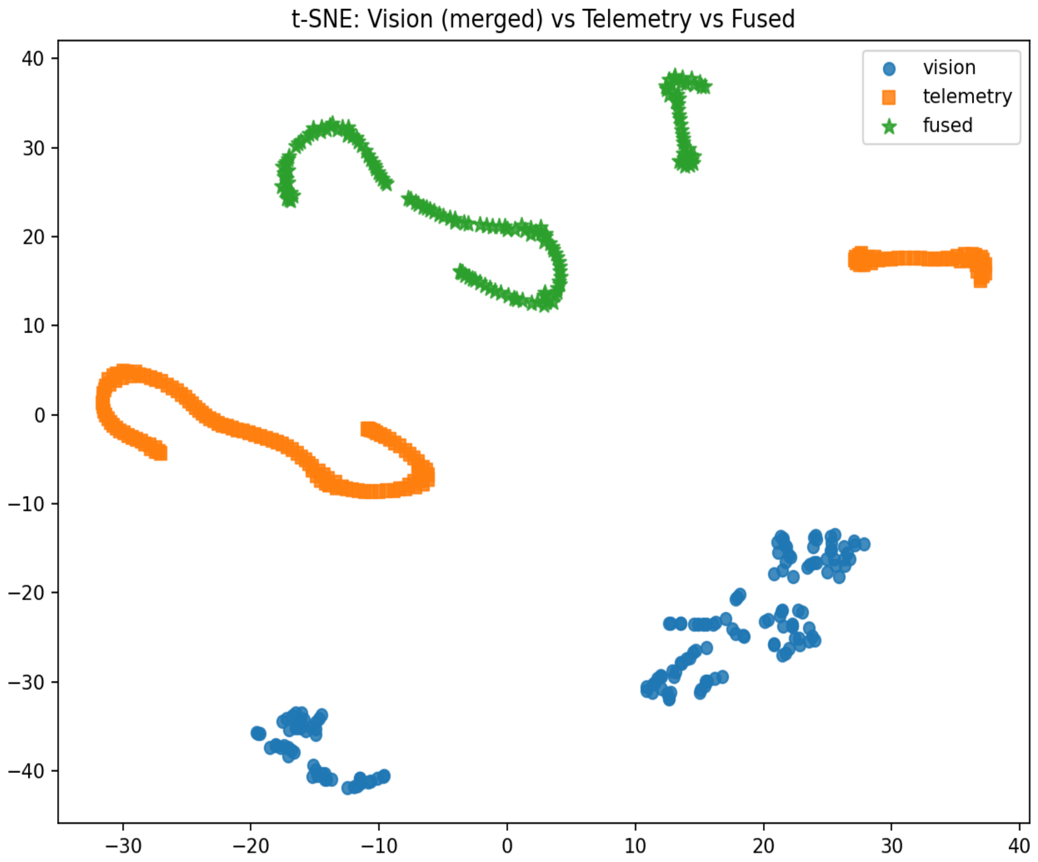
$$g = \sigma(W_g \cdot [h_{img} \,||\, h_{scene} \,||\, h_{tel}])$$

- **g** acts as soft attention mask.
- Higher g - Representation relies on semantic image cues.
- When g is low, the representation prioritizes spatial scene structure and telemetry.

$$h_{fused} = \mathbf{ReLU}(g \odot h_{img} + (1 - g) \odot (h_{scene} + h_{tel}))$$

- Where:
- $\odot$ = element-wise multiplication
- Output H → fused 512-D representation

Results: t-SNE Plot (Vision vs Telemetry vs Fused)

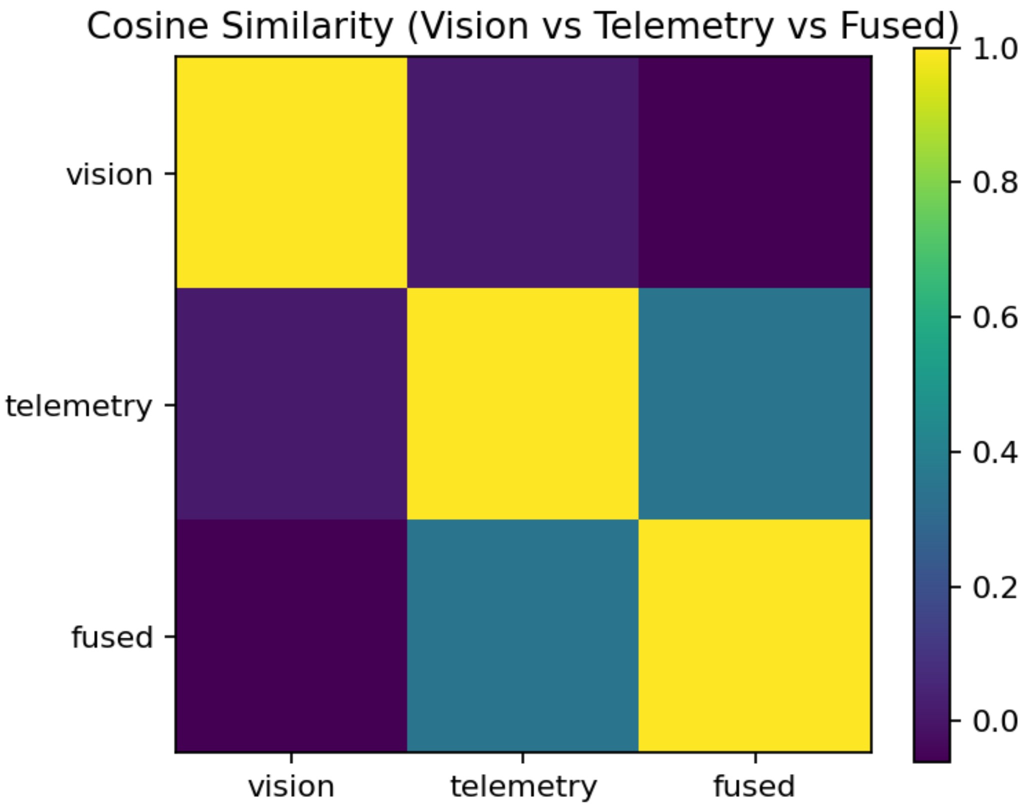


Interpretation :

- The **vision features** group together because they describe **object appearance and scene meaning**.
- The **telemetry features** group together because they represent **drone state** (altitude, angle, speed).
- The **fused points are located between the two clusters**, not on top of one cluster.

SHIV NADAR UNIVERSITY CHENNAI

Cosine Similarity Heatmap



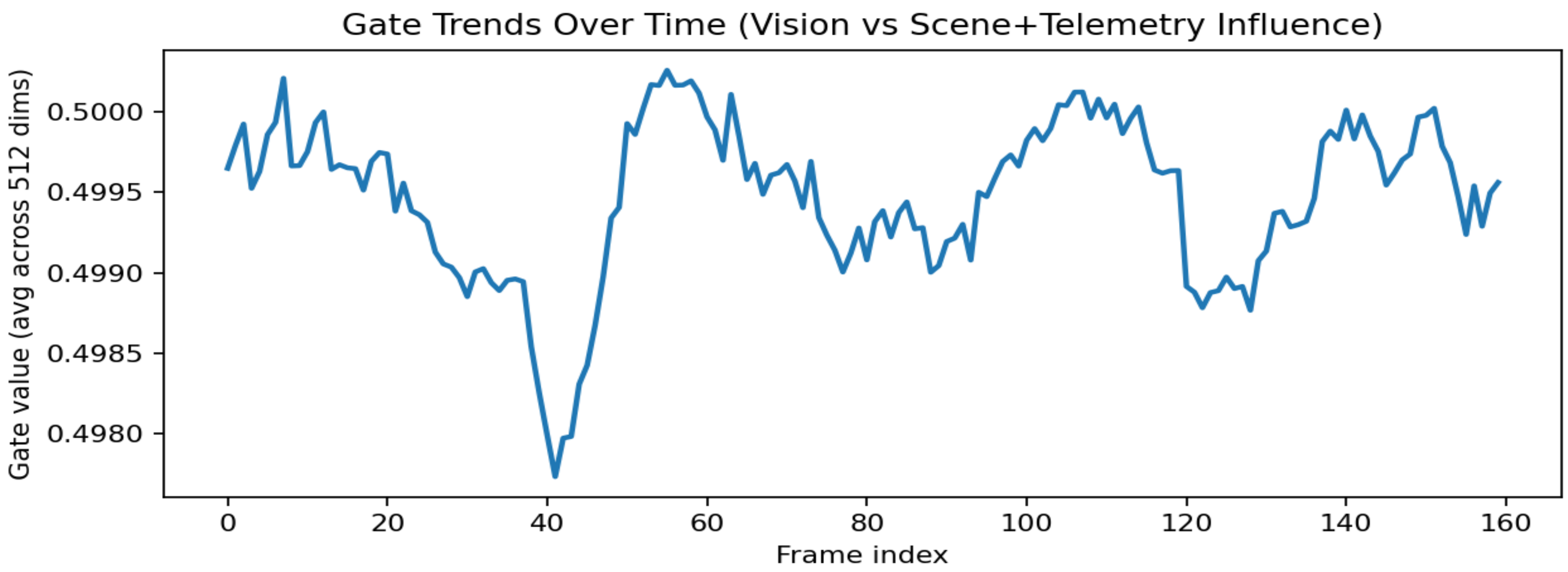
Interpretation:

- **Vision ↔ Vision = 1** (same feature → same representation)
- **Telemetry ↔ Telemetry = 1** (same reason)
- **Vision ↔ Telemetry** is **low**, meaning they provide **different types of information**.
- **Fused ↔ Vision** and **Fused ↔ Telemetry** are **moderate**, not extreme.

Why this is important:

- It shows that the fused output **does not ignore any modality** — it takes meaningful contribution from both.

Gate Trend Plot (Over Time)



What the line graph shows:

- The **gate value** is a number between **0 and 1**.
- Higher gate: **model trusts vision more**
- Lower gate: **model trusts telemetry + scene layout more**

How to read it:

- The line changes over time (frame index).
- When visuals are clear → values are slightly higher.
- When visuals become unclear (motion shake, lighting change) → values drop.