ViFi-Loc: Multi-modal Pedestrian Localization using GAN with Camera-Phone Correspondences

Hansi Liu
Rutgers University
hansiiii@winlab.rutgers.edu

Hongsheng Lu
Toyota Motor North America
hongsheng.lu@toyota.com

Kristin Dana
Rutgers University
kristin.dana@rutgers.edu

Marco Gruteser
Rutgers University
gruteser@winlab.rutgers.edu

ABSTRACT

In Smart City and Vehicle-to-Everything (V2X) systems, acquiring pedestrians' accurate locations is crucial to traffic and pedestrian safety. Current systems adopt cameras and wireless sensors to estimate people's locations via sensor fusion. Standard fusion algorithms, however, become inapplicable when multi-modal data is not associated. For example, pedestrians are out of the camera field of view, or data from the camera modality is missing. To address this challenge and produce more accurate location estimations for pedestrians, we propose a localization solution based on a Generative Adversarial Network (GAN) architecture. During training, it learns the underlying linkage between pedestrians' camera-phone data correspondences. During inference, it generates refined position estimations based only on pedestrians' phone data that consists of GPS, IMU, and FTM. Results show that our GAN produces 3D coordinates at 1 to 2 meters localization error across 5 different outdoor scenes. We further show that the proposed model supports self-learning. The generated coordinates can be associated with pedestrians' bounding box coordinates to obtain additional camera-phone data correspondences. This allows automatic data collection during inference. Results show that after fine-tuning the GAN model on the expanded dataset, localization accuracy is further improved by up to 26%.

KEYWORDS

Localization; Multi-modal; Computer Vision; WiFi FTM; IMU; GAN

ACM Reference Format:
Hansi Liu, Hongsheng Lu, Kristin Dana, and Marco Gruteser. 2023. ViFi-Loc: Multi-modal Pedestrian Localization using GAN with Camera-Phone Correspondences. In *INTERNATIONAL CONFERENCE ON MULTIMODAL INTERACTION (ICMI '23), October 9–13, 2023, Paris, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3577190.3614119

1 INTRODUCTION

In V2V (vehicle to vehicle) and V2X (vehicle to everything) communities, roadside units (RSU) are becoming more significant as they are deployed at a larger scale. Equipped with sensors such as RGBD cameras and wireless communication devices, RSUs can communicate with traffic participants in the vicinity, provide additional information, and enhance traffic mobility and safety. They are the common building blocks for outdoor edge computing applications ranging from smart city, traffic management, collaborative perception, self-driving, etc. In these applications, acquiring accurate location estimations for vehicles and pedestrians is important. Standard localization solutions typically rely on GPS or other GNSS services. However, their accuracy degrades significantly in complex environments like urban-canyon.

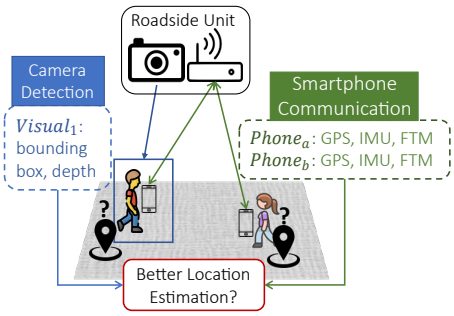

Figure 1: Motivation. The roadside unit collects user participants' multi-modal data using RGBD camera sensing and wireless communication. Can we provide the users with accurate location estimations leveraging the multi-modal data that is not necessarily associated?

To provide a better location service when standard location services can not be obtained, current systems leverage RSU's vision and wireless sensing capabilities, and edge computing resources to estimate and share traffic participants' locations. But the estimated pedestrian locations may not be accurate because each sensing modality has its own limitations. Camera RGBD sensing, while providing accurate depth information of the detected persons, is limited to non-line-of-sight (NLOS) scenarios and suffers from drastic illumination changes. Wireless sensing such as Fine-time-measurement (FTM) [8] is more robust to NLOS conditions, but its ranging performance is degraded by multi-path and shadowing in a complex environment. It is desirable to combine both modalities to achieve better localization for pedestrians. The state-of-the-art multi-modal sensor fusion algorithms usually provide a state estimation that is more accurate than the measurement from every single modality. For example, fusing camera depth measurement

ICMI '23, October 9–13, 2023, Paris, France

ACM ISBN 979-8-4007-0055-2/23/10…$15.00
https://doi.org/10.1145/3577190.3614119

with FTM allows ranging measurement to be more accurate; Fusing GPS with IMU allows fine-grained localization that is robust to sensor noise and drifting.

These standard data fusion approaches, however, are applicable only under the condition that measurements from both modalities are available and data association is known. If a subject's multi-modal data is not correctly associated, the fused measurement would be inaccurate. Moreover, measurements of a pedestrian might not always include both modalities. For a situation depicted in Figure 1, when a person is out of the camera view or the detection algorithm fails to detect her and only phone data is available, it would be infeasible to leverage data from camera modality to improve the performance of wireless ranging and localization. This challenge motivates us to come up with a solution that accurately estimates a person's location using information from both camera and phone modalities while not depending on pre-computed data association.

When a pedestrian's camera data contains bounding box coordinates and phone data includes GPS, IMU measurements, and FTM, we can view the localization task as *refining* or *correcting* a person's raw GPS data using his camera RGBD data, IMU measurements, and FTM. An important intuition lies in the fact that GPS localization error, affected by satellite constellation, tends to be correlated for pedestrians in the vicinity within a period of time [30]. Thus, if we can learn a cross-modal mapping between a group of pedestrians' existing camera-phone data correspondences, we will be able to use the same mapping to localize other pedestrians in the same area, by translating their phone data into local camera spatial coordinates, even if they are out of the camera field of view.

We propose a cross-modal Generative Adversarial Network (GAN) architecture that learns the linkage between a person's camera modal measurement and phone modal measurement. During training, a pedestrian's multi-modal measurements within a time window will be fed into the network. Measurements from phone modality include FTM, IMU data, and smartphone GPS readings; Measurements from camera modality include bounding box centroids coordinates and depth measurements. Although phone measurements and camera measurements are not directly comparable, they both describe and encode the same pedestrian's kinematic information. To reflect this linkage, the network extracts features from input measurements and enforces them to be close to each other in the hyper-space. A decoder is then applied to the feature vector of phone modality to generate a location estimation with respect to the camera coordinate frame. The generated coordinate will be examined by a discriminator to ensure that the produced coordinate is within the distribution of true locations. During inference, the proposed network is capable of generating a person's estimated coordinates with respect to the local camera coordinate frame based on the person's phone measurements. We evaluate our proposed methods on a large-scale real-world dataset and develop a procedure to estimate the RSU's world-camera transformation and comprehensively evaluate our methods' accuracy and ability to generalize.

To facilitate real-world deployment and larger-scale training, we propose a self-learning mechanism that leverages the network output to automatically produce more multi-modal data correspondences. Upon obtaining the estimated locations from the pedestrian's phone measurement, we associate the GAN-produced coordinates with the existing bounding box centroids 3D coordinates. Since each GAN-produced location corresponds to a pedestrian's phone measurement, associating the GAN-produced location with camera modality measurement is essentially acquiring additional camera-phone data correspondences. The semi-supervised approach allows us to easily obtain large-scale reliable training data without dedicated data collection and manual labeling.

To the best of our knowledge, we are the first to apply GAN to generate location measurements. Unlike existing works that use GAN to generate synthesized images [10, 17, 24], texts [16], or WiFi signals [31, 32], we focus on generating 3D locations from wireless ranging measurements, IMU measurements, and camera bounding box coordinates and depth.

Summary of Contributions. As a summary, ViFi-Loc makes the following contributions:

- Designing a GAN architecture that learns GPS correction models of different environments based on users' multi-modal data. During inference it only requires phone modality data to produce accurate location estimations, not depending on pre-computed data association.
- Developing a self-learning mechanism to facilitate real-world deployment and larger-scale data collection and training. The proposed mechanism associates the GAN-produced coordinates with pedestrians' camera coordinates to automatically accumulate additional data during inference.
- Systematically evaluating our proposed methods on a large-scale real-world dataset. We develop a procedure to estimate the RSU's world-camera transformation and comprehensively evaluate our methods' accuracy and ability to generalize.

Artifact Availability: We plan to open-source our code implementation. For review purposes, all the source code can be found in the submitted supplementary materials.

2 BACKGROUND AND RELATED WORK

Vision-based localization There are many related works on pedestrian localization. These works can be categorized based on sensor types and modalities. In the vision domain, localization can be achieved by cameras or other optical sensors such as lidar. Using RGBD or 3D point cloud information and state-of-the-art human detectors, a person's spatial location can be estimated. Off-the-shelf RGBD cameras with person detection and localization functionality such as ZED and RealSense offer depth accuracy of 1% to 9% of the distance from near range to far range within 20 meters [1, 2].

GPS GPS is one of the typical localization solutions. With different chipset configurations and services, they provide a varying range of localization granularity from meter level to sub-centimeter level. GPS-enabled smartphones are typically accurate within a 5-meter radius under open sky [3]. However, their performances are usually degraded by factors including satellite constellation, poor weather conditions, environmental variation, and multipath due to tall buildings, bridges, and trees. Pocket-size GPS receivers with moderate prices offer positioning accuracy around tens of meters [26]. The study in [34] suggests that the observation quality

of Android smartphone GNSS observations are difficult to achieve meter-level accuracy if using only pseudo-range observations. While survey-grade GPS equipment achieves sub-centimeter accuracy, it requires specialized equipment and expensive configurations with extra subscription services such as Real Time Kinematics (RTK) and Real Time Differential [15].

WiFi localization Another major category of studies on localization focuses on WiFi signals. Received Signal Strength Indicator (RSSI) can be used in fingerprinting [13] and trilateration [20]. More recently, WiFi Fine Time Measurements (FTM) [8] has been extensively explored in localization tasks. [18] confirms that the FTM protocol can achieve meter-level accuracy in open space environments although degrades in high multipath environments.

Inertial aided localization Inertial Measurement Units (IMU) are often adopted as auxiliary sensors due to their easy accessibly and cheap prices. IMU provides kinematic information at a higher sample rate than GPS and WiFi message exchange rates. Localization based on IMU dead-reckoning alone, however, suffers from cumulative error in the long term. Standard approaches to reduce cumulative error include filtering techniques that incorporate IMU with GPS and WiFi measurements. For example, [12] fuses WiFi RSSI fingerprinting, GPS, and IMU using an Extended Kalman Filter. Wi-Go [19] fuses WiFi FTM, GPS, and vehicle odometry information using a particle filter and achieves an outdoor vehicular localization error of 1.3 m median.

Multi-modal sensor fusion/association Vision-based localization and wireless-based localization have complementary characteristics. Camera sensing provides more accurate spatial information in the near field through RGBD sensing, but they suffer from occlusion, appearance, and illumination variation; wireless sensing, on the other hand, can work in non-line-of-sight and poor illumination conditions. But its ranging performance can be degraded by complex environments with multi-path and shadow fading. Combining vision and wireless sensing in a localization system has gained more attention recently as it combines both modalities' advantages. Related work in the cross-field includes Simultaneous Localization and Mapping (SLAM), where a mobile agent relies on vision and wireless data to locate itself while creating a representation of the surrounding. SLAM can be achieved using vision only [27], vision+IMU [11, 29], WiFi+IMU [9, 19], etc.. Compared with traditional Filtering approaches such as EKF and particle filters, Bundle Adjustment, pose graph, or factor graph optimization [14, 23] provides better performance on a larger scale.

The above-mentioned sensor fusion approaches have a limitation in that vision and wireless data need to be available and associated at the same time during inference. In our task, however, a person's camera data could be unavailable due to a limited field of view. As a result, typical Kalman filter or SLAM approaches become inapplicable. A novel approach is needed to fully exploit pedestrians' camera data and phone data.

3 MULTI-MODAL LOCATION ESTIMATION

Figure 2 presents an overview of our methodology. The model is first trained with a manually labeled dataset that consists of camera-phone data correspondences of multiple pedestrians. During inference, the network produces location estimations based only on pedestrians' phone data. The produced coordinates are then associated with bounding box coordinates from the camera modality. Then the associated data correspondences are combined with the original dataset to form a larger-scale training set that can be used to further train the network. This feedback loop enables our network to achieve self-learning — using the network's output to produce more training data during inference. Next, we introduce our GAN architecture, how it is trained, and how the generated locations are associated with camera observations to obtain more training data.

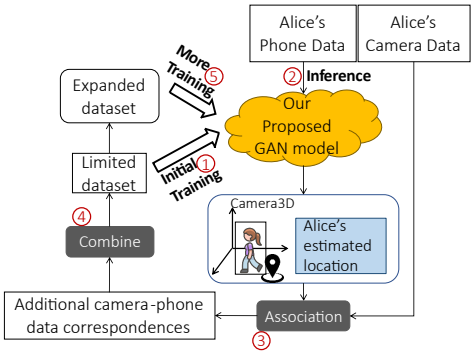

Figure 2: Method overview. A GAN model is trained on an initially limited dataset that contains pedestrians' camera-phone data correspondences. During inference, the model generates a location estimation for the user based only on her phone data. The generated coordinates can be used to associate with camera bounding box coordinates to produce additional camera-phone data correspondences, which allows the dataset to expand automatically. More training and fine-tuning on the expanded dataset improve the model's localization accuracy.

3.1 GAN Architecture

Figure 3 shows the proposed GAN architecture. The network takes as input a pedestrian's sequential multi-modal data within a time window k. For every timestamp t, the vision data $\boldsymbol{v}_t$ takes the form

$$\boldsymbol{v}_t = \begin{bmatrix} d, & x, & y, & X, & Y, & Z \end{bmatrix} \in \mathbb{R}^6, \tag{1}$$

where d is the depth value of the pedestrian's bounding box centroid; $[x, y]$ is the pixel coordinate of the bounding box centroid; $[X, Y, Z]$ is the bounding box centroid's 3D coordinate with respect to the camera coordinate frame.

For wireless data, the input at timestamp t takes the form

$$\boldsymbol{p}_t = \begin{bmatrix} r_{\text{ftm}}, & std_{\text{ftm}}, & Acc, & Gyr, & Mag, & GPS \end{bmatrix} \in \mathbb{R}^{14}. \tag{2}$$

It contains FTM range r_{ftm}, FTM standard deviation std_{ftm}, 9-axis IMU data (accelerometer $[x_{\text{acc}}, y_{\text{acc}}, z_{\text{acc}}]$, gyroscope $[x_{\text{gyr}}, y_{\text{gyr}}, z_{\text{gyr}}]$, and magnetometer $[x_{\text{mag}}, y_{\text{mag}}, z_{\text{mag}}]$) as well as GPS coordinates $[X_{\text{gps}}, Y_{\text{gps}}, Z_{\text{gps}}]$ with respect to the local camera coordinate frame.

The synchronized vision and wireless sequential data are rendered into feature embeddings $\boldsymbol{e}_v$ and $\boldsymbol{e}_p$ by two independent bi-directional LSTM modules. The embeddings contain spatial and temporal cues of the person's camera modality input and phone modality input. We adopt LSTM units as feature extractors for the multi-modal input considering they offer significant advantages

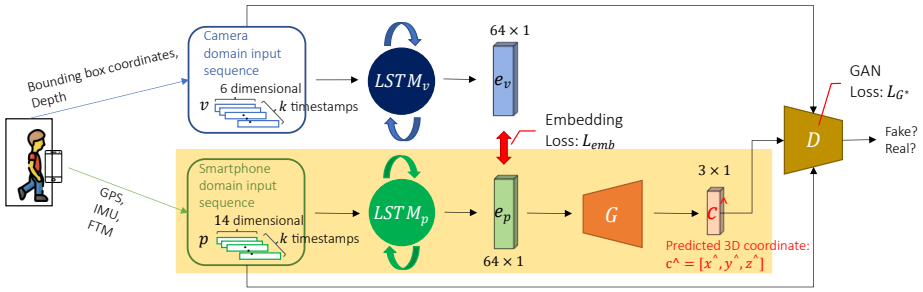

Figure 3: Proposed GAN architecture. The input includes pedestrians' camera-phone data correspondences. Camera domain input consists of the person's bounding box centroid and depth information. Phone domain input contains FTM range, standard deviation, 9-axis IMU, and GPS coordinates w.r.t. local camera 3D coordinate frame. The input goes through two independent bi-directional LSTM units. The output embeddings e_v and e_p encode spatial and temporal cues of the person's camera data and phone data. They are constrained by the embedding loss because they come from the same pedestrian. A Generator $\mathcal{G}$ renders the phone modality embedding e_p into a coordinate $c^\wedge$. A discriminator $\mathcal{D}$ is used to examine whether $c^\wedge$ is genuine or fake. The detailed configurations of $\mathcal{G}$ and $\mathcal{D}$ are listed in Table 1 and 2. Training the network uses data from both modalities. Inference only requires phone data as input, as shown in the yellow-shaded region.

over other vanilla multi-layer network architectures when extracting features from sequential or time-series data. Because these feature vectors represent the same pedestrian, we use the Embedding Loss to force them to be close to each other in the high-dimensional space. The Embedding Loss takes the form $L_{emb} = \|\boldsymbol{e}_v - \boldsymbol{e}_p\|_2$. It ensures the linkage between vision modality and wireless modality.

Next, the wireless modality feature vector $\mathbf{e}_p$ goes through a generator $\mathcal{G}$ that consists of a series of fully connected layers, batch normalization layers, and dropout units. The detailed architecture is listed in Table 1. The generator renders the feature vector $\mathbf{e}_p$ into a coordinate $\boldsymbol{c}^\wedge$ in the camera's 3D local coordinate frame. The generated coordinate and the network input are then examined by a discriminator $\mathcal{D}$ whose detailed architecture is listed in Table 2. $\mathcal{G}$'s purpose is to generate valid coordinates that are within the distribution of true location; $\mathcal{D}$'s purpose is to stringently discriminate or examine if a generated coordinate is good enough, i.e., within the distribution of the pedestrian's true coordinates. The output of the discriminator $\mathcal{D}$ is 0 or 1 indicating whether the examined coordinate is unrealistic (fake) or realistic (true). We use the Generative loss to train the generator and discriminator. The Generative Loss L_{G^*} takes the form

$$L_{G^*} = \min_{\mathcal{G}} \max_{\mathcal{D}} L_{LSGAN}(\mathcal{G}, \mathcal{D}) + g(\boldsymbol{c}_{\text{gnd}}, \boldsymbol{c}^\wedge), \tag{3}$$

where $L_{LSGAN}(.)$ is the standard Least Squares GAN Loss [25]

$$L_{LSGAN}(\mathcal{G}, \mathcal{D}) = \mathbb{E}[(\mathcal{D}(\boldsymbol{p}, \boldsymbol{v}, \boldsymbol{c}_{\text{gnd}}) - 1)^2] + \mathbb{E}[\mathcal{D}(\boldsymbol{p}, \boldsymbol{v}, \mathcal{G}(\boldsymbol{e}_p))^2], \tag{4}$$

and $g(.)$ is the regularization term

$$g(\boldsymbol{c}_{\text{gnd}}, \boldsymbol{c}^\wedge) = |\boldsymbol{c}_{\text{gnd}} - \boldsymbol{c}^\wedge| + \|\boldsymbol{c}_{\text{gnd}} - \boldsymbol{c}^\wedge\|_2. \tag{5}$$

$g(.)$ penalizes the reconstruction loss of the predicted coordinate $\boldsymbol{c}^\wedge$ and the ground-truth coordinate $\boldsymbol{c}_{\text{gnd}}$. The total loss is the sum of the Embedding Loss and the Generative Loss

$$L = L_{emb} + L_{G^*}. \tag{6}$$

During training, both $\mathcal{G}$ and $\mathcal{D}$ will improve as they combat with each other. $\mathcal{G}$ will be better at predicting valid coordinates, and $\mathcal{D}$ will be better at determining fake generated coordinates.

Table 1: Detailed configuration of $\mathcal{G}$.

Input	$\boldsymbol{v} \in \mathbb{R}^{6\times10}$	$\boldsymbol{p} \in \mathbb{R}^{14\times10}$
Feature extractor	LSTM_v (6, 64)	LSTM_p (14, 64)
Extracted feature	$\boldsymbol{e}_v \in \mathbb{R}^{64\times1}$	$\boldsymbol{e}_p \in \mathbb{R}^{64\times1}$
Layers of $\mathcal{G}$		FC1 (64, 64), BatchNorm1D Leaky-ReLU, Dropout
		FC2 (64, 64), BatchNorm1D Leaky-ReLU, Dropout
		FC3 (64, 64), BatchNorm1D Leaky-ReLU, Dropout
		FC4 (64, 32), BatchNorm1D Leaky-ReLU
		FC5 (32, 3)
Output		$\boldsymbol{c}^\wedge \in \mathbb{R}^{3\times1}$

Table 2: Detailed configuration of $\mathcal{D}$.

Input	$\boldsymbol{v} \in \mathbb{R}^{6\times10}$	$\boldsymbol{p} \in \mathbb{R}^{14\times10}$	$\boldsymbol{c}^\wedge \in \mathbb{R}^{3\times1}$
Layers of $\mathcal{D}$	LSTM_v (6, 8)	LSTM_p (14, 8)	
	FC1 (19, 8), BatchNorm1D Leaky-ReLU		
	FC2 (8, 4), BatchNorm1D Leaky-ReLU		
	FC3 (4, 1)		
Output	$d \in \mathbb{R}$		

Eventually, an equilibrium is achieved during training, and the generated coordinates will be used as location estimations.

We implement the network architecture using PyTorch [28] – the detailed configurations of the network layers are listed in Table 1 and 2. We train the network with an NVIDIA 1080-Ti GPU with a batch size of 32 and a learning rate of 0.001 (0.0001 after 100 epochs).

3.2 Self-learning with association

Training the proposed network architecture requires a large amount of labeled vision-phone data correspondences. Obtaining sufficient data correspondences requires collecting multi-modal data from multiple pedestrians in various outdoor scenarios. Moreover, a large amount of extra effort is needed to determine and label the vision-phone correspondences from the collected multi-modal data so that they can be used in the training process. Although there

exists available labeled muti-modal datasets that are of reasonable scale for us to initially train the proposed network, it is always a challenge to bring in more training data to improve the network's performance.

To address this challenge, we propose a self-learning mechanism for our network to acquire more vision-phone data correspondences during inference. We associate pedestrians' camera domain coordinates with our GAN output coordinates. Since the input of the GAN during inference is pedestrians' phone data, solving the association problem is equivalently finding the correct vision-phone correspondences in the test data. The network is first trained with a limited portion of the labeled data correspondences. During the test phase, suppose at a timestamp in the test data there are M camera detected bounding boxes and N available phone data sequences. Using RGBD information, we can obtain M camera 3D coordinates $\{p^{\text{camera}}\}$; using our GAN to perform inference on these phone data sequences, we have N generated coordinates $\{p^{\text{phone}}\}$ that are with respect to the camera local 3D coordinate frame. Because the GAN is trained to produce realistic coordinates that are close to the pedestrian's true camera coordinates, for a true camera-phone data correspondence, the distance between its camera coordinates and its GAN-generated coordinates should be smaller than that of non-correspondences.

Using this heuristic, we choose the camera observation whose bounding box coordinate has the smallest Euclidean Distance to the GAN-produced coordinate as the associated identity from camera modality for every identity in the phone modality:

$$\text{AssociatedID}_i = \underset{j\in[1,M]}{\operatorname{argmin}} \|p_i^{\text{phone}} - p_j^{\text{camera}}\|_2. \tag{7}$$

In this way, we obtain good quality camera-phone correspondences as additional training samples without the dedicated effort of data collection or manual labeling. Then we combine the associated data with the initial labeled data to form a larger dataset that can be used to retrain or fine-tune the network. The feedback loop in Fig. 2 indicates that the pipeline of train-association-retrain can be executed in multiple iterations, with each iteration the association can bring in more new data correspondences. This allows the network to evolve on its own after it is initially trained with a limited amount of labeled data correspondences.

4 EVALUATION

Dataset We adopt the multimodal dataset in Vi-Fi [22] to train the network. The dataset contains pedestrians' camera data and their smartphone's wireless measurements. There are 3 user participants carrying smartphones and up to 12 passerby pedestrians in the camera view simultaneously. Camera data includes bounding box centroid and depth measurements; wireless measurements contain smartphone GPS readings, FTM ranging, and IMU measurements. The setup of dataset collection contains a roadside unit (RSU) that consists of an RGBD camera and WiFi access point that are placed together. A mounted Stereolabs ZED2 [4] (RGB-D) camera is set at the height of 2.4 to 2.8 meters with a proper field of view to record video at 10 fps, which collects depth information from 0.2 m to 20 m away from the camera. The smartphones are set to exchange FTM messages at 3 Hz frequency with a Google Nest WiFi Access Point anchored beside the camera. Each smartphone also logs its IMU sensor data at 50 Hz and GPS readings at 1 Hz (in Dataset B only). The smartphones and the camera are connected to the Internet to achieve synchronization.

Algorithm 1: Pseudo-code of the Particle Filter baseline

```
l = ENUcoord(RSU.lat, RSU.lon) ;            /* the RSU's position */
foreach GPSdata in GPSdataStream do
    Σ_GPS = CovMat(GPSdata.radius);
    FTMrange, FTMstd = fetchFTMdata(GPSdata.timestamp);
    for i = 0, 1, 2, ...N − 1 do
        μ = ENUcoord(GPSdata.lat, GPSdata.lon);
        p_i ~ N(μ, Σ_GPS) ;                 /* a particle's position */
        w_i = 1 ;
        /* Update the particle's weight */
        w_i ← w_i * N(‖p_i − l‖_2, FTMstd).pdf(FTMrange);
    end
    w ← w / Σ_{i=1}^{N} w_i ;                /* normalize weights */
    /* weighted average as final estimation */
    est = (1/N) Σ_{i=1}^{N} w_i p_i ;
end
```

The dataset contains in total 79 3-minute video sequences across 5 outdoor scenarios. We randomly choose 1 sequence from each scenario and use the vision-phone data correspondences from multiple pedestrians to construct the test set. We use the data from the rest 74 sequences to construct the training set. To match the timestamps of multi-modal data that have different sample rates (camera frames at 10 fps, FTM measurements at 3 Hz, IMU data at 50 Hz, and GPS readings at 1 Hz), we upsample the GPS readings with repetition and downsample the camera frames and IMU stream to 3 Hz. As discussed in Section 3.1, each data entry contains multi-modal data within a time window that contains data from k timestamps. In the evaluation, we empirically set $k = 10$. As a result, each data entry contains a pedestrian's multi-modal data whose duration is about 3 seconds. The rationale for setting $k = 10$ is that we want the time window to contain sufficient information about the pedestrians while not consuming too much memory. Maintaining a 3-second time window is feasible if the system needs to run in real-time. The total number of training entries is 110141 and the total number of testing entries is 6951.

Particle filter baseline Since the problem of cross-modal coordinate generation has not been specifically addressed in the literature, there are no off-the-shelf model architectures to compare against. Common solutions to localization adopt filter-based approaches to fuse measurements from multiple sources to obtain better estimations. In our context, however, the multi-modal data from pedestrians are not associated. When the camera data is not available, we can only rely on pedestrians' phone data to estimate their locations. Therefore, we use a particle filter as our baseline. It fuses phone GPS with FTM. The phone GPS data is obtained by an Android API function that returns the standard GNSS observations [5]. The particle filter approach corrects each GPS measurement with the RSU's FTM ranging information.

The particle filter contains two phases: prediction and update. In the prediction phase, the algorithm adopts the phone's GPS reading to construct a group of particles within a circle whose center position is the GPS reading and radius is the GPS's lateral error. Each particle is assigned the same weight, suggesting that the true position could be anywhere within the circle. When corresponding

FTM data arrives, the algorithm enters the update phase. Here, we update each particle's weight based on the difference between the FTM range and the particle's distance to the roadside unit position. The larger the difference, the smaller the weight is. In other words, we penalize the weights of those particles that are far from the FTM range circle. Finally, a pedestrian's location is computed as the weighted average of all the particles. A more detailed pseudo-code of the algorithm is presented in Algorithm 1.

4.1 Localization accuracy

The coordinates from the camera modality input are with respect to the local camera 3D coordinate frame; the coordinates from the phone modality input are with respect to the world's GPS system. To train our network and evaluate the accuracy of the predicted coordinates, the coordinates from multi-modal input need to be in the same reference system. We choose the camera's local 3D coordinate system as the reference frame. To convert pedestrians' GPS readings into coordinates with respect to the camera's 3D coordinate frame, we need to first obtain the coordinate transformation between the world and the camera.

Consider a reference point's coordinate in GPS format (latitude, longitude, altitude), it's coordinate can be converted into 3D world Cartesian format $P = [X_W, Y_W, Z_W]$ using the WGS84 model [6, 7, 33]. Its corresponding pixel coordinate on the image is $p = [u, v]$. The corresponding 3D world Cartesian coordinate and pixel coordinate have the relationship $[p, 1]^\top = \mathbf{K} \cdot {}^C\mathbf{T}_W \cdot [P, 1]^\top$, where $\mathbf{K} \in \mathbb{R}^{3\times3}$ represents the camera's intrinsics. ${}^C\mathbf{T}_W = [{}^C\mathbf{R}_W \quad {}^C\mathbf{t}_W]$ is the transformation matrix from the world to camera 3D coordinate frame. It contains a rotation matrix ${}^C\mathbf{R}_W \in \mathbb{R}^{3\times3}$ and a translation vector ${}^C\mathbf{t}_W \in \mathbb{R}^{3\times1}$.

We adopt the AP3P [21] algorithm to estimate this transformation. It takes as input 4 pairs of 3D-2D point correspondences with minimal measurement noise and outputs the estimated transformation matrix. As shown in Figure 4, we collect 6 reference points' GPS coordinates at each experimental field using a survey-grade Trimble-R2 GPS receiver (meter-level accuracy). We collect the reference points' corresponding 2D pixel coordinates with a pixel information tool that displays pixel coordinates in an image that the mouse pointer is positioned over.

We implement the AP3P algorithm using OpenCV's "solvePnP" method. For each scene that has 6 pairs of 3D-2D reference points, we iterate through all 4-point subsets and evoke AP3P to compute the transformation matrix multiple times. We choose the transformation matrix that has the lowest re-projection error as our final estimation. Once the world-camera transformation ${}^C\mathbf{T}_W$ is estimated, the camera-world transformation ${}^W\mathbf{T}_C$ can be derived as ${}^W\mathbf{T}_C = \left[{}^C\mathbf{R}_W^\top \quad -{}^C\mathbf{R}_W^\top \cdot {}^C\mathbf{t}_W\right]$. From the estimated camera-world transformation matrix, we can directly obtain the estimated RSU location P_{RSU} by fetching the last column of the ${}^W\mathbf{T}_C$.

To evaluate the quality of the transformation matrix, we compare the estimated RSU location P_{RSU} with the surveyed RSU location P^*_{RSU} measured by Trimble R2 and compute their Euclidean distance as the RSU position error $err_{\text{RSU}} = \|P_{\text{RSU}} - P^*_{\text{RSU}}\|_2$.

We also examine the reprojection error by projecting the reference points' 3D coordinates back into the image plane using the estimated transformation matrix and compare the projected pixel coordinates with their original 2D pixel coordinates. We compute the average and the standard deviation for all reference point's reprojection error $err_{\text{reproject}} = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{K} \cdot {}^C\mathbf{T}_W \cdot P_i - p_i\|$, where p_i and P_i are the i-th reference point's pixel coordinate and 3D world coordinate, respectively.

Table 3: Roadside unit position error and reprojection error for the estimated transformation matrix in each scene.

	RSU position error (m)	Reprojection error (pixel)	
		avg	std
Scene 1	1.455	36.3	46.3
Scene 2	1.876	31.0	21.8
Scene 3	0.886	27.1	31.2
Scene 4	1.077	25.6	28.7
Scene 5	1.864	33.5	38.7

Table 3 shows the RSU localization error and reprojection error for 5 scenes' world-camera transformation matrices. The small magnitudes of reprojection error and RSU position error suggest that the estimated transformation matrices are qualitatively satisfactory.

We use the above transformation matrices to convert baselines' GPS coordinates into the local camera coordinate system and compare them with our GAN-produced coordinates. The ground truth locations for pedestrians are their camera 3D coordinates which are derived from their bounding boxes' centroid and depth value.

We first present the localization results in a visual way for the 5 test scenes. As Figure 5 shows, different users' locations are highlighted by markers of different colors. We project each user's estimated 3D coordinates (with respect to the RSU's camera) into image pixel coordinates and highlight them using different markers of the same color. The solid squares represent the ground truth location of the users; the hollow squares represent raw GPS measurements; the hollow circles represent the FTM-fused GPS locations; the solid circles represent our GAN-generated location estimations. Compared to raw GPS measurements and the FTM-fused GPS location estimated by the particle filter, our method's estimated locations are the closest to the ground truth.

Table 4 provides more detailed quantitative results on localization error for the 5 test scenes. For baseline approaches, the phone's GPS readings have an average localization error of 7.443 m; the fused locations estimated by the particle filter have an average localization error of 5.339 m. In comparison, the estimated coordinates generated by our proposed GAN have an average localization error of 1.554 m. Moreover, for baseline approaches, the phone GPS readings and particle filter estimated locations exhibit large deviations across different scenes. In scene 5, specifically, where GPS performance degrades significantly due to tall buildings, the localization errors for baseline approaches are more than 10 meters. In contrast, our method produces consistent location estimations with errors varying between 1 to 2 meters. These comparisons suggest that our method is capable of producing location estimations that are consistently better than fused GPS location for different surrounding environments.

4.2 Perturbation on coordinate transformation

Readers might argue that the transformation matrix obtained by AP3P with reference points inevitably contains errors due to the measurement noise of the GPS collector. The true transformation could result in different localization errors, which might increase

Figure 4: Experimental fields and reference points that are used to estimate world-camera transformation (Best viewed zoomed).

Figure 5: Showcasing localization results for five test scenes. Each user's estimated 3D coordinates with respect to the RSU's camera are projected into the image and highlighted by different markers of the same color. The solid squares represent the ground truth location of the users; the hollow squares represent raw GPS measurements; the hollow circles represent the FTM-fused GPS locations; the solid circles represent our GAN-generated location estimations. Our method's estimated locations are the closest to the ground truth (Best viewed zoomed).

Table 4: Localization error (m) in average and standard deviation for different scenes in the test set.

	Phone GPS		Phone GPS + FTM		**Ours**	
	avg	std	avg	std	avg	std
Scene 1	3.460	1.897	2.030	1.092	1.620	0.951
Scene 2	7.314	4.509	6.055	2.468	1.822	1.581
Scene 3	3.899	2.439	3.807	2.398	1.678	1.331
Scene 4	3.728	1.552	2.940	2.289	1.432	0.927
Scene 5	16.96	6.263	10.76	4.429	1.351	0.849
Overall	7.443	6.727	5.339	4.366	**1.554**	**1.143**

Table 5: Perturbation Study results.

Perturbation		Phone GPS		Phone GPS + FTM		**Ours**	
σ_θ (°)	σ_t (m)	avg	std	avg	std	avg	std
0	0	7.443	6.727	5.339	4.366	**1.554**	**1.143**
5	0.5	7.869	6.930	5.777	4.538	1.584	1.110
10	1.0	7.720	6.508	5.502	4.268	1.688	1.242
15	1.5	8.220	6.399	6.243	4.211	1.702	1.236
20	2.0	7.692	5.141	5.956	3.450	1.930	1.423
25	2.5	9.582	6.361	7.830	4.298	1.915	1.495
30	3.0	9.487	5.530	7.559	4.055	2.224	1.842

the GAN estimated localization error and reduce the original GPS error. To address this concern, we conduct a perturbation study on the transformation matrix and evaluate how small perturbations affect pedestrians' localization errors.

We perturb the camera-world transformation by applying a small rotation $\mathbf{R}_p$ and a small translation $\mathbf{t}_p$ to the original transformation. The perturbation rotation $\mathbf{R}_p \in \mathbb{R}^{3\times3}$ consists of rotations with respect to the world's X, Y, and Z axis. The rotation angles θ_X, θ_Y, and θ_Z are randomly drawn from a zero-mean Gaussian distribution with standard deviation σ_θ. The perturbation translation $\mathbf{t}_p \in \mathbb{R}^{3\times1}$ is a zero-mean Gaussian random vector with covariance matrix $\mathbf{I} \cdot \sigma_t^2$. We vary the perturbation to the transformation matrix ${}^{W}T_C$ by changing the value of σ_θ and σ_t. In the evaluation, we choose $\sigma_\theta = \{5°, 10°, 15°, 20°, 25°, 30°\}$ and $\sigma_t = \{0.5\text{ m}, 1\text{ m}, 1.5\text{ m}, 2\text{ m}, 2.5\text{ m}, 3\text{ m}\}$ respectively. For each perturbed transformation matrix, we re-train our network and compute the average localization error across 5 scenes.

The results are shown in Table 5. Despite the fact that perturbing the World-Camera Transformation matrix will change localization errors for both the original GPS and our GAN method, the relationship between them stays the same. Our proposed method always provides more than 70% less localization error compared to smartphone's GPS readings. This suggests that the performance gain of our method is independent of the uncertainty of the transformation matrix. Considering the perfect World-Camera transformation is nearly impossible to derive, the perturbation study on the transformation matrix substantiates the argument that under reasonable evaluation metrics, our proposed GAN-estimated positions have lower localization errors than the baseline approaches.

4.3 The ability to generalize

Unseen Participants To evaluate if the network can generalize to unseen participants, we train the network using 2 users' data and test the network using the 3rd user's data that is not seen in the training set. In Figure 6, we see that our method produces better location estimations than baseline approaches. These results suggest that the network can be used to generate accurate locations for other unseen pedestrians at the same place. This is consistent with our previous argument that the GPS error for multiple pedestrians at a specific scene is correlated, and the learned GPS correction mapping can be applied to others. In real-world scenarios, there are many pedestrians walking across the intersections every day and it is infeasible to collect multi-modal data for everyone. But our network does not require to be trained on everyone's data. We can just train the network using a reasonable amount of data from a group of pedestrians and let it do the work for others. We will leverage this characteristic in the evaluation of self-learning.

Unseen environments Unlike unseen participants, we don't expect the network trained in one place to produce accurate locations for pedestrians in a new environment without any fine-tuning, because our model is scene-dependent. For a specific scene, it essentially learns the GPS error correction model that is determined by environmental factors such as satellite constellation. We observe from experiments that if a pre-trained network is deployed at a new scene (where the satellite constellation is different and the GPS error model changes), our method's localization error is no longer significantly less than, but similar to the raw GPS readings.

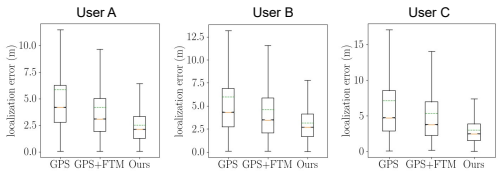

Figure 6: Generalization to unseen participants. For each sub-figure, the network is trained with 2 users' data and tested on the 3rd user's data across all scenes.

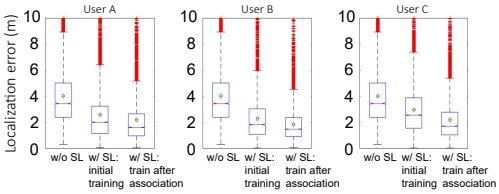

Figure 7: Self-learning results. Each sub-figure shows the effect of self-learning when a specific user's data is used as the initial training set.

4.4 Self-learning with associated data

If we deploy the network at a new scene, we will have to train the network with the new scene's data. But collecting new data and labeling the correspondences at each new scene is exhausting and sometimes infeasible. Fortunately, our proposed self-learning approach alleviates this problem and allows the network to work at a larger scale without excessive manual data collection.

We evaluate the self-learning mechanism by comparing the localization error under three different configurations. In Figure 7, each sub-figure shows the improvement of self-learning when a specific user's data is used as the initial training set. The test set contains 5 video sequences, each taken at 5 different scenes. They are treated as unseen environments. The first configuration is without self-learning (w/o SL). We train the network using data from 4 locations and test it on the data from the 5th location. We do this five times for 5 different scenes and plot the overall localization error in the left part of all three sub-figures. The second configuration is self-learning with initial training (w/ SL initial training). Here, we obtain 3 initially trained models, each trained on a different user's data. We test these initially trained models on the test set and plot their localization errors in the middle part of the three sub-figures. The last configuration is fine-tuning after self-learning's association (w/ SL train after association). We deploy the initially trained model (in the second configuration) to make inferences for other users and run the association to automatically accumulate additional data correspondences for the other two users. We then fine-tune the network using the expanded dataset. We then compute and plot the localization error on the same test set on the right part of the three sub-figures. From these results, we see that self-learning can further improve localization accuracy.

Table 6 shows the association precision and compares the localization error before and after training on the additional data correspondences that are obtained autonomously by association. The association precision measures how many associated camera-phone data pairs are true correspondences. Our association method

Table 6: Average localization (m) before and after training on additional data produced by the association.

	Train on one person's data	Association precision	Train on associated data	*Localization accuracy gain*
User A	2.577	78.7%	2.176	*15.6%*
User B	2.300	82.2%	1.857	*19.3%*
User C	2.954	71.2%	2.181	*26.2%*

produces a majority of good-quality data correspondence. The gain in localization accuracy varies from 15.6% to 26.2%. This suggests that the additional data correspondences obtained by the association are helping the GAN to learn a better GPS correction model. It is worth mentioning that the improvement in localization does not require perfect association. The precision of association is 70 – 80 percent, meaning that there are false-positive matches in the auto-generated dataset. But our mechanism is tolerable to these wrong associations. Because 1), most of the association is correct, so the data from true association plays a dominant role in the training phase; 2), for the camera bounding boxes that are incorrectly associated, their coordinates are not too far from the true locations. So they won't significantly degrade the error correction model that the network tries to learn. These results provide a promising semi-supervised direction as the proposed association mechanism allows the network to use its output to generate more training data and improve on its own.

We can leverage the network's ability to generalize on unseen participants and the self-learning mechanism to make ViFi-Loc adapt to new environments. When we deploy the network at a new place, we can first train the network using a small dataset that is easy to obtain, then use the self-learning approach to automatically accumulate additional data and further reduce the localization error through subsequent training and fine-tuning.

5 CONCLUSION

In this paper, we propose a network architecture that can be used in V2X applications to improve pedestrian and traffic safety. It is trained with multi-modal data including camera bounding boxes information and smartphone IMU, GPS, and FTM measurements. During inference, no camera data or multi-modal data association is required. The network produces accurate location estimations based only on pedestrians' phone data sequences. Our method outperforms the phone GPS and a particle filter baseline with an average localization error of 1.5 m. To alleviate manual labeling and data collection and enable the network to be deployed on a larger scale, we propose a self-learning approach that allows the network to use its output to generate more training data during test phases. By associating the produced coordinates with the coordinates from the camera-observed pedestrians, more vision-phone data correspondences can be obtained autonomously. Trained on the additional data correspondences, the localization accuracy of the generated coordinates is further improved by up to 26%. Extensive evaluation shows a promising direction for our proposed method to be deployed in large-scale real-world scenarios.

6 ACKNOWLEDGEMENT

This research has been supported by the National Science Foundation (NSF) under Grant No. CNS-1901355.

REFERENCES

[1] https://support.stereolabs.com/hc/en-us/articles/206953039-How-does-the-ZED-work-.

[2] https://www.intel.com/content/www/us/en/support/articles/000026260/emerging-technologies/intel-realsense-technology.html.

[3] https://www.gps.gov/systems/gps/performance/accuracy/.

[4] https://www.stereolabs.com/docs/object-detection/.

[5] https://developer.android.com/reference/android/location/LocationManager/.

[6] https://en.wikipedia.org/wiki/World_Geodetic_System/.

[7] https://gssc.esa.int/navipedia/index.php/Ellipsoidal_and_Cartesian_Coordinates_Conversion/.

[8] "IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". *"IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)"*, pages 1–3534, Dec 2016.

[9] A. Arun, R. Ayyalasomayajula, W. Hunter, and D. Bharadia. P2slam: Bearing based wifi slam for indoor robots. *IEEE Robotics and Automation Letters*, 7(2):3326–3333, 2022.

[10] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua. Cvae-gan: fine-grained image generation through asymmetric training. In *Proceedings of the IEEE international conference on computer vision*, pages 2745–2754, 2017.

[11] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[12] J. Cheng, L. Yang, Y. Li, and W. Zhang. Seamless outdoor/indoor navigation with wifi/gps aided low cost inertial navigation system. *Physical Communication*, 13:31–43, 2014.

[13] Y.-C. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm. Accuracy characterization for metropolitan-scale wi-fi localization. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 233–245, 2005.

[14] F. Dellaert. Factor graphs and gtsam: A hands-on introduction. Technical report, Georgia Institute of Technology, 2012.

[15] R. Gao, L. Xu, B. Zhang, and T. Liu. Raw gnss observations from android smartphones: Characteristics and short-baseline rtk positioning performance. *Measurement Science and Technology*, 32(8):084012, 2021.

[16] S. K. Gorti and J. Ma. Text-to-image-to-text translation using cycle consistent adversarial networks. *arXiv preprint arXiv:1808.04538*, 2018.

[17] C. Han, H. Hayashi, L. Rundo, R. Araki, W. Shimoda, S. Muramatsu, Y. Furukawa, G. Mauri, and H. Nakayama. Gan-based synthetic brain mr image generation. In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 734–738. IEEE, 2018.

[18] M. Ibrahim, H. Liu, M. Jawahar, V. Nguyen, M. Gruteser, R. Howard, B. Yu, and F. Bai. Verification: Accuracy evaluation of wifi fine time measurements on an open platform. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, pages 417–427. ACM, 2018.

[19] M. Ibrahim, A. Rostami, B. Yu, H. Liu, M. Jawahar, V. Nguyen, M. Gruteser, F. Bai, and R. Howard. Wi-go: accurate and scalable vehicle positioning using wifi fine timing measurement. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pages 312–324, 2020.

[20] M. I. M. Ismail, R. A. Dzyauddin, S. Samsul, N. A. Azmi, Y. Yamada, M. F. M. Yakub, and N. A. B. A. Salleh. An rssi-based wireless sensor node localisation using trilateration and multilateration methods for outdoor environment. *arXiv preprint arXiv:1912.07801*, 2019.

[21] T. Ke and S. I. Roumeliotis. An efficient algebraic solution to the perspective-three-point problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7225–7233, 2017.

[22] H. Liu, A. Alali, M. Ibrahim, B. B. Cao, N. Meegan, H. Li, M. Gruteser, S. Jain, K. Dana, A. Ashok, et al. Vi-fi: Associating moving subjects across vision and wireless sensors. In *2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 208–219. IEEE, 2022.

[23] H.-A. Loeliger. An introduction to factor graphs. *IEEE Signal Processing Magazine*, 21(1):28–41, 2004.

[24] Y. Lu, S. Wu, Y.-W. Tai, and C.-K. Tang. Image generation from sketch constraint using contextual gan. In *Proceedings of the European conference on computer vision (ECCV)*, pages 205–220, 2018.

[25] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.

[26] P. Misra, B. P. Burke, and M. M. Pratt. Gps performance in navigation. *Proceedings of the IEEE*, 87(1):65–85, 1999.

[27] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[29] T. Qin, P. Li, and S. Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[30] A. Rostami, B. Cheng, H. Lu, J. B. Kenney, and M. Gruteser. A light-weight smartphone gps error model for simulation. In *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, pages 1–5. IEEE, 2019.

[31] Y. Shi, K. Davaslioglu, and Y. E. Sagduyu. Generative adversarial network for wireless signal spoofing. In *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*, pages 55–60, 2019.

[32] Y. Shi, K. Davaslioglu, and Y. E. Sagduyu. Generative adversarial network in the air: Deep adversarial learning for wireless signal spoofing. *IEEE Transactions on Cognitive Communications and Networking*, 7(1):294–303, 2020.

[33] J. S. Subirana, J. J. Zornoza, and M. Hernández-Pajares. Ellipsoidal and cartesian coordinates conversion, 2016.

[34] X. Zhang, X. Tao, F. Zhu, X. Shi, and F. Wang. Quality assessment of gnss observations from an android n smartphone and positioning performance analysis using time-differenced filtering approach. *Gps Solutions*, 22(3):1–11, 2018.