FINAL REPORT - ENG 573

Benchmarking different SLAM Approaches

Narathip Rodwarna (nr44), Sridharan Subramanian (ss233), Yifei Li (yifei16)

I. Abstract

This project evaluates and enhances visual SLAM algorithms to improve UAV performance in GPS-denied environments. We compare ORB-SLAM3, OpenVSLAM, and LSD-SLAM using Absolute Pose Error (APE) and Relative Pose Error (RPE) metrics. Additionally, our fusion approach aims to enhance system robustness and accuracy through intelligent algorithm switching. Experiments on diverse datasets show that OpenVSLAM generally outperforms others in accuracy and tracking recovery. Despite initial hypotheses that LSD-SLAM would perform better in low-texture areas, it does not effectively complement OpenVSLAM in challenging scenarios. Future work should explore alternative direct algorithms and the impact of different Bag-of-Words (BoW) implementations. Furthermore, while RNN-LSTM models predict trajectory well, challenges remain in quaternion angle learning and agile Z motion, suggesting potential benefits from Encoder-Decoder architectures. Further research into failure analysis with logistic regression or other classification algorithms is recommended with a more comprehensive dataset.

II. Introduction

Autonomous navigation, localization, and mapping are crucial for the effective operation of Unmanned Aerial Vehicles (UAVs) in unknown environments where other modes of perception are unavailable. This project aims to evaluate and improve various visual SLAM algorithms to equip UAV systems with enhanced capabilities, allowing them to perform efficiently in such challenging settings. By focusing on these technologies, we seek to address the limitations faced by UAVs in GPS-denied environments and contribute to advancements in autonomous aerial operations.

This study evaluates and compares three prominent visual SLAM algorithms: ORB-SLAM3, OpenVSLAM, and LSD-SLAM. We assess their performance using Absolute Pose Error (APE) and Relative Pose Error (RPE) metrics. Building on these individual evaluations, we propose a novel fusion approach to enhance system robustness and accuracy. Our methodology includes failure prediction mechanisms to enable intelligent algorithm switching, ensuring system reliability. Section VII presents the comparative analysis, Sections VIII and IX discuss online and offline fusion, respectively, Section X addresses technical challenges and limitations, and Section XI concludes with our findings and suggestions for future work.

III. Algorithm Review

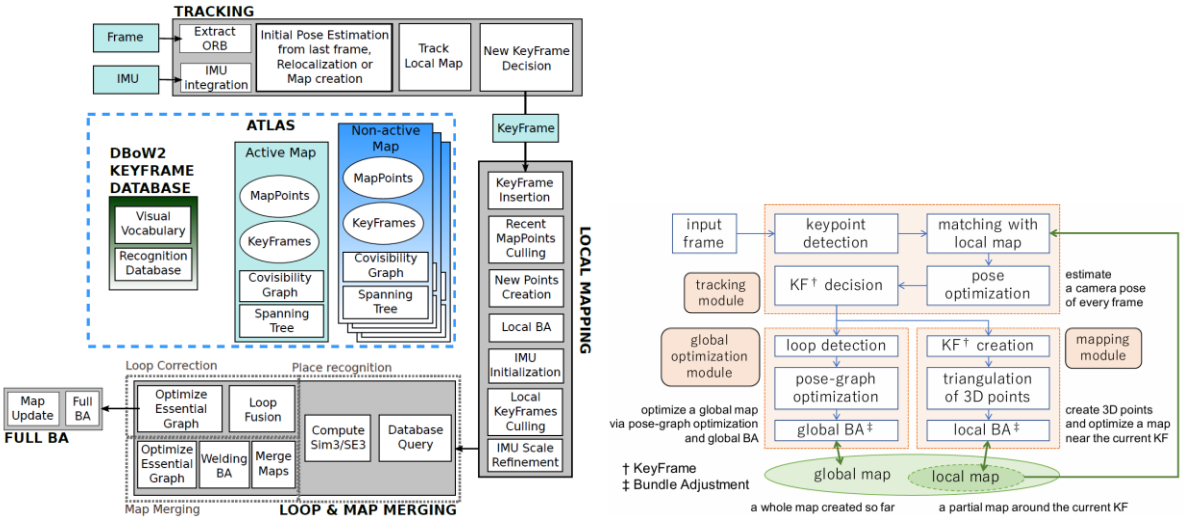
Visual SLAM algorithms can be categorized into direct and indirect methods, each with distinct approaches to processing visual data for localization and mapping. Direct VSLAM utilizes raw pixel intensities, minimizing photometric errors to estimate motion and create dense maps. This method excels in low-texture environments but can be sensitive to lighting changes. In

contrast, indirect VSLAM extracts and tracks specific features from images, minimizing geometric reprojection errors. It performs well in textured scenes and is more robust to lighting variations, but typically produces sparser maps. The selection of a VSLAM approach is influenced by the specific application requirements and the characteristics of the target environment. Moreover, these algorithms can be used in conjunction to leverage their complementary strengths and mitigate their individual weaknesses.

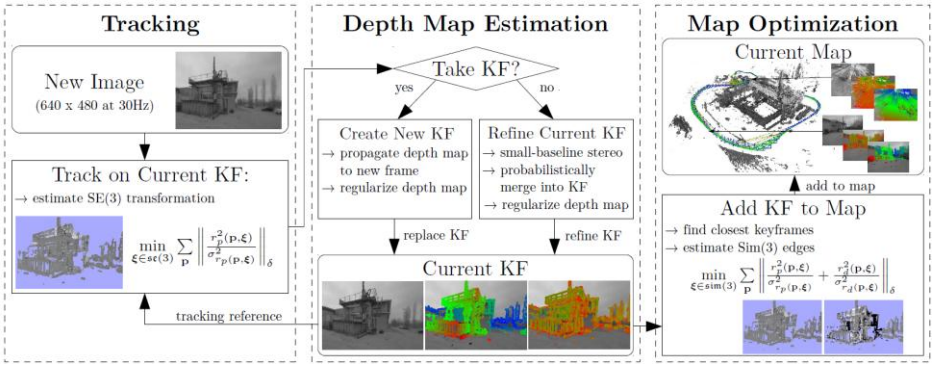
First, ORB-SLAM3 is a multimap SLAM system that supports monocular, stereo, and RGB-D cameras with both pinhole and fisheye lens models (Campos et al., 2021). It is built on top of ORB-SLAM. This algorithm uses ORB features because they are fast to extract and match. The key innovation is a multimap SLAM capability that allows merging of disconnected maps and multi-session SLAM. ORB-SLAM3 consists of several main components: the Atlas multi-map representation, a tracking thread for real-time pose estimation, a local mapping thread for map refinement, and a loop and map merging thread for detecting common regions and correcting loops. Overall, ORB-SLAM3 represents a significant advancement in SLAM technology, offering superior performance across a wide range of sensor configurations and environmental conditions

Second, OpenVSLAM is a framework designed to provide high usability and extensibility for various applications (Sumikura et al., 2019). Unlike existing frameworks such as ORB-SLAM or LSD-SLAM, OpenVSLAM addresses limitations in usability and customization by offering a modular structure and compatibility with multiple camera types, including perspective, fisheye, and equirectangular models. It uses an indirect SLAM approach for robust tracking and mapping, dividing its functionality into three main modules: tracking, mapping, and global optimization. A key feature is its ability to store and load maps, allowing maps to be reused for localization tasks later. Benchmark evaluations demonstrate that OpenVSLAM achieves comparable or better tracking accuracy and faster processing times than ORB-SLAM2. The original OpenVSLAM utilizes FBoW (Fast Bag of Words), which is a more optimized and efficient than DBoW2 library used in ORB-SLAM3, offering faster processing and reduced memory usage for tasks such as feature matching and place recognition.

Last, LSD-SLAM (Large-Scale Direct Monocular SLAM) is a direct monocular SLAM algorithm that builds large-scale, consistent maps of the environment in real time by leveraging direct image alignment and semi-dense depth maps (Engel et al.,2014). Unlike feature-based methods, it operates directly on image intensities, enabling it to utilize all available image information for accurate pose estimation and 3D reconstruction. The system consists of three main components: tracking, depth map estimation, and map optimization. Tracking estimates the camera's pose by minimizing photometric errors between frames using a scale-aware similarity transform that detects and corrects scale drift. Depth maps are generated through pixel-wise stereo comparisons and refined with spatial regularization, focusing on regions with sufficient texture to create semi-dense reconstructions. The global map is represented as a pose graph of keyframes connected by similarity transforms, allowing for loop closure detection and global optimization. LSD-SLAM incorporates probabilistic uncertainty into its tracking process to improve robustness and runs efficiently in real time on a CPU. Its ability to handle large-scale environments and texture-less regions makes it well-suited for applications in robotics.



(a) (b)



(c)

Fig.1 Algorithm Architecture
(a) ORB-SLAM3 (Campos et al., 2021 (b) OpenVSLAM (Sumikura et al., 2019)
(c) LSD-SLAM (Engel et al.,2014)

IV. Setup

i.) Environment Setup

To ensure a consistent and replicable development environment, Docker was used to containerize each VSLAM algorithm.

a. Ubuntu and ROS

This project employs Docker to implement three SLAM systems with distinct environmental configurations. LSD-SLAM operates on ROS Indigo with Ubuntu 14.04, while ORB-SLAM3 requires ROS Noetic on Ubuntu 20.04. In contrast, OpenVSLAM runs directly on Ubuntu 22.04 without ROS dependencies.

b. Docker

Installation: Docker was installed on Ubuntu using the standard Docker Engine installation guide.

Containerization: Each SLAM algorithm (OpenVSLAM, LSD-SLAM, ORB-SLAM3) was installed within separate Docker containers to ensure isolation and avoid any conflicts between dependencies. This setup allowed seamless switching between different algorithms without interfering with one another.

While ORB-SLAM3 and OpenVSLAM utilized existing containers from third-party repositories (Jahaniam, 2024) and original authors (“Stella CV Documentation,” 2024) respectively. Further, we developed a custom Docker environment for LSD-SLAM to ensure system compatibility.

ii.) Algorithm Setup

Each SLAM algorithm was configured with the appropriate camera calibration parameters. For ORB-SLAM3, the image loading routines were specifically modified to process the sequential imagery from the Kagaru dataset. The algorithms were installed following their respective documentation: OpenVSLAM through the official installation guideline, ORB-SLAM3 using third-party installation guide, and LSD-SLAM through our custom Docker with our guideline.

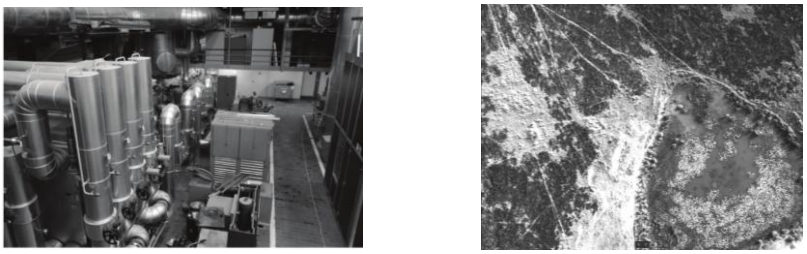
V. Dataset

i.) EuRoC MAV Dataset

The EuRoC MAV dataset (Burri et al., 2016) was downloaded from the official website ("KMAV visual-inertial datasets," 2024). The dataset is composed of 11 sequences, each providing images and inertial data from a Micro Aerial Vehicle (MAV) in indoor environments including a machine hall and 2 Vicon rooms. The sequences range from easy (slow movements, little rotation) to difficult (high speed, complex rotations).

ii.) Kagaru Airborne Dataset

The Kagaru Airborne dataset (Warren et al., 2012) provides data captured from an airborne platform, focusing on downward-facing stereo imagery, which is particularly useful for aerial SLAM evaluations. The Kagaru dataset can be downloaded from the official website (Warren et al., 2024). The dataset includes stereo images taken from a small Unmanned Aerial System, which operates in large outdoor environments. The dataset has ground truth measurements obtained through GPS instrumentation (latitude, longitude, and altitude). Two preprocessing steps were implemented to prepare the data for visual SLAM evaluation. First, the GPS-based ground truth measurements were transformed from geodetic coordinates into a Cartesian coordinate reference system to facilitate geometric calculations. Second, the image sequences were converted to ROS bag format because LSD-SLAM lacks an input channel for timestamps, making it unable to generate a trajectory based on them. Therefore, using a ROS bag is necessary.



(a) (b)

Fig.2 Example of dataset (a) EuRoC MAV (b) Kagaru Airborne

Table.1 Dataset characteristics

Dataset	Test Case	Length [meter]	Duration [second]
EuroC (Indoor)	MH_01_easy	80.6	182
	MH_02_easy	73.5	150
	MH_03_medium	130.9	132
	MH_04_difficult	91.7	99
	MH_05_difficult	97.6	111
	V1_01_easy	58.6	144
	V1_02_medium	75.9	84
	V1_03_difficult	79.0	105
	V2_01_easy	36.5	112
	V2_02_medium	83.2	115
	V2_03_difficult	86.1	115
Kagaru (Outdoor)	Kagaru	11,100.7	363

VI. Evaluation Setup

i.) Data Preprocessing

- Trajectory Extraction: The output trajectories from each SLAM algorithm (ORB-SLAM3, OpenVSLAM, and LSD-SLAM) were saved in standard format (TUM format for Evo). These trajectories were aligned with the ground truth from the datasets (EuRoC MAV and Kagaru Airborne) to ensure a direct comparison.
- Ground Truth Preparation: The ground truth poses provided by the datasets were formatted to be compatible with Evo.

ii.) Evaluation Metrics:

- Absolute Pose Error (APE): This metric is used to evaluate the accuracy of pose estimation. It measures the difference between the estimated pose of a robot or camera and its true pose (ground truth) at each time step.
- Relative Pose Error (RPE): RPE was calculated to measure the drift between consecutive poses in the trajectory, giving insight into how well the SLAM system performs over time. RPE was particularly useful for identifying small-scale errors that accumulate over long trajectories.

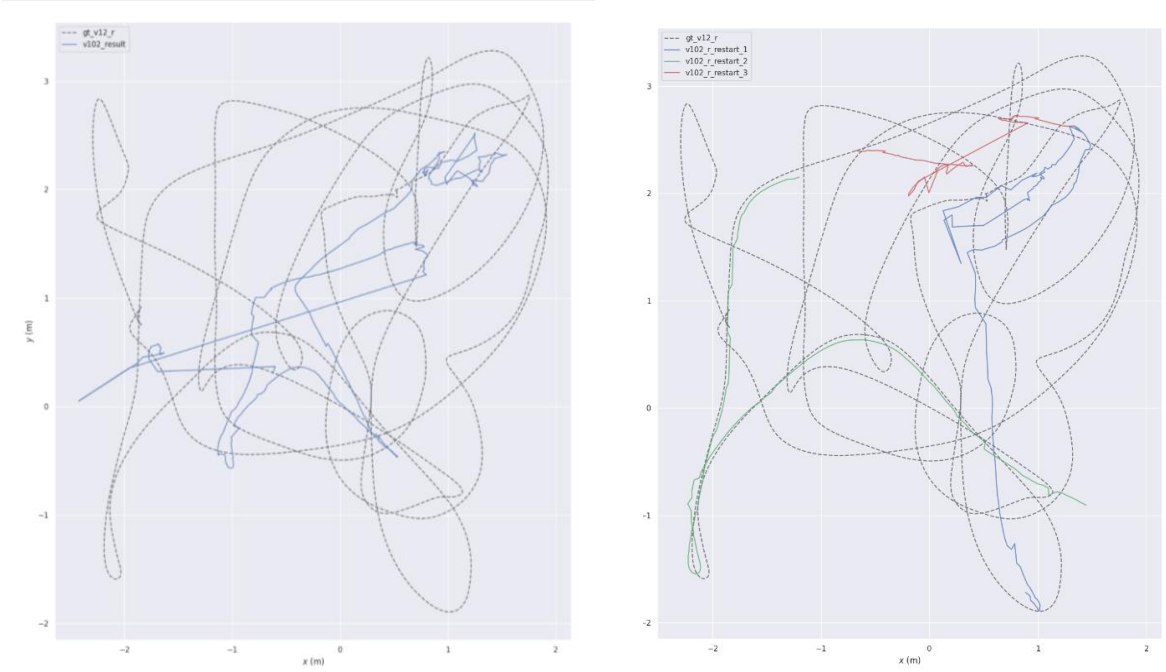
iii.) Evo Package

The Evo package (Grupp, 2024) is a Python-based tool designed for evaluating the performance of SLAM and odometry algorithms. It provides metrics like Absolute Trajectory Error (ATE) and Relative Pose Error (RPE), which are essential for comparing estimated trajectories with ground truth data. Evo also provides visualization tools to plot the estimated trajectory against the ground truth.

VII. Algorithm Evaluation

This section provides evaluation of the three selected SLAM algorithms: ORB-SLAM3, OpenVSLAM, and LSD-SLAM. Each algorithm was evaluated based on its performance using the EuRoC MAV and Kagaru datasets, focusing on key metrics APE and RPE. Each dataset, we run 3 times and select the median result. We used the evo package for evaluation. In each evaluation, the trajectories are aligned with command '-as' from the evo package.

Using the evo package, it's easy to draw the plot of the trajectory of the result. However, sometimes the slam algorithm may lose tracking and the trajectory will be divided into several pieces. The evo package cannot handle this very well. So we cut the trajectory into small pieces, each piece having equal length and the evo package can align each piece with the ground truth as shown in Fig.3.



(a) (b)

Fig.3 Original and splits trajectory result of V102 dataset (a) Original (b) Splitted

OpenVSLAM exhibits superior performance characteristics, particularly in tracking robustness and Absolute Pose Error (APE) metrics. In the comprehensive Kagaru trajectory evaluation, which represents the most extensive test sequence, the system demonstrated remarkable stability by experiencing an average of only two tracking losses, significantly outperforming alternative algorithms. Additionally, it achieved faster map initialization compared to ORB-SLAM3.

Fig.4 Blurred images that cause loss of tracking

ORB-SLAM3 performs well in the easy and medium case of the machine hall and Vicon room sequences in the EuRoC dataset, where there is no high speed motion. It experiences minimal tracking loss and is able to recover and maintain map continuity. However, its performance deteriorates in low-texture environments, such as those encountered in the Kagaru dataset. As noted in the original paper, ORB-SLAM3 relies on ORB features for tracking, which

struggle to extract sufficient key points from low-texture surfaces like grass fields or roads. This limitation leads to frequent tracking failures in such scenarios, highlighting a key challenge for feature-based SLAM systems when operating in environments with sparse visual information.

Fig.5 Dark scene

LSD-SLAM, the only image intensity-based visual SLAM algorithm in our study, achieves the highest APE in almost all scenarios. Even with perfect tracking without any loss of tracking, its performance is significantly inferior to the other two algorithms. In medium and difficult cases, LSD-SLAM struggles with tracking recovery, leading to incomplete maps that fail to capture the full trajectory, often appearing as fragmented segments. It was initially hypothesized that LSD-SLAM might outperform other algorithms in areas where they typically fail, particularly in low-texture environments. However, experimental results indicate that LSD-SLAM also fails to demonstrate superior performance in these conditions, facing similar difficulties in maintaining accurate tracking. Further, it is mentioned by another paper that the direct method does not perform well with dark images, but it cannot be evaluated in this test because it failed before entering the dark area.

Table.2 APE result

APE	**OpenVSLAM**	**ORB-SLAM3**	**LSD-SLAM**
MH01	0.0292	0.0331	0.5519
MH02	0.0670	0.0267	0.2873
MH03	0.0379	0.0327	2.7663
MH04	0.0997	0.0496	5.6413
MH05	0.0498	0.1473	1.0855
V101	0.0815	0.0821	0.2406
V102	0.0668	0.0613	0.4309
V103	0.0955	0.1332	0.4926
V201	0.0504	0.1523	0.1654
V202	0.1169	0.0498	0.0164
V203	0.0977	0.0748	1.0379
Kagaru	11.271	8.4890	222.2001

Table.3 RPE result

RPE	OpenVSLAM	ORB-SLAM3	LSD-SLAM
MH01	0.0230	0.0232	0.0210
MH02	0.0242	0.0240	0.0209
MH03	0.0561	0.0562	0.0515
MH04	0.0550	0.0563	0.0528
MH05	0.0497	0.0522	0.0576
V101	0.0226	0.0235	0.0201
V102	0.0456	0.0472	0.0263
V103	0.0430	0.0508	0.0382
V201	0.0163	0.0212	0.0143
V202	0.0371	0.0371	0.0301
V203	0.0555	0.0503	0.0624
Kagaru	1.3350	1.2360	0.7298

Table.4 Comparison of output trajectory on Kagaru dataset

Method	Percentage of Completed Trajectory	APE		RPE	
		mean	median	mean	median
ORB-SLAM3	39.35	8.379	8.496	1.237	1.140
OpenVSLAM	69.33	12.310	11.683	1.384	1.330

A critical challenge in visual SLAM systems is their ability to recover from tracking failures. Our comparative analysis reveals significant variations in recovery capabilities across the three evaluated algorithms when tested on the Kagaru that has the longest trajectory.

OpenVSLAM demonstrated superior recovery performance through its robust map merging mechanism. When tracking is lost, the system successfully reconstructs and merges disconnected trajectory segments into a coherent global map, maintaining spatial consistency throughout the sequence. However, overall it takes longer to recover than ORB-SLAM3.

ORB-SLAM3's framework provides moderate recovery capabilities through its multimap management system. While it successfully creates and maintains multiple submaps, the merging process is more restrictive. When submap fusion fails, the system retains only the initially created map, potentially discarding valuable trajectory information. As shown in Fig. 6, three submaps were generated (blue, red and pink), but only the first submap created was preserved in the final

output. Table 4 further compares the resulting trajectories between OpenVSLAM and ORB-SLAM3.

LSD-SLAM demonstrated limited recovery capabilities, primarily due to its reliance on previously observed keyframes for relocalization. If areas have not been previously explored, the system is unable to recover, which represents a significant disadvantage. One potential improvement could involve increasing the frequency of keyframe saving. However, this approach is inefficient and may not effectively resolve the underlying limitations.

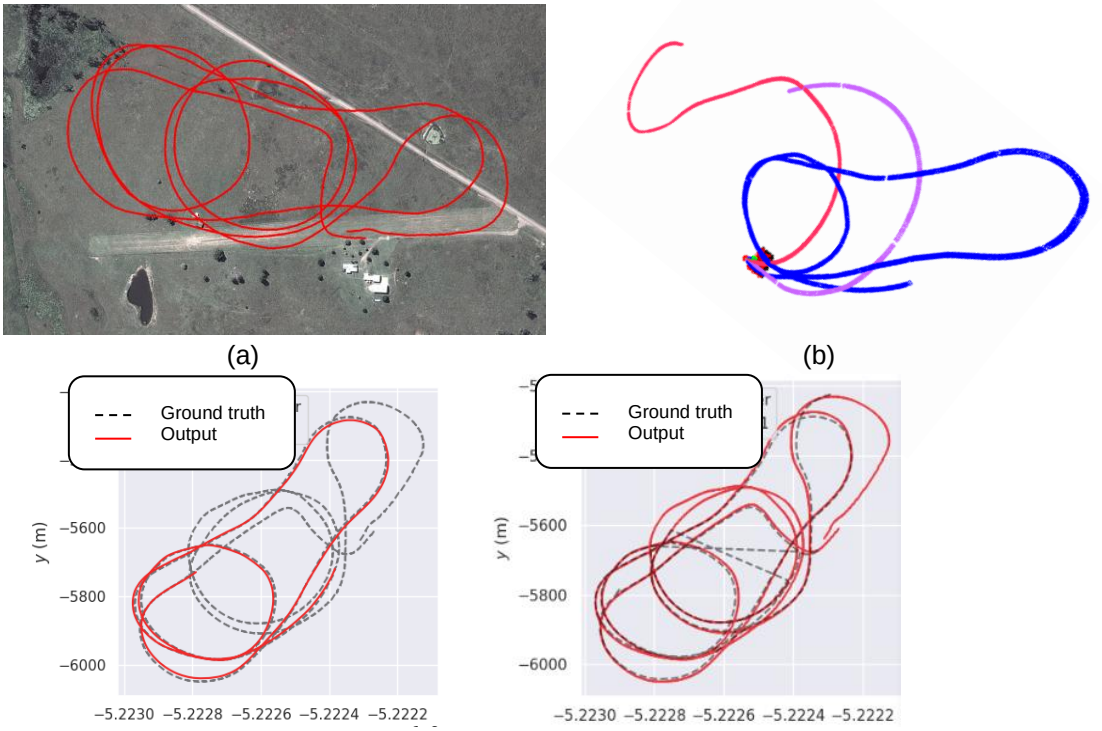


Fig.6 Trajectory outputs on the Kagaru dataset: (a) Ground truth, (b) Atlas of ORB-SLAM3, (c) Output trajectory of ORB-SLAM3, (d) Output trajectory of OpenVSLAM

VIII. Trajectory Fusion

i.) RNN-LSTM Fusion Implementation

a. Intro & Aim

As follows from previous headings, we've seen the trajectory prediction becomes crucial especially when looking for a system where elaborate tests aren't possible due to various constraints. Since this prediction helps with planning and decision-making by anticipating the movement or future positions of agents, or in this particular context a flight, robust trajectory prediction aids in the improvement of UAV pose estimates in the setting of Simultaneous Localization and Mapping (SLAM), resulting in more precise navigation.

Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) networks, are better suited for handling sequential data like trajectories here because they are designed to capture temporal dependencies in the input data. We train the LSTM model from keras, from scratch, with trajectory outputs from the SLAM modules such as OpenVSLAM and

ORB-SLAM3. This work aimed towards understanding how well an LSTM network can capture not just position but also quaternion data throughout an agile flight. The model leverages sequential learning to predict each subsequent position and orientation(through quaternion) based on past positions and orientations. This can be used in case there are any discrepancies in the trajectory dataset, or when there is missing data/tracking during agile movement of the drone of flight.

b. Working

This implementation of RNN works by training an LSTM model using output data from both ORB-SLAM and OpenVSLAM. This also involves aligning them with corresponding ground-truth trajectories. This aligning with Ground truth can be done by comparing the Absolute Pose Error of both modules(or whichever SLAM modules that are put into use case for that matter) and synchronizing with the output with least APE.

This work then follows to form a tool that can look into the ability of LSTM to understand and improve the quality of existing SLAM trajectory predictions especially into intricate temporal details of orientation changes. The RNN-LSTM model learns to recognize patterns in the trajectories generated by SLAM and aims to refine them towards a more accurate representation of the ground truth. Here, the input trajectories from both SLAM systems are synchronized with ground truth data based on timestamps. As part of the LSTM, a sequence of these input trajectory points is then fed into the LSTM model, which is trained to predict future ground-truth positions and orientations. We have tested multiple configurations of hyperparameters such as LSTM units, sequence length, dropout rate, and learning rate to find the best-performing model based on validation loss. The best model based on the validation loss is then used to predict trajectories for test data which are also synced with each other, and finally, we save the output with corresponding timestamps for further use or evaluation. These timestamps are finally attached to the predicted trajectory that comes out of LSTM without timestamps.

c. Data Preparation

Ground Truth:

The trajectory of the drone detailing its x,y,z coordinates and the orientation in terms of quaternion form the major part of ground truth input for the LSTM training. Though there are other variables like total velocity, and angular velocity at each timestamp, we discard this information to focus only on the pose of the drone or flight. These values accompanied with a timestamp are collected from the EuRoC MAV dataset, and processed with Evo to convert it in terms of TUM format. This TUM format is ***<timestamp x y z qx qy qz qw>.*** This makes it easier in evaluation using evaluation pipelines like Evo.

SLAM trajectory outputs:

Once we run the OpenVSLAM algorithm and ORB SLAM-3 algorithms, we extract the output trajectory in the form of .txt files containing the timestamps and information about pose(position and orientation in quaternion).

Data format:

This output data is prepared in the same format of TUM dataset, with ***<timestamp x y z qx qy qz qw>*** in order to be able to run the evaluation pipeline. Here, the x,y,z positions are in metres.

How timestamps were synchronized with GT and trajectories outputs?:

Before using the dataset prepared above to train on LSTM, it is necessary to make sure both the trajectory outputs from OpenVSLAM and ORB SLAM-3 algorithms contain the same amount of data. This is essential as the RNN learns from the pattern of drone movement from each of these algorithms. Here, the timestamps are extracted and sent into a fusion algorithm to get and match with the closest timestamps in the other algorithm. The remaining extra timestamps are discarded from the trajectory file which has the maximum number of trajectories in it.

This way, both the trajectory outputs are synced. The second way to do this is also by normalizing and clipping the timestamps to [0,1] so that it can universally be applied to any algorithm with its own timestamp definition. This method is also applied to ground truth as we use that in the training as outputs. The ground truth is synced with either of the trajectories based on which algorithm yields the smallest Absolute Pose Error.

d. Dataset

Training & Testing dataset:

Multiple configurations of training have been performed. We've primarily used the EuRoC MAV dataset from both the Vicon rooms, and the Machine Hall environments as it covers a wide range of slow to agile movement of the drone.

e. EuRoC MAV

Vicon Rooms:

V101	Easy
V102	Medium
V103	Difficult with agile movement
V201	Easy
V202	Medium
V203	Difficult with agile movement

Machine Hall:

MH01	Easy1
MH02	Easy2
MH03	Medium
MH04	Difficult with agile movement
MH05	Difficult with most agile movement

The training was performed using the MH01 easy as the input, and testing performed on the MH05 difficult to measure the performance of the LSTM algorithm to adapt to much agile movement. On the other hand, training with MH05 as input, and testing out on MH01 proved to give better results on the X,Y,Z components prediction and it closely followed the trajectory of MH04(similarly agile) when compared to MH01 as training input. However, the training time was found to be the same for both MH01 and MH05.

f. About the Model and RNN Pipeline overview

As with an LSTM, the trajectory data is converted into sequences for LSTM input. We use a sliding window approach to create overlapping trajectory sequences of fixed lengths. This whole architecture is based on an LSTM-based regressor that's made to predict vectors from sequences. This lightweight model utilizes Lstm_units that capture the data complexity and the dropout layers ignore the input neurons for preventing overfitting of the training data as we try to keep the training input very short.

g. Hyperparameters & Optimization Explored

Here, we explored several hyperparameters to find the optimal set of the same. This started with the number of LSTM units(from the set of (32, 64, 128)). The aim of exploring this

was to identify the model's capacity to learn sequential trajectory inputs. Later, the number of sequential points were also explored along with the dropout rates from the set of (0.2,0.3,0.4). Dropout was used to prevent overfitting by deactivating neurons/units during training. This was particularly needed in this work as the size of the dataset is quite limited compared to training with images which aren't sequential in general. Additionally, the learning rates(0.0001,0.001,0.01) were also tested to figure out the model's ability to update the weights. Additional improvements can be made in terms of optimization by utilizing early stopping.

h. Training Results

This below image shows what the training loss looks like with 50 epochs of training with MH01(easy). Also, the best 5 and the worst 5 validation losses have been taken for the visualization.

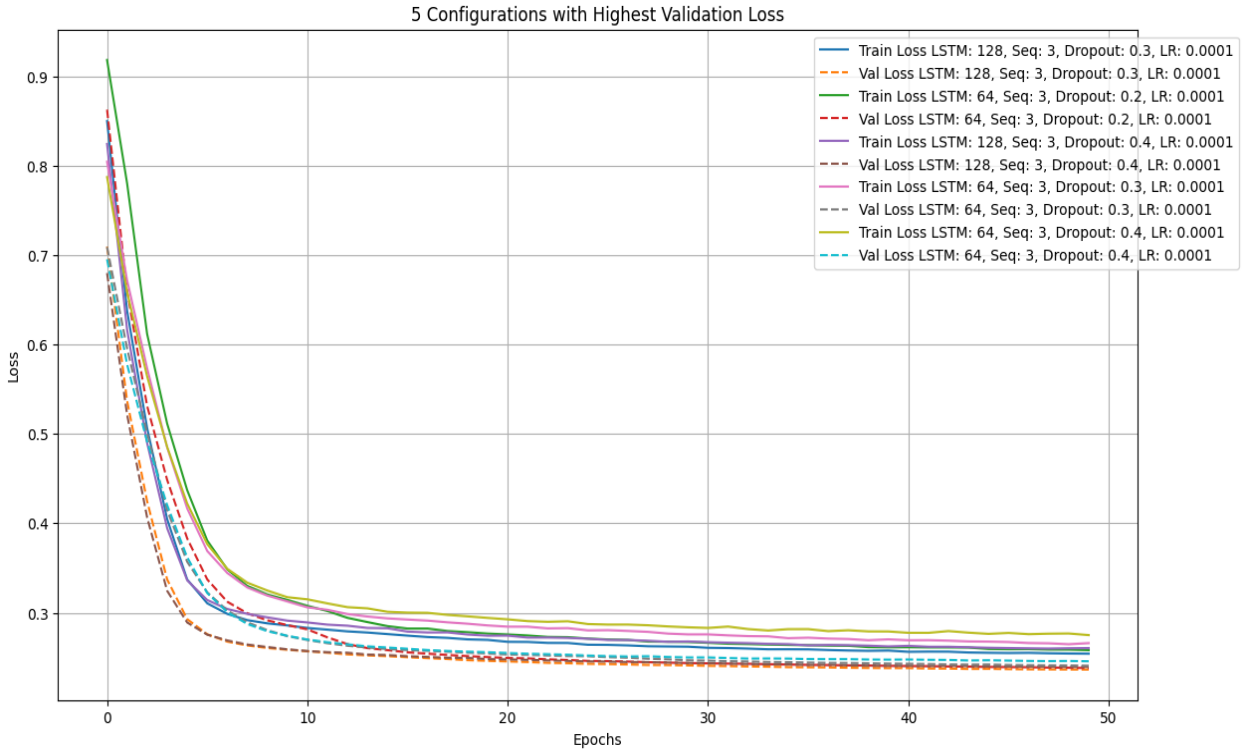


Fig.7 Worst 5 models - Training + Validation losses with parameter plot

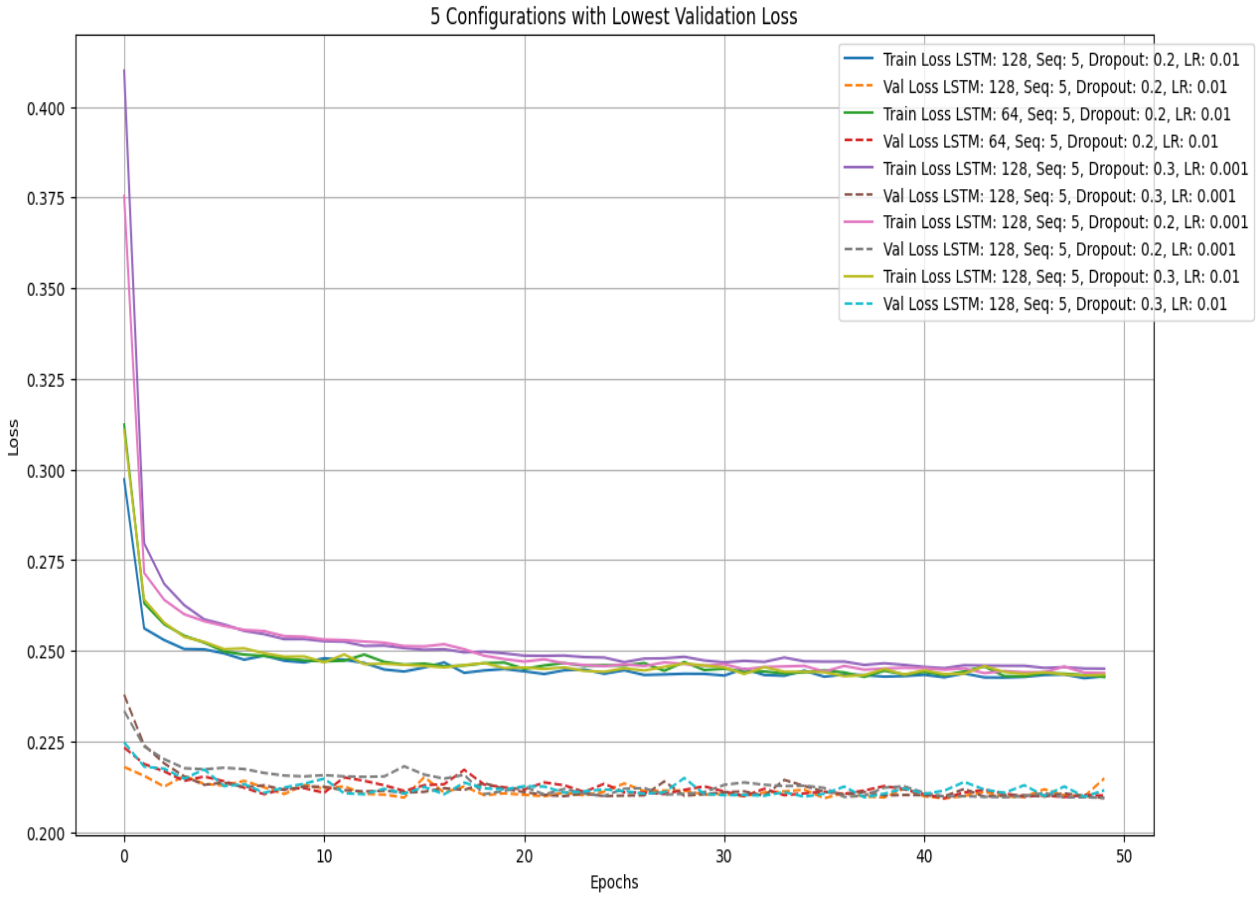


Fig.8 Best 5 models - Training + Validation losses with parameter plot

i. Output trajectory performance against Ground Truth (APE metric):

XYZ plot of Machine Hall 04: Trained with MH05 (difficult)

XYZ plot of Machine Hall 04: Trained with MH01 (easiest)

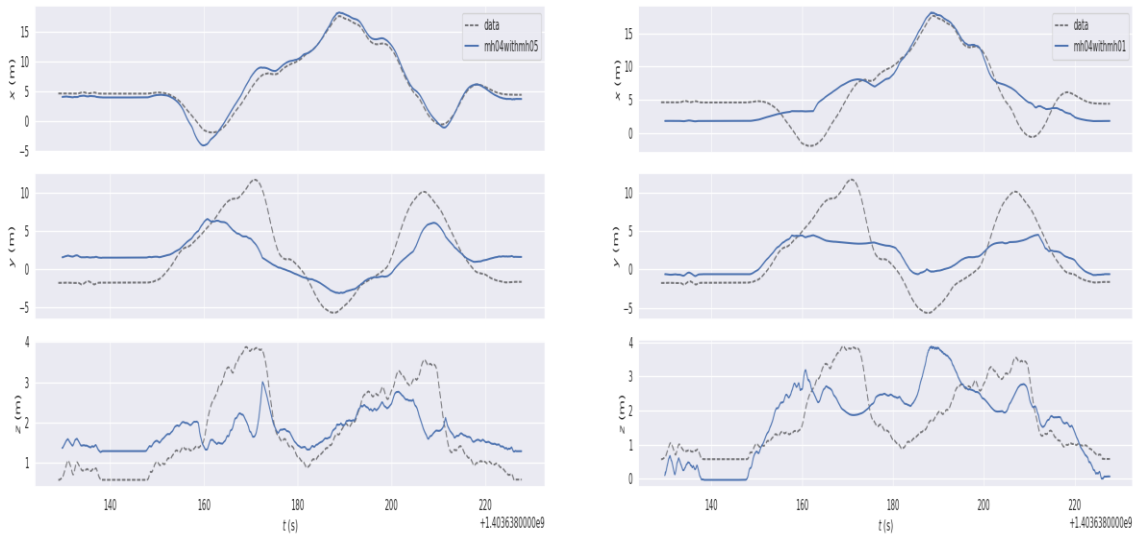


Fig.9 Result when trained with mh01, mh05(using with Vicon room series)

Adding to what we discussed above in the Datasets section, training with Mh05 and MH01 poses some contrasting results when tested with MH04(similarly agile to MH05). The main bottleneck in our training and testing comes in capturing the motion along the Z axis where the drone shows the most random movement. This involves changing the magnitude of velocity as well as the direction. Training with MH04 rightly captures the movement along X which points in the forward direction of the drone, and Y movement is captured better by the MH01 as it is less agile throughout, but for the Z motion, even though the motion is offset by constant amount for MH05, the overall motion is still very random for this lightweight LSTM module and training pipeline to learn from.

Below is the demonstration of the resultant trajectory capturing total velocity with respect to ground truth. Here, similar to the above result, training with MH05(difficult) captures the trend of velocity and acceleration better than training with MH01(easy).

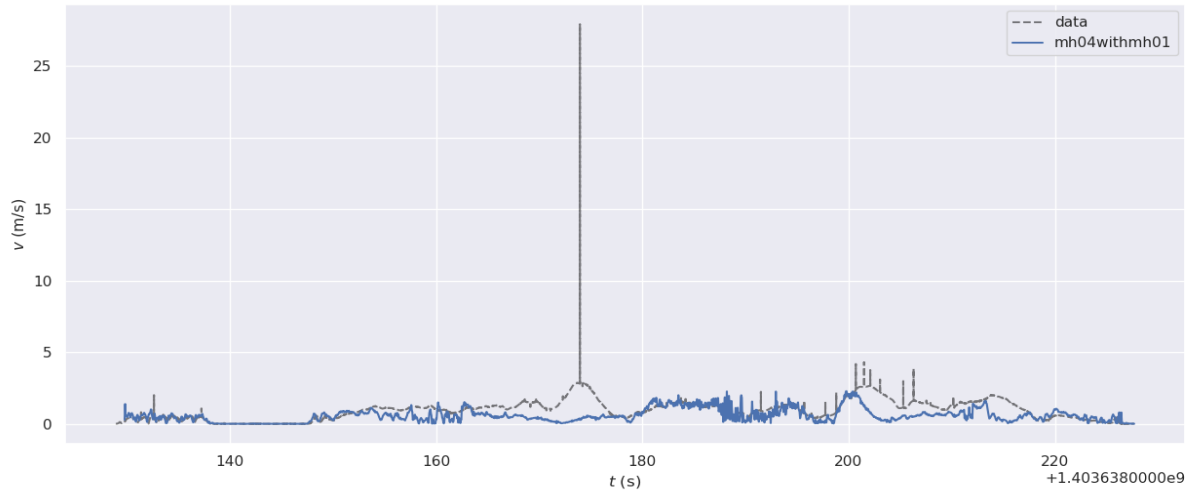


Fig.10 Velocity plot of Machine Hall 04: Training with MH01 (easiest)

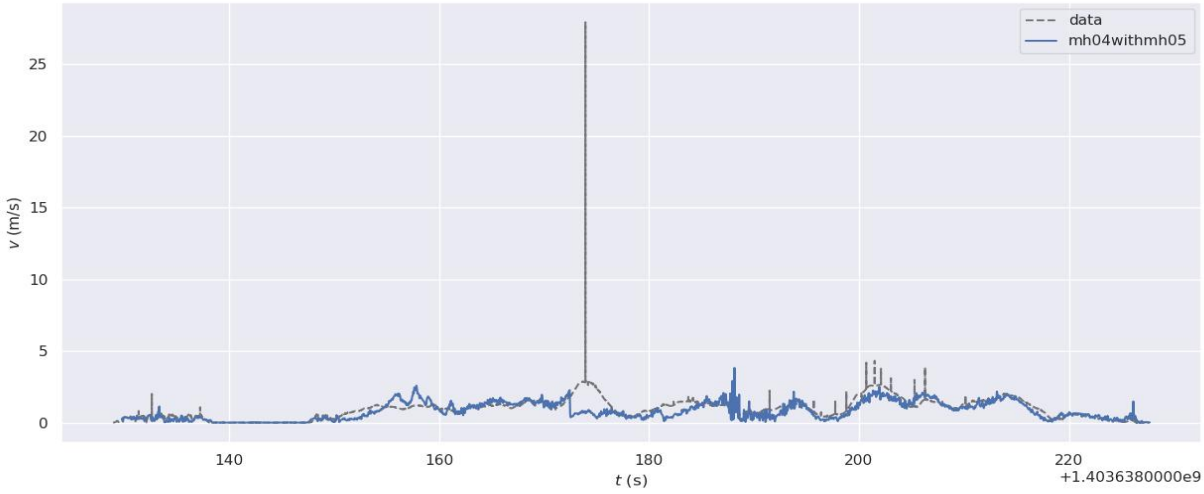


Fig.11 Velocity plot of Machine Hall 04: Training with MH05 (difficult)

ii.) Switch of Trajectory Fusion

In another attempt to get the best out of both algorithms(which can later be extended to more algorithms), we extracted a representative APE value of consecutive timestamps. For instance, it could be the mean APE of 5 consecutive timestamps across both the trajectory outputs from OpenVSLAM and ORB-SLAM3. This straightforward algorithm can also be combined with other algorithms in this work when making a switch based on the image features and suitable SLAM algorithm.

We also explored these factors and how they affect the Mean and RMSE of APE(Absolute Pose Error) metric.

1. Orientation error calculation
    a. Subtracting directly - The difference between quaternions
    b. Using SciPy's rotation matrix - relative rotation between two quaternions to get the difference.
2. Mean vs median of that set of APEs to decide between two trajectories.

    Here, we observed how the APE representation of those few timestamps affects the mean and RMSE of the resultant trajectory against ground truth.

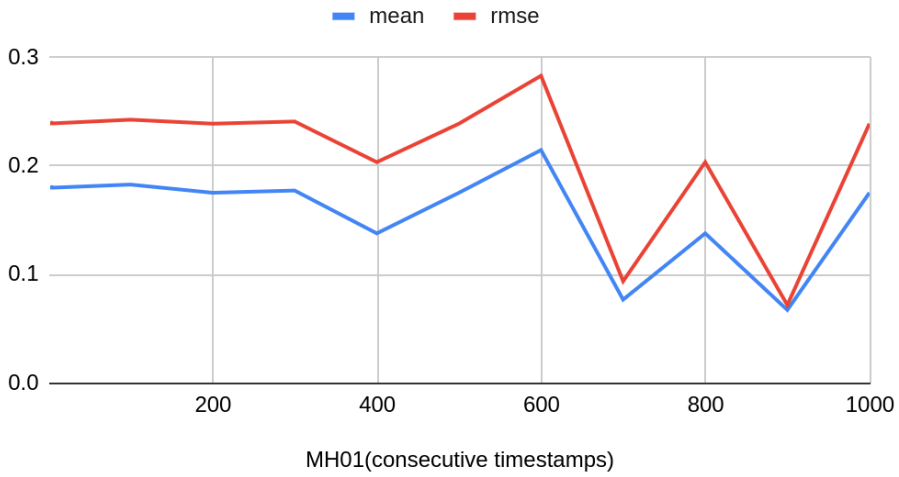


Fig.12 When we take Mean APE as basis for switching trajectories

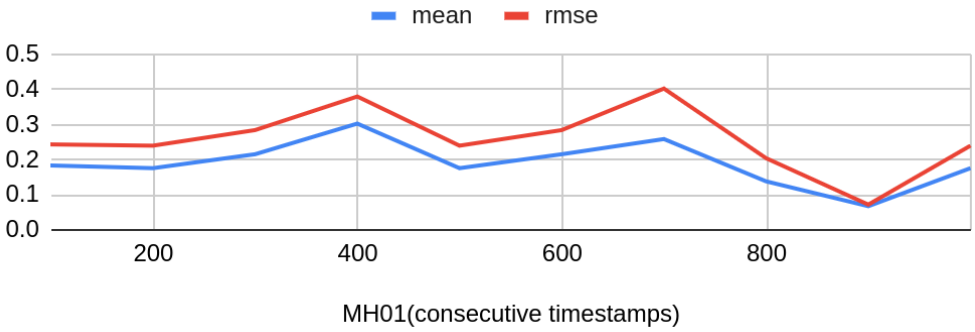


Fig.13 When we take median APE as basis for switching trajectories:

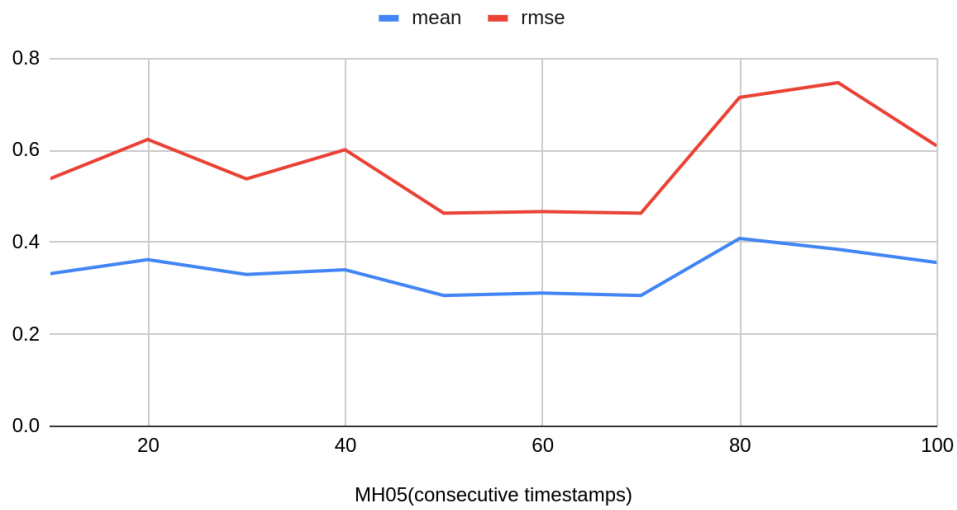


Fig.14 The number of consecutive timestamps we considered to calculate APE and switch between algorithms vs the mean/RMSE of resultant trajectory

Table.5 Tabular representation of above graph for MH05

EuRoC MH05	Mean [meter]	RMSE [meter]
10	0.330924	0.536472
20	0.361708	0.622753
30	0.329411	0.536928
40	0.33965	0.6005
50	0.283708	0.462489
60	0.289012	0.465885
70	0.283708	0.462489
80	0.407744	0.714185
90	0.38404	0.746228
100	0.355426	0.608614

iii.) Weighted Average Trajectory Fusion

The weighted average fusion method is the one we started with where we tried to make a fusion algorithm working purely based on the overall APE. This takes in both the trajectory outputs from OpenVSLAM and ORB-SLAM3, and uses the inverse ratio of APE to weigh them accordingly. In this way, the lesser APE SLAM method gets more weightage over the other.

This method proves to give insight over which method performs better as well. Take a look at this table:

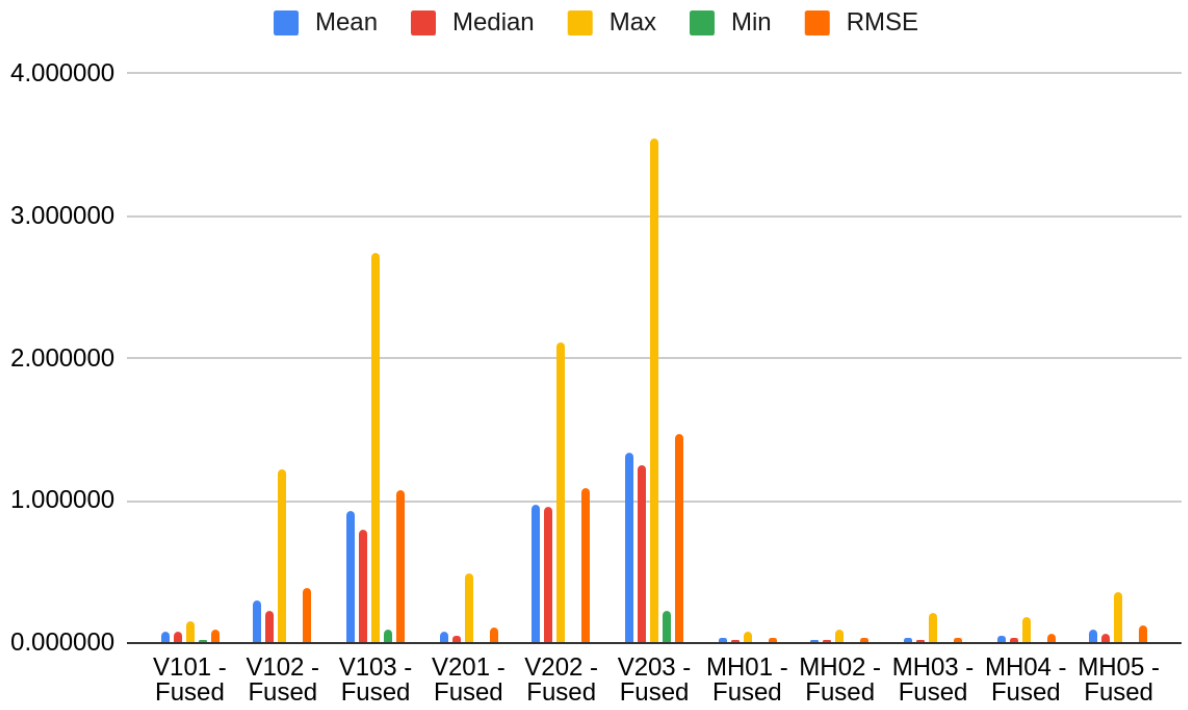


Fig.15 Mean, RMSE, Max, Min of APE across all the environments in EuRoC dataset

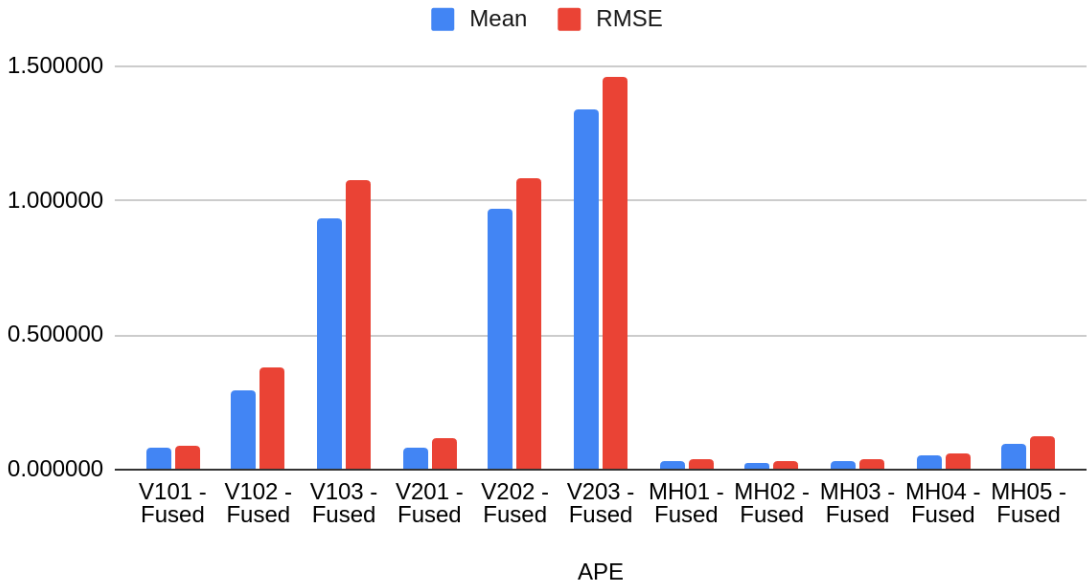


Fig.16 Mean and RMSE of APE across all the environments in EuRoC dataset

For all the Machine Hall environments-weighted fusion did better in terms of APE than at least one of the individual SLAM methods. In Vicon Rooms, apart from difficult room/movement settings(V103, V203), the weighted fusion method performed better than at least one SLAM method.

IX. Online Fusion

Apart from offline fusion, which aims to combine only the most accurate parts of each algorithm to find the optimal solution, the next step is to consider online fusion, which reflects real-world situations where ground truth is not always known in advance. As the results of individual algorithm evaluation show, OpenVSLAM is the most likely to perform best in all cases. Therefore, we select it as the main algorithm and switch to other algorithms when it tends to fail. The primary goal of this task is to analyze image features and motion-related parameters where failures occur in OpenVSLAM.

In this study, we utilized several variables to analyze and model the data. Below is a description of each variable:

- **laplacian_variance:** Represents the variance of the Laplacian of an image, which is used to measure image sharpness. A higher variance suggests a sharper image, while a lower variance indicates blurriness
- **entropy:** Measures the amount of disorder or randomness in an image. It quantifies the complexity or information content by evaluating the probability distribution of pixel intensities
- **brightness:** Refers to the perceived intensity of light emitted or reflected from an image
- **contrast:** The difference in luminance or color that makes objects distinguishable
- **disparity_mean, disparity_std:** These relate to depth perception in stereo vision systems. disparity_mean indicates the average disparity, while disparity_std represents the standard deviation, reflecting depth variation across an image
- **v_x, v_y, v_z:** Velocity components; v_x, v_y and v_y refer to velocities in the x-axis, y-axis and z-axis, respectively, while v indicates overall velocity.
- **v:** overall magnitude of velocities
- **d_vx, d_vy, dv_z, d_v:** Absolute changes in velocities; d_vx, d_vy and dv_z refer to changes along the x-axis, y-axis and z-axis, respectively, while d_v indicates changes in overall velocity
- **w:** overall magnitude of Angular velocities
- **d_wx, d_wy, d_wz, d_w:** Absolute changes in angular velocities; d_wx, d_wy and d_wz refer to changes along the x-axis, y-axis and z-axis, respectively, while d_w indicates changes in overall angular velocity

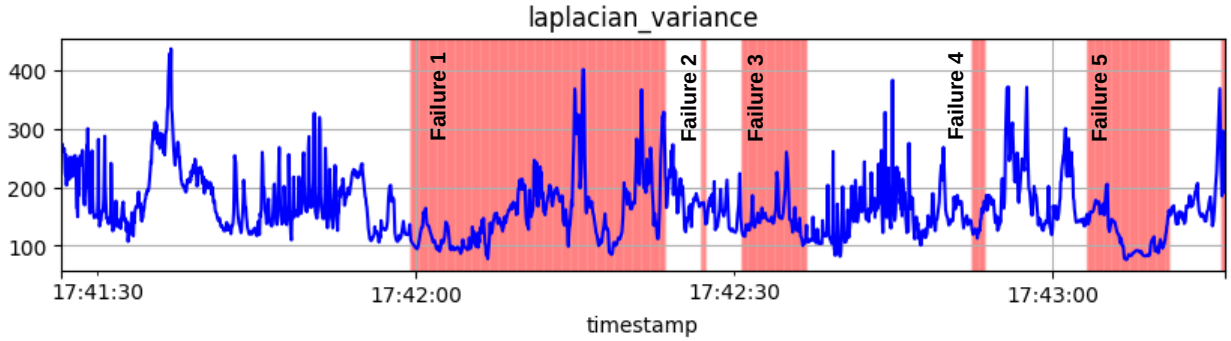
i.) Data Visualization

Before applying statistical methods, we first visualized the data to identify any potential features contributing to tracking failures. Our observations suggest that certain variables may be associated with these failures:

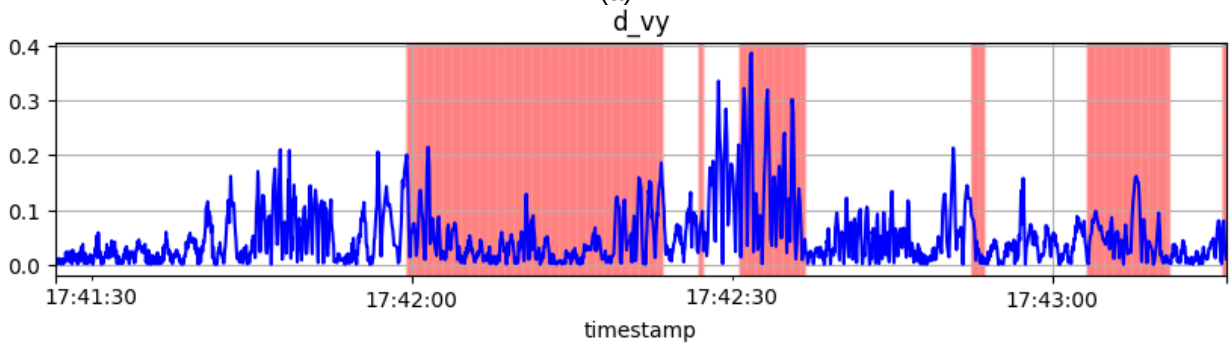
- At the beginning of the first and fourth failures, the laplacian_variance appeared notably low, indicating that image blurriness might contribute to these failures.
- During the third failure, a high value of dv_y was observed, suggesting that significant changes in vertical velocity may lead to tracking loss. Additionally, v_z was elevated during this failure, indicating that the drone was ascending rapidly.

- For the second and fourth failures, w_x values were high, implying that rapid rotation around the x-axis might be a factor in these failures.

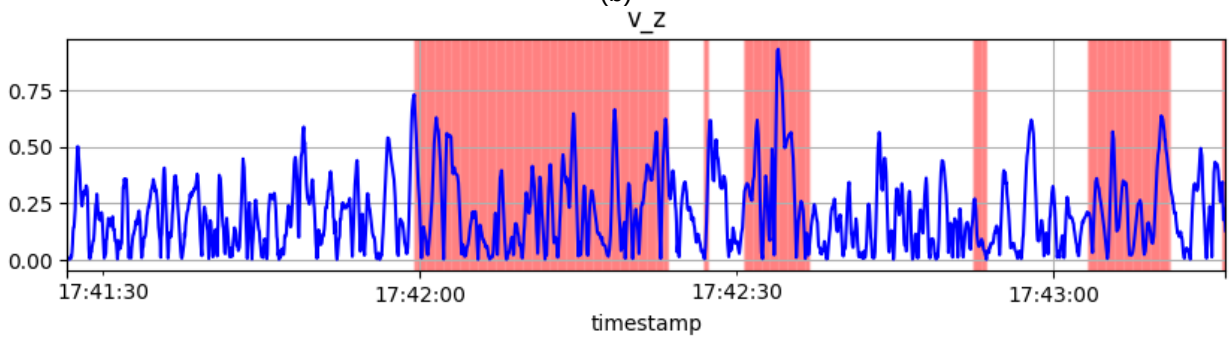
These observations form preliminary hypotheses regarding potential causes of tracking loss. To rigorously test these hypotheses, further analysis using autoregressive models is necessary.



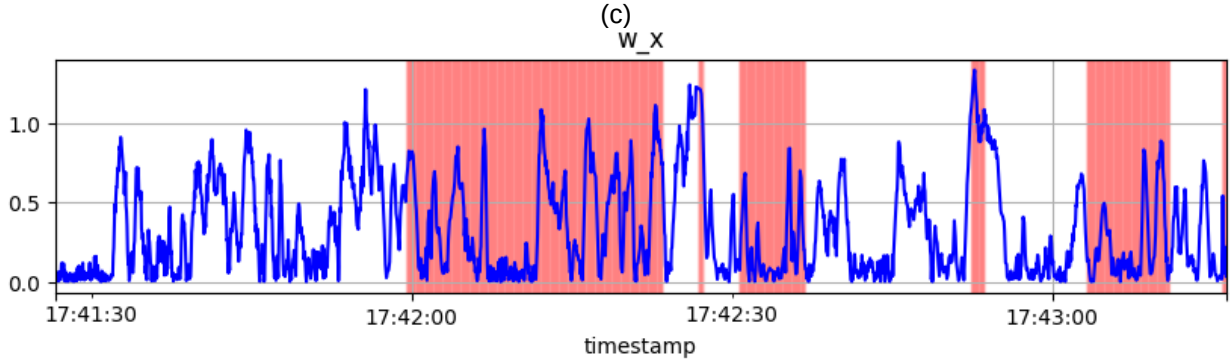
(a)



(b)



(c)



(d)

Fig.17 Raw data of V203 (a) laplacian_variance (b) d_vy (c) v_z (d) w_x

ii.) Multivariable Prediction

a. Method

As the data are time series, previous information has a potential effect on the current timestep. The method selected to evaluate this is Vector Autoregression (VAR), as it fits all input data with weights.

Vector Autoregression (VAR) is a multivariate forecasting algorithm that's used when two or more time series influence each other. In VAR, each variable is a linear function of past lags of itself and past lags of the other variables. The steps are as follows:

1.Train and Test Separation

The data is divided into two groups: training and test sets.

2.Feature Extraction

Multiple parameters that are expected to have a relationship with failure are extracted:

- Image features including variance of laplacian, entropy, brightness, contrast, disparity
- Each dataset provides different parameters, but common ones include linear velocity, angular velocity, and acceleration in three axes.

3. Data Preprocessing

- Resampling data into constant intervals
- Normalization to scale all data to a min-max range of 0 to 1

4.Stationarity Test

To predict data in time series, it must be stationary, meaning its statistical properties such as mean, variance, and covariance do not change over time. The Augmented Dickey-Fuller (ADF) test is used to check for stationarity by testing the null hypothesis that a unit root is present in the time series. If the null hypothesis is rejected, it suggests that the series is stationary or trend-stationary. If all parameters are found to be stationary, predictions are likely to be more accurate; if not, predictions can still be made but may be less reliable.

5.Feature Selection

The Granger Causality Test is implemented to determine whether a parameter has an effect on predicting other parameters. A p-value threshold is set to identify significant features, indicating whether the inclusion of a parameter improves the predictive power of the model beyond what is achieved by the other parameters alone

6.Optimal Lag Search

We fit the training data to a range of lags to search for the optimal lag (look-back timestep), using the Akaike Information Criterion (AIC) as the selection criterion.

7.Data Fitting

The training data is used to fit the VAR model with optimal lag to find the equation.

8.Testing

The model is tested on the test data, and residuals are computed to evaluate performance.

This methodology provides a comprehensive approach to analyzing time series data from image inputs, considering both visual features and motion parameters.

b. Result

Following the completion of steps 1 through 4, stationarity tests indicated that the p-values for Laplacian variance, brightness, and contrast exceeded 0.05, signifying non-stationarity in the data. This non-stationarity suggests potential challenges in model training due to the violation of assumptions required by many time series models. Subsequently, a Granger causality test was conducted to identify variables for model inclusion. The selected variables were laplacian_variance, entropy, brightness, contrast, disparity_mean, disparity_std, v_x, d_vx, v_y, d_vy, v, d_v, d_wx, w_y, d_wy, d_wz, w, and d_w. The result of p-values test is shown in the Appendix B.

The optimal lag length was determined to be 2, as shown in Figure X. Consequently, we trained the model with a lag of 2 and tested it. The testing revealed that the model's predictions closely aligned with the ground truth, resulting in a mean squared error (MSE) of 6.86E-2 at its worst.

To identify potential failure cases, we analyzed the sum of errors over 20-timestep windows (see Figure X). Although some areas exhibited high residual sums indicating potential failures, other areas with similarly high residual sums did not result in failures, suggesting that the relationship is not conclusive. Consequently, no definitive cause of failure could be identified from this data.

In conclusion, while our analysis identified areas with potential prediction discrepancies, further investigation is necessary to understand the underlying causes of these discrepancies. Future research should focus on addressing data non-stationarity and exploring alternative model specifications to enhance predictive accuracy.

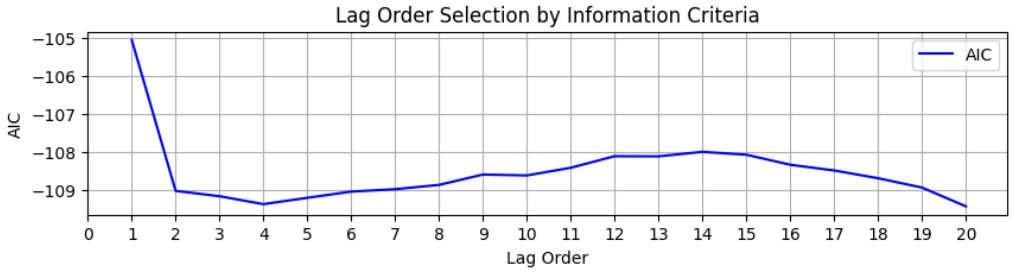


Fig.18 Lag order selection using AIC

Table.6 Mean Squared Error of Multivariable Prediction on V203

Feature	MSE
laplacian_variance	5.34E-03
entropy	3.70E-03
brightness	7.08E-03
contrast	8.17E-03
disparity_mean	1.22E-03
disparity_std	2.51E-03
v_x	1.55E-04
d_vx	5.23E-03
v_y	5.74E-04

Feature	MSE
d_vy	9.94E-03
v	1.43E-04
d_v	1.03E-02
d_wx	6.86E-02
w_y	7.40E-03
d_wy	1.40E-02
d_wz	4.44E-02
w	5.10E-03
d_w	2.31E-02

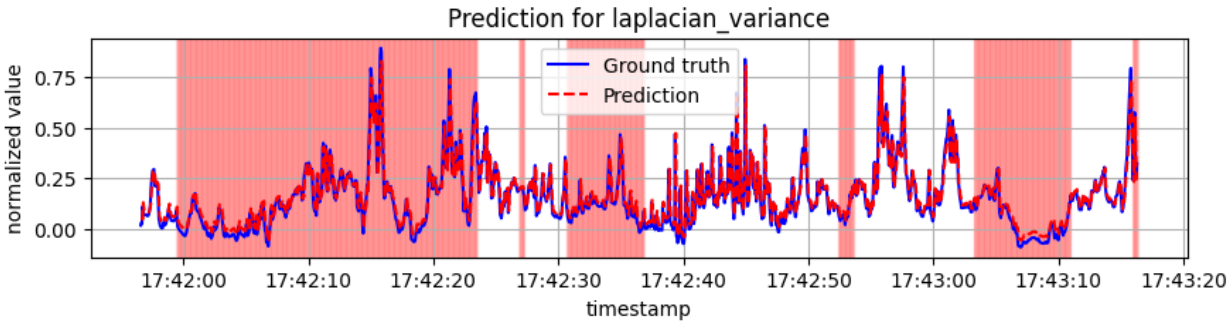
iii.) Univariable Prediction

a. Method

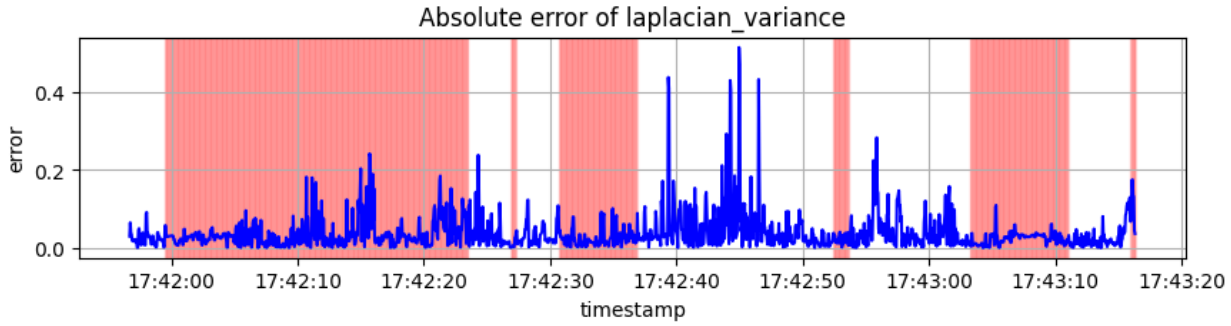
Since the multivariable predictions showed no failure indicators, we conducted univariate analysis to identify potential tracking loss causes. The method remained consistent with the multivariable approach, with two key modifications: (1) step 5 involving cross-variable comparisons was omitted and (2) replacement of Vector Autoregression (VAR) with univariate Autoregression (AutoReg) modeling for single-variable time series predictions. The model utilizes ground truth data from the previous timesteps as input, with the predicted current timestep (t) serving only as a reference. This design choice prevents error propagation by avoiding the use of predicted values as inputs for subsequent predictions.

b. Result

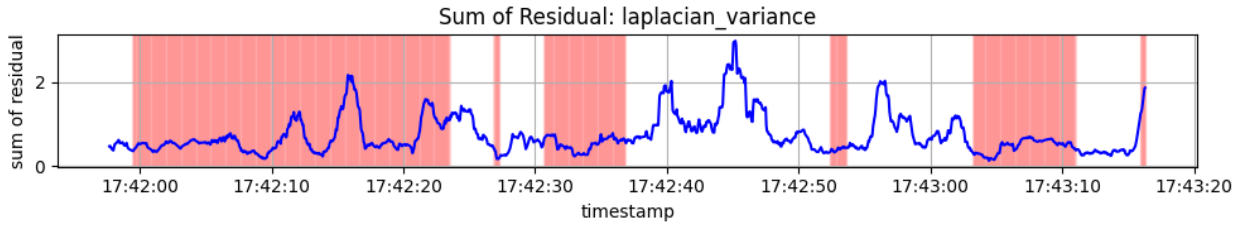
We used only image features to examine the difference between predictions and ground truth, maintaining a lag order of 2, as in the previous case. The prediction model demonstrated robustness when tested with new data, contrary to our initial hypothesis that discrepancies between predictions and ground truth would reveal failure signals. The worst mean squared error (MSE) was 5.01E-02 on d_wx, indicating no significant difference. Analysis of both instantaneous absolute errors and sum of residuals over 20-timestep windows (equivalent to one second of operation) revealed no significant failure indicators, as shown in Fig.19. Therefore, these image feature predictions cannot serve as indicators of failure. Additional tables are provided in Appendix B.



(a)



(b)



(c)

Fig. 19 Prediction for Laplacian Variance
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals

Table.7 Mean Squared Error of Univariable Prediction on V203

Feature	MSE
laplacian_variance	3.42E-03
entropy	2.84E-03
brightness	3.91E-03
contrast	3.49E-03
disparity_mean	7.44E-04
disparity_std	1.81E-03
v_x	1.40E-04
d_vx	4.85E-03
v_y	4.71E-04

Feature	MSE
d_vy	9.02E-03
v	1.21E-04
d_v	8.86E-03
d_wx	5.01E-02
w_y	7.52E-03
d_wy	1.46E-02
d_wz	3.42E-02
w	4.71E-03
d_w	1.83E-02

iv.) Conclusion

Neither multiple nor singular feature regression shows any signs of predictable failure in image tracking, contrary to our initial assumptions. We hypothesized that tracking failures might occur due to low image contrast hindering ORB feature extraction, or insufficient texture distinctiveness leading to incorrect feature point matching. However, these assumptions could not be validated through autoregression analysis.

X. Technical Challenges and Limitations

i.) Visualization issue

- **Finding the appropriate 3rd party visualization:**

  Here, as we can see, the OpenVSLAM's output should be visualized using a 3rd party viewer such as Pangolin Viewer or Iridescence viewer that is easier for debugging and configuration of settings.

- **Impact:**

  The initial viewer which was set up: Iridescence viewer. This was quite troublesome in the sense that the viewer wasn't properly supported by the Graphics Driver NVIDIA_535 in Ubuntu 22.04 - resulting in a very small window size making it difficult for visualization and configuring loop closure detection.

- **Solution:** The next step was to try Pangolin Viewer which works with the said driver.

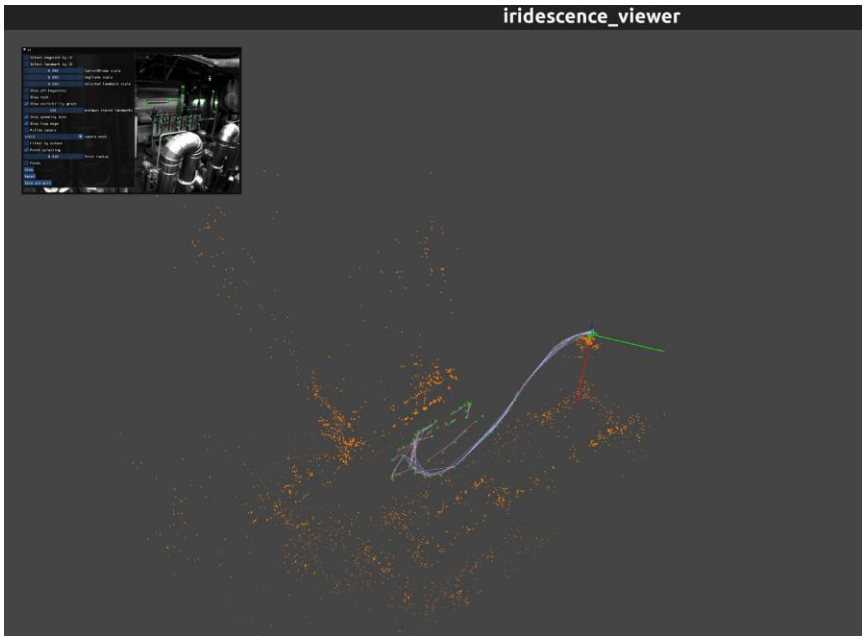


Fig. 20 Iridescence_viewer support issue with NVIDIA driver

- **Empty Mapping in Iridescence viewer:**

  Iridescence Viewer when combined with ROS Wrapper for the OpenVSLAM, puts out an empty map during the mapping session.

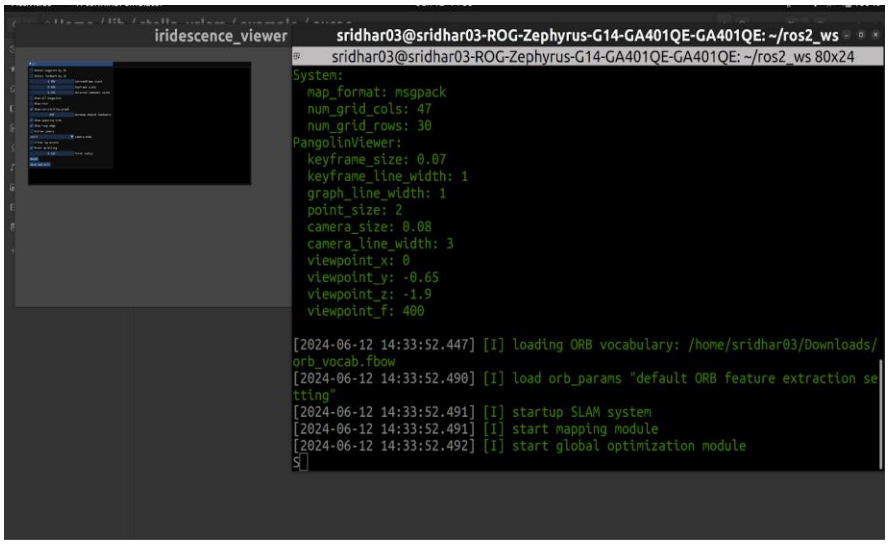


Fig. 21 Iridescence_viewer mapping issue with ROS Wrapper

- So, instead of ROS Wrapper - which also has the dependency issue with Backward_CPP and Backward_ROS, the OpenVSLAM system was set up successfully on Ubuntu 22.04 without ROS Wrapper, which solved the issue of mapping as well as that of Backward_CPP.

ii.) System Implementation Constraints

The implementation faced several hardware and software constraints across different development environments. Docker containerization encountered compatibility issues specifically with MacOS systems, impacting development flexibility. Storage requirements also posed a significant challenge, as the Kagaru dataset's image sequences required a large amount of storage capacity in ROS bag format. This constraint limited multi-device development capabilities, though execution was successfully achieved on a single high storage laptop.

iii.) Limitation of Monocular Camera

The utilization of a monocular camera as the sole sensor presents significant challenges in real-world SLAM applications. A fundamental limitation is the inherent scale ambiguity in monocular vision, where the generated map can only be reconstructed up to scale factor. Without prior metric information, this scale ambiguity can lead to significant inaccuracies in map reconstruction and pose estimation. Additionally, monocular systems demonstrate particular vulnerability to environmental challenges, including adverse weather conditions (snow, rain) that degrade image quality, and dynamic motion patterns such as rapid acceleration or abrupt directional changes. Initial attempts to evaluate these algorithms using drone racing and Tartan Air datasets proved unsuccessful, as the systems were unable to extract sufficient visual information for reliable tracking and mapping under these demanding conditions. These limitations underscore the inherent constraints of monocular vision-based SLAM systems in challenging scenarios.

iv.) Prediction Model Limitations

The vector regression approach for tracking failure prediction did not provide significant results, contrary to our initial hypothesis. This limitation can be attributed to insufficient failure cases in the test sequences (n <= 5), resulting in an inadequate training dataset.

XI. Conclusion and Future work

We have successfully implemented OpenVSLAM, ORB-SLAM3, and LSD-SLAM on two diverse datasets: an agile indoor environment and outdoor flight sequences from the Kagaru Airborne dataset. Quantitative evaluation metrics, including Absolute Pose Error (APE) and Relative Pose Error (RPE), were used to compare the performance of these SLAM systems. The results indicate that OpenVSLAM generally outperforms the other algorithms in terms of accuracy and ability to recover from lost tracking. We hypothesize that OpenVSLAM's superior performance is due to its more effective feature extraction and optimized choice of Bag-of-Words (BoW) model. Future work could explore the effect of different BoW implementations on performance in map merging tasks.

Our initial hypothesis proposed using OpenVSLAM as the primary algorithm, with LSD-SLAM serving as a complementary system for challenging scenarios such as low-texture areas and repetitive patterns where OpenVSLAM might falter. However, our experiments revealed that LSD-SLAM was unable to compensate for OpenVSLAM's shortcomings, as it encountered similar difficulties in these problematic environments. Subsequent research could focus on identifying alternative direct algorithms capable of performing well in such situations.

The RNN-LSTM gives the prediction of trajectory and moves towards the ground truth for the datasets we mentioned above. While the LSTM is able to capture the motion along X and Y pretty well, the quaternion angle learning and the agile Z motion is still a bottleneck. So, another approach to effectively learning sequential trajectory data is through Encoder-Decoder architecture. While this might be heavier in computation for real time systems compared to the lightweight LSTM architecture, this model can be custom trained to learn specifically for XYZ positions and Quaternion angles especially given the quick changes in the orientation and Z-axis position.

Furthermore, as our failure analysis was inconclusive, further investigation could explore alternative approaches such as logistic regression or other classification algorithms. This avenue of research would be contingent upon collecting a more comprehensive dataset of failure cases.

XII. Value of Capstone Project to The Organization

GE Aerospace’s mission is to commit to advancing the future of aviation through innovative technologies that enhance safety, efficiency, and sustainability. A key aspect of this mission is the development and integration of advanced avionics systems for semi- and fully autonomous platforms, which require robust and reliable navigation capabilities even in challenging environments where traditional systems like GPS may fail. Our work contributes to this mission by exploring and benchmarking various SLAM approaches and improving accuracy by fusing them. These techniques are crucial for enabling autonomous platforms to maintain situational awareness and operate safely by continuously mapping their environment and localizing themselves within it, even when external references are unavailable or unreliable. By evaluating different SLAM algorithms and their combinations, our research aims to identify solutions that provide high-quality mapping with minimal computational demands, thereby enhancing the operational capabilities of GE Aerospace's autonomous systems. This work not only supports GE Aerospace's goal of pioneering safer and more efficient flight technologies but also contributes to broader industry efforts to improve autonomous navigation systems.

The SLAM problem is a common challenge in nearly every scenario involving autonomy. Currently, there are numerous SLAM algorithms, including those we have explored in our work. From an organizational perspective, selecting the most suitable algorithm can be time-consuming, particularly when dealing with semi or fully autonomous aerial systems, as proposed by GE Aerospace. Our work addresses this issue by benchmarking various SLAM methods against standard datasets that encompass a wide range of lighting conditions, movements, and environments. We have discussed with GE representative Paul Ardis the possibility of providing GE access to this work to serve as an initial guide and allow them to further develop it at their discretion.

Beyond the algorithms themselves, their application across multiple scenarios to enhance task effectiveness is a significant advantage. A related project by Usenko et al. (2020) involves developing vision-based localization and navigation technologies for Micro Aerial Vehicles (MAVs), which is crucial for enhancing autonomy in GPS-restricted or GPS-denied environments. This development is essential for improving the efficiency and accuracy of inspections in large structures, such as bridges, by automating these processes and addressing significant challenges in cost-effectiveness and systematic inspection quality, which are critical for infrastructure maintenance. Additionally, service robots that navigate in low-visibility conditions benefit from sensor fusion techniques that enhance the reliability of autonomous systems by maintaining

operational capabilities despite environmental challenges (Alejo et al., 2023). These advancements are crucial as they enable industries to increasingly rely on autonomous systems for improved efficiency and safety, reflecting broader trends and challenges in the field of robotics today.

XIII. Value of Capstone Project to Individual

i.) Yifei Li

- How did your experience connect with your academic studies?

  The capstone project directly integrated knowledge from various areas, including robotics, computer vision, and machine learning. This project allowed us to bridge theoretical concepts with practical implementation, fostering a deeper understanding of their real-world applications. Also, during the project, it also helped me improve my communication and presentation skills.

- Describe your most significant opportunities to network, collaborate, and learn new skills during your experience.

  This project offered significant opportunities to network with experts in the field through conferences, discussions with advisors, and collaborative coding sessions with peers. Additionally, I gained hands-on experience with tools such as Docker, github and Evo package for trajectory evaluation These skills not only enriched my technical expertise but also prepared me for collaborative research in robotics.

ii.) Sridharan Subramanian

- How did your experience connect with your academic studies?

  This experience involved SLAM in detail where I'd to work on Visual SLAM and how different algorithms differ from each other, not just in the method they use, but also in the results that we clearly noticed. The SLAM content learnt in Mobile Robotics was useful often when reading through literature. This also involved dataset preparation and manipulation before using it for training and how I translate the prediction back to get the visualizable output. This was similar to and extended what I've learnt in Applied Machine Learning coursework as well.

- Describe your most significant opportunities to network, collaborate, and learn new skills during your experience.

  This capstone opportunity opened me to industry experience which in itself is very valuable to gain and learn from. I learnt how projects go on from problem formulation to completion in corporate and research environments, and how I can work in a team to provide meaningful results. This required revising SLAM algorithms and basics which was very useful, and learning docker systems was one of the major learnings I took from this, especially given it will be directly needed at any organization throughout my career. Concepts of version control software GitHub and adapting the code to work between Collab-like notebook environments and local systems have also been improved. Finally, as I've mentioned later on, interaction with Paul gave me a good glimpse into how to

communicate technical concepts with officials, which I consider to be very crucial moving forward at any organization.

iii.) Narathip Rodwarna

- How did your experience connect with your academic studies?

  This research experience integrated various subjects from my academic studies, as well as areas I had not previously explored. For instance, while SLAM (Simultaneous Localization and Mapping) was only briefly covered in the context of LiDAR in my courses, this project allowed me to delve into its details with different sensor modalities. Additionally, I was able to apply my background knowledge from computer vision and deep learning courses to explore new areas. Having worked with a small robot in another project, I could see how these systems benefit mobile robotics. Furthermore, I had the opportunity to apply time-series prediction techniques, which I had only briefly encountered in my deep learning class, providing a valuable chance to practice this skill.

- Describe your most significant opportunities to network, collaborate, and learn new skills during your experience.

  With this project, collaborating with Paul, the advisor from GE, provided me with the invaluable opportunity to work alongside an expert in the field. His guidance and advice were instrumental in helping me navigate the project effectively and delve deeper into the topic with clear direction. As a team, we were initially a group of individuals unfamiliar with each other, but came together to work on a shared area of interest. This collaboration allowed us to exchange ideas, learn how to be both leaders and team players, and develop lasting friendships.

  Additionally, I received excellent support from Professor Bhowmik and Professor Mitra, who facilitated access to computing resources in the ECE building. In terms of skill development, I learned several tools essential for real-world applications. While my coursework primarily involved Python, this project required me to expand my knowledge of C++, as many robotics algorithms are written in it. Further, this experience also enhanced my skills related to Linux and Docker systems.

XIV. Reflection on Experience

i.) Yifei Li

- How did the capstone project affect you? In what ways did you grow personally and intellectually?

  The capstone project was a transformative experience, helping me grow both personally and intellectually. On a personal level, I developed adaptability when faced with technical challenges, such as debugging algorithm failures or aligning datasets. And I also cover my drawback of not willing to seek help from my advisor and peers. Intellectually, I gained a profound understanding of visual SLAM systems, trajectory evaluation metrics. They are important key parts of robotics.

- What were your goals? How were they met?

  My primary goals for this project were to (1) benchmark SLAM algorithms with datasets, (2) develop a trajectory fusion pipeline. These goals were met through meticulous testing of SLAM algorithms, successful implementation of LSTM for trajectory fusion, and comprehensive analysis documented in our final report.

- How did your experiences impact your perception of Autonomy and Robotics industry or research?

  This experience broadened my perspective on the challenges and opportunities in the autonomy and robotics industry. There are many challenges in some specific areas such as VSLAM, many improvements can be done and there are many opportunities.

- In what ways has your experience prepared you for a career in Autonomy and Robotics?

  The project equipped me with practical skills such as containerization, data preprocessing, and efficient communication, which are critical for careers in robotics. It also proves the ability of analytical thinking, problem-solving, and technical communication, all of which are essential for success in the field.

- What is your greatest “take away” from your experience?

  My greatest takeaway from this experience is the importance of persistence in tackling complex problems. The iterative process of debugging, experimenting, and refining our approach taught me that progress often comes from learning through failures and embracing challenges as opportunities for growth.

ii.) Sridharan Subramanian

- *How did the capstone project affect you? In what ways did you grow personally and intellectually?*

  This capstone project helped me shape my approach to work in a positive way that builds on my previous work experience as well. It has been a good means to prepare myself towards a shared goal and how my individual contribution should align with the collective result.

- *What were your goals? How were they met?*

  Apart from the clear technical goals associated with the project, my main goal was to experience and prepare myself to be involved in technical work at a company in the US, and to equip myself to communicate effectively and deliver results in a professional setting. This was made possible by regular meetings with Paul Ardis from GE Aerospace.

- *How did your experiences impact your perception of Autonomy and Robotics industry or research?*

  This project opened me up to some real-world challenges and works going on at one of the most impactful corporations in the world. This made me learn how a problem is approached step by step, and how teams come together to solve these problems in the Autonomy and Robotics domain which at many times may not be intuitive in formulation.

- *In what ways has your experience prepared you for a career in Autonomy and Robotics?*

  This project helped me improve in problem formulation and revise crucial concepts and literatures in Visual SLAM which is one of the common elements in a Robotics career. My learnings from especially implementation of RNN from scratch and learning about autoregression models will also prove useful in the Autonomy and Robotics domain.

- *What is your greatest "take away" from your experience?*

  The greatest take away from my experience is how to combine with a team and communicate effectively to a manager at a corporation. This time definitely helped me improve my way of communicating with an official on subject matter which I believe is one of the most crucial and sought after qualities at top organizations.

iii.) Narathip Rodwarna

- *How did the capstone project affect you? In what ways did you grow personally and intellectually?*

  The capstone project provided an opportunity to apply classroom knowledge in a practical setting. I learned multiple aspects from working on this project. First, I collaborated with an amazing team of supportive and hardworking individuals. We motivated each other to achieve our goals. I also developed my teamwork skills and took on leadership roles, allowing me to practice and enhance my leadership abilities throughout the project. Given the project's long duration, alongside attending classes and searching for jobs, I have learned effective time management skills, which were crucial for success in all these endeavors. Intellectually, I acquired new skills and deepened my knowledge in the field of computer vision, which is my desired area of future work. Concepts such as Lie algebra and optimization techniques are fundamental and important, with potential applications in the future.

- *What were your goals? How were they met?*

  My primary goal was to evaluate the target algorithms and analyze their pros and cons. My secondary goal was to analyze failure prediction. I achieved my primary goal by studying research papers and source codes to understand how each algorithm works and by testing all algorithms in collaboration with my teammates. We completed testing on our target datasets and gained valuable insights from the results.

  For the second goal, I discussed with Paul, and we decided to use autoregression to identify signs of potential failures. Although we completed the time-series analysis, we were unable to draw a conclusion. We suspect this was due to insufficient data, as loss of tracking occurred fewer than 10 times per case, which was inadequate for more sophisticated models to detect patterns.

- *How did your experiences impact your perception of Autonomy and Robotics industry or research?*

  As I researched multiple algorithms for the project, I gained a deeper understanding of fundamental techniques and concepts essential to the field of autonomy and robotics. I was surprised by how rapidly advancements occur, with technologies

evolving and becoming outdated within a decade. This experience highlighted the nature of research in this field, where collaboration and contributions from researchers worldwide drive innovation. It made me realize that continuous learning and collaboration are crucial, as many advanced algorithms build upon foundational work developed by other researchers. This insight has significantly shaped my perception of the autonomy and robotics industry, emphasizing the importance of staying current with technological advancements and fostering collaborative efforts.

- *In what ways has your experience prepared you for a career in Autonomy and Robotics?*

  Through this experience, I have gained exposure to corporate environments and become familiar with tools and theories beneficial for a career in autonomy and robotics. I have developed proficiency in C++, a crucial programming language in the field. Additionally, I have deepened my understanding of state-of-the-art techniques and their performance, which is essential for success in this industry. Moreover, I have honed my soft skills, such as communication, teamwork, work ethic, and problem-solving, which are as important as technical skills. An integral part of this experience was learning to give and receive feedback effectively, which enhanced my ability to collaborate and improve continuously. These experiences not only prepare me for a role in robotics but also equip me to be a valuable asset in any industry.

- *What is your greatest "take away" from your experience?*

  My greatest takeaway from this experience is the importance of continuous learning. Initially, we may not have had extensive knowledge of how to solve the target problem, but by continuously learning and staying focused on our goals, we equipped ourselves with the tools needed for success. Being open to feedback and guidance provided us with direction, while teamwork brought diverse ideas that led to innovative solutions we might not have conceived individually. Additionally, understanding the problem's application and constraints was crucial; for instance, knowing that the drone needed to be lightweight and equipped with only one camera meant we had to find solutions within these restrictions rather than suggesting enhancements to the camera's capabilities.

XV. References

Campos, C., Elvira, R., Gómez Rodríguez, J. J., Montiel, J. M., & Tardós, J. D. (2021). ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multi-map SLAM. *IEEE Transactions on Robotics, 37*(6), 1874–1890.

UZ-SLAMLab. (n.d.). *ORB-SLAM3: An open-source library for visual, visual-inertial, and multi-map SLAM.* GitHub. Retrieved December 4, 2024, from https://github.com/UZ-SLAMLab/ORB_SLAM3

Jahaniam. (n.d.). *ORB-SLAM3 Docker: A dockerized version of ORB-SLAM3.* GitHub. Retrieved December 4, 2024, from https://github.com/jahaniam/orbslam3_docker

Thien94. (n.d.). *ORB-SLAM3 ROS wrapper: Integration with ROS for ORB-SLAM3.* GitHub. Retrieved December 4, 2024, from https://github.com/thien94/orb_slam3_ros_wrapper

Sumikura, N., Shibuya, T., & Amano, H. (2019). *OpenVSLAM: A versatile visual SLAM framework*. Retrieved December 4, 2024, from https://github.com/xdspacelab/openvslam

Stella CV Documentation. (n.d.). *Stella CV: Overview.* Retrieved December 4, 2024, from https://stella-cv.readthedocs.io/en/latest/overview.html

Engel, J., Schöps, T., & Cremers, D. (2014). LSD-SLAM: Large-scale direct monocular SLAM. In *Proceedings of the European Conference on Computer Vision (ECCV)* (Vol. 8690, pp. 834–849). Springer.

Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M. W., & Siegwart, R. (2016). The EuRoC micro aerial vehicle datasets, *The International Journal of Robotics Research, 35*(10). https://doi.org/10.1177/0278364915620033

Autonomous Systems Lab (ASL). (n.d.). *KMAV visual-inertial datasets.* Retrieved December 4, 2024, from https://projects.asl.ethz.ch/datasets/doku.php?id=kmavvisualinertialdatasets

Warren, M., McKinnon, D., He, H., Glover, A., Shiel, M., & Upcroft, B. (2012). Large Scale Monocular Vision-only Mapping from a Fixed-Wing sUAS, *International Conference on Field and Service Robotics*, Matsushima, Japan.

Warren, M., Upcroft, B., & Shiel, M. (n.d.). *Kagaru Airborne Stereo Dataset*. Retrieved October 10, 2024, from https://michaelwarren.info/docs/datasets/kagaru-airborne-stereo

Grupp, M. (n.d.). *evo: Python package for the evaluation of odometry and SLAM.* GitHub. Retrieved December 4, 2024, from https://github.com/MichaelGrupp/evo

Staudemeyer, R. C., & Morris, E. R. (2019). Understanding LSTM--a tutorial into long short-term memory recurrent neural networks. arXiv preprint arXiv:1909.09586.

Olah, C. (2024, July 5). *Understanding LSTM Networks*. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Skipper, S., & Josef, P. (2010). statsmodels: Econometric and statistical modeling with python, *9th Python in Science Conference*

Usenko, V., von Stumberg, L., Stückler, J., & Cremers, D. (2020). TUM Flyers: Vision-based MAV navigation for systematic inspection of structures, *Bringing innovative robotic technologies from research labs to industrial end-users*, *136*, 109–123. https://doi.org/10.1007/978-3-030-34507-5_8

Alejo, D., Rey, R., Cobano, J.A., Caballero, F., Merino, L. (2023). Data Fusion of RADAR and LIDAR for Robot Localization Under Low-Visibility Conditions in Structured Environments, *ROBOT2022: Fifth Iberian Robotics Conference. ROBOT 2022*, *590*. https://doi.org/10.1007/978-3-031-21062-4_25

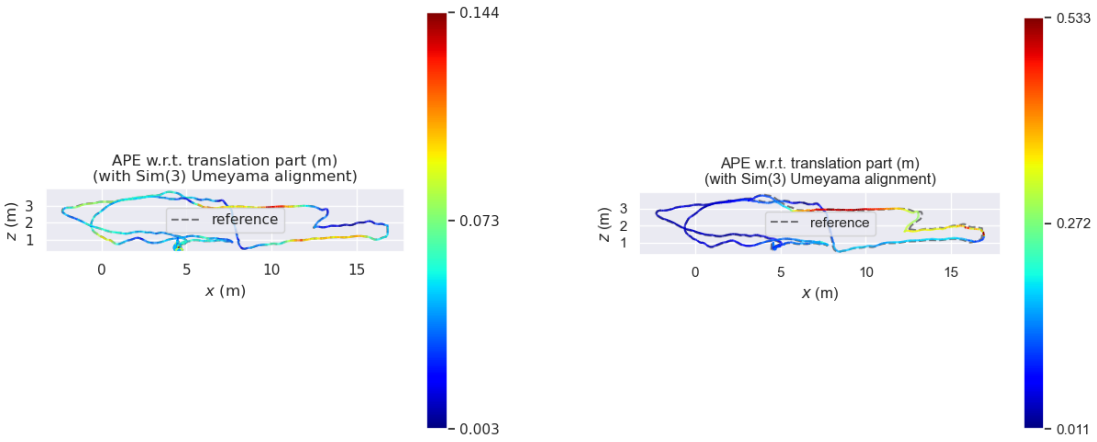
XVI. Appendices

i.) Appendix A

We have taken two quantitative aspects such as ATE and RPE.

- APE - Absolute Pose Error is the most common benchmarking aspect.
- RPE - Relative Pose Error

Here are some results on ATE from OpenVSLAM and ORB_SLAM3:



(a) (b)

Fig.22 Heatmap of APE across the map of EuRoC MACHINE_HALL_05_DIFFICULT (a) OpenVSLAM, (b) ORB-SLAM3

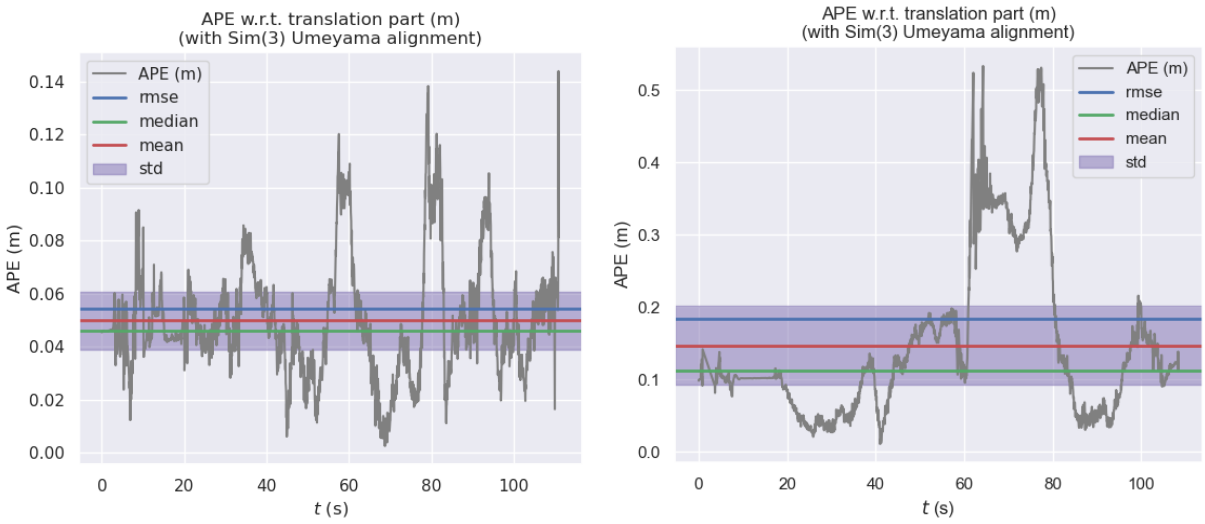


Fig.23 Raw graph of APE variation for EuRoC MACHINE_HALL_05_DIFFICULT
(a) OpenVSLAM, (b) ORB-SLAM3

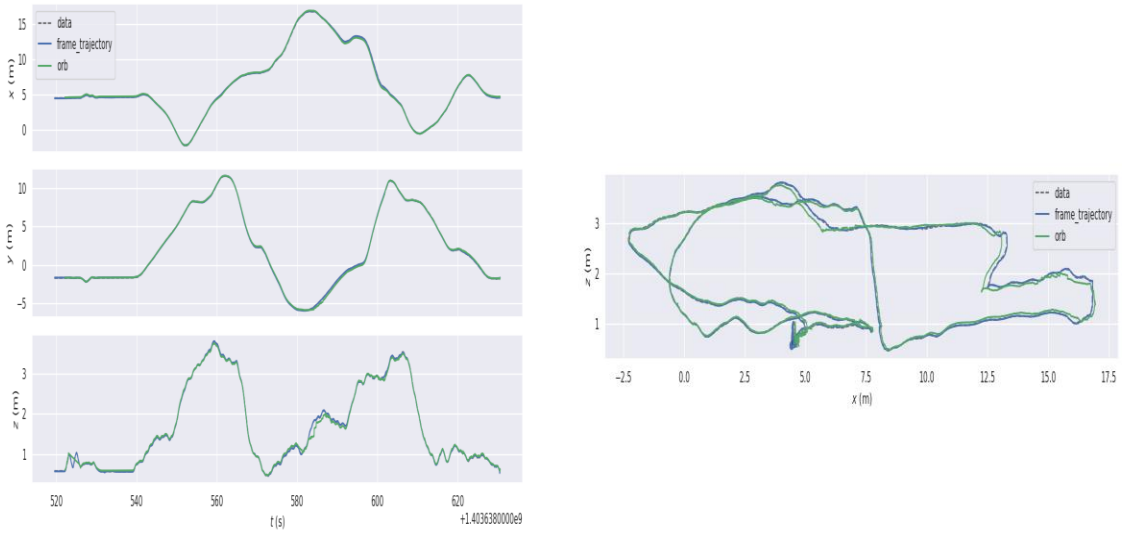
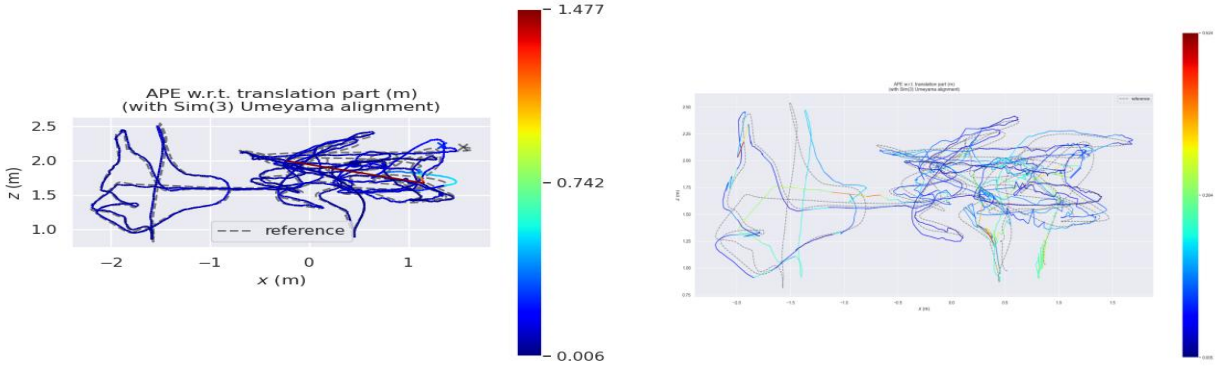
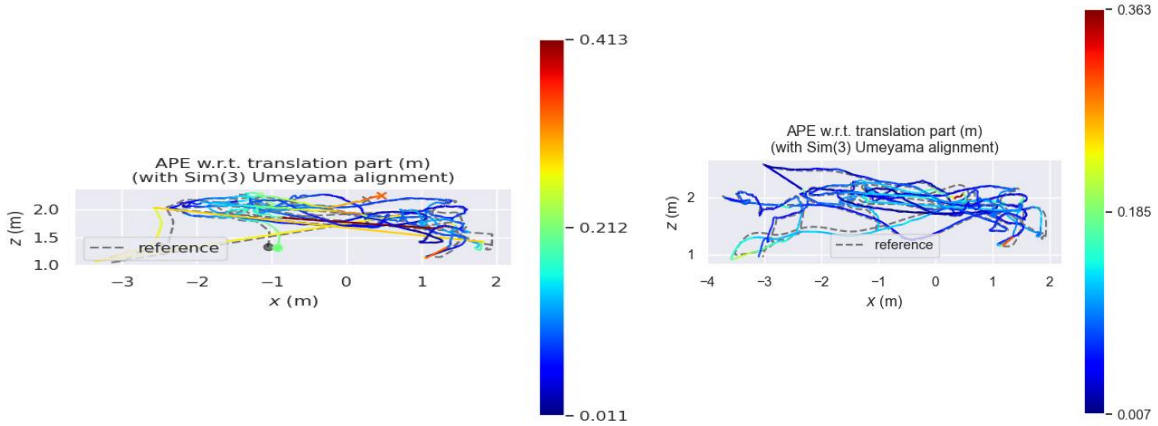


Fig.24 Trajectory Comparison between OpenVSLAM and ORB-SLAM3
of EuRoC MACHINE_HALL_05_DIFFICULT



(a) (b)

Fig.25 Heatmap of APE across the map of EuRoC VICON_ROOM_103_DIFFICULT
(a) OpenVSLAM, (b) ORB-SLAM3



(a) (b)

Fig.26 Heatmap of APE across the map of EuRoC VICON_ROOM_203_DIFFICULT
(a) OpenVSLAM, (b) ORB-SLAM3

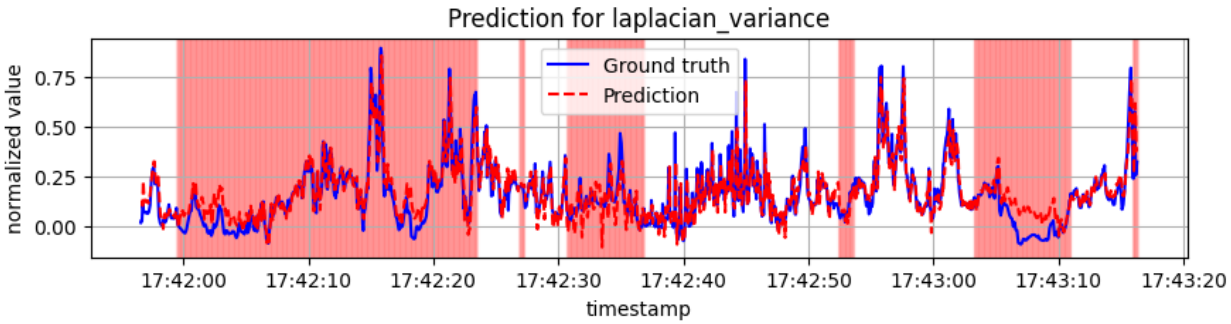
ii.) Appendix B

B.I Multivariable Prediction

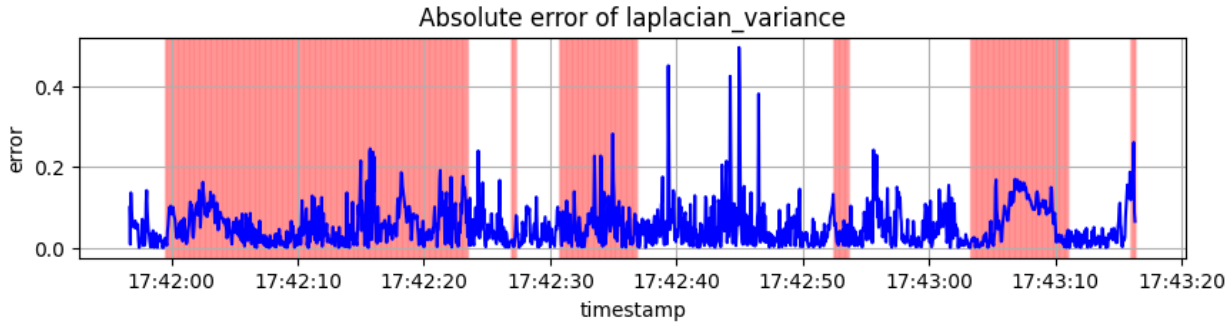
Table.8 Result of stationary test for V203

Feature	p-value
laplacian_variance	8.97E-02
entropy	3.84E-03
brightness	1.86E-01
contrast	2.34E-01
disparity_mean	3.77E-02
disparity_std	1.07E-02
v_x	1.76E-03
d_vx	1.15E-06
v_y	3.40E-04

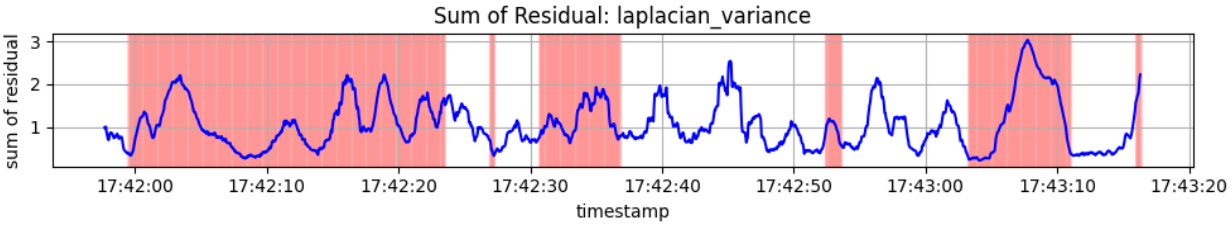
Feature	p-value
d_vy	3.23E-03
v	1.73E-03
d_v	1.10E-05
d_wx	2.90E-19
w_y	3.35E-01
d_wy	7.69E-02
d_wz	8.35E-12
w	5.26E-02
d_w	5.82E-02



(a)

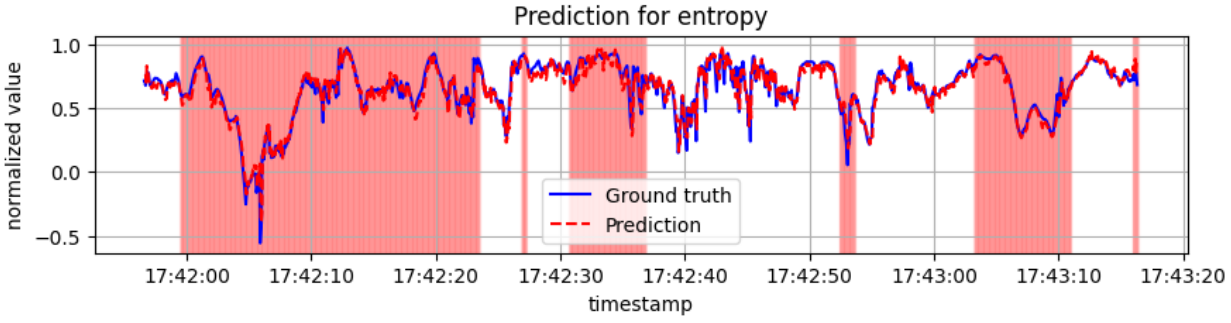


(b)

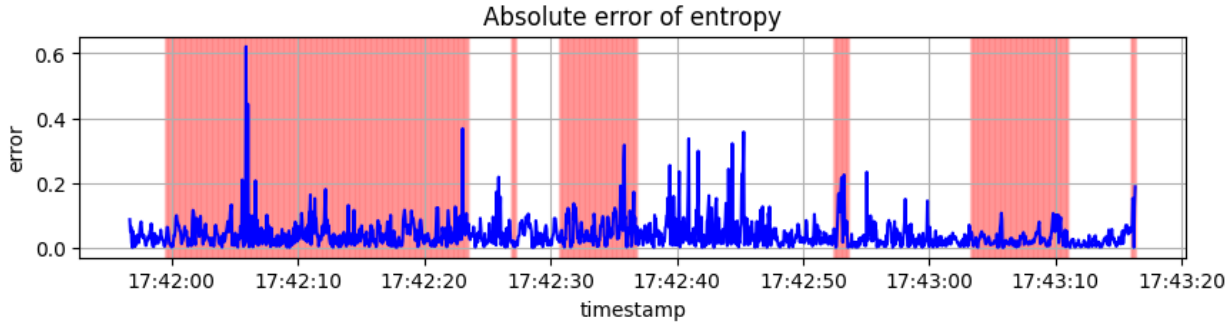


(c)

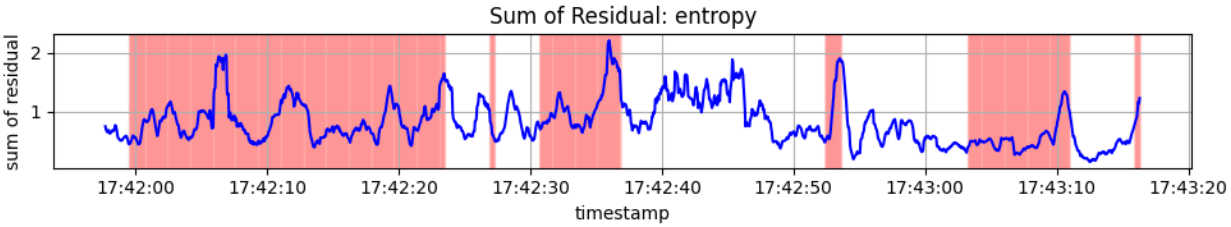
Fig.27 Multivariable Prediction for laplacian variance
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

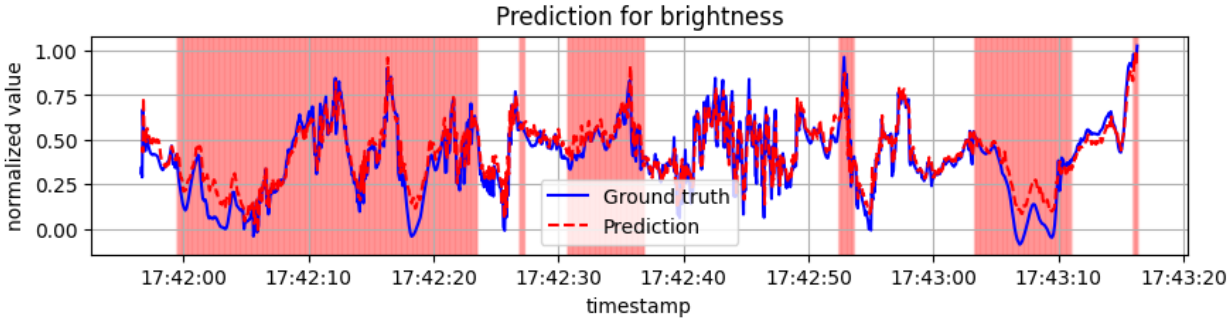


(b)

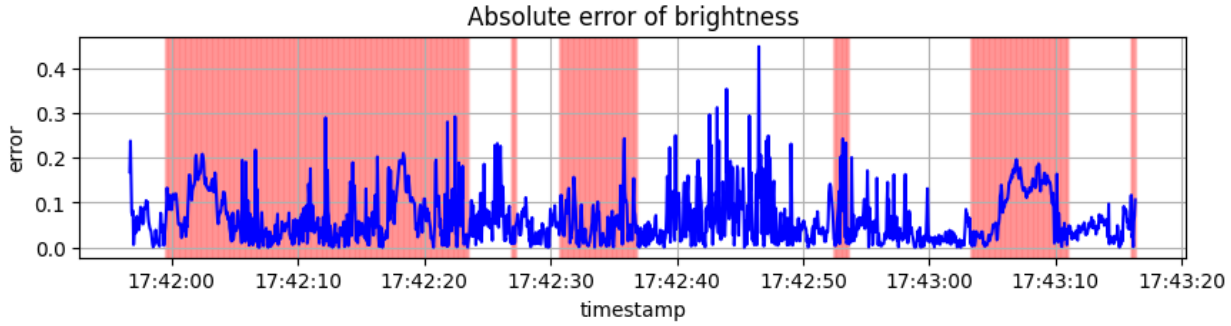


(c)

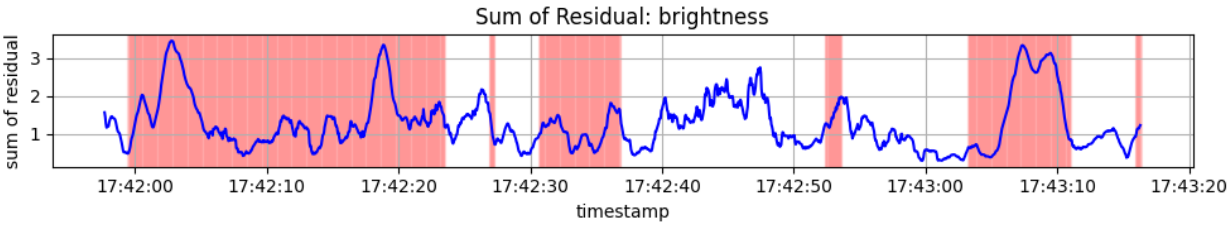
Fig.28 Multivariable Prediction for entropy
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



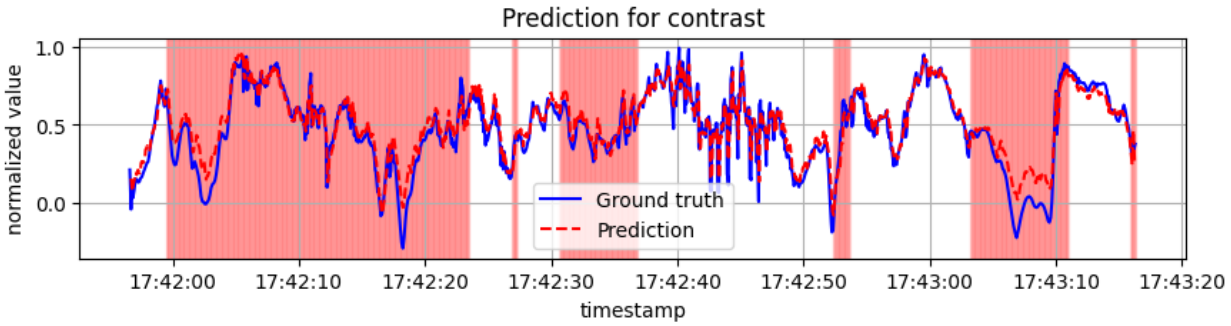
(b)



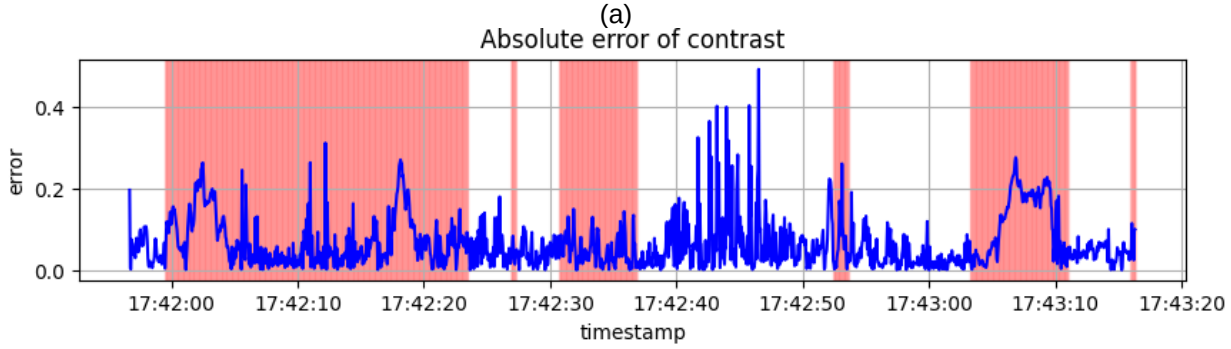
(c)

Fig.29 Multivariable Prediction for brightness

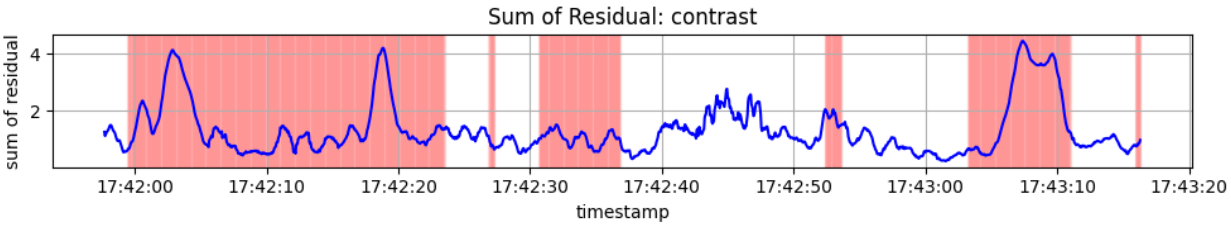
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



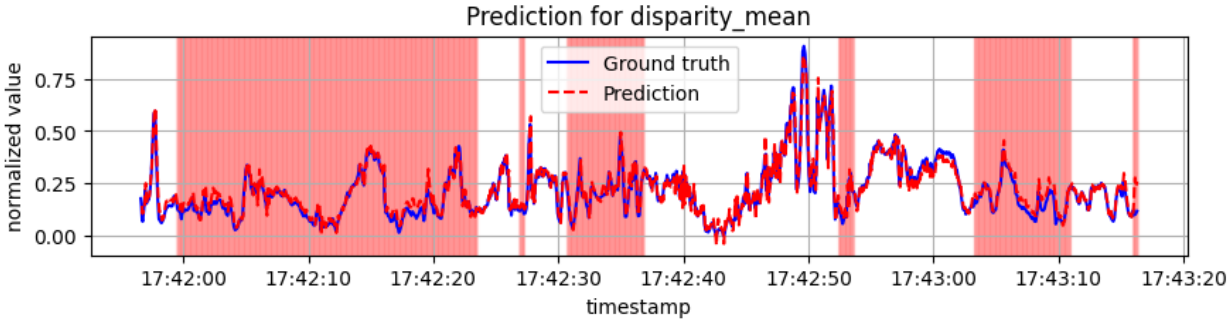
(b)



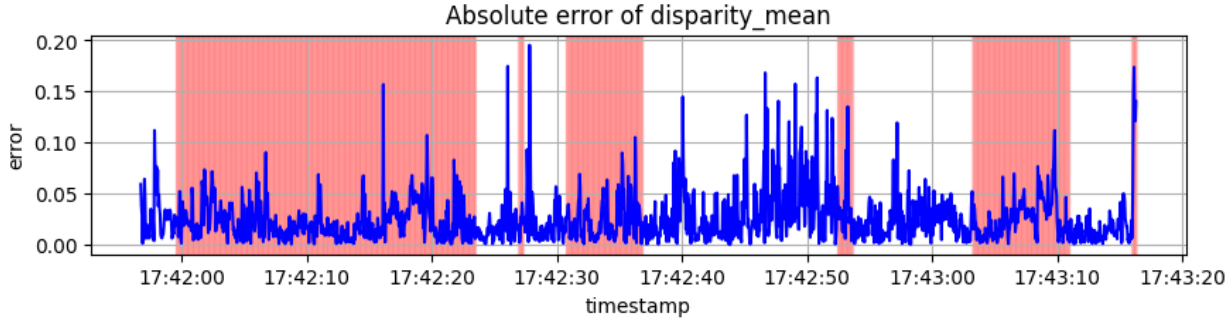
(c)

Fig.30 Multivariable Prediction for contrast

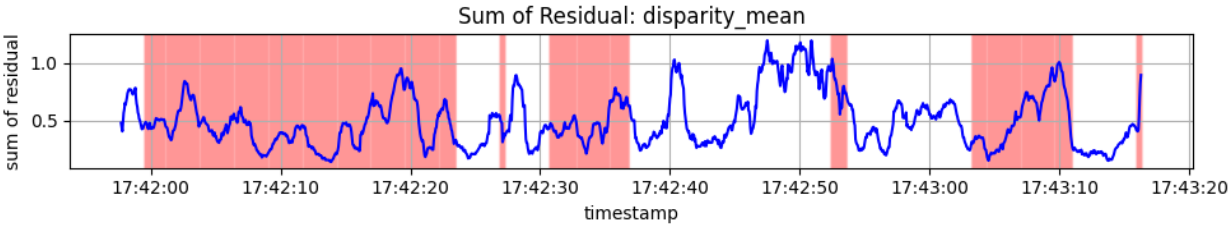
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

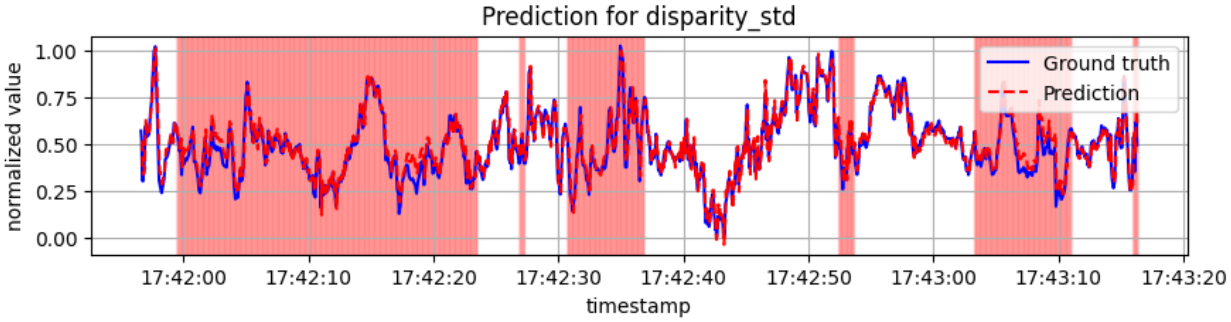


(b)

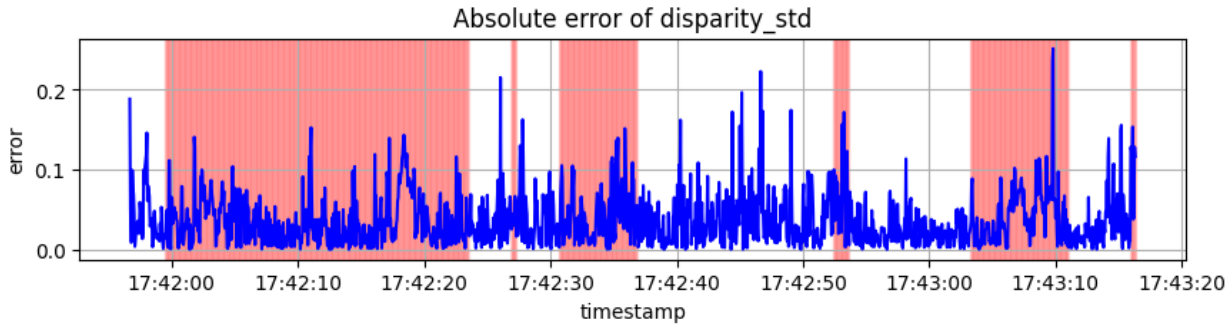


(c)

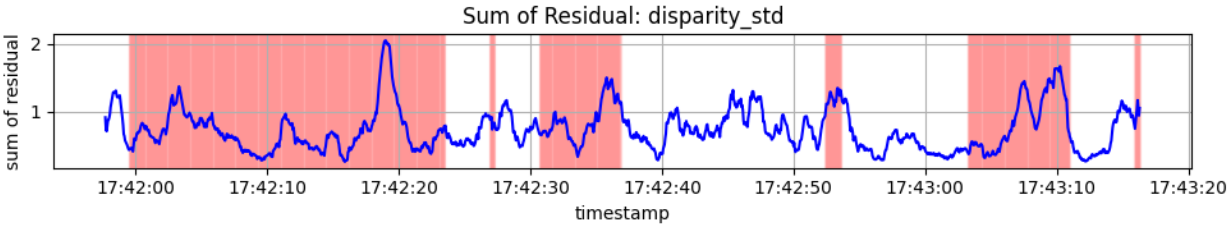
Fig.31 Multivariable Prediction for disparity_mean
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

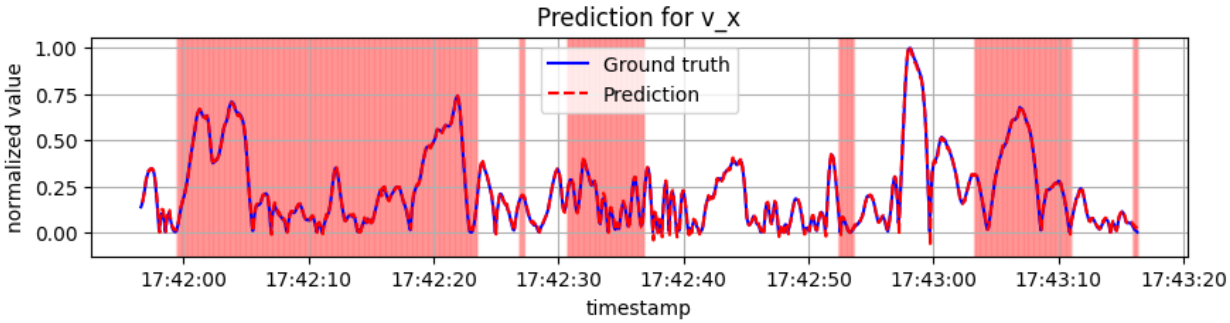


(b)

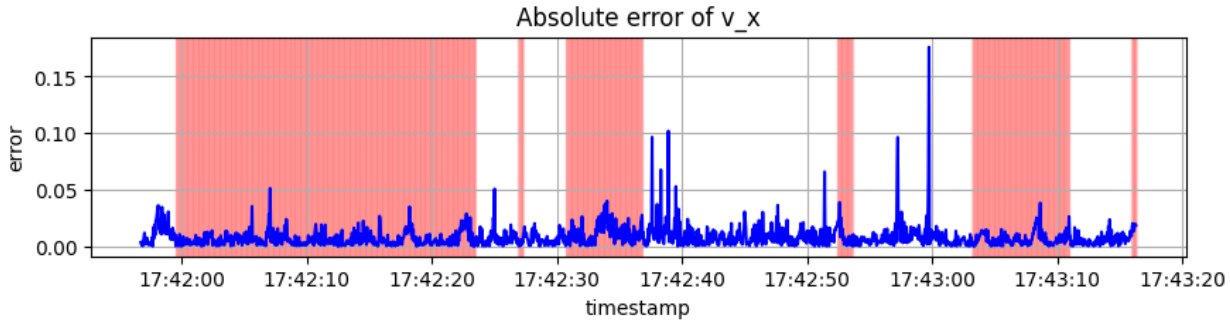


(c)

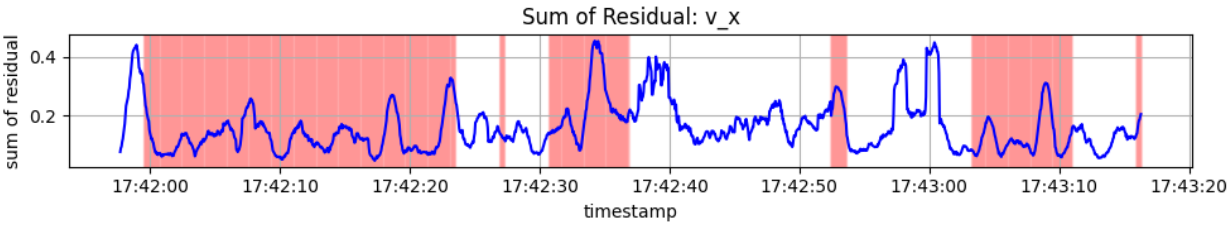
Fig.32 Multivariable Prediction for disparity_std
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

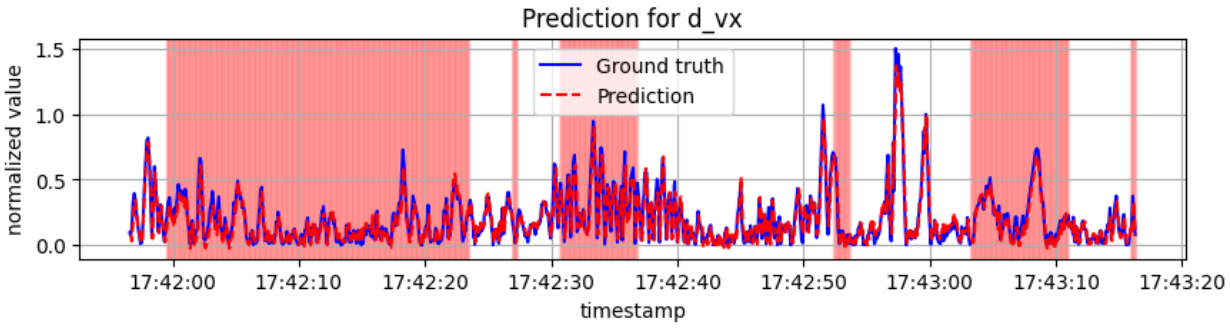


(b)

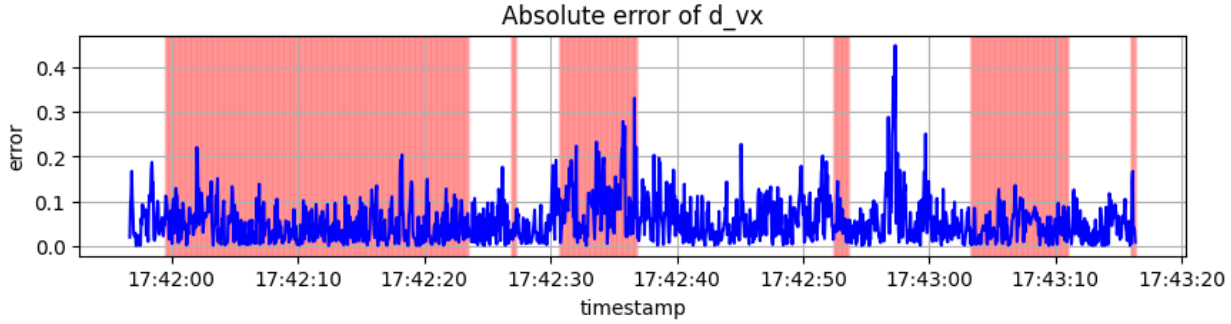


(c)

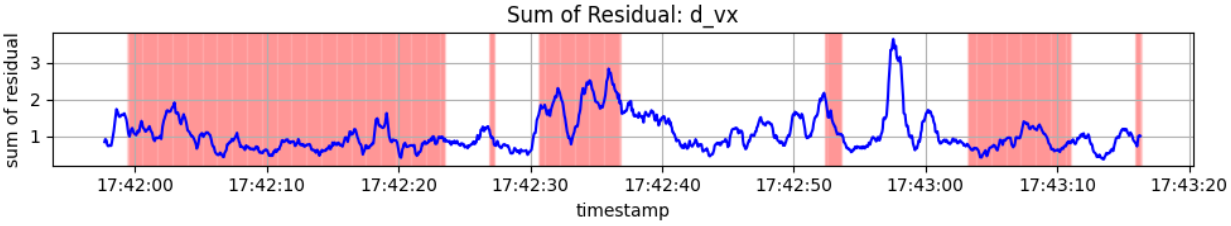
Fig.33 Multivariable Prediction for v_x
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

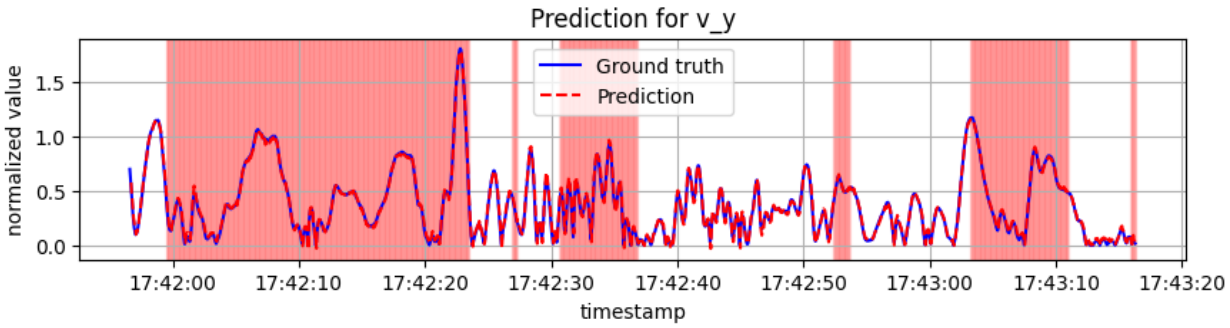


(b)

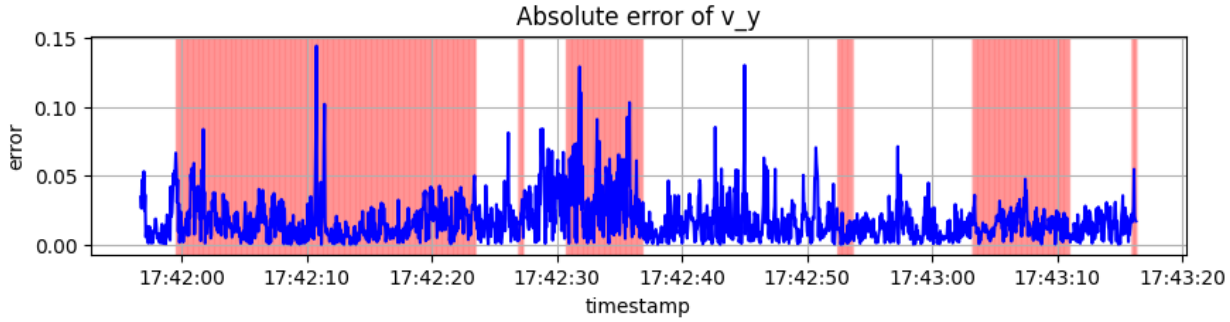


(c)

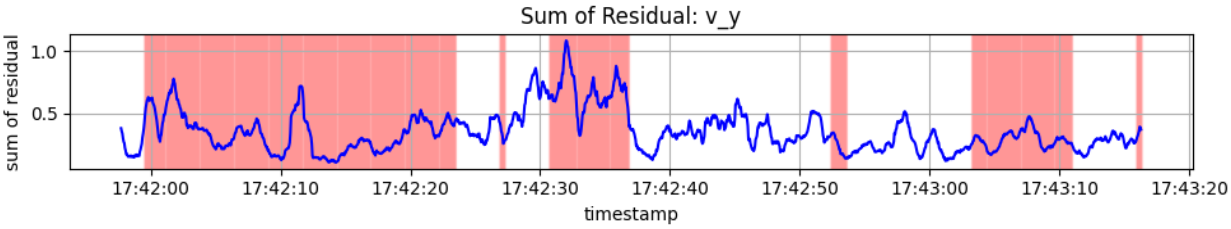
Fig.34 Multivariable Prediction for d_vx
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

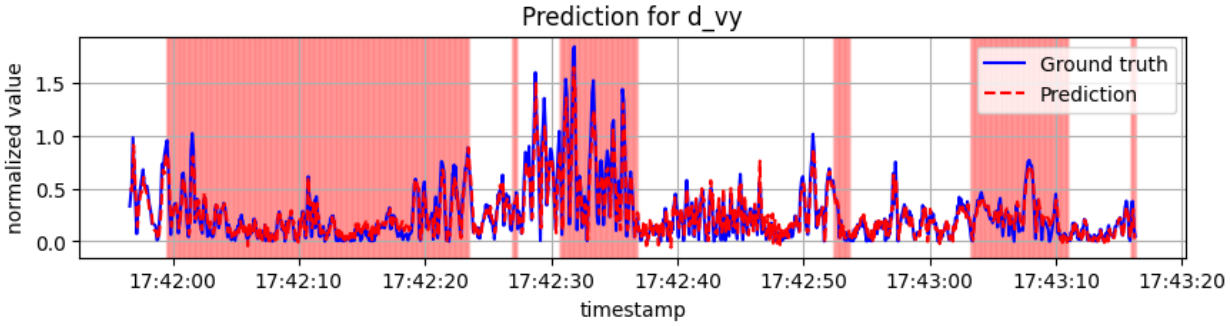


(b)

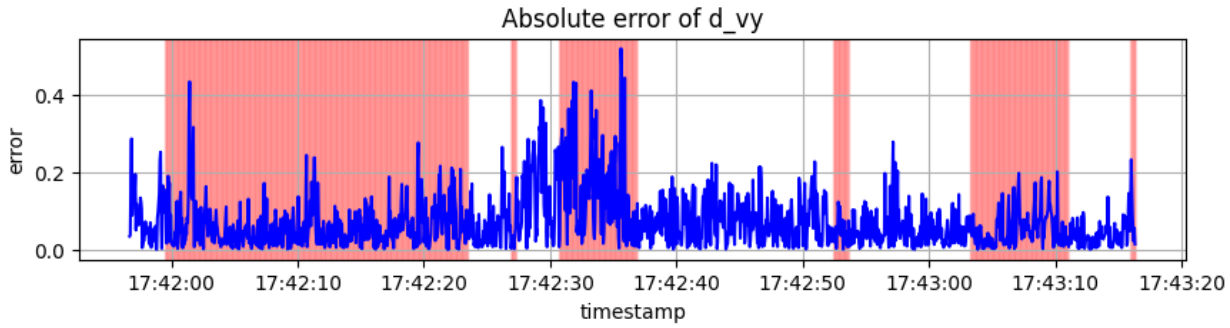


(c)

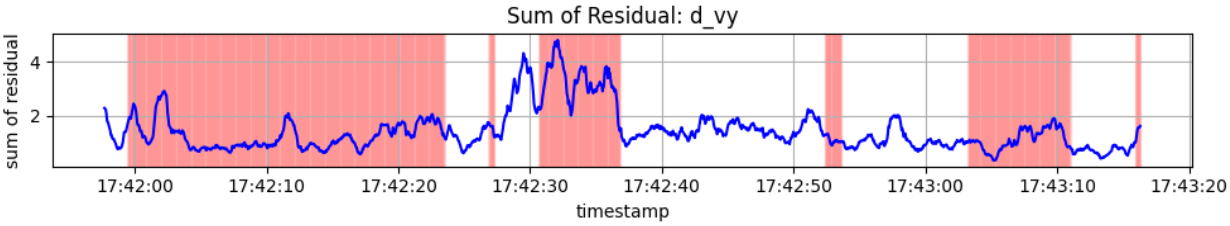
Fig.35 Multivariable Prediction for v_y
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

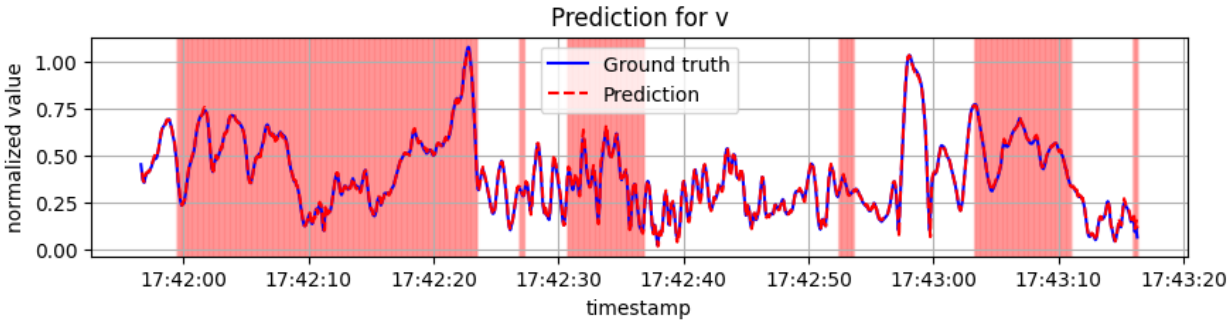


(b)

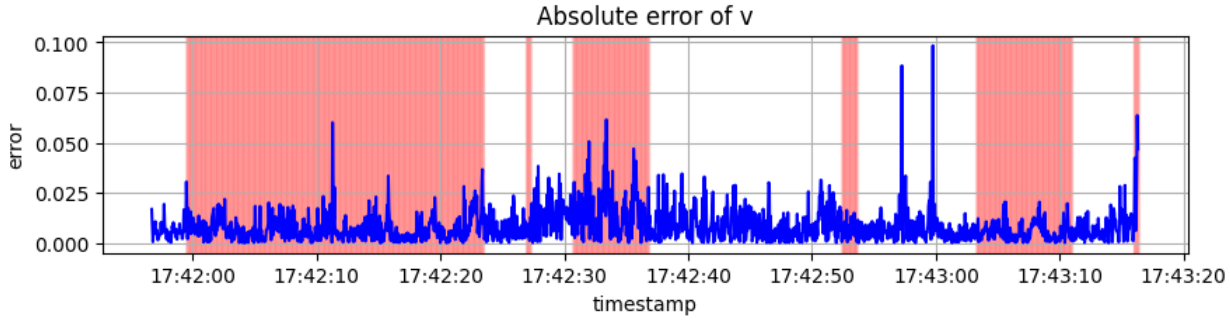


(c)

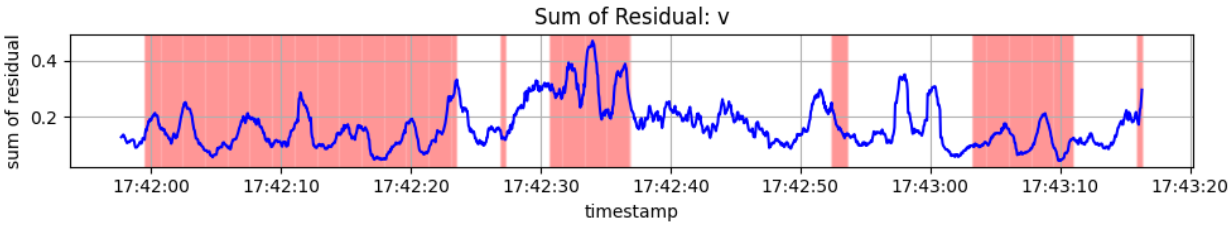
Fig.36 Multivariable Prediction for d_vy
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



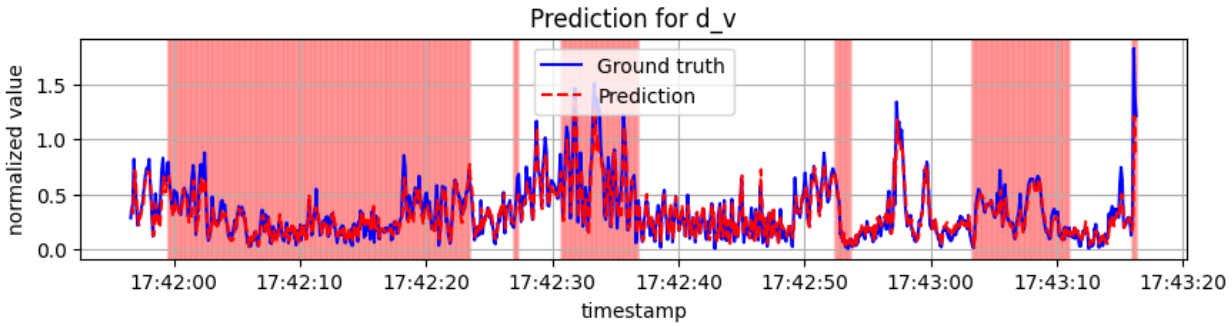
(b)



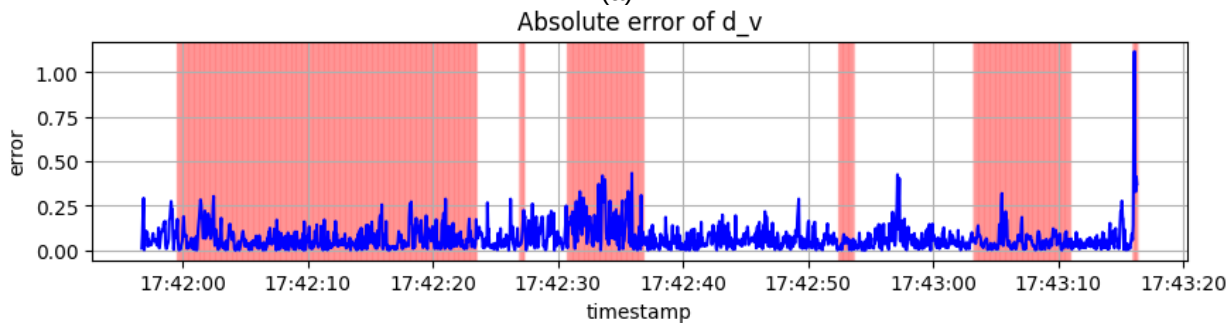
(c)

Fig.37 Multivariable Prediction for v

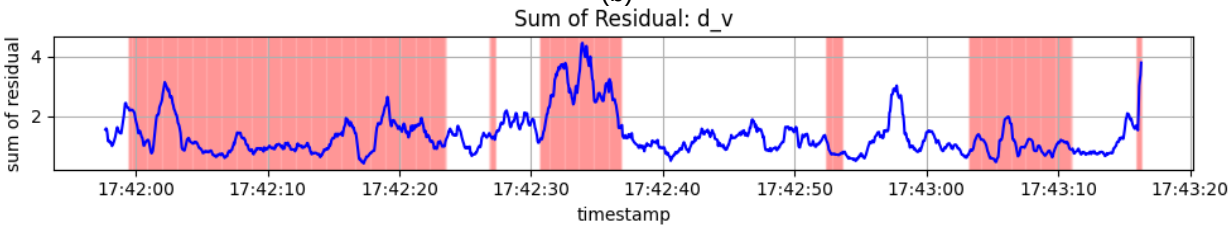
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



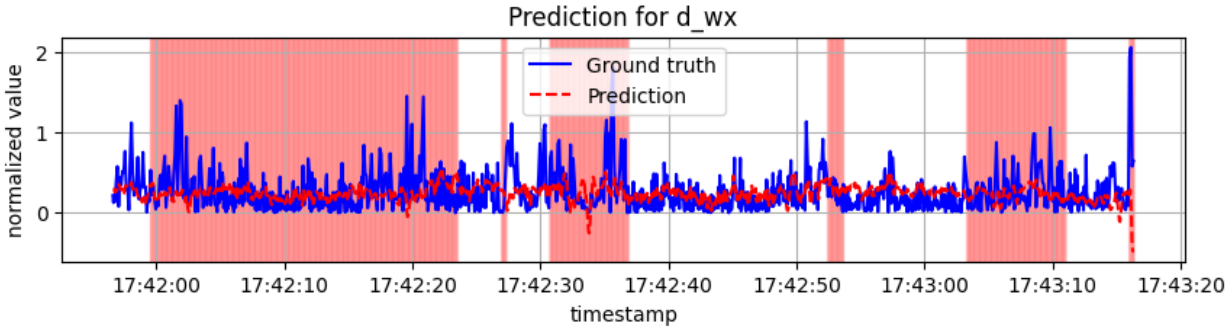
(b)



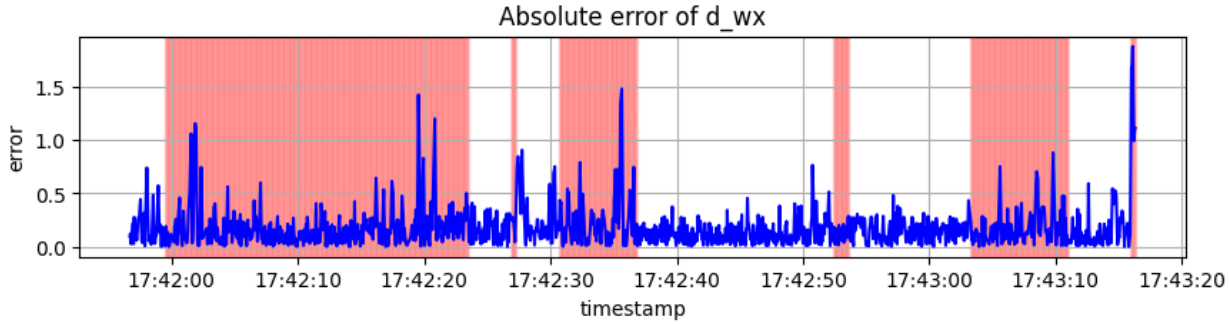
(c)

Fig.38 Multivariable Prediction for d_v

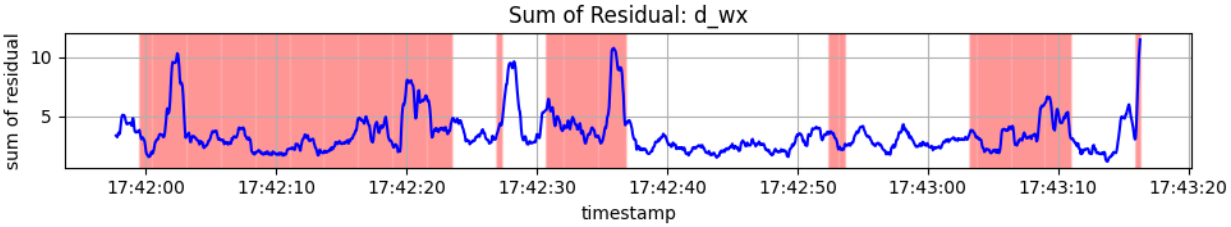
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

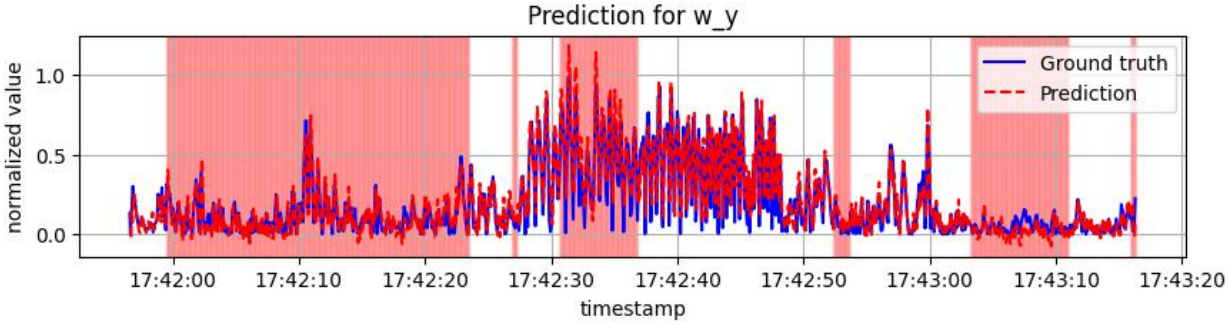


(b)

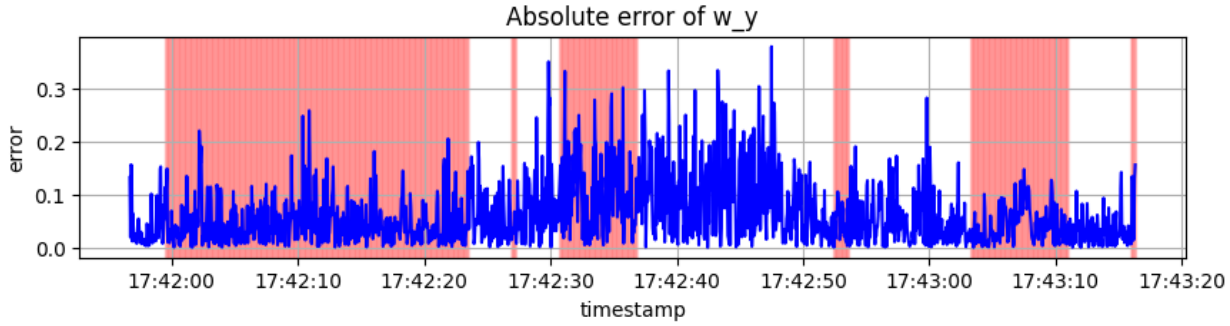


(c)

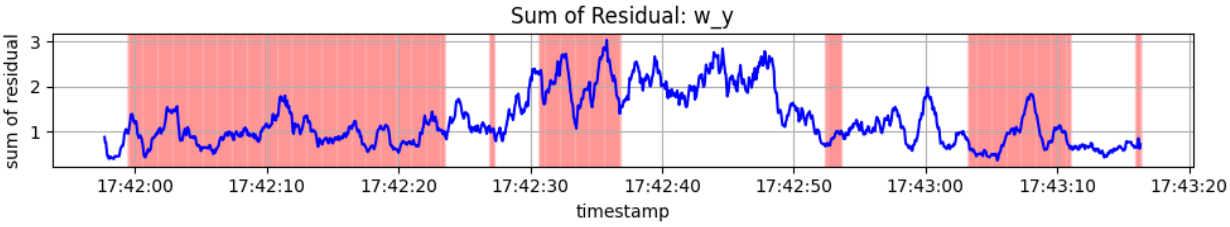
Fig.39 Multivariable Prediction for d_wx
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

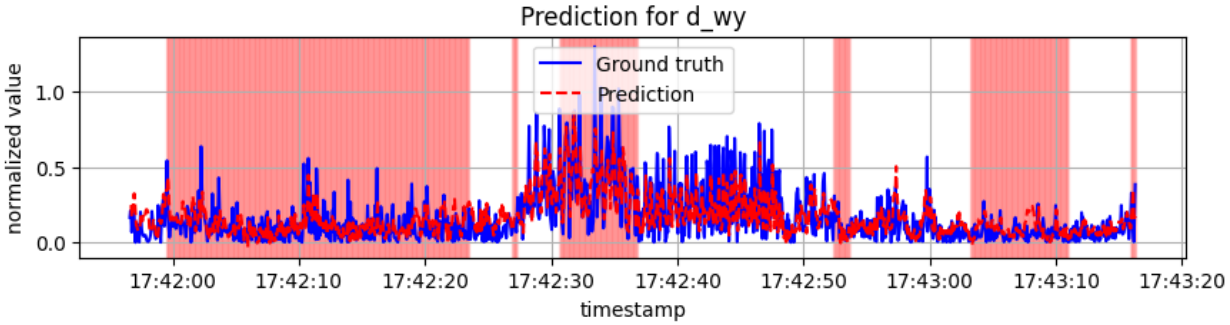


(b)

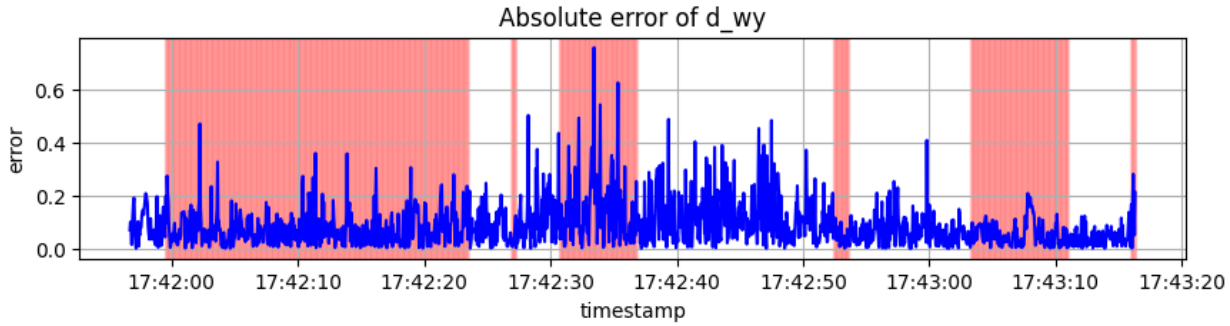


(c)

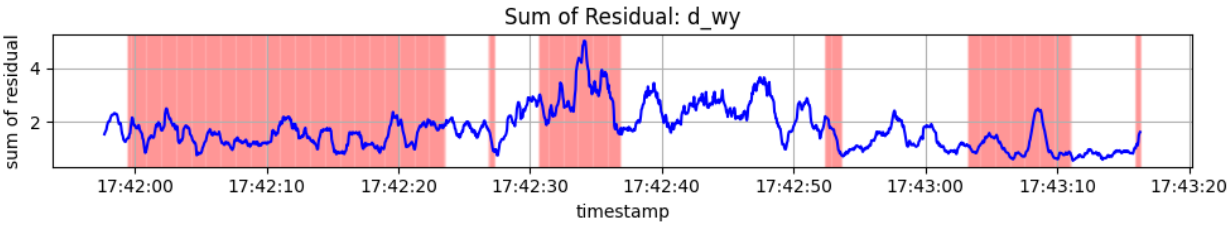
Fig.40 Multivariable Prediction for w_y
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

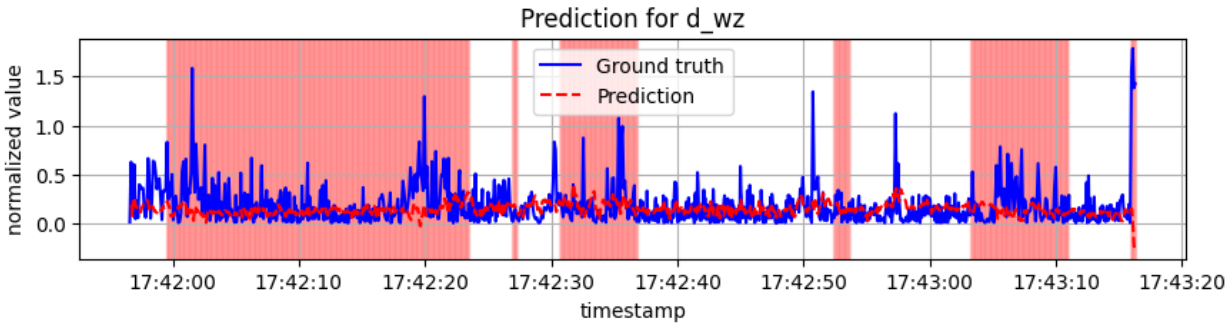


(b)

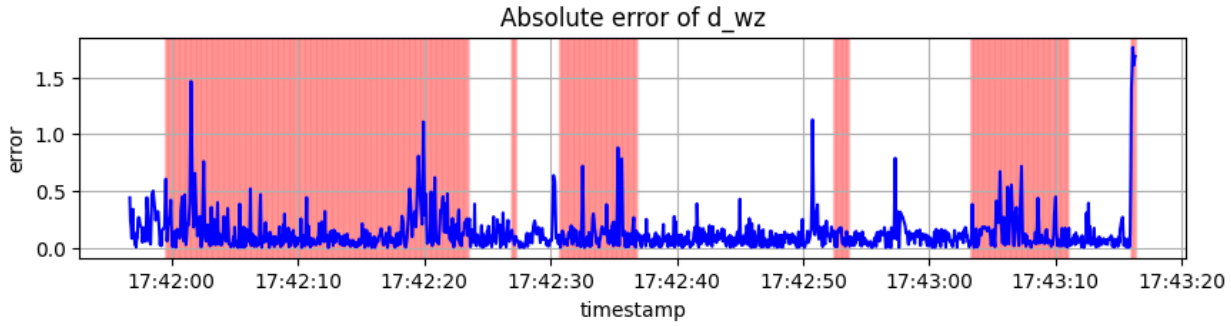


(c)

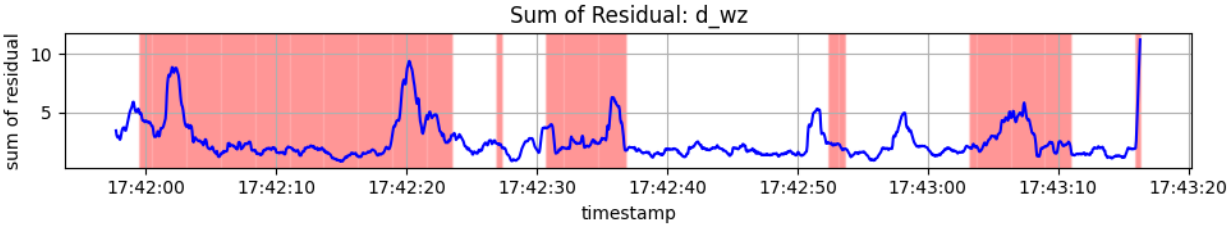
Fig.41 Multivariable Prediction for d_wy
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

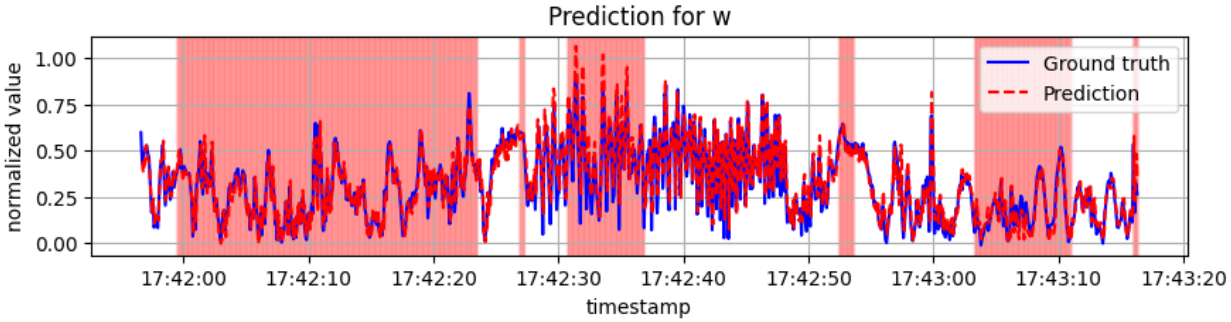


(b)

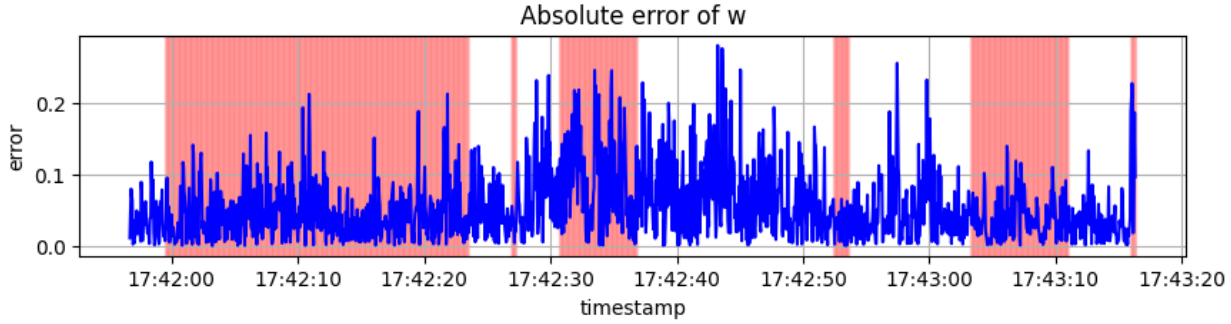


(c)

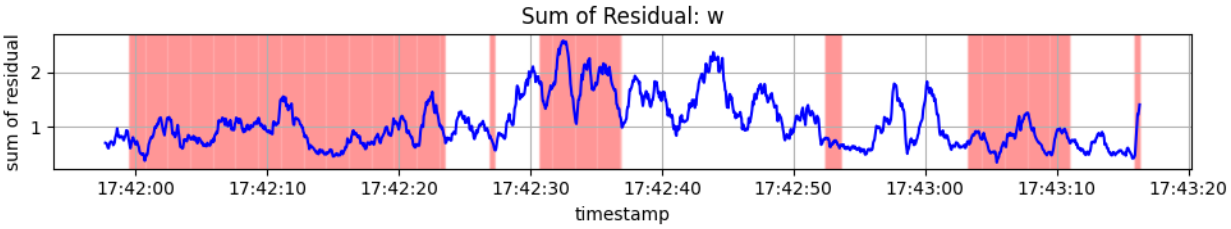
Fig.42 Multivariable Prediction for d_wz
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



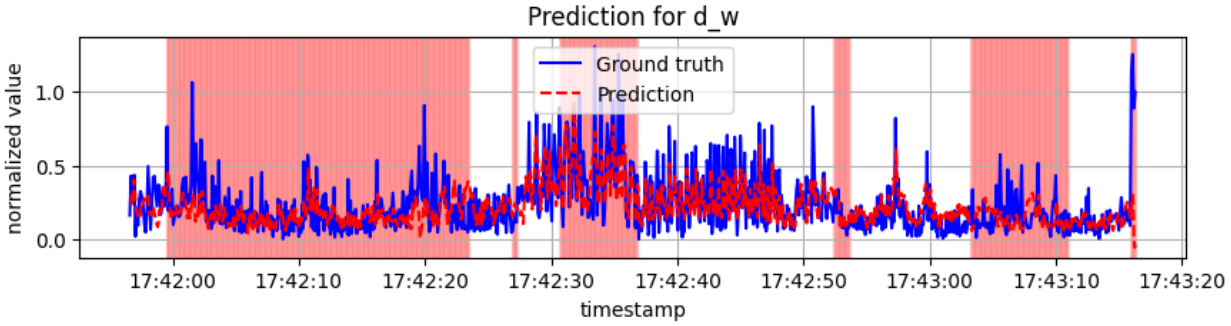
(b)



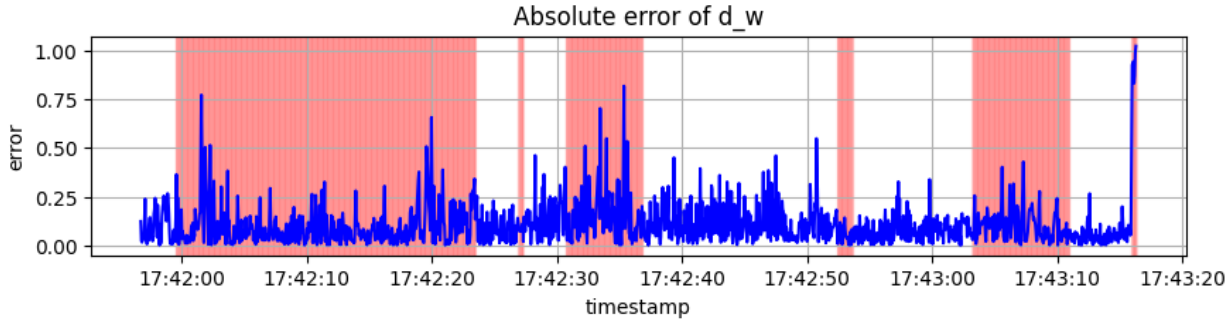
(c)

Fig.43 Multivariable Prediction for w

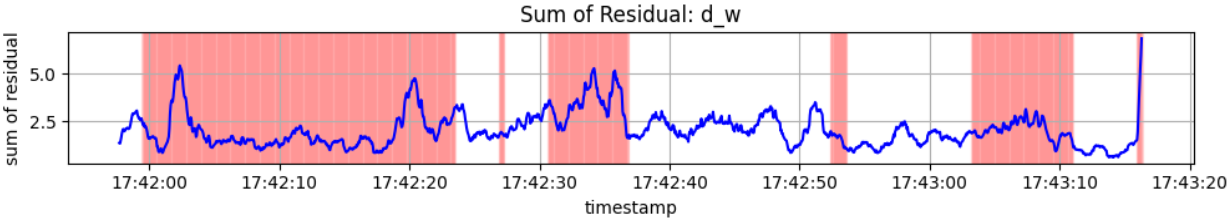
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



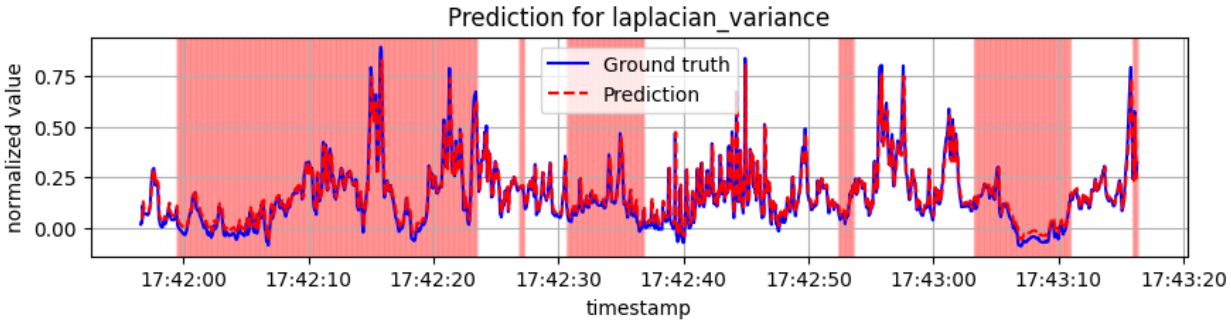
(b)



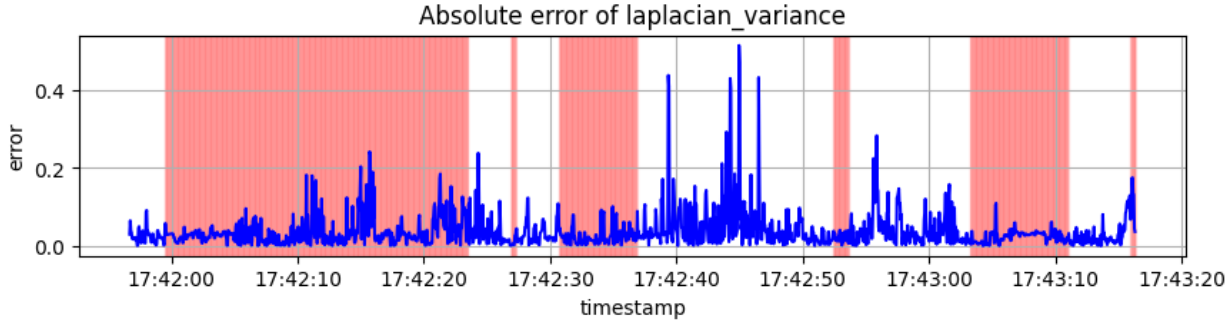
(c)

Fig.44 Multivariable Prediction for d_w
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals

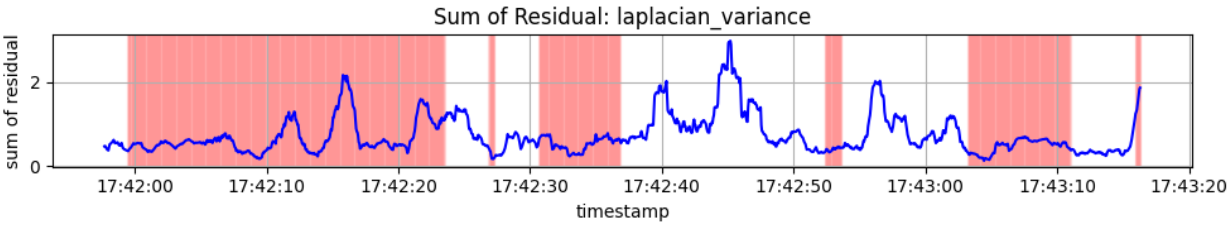
B.II Univariable Prediction



(a)

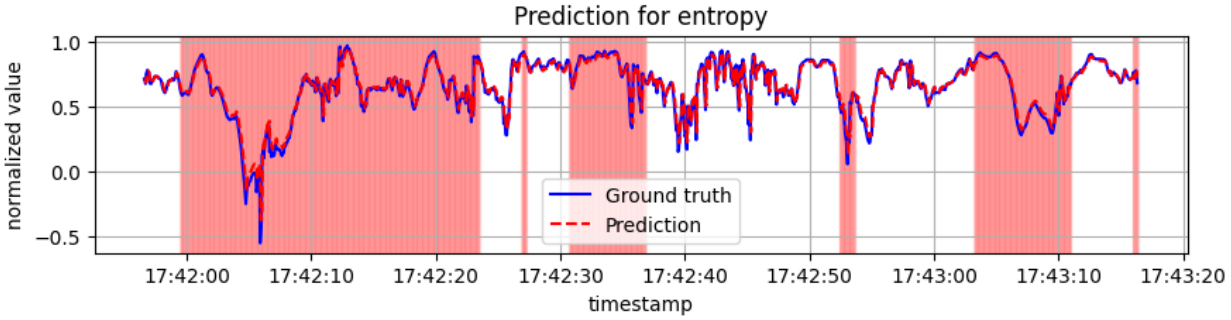


(b)

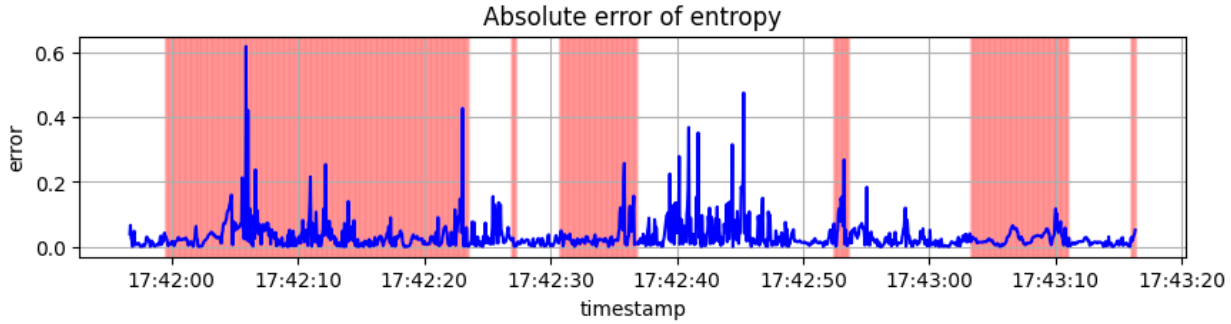


(c)

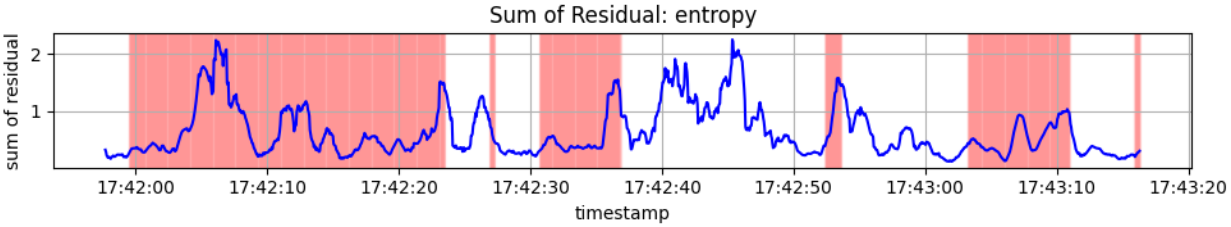
Fig.45 Univariable Prediction for laplacian_variance
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

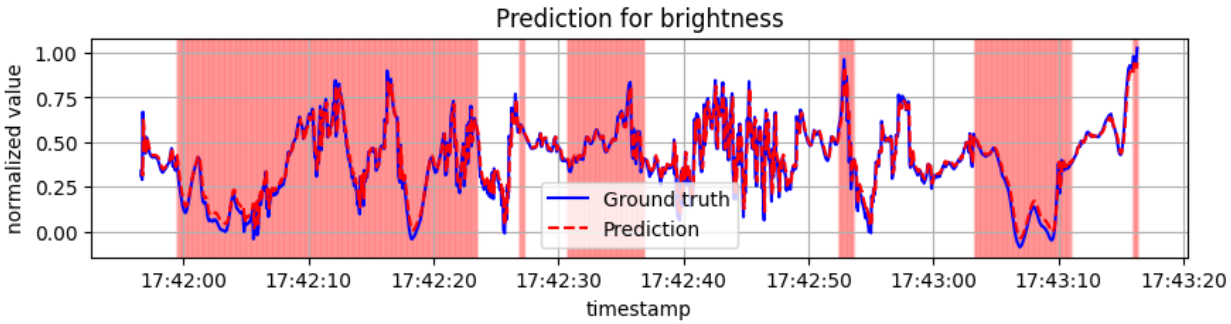


(b)

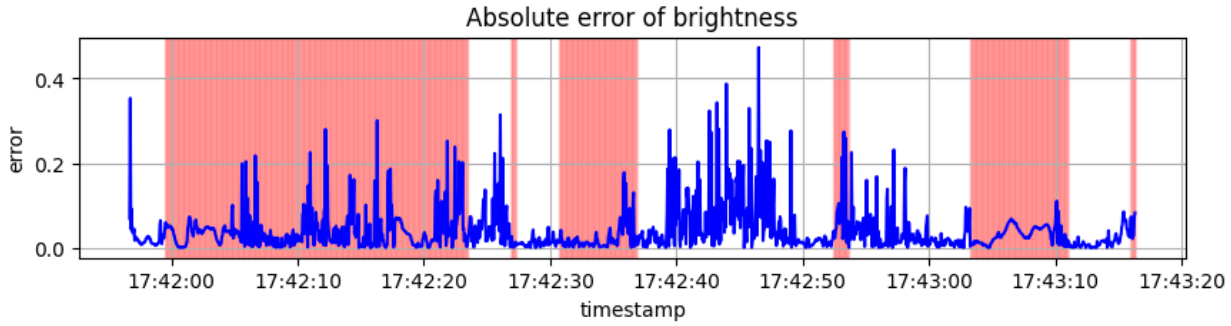


(c)

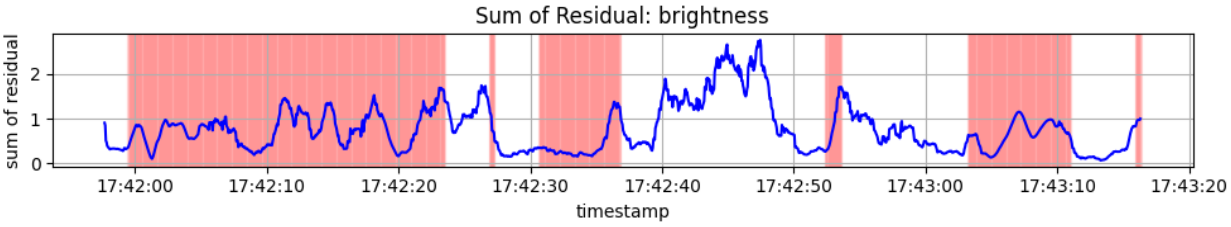
Fig.46 Univariable Prediction for entropy
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

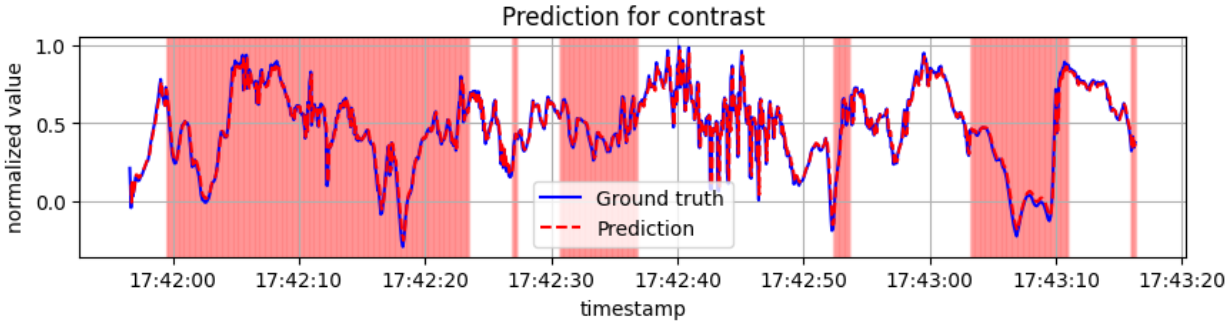


(b)

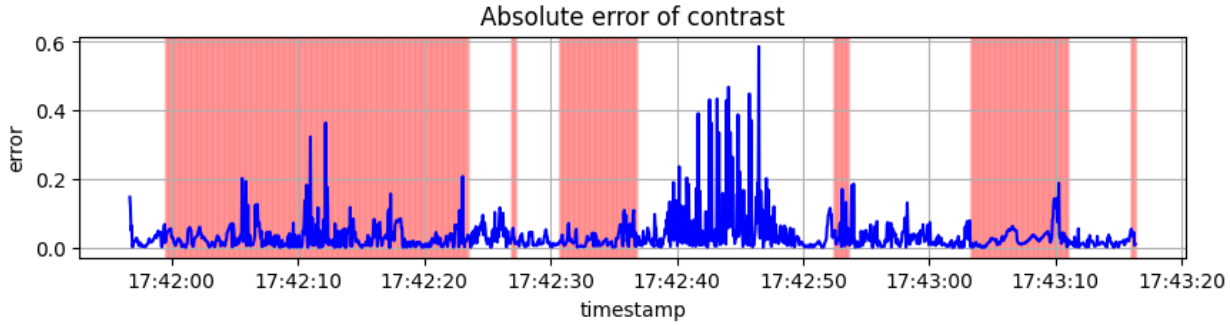


(c)

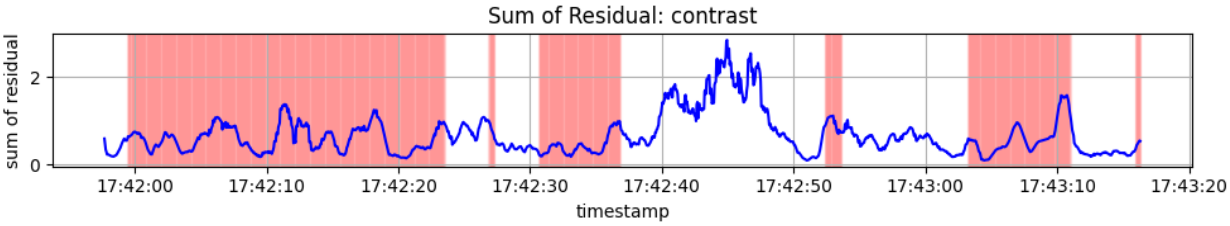
Fig.47 Univariable Prediction for brightness
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

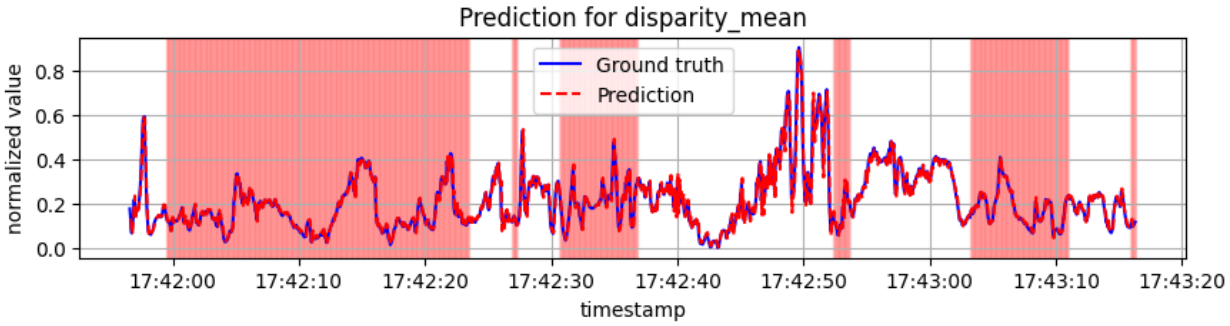


(b)

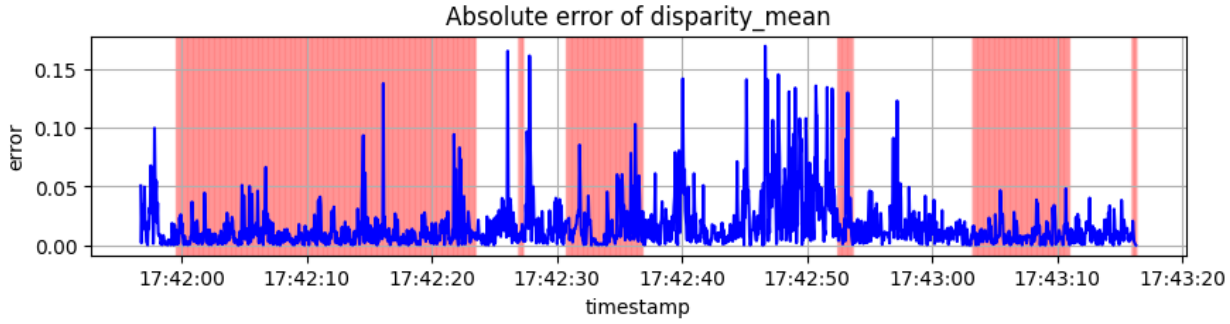


(c)

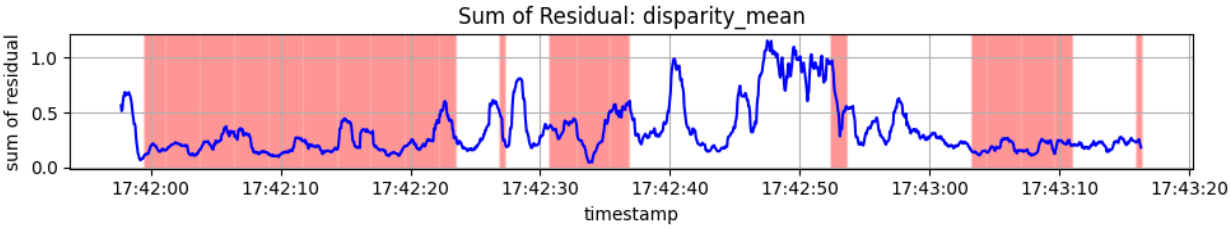
Fig.48 Univariable Prediction for contrast
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

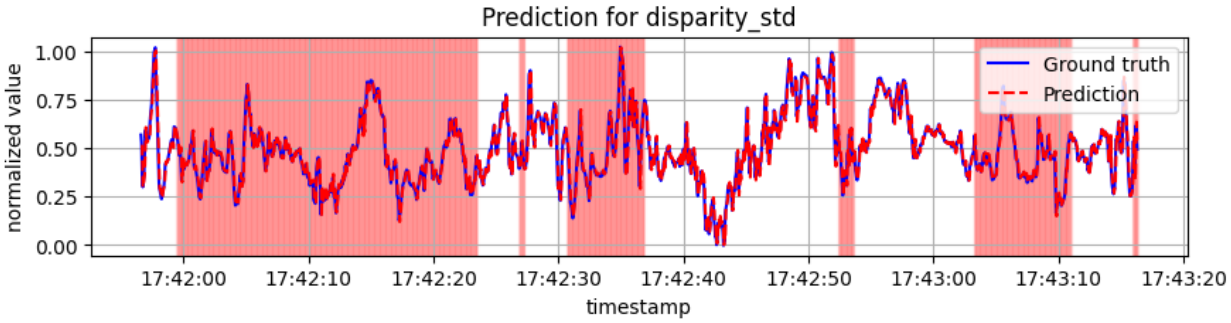


(b)

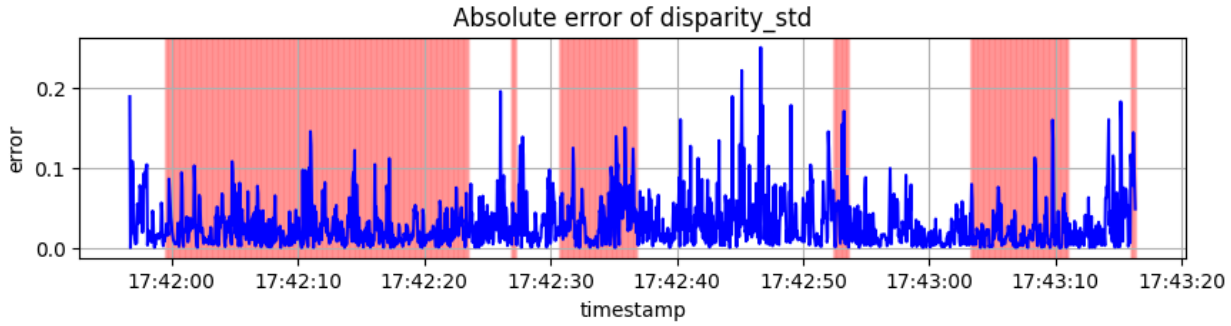


(c)

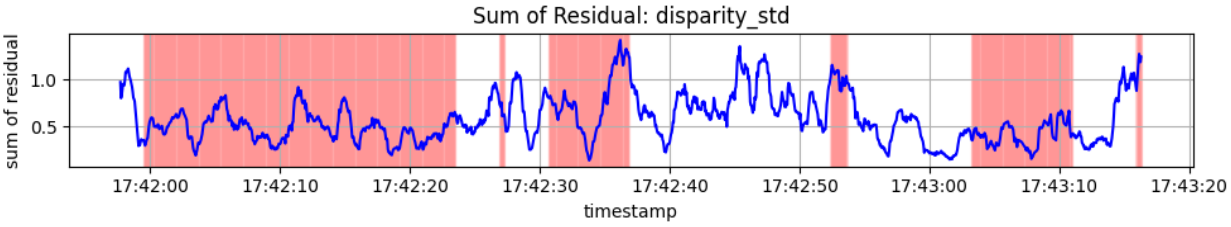
Fig.49 Univariable Prediction for disparity_mean
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

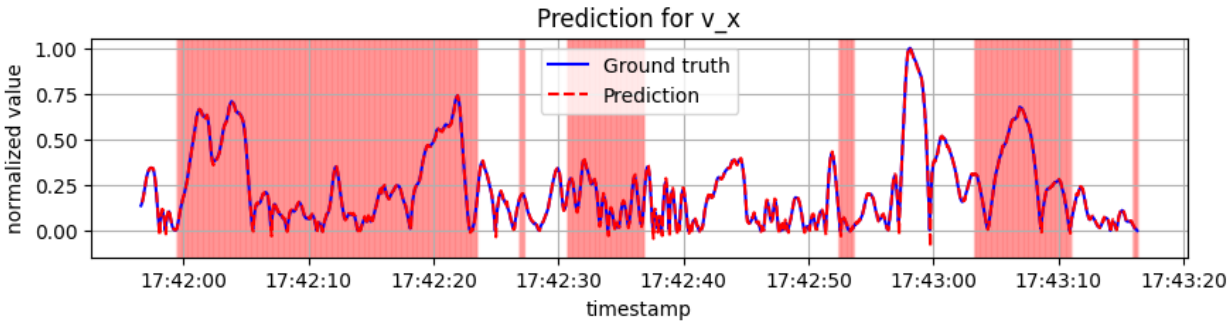


(b)

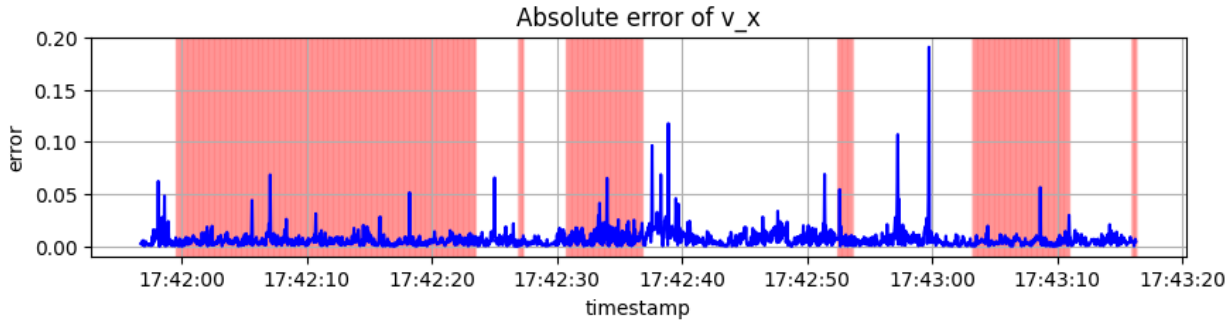


(c)

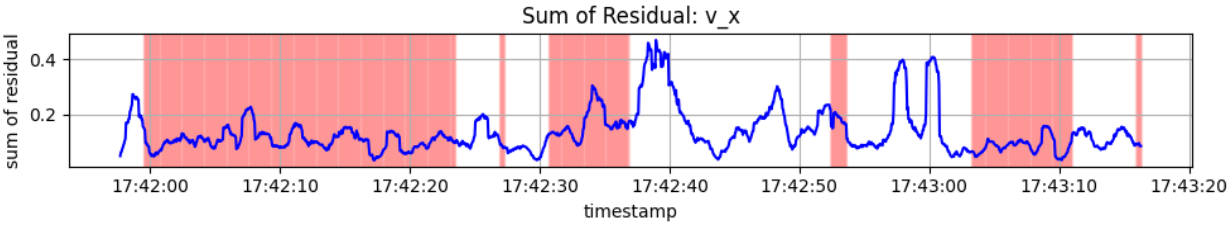
Fig.50 Univariable Prediction for disparity_std
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

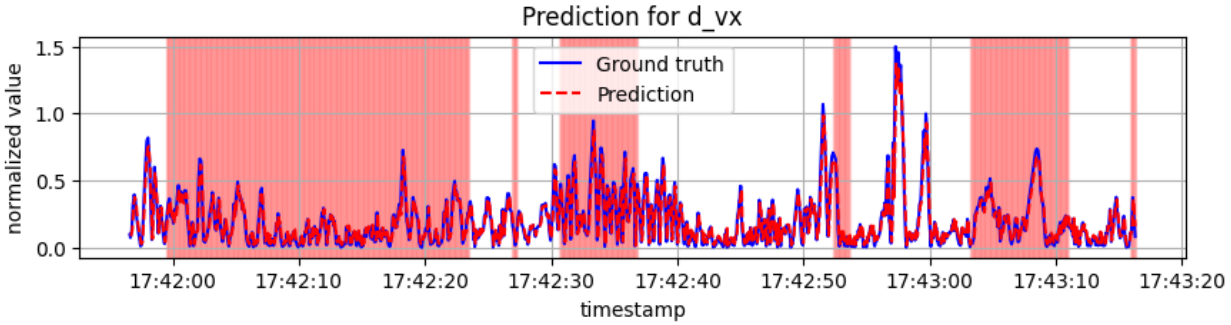


(b)

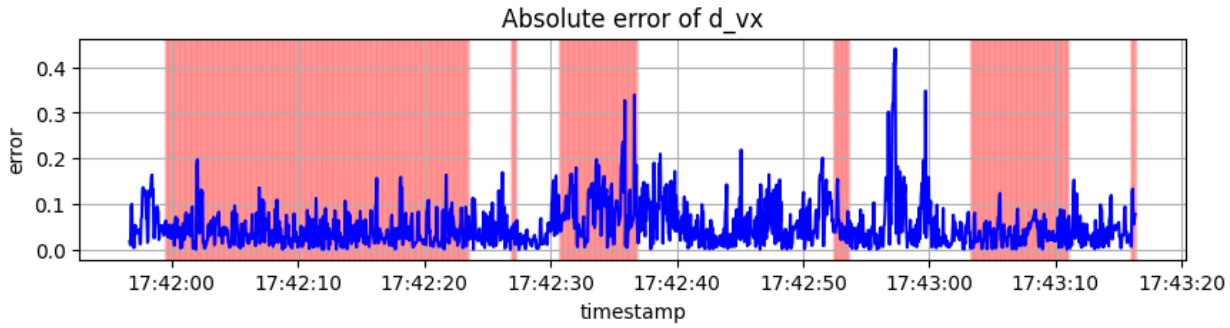


(c)

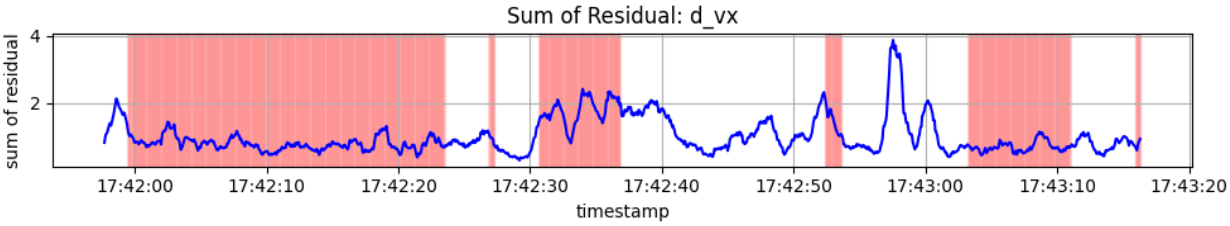
Fig.51 Univariable Prediction for v_x
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

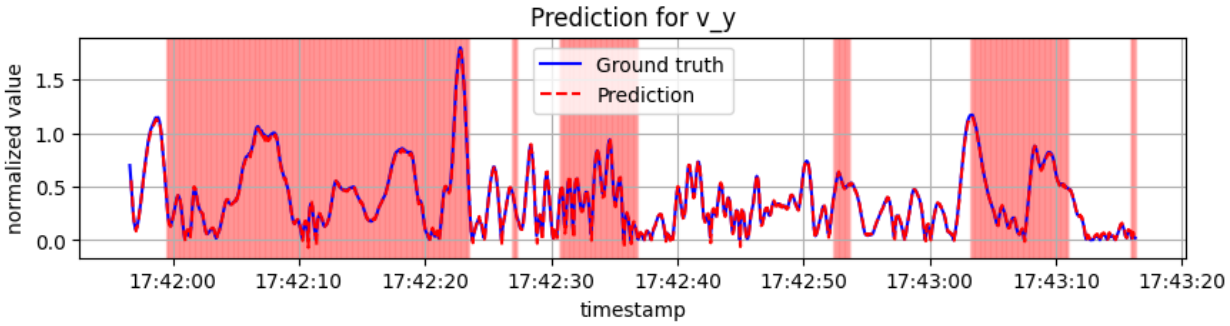


(b)

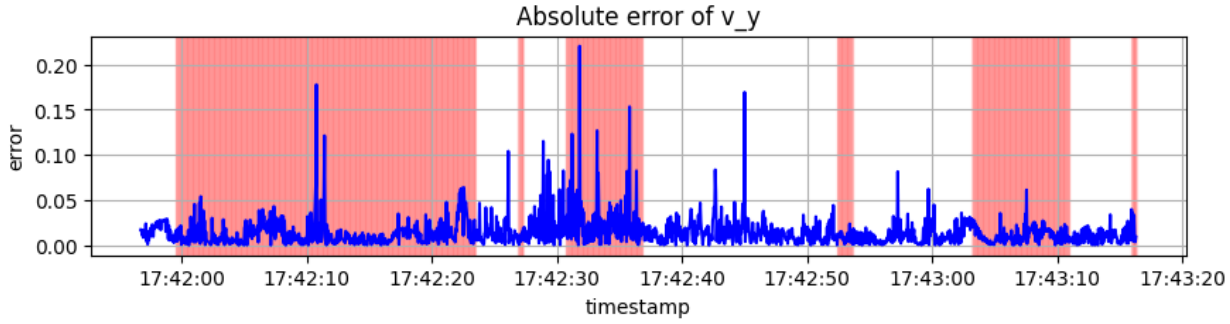


(c)

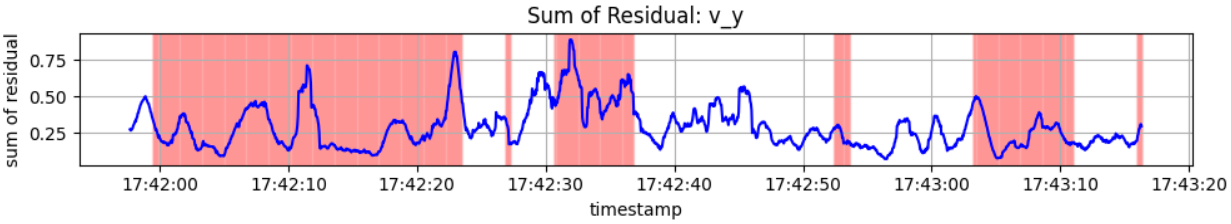
Fig.52 Univariable Prediction for d_vx
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



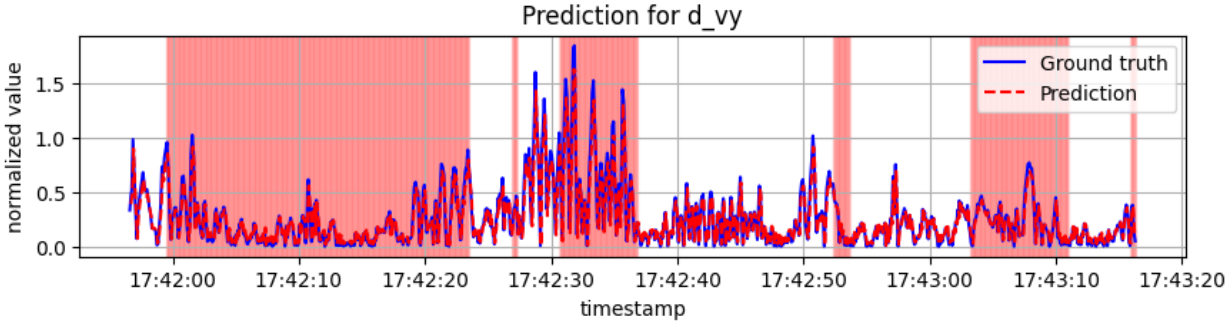
(b)



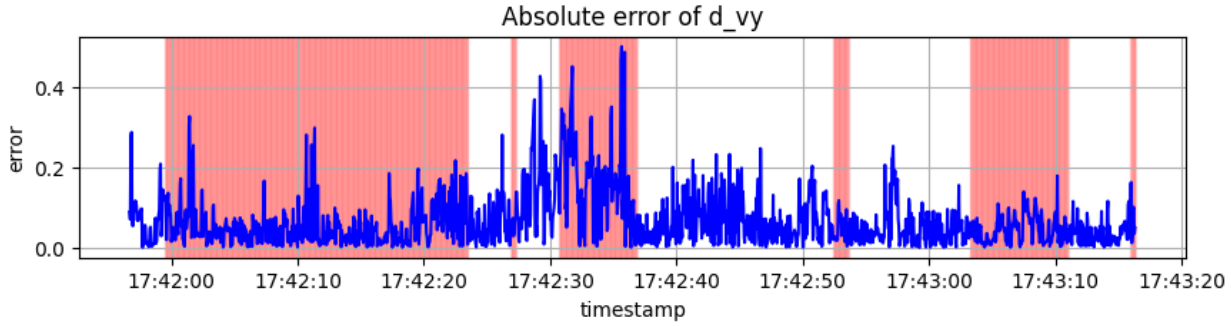
(c)

Fig.53 Univariable Prediction for v_y

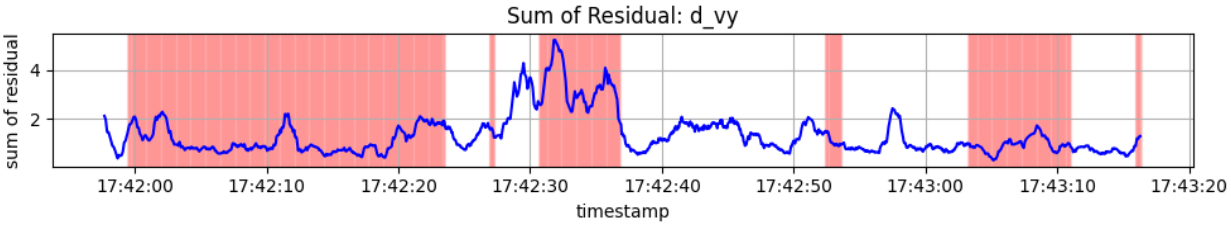
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



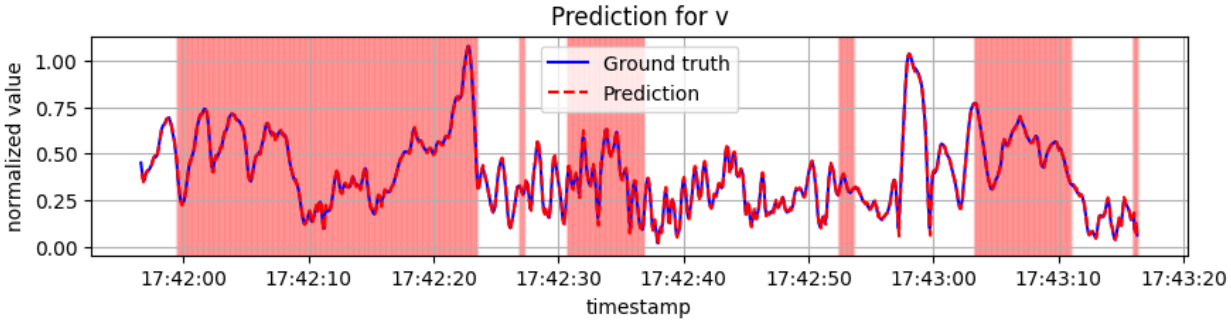
(b)



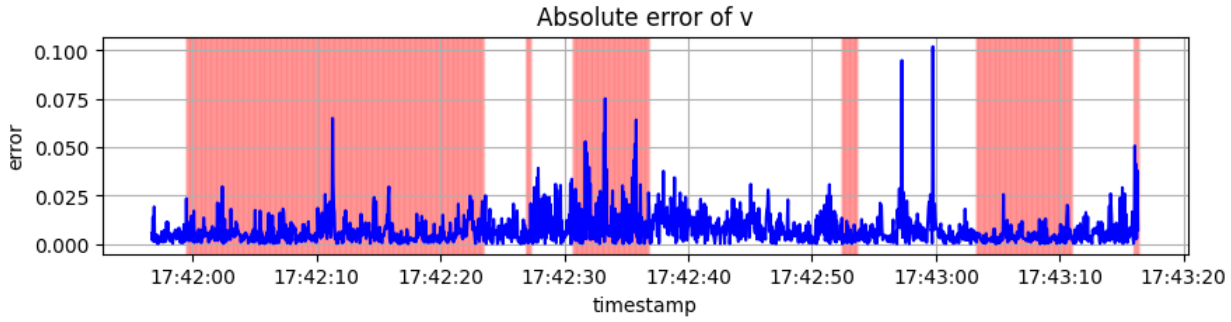
(c)

Fig.54 Univariable Prediction for d_vy

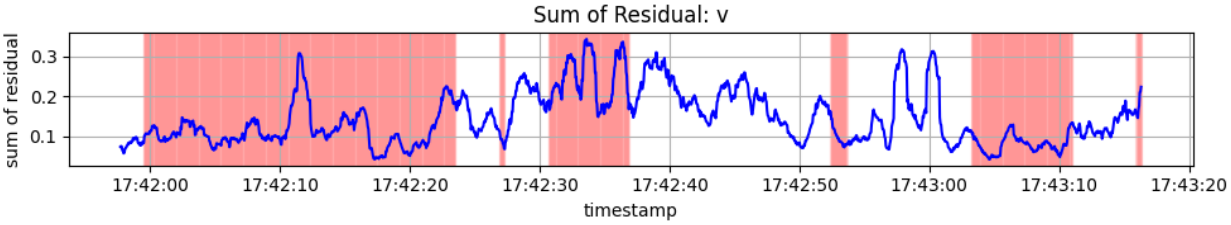
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



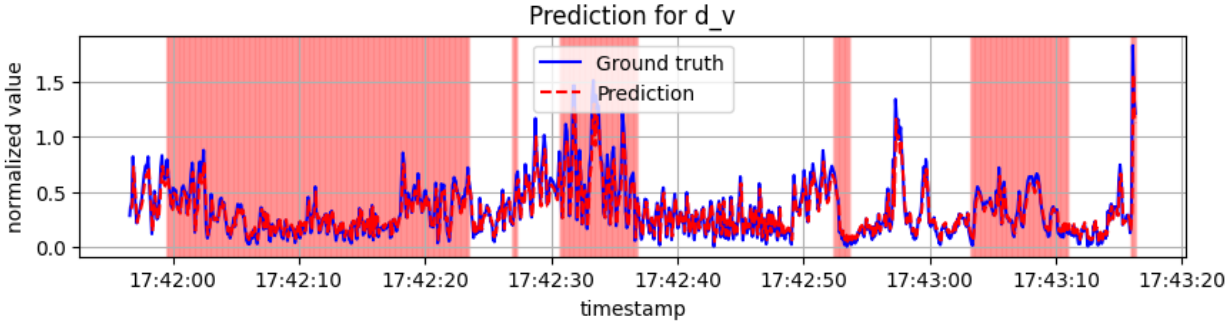
(b)



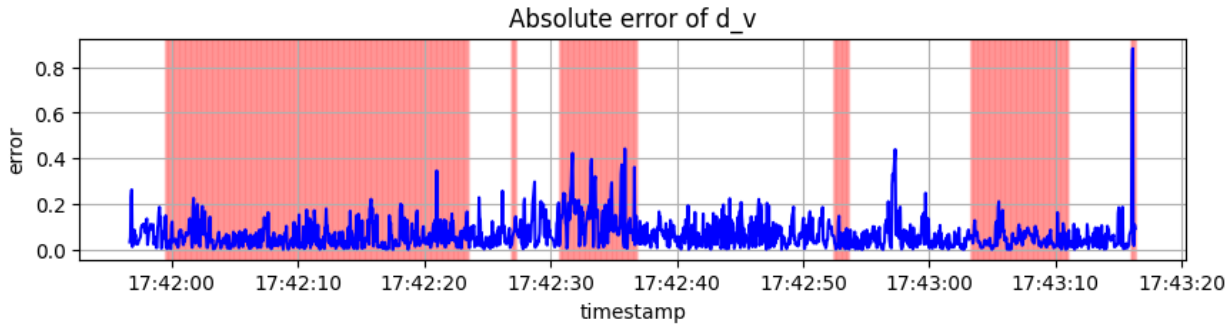
(c)

Fig.55 Univariable Prediction for v

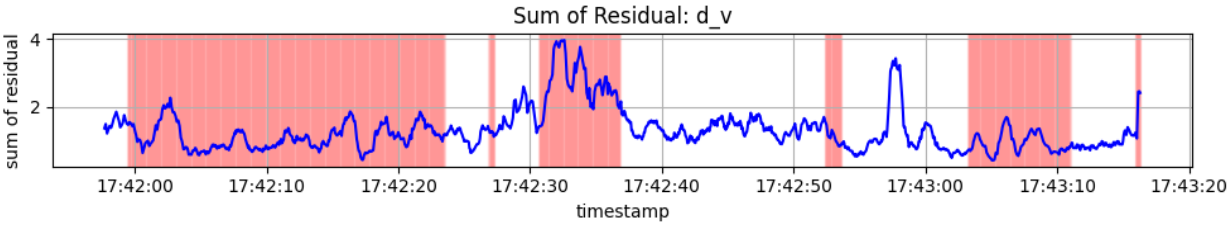
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

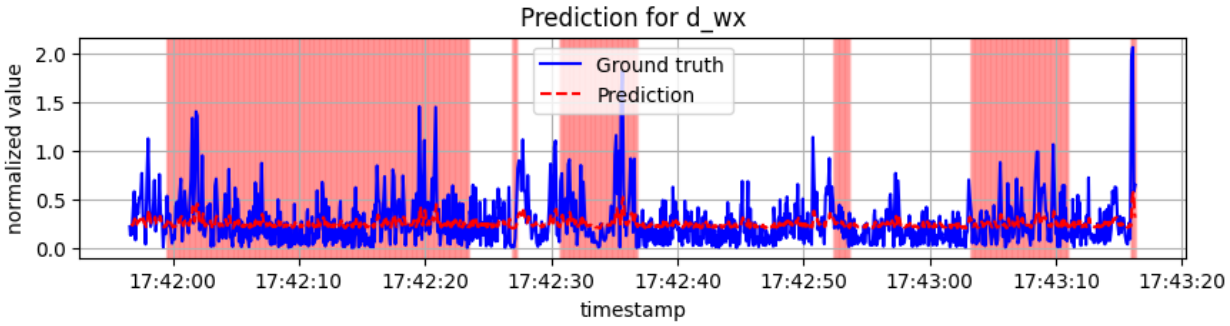


(b)

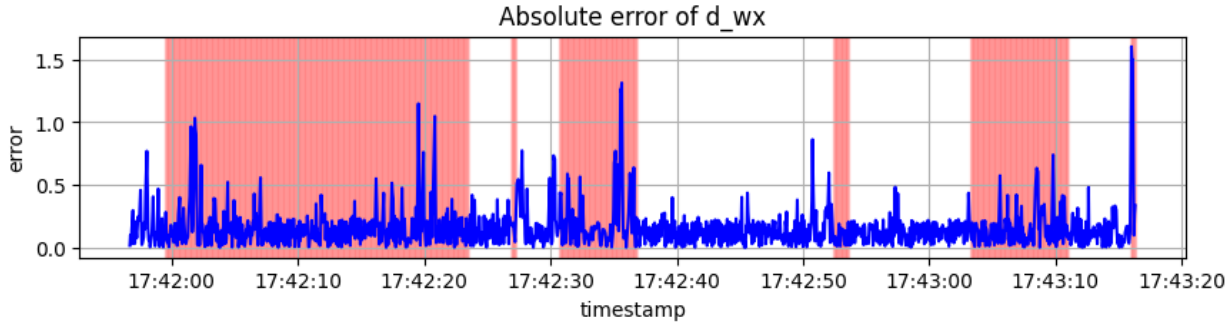


(c)

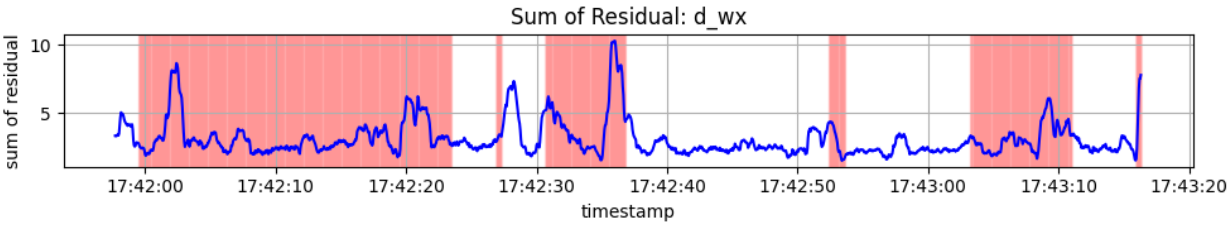
Fig.56 Univariable Prediction for d_v
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



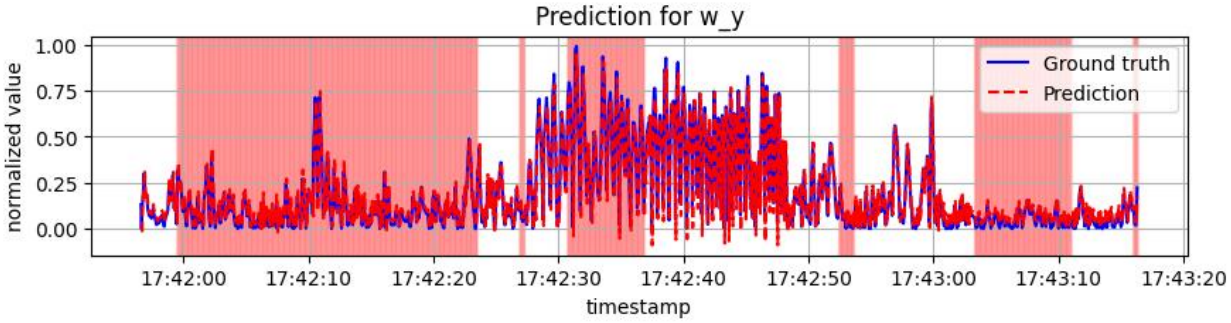
(b)



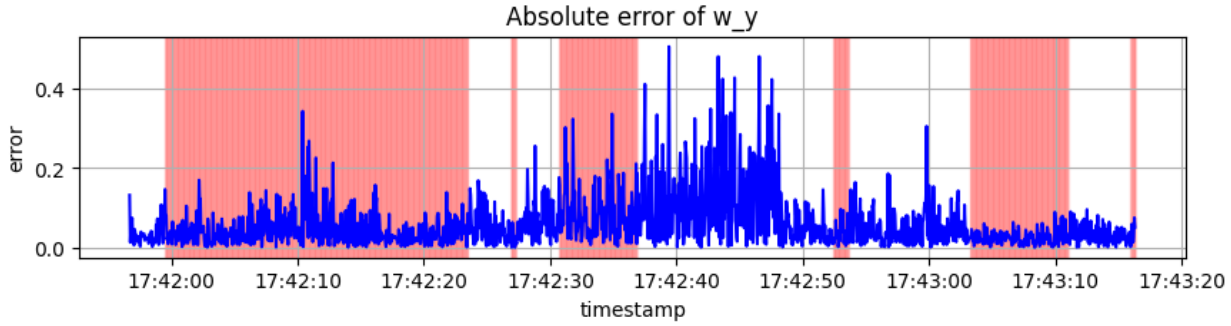
(c)

Fig.57 Univariable Prediction for d_wx

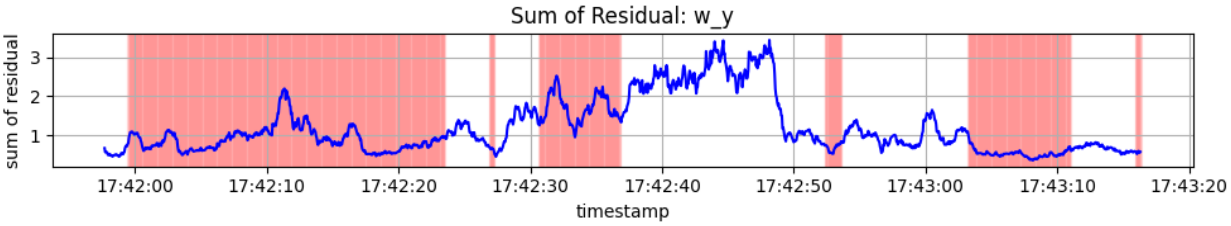
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



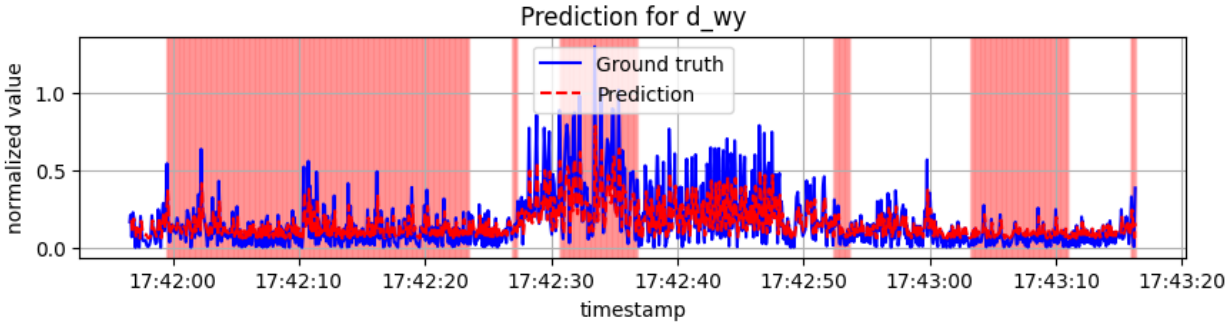
(b)



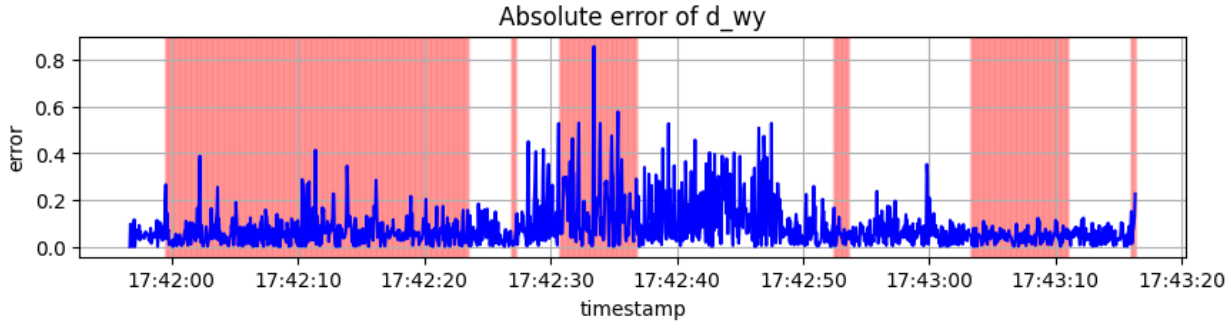
(c)

Fig.58 Univariable Prediction for w_y

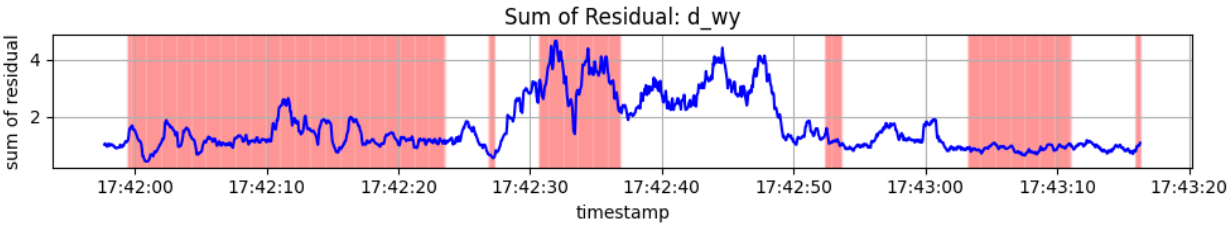
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)

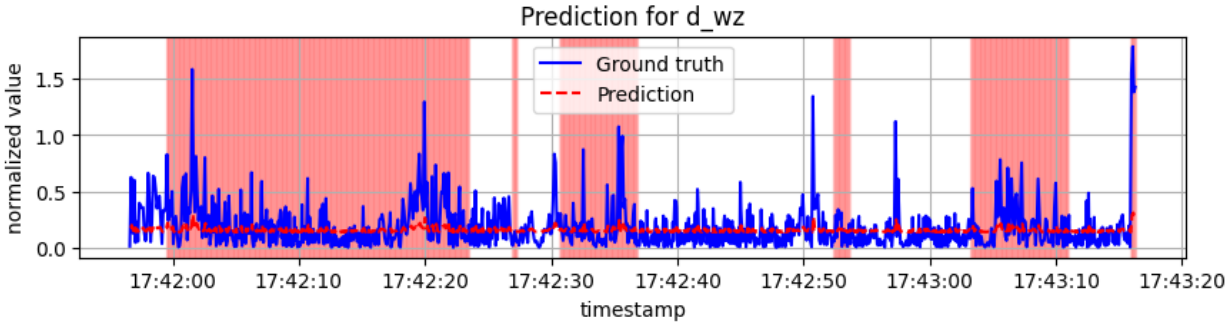


(b)

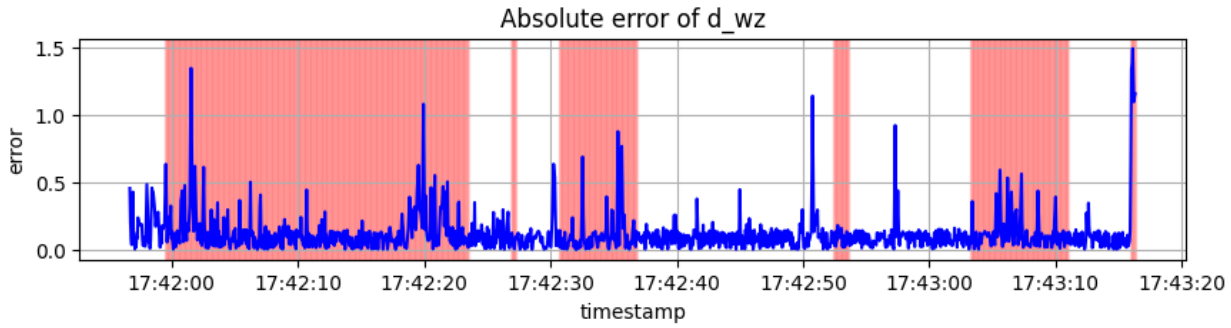


(c)

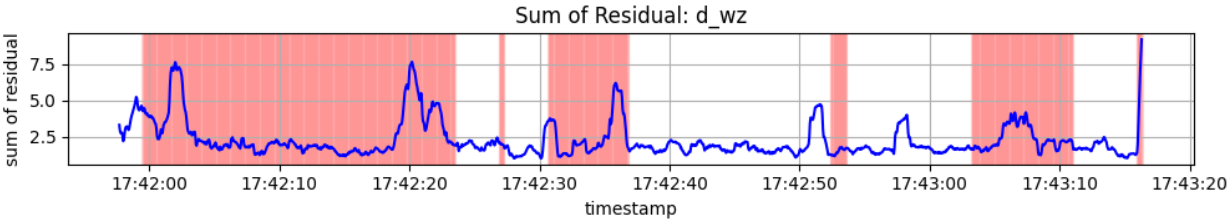
Fig.59 Univariable Prediction for d_wy
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



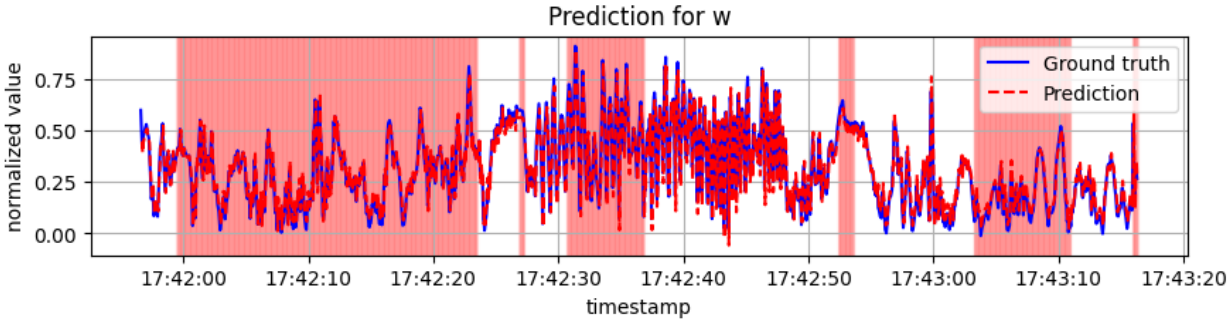
(b)



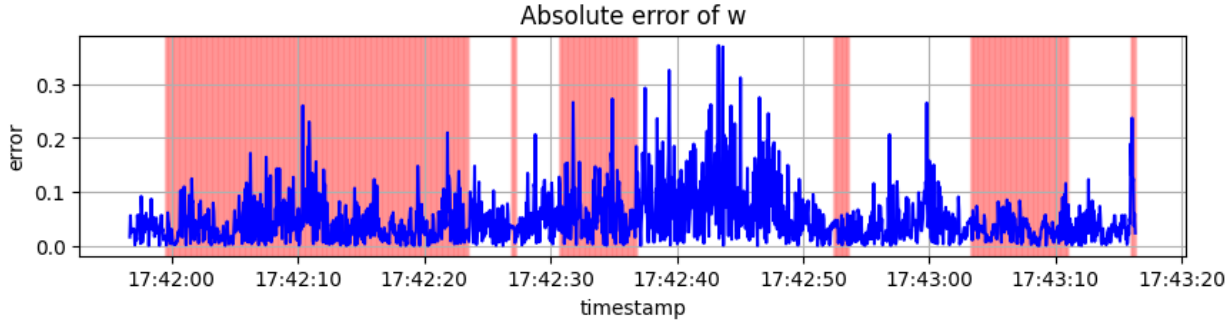
(c)

Fig.60 Univariable Prediction for d_wz

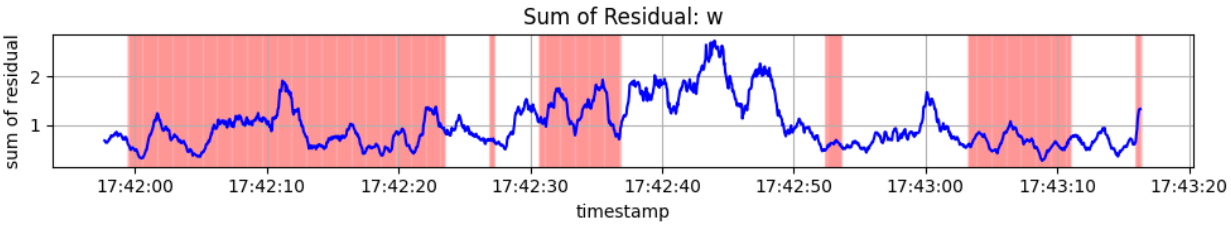
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



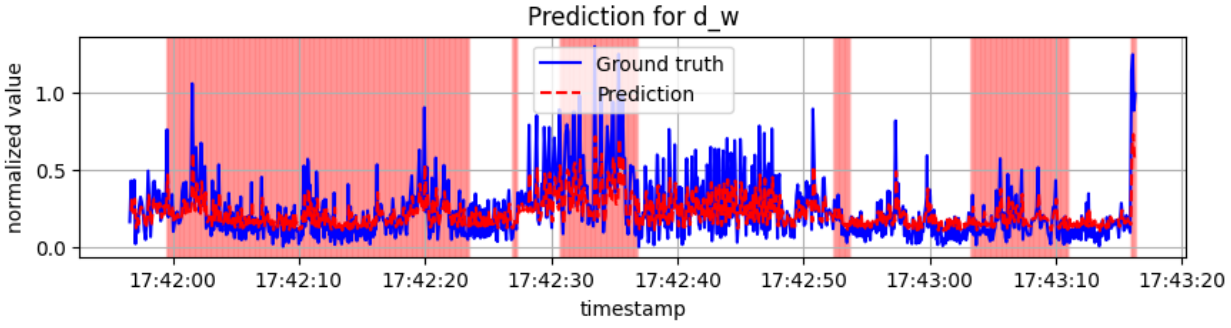
(b)



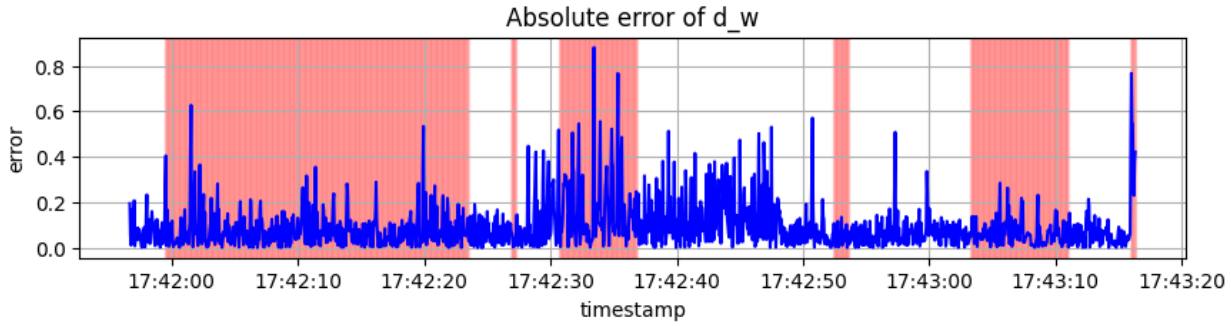
(c)

Fig.61 Univariable Prediction for w

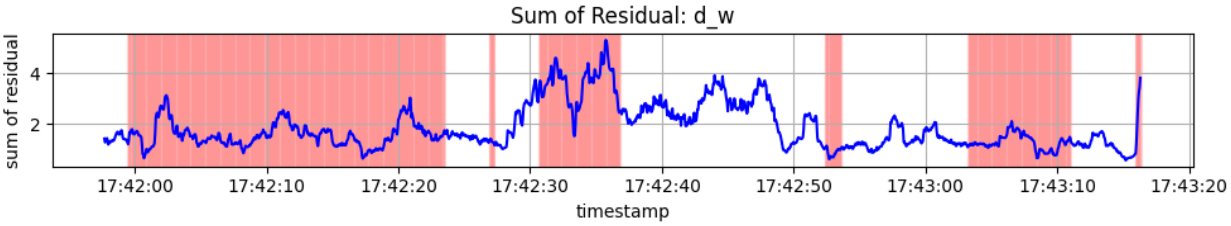
(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals



(a)



(b)



(c)

Fig.62 Univariable Prediction for d_w

(a) Prediction Comparison, (b) Absolute Error, (c) Sum of Residuals