BicycleNet: A Temporal Convolutional Network for Ego Cyclist Trajectory Prediction Using Multi-Modal Bicycle-Mounted Sensors

Master's Thesis by Gijs de Smit

Committee:
Luuk Spreeuwers
Yanqiu Huang
Akhil Pallamreddy
Özlem Durmaz-Incel

June, 2025

Pervasive Systems
University of Twente
Computer Science (Data Science)

***Abstract*—Cycling is a popular and indispensable mode of transportation, but there are many traffic accidents involving cyclists. Predicting cyclist trajectories could prevent accidents by sharing them with surrounding traffic for timely warnings and interventions. However, bicycles pose challenges due to their resource constraints, and existing studies classified future maneuvers rather than predicting trajectories, lack systematic comparisons of sensor placement and modality, and used experimentally constrained datasets. This paper introduces two custom lightweight multi-modal models for cyclist trajectory prediction: BicycleNet, a Temporal Convolutional Network (TCN), and a Convolutional Long Short-Term Memory Neural Network (CNN-LSTM) model. The models are evaluated on unseen cyclists and new locations using two large, realistic datasets comprising sensor data from 63 participants, including four inertial measurement units (IMUs) mounted on the helmet, handlebar, frame, and pedal, as well as GPS and a forward-facing camera. BicycleNet achieves an average distance error of 1.86 m and a final distance error of 3.50 m over a prediction horizon of 5 seconds. Both models achieved similar prediction accuracy, while BicycleNet uses 3.5× fewer parameters than the CNN-LSTM. To determine an optimal sensing configuration, an analysis into sensing modality and IMU placement was carried out, which revealed that the best single IMU placement is on the pedal, closely followed by the frame and handlebar, while the helmet performed worst. Combining all four IMUs with GPS provides the best overall performance. Lastly, the results show promise for deployment in real-world collision avoidance applications, with a relative positional error of 9–17% at 5 seconds into the future.**

***Index Terms*—Ego Trajectory Prediction, Cycling, IMU, GPS, Camera, Deep Learning, TCN, CNN-LSTM**

I. Introduction

Cycling is a popular mode of transportation in Europe due to its affordability, sustainability, and health benefits. However, over the past decade, the number of cyclist fatalities in crashes remained constantly high, while the number of fatalities of other modes of transport has decreased between 10% and 40% [1]. Consequently, it is important to improve cycling safety in traffic.

Passive protection measures such as helmets and airbags [2] reduce the risk of serious injuries, but many cyclists choose not to wear them, and they cannot prevent accidents from occurring. In contrast, active protection measures such as collision avoidance systems have been proposed as a solution to improve traffic safety [3]. In order to prevent crashes, these systems rely on predicting the future trajectories of the ego vehicle and nearby traffic participants. Consequently, trajectory prediction has become an important and substantial area of research. This prediction task has been studied for motorized vehicles in structured environments like highways [4], [5], with more recent literature focused on handling VRUs in urban environments [6]. These trajectory prediction systems are particularly important for autonomous vehicle (AV) path planning, for which there are many proven methods to predict ego and other traffic participant trajectories in real-time [7]. AVs are equipped with advanced sensors, such as radar, LiDAR, and cameras to monitor the behavior of surrounding traffic. These sensors require a line of sight to measure the motion of other traffic participants, such as bicycles, which can be obstructed and result in noisy data. Another popular sensing approach for prediction setups is to use stationary sensors, such as cameras, to observe a piece of road or intersection [8], [9]. However, stationary setups also face the same indirect sensing inaccuracies as AVs.

A more accurate way to measure the motion of cyclists for trajectory prediction is to place sensors on the bicycle or cyclist. These bicycle-mounted sensors can measure more subtle and intricate motion features that can be future ego path indicators, such as hand gestures, head rotation [10], paddle movement, and countersteering [11]. Ego bicycle trajectories can then be communicated via vehicle-to-vehicle (V2V) communication to surrounding vehicles, such as AVs, to prevent collisions. However, it remains a challenge to transfer the existing trajectory prediction methods for AVs and the necessary hardware to bicycles due to their limited capacity to carry resource-intensive sensing and computing technology.

Existing predictive cycling methods that used bicycle-mounted sensors were limited in several key ways: they used a single or a small number of sensors, trained and evaluated on experimentally constrained datasets, rely on access to explicit maneuver labels, and/or classified a small set of future maneuvers rather than predicting trajectories. This work aims to fill the research gap by proposing two lightweight trajectory prediction models, designed to integrate multi-modal data from GPS, a front-facing camera, and IMUs mounted on the handlebar, frame, pedal, and helmet. The models are trained and evaluated on two separate large datasets, collected by the Pervasive Systems research group from a total of 63 participants that cover routes through real traffic. The main contributions are as follows.

- **Multi-Modal Ego Cyclist Trajectory Prediction:** This work proposes two novel models for predicting future ego cyclist trajectories from multi-modal inputs including IMU, camera, and GPS data. The first model, BicycleNet, is a TCN and the second is a CNN-LSTM architecture. Both models do not require future maneuver labels in the input as needed by existing work.
- **Sensing Configuration Analysis:** Evaluation of how IMU sensor placement and sensing modality affect trajectory prediction, identifying the optimal configuration for performance.
- **Realistic Situation Investigation:** Deployment and validation on a large realistic dataset collected in urban traffic environments, ensuring generalizability to different cyclists and locations.

The remainder of the paper is structured as follows. Section II reviews related works that deal with both bicycle-mounted sensors and deep learning methods. Section III outlines the methodology. Section IV describes the data collection, including the sensing hardware and datasets. The experimental results are presented in Section V. Section VI discusses the limitations of the study. Finally, Section VII concludes the paper.

TABLE I: Comparison of studies on ego cyclist activity recognition, maneuver prediction and trajectory prediction that use bicycle-mounted sensors and deep learning.

Study	Sensors	Task	Dataset	Model	Results
BikeSense [12]	IMUs on frame, handlebar, pedals, pocket, and backpack	Activity recognition	Small, 140 m straight road, experimentally controlled	CNN-LSTM	0.97 F1-score
HeadMon [10]	Helmet IMU	Maneuver prediction	Large, campus and city traffic, 20 participants	CNN-LSTM	>85% precision, 4s in advance
RideGuard [13]	Handlebar IMU	Maneuver prediction	Large, campus and city traffic, 20 participants	CNN-LSTM	94% precision, 5s in advance
[11]	IMUs on handlebar and frame	Maneuver prediction	Small, single artificial intersection, 20 participants, experimentally controlled	CNN-LSTM	0.72 F1-score, 0.5s in advance
[14]	Front camera, unspecified IMU, turn signal indicator	Conditional trajectory prediction	Medium, 5 hours of cycling, single participant, city traffic	FCN + CNN, conditional on maneuver label	~ 3.5 m final distance error[1] ($H = 5$ s)
BicycleNet (this work)	Front camera; IMUs on helmet, pedal, frame, handlebar; GPS	Trajectory prediction	Large, campus and city traffic, 63 participants	TCN + ResNet	1.86 m average distance error, 3.50 m final distance error ($H = 5$ s), when using IMU & GPS only

II. Literature Review

This section reviews the existing literature. It begins by comparing trajectory prediction and maneuver prediction, then motivates the use of deep learning and introduces two state-of-the-art model architectures. Next, it discusses limitations in sensor setups and dataset realism of related work. A complete overview of cycling studies involving deep learning and bicycle-mounted sensors is provided in Table I.

Most existing predictive cycling research has focused on maneuver prediction, rather than trajectory prediction, but there is a key distinction between these two tasks. Maneuver prediction, as in [10], [11], and [13], outputs one of several predefined classes such as 'left turn' or 'straight', whereas trajectory prediction outputs a time window of future spatial points. The advantage of trajectory prediction over maneuver prediction is that the extent and timing of the maneuver are apparent from the future spatial points. This helps to detect potential dangers such as collisions more precisely. One study has addressed ego cyclist trajectory prediction using a conditional imitation learning (CIL) approach [14]. Their model is given the future maneuver class as input, obtained from a turning indicator, and uses class-specific output branches. This dependence on predefined intent limits the model's applicability in real-world scenarios where the cyclist's intent is not explicitly indicated (e.g. when no turning indicator is used). For that reason, this work will focus on direct trajectory prediction without CIL, to capture both the timing and extent of cyclist maneuvers, and to make predictions without relying on future maneuver labels.

A recent survey on trajectory prediction methods for autonomous vehicles [7] highlights that deep learning models outperform traditional physics-based and probabilistic models for trajectory prediction, especially in complex scenarios. This shows the promise of deep learning for ego cyclist trajectory prediction, but deploying such models on bicycles is challenging due to resource constraints, which are an important consideration in model design. One model architecture that has shown its capability for the cycling maneuver prediction task is CNN-LSTM, which is a combination of the convolutional neural network (CNN) and long short-term memory (LSTM) layer, such as the one used in HeadMon [10]. The main benefit of this architecture is that it is quite computationally inexpensive, referring to the use of convolutional layers. However, the architecture also contains LSTM and attention layers, which form a bottleneck in terms of computational efficiency. This architecture has also been adopted in several other studies [11], [12], [13], which is why a custom CNN-LSTM architecture is included as one of the models in this work. Moreover, a competitor model to CNN-LSTM is the Temporal Convolutional Network (TCN), which is a type of CNN that uses causal 1D convolutions to capture long-term time dependencies. TCNs offer similar (and sometimes better) performance with lower memory usage and faster inference [15] [16], making them especially suitable for resource-constrained setups such as bicycles. For that reason, the second custom model included in this work, called BicycleNet, uses a TCN-based architecture.

To gain insight into cyclists, recent research has investigated the deployment of inertial measurement units (IMUs) on bicycles. An IMU combines an accelerometer and a gyroscope to measure acceleration and angular velocity, each in three dimensions. IMUs offer the benefit of being light-weight, low-cost, and energy-efficient and are widely used for human activity recognition [17], which implies their potential in a cycling context. Table I shows that IMUs have been placed on the frame, handlebar, pedals, and helmet, to classify behaviors (such as braking, coasting, or turning [12]) and predict maneuvers [10], [11], [13]. One study incorporated a multimodal setup, combining an IMU with a camera and a turn indicator [14] for CIL-based trajectory prediction. These studies demonstrate that bicycle-mounted sensors can be effective, but none of the studies systematically compare sensor placements

[1]The FDE was not explicitly reported in [14], instead it was estimated from four maneuver-specific line graphs showing the distance error of their best model over the prediction horizon.

and modalities. One contribution of this work is a systematic comparison of sensor placement and modality to identify an optimal sensing configuration.

Furthermore, data realism and model evaluation strategies vary strongly across related work. Ranging from controlled or single-location environments [11], [12] without traffic interactions, to realistic routes through urban traffic with many participants [10], [13], as detailed in Table I. Adding to that, the training and testing set split is typically performed based on participant id (as in [11]) or location (as in [14]), or, in some cases, left unspecified (as in [10], [13]). These approaches risk spatial and/or participant leakage, making it difficult to assess how well models generalize to unseen locations and new cyclists, which is crucial for real-world deployment. For that reason, this work uses two datasets with a total of 63 participants, each recorded on a different route with a different set of participants. This enables an evaluation of the model with both a location-based and participant-based train-test split, ensuring realistic performance assessment without data leakage.

In summary, several key limitations in existing research persist. First, maneuver prediction has been extensively explored, but trajectory prediction remains understudied, with only one prior work [14] addressing it under the restrictive condition of future maneuver labels. Second, sensor placements and modalities have not been systematically compared, which is important to determine an optimal sensing setup for resource-constrained bicycles. Third, most studies rely on experimentally constrained datasets or use train-test splits that leak location or participant information, limiting real-world applicability. This work addresses these gaps by proposing two novel models that perform multi-modal unconditional trajectory prediction, one based on the TCN architecture and one based on the CNN-LSTM architecture. The models are validated on realistic datasets with an analysis to optimize sensor configuration to enable efficient cyclist collision avoidance.

III. Methodology

This section describes the methodology for ego cyclist trajectory prediction, which is based on deep learning. The data processing pipeline is presented, which covers model input and output design, IMU feature extraction, normalization, data augmentation, and detailed model architectures.

A. Model Input & Output

The model takes as input multi-modal sensor data from GPS, n IMUs (with $n \geq 1$), and a forward-facing RGB camera, all mounted on a bicycle. These inputs are combined into a time window $(t_p - W, t_p]$ of duration W seconds at the current point in time t_p. The configuration of a time window and its parameters can be seen in Figure 1. A window contains time-series data from GPS and IMU collected over the past W seconds, and a single camera image at t_p. More precisely, it consists of $W_i = W \cdot f_{imu}$ IMU timesteps and $W_{gp} = W \cdot f_{gps}$ GPS timesteps of the past trajectory, where

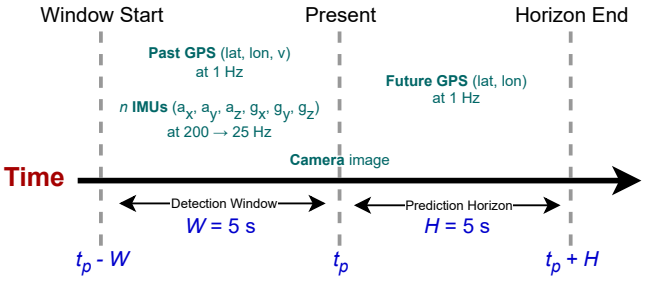


Fig. 1: Time window parameters.

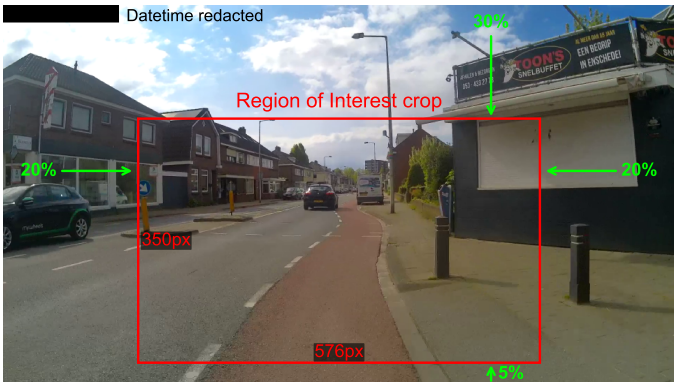


Fig. 2: Image region of interest crop example.

f_{imu} and f_{gps} denote the sampling rates of the IMU and GPS sensors, respectively. A combined IMU object is created by stacking the dimensions of n IMUs, each recording 6 dimensions $(a_x, a_y, a_z, g_x, g_y, g_z)$, having shape $(6n, W_i)$. A GPS object is created which contains the latitude, longitude, and velocity of the past trajectory, which is of shape $(3, W_{gp})$. The IMU and GPS objects, along with the image object, are passed to different input branches of the model, as shown later in Section III-E.

For a time window, only a single image is captured at t_p. This is less computationally expensive than processing multiple frames, and the most recent frame contains the most relevant information about the future path. Including additional frames in a time window might be redundant as the dynamics of the bicycle are already captured by the IMU and GPS. Subsequently, the image from the camera is scaled and cropped to retain the most important information for predicting the future trajectory, mainly the road layout, while filtering out parts such as the sky and nearby buildings that do not add any value. The image is scaled to 540×960 and cropped by removing 20% from both the left and right sides, 30% from the top, and 5% from the bottom. An illustration of this cropping procedure is shown in Fig. 2. After cropping, an image object is left with shape $(3, 350, 576)$, corresponding to the three RGB channels and spatial dimensions.

The ground truth corresponding to a time window is the future GPS trajectory during the prediction horizon $(t_p, t_p+H]$ of duration H seconds. This trajectory contains only latitude and longitude (not velocity), resulting in $W_{gf} = H \cdot f_{gps}$

future GPS timesteps. The resulting ground-truth trajectory therefore has shape $(2, W_{gf})$. Moreover, IMU data is not used for positional ground truth in this work, because the urban dataset (see Section IV-B) only contains accelerometer (with the effect of gravity) and gyroscope measurements, without any reference to absolute position. Estimating the position from IMU requires double integration, which leads to significant drift over time that cannot be corrected without reference sources such as compass readings.

B. Feature Extraction: IMU Integration

As an optional step, the change in angle and velocity over a time window can be included as additional features. These quantities indicate how much the bicycle has rotated or accelerated in the last few seconds, which might help the model better capture overall movement patterns. From visual inspection of plots, these changes often provide a clearer sense of trajectory than raw accelerometer and gyroscope signals, especially when trying to understand maneuvers such as turns or braking. Using Equation 1, the change in angle $\Delta\theta_i$ around axis i is computed as the integral of angular velocity g_i over time.

$$\Delta\theta_i = \int_{t_p-W}^{t_p} g_i \; dt \tag{1}$$

When the bicycle is moving, the angular velocity measured by the gyroscope reflects the bicycle's rotational movement, which is 0 when stationary. In the case of acceleration, if effect of gravity is also measured, it needs to be removed before integrating. The average acceleration $\bar{a}_i$ is estimated over the entire recording to approximate the gravity component. Using Equation 2, the change in velocity Δv_i along axis i is then computed as:

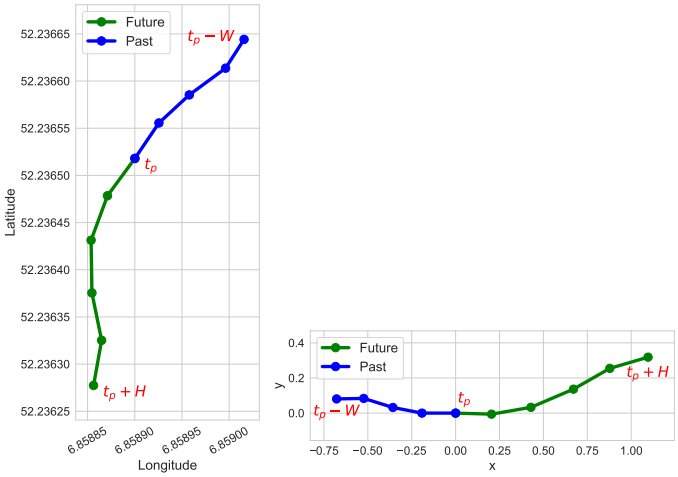
$$\Delta v_i = \int_{t_p-W}^{t_p} (a_i - \bar{a}_i) \; dt \tag{2}$$

The original IMU object of shape $(6n, W_i)$ can then be extended by stacking these features to obtain an updated IMU object of shape $(12n, W_i)$.

C. Normalization

This subsection describes how a time window and its ground truth are normalized for deep learning, ensuring that all input features contribute equally during training and improving model convergence. Firstly, all raw IMU dimensions are individually normalized using z-score (see Equation 3), with the standard deviation σ estimated across the entire training dataset and the mean μ estimated from the specific recording to which the time window belongs, to account for slight IMU orientation differences that affect the gravity vector. For the angle and velocity features derived from the IMUs, μ and σ are computed after generating all time windows, since these features depend on the chosen window size W. Also, the velocity feature of the past GPS data is normalized using z-score, with μ and σ estimated across the whole training set.

$$z = \frac{x - \mu}{\sigma} \tag{3}$$



(a) Raw trajectory (b) Normalized trajectory

Fig. 3: Normalization of past and future GPS trajectories of a time window.

Secondly, images are normalized using z-score, where each of the three RGB channels is normalized independently using the mean and standard deviation estimated across the entire training set. Third, the normalization of the past and future GPS trajectories (latitude, longitude) consists of several steps. A time window's combined past and future GPS data is first converted from geographic coordinates to metric UTM coordinates. The combined trajectories are centered at the origin by subtracting the current point in time t_p, which is the last point of the past trajectory. The current heading is estimated as the angle between the last two points of the past trajectory and used to rotate the coordinate system so that the cyclist's current heading aligns with the x-axis. Finally, the scaling factor s (see Equation 4) is applied so that the magnitude of the trajectory is normalized based on the average cycling speed v_{avg} (in m/s), the number of points, and the GPS sampling rate f_{gps}. The result of the trajectory normalization strategy is shown in Figure 3.

$$s = \frac{1}{\left(v_{avg} \cdot \frac{W_{gp}+W_{gf}}{f_{gps}}\right)} \tag{4}$$

D. Data Augmentation

To improve the generalization and robustness of the model, data augmentation is applied during training. The augmentations are applied probabilistically to normalized data upon batch retrieval. The following list shows the data augmentations applied specific to each sensing modality.

- **GPS**: Random scaling is applied to both the past and future GPS trajectories with a probability of 0.4 and a scaling range of $\pm 8\%$, and Gaussian noise (standard deviation 0.01) is added to the past GPS trajectory with a probability of 0.4.
- **IMU**: All IMU data (including integrated signals) are scaled with a probability of 0.4 and a scaling range of

$\pm 8\%$. Additionally, Gaussian noise (standard deviation 0.08) is added to each IMU signal with a probability of 0.5.

- **Image**: Gaussian noise with a standard deviation randomly sampled between 0.04 and 0.11 is added with a probability of 0.5. Images are also randomly resized (with padding or cropping to maintain resolution) with a probability of 0.4 and range of $\pm 10\%$.

E. Trajectory Prediction Deep Learning Models

Two custom models are proposed: one based on the temporal convolutional network architecture, called BicycleNet, and another based on a CNN-LSTM architecture. The models are compared in terms of predictive performance and computational efficiency in the experimental results section.

1) CNN-LSTM: The CNN-LSTM architecture has shown its capability in predictive cycling studies, as instances of this architecture are used in [10], [11], [12], and [13]. Unlike those studies, this work designs an architecture to support trajectory prediction and to handle multiple input modalities. The CNN-LSTM model designed by this work can be seen in Figure 4. Each input modality has their own branch. The IMU branch is exactly the model of HeadMon [10]. It consists of two 1D convolutional layers (Conv1D) with kernel size 7. The first layer has 16 channels, and the second has 4 channels. Each convolution is followed by a ReLU activation and a 1D maxpooling operation with kernel size 2. The features are then passed through an LSTM layer with 64 hidden units. An attention layer is applied to the LSTM output, followed by a dropout layer with a rate of 0.5. The IMU branch output is produced by a dense layer that maps the 64-dimensional input to an output of size $2H = 10$. In addition, the GPS branch is a smaller CNN-LSTM model. It starts with a 1D convolutional layer with kernel size 3 and 5 output channels, followed by a ReLU activation. The features are passed through an LSTM with 16 hidden units. The last time step of the LSTM output is taken and passed through a dropout layer with a rate of 0.3. Finally, a dense layer maps the output to the desired size $2H$. The camera path consists of an untrained ResNet-18 model [18] whose final dense layer is removed and replaced by a small fully connected network. ResNet-18 was chosen for its lightweight design, suitability for resource-constrained setups, and it experimentally showed the deepest and earliest training convergence compared to other popular image backbones. ResNet's 512-dimensional output is passed through a dropout layer ($p = 0.2$), followed by a dense layer reducing it to 256 dimensions, another dropout layer ($p = 0.2$), and finally a dense layer that projects to $2H$. After that, the output of the three different modality branches is concatenated along one dimension and passed through a fully connected network consisting of three layers with 64, 32, and $2H$ neurons, respectively. The output is then reshaped, after which the predicted trajectory is obtained, having the shape $(H, 2)$.

2) BicycleNet: BicycleNet is based primarily on the Temporal Convolutional Network (TCN) architecture [15], and also includes Time-Distributed Dense (TDD) layers and ResNet-18. The TCN architecture was chosen for its low memory usage and fast inference, as it avoids bottleneck layers such as LSTMs and attention mechanisms, making it ideal for real-time use on resource-constrained devices like bicycles. The model architecture of BicycleNet is shown in Figure 5. If no activation function is specified, then the linear activation function is used. The IMU and GPS data are each passed through a separate TCN model, which consists of residual blocks that apply 1D causal convolutions. A residual block can be seen in Fig. 6, which is taken directly from [15]. A block is specified by the following parameters: kernel size, number of channels in/out, dropout, and dilation. In the Figure of the BicycleNet architecture, the configuration of residual blocks is specified in each TCN rectangle. For example, the TCN that processes the IMU data consists of 5 residual blocks, all having 10 channels. In each TCN, the dilation rate is set to 2^l, where l is the block index starting at 0, which means that the dilation doubles each block. Furthermore, an important concept is the receptive field, which is defined as the range of input visible by a single neuron at the output of a TCN. This is directly affected by the number of layers, kernel size, and dilation. For a constant kernel size k and dilation rate 2^l, the receptive field R can be calculated using Equation 5. The receptive field of the IMU branch TCN is 125 timesteps matching W_i and the receptive field of the GPS branch TCN is 7 timesteps encapsulating W_{gf}.

$$R = 1 + (k-1) \sum_{l=0}^{L-1} \times 2^l \tag{5}$$

Moreover, TDD layers are used to extract features individually at each timestep by applying a dense layer across the channel dimension. A TDD layer does not mix information across timesteps, only across channels/features within each timestep. For each TDD layer, the number of channels in (C_{in}) and out (C_{out}) is indicated in Figure 5. Together, the TCN residual blocks and TDD layers maintain the temporal structure and sequence length of the time-series data in the first half of the model. This allows the model to reason on temporal structure, which is a key aspect of this architecture. The last step of the GPS branch is a linear interpolation from W_{gp} to W_i, after which the IMU and GPS channels are concatenated and passed to a small fusion network. The 'IMU & GPS Fusion' network consist of three TDD and one dense layer. The TDD layers here combine the channels of the two modalities, reducing eventually to only 2 channels for x and y position. The dense layer reduces the window size W_i to the prediction horizon of $H = 5s$. The camera path is identical to that used in the CNN-LSTM model, where a single image is passed through an untrained ResNet-18 whose final dense layer is replaced by a small fully connected network with dropout that outputs $2H$ features. The image branch output is then concatenated with the output of the fusion network. The last step is a fully connected network, after which the predicted trajectory is obtained, having shape $(H, 2)$.

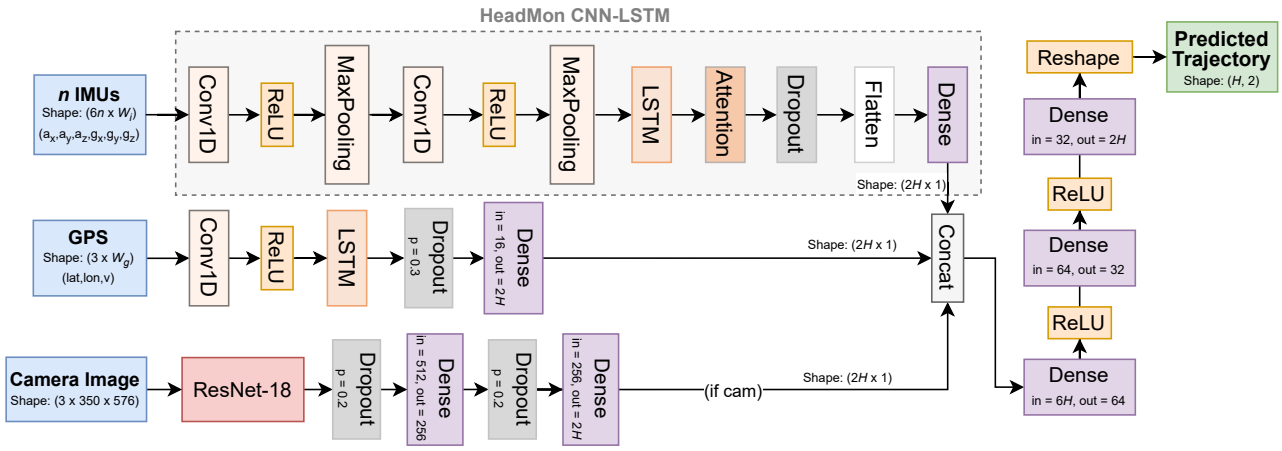


Fig. 4: Custom CNN-LSTM model for trajectory prediction with multi-modal input. The feature extraction of the IMU and GPS mainly consist of 1D non-causal convolutions and LSTM. The IMU input branch is the model from HeadMon [10]. Camera input is processed through ResNet-18 [18].

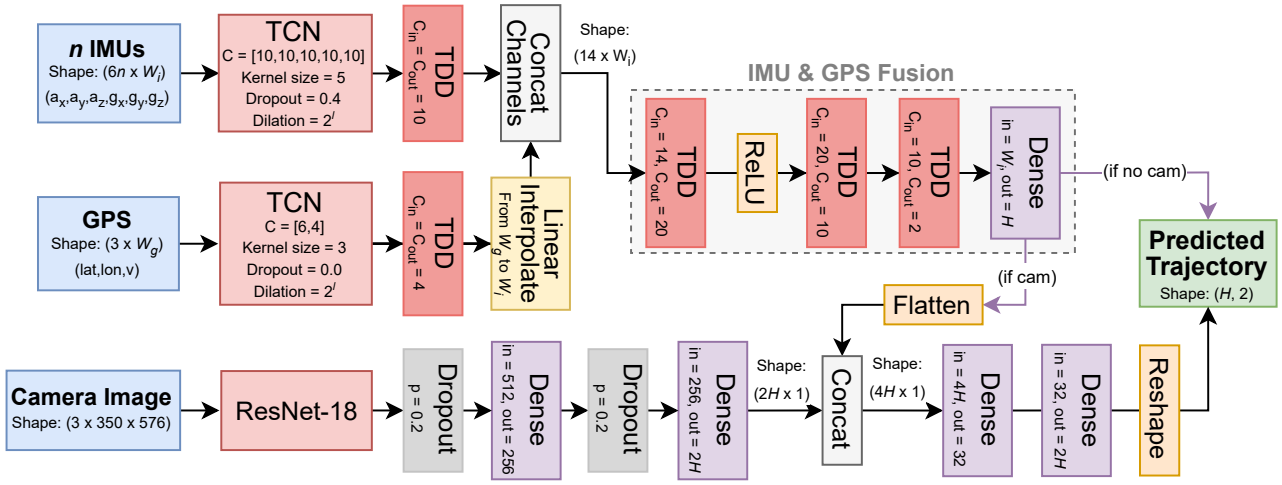


Fig. 5: Custom trajectory prediction model with multi-modal input, referred to as BicycleNet. Features from the IMU and GPS input are extracted using temporal convolutional networks, which consist only of residual blocks (see Figure 6). Time-distributed dense layers are used to combine the IMU and GPS modalities. Camera input is processed through ResNet-18 [18].

IV. Data Collection

This section describes the datasets that were used to evaluate the methods. Two separate rounds of data collection have been conducted by the research group Pervasive Systems of the University of Twente, each having the same hardware sensing configuration. Subsection IV-B describes dataset 1, the urban dataset, which was used to train and validate the model. Subsection IV-C describes dataset 2, the campus dataset, which was used to test the model.

A. Sensor Hardware on the Bicycle

To collect data from participants, a helmet and a medium sized women's electric city bicycle were equipped with a camera, a GPS, and four IMUs. The mounting orientation of each sensor can be seen in Figure 7. A Nokia C32 smartphone (P) was mounted on the handlebar to record latitude, longitude, and speed of the GPS at a sampling rate of $f_{gps} = 1$ Hz. For dataset 1, the phone was running a custom application to record the GPS variables, while for dataset 2, the application 'Phyphox' was used. Also, an Akamduman Action Camera (C) was mounted facing forward on the handlebar to record the cyclist's point of view at a resolution of 1920×1080 and 30 frames per second. In addition, Inertia ProMove Mini inertial measurement units were placed on the handlebar (I_{HB}), frame (I_F), pedal (I_P), and helmet (I_{HM}) each recording three-axis accelerometer and three-axis gyroscope data at a sampling rate of 200 Hz.

All devices were manually time-synchronized by this work, because each device has its own internal clock. The four IMUs were automatically synced via an Inertia Gateway device and therefore share a common clock. Synchronization between GPS, IMU, and camera was achieved by performing a left–right–left shake of the handlebar at the beginning of each recording. The motion of the shake exhibits distinct features,

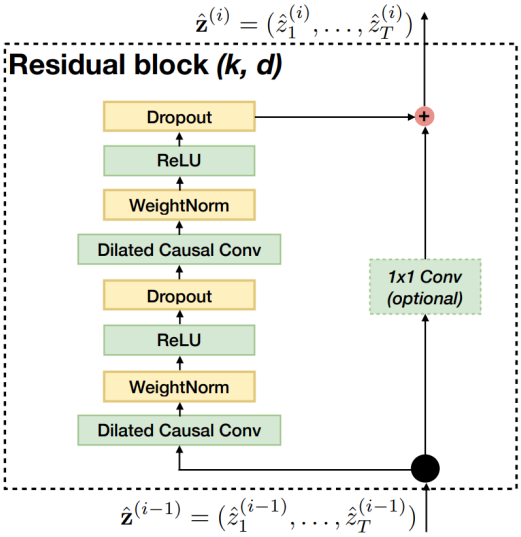


Fig. 6: A residual block from the TCN architecture. This Figure is taken directly from [15]. The parameters of one residual block are: kernel size, number of channels in/out, dropout, and dilation.

Fig. 7: The bicycle and helmet that were used for data collection equipped with camera (C), phone (P), and four IMUs (I_{HM}, I_{HB}, I_F, I_P). The mounting orientation of the IMUs and their individual coordinate systems are denoted using vectors.

such as a start and end point, and clearly defined maximum left and right orientations, which appear in all sensors and can be used to align their timelines. Specifically, the shake was visible in the handlebar IMU (I_{HB}), the video footage from the front-facing camera (C), and, in the urban dataset, the phone's onboard accelerometer (P). By manually aligning these features across modalities, a common time zero was established. Due to the difference in software used on the phone, a slightly different approach was used for the campus dataset, where GPS heading was aligned with compass readings from I_{HB}. A detailed explanation and visualizations of the time-synchronization process can be found in Appendix A.

Using the aforementioned data collection setup, two separate datasets were collected, each with a different route and set of participants. Before the ride, participants were instructed to follow all standard traffic and safety rules during the experiment. They were not given any information that could influence or bias their natural cycling behavior. The bicycle saddle and helmet were adjusted to each participant's comfort. Although there was no imposed speed limit during the ride, the built-in motor assistance of the e-bike was limited to 25 km/h. Participants had the option of using motorized assistance, but some chose not to use it.

B. Dataset 1: Urban (Training & Validation Sets)

The collection of this dataset was conducted by members of the Smart Connected Bikes project of the Pervasive Systems research group of the University of Twente [19]. Three instrumented bicycles were developed following the sensor configuration presented in the previous subsection. Data collection was carried out with 42 participants cycling in an urban environment during April and May 2024. Each participant used one of the three instrumented bicycles. Participants ranged from 20 to 80 years of age, included both males and females, and had varying levels of cycling experience ranging from a few months to several years. The recordings were made between 10 am and 6 pm and each participant cycled the route shown in Fig. 8 between one and three times in a row, depending on their desired level of participation. The route is 4.0 km long, covers a diverse range of traffic scenarios, and takes about 12 minutes on average to complete. To navigate the route, a second phone was mounted on the handlebar which displayed the route in the application 'Komoot'. The route starts on the university campus, passes through a park, crosses Hengelosestraat (a major arterial road) and continues alongside it, passes through urban living and shopping areas in the district Twekkelerveld, and finally returns to the university campus. This route consists of 16 unique turns and many intersections. In total, the route was traversed 94 times, resulting in 18 hours and 36 minutes of cycling data. The dataset is split into the training and validation set using an 80/20 ratio, based on participant id.

C. Dataset 2: Campus (Testing Set)

A second round of data collection was conducted with 21 new participants using only one instance of the bicycle on the University of Twente campus in May 2024 by I. Kaniščev [20]. The participants consisted of males and females who were all students, which means a narrower distribution in both age range and level of experience compared to dataset 1. Each

Fig. 8: Route of the urban dataset, which is divided into the training and validation set. The route starts and ends on the University of Twente campus in Enschede and passes through the district Twekkelerveld. Each participant cycled this route up to three times clockwise.

Fig. 9: Route of the campus dataset, which is used as the testing set. Each participant cycled the rectangular lap twice counterclockwise and twice clockwise. The route is fully located within the University of Twente campus.

participant cycled the rectangle shape route shown in Figure 9, which is located fully within the university campus. The route first consists of two laps counterclockwise, a u-turn, and finally two laps clockwise, resulting in a total distance of 4.6 km containing a total of 16 turns. No map was displayed and participants were told to memorize the route, so that they did not have to look down to the handlebar for navigation. The start, u-turn, and finish points are all at the same location. In addition, the route contains fewer intersections and less traffic compared to the urban dataset. On average, it took participants 12 minutes to cycle the whole course, making a total of 4 hours and 9 minutes of cycling data, which makes up the testing set.

V. Experimental Results

This section presents the experimental setup and results. First, the training procedure and evaluation method are described. Then, the results are shown for the sensing configuration analysis in terms of IMU placement, IMU feature extraction, and sensing modality. Finally, BicycleNet and the CNN-LSTM are compared in terms of predictive performance and suitability for real-time use.

A. Training Setup & Parameters

This subsection describes the parameters that define the model input and output as well as the setup that was used to train the deep learning models. First of all, the window size W is fixed to 5 seconds, as it captures early maneuver preparation while offering a good balance between performance and computational resources, as shown in [10] and [13]. The prediction horizon H is also set to 5 seconds, as motivated in [14]. This allows enough time for downstream assistance systems to react (e.g., a 4-second warning [21]) while still maintaining high accuracy and avoiding the uncertainty of longer horizons. The sampling rate of the IMUs (f_{imu}) is sampled at 25 Hz, as it offers a good trade-off between preserving relevant motion information, suppressing high-frequency road noise, and maintaining computational efficiency, as shown in [10] and [13]. A GPS sampling rate (f_{gps}) of 1 Hz is used, reflecting the typical rate available on smartphones such as the one used in the datasets.

Following that, the 42 participants from the urban dataset (see Section IV-B) were randomly split (depending on the seed) into 33 for training and 9 for validation, following an approximate ratio of 80/20. The 21 participants from the campus dataset (see Section IV-C) were exclusively used for the testing set. To extract samples from the recordings, the sliding window method was employed with an overlap of 3 seconds (60%) between consecutive windows. The samples within each set were randomly shuffled. In total, there are $33,042$ combined training and validation samples, and $7,382$ testing samples. Secondly, the optimization of the hyperparameters and selection of the loss function were performed through experimentation where the performance metrics on the validation set were evaluated. Several loss functions were

experimented with, these include: mean squared error (MSE) loss, velocity loss, cosine similarity tangent loss, and their various combinations. The loss selected from experimentation is the MSE loss, as it was found to have the smoothest training, best/earliest convergence, and lowest distance error on the validation set. In addition, the Adam optimizer was used with an initial learning rate of 0.001 and the batch size was set to 80. The models were trained for 45 epochs, corresponding to the point at which the validation loss converged. For each training run, the model with the lowest validation loss across the epochs is selected for evaluation on the testing set. Following that, the seed was set for all random operations (excluding data augmentation). Doing so allows for comparable experiments, as for example the model weights are initialized in the same way and the same validation set is used in each run. Runs are repeated 12 times with different seeds to add significance to the results.

Training was conducted on a single compute node from the high-performance cluster (HPC) of the University of Twente due to the large size of the dataset. The compute node consists of an NVIDIA Tesla L40 GPU with 48 GB of dedicated video memory (VRAM) and 256 GB of system memory (RAM). The computational bottleneck is image data, which is significantly larger compared to IMU and GPS data. To optimize storage and to fit into system memory, the images were stored in float16 format, while all other data remained in float32. Since the original images are 8-bit integers, converting them to float16 introduces negligible loss in precision compared to float32. The full dataset is approximately 170 GB, which would have been nearly twice as large if the images had been stored in float32. For faster training, the entire dataset is loaded into the system memory of the node, as lazy loading was found to be considerably slower in practice.

B. Model Evaluation

This subsection describes the distance-based metrics used to assess the model's trajectory prediction performance. Given an actual and predicted future trajectory, the average distance error (ADE) measures the mean Euclidean distance between the two trajectories across the prediction horizon H, as shown in Equation 6. This metric captures how well the model tracks the overall shape of the trajectory.

$$\text{ADE} = \frac{1}{H} \sum_{t=t_p}^{t_p+H} |\hat{p}_t - p_t| \tag{6}$$

Here, $\hat{p}_t$ is the predicted position and p_t the ground truth position at time step t. ADE is then aggregated across all samples in the test set to compute the average performance. Additionally, the final distance error (FDE) is computed, which considers only the last point of the predicted trajectory. This is shown in Equation 7.

$$\text{FDE} = |\hat{p}_{t_p+H} - p_{t_p+H}| \tag{7}$$

FDE is especially important in scenarios where the final position is more critical than the intermediate steps. For example, when the model is used for decision-making or interaction planning, where reaching the right endpoint matters most. Both ADE and FDE are reported in meters and strongly depend on the prediction horizon H. A longer prediction horizon typically results in higher errors due to the compounding error in motion. Lower ADE and FDE correspond to more accurate model predictions.

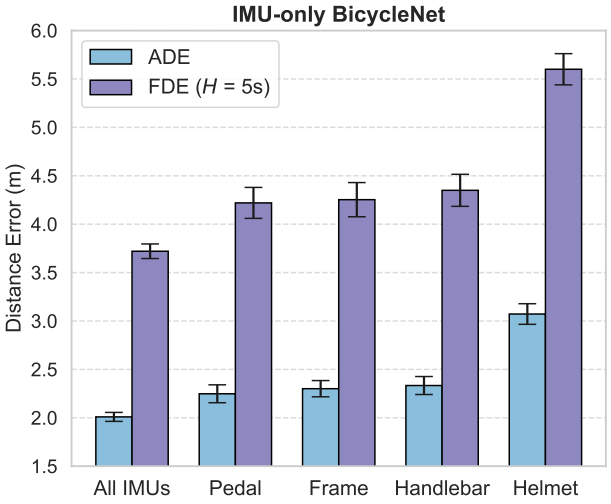


Fig. 10: Comparison of the IMU sensor placements. BicycleNet with only the IMU input branch was trained on different IMU inputs. The average distance error (ADE) and final distance error (FDE) over a horizon of $5s$ are reported on the testing set.

C. IMU Placement

To measure the contribution of each IMU sensor placement, the BicycleNet model with only the IMU input branch was trained on different IMU input configurations. The configurations that were evaluated include each IMU individually and all four IMUs combined. The training process was repeated 12 times with different random seeds for each configuration and the results were averaged. Recall that changing the random seed affects both the training/validation split and the weight initialization. ADE and FDE were computed on the test set, and the error bars indicate the standard deviation between the runs.

Figure 10 shows the result of the IMU comparison experiment. It can be seen that there is a similar pattern for ADE and FDE across different input configurations, which makes sense as they are correlated metrics. The three single IMU configurations of pedal, frame, and handlebar offer similar performance. There is no significant single best IMU placement, but an IMU placed only on the pedal offers the lowest ADE of 2.25 $\pm$ 0.09 m. Using the pedal IMU might improve model predictions for stopping and slowing-down situations, which involve longitudinal motion. Following that, the frame and handlebar IMUs potentially improve the model predictions in leaning and steering maneuvers, which involve lateral motion. Furthermore, the IMU on the helmet is the

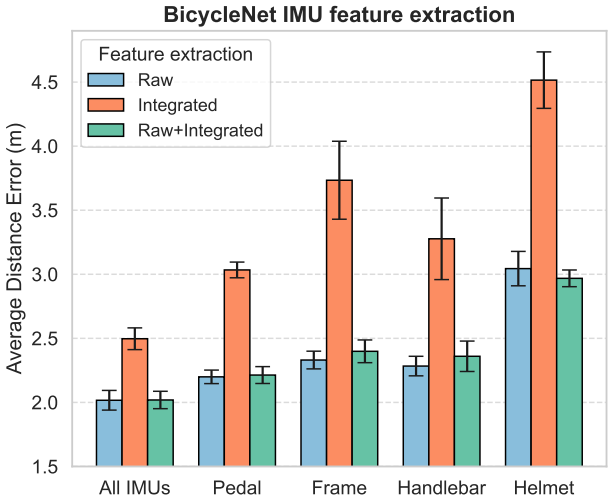


Fig. 11: Feature extraction of the different IMUs. BicycleNet with only the IMU input branch was trained on different IMU inputs with and without feature extraction obtained from integration (velocity and angle). The average distance error (ADE) over a horizon of $5s$ is reported on the testing set.

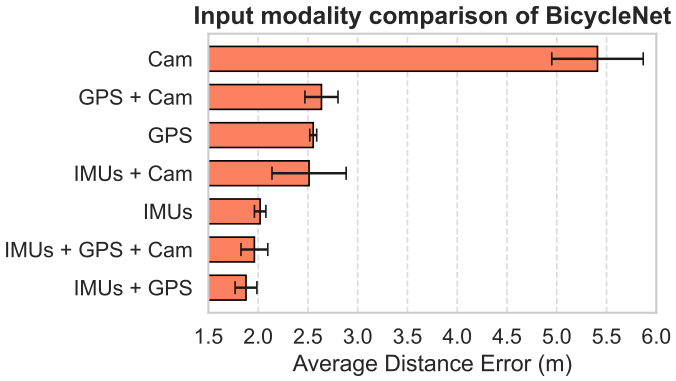


Fig. 12: Comparison of different sensing modality configurations of BicycleNet. The average distance error (ADE) over a horizon of $5s$ is reported on the testing set.

least accurate, having a relatively high ADE of 3.07 $\pm$ 0.11 m. This might be caused by cyclists looking around while cycling straight and not entering a turning maneuver (a false positive turn is detected), which can occur around intersections. The effectiveness of the helmet IMU might become more apparent when combined with other IMU placements, which can prevent this type of trajectory uncertainty. For example, if the cyclist is looking around but is still pedaling, it likely indicates that they are continuing straight rather than initiating a turning maneuver. Lastly, combining all IMUs provides the best performance with an ADE of 2.01 $\pm$ 0.05 m. Each IMU placement captures different aspects of the cyclist's behavior, such as pedaling activity, steering dynamics, and head gaze. The model is able to extract features across IMU placements (inter-IMU features), increasing the accuracy of the prediction.

D. IMU Feature Extraction

The IMU placement comparison experiment is expanded by including the angle and velocity features extracted from the IMUs (using integration as described in Section III-B). For each individual IMU and for all IMUs combined, three configurations are compared: raw IMU signals, integrated features, and a combination of both. Figure 11 shows the ADE on the testing set, averaged over 12 runs of the IMU-only BicycleNet model, where error bars represent the standard deviation across runs. The plot shares the light blue bars with the previous experiment (Figure 10), which corresponds to the ADE of the raw IMU configurations. Like before, the pedal is the best single IMU placement overall, achieving the lowest ADE in each of the three categories: raw, integrated, and raw+integrated. From the orange bars, it is derived that only using integrated features as model input shows an increase of roughly 30% or more in ADE compared to only using raw IMU data. Moreover, for single IMU setups except for the helmet, the green bars show that raw+integrated IMU features show a slight increase in ADE compared to using only the raw IMU data. However, this is not the case when combining all IMUs, where raw and raw+integrated features do not show any significant difference in performance. This suggests that individual IMU sensor setups do not benefit from the feature extraction of angle and velocity, as raw+integrated data generally results in worse performance compared to raw data alone. Also, for the setup using multiple IMUs, the raw+integrated data do not provide a noticeable improvement over raw data alone. For the final configuration of the model, the raw data from all IMUs are used without additional feature extraction.

E. Sensing Modality

In this subsection, a performative comparison is made between the different sensing modalities of IMU, GPS, and camera. Multiple training runs were conducted with BicycleNet for each possible combination of the three modality input branches. Figure 12 shows the ADE on the testing set, averaged over 12 runs of the BicycleNet model trained on different input configurations, where error bars represent the standard deviation across runs. The best performing combination of modalities is IMUs+GPS, achieving an ADE of 1.88 $\pm$ 0.11 m, showing a clear improvement over using IMUs or GPS individually. GPS captures global movement trends and provides a broad situational context, such as the beginning or end of a turn, and gradual speed changes. On the other hand, IMU data captures more fine-grained motion patterns, such as the pre-maneuver indicators described in AppendixB. Together, the modalities give a comprehensive view of a cyclist's motion pattern, resulting in a strong model performance. Following that, GPS alone performs worse than IMU alone, indicating that while GPS provides useful global context, it lacks fine-grained motion details. Adding the camera to IMU and GPS (IMUs+GPS+Cam) does not improve performance (1.89 $\pm$ 0.06 m), suggesting that image input does not contribute to what is already captured by the motion sensors. In fact,

combining the camera with just the IMUs (IMUs+Cam) shows a significant drop in performance compared to the IMUs alone, showing that the image input adds uncertainty and may confuse the model. Moreover, the camera alone performs the worst (5.45 $\pm$ 0.48 m), with a high ADE and a large variance. These findings suggest that the image branch of the model may fail to extract general road layout information. The lack of performance when using camera input is further discussed in Section VI.

F. BicycleNet vs. CNN-LSTM

In this subsection, the BicycleNet model is compared to the CNN-LSTM model in terms of trajectory prediction performance and computational efficiency. For comparison, both models were trained without the image branch, meaning the input is just IMUs+GPS, without additional feature extraction.

Figure 13 shows the distance error across the prediction horizon for both models. It can be seen that BicycleNet achieves an FDE of 3.50 $\pm$ 0.07 m, and CNN-LSTM reaches 3.55 $\pm$ 0.05 m. BicycleNet achieves a slightly (but not significantly) lower FDE. This indicates that the causal convolutional (TCN-based) BicycleNet performs as well as the CNN-LSTM model, which combines convolutional layers with LSTM. Furthermore, a distance error of 3.5 m at 5 seconds into the future is quite promising for collision avoidance applications. Considering that cyclists typically travel at 15–27 km/h (covering roughly 20–38 m in 5 seconds), a 3.5 m deviation corresponds to a relative error of approximately 9–17%. Combined with a conservative 1-second processing time, a 4-second advance warning to the cyclist could yield a potential safety benefit of up to 98%, as demonstrated in [21].

Interestingly, the BicycleNet model performs on par with another study [14] that relies on conditional imitation learning (CIL), which assumes access to maneuver labels through a turning signal indicator. That study was also able to improve model performance by using an image captured by the front-facing camera. Their (IMU+Cam) CIL model achieves an FDE of 3.5 m. In contrast, the model presented here predicts without access to maneuver labels and without using the camera and still achieves a comparable FDE of 3.5 m, suggesting that the model successfully anticipates cyclist turns.

1) Timing and Memory Benchmarks: This paragraph describes the experiment performed to measure real-time suitability of the models. Table II shows the results of the experiment. The mean inference time over 200 forward passes of a model is reported in milliseconds (ms), along with the throughput as the number of samples that can be predicted per second and the VRAM usage of the GPU. The inference batch size is set to 1, as this is likely to be used for real-world deployment on a single bicycle. From Table II it is seen that both BicycleNet and the CNN-LSTM model are lightweight and suitable for real-time use on edge devices. When using only the IMU and GPS branches, the models are extremely small. BicycleNet with 7,816 parameters is 3.5 times smaller than the CNN-LSTM model with 27,237 parameters, resulting in less memory usage. The inference timings are very low on the high-end GPU for both models, especially for CNN-LSTM which achieves 0.93 ms per forward pass, compared to 1.74 ms for BicycleNet. This difference is due to BicycleNet having more layers, resulting in more sequential processing. In short, BicycleNet achieves comparable performance to that of the CNN-LSTM model while using 3.5$\times$ less parameters.

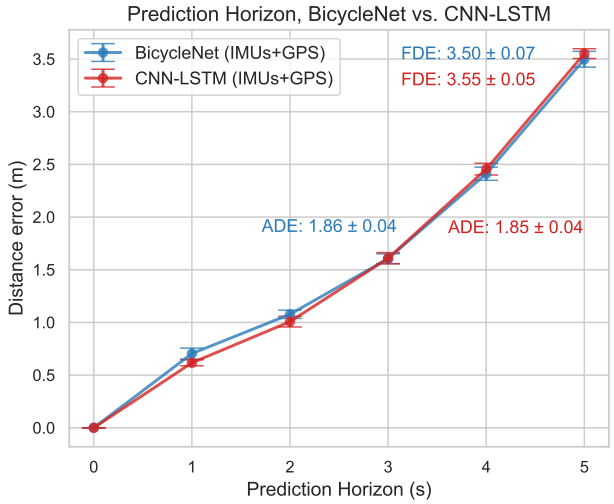


Fig. 13: BicycleNet and CNN-LSTM models testing set distance error across the prediction horizon when trained the modalities IMUs+GPS (without additional feature extraction). The average distance error (ADE) and final distance error (FDE) are denoted in the plot.

VI. Discussion

The effectiveness of BicycleNet and the custom CNN-LSTM model was demonstrated in the results section. However, some limitations were identified. This section discusses the study's limitations in detail and describes potential improvements.

First, the helmet-mounted IMU performed worse than the other IMU placements in terms of trajectory prediction performance, which contrasts with the findings of HeadMon [10], which demonstrated the effectiveness of the helmet IMU for maneuver prediction. This discrepancy might be attributed to cyclists looking around without initiating a turning maneuver, differences between the task of predicting maneuvers (classification) and predicting trajectories (regression), as well as differences in dataset composition in terms of class balancing. This research project did not balance the different types of maneuver to train the trajectory prediction models and instead included all available samples. In contrast, HeadMon [10] used a subset of samples for the 'cycling straight' class, potentially missing cases where the cyclist looks around while cycling straight. The training route consists of many intersections and other traffic scenarios where looking around is crucial for cycling safety, even when cycling straight. This could explain the lack of performance of the helmet IMU, since the cyclist may look around without initiating a maneuver. Moreover, the participants in the training set had to look at a map displayed

TABLE II: Inference timings and memory usage of the trajectory prediction models. The batch size is set to 1 and data processing, transformation, and initialization steps are excluded from the timings. The utilized GPU for this experiment is the 'NVIDIA RTX 4070 SUPER'. The mean of 200 forward passes is reported with standard deviation.

Model	**Modalities (input branches)**	**#Parameters**	**Inference Time (ms) (mean ± std)**	**Throughput (samples per second)**	**VRAM Usage (MB): model + remaining**
BicycleNet	IMU, GPS	7,816	1.74 ± 0.08	549	0.03 + 9.30
CNN-LSTM	IMU, GPS	27,237	0.93 ± 0.31	1208	0.10 + 16.43

on the phone to navigate the route, whereas this was not the case for participants in the testing set. Frequent looking down at the phone could introduce noise, thereby decreasing the performance of the helmet IMU.

Second, the camera input underperformed, which could be due to either the architecture of the image branch or the nature of the data. Despite testing several CNN backbones for the image branch, no configuration improved over the ResNet model. This suggests a limitation in the extraction of generalizable features from the images. This is in contrast to previous work [14], which reported improvements when using the camera image for conditional trajectory prediction. One potential cause is that the camera's pitch angle (vertical tilt) was inconsistent across participants and bicycles due to imperfect mounting and road vibrations. There was roughly a $\pm 20°$ variability in the height of the horizon in an image. As a result, the road layout appears inconsistently across training samples, with parts sometimes occluded. Another potential cause is overfitting to the training route. Additionally, many scenarios include cycling behind other cyclists, interacting with traffic, unexpected u-turns, or even cycling on the sidewalk to avoid a road blockage. These scenarios make it difficult to extract meaningful clues for the future trajectory from the image. A potential improvement is the inclusion of road segmentation, lane detection, and/or object detection as a preprocessing step for a trajectory prediction model. In this way, existing generalized knowledge of traffic images can be transferred to the trajectory prediction model to prevent overfitting to route-specific features. In addition, dynamic understanding of the past trajectory could be improved by including multiple frames in the model input stage and adapting the model camera branch accordingly.

Third, several limitations are identified that stem from the data processing pipeline. One limitation is that time synchronization between the different sensors (IMU, GPS, and camera) was performed manually as a post-processing step after data collection, based on aligning measurements of a handlebar shake. This method can introduce a random misalignment error of up to ± 500 ms that is different for every recording, which could limit the model's ability to learn time-correlated features across modalities. While this may not critically affect the GPS (1 Hz) or the camera (single frame), it could still introduce noise. In future work, a more robust solution would be to perform (automatic) real-time synchronization before a recording is started. Following that, the method of obtaining the angle and velocity has the limitation of being relative to the starting point of a time window. This means that the model lacks information about the absolute velocity and angle of the cyclist, for example the absolute angle of the pedal IMU is unknown. A potential improvement is to obtain the absolute angle and velocity during data collection by recording the compass and quaternions of the IMUs. Additionally, the acceleration was recorded with the effect of gravity included, which could be removed in future setups to isolate the cyclist's motion. Furthermore, the current system is constrained to predicting at 1 Hz, due to the GPS sampling rate. Also, GPS itself can be an inaccurate source of positional ground truth, as it is subject to random errors of up to several meters that might limit the achievable model performance. A more accurate approach is to use localization methods of higher frequency and better accuracy, such as position tracking using visual-inertial odometry or high-precision RTK-GNSS systems.

VII. Conclusion

This paper introduced a novel cyclist trajectory prediction method that incorporates multi-modal input from GPS, IMUs, and a camera. Two custom models were proposed in this work: BicycleNet, based on the temporal convolutional architecture, and a CNN-LSTM model. The models were trained and evaluated on a large and diverse dataset collected from multiple participants cycling in a real urban environment. When configured with only IMU and GPS input branches, both models achieved comparable performance, with BicycleNet requiring less memory because it has 3.5x fewer parameters. BicycleNet achieved an average distance error of 1.86 m and a final distance error of 3.50 m over a prediction horizon of 5 s. This level of accuracy was shown to be promising for collision avoidance applications, due to a relative positional error of 9–17% at 5 seconds into the future. An analysis of IMU sensor placement showed that an IMU placed on the pedal achieved the best single-IMU trajectory prediction performance, closely followed by the frame and handlebar placements, while the helmet-mounted IMU performed the worst. The helmet IMU was likely less reliable due to frequent head movements that are not associated with future maneuvers. Combining multiple IMU placements enabled the model to extract inter-IMU features, improving trajectory prediction performance. Moreover, including GPS sensor data of the past trajectory along with the combined IMU data showed the best performance compared to all the other possible combinations of modalities. Also, the camera input consistently did not improve prediction performance. Lastly, the models

are lightweight and fast enough for real-time inference on resource-constrained setups such as bicycles.

A. Future Works

In real-world traffic scenarios, blind spots and the sudden appearance of unexpected objects are common. Understanding the trajectory intentions of surrounding vehicles can be crucial to help prevent potential collisions. One promising application lies in Vehicle-to-Everything (V2X) communication for bicycles [22], which enables efficient short-range communication. Using V2X, a cyclist's predicted trajectory can be broadcast in real time to nearby vehicles, allowing them to respond appropriately or issue warnings to their drivers. For instance, when a collision is predicted, a safety mechanism such as automatic braking via an e-bike's motor could prevent it. In cases where a collision is unavoidable, a deployable helmet airbag could be activated to minimize injury.

Beyond that, advanced bicycle-mounted sensing systems like the one presented in this work could also be used to assess rider behavior and condition. For example, indicators of fatigue, intoxication, stress, or inexperienced riding could be detected and used to tailor real-time riding assistance or restrict bicycle operation.

Finally, future work could explicitly detect pre-maneuver indicators, such as head checking, countersteering, and stop pedaling (as discussed in Appendix B), and integrate them into a model to improve trajectory prediction performance.

References

[1] M. Mellauner, M. Fleischer, M. Donabauer, and A. Soteropoulos, "Facts and figures: Cyclists," tech. rep., European Road Safety Observatory, Directorate General for Transport, Brussels, Belgium, June 2024.

[2] J. Woo, S.-H. Jo, G.-S. Byun, B.-S. Kwon, and J.-H. Jeong, "Wearable airbag system for real-time bicycle rider accident recognition by orthogonal convolutional neural network (o-cnn) model," *Electronics*, vol. 10, June 2021.

[3] M. Hamidaoui, M. Z. Talhaoui, M. Li, M. A. Midoun, S. Haouassi, D. E. Mekkaoui, A. Smaili, A. Cherraf, and F. Z. Benyoub, "Survey of autonomous vehicles' collision avoidance algorithms," *Sensors*, vol. 25, no. 2, p. 395, 2025.

[4] Y. Hu, W. Zhan, and M. Tomizuka, "A framework for probabilistic generic traffic scene prediction," 2018.

[5] R. Toledo-Moreo and M. A. Zamora-Izquierdo, "Imm-based lane-change prediction in highways with low-cost gps/ins," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 180–185, 2009.

[6] E. Schuetz and F. B. Flohr, "A review of trajectory prediction methods for the vulnerable road user," *Robotics*, vol. 13, no. 1, 2024.

[7] V. Bharilya and N. Kumar, "Machine learning for autonomous vehicle's trajectory prediction: A comprehensive survey, challenges, and future research directions," *Vehicular Communications*, vol. 46, p. 100733, April 2024.

[8] J. Li, Y. Ni, and J. Sun, "A two-layer integrated model for cyclist trajectory prediction considering multiple interactions with the environment," *Transportation Research Part C: Emerging Technologies*, vol. 155, p. 104304, October 2023.

[9] Z. Liu, N. Zhong, J. Chen, and B. Gao, "A modeling method for two-dimensional two-wheeler driving behavior during severe conflict interaction at intersections," *Accident Analysis & Prevention*, vol. 205, p. 107668, September 2024.

[10] Z. Han, L. Xu, X. Dong, Y. Nishiyama, and K. Sezaki, "Headmon: Head dynamics enabled riding maneuver prediction," in *2023 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, PerCom '23, pp. 22–31, IEEE, 2023.

[11] G. d. Smit, D. Yeleshetty, P. J. Havinga, and Y. Huang, "Predicting turn maneuvers of cyclists using bicycle-mounted imu with cnn-lstm," in *2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, PerCom Workshops '24, pp. 587–592, IEEE, 2024.

[12] L. Mathuseck, J. Götz, L. Busch, and K. David, "Bikesense: Riding behaviour recognition using an instrumented bicycle," in *2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pp. 1–6, 2024.

[13] Z. Han, X. Dong, L. Xu, Z. Zhu, E. Wang, Y. Nishiyama, and K. Sezaki, "Rideguard: Micro-mobility steering maneuver prediction with smartphones," in *2024 IEEE 44th International Conference on Distributed Computing Systems (ICDCS)*, ICDCS '24, pp. 1039–1049, IEEE, 2024.

[14] A. Weißmann and D. Görges, "An empirical study on ego vehicle trajectory prediction for bicycles in urban environments based on conditional imitation learning," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, ITSC '21, pp. 1482–1489, IEEE, 2021.

[15] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018.

[16] Y. Li, L. Xiao, H. Wei, D. Li, and X. Li, "A comparative study of lstm and temporal convolutional network models for semisubmersible platform wave runup prediction," *Journal of Offshore Mechanics and Arctic Engineering*, vol. 147, no. 1, 2025. Cited by: 1.

[17] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019. Deep Learning for Pattern Recognition.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[19] "Smart connected bikes field trials featured in u-today." https://www.utwente.nl/en/eemcs/ps/news/2024/4/1458699/smart-connected-bikes-field-trials-featured-in-u-today, Apr. 2024. Accessed: 2025-06-12.

[20] I. Kaniščev, "Cyclist maneuvers prediction with bicycle-mounted imus," master's thesis, University of Twente, Enschede, The Netherlands, July 2024.

[21] L. Schories, N. Dahringer, U. Piram, A. Raut, S. Nikolaou, I. Gragkopoulos, I. Tsetsinas, and M. Panou, "Safety performance assessment via virtual simulation of v2x warning triggers to cyclists with models created from real-world testing," *Sustainability*, vol. 16, p. 610, Jan. 2024.

[22] S. Blanco, "Ces 2021: Bicycles want to get into the connected vehicle game." Online Article, Jan 2021.

[23] J. Fajans, "Steering in bicycles and motorcycles," *American Journal of Physics*, vol. 68, pp. 654–659, 07 2000.

APPENDIX A
TIME SYNC DEVICES: HANDLEBAR SHAKE

This appendix describes how the three sensing modalities of IMU, GPS, and camera were synced in time. The syncing was performed in a manual post-processing manner, because the different sensors were not all connected to a shared device that handles automatic syncing during data collection. The synchronization method for the urban dataset is described, which relies on the motion of a handlebar shake. The synchronization of the campus dataset follows a similar approach, but with one key difference, which is described in the last subsection.

Each sensing modality uses a different clock. The four IMUs were connected to an Inertia Gateway device, which automatically syncs the four IMUs using its clock. GPS data was recorded using the phone, which also had its internal accelerometer enabled (only in the urban dataset). The front-facing camera simply recorded video and also uses its own clock. To achieve synchronization between the devices, a left-right-left shake of the handlebar was performed at the beginning of each recording. The motion of the shake has distinct features that are captured by each device. These features include a start and end point, as well as clearly defined maximum left and right orientations. By marking the device-specific timestamps at which these features occur in each recording, the time offset between the clocks can be calculated as the mean difference between the corresponding points. The time axis of each device is then shifted accordingly, so that the shake event aligns across devices and effectively defines a common time zero. The more distinct timestamps that are labeled for a recording, the more accurate the syncing. The following subsections cover each device individually and show visualizations of the handlebar shake captured by each device.

A. Camera

Figure 14 shows the shake of the handlebar captured by the camera. The camera recorded video at 30 frames per second, allowing for precise timestamp selection at which the handlebar-mounted camera starts and stops moving. The human error in selecting these timestamps is estimated to be at most 3 frames, corresponding to 100 milliseconds.

B. IMU Handlebar

Figure 15a shows the shake of the handlebar recorded by the IMU on the handlebar at 200 Hz. To find the distinct features of the handlebar shake, the accelerometer data from the IMU is plotted, as this is also available on the phone, which did not have a gyroscope enabled. The axis that measures the shake the most is a_y, which corresponds to the left-right movement along the handlebar. The human error due to manual selection of points is estimated to be around 80 ms. The error of syncing is reduced if more distinct points are selected for a single shake.

C. Phone Accelerometer (Urban Dataset)

The phone accelerometer was only enabled during the data collection of the urban dataset. Figure 15b shows the shake of the handlebar recorded by the phone accelerometer at 10 Hz.

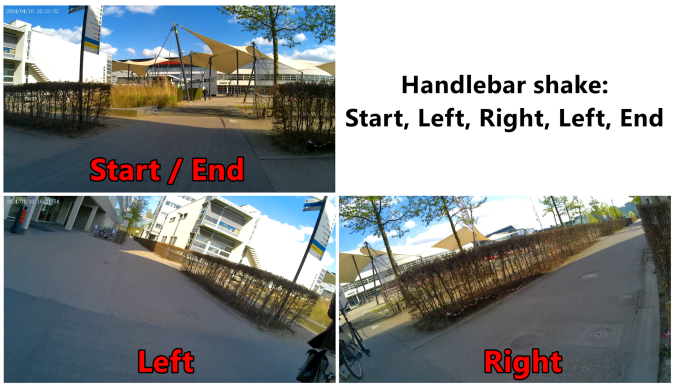
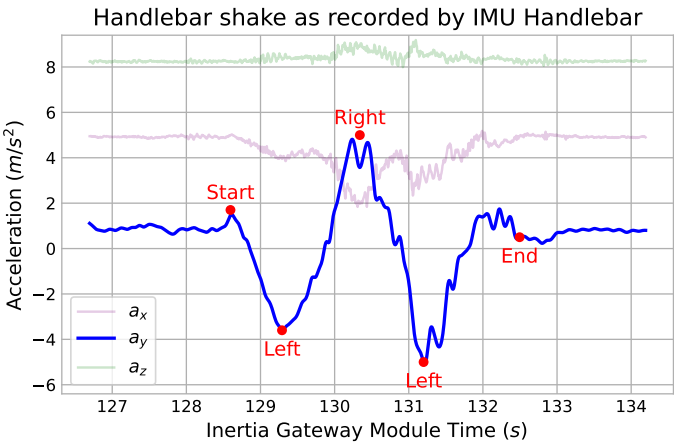
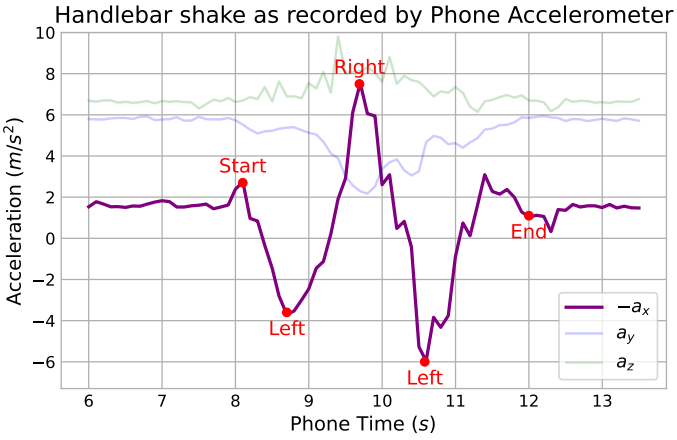


Fig. 14: The handlebar shake for syncing the devices captured by the camera. Three snapshots illustrate key moments in the motion: the start, the leftmost point, the rightmost point, and the end of the shake.



(a) Handlebar shake captured by the IMU on the handlebar at 200 Hz.



(b) Handlebar shake captured by the phone's accelerometer at 10 Hz (urban dataset only).

Fig. 15: The handlebar shake as captured by the IMU and phone accelerometer, used to align device clocks during post-processed synchronization.

The coordinate system used by the phone is different from that of the IMU on the handlebar. The a_y axis of the handlebar IMU roughly aligns with the $-a_x$ axis of the phone. The human error due to manual selection of points is estimated to be around 100 ms.

D. IMU Compass and GPS Heading (Campus Dataset)

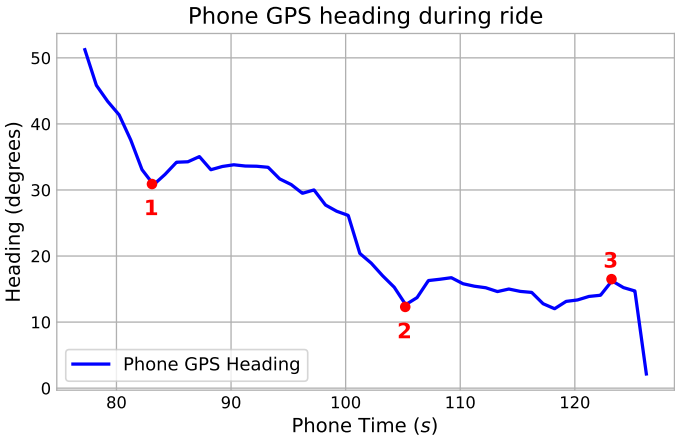
The campus dataset requires a slightly different approach to sync the devices in time. Similarly to the urban dataset, the camera and IMU clocks are aligned by matching the movement of the handlebar shake at the start of a recording. However, only for the data collection of the campus dataset the accelerometer of the phone disabled, but the compass of the IMU was enabled.

To sync the GPS with the other devices, the GPS heading of the phone was aligned with the compass heading of the IMU on the handlebar. The heading of the IMU compass is estimated using Equation 8. This method assumes that the IMU is level with the ground and uses only the horizontal components of the Earth's magnetic field. Specifically, c_x and c_y represent the magnetometer readings along the local horizontal axes (e.g., forward and lateral directions). The heading angle θ indicates the direction in which the sensor is facing relative to magnetic north.

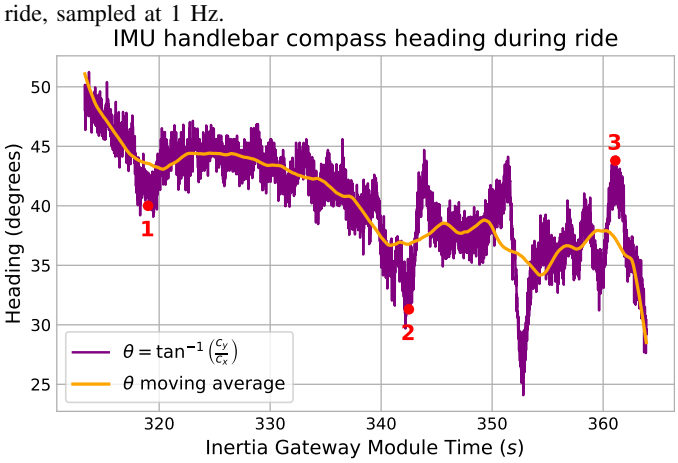
$$\theta = \tan^{-1}\left(\frac{c_y}{c_x}\right) \tag{8}$$

Figure 16 shows the GPS and IMU heading during a 50 second window of the ride. A moving average of the IMU is included to more easily match distinct features between the modalities, because the IMU is sampled at 200 Hz and the GPS is sampled at only 1 Hz. The cyclist is mostly going straight, but makes noticeable changes in his heading, which can be detected in both the IMU and GPS. These events are marked with a red dot and a number. Like in the case of the handlebar shake, these distinct features can then be used to align the clocks of the Inertia Gateway and the phone. Since the Inertia Gateway can still be synced with the camera through the handlebar shake, the syncing process is complete for the campus dataset.

There are some inaccuracies with the method of aligning based on heading. Firstly, in Figure 16 there is a noticeable difference in the scaling on the y-axis between the modalities. This is likely due to the assumption of the IMU to be in a plane tangent to Earth's surface, but it is actually mounted at an incline on a bicycle that moves. In addition, the time window needed to align the IMU and GPS based on heading is much larger than when aligning using the handlebar shake because the GPS sampling rate is 1 Hz. This makes it more difficult to find features that can be precisely matched. To offset this, multiple points are selected along the ride to ensure good syncing. The human error is expected to be somewhere between 200 and 500 ms.



(a) GPS heading measurements during a 50-second segment of the ride, sampled at 1 Hz.



(b) IMU compass heading during the same time window, sampled at 200 Hz and smoothed with a moving average.

Fig. 16: GPS and IMU compass heading signals used for time synchronization in the campus dataset. Distinct heading changes that match in both graphs are marked with numbers. These enable alignment despite differences in sampling rates and sensor mounting.

APPENDIX B
PRE-MANEUVER INDICATORS

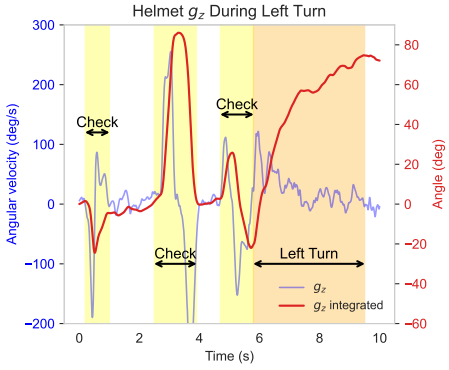
In this appendix, an experiment is described in which pre-maneuver indicators are identified. These indicators serve as explainability for the models by revealing why future trajectories can be predicted using bicycle mounted sensors. It is important for cyclists to plan and anticipate several seconds ahead in order to follow traffic rules and ensure safety in their environment. This behavior is shown through physical cues in the cyclist's motion, referred to as pre-maneuver indicators. There are three types of pre-maneuver indicators that will be showcased here, these include: head checking, countersteering, and stop pedaling. Examples are taken from the urban dataset, described in Section IV-B. To identify indicators, plots of the angular velocity of the Z axis of the gyroscope (g_z) are made of I_{HM}, I_P, and I_{HB}. The angle obtained from integrating g_z using Equation 1 is also shown in the plots.

Figure 17a reveals that cyclists perform head-turning checks prior to turn maneuvers as measured by I_{HM}. This is easy to understand, as it is imperative for a cyclist to observe the surroundings before making a change in trajectory. The duration, extent, and number of these head checks likely depend on the extent of the upcoming turn. In this example, the cyclist performs three head checks, including one where he looks over his shoulder for almost two seconds before entering a relatively long turning maneuver of almost four seconds. In particular, HeadMon [10] uses only an IMU on the helmet for the classification of upcoming maneuvers with 85% accuracy four seconds before the maneuver.

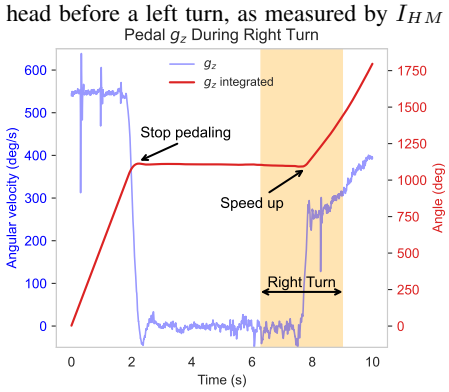
Furthermore, Figure 17b shows that cyclists often stop pedaling before initiating a turn, as captured by I_P. In this example, pedaling stops approximately four seconds before a right turn, with acceleration occurring midway through the turn. This suggests that the action of stopping to pedal can serve as a pre-maneuver indicator. However, this action might not occur before all turns, as gradual or non-intersection turns may not require deceleration.

Another pre-maneuver indicator is countersteering as measured by I_{HB} that is shown in Figure 17c. A countersteer occurs shortly before a turn as a preparatory action. In the example, the cyclist first steers left before entering a right turn. A cyclist consciously or subconsciously performs a countersteer to place their center of mass inside the curve of a turn [23]. The extent can be small or large, depending on the speed through the turn. Notably, [11] uses only an IMU on the handlebar to classify upcoming maneuvers with significant accuracy 0.5 seconds before the maneuver.

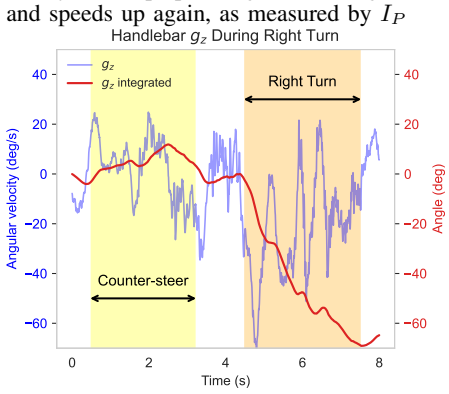
Additionally, the pre-maneuver indicators can appear before a variety of maneuvers, like stopping completely or making a u-turn. The three examples shown here are likely only a subset of a broader range of ways cyclists prepare for the upcoming trajectories. Other pre-maneuver indicators may exist, such as slight braking, small handlebar movements, or shifts in body posture, as mentioned in RideGuard [13]. This motivates the use of deep learning, which can learn to recognize these and even more nuanced pre-maneuver indicators. Indicators may also vary between cyclists, influenced by factors such as experience, age, or individual riding style. Sometimes, a cyclist can show indicators, even when cycling straight, resulting in a wrong indication of the future maneuver (i.e., false positive). To combat this, different indicators can be combined through multiple IMU placements, to reduce uncertainty and improve trajectory prediction performance.



(a) Cyclist performs three checks with the head before a left turn, as measured by I_{HM}



(b) Cyclist stops pedaling before a right turn and speeds up again, as measured by I_P



(c) Cyclist performs a countersteer to prepare for a right turn, as measured by I_{HB}

Fig. 17: Instances of the pre-maneuver indicators: head checking, stop pedaling, and countersteering. Each plot shows the gyroscope z-axis angular velocity from an IMU sensor and the corresponding angle change over time from integration. Each example is performed by a different participant.