This paper appears in IEEE Transactions on Intrumentation and Measurement
https://doi.org/10.1109/TIM.2025.3551564

Indoor Temporally Constrained Instantaneous Ego-Motion Estimation using 4D Doppler Radar

Samuel Lovett, Kade MacWilliams, Sreeraman Rajan, and Carlos Rossa

***Abstract*—Indoor ego-motion estimation using millimetre-wave Doppler sensors is challenging due to high levels of outliers, primarily caused by multi-path reflections. A standard approach to mitigate these outliers is RANdom SAmpling Consensus (RANSAC), where the ego-motion model is derived from data collected at a single time step, overlooking the continuity between successive measurements. In this paper, we demonstrate that leveraging temporal relationships across multiple time steps can improve ego-motion estimation accuracy in indoor environments. We introduce two novel RANSAC-based methods that incorporate a weighted sliding window to enhance ego-motion estimation: Temporal Sample Consensus (TEMPSAC) and Temporally Weighted Least-Squares (TWLSQ). In TEMPSAC, samples are selected with a probability weighted by their temporal proximity, and the velocity model is generated using least squares regression (LSQ). In TWLSQ, samples are uniformly selected, but the velocity model is parameterized with a temporally weighted LSQ. Both methods calculate the platform's motion by prioritizing temporally consistent inliers. Experimental validation of 18 indoor trajectories shows an average position accuracy improvement of 27% compared to previous RANSAC-based ego-motion implementations. The results demonstrate the effectiveness of incorporating temporal information into mmWave-based ego-motion estimation.**

***Index Terms*—Ego-motion, mmWave, weighted RANSAC, weighted least-squares, sliding window, temporally constrained.**

I. Introduction

EGO-MOTION estimation is the process of calculating a sensing platform's motion relative to its environment. The motion the sensor observes is either caused by the sensor's self (*ego*) motion or by objects moving in the sensor's environment. When the environment is static, this relative motion is used to infer the platform's position over time [1].

In mobile robotics, sensor-on-chip frequency modulated continuous wave (FMCW) radar in the millimetre wave bandwidth (mmWave sensor) have gained popularity for ego-motion estimation [2]. Their low-cost, and small form factor make them a favourable option for designers [3], [4]. In addition, they leverage the Doppler effect to measure the velocity of objects [5] and their wavelength allows them to operate in low visibility conditions like smoke, dust, or fog, where sensors like LIDAR and cameras typically fail [6]–[8].

Despite these advantages, mmWave sensor measurements have two key drawbacks. The first drawback is their high level of noise caused by specular (mirror-like) reflections [9], diffraction [10], and multi-path reflections. Multi-path reflections occur when the mmWave sensor's transmitted signal is reflected off multiple objects before returning to the sensor, as shown in Fig. 1 A). These multi-path reflections result in "ghost points", which act as outliers within the measurement cloud. Fig. 1 B) shows an example of a 2D mmWave point cloud, highlighting the high number of outliers present. In severe multi-path scenarios (i.e., hallway corners), these outliers can account for over 75% of samples within the measurement cloud [9], [11]. The second drawback is that mmWave point clouds are sparse, often containing two orders of magnitude fewer points than a 2D-LIDAR scan [11]. These drawbacks compound to result in a high ratio of outliers to inliers severely impacting ego-motion estimation.

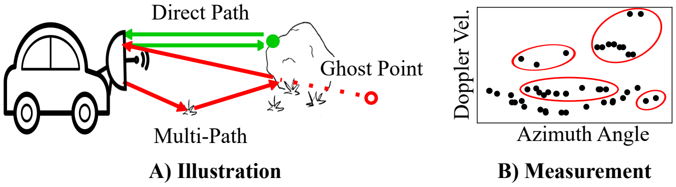

Fig. 1. Multi-path reflection noise. A) The multi-path wave is perceived by the sensor as a reflection from behind the rock, not the rock itself, generating a ghost point. B) A 2D mmWave point cloud with ghost points due to multi-path reflections circled in red. These ghost points must be removed before estimating the platform's motion.

To address these challenges for 3-degrees-of-freedom (DOF) (two translations with one in-plane rotation), researchers developed methods of estimating ego-motion using a single measurement. These "instantaneous" methods leverage the Doppler velocity and angle of arrival of static objects to infer the sensor's motion [12]–[14]. For each measurement the following 3 iterative steps are conducted: 1) fit a velocity model to a subset of samples within the measurement, 2) use the model to remove outliers, and 3) rerun step 1) without outliers to refine the model. Once the model is parametrized, the velocity is integrated to calculate the sensor's motion.

To remove measurement outliers during velocity model fitting, different algorithms have been proposed. A commonly used algorithm is the random sample consensus (RANSAC), an iterative model fitting algorithm designed for data containing outliers [12], [14]–[18]. For each measurement, RANSAC randomly selects several Doppler velocities to fit the model. Then the error between the model's estimated Doppler velocity and the measured Doppler velocity is calculated, and based on this error, samples are categorized as inliers or outliers. The

The authors are with the Department of Systems and Computer Engineering, Carleton University, Ottawa, Ontario, Canada.

This research is supported by the Queen Elizabeth II Scholarship in Science and Technology and the Society of Fire Protection Engineers Foundation Student Scholar Award. Cette recherche est appuyée par la bourse d'études supérieures en sciences et technologie de la reine Élizabeth II et le prix Student Scholar Award de la Society of Fire Protection Engineers Foundation.

process of selecting samples, fitting the model, and calculating the error is then repeated a set number of times. The model with the largest number of inliers and lowest error is selected. However, the accuracy of these methods heavily depends on the model's ability to correctly identify and reject outliers [13]. With a large number of outliers, the velocity estimate may not converge to the sensor's true velocity.

To mitigate this issue, our previous paper [19] proposed a temporally weighted sliding window. By incorporating techniques from LIDAR ego-motion ([20]) to an instantaneous ego-motion framework, the ego-motion accuracy was improved by over 19.5% compared to [12]. However, if 3-DOF estimation methods are used for platforms experiencing 6-DOF motion (three translations and three rotations) such as robots in rough terrains, drones, and wearables, the estimate cannot converge to the correct motion [21].

In this paper, we propose an extended formulation of the methods presented in [19], Temporal Sampling Consensus (TEMPSAC) and Temporally Weighted Least Squares (TWLSQ), for platforms experiencing 6-DOF motion. Both methods estimate ego-motion over a temporally weighted sliding window, as shown on the left in Fig 2. The sliding window is a first-in-first out buffer where each radar point cloud measurement is assigned a weight based on its temporal location. These point clouds are then concatenated together, and the following steps are used to calculate the ego-motion: 1) TEMPSAC selects three samples from the sliding window using a temporally weighted random selection, biasing model selection in time, as seen in the top of Fig. 2. Then least-squares regression (LSQ) is used to fit the 6-DOF Doppler velocity model to the three samples. In contrast, in TWLSQ all samples in the sliding window having an equal likelihood of being selected. The three samples selected are then used to fit the 6-DOF Doppler velocity model weighted-LSQ (WLSQ), biasing model fitting in time, as seen in the bottom of Fig. 2. 2) For both methods the error between the fitted model and the measured Doppler values is calculated, and based on this error, samples are categorized as inliers or outliers. 3) If the number of inliers is above a set threshold the model is re-calculated over all inliers, using LSQ for TEMPSAC, and WLSQ for TWLSQ. Then the average error between the model and the inliers is calculated. The process of selecting samples, fitting the model, and calculating the error is then repeated a set number of times. Then the model having the lowest error is selected as the ego-motion estimation.

The proposed methods improve estimation accuracy by concatenating multiple point clouds. Thereby, the regions of the point cloud corresponding to the true sensor velocity become densely populated with inliers, and the outliers stay randomly scattered throughout the sliding window. Therefore, the probability that samples from the true sensor velocity regions are selected to fit the Doppler velocity model is higher, leading to better initial guesses and therefore, improved convergence.

This paper makes the following major contributions:

- To the best of the authors' knowledge, this is the first application of a temporally weighted sliding window for 6-DOF ego-motion estimation using radar data. While prior studies have achieved 6-DOF estimation with radar, they rely on expensive sensors, and complex sensor fusion. In contrast, this work employs a single low-cost sensor-on-chip FMCW mmWave sensor for real-time localization.
- The benefits of a sliding window were directly evaluated by implementing the estimation method from [22] in two versions: one with a sliding window and one without. Results demonstrate that the sliding window improves performance over the baseline in 99.94% of evaluations.
- The results from our prior work on 3-DOF ego-motion estimation were replicated with an additional 15 trajectories and compared to two additional state-of-the-art methods, demonstrating the effectiveness of the proposed approach.

The paper is organized as follows. Section II summarizes the related work. Section III-A presents the theoretical formulation for 3-DOF ego-motion estimation assuming ideal data. Section III-B extends this formulation to 6-DOF. Section IV presents how mmWave measurements are filtered, with Section IV-A providing an overview of RANSAC filtering. Section IV-C & Section IV-D present the proposed methods for ego-motion estimation. In Section VI the methods under test (MUT) are experimentally validated over 1900 evaluations across four unique settings (indoor room, indoor hallways, large open underground mines, and narrow mine passageways), and the results compared to recent state-of-the-art methods [23], [24] The proposed algorithms are publicly available[1].

II. Related Work

To address the challenges related to multi-path reflections, various methods have been proposed. To review them, we assume that the mmWave sensor provides a measurement cloud of samples, and each sample contains the distance to a reflection point, the angle of arrival, and the Doppler velocity. The process of estimating these quantities from the transmitted and reflected mmWave signals is beyond the scope of this paper. However, interested readers are directed to the Texas Instruments fundamentals of mmWave sensors in [5]. Additionally, discussion of advanced signal processing techniques including feature recognition and machine learning are presented in [25]. As for methods estimating ego-motion, they can be classified as *indirect* or *direct* [26]. The *indirect* methods use radar images or point clouds for registration, whereas the *direct* methods forgo registration. The registration of measurements solves for the spatial transformation that aligns them, and is well explored in literature [20], [27].

A. 3-DOF Techniques

1) Indirect Ego-motion Estimation: For radar images or point clouds, registration is done using computer vision techniques such as SIFT [28], FAST [29], DBSCAN [30], or MSER [31] to identify key features within the measurement. These features are used to solve for the spatial transformation [32]–[34]. However, these techniques cannot handle high levels of outliers. The authors of [35], [36] used the radar's power-range spectra to filter outliers before point cloud

[1] https://github.com/samuelLovett/tempEgo3D.git

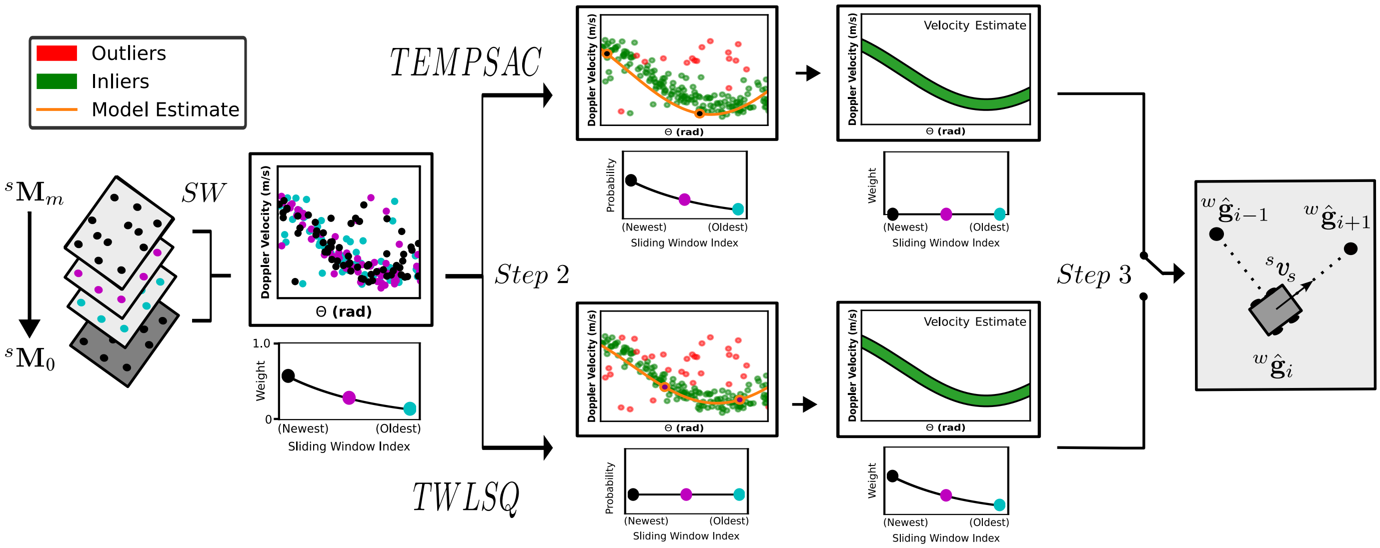

Fig. 2. The proposed ego-motion estimation methods. Measurements are placed within the sliding window and then combined. In TEMPSAC, samples are selected based on a temporally weighted probability and then the velocity model is generated using LSQ. In TWLSQ, the samples are selected having a uniform probability, but the model is parameterized with a temporally weighted LSQ. The motion of the platform is then calculated for both methods.

creation. While registration improved, identifying key features is still an issue, especially in uniform environments [35].

To sidestep the challenge of identifying features entirely, the Fourier-Mellin Transform can register radar images directly [37]. However, it requires monetarily expensive radar that creates dense 360° images. For cheaper radar, the data sparsity problem can be alleviated by using a Convolutional Generative Adversarial Network (CGAN) to create *LIDAR like* point clouds [11]. The CGAN is trained on both LIDAR and radar point clouds, and once finished only the radar is needed for perception, with the *LIDAR like* clouds used for registration.

2) Direct Ego-motion Estimation: In [14] the instantaneous strategy introduced in Section I was adapted for imaging radar. The algorithm fits a polynomial curve directly to the radar image to estimate the velocity model rather than using LSQ. Separately, the work in [12] was extended to use multiple radar sensors in [13]. In the extended work, RANSAC was replaced by Maximum-Likelihood Sampling Consensus (MLESAC) to maximize the solution's log likelihood. But both solutions are still limited by the number of outliers in a measurement. With too many outliers present, the estimates may not converge.

B. 6-DOF Techniques

The following is a description of the related 6-DOF machine learning and *direct* ego-motion estimation techniques, other techniques like feature-based methods, are reviewed in [8].

1) Machine Learning Techniques: To obtain reliable ego-motion estimates in 6-DOF machine learning (ML) is often used. In [38], a dense neural network (NN) and a convolutional NN were used to localize a human from a static mmWave generated point cloud. The networks independently downsample the cluttered and outlier filled point clouds to a single point, more accurately estimating the person's location. The results show an improved localization accuracy by 88% compared to localizing the person by averaging all samples in the cloud. However, the results may not generalize to a moving scenario. Other machine learning techniques offer an end-to-end solution to the registration and mapping problem. RadarLoc uses a NN with geometric constraints between radar measurements to improve localization accuracy [39]. Whereas, Milli-RIO uses an unscented Kalman filter to fuse the data from a mmWave sensor and an Inertial Measurement Unit (IMU) before utilizing a long-term short-term memory motion model to estimate the ego-motion [40]. On the other hand, MilliEgo fuses data from a mmWave radar and an IMU using a multi-modal sensor fusion network, achieving real-time estimation performance [9]. While the results of these ML methods are impressive, they require costly computations, large amounts of data, and their ability to generalize to new locations is limited.

2) Direct Ego-motion Estimation: Instead of using machine learning, ego motion estimation can be posed as a state estimation problem [17], [23], [24], [41]. In [23], a hybrid *direct* / *indirect* approach is used where an iterated Kalman filter combines the *indirect* ego-motion estimates from radar image registration, with the *direct* estimates from an IMU and the graduated non-convexity method (an alternative to RANSAC [42]) to estimate the platform's motion. The results, validated using in-house data and the ColoRadar dataset [43], show metre level accuracy. However, this method is only applicable to imaging radar, which generates a larger number of samples at a higher resolution than low-cost radar sensors.

As an alternative to imaging radar, a sliding window of sparse measurements from a low-cost mmWave sensors and IMU can be used [41]. The window is batch optimized, jointly estimating the platform's states using the Doppler data and applying motion constraints from the IMU. This method outperformed the Intel Realsense stereo camera's odometry output. To estimate ego-motion without a sliding window, a hybrid *direct* / *indirect* approach is used in [24]. Modelling the measurement uncertainty of the mmWave and the IMU improves velocity estimation and point cloud registration. However, current research shows that for a mmWave sensor and an IMU, *direct* methods will outperform *indirect* methods [44]. Alternatively, two radar sensors can be mounted orthogonally from one another (i.e., one facing forward and the other sensor facing the ground), increasing the number of samples per point cloud [17], and outperforming [9], [45]. However, all the above methods require the Jacobean to be calculated, which can be computationally demanding and time-consuming.

A single radar sensor can be used for ego-motion estimation by replacing the original sinusoidal velocity relationship presented in [12] with a sinusoidal surface model [22]. The outlier rejection is conducted using MLESAC, and the model was fit using orthogonal distance regression (ODR). ODR minimizes the orthogonal distances (error) between the measurement and model along all variables, unlike LSQ [46]. This better accounts for the measurement uncertainty inherent to mmWave sensors [13], [22]. The ODR formulated in [22] is a nonlinear optimization problem requiring the Jacobian to be computed, which can be difficult and time-consuming. As with [12], this estimate diverges from the true motion if the measurement has a high number of outliers.

III. Instantaneous Ego-Motion Estimation

The following subsections detail the theoretical foundation of ego-motion estimation for 3-DOF motion (III-A) and 6-DOF motion (III-B). These sections assume that the radar measurements are outlier free point clouds (containing only static samples and no multi-path ghost points).

Hereafter a matrix is written in bold uppercase, a vector in bold lowercase, a scalar in unbolded lowercase, and their superscript defines the reference frame in which they are being expressed. Further, the sensing platform in this section is equipped with a mmWave sensor, and three coordinate frames defined as shown in Fig. 3A. The fixed world reference frame RF_w is where the ego-motion of the platform's body frame RF_b is observed. Attached to the RF_b is the sensor reference frame RF_s where the mmWave measurements are generated. As the mmWave sensor moves in a static environment, it can equivalently be represented as a stationary mmWave sensor with moving environment. Using this representation moving forward, the environment has a relative motion equal and opposite to the sensor's original motion.

A. Three Degrees-of-Freedom Ego Motion Estimation

For a given measurement index i, a mmWave point cloud ${}^{s}\mathbf{M}_i$ is defined as a collection of a_i samples ${}^{s}\mathbf{p}_{j=1,\ldots,a_i}$ measured at i such that

$$ {}^{s}\mathbf{M}_i = \begin{bmatrix} {}^{s}\mathbf{p}_1 & \ldots & {}^{s}\mathbf{p}_{a_i} \end{bmatrix} \tag{1} $$

$$ {}^{s}\mathbf{p}_j = \begin{bmatrix} {}^{s}\theta_j & {}^{s}v_{d_j} \end{bmatrix}^T \tag{2} $$

where ${}^{s}\theta_j$ is the angular location and ${}^{s}v_{d_j}$ is the Doppler speed of the j^{th} point ${}^{s}\mathbf{p}_j$ in RF_s, respectively, as illustrated in Fig. 3A. The Doppler speed ${}^{s}v_{d_j}$ is the component of the object's velocity (-${}^{s}v_s$) in the radial direction. In other words, the component along the line projecting from the object to the centre of RF_s at the angle ${}^{s}\theta_j$ with respect to the horizontal axis ${}^{s}\mathbf{x}$ of RF_s. As the platform moves through the environment with a constant speed, the measured Doppler speed of static objects will change depending on their location. This is shown in Fig. 3A by the snapshots of the platform at measurement index $i = 1$ and $i = 2$. When multiple objects are observed within one measurement, like the black circles in Fig. 3B, the Doppler speed has a sinusoidal relationship with the sensor's speed ${}^{s}v_s$ [12]. By transforming these measurements from the ${}^{s}\mathbf{x}$-${}^{s}\mathbf{y}$ domain to the ${}^{s}\theta$-${}^{s}v_d$ domain, as in Fig. 3C, the object's Doppler speed can be modelled as,

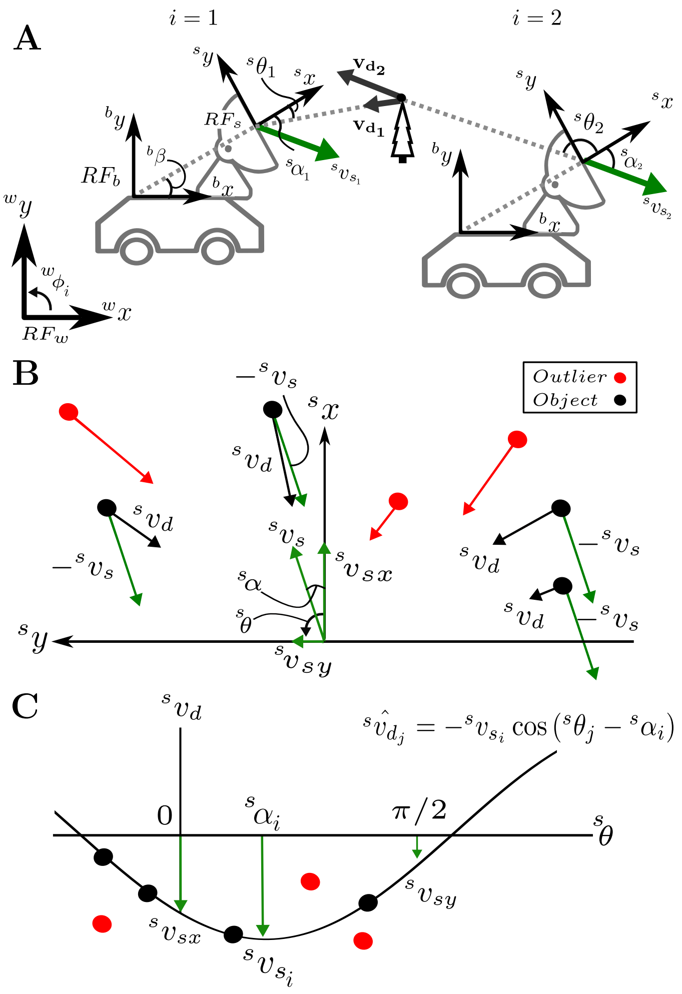

Fig. 3. **A)** Doppler velocity measurement ${}^{s}\mathbf{v}_{d_i}$ from a mobile platform moving with speed ${}^{s}v_s$ at an angle ${}^{s}\alpha$ with respect to (w.r.t) ${}^{s}\mathbf{x}$. At $i = 1$ and $i = 2$, the sensor measures ${}^{s}\mathbf{v}_{d_1}$ and ${}^{s}\mathbf{v}_{d_2}$, respectively, in the sensor reference frame RF_s along ${}^{s}\theta$ w.r.t. ${}^{s}\mathbf{x}$. The orientation of the body reference frame RF_b w.r.t to the world reference frame RF_w, and of RF_s w.r.t RF_b, are given by ${}^{w}\phi$ and ${}^{b}\beta$, respectively. **B)** Example of a mmWave measurement ${}^{s}\mathbf{M}_i$ with all static samples shown in black and outlier samples shown in red. In RF_s objects appear moving with velocity $-{}^{s}v_s$. Here the i and j subscripts have been omitted for readability. **C)** Samples from **B)** are viewed in the ${}^{s}\theta$-${}^{s}v_d$ domain. The inlier samples fit a sinusoidal curve ${}^{s}\hat{v}_{d_j} = -{}^{s}v_{s_i}\cos({}^{s}\theta_j - {}^{s}\alpha_i)$ used to estimate the platform's velocity.

$$ {}^{s}\hat{v}_{d_j} = -{}^{s}v_{s_i}\cos\left({}^{s}\theta_j - {}^{s}\alpha_i\right) \tag{3} $$

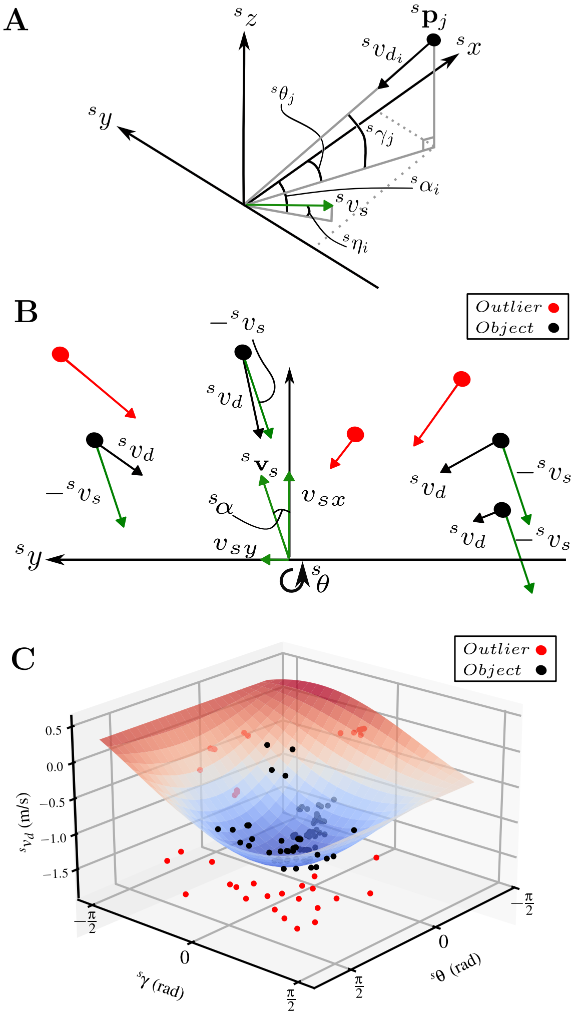

Fig. 4. **A)** Example of $^s\mathbf{p}_j$ and $^s\mathbf{v}_s$ in RF_s. RF_b and RF_w follow the same structure as in Fig. 3A, however, now having an additional axis ($^s\mathbf{z}$) sticking out of the page. **B)** Cloud $^s\mathbf{M}_i$ of all samples measured at measurement index i from objects (black) and outliers (red). In RF_s objects appear moving with velocity $-^sv_s$. For readability, only the $^sx-^sy$ axes are shown. **C)** The cosine surface from (12) plotted along side $^s\mathbf{M}_i$ in the $^s\theta$-$^s\gamma$-^sv_d domain. The object samples (black) fit the cosine surface, whereas the false measurements (red) do not.

where $^s\hat{v}_{d_j}$ is the estimated Doppler speed of the model, which is parameterized using $^s\alpha_i$ and $^sv_{s_i}$. Here, $^s\alpha_i$ is the angular direction of the sensor's speed $^sv_{s_i}$ at measurement index i. Expanding (3) the model becomes,

$$^s\hat{v}_{d_j} = -^sv_{s_i}\left[\cos(^s\theta_j)\cos(^s\alpha_i) + \sin(^s\theta_j)\sin(^s\alpha_i)\right]. \quad (4)$$

To solve for the unknown $^s\alpha_i$ and $^sv_{s_i}$ parameters in (3), Eq. (4) can be reformulated as a matrix equation:

$$\begin{bmatrix} ^sv_{d_1} \\ \vdots \\ ^sv_{d_j} \end{bmatrix} = \begin{bmatrix} \cos{^s\theta_1} & \sin{^s\theta_1} \\ \vdots & \vdots \\ \cos{^s\theta_j} & \sin{^s\theta_j} \end{bmatrix} \begin{bmatrix} ^sv_{s_i}\cos(^s\alpha_i) \\ ^sv_{s_i}\sin(^s\alpha_i) \end{bmatrix} \quad (5)$$

containing all objects $j = 1, \ldots, a_i$ in $^s\mathbf{M}_i$, and solved using the measured Doppler velocities via LSQ. The resulting $^s\alpha_i$ and the sensor speed $^sv_{s_i}$ are then converted to the Cartesian vector, $^s\mathbf{v}_{s_i}$. In order to calculate the motion of the platform, $^s\mathbf{v}_{s_i}$ must be transformed into RF_w through a series of homogeneous transformations,

$$^w\tilde{\mathbf{v}}_{s_i} = {^w\mathbf{H}_{b_i}}\,{^b\mathbf{H}_s}\,{^s\tilde{\mathbf{v}}_s} \quad (6)$$

where $^s\tilde{\mathbf{v}}_{s_i} = [^s\mathbf{v}_{s_i} \quad 1]^T$, $^b\mathbf{H}_s$ is the homogeneous transformation from RF_b to RF_s governed by rotation angle $^b\beta$ between $^s\mathbf{x}$ and $^b\mathbf{x}$ about $^s\mathbf{x} \times {^s\mathbf{y}}$, and $^w\mathbf{H}_{b_i}$ is the homogeneous transformation from RF_w to RF_b governed by the rotation angle $^w\phi_i$ between $^b\mathbf{x}$ and $^w\mathbf{x}$ about $^w\mathbf{x} \times {^w\mathbf{y}}$ at i (i.e, the platform's orientation in RF_w, see Fig. 3A).

As all quantities are now in the world coordinate frame and the only velocity left is the sensor velocity $^w\mathbf{v}_s$, moving forward the superscripts and subscripts are dropped for convenience. With the sensor velocity $\mathbf{v}_i$ known, all that is needed to estimate the platform's change in pose ($\Delta\hat{\mathbf{g}}_i$) is the change in time between the current and past measurement (i.e., Δt_i = $t_i - t_{i-1}$). Therefore,

$$\Delta\hat{\mathbf{g}}_i = [\mathbf{v}_i \cdot \Delta t_i \quad \phi_i]^T. \quad (7)$$

The platform's current displacement is the sum of all prior relative displacements (i.e., $\mathbf{v}_i{\cdot}\Delta t_i$) from the first measurement index $f = 1$ to the current measurement index i,

$$\mathbf{d}_i = \sum_{f=1}^{i} \mathbf{v}_f \cdot \Delta t_f. \quad (8)$$

The absolute pose estimate of the platform $\hat{\mathbf{g}}_i$ is defined as

$$\hat{\mathbf{g}}_i = [\mathbf{d}_i \quad \phi_i]^T. \quad (9)$$

The series of absolute pose estimates from measurement f to i represents the estimated trajectory the platform followed.

B. Six Degrees-of-Freedom Ego Motion Estimation

The mmWave point clouds is now redefined as $^s\mathbf{M}_i$,

$$^s\mathbf{M}_i = \begin{bmatrix} ^s\mathbf{p}_1 & \ldots & ^s\mathbf{p}_{a_i} \end{bmatrix} \quad (10)$$

$$^s\mathbf{p}_j = \begin{bmatrix} ^s\theta_j & ^s\gamma_j & ^sv_{d_j} \end{bmatrix}^T \quad (11)$$

where $^s\theta_j$ is the azimuth angle, $^s\gamma_j$ is the elevation angle and $^sv_{d_j}$ is the 1D Doppler velocity of the j^{th} point $^s\mathbf{p}_j$ in RF_s, respectively, as illustrated in Fig. 4A. The Doppler velocity $^sv_{d_j}$ is the component of the object's velocity (-^sv_s) in the radial direction. In other words, the component along the line projecting from the object to the centre of RF_s at azimuth $^s\theta_j$ and elevation $^s\gamma_j$. $^s\theta_j$ is defined with respect to the horizontal axis $^s\mathbf{x}$ and spans the $^s\mathbf{x}$-$^s\mathbf{y}$ plane of RF_s, while $^s\gamma_j$ is defined with respect to $^s\theta_j$ and spans the $^s\theta_j$-$^s\mathbf{z}$ plane of RF_s. As the platform moves through the environment with a constant velocity, the measured Doppler velocity of static objects will change as their relative location changes [12] [19]. In Section III-A this relationship is sinusoidal in nature, however, for motion in 6-DOF it is governed by a sinusoidal surface. By transforming the measurement from Cartesian ($^s\mathbf{x}$-$^s\mathbf{y}$-$^s\mathbf{z}$) to the azimuth-elevation-Doppler domain ($^s\theta$-$^s\gamma$-^sv_d), as in Fig. 4C the object's Doppler velocity can be modelled as,

$$\begin{aligned} ^s\hat{v}_{d_j} = &-^sv_{s_i}\cos(^s\gamma_j)\cos(^s\eta_i)\cos\left(^s\theta_j - {^s\alpha_i}\right) \\ &-^sv_{s_i}\sin(^s\gamma_j)\sin(^s\eta_i) \end{aligned} \quad (12)$$

where ${}^s\hat{v}_{d_j}$ is the estimated Doppler velocity of the model, which is parameterized using ${}^s\eta_i$, ${}^s\alpha_i$, and ${}^s v_{s_i}$. Here, ${}^s\eta_i$ and ${}^s\alpha_i$ are the angular directions of the sensor's velocity ${}^s v_{s_i}$ at measurement index i. Expanding (12) the model becomes,

$$\begin{aligned}{}^s\hat{v}_{d_j} = -{}^s v_{s_i}[&\cos({}^s\gamma_j)\cos({}^s\eta_i)\cos({}^s\theta_j)\cos({}^s\alpha_i)+\\ &\cos({}^s\gamma_j)\cos({}^s\eta_i)\sin({}^s\theta_j)\sin({}^s\alpha_i)+\\ &\sin({}^s\gamma_j)\sin({}^s\eta_i)].\end{aligned} \tag{13}$$

The first term corresponds to the ${}^s\mathbf{x}$-component of the Doppler velocity, whereas the second and third term correspond to the ${}^s\mathbf{y}$ and ${}^s\mathbf{z}$-component, respectively.

To solve for the unknown ${}^s\eta_i$, ${}^s\alpha_i$ and ${}^s v_{s_i}$ parameters in (12), Eq. (13) can be reformulated as a matrix equation:

$$\begin{bmatrix}{}^s v_{d_1}\\ \vdots \\ {}^s v_{d_j}\end{bmatrix} = \begin{bmatrix}\cos({}^s\gamma_1)\cos({}^s\theta_1) & \cos({}^s\gamma_1)\sin({}^s\theta_1) & \sin({}^s\gamma_1)\\ \vdots & \vdots & \vdots \\ \cos({}^s\gamma_j)\cos({}^s\theta_j) & \cos({}^s\gamma_j)\sin({}^s\theta_j) & \sin({}^s\gamma_j)\end{bmatrix} \cdot \begin{bmatrix}-{}^s v_s\cos({}^s\eta_i)\cos({}^s\alpha_i)\\ -{}^s v_s\cos({}^s\eta_i)\sin({}^s\alpha_i)\\ -{}^s v_s\sin({}^s\eta_i)\end{bmatrix} \tag{14}$$

containing all objects $j = 1, \ldots, a_i$ in ${}^s\mathbf{M}_i$. (14) is solved via regression using the measured angles and Doppler velocities. The resulting ${}^s\eta_i$, ${}^s\alpha_i$ and the sensor velocity ${}^s v_{s_i}$ are then converted to the Cartesian vector, ${}^s\mathbf{v}_{s_i}$. Next, to calculate the motion of the platform, ${}^s\mathbf{v}_{s_i}$ must be transformed into RF_w through a series of homogeneous transformations,

$${}^w\tilde{\mathbf{v}}_{s_i} = {}^w\mathbf{H}_{b_i}\,{}^b\mathbf{H}_s\,{}^s\tilde{\mathbf{v}}_{s_i} \tag{15}$$

where ${}^s\tilde{\mathbf{v}}_{s_i} = [{}^s\mathbf{v}_{s_i} \quad 1]^T$, ${}^b\mathbf{H}_s$ is the homogeneous transformation from RF_b to RF_s governed by the rotation with respect to RF_b which aligns RF_b with RF_s, and ${}^w\mathbf{H}_{b_i}$ is the homogeneous transformation from RF_w to RF_b governed by the rotation with respect to RF_w which aligns RF_w with RF_b at i (see Fig. 4A).

As all quantities are now in the world coordinate frame and the only velocity left is the sensor velocity ${}^w\mathbf{v}_s$, in the rest of this section the superscripts and subscripts are dropped for convenience. With the sensor velocity $\mathbf{v}_i$ known and given the change in time between the current and past measurement (i.e., $\Delta t_i = t_i - t_{i-1}$), the platform's change in pose ($\Delta\hat{\mathbf{g}}_i$) is:

$$\Delta\hat{\mathbf{g}}_i = [\mathbf{v}_i \cdot \Delta t_i \quad \mathbf{q}_i]^T, \tag{16}$$

and $\mathbf{q}_i$ is the platform's current orientation as a quaternion. The platform's current displacement is then calculated as in (8), and the absolute pose estimate of the platform $\hat{\mathbf{g}}_i$ is

$$\hat{\mathbf{g}}_i = [\mathbf{d}_i \quad \mathbf{q}_i]^T. \tag{17}$$

The series of absolute pose estimates from the first measurement f to i represents the estimated trajectory followed.

IV. Filtering Using Random Sample Consensus

The previous section assumed that all samples are inliers (static and not ghost points). When measurements contain outliers filtering is required to accurately to fit the velocity model. The following subsections present an overview of the proposed filtering methods. Subsection IV-A gives a brief overview of RANSAC filtering, Subsection IV-B defines the proposed temporally weighted sliding window, and Subsection IV-C and IV-D describes how the temporally weighted sliding window is utilized within a RANSAC framework for the proposed filtering methods, Temporal Sampling Consensus (TEMPSAC), and Temporally Weighted Least-Squares (TWLSQ), for ego-motion estimation respectively. The presented formulations combine the notation and variables for the 3-DOF and 6-DOF case. The cases are distinguished by the dimension of the point being used ${}^s\mathbf{p}_j$.

A. Random Sample Consensus (RANSAC)

RANSAC is a popular model estimation algorithm robust to data with many outliers [15], [47]. RANSAC relies on a hypothesize-and-verify approach, repeatedly estimating the model based on randomly selected samples [48]. This approach allows for RANSAC to globally search the estimation space for the best solution instead of iteratively improving a single local solution, like gradient descent. RANSAC has three main steps, 1) estimate model parameters, 2) partition data, and 3) determine best partition and refit model.

Step 1 — Estimate model parameters: Given a measurement ${}^s\mathbf{M}_i$ of a_i samples ${}^s\mathbf{p}_j$, a *random* selection of n samples from ${}^s\mathbf{M}_i$ can be used to find an initial estimate of the sensor velocity by solving (5) or (14) through LSQ regression [15]. For a cosine curve, the minimum number of samples to fit the curve is $n = 2$. Whereas for a cosine surface, the minimum number of samples to fit the curve is $n = 3$ [22].

The random selection of an entry ${}^s\mathbf{p}_j$ is governed by the uniform probability distribution such that,

$$\text{Pr}({}^s\mathbf{p}_j) = U[1, a_i], \tag{18}$$

where $\text{Pr}(\cdot)$ is the probability function of a random variable.

Step 2 — Partition the data: For a point ${}^s\mathbf{p}_{j=1,\ldots,a_i} \in^s \mathbf{M}_i$, the error e_j between the measured point's Doppler speed ${}^s v_{d_j}$ and the speed estimated from the model fit in Step 1 ${}^s\hat{v}_{d_j}$ is,

$$e_j = ({}^s v_{d_j} - {}^s\hat{v}_{d_j})^2. \tag{19}$$

A point ${}^s\mathbf{p}_j$ is considered an inlier if the error is below a predefined threshold ε, that is:

$$inliers = \{{}^s\mathbf{p_j} \mid e_j < \varepsilon\}. \tag{20}$$

The number of inliers c is then counted. If c is larger than a predefined threshold ρ, then the algorithm proceeds to Step 3, otherwise it goes back to Step 1.

Step 3 — Determine best partition and refit model: LSQ is then repeated to update the parameters in (3) or (12) using all identified inliers. The error in (19) is re-evaluated for all inliers. The quality of the model is calculated as the mean error of all inliers:

$$\bar{e} = \frac{1}{c}\sum_{j=1}^{c} e_j \qquad {}^s\mathbf{p_j} \in inliers. \tag{21}$$

This process is repeated for k initial estimates. The model parameters that result in the lowest error parametrizes (3) and (12), which is used to find the platform's motion ${}^w\mathbf{d}_i$.

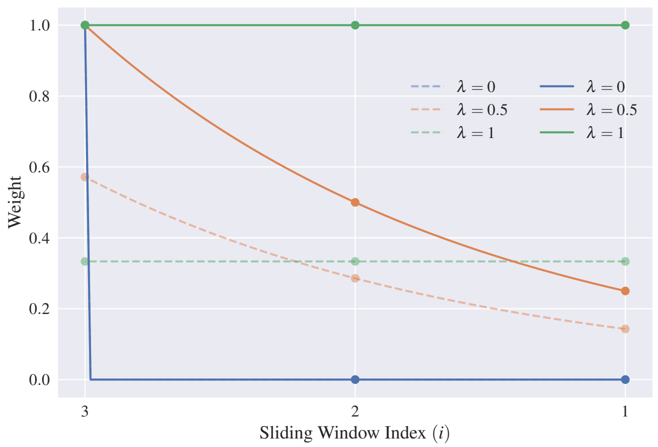

Fig. 5. A family of temporal weight curves for a sliding window length of $m = 3$ with $\lambda = 0$ (blue), $\lambda = 0.5$ (orange), and $\lambda = 1$ (green). The solid lines, and dashed lines are calculated using (25), and (26), respectively. The dots along each curve correspond to the weight assigned to each sliding window index (i). With $i = 3$ being the newest measurement and $i = 1$ the oldest. Comparing the $\lambda = 0.5$ curves, the solid curve where w_i' is not normalized has a larger difference between each weight value.

B. Temporally Weighted Sliding Window

As shown in [19], by estimating ego-motion over a sliding window, the temporal dependence of inlier samples and the temporal independence of outlier samples between measurements can be leveraged to improve ego-motion estimation. The first-in-first-out sliding window **SW** is defined as the set of m recent measurements, as shown on the left of Fig. 2. Henceforth, i will be used as the index within **SW** such that,

$$\mathbf{SW} = \left\{ {}^s\mathbf{M}_{i=0}, \quad \dots, \quad {}^s\mathbf{M}_{i=m} \right\} \tag{22}$$

where $i = 0$ is the oldest measurement and $i = m$ the newest measurement. Further, **SW** is also the superset containing all samples ${}^s\mathbf{p}_{j=1,\dots,a_i} \;\forall\; {}^s\mathbf{M}_i \in \mathbf{SW}$, therefore **SW** can equivalently be defined as the set of all samples

$$\mathbf{SW} = \left\{ {}^s\mathbf{p}_1, \quad \dots, \quad {}^s\mathbf{p}_{a_{sw}}, \right\} \tag{23}$$

where $a_{sw} = \sum_{i=0}^{m} a_i$.

However, by calculating ego-motion over **SW** a time delay is introduced into the estimation. To account for this and ensure that the proposed methods respond quickly to changes in motion, a temporal weighting w_i is applied to each measurement within **SW** such that equation (2) and (10) become

$${}^s\mathbf{p}_j = \begin{bmatrix} {}^s\theta_j & {}^s v_{d_j} & w_i \end{bmatrix}^T \;\&\; {}^s\mathbf{p}_j = \begin{bmatrix} {}^s\theta_j & {}^s\gamma_j & {}^s v_{d_j} & w_i \end{bmatrix}^T \tag{24}$$

$$\forall \quad {}^s\mathbf{M}_i \in \mathbf{SW},$$

for the 3-DOF and 6-DOF case respectively. This index dependent weight is calculated as,

$$w_i = \lambda^{m-i} \qquad\qquad \lambda \in [0, 1] \tag{25}$$

where $0 \leq \lambda \leq 1$ is the user defined fixed forgetting factor as in [49]. As λ goes from 0 to 1, w_i goes from placing all the weight on the most recent measurement to weighting all measurements equally, as shown in Fig. 5.

C. Ego-Motion Estimation Using Temporal Sampling Consensus (TEMPSAC)

The first proposed solution is Temporal Sampling Consensus (TEMPSAC), a RANSAC variant which estimates the velocity across the sliding window and normalizes the temporal weights to favour model generation from newer measurements in time. The normalized weights w_i' are calculated as,

$$w_i' = \frac{\lambda^{m-i}}{\sum_{i=0}^{m} \lambda^{m-i}} \qquad\qquad \lambda \in [0, 1] \tag{26}$$

TEMPSAC follows the same three-step format as RANSAC,

Step 1 — Estimate model parameters: Given the set **SW** from (23) containing $a_{\mathbf{sw}}$ samples ${}^s\mathbf{p}_j$, n samples are *randomly* selected and used to find an initial estimate of the sensor velocity by solving (5) or (14) through LSQ regression. In contrast to RANSAC the random selection of an entry ${}^s\mathbf{p}_j$ is now governed by the w_i',

$$\Pr({}^s\mathbf{p}_j) = \frac{w_i'}{a_i} \qquad\qquad \forall \; {}^s\mathbf{M}_i \in \mathbf{SW}. \tag{27}$$

With the weights normalized w_i' can be thought of as the probability of remembering a measurement ${}^s\mathbf{M}_i$ from **SW** i.e., $\Pr(\mathbf{M} = {}^s\mathbf{M}_i) = w_i'$. This probability distribution, shown in Fig. 5 by the dashed lines, and is governed by (25), biasing TEMPSAC to generate models from the samples in more recent measurements.

Step 2 — Partition the data: For a point ${}^s\mathbf{p}_{j=1,\dots,a_{\mathbf{sw}}} \in \mathbf{SW}$, the error e_j between the measured point's Doppler velocity ${}^s v_{d_j}$ and the velocity estimated from the model fit in Step 1 ${}^s\hat{v}_{d_j}$ is given by (19). ${}^s\mathbf{p}_{j=1,\dots,a_{\mathbf{sw}}}$ is an inlier if $e_j < \varepsilon$. If the number of inliers c is larger than ρ the algorithm proceeds to Step 3, otherwise a new model is generated in Step 1.

Step 3 — Determine the best partition and refit the model: With the outliers now discarded, the estimated model (12) is improved by recomputing LSQ and then (19) is re-evaluated. The quality of the model is calculated as (21), and the model with the lowest error after k initial estimates parametrizes (3) or (12) and is used to find the platform's motion ${}^w\mathbf{d}_i$.

D. Ego-Motion Estimation Using Temporally Weighted Least Squares (TWLSQ)

The second proposed solution is Temporally Weighted Least Squares (TWLSQ), another RANSAC variant which estimates the velocity across a sliding window. In TEMPSAC a weighted random selection is used to bias the initial model estimate in time, whereas TWLSQ's uniform selection generates an initial guess where successive measurements overlap, as at the bottom of the cosine surface in Fig. 4 C). This biases the initial guess to be consistent with the motion experienced across the sliding window. But to ensure that the estimates do not lag behind the true motion, the temporal weights w_i are used to bias the regression steps. With (25) is no longer normalized as in [19], the difference in weight for each measurement in the sliding window is larger, as seen in Fig.5. The implication being that weights will now have a greater impact on the WLSQ stages of TWLSQ. With more weight placed on the most recent measurements, the regression will more closely fit the newest inliers which most closely reflect the sensing platform's current motion.

TWLSQ follows the same three-step format as RANSAC,

Step 1 — Estimate model parameters: Given the set $\mathbf{SW}$ containing $a_{\mathbf{sw}}$ samples ${}^s\mathbf{p}_j$, a *random* selection of n samples can be used to find the sensor velocity by solving (5) or (14) through a *weighted-LSQ (WLSQ) regression*. The random selection of samples is governed by (18).

Step 2 — Partition the data: For a point ${}^s\mathbf{p}_{j=1,\ldots,a_{\mathbf{sw}}} \in \mathbf{SW}$, the *weighted* error e_j between ${}^sv_{d_j}$ and ${}^s\hat{v_{d_j}}$ is given by,

$$e_j({}^s\theta_j) = w_i({}^sv_{d_j} - {}^s\hat{v_{d_j}})^2, \tag{28}$$

and ${}^s\mathbf{p}_j$ is an inlier if $e_j < \varepsilon$. Like before, if $c > \rho$ proceed to Step 3, otherwise go back to Step 1.

Step 3 — Determine best partition and refit model: With the outliers discarded, WLSQ is repeated to improve parameters in (12), and then the error is (28) is re-evaluated. The quality of the improved model is calculated as the mean *weighted* error,

$$\bar{e} = \frac{1}{c}\sum_{j=1}^{c} e_j \qquad {}^s\mathbf{p_j} \in inliers. \tag{29}$$

The model parameters which result in the lowest error after k initial estimates is used to parametrize (3) or (12) which is then used to determine the platform's motion ${}^w\mathbf{d}_i$.

V. Hyperparameter Optimization

TEMPSAC and TWLSQ both require five hyperparameters to be specified by the user: 1) the fixed forgetting factor λ, 2) the sliding window length m, 3) the maximum number of iterations k before terminating the estimation, 4) the threshold distance ε away from the model along the Doppler velocity axis to define an inlier, and 5) the minimum required number of inliers ρ to certify a usable model. To optimize these hyperparameters a multi-variable genetic algorithm (GA) was implemented. As the name implies, a GA is an optimization heuristic inspired by the principles of natural selection and genetics [50]. The algorithm mimics the evolutionary process by improving a population of solution candidates over multiple generations to find the optimal or near-optimal solution [51].

A GA begins by randomly initializing a population of solutions that will cover a large portion of the search space. Each solution is represented as a chromosome and its quality scored using a cost function. Using the score, solutions are selected as parents with the "more fit" solutions having a higher chance of being chosen. The parents then undergo *crossover*, where parts of their chromosomes are exchanged, making a new solution (offspring). To reduce the likelihood of converging to local minima, genetic diversity is maintained through *mutation*; applying small changes to some offspring. The new offspring then replace the old population and the cycle repeated over generations until terminated. This flexible structure allows GAs to simultaneous optimize continuous variables (λ, ε) and integer variables (m, k, ρ).

To assert that one set of hyperparameters produces a better score than another, the model must generate the same output when given identical inputs. Yet, the proposed models are non-deterministic as they rely on randomness to select samples to parameterize the velocity model. Therefore, when provided with the same input data, the proposed methods can generate different results, as seen in (18) and (27). This randomness is known to introduce uncertainty in the final estimation [52], and thereby in determining the best set of hyperparameters. While the hyperparameters obtained from multiple optimization runs could be averaged, as in [19], this can still cause the model to have a wide error distribution. To ensure that only a single optimization is needed and estimates are more precise, a new optimization strategy is required that accounts for both the *estimation error* and the *estimation uncertainty*.

To account for the *estimation error* the relative pose error (RPE_i) is included in the cost function. The relative pose error quantifies ego-motion accuracy by comparing the estimated change in trajectory ((7) and (16)) with the ground truth change in trajectory ($\Delta\mathbf{g}_i$) [53], [54]. It is calculated as $RPE_i = \hat{\Delta\mathbf{g}_i} \ominus \Delta\mathbf{g}_i$, where $\ominus$ is an operator that computes the relative transformation between poses [55]. For a trajectory of l poses, the mean relative error per estimate is computed as the root-mean-squared relative pose error (*RMS-RPE*),

$$\textit{RMS-RPE} = \sqrt{\frac{1}{l}\sum_{i=1}^{l}(\hat{\mathbf{g}}_i^w \ominus \mathbf{g}_i^w)^2}. \tag{30}$$

To account for the *estimation uncertainty, we assume that the ego-motion estimation for a given measurement can be modelled as an unknown Gaussian distribution whose mean value μ corresponds to the estimations true value (resulting in a relative pose error of zero), and whose standard deviation σ corresponds to the estimation's uncertainty. Therefore, any deviation in the relative pose error from zero is caused by the estimation's uncertainty. The larger the uncertainty, the greater the pose estimate will deviate from the ground truth, producing a wider error distribution, and a larger standard deviation of the relative pose error. By finding the hyper-parameters that minimize the distribution's spread (σ) the distribution is forced to be narrow, reducing the uncertainty.*

Given a trajectory l, the standard deviation is calculated as,

$$\sigma_{RPE} = \frac{1}{l}\sum_{i=0}^{l}\left(RPE_i - \overline{RPE}\right)^2. \tag{31}$$

The proposed cost function must minimize the average relative error and its standard deviation. Therefore, the cost function for the hyperparameter and uncertainty optimization is formulated as a mixed integer programming problem,

$$\begin{array}{rl} \text{minimize} & \textit{RMS-RPE} + r\sigma_{RPE} \\ \text{subject to:} & 0 \leq \lambda \leq 1 \qquad \varepsilon > 0 \\ & 1 \leq \rho \leq \min\limits_i \{a_i\} \qquad m, k, \rho, \in \mathbb{N}^+. \end{array} \tag{32}$$

Due to the size difference between RPE and its variance, a scaling value r is applied to ensure equal weighting of both.

A. Genetic Algorithm Implementation

The GA was implemented using the *pymoo* Python package [56]. The algorithm used (32) as the cost function, and a population size of 100. The termination criterion was set to

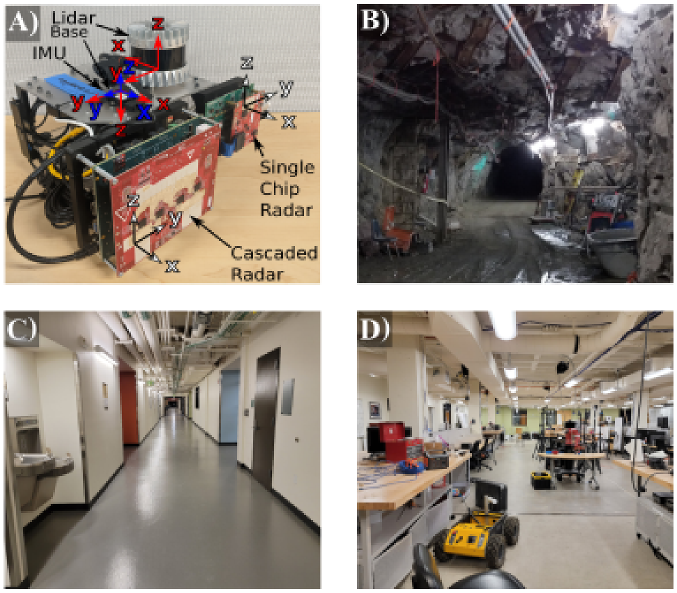

Fig. 6. Handheld sensing platform used for acquiring the ColoRadar dataset (A). The sensors and coordinate frames are labelled, with the base frame shown in blue. Images of the ColoRadar dataset locations. B) Trajectory 1-5, a large open underground mine. C) Trajectory 10-14 location, indoor hallways. D) Trajectories 15-18, an indoor room. Modified from [43].

TABLE I
ColoRadar Datasets Used.

Trajectory	ColoRadar Dataset Sequence	Length (m)
Trajectory 1	2_23_2021_edgar_army_run0	341
Trajectory 2	2_23_2021_edgar_army_run1	194
Trajectory 3[a]	2_23_2021_edgar_army_run2	131
Trajectory 4	2_23_2021_edgar_army_run3	240
Trajectory 5	2_23_2021_edgar_army_run4	344
Trajectory 6	2_23_2021_edgar_classroom_run2	207
Trajectory 7	2_23_2021_edgar_classroom_run3	306
Trajectory 8[a]	2_23_2021_edgar_classroom_run4	183
Trajectory 9	2_23_2021_edgar_classroom_run5	187
Trajectory 10	12_21_2020_ec_hallways_run0	111
Trajectory 11	12_21_2020_ec_hallways_run1	181
Trajectory 12	12_21_2020_ec_hallways_run2	272
Trajectory 13	12_21_2020_ec_hallways_run3	278
Trajectory 14[b]	12_21_2020_ec_hallways_run4	102
Trajectory 15	12_21_2020_irl_lab_run0	100
Trajectory 16[a]	12_21_2020_irl_lab_run1	87
Trajectory 17	12_21_2020_irl_lab_run3	90
Trajectory 18	12_21_2020_irl_lab_run4	95
Trajectory 19	2_23_2021_edgar_classroom_run0	188

[a] Used previously in [19].
[b] Only used for the optimization in Sec. VI-A.

TABLE II
Radar Sensor parameters [43].

Parameter	Value	Parameter	Value
Frame rate	10 Hz	TX antennas	3
Frequency	77 GHz	Waveform	FMCW
Max range	12 m	RX antennas	4
Resolution	0.125 m	Azimuth resolution	11.3°
Doppler velocity res.	0.04 m/s	Data rate	63 Mbps
Max Doppler velocity	2.56 m/s	Elevation resolution	45°

100 generations or a change in score of less than a 1×10^{-6} over the past ten generations [56]. The algorithm leveraged a simulated binary crossover (SBX) and binary tournament selection. SBX generates two offspring by sampling around the parent solutions [57], while binary tournament selection selects two solutions at random and compares their scores. The solution with the better score is selected as a parent, ensuring the pool of parents has a better average score than the average population [58]. We used polynomial mutation to modify the gene of an offspring by generating a new value from a polynomial distribution centred around the solution [59]. A mutation rate of 90% was used to ensure a diverse population and that the optimization covered a large solution space. It also reduces the chance of premature convergence due to tournament selection [58], at the expense of increased optimization time.

VI. Experimental Validation

The proposed methods are compared against three state-of-the-art "instantaneous" ego-motion estimation methods: 1) standard RANSAC estimation introduced in [12] further known as RANSAC, 2) the method implemented in [22] that uses MLESAC estimation and ODR further known as MLESAC, and 3) a novel version of 2) using a first-in-first-out sliding window, further known as SW-MLESAC.

The results for this section were generated using the ColoRadar dataset [43]. ColoRadar is a multi-sensor dataset spanning 6 locations and over 13 kilometres. In this paper, all trajectories selected were captured by a user walking though the environment with the handheld sensing platform shown in Fig. 6 A). As the user walked, the platform was moved so that it contained full 6-DOF motion. This often involved pitching and rolling the platform intentionally in addition to raising, lowering, and translating horizontally while yawing to simulate micro aerial vehicles moving along 6-DOF trajectories [43]. In total, 19 trajectories were selected, with 18 used for experimental validation, and one used to optimize the methods under test (MUT). The exact trajectories used are detailed in Table I, with trajectories 1-5 shown in Fig. 6 B) generated in a large open underground mine, trajectories 6-9, and 19 generated in a narrow underground mine passageways, trajectories 10-14 shown in Fig. 6 C) generated in 4 different indoor hallway environments, and trajectories 15-18 shown in Fig. 6 D) generated from 4 distinct paths of an indoor room.

For the purposes of the investigation, only the single chip radar sensor (TI AWR1843BOOST-EVM) and the IMU (Lord Microstrain 3DM-GX5-15) are considered. Additionally, a static offset is applied to the elevation angle of the radar point clouds. The methods within this paper make no claim to deal with radar bias. The scope of this paper is the filtering of radar point clouds for ego-motion estimation. Methods of removing radar bias are presented in [25]. The radar sensor parameters are detailed in Table II. It is important to note that "resolution" for radar sensors refers to the required separation between targets to distinguish them in the sensor's output. The orientation of the platform is provided by the IMU.

Before the MUT were experimentally validated for 3-DOF motion & 6-DOF motion, their hyperparameters were optimized using trajectory 14, see Section VI-A. Post optimization,

TABLE III
HYPERPARAMETERS IN 3-DOF AND 6-DOF MOTION VALIDATION.

	Parameter	TEMPSAC	TWLSQ	RANSAC
3-DOF	Fixed forgetting factor (λ)	0.1449	0.4845	N/A
	Sliding window length (m)	2	3	N/A
	Maximum iterations (k)	398	972	1350
	Inlier threshold (ε)	14.05	0.00382	13.99
	Min. number of inliers (ρ)	22	31	25
6-DOF	Fixed forgetting factor (λ)	0.7160	0.1576	N/A
	Sliding window length (m)	2	2	N/A
	Maximum iterations (k)	1494	68	608
	Inlier threshold (ε)	14.84	14.39	0.00619
	Min. number of inliers (ρ)	24	24	31

TABLE IV
TIMING STATISTICS (MAX-MIN/AVERAGE) OF 3-DOF AND 6-DOF MUT OVER TRAJECTORY 19 (187 SECONDS LONG).

	Method	Per Estimate (ms)		Trajectory (s)	
		min-max	ave.	min-max	ave.
3-DOF	TEMPSAC	58-186	74	117-122	120
	TWLSQ	26-65	35	57-59	58
	RANSAC	38-182	45	72-74	73
	MLESAC	1-182	15	24-26	25
	SW-MLESAC	2-284	49	79-87	83
6-DOF	TEMPSAC	130-282	166	263-271	267
	TWLSQ	4-31	6	9.7-10	9.9
	RANSAC	18-77	24	39-40	39.7
	MLESAC	5-425	84	130-142	138
	SW-MLESAC	12-800	199	316-334	326

the proposed methods were validated across 18 trajectories in 3-DOF motion and 6-DOF motion scenarios respectively, in Section VI-B and Section VI-C. Due to the non-deterministic nature of the MUT, 100 evaluations per trajectory are performed to provide an accurate representation of performance.

A. Hyperparameter Optimization Results

RANSAC, TEMPSAC, and TWLSQ were optimized separately using (32) as the cost function and the GA implemented in Section V-A. Trajectory 14 was selected as the hyperparameter optimization dataset since a hallway environment should generalize well to other indoor environments [11], and it was the trajectory used in [19]. By optimizing over the same trajectory, the results here can be directly compared to [19].

In total 18,600 distinct hyperparameter combinations were tested across all 6 optimizations. The parameters that minimized (32) are shown in Table III. The fixed forgetting factor (λ) varied between favouring more recent measurements ($\lambda \to 0$) to more evenly weighting measurements ($\lambda \to 1$). Additionally, λ had an impact on the maximum number of iterations k. Instances where $\lambda \to 1$ resulted in double or more iterations required. One reason for this is because as $\lambda \to 1$, the measurements weights approach equal, increasing the number of equally likely solutions. With only a finite number of hypotheses able to be tested, this larger search area increases the estimation uncertainty. The increase in uncertainty and its accompanying error can be offset by searching more of the area (i.e., a larger k). For the sliding window length, the optimization found $m = 2$, as expected due to the radar sensor's low frame rate. The larger the sliding window, the less the measurements correlate. As measurements correlate less, fewer inliers overlap and instead start to act like noise, decreasing estimation accuracy. The inlier threshold was found to be comparatively large ($>> 1$) or small ($<< 1$). This is due to ε being the squared error threshold and the square-root of a number larger than one is a smaller number, but the square-root of a number smaller than one is a larger number. The minimum number of inliers (ρ) was similar between all the methods; the largest difference was seven inliers. On average, $\rho = 27$, with the given measurement from the single chip radar having 140 samples. Therefore, for the velocity estimate must capture at least 19% of the measurement as inliers.

MLESAC and SW-MLESAC were not optimized since they have no hyperparameters. To ensure a fair comparison, the sliding window length for SW-MLESAC was the same as the majority of other sliding window methods ($m = 2$).

The hyperparameters had a major impact on the computation time of the MUT, as shown in Table IV. The lower the number of iterations k, the faster the execution time. Furthermore, as the number of samples n being estimated increases, the execution time increases by $O(n^2)$. TEMPSAC's slow performance is a result of the sample selection's implementation. MLESAC and SW-MLESAC's wide range of per estimate execution times is due to the dynamic number of iterations required for the model to converge [22]. All methods show online performance, except 6-DOF SW-MLESAC. The increased number of samples due to the sliding window doubled the execution time compared to 6-DOF MLESAC.

B. Validation for Three Degrees-of-Freedom Motion

The ColoRadar dataset contains 6-DOF motion and measurements. In order to evaluate the five MUT in a 3-DOF scenario the dataset was projected onto the xy-plane. The MUT are compared based on their root-mean-squared absolute pose error (*RMS-APE*) as defined in [54] and the average root-mean-squared absolute pose error over the 100 evaluations (*AAPE*). Fig. 7 shows the estimated xy-path (A, B, C) from Trajectory 3 (left), Trajectory 8 (centre), and Trajectory 16 (right), the *RMS-APE* error over time from a single evaluation (D, E, F), and the *RMS-APE* distribution for the 100 evaluations (G, H, I). These trajectories were previously evaluated in [19]. The methods proposed within this paper generated narrower *RMS-APE* distributions compared to the implementation in [19].

The xy-paths in Fig. 7 (A, B) show that RANSAC performed the worst, consistently underestimating the ego-motion, confirming the findings in [19]. The *RMS-APE* and the distribution box plots in Fig. 7 (D, G, E, H) show that RANSAC has the largest peak *RMS-APE* (15 m), the largest median *RMS-APE* (8.8 m), and the widest error distribution.

RANSAC's poor *RMS-APE* performance is caused by its reliance on a single measurement to estimate ego-motion. When this measurement has a high outlier to inlier ratio, RANSAC is prone to fitting the velocity model to outliers [12]. This can be seen in the *RMS-APE* plot in Fig. 7 (E) at 65 seconds. At this time, the errors in RANSAC's estimate

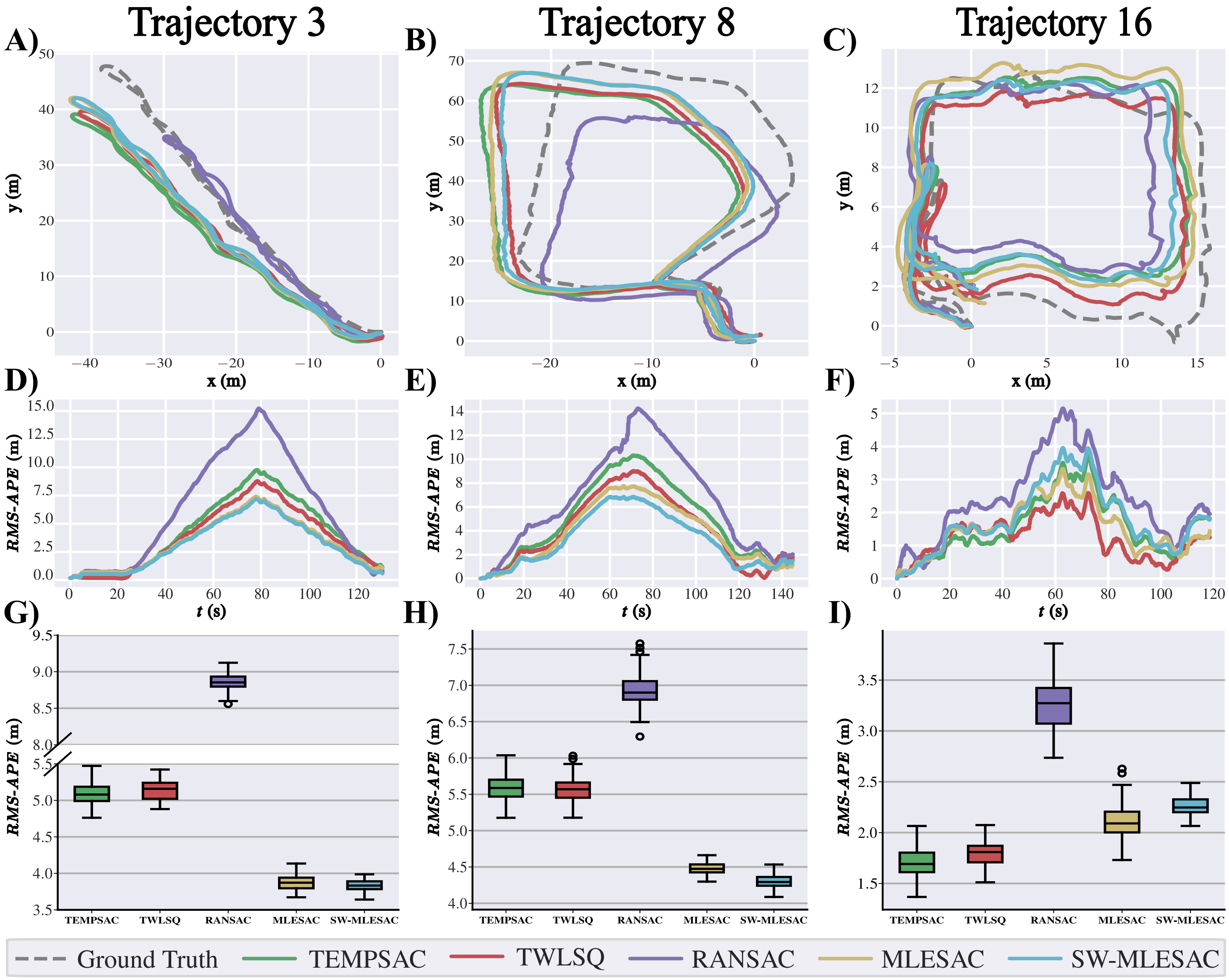

Fig. 7. Results from TEMPSAC (green), TWLSQ (red), RANSAC (purple), MLESAC (gold), and SW-MLESAC (blue) for trajectory 3 (left column), 8 (centre column), and 16 (right column). The rows show the xy-path (A, B, C), absolute pose error over time (D, E, F), and error distribution over the 100 evaluations as box plots (G, H, I). On average, SW-MLESAC outperforms all other methods.

are more pronounced than other MUT, varying up and down over a short period of time before experiencing a large spike in error of 3 m. MLESAC also estimates ego-motion using a single measurement. However, it is the second-best performing MUT, as shown in Fig. 7 (G, I). This is because MLESAC maximizes the log-likelihood of an estimate's existence, leading to estimates with lower error.

A sliding window provides a marked improvement in accuracy for TWLSQ and TEMPSAC over RANSAC, and also improves the estimates provided by SW-MLESAC compared to those from MLESAC. Use of sliding window increases the inlier density, thereby producing more reliable estimates.

Table V presents the *AAPE* for all 18 trajectories, highlighting the best performing method in bold. The proposed methods reduced *AAPE* by up to a 27% in trajectories 3 and 16 compared to [19]. However, all methods experienced an increase in *AAPE* for trajectory 8. The results from Table V show that the proposed methods outperformed RANSAC over all trajectories. While TEMPSAC outperformed its implementation in [19], it underperformed TWLSQ in all but trajectory 17, where it achieved a 0.08 m improvement in *AAPE*. TWLSQ outperformed TEMPSAC and its implementation in [19] due to the reformulated temporal weights (25). In [19] the weights were normalized, whereas in (25) they are not. TWLSQ outperformed all MUT over 2 trajectories by 4.3% on average over the next best method. MLESAC outperformed all MUT over 4 trajectories by 9% on average over the next best method. SW-MLESAC performed the best overall, outperform all other methods in 66.7% of trajectories, and outperforming MLESAC (the next best method) by up to 15%.

C. Validation for Six Degrees-of-Freedom Motion

The ColoRadar dataset containing full 6-DOF motion is used for this section. As before, to evaluate the five MUT, the root-mean-squared absolute pose error (*RMS-APE*) is used, as well as the averaged absolute pose error (*AAPE*) for the 100

TABLE V

Comparison of average absolute pose error (*AAPE*) for 3-DOF and 6-DOF motion estimation. *AAPE* is in metres.

Tr. #	3-DOF TEMPSAC	3-DOF TWLSQ	3-DOF RANSAC	3-DOF MLESAC	3-DOF SW MLESAC	6-DOF TEMPSAC	6-DOF TWLSQ	6-DOF RANSAC	6-DOF MLESAC	6-DOF SW MLESAC
1	12.03	11.80	18.74	8.64	**8.35**	8.48	**8.37**	11.35	10.08	9.70
2	3.99	3.46	8.95	2.85	**2.70**	5.68	5.99	6.31	3.18	**3.16**
3	5.50	5.15	8.86	3.88	**3.83**	3.85	**3.84**	5.20	4.02	3.97
4	9.75	9.09	15.03	6.76	**6.48**	5.67	**5.62**	15.32	8.57	7.87
5	14.61	13.86	25.85	9.67	**9.57**	11.58	**11.57**	27.37	12.27	11.8
6	7.19	6.95	9.30	5.29	**4.82**	**3.12**	3.13	11.64	6.49	5.56
7	5.06	4.85	5.13	4.03	**3.42**	4.27	**4.26**	6.13	4.86	4.38
8	6.04	5.58	6.95	4.48	**4.31**	3.66	**3.55**	8.38	5.48	5.21
9	5.71	5.38	7.32	3.87	**3.37**	3.26	**3.24**	7.72	4.43	3.76
10	3.45	3.01	3.71	2.48	**2.47**	**3.56**	3.64	5.48	3.97	3.91
11	7.09	6.38	8.40	5.53	**5.50**	6.69	6.73	6.37	5.90	**5.88**
12	5.05	4.68	6.62	**3.95**	5.28	5.29	5.41	**3.21**	4.13	4.20
13	6.10	5.37	6.48	**4.26**	4.81	7.60	7.56	9.65	5.46	**5.36**
15	1.21	**1.14**	2.71	1.17	1.49	3.23	3.29	2.453	1.53	**1.49**
16	1.82	**1.71**	3.29	2.13	2.26	4.40	4.44	**1.93**	2.32	2.14
17	0.96	1.04	1.218	0.98	**0.75**	1.99	2.13	1.82	1.65	**1.50**
18	1.80	1.78	2.539	**1.66**	1.87	4.19	4.08	**1.60**	2.09	2.07
19	5.09	5.24	7.29	**4.75**	4.86	**4.49**	4.54	7.82	5.48	5.09

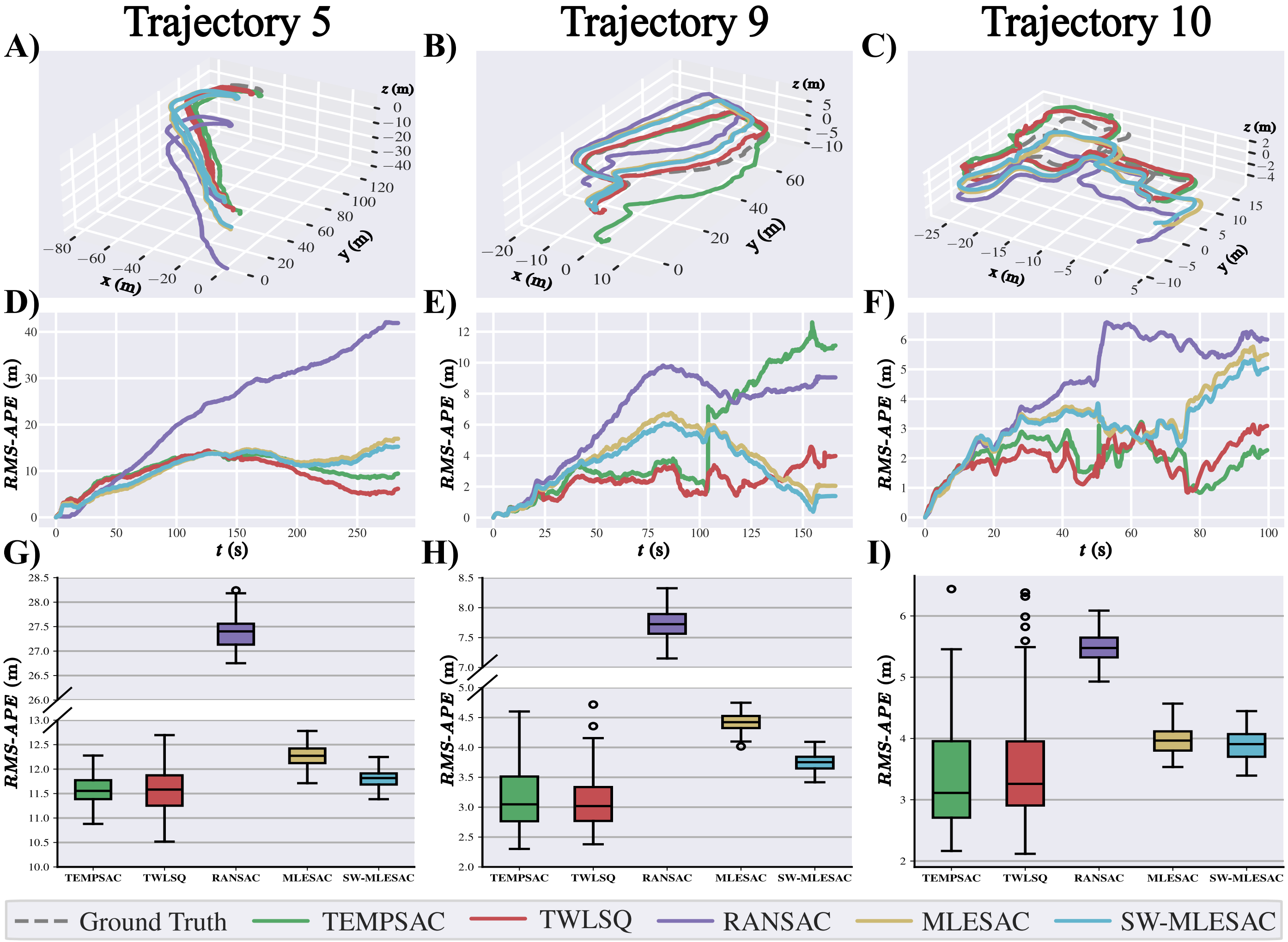

Fig. 8. Results from TEMPSAC (green), TWLSQ (red), RANSAC (purple), MLESAC (gold), and SW-MLESAC (blue) for trajectories 5 (left column), 9 (center column), and 10 (right column). The rows show the xyz-path (A, B, C), absolute pose error over time (D, E, F), and error distribution over the 100 evaluations as box plots (G, H, I). TWLSQ performs best on average but shows higher variability than MLESAC and SW-MLESAC.

TABLE VI
EXTERNAL COMPARISON OF ABSOLUTE POSE ERROR (*RMS-APE*) FOR 6-DOF MOTION. *RMS-APE* IS IN METRES.

Method	Trajectory 10	Trajectory 19
TEMPSAC	**3.56**	4.49
TWLSQ	3.64	4.54
RANSAC	5.48	7.82
MLESAC	3.97	5.48
SW-MLESAC	3.91	5.09
EKF-RIO [24]	9.352	10.99
Ours-P2D [24]	7.62	8.382
Ours-D2M [24]	12.65	14.05
Ours [24]	5.223	8.340
EKF-RIO [23]	N/A	5.05
iRIO [23]	N/A	2.377
iRIOM [23]	N/A	**0.916**

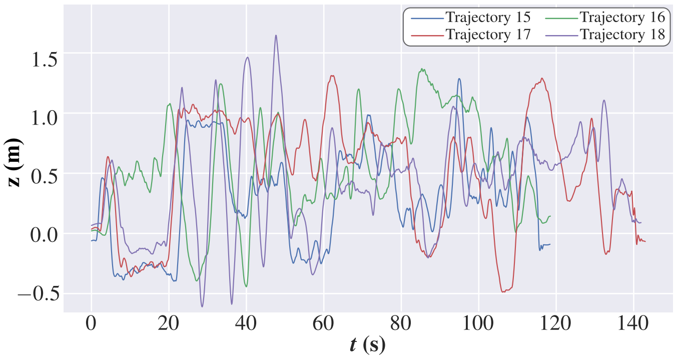

Fig. 9. Ground truth motion along the $^s\mathbf{z}$-axis for trajectories 15-18. All trajectories have erratic z-motion.

evaluations. Fig. 8 summarizes the results from trajectory 5 (left), trajectory 9 (centre), and trajectory 10 (right). It shows the estimated xyz-path (A, B, C), the *RMS-APE* (D, E, F), and the *RMS-APE* distribution box plots (G, H, I). Trajectories 5, 9, and 10 were selected to evaluate the MUT's performance for their unique and diverse locations. Trajectory 5 is a large open underground mine environment with rough unstructured walls providing many surfaces for multi-path reflections. Trajectory 9 is a narrow underground mine passageway with similarly rough and unstructured walls. Trajectory 10 is a structured indoor hallway environment in a different location than trajectory 14 used in Section VI-A.

As seen in Fig. 8 (A, C), RANSAC performs the worst when estimating motion along z with a final pose 10 m lower than the ground truth. RANSAC's performance was expected given the results from Section VI-B and [19]. However, Fig. 8 (B) shows that TEMPSAC performed the worst with 5 m lower displacement along z than the other MUT. The *RMS-APE* results in Fig. 8 E) show that TEMPSAC also has the highest peak *RMS-APE* at 13 m. The cause for this unexpectedly poor performance can be seen at around 100 seconds. At this time TEMPSAC makes multiple incorrect motion estimates resulting in a near instantaneous increase in *RMS-APE* of 5 m. These incorrect estimations were caused by TEMPSAC fitting the velocity model to outlier samples over consecutive time steps. Due to TEMPSAC's sampling strategy (detailed in (27)), both inliers and outliers from newer measurements are favoured for sampling and ego-motion estimation. Therefore, when the most recent measurements have a high ratio of outliers to inliers, the ego-motion estimate is more likely to fit those outliers, potentially causing large changes in *RMS-APE*. In Fig. 8 (E), at 100 seconds all MUT excluding RANSAC experienced an increase in *RMS-APE*, and before 100 seconds TEMPSAC had similar performance to TWLSQ.

The error distributions in Fig. 8 (G, I) shows that RANSAC performs the worst, having the largest median *RMS-APE* in all 3 trajectories. SW-MLESAC consistently outperforms MLESAC. TEMPSAC and TWLSQ perform similarly across all trajectories, with lower median *RMS-APE* than MLESAC and SW-MLESAC. However, TEMPSAC and TWLSQ have a wider distribution. The distribution is wide enough that nearly 100% of MLESAC and SW-MLESAC's evaluations fall into the largest 50% of TEMPSAC and TWLSQ *RMS-APE* results. This can be seen by the amount of overlap between the box plots in Fig. 8 (G, H, I). Meaning that while TEMPSAC and TWLSQ produce more accurate estimates for a given evaluation, MLESAC and SW-MLESAC have a lower variance over multiple evaluations. This suggests that future work on the hyperparameter optimization's cost function (32), is required to minimize the error distribution's spread (σ_{RPE}).

Table V shows the *AAPE* for all 18 trajectories, with the best performing method shown in bold. As expected given the results in Fig. 8, the proposed methods outperform RANSAC in all trajectories other than 15-18. Trajectories 15-18 are different paths (100 m or less) of the room shown in Fig. 6. The room is very cluttered with desks, lab equipment, and other obstacles, and the trajectories' ground truth have highly variable z-motion over short periods of time, as seen in Fig. 9. This unpredictability causes measurements to overlap less and the proposed methods' to generate worse motion estimates. The proposed methods performed very similar, with TWLSQ showing an average improvement of 0.7% over TEMPSAC across all trajectories. Additionally, the proposed methods outperformed all other methods across 55% of trajectories evaluated. SW-MLESAC performed the next best, outperforming all other methods in 27.8% of the trajectories.

Table VI compares the MUT results over trajectory 10 and 19 with other reported methods, i.e., EKF-RIO, Ours, Ours-P2D, and Ours-D2M presented in [24] and EKF-RIO, iRIO, and iRIOM presented in [23]. The *RMS-APE* values corresponding to the MUT are their *AAPE* and the other values are reused from [23], [24]. Both papers presented an implementation of EKF-RIO [60], the best-performing open-source 4D radar inertial odometry software [23]. Ours is a hybrid *direct / indirect* approach that incorporates a measurement uncertainties model. Ours-P2D, and Ours-D2M are similar to Ours but use different registration methods and uncertainty residuals (based on LIDAR, and mmWave, respectively) [24]. iRIO (imaging radar inertial odometry) [23], estimates ego-motion using an extended Kalman filter to fuse IMU motion estimates, radar image registration, and graduated non-convexity estimates. iRIOM is identical to iRIO, except the final ego-motion estimate is used to build a map and loop closure is performed to improve accuracy.

Over trajectory 10, out of the methods presented in [24]

Ours performs the best, and outperformed RANSAC by 4.8%. However, it performed worse than all other MUT, underperforming TEMPSAC (the best MUT) by 31.8%. Over trajectory 19, iRIOM performed the best with sub-metre accuracy thanks to loop closure. The second-best method iRIO was a more direct comparison to the MUT since there is no map making. While its error is almost half of TEMPSAC's (the best MUT), both iRIO and iRIOM are using an imaging radar that is over 300% more expensive than the radar the MUT use.

VII. Discussion

The 3-DOF validation shows that the proposed methods provide up to a 27% improvement and generate narrower *AAPE* distributions than their previous implementation. These results suggest that the optimization presented in Section V succeeded in selecting hyperparameters which minimized the *RPE* and its variance. However, this highlights a potential trade-off where the higher precision may result in a higher median *RMS-APE*, as was the case with trajectory 8. Comparing all the MUT, SW-MLESAC performed the best overall, outperforming all other methods in 66.7% of trajectories, and outperforming MLESAC, the next best method, by up to 15%.

The results from the 6-DOF validation shows more variability over all MUT due to the increase in DOF of the estimation. The best method per trajectory differs significantly from the 3-DOF validation, while the corresponding *AAPE* is very similar. On average, the best performing 6-DOF estimate is 3.9% worse than the corresponding 3-DOF estimate. Directly comparing implementations of SW-MLESAC (the best performing 3-DOF MUT), the 3-DOF version outperforms the 6-DOF version in 61% of trajectories. The reason for this difference in performance, besides the increase in DOF, is the low elevation resolution of the mmWave radar. The elevation resolution results in objects which are far apart and have different Doppler velocities being blurred together [61], [62]. With an elevation resolution of $45°$ and a distance from the sensor of 10 m, any objects less than 10 m apart from one another along the ${}^{s}\mathbf{z}$-axis will be indistinguishable.

Overall, when presented with 6-DOF motion conditions, the proposed methods outperform all MUT in 55% of trajectories. Of those trajectories, the majority are from an open underground mine (trajectories 1-5) and a narrow mine passageway (trajectories 6-9, and 19). Yet, in Section V the GA was trained on data from a structured indoor environment, suggesting some generalization. Overall, TWLSQ performed the best, showing a 0.7% improvement on average over TEMPSAC. The improvement can be attributed to TWLSQ's uniform sampling strategy, which ensures that samples from high density regions within the sliding window are more likely to parameterize the velocity model. These high density regions correspond to inliers because over short periods of time all inliers are a result of the same platform motion, i.e., are temporally consistent. Concurrently, the likelihood of generating a model using outliers is reduced, since outliers in consecutive measurements have no temporal relation with one another. These findings are consistent with [19] and the principles of LIDAR scan matching [21]. Further evidence of these effects can be seen by comparing the results from SW-MLESAC to MLESAC. The only difference between the two methods is the use of a sliding window. Yet, SW-MLESAC outperforms MLESAC in 99.94% of trajectories, showing that temporally consistent mmWave measurements can constrain the model estimation, thereby improving performance. These results open the door for future research on similar techniques used in different environments and other mmWave-based applications such as target tracking and environmental modelling.

Another area of potential improvement is to apply similar techniques to scenarios with multiple moving objects. Traditional "instantaneous" ego-motion estimation is more likely to fail in these scenarios due to their RANSAC framework [12]. As the number of samples from moving objects with similar velocities increases, the velocity model will be parameterized using those samples rather than the samples of stationary objects. Due to the problems similar nature with the one addressed in this paper, applying a similar technique could make the ego-motion estimation more robust, but further research is required.

While the results presented in this paper also suggest a level of generalization of the proposed optimization method, future work is required to strengthen this claim. Further, to the best of the authors' knowledge, no other research has been conducted examining the generalization of the instantaneous ego-motion velocity model to different environments. Future work in both these areas would pave the way for more robust and reliable applications of mmWave odometry that leverages temporal relationships in Doppler velocity measurements.

VIII. Conclusion

Millimetre wave radar sensors are becoming increasingly popular for ego-motion estimation because they are more robust to environmental factors and do not require mapping for motion estimation. However, it is challenging to filter out the high number of samples caused by specular (mirror-like) reflections, diffraction, and multi-path reflections. Yet, traditional methods of estimating ego-motion using Doppler velocity only use a single measurement cloud [12]–[14], [17], [22]. These methods ignore the time-dependency of successive measurements, a dependency that is critical to other ego-motion techniques like LIDAR scan matching.

This paper presents two methods of estimating 6-DOF ego-motion, building upon [19]. The proposed methods use a temporally weighted sliding window to leverage the temporal relationship between sparse measurements. Experimental results show that this relationship improves estimation accuracy when compared to using a single measurement, as evidenced by the 5.5% improvement of SW-MLESAC over MLESAC, where the only difference between the two implementations of [22] is the sliding window. The proposed method were also compared with the state-of-the-art methods presented in [22]–[24], outperforming all other methods in 53% of trajectories tested. In this comparison, iRIO and iRIOM performed the best but use a higher quality radar (360% more expensive than the one used in this paper). However, the proposed methods performed better than all other methods while having a simpler architecture and a cheaper sensor. Finally, the experimental validation of [19] was extended to include 18 unique trajectories along with a comparison against two additional state-of-the-art ego-motion estimation methods, confirming that the proposed sliding winding improves motion estimation for both 3-DOF and 6-DOF indoor environments.

References

[1] J. Leonard and H. Durrant, "Simultaneous map building and localization for an autonomous mobile robot," *Int. Conf. on Intell. Robots and Syst.*, pp. 1442–1447, 1991.

[2] Z. Zhang *et al.*, "MRPT: Millimeter-wave radar-based pedestrian trajectory tracking for autonomous urban driving," *IEEE Trans. on Instrum. and Meas.*, vol. 71, pp. 1–17, 2022.

[3] H. Xu, X. Zhang, J. He, Y. Yu, and Y. Cheng, "Real-time volumetric perception for unmanned surface vehicles through fusion of radar and camera," *IEEE Trans. on Instrum. and Meas.*, vol. 73, pp. 1–12, 2024.

[4] L.-F. Shi, Y.-F. Lv, W. Yin, and Y. Shi, "Autonomous multiframe point cloud fusion method for mmwave radar," *IEEE Trans. on Instrum. and Meas.*, vol. 72, pp. 1–8, 2023.

[5] C. Iovescu and S. Rao, "The fundamentals of millimeter wave sensors," *Texas Instruments*, pp. 1–8, 2017.

[6] H. Yin, P. X. Liu, and M. Zheng, "Ego-motion estimation with stereo cameras using efficient 3D–2D edge correspondences," *IEEE Trans. on Instrum. and Meas.*, vol. 71, pp. 1–11, 2022.

[7] K. Harlow, H. Jang, T. D. Barfoot, A. Kim, and C. Heckman, "A new wave in robotics: Survey on recent mmwave radar applications in robotics," *IEEE Trans. Robot.*, 2024.

[8] A. Kramer *et al.*, "Radar-inertial ego-velocity estimation for visually degraded environments," in *Intl. Conf. on Robot. and Automat.*, 2020, pp. 5739–5746.

[9] X. Lu *et al.*, "milliEgo: Single-chip mmwave radar aided egomotion estimation via deep sensor fusion," in *18th Conf. on Embed. Netw. Sensor Syst.*, 2020, pp. 109–122.

[10] G. MacCartney, S. Deng, S. Sun, and T. Rappaport, "Millimeter-Wave Human Blockage at 73 GHz with a Simple Double Knife-Edge Diffraction Model and Extension for Directional Antennas," in *IEEE 84th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2016, pp. 1–6.

[11] C. Lu *et al.*, "See through smoke: Robust indoor mapping with low-cost mmwave radar," in *ACM MobiSys 2020*, 2020, pp. 14–27.

[12] D. Kellner *et al.*, "Instantaneous ego-motion estimation using doppler radar," in *16th ITSC*, 2013, pp. 869–874.

[13] D. Kellner *et al.*, "Instantaneous ego-motion estimation using multiple doppler radars," in *Inter. Conf. on Robot. and Aut.*, 2014, pp. 1592–1597.

[14] S. Kwon *et al.*, "Radar sensor-based ego-motion estimation and indoor environment mapping," *IEEE Sensors J.*, 2023.

[15] M. Fischler and R. Bolles, "RANSAC: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[16] S.-W. Yang and C.-C. Wang, "Multiple-model RANSAC for ego-motion estimation in highly dynamic environments," in *Intl. Conf. on Robot. and Automat.*, 2009, pp. 3531–3538.

[17] Y. S. Park, Y. S. Shin, J. Kim, and A. Kim, "3D ego-Motion Estimation Using low-Cost mmWave Radars via Radar Velocity Factor for Pose-Graph SLAM," *IEEE Robot. and Automat. Lett.*, vol. 6, no. 4, pp. 7691–7698, Oct. 2021.

[18] Z. Zeng *et al.*, "Joint velocity ambiguity resolution and ego-motion estimation method for mmwave radar," *IEEE Rob. Automat. Lett.*, 2023.

[19] S. Lovett, K. MacWilliams, S. Rajan, and C. Rossa, "Enhancing doppler ego-motion estimation: A temporally weighted approach to RANSAC," in *2024 IEEE Sensors Applications Symposium (SAS)*, 2024, pp. 1–6.

[20] F. Pomerleau *et al.*, "A review of point cloud registration algorithms for mobile robotics," *Found. and Trends in Robot.*, vol. 4, no. 1, pp. 1–104, 2015.

[21] S. Lovett *et al.*, "Level plane slam: Out-of-plane motion compensation in a globally stabilized coordinate frame for 2D SLAM," in *Int. Conf. on Syst. Man and Cybern.*, 2023, pp. 3355–3360.

[22] C. C. Stahoviak, *An Instantaneous 3D Ego-Velocity Measurement Algorithm for Frequency Modulated Continuous Wave (FMCW) Doppler Radar Data*, 2019.

[23] Y. Zhuang, S. Member, B. Wang, J. Huai, and M. Li, "4D iRIOM: 4D Imaging Radar Inertial Odometry and Mapping," *IEEE Robot. and Automat. Lett.*, vol. 8, no. 6, 2023.

[24] Y. Xu, Q. Huang, S. Shen, and H. Yin, "Incorporating point uncertainty in radar slam," *IEEE Robotics and Automation Letters*, vol. 10, no. 3, pp. 2168–2175, 2025.

[25] S. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Sign. Process. Mag.*, vol. 34, no. 2, pp. 22–35, 2017.

[26] M. Yang *et al.*, "Sensors and sensor fusion methodologies for indoor odometry: A review," *Polymers*, vol. 14, no. 10, 2022.

[27] A. Goshtasby, *Image registration: Principles, tools and methods*. Springer Sci. & Bus. Media, 2012.

[28] G. A. Reis, J. Chang, N. Vachharajani, R. Rangan, and D. I. August, "Swift: Software implemented fault tolerance," in *Int. symposium on Code generation and optimization*, IEEE, 2005, pp. 243–254.

[29] D. G. Viswanathan, "Features from accelerated segment test (fast)," in *Proc. 10th workshop on image analysis for multimedia interactive services*, 2009, pp. 6–8.

[30] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "Dbscan revisited, revisited: Why and how you should (still) use dbscan," *ACM Trans. on Database Syst. (TODS)*, vol. 42, no. 3, pp. 1–21, 2017.

[31] M. Donoser and H. Bischof, "Efficient maximally stable extremal region (mser) tracking," in *2006 IEEE Com-*

put. Soc. Conf. on Comput. Vis. and Pattern Recognit. (CVPR'06), Ieee, vol. 1, 2006, pp. 553–560.

[32] J. Callmer, D. Törnqvist, F. Gustafsson, H. Svensson, and P. Carlbom, "Radar slam using visual features," *EURASIP J.on Advances in Signal Processing*, vol. 2011, no. 1, p. 71, Sep. 2011.

[33] F. Schuster, C. G. Keller, M. Rapp, M. Haueis, and C. Curio, "Landmark based radar slam using graph optimization," in *IEEE 19th Intl. Conf. Intell. Transp. Syst. (ITSC)*, 2016, pp. 2559–2564.

[34] K. Werber, J. Klappstein, J. Dickmann, and C. Waldschmidt, "Interesting areas in radar gridmaps for vehicle self-localization," in *IEEE MTT-S Intl. Conf. on Microw. for Intell. Mobility (ICMIM)*, 2016, pp. 1–4.

[35] S. H. Cen and P. Newman, "Precise ego-motion estimation with millimeter-wave radar under diverse and challenging conditions," in *Intl. Conf. on Robot. and Automat. (ICRA)*, 2018, pp. 6045–6052.

[36] R. Aldera, D. D. Martini, M. Gadd, and P. Newman, "Fast radar motion estimation with a learnt focus of attention using weak supervision," in *Intl. Conf. on Robot. and Automat. (ICRA)*, 2019, pp. 1190–1196.

[37] P. Checchin, F. Gérossier, C. Blanc, R. Chapuis, and L. Trassoudaine, "Radar scan matching slam using the fourier-mellin transform," in *Field and Service Robotics*, Berlin, Heidelberg: Springer, 2010, pp. 151–161.

[38] B. Amjad *et al.*, "Deep learning approach for optimal localization using an mm-wave sensor," *IEEE Trans. on Instrum. and Meas.*, vol. 72, pp. 1–15, 2023.

[39] J. Komorowski, M. Wysoczanska, and T. Trzcinski, "Large-scale topological radar localization using learned descriptors," in *Intl. Conf. on Neural Information Processing*, Springer, 2021, pp. 451–462.

[40] Y. Almalioglu *et al.*, "Milli-RIO: Ego-motion estimation with low-cost mmwave radar," *IEEE Sensors J.*, vol. 21, no. 3, pp. 3314–3323, 2020.

[41] A. Kramer, C. Stahoviak, A. Santamaria-Navarro, A. A. Agha-Mohammadi, and C. Heckman, "Radar-Inertial Ego-Velocity Estimation for Visually Degraded Environments," *Intl. Conf. Robot. and Automat.*, pp. 5739–5746, May 2020, Publisher: Institute of Electrical and Electronics Engineers Inc. ISBN: 9781728173955.

[42] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated Non-Convexity for Robust Spatial Perception: From Non-Minimal Solvers to Global Outlier Rejection," *IEEE Robot. and Automat. Lett.*, vol. PP, pp. 1–1, Jan. 2020.

[43] A. Kramer et al, "Coloradar: The direct 3D millimeter wave radar dataset," *The Int. J. Robot. Res.*, vol. 41, no. 4, pp. 351–360, 2022.

[44] V. Kubelka, E. Fritz, and M. Magnusson, "Do we need scan-matching in radar odometry?" In *2024 IEEE Intl. Conf. Robot. and Automat.*, 2024, pp. 13 710–13 716.

[45] J. Zhang and S. Singh, "Loam : Lidar odometry and mapping in real-time," *Robot.: Sci. and Syst. Conf. (RSS)*, pp. 109–111, Jan. 2014.

[46] P. T. Boggs and J. R. Donaldson, *Orthogonal distance regression:* en, Jan. 1989.

[47] M. Waine *et al.*, "3D shape visualization of curved needles in tissue from 2D ultrasound images using RANSAC," in *Intl. Conf. on Robot. and Automat.*, 2015, pp. 4723–4728.

[48] R. Raguram, J. Frahm, and M. Pollefeys, "A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus," in *Comput. Vis. – ECCV 2008*, Springer Berlin Heidelberg, 2008, pp. 500–513.

[49] D. Bodenham *et al.*, "Continuous monitoring for changepoints in data streams using adaptive estimation," *Statist. and Comput.*, vol. 27, no. 5, pp. 1257–1270, Sep. 2017.

[50] S. Katoch, S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools and Applications*, vol. 80, no. 5, pp. 8091–8126, Feb. 2021.

[51] C. Yang, "Parallel–series multiobjective genetic algorithm for optimal tests selection with multiple constraints," *IEEE Trans. on Instrum. and Meas.*, vol. 67, no. 8, pp. 1859–1876, 2018.

[52] A. Hast, J. Nysjö, and A. Marchetti, "Optimal RANSAC-towards a repeatable algorithm for finding the optimal set," *J. WSCG*, vol. 21, no. 1, 2013.

[53] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *Intl. Conf. on Intell. Robots and Syst.*, 2018, pp. 7244–7251.

[54] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, 2012, pp. 573–580.

[55] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Automat. robots*, vol. 4, pp. 333–349, 1997.

[56] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.

[57] K. Deb, K. Sindhya, and T. Okabe, "Self-adaptive simulated binary crossover for real-parameter optimization," in *Proc. 9th Annu. Conf. on Genetic and Evolutionary Computation*, ser. GECCO '07, London, England: Association for Comput. Machinery, 2007, pp. 1187–1194.

[58] B. Miller and D. Goldberg, "Genetic algorithms, tournament selection, and the effects of noise," *Complex Syst.*, vol. 9, 1995.

[59] K. Deb and D. Deb, "Analysing mutation schemes for real-parameter genetic algorithms," *Int. J. Artif. Intell. and Soft Comput.*, vol. 4, pp. 1–28, Feb. 2014.

[60] C. Doer and G. F. Trommer, "An ekf based approach to radar inertial odometry," in *2020 IEEE Intl. Conf. on Multisensor Fusion and Integration for Intell. Syst. (MFI)*, 2020, pp. 152–159.

[61] P. Li, P. Wang, K. Berntorp, and H. Liu, "Exploiting temporal relations on radar perception for autonomous driving," in *2022 IEEE/CVF Conf. on Comput. Vis. and Pattern Recognit. (CVPR)*, 2022, pp. 17 071–17 080.

[62] B.-S. Kim, Y. Jin, J. Lee, and S. Kim, “Fmcw radar estimation algorithm with high resolution and low complexity based on reduced search area,” *Sensors*, vol. 22, no. 3, 2022.

Samuel Lovett received his BEng in mechatronics engineering from Ontario Tech University, Oshawa, Canada, in 2022. He won second place in the capstone design competition for the design of a human-like collaborative robotic arm. He received his Master of Applied Science in Electrical and Computer Engineering from Carleton University, Ottawa, Canada, in 2025. He is the recipient of the Queen Elizabeth II Scholarship in Science and Technology for 2022-23. His research interests include localization and mapping of mobile platforms.

Kade MacWilliams received his bachelor’s in biomedical engineering at Carleton University, Ottawa, Canada in 2024. He is currently pursuing a Master of Applied Science in biomedical engineering at Carleton University, Ottawa, Canada. He is the recipient of the Queen Elizabeth II Scholarship in Science and Technology for 2023-24. His research interests include new technologies to assist surgeons in training for pediatric laparoscopic procedures.

Dr. Sreeraman Rajan received his bachelor degree in Electronics and Communications Engineering from Bharathiyar University, Coimbatore, India in 1987, his Master of Science in Electrical Engineering from Tulane University, New Orleans, USA, and his Ph.D. in Electrical and Computer Engineering from University of New Brunswick, Canada. He is a Professor in the Department of Systems and Computer Engineering at Carleton University, Ottawa, Canada, and a Tier II Canada Research Chair (Sensor Systems). His research interests include sensors and sensor systems, compressive sensing, signal processing (including biomedical signal processing, statistical signal processing, adaptive signal processing), machine learning, and pattern classification.

Dr. Carlos Rossa received his BEng and MSc degrees in Mechanical Engineering from the Ecole Nationale d’Ingénieurs de Metz, Metz, France, both in 2010, and earned his PhD degree in mechatronics and robotics from the Sorbonne Université (UPMC), Paris, France, in 2014 under the auspices of the Commissariat à l’Energie Atomique (CEA). He is an Associate Professor in the Department of Systems and Computer Engineering at Carleton University, Ottawa, Canada. His research interests include medical robotics, autonomous systems, robot-assisted surgery, image-guided surgery, medical imaging, sensors and instrumentation, haptic feedback, and human-robot interaction.