Real-Time Vehicle Inference System with Occlusion Handling

Overview

This document describes the modified real-time vehicle inference system, which explicitly demonstrates how the system handles the scenario where a person goes out of the camera frame (occlusion).

The core principle is that the **Unscented Kalman Filter (UKF) maintains the trajectory identity** through prediction, while the **Trajectory Matcher correctly ceases assignment** due to the lack of visual data.

Demonstration Results

The simulation introduced a controlled occlusion period between **Frame 151 and Frame 250** (a duration of 100 frames, or 3.33 seconds).

Key Observations During Occlusion

Frame Range	UKF Uncertainty (Trace of Covariance)	Assignment Status	Observation
0000 - 0150	Low (e.g., 2.10 - 3.09)	UWB 0 -> Cam 101	**Identified.** UKF is updated by camera data.
0151 - 0250	**Grows rapidly** (e.g., 3.09 -> 351.24)	**None** (OCCLUDED)	**Occlusion.** UKF relies solely on prediction, causing uncertainty to skyrocket. Matcher correctly finds no valid assignment.
0251 - 0300	**Drops immediately** (e.g., 351.24 -> 2.49)	UWB 0 -> Cam 101	**Re-identified.** UKF update resumes, uncertainty drops, and assignment is restored.

Log Snippet Demonstrating Occlusion

Plain Text

```
Frame 0120 | Time: 4.00s | Proc Time: 0.85ms | UKF Uncert: 2.46 |
Assignment: UWB 0 -> Cam 101
Frame 0150 | Time: 5.00s | Proc Time: 0.90ms | UKF Uncert: 3.09 |
Assignment: UWB 0 -> Cam 101 (OCCLUDED)
--- Occlusion Start at Frame 151 ---
Frame 0180 | Time: 6.00s | Proc Time: 0.14ms | UKF Uncert: 44.85 |
Assignment: None (OCCLUDED)
Frame 0210 | Time: 7.00s | Proc Time: 0.12ms | UKF Uncert: 150.90 |
Assignment: None (OCCLUDED)
Frame 0240 | Time: 8.00s | Proc Time: 0.13ms | UKF Uncert: 351.24 |
Assignment: None (OCCLUDED)
--- Occlusion End at Frame 250 ---
Frame 0270 | Time: 9.00s | Proc Time: 0.75ms | UKF Uncert: 2.49 |
Assignment: UWB 0 -> Cam 101
```

Code Modifications

1. `vehicle_data_streamer.py`

A controlled occlusion window was introduced to simulate the person leaving the frame:

Python

```python
# In vehicle_data_streamer.py
# Introduce occlusion period: Frames 150 to 250 (3.33 seconds)
if 150 <= frame <= 250:
    camera_tracklets = [] # Person out of frame
else:
    # ... generate tracklet data ...
```

2. `main_realtime_inference.py`

The main script was updated to explicitly handle the empty `camera_tracklets` list:

Python

```python
# In main_realtime_inference.py
is_visible = len(camera_tracklets) > 0

if is_visible:
    # Update UKF with camera tracklet position
    self.ukf.update_camera(tracklet_pos, frame, timestamp)
else:
    # Person is out of frame (occluded). UKF continues to predict.
    # The UKF update is skipped, relying purely on the prediction step.
    # ... logging occlusion start/end ...
```

Conclusion

The modified code successfully demonstrates the system's resilience to camera tracklet loss:

1. **Identity is Maintained:** The UKF continues to track the UWB tag's predicted trajectory.
2. **Uncertainty is Tracked:** The growing UKF uncertainty provides a measure of how unreliable the prediction is during occlusion.
3. **Assignment is Correct:** The trajectory matcher correctly stops assigning the UWB tag to a camera tracklet when the person is not visible.

Files Delivered

File	Description
`uwb_ukf_2d_streaming.py`	Streaming UKF with buffer management (Unchanged)
`trajectory_matching_2d_streaming.py`	Real-time trajectory matching (Unchanged)
`vehicle_data_streamer.py`	**MODIFIED:** Introduces a controlled occlusion period
`main_realtime_inference.py`	**MODIFIED:** Handles and logs the occlusion scenario
`REALTIME_VEHICLE_INFERENCE_OCCLUSION_README.md`	This documentation

The complete, modified code is attached.