Extrinsic Calibration of 2D Millimetre-Wavelength Radar Pairs Using Ego-Velocity Estimates

Qilong Cheng†, Emmett Wise†, and Jonathan Kelly

***Abstract*— Correct radar data fusion depends on knowledge of the spatial transform between sensor pairs. Current methods for determining this transform operate by aligning identifiable features in different radar scans, or by relying on measurements from another, more accurate sensor. Feature-based alignment requires the sensors to have overlapping fields of view or necessitates the construction of an environment map. Several existing techniques require bespoke retroreflective radar targets. These requirements limit both where and how calibration can be performed. In this paper, we take a different approach: instead of attempting to track targets or features, we rely on ego-velocity estimates from each radar to perform calibration. Our method enables calibration of a subset of the transform parameters, including the yaw and the axis of translation between the radar pair, without the need for a shared field of view or for specialized targets. In general, the yaw and the axis of translation are the most important parameters for data fusion, the most likely to vary over time, and the most difficult to calibrate manually. We formulate calibration as a batch optimization problem, show that the radar-radar system is identifiable, and specify the platform excitation requirements. Through simulation studies and real-world experiments, we establish that our method is more reliable and accurate than state-of-the-art methods. Finally, we demonstrate that the full rigid body transform can be recovered if relatively coarse information about the platform rotation rate is available.**

***Index Terms*— Calibration & Identification, Radar, Robot Sensing Systems, Sensor Fusion**

I. Introduction

Millimetre-wavelength radar has proven to be a valuable sensing modality for autonomous vehicles (AVs) because radar is relatively robust to inclement weather and is able to provide velocity information [1]. However, the field of view of most radar sensors is limited and radars are known to produce noisy range and range-rate measurements. To provide full coverage of the environment and to ensure redundancy, AVs often aggregate data from multiple radars. Data fusion, in turn, requires accurate knowledge of the spatial transform(s) between the sensors. The process of determining the sensor-to-sensor transform is known as extrinsic calibration. While the sensors on board many AVs are factory-calibrated prior to deployment, the spatial transform may change during operation for a variety of reasons (e.g., collisions, wear and tear, etc.). An ability to perform online, in-situ extrinsic calibration is therefore important for safety and reliability.

† denotes equal contribution.

All authors are with the Space & Terrestrial Autonomous Robotics Systems Laboratory at the University of Toronto Institute for Aerospace Studies, Toronto, Canada. Jonathan Kelly is a Vector Institute Faculty Affiliate. <first>.<last>@robotics.utias.utoronto.ca

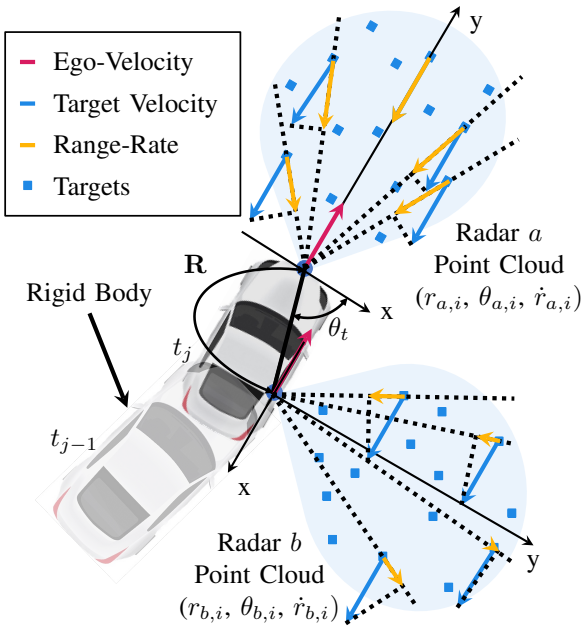


Fig. 1: Illustration of our 2D radar-to-radar extrinsic calibration problem. Radars a and b measure the range, azimuth, and range-rate to targets in the environment. Assuming that the targets are stationary in a fixed, world reference frame, we can estimate the ego-velocity of each radar. Our algorithm fuses the ego-velocity estimates from radars a and b to estimate the rotation, $\mathbf{R}$, and translation direction θ_t between the the two sensors.

Existing extrinsic calibration methods for 2D radar pairs either align clouds of 3D points derived from radar measurements [2] or involve other (non-radar) sensors that have better accuracy (e.g., lidar units) [3]–[5]. The method in Olutomilayo et al. [2] relies on specialized trihedral radar retroreflectors to ensure sufficient environmental structure for calibration and to simplify the data association problem, for example. However, these retroreflectors are bespoke and are unavailable outside of the laboratory, limiting the possibility for calibration in the field, during operation.

In this paper, we study the radar-to-radar extrinsic calibration problem and develop a calibration approach that does not require specialized targets or other sensors. To avoid the challenges of data association, we instead propose a method that relies on instantaneous ego-velocity estimates from each radar only, as shown in Figure 1. More specifically, we determine the rotation angle and translation axis (i.e., unit vector) between the sensors. Radar data fusion, especially radar ego-velocity fusion, is most sensitive to these parameters. Under nominal operating conditions, the distance

between the sensors is unlikely to change appreciably from specifications. In contrast, the orientation of each radar can easily be altered (e.g., by a minor impact) and is very difficult to measure by hand. To the best of the authors' knowledge, this is the first work to explore extrinsic calibration of sensor pairs that provide velocity estimates only. That is, no direct information about the rotation or rotation rate of the sensor platform is considered (instead, rotation must be inferred from the velocities). Herein, we:

1) show that the yaw angle and the direction of translation between pairs of coplanar 2D radar units can be determined from ego-velocity estimates only;
2) formulate a batch solver for the calibration problem and prove that the parameters are identifiable, given sufficient excitation;
3) carry out simulation studies to analyze the sensitivity of our method to varying levels of measurement noise;
4) confirm via real-world experiments that our approach is more accurate and reliable than two state-of-the-art methods; and
5) demonstrate that the full spatial transform can be recovered when an additional, coarse source of information about the platform rotation rate is available.

The remainder of the paper is organized as follows. In Section II, we review existing radar extrinsic calibration algorithms. Section III formulates our batch optimization problem. We prove that the calibration parameters are identifiable and establish the necessary trajectory excitation requirements in Section IV. In Section V, we show, through simulations and real-world experiments, that our algorithm is more reliable and accurate at estimating the yaw angle and the translation axis between radar pairs than two state-of-the-art-methods. Finally, we summarize our work and discuss future research directions in Section VI.

II. Related Work

In this section, we survey pairwise extrinsic calibration methods where one (or both) of the sensors in the pair is a mm-wavelength radar. Sections II-A and II-B detail target-based and target-free extrinsic calibration algorithms, respectively, that rely on feature detection and matching. In Section II-C, we review extrinsic calibration methods that relate the instantaneous radar ego-velocity to the motion of the second sensor.

A. Target-Based Methods

Most existing radar-camera extrinsic calibration algorithms estimate the projective transform (homography) between the horizontal 2D radar sensing plane and the camera image plane. Due to the sparse and noisy nature of radar measurements, these methods rely on specialized trihedral retroreflectors that produce point-like detections in the radar and camera data while simplifying the correspondence problem [6]–[9]. Additionally, although 2D radars are incapable of estimating elevation, the sensors do often detect off-plane targets, and these detections bias the homography estimate. Since signal returns from targets on the radar horizontal plane are stronger than those from off-plane targets, Sugimoto et al. [6] filter on-plane targets by maximizing the measured radar cross section (RCS) of the targets.

The most common error metric for radar-to-sensor extrinsic calibration is a form of 'reprojection' error, which defines the misalignment between identifiable targets (objects) viewed by both sensors. For example, Olutomilayo et al. [2] estimate the 2D transform between the radar frame and a vehicle coordinate frame by aligning radar measurements of stationary retroreflectors with a known map in the vehicle coordinate frame. The approaches of El Natour et al. [10], Domhoff et al. [11], and Peršić et al. [12] treat all radar measurements as lying on spherical arcs with constant range and azimuth (i.e., the measurements vary only in elevation). To estimate the 3D transform between sensor pairs, the arcs are aligned with the measurements from a second sensor. In order to account for the elevations of the retroreflector targets relative to the horizontal radar sensing plane, these methods introduce additional calibration constraints by designing specific target arrangements [10], [11] or explicitly modelling the radar-target interactions [12]. All of these techniques require the sensor pair to simultaneously view one or more specialized targets, so the sensors must share overlapping fields of view. Our approach does not require specialized infrastructure or a shared field of view, allowing for calibration of a wider range of sensor configurations and in more environments.

B. Target-Free Methods

In contrast to methods that rely on specialized radar retroreflectors, target-free or 'targetless' algorithms estimate the radar-to-sensor transform by aligning identifiable environment features (observed by both sensors). Schöller et al. [13] train a neural network to correct an inaccurate rotation estimate between a 2D radar-camera pair using raw camera and radar vehicle detection data. Peršić et al. [14] align tracked objects to determine the yaw angle between 2D radar-camera and radar-lidar pairs. Due to the challenge of tracking environmental features consistently across radar scans, these methods only calibrate the rotation between the sensors. Burnett et al. [3] estimate the transform between a 2D radar-lidar pair, where both sensors have a 360° field of view. The method in [3] aligns measured radar and lidar point clouds, which requires a large number of jointly-observed features. Heng et al. [15] estimate the extrinsic calibration parameters between 3D radar-lidar pairs by constructing a lidar point cloud map and localizing the 3D radar units within the map. While this approach could possibly be used for 2D radars, the method requires the known poses of the vehicle and the construction of dense map. Our approach does not require tracking of environmental features, which simplifies the calibration process.

C. Ego-Velocity Methods

Ego-velocity methods estimate the transform by minimizing the error between radar ego-velocity estimates and the motion of another sensor [1], [16]. By minimizing the lateral

velocity error between a radar and an inertial measurement unit (IMU), Kellner et al. [17] estimate the rotation angle between a radar-IMU pair, but their scheme requires accurate knowledge of the translation between the sensors. Doer et al. [5] and Wise et al. [4] extend this approach to 3D radar-IMU and radar-camera extrinsic calibration. To date, each ego-velocity method relies on one sensor that provides rotation information (e.g., angular velocity or $\mathrm{SO}(2)$ measurements relative to an inertial frame). Herein, we determine the subset of extrinsic calibration parameters that are identifiable without angular velocity measurements.

III. Problem Formulation

A. Notation

In this paper, Latin and Greek letters, such as a and α, represent scalar variables. Lowercase (e.g., $\mathbf{h}$ and $\boldsymbol{\sigma}$) and uppercase (e.g., $\boldsymbol{\Theta}$ and $\mathbf{C}$) boldface characters are reserved for vectors and matrices, respectively. A Cartesian reference frame is identified by $\underrightarrow{\mathcal{F}}$. The translation vector from $\underrightarrow{\mathcal{F}}_a$ to $\underrightarrow{\mathcal{F}}_b$, expressed in $\underrightarrow{\mathcal{F}}_a$, is denoted by $\mathbf{t}_a^{ba}$. The function $\mathbf{R}(\theta)$ maps $\theta \in \mathbb{R}$ to an element of $\mathrm{SO}(2)$; for example, $\mathbf{R}(\theta_{ab})$ defines the rotation from $\underrightarrow{\mathcal{F}}_b$ to $\underrightarrow{\mathcal{F}}_a$. We use $\mathbf{I}_n$ to denote the n-by-n identity matrix. The operator $\times$ is the cross product operator. The unary operator $^\wedge$ acts on $r \in \mathbb{R}$ to produce

$$r^\wedge = \begin{bmatrix} 0 & -r \\ r & 0 \end{bmatrix}. \tag{1}$$

B. Radar Ego-Velocity Estimation

Let the static world and moving radar frames be $\underrightarrow{\mathcal{F}}_w$ and $\underrightarrow{\mathcal{F}}_r$, respectively. At each time index j, the radar detects N stationary, environmental features in $\underrightarrow{\mathcal{F}}_w$. The resulting radar measurement is $\{\begin{bmatrix} r_1 & \theta_1 & \dot{r}_1 \end{bmatrix}, \ldots, \begin{bmatrix} r_N & \theta_N & \dot{r}_N \end{bmatrix}\}$, where r_i, θ_i, and $\dot{r}_i$ are, respectively, the range, azimuth, and range-rate (i.e., the Doppler velocity) of feature i in $\underrightarrow{\mathcal{F}}_r$. If we assume that each feature lies on the horizontal plane of the radar, then the range-rate of a feature is

$$\dot{r}_i = -\begin{bmatrix} \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \mathbf{h}_r^j, \tag{2}$$

where $\mathbf{h}_r^j$ is the 2D radar ego-velocity at time j. A depiction of the relationship between the range-rate of stationary features and ego-velocity of the radar is shown in Figure 1.

Radar ego-velocity estimation can be cast as a linear least squares problem, where the measurement model is

$$\underbrace{\begin{bmatrix} -\dot{r}_1 \\ -\dot{r}_2 \\ \vdots \\ -\dot{r}_N \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} \sin(\theta_1) & \cos(\theta_1) \\ \sin(\theta_2) & \cos(\theta_2) \\ \vdots & \\ \sin(\theta_N) & \cos(\theta_N) \end{bmatrix}}_{\mathbf{A}} \mathbf{h}_r^j. \tag{3}$$

The resulting error equation is

$$\boldsymbol{\epsilon} = \mathbf{y} - \mathbf{A}\mathbf{h}_r^j, \tag{4}$$

and the radar ego-velocity estimation problem is

$$\min_{\mathbf{h}_r^j \in \mathbb{R}^2} \boldsymbol{\epsilon}^T \boldsymbol{\epsilon}. \tag{5}$$

As a result, the estimated ego-velocity at time j is

$$\mathbf{h}_r^{j\star} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{y}, \tag{6}$$

with covariance

$$\boldsymbol{\Sigma}_r^j = \frac{(\boldsymbol{\epsilon}^T \boldsymbol{\epsilon})(\mathbf{A}^T \mathbf{A})}{N-2}. \tag{7}$$

We leverage RANSAC to remove outliers such as targets moving relative to $\underrightarrow{\mathcal{F}}_w$ and multipath radar reflections [18].

C. Radar Ego-Velocity Measurement Models

Let $\underrightarrow{\mathcal{F}}_a$ and $\underrightarrow{\mathcal{F}}_b$ be the reference frames of two rigidly attached radars that share and move along one horizontal sensing plane. The ego-velocity measurement models for radars a and b, at time j, are

$$\begin{aligned} \mathbf{h}_{r,a}^j &= \mathbf{v}_{r,a}^j + \mathbf{n}_{r,a}^j, \\ \mathbf{n}_{r,a}^j &\sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}_{r,a}^j\right), \end{aligned} \tag{8}$$

$$\begin{aligned} \mathbf{h}_{r,b}^j &= \mathbf{R}(\theta_{ba})(\omega^{j\wedge} \mathbf{t}_a^{ba} + \mathbf{v}_{r,a}^j) + \mathbf{n}_{r,b}^j, \\ \mathbf{n}_{r,b}^j &\sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}_{r,b}^j\right), \end{aligned} \tag{9}$$

where $\mathbf{v}_{r,a}$ is the ego-velocity of the radar, θ_{ba} is the rotation from radar a to radar b, ω^j is the angular velocity of the rigid body, and $\mathbf{t}_a^{ba}$ is the translation from radar a to radar b, expressed in the reference frame of a. The vectors $\mathbf{n}_{r,a}^j$ and $\mathbf{n}_{r,b}^j$ are additive zero-mean Gaussian noise terms with covariances $\boldsymbol{\Sigma}_{r,a}^j$ and $\boldsymbol{\Sigma}_{r,b}^j$, respectively. The values of $\boldsymbol{\Sigma}_{r,a}^j$ and $\boldsymbol{\Sigma}_{r,b}^j$ are determined with use of Equation (7).

The error equations corresponding to the ego-velocity estimates are

$$\begin{aligned} \mathbf{e}_{r,a}^j &= \mathbf{h}_{r,a}^j - \mathbf{v}_{r,a}^j, \\ \mathbf{e}_{r,a}^j &\sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}_{r,a}^j\right), \\ \mathbf{e}_{r,b}^j &= \mathbf{h}_{r,b}^j - \mathbf{R}(\theta_{ba})(\omega^{j\wedge} \mathbf{t}_a^{ba} + \mathbf{v}_{r,a}^j), \\ \mathbf{e}_{r,b}^j &\sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}_{r,b}^j\right), \end{aligned} \tag{10}$$

where $\mathbf{h}_{r,a}^j$ and $\mathbf{h}_{r,b}^j$ are the values from Equation (6) for radars a and b, respectively.

D. Batch 2D Radar to Radar Extrinsic Calibration

Given M pairs of synchronized radar measurements, the vector of parameters that we wish to estimate includes the ego-velocity of radar a from time 1 to M, the angular velocity of radar a from time 1 to M, the translation from radar a to b expressed in $\underrightarrow{\mathcal{F}}_a$, and the rotation from radar $\underrightarrow{\mathcal{F}}_a$ to $\underrightarrow{\mathcal{F}}_b$,

$$\mathbf{x}^T = \begin{bmatrix} \mathbf{v}_{r,a}^{1\,T} & \omega^1 & \cdots & \mathbf{v}_{r,a}^{M\,T} & \omega^M & \mathbf{t}_a^{baT} & \theta_{ba} \end{bmatrix}. \tag{11}$$

Our calibration problem is to solve

$$\min_{\mathbf{x}} \sum_{j=1}^{M} \mathbf{e}_{r,a}^{j\,T}\, \boldsymbol{\Sigma}_{r,a}^{j\,-1} \mathbf{e}_{r,a}^j + \mathbf{e}_{r,b}^{j\,T}\, \boldsymbol{\Sigma}_{r,b}^{j\,-1} \mathbf{e}_{r,b}^j. \tag{12}$$

E. Scale and Angular Velocity Indistinguishability

Unfortunately, the optimization problem defined by Equation (12) has infinitely many indistinguishable solutions.

Given any solution that minimizes Equation (12), another minimizer can be found by arbitrarily scaling $\omega^j \ \forall \ j = 1, \ldots, M$ and $\mathbf{t}_a^{ba}$ by $\gamma \in \mathbb{R}$ and $\frac{1}{\gamma}$, respectively. However, the problem can be made distinguishable with additional constraints.

To make the optimization problem identifiable (see Section IV), we constrain

$$\left\|\mathbf{t}_a^{ba}\right\|_2 = 1. \tag{13}$$

We enforce this constraint by setting

$$\mathbf{t}_a^{ba} = \begin{bmatrix}\cos(\theta_t)\\ \sin(\theta_t)\end{bmatrix}, \tag{14}$$

where θ_t is the angle from the x-axis of radar a to the line of possible translations between radars a and b. Since the angle to the line is periodic with period π, we bound $0 \leq \theta_t < \pi$. We denote the resulting unscaled angular velocity as ω_γ^j. Consequently, our vector of parameters for the optimization problem becomes

$$\mathbf{x}^T = \begin{bmatrix}\mathbf{v}_{r,a}^{1\ T} & \omega_\gamma^1 & \cdots & \mathbf{v}_{r,a}^{M\ T} & \omega_\gamma^M & \theta_t & \theta_{ba}\end{bmatrix}. \tag{15}$$

We substitute Equation (14) into Equation (12) and solve the problem using the Levenberg-Marquardt algorithm.

F. Problem Initialization

Since the two radar sensors provide no rotational information, we require a method to initialize the angular velocities that appear in Equation (15). To start, we determine θ_{ba} by finding the K pairs of ego-velocity estimates that have similar magnitudes. Using these 'velocity pairs,' we compute

$$\theta_{ba}^l = \begin{bmatrix}0 & 0 & 1\end{bmatrix}\left(\begin{bmatrix}\mathbf{h}_{r,a}^l\\ 0\end{bmatrix} \times \begin{bmatrix}\mathbf{h}_{r,b}^l\\ 0\end{bmatrix}\right). \tag{16}$$

The initial θ_{ba} is the median of $\theta_{ba}^l \ \forall \ l = 1, ..., K$. To initialize θ_t, we use

$$\begin{aligned}\mathbf{b}^j &= \begin{bmatrix}b_x^j & b_y^j\end{bmatrix}^T = \mathbf{R}(\theta_{ba})^T\,\mathbf{h}_{r,b}^j - \mathbf{h}_{r,a}^j,\\ \theta_t^j &= \text{arctan2}\left(\frac{b_y^j}{\left\|\mathbf{b}^j\right\|_2}, -\frac{b_x^j}{\left\|\mathbf{b}^j\right\|_2}\right).\end{aligned} \tag{17}$$

Each θ_t^j is mapped to the corresponding value within $[0, \pi)$ and the initial θ_t is the median of $\theta_t^j \ \forall \ j = 1, ..., M$. By fixing θ_t and θ_{ba} to our initial estimates, Equation (12) becomes an unconstrained quadratic problem. We solve this problem to initialize ω_γ^j and $\mathbf{v}_{r,a}^j \ \forall \ j = 1 \ldots M$.

IV. Identifiability

In this section, we prove that, given sufficient excitation of the system, the extrinsic calibration problem is identifiable. Since a problem that is locally weakly observable is also identifiable (in the batch setting), we use the rank criterion defined by Hermann and Krener in [19] in our proof. Section IV-A reviews the rank criterion. In Section IV-B, we demonstrate that our problem is locally weakly observable. Finally, we highlight important degenerate motions in Section IV-C that result in a loss of observability and potentially also identifiability.

A. The Observability Rank Criterion

Consider the system

$$S\begin{cases}\dot{\mathbf{x}} = \mathbf{f}_0(\mathbf{x}) + \sum_{j=1}^p \mathbf{f}_j(\mathbf{x})\,u_j\\ \mathbf{y} = \mathbf{h}(\mathbf{x})\end{cases}, \tag{18}$$

where $\mathbf{x}$ is the state vector, $\mathbf{f}_0(\mathbf{x})$ is the drift vector field, $\mathbf{f}_j(\mathbf{x})$ is a vector field on the state manifold that is linear with respect to the control input u_j, $\mathbf{y}$ is the measurement vector, and $\mathbf{h}(\mathbf{x})$ is the measurement model. Given the vector field $\mathbf{f}(\mathbf{x})$, we can compute the Lie derivative of $\mathbf{h}$ with respect to $\mathbf{f}$, which is defined as

$$L_\mathbf{f}\mathbf{h}(\mathbf{x}) = \nabla_\mathbf{f}\mathbf{h}(\mathbf{x}) = \frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}). \tag{19}$$

The n^{th} Lie derivative of h with respect to $\mathbf{x}$ along vector field $\mathbf{f}$ is defined as,

$$L_\mathbf{f}^n\mathbf{h}(\mathbf{x}) = \frac{\partial L_\mathbf{f}^{n-1}\mathbf{h}(\mathbf{x})}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}), \tag{20}$$

where $L^0\mathbf{h}(\mathbf{x}) = \mathbf{h}(\mathbf{x})$.

The Lie derivatives can be vertically stacked to form the observability matrix $\mathbf{O}$. From Hermann and Krener [19], a system is locally weakly observable at $\mathbf{x}$ if the matrix $\mathbf{O}$ is full column rank at $\mathbf{x}$.

B. Observability of Extrinsic Calibration

Let $\mathbf{h}_{r,a}$ and $\mathbf{h}_{r,b}$ be 2D radar ego-velocity measurements. We define the state at timestep t_j as

$$\mathbf{x}^T(t_j) = \begin{bmatrix}\omega_\gamma(t_j) & \alpha_\gamma(t_j) & \theta_t & \theta_{ba}\end{bmatrix}, \tag{21}$$

where $\alpha_\gamma(t_j)$ is the instantaneous angular acceleration.[1] We assume the vehicle follows the constant angular acceleration model given by

$$\begin{aligned}\dot{\alpha}_\gamma(t_j) =& n_\alpha,\\ n_\alpha \sim& \mathcal{N}\left(0, \sigma_\alpha^2\right),\end{aligned} \tag{22}$$

where n_α is an additive zero-mean Gaussian noise term with variance σ_α^2. Since the motion is noiseless in this analysis (i.e., $\dot{\alpha}_\gamma(t_j) = 0$), the motion model is

$$\dot{\mathbf{x}}^T(t_j) = \begin{bmatrix}\alpha_\gamma(t_j) & \mathbf{0}_{3\times1}^T\end{bmatrix}. \tag{23}$$

We can substitute Equation (8) into Equation (9), which simplifies the measurement model to

$$\mathbf{h}_{r,b}^j = \mathbf{R}(\theta_{ba})\left(\omega_\gamma(t_j)^\wedge \begin{bmatrix}\cos\theta_t\\ \sin\theta_t\end{bmatrix} + \mathbf{h}_{r,a}^j\right). \tag{24}$$

The observability matrix of this system can be written as

$$\mathbf{O} = \begin{bmatrix}\nabla_\mathbf{x} L^0\mathbf{h}_{r,b}^j\\ \nabla_\mathbf{x} L_f^1\mathbf{h}_{r,b}^j\end{bmatrix}, \tag{25}$$

which is full column rank[2] except when the sensor platform motion is degenerate, as discussed below.

[1] The analysis can, in fact, be simplified by removing the angular acceleration state; we use this formulation, specifically, in Section IV-C.

[2] The rank of $\mathbf{O}$ can be determined using a symbolic math package. We omit the full proof for brevity.

C. Degeneracy Analysis

The system is unobservable (and potentially unidentifiable) when $\mathbf{O}$ does not have full column rank. The determinant of the observability matrix is

$$\det(\mathbf{O}) = \begin{bmatrix} 0 & 0 & \alpha_\gamma(t_j) \end{bmatrix} \left(\begin{bmatrix} \mathbf{h}_{r,a}^j \\ 0 \end{bmatrix} \times \begin{bmatrix} \cos\theta_t \\ \sin\theta_t \\ 0 \end{bmatrix} \right), \quad (26)$$

which is rank-deficient when $\det(\mathbf{O}) = 0$. As a result, the system must have nonzero angular acceleration, α_γ, and nonzero ego-velocity, $\mathbf{h}_a$. Additionally, the direction of ego-motion must not align with the sensor translation axis.

V. Experiments

To verify the performance of our algorithm, we conducted a series of simulated and real-world experiments. In Section V-A we show, using simulated data, that our algorithm is robust to realistic levels of radar measurement noise and that it yields an improved ego-velocity estimate. In Section V-B, we compare our approach to two state-of-the-art methods on the publicly-available Endeavour dataset.[3]

A. Simulation Studies

We performed a series of simulation studies to evaluate the robustness of our algorithm to measurement noise. We varied the simulation duration and the level of noise and generated 100 randomized trials with each pair of settings. Each simulation ranged in duration from 15 s to 120 s; the simulated sensor platform followed a periodic, nominal (noise-free) trajectory with sufficient excitation for our calibration problem (see Figure 2). The radar ego-velocity estimates for radars a and b were computed using the ground truth linear and angular velocities of the platform along the trajectory. Ego-velocity measurements from radars a and b were then corrupted with zero-mean Gaussian noise ($\mathbf{\Sigma}_{r,a}^j = \mathbf{\Sigma}_{r,b}^j = \sigma_r^2 \mathbf{I}_2$), where the standard deviation of the noise (σ_r) ranged from 0.05 m/s to 0.2 m/s. Based on our experiments (discussed in more detail in Section V-B), we found the real-world measurement noise to be at the lower end of this range.

The error distribution of the estimated calibration parameters is shown in Figure 3. For most noise levels and durations, our estimated translation direction and rotation angle are, respectively, within 2° and 3° of the ground truth. Figure 4 shows that the median of the estimated ego-velocity errors for radars a and b are both 4 cm/s lower than the raw estimates. Importantly, this improvement can be achieved without the need for additional rotation information.

B. Real-World Experiments

We demonstrate the reliability of our method and compare to two state-of-the-art algorithms on the Endeavour dataset. Post-hoc extrinsic calibration for this dataset is challenging because the environments contain no trihedral reflectors. We demonstrate that the lack of trihedral reflectors has a negligible impact on our method, but is detrimental to the method in Olutomilayo et al. [2]. Additionally, we show that the parameters estimated by our method result in smaller velocity errors than the parameters estimated by two state-of-the-art methods. The first technique follows the approach in Olutomilayo et al. [2]. To build the required map for this method, we collate measurements from one radar while the vehicle is stationary. The second method is similar to the approach in Burnett et al. [3]. The parameters estimated by this method are included in the Endeavour dataset.[4] Next, we demonstrate that normal driving motions provide sufficient excitation to calibrate radar pairs that have translation axes which align with the forward direction of the vehicle. Finally, we show that, when a source of angular velocity information is available, the scale of the translation between the radar pair can be estimated.

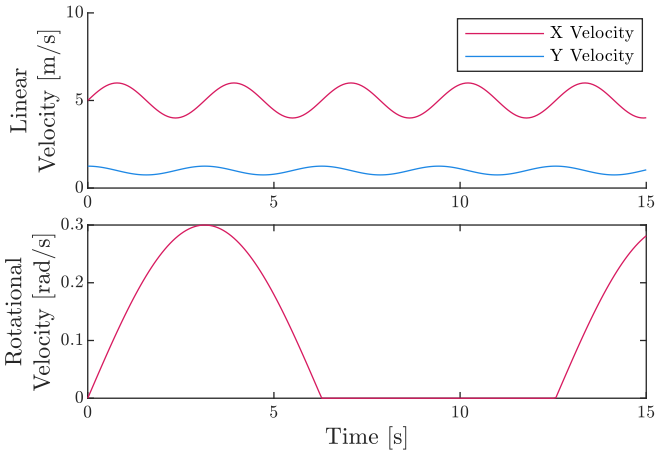


Fig. 2: Top: ego-velocity of radar a over 15 s. Bottom: angular velocity of radar a over 15 s. Both plots show the full period of the velocity functions.

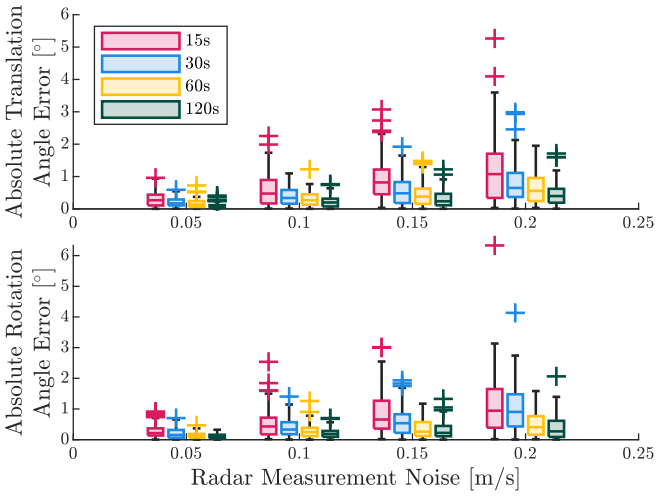


Fig. 3: Absolute translation direction and rotation angle error for our algorithm at varying levels of measurement noise and simulation durations. The translation direction is the angle from the x-axis of radar a to the line of indistinguishable translations between radars a and b.

[3]The dataset is available at: https://gloryhry.github.io/2021/06/25/Endeavour_Radar_Dataset.html

[4]The radar to lidar extrinsic calibration code for the Endeavour dataset can be found at: https://github.com/gloryhry/radar_lidar_static_calibration

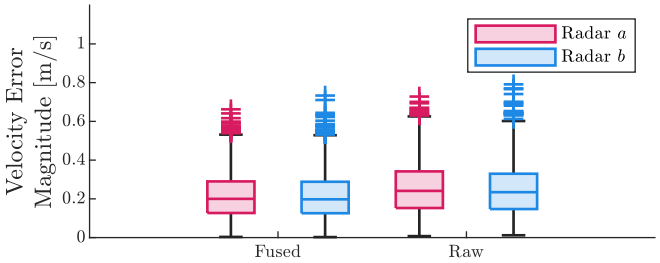


Fig. 4: Raw and fused ego-velocity estimate errors of radars a and b for an experiment that is 120 s in duration with a measurement noise level of 0.2 m/s.

The Endeavour dataset was collected from a small shuttle bus driving around three different loops in a campus setting. The dataset contains two runs for each loop, where each run is roughly 10 minutes long. The shuttle bus has a BDStar Navigation Npos320 RTK GNSS, four Velodyne VLP-16 lidars, and five Continental ARS430 radars, which operate at 100 Hz, 10 Hz, and 14 Hz, respectively. The radar labelled `Near_5` is mounted on the front bumper and observes the environment in front of the vehicle. Radar pairs `Near_3`–`Near_1` and `Near_4`–`Near_2` observe the environment surrounding the sides of the vehicle. Radars `Near_3` and `Near_1` are mounted on the front and back driver's side of the vehicle, while radars `Near_4` and `Near_2` are mounted on the front and back passenger side of the vehicle.[5]

Before applying our method to the Endeavour dataset, we tuned our RANSAC-based ego-velocity estimator, synchronized the radar measurement timestamps and removed zero-velocity measurement pairs. For RANSAC, the inlier and outlier thresholds were set to 40% of the number of measured reflections and 0.025 m/s, respectively. These thresholds were determined using the radar and GNSS velocity data from `East_2`. To temporally synchronize the data streams, we aligned the radar measurement timestamps using linear interpolation. Finally, we removed ego-velocity measurement pairs with magnitudes less than 0.05 m/s to improve the signal-to-noise ratio in the calibration problem.

Estimating the transforms for Olutomilayo et al. [2] required two pre-processing steps. First, we identified stationary radar measurements using the RTK GNSS data and removed points that were observed less than five times. Next, we expressed (using the Endeavour parameters) the radar point clouds in a common reference frame and associated points that were within a 10 cm threshold. Finally, the extrinsic transforms from Olutomilayo et al. [2] and Endeavour were chained together to compute the rotation angles and translation axes relative to `Near_5`.

Table I shows that our parameters are within 3° of the provided parameters while the parameters from Olutomilayo et al. [2] deviate very significantly. This deviation is due to the narrow overlap between the fields of view of some of the radar pairs, which results in sparse overlapping point clouds. Often, this systematic issue results in the data collection runs having insufficient information for the method in [2] to operate properly (see Table II).

[5]Additional detailed information is available at: `https://gloryhry.github.io/2021/06/25/Endeavour_Radar_Dataset.html`

TABLE I: θ_t and θ_{ba} parameters estimated by each method on `East2`. Radar a is `Near_5` for every parameter given.

	Near_1		Near_2		Near_3		Near_4	
Method	θ_t	θ_{ba}	θ_t	θ_{ba}	θ_t	θ_{ba}	θ_t	θ_{ba}
Endeavour	1.71	-1.57	1.44	1.59	2.73	-1.58	0.33	1.56
Olutomilayo [2]	1.36	-1.60	1.41	1.58	2.16	-1.61	0.39	1.56
Ours	1.74	-1.58	1.41	1.61	2.79	-1.58	0.35	1.56

*All angles are in radians.

We use the mean velocity error magnitude to evaluate the accuracy of the calibration parameters given in Table I. The velocity error of a radar measurement pair, $\mathbf{h}^j_{r,a}$ and $\mathbf{h}^j_{r,b}$, is $\mathbf{e}^j_{r,b}$ from Equation (10), where $\mathbf{v}^j_a = \mathbf{h}^j_{r,a}$, θ_{ba} is the estimated rotation, θ_t is the estimated translation axis, and ω^j_γ is the value that minimizes the magnitude of $\mathbf{e}^j_{r,b}$. Table III shows the mean velocity error magnitude for each run, radar pair, and set of parameters. Our parameters yield lower velocity errors in almost all cases, reducing the mean error magnitude for the `Near_5`–`Near_4` radar pair by over 1 cm/s.

Due to the configuration of radar pairs `Near_3`–`Near_1` and `Near_4`–`Near_2`, the ego-motion of the vehicle driving forward aligns with the translation axes of these pairs, which, in theory, should make the calibration data poorly conditioned. However, these radar pairs are mounted on the periphery of the vehicle, so any angular velocity induces unaligned ego-motion measurements. Carrying out calibration on data from pairs `Near_3`–`Near_1` and `Near_4`–`Near_2`, with initial calibration parameters greater than 20° from the Endeavour values, results in estimated parameters that are consistently within 3° of the Endeavour values; this indicates that the problem is not poorly conditioned.

By including a third sensor that is able to measure angular velocity, we can estimate the (metric) scale of the translation between the radars, without requiring the exact extrinsic transform of the third sensor to be known. The magnitude of the angular velocity of a rigid body is the same for all points on the body, allowing us to match the unscaled radar estimate to the angular velocity of the third sensor. For example, assuming that the z-axis of an on-board GNSS receiver is roughly perpendicular to the sensing plane of the radar units, we can apply constant-acceleration smoothing and linear interpolation of the GNSS pose measurements to estimate angular velocity. We tried this approach on the

TABLE II: Identifiability of Olutomilayo et al. [2] for each run and radar pair in the Endeavour dataset. Our algorithm is identifiable for each run and radar pair.

	Data Collection Run					
Radar Pairs	East1	East2	Mid1	Mid2	West1	West2
Near_1–Near_3	✓	✓	✓	✓	✓	✓
Near_2–Near_4	✓	✓	✓	✓	✓	—
Near_3–Near_5	—	✓†	—	—	—	—
Near_4–Near_5	—	✓	—	—	—	—

† This problem is only identifiable using features that appear in less than 5% of measurements.

TABLE III: Endeavour mean velocity error magnitude for each run excluding East2, which was used to estimate the calibration parameters. The errors presented below show how consistent the estimated parameters are when explaining the velocity vector field of a system with a unit moment arm.

	Near_5–Near_1			Near_5–Near_2			Near_5–Near_3			Near_5–Near_4		
Data	Endeavour	Olutomilayo [2]	Ours	Endeavour	Olutomilayo [2]	Ours	Endeavour	Olutomilayo [2]	Ours	Endeavour	Olutomilayo [2]	Ours
Mid1	0.0218	0.0591	**0.0175**	0.0232	0.0242	**0.0173**	0.0185	0.0928	**0.0173**	0.0411	0.0334	**0.0184**
Mid2	0.0215	0.0580	**0.0170**	0.0228	0.0235	**0.0166**	**0.0195**	0.0885	0.0288	0.0289	0.0240	**0.0140**
East1	0.0206	0.0657	**0.0152**	0.0208	0.0212	**0.0157**	0.0152	0.0846	**0.0139**	0.0305	0.0249	**0.0121**
West1	0.0206	0.0574	**0.0175**	0.0247	0.0249	**0.0179**	0.0151	0.1323	**0.0146**	0.0354	0.0285	**0.0149**
West2	0.0210	0.0558	**0.0176**	0.0259	0.0266	**0.0192**	0.0189	0.1320	**0.0178**	0.0515	0.0411	**0.0224**

*All values are in m/s.

Endeavour dataset. After removing measurement pairs with angular velocity magnitudes less than 0.1 rad/s, we computed the translation estimates. Table IV shows that, in most cases, the metric translation values recovered by our algorithm are closer to the ground-truth Endeavour dataset values than those estimated by Olutomilayo et al. [2]. While the sign of the translation may still be positive or negative (i.e., one z-axis could be inverted), this information can be easily determined from a rough model of the system.

VI. Conclusion

In this paper, we presented a 2D radar-to-radar extrinsic calibration algorithm that uses radar ego-velocity data only. We proved that the yaw angle and the axis of translation between the sensors can be identified given sufficient excitation. Using simulations, we demonstrated that our calibration method is robust to varying levels of radar measurement noise and that we are able to improve the raw radar ego-velocity estimates. Finally, we showed, using data from a vehicle, that our algorithm was more reliable and accurate than a state-of-the-art method.

There are multiple potential directions for future research. Our approach could be extended to pairs of 3D radar sensors, similar to those discussed in Wise et al. [4]. Another possibility is to perform temporal calibration using ego-velocity estimates, which could simplify the estimation problem for some systems.

References

[1] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous ego-motion estimation using doppler radar," in *16th Int. IEEE Conf. on Intelligent Transportation Systems (ITSC)*, 2013, pp. 869–874.

[2] K. T. Olutomilayo, M. Bahramgiri, S. Nooshabadi, and D. R. Fuhrmann, "Extrinsic calibration of radar mount position and orientation with multiple target configurations," *IEEE Trans. Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

TABLE IV: Estimated translation magnitude for each method on East2. Radar a is Near_5 for every parameter given. The bolded values are the translation magnitudes closest to the Endeavour parameters.

Method	Near_1	Near_2	Near_3	Near_4
Endeavour	5.68	5.82	0.83	0.86
Olutomilayo [2]	4.87	**6.04**	0.47	0.96
Ours	**6.08**	5.54	**0.77**	**0.95**

*All values are in m.

[3] K. Burnett *et al.*, "Boreas: A multi-season autonomous driving dataset," *Intl. J. Robotics Research (IJRR)*, vol. 42, no. 1-2, pp. 33–42, 2023.

[4] E. Wise, J. Peršić, C. Grebe, I. Petrović, and J. Kelly, "A continuous-time approach for 3D radar-to-camera extrinsic calibration," in *IEEE Intl. Conf. Robotics and Automation (ICRA)*, 2021, pp. 13 164–13 170.

[5] C. Doer and G. F. Trommer, "Radar inertial odometry with online calibration," in *European Navigation Conf. (ENC)*, 2020, pp. 1–10.

[6] S. Sugimoto, H. Tateda, H. Takahashi, and M. Okutomi, "Obstacle detection using millimeter-wave radar and its visualization on image sequence," in *Int. Conf. Pattern Recog. (ICPR)*, 2004, pp. 342–345.

[7] T. Wang, N. Zheng, J. Xin, and Z. Ma, "Integrating millimeter wave radar with a monocular vision sensor for on-road obstacle detection applications," *Sensors*, vol. 11, no. 9, pp. 8992–9008, 2011.

[8] D. Y. Kim and M. Jeon, "Data fusion of radar and image measurements for multi-object tracking via Kalman filtering," *Information Sciences*, vol. 278, pp. 641–652, 2014.

[9] J. Kim, D. S. Han, and B. Senouci, "Radar and vision sensor fusion for object detection in autonomous vehicle surroundings," in *4th Int. Conf. Ubiquitous and Future Networks (ICUFN)*, 2018, pp. 76–78.

[10] G. El Natour, O. Ait Aider, R. Rouveure, F. Berry, and P. Faure, "Radar and vision sensors calibration for outdoor 3D reconstruction," in *IEEE Int. Conf. Robotics and Automation (ICRA)*, 2015, pp. 2084–2089.

[11] J. Domhof, J. F. P. Kooij, and D. M. Gavrila, "An extrinsic calibration tool for radar, camera and lidar," in *IEEE Int. Conf. Robotics and Automation (ICRA)*, 2019, pp. 8107–8113.

[12] J. Peršić, I. Marković, and I. Petrović, "Extrinsic 6DoF calibration of a radar–lidar–camera system enhanced by radar cross section estimates evaluation," *Robotics and Autonomous Systems*, vol. 114, pp. 217–230, 2019.

[13] C. Scholler *et al.*, "Targetless rotational auto-calibration of radar and camera for intelligent transportation systems," in *2019 IEEE Intelligent Transportation Systems Conf. (ITSC)*, 2019, pp. 3934–3941.

[14] J. Peršić, L. Petrović, I. Marković, and I. Petrović, "Online multi-sensor calibration based on moving object tracking," *Advanced Robotics*, vol. 35, no. 3-4, pp. 130–140, 2021.

[15] L. Heng, "Automatic targetless extrinsic calibration of multiple 3D lidars and radars," in *IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (IROS)*, 2020, pp. 10 669–10 675.

[16] C. C. Stahoviak, "An instantaneous 3D ego-velocity measurement algorithm for frequency modulated continuous wave (FMCW) Doppler radar data," Master's thesis, University of Colorado at Boulder, 2019.

[17] D. Kellner, M. Barjenbruch, K. Dietmayer, J. Klappstein, and J. Dickmann, "Joint radar alignment and odometry calibration," in *18th Int. Conf. Information Fusion (FUSION)*, 2015, pp. 366–374.

[18] M. A. Richards, J. A. Scheer, and W. A. Holm, Eds., *Principles of Modern Radar: Basic principles*. Inst. of Eng. and Technol., 2010, vol. 1.

[19] R. Hermann and A. Krener, "Nonlinear controllability and observability," *IEEE Trans. Automatic Control*, vol. 22, no. 5, pp. 728–740, 1977.

Spatiotemporal Calibration of 3D Millimetre-Wavelength Radar-Camera Pairs

Emmett Wise, Qilong Cheng, and Jonathan Kelly

***Abstract*—Autonomous vehicles (AVs) often depend on multiple sensors and sensing modalities to impart a measure of robustness when operating in adverse conditions. Radars and cameras are popular choices for use in combination; although radar measurements are sparse in comparison to camera images, radar scans are able to penetrate fog, rain, and snow. Data from both sensors are typically fused prior to use in downstream perception tasks. However, accurate sensor fusion depends upon knowledge of the spatial transform between the sensors and any temporal misalignment that exists in their measurement times. During the life cycle of an AV, these calibration parameters may change, so the ability to perform in-situ spatiotemporal calibration is essential to ensure reliable long-term operation. State-of-the-art 3D radar-camera spatiotemporal calibration algorithms require bespoke calibration targets that are not readily available in the field. In this paper, we describe an algorithm for *targetless* spatiotemporal calibration that is able to operate without specialized infrastructure. Our approach leverages the ability of the radar unit to measure its own ego-velocity relative to a fixed, external reference frame. We analyze the identifiability of the spatiotemporal calibration problem and determine the motions necessary for calibration. Through a series of simulation studies, we characterize the sensitivity of our algorithm to measurement noise. Finally, we demonstrate accurate calibration for three real-world systems, including a handheld sensor rig and a vehicle-mounted sensor array. Our results show that we are able to match the performance of an existing, target-based method, while calibrating in arbitrary, infrastructure-free environments.**

***Index Terms*—Calibration & Identification, Sensor Fusion, Robot Sensing Systems, Radar, Computer Vision**

I. Introduction

The widespread deployment of autonomous vehicles (AVs) depends critically on their ability to operate safely under a range of challenging environmental conditions. To ensure sufficient redundancy, most AV perception systems incorporate multiple sensors and sensing modalities. In this paper, we consider 3D mm-wavelength radar as a complementary sensor to standard cameras for safe AV perception.

The operating principle of mm-wavelength radars (i.e., the active emission of mm-wavelength electromagnetic (EM) radiation) makes these sensors relatively immune to adverse conditions that negatively affect cameras. Radars also provide information that cameras do not, including *range-rate* measurements of the relative velocity of targets in the environment. However, radar data are much lower resolution and significantly more noisy, than visual measurements under nominal conditions. Together, radars and cameras are highly complementary, providing situational awareness under both nominal and visually-degraded conditions.

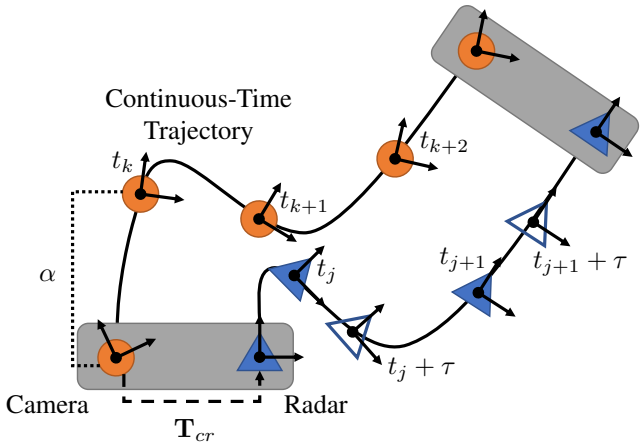


Fig. 1: The radar (triangle) and camera (circle) are assumed to be rigidly connected. Our calibration problem involves estimating the transform between the camera and radar, $\mathbf{T}_{cr}$, the translation scale factor, α, for the camera pose measurements, and the temporal offset, τ. The unfilled triangles represent radar measurements at "shifted" points in time due to the offset bias, which must be considered to ensure the correct radar ego-velocity estimate.

To be used jointly in the AV perception stack, radar and camera sensors must be calibrated with respect to each other. The spatial (6-DoF) transform between a radar-camera pair must be known accurately in order to express their data in a common reference frame. An AV may undergo calibration 'at the factory' prior to operation, but maintenance and general wear and tear can alter the spatial calibration parameters. Further performance gains are achieved when the sensor data streams are temporally aligned in addition to spatial alignment. Even when the sensors are externally triggered, internal signal processing delays can result in shifted measurement timestamps. If this time offset is not accounted for, then, for example, moving targets will be *spatially* shifted in the radar and camera data. Further, in some systems, power cycling or reconfiguring the sensors may change the time offset. In turn, temporal calibration may need to be performed routinely to ensure the accuracy and integrity of fused sensor measurements.

An in-situ method to estimate the spatial transformation

Emmett Wise, Qilong Cheng, and Jonathan Kelly are with the Space and Terrestrial Autonomous Robotic Systems Laboratory, University of Toronto, Institute for Aerospace Studies, Toronto, Canada. {<first name>.<last name>@robotics.utias.utoronto.ca}

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC). Jonathan Kelly was supported by the Canada Research Chairs Program. Jonathan Kelly is a Vector Institute for Artificial Intelligence Faculty Affiliate.

from the radar to the camera and the temporal offset of the sensor data streams, would enable long-term AV operation in the field. However, existing radar-camera spatial and spatiotemporal calibration algorithms are restricted to certain environments and sensor configurations [1], [2]. Primarily, these methods rely on the assumption that the radar measures 'point-like' reflections from objects. In general, a radar measurement, determined from a reflected EM pulse, is a complex function of the shape, relative orientation, size, and composition of an object [3]. Additionally, multipath reflections introduce outlier measurements of ghost 'objects' [3]. To avoid these problems, specialized trihedral retroreflective radar targets are used to produce the desired point-like radar returns. A visual fiducial can be placed over or alongside a trihedral target, allowing radar-camera measurement correspondences to be established. The use of targets, however, means that calibration must be carried out in specialized areas or with infrastructure that is not usually available during regular AV operation. Additionally, these algorithms require that the radar-camera pair(s) share overlapping fields of view, which may not be possible for all radar-camera systems.

Herein, we extend the method in [4] to jointly estimate the extrinsic calibration parameters and temporal offset of a 3D mm-wavelength radar-camera pair in a fully targetless manner. Importantly, our approach does not require the sensors to share overlapping fields of view. Instead, we use radar measurements to estimate the instantaneous radar ego-velocity — that is, the velocity of the radar unit relative to an external reference frame expressed in the radar reference frame [5]. By relying on velocity information, we remove the need for specialized calibration targets while also avoiding the difficult problems of radar and cross-modal data association. We make the following contributions:

- we extend the work in [4] to enable full spatiotemporal calibration of monocular camera-3D radar pairs in arbitrary configurations;
- we prove that the calibration problem is identifiable and determine the motions that are required for reliable calibration;
- we analyze the accuracy of spatiotemporal calibration with varying amounts of sensor noise through an extensive series of simulation studies; and
- we carry out three different real-world experiments, which demonstrate that our algorithm is able to match the accuracy of an existing, target-based method and that we are able to perform calibration in different environments, including for sensors on board an AV.

In Section II, we survey existing extrinsic and spatiotemporal calibration algorithms for mm-wavelength radar sensors. Section III formulates spatiotemporal calibration as a batch, continuous-time estimation problem. We examine the identifiability of the calibration problem in Section IV. In Section V, we describe two simulation experiments designed to evaluate the robustness of our algorithm. In Section VI, we demonstrate the accuracy and flexibility of our algorithm by reporting on three real-world experiments in different environments. Finally, we summarize our work in Section VII.

II. Related Work

In this section, we survey spatial and spatiotemporal calibration algorithms that can be applied to mm-wavelength radars in relation to another, complementary sensor. Section II-A reviews algorithms for target-based extrinsic calibration, while Section II-B describes algorithms for target-free, or *targetless*, extrinsic calibration. In Section II-C, we discuss prior work on target-based spatiotemporal calibration.

A. Target-Based Extrinsic Calibration

Early radar extrinsic calibration algorithms, developed prior to the widespread availability of 3D mm-wavelength radar units, were designed to enable 2D radar-camera data fusion. Many of these early extrinsic calibration techniques operate by computing the projective homography that maps points on the horizontal (sensing) radar plane to points on the camera image plane. Because radar sensors are inherently noisy, most calibration algorithms require specialized trihedral reflectors, shown in Fig. 8, that produce coincident, point-like 'signals' in both the radar and camera data, making the correspondence problem easier to solve [6]–[9]. Although 2D radar sensors are not able to properly measure the elevation of remote targets, they do detect targets at a small elevation angle above the radar horizontal plane. Since the distance to off-plane targets will be slightly different, accurate calibration requires that detected reflectors *do* lie on the radar horizontal plane. Sugimoto et al. [6] constrain the trihedral reflector position using the radar return signal strength. During calibration, the approach in [6] filters radar-camera measurement pairs by return intensity; the intensity is maximal for reflectors that lie on the horizontal plane.

More recent 2D radar extrinsic calibration algorithms often minimize a type of 'reprojection error,' that is, the error in the alignment of identifiable objects that appear within both sensors' fields of view. Kim et al. [10] leverage reprojection error to estimate the radar-to-camera transform but assume that radar measurements are strictly constrained to the zero-elevation plane. El Natour et al. [11] determine the radar-to-camera transform by intersecting backprojected camera rays with the 3D 'arcs' along which the 2D radar measurements must lie. Domhof et al. [12] use a specialized calibration target that provides scale for the camera measurement, enabling extrinsic calibration via point cloud alignment. Peršić et al. [13] also perform extrinsic calibration via 3D point cloud alignment but improve overall accuracy by modelling the relationship between target return intensity and elevation angle. The 'homography' and 'reprojection' methods are summarized and compared by Oh et al. in [14], where the authors conclude that both have similar performance. Due to the infrastructure needs (i.e., specialized targets), the methods above are restricted to sensor pairs that share overlapping fields of view. This requirement may be impossible to satisfy for certain sensor configurations. By leveraging constraints induced by the motion of a rigidly-connected radar-camera pair, we are able to calibrate sensors that do not share overlapping fields of view. Further, our approach operates without added infrastructure, enabling calibration under a wider range of conditions.

B. Target-Free Extrinsic Calibration

Some extrinsic calibration algorithms do not require specialized retroreflective targets. Schöller et al. [15] train a neural network end-to-end to regress a rotation correction from raw camera images and radar data, for example. Peršić et al. [16] estimate the yaw angles (only) between radar, camera, and lidar sensors by aligning the trajectories of tracked objects. Both of these methods require manual measurement of the translation parameters and overlapping fields of view.

Heng [17] presents the first reprojection error-based 3D radar-lidar extrinsic calibration algorithm that does not require specialized targets or overlapping sensor fields of view. The approach in [17] estimates the extrinsic calibration between several lidar units and, using a known vehicle trajectory, constructs a 3D point cloud map. The radar-lidar extrinsic calibration parameters are then determined by minimizing two weighted residuals: i) the distance from the radar point measurements to the closest plane in the lidar map and ii) the radial velocity error. However, this method requires the construction of a dense lidar map and known vehicle poses.

Instead of using feature positions, a subset of extrinsic calibration algorithms fuse ego-velocity and ego-motion measurements from the radar and second sensor, respectively. Since the motion of each sensor is estimated separately, these methods do not perform radar or cross-modal data association and are inherently 'target-free.' Kellner et al. [18] estimate the rotation between a car-mounted 2D radar and an inertial measurement unit (IMU) by minimizing the difference in estimated lateral velocities expressed in the radar frame. While the radar ego-velocity measurements provide lateral velocity directly, determining the lateral velocity of the radar from IMU measurements requires both the IMU angular velocity and accurate knowledge of the radar-IMU translation. Doer et al. [19] extend the approach in [18] to estimate the full extrinsic calibration for a 3D radar-IMU pair. Their method is able to achieve a spatial calibration accuracy of 5 cm and 5° when using simulated, low-noise (our designation, see Section V) radar measurements. Wise et al. [4] perform extrinsic calibration in continuous time using instantaneous radar ego-velocity measurements and camera egomotion measurements. Given an unknown but fixed temporal offset, the spatial calibration parameters estimated by this method are within 3 cm and 1°, per axis, of those determined by [2]. All of these techniques rely on ad-hoc temporal calibration schemes. Herein, we incorporate a principled temporal calibration method.

C. Target-Based Spatiotemporal Calibration

To date, only two radar spatiotemporal calibration algorithms have appeared in the literature, by Lee et al. [1] and by Peršić et al. [2]. The algorithm in [1] first calibrates the 2D radar-lidar spatial transform using the method of Peršić et al. [13]. As a second step, the lidar measurements are expressed in the radar reference frame, and the azimuth error to distant targets is minimized to determine the temporal offset between the sensor data streams. Peršić et al. [2] represent the trajectory of a target moving through the fields of view of multiple sensors using a continuous-time Gaussian process model. This representation allows their algorithm to estimate the spatiotemporal calibration parameters by aligning the sensors' trajectories. In general, jointly determining all parameters as part of one maximum likelihood estimation problem yields superior accuracy [2], [20]. Notably, since the methods in [1] and [2] rely on known targets, they have the same limitations as the methods discussed in Section II-A.

III. Methodology

We formulate radar-to-camera spatiotemporal calibration as a continuous-time batch estimation problem. In Section III-A, we describe the mathematical notation used throughout the paper. We choose to parameterize the smooth radar-camera trajectories using continuous-time B-splines; we review the properties of this representation in Section III-B. In Section III-C, we derive our radar and camera measurement models. With the necessary preliminaries in place, we then define the full estimation problem in Section III-D.

A. Notation

Latin and Greek letters (e.g., a and α) denote scalar variables, while boldface lower- and uppercase letters (e.g., $\mathbf{x}$ and $\boldsymbol{\Theta}$) denote vectors and matrices, respectively. A parenthesized superscript pair, for example, $\mathbf{A}^{(i,j)}$, indicates the ith row and the jth column of the matrix $\mathbf{A}$. A three-dimensional reference frame is designated by $\underrightarrow{\mathcal{F}}$. The translation vector from point a (often a reference frame origin) to b, expressed in $\underrightarrow{\mathcal{F}}_a$, is denoted by $\mathbf{r}_a^{ba}$. The translational velocity vector of point b relative to point a, expressed in $\underrightarrow{\mathcal{F}}_a$, is denoted by $\mathbf{v}_a^{ba}$. The angular velocity of frame $\underrightarrow{\mathcal{F}}_a$ relative to a frame $\underrightarrow{\mathcal{F}}_i$, expressed in $\underrightarrow{\mathcal{F}}_a$, is denoted by $\boldsymbol{\omega}_a^{ai}$.

We denote rotation matrices by $\mathbf{R}$. For example, $\mathbf{R}_{ab} \in \mathrm{SO}(3)$ defines the rotation from $\underrightarrow{\mathcal{F}}_b$ to $\underrightarrow{\mathcal{F}}_a$. We reserve $\mathbf{T}$ for $\mathrm{SE}(3)$ transformation matrices. For example, $\mathbf{T}_{ab}$ is the 4×4 homogeneous matrix that defines the rigid-body transform from frame $\underrightarrow{\mathcal{F}}_b$ to $\underrightarrow{\mathcal{F}}_a$. Our $\mathrm{SE}(3)$ matrix entries will generally be functions of time; we denote the transform from frame $\underrightarrow{\mathcal{F}}_b$ to $\underrightarrow{\mathcal{F}}_a$ at time t by

$$\mathbf{T}_{ab}(t) = \begin{bmatrix} \mathbf{R}_{ab}(t) & \mathbf{r}_a^{ba}(t) \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{1}$$

where $\mathbf{R}_{ab}(t) \in \mathrm{SO}(3)$ and $\mathbf{r}_a^{ba}(t) \in \mathbb{R}^3$. We use $\mathbf{I}_n$ to denote the n-by-n identity matrix.

The unary operator $^\wedge$ acts on $\mathbf{r} \in \mathbb{R}^3$ to produce a skew-symmetric matrix such that $\mathbf{r}^\wedge \mathbf{s}$ is equivalent to the cross product $\mathbf{r} \times \mathbf{s}$. The operators $\exp(\cdot)$ and $\log(\cdot)$ map from the Lie algebra $\mathfrak{so}(3)$ to the Lie group $\mathrm{SO}(3)$ and vice versa, respectively [21].

B. Continuous-Time Trajectory Representation

Temporal calibration is most easily formulated as a continuous-time problem, in part because the batch optimization procedure incrementally time-shifts the measurements from one sensor. In turn, we require the ability to query the pose of the radar or the camera at arbitrary points in time.

To enable this, we parameterize the trajectory of the radar-camera pair using the B-spline representation from Sommer et al. [22]. This representation is briefly reviewed below. We refer readers to Sommer et al. [22], de Boor [23], and Qin [24] for additional details.

A B-spline of order k is a function of one continuous parameter (e.g., time) and a finite set of control points; for brevity, we restrict our example here to control points $\{\mathbf{p}_0, \ldots, \mathbf{p}_N \mid \mathbf{p}_i \in \mathbb{R}^d\}$. In a uniformly-spaced B-spline, each control point is assigned a time (or *knot*) $t_i = t_0 + i\Delta t$, where t_0 marks the beginning of the spline and Δt is the time between knots. Evaluating a k^{th} order B-spline at time t, where $t_i \leq t < t_{i+1}$, requires the set of k control points over the knot sequence $t_i, \ldots, t_{i+k-1}$. As a result, the end point of a B-spline of length N and order k is at time t_{N-k+1}.

The first step in computing the value of a k^{th} order B-spline at time t is to convert t to the 'normalized' time $u = \frac{t-t_i}{t_{i+1}-t_i}$. Given u, the value of the k^{th} order B-spline is defined as

$$\mathbf{p}(u) = \begin{bmatrix}\mathbf{p}_i & \mathbf{d}_1^i & \ldots & \mathbf{d}_{k-1}^i\end{bmatrix} \tilde{\mathbf{M}}_k \mathbf{u}, \tag{2}$$

where $\mathbf{u}^T = [1 \; u \; u^2 \; \ldots \; u^{k-1}]$ and $\mathbf{d}_j^i = \mathbf{p}_{i+j} - \mathbf{p}_{i+j-1}$. The elements of the $k \times k$ mixing matrix, $\tilde{\mathbf{M}}_k$, are defined by,

$$\tilde{\mathbf{M}}_k^{(a,n)} = \sum_{s=a}^{k-1} m_k^{(s,n)}, \tag{3}$$

$$m_k^{(s,n)} = \frac{C_{k-1}^n}{(k-1)!} \sum_{l=s}^{k-1} (-1)^{l-s} C_k^{l-s} (k-1-l)^{k-1-n} \tag{4}$$
$$a, s, n \in \{0, \ldots, k-1\},$$

where scalar $C_j^i = \frac{j!}{i!(j-i)!}$. Substituting $\boldsymbol{\lambda}(u) = \tilde{\mathbf{M}}_k \mathbf{u}$ into Equation (2) results in

$$\mathbf{p}(u) = \mathbf{p}_i + \sum_{j=1}^{k-1} \boldsymbol{\lambda}_j(u) \mathbf{d}_j^i. \tag{5}$$

Equation (5) can describe the smooth translation of a rigid-body in continuous time (see Figure 10 in Section VI for an example).

While our development above focuses on vector space splines, B-splines can also be defined on Lie groups, including the group $\mathrm{SO}(3)$ of rotations,

$$\mathbf{R}(u) = \mathbf{R}_i \prod_{j=1}^{k-1} \exp(\lambda_j(u) \boldsymbol{\phi}_j^i), \tag{6}$$

where $\mathbf{R}_i$ is a control point of the rotation spline and $\boldsymbol{\phi}_j^i = \log(\mathbf{R}_{i+j-1}^T \mathbf{R}_{i+j})$. We use two B-splines, one on $\mathrm{SO}(3)$ and one on $\mathbb{R}^3$, as our complete continuous-time representation of the radar-camera trajectory.

C. Sensor Measurement Models

In order to perform spatiotemporal calibration, we require a measurement model for the radar unit. Radars emit EM waves that reflect off of surfaces in the environment. Due to the relatively long EM wavelength used by radars, the reflected "location" of a target can vary based on the relative orientation between the radar and target [3]. Additionally, multipath reflections can occur when the wave bounces off of multiple surfaces before returning to the radar receiver, which can bias measurements of targets and introduce false detections [3].

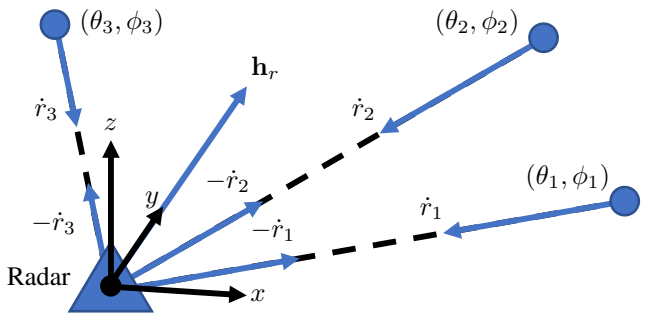


Fig. 2: Illustration of our radar measurement model. The radar EM wave reflects off of three (or more) non-collinear, stationary landmarks in the environment, yielding azimuth, elevation, and range-rate measurements to each landmark. Using these data, we estimate the radar velocity relative to the world reference frame expressed in the radar reference frame.

For each received reflection l from an environmental feature (i.e., an object that we identify as a *landmark*), the radar measures the range r_l, azimuth θ_l, elevation ϕ_l, and range-rate $\dot{r}_l$. We assume that the observed landmarks are stationary with respect to a world frame, $\underrightarrow{\mathcal{F}}_w$; measurements are resolved in the radar reference frame, $\underrightarrow{\mathcal{F}}_r$. The range-rate measurement is the dot product between the velocity of the radar unit itself, $\mathbf{h}_r \in \mathbb{R}^3$, and the unit vector $\hat{\mathbf{r}}_l \in \mathrm{S}^2$ defined by θ_l and ϕ_l. Given radar measurements to at least three non-collinear, stationary landmarks, the unit direction vectors and their associated range-rates can be used to reconstruct the radar velocity, $\mathbf{h}_r$, as shown in Figure 2.

Stahoviak [5] and Doer et al. [25] demonstrate that, given $N > 3$ landmarks, one can estimate the ego-velocity of the radar by solving the over-constrained linear least-squares problem

$$\mathbf{h}_r^\star = \min_{\mathbf{h}_r} \; \mathbf{e}_{ego}{}^T \mathbf{e}_{ego}, \tag{7}$$

where

$$\mathbf{e}_{ego} = \mathbf{H}\mathbf{x} - \mathbf{y} = \begin{bmatrix}\hat{\mathbf{r}}_0{}^T \\ \vdots \\ \hat{\mathbf{r}}_N{}^T\end{bmatrix} \mathbf{h}_r - \begin{bmatrix}\dot{r}_0 \\ \vdots \\ \dot{r}_N\end{bmatrix}. \tag{8}$$

Equation (8) has its specific form because we wish to estimate the velocity of the radar with respect to the static world frame — not vice versa. The estimated ego-velocity covariance is

$$\boldsymbol{\Sigma}_v = \frac{(\mathbf{e}_{ego}{}^T \mathbf{e}_{ego})(\mathbf{H}^T \mathbf{H})}{N-3}. \tag{9}$$

We use RANSAC [5], [25] and radar cross-section thresholding to remove outliers. The two main sources of outliers are targets that move relative to the inertial reference frame and spurious multipath reflections. Empirically, RANSAC successfully eliminates outliers from these two sources if the range-rates to the targets deviate significantly from other stationary landmarks that are close in terms of angle. However, there are two subtle cases where multipath returns may appear to

be valid measurements of stationary landmarks. In the first case, the difference between the transmission and return angles is small. In the second case, the transmission and return angles are symmetric about the radar ego-velocity direction (see Section 8.9 in [3]). The returns from these reflections have a small radar cross-section and are rejected by cross-section thresholding.

Leveraging our continuous-time trajectory representation, the measurement model for the radar ego-velocity at time t_j is

$$\begin{aligned}\mathbf{h}_{r_j} &= -\dot{\mathbf{r}}_r^{wr}(t_j+\tau) - \boldsymbol{\omega}_r^{rw}(t_j+\tau)^{\wedge}\mathbf{r}_r^{wr}(t_j+\tau) \\ &\quad + \mathbf{n}_{v_j}, \\ \mathbf{n}_{v_j} &\sim \mathcal{N}\left(\mathbf{0}_{3\times 1}, \boldsymbol{\Sigma}_{v_j}\right),\end{aligned} \tag{10}$$

where τ is the temporal offset of the radar measurements relative to the camera measurements and $\mathbf{n}_{v_j}$ is the radar velocity measurement noise term. We assume that the noise is a zero-mean Gaussian with covariance matrix $\boldsymbol{\Sigma}_{v_j}$ (see Equation (9)). From the radar ego-velocity model, the error residual is

$$\begin{aligned}\mathbf{e}_{v_j} = \mathbf{h}_{r_j} + \dot{\mathbf{r}}_r^{wr}(t_j+\tau) + \\ \boldsymbol{\omega}_r^{rw}(t_j+\tau)^{\wedge}\mathbf{r}_r^{wr}(t_j+\tau) - \mathbf{n}_{v_j}.\end{aligned} \tag{11}$$

To estimate the ego-motion of the camera, we use a monocular simultaneous localisation and mapping (SLAM) algorithm that operates independently of the radar. By observing fixed landmarks in the environment, monocular SLAM is capable of determining the transformation between the camera reference frame, $\underrightarrow{\mathcal{F}}_c$, and the world frame, $\underrightarrow{\mathcal{F}}_w$, up to an unknown scale factor α [26]. Our (scaled) camera pose measurement model is given by

$$\begin{aligned}\mathbf{R}_{cw,t_k} &= \exp(\mathbf{n}_{r,k})\mathbf{R}_{cr}\,\mathbf{R}_{wr}(t_k)^T, \\ \mathbf{n}_{r,k} &\sim \mathcal{N}\left(\mathbf{0}_{3\times 1}, \boldsymbol{\Sigma}_r\right),\end{aligned} \tag{12}$$

$$\begin{aligned}\mathbf{r}_{c,t_k}^{wc} &= \alpha(\mathbf{R}_{cr}\mathbf{r}_r^{wr}(t_k) + \mathbf{r}_c^{rc}) + \mathbf{n}_{t,k}, \\ \mathbf{n}_{t,k} &\sim \mathcal{N}\left(\mathbf{0}_{3\times 1}, \boldsymbol{\Sigma}_t\right),\end{aligned} \tag{13}$$

where $\mathbf{n}_{r,k}$ and $\mathbf{n}_{t,k}$ are zero-mean Gaussian noise terms of covariances matrice $\boldsymbol{\Sigma}_r$ and $\boldsymbol{\Sigma}_t$ for the camera rotation and translation measurements. The resulting error equations are

$$\mathbf{e}_{r,t_k} = \log(\mathbf{R}_{cw,t_k}\mathbf{R}_{wr}(t_k)\,\mathbf{R}_{cr}{}^T), \tag{14}$$

$$\mathbf{e}_{t,t_k} = \mathbf{r}_{c,t_k}^{wc} - \alpha(\mathbf{R}_{cr}\mathbf{r}_r^{wr}(t_k) + \mathbf{r}_c^{rc}) - \mathbf{n}_{t,k}. \tag{15}$$

We note that a monocular visual odometry (VO) algorithm (i.e., localization without loop closure) can provide camera ego-motion estimates, but visual drift will bias these estimates and decrease calibration accuracy. Also, the measured radar ego-velocity is a local property of a trajectory and cannot fully correct for pose errors induced by visual drift.

D. The Spatiotemporal Calibration Problem

The set of parameters, $\mathbf{x}$, that we wish to estimate are the spline control points ($\mathbf{r}_{0\ldots N} \in \mathbb{R}^3$, $\mathbf{R}_{0\ldots N} \in \mathrm{SO}(3)$), the extrinsic calibration parameters ($\mathbf{R}_{cr}, \mathbf{r}_c^{rc}$), the camera translation scale factor (α), and the temporal offset (τ),

$$\begin{aligned}\mathbf{x} = \big\{&\mathbf{r}_0, \quad \ldots, \quad \mathbf{r}_N, \quad \mathbf{R}_0, \quad \ldots, \quad \mathbf{R}_N, \\ &\mathbf{R}_{cr}, \quad \mathbf{r}_c^{rc}, \quad \alpha, \quad \tau\big\}.\end{aligned} \tag{16}$$

Given N_r radar measurements and N_c camera measurements, we minimize the following cost function,

$$\begin{aligned}\mathbf{x}^{\star} = \min_{\mathbf{x}} &\sum_{j=1}^{N_r} \mathbf{e}_{v_j}{}^T \boldsymbol{\Sigma}_{v_j}^{-1}\mathbf{e}_{v_j} + \\ &\sum_{k=1}^{N_c} \mathbf{e}_{r,t_k}{}^T \boldsymbol{\Sigma}_r^{-1}\mathbf{e}_{r,t_k} + \mathbf{e}_{t,t_k}{}^T \boldsymbol{\Sigma}_t^{-1}\mathbf{e}_{t,t_k}.\end{aligned} \tag{17}$$

We perform this minimization using the Ceres solver, a standard nonlinear least squares solver [27]. The ability to calibrate all of the relevant parameters depends upon the identifiability of problem, which we discuss in the next section.

IV. Identifiability

In this section, we show that the calibration problem is identifiable given sufficient excitation of the radar-camera system. Our approach is to determine the observability, or 'instantaneous identifiability,' of the system at several different points in time, assuming that the system follows a varying trajectory. We consider local identifiability (cf. locally weak observability) along a trajectory segment in Section IV-B, after introducing the requisite observability rank condition in Section IV-A. A similar approach has been taken in [28] and [29] and elsewhere. In Section IV-C, we describe several 'degenerate' motions for which the identifiability condition does not hold. We leave the complete characterization of the set of unidentifiable trajectories as future work.

A. The Observability Rank Condition

We make use of the criterion from Hermann and Krener [30] as part of our identifiability analysis. A system S, written in control-affine form as

$$S\begin{cases}\dot{\mathbf{x}} = \mathbf{f}_0(\mathbf{x}) + \sum_{j=1}^{p}\mathbf{f}_j(\mathbf{x})u_j \\ \mathbf{y} = \mathbf{h}(\mathbf{x})\end{cases}, \tag{18}$$

with the drift vector field $\mathbf{f}_0(\mathbf{x})$ and control inputs u_j (for $j = 1, \ldots, p$), is locally weakly observable if the matrix $\mathbf{O}$ of the gradients of the Lie derivatives with respect to the system state has full column rank.

The Lie derivative, or directional derivative, of a smooth scalar function h with respect to the smooth vector field $\mathbf{f}$ at the point $\mathbf{x}$ is

$$L_{\mathbf{f}}h(\mathbf{x}) = \nabla_{\mathbf{f}}h(\mathbf{x}) = \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}). \tag{19}$$

The n^{th} Lie derivative of h with respect to $\mathbf{x}$ along $\mathbf{f}$ is defined recursively as

$$L_{\mathbf{f}}^n h(\mathbf{x}) = \frac{\partial L_{\mathbf{f}}^{n-1}h(\mathbf{x})}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}), \tag{20}$$

where $L^0 h(\mathbf{x}) = h(\mathbf{x})$. We note that the matrix $\mathbf{O}$ has an infinite number of rows, but it is sufficient to show that a finite subset of rows yield a matrix of full column rank.

B. Identifiability of Radar-Camera Calibration

We begin by simplifying the state (and parameter) vector that we aim to estimate. We are able to measure the camera

pose up to scale [26] and the radar velocity in the radar frame [5]. Since we are working in continuous time (or, roughly equivalently, if there are a sufficient number of closely-spaced radar and camera measurements), then the scaled velocity of the camera in the camera reference frame $\alpha \mathbf{v}_c^{cw}(t_i)$, the angular velocity of the camera $\boldsymbol{\omega}_c(t_i)$ in the camera frame, the radar velocity $\mathbf{v}_r^{rw}(t_i+\tau)$ in the radar frame, and the time derivative of the radar velocity $\dot{\mathbf{v}}(t_i+\tau)$ in the radar frame are all available. For the purposes of identifiability, we are able to define the following, modified measurement model,

$$\mathbf{h}(t_i) = \alpha(\mathbf{R}_{cr}\mathbf{v}_r^{rw}(t_i+\tau) - \boldsymbol{\omega}_c(t_i)^\wedge \mathbf{r}_c^{rc}), \tag{21}$$

where $\mathbf{h}(t_i)$ is the scaled linear velocity of the camera ($\mathbf{v}_c^{cw}$) and $\boldsymbol{\omega}_c$ is the angular velocity of the camera, both relative to the camera frame. This modified measurement model does not directly rely on the pose of the radar, thus simplifying the set of parameters that we wish to determine to

$$\tilde{\mathbf{x}} = \{\mathbf{r}_c^{rc}, \quad \mathbf{R}_{cr}, \quad \alpha, \quad \tau\}. \tag{22}$$

To decrease the notational burden, we drop the superscripts and subscripts defining the velocities and extrinsic transform parameters. The gradient of the zeroth-order Lie derivative of the ith measurement is

$$\begin{aligned}\nabla_{\tilde{\mathbf{x}}} L_0 \mathbf{h}(t_i) = \big[&-\alpha\boldsymbol{\omega}(t_i)^\wedge \quad -\alpha(\mathbf{R}\mathbf{v}(t_i+\tau))^\wedge \mathbf{J} \\ &\mathbf{R}\mathbf{v}(t_i+\tau) - \boldsymbol{\omega}(t_i)^\wedge \mathbf{r} \quad \alpha\mathbf{R}\dot{\mathbf{v}}(t_i+\tau)\big],\end{aligned} \tag{23}$$

where $\mathbf{J}$ is the Lie algebra left Jacobian of $\mathbf{R}_{cr}$ [21]. Since the parameters of interest are constant with respect to time, we are able to stack the gradients of several Lie derivatives (at different points in time) to form the observability matrix,

$$\mathbf{O} = \begin{bmatrix} \nabla_{\tilde{\mathbf{x}}} L_0 \mathbf{h}(t_1) \\ \nabla_{\tilde{\mathbf{x}}} L_0 \mathbf{h}(t_2) \\ \nabla_{\tilde{\mathbf{x}}} L_0 \mathbf{h}(t_3) \end{bmatrix}, \tag{24}$$

which, using block Gaussian elimination, can be shown to have full column rank when three or more sets of measurements are available.[1]

Two comments regarding the analysis are in order. First, we note that the analysis is simplified by considering the modified measurement equation only, which avoids the use of higher-order Lie derivatives. Second, there is a subtlety involved in stacking the gradients of the Lie derivatives at different points in time. The modified measurement equation depends upon the time derivatives of the camera pose and the radar ego-velocity—this implies that, although we do not consider specific control inputs, the system dynamics must be non-null. Stated differently, varied motion of the radar-camera pair is necessary to ensure identifiability; we discuss this requirement further in the next section. Further, it is worth noting that the the observation times must span the temporal offset period [31].

C. Degenerate Motions

There are motions that cause the matrix $\mathbf{O}$ in Equation (24) to lose full column rank. First, the Lie derivatives include linear and rotational velocities and accelerations, so the matrix will lose full column rank when the system is stationary with respect to the world frame or moving with constant linear or angular velocity. Second, in Wise et al. [4], we showed that the system must undergo rotation about two nonparallel axes in order for the observability matrix to be full rank. This requirement also applies to the present analysis. To show this, we can align the angular velocity and angular acceleration vectors by substituting $\boldsymbol{\alpha}_c^{ci} = \eta\, \boldsymbol{\omega}_c^{cw}$, where η is an arbitrary constant, into Equation (41). This substitution is equivalent to asserting that the system rotates about one axis only, resulting in an observability matrix that is rank-deficient.

[1] We omit the full derivation for brevity, and note that the rank condition can be verified in this case using any symbolic algebra package.

V. Simulation Studies

In order to test the robustness of our algorithm to measurement noise, we carried out a series of simulation studies. We generated two simulated camera-radar datasets using two different trajectories and with varying amounts of (synthetic) noise (see Figures 3 and 4). The nominal (noise-free) trajectories were selected to ensure sufficient excitation of the camera-radar pair. The median linear and rotational velocities for the trajectory shown in Figure 3 were, respectively, higher and lower than the velocities for the trajectory shown in Figure 4.

After constructing the trajectories, we computed the simulated radar ego-velocity and camera pose measurements. Since radar measurements are antenna configuration- and environment-specific, these measurements were not generated at the EM propagation level. Radar ego-velocity measurements (i.e., $\mathbf{h}_{r_k}$) were computed using the known linear and rotational velocities defined by the trajectory. Consequently, our simulated radar measurements generalize to any radar and environment that produce an unbiased 3D ego-velocity estimate. Simulated camera pose measurements (i.e., $\mathbf{R}_{cw,t_k}$ and $\mathbf{t}_{c,t_k}^{wc}$) were derived from simulated observations of a series of landmark points, arranged in a 2D grid. This configuration of points matches the configuration of a standard 'checkerboard' camera calibration target. In the targetless setting, we can only estimate the position of the camera up to an unknown scale [26], so the checkerboard tracking algorithm is given an incorrect size for the checkerboard squares.

For each individual simulation, we added zero-mean Gaussian noise to the radar ego-velocity measurements ($\boldsymbol{\Sigma}_{v_j} = \sigma_r^2\, \mathbf{I}_{3\times 3}$) and to the camera measurements of the checkerboard corners on the simulated image plane ($\boldsymbol{\Sigma}_{p_k} = \sigma_c^2\, \mathbf{I}_{2\times 2}$). In our experiments, we adjusted the radar ego-velocity noise standard deviation (σ_r) between 0.05 m/s and 0.15 m/s. Based on our real-world experiments (see Section VI), we have found that the radar ego-velocity measurement noise is closer to the lower end of this range, unless the environment is sparse and too few valid radar returns are captured. We adjusted the standard deviation of the noise added to the measured checkerboard corner coordinates (σ_c) between 0.2 and 0.4 pixels; these noise levels are similar to the observed noise in our real-world experiments [4].

The error distributions for the spatial calibration parameter estimates ($\mathbf{R}_{cr}, \mathbf{r}_c^{rc}$), scale factor ($\alpha$), and temporal offset ($\tau$) are shown in Figures 5 and 6, across 100 simulation trials

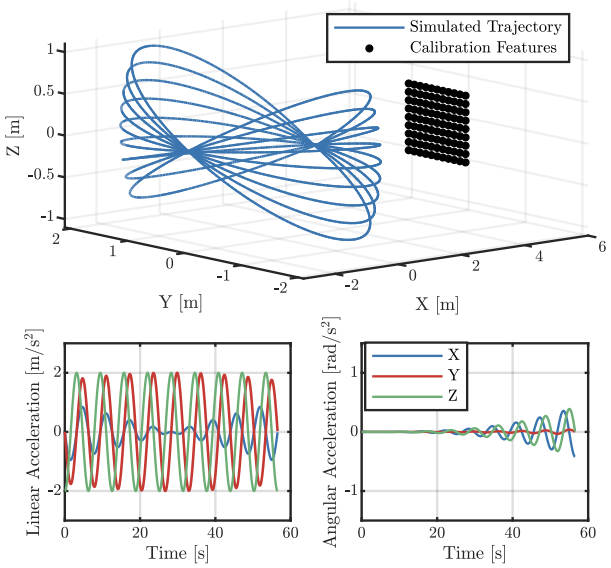


Fig. 3: High linear and low angular velocity trajectory (top) for the simulation experiments, with associated linear (bottom left) and angular acceleration (lower right) plots.

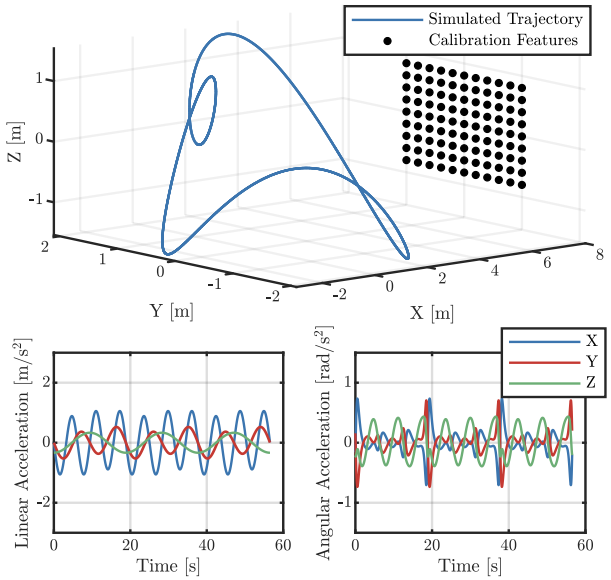


Fig. 4: Low linear and high angular velocity trajectory (top) for the simulation experiments, with associated linear (bottom left) and angular acceleration (lower right) plots.

for each (nominal) trajectory. For the high linear and low angular velocity trajectory, even in the high-noise regime, the error in the rotation and scale estimates remains less than two degrees and one percent, respectively. However, high levels of noise in the radar ego-velocity measurements result in substantially larger (and more widely distributed) errors in the estimate of the relative translation of the sensors and of the temporal offset; the errors can be as large as 15 cm and 30 ms, respectively. This sensitivity indicates that, prior to use in our algorithm, the radar data should be filtered to remove high-noise measurements whenever possible.

If the system follows the low linear and high angular velocity trajectory in Figure 4, then radar data filtering may not be necessary. As shown in Figure 6, calibrating the radar along this trajectory results in similar scale and rotation estimation accuracy as for the other trajectory, but drastically improves the translation and temporal offset estimates; the errors are within 10 cm and 10 ms, respectively. Additionally, our algorithm achieves a comparable spatial calibration accuracy to Doer et al. [19] on the noisier radar ego-velocity data. However, the high angular velocity trajectory is challenging for real-world camera localization and the amount of excitation is not necessary if the radar data are sufficiently accurate.

VI. Real-World Experiments

To verify the performance and accuracy of our algorithm, we carried out a series of real-world experiments involving three different radar-camera systems. We discuss the various systems and their implementation details in Section VI-A. In Section VI-B, we show that the set of spatiotemporal calibration parameters estimated by our algorithm have a similar level of alignment accuracy as the parameters estimated by the target-based method of Peršić et al. [2]. In Section VI-C, we demonstrate how spatiotemporal calibration can improve the performance of camera-radar-IMU odometry. Finally, in Section VI-D, we evaluate the accuracy of our algorithm in a challenging situation involving sensors mounted on an autonomous vehicle.

A. Data Collection and Data Preprocessing

The data collection systems are different for each experiment, but each system includes at least one radar and one camera. The system discussed in Section VI-B is a handheld rig that incorporates a Texas Instruments (TI) AWR1843BOOST radar and Point Grey Flea3 USB camera. The measurement update rates for the sensors are 20 Hz and 30 Hz, respectively. For the experiments in Section VI-C, the data are from the publicly available IRS Radar Thermal Visual Inertial dataset [32]. The data collection system [32] is a handheld rig that mounts on a drone, where measurements are acquired from a TI IWR6843AOP radar, an IDS UI-3241 camera, and an Analog Devices ADIS16448 IMU, operating at frequencies of 10 Hz, 20 Hz, and 409 Hz, respectively. Doer et al. [32] provides additional details about this system. In Section VI-D, the data collection system [33] includes a vehicle-mounted TI AWR1843BOOST radar and three Point Grey Flea3 GigE cameras operating at frequencies of 25 Hz and 16 Hz, respectively.

In our real-world experiments, we use two similar radars that primarily differ in their angular resolutions. If two targets have identical ranges and range-rates but are separated by less than the angular resolution, then the targets will blend together, biasing the radar measurement output. The AWR1843BOOST has azimuth and elevation resolutions of 15° and 58°, respectively, while the IWR6843AOP has azimuth and elevation resolutions of 30°. As we show in Sections VI-B, VI-C, and VI-D, our algorithm is capable of calibrating both radars even though they have differing angular resolutions. For additional information on the radars used in our experiments, we refer

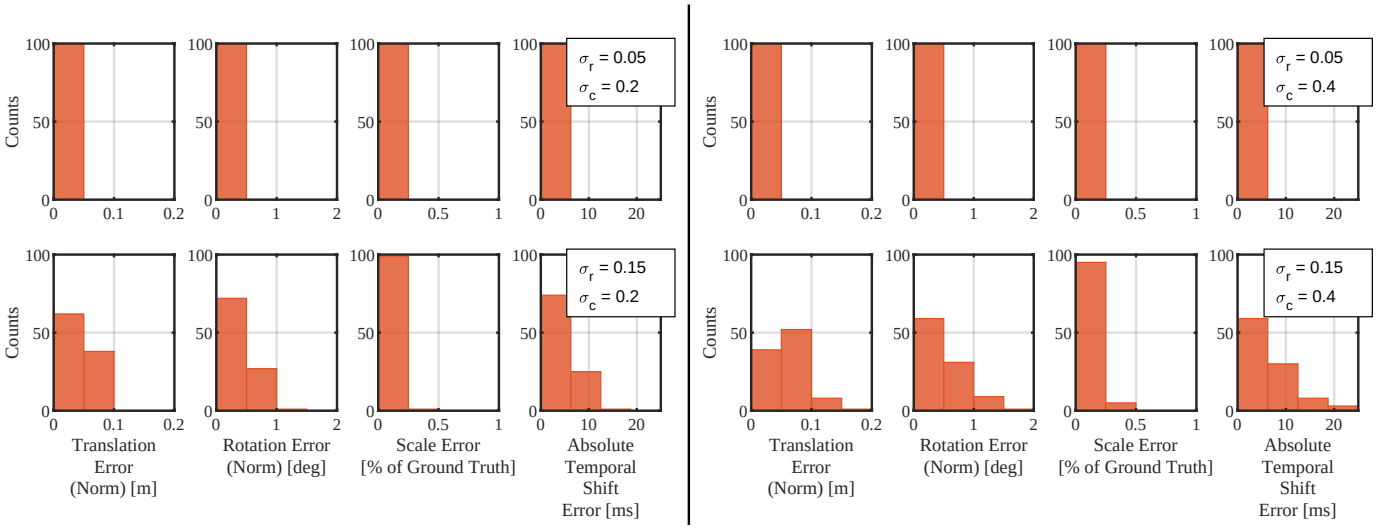


Fig. 5: High linear and low angular velocity trajectory calibration results from our simulation experiments. Each subplot is a histogram of the error between the estimated and true parameter values for 100 trials at a given level of measurement noise. Each row presents the results for a level of measurement noise. The levels of noise are a combination of two radar measurement noise levels ($\sigma_r = 0.05$ or 0.15 m/s) and two camera pixel measurement noise levels ($\sigma_c = 0.2$ or 0.4 pixels).

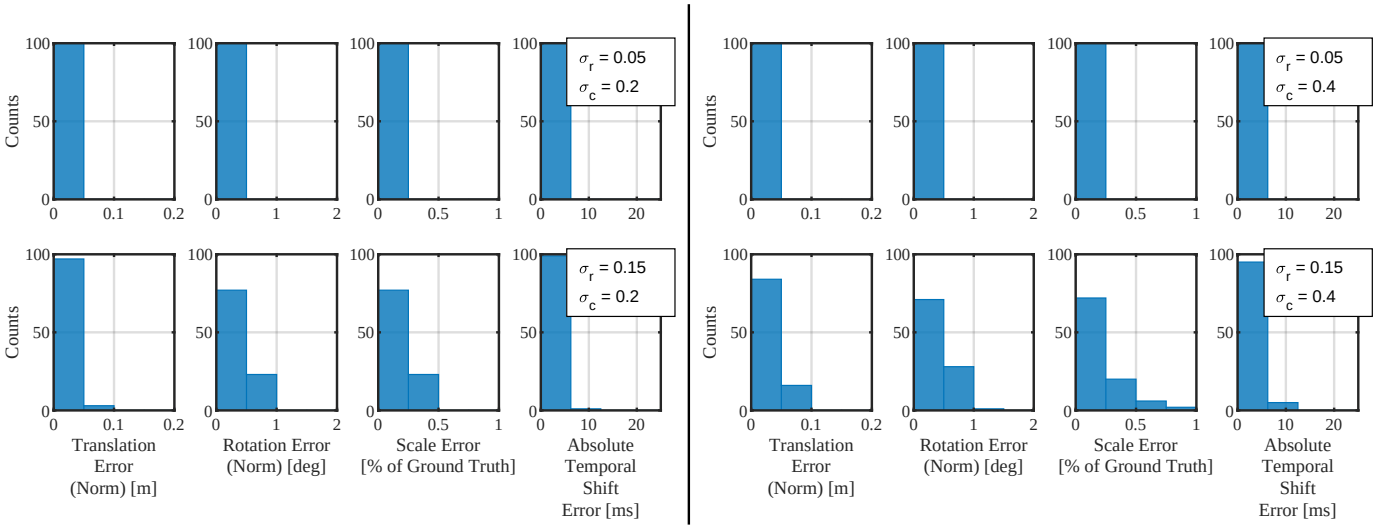


Fig. 6: Low linear and high angular velocity trajectory calibration results from our simulation experiments. Each subplot is a histogram of the error between the estimated and true parameter values for 100 trials at a given level of measurement noise. Each row presents the results for a level of measurement noise. The levels of measurement noise are a combination of two radar measurement noise levels ($\sigma_r = 0.05$ or 0.15 m/s) and two camera pixel measurement noise levels ($\sigma_c = 0.2$ or 0.4 pixels).

the reader to the AWR1843BOOST and IWR6843AOP user manuals [34], [35].

To ensure reliable ego-velocity estimation in our experiments, we set the maximum measurable range-rate and constant false alarm rate (CFAR) thresholds for our radar units. The maximum range-rate of the radar must be set above the maximum velocity of the data collection platform because the ego-velocity estimates will saturate at this value. However, an inverse relationship exists between the maximum range-rate and maximum range settings and these must be properly balanced for the operating environment [3]. The on-board radar preprocessing pipeline incorporates a CFAR detector that differentiates targets from background noise in the received EM signal [3]. Since the definition of background noise is also environment-dependent, we set the CFAR threshold to ensure that the ego-velocity estimator returned a sufficient number of inliers while minimizing the number of outliers. Before each experiment, we performed a series of 'test' data collection runs to tune these settings, making sure that the ego-velocity estimates were not saturating, that there were at least 15 inliers for each measurement, and that the inlier-to-outlier ratio was above 50%.

There are three data preprocessing steps for the experiments discussed in Sections VI-B and VI-C, while the experiment in Section VI-D requires a fourth preprocessing step. Prior to estimating the calibration parameters using our algorithm, we first determine radar ego-velocity estimates using the

algorithm from [25].[2] Second, we rectify the camera images to remove lens distortion effects. Third, we use the feature-based, monocular SLAM algorithm ORB-SLAM3 [36] to provide an initial estimate of the (arbitrarily-scaled) pose of the camera at the time of each image acquisition. While camera pose estimation is possible with any monocular SLAM, we chose this package for its robustness and accuracy [36]. Finally, for the experiment in Section VI-D, we remove outlier radar ego-velocity and camera pose estimates using a median filter. The median filter computes the local median and standard deviation of the signals across a window of time—200 ms and 850 ms for the radar and the camera, respectively. If the measurement at the centre of the window is greater than a chosen threshold from the median, the measurement is treated an outlier. For the tests in Section VI-D, the threshold is set to three standard deviations from the median, since this value eliminates gross outliers without removing noisy, but valid, portions of the signals. We found that this step was necessary to ensure data integrity.

B. Handheld Rig Experiment

In this experiment, we compared the calibration parameters estimated by our algorithm against the parameters determined by the target-based method in Peršić et al. [2]. To compare the two approaches, we used a handheld rig to collect a dataset consisting of two parts: one part with no visible calibration targets (for our algorithm) and one part with visible targets for target-based calibration. We collected both parts during one continuous run, without power-cycling the sensors. Our quality metric in this case is based on the results from target-based calibration (which can be treated as the 'gold standard,' effectively).

We used the first part of the dataset to perform targetless radar-camera calibration with our algorithm. The procedure consisted of moving the sensor rig, shown in Figure 7, throughout the office environment shown in Figure 9. A segment of the trajectory recovered by our algorithm is plotted in Figure 10. Then, we used the second part of the dataset to perform target-based calibration with the algorithm described in Peršić et al. [2]. In this case, the procedure consisted of moving a trihedral retroreflective target, shown in Figure 8, in front of the stationary radar-camera rig. The second part of the dataset was also used to evaluate the relative accuracy of the parameters estimated by both algorithms.

Our trihedral retroreflective target is specially constructed for calibration evaluation, consisting of a trihedral radar retroreflective 'corner' and a visual AprilTag [37] pattern printed on paper (which is EM-transparent). The target, shown in Figure 8, has the AprilTag mounted in front of the retroreflector. Using the known AprilTag scale, the pose of the camera relative to the AprilTag reference frame can be established. The distance from the origin of the AprilTag frame to the corner of the retroreflector is also known. During data collection, we kept the reflector opening pointed at the radar to ensure a consistent radar reflection.

[2]Available at: https://github.com/christopherdoer/reve

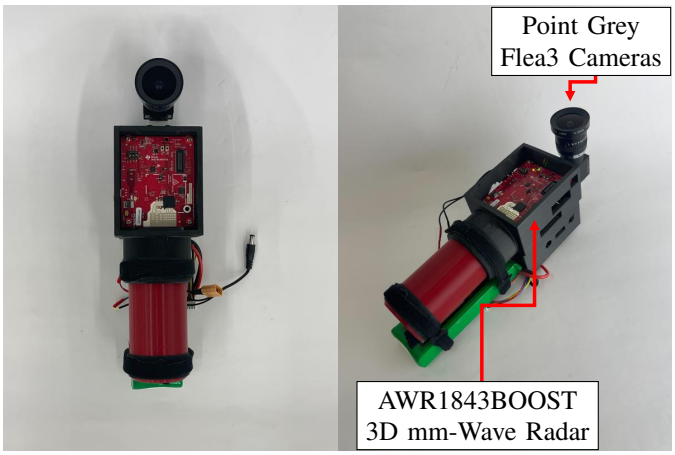


Fig. 7: Two pictures of our handheld sensor rig. The left image is a front view and the right image is an isometric view of the radar-camera unit. The radar antennas are mounted in the white area on the red circuit board. From our CAD model of the handheld rig, the radar-camera translation parameters (i.e., the components of $\mathbf{r}_c^{rc}$) are $r_x = 0.1$, $r_y = 10.5$, and $r_z = -1.0$ cm.

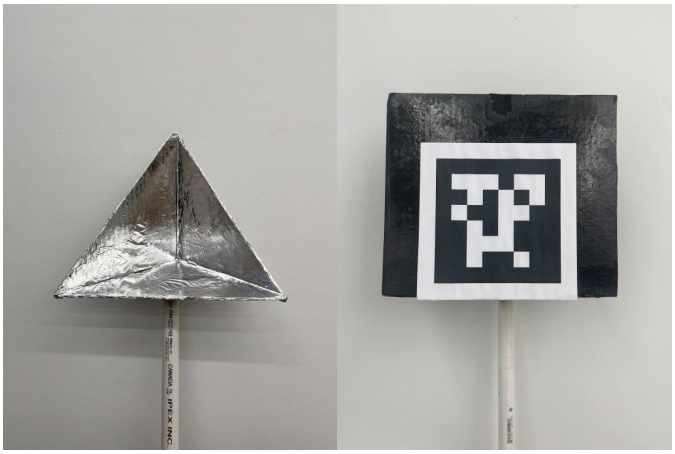


Fig. 8: Our specialized retroreflective radar target used for calibration verification. The left image shows the retroreflector alone, while the right image shows an AprilTag mounted to a flat cardboard backing that is attached to the front of the retroreflector. The cardboard material is fully transparent to the radar EM wave.

We quantify the calibration accuracy based on a 'reprojection error' metric. The reprojection error is the distance between the position of the retroreflector corner predicted from the camera observations and the position measured by the radar, both expressed in the radar frame. The retroreflector is more consistently detected than the AprilTag, and so we linearly interpolate the measured position of the trihedral retroreflector corner to the image timestamps.

Overall, our algorithm achieves results that are comparable to the method from Peršić et al. [2]. Table I shows that the estimated translation and rotation are, per axis, within 1.6 cm and 3 degrees, respectively, of the values estimated by the target-based method. Additionally, our estimated temporal offset differs from the target-based method by only 6 ms. Figure 11 shows that our algorithm, in a completely targetless manner, produces a reprojection error distribution with a

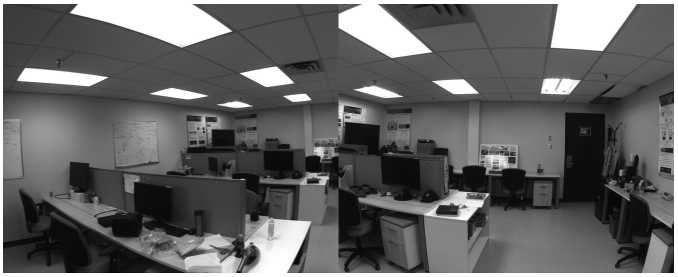


Fig. 9: Images from our handheld sensor rig calibration dataset, showing two views of the feature-rich indoor test environment.

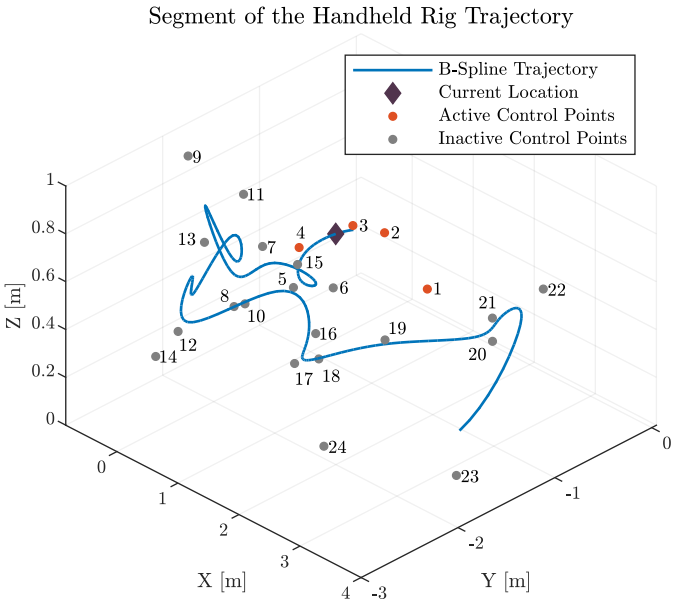


Fig. 10: A segment of the estimated $\mathbf{r}_r^{wr}$ B-spline for the handheld rig during calibration. The purple diamond is position of the rig 6.3 s from the start of the trajectory. The active control points at 6.3 s are shown in orange. As the rig continues along the trajectory, the active control points change.

median that is only 3 mm larger than the target-based method.

C. IRS Radar Thermal Visual Inertial Datasets

In this section, we demonstrate the versatility of our approach by making use of our estimated calibration parameters to improve the accuracy of camera-radar-IMU odometry. Specifically, we evaluate on the dataset and against the camera-radar-IMU odometry algorithm, known as RRxIO, described by Doer and Trommer in [32]. The extrinsic calibration parameters that accompany the IRS dataset were determined using the radar-IMU extrinsic calibration process described in [19] with ad hoc temporal calibration. Post hoc calibration of the radar and camera is challenging because the test environments do not contain any trihedral reflectors and the motion of the sensor platform is constrained (i.e., there are no deliberate excitations for calibration). To the best of the authors' knowledge, our approach is the only technique that can estimate all of the spatiotemporal calibration parameters for the dataset described in [32].

TABLE I: Calibration parameters for our handheld dataset. The values in each row are estimated by a different algorithm. The rotation between the sensors is given in roll-pitch-yaw (i.e., θ_x, θ_y, θ_z) Euler angle form.

	r_x [cm]	r_y [cm]	r_z [cm]	θ_x [rads]	θ_y [rads]	θ_z [rads]	τ [ms]
Peršić [2]	-1.60	11.9	-5.02	-1.59	0.07	-3.12	-63.8
Ours	-0.48	12.2	-3.42	-1.62	0.02	-3.15	-57.9

We chose to calibrate (and to evaluate calibration quality) for three of nine trajectories in the IRS dataset: `Gym`, `MoCap Easy`, and `MoCap Medium`. Data were collected in two environments with varying numbers of features: a large, sparse gymnasium and a feature-rich office setting. For the other six trajectories, poor lighting conditions and rapid motions caused ORB-SLAM3 to fail. To evaluate on a given trajectory, we compute the radar-camera spatiotemporal calibration parameters using our algorithm, and then run RRxIO on the same dataset with our estimated parameters. During evaluation, we disable the live 'camera-to-IMU' extrinsic calibration algorithm that operates as part of RRxIO. Using the known ground truth and the estimated RRxIO trajectories, we are able to determine the quality of our calibration using the following odometric error metrics: the relative translational root mean square error (RMSE RTE), relative rotational RMSE (RRE), absolute translational RMSE (ATE), and absolute rotational RMSE (ARE).

While the parameters estimated by our algorithm, listed in Table II, are relatively close to the parameters provided in Doer and Trommer [32], use of our parameters yields more accurate odometry estimates. The estimated temporal offset for the `Gym` trajectory is the only large deviation from the values in Doer and Trommer [32]. Table III reports the absolute and relative translation and rotation errors for the RRxIO trajectories after a yaw alignment. The parameters estimated by our algorithm improve the translation error on all datasets and rotation error for two of the datasets. Notably, the `Gym` dataset, which has the largest temporal offset, improves the most.

D. Vehicle Experiments

In this section, we verify the accuracy of our calibration algorithm by estimating the distance between cameras mounted on an autonomous vehicle. This task was challenging because,

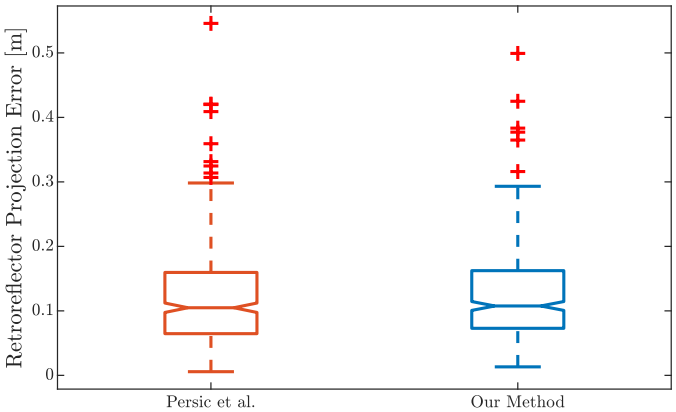


Fig. 11: The reprojection error distributions for the state-of-the-art method in [2] and ours.

TABLE II: Radar-IMU calibration parameters determined for three IRS trajectories and by RRxIO. The radar-IMU calibration parameters listed for our algorithm are a combination of the IRS IMU-camera parameters and our camera-radar parameters. The rotation between the sensors is given by roll-pitch-yaw (i.e., θ_x, θ_y, θ_z) Euler angles.

	r_x [cm]	r_y [cm]	r_z [cm]	θ_x [rads]	θ_y [rads]	θ_z [rads]	τ [ms]
RRxIO	6.00	4.00	-4.00	-3.14	0.02	-1.59	8.00
ME† (ours)	4.08	4.71	-5.05	-3.12	0.01	-1.59	13.1
MM† (ours)	3.90	4.46	-5.63	-3.11	0.01	-1.59	15.4
Gym (ours)	3.27	4.48	-3.62	-3.15	-0.06	-1.60	40.7

† These datasets are MoCap Easy (ME) and MoCap Medium (MM).

as shown in Figure 13, the radar-camera pairs do not share overlapping fields of view, so it is impossible to perform calibration using a target-based method. Additionally, the constrained motion of the car results in a poorly conditioned problem (i.e., the minimum eigenvalue of the identifiability matrix in Equation (24) is close to zero). The poor conditioning of the problem makes the estimated parameters very sensitive to sensor measurement noise, which can lead to inaccurate results. To overcome the poor conditioning of this system, we add an extrinsic calibration prior,

$$\begin{aligned} \mathbf{e}_{prior} &= \log(\mathbf{T}_{cr}^{-1}\mathbf{T}_{cr,prior}), \\ J_{prior} &= \mathbf{e}_{prior}{}^{T}\,\boldsymbol{\Sigma}_{prior}^{-1}\mathbf{e}_{prior}, \\ \boldsymbol{\Sigma}_{prior} &= \begin{bmatrix} \sigma_t^2\,\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \sigma_\theta^2\,\mathbf{I}_{3\times3} \end{bmatrix}, \end{aligned} \tag{25}$$

to the optimization problem. For our experiments, the prior for the extrinsic calibration parameters ($\mathbf{T}_{cr,prior}$) is derived from hand measurement. We set the prior uncertainty for the translation (σ_t) to 0.1 m along each axis, and the prior uncertainty for the rotation to (σ_θ) 30 degrees. The addition of this term stabilizes the estimation of the vertical translation between the radar and cameras, in particular. After optimization, less than 1% of the final cost value is due to the prior error term.

The mounting positions of the radar and three cameras on the car are shown in Figure 12, and the corresponding fields of view are shown in Figure 13. The first camera is positioned at the centre of the car and faces the direction of travel. The other two cameras are placed to the left and right of the centre camera and point roughly 45 degrees left and right from the forward axis, respectively. The 3D radar is more than one metre away from all of the cameras, facing towards the rear of the car, opposite the direction of travel.

We collected a total of nine datasets from the radar and the cameras (three datasets per camera) while driving two laps of a figure eight pattern. Data collection took place

TABLE III: Odometry performance evaluation for RRxIO and for our algorithm on three IRS trajectories.

	RTE [%]		**RRE [deg/m]**		**ATE [m]**		**ARE [deg]**	
Dataset	RRxIO	Ours	RRxIO	Ours	RRxIO	Ours	RRxIO	Ours
ME†	0.809	0.669	0.084	0.089	0.177	0.144	1.567	1.918
MM†	1.377	1.097	0.122	0.095	0.351	0.260	2.522	2.027
Gym	1.170	0.752	0.076	0.054	0.308	0.195	2.087	1.349

† These datasets are MoCap easy (ME) and MoCap Medium (MM).

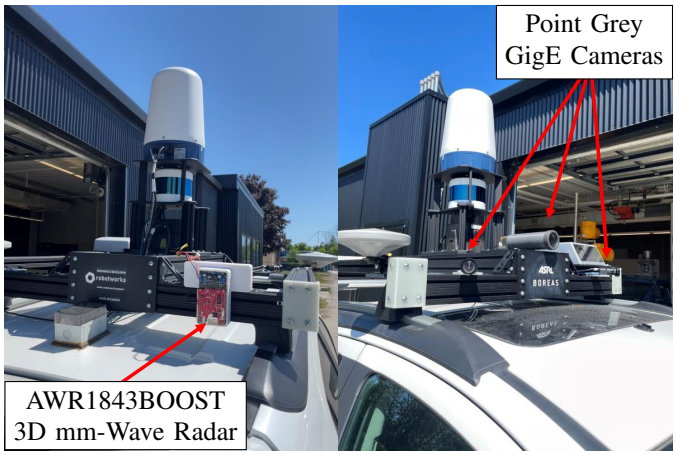


Fig. 12: Two views of the radar and camera mounting positions on the vehicle used in our experiments. The left image shows the mounting position of the TI radar. The right image shows the mounting positions of the three Point Grey cameras. The radar and the cameras do not share overlapping fields of view.

in a sparse parking lot environment, where the radar and camera features were at a substantial distance from the vehicle. We evaluated the accuracy of our estimated parameters by comparing the estimated distances between the centre camera and the two side cameras to the distances measured using a Leica Nova MS50 MultiStation. This method of comparison was selected in part because camera-to-camera extrinsic calibration is difficult for camera pairs that have minimal field of view overlap. Additionally, structural components of the car prevent direct measurement of the distance between the radar and cameras. Each run of our spatiotemporal calibration algorithm produced an estimated extrinsic calibration, for a total of three sets of estimated extrinsic calibration parameters for each camera. The transformations between the centre-to-left and -right cameras are computed by combining two radar extrinsic calibration estimates, which give a total of 18 camera-to-camera extrinsic calibration estimates (nine left and nine right).

Figure 14 shows the distribution of distance errors. The

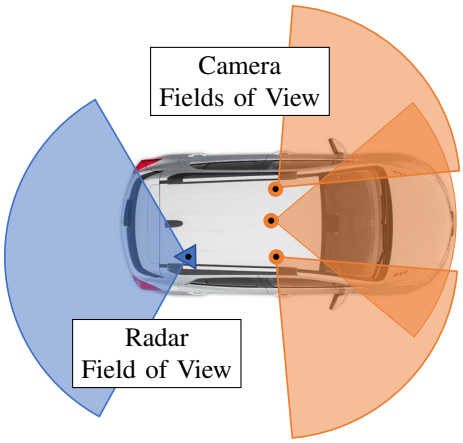


Fig. 13: Fields of view of the radar and cameras sensors used for our vehicle experiments.

majority of estimated extrinsic calibration parameters result in a camera-to-camera distance error of less than 5 cm, with two values that are greater than 10 cm. This error is reasonable given the 'chained' nature of the two radar-camera transforms. The accuracy of the estimated camera-camera distance depends on the accuracy of the translation and rotation parameters for both transforms. For this experiment, the radar-camera transforms have translation magnitudes greater than 1 m, so a small error in either estimated rotation results in a large distance error. In turn, we expect the estimated rotation and translation parameters to be within 5° and 5 cm of the their true values.

E. Calibration Environment

Several notes are in order regarding environments that are suitable for calibration. Although our algorithm does not require any retroreflective targets for the radar or a specific calibration pattern for the camera, there are nonetheless some limitations on where calibration can be performed. To ensure accurate ego-velocity estimation, the calibration environment should contain, at minimum, four stationary landmarks, with more being better. Also, to ensure accurate camera pose estimation using ORB-SLAM3, the scene should have sufficient lighting and visual texture. As a result, calibration should generally not be performed in scenes with many moving targets, dim lighting, or inclement weather such as fog. The accuracy of the camera pose estimates ultimately depends upon the specific SLAM algorithm that is chosen.

VII. Conclusion

In this paper, we described an algorithm that leverages radar ego-velocity estimates, unscaled camera pose measurements, and a continuous-time trajectory representation to perform targetless radar-to-camera spatiotemporal calibration. We proved that the calibration problem is identifiable and determined the necessary conditions for successful calibration. Through simulation studies, we demonstrated that our algorithm is accurate, but can be sensitive to the amount of noise present in the radar range-rate measurements. Further, we evaluated our algorithm in three different, real-world environments. First, we showed, using data from a handheld sensor rig, that our approach can match the accuracy of target-based calibration methods. Second, we presented results indicating that calibration can improve the localization performance of a hardware-triggered radar-camera-IMU system. Finally, we established that our calibration framework can be applied to AV systems, where the radar and camera are mounted at a significant distance from each other and do not share overlapping fields of view.

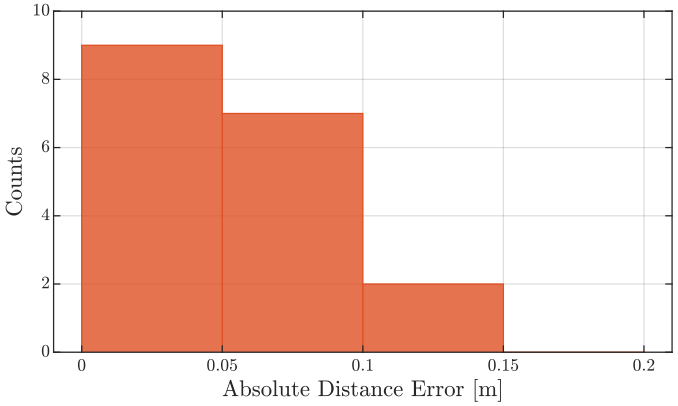


Fig. 14: Results from the vehicle calibration experiment, where the radar and the cameras do not share overlapping fields of view. The distance error is the difference between the estimated and measured distances between the center-left and center-right cameras. The ground truth distance was determined using a Leica MultiStation.

There are several potential directions for future research. It would be valuable to develop a method to automatically determine the knot spacing required for the continuous-time spline representation. Our calibration approach could naturally be extended to the multi-camera and multi-radar setting. Other pairs of sensors could also be considered beyond radar-camera pairs, including radar-inertial sensor combinations, for example.

Appendix A
An Extension on the Observability of Radar-to-Camera Extrinsic Calibration from Wise et. al [4]

In this appendix, we provide an extension to our earlier work in [4] demonstrating that radar-to-camera spatial calibration (with a known temporal offset) is locally weakly observable.

A. Notation for Nonlinear Observability Analysis

In Section III, we represent rotations as orthonormal matrices, which is convenient for the calibration problem but slightly more difficult to use for observability analyses. In this section, we rely on unit quaternions to represent rotations, avoiding the need to use exponential functions. We write a unit quaternion in 'vector' form as

$$\mathbf{q} = \begin{bmatrix} q_0 & \mathbf{q}_v \end{bmatrix}, \tag{26}$$

with the scalar component q_0 and vector component $\mathbf{q}_v$, such that $\|\mathbf{q}\|_2 = 1$. The conversion from unit quaternion to rotation matrix is given by the formula

$$\mathbf{R}_{ab} = \mathbf{R}(\mathbf{q}_{ab}) = (2q_0^2 - 1)\mathbf{I}_3 + 2\mathbf{q}_v\,\mathbf{q}_v{}^T + 2q_0\,\mathbf{q}_v^\wedge. \tag{27}$$

The quaternion kinematics are defined by

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Xi}(\mathbf{q})\,\boldsymbol{\omega} = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega})\mathbf{q}, \tag{28}$$

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -\boldsymbol{\omega}^\wedge \end{bmatrix}, \tag{29}$$

$$\boldsymbol{\Xi}(\mathbf{q}) = \begin{bmatrix} -\mathbf{q}_v{}^T \\ q_0\mathbf{I}_3 + \mathbf{q}_v^\wedge \end{bmatrix}, \tag{30}$$

where $\boldsymbol{\omega}$ is the angular velocity vector. The following identity is useful for the observability proof,

$$\begin{aligned} \frac{\partial \mathbf{R}(\mathbf{q})\mathbf{p}}{\partial \mathbf{q}} =& [(4q_0\mathbf{I}_3 + 2\mathbf{q}_v^\wedge)\mathbf{p} \\ & 2((\mathbf{q}_v{}^T\,\mathbf{p})\mathbf{I}_3 + \mathbf{q}_v\,\mathbf{p}^T - q_0\mathbf{p}^\wedge)], \end{aligned} \tag{31}$$

where $\mathbf{p}$ is an arbitrary 3×1 vector. If we transpose the rotation matrix on the left side of Equation (31), then the skew-symmetric terms on the right side change sign from positive to negative and vice versa.

B. Local Weak Observability

Following the procedure outlined in Section IV-A, we define the system equations, compute the respective Lie derivatives, and demonstrate that the nonlinear observability matrix has full column rank. In the analysis here, the pose, velocity, and acceleration states of the radar-camera system are camera-centric (i.e., taken with respect to the camera and not the radar). Since the camera-centric states can be used to determine the radar-centric states, this change does not affect the observability result.

Considering the camera frame $\underrightarrow{\mathcal{F}}_c$, the radar frame $\underrightarrow{\mathcal{F}}_r$, and the world frame $\underrightarrow{\mathcal{F}}_w$, the state vector for the observability analysis is defined as

$$\begin{aligned}\mathbf{x} = \begin{bmatrix} \mathbf{r}_w^{cwT} & \mathbf{q}_{wc}{}^T & \mathbf{v}_w^{cwT} & \boldsymbol{\omega}_c^{cwT} & \mathbf{a}_w^{cwT} & \boldsymbol{\alpha}_c^{cwT} \\ \gamma & \mathbf{r}_c^{rcT} & \mathbf{q}_{cr}{}^T \end{bmatrix}^T,\end{aligned} \tag{32}$$

where $\mathbf{r}$, $\mathbf{v}$, and $\mathbf{a}$ denote translation, linear velocity, and linear acceleration, respectively. The vectors $\boldsymbol{\omega}$ and $\boldsymbol{\alpha}$ are the angular velocity and the angular acceleration, respectively. Finally, γ is the scale factor for the camera translation (for a monocular camera system). The motion model for the system is

$$\dot{\mathbf{x}} = \mathbf{f}_0(\mathbf{x}) + \mathbf{f}_1(\mathbf{x}) = \begin{bmatrix} \mathbf{0}_{3\times1} \\ \frac{1}{2}\Xi(\mathbf{q}_{wc})\boldsymbol{\omega}_c^{cw} \\ \mathbf{0}_{3\times1} \\ \boldsymbol{\alpha}_c^{cw} \\ \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times1} \\ 0 \\ \mathbf{0}_{3\times1} \\ \mathbf{0}_{4\times1} \end{bmatrix} + \begin{bmatrix} \mathbf{v}_w^{cw} \\ \mathbf{0}_{4\times1} \\ \mathbf{a}_w^{cw} \\ \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times1} \\ \mathbf{0}_{3\times1} \\ 0 \\ \mathbf{0}_{3\times1} \\ \mathbf{0}_{4\times1} \end{bmatrix}. \tag{33}$$

The measurement model equations for the (scaled) camera translation and rotation are, respectively,

$$\begin{aligned}\mathbf{h}_1 &= \gamma\, \mathbf{r}_w^{cw}, \\ \mathbf{h}_2 &= \mathbf{q}_{wc}.\end{aligned} \tag{34}$$

Using the camera-centric model, it is possible to directly measure $\mathbf{q}_{wc}$ and, following the result in Section IV-B, to determine $\boldsymbol{\omega}_c^{cw}$ and $\boldsymbol{\alpha}_c^{cw}$. Finally, the radar ego-velocity measurement equation is

$$\mathbf{h}_3 = \mathbf{R}^T(\mathbf{q}_{cr})(\mathbf{R}^T(\mathbf{q}_{wc})\mathbf{v}_w^{cw} + \boldsymbol{\omega}_c^{cw\wedge}\mathbf{r}_c^{rc}). \tag{35}$$

The observability analysis requires the zeroth-, first-, and second-order Lie derivatives. The zeroth-order Lie derivatives are

$$\begin{aligned}\nabla L^0\mathbf{h}_1 &= \begin{bmatrix}\gamma\mathbf{I}_3 & \mathbf{0}_{3\times16} & \mathbf{r}_w^{cw} & \mathbf{0}_{3\times7}\end{bmatrix}, \\ \nabla L^0\mathbf{h}_2 &= \begin{bmatrix}\mathbf{0}_{4\times3} & \mathbf{I}_4 & \mathbf{0}_{4\times20}\end{bmatrix}, \\ \nabla L^0\mathbf{h}_3 &= [\mathbf{0}_{3\times3} \quad \mathbf{A} \quad \mathbf{R}^T(\mathbf{q}_{cr})\,\mathbf{R}^T(\mathbf{q}_{wc}) \\ &\quad - \mathbf{R}^T(\mathbf{q}_{cr})\mathbf{r}_c^{rc\wedge} \quad \mathbf{0}_{3\times7} \\ &\quad \mathbf{R}^T(\mathbf{q}_{cr})\boldsymbol{\omega}_c^{cw\wedge} \quad \mathbf{B}],\end{aligned} \tag{36}$$

where

$$\begin{aligned}\mathbf{A} &= \mathbf{R}^T(\mathbf{q}_{cr})\frac{\partial\, \mathbf{R}^T(\mathbf{q}_{wc})\mathbf{v}_w^{cw}}{\partial \mathbf{q}_{wc}}, \\ \mathbf{B} &= \frac{\partial \mathbf{R}^T(\mathbf{q}_{cr})(\mathbf{R}^T(\mathbf{q}_{wc})\mathbf{v}_w^{cw} + \boldsymbol{\omega}_c^{cw\wedge}\mathbf{r}_c^{rc})}{\partial \mathbf{q}_{cr}}.\end{aligned} \tag{37}$$

The first-order Lie derivatives are

$$\begin{aligned}\nabla L^1_{\mathbf{f}_1}\mathbf{h}_1 &= \begin{bmatrix}\mathbf{0}_{3\times7} & \gamma\mathbf{I}_3 & \mathbf{0}_{3\times9} & \mathbf{v}_w^{cw} & \mathbf{0}_{3\times7}\end{bmatrix}, \\ \nabla L^1_{\mathbf{f}_0}\mathbf{h}_2 &= [\mathbf{0}_{4\times3} \quad \tfrac{1}{2}\Omega(\boldsymbol{\omega}_c^{cw}) \quad \mathbf{0}_{4\times3} \\ &\quad \tfrac{1}{2}\Xi(\mathbf{q}_{wc}) \quad \mathbf{0}_{4\times14}], \\ \nabla L^1_{\mathbf{f}_0}\mathbf{h}_3 &= [\mathbf{0}_{3\times3} \quad \mathbf{C} \quad \mathbf{D} \quad \mathbf{E} \quad \mathbf{0}_{3\times3} \\ &\quad \mathbf{F} \quad \mathbf{0}_{3\times1} \quad \mathbf{R}^T(\mathbf{q}_{cr})\boldsymbol{\alpha}_c^{cw\wedge} \quad \mathbf{G}], \\ \nabla L^1_{\mathbf{f}_1}\mathbf{h}_3 &= [\mathbf{0}_{3\times3} \quad \mathbf{H} \quad \mathbf{0}_{3\times6} \\ &\quad \mathbf{R}^T(\mathbf{q}_{cr})\,\mathbf{R}^T(\mathbf{q}_{wc}) \quad \mathbf{0}_{3\times7} \quad \mathbf{L}],\end{aligned} \tag{38}$$

where

$$\begin{aligned}\mathbf{H} &= \mathbf{R}^T(\mathbf{q}_{cr})\frac{\partial\, \mathbf{R}^T(\mathbf{q}_{wc})\mathbf{a}_w^{cw}}{\partial \mathbf{q}_{wc}}, \\ \mathbf{L} &= \frac{\partial\, \mathbf{R}^T(\mathbf{q}_{cr})\,\mathbf{R}^T(\mathbf{q}_{wc})\mathbf{a}_w^{cw}}{\partial \mathbf{q}_{cr}}.\end{aligned} \tag{39}$$

We do not explicitly require the nonzero matrices, $\mathbf{C}$, $\mathbf{E}$, and $\mathbf{F}$, in Equation (38) because the submatrix formed from the columns corresponding to the rotation states can be shown to be full rank. The matrices $\mathbf{D}$ and $\mathbf{G}$ are required for the analysis, but we omit them here for brevity. The second-order Lie derivatives are

$$\begin{aligned}\nabla L^2_{\mathbf{f}_1}\mathbf{h}_1 &= \begin{bmatrix}\mathbf{0}_{3\times13} & \gamma\mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{a}_w^{cw} & \mathbf{0}_{3\times7}\end{bmatrix}, \\ \nabla L^2_{\mathbf{f}_0}\mathbf{h}_2 &= [\mathbf{0}_{4\times3} \quad \tfrac{1}{4}(2\boldsymbol{\Omega}(\boldsymbol{\alpha}_c^{cw}) - \boldsymbol{\omega}_c^{cwT}\boldsymbol{\omega}_c^{cw}\mathbf{I}_4) \\ &\quad \mathbf{0}_{4\times3} \quad -\tfrac{1}{2}\mathbf{q}_{wc}\,\boldsymbol{\omega}_c^{cwT} \quad \mathbf{0}_{4\times3} \\ &\quad \tfrac{1}{2}\Xi(\mathbf{q}_{wc}) \quad \mathbf{0}_{4\times8}].\end{aligned} \tag{40}$$

Stacking the gradients of the Lie derivatives, we arrive at the nonlinear observability matrix,

$$\mathbf{O} = \begin{bmatrix} \nabla L^0\mathbf{h}_1 \\ \nabla L^1_{f_1}\mathbf{h}_1 \\ \nabla L^2_{f_1 f_1}\mathbf{h}_1 \\ \nabla L^0\mathbf{h}_2 \\ \nabla L^1_{f_0}\mathbf{h}_2 \\ \nabla L^2_{f_0 f_0}\mathbf{h}_2 \\ \nabla L^0\mathbf{h}_3 \\ \nabla L^1_{f_0}\mathbf{h}_3 \\ \nabla L^1_{f_1}\mathbf{h}_3 \end{bmatrix}. \tag{41}$$

This matrix can be shown to be full column rank (except when excitation of the system is insufficient), hence the system is locally weakly observable.

REFERENCES

[1] C.-L. Lee, Y.-H. Hsueh, C.-C. Wang, and W.-C. Lin, "Extrinsic and Temporal Calibration of Automotive Radar and 3D Lidar," in *2020 IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, Oct. 25, 2020–Jan. 24, 2021 2020, pp. 9976–9983.

[2] J. Peršić, L. Petrović, I. Marković, and I. Petrović, "Spatiotemporal multisensor calibration via Gaussian processes moving target tracking," *IEEE Trans. Robotics*, vol. 37, no. 5, pp. 1401–1415, Mar. 2021.

[3] M. A. Richards, J. A. Scheer, and W. A. Holm, Eds., *Principles of Modern Radar: Basic Principles*. Institution of Eng. and Technol., 2010, vol. 1.

[4] E. Wise, J. Peršić, C. Grebe, I. Petrović, and J. Kelly, "A continuous-time approach for 3D radar-to-camera extrinsic calibration," in *2021 IEEE Intl. Conf. Robotics and Automation (ICRA)*, Xi'an, China, May 30–Jun. 5 2021, pp. 13 164–13 170.

[5] C. C. Stahoviak, "An instantaneous 3D ego-velocity measurement algorithm for frequency modulated continuous wave (FMCW) Doppler radar data," Master's thesis, University of Colorado at Boulder, 2019.

[6] S. Sugimoto, H. Tateda, H. Takahashi, and M. Okutomi, "Obstacle detection using millimeter-wave radar and its visualization on image sequence," in *Int. Conf. Pattern Recognition (ICPR)*, Cambridge, England, Aug. 23–26 2004, pp. 342–345.

[7] T. Wang, N. Zheng, J. Xin, and Z. Ma, "Integrating millimeter wave radar with a monocular vision sensor for on-road obstacle detection applications," *Sensors*, vol. 11, no. 9, pp. 8992–9008, Sep. 2011.

[8] D. Y. Kim and M. Jeon, "Data fusion of radar and image measurements for multi-object tracking via Kalman filtering," *Information Sciences*, vol. 278, pp. 641–652, Sep. 2014.

[9] J. Kim, D. S. Han, and B. Senouci, "Radar and vision sensor fusion for object detection in autonomous vehicle surroundings," in *2018 4th Int. Conf. Ubiquitous and Future Networks (ICUFN)*, Prague, Czech Republic, Jul. 3–6 2018, pp. 76–78.

[10] T. Kim, S. Kim, E. Lee, and M. Park, "Comparative analysis of RADAR-IR sensor fusion methods for object detection," in *2017 17th Int. Conf. Control, Automation and Systems (ICCAS)*, Jeju, Korea, Oct. 18–21 2017, pp. 1576–1580.

[11] G. El Natour, O. Ait Aider, R. Rouveure, F. Berry, and P. Faure, "Radar and vision sensors calibration for outdoor 3D reconstruction," in *2015 IEEE Int. Conf. Robotics and Automation (ICRA)*, Seattle, WA, USA, May 25–30 2015, pp. 2084–2089.

[12] J. Domhof, J. F. P. Kooij, and D. M. Gavrila, "An extrinsic calibration tool for radar, camera and lidar," in *2019 Int. Conf. Robotics and Automation (ICRA)*, Montréal, Canada, May, 20–24 2019, pp. 8107–8113.

[13] J. Peršić, I. Marković, and I. Petrović, "Extrinsic 6DoF calibration of a radar–lidar–camera system enhanced by radar cross section estimates evaluation," *Robotics and Autonomous Systems*, vol. 114, pp. 217–230, Apr. 2019.

[14] J. Oh, K. Kim, M. Park, and S. Kim, "A comparative study on camera-radar calibration methods," in *2018 15th Int. Conf. Control, Automation, Robotics and Vision (ICARCV)*, Singapore, Nov. 18–21 2018, pp. 1057–1062.

[15] C. Schöller, M. Schnettler, A. Krämmer, G. Hinz, M. Bakovic, M. Güzet, and A. Knoll, "Targetless rotational auto-calibration of radar and camera for intelligent transportation systems," in *2019 IEEE Intelligent Transportation Systems Conf. (ITSC)*, Auckland, New Zealand, Oct. 27–30 2019, pp. 3934–3941.

[16] J. Peršić, L. Petrović, I. Marković, and I. Petrović, "Online multi-sensor calibration based on moving object tracking," *Advanced Robotics*, vol. 35, no. 3–4, pp. 130–140, Sep. 2021.

[17] L. Heng, "Automatic targetless extrinsic calibration of multiple 3D lidars and radars," in *2020 IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, Oct. 25, 2020–Jan. 24, 2021 2020, pp. 10 669–10 675.

[18] D. Kellner, M. Barjenbruch, K. Dietmayer, J. Klappstein, and J. Dickmann, "Joint radar alignment and odometry calibration," in *2015 18th Int. Conf. Information Fusion (FUSION)*, Washington, DC, USA, Jul. 6–9 2015, pp. 366–374.

[19] C. Doer and G. F. Trommer, "Radar inertial odometry with online calibration," in *2020 European Navigation Conf. (ENC)*, Nov. 23–24 2020, pp. 1–10.

[20] J. Rehder, R. Siegwart, and P. Furgale, "A general approach to spatiotemporal calibration in multisensor systems," *IEEE Trans. Robotics*, vol. 32, no. 2, pp. 383–398, Apr. 2016.

[21] T. D. Barfoot, *State estimation for robotics*. Cambridge, UK: Cambridge Univ. Press, 2017.

[22] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient derivative computation for cumulative B-splines on Lie groups," in *2020 IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, Jun. 14–19 2020, pp. 11 145–11 153.

[23] C. de Boor, *A Practical Guide to Splines*, ser. Applied Mathematical Sciences. Springer-Verlag, Jan. 1978, vol. 27.

[24] K. Qin, "General matrix representations for b-splines," in *6th Pacific Conf. on Computer Graphics and Applications*, Singapore, Oct. 26–29 1998, pp. 37–43.

[25] C. Doer and G. F. Trommer, "An EKF based approach to radar inertial odometry," in *2020 IEEE Intl. Conf. Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Karlsruhe, Germany, Sep. 14–16 2020, pp. 152–159.

[26] A. Chiuso, P. Favaro, Hailin Jin, and S. Soatto, "Structure from motion causally integrated over time," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 523–535, Apr. 2002.

[27] S. Agarwal *et al.*, *Ceres Solver*. [Online]. Available: http://ceres-solver.org

[28] M. Li and A. I. Mourikis, "Online temporal calibration for camera-IMU systems: Theory and algorithms," *Intl. J. Robotics Research*, vol. 33, no. 7, pp. 947–964, 2014.

[29] R. A. Hewitt and J. A. Marshall, "Towards intensity-augmented SLAM with LiDAR and ToF sensors," in *Proc. IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (IROS)*, Hamburg, Germany, September/October 2015, pp. 1957–1961.

[30] R. Hermann and A. Krener, "Nonlinear controllability and observability," *IEEE Trans. Automatic Control*, vol. 22, no. 5, pp. 728–740, Oct. 1977.

[31] J. Kelly, C. Grebe, and M. Giamou, "A question of time: Revisiting the use of recursive filtering for temporal calibration of multisensor systems," in *Proc. IEEE Intl. Conf. Multisensor Fusion and Integration (MFI)*, Karlsruhe, Germany, 2021.

[32] C. Doer and G. F. Trommer, "Radar visual inertial odometry and radar thermal inertial odometry: Robust navigation even in challenging visual conditions," in *2021 IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (IROS)*, Prague, Czech Republic, Sep. 27 – Oct. 1 2021, pp. 331–338.

[33] K. Burnett, D. J. Yoon, Y. Wu, A. Z. Li, H. Zhang, S. Lu, J. Qian, W.-K. Tseng, A. Lambert, K. Y. Leung, A. P. Schoellig, and T. D. Barfoot, "Boreas: A multi-season autonomous driving dataset," *The International Journal of Robotics Research*, vol. 42, pp. 33–42, 2023.

[34] Texas Instruments, *xWR1843 Evaluation Module (xWR1843BOOST) Single-Chip mmWave Sensing Solution*, May 2020. [Online]. Available: https://www.ti.com/lit/ug/spruim4b/spruim4b.pdf

[35] ——, *IWR6843AOP Single-Chip 60- to 64-GHz mmWave Sensor Antennas-On-Package (AOP)*, July 2022. [Online]. Available: https://www.ti.com/lit/ds/symlink/iwr6843aop.pdf

[36] C. Campos, R. Elvira, J. J. G. Rodr'iguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robotics*, vol. 37, no. 6, pp. 1874–1890, May 2021.

[37] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *2011 IEEE Intl. Conf. Robotics and Automation (ICRA)*, Shanghai, China, May 9–13 2011, pp. 3400–3407.

Emmett Wise received his Bachelor of Applied Science in engineering physics from Queens University, Kingston, Canada, in 2016. After graduation, he worked to automate the manufacturing of nanoscale coatings at 3M Canada. He is currently a Ph.D. Candidate in the Space and Terrestrial Autonomous Robotics (STARS) Laboratory at the Institute for Aerospace Studies, Toronto, Canada. His research interests include perception, calibration, and state estimation.

Qilong (Jerry) Cheng is currently pursuing his Master of Engineering degree in electrical and computer engineering, having completed a Bachelor's in Mechanical Engineering at the University of Toronto. He was a developer at Autodesk from 2019 to 2020. From 2020 to 2021, he made design patent contributions to a high pressure spray nozzle design for the China State Shipbuilding Corporation. He is currently a graduate research student in the Space and Terrestrial Autonomous Robotics (STARS) Laboratory at the University of Toronto, focusing on calibration and state estimation.

Jonathan Kelly received the Ph.D. degree in Computer Science from the University of Southern California, Los Angeles, USA, in 2011. From 2011 to 2013 he was a postdoctoral associate in the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology, Cambridge, USA. He is currently an associate professor and director of the Space and Terrestrial Autonomous Robotic Systems (STARS) Laboratory at the University of Toronto Institute for Aerospace Studies, Toronto, Canada. Prof. Kelly holds the Tier II Canada Research Chair in Collaborative Robotics. His research interests include perception, planning, and learning for interactive robotic systems.

VibraForge: A Scalable Prototyping Toolkit For Creating Spatialized Vibrotactile Feedback Systems

Bingjian Huang
bingjian20@dgp.toronto.edu
Dynamic Graphics Project Lab, University of Toronto
Toronto, Ontario, Canada

Siyi Ren
siyi.ren@mail.utoronto.ca
Division of Engineering Science, University of Toronto
Toronto, Ontario, Canada

Yuewen Luo
sofia.luo@mail.utoronto.ca
Division of Engineering Science, University of Toronto
Toronto, Ontario, Canada

Qilong Cheng
qilong.cheng@mail.utoronto.ca
Mechanical Engineering, University of Toronto
Toronto, Ontario, Canada

Hanfeng Cai
hanfeng.cai@mail.utoronto.ca
Department of Electrical and Computer Engineering, University of Toronto
Toronto, Ontario, Canada

Yeqi Sang
abel.sang@mail.utoronto.ca
Department of Mechanical & Industrial Engineering, University of Toronto
Toronto, Ontario, Canada

Mauricio Sousa
mauricio@dgp.toronto.edu
Dynamic Graphics Project Lab, University of Toronto
Toronto, Ontario, Canada

Paul H. Dietz
dietz@cs.toronto.edu
Dynamic Graphics Project Lab, University of Toronto
Toronto, Ontario, Canada

Daniel Wigdor
daniel@dgp.toronto.edu
Dynamic Graphics Project Lab, University of Toronto
Toronto, Ontario, Canada

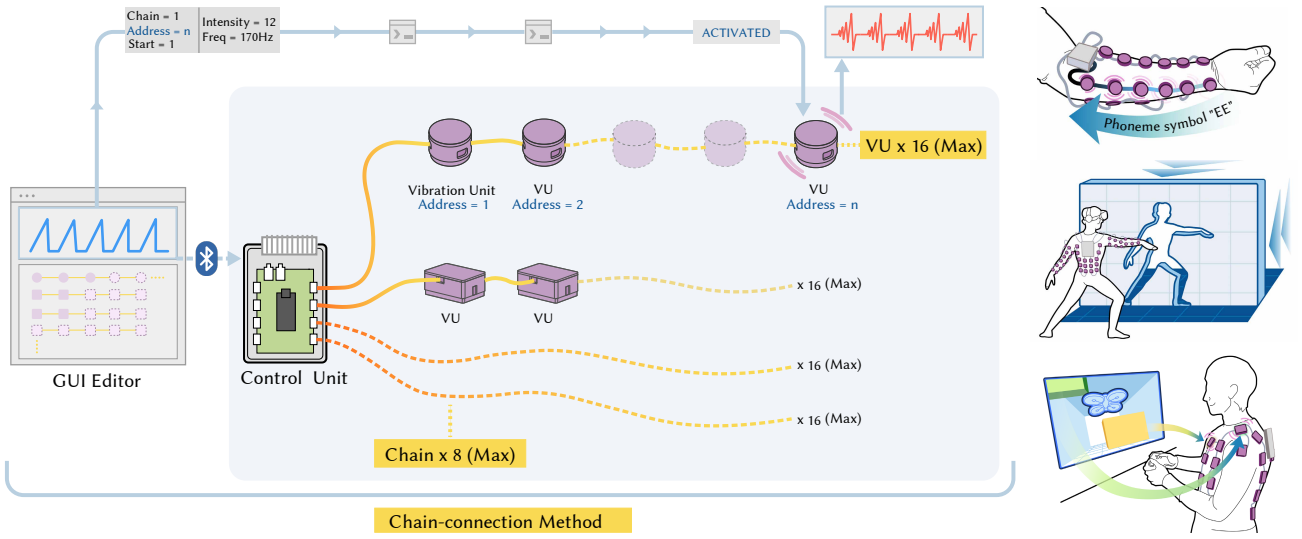


Figure 1: VibraForge is a scalable, open-source vibrotactile toolkit that supports the creation of spatialized vibrotactile feedback systems. Leveraging a chain-connection method, one control unit in the system can independently control up to 128 vibration units. Applications of the toolkit are shown on the right.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

Abstract

Spatialized vibrotactile feedback systems deliver tactile information by placing multiple vibrotactile actuators on the body. As increasing numbers of actuators are required to adequately convey information in complicated applications, haptic designers find it difficult to

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

create such systems due to limited scalability of existing toolkits. We propose VibraForge, an open-source vibrotactile toolkit that supports up to 128 vibrotactile actuators. Each actuator is encapsulated within a self-contained vibration unit and driven by its own microcontroller. By leveraging a chain-connection method, each unit receives independent vibration commands from a control unit, with fine-grained control over intensity and frequency. We also designed a GUI Editor to expedite the authoring of spatial vibrotactile patterns. Technical evaluations show that vibration units reliably reproduce audio waveforms with low-latency and high-bandwidth data communication. Case studies of phonemic tactile display, virtual reality fitness training, and drone teleoperation demonstrate the potential usage of VibraForge within different domains.

CCS Concepts

• **Human-centered computing → Haptic devices**; **User interface toolkits**.

Keywords

Haptics, Vibrotactile, Toolkits, Wearable Devices

ACM Reference Format:
Bingjian Huang, Siyi Ren, Yuewen Luo, Qilong Cheng, Hanfeng Cai, Yeqi Sang, Mauricio Sousa, Paul H. Dietz, and Daniel Wigdor. 2018. VibraForge: A Scalable Prototyping Toolkit For Creating Spatialized Vibrotactile Feedback Systems. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 17 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Tactile feedback conveys sensory information to the human body through the sense of touch [8]. Vibrotactile feedback, which produces tactile sensations via actuators vibrating on the skin, has gained much popularity because it is compact, affordable, and accessible [6]. It has been widely adopted in various domains, such as virtual reality (VR) [4, 17, 52], robotics [1, 31, 57], rehabilitation [3, 23, 35], and accessibility [16, 37, 55].

To provide adequate feedback to users during complex tasks, spatialized vibrotactile feedback systems have been proposed [22, 24]. These systems utilize actuator positions as an additional channel to enhance information transmission. For example, Wang et al. designed a facial mask with 36 eccentric rotating mass (ERM) actuators to study the perception of localization and orientation of objects in VR [52]. The bHaptics TactSuit X40 [4] utilized 40 ERMs on a vest to create localized feedback for immersive VR gaming experiences. The bHaptics TactSuit was also used by Xia et al. to convey hydrodynamic flow intensity during submarine teleoperation [57]. Jung et al. designed a wireless haptic interface with 36 ERMs to convey detailed fingertip touch sensations on the upper arm for amputees [16]. Sanchez et al. created OpenVNAVI, a navigation aid system with 128 ERMs to guide visually impaired people [45]. A 64-actuator tactile jacket was also created by Lemmens et al. to enrich movie viewing experiences [20]. Similarly, West et al. created the Body:Suit:Score system with 60 ERMs to enhance musical experiences [53].

Despite the great potential of spatialized vibrotactile feedback systems, previous implementations of such systems revealed scalability and expressivity limitations. Systems that depend on direct connections between actuators and microcontroller units (MCUs) exhibit limited scalability, as the number of actuators is intrinsically restricted by the available GPIOs or PWM channels on a MCU [17, 37, 52]. Expanding the number of actuators requires multiple MCUs, resulting in exponential increases in system complexity and communication overhead. For systems that use custom drivers and data bus protocols to drive actuators indirectly, the multi-layer connections support more actuators but compromise the vibration expressivity [20, 21, 53]. These systems cannot render complex waveforms such as audio signals due to bandwidth limits, thereby limiting the types of actuators that can be used to simple actuators like ERMs. Moreover, while various toolkits have been developed to assist vibrotactile design, they typically employ direct or multi-layer connections and face the same limitations (e.g., TECHTILE [29], VITAKI [26], and VHP [7]).

To overcome these limitations, this work proposes VibraForge, an open-source vibrotactile toolkit that leverages a scalable, modular design to enable the rapid prototyping of vibrotactile systems. VibraForge consists of control units and vibration units. Control units are responsible for receiving vibration control commands from a Bluetooth server and sending them to vibration units. Vibration units are self-contained modules with an actuator driven by their own microcontroller. Vibration units are connected to a control unit via a chain-connection method, where a single GPIO pin on a control unit can control up to sixteen vibration units via a custom UART protocol. To better support spatial pattern authoring, we also designed an accompanying GUI Editor that enables haptic designers to intuitively author multi-actuator patterns.

Technical evaluations of VibraForge showed that the toolkit has a high bandwidth (200 Hz), low latency (16 ms), and can reliably render complex audio signals in real time. Through three case studies, we demonstrate that VibraForge not only supports the efficient replication of existing research prototypes such as phonemic displays [37], but also improves user performance in various applications such as virtual reality fitness gaming and robot teleoperation.

The contributions of this work are:

- an open-source hardware and software toolkit, VibraForge, that supports the design of spatialized vibrotactile feedback systems,
- technical evaluations of VibraForge, including power consumption, bandwidth, latency and acceleration, etc., and
- case studies demonstrating replicated research prototypes and novel applications enabled by VibraForge.

2 Related Work

Herein, we provide an overview of existing vibrotactile toolkits, highlighting their strengths and limitations. Then, we summarize previous spatialized vibrotactile feedback systems and discuss the benefits and drawbacks of their design strategies. Lastly, we summarize existing software for vibration pattern design.

2.1 Existing Vibrotactile Toolkits

Vibrotactile actuators such as eccentric rotation mass vibration motors (ERM), linear resonant actuators (LRA), voice coil actuators (VCA), or piezoelectric actuators (PIEZO) create mechanical vibrations via electromagnetic motions. Each actuator type and model has distinct voltage ratings, frequency responses, and vibration characteristics, making it challenging for designers to explore different configurations for wearable tactile devices. To assist the design process, various vibrotactile toolkits have been developed. These toolkits abstract low-level technical details and offer intuitive design interfaces, so researchers and designers can concentrate on creating vibration effects without being overwhelmed by the complexities of electronic, mechanical, and software design.

Vibrotactile prototyping toolkits can be categorized into those for education purposes and those for design purposes. Toolkits for education purposes often have simple compact widgets and limited output channels so they can teach novice users about the basics of haptic feedback and the design of simple vibration effects. For example, the TECHTILE toolkit was composed of a microphone, a signal amplifier, and two actuators [29]. Audio signals were captured by the microphone, amplified through the amplifier, and delivered to the actuators. The Stereohaptics toolkit used a similar principle, enabling users to create, record, and playback simple haptic waveforms, making haptic design accessible through familiar audio tools [15].

Toolkits for design purposes often have more sophisticated hardware and complex functionality that enable researchers and designers to explore different configurations of actuators and vibration patterns. For example, VITAKI was an ERM-based toolkit targeting virtual reality and video game applications [26]. It used shift registers and PWM drivers to control up to sixteen ERMs (i.e., the number of channels available on the PWM drivers). TactJam supported up to eight actuators and was intended for collaborative design [54]. With Syntacts, designers could design vibration effects in a GUI Editor and then send them through an audio amplifier to actuators for testing [33]. The Syntacts audio amplifier, however, only supported eight channels and was too bulky to wear on the body. To enable the creation of wearable devices, the VHP toolkit used a custom designed PCB and flexible PCB connectors so devices could fit to different body shapes [7]. The Nordic nRF52840 microcontrollers used in the toolkit had three PWM modules that generated 12 channels of PWM signals to drive the actuators.

While existing toolkits provide rich features to support vibrotactile design, they are limited by their scalability, particularly in the number of actuators they can support. This is because the actuators are directly controlled via GPIO pins or PWM channels on MCUs, with most offering a maximum of 16 channels. As vibrotactile feedback continues to be used for more complicated tasks and applications, these toolkits cannot meet the demands of emerging systems. Therefore, when designing VibraForge, we drew insights from previous spatialized vibrotactile feedback systems and propose a new connection method to overcome these scalability limitations.

2.2 Spatialized Vibrotactile System Design

Two connection methods have been often deployed When building spatialized vibrotactile systems (Figure 2). Some have tried maintaining a "direct connection" to each actuator by adding more controllers to a system and thus increasing the number of PWM channels. For example, Wang et al. used PWM outputs from four Arduino Mega2560 boards to drive their facial display [52]. Similarly, Kaul et al. used four Arduino Nanos to drive 17 actuators mounted on their HapticHead display [17]. Reed et al. used three custom-built audio amplifier boards to send individual waveforms to 24 actuators on a sleeve to deliver speech phonemic cues to deaf users [37]. Although complex waveforms could be sent to the actuators via direct connections, communication overhead and system complexity increased significantly when the system scaled up.

Others have explored using "multi-layer connections" that employ custom driver boards to interface between the microcontrollers and actuators [20, 21, 45, 53]. On the input side, these driver boards received vibration commands from a main controller board via data bus protocols such as I2C [45] or SPI [20, 53]. On the output side, they converted vibration commands to PWM signals and drove four [20], six [53], eight [45], or sixteen [21] actuators. By leveraging a multi-layer design, these systems were more scalable than previous solutions, however, the indirect control afforded by the data bus protocols made it impossible to process complex waveforms. This limited the expressivity of the actuators so only simple actuators like ERMs were used in these systems.

There is thus a trade-off between system scalability and actuator expressivity: a system that supports fine-grained control of individual actuators is difficult to scale, and vice versa. In this work, we propose a chain-connection method (Figure 2), where actuators are encapsulated into self-contained vibration units and driven by their own PCB drivers. Chains of vibration units are connected to a central MCU. Each PCB driver receives commands from the central MCU and generates PWM signals for its own actuator, thus supporting scalability. The high bandwidth and low latency control of vibration characteristics enables actuators to render complex waveforms, thus enhancing expressivity

2.3 Software for Vibration Pattern Design

Vibration pattern design is crucial in spatialized vibrotactile feedback systems. For a comprehensive review of previous design software and their public availability, please see [48]. Here we discuss the common interfaces used to design single-actuator waveforms and multi-actuator patterns.

For single-actuator design, one common method is to use predefined waveforms. Designers start with standard waveforms such as sine, square, or triangle, multiply them to generate new waveforms, and modify them using envelopes [33]. While this method provides authoring flexibility, it requires that designers have a deep understanding of how waveforms can be composed through combinations of standard signals. This makes it difficult for novice designers to use. Others have created editors that enable the direct manipulation of waveforms. With these editors, designers have access to two parallel windows to edit vibration amplitude and frequency. For example, in the Macaron Editor, designers could directly move keyframe dots to modify a waveform, making the design process more intuitive [40]. This method was later adopted by several industrial software tools, including Meta Haptics Studio [27].

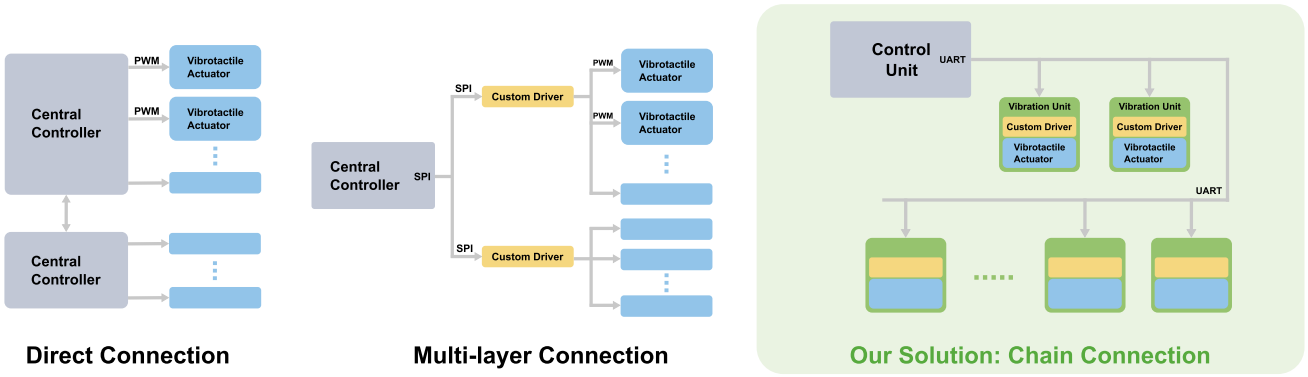


Figure 2: Depiction of the direct connection (left) and multi-layer connection (middle) methods commonly used in spatialized vibrotactile systems and our proposed solution (right), which uses a chain connection method.

For multi-actuator authoring, the design focus shifts from waveform editing to pattern design. Designers compose vibration paths across multiple actuators on a canvas and the system converts the drawn paths to a series of vibration commands that are executed over time. For example, Schneider et al. utilized phantom sensation illusions to interpolate points between actuators to produce more continuous vibration sensations in Mango [39].

VibraForge's GUI Editor combines single-actuator waveform design with a multi-actuator canvas to support the efficient design for spatialized systems. It provides a library of standard waveforms and envelops for composing complex waveforms. It is also compatible with existing tools such as Syntacts [33] and VIREO [48] to support the reuse of workflows and prior knowledge. The canvas shows a virtual layout of actuators that replicates the physical chain connections so designers can use it as reference when designing spatial patterns.

3 VibraForge Design

As the discussion of prior work highlighted, there is a trade-off between actuator expressivity and system scalability. We aimed to mitigate this trade-off by using a novel chain-connection method with custom designed vibration units and control units. This section provides an overview of the VibraForge toolkit[1], including its design requirements, vibration units, control units, chain-connection method, signal segmentation algorithm, and vibration pattern GUI Editor.

3.1 Design Requirements

VibraForge's design requirements were drawn from design challenges discussed in prior work on wearable spatial vibrotactile design or were identified by the authors based on prior prototyping experiences.

From spatial vibrotactile design, three design requirements were proposed:

- *DR1: Scalability*. The toolkit should support large quantities of actuators. Designers should be able to use different types of actuators such as ERM, LRA, and VCA.
- *DR2: Fine-Grained Control*. The toolkit should provide fine-grained control of individual actuator vibration characteristics such as intensity and frequency to support expressivity. Regardless of the number of actuators employed in the system, the control mechanism should have a high bandwidth and low latency.
- *DR3: Usability*. The toolkit should provide an intuitive design interface for authoring vibrotactile spatial patterns. To leverage designers' prior design knowledge, the toolkit should be compatible with existing vibrotactile waveform editors such as Syntacts [33], VIREO [48], and Meta Haptics Studio [27].

From wearable design, two design requirements were proposed:

- *DR4: Flexibility*. The toolkit should enable designers to modify settings, reprogram behaviors, and physically rearrange components without difficulty. This requirement was derived from design challenges faced by West et al. when attempting to relocate or replace components that were sewn onto a garment [53].
- *DR5: Portability*. Devices built with the toolkit should be portable and powered by batteries, communicate with external devices via wireless protocols, and use lightweight components to ensure a comfortable wearing experience.

3.2 Vibration Units

The vibrations units were designed as self-contained modules that could drive actuators by themselves. Each vibration unit consisted of a vibrotactile actuator, a custom designed PCB, and 3D printed components (Figure 3). We designed three versions of the vibration unit to support a range of actuators (DR1).

3.2.1 Vibrotactile Actuators. An X-axis, square-shaped, 9.5 mm x 9.5 mm LRA manufactured by the Jinlong Machinery and Electronics Company (Model L959535) [25] was chosen for the vibration unit. Its resonant frequency was 170 Hz. When powered by 0.9 Vrms, it had a maximum current of 145 mArms and an acceleration of 0.65±0.12 Grms. The other two actuators were a Z-axis LRA manufactured by Jinlong (Model G0832) [25] and a VCA manufactured by PUI Audio, Inc. (Model HD-VA3222) [36]. The Z-axis LRA was cylindrical-shaped, with an 8 mm diameter and a height of 3.2 mm.

[1]The project resource can be found at: https://anonymous.4open.science/r/VibraForge

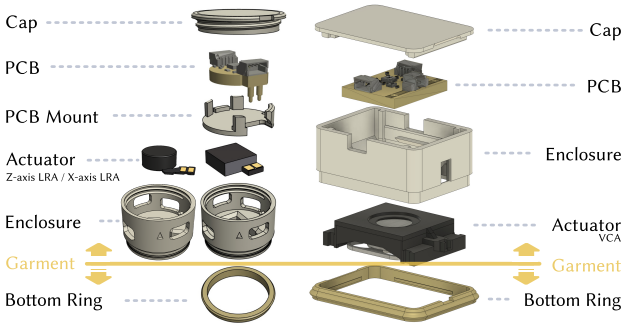


Figure 3: (Left) Exploded views of the small vibration unit for X-axis LRAs, including the cap, the PCB, the PCB mount, the enclosure, and the bottom ring. Z-axis LRAs shared most components except the enclosure. (Right) Exploded view of the large vibration unit for VCAs with a similar structure. Note that the bottom ring was placed underneath the garment to attach the unit.

Its resonant frequency was 235 Hz. When powered by 1.8 Vrms, it had a maximum current of 90 mArms and an acceleration of 0.50±0.08 Grms. The VCA was 32 mm x 22 mm and had a frequency response range from 80 - 500 Hz. When driven by a 133 Hz 1.5 Vrms signal, it had a maximum acceleration of 2.52 Gp-p.

These actuators were chosen because they cover a variety of vibration directions and intensities, thus satisfying the tactile perception requirements of different body parts (i.e., spatial acuity [9] and perceived intensity [49]). Since they all share the same PCB schematics and similar enclosure structures, in the following sections we only describe the vibration unit designed for the X-axis LRA. The other designs can be seen in Figure 3.

3.2.2 Printed Circuit Board Design. Due to bandwidth limitations, as the number of actuators increases it becomes difficult to drive them using real-time PWM signals sent from a central unit. Therefore, we allocated an MCU to each vibration unit to drive each actuator. Each vibration unit had its own driver PCB to receive UART commands from a central board and provide fine-grained control over vibration characteristics. The driver PCB had a PIC16F18313 MCU, a DRV8837 H-bridge motor driver, two 4-pin JST-SH headers, two pogo pins, and supportive components (Figure 4). The MCU was used to transmit commands and generate waveforms. The JST headers were used as input and output for the power and signal connections. The number of connector pins was minimized to four (i.e., actuator power, MCU power, ground, and data). The pogo pins were mounted on the bottom to ensure direct contact with the actuator pads.

When the MCU received a "start" signal, it sent a waveform to the H-bridge to modulate the actuator. We utilized the PWM generator inside the MCU for fine-grained control over the actuators' intensities, frequencies, and waveforms. Because there is no address system for each unit, designers only need to load their program into the MCU once. They can then plug-and-play each unit as desired. The communication protocol is further described in Section 3.4.

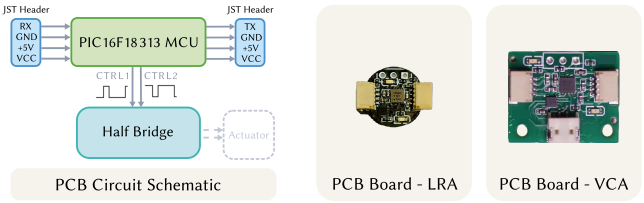


Figure 4: The circuit schematics and PCB board designs of vibration units.

3.2.3 3D Printed Enclosures. The 3D printed enclosures were designed to enclose each vibrotactile actuator and PCB board and provide a method to easily attach and detach each vibration unit to a garment (DR4). Similar to the size of a coin, the enclosure was compact and lightweight, enabling for a comfortable wearing experience (DR5).

The enclosure consisted of four parts: a cap, a PCB mount, an enclosure, and a bottom ring (Figure 3). The cap employed a screw-nut fastening system, which added top-down pressure to stabilize the PCB. The PCB mount offered protection for the PCB and aligned the pogo pins directly with the actuator pads. This improved the robustness of the vibration unit as no soldering was required during assembly. The enclosure served as a protective housing for all the parts. It had two holes for JST connectors, as well as two asymmetrically positioned vents for passive heat dissipation. The bottom ring was designed to be positioned beneath the garment to maintain direct contact with the skin without the need for any sewing (DR4).

The same design can be easily adapted to other vibration unit structures for other types of actuators. In Figure 3, other vibration units that share the same PCB schematics and slightly different enclosure designs are shown.

3.3 Control Unit

The control unit was designed to be fully portable (DR6). It consisted of a PCB extension board with an ESP32 microcontroller (MCU), two LiPo batteries, and a 3D-printed enclosure (Figure 5). Multiple versions of control units were designed to accommodate different projects, ranging from simple arm bands to more spatialized body displays. They varied in the (a) number of chains supported, (b) model of the ESP32 board, and (c) battery capacity.

We designed a large control unit that supported up to 8 chains of vibration units and 16 units per chain, for a total of 128 vibration units.The extension board drew power from two serial 3.7 V 8200 mAh LiPo batteries. It stabilized the voltage at 5 V through two power regulators: a linear regulator that powered all the MCUs and a buck regulator that powered all actuators in the system. These separate power supplies ensured that MCUs operated on stable voltages without interference from the actuators' power consumption. The extension board had eight JST connectors. Each connector was connected to a GPIO pin on the ESP32 MCU and drove a chain of vibration units. The MCU was an Adafruit Feather ESP32 V2[2]. It was responsible for converting external Bluetooth commands to UART commands and sending them to the corresponding vibration unit.

[2]https://learn.adafruit.com/adafruit-esp32-feather-v2

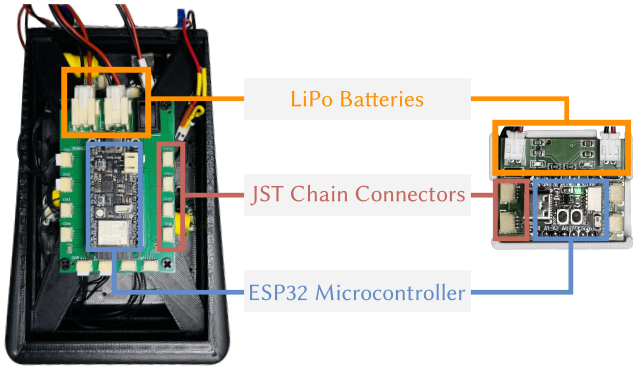


Figure 5: The control unit, which contained an ESP32 MCU, an extension board, chain connectors, and LiPo batteries.

Since the number of required vibration units varies across different tasks, we also designed a small control unit with fewer supported chains. This compact unit design contained two 3.7 V 500 mAh LiPo batteries and an Adafruit QT Py ESP32-S3 [3]. This MCU had four general purpose GPIO pins that could be used for chain connections.

3.4 Chain-Connection Method and Data Communication

VibraForge differs from existing toolkits due to its use of a chain-connection method (Figure 6). This method enables the toolkit to address over 100 vibration units using a single control unit (DR1) and support the simultaneous control over the vibration characteristics of multiple actuators (DR2).

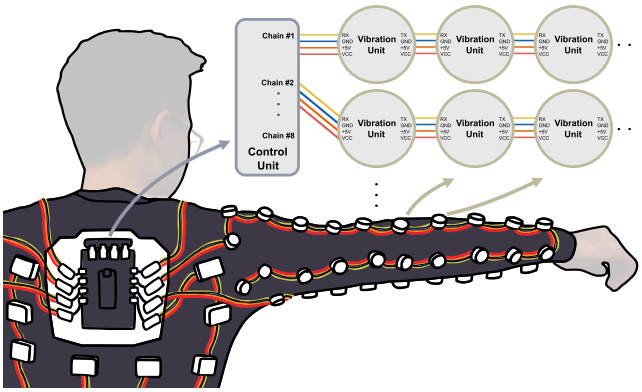


Figure 6: The multiple chain-connection method used within VibraForge, depicting the connections required for a given chain.

As explained in Section 3.3, each GPIO pin on the ESP32 MCU was routed to a JST connector on the extension board. Because vibration units shared the same JST connectors, they could be connected in chains and receive commands through the same GPIO pin.

[3]https://learn.adafruit.com/adafruit-qt-py-esp32-s3/pinouts

There were two options for the data transmission design. The first option was to use data bus protocols such as CAN or I2C, where all the vibration units would receive the same command simultaneously and use the command's address to determine whether the command was intended for itself. While this option would allow commands to be transmitted with a minimal latency, the vibration units would need to be programmed with different addresses prior to use, which would be burdensome when working with multiple units. Instead, we decided to send commands through each chain using a serial protocol, where each unit would receive a command, process it, and resend it to the next unit. While this may increase the overall latency, it allowed each unit to operate with a "virtual address", enabling independent plug-and-play control without additional programming.

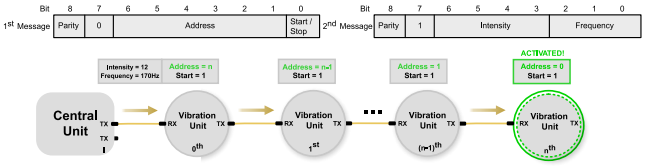


Figure 7: The two-byte message bits definition and an example of how vibration commands are transmitted along a chain using the "virtual address" system.

Vibration commands were transmitted via a UART protocol at 115.2 kHz with parity checks (Figure 7). Each command consisted of two bytes: the first byte encoded the address and start/stop status of the target vibration unit, and the second byte detailed the vibration characteristics. Upon receiving the first byte, a vibration unit checked the address. A non-zero address indicated the command was for subsequent units, prompting the unit to decrement the address and forward the message. This process continued until the address reached zero and signified the targeted unit, which then acted based on the start/stop bit: '1' (START) awaited the second byte, while '0' (STOP) disabled its actuator. When a unit was "signified", it waited for the second-byte message, which contained details about vibration characteristics, including the 16 intensity levels (evenly distributed from 0% to 100% duty cycle) and 8 frequencies (i.e., 123, 145, 170, 200, 235, 275, 322, and 384 Hz). The frequency levels were chosen so that each level was 1.18 times higher than the previous level. This followed Weber's factor for vibrotactile frequency just-noticeable differences [34].

3.5 Signal Segmentation Algorithm

To render complex waveforms such as audio signals on vibration units (DR2), we implemented a signal segmentation algorithm that converted high-frequency input signals to low-frequency serial commands (Figure 8).For instance, consider an input waveform of 44100 Hz audio signals that composed of 200 Hz and 5 Hz sine waves. High-frequency components (200 Hz) in the waveform were extracted using a short-time Fourier transform (STFT), which captured how the dominant frequency changed over time. Low-frequency components (5 Hz) were extracted using a Hilbert transform, which captured the envelope of the waveform. Both components were

downsampled to 200 Hz and sent to vibration units. Since our frequency levels ranged from 123 Hz to 384 Hz, the segmentation threshold was set to 100 Hz. Frequency components above 100 Hz were rendered using frequency levels, while components below 100 Hz were rendered using intensity levels. The technical evaluations in Section 4 demonstrated that this algorithm could reliably render complex waveforms such as audio signals using serial command control.

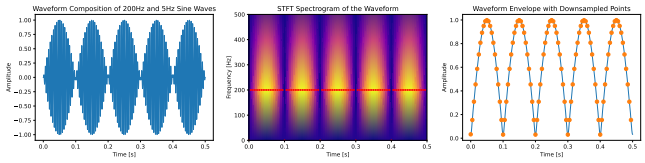


Figure 8: The signal segmentation workflow, showing how low frequency and high frequency components were extracted from a complex waveform.

3.6 GUI Editor for Designing Spatialized Patterns

We designed a GUI Editor (Figure 9) to enable designers to create spatialized vibration patterns (DR3). This editor differed from previous editors in that it combined the waveform editing from single-actuator design software with the canvas design used for multi-actuator design. The key components of the GUI Editor were:

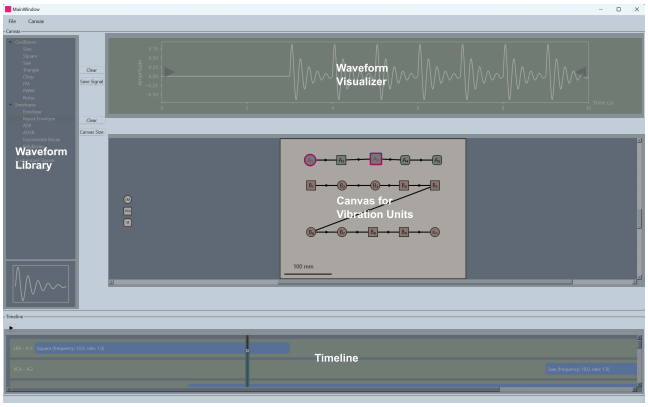


Figure 9: The layout of the GUI Editor, with its Waveform Library, Waveform Visualizer, Canvas, and Timeline.

- **Waveform Library**: The Waveform Library contained waveform templates such as oscillators and envelopes. Designers could use them to create a custom waveform and save it to the Waveform Library. In addition, designers could import waveforms in the JSON format from popular waveform editors such as Syntacts [33], VIREO [48], and Meta Haptics Studio [27] so they could reuse previous designs (DR3).
- **Waveform Visualizer**: The Waveform Visualizer served two purposes. When designers were creating a custom waveform, they could drag and drop waveform templates from the Waveform Library onto the Waveform Visualizer and it would show the combined waveform. When designers were composing vibration patterns for individual vibration units, the Waveform Visualizer also show the composed waveform for the selected unit.
- **Canvas**: The Canvas enabled designers to drag-and-drop vibration units onto the canvas so they could be connected into chains that mimicked the physical layout of the vibration units. In addition, a "Create New Chain" function enabled designers to generate vibration units in batches and arrange them using grid layouts. When clicking on a unit, its corresponding waveform would show up in the Waveform Visualizer.
- **Timeline**: The Timeline served as a high-level summary of the vibration patterns. The blue blocks showed the range of time when the vibration unit was activated. By moving the slider left and right, the vibration units that were activated at that moment would be highlighted in the Canvas so designers could understand the vibration patterns over time. By clicking the "Play" button, the waveform would be converted into vibration commands with timestamps and sent to the control unit via Bluetooth.

In a typical spatial pattern design workflow, designers would begin by creating a new design and specify the number of chains and actuators to be used in their system. The GUI would then generate a default Canvas with the designated number of actuators. Designers would then freely adjust the layout to match their physical device. Then, designers would assign waveforms to each vibration unit and specify the start and end time. Each waveform could be designed from scratch using the waveform templates in the Waveform Library, or imported from other software. To review the spatialized patterns, designers would click on any vibration unit on the Canvas and the Waveform Visualizer would show the corresponding waveform. Alternatively, designers could see a high-level summary of how the vibration patterns of multiple units would change over time by moving the Timeline slider.

Additionally, we also provided a Python server with VibraForge APIs, so that vibration units can be directly activated and controlled via code (DR3). Later in Section 5, Python servers are integrated into external programming environments (e.g., Unity) to trigger real-time feedback for applications.

4 Technical Evaluation

In this section, we summarize various performance metrics of the VibraForge toolkit to demonstrate how it fulfills the aforementioned design requirements. All tests below used vibration units of X-axis LRAs and the small control unit with default level-7 intensity (50% duty cycle) and 170 Hz frequency, unless otherwise stated.

4.1 Latency

Having fine-grained control (DR2) over the actuators in a vibrotactile system means that the data transmission of the system must have low latency and high bandwidth. To determine if the latency of VibraForge was below the human perceivable latency threshold of 45 ms [51], we measured the end-to-end latency of a PC to a control unit (i.e., the transmission latency of the Bluetooth Low

Energy protocol (BLE)), and of a control unit to a vibration unit (i.e., the latency of the custom UART protocol).

To measure the PC to control unit latency, a BLE write GATT command was sent and immediately followed by a BLE read GATT command. A Python timer recorded the write command sending time and read command receiving time and computed the difference as the latency. We repeated this measurement 100 times and found that the average round-trip transmission latency was 26.67±6.53 ms. Therefore, the one-way transmission latency was approximately 14 ms. For the control unit to vibration unit latency, repeated measurements using an oscilloscope showed that each command took 125 us to be transmitted from one unit to the next unit on a chain. Therefore, latency on the longest chain with 16 actuators would accumulate to 2 ms. Adding the two measurements together would thus result in an overall latency of approximately 16 ms, which would be virtually imperceptible by humans [51].

4.2 Bandwidth

The bandwidth of VibraForge was the number of commands that could be sent and processed per second by control units and vibration units. Bandwidth differs from latency in that asynchronous BLE commands could be continually sent without needing to wait for the previous command to be received (Figure 10). Therefore, bandwidth represented the refresh rate of the vibration commands for each actuator. Since BLE commands (millisecond level) would be processed much slower than UART commands (microsecond level), the bandwidth bottleneck of the system depended on the processing speed of BLE commands on the control unit.

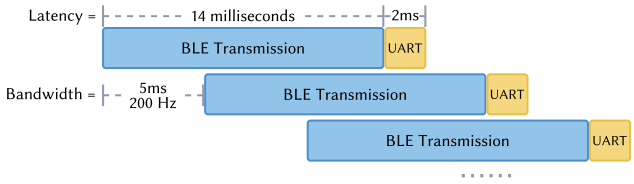


Figure 10: Data transmission bandwidth breakdown.

To measure VibraForge's bandwidth, we wrote a Python testing program to repetitively send standard BLE commands without latency, and another program on the ESP32 (i.e., control unit) to receive the commands and print the receiving time to the serial port. The command processing time was computed as the difference between the receiving time for each command. We sent 1,000 BLE commands with five UART commands in each of them, and computed an average processing time as 2.96 ± 1.20 ms. Therefore, for the actual operation of the toolkit, we set the BLE command sending delay to be 5 ms with some tolerance. This means that the control unit could simultaneously update five vibration units at 200 Hz refresh rate, satisfying the signal segmentation algorithm requirements.

Although vibration units were not directly driven by continuous audio signals from the central MCU, the high bandwidth ensured that they could approximate continuous waveforms using discrete commands (DR2).

4.3 Power Consumption and Battery Life

Power consumption measurements would provide an estimation of how long wearable devices built using VibraForge would function before recharging was necessary (DR6). Power consumption was measured using a bench power supply with a real-time measurement of voltage, current, and power. When only the control unit was connected, it consumed 106 mA current. When multiple vibration units were connected to the control unit, each vibration unit consumed 2.5 mA when idle. When a vibration unit was activated, the actuator consumed 150 mA. Therefore, for a vibrotactile arm display with one control unit and two full chains of vibration units (16x2), the idle current would be 186 mA and a 500 mAh LiPo battery would last 2.69 hours. Even in extreme situations such as multimedia entertainment, running two actuators consistently would consume 486 mA, leading to 1.03 hours of operating time. For a larger system with four chains of vibration units (16x4), the idle current would be 266 mA, so a 8200 mAh LiPo battery would last 30.83 hours. In an extreme situation where eight actuators were continuously activated, the overall current would be 1460 mA and the battery would last 5.62 hours.

4.4 Maximum Number of Vibration Units On Each Chain

Here we described how the maximum number of units on each chain was determined (DR1). Due to incremental wire resistance along the chains, voltage drops will become more significant when additional vibration units are connected or when more units are activated simultaneously. To maintain stable operation, the MCU power line must sustain a minimum of 2.3 V (i.e., the MCU's minimum operational voltage) and the actuator power line should not fall below 0.9 V (i.e., the LRA-X actuator's rated voltage).

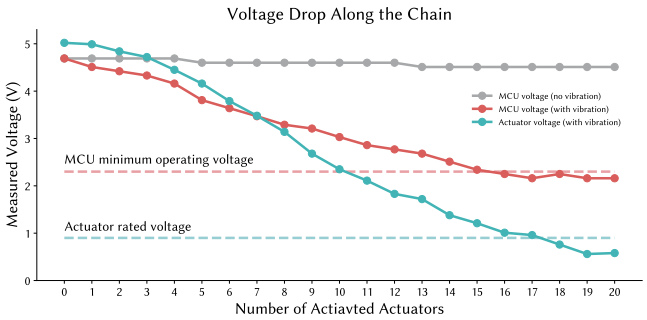


Figure 11: Voltage drops on the MCU power line and the actuator power line with an increasing number of actuators being activated.

We measured the voltage drop threshold by incrementally activating vibration units on the chain. An oscilloscope was used to measure the MCU power line and actuator power line at the last vibration unit of the chain. If the last unit maintained a stable voltage, then one could be certain that other units were stable as well. Figure 11 shows the relationship between voltage drop number of actuators activated. When the number of actuators surpassed 16, the MCU voltage dropped below the operating range. When it

surpassed 17, the actuator voltage dropped below its rated voltage. Therefore, control units could support up to 16 actuators on each chain.

4.5 Vibration Characteristics

To demonstrate that the toolkit can provide precise control over different vibration characteristics (DR2), we measured the acceleration of vibration units at various intensities and frequencies.

For intensity, we used an ADXL345 accelerometer for the measurements. Since most ADXL345 breakout boards are larger then the vibration unit itself, we designed a FlexPCB version and attached it between the skin and the vibration unit for precise measurements. A vibration unit was mounted on a sleeve and placed near the wrist of a human subject. During the testing, the vibration unit was activated for three seconds at each intensity level and the vibration frequency was kept constant at 170 Hz. The average acceleration was measured as the RMS value over the 1-second period of stable vibration. Acceleration was found to increase Monotonically as the intensity level increased (Figure 12 Left).

Similarly, for frequency, the vibration unit was placed near the wrist and activated for three seconds at each frequency level with level-3 intensity (25% duty cycle). To obtain the main frequency response, the acceleration data was processed using a fast Fourier transform. We found that the main frequency response was well aligned with the designated frequency at each level, thereby demonstrating the precise PWM signal control (Figure 12 Right).

4.6 Waveform Approximation

Because vibration units are controlled via discrete UART commands rather than continuous audio signals, one might worry that this would reduce the expressivity of the vibration waveform delivered through the units. Here we show that by utilizing high-bandwidth communication (DR2), VibraForge was capable of approximating audio signals and creating similar feedback. We used standard vibration waveforms from VibViz [41], converted them from WAV files to HAPTIC files through Meta Haptics Studio [27], and sent them to the GUI Editor. The editor automatically converted the haptic files to a sequence of vibration commands that were used to drive a vibration unit. We then measured the acceleration of the vibration unit and compared it with the original audio envelope. The results showed that they were closely approximated to each other (Figure 13).

5 Case Studies

We conducted three case studies to demonstrate the potential usage of VibraForge. The first case study showcased how VibraForge can be used to efficiently reproduce previous research prototypes. The second and third case studies showed how VibraForge can be used to build novel spatial vibrotactile systems that improve user performance in various applications.

5.1 Case Study 1: Replicating A Phonemic-Based Tactile Display

In Reed et al.'s work, a phonemic-based tactile display was designed to aid speech communication [37]. This tactile display used 24 actuators on the forearm to deliver unique tactile codes of 39 English phonemes to deaf and hard-of-hearing users. Phonemes were mapped to vibration patterns using frequency, waveform, amplitude, spatial location, and movement. Although the display was shown to be useful, the process of creating the display was challenging. To drive the 24 actuators independently, the researchers had to use a desktop audio device (i.e., a MOTU 24Ao) with three custom amplifier boards to provide output signals, which made the whole system bulky and unable to be used in real-world situations. In this section, we demonstrated how the VibraForge toolkit could be utilized to simplify the development process and enhance the portability of the display.

5.1.1 Vibrotactile System Development. The first step in replicating the phonemic tactile display was to determine the layout of the actuators and the chain connections. Since the original display used a 4x6 layout on the forearm, we decided to use four chains with six vibration units on each chain and a small control unit.

The second step was to assemble the display. The display required a sleeve garment and multiple modules from the toolkit (Figure 14). We first assembled the vibration units, then attached them to the garment and connected JST wires between them. The control unit was attached to the garment and four chains of vibration units were then connected to the control unit. The mapping between chains and connection ports followed the order of actuators described in the original paper. The whole assembly process took 16 minutes.

Once the hardware was built, the next step was to design the vibration patterns. Each phoneme needed to represented by a distinctive waveform that incorporated a high-frequency primary component and a low-frequency envelope, with a duration ranging from 100 to 480 ms. Consonants were designed as static patterns with multiple actuators being activated concurrently, whereas vowels were designed as dynamic patterns with actuators being activated in succession. The GUI Editor streamlined the process of creating the large variety of waveforms.

For example, to design the waveform for the consonant "V" (IPA symbol /v/), we first dragged a sine wave into the Waveform Visualizer and set the frequency to be 300 Hz (primary component). Then we multiplied it with another sine wave at 8 Hz (envelope). The combined waveform was then saved in the Waveform Library as "Consonant_B". Alternatively, when the envelope was not available in the Waveform Library (e.g., cos^2 for consonant "H"), we designed the waveform in Syntacts and imported the .CSV file into our GUI Editor. We then used the "create new chain" function to create a virtual layout with four chains of vibration units on the canvas. We selected each of the designated vibration units (i.e., i6, ii6, iii6, iv6), assigned the custom waveform, and set the time range from 0 to 400 ms. Finally, we clicked the "play" button to validate the design. The average time for an experienced haptic researcher on our research team to create one phoneme was one minute and five seconds. Therefore, the whole phoneme design process could be finished in 42 minutes. While the original paper did not mention the waveform design time, we believe the streamlined waveform creation process led to a significant reduction of design time.

5.1.2 Lessons Learned. The use of VibraForge to create the phonemic tactile display had several benefits. First, compared with the original device, our system was fully portable (DR5), with a small control unit and 24 vibration units directly attached to the sleeve.

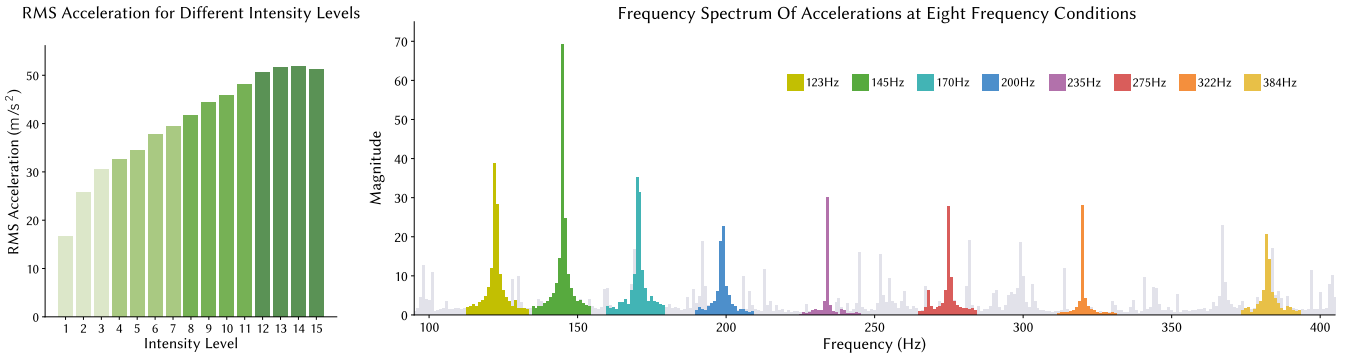


Figure 12: The acceleration measurements at various intensity and frequency levels.

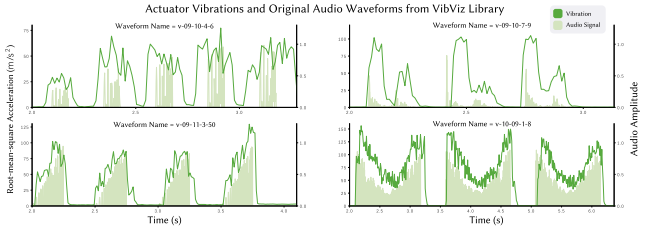


Figure 13: Comparison of actuator vibrations (green lines) and original audio waveforms (light green shaded areas) from the VibViz Library

Second, VibraForge simplified the overall development process. The hardware system was reduced to module assembly without soldering, and the waveform design was supported by the GUI Editor (DR3). Third, the flexible design (DR4) enabled personalization opportunities. The original paper mentioned that the inter-motor spacing of the actuators needed to be adjusted for different participants and that the process of adjusting the layout was rather tedious. With our system, because the vibration units could be easily attached and detached, researchers could easily adjust the positions even if the garment was worn on the arm, thus giving them much more flexibility.

Meanwhile, some challenges were found when using the toolkit. While designers could easily author patterns using the GUI Editor, it mainly served as a testing site, because the current design of GUI Editor only allows users to manually click the "play/stop" button to control the vibration sequences. This has constrained its use for real world interaction, where speech is segmented into phonemes and used to trigger corresponding patterns in real time. Right now the Python server does not support this kind of interaction so an additional API would be needed.

5.2 Case Study 2: Fitness Gaming in Virtual Reality

Virtual Reality (VR) fitness applications like OhShape [18] and Supernatural [43] have gained popularity by turning traditional fitness routines into interactive, gamified experiences. However, users often face challenges due to the absence of real-time, precise feedback about the physical positioning of their body [56]. While prior research has demonstrated the utility of vibrotactile feedback for movement guidance in rehabilitation applications [3, 35], systems used in these applications only deployed actuators on major joint locations such as the wrists or elbows, thus offering limited and coarse guidance. Inspired by OhShape [18] and the TV show "Hole in the Wall" [10], we built a VR fitness application in which participants were asked to adjust their body poses to mimic cutouts in virtual walls when their digital avatar moved through the wall (Figure 15). We used the VibraForge toolkit to construct an upper-body suit that would provide spatial vibrotactile feedback to improve training performance.

5.2.1 Vibrotactile System Development. The first step of our development was to determine the layout of vibration units on the body. We referenced prior work that measured spatial acuity [12] and determined that vibration units should be uniformly distributed on the arm with 4 cm of spacing between them and on the body with 8 cm of spacing between them. As a result, the suit had 36 vibration units on the body and 40 vibration units on each arm (Figure 15).

The second step was to mount the control units and vibration units on the garment, which was an exercise compression shirt. Since the total number of vibration units was within the upper limit of the large control unit, we only needed to use one control unit. It was worn on the upper back to help users maintain their balance during fitness training. Vibration units were mounted one-by-one and JST wires were used to connect them.

The third step was to integrate the hardware into the fitness game, which was built using the Unity Game Engine [47]. The physical layout of the vibration units on the suit was manually converted into a virtual layout and overlaid on the user's avatar, so that the virtual vibration units would move following the human body movement. Whenever a collision was detected between a virtual vibration unit and an obstacle, the system would determine which vibration units should be activated and send the vibration commands to the control unit through a Python server (Figure 15). Since collisions were binary events (i.e., collided / not collided), the vibration intensity was set to constant level (i.e., 7), as well as frequency (i.e., 170 Hz).

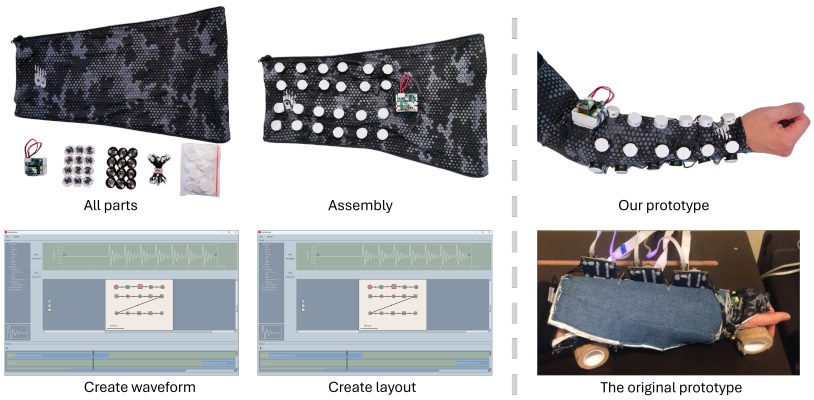


Figure 14: The development process of Case Study 1's phonemic tactile display re-implementation using VibraForge.

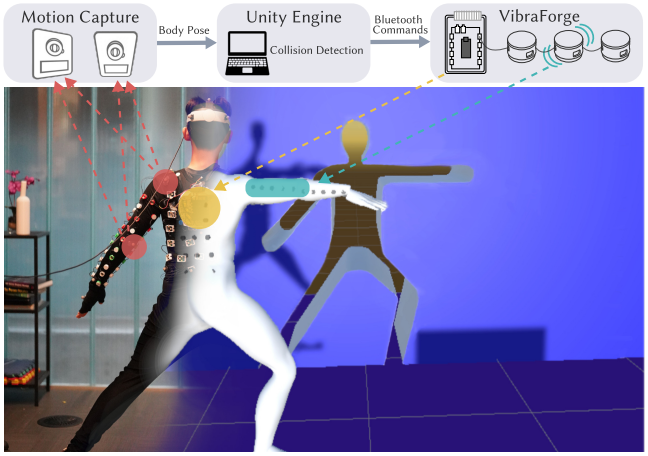


Figure 15: An overview of the pipeline used in Case Study 2's VR fitness game. Participant body movements were tracked by Vicon Motion Cameras and sent to a Unity scene for collision detection. Collision commands were then sent to a control unit on the participant's back through Bluetooth and converted to UART commands for the vibration units.

5.2.2 User Study Design. A preliminary study was conducted to evaluate the effectiveness of the vibrotactile suit on fitness training performance. The VR application was run on an Oculus Quest 2 headset [28] through a Quest Link connection to a PC (i.e., Alienware M15 Laptop with RTX 3060Ti GPU). Vicon Motion Capture cameras [50] were used to track the body poses of each participant and the poses were then mapped to the virtual avatar in the Unity scene. Nine participants (i.e., 4 Males and 5 Females; mean age = 25 years, SD = 3 years) were recruited to participate in the study, which lasted approximately one and a half hours, and each received $40 compensation. The study was approved by our institutional research ethics board and conducted in a university lab space.

During the study, each participant would experience two conditions, i.e., no feedback and vibrotactile feedback. In the no feedback condition, participants only saw the incoming virtual walls and their own avatar and had to rely on proprioception to adjust their body position. In the vibrotactile feedback condition, participants would feel vibrations at the body positions where collisions occurred between vibration units and walls. For each condition, 24 pairs of walls with different cutouts were presented in sequence. Two walls in each pair were identical. The first wall served as the "preview wall" to help the participant adjust their body pose, and the second wall served as the "test wall" to measure any body position changes. We measured the number of collisions occurring during each condition. Our hypotheses were that (1) vibrotactile feedback would cause fewer collisions than no feedback, and (2) test walls would have fewer collisions than preview walls.

5.2.3 Results. The average number of collisions between vibration units and walls was computed for each feedback condition. Then, a repeated measures ANOVA was conducted using SPSS to determine if feedback condition had an effect on the number of collisions that occurred. If there were significant differences between each feedback condition, pair-wise comparisons with a Bonferroni correction were then performed.

The RM-ANOVA determined that both feedback type ($F(1,8) = 6.450, p < 0.05$) and wall type ($F(1,8) = 5.582, p < 0.05$) had significant effects on the number of collisions. No interaction effect was found (($F(1,8) = 3.936, p = 0.083$). For feedback type, the vibrotactile feedback (M=26.289, std=9.897) led to fewer collisions than the no feedback type (M=31.489, std=7.162, $p < 0.05$). For wall type, the test walls (M=27.552, std=10) led to fewer collisions than the preview walls (M=30.226, std=7.694, $p < 0.05$). Therefore, we concluded that both hypotheses were accepted.

5.2.4 Lessons Learned. The benefits of using VibraForge to build this high-density vibrotactile suit were multi-fold. First, scalability (DR1) enabled us to support a large number of actuators and provide highly localized feedback to guide body movement. Second, the modular and flexible design (DR4) made the system assembly process as simple as press-fitting the vibration units onto the garment and connecting wires between them. The entire process took less than 30 minutes. Compared with prior work that used sewing techniques [20, 53], our solution was more flexible and efficient. Third, the low latency and high bandwidth communication (DR2) helped deliver almost real-time vibrotactile feedback to users when

collisions occurred. Participants did not report any perceivable delays during the study.

Nonetheless, there were several challenges during the process. First, the integration of the vibrotactile system into the interactive game design was difficult. This was because we had to manually convert the physical layout of units on the body to a virtual layout in the Unity Engine and assign vibration units to their positions. This process would have been easier if we could detect the physical positions of the units and generate a virtual mapping automatically. Second, two participants reported that vibration units near the waist were not consistently perceived because body movement during fitness training would loosen the contact with garment. Our temporary solution was to tighten the suit so that movement was constrained. In the future, other methods would be needed to ensure that the vibration units maintained contact with the skin.

5.3 Case Study 3: Teleoperation Assistance for Unmanned Aerial Vehicle Operation

In this case study, we used VibraForge to design a wearable vibrotactile device to provide Unmanned Aerial Vehicle (UAV) operators with spatial information about obstacles. Current UAV teleoperation is dependent on visual feedback from onboard cameras, which presents significant challenges as a camera's limited field of view can hinder situation awareness and lead to potential collisions with obstacles. While there have been prior attempts to use haptic feedback to notify operators of nearby obstacles [5, 11, 57, 58], they required an operator to switch from using radio control (RC) controllers to hand-held haptic joysticks, only rendered force feedback in one direction at a time, or had fixed actuator layouts that could not represent lateral movements due to the absence of actuators on the side.

By creating a new vibrotactile feedback system using VibraForge, one could flexibly attach multiple vibration units to an operator's body that would render simultaneous spatial cues about surrounding obstacles while the operator was using an RC controller. This would enhance operator's collision avoidance ability and situational awareness in environments where visual feedback is limited.

5.3.1 Vibrotactile System Development. The first step of the development was to determine the layout of the vibration units on the body. Since there is no prior work that studied the mapping between tactile feedback positions on the body and human perception of 3D directions, we conducted a perceptual study with 10 participants to identify a mapping. The layout was initialized with 46 vibration units uniformly distributed on the body and optimized using user data. The final layout used 32 vibration units to achieve comprehensive coverage of the 3D space (Figure 16). More details on the perceptual study and layout optimization can be found in [13].

The second step was to affix control units and vibration units on the garment, which was a compression shirt similar to that used in Case Study 2. Thirty-two VCA vibration units were press-fit to the garment. A large control unit was placed on the operator's back. The vibration units were grouped into four chains, each covering the front left, front right, back left, and back right.

The third step was to integrate the hardware into the simulated environment, which was built using Microsoft AirSim [42] with Unreal Engine 4 and Python APIs. Since the mapping between vibration units and 3D directions were known from the perceptual study, when a potential collision was detected, the system would trigger the unit that had the nearest direction with the colliding object in the space. This avoided the need to reproduce the physical layout of the vibration units in the system. The risks of colliding with surrounding obstacles were computed using control barrier functions [2] and scaled to $[0, 15]$ to match the actuators' intensity levels. If the risk of an obstacle was above 0.5, the vibration unit in the corresponding direction would be activated by sending commands through a Python server.

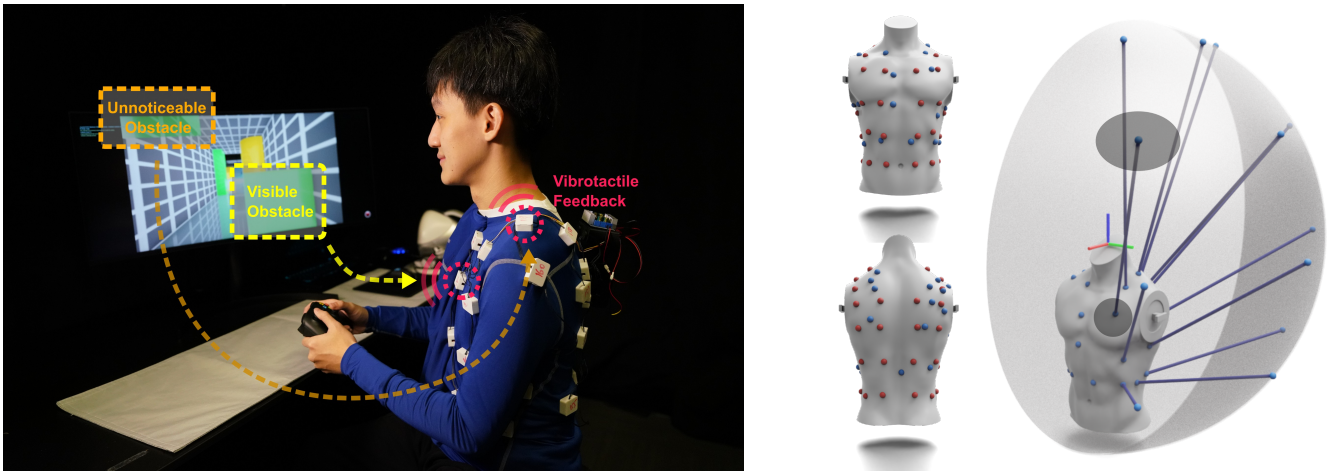
5.3.2 User Study Design. A preliminary study was conducted to evaluate the effectiveness of the vibrotactile system on teleoperation performance. The simulation environment ran on an Alienware M15 laptop with RTX 3060Ti GPU. Twelve participants were recruited for the study (11 male, 1 female; mean age = 24 years, std = 3 years). Seven participants had previous experience operating quadrotors. Each study lasted 70 minutes and each participant received $40 as compensation. The study was approved by our institutional research ethics board.

During the study, each participant went through three feedback conditions in a randomized order: no feedback (**NA**), force shared control (**FSC**), and vibrotactile shared control (**VSC**). For the NA and VSC conditions, participants used an Xbox controller for input, which had control mechanism similar to an RC controller. The output device for the VSC condition was the vibrotactile system. For the FSC condition, a Novint Falcon haptic joystick [14] was used for both input and output, similar to previous studies [30, 38].

For each condition, the experimental scene was a $5m \times 5m \times 50m$ tunnel (Figure 17a) that required participants to steer quadrotors forward, right, or upward. Different flying directions were used to vary visual information capacity as obstacles are often outside the camera's field of view when flying right and upward. The tunnel contained fifteen objects that were randomly placed to avoid learning effects. Based on these conditions, we hypothesize that (a) vibrotactile feedback would enhance teleoperation performance compared to no feedback, and (b) the improvement would be more effective in the right and upward directions than the forward direction.

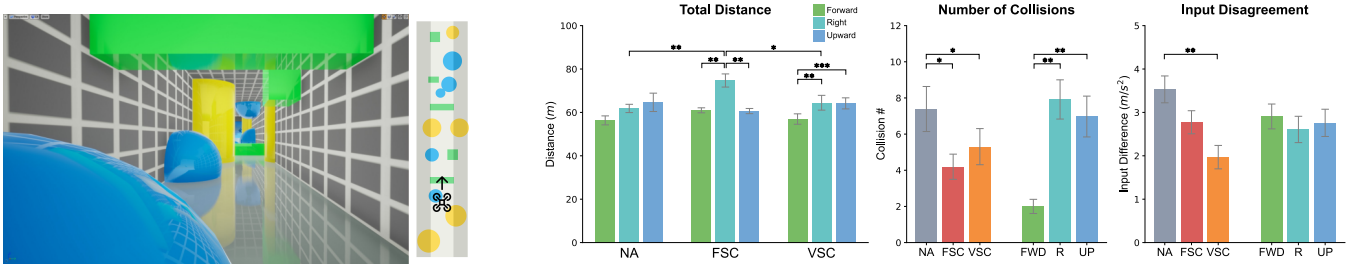
5.3.3 Analysis and Results. We evaluated three metrics: total distance travelled, number of collisions, and input disagreement (i.e., the deviation of user steering commands from the safe input range during flight). Using a two-way repeated measures ANOVA in SPSS, we assessed the effect of feedback condition and flying direction. Post-hoc pairwise comparisons with a Bonferroni correction were conducted for significant results (Figure 17b).

The RM-ANOVA determined that feedback condition had a significant effect on total distance travelled ($F(2, 22) = 3.585, p < 0.05$), as did with flying direction ($F(2, 22) = 17.868, p < 0.001$). The interaction between feedback condition and flying direction was also significant ($F(1.957, 21.527) = 4.916, p < 0.01$). The post-hoc analysis of the interaction effect revealed that FSC-R had longer distance than FSC-FWD ($p < 0.01$), FSC-UP ($p < 0.01$), NA-R ($p < 0.01$), and VSC-R ($p < 0.05$), while VSC-FWD had a significantly shorter distance than VSC-R ($p < 0.01$) and VSC-UP ($p < 0.001$).



(a) (b)

Figure 16: (a) The vibrotactile device designed during Case Study 3 assists UAV operators with collision avoidance by delivering obstacle directions via multi-point vibrotactile feedback on the body. Operators not only see and feel visible obstacles (yellow), but also perceive obstacles outside their field of view (orange). (b) The layout of vibration units on the upper body. Red dots represent the initial layout with uniform spacing, while blue dots represent the final layout and are mapped to directions in the 3D space.



(a) (b)

Figure 17: (a) A first-person view of the simulation environment used in Case Study 3's study, with randomly positioned obstacles and a top-down view on the right. (b) The results for the total distance travelled, number of collisions, and input disagreement metrics for the feedback conditions (NA, FSC, VSC) and flying directions (forward-FWD, right-R and upward-UP). The error bars represent the standard error of the mean (SEM), $^{*} = p < 0.05$, $^{**} = p < 0.01$, and $^{***} = p < 0.001$.

For the number of collisions, the RM-ANOVA determined that feedback condition had a significant effect ($F(2, 22) = 8.095, p < 0.01$), as did flying direction ($F(2, 22) = 15.653, p < 0.001$). The interaction between feedback condition and flying direction was also not found to be significant ($F(2.168, 23.846) = 1.910, p = 0.168$). The pairwise comparisons between feedback conditions revealed that NA caused more collisions than FSC ($p < 0.05$) and VSC ($p < 0.05$). The pairwise comparisons of flying directions revealed that flying forward caused fewer collisions than right ($p < 0.01$) and upward ($p < 0.01$).

For input disagreement, the RM-ANOVA determined that feedback condition had a significant effect ($F(2, 22) = 4.798, p < 0.05$), while flying direction did not ($F(1.254, 13.789) = 0.628, p = 0.476$). The interaction between feedback condition and flying direction was also not significant ($F(4, 44) = 1.566, p = 0.200$). The pairwise comparison between feedback conditions revealed that there was less disagreement with VSC compared to NA ($p < 0.01$)

In summary, VSC caused fewer collisions and less input disagreement compared to NA, and achieved comparable performance with FSC. Additionally, the right and upward directions had more collisions than the forward condition, highlighting the increasing reliance users had on haptic feedback when under limited visual capacity. Thus, our hypotheses were accepted.

5.3.4 Lessons Learned. The process of using VibraForge to build the teleoperation vibrotactile system demonstrated several benefits. First, the flexible design (DR4) allowed us to iterate on the layout of vibration units rapidly during the perceptual study. From the initial 46 actuator layout to the final 32 actuator layout, we could easily adjust the positions of vibration units on the body and test various

spatial mappings. Second, the low latency vibration responses (DR2) also provided instant feedback for participants, which is essential in high speed teleoperation tasks. In addition, the varied intensities (DR2) provided more nuanced information for participants, helping them prioritize their responses towards obstacles with higher risks. Together, these benefits demonstrate the usability of the toolkit for a complex application.

We also encountered several design challenges. One issue was that participants reported that the same intensity level was perceived differently across body areas. Locations with bone contacts were perceived to be stronger, while those with more subcutaneous tissues were perceived to be weaker. This variability may affect feedback precision during critical applications, and require a deeper investigation into how this can be overcome. Another issue was encountered during the layout iteration. Because the final layout was relatively sparse, the original connection wires that were used were too short. Therefore, we had to solder and assemble new wires. This, however, helped us make the toolkit adaptable to more diverse future use cases.

6 Discussion

In this section, we discussed the implications drawn from case studies, including strengths and limitations of VibraForge. Additionally, from our first-hand experiences in case studies, we synthesized a generalized five-step pipeline for building spatialized vibrotactile systems. Potential applications of VibraForge were highlighted at the end.

6.1 Strengths of VibraForge

VibraForge aims to provide a "least resistance path" [19] towards creating spatialized vibrotactile systems, which was demonstrated through the three case studies.

VibraForge offers great scalability. Systems designed in the case studies are of different scales, ranging from sleeve-type phonemic display with 24 vibration units on the forearm, to body-scale system with over 100 vibration units spanning across the upper body. Compared existing toolkits that are often constrained intrinsically to less than 16 actuators, VibraForge's chain connection method enables designers to easily scale their system to more than 100 vibration units, significantly expanding the potential design space. Furthermore, adding a new unit to an existing system is also as simple as plug-and-play, without the need for any soldering or programming.

VibraForge offers great expressivity. Information of various complexity are adequately conveyed through spatial vibrotactile feedback in the case studies, ranging from simple on/off status for VR collision detection, to continuous risk levels in teleoperation and symbolic phoneme representation in the phonemic display. Compared with prior spatialized vibrotactile systems that are constrained to using ERMs and low-bandwidth control, our toolkit offers various actuator options and provides fine-grained control over vibration characteristics of each actuator. This enables vibration units to reliably reproduce audio waveforms in technical evaluations.

The GUI Editor of VibraForge features usability. In the phonemic display study, it effectively expedited the pattern authoring process by reducing the average pattern authoring time to one minute. Compared with prior design interfaces, our editor highlighted the canvas and the timeline that provided substantial support for multi-actuator pattern design.

VibraForge also features portability and flexibility. Compared with the original phonemic display that was wired to a bulky desktop audio device, components used in our prototype were more portable and easy to wear, such as the small control unit that communicates via Bluetooth. In the UAV teleoperation study, flexibility ensured that vibration units could be frequently adjusted during the layout optimization perceptual study. Compared to using commercial devices that often have fixed feedback positions, VibraForge enables designers to easily adjust the positions of vibrations units, thus enabling rapid design iterations.

6.2 Limitations of VibraForge

Although VibraForge can enable designers to quickly prototype and evaluate new multi-actuator vibrotactile devices and experiences, the toolkit does not come without limitations. For example, the open-loop control used in the chain connection topology does pose some issues. First, when a control unit sends a vibration command, it is unclear if the command was successfully transmitted to the target vibration unit or not. This is especially important when sending a "stop" command, because the vibration unit won't stop vibrating if the commands was missed during the transmission process. Second, when multiple actuators were activated, there was a noticeable voltage drop along the chains. Although the technical evaluation showed that the supply voltage remained in the MCU operating range and actuator voltage range, it would still affect the users' subjective experience of the vibrations.

By adding extra command receiver pins on the control unit, one could instead use closed-loop control to solve these problems. With such a design, the vibration units and the control unit would form a loop, wherein the last vibration unit of the chain would be connected back to the control unit using a receiver pin. Whenever a vibration unit receives a command for itself, it would not only process the command and trigger the vibration, but also pass a confirmation message to the next unit. When the last unit in the chain receives this confirmation message, it would send the message back to the central unit, thus closing the loop. In cases where commands fail during the transmission process, the central unit would miss the confirmation message and resend the message. A closed loop connection would also reduce the voltage drop along the chain, as both ends of the chain would be connected to the power source. An LTspice simulation shows that an open-loop, eight-unit chain would result in voltage drop from 5 V to 3.1 V, while a closed-loop version would increase the voltage to 4.2 V.

Additionally, while the GUI Editor's waveform design was advantageous for the phonemic display pattern design, designers would face difficulties when designing more complex spatial patterns. For example, if a designer wanted to create the effect of actuators vibrating from a central point and expanding out in concentric circles, a designer would have to manually design the waveform, assign the waveform to each vibration unit, and specify the start and end time, which is very tedious. A better solution would be to use graphical

design metaphors and breakdown spatial patterns to low-level animations of point, line, and shapes. By leveraging phantom sensation illusions [39], designers would only need to draw the shapes and trajectories, and they would be automatically converted to vibration commands for each unit. In this way, complex patterns could be created with ease.

Our case studies also emphasized the need for an automation pipeline to detect the positions of vibration units on the body and convert them to virtual layout that can be later used in various programming environment. We believe this need can be fulfilled with computer vision and motion capture techniques in the future. Using RGB images of the body, computer vision methods could detect the human body outline, segment vibration units from the body, and record the units' positions. Similarly, retro-reflective markers could be mounted on top of each vibration unit to create a tracking object in the motion capture system. These positions could then be tracked in real-time and loaded into a development environment such as Unity Engine. In this way, designers could focus on the interaction design rather than manually creating the virtual layout.

6.3 Pipeline for Creating Spatialized Vibrotactile Systems

Synthesizing the processes followed during the three case studies, herein we summarize a generalized five-step pipeline for creating spatialized vibrotactile systems.

The first step is to determine **what** information to be delivered through vibrotactile feedback. For example, the purpose of the phonemic display was to convey vowels and consonants to deaf and hard-of-hearing people, so the types of vowels and consonants in words were the target information. In the drone teleoperation scenario, the key information was the potential risk of colliding with nearby obstacles and could be further divided into the directions of obstacles and their risk levels.

The second step is to decide **how** to convey the information. Since vibrations can be configured with different characteristics such as intensity, frequency, duration, position, and so on, the common practice is to breakdown the information and deliver it through multiple channels to maximize information transmission [46]. For example, vowels and consonants for the phonemic displays were distinguished by using moving patterns and static patterns, while different consonants were distinguished by actuator position, frequency modulation and vibration duration. As shown in prior research, providing information through multiple channels with fewer levels in each channel can enhance the overall information transfer for vibrotactile systems.

The third step is to decide **where** to install the system and derive the layout of vibration units. In cases where the information was spatially connected to the surrounding environment, the layout should have a good mapping between tactile positions and spatial direction. For example, in the VR fitness application, vibration units were mainly placed on the arm to provide localized feedback, indicating the arm pose differences between the user and the wall. In the UAV teleoperation, locations of vibrations were perceptually mapped to the directions of incoming obstacle. Perceptual limits should also be considered, e.g., spatial acuity indicated the upper limits of layouts, as adding more actuators to the layout would not increase the information transmission.

The fourth step is to **build** the system. Control units and vibration units can be chosen based on the scale of the system. Learning from the building process of case studies, We recommend using the small control unit for localized systems such as sleeves and headbands, as well as applications that require short-term usage. Alternatively, the large control unit is suitable for long-term usage and body-scale systems, such as immersive virtual experience and robot teleoperation. When designing vibration patterns, if the information involves complex waveforms, designers can utilize the waveform library in the GUI Editor to expedite the authoring process, just like how we designed patterns for the phonemic display. If the information only requires changing levels of intensity or frequency, the Python server with APIs should be handy.

The final step is to **test** the system and **iterate** on the design. If the desired information cannot be transmitted as expected, the designer can reconfigure the system by changing the layout or experiment different characteristics for conveying the information.

6.4 Potential Applications for VibraForge

A toolkit such as VibraForge could benefit several domains. In VR, spatial vibrotactile feedback can enhance immersive social experiences. In VR social platforms, adding tactile interactions can help users express their emotions through social touch gestures [44]. If developers are creating devices for users with sensory issues in certain areas of their body, they can leverage the toolkit to quickly and easily adjust the layout of actuators to avoid discomfort.

During human-robot interaction, spatial feedback could be useful for assisting humanoid robot teleoperation. Humanoid robot training collects data from human operators performing tasks remotely. However, the absence of tactile feedback mechanisms makes it difficult for operators to execute daily tasks that are typically performed with ease using their own bodies [31]. Spatial tactile feedback can provide additional information about remote interaction, such as when a robot hits a wall or other obstacle. The choice of tactile feedback positions can also be configured to match the positions of sensors on the robot. This feedback could thus increase an operator's situational awareness and close the interaction loop.

Last but not least, we envision that VibraForge can be used to build custom prototypes for tactile perceptual studies, such as those measuring spatial acuity, information transfer, or other perceptual characteristics. For example, Park et al. performed perceptual studies to measure information transfer across five body positions [32]. These positions were not chosen by the researchers but instead were constrained by the design of the commercial system they used (i.e., tactile modules from bHaptics [4]). By using VibraForge, researchers could freely adjust the layout of vibration units on the body and adapt it to any test they want to conduct.

7 Conclusion

Acknowledging the limits with existing spatialized vibrotactile feedback systems and toolkits, we built the VibraForge toolkit. Our toolkit enables scalability via a modular design and the use of chain-connection data transmission methods. Technical evaluations validated the toolkit design, and case studies demonstrated the wide

range of experiences that could be authored. The lessons learned during the case studies contributed to our understanding of a generalized pipeline for creating spatialized vibrotactile systems, and highlighted potential research directions of toolkit refinement and future applications. The toolkit is in its final stages of preparation, and will be released to the public later this year. We are excited to see how researchers and designers will harness the toolkit to elevate their own research and push the boundary of design innovation when using spatialized vibrotactile feedback.

Acknowledgments

References

[1] Marco Aggravi, Florent Pausé, Paolo Robuffo Giordano, and Claudio Pacchierotti. 2018. Design and Evaluation of a Wearable Haptic Device for Skin Stretch, Pressure, and Vibrotactile Stimuli. *IEEE Robotics and Automation Letters* 3, 3 (2018), 2166–2173. https://doi.org/10.1109/LRA.2018.2810887

[2] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. 2019. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*. IEEE, 3420–3431.

[3] Karlin Bark, Emily Hyman, Frank Tan, Elizabeth Cha, Steven A Jax, Laurel J Buxbaum, and Katherine J Kuchenbecker. 2014. Effects of vibrotactile feedback on human learning of arm motions. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 23, 1 (2014), 51–63.

[4] bHaptics. 2024. bHaptics: Tactile Feedback for VR, Gaming, and Music. https://www.bhaptics.com/

[5] Adam M Brandt. 2009. Haptic Collision Avoidance for a Remotely Operated Quadrotor UAV in Indoor Environments. (2009).

[6] Seungmoon Choi and Katherine J. Kuchenbecker. 2013. Vibrotactile Display: Perception, Technology, and Applications. *Proc. IEEE* 101, 9 (2013), 2093–2104. https://doi.org/10.1109/JPROC.2012.2221071

[7] Artem Dementyev, Pascal Getreuer, Dimitri Kanevsky, Malcolm Slaney, and Richard F Lyon. 2021. VHP: vibrotactile haptics platform for on-body applications. In *The 34th Annu. ACM Symp. on User Interface Softw. and Technol.* 598–612.

[8] Abdulmotaleb El Saddik, Mauricio Orozco, Mohamad Eid, and Jongeun Cha. 2011. *Haptics technologies: Bringing touch to multimedia.* Springer Science & Business Media.

[9] Hesham Elsayed, Martin Weigel, Florian Müller, Martin Schmitz, Karola Marky, Sebastian Günther, Jan Riemann, and Max Mühlhäuser. 2020. Vibromap: Understanding the spacing of vibrotactile actuators across the body. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 4 (2020), 1–16.

[10] Fox. 2008. Hole in the Wall (American game show). https://en.wikipedia.org/wiki/Hole_in_the_Wall_(American_game_show) Accessed: September, 2023.

[11] Xiaolei Hou, Pengfei Fang, and Yaohong Qu. 2017. Dynamic kinesthetic boundary for haptic teleoperation of unicycle type ground mobile robots. In *2017 36th Cine. Control Conf. (CCC)*. IEEE, 6246–6251.

[12] Bingjian Huang, Paul H. Dietz, and Daniel Wigdor. 2024. Investigating the Effects of Intensity and Frequency on Vibrotactile Spatial Acuity. *IEEE Transactions on Haptics* (2024), 1–13. https://doi.org/10.1109/TOH.2024.3350929

[13] Bingjian Huang, Zhecheng Wang, Qilong Cheng, Siyi Ren, Hanfeng Cai, Antonio Alvarez Valdivia, Karthik Mahadevan, and Daniel Wigdor. 2024. AeroHaptix: A Wearable Vibrotactile Feedback System for Enhancing Collision Avoidance in UAV Teleoperation. *arXiv preprint arXiv:2407.12105* (2024).

[14] Novint Technologies Inc. 2024. Novint Falcon. https://hapticshouse.com/pages/novints-falcon-haptic-device. Accessed: Feb 24, 2024.

[15] Ali Israr, Siyan Zhao, Kyna McIntosh, Zachary Schwemler, Adam Fritz, John Mars, Job Bedford, Christian Frisson, Ivan Huerta, Maggie Kosek, Babis Koniaris, and Kenny Mitchell. 2016. Stereohaptics: a haptic interaction toolkit for tangible virtual experiences. In *ACM SIGGRAPH 2016 Studio* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 13, 57 pages. https://doi.org/10.1145/2929484.2970273

[16] Yei Hwan Jung, Jae-Young Yoo, Abraham Vázquez-Guardado, Jae-Hwan Kim, Jin-Tae Kim, Haiwen Luan, Minsu Park, Jaeman Lim, Hee-Sup Shin, Chun-Ju Su, et al. 2022. A wireless haptic interface for programmable patterns of touch across large areas of the skin. *Nature Electronics* 5, 6 (2022), 374–385.

[17] Oliver Beren Kaul and Michael Rohs. 2017. Haptichead: A spherical vibrotactile grid around the head for 3d guidance in virtual and augmented reality. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 3729–3740.

[18] Odders Lab. 2019. OhShape. https://store.steampowered.com/app/1098100/OhShape/ Accessed: September, 2023.

[19] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–17. https://doi.org/10.1145/3173574.3173610

[20] Paul Lemmens, Floris Crompvoets, Dirk Brokken, Jack Van Den Eerenbeemd, and Gert-Jan de Vries. 2009. A body-conforming tactile jacket to enrich movie viewing. In *World Haptics 2009-Third Joint EuroHaptics conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*. IEEE, 7–12.

[21] Yan LI, Yuki OBATA, Miyuki KUMAGAI, Marina ISHIKAWA, Moeki OWAKI, Natsuki FUKAMI, and Kiyoshi TOMIMATSU. 2013. A design study for the haptic vest as a navigation system. *International Journal of Asia Digital Art and Design Association* 17, 1 (2013), 10–17.

[22] Robert W. Lindeman, Robert Page, Yasuyuki Yanagida, and John L. Sibert. 2004. Towards full-body haptic feedback: the design and deployment of a spatialized vibrotactile feedback system. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology* (Hong Kong) *(VRST '04)*. Association for Computing Machinery, New York, NY, USA, 146–149. https://doi.org/10.1145/1077534.1077562

[23] Robert W Lindeman, Yasuyuki Yanagida, Kenichi Hosaka, and Shinji Abe. 2006. The TactaPack: A wireless sensor/actuator package for physical therapy applications. In *2006 14th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*. IEEE, 337–341.

[24] Cephise Louison, Fabien Ferlay, and Daniel R Mestre. 2017. Spatialized vibrotactile feedback contributes to goal-directed movements in cluttered virtual environments. In *2017 IEEE Symp. on 3D User Interfaces (3DUI)*. IEEE, 99–102.

[25] Jinlong Machinery and Electronics. Co. 2022. LRA - Haptic Motors. http://en.kotl.com.cn/prod_view.aspx?TypeId=75&Id=252&Fid=t3:75:3

[26] Jonatan Martínez, Arturo S García, Miguel Oliver, José P Molina, and Pascual González. 2014. Vitaki: a vibrotactile prototyping toolkit for virtual reality and video games. *International Journal of Human-Computer Interaction* 30, 11 (2014), 855–871.

[27] Inc. Meta Platforms. 2023. Haptics SDK Studio for Meta Quest VR. https://developer.oculus.com/blog/haptics-sdk-studio-meta-quest-vr/ Accessed: 2024-08-26.

[28] Inc Meta Platforms. 2023. Meta Quest 2. https://www.meta.com/ca/quest/products/quest-2/?utm_source=www.google.com&utm_medium=oculusredirect

[29] Kouta Minamizawa, Yasuaki Kakehi, Masashi Nakatani, Soichiro Mihara, and Susumu Tachi. 2012. TECHTILE toolkit: a prototyping tool for design and education of haptic media. In *Proceedings of the 2012 Virtual Reality International Conference*. 1–2.

[30] Sammy Omari, Minh-Duc Hua, Guillaume Ducard, and Tarek Hamel. 2013. Bilateral haptic teleoperation of VTOL UAVs. In *2013 IEEE Int. Conf. on Robot. and Automat.* IEEE, 2393–2399.

[31] Claudio Pacchierotti and Domenico Prattichizzo. 2024. Cutaneous/Tactile Haptic Feedback in Robotic Teleoperation: Motivation, Survey, and Perspectives. *IEEE Transactions on Robotics* 40 (2024), 978–998. https://doi.org/10.1109/TRO.2023.3344027

[32] Jaejun Park, Junwoo Kim, Sangyoon Han, Chaeyong Park, Junseok Park, and Seungmoon Choi. 2023. Information Transfer of Full-Body Vibrotactile Stimuli: An Initial Study with One to Three Sequential Vibrations. In *2023 IEEE World Haptics Conference (WHC)*. IEEE, 41–47.

[33] Evan Pezent, Brandon Cambio, and Marcia K O'Malley. 2020. Syntacts: Open-source software and hardware for audio-controlled haptics. *IEEE Transactions on Haptics* 14, 1 (2020), 225–233.

[34] Helena Pongrac. 2006. Vibrotactile perception: Differential effects of frequency, amplitude, and acceleration. In *2006 ieee international workshop on haptic audio visual environments and their applications (have 2006)*. IEEE, 54–59.

[35] Deepa Prabhu, Muhammad Mehedi Hasan, Lisa Wise, Clare MacMahon, and Chris McCarthy. 2020. VibroSleeve: A wearable vibro-tactile feedback device for arm guidance. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 4909–4912.

[36] Inc. PUI Audio. 2023. HD-VA3222. https://puiaudio.com/product/haptics/hd-va3222 Accessed: September, 2023.

[37] Charlotte M Reed, Hong Z Tan, Zachary D Perez, E Courtenay Wilson, Frederico M Severgnini, Jaehong Jung, Juan S Martinez, Yang Jiao, Ali Israr, Frances Lau, et al. 2018. A phonemic-based tactile display for speech communication. *IEEE transactions on haptics* 12, 1 (2018), 2–17.

[38] Sergio Reyes, Hugo Romero, Sergio Salazar, Rogelio Lozano, and Omar Santos. 2015. Outdoor haptic teleoperation of a hexarotor UAV. In *2015 Int. Conf. on Unmanned Aircr. Syst. (ICUAS)*. IEEE, 972–979.

[39] Oliver S Schneider, Ali Israr, and Karon E MacLean. 2015. Tactile animation by direct manipulation of grid displays. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. 21–30.

[40] Oliver S Schneider and Karon E MacLean. 2016. Studying design process and example use with Macaron, a web-based vibrotactile effect editor. In *2016 IEEE Haptics Symposium (Haptics)*. IEEE, 52–58.

[41] Hasti Seifi, Kailun Zhang, and Karon E MacLean. 2015. VibViz: Organizing, visualizing and navigating vibration libraries. In *2015 IEEE World Haptics Conference (WHC)*. IEEE, 254–259.

[42] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. 2017. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Field and Service Robot*. arXiv:arXiv:1705.05065 https://arxiv.org/abs/1705.05065

[43] Supernatural. 2024. Supernatural - Virtual Reality Fitness. https://www.getsupernatural.com/. Accessed: 2024-09-02.

[44] Philipp Sykownik and Maic Masuch. 2020. The experience of social touch in multi-user virtual reality. In *Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology*. 1–11.

[45] David Antón Sánchez and René Bohne. 2015. OpenVNAVI: A Vibrotactile Navigation Aid for the Visually Impaired. https://hci.rwth-aachen.de/openvnavi Accessed: September, 2023.

[46] Hong Z Tan, Seungmoon Choi, Frances WY Lau, and Freddy Abnousi. 2020. Methodology for maximizing information transmission of haptic devices: A survey. *Proc. IEEE* 108, 6 (2020), 945–965.

[47] Unity Technologies. 2020. Unity Editor: What's New in 2020.3.31. https://unity.com/releases/editor/whats-new/2020.3.31

[48] Mihail Terenti and Radu-Daniel Vatavu. 2023. VIREO: Web-based Graphical Authoring of Vibrotactile Feedback for Interactions with Mobile and Wearable Devices. *International Journal of Human–Computer Interaction* 39, 20 (2023), 4162–4180.

[49] Ronald T Verrillo, Anthony J Fraioli, and Robert L Smith. 1969. Sensation magnitude of vibrotactile stimuli. *Perception & Psychophysics* 6, 6 (1969), 366–372.

[50] Vicon. 2023. Vicon Shogun Software System. https://www.vicon.com/software/shogun/ Accessed: September, 2023.

[51] Ingrid MLC Vogels. 2004. Detection of temporal delays in visual-haptic interfaces. *Human Factors* 46, 1 (2004), 118–134.

[52] K. Wang and et al. 2021. A Research on Sensing Localization and Orientation of Objects in VR with Facial Vibrotactile Display. In *Virtual, Augmented and Mixed Reality. HCII 2021. Lecture Notes in Computer Science*, J.Y.C. Chen and G. Fragomeni (Eds.), Vol. 12770. Springer, Cham. https://doi.org/10.1007/978-3-030-77599-5_17

[53] Travis J West, Alexandra Bachmayer, Sandeep Bhagwati, Joanna Berzowska, and Marcelo M Wanderley. 2019. The design of the body: suit: score, a full-body vibrotactile musical score. In *Human Interface and the Management of Information. Information in Intelligent Systems: Thematic Area, HIMI 2019, Held as Part of the 21st HCI International Conference, HCII 2019, Orlando, FL, USA, July 26-31, 2019, Proceedings, Part II 21*. Springer, 70–89.

[54] Dennis Wittchen, Katta Spiel, Bruno Fruchard, Donald Degraen, Oliver Schneider, Georg Freitag, and Paul Strohmeier. 2022. Tactjam: An end-to-end prototyping suite for collaborative design of on-body vibrotactile feedback. In *16th Int. Conf. on Tangible, Embedded, and Embodied Interact.* 1–13.

[55] Heidi JB Witteveen, Hans S Rietman, and Peter H Veltink. 2015. Vibrotactile grasping force and hand aperture feedback for myoelectric forearm prosthesis users. *Prosthetics and orthotics international* 39, 3 (2015), 204–212.

[56] Yihong Wu, Lingyun Yu, Jie Xu, Dazhen Deng, Jiachen Wang, Xiao Xie, Hui Zhang, and Yingcai Wu. 2023. AR-Enhanced Workouts: Exploring Visual Cues for At-Home Workout Videos in AR Environment. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–15.

[57] Pengxiang Xia, Kevin McSweeney, Feng Wen, Zhuoyuan Song, Michael Krieg, Shuai Li, Xiao Yu, Kent Crippen, Jonathan Adams, and Eric Jing Du. 2022. Virtual telepresence for the future of ROV teleoperations: opportunities and challenges. In *SNAME Offshore Symp.* SNAME, D011S001R001.

[58] Dawei Zhang, Guang Yang, and Rebecca P Khurshid. 2020. Haptic teleoperation of uavs through control barrier functions. *IEEE Trans. on Haptics* 13, 1 (2020), 109–115.

AeroHaptix: A Wearable Vibrotactile Feedback System for Enhancing Collision Avoidance in UAV Teleoperation

Bingjian Huang, Zhecheng Wang, Qilong Cheng, Siyi Ren, Hanfeng Cai,
Antonio Alvarez Valdivia, Karthik Mahadevan, and Daniel Wigdor

***Abstract*— Haptic feedback enhances collision avoidance by providing directional obstacle information to operators during unmanned aerial vehicle (UAV) teleoperation. However, such feedback is often rendered via haptic joysticks, which are unfamiliar to UAV operators and limited to single direction force feedback. Additionally, the direct coupling between the input device and the feedback method diminishes an operators' sense of control and causes oscillatory movements. To overcome these limitations, we propose AeroHaptix, a wearable haptic feedback system that uses spatial vibrations to communicate multiple obstacle directions to operators simultaneously, without interfering the input control. The layout of vibrotactile actuators was determined via a perceptual study to eliminate perceptual biases and achieve uniform spatial coverage. A novel rendering algorithm, MultiCBF, extends control barrier functions to support multi-directional feedback. Our system evaluation showed that compared to the baseline condition, AeroHaptix effectively reduced the number of collisions and input disagreement. Additionally, operators reported that AeroHaptix was more helpful than the force feedback method, with comparable workload and situational awareness.**

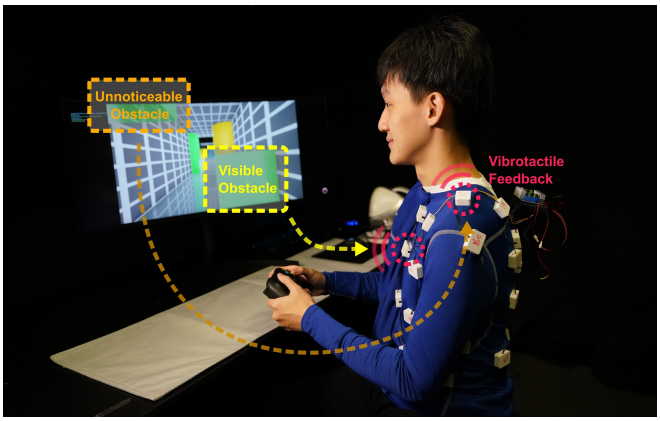


Fig. 1: AeroHaptix assists UAV operators with collision avoidance by delivering spatial vibrotactile feedback on the body about multiple obstacle directions. Operators can not only see and feel visible obstacles (yellow), but also perceive obstacles out of view (orange).

I. Introduction

Unmanned aerial vehicle (UAV) teleoperation enables operators to pilot UAVs beyond their visual line of sight, enabling for tasks to be completed in situations that are either difficult or dangerous for humans to be present. Teleoperation is challenging, however, because the operator is physically separated from the UAV, which limits an operator's ability to perceive obstacles and avoid collisions. Recent UAV teleoperation systems have utilized haptic feedback to deliver obstacle information so that operators can safely steer UAVs. To translate obstacle information into haptic feedback, prior work has developed collision avoidance algorithms such as parametric risk fields [1], time-to-impact [2], dynamic kinesthetic boundary [3], and control barrier functions (CBF) [4].

Despite the advancement of collision avoidance algorithms, the devices used to render haptic feedback are primarily commercial haptic joysticks with three degree-of-freedom (DoF) force feedback [5]–[7]. Although these devices have proven useful in collision avoidance tasks [8], [9], they are rarely adopted in real-world UAV operations due to the high transition cost from standard radio control (RC) controllers to such unfamiliar devices. The use of such devices is further constrained to indoor environments as haptic joysticks must be mounted to an immovable surface to provide accurate force feedback. Furthermore, the limited information bandwidth of haptic joysticks restricts force feedback to being rendered in one direction [4], [10], thus compromising situational awareness in environments with multiple obstacles. Since haptic joysticks exert force feedback on the hand, the direct coupling of input and output channels impairs UAV control precision, leading to oscillatory behavior in cluttered environments [9] and low user acceptance [11]. Alternative devices such as cable-driven exoskeletons [12], [13] solved the mobility issue, but added new challenges such as physical fatigue and limited degrees of freedom.

To address these limitations, we designed a wearable haptic feedback system, AeroHaptix, that renders vibrotactile feedback at thirty-two upper body positions to deliver obstacle directions. Since the feedback is provided as vibration cues on the body, it does not hinder hand movement and can be seamlessly integrated into existing teleoperation control workflows (e.g., using RC controllers). We conducted a perceptual study with ten participants to collect data on the mapping between body positions and spatial directions, and employed a neural network to generate the layout of vibrotactile actuators on the body. We also designed MultiCBF, a novel collision avoidance algorithm that extends previous CBF methods to render multi-directional haptic feedback. Our system evaluation showed that compared to the baseline condition, AeroHaptix effectively reduced the number of collisions and input disagreement. Additionally, operators reported that AeroHaptix was more helpful than the force feedback method, with comparable workload and situational awareness.

II. Related Work

Of most relevance to the present research is prior literature on (a) collision avoidance algorithms for UAV teleoperation and (b) vibrotactile feedback to convey spatial information.

A. Collision Avoidance Algorithms

Collision avoidance algorithms were first utilized for unmanned ground vehicle teleoperation. In 1998, Hong et al. proposed an artificial force field algorithm to generate artificial forces based on the potential fields of vehicle and obstacles [14]. This concept was later adopted for UAVs by Boschloo et al. [15] and Lam et al. [1], who developed basic risk field and parametric risk field algorithms respectively. These methods successfully reduced collisions but faced difficulties when users navigated through narrow spaces. To further improve operation, Brandt and Colton introduced time-to-impact and virtual string algorithms that calculated the virtual impact forces between a UAV and obstacles [2]. While these algorithms further reduced collisions and workloads, the force feedback they generated was found to cause UAVs to oscillate in cluttered environments [9].

Alternatively, other researchers have proposed algorithms that override user input with safer values, such as dynamic kinesthetic boundary [3], [16] and obstacle avoidance system for teleoperation [17]. While these algorithms eliminated collisions, they reduced operators' sense of control and user acceptance. More recently, Zhang et al. applied control barrier functions (CBF) to UAV teleoperation [4], which modify an input control signal to closely match the original while adhering to safety constraints. They also designed a haptic shared autonomy control scheme that enhanced operators' sense of control [8]. We extended their CBF algorithm to support multi-point haptic feedback and thus the simultaneous representation of multiple obstacles.

B. Vibrotactile Feedback to Convey Spatial Information

Vibrotactile feedback conveys sensory information to humans via actuators vibrating on the skin. Prior work has demonstrated the application of vibrotactile feedback in various domains where spatial information is crucial. For example, vibrotactile headbands have been used to assist users in locating 3D objects and improving spatial awareness in virtual reality [18], [19]. Vibrotactile actuators have also been placed on the body to enhance obstacle detection and navigation for blind and visually-impaired people [20]–[22]. Within robotics, vibrations have been used to communicate handover positions and predicted trajectories of robotic arms [23], [24], hydrodynamic flow near underwater robots [25], and obstacle positions around UGVs [26]. These examples reaffirm the applicability of using vibrotactile feedback to convey spatial information.

Compared to previous applications, delivering obstacle directions during UAV teleoperation is more challenging, because it requires numerous actuators placed in a custom layout on the body to precisely represent obstacle directions. Current vibrotactile systems such as TactJam [27] and VHP [28]) offer great customizability but only support up to 12 actuators. In contrast, solutions like bHaptics X40 [29]) support numerous actuators but have predetermined positions on the body and limited body coverage. Therefore, we developed custom hardware for AeroHaptix that overcame the above challenges.

III. AeroHaptix Hardware Design

To support UAV teleoperation tasks, AeroHaptix was designed to support numerous actuators, customized layouts, and low-latency control. Its design had three requirements:

- **R1**: support the fine-grained control of a large number of actuators to distinguish obstacles from different directions;
- **R2**: support the reconfiguration of actuator layouts so actuators could be freely adjusted as needed; and
- **R3**: support low-latency communication so multi-point feedback could be activated without a noticeable delay.

AeroHaptix had a central unit and multiple chains of vibration units (Figure 2). Each vibration unit consisted of a voice coil actuator (PUI Audio, Model# = HD-VA3222), a custom-designed PCB board, and 3D-printed parts. The actuator was 32 mm by 22 mm, with frequency responses from 80 - 500 Hz. When driven by 133 Hz 1.5 V_{rms} signal, it had a maximum acceleration of 2.52 Gp-p. The PCB included a PIC16F18313 Microcontroller Unit (MCU) and a DRV8837 H-bridge motor driver. The MCU received and transmitted UART commands and generated waveforms. When the MCU received a *start* signal, it sent a waveform to the H-bridge to activate the actuator. The MCU had fine-grained control (**R1**) over 16 intensity levels, 4 frequencies, and two waveforms. The 3D-printed parts included a cap and an enclosure that ensured components' stability during vibrations, and a bottom ring that enabled easy repositioning on garment (**R2**) via press-fit.

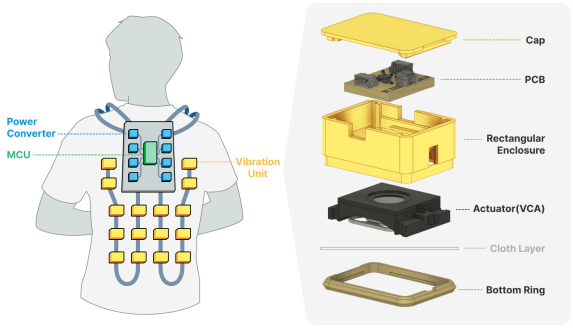


Fig. 2: Aerohaptix's hardware design, with an exploded view of a vibration unit.

To support low-latency communication (**R3**), we designed a chain-connection topology and a high-speed UART protocol (Figure 3). Each chain had a central unit, a power converter, and a maximum of 20 vibration units. Each vibration unit received a UART command from the previous unit and determined whether to execute it or propagate it to the next unit. The UART protocol transmitted two-byte messages at a baud rate of 115.2 kHz with parity check

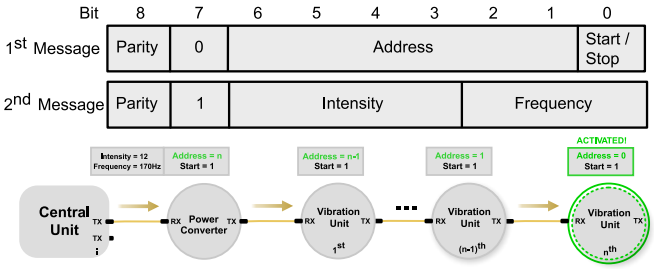


Fig. 3: The data transmission on each chain used a two-byte UART protocol. The central unit sent a command with address n, and each unit deducted the address by 1 until it reached the target unit.

bits. The first byte contained the target unit address and a *start/stop* toggle bit. The second byte contained vibration parameters for the target unit (i.e., intensity and frequency). A technical evaluation demonstrated that there was a 125 μs delay between two vibration units. With 20 units, the total delay was only 2.5 ms, fulfilling the requirement.

IV. Haptic Actuator Layout Optimization

To achieve comprehensive coverage of obstacle directions (**R1**), one naive approach is to uniformly distribute actuators around the body. However, previous research has shown that due to proprioceptive biases, a uniform actuator layout on the waist could lead to uneven coverage of 2D directions on the azimuth plane [30]. Therefore, we designed a perceptual study to collect data about position-direction mappings on the upper body. Then, we trained a neural network to generate an optimized actuator layout.

Ten participants (6 male, 4 female; mean age = 25 years, std = 2 years) were recruited for the study. Each study lasted 40 minutes and each participant received $20 CAD as compensation. The study was approved by our university's Research Ethics Board.

A. Data Collection Procedure

The initial layout used during the study was a 46-actuator grid that was uniformly distributed on the upper body (i.e., red dots in Figure 5). Each actuator was positioned equidistantly from its neighbors, with 18 actuators on the front and back, 3 on each side of the waist, and 2 on each shoulder. Following VibroMap [31], the inter-actuator distances were about 8 cm.

A Unity 3D virtual reality application was developed for the study. Participants wore a Meta Quest 2 headset and stood in a virtual space with coordinates surrounding them (Figure 4). They received a vibrotactile cue from an individual actuator and used an extended ray emitted from a Quest 2 controller to point in the perceived direction of the actuator. They then clicked the trigger button on the controller to confirm their perceived direction. Each participant completed 230 trials in a randomized order (i.e., 46 actuators x 5 repetitions).

B. Layout Optimization

We collected 2,300 mappings between an actuator positions on participants' body surfaces $p \in \Omega \subset \mathbb{R}^3$ and

Fig. 4: Left: An illustration of the virtual reality study environment where a user is pointing in the direction of the vibrotactile cue they perceived on their body. Right: Example data points collected from one participant, where points of the same color were from the same actuator.

the obstacle directions participants perceived (ϑ, φ) on a 2-sphere S^2, parameterized by a polar angle ϑ and an azimuth angle φ. This data was used to approximate an inverse mapping from S^2 to $\Omega \subset \mathbb{R}^3$ using a Multi-layer Perceptron neural network:

$$f_\theta = \arg\min_\theta \sum_{i=1}^{N} \|f_\theta(\vartheta,\varphi) - p\|^2 + \lambda \sum_{i=1}^{N} |\nabla f_\theta(\vartheta,\varphi)|, \quad (1)$$

where θ represented the neural network parameters and λ was the regularization term weight. The network featured five 64-unit hidden layers with `ReLU` activation. To promote mapping smoothness, we incorporated a total variation regularization term into the mean squared error loss function.

C. Final Layout

Previous studies have found that humans can discriminate force feedback directions on the hand with 30° resolutions [32], [33]. To achieve similar discriminability with vibrations on the body, we uniformly sampled 32 directions $\phi = (\vartheta, \varphi)$ in 3D space S^2, where $\vartheta \in \{\frac{\pi}{6}, \frac{\pi}{3}, \frac{\pi}{2}, \frac{2\pi}{3}\}$ and $\varphi \in \{-\frac{3\pi}{4}, -\frac{\pi}{2}, -\frac{\pi}{4}, 0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi\}$. We then used f_θ to find the corresponding actuator positions (i.e., blue dots in Figure 5). The final layout indicated that actuators should be positioned on body surface with greater curvatures (e.g., shoulders, waist) for optimal coverage of potential obstacle directions (**R1**). Directions with $\vartheta = \frac{5\pi}{6}$ were omitted from the final layout because they did not align with upper body vibrations perceived by participants.

In our collision avoidance algorithm MultiCBF, this optimized layout was represented by a set of 32 actuators $\mathcal{A} = \{\mathcal{A}_i\}_{i=1}^{32}$ and the corresponding directions set $\hat{r}$ where $\hat{r}_i$ represented the direction associated with actuator $\mathcal{A}_i$.

V. Collision Avoidance Algorithm

In prior work, collision avoidance algorithms such as PRF [34] and CBF [4] only considered single-directional feedback due to the limited bandwidth of haptic joysticks. Since AeroHaptix was designed to support multi-point vibrations (**R3**), we extended CBF [4] to build a customized multi-directional feedback algorithm, **MultiCBF**:

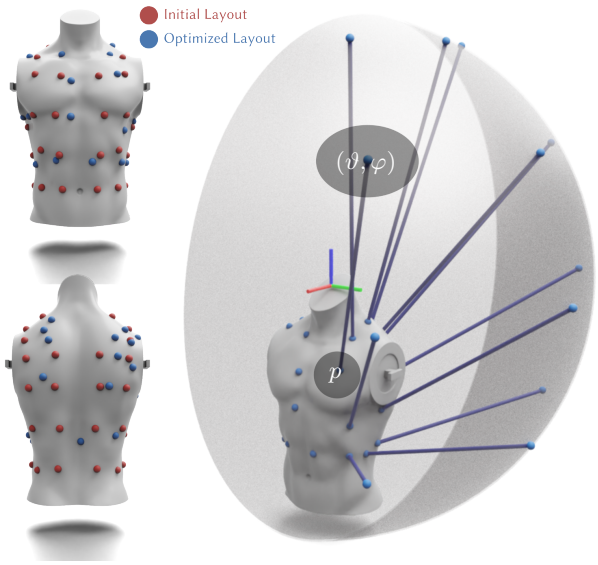


Fig. 5: The vibrotactile actuator layout, where the red dots depict the initial uniform distribution layout used in the perception study and the blue dots depict the final optimized layout. The lines illustrate the mappings between the perceived directions and body positions on the left side of the body.

1) Consider that the continuous-time dynamics of a UAV can be modelled with a double integrator, where the control input u corresponds to the acceleration command of the UAV. Let $x = \begin{bmatrix} q & \dot{q} \end{bmatrix}^T$ with q and $\dot{q}$ being the position and the velocity of the UAV respectively. The dynamics of the system then become

$$\dot{x} = \begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix} = f(x) + g(x)u, \tag{2}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$, $u \in \mathcal{U} \subset \mathbb{R}^m$, $f : \mathcal{X} \to \mathbb{R}^n$ and $g : \mathcal{X} \to \mathbb{R}^{n \times m}$ are Lipschitz continuous functions.

2) Consider a space with a set of obstacles $\mathcal{B} = \{b_i\}_{i=1}^{|\mathcal{B}|}$, where every obstacle b_i is associated with the center of mass q_{b_i}. For each obstacle b_i, we construct a set of safety constraints $\mathcal{C}_i \in \mathcal{X}$ defined as

$$\mathcal{C}_i := \{x \in \mathcal{X} : h_i(x) \geq 0\}, \tag{3}$$

where $h_i : \mathcal{X} \to \mathbb{R}$ is a continuously differentiable function that defines the safety boundary of obstacle b_i, $h_i(x) = 0 \Leftrightarrow x \in \partial\mathcal{C}_i$.

3) We define the CBF $\mathcal{U}_{\texttt{CBF}}$ for second-order systems as the set of all control inputs that keep the systems in the safe set C. For the CBF associated with C_i

$$\begin{aligned} \mathcal{U}_{\texttt{CBF},i}(h_i(x)) = \{u \in \mathcal{U} : L_f^2 h_i(x) + L_g L_f h_i(x) u \\ + K[h_i(x)\ L_f h_i(x)]^T \ \} \geq 0, \end{aligned} \tag{4}$$

where $L_f h_i(x) = \nabla h_i(x)^T f(x)$, $L_g h_i(x) = \nabla h_i(x)^T g(x)$ are the Lie derivatives of $h_i(x)$, $L_f^2 h_i(x)$ is the second-order Lie derivative of $h_i(x)$, and K is a positive constant that adjusts the safety margin.

4) Given user input $u_{\texttt{ref}} \in \mathcal{U}$, the optimization of a safe input $u_{\texttt{safe},i}$ for obstacle b_i and its $h_i(x)$ can be formulated as a quadratic program to find a **local safe input** $u_{\texttt{safe},i} \in \mathcal{U}_{\texttt{CBF},i}$ (Eq. 4) that is closest to $u_{\texttt{ref}}$

$$u_{\texttt{safe},i} = \arg\min_{u \in \mathcal{U}} \frac{1}{2} \|u - u_{\texttt{ref}}\|^2 \text{ s.t. } u \in \mathcal{U}_{\texttt{CBF},i}(h_i(x)). \tag{5}$$

If $u_{\texttt{safe},i} \neq u_{\texttt{ref}}$, it means the user input $u_{\texttt{ref}}$ violates the safety constraint C_i, then an actuator $\mathcal{A}_j$ is triggered to notify the user. The choice of actuator $\mathcal{A}_j$ is determined by the most aligned actuator direction $\hat{r}_k$

$$j = \arg\max_{k \in 1:32} \left(\frac{q_{b_i} - q}{\|q_{b_i} - q\|} \cdot \hat{r}_k \right). \tag{6}$$

The vibration intensity I_j of actuator $\mathcal{A}_j$ is then determined by the difference between the user input $u_{\texttt{ref}}$ and the safe input $u_{\texttt{safe},i}$ multiplied with a gain factor K_v

$$I_j = \|K_v(u_{\texttt{safe},i} - u_{\texttt{ref}})\|. \tag{7}$$

Previous CBFs only considered a global safe input $u_{\texttt{safe}}$ computed from a global safety set $h(x) = \{h_i(x)\}_{i=1}^{|\mathcal{B}|}$ [4]. While this global safe input helped operators avoid obstacles, it was not sufficient to represent multiple obstacles and thus hindered operator's situational awareness. In contrast, MultiCBF computed local safe input for each obstacle and rendered haptic feedback independently.

VI. System Evaluation

To evaluate AeroHaptix's ability to assist UAV teleoperation, we designed a study in which participants maneuvered a simulated UAV through cluttered tunnels. Twelve participants were recruited from our university campus to participate in the study (11 male, 1 female; mean age = 24 years, std = 3 years). Seven participants had previous experience operating UAVs. Each experiment lasted 70 minutes and each participant received $40 CAD as compensation. The study was approved by our university's Research Ethics Board.

A. Experimental Conditions

Participants experienced three **feedback conditions**: no feedback (**NA**), force shared control (**FSC**), and vibrotactile shared control (**VSC**). For the NA and VSC conditions, participants used an Xbox controller for input, which had a control mechanism similar to an RC controller. The output device was AeroHaptix with vibrotactile feedback rendered using MultiCBF. For the FSC condition, a Novint Falcon haptic joystick [5] was used for both input and output, similar to previous studies [35], [36]. Force feedback during FSC was rendered using CBF [4]. We hypothesized that VSC would enhance collision avoidance and input safety compared to NA, and reduce workload and increase sense of control compared to FSC.

Because previous studies often used complex experiment environments with varying visual capacities [3], [9], [17], it was unclear when haptic feedback was beneficial. In this study, we used three **flying directions** to assess how visual capacity influenced participants' reliance on haptic feedback: forward (**FWD**), right (**R**) and upward (**UP**). In the forward direction, participants received substantial visual information about upcoming obstacles because the UAV was oriented to

face the flying direction. In contrast, visual capacity in other directions were limited, and this may increase the difficulty for collision avoidance. Due to the MultiCBF algorithm, we also hypothesized that VSC would outperform FSC in delivering obstacle information.

Our hypotheses were that:

(H_1) compared to NA, VSC would reduce the number of collisions and input disagreement

(H_2) compared to flying forward, the number of collisions would be higher when flying right and upward

(H_3) compared to FSC, VSC would reduce task workload

(H_4) compared to FSC, VSC would increase participants' sense of control

(H_5) compared to FSC, VSC would be more effective at increasing situational awareness

B. Experimental Setup

The study was performed in a simulated UAV environment that was built using Microsoft AirSim [37]. During the study, participants operated a simulated quadrotor with a front-facing camera. The simulation was run on an Alienware M15 Laptop with RTX 3060 Ti GPU.

The experimental scene was a $5 \times 5 \times 50$m tunnel (Figure 6) that could face forward, right, or upward depending on the flying directions. The scene contained four planes and fifteen obstacles that were of three types, i.e., cubes, spheres, and cylinders. The safety boundary of plane b_i was defined as: $h_i(x) = h_i(\begin{bmatrix} q & \dot{q} \end{bmatrix}^T) = (q - q_{b_i}) \times \hat{n}_{b_i}$, where q_{b_i} was the center of mass and $\hat{n}_{b_i}$ was the unit normal of plane b_i. The safety boundaries of other obstacles were approximated using super-ellipsoids $h_i(x) = h(\begin{bmatrix} q & \dot{q} \end{bmatrix}^T) = (\frac{q_1 - q_{b_i,1}}{a_1})^n + (\frac{q_2 - q_{b_i,2}}{a_2})^n + (\frac{q_3 - q_{b_i,3}}{a_3})^n$, where q_{b_i} was the center of mass and a was the scaling vector of obstacle b_i. Obstacle positions were randomized to avoid learning effects.

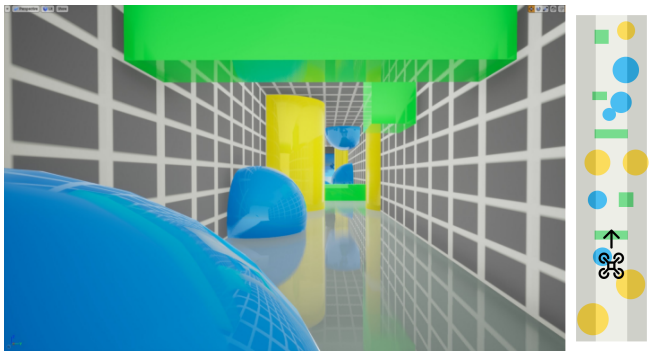


Fig. 6: The first-person view of the simulation environment used in the study, with randomly positioned obstacles. A top-down overview is on the right.

On each simulation frame, the system detected input commands from the controller, received UAV state updates from AirSim, and checked if safety constraints were violated using collision avoidance algorithms. If haptic feedback was enabled, then the system would send vibration or force feedback commands to the corresponding output device. At the end of each frame, the system updated the UAV's state in AirSim.

At the beginning of the study, the participant reviewed and signed a consent form. Then, the participant was briefed on the study purpose and overall procedure. Following this, the participant experienced the three feedback conditions in a randomized order (3×3 Latin square). For each feedback condition, the participant underwent a practice round with designated devices for five minutes. Then, the participant operated the UAV to fly through three tunnels, one for each flying direction. After each condition, the participant completed a questionnaire about task workload and their sense of control.

C. Metrics and Analysis

We collected both objective and subjective measurements. Objective measurements included the total distance travelled, number of collisions, and input disagreement (computed as the difference between participant input and safe input $\|u_{\texttt{ref}} - u_{\texttt{safe}}\|$), which evaluated the collision avoidance performance (**H1,H2**). For each metric, we conducted a two-way repeated measures ANOVA using SPSS. If significant effects were found, post-hoc pairwise comparisons using a Bonferroni correction were conducted.

Subjective measurements included NASA-TLX [38] questions that evaluated task load (**H3**), and four 7-point Likert questions adapted from previous work [8] that probed sense of control (**H4**):

1) How easy was it to control drone with this input device?
2) How much control did you feel you had over the drone?
3) How well did the drone's motion match your intention?
4) If you felt haptic feedback, how much did the feedback help you navigate the robot?

For each metric, we conducted a Friedman test. If significant effects were found, post-hoc pairwise comparisons using Wilcoxon signed-rank tests were conducted.

To evaluate situational awareness (**H5**), a pop-up window appeared at a random time during the flight for each direction (with the simulation paused) and participants were asked to report any perceived obstacles. This technique was adapted from situational awareness probing techniques like SAGAT and SPAM [39]. Reported obstacles were categorized into *visual obstacles* that were visible at pause, and *haptic obstacles* that were not seen but perceived through haptic feedback. Since the visual capacity is consistent across conditions, haptic obstacles represent the enhanced situational awareness through haptic feedback. A two-way repeated measures ANOVA is run to determine the effect of haptic feedback conditions and flying directions on situational awareness.

D. Results

1) Objective Measurements: For total distance travelled, the RM-ANOVA determined that feedback condition had a significant effect ($F(2, 22) = 3.585, p < 0.05$), as did flying direction ($F(2, 22) = 17.868, p < 0.001$). The interaction between feedback condition and flying direction was also significant ($F(1.957, 21.527) = 4.916, p < 0.01$). The post-hoc analysis of the interaction effect revealed that FSC-R had a significantly longer distance than FSC-FWD ($p < 0.01$),

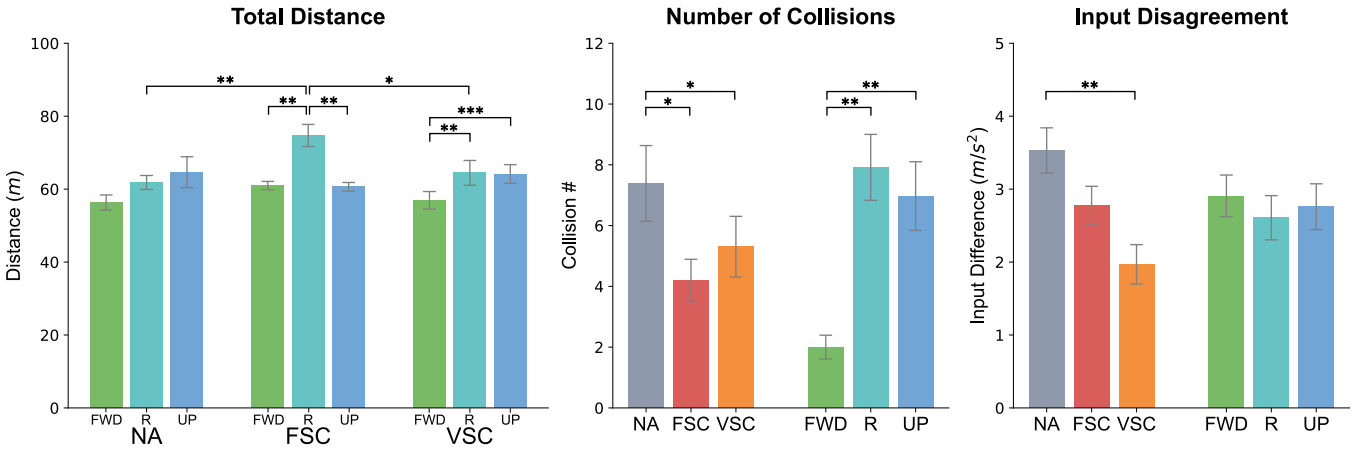


Fig. 7: Objective measurement results grouped by feedback condition (NA - no feedback, FSC - force shared control, VSC - vibrotactile shared control) and flying direction (FWD - forward, R - right, UP - upward). The error bars represent the standard error of the mean (SEM), $^{*}p < 0.05$, $^{**}p < 0.01$, and $^{***}p < 0.001$.

FSC-UP ($p < 0.01$), NA-R ($p < 0.01$), and VSC-R ($p < 0.05$), while VSC-FWD had a significantly shorter distance than VSC-R ($p < 0.01$) and VSC-UP ($p < 0.001$).

For the number of collisions, the RM-ANOVA determined that feedback condition had a significant effect ($F(2, 22) = 8.095, p < 0.01$), as did flying direction ($F(2, 22) = 15.653, p < 0.001$). No interaction was found between feedback condition and flying direction ($F(2.168, 23.846) = 1.910, p = 0.168$). Post-hoc pairwise comparisons revealed that NA caused more collisions than FSC ($p < 0.05$) and VSC ($p < 0.05$) and that flying forward caused fewer collisions than right ($p < 0.01$) and upward ($p < 0.01$).

For input disagreement, the RM-ANOVA determined that feedback condition had a significant effect ($F(2, 22) = 4.798, p < 0.05$), while flying direction did not ($F(1.254, 13.789) = 0.628, p = 0.476$). The interaction between feedback condition and flying direction was also not significant ($F(4, 44) = 1.566, p = 0.200$). Post-hoc pairwise comparisons revealed that there was less disagreement with VSC than NA ($p < 0.01$)

In summary, VSC caused fewer collisions and less input disagreement than NA. Additionally, the right and upward directions caused more collisions than the forward direction, highlighting the impact of reduced visual capacity when the camera was not aligned with the flying direction. Thus, **H1** and **H2** were accepted.

2) Subjective Measurements: physical demand: NA ¡ FSC, effort: NA ¡ FSC,

Friedman tests determined that feedback condition had a significant effects on physical demand ($\chi^2(2) = 12.950, p < 0.01$), effort ($\chi^2(2) = 7.294, p < 0.05$), overall task load ($\chi^2(2) = 6.048, p < 0.05$), Q3 Matching Intention ($\chi^2(2) = 7.056, p < 0.05$), and Q4 Haptic Usefulness ($\chi^2(2) = 21.378, p < 0.001$). Post-hoc pairwise comparisons revealed that NA required less physical demand than FSC ($p < 0.01$), VSC required less effort than FSC ($p < 0.05$), and VSC was more helpful than FSC ($p < 0.05$). Since significant differences were only found for some metrics, **H3** and **H4** were not accepted.

3) Situational Awareness: The numbers of reported haptic obstacles are reported in Figure 8. the RM-ANOVA determined that feedback condition had a significant effect ($F(1, 11) = 7.436, p < 0.05$), while flying direction did not ($F(2, 22) = 2.131, p = 0.143$). The interaction between feedback condition and flying direction was also not significant ($F(2, 22) = 0.517, p = 0.604$). Post-hoc pairwise comparisons revealed that VSC was more effective at enhancing situational awareness than FSC ($p < 0.05$). Therefore, **H5** was accepted.

VII. Discussion

In this section, we discuss the design implications arising from the experiment results.

A. Benefits of Vibrotactile Feedback

Vibrotactile feedback resolved the direct coupling problem with haptic joystick systems and ensured stable operation. When using FSC, operators struggled to understand the intent of the force feedback and often tried to override the input with manual force in the opposite direction. This inevitably resulted in risky and oscillatory movements in the tunnel, which was further confirmed by the longer flying distances and higher input disagreement. When using VSC, however, the decoupling of the input and output channels via the RC controller and AeroHaptix provided operators with full control over their UAV. Operators had sufficient time to process the vibrotactile cues and then steer the UAV towards a safe trajectory. Thus, VSC promoted safer and more cautious operation, reducing the risk of collisions in real world scenarios.

Additionally, vibrotactile feedback enhanced situational awareness by delivering multi-directional cues through the MultiCBF algorithm. More obstacles were identified through haptic feedback when using VSC than FSC. This indicated that operators could effectively understand vibrotactile feedback and became more aware of the environment.

We acknowledged that the subjective assessment of VSC, task workload and sense of control, showed no statistical

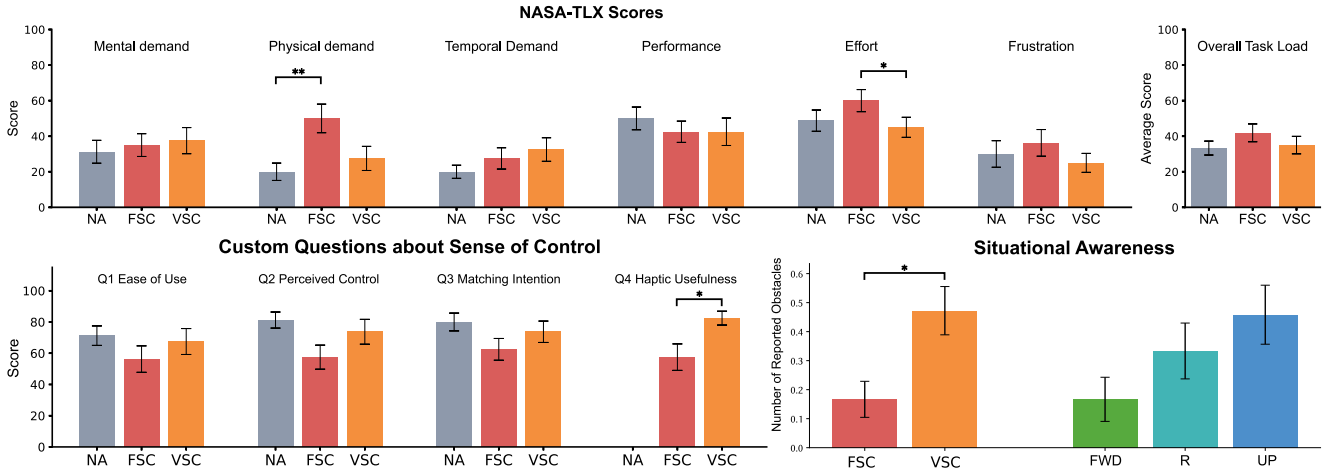


Fig. 8: The subjective measurements and situational awareness results grouped by feedback condition (NA - no feedback, FSC - force shared control, VSC - vibrotactile shared control) and flying direction (FWD - forward, R - right, UP - upward). The error bars represent the standard error of the mean (SEM), $^{*}p < 0.05$, $^{**}p < 0.01$, and $^{***}p < 0.001$.

difference from FSC. This was likely due to the short duration of the experimental tasks, which lasted approximately 1 minute per flight. This prevented participants from experiencing the types of workloads typically associated with longer real-world teleoperation. Future research should consider extending the task duration to gain a deeper understanding of the differences between the two.

B. Effect of Visual Capacity

The isolated flying directions helped assess how visual capacity influenced operator's reliance on haptic feedback. When obstacles were out of sight in right and upward directions, the numbers of collisions were three times more than the forward direction. This suggested that haptic feedback was more helpful in low visual capacity conditions. Due to the limited obstacles in the tunnels, no statistical difference was found across flying directions on situational awareness. Future research can increase obstacles to highlight the effectiveness of haptic feedback in low visibility scenarios.

Besides flying directions, factors such as light sources and dust [40] can also impair visual capacity in real-world scenarios. Future research could investigate these factors and refine algorithms to dynamically adjust haptic feedback.

C. Intuitive Mappings from Layout Optimization

Our layout optimization process eliminated perceptual biases and ensured an intuitive mapping between body positions and obstacle directions, which helped reduce perceived workloads during teleoperation with vibrotactile assistance. As no significant differences were found between the NA and VSC conditions, the addition of vibrotactile feedback on the body likely did not impose extra processing workload.

On the contrary, FSC caused significantly higher physical demand than NA and required more effort than VSC. This indicates that the removal of the RC controller and the direct coupling of input and output channels imposed control challenges that could be attributed to force feedback misalignment. Previous approaches that rendered force feedback often overlooked the anisotropy of force perception and magnitude on the hand [32], [33], leading to perceptual differences in force feedback from various directions. These findings thus reinforce the importance of addressing perceptual biases in future haptic device design for UAV teleoperation.

VIII. Limitations and Future Work

Although the results showed that on-body vibrotactile feedback was effective at enhancing collision avoidance during UAV teleoperation, there are a few limitations. First, as discussed in Section IV-C, upper-body vibrations are limited in representing obstacles below the drone. Currently, we had to map them to the lower back actuators. In the future, We plan to extend vibrotactile feedback to the upper limbs or lower body to alleviate this problem.

Second, UAV teleoperation was simulated in a simplified virtual environment, without considering real-world factors such as communication delays, inaccurate UAV state estimations, or control input constraints, as outlined by Lam et al. [34]. To understand the role of such factors, we plan to integrate AeroHaptix into commercial UAVs using developer tools such as DJI SDKs.

Third, we only modulated the actuator positions and vibration intensities to convey obstacle directions. Future work could utilize more vibration parameters to enrich the information provided about obstacles, such as obstacle types and mobility. As prior research has suggested, the use of frequency, spatial, and temporal patterns could also be viable when rendering such information [19].

IX. Conclusion

This work introduced AeroHaptix, a novel vibrotactile feedback system for collision avoidance during UAV teleoperation. The system was built using custom hardware that featured high-density actuators, fine-grained control, and low-latency communication. An optimal actuator layout was derived from a perceptual study to ensure obstacle directions could be uniformly conveyed across the upper body. Using

a novel multi-point feedback algorithm (i.e., MultiCBF), AeroHaptix was found to enhance collision avoidance and reduce input disagreement, without inducing extra workload. We encourage future researchers to build on our work to innovate haptic feedback devices to assist operators with UAV teleoperation.

REFERENCES

[1] T. M. Lam, H. W. Boschloo, M. Mulder, and M. M. Van Paassen, "Artificial force field for haptic feedback in uav teleoperation," *IEEE Trans. on Syst., Man, and Cybern.-Part A: Syst. and Humans*, vol. 39, no. 6, pp. 1316–1330, 2009.

[2] A. M. Brandt and M. B. Colton, "Haptic collision avoidance for a remotely operated quadrotor uav in indoor environments," in *2010 IEEE Int. Conf. on Syst., Man and Cybern.* IEEE, 2010, pp. 2724–2731.

[3] X. Hou and R. Mahony, "Dynamic kinesthetic boundary for haptic teleoperation of vtol aerial robots in complex environments," *IEEE Trans. on Syst., Man and Cybern.: Syst.*, vol. 46, no. 5, pp. 694–705, 2015.

[4] D. Zhang, G. Yang, and R. P. Khurshid, "Haptic teleoperation of uavs through control barrier functions," *IEEE Trans. on Haptics*, vol. 13, no. 1, pp. 109–115, 2020.

[5] N. T. Inc, "Novint falcon," https://hapticshouse.com/pages/novints-falcon-haptic-device, 2024, accessed: Feb 24, 2024.

[6] D. Syst., "Touch haptic device," https://www.3dsystems.com/haptics-devices/touch, 2024, accessed: Feb 24, 2024.

[7] F. Dimension, "Force dimension - haptic devices," https://www.forcedimension.com/products, 2024, accessed: Feb 24, 2024.

[8] D. Zhang, R. Tron, and R. P. Khurshid, "Haptic feedback improves human-robot agreement and user satisfaction in shared-autonomy teleoperation," in *2021 IEEE Int. Conf. on Robot. and Automat. (ICRA)*. IEEE, 2021, pp. 3306–3312.

[9] R. M. Philbrick and M. B. Colton, "Effects of haptic and 3d audio feedback on operator performance and workload for quadrotor uavs in indoor environments," *J. of Robotics and Mechatronics*, vol. 26, no. 5, pp. 580–591, 2014.

[10] A. M. Brandt, "Haptic collision avoidance for a remotely operated quadrotor uav in indoor environments," 2009.

[11] V. Ho, C. Borst, M. M. van Paassen, and M. Mulder, "Increasing acceptance of haptic feedback in uav teleoperation by visualizing force fields," in *2018 IEEE Int. Conf. on Syst., Man, and Cybern. (SMC)*. IEEE, 2018, pp. 3027–3032.

[12] C. Rognon, A. R. Wu, S. Mintchev, A. Ijspeert, and D. Floreano, "Haptic guidance with a soft exoskeleton reduces error in drone teleoperation," in *Haptics: Science, Technology, and Applications: 11th International Conference, EuroHaptics 2018, Pisa, Italy, June 13-16, 2018, Proceedings, Part II 11*. Springer, 2018, pp. 404–415.

[13] C. Rognon, V. Ramachandran, A. R. Wu, A. J. Ijspeert, and D. Floreano, "Haptic feedback perception and learning with cable-driven guidance in exosuit teleoperation of a simulated drone," *IEEE transactions on haptics*, vol. 12, no. 3, pp. 375–385, 2019.

[14] S.-G. Hong, B. S. Kim, S. Kim, and J.-J. Lee, "Artificial force reflection control for teleoperated mobile robots," *Mechatronics*, vol. 8, no. 6, pp. 707–717, 1998.

[15] H. W. Boschloo, T. M. Lam, M. Mulder, and M. Van Paassen, "Collision avoidance for a remotely-operated helicopter using haptic feedback," in *2004 IEEE Int. Conf. on Syst., Man and Cybern.*, vol. 1. IEEE, 2004, pp. 229–235.

[16] X. Hou and R. Mahony, "Dynamic kinesthetic boundary for haptic teleoperation of aerial robotic vehicles," in *2013 IEEE/RSJ Int. Conf. on Intell. Robots and Syst.* IEEE, 2013, pp. 4549–4950.

[17] H. Courtois, N. Aouf, K. Ahiska, and M. Cecotti, "Oast: Obstacle avoidance system for teleoperation of uavs," *IEEE Trans. on Human-Machine Syst.*, vol. 52, no. 2, pp. 157–168, 2022.

[18] C. Louison, F. Ferlay, and D. R. Mestre, "Spatialized vibrotactile feedback contributes to goal-directed movements in cluttered virtual environments," in *2017 IEEE Symp. on 3D User Interfaces (3DUI)*. IEEE, 2017, pp. 99–102.

[19] V. A. de Jesus Oliveira, L. Brayda, L. Nedel, and A. Maciel, "Designing a vibrotactile head-mounted display for spatial awareness in 3d spaces," *IEEE Trans. on Vis. and Comp. Graph.*, vol. 23, no. 4, pp. 1409–1417, 2017.

[20] J. Y. F. Lee, N. Rajeev, and A. Bhojan, "Goldeye: Enhanced spatial awareness for the visually impaired using mixed reality and vibrotactile feedback," in *ACM Multimedia Asia*, 2021, pp. 1–7.

[21] G. Flores, S. Kurniawan, R. Manduchi, E. Martinson, L. M. Morales, and E. A. Sisbot, "Vibrotactile guidance for wayfinding of blind walkers," *IEEE Trans. on Haptics*, vol. 8, no. 3, pp. 306–317, 2015.

[22] H.-C. Wang, R. K. Katzschmann, S. Teng, B. Araki, L. Giarré, and D. Rus, "Enabling independent navigation for visually impaired people through a wearable vision-based feedback system," in *2017 IEEE Intl. Conf. on Robotics and Automat. (ICRA)*. IEEE, 2017, pp. 6533–6540.

[23] M. A. B. Mohammed Zaffir and T. Wada, "Presentation of robot-intended handover position using vibrotactile interface during robot-to-human handover task," in *Proc. of the 2024 ACM/IEEE Int. Conf. on Human-Robot Interact.*, 2024, pp. 492–500.

[24] S. Grushko, A. Vysockỳ, D. Heczko, and Z. Bobovskỳ, "Intuitive spatial tactile feedback for better awareness about robot trajectory during human–robot collaboration," *Sensors*, vol. 21, no. 17, p. 5748, 2021.

[25] P. Xia, K. McSweeney, F. Wen, Z. Song, M. Krieg, S. Li, X. Yu, K. Crippen, J. Adams, and E. J. Du, "Virtual telepresence for the future of rov teleoperations: opportunities and challenges," in *SNAME Offshore Symp.* SNAME, 2022, p. D011S001R001.

[26] P. G. De Barros, R. W. Lindeman, and M. O. Ward, "Enhancing robot teleoperator situation awareness and performance using vibro-tactile and graphical feedback," in *2011 IEEE Symp. on 3D User Interfaces (3DUI)*. IEEE, 2011, pp. 47–54.

[27] D. Wittchen, K. Spiel, B. Fruchard, D. Degraen, O. Schneider, G. Freitag, and P. Strohmeier, "Tactjam: An end-to-end prototyping suite for collaborative design of on-body vibrotactile feedback," in *16th Int. Conf. on Tangible, Embedded, and Embodied Interact.*, 2022, pp. 1–13.

[28] A. Dementyev, P. Getreuer, D. Kanevsky, M. Slaney, and R. F. Lyon, "Vhp: vibrotactile haptics platform for on-body applications," in *The 34th Annu. ACM Symp. on User Interface Softw. and Technol.*, 2021, pp. 598–612.

[29] bHaptics, "bhaptics: Tactile feedback for vr, gaming, and music," 2023. [Online]. Available: https://www.bhaptics.com/

[30] J. B. Van Erp, "Presenting directions with a vibrotactile torso display," *Ergonomics*, vol. 48, no. 3, pp. 302–313, 2005.

[31] H. Elsayed, M. Weigel, F. Müller, M. Schmitz, K. Marky, S. Günther, J. Riemann, and M. Mühlhäuser, "Vibromap: Understanding the spacing of vibrotactile actuators across the body," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 4, pp. 1–16, 2020.

[32] F. E. Van Beek, W. M. B. Tiest, and A. M. Kappers, "Anisotropy in the haptic perception of force direction and magnitude," *IEEE Trans. on Haptics*, vol. 6, no. 4, pp. 399–407, 2013.

[33] H. Z. Tan, F. Barbagli, J. Salisbury, C. Ho, and C. Spence, "Force-direction discrimination is not influenced by reference force direction (short paper)," 2006.

[34] T. M. Lam, M. Mulder, and M. Van Paassen, "Collision avoidance in uav tele-operation with time delay," in *2007 IEEE Int. Conf. on Syst., Man and Cybern.* IEEE, 2007, pp. 997–1002.

[35] S. Omari, M.-D. Hua, G. Ducard, and T. Hamel, "Bilateral haptic teleoperation of vtol uavs," in *2013 IEEE Int. Conf. on Robot. and Automat.* IEEE, 2013, pp. 2393–2399.

[36] S. Reyes, H. Romero, S. Salazar, R. Lozano, and O. Santos, "Outdoor haptic teleoperation of a hexarotor uav," in *2015 Int. Conf. on Unmanned Aircr. Syst. (ICUAS)*. IEEE, 2015, pp. 972–979.

[37] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robot.*, 2017. [Online]. Available: https://arxiv.org/abs/1705.05065

[38] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in *Advances in psychology*. Elsevier, 1988, vol. 52, pp. 139–183.

[39] M. R. Endsley, "A systematic review and meta-analysis of direct objective measures of situation awareness: a comparison of sagat and spam," *Human factors*, vol. 63, no. 1, pp. 124–150, 2021.

[40] T.-C. Hung, Y.-R. Li, and C.-C. Peng, "Uav inspection in heavy industry: Preliminary trial and challenge," in *2024 International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan)*. IEEE, 2024, pp. 135–136.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 23, 2022, accepted March 20, 2022, date of publication March 30, 2022, date of current version April 7, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3163302

Weakly Supervised Semantic and Attentive Data Mixing Augmentation for Fine-Grained Visual Categorization

MENGQI HE[1], QILONG CHENG[2], AND GUANQIU QI[3]
[1]College of Engineering and Computer Science, The Australian National University, Canberra, ACT 2600, Australia
[2]Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, ON M5S 1A1, Canada
[3]Computer Information Systems Department, The State University of New York at Buffalo State, Buffalo, NY 14222, USA

Corresponding author: Guanqiu Qi (qig@buffalostate.edu)

ABSTRACT As a key factor, the availability of large-scale training samples determines the improvement of visual performance. However, the size of Fine-Grained Visual Categorization (FGVC) datasets is always limited. Therefore, overfitting as an issue in FGVC-related training needs to be solved. Data mixing augmentation is a widely-used data augmentation method. In most of the recently proposed data mixing augmentation methods, random patch selection may generate meaningless training samples and result in model instability during the training process. This paper proposes a data mixing augmentation strategy termed Semantic and Attentive Data Mixing (SADMix) to select semantic patches for the generation of new training samples. In SADMix, a certain number of critical regions are localized according to convolutional activations. An image patch is selected from these localized regions for the generation of new training samples. The size, aspect ratio, and center location of these image patches are changed according to the random values from a beta distribution. These image patches with semantic information are used to mix two training images. According to the class activation map (CAM), training images and their labels are mixed proportionally to generate new mixed training samples and the corresponding labels. The proposed SADMix is tested on three fine-grained datasets, which are CUB-200-2011, FGVC Aircraft, and Stanford Cars, respectively. The experimental results confirm the effectiveness of the proposed SADMix.

INDEX TERMS Data augmentation, fine-grained visual categorization, attention.

I. INTRODUCTION

In order to prevent overfitting caused by the limited training data, random data augmentation methods used in machine learning are applied to increase the size and diversity of training data. Additionally, random data augmentation methods can also improve the generalization ability of models. Therefore, data augmentation methods, such as random image cropping, flipping and rotation, are widely used in the training phase of visual tasks such as object recognition and object detection, etc. Recently proposed data augmentation methods are categorized into two groups. As the first group, region-erasing methods [1], [2] erase partial image regions to encourage models to identify more discriminative regions. As the second group, data mixing methods [2]–[7] generate new training data by combining multiple images and fusing their labels accordingly. Data mixing augmentation methods have been proven to be effective in both general and fine-grained image classification [6]–[8]. Therefore, they are receiving more and more attention. In Mixup [5], two images are combined linearly and their labels are mixed by using the same combination coefficients. In CutMix [7], an image region is first cut out and then pasted on another image. Labels of these two images are mixed according to the corresponding area proportion.

Data mixing augmentation methods have two major drawbacks. As the first drawback, the diversity of augmented data is limited due to the symmetrically blended image regions. As a result, the selected regions are restricted to be complementary. As the second drawback, label noise may be generated by the random selection of image regions for mixing. In Fig. 1, the randomly generated region may cover the meaningless region in image (a), and this region is copied and pasted on the critical regions in image (b) to generate

The associate editor coordinating the review of this manuscript and approving it for publication was Tai-Hoon Kim.

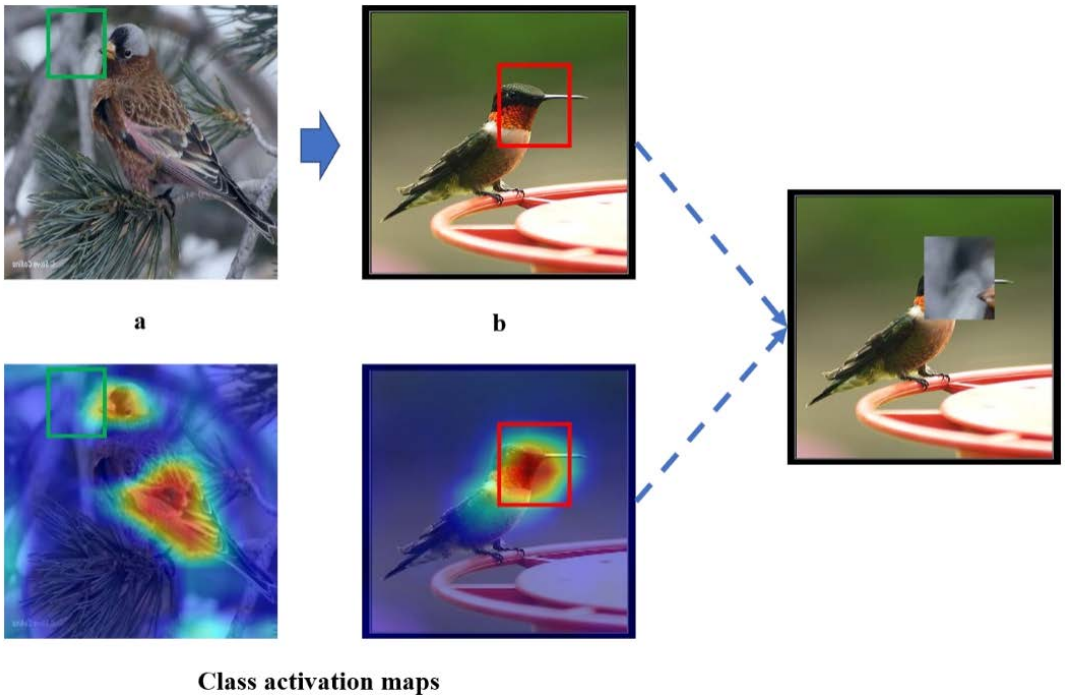


FIGURE 1. Meaningless training samples generated in SnapMix.

a meaningless training sample. Moreover, the meaningless region in image (a) may still have some semantic information in the class activation map (CAM). Due to the subtle differences in some small image regions, these two drawbacks limit the recognition performance, especially in the field of Fine-Grained Visual Categorization (FGVC).

Due to both high intra-class variances and low inter-class variances, FGVC is a challenging visual task. Additionally, the size of fine-grained datasets is usually small because the obtainment of partial manual annotations requires expert-level domain knowledge. In essence, FGVC mainly focuses on learning fine-grained features from the limited data. As discussed in [9], FGVC feature learning methods can be divided into three paradigms, fine-grained object recognition (1) with localization-classification subnetwork, (2) with end-to-end feature encoding, and (3) with external information. Due to the limited dataset size, data augmentation is also important for FGVC.

Some existing study focuses on data augmentation for FGVC [6], [9], [10]. A data mixing augmentation method called Semantically Proportional Mixing (SnapMix) was proposed [6]. Mixed labels are generated by exploiting Class Activation Map (CAM) to reduce negative influence [10]. The generated labels, that are normalized to sum to 1, are used to weigh the mixed images in SnapMix. Different from CutMix, cut-and-paste operations are set to be asymmetric in SnapMix, which can boost data diversity. Due to the importance of attribute-level features in discriminating subcategories, the discriminative and transferable attribute features are explored and used to scale up the fine-grained training samples in Attribute Mix [8]. A weakly supervised data augmentation network (WS-DAN) was proposed to generate attention maps for the representation of discriminative regions by weakly-supervised learning [10]. The generated attention maps were then used to guide image augmentation, such as attention cropping and attention dropping. The fine-grained features were extracted through an effective module called bilinear attention pooling (BAP).

This paper proposes a data mixing augmentation method called Semantic and Attention Data Mixing (SADMix) for FGVC. In SADMix, the regions selected for mixing are localized by using CNN's activation maps. According to the observation of CNN's activation maps, the regions with high activation values often correspond to the localized key parts. When the activation value is getting larger, more information is involved in the representation of the corresponding region. Since the regions with relatively large activation values are often adjacent to the region with the largest activation value, non-maximum suppression (NMS) is first adopted to select a fixed number of box regions with different scales. Then a box region is randomly selected from these box regions for mixing. In order to improve the corresponding performance, random adjustment is applied to change the size and aspect ratio of the selected box region according to a random value generated by Beta distribution. According to the conclusion in [11], the regions with relatively large activation values are often adjacent to the region with the largest activation value. Therefore, non-maximum suppression (NMS) is first adopted to select a fixed number of box regions with different scales and less redundancy.

The main contributions of this paper are summarized as follows: (1) A data mixing augmentation method called SADMix is proposed for fine-grained visual categorization. SADMix can mix images asymmetrically and generate the corresponding label of the mixed image according to the normalized CAM. A box region used for mixing is selected by a Semantic and Attentive Part Localization (SAPL) module. The box region for image mixing is always located in critical object parts. (2) Due to the fixed size of box regions selected from a SAPL module, it is necessary to randomly change the size and aspect ratio of the selected semantic and attentive box region for integrating more useful information. Experimental results confirm that even a simple network with the proposed SADMix can achieve competitive performance compared with the state-of-the-art methods.

II. RELATED WORK

This section briefly reviews FGVC methods and several recently proposed data augmentation methods, which are closely related to the proposed SADMix.

A. FGVC METHODS

This paper only focuses on the weakly supervised FGVC methods with localization-classification subnetwork [11]–[16]. Weakly supervised FGVC methods can localize the critical regions and learn fine-grained features only depending on the image-level labels. Navigator-Teacher-Scrutinizer Network (NTS-Net) was proposed and its localization subnetwork was trained to localize informative regions without any manual annotations [11]. The features extracted from the whole image and informative regions were concatenated for recognition. In multi-branch and multi-scale learning network (MMAL-Net) [11], three branches are used to learn features from the whole input image, an object and parts. Localization branches in MMAL-Net only involve a small number of parameters that need to be trained. A discriminative filter learning network (DFL-Net) was proposed [17] to learn discriminative mid-level patches in an end-to-end fashion. Both discriminative local and global features were learnt by DFL-Net. Destruction and construction learning (DCL) [18] encourages the CNN to learn fine-grained features by the destruction of the global image structure. WS-DAN [19] first generates attention maps to represent the discriminative parts by weakly-supervised learning. The generated attention maps are then used to guide image augmentation, such as attention cropping and attention dropping. The fine-grained features are extracted by an effective module called bilinear attention pooling (BAP).

B. DATA MIXING AUGMENTATION

In this subsection, the recently published data mixing augmentation methods are introduced. Mixup [5] was proposed to mix data to extend the training distribution. It generated images by linearly combining training images and fusing their labels with the same coefficients. CutMix [7] can generate a mixed image by cutting out one region and pasting it on another image. The labels are also mixed proportionally according to the area of the corresponding regions. To avoid the label noise generated by Mixup and CutMix and increase the diversity of training data, SnapMix [6] mixes images asymmetrically and generates the new labels for the mixed images by estimating semantic compositions according to the normalized CAM. The details of these related data augmentation methods will be discussed in subsection 3.1.

III. THE PROPOSED METHOD

This section discusses the proposed SADMix in detail. The subsection 3.1 reviews Mixup, CutMix and SnapMix. The subsection 3.2 specifies the details of the proposed SADMix.

A. BACKGROUND

The original training dataset is defined as $\{(\mathbf{I}_i, y_i), i = 1, 2, \ldots, N\}$, where $\mathbf{I}_i \in \mathbf{R}^{3\times H\times W}$ and y_i refer to an image and its label, respectively. For a data pair $((\mathbf{I}_a, y_a), (\mathbf{I}_b, y_b))$ and a random variable λ from Beta distribution $Beta(\alpha, \alpha)$, a mixed image $\hat{\mathbf{I}}$ and two label weights β_a and β_b are generated. The label weights β_a and β_b are used to generate the mixed label $\hat{y}$.

In Mixup [5], two images and their labels are combined linearly. The linear combination is expressed as follows.

$$\hat{\mathbf{I}} = \lambda \mathbf{I}_a + (1-\lambda)\mathbf{I}_b \tag{1}$$

$$\hat{y} = \beta_a y_a + \beta_b y_b \tag{2}$$

$$\beta_a = \lambda \tag{3}$$

$$\beta_b = 1 - \lambda \tag{4}$$

In Mixup, the whole image is applied to the linear combination. So, the robustness of CNN to adversarial examples is improved.

Different from Mixup, CutMix adopts cut-and-paste operations for mixing two images. The mixed labels are combined according to the area ratio. The combination in CutMix can be expressed as follows.

$$\hat{\mathbf{I}} = M_\lambda \bigodot \mathbf{I}_a + (1 - M_\lambda) \bigodot \mathbf{I}_b \tag{5}$$

$$\hat{y} = \lambda y_a + (1-\lambda) y_b \tag{6}$$

where $\bigodot$ denotes the element-wise multiplication and $M_\lambda \in \mathbf{R}^{H\times W}$ is the binary mask of a randomly selected box region whose area ratio to the image is λ. In addition to improving the robustness of CNN, CutMix can enhance the localization capacity of CNN.

Area ratio used in CutMix cannot effectively reflect the intrinsic semantic composition of the mixed image. A large area may contain less semantic information, but it takes a large proportion in the label mixing process. This results in noise mixing labels. In order to solve this problem, class activation maps (CAM) [10] is applied to estimate the label composition of mixed images in SnapMix. Different from mixing images at symmetric locations in CutMix, an area is first cropped at a randomly selected location and then transformed and pasted to a randomly selected location on

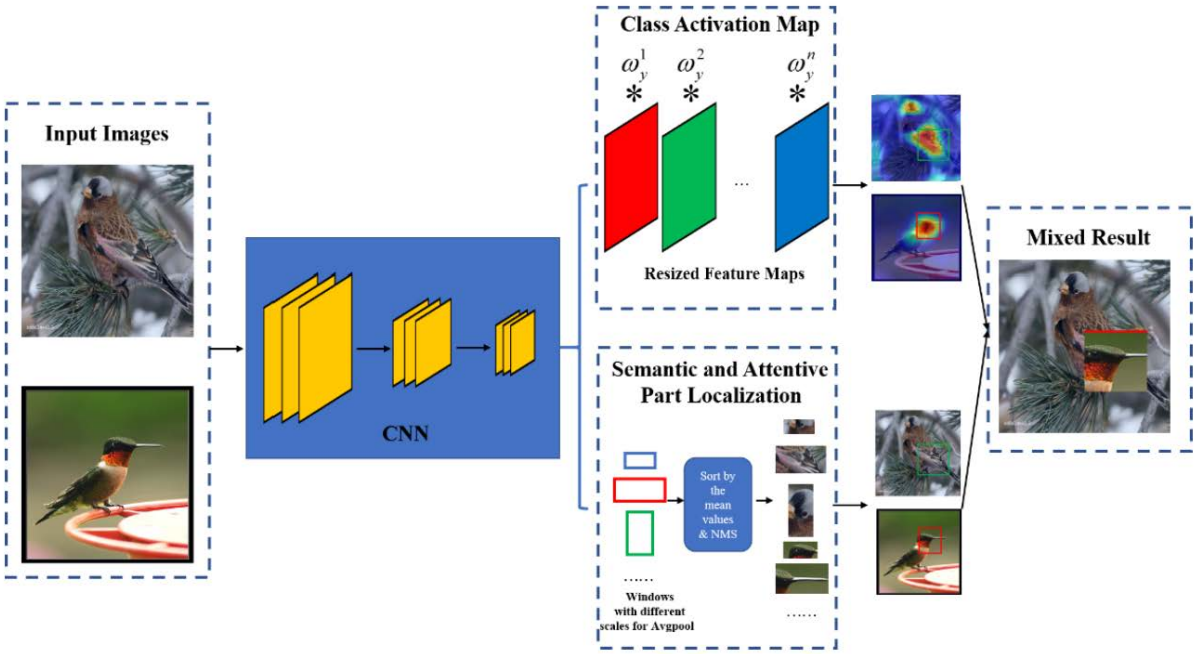


FIGURE 2. An overview of SADMix.

another image in SnapMix. The mixing of two images can be expressed as follows.

$$\hat{\mathbf{I}} = T_\theta(M_{\lambda_a} \bigodot \mathbf{I}_a) + (1 - M_{\lambda_b}) \bigodot \mathbf{I}_b \tag{7}$$

where M_{λ_a} and M_{λ_b} are two binary masks containing the randomly selected box regions with area ratios λ_a and λ_b, and T_θ is a transformation that makes the cutout region size of $\mathbf{I}_a$ to match the corresponding box region size of $\mathbf{I}_b$. In the process of label generation, the CAM of the input image is calculated as follows.

$$Cam(\mathbf{I}_i) = Upsample(\sum_{j=1}^{C} \omega_{y_i}^j F_j(\mathbf{I_i})) \tag{8}$$

where $\mathbf{I}_i$ is the input image, $F(\mathbf{I_i}) \in \mathbf{R}^{c\times h\times w}$ is the output of the last convolutional layer, $F_j(\mathbf{I_i}) \in \mathbf{R}^{h\times w}$ is the j-th feature map of $F(\mathbf{I_i})$, and $\omega_{y_i}^j \in \mathbf{R}^c$ is the weight corresponding to class y_i in the FC layer. The CAM of the input image should be normalized to sum-to-1. The normalized CAM of the input image $\mathbf{I}_i$, which is defined as $N(\mathbf{I}_i)$, is calculated as follows.

$$N(\mathbf{I}_i) = \frac{Cam(\mathbf{I}_i)}{sum(Cam(\mathbf{I}_i)))} \tag{9}$$

The mixed label in SnapMix is generated as follows.

$$\hat{y} = \beta_a y_a + \beta_b y_b \tag{10}$$

$$\beta_a = sum(M_{\lambda_a} \bigodot N(\mathbf{I}_a)) \tag{11}$$

$$\beta_b = 1 - sum(M_{\lambda_b} \bigodot N(\mathbf{I}_b)) \tag{12}$$

In SnapMix, two images are mixed asymmetrically to generate a mixed image and the target labels of the mixed image are generated by estimating its intrinsic compositions through the normalized CAM. Experimental results confirm SnapMix can consistently outperform existing data mixing augmentation methods such as Cutout, Mixup and CutMix.

B. SADMix

Different from SnapMix, the random box region generation is replaced by Semantic and Attentive Part Localization (SAPL) module in SADMix.

According to the observation of the feature map $\mathbf{F} \in \mathbf{R}^{c\times h\times w}$ of the last convolutional layer in CNN, where c is the channel of a feature map, h and w ere the height and width of the feature map, windows with different sizes and aspect ratios are used to perform average pooling operation on the feature map, which is similar to [20]. The activations mean value of each window's feature map $\mathbf{F}_\omega$ is calculate as follows.

$$\hat{f}_\omega = \frac{\sum_{x=0}^{W_\omega - 1} \sum_{y=0}^{H_\omega - 1} \mathbf{F}_\omega(x, y)}{H_\omega \times W_\omega} \tag{13}$$

where H_ω and W_ω are the height and width of a window's feature map. The activations mean maps are summed in channel dimension. When the $\hat{f}_\omega$ is getting larger, the region corresponding to this mean value becomes more informative. The regions with relatively high informativeness are selected for data mixing. However, they are usually adjacent to the largest $\hat{f}_\omega$ windows and almost contain the same part of the object. To reduce region redundancy, non-maximum suppression (NMS) is used to select a number of regions with different scales.

The overall process of SADMix is shown in Fig. 2 Two original training images $((\mathbf{I}_a, y_a), (\mathbf{I}_b, y_b))$ are processed through the backbone network. The feature maps of the last convolutional layer in the backbone network are applied to both CAM and SAPL. The normalized CAMs $N(\mathbf{I}_a)$ and $N(\mathbf{I}_b)$ of training images are calculated through the methods reviewed in subsection 3.1. The image mixing is expressed as follows.

$$\hat{\mathbf{I}} = T_\theta(M_{\lambda_a}^{SAPL} \bigodot \mathbf{I}_a) + (1 - M_{\lambda_b}^{SAPL}) \bigodot \mathbf{I}_b \qquad (14)$$

where $M_{\lambda_a}^{SAPL}$ and $M_{\lambda_b}^{SAPL}$ are two binary masks containing semantic and attentive box regions with the area ratios λ_a and λ_b. In order to improve the performance, the aspect ratio and size of the semantic and attentive box regions are determined by a random value between 0 and 1 that also follows a Beta distribution.

IV. EXPERIMENTS AND RESULTS

In this section, the performance of the proposed SADMix is evaluated on three fine-grained datasets. Two network architectures (ResNet34 and ResNet50) are used as backbones. The implementation details of baselines and SADMix are introduced in subsection 4.3. The performance comparison between the proposed SADMix and the related data augmentation methods is shown in subsection 4.4. In order to further show the effectiveness of SADMix, the class activation maps (CAMs) of all the methods are compared and analyzed in subsection 4.5.

A. DATASETS

In comparative experiments, three fine-grained datasets are used for evaluation.

- CUB-200-2011 [21] is the most widely used fine-grained object dataset, which contains 11,788 images spanning 200 bird species. There are 5,994 training images and 5,794 testing images, respectively.
- Stanford-Cars [22] contains 16,185 images involving 196 manufacturer classes. There are 8,144 training images and 8,041 testing images, respectively.
- FGVC-Aircraft [23] consists of 10,000 images, which can be divided into 100, 30 and 70 categories by Variants, Manufacturers and Family, respectively. There are 6,667 training images and 3,333 testing images, respectively. The comparative experiments only use the images from 100 categories of variants.

B. BACKBONE NETWORKS AND BASELINES

Two backbone networks of ResNet34 and ResNet50 pretrained on ImageNet dataset [24] are applied to the proposed SADMix and other four representative data augmentation methods including CutOut [2], Mixup [5], CutMix [7] and SnapMix [6]. These two networks pretrained on ImageNet are also used as baselines in the experiments. Similar to [6], a strong baseline termed as mid-level baseline that incorporates mid-level features is used. There are two branches in the mid-level baseline. One is the original classification branch in baseline. The other is mid-level classification branch, which is composed of $Conv_{1\times1}$, MaxPooling and FC layer. The mid-level classification branch is placed after the last convolutional layer in the network. In the inference phase, the sum of the outputs of original classification branch and mid-level classification branch is used for prediction.

1) DATA AUGMENTATION METHODS

According to the experiment setup in [6], the probability of performing data augmentation is set to 0.5 for CutOut and Mixup and 1.0 for CutMix, SnapMix and SADMix. The α in Beta distribution $Beta(\alpha, \alpha)$ is set to 1.0 for MixUp and 3.0 for the remaining methods.

2) FGVC METHODS

In order to further evaluate the performance of SADMix, five state-of-the-art FGVC methods including NTS-Net [12], MMAL-Net [11], DFL-CNN [17], DCL [18] and WS-DAN [19] are used for comparison in the experiments.

C. IMPLEMENTATION DETAILS OF SADMIX

In this subsection, the implementation of SADMix is discussed in detail. In the experiments, the input images are first resized to 512 × 512 and then randomly cropped to 448 × 448. Similar to the experiments setup in [11], windows with three broad scale categories are constructed: [4 × 4, 3 × 5, 5 × 3], [6 × 6, 5 × 7, 7 × 5], [8 × 8, 6 × 10, 10 × 6, 7 × 9, 9 × 7, 7 × 10, 10 × 7] for the feature maps of 14 × 14 output by the last convolutional layer in ResNet. After using the NMS, different number of windows need to be selected on each broad scale category. The selection of window sizes is consistent with that of [1], which presents superior results in our experiments, indicating that this selection can obtain the regions with good discriminability. Avg. pooing operation is required for each window size, and multiple feature maps obtained will be sorted from large to small values. Also, NMS is used to select the region with the largest mean value. In the experiments, $N_1 = 2$, $N_2 = 3$ and $N_3 = 2$ are set. One window is randomly selected from these windows as the final semantic and attentive box region. In order to improve the performance, the selected semantic and attentive box regions are then randomly adjusted in size and aspect ratio according to a randomly generated value.

The experiments are implemented by PyTorch and trained on a PC with four TITAN-X GPUs. Similar to [6], stochastic gradient descent (SGD) with momentum 0.9 is used, and learning rate is set to 0.001 for the pre-trained parameters and 0.01 for new parameters respectively. All models are trained for 200 epochs and the learning rate is decayed by factor 0.1 every 80 epochs.

D. PERFORMANCE EVALUATION

In this subsection, the results of SADMix and performance comparisons with comparative approaches are presented. In order to show the effectiveness of random adjustment

TABLE 1. Performance comparison of SADMix with/without random adjustment.

	CUB-200-2011		Stanford-Cars		FGVC-Aircraft	
	ResNet34	ResNet50	ResNet34	ResNet50	ResNet34	ResNet50
SADMix	86.96%	88.01%	93.81%	94.25%	92.55%	92.96%
SADMix + rand adjustment	87.37%	88.23%	94.08%	94.43%	92.92%	93.13%

TABLE 2. Performance comparison of data augmentation methods on fine-grained object datasets.

	CUB-200-2011		Stanford-Cars		FGVC-Aircraft	
	ResNet34	ResNet50	ResNet34	ResNet50	ResNet34	ResNet50
Baseline	84.98%	85.49%	92.02%	93.04%	89.92%	91.07%
CutOut	83.36%	83.55%	92.84%	93.76%	89.90%	91.23%
MixUp	85.22%	86.23%	93.28%	93.96%	91.02%	92.24%
CutMix	85.69%	86.15%	93.61%	94.18%	91.26%	92.23%
SnapMix	87.06%	87.75%	93.95%	94.30%	92.36%	92.80%
SADMix	87.37%	88.23%	94.08%	94.43%	92.92%	93.13%

TABLE 3. Performance comparison with the state-of-the-art FGVC methods on fine-grained object datasets.

Method	**Backbone**	**CUB-200-2011**	**Stanford-Cars**	**FGVC-Aircraft**
NTS-Net	ResNet50	87.50%	93.90%	91.40%
DFL-CNN	VGG-16	86.70%	93.80%	92.00%
DCL	ResNet50	87.80%	94.50%	93.00%
MMAL-Net	ResNet50	89.60%	95.00%	94.70%
WS-DAN	Inception v3	89.40%	94.50%	93.00%
Baseline	ResNet50	85.49% (85.85%)	93.04% (93.17%)	91.07% (91.30%)
Mid-level baseline	ResNet50	87.13%	93.80%	91.68%
Baseline + SnapMix	ResNet50	87.75% (88.01%)	94.30% (94.59%)	92.80% (93.16%)
Mid-level baseline + SnapMix	ResNet50	88.70% (88.97%)	95.00% (95.16%)	93.24% (93.49%)
Baseline + SADMix	ResNet50	88.23% (88.75%)	94.43% (94.55%)	93.13% (93.43%)
Mid-level baseline + SADMix	ResNet50	89.01% (89.23%)	95.19% (95.33%)	93.53% (93.71%)

on the selected semantic and attentive box region, the performance comparison of SADMix with/without random adjustment is shown in Table 1. Further, the performance comparison of data augmentation methods is shown in Table 2. Then, SADMix is tested using two baselines and compared with the state-of-the-art FGVC methods, and the corresponding results are shown in Table 3. Top-1 accuracy is used as a performance evaluation indicator. Both the best accuracy and average accuracy of SADMix are provided.

Each data augmentation method is reimplemented in the experiments. When the experimental results are presented, the original performance is shown if the reimplemented performance is lower than the original performance.

Table 1 shows the results of SADMix with/without random adjustment on three fine-grained object datasets. SADMix with random adjustment outperforms the one without random adjustment. As a main reason, the random adjustment on the selected semantic and attentive box region can introduce more useful information in the training phase. Table 1 shows the average accuracy of the last 10 epochs. Since the validity has been verified, all subsequent experimental results are obtained by SADMix with random adjustment.

Table 2 shows the performance comparison of data augmentation methods on three fine-grained object datasets. The proposed SADMix consistently outperforms the comparative data augmentation methods. The experimental results show that deeper CNNs have better performance in all data augmentation methods. The results listed in Table 2 are obtained by using the baselines without mid-level features. Table 2 shows the average accuracy of the last 10 epochs.

Table 3 shows the performance comparison with the state-of-the-art FGVC methods. In Table 3, the results of FGVC methods are directly cited from the original papers. The results of SnapMix and SADMix are calculated by standard baselines and mid-level baselines, respectively. Moreover, the average accuracy of the last 10 epochs is reported and the best accuracy is shown in the brackets. As shown in Table 3, MMAL-Net performs better on CUB-200-2011 dataset than all other comparative methods. WS-DAN achieves sub-optimal performance on CUB-200-2011. SADMix with mid-level baseline achieves 89.23% accuracy, which is lower than MMAL-Net and WS-DAN. As a possible reason, CUB-200-2011 is more complex, which requires the model to have more powerful feature representation ability. MMAL-Net has three branches to describe global and local features simultaneously, while WS-DAN has a BAP module for feature representation. In FGVC-Aircraft, MMAL-Net also achieves the best performance. The proposed SADMix with mid-level baselines achieves the second-best performance. In Stanford-Cars, SADMix with mid-level baselines outperforms than all comparative methods.

The performance improvements of SADMix over the data augmentation methods are relatively low on Stanford-Cars

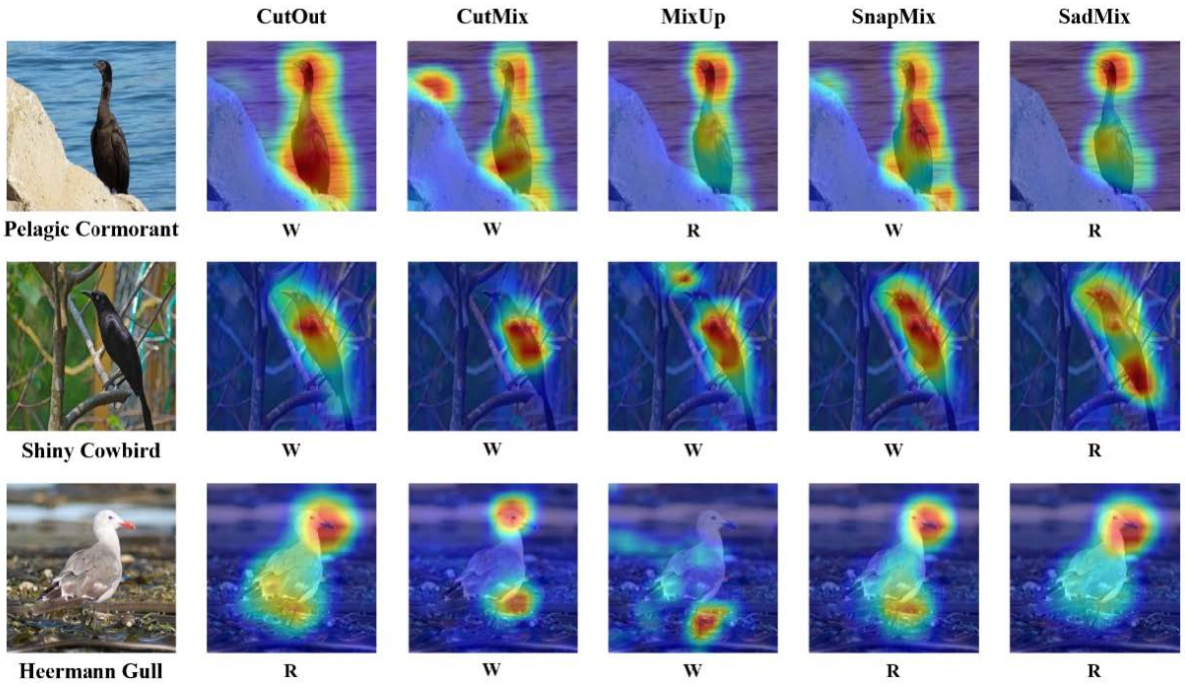


FIGURE 3. Class activation maps visualization of different data augmentation methods on CUB-200-2011.

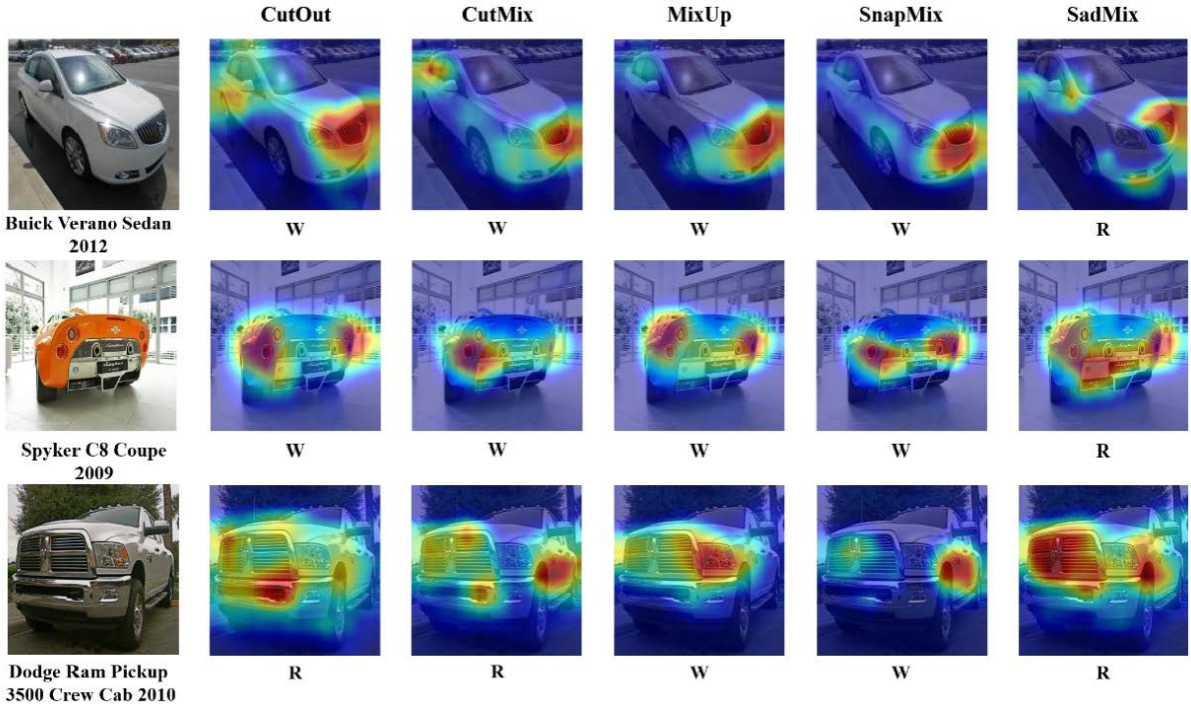


FIGURE 4. Class activation maps visualization of different data augmentation methods on Stanford-Cars.

and FGVC-Aircraft than on CUB-200-2011. The main reason is that the structure of vehicles and aircraft is relatively fixed. Compared with birds, the structure variation of vehicles and aircrafts is limited.

E. VISUALIZATION ANALYSIS

Class activation map visualization of different data augmentation methods is shown in Figures 3, 4, and 5. Three images are selected from each dataset used in the experiments.

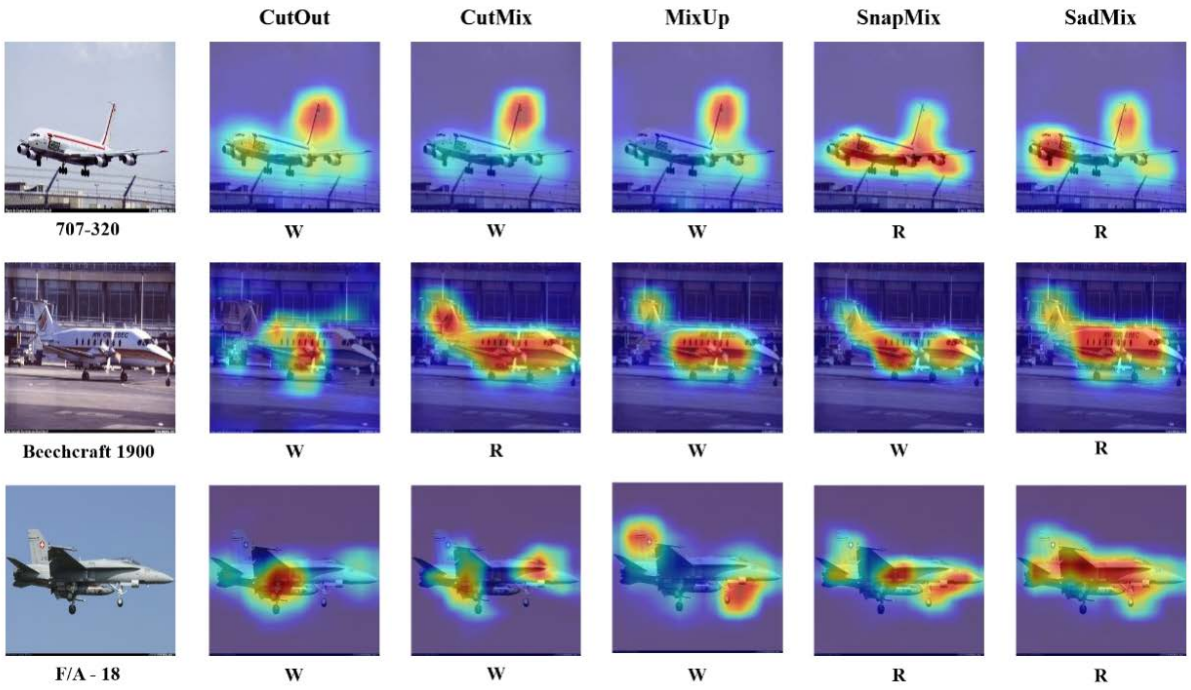


FIGURE 5. Class activation maps visualization of different data augmentation methods on FGVC-Aircraft.

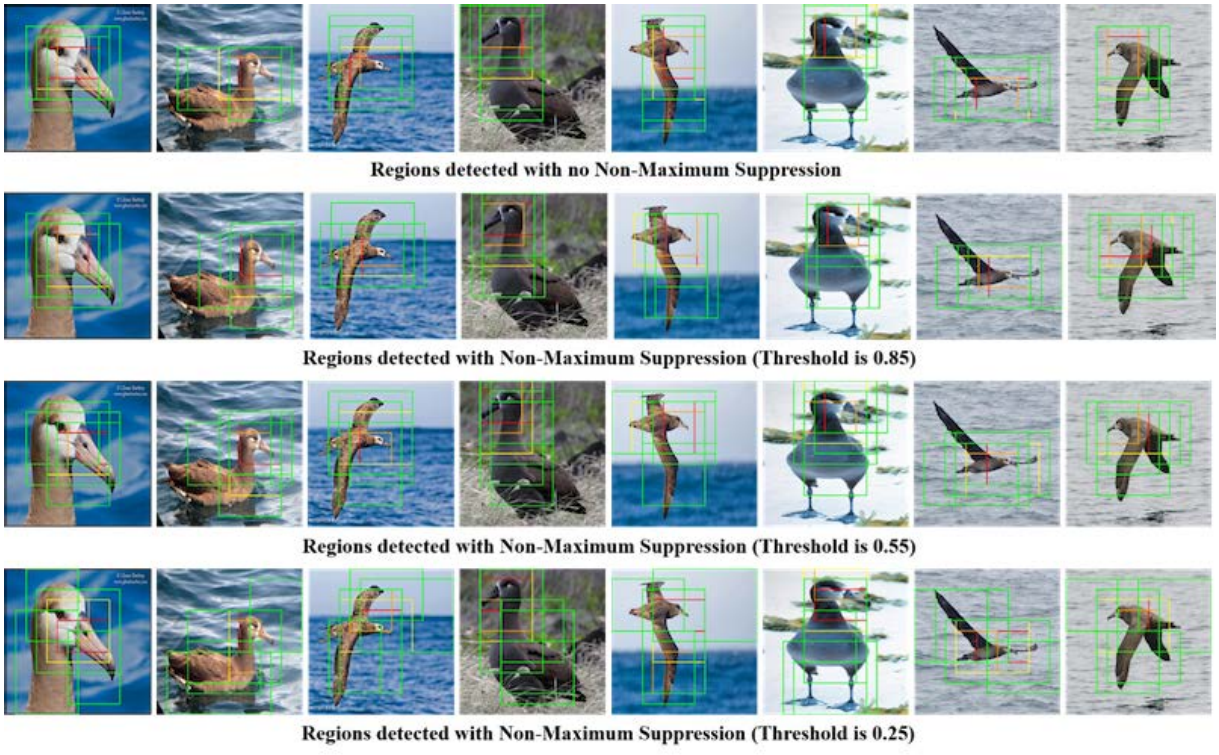


FIGURE 6. The effect of NMS with different threshold values.

Each image is correctly predicted by SADMix and maybe misclassified by several other methods. If an image is correctly classified, the image is marked in a capital R. If an image is misclassified, the image is marked in a capital W.

In order to reflect the ability of the network models trained by different data augmentation methods in the localization of critical regions, class activation map (CAM) [10] is used to make visualization analysis of the regions concerned by

TABLE 4. Performance comparison of SADMix with different NMS thresholds.

	CUB-200-2011		Stanford-Cars		FGVC-Aircraft	
Thresholds	ResNet34	ResNet50	ResNet34	ResNet50	ResNet34	ResNet50
0.25	87.16%	87.98%	93.78%	94.02%	92.85%	92.88%
0.55	87.37%	88.23%	94.08%	94.43%	92.92%	93.13%
0.85	87.03%	87.85%	93.57%	93.84%	92.60%	92.71%
No NMS	86.88%	87.52%	93.43%	93.65%	92.07%	92.35%

the networks. In CAM, the output feature maps of the final convolutional layer are fed into global average pooling. The global average pooling results are used as the features for classification. The importance of the image regions can be identified by projecting the weights of the fully connected layer on the final convolutional feature maps. In visualization analysis, the weights corresponding to the predicted class with the highest values in the fully connected layer are projected on the convolutional feature maps. Resnet50 is used in visualization analysis.

As shown in Figures 3, 4, and 5, the proposed SADMix, which outperforms the related data augmentation methods on three fine-grained datasets, not only localizes the critical regions more accurately, but also explores more critical region information. For example, the network trained by SADMix can localize both heads and tails of Shiny Cowbird for recognition. Networks trained by the remaining related data augmentation methods only localize parts of critical regions and cause misclassification.

F. EFFECT OF NMS

Figure 6 shows the effect of the NMS method on region detection at different thresholds. Each region is selected based on the activation mean value in the feature map. The boxes marked with different colors have different window sizes. It is observed that when NMS is not used, the detected regions are clustered together with a large area of overlapping. As the threshold value in the NMS decreases from 0.85 to 0.25, the selected regions gradually disperse and tend to cover different semantic parts of the object. This observation allows us to understand the purpose of NMS, which effectively reduces the redundancy and enhances the diversity of the highlighted semantic regions, leading to a performance gain.

To further examine the effect of NMS on the model performance, we evaluated SADMix with three NMS threshold values, including 0.25, 0.55, 0.85. Also, a baseline model without the usage of NMS was evaluated (i.e., a threshold of 1.0). Results on the three aforementioned datasets are reported in Table 4. it is observed that with an NMS threshold of 0.55, the resulting model consistently outperforms models with other NMS thresholds. This result verifies our hypothesis that as the threshold increases, a more diversified set of semantic regions can be selected and used for training sample augmentation, leading to performance gains. However, as the threshold keeps decreasing, the selected regions start to disperse and more background contents are included, which lowers the quality of the patches used for augmentation. It is expected that the model without NMS has the lowest accuracy due to the excessive redundancy among the selected regions.

V. CONCLUSION

This paper proposes a data mixing augmentation method termed as SADMix for fine-grained visual categorization. New training samples are generated by SADMix with the normalized CAM and SAPL modules. Different from existing solutions, random box region generation is replaced by a SAPL module. Adding a certain amount of random adjustment to windows with the fixed size and aspect ratio during localization is conducive to improving performance. As shown in subsection 4.4 and 4.5, the networks trained by SADMix can achieve better classification performance and stronger localization capacity.

It is worth pointing out that the computational complexity of SADMix is higher than related data augmentation methods, such as MixUp and SnapMix, due to the added process of localizing discriminative regions. Although the complexity has increased, the proposed SADMix has achieved consistent performance gains on three datasets. In real-world deep learning applications, there is always a trade-off between performance and speed. The latter can be improved by upgraded hardware, while the former can only be improved with a better model design or an enhanced dataset. Compared to its peers, SADMix aims to maximize a model's ability to mine discriminative patterns via data augmentation, outperforming the SOTA. Therefore, the superiority of SADMix can be justified and validated through the conducted experiments. Meanwhile, our investigation shows that SADMix and its peers lack comprehensive algorithm complexity analysis. This interesting future direction can offer theoretical support for certain building blocks utilized in this study.

The conclusions of this paper can be summarized as follows. 1) Random region generation in previous data augmentation methods, such as Cutout, Cutmix, Mixup and Snapmix, may introduce meaningless training samples. 2) The region generated by the attention mechanism module for mixing data can avoid generating meaningless training samples, and the generalization of the network and useful information can be improved by the random modification of the regions generated by the attention mechanism module.

Two research directions are worth further investigation. First, is it better to use contours, as shown in the activated areas in the Figure 5, rather than bounding boxes to perform SADMix? Intuitively, the contours contain more descriptive fine-grained information with less background noise. However, we anticipate two challenges: 1) different contours vary

in shape and size, leading to a difficulty in mixing them; 2) extracting contours is less straightforward compared to the extraction of bounding boxes. With proper strategies to resolve these challenges, contours can be more promising. Second, the usage of a generative model such as a generative adversarial network (GAN) in the FGVC task is also interesting. One benefit of GAN is its ability to generate synthetic samples that are not seen by the learning algorithm, leading to a more diverse training set. The challenge of GAN is that for FGVC, the learning algorithm should exploit the fine-grained features, which should be emphasized during training. Regular GANs, however, are more focused on the global features rather than the fine-grained ones. The proposed SADMix method allows the learning algorithm to focus on the fine-grained patterns, which may be highlighted by the semantic regions used for data mixing. Both methods have their pros and cons. It would be interesting to design a custom GAN in a way that both global and fine-grained features are both well captured to generate high quality training samples specific for the FGVC task. A more in-depth study is needed to apply GAN for this purpose.

ACKNOWLEDGMENT

The authors thank the anonymous reviewers for their careful reading of their manuscript and their insightful comments, which are essential to the improvement of the manuscript quality.

REFERENCES

[1] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 7, pp. 13001–13008.

[2] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," 2017, *arXiv:1708.04552*.

[3] Y. Tokozume, Y. Ushiku, and T. Harada, "Between-class learning for image classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5486–5494.

[4] H. Inoue, "Data augmentation by pairing samples for images classification," 2018, *arXiv:1801.02929*.

[5] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," 2017, *arXiv:1710.09412*.

[6] S. Huang, X. Wang, and D. Tao, "SnapMix: Semantically proportional mixing for augmenting fine-grained data," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 2, pp. 1628–1636.

[7] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "CutMix: Regularization strategy to train strong classifiers with localizable features," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6023–6032.

[8] H. Li, X. Zhang, Q. Tian, and H. Xiong, "Attribute mix: Semantic data augmentation for fine grained recognition," in *Proc. IEEE Int. Conf. Vis. Commun. Image Process. (VCIP)*, Dec. 2020, pp. 243–246.

[9] X.-S. Wei, J. Wu, and Q. Cui, "Deep learning for fine-grained image analysis: A survey," 2019, *arXiv:1907.03069*.

[10] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2921–2929.

[11] F. Zhang, M. Li, G. Zhai, and Y. Liu, "Multi-branch and multi-scale attention learning for fine-grained visual categorization," in *Proc. 27th Int. Conf. MultiMedia Modeling (MMM)*, Prague, Czech Republic, vol. 12572, in Lecture Notes in Computer Science. Berlin, Germany: Springer, Jun. 2021, pp. 136–147.

[12] Z. Yang, T. Luo, D. Wang, Z. Hu, J. Gao, and L. Wang, "Learning to navigate for fine-grained classification," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 420–435.

[13] B. Zhao, X. Wu, J. Feng, Q. Peng, and S. Yan, "Diversified visual attention networks for fine-grained object classification," *IEEE Trans. Multimedia*, vol. 19, no. 6, pp. 1245–1256, Jun. 2017.

[14] H. Zheng, J. Fu, T. Mei, and J. Luo, "Learning multi-attention convolutional neural network for fine-grained image recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5209–5217.

[15] M. Sun, Y. Yuan, F. Zhou, and E. Ding, "Multi-attention multi-class constraint for fine-grained image recognition," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 805–821.

[16] H. Zheng, J. Fu, Z.-J. Zha, and J. Luo, "Looking for the devil in the details: Learning trilinear attention sampling network for fine-grained image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5012–5021.

[17] Y. Wang, V. I. Morariu, and L. S. Davis, "Learning a discriminative filter bank within a CNN for fine-grained recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4148–4157.

[18] Y. Chen, Y. Bai, W. Zhang, and T. Mei, "Destruction and construction learning for fine-grained image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5157–5166.

[19] T. Hu, H. Qi, Q. Huang, and Y. Lu, "See better before looking closer: Weakly supervised data augmentation network for fine-grained visual classification," 2019, *arXiv:1901.09891*.

[20] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated recognition, localization and detection using convolutional networks," 2013, *arXiv:1312.6229*.

[21] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD birds-200-2011 dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2011-001, 2011.

[22] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Dec. 2013, pp. 554–561.

[23] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," 2013, *arXiv:1306.5151*.

[24] O. Russakovsky, J. Deng, H. Su, and J. Krause, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

MENGQI HE is currently pursuing the degree in advance computing with The Australian National University, specializing in machine learning. He is expected to graduate, in 2022, and continue his master's degree in machine learning and computer vision. He is experienced and passionate in algorithm and software development. He is interned as an Algorithm Engineer at PwC, JD.com, and CAS. In 2021, during his intern at CAS, he has contributed a reinforcement learning-based PID control algorithm for the Institute of Launch Vehicle.

QILONG (JERRY) CHENG is currently pursuing the degree in mechanical engineering with the University of Toronto. He is expected to graduate, in 2022, and keep on pursuing his M.Eng. degree. He was selected as the University Campus Ambassador and a Developer by Autodesk. He had been the President of Fusion Design Association. From 2020 to 2021, he has contributed a high pressure spray nozzle design patent for China State Shipbuilding Corporation during his intern.

GUANQIU QI received the Ph.D. degree in computer science from Arizona State University, in 2014. He is currently an Assistant Professor with the Computer Information Systems Department, The State University of New York at Buffalo. His primary research interests include deep learning, machine learning, and image processing, and also span many aspects of software engineering, such as software-as-a-service (SaaS), testing-as-a-service (TaaS), big data testing, combinatorial testing, and service-oriented computing.