DICP: Doppler Iterative Closest Point Algorithm

Bruno Hexsel, Heethesh Vhavle and Yi Chen
Aeva, Inc

***Abstract*—In this paper, we present a novel algorithm for point cloud registration for range sensors capable of measuring per-return instantaneous radial velocity: Doppler ICP. Existing variants of ICP that solely rely on geometry or other features generally fail to estimate the motion of the sensor correctly in scenarios that have non-distinctive features and/or repetitive geometric structures such as hallways, tunnels, highways, and bridges. We propose a new Doppler velocity objective function that exploits the compatibility of each point's Doppler measurement and the sensor's current motion estimate. We jointly optimize the Doppler velocity objective function and the geometric objective function which sufficiently constrains the point cloud alignment problem even in feature-denied environments. Furthermore, the correspondence matches used for the alignment are improved by pruning away the points from dynamic targets which generally degrade the ICP solution. We evaluate our method on data collected from real sensors and from simulation. Our results show a significant performance improvement in terms of the registration accuracy with the added benefit of faster convergence guided by the Doppler velocity gradients.**

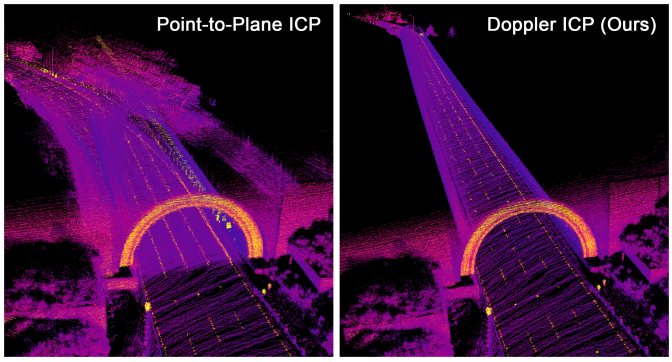

Fig. 1: Comparison of tunnel reconstructions using point-to-plane ICP (left) and Doppler ICP (right) methods with measurements collected by an FMCW LiDAR. Point-to-plane ICP fails in this degenerate case due to the lack of features in the scene whereas the Doppler ICP algorithm is able to reconstruct the scene with very low error.

I. Introduction

The problem of geometrically aligning two point clouds, also known as point cloud registration, has many applications in robotics, healthcare, and others [4]. In the past decades, methods for solving point cloud registration have been extensively researched and many algorithms have been developed for this purpose [5]. Among those, one popular algorithm is Iterative Closest Point (ICP) [9]. In ICP, two point clouds, defined as the source and the target point clouds, are matched by iteratively finding a transform that minimizes some distance metric between them. Generally speaking, registration methods based on ICP can lead to high-precision results but remain vulnerable in the presence of geometrically non-distinctive or repetitive environments, such as tunnels, highways, or bridges [1], as shown in Figure 1.

Despite these shortcomings, most rangefinder sensors that generate a point cloud can make use of the ICP algorithm for point cloud registration. For instance, RGB-D sensors provide measurements of images synchronized with depth images and can use the point cloud generated from the depth image along with information from the image for point cloud registration [18]. Light detection and ranging (LiDAR) sensors are another class of rangefinder sensors with the broad use of ICP methods. These sensors use coherent light to measure the bearing, range, and often intensity of return points. One promising development in LiDAR technology in recent years is the advent of coherent frequency-modulated continuous-wave (FMCW) LiDARs, which employs direct modulation and demodulation of the laser waveform in frequency domain [21]. Compared to two other popular LiDAR schemes of pulsed and amplitude-modulated continuous-wave (AMCW) [3], FMCW LiDARs measure the beat frequency, a frequency difference resulting from an alternating constructive and destructive interference pattern caused by the outgoing and incoming signal, to indirectly measure the range of the target [24]. This scheme also allows FMCW LiDARs to be able to measure the relative velocity between each measured point to the sensor along the radial direction (Doppler velocity) by the Doppler effect, therefore providing additional motion information about the target other than range and intensity that other types of LiDARs cannot support. Furthermore, the FMCW technology can also be applied to radars to allow them to measure the Doppler velocity [28].

The advancement of the FMCW technology in range sensors offers a new dimension of information in the raw measurement level for each measured point, creating opportunities for traditional methodologies that only rely on range information to be revisited. For instance, the Doppler velocity measurement could be used to tackle the aforementioned feature-denied challenging environments for ICP algorithms. However, to our best knowledge, no such work has been reported to explore this possibility and we aim to fulfill this gap with our proposed method.

In this paper, we present a novel Doppler ICP (DICP) algorithm that takes into consideration the measured Doppler velocity for each measured point by FMCW LiDARs or radar sensors in the registration process. In DICP, we first establish the relationship between the Doppler velocity of an observed point and the transformation to be estimated. A Doppler

velocity residual is then defined and included in the objective function to be minimized jointly with classical geometric objectives used in ICP. The method is validated using both real-world data collected using an FMCW LiDAR on a driving vehicle and simulated data using the CARLA simulator [12].

The key contribution of our work can be summarized as:

- By deriving the Doppler error distance metrics and their gradients, we show that virtually any variant of ICP can be extended to include these terms to improve registration accuracy and robustness.
- We demonstrate that, by using gradients derived from Doppler velocity measurements, the optimization can be guided by terms that are independent of the structure of the environment, which is of particular importance for environments that are challenging to most variants of ICP as discussed above.
- The speed of convergence is greatly improved by having a guided optimization process by using the Doppler error component and gradient terms in addition to point cloud matching terms.
- We show that correspondence matches used for the optimization step in the registration algorithm are improved by exploiting the compatibility of each point's Doppler measurement and the current sensor's rigid motion estimate. This enhances the algorithm performance by eliminating most moving objects in a scene that would otherwise degrade the performance of point cloud registration.

II. Related Work

Several enhancements to the ICP algorithm have been proposed with different performance profiles in real-world datasets [4]. In [26], the augmentation of ICP is discussed by using normals and tangents to the surface as part of the optimization step. A point-to-plane algorithm is combined with the classical ICP algorithm [25].

We discuss a formulation where the linear and angular components are considered to derive the 3D rigid transformations used in the computation of ICP. A similar formulation to this paper can be found in [17], where the authors use the sensor velocity components to estimate the distortion effect from the sensor motion while building the point cloud.

Our proposed algorithm uses a formulation for instantaneous radial velocity measured as Doppler velocity to apply to a point cloud matching algorithm. In the experiments section, this measured point cloud is obtained by making use of an FMCW LiDAR. In the context of research for using Doppler enabled sensors for positioning, [28] presents an approach where a rotating FMCW radar is used to estimate ego-motion and build a map based on the static returns. In [27] and [11], the FMCW radar spectral information is used for place and pose estimation. [15] proposed a Doppler velocity-based cluster and velocity estimation algorithm using an FMCW LiDAR.

One of the issues addressed in our work, outlier rejection of dynamic objects in a scene, is addressed in [10, 18, 23]. Note that the algorithm described in this paper takes advantage of the Doppler velocity measurements for outlier rejection independently of robust outlier rejection via any other method involving a robust loss functions. In practice, DICP can be combined with robust loss methods to further enhance the performance of the point cloud registration algorithm.

III. Method

A. Notation

惯性坐标系

We denote $\mathcal{F}_I$ to be the inertial frame, $\mathcal{F}_V$ to be the vehicle body frame, and $\mathcal{F}_L$ to be the LiDAR body frame. For convenience, we also denote the vehicle body frame for the previous point cloud as the *source frame*, $\mathcal{F}_S$, and for the current point cloud; *target frame*, $\mathcal{F}_T$.

Generally, we denote ${}_F\mathbf{r}_{AB} \in \mathbb{R}^3$ as a vector from point A to point B expressed in the frame $\mathcal{F}_F$. In this configuration, the transformation matrix $\mathbf{T}_{AB} \in SE(3)$ transforms a vector ${}_B\mathbf{r}_{AB}$ expressed in $\mathcal{F}_B$ to the frame $\mathcal{F}_A$. $\mathbf{R}_{AB} \in SO(3)$ rotates a vector expressed in frame $\mathcal{F}_B$ to frame $\mathcal{F}_A$.

We denote the source point cloud $\mathcal{P}$ as a set of points such that P_j is the j-th point and p_j is the corresponding tuple of the point containing at least the 3D vector ${}_S\mathbf{r}_{SP_j}$ (which is the vector from the origin $\mathcal{F}_S$ to a source point P_j expressed in the source frame) and the measured Doppler velocity v_{meas_j}. For the target point cloud $\mathcal{Q}$, we denote Q_j to be the j-th point and q_j to be the corresponding point tuple containing the 3D vector ${}_T\mathbf{r}_{TQ_j}$ representing a vector from the origin $\mathcal{F}_T$ to the target point Q_j expressed in the target frame. The tuple q_j may also contain other relevant information such as the surface normal at the point Q_j denoted as ${}_T\mathbf{n}_{Q_j}$.

Last but not least, $\hat{\mathbf{a}}$ is denoted as the skew-symmetric matrix of vector $\mathbf{a}$, and a homogeneous coordinate of a vector $\mathbf{a} \in \mathbb{R}^3$ is defined as $\bar{\mathbf{a}} = \begin{bmatrix}\mathbf{a}^\top & 1\end{bmatrix}^\top$.

B. A Primer on ICP and State Vector Representation

The ICP algorithm aims to estimate a rigid-body transformation, $\mathbf{T}_{TS}$ that best aligns the source point cloud $\mathcal{P}$ to the target point cloud $\mathcal{Q}$. The general procedure of ICP can be described as correspondence association, error minimization, and outlier filtering. In the first stage, the correspondence between points is established using the Euclidean distance [7], geometric features [9] or colors of points [20]. Next, $\mathbf{T}_{TS}$ is estimated by iteratively minimizing

$$\mathbf{T}^*_{TS} = \arg\min_{\mathbf{T}_{TS}} E\left(\mathbf{T}_{TS}, \mathcal{P}, \mathcal{Q}\right), \tag{1}$$

where $E\left(\mathbf{T}_{TS}, \mathcal{P}, \mathcal{Q}\right)$ is the error function for all matched correspondences between point cloud $\mathcal{P}$ and $\mathcal{Q}$ with all points in $\mathcal{P}$ transformed using $\mathbf{T}_{TS}$. After the transformation is estimated and applied, the process is repeated by removing outliers and redefining correspondences. This process is iterated until either the solution converges or the termination criteria is reached, usually when the overall error function stops evolving

significantly. One widely adopted ICP variant is the point-to-point ICP algorithm, where the error function is defined based on the Euclidean distance between correspondences:

$$E\left(\mathbf{T}_{TS}, \mathcal{P}, \mathcal{Q}\right) = \sum_{j=1}^{N} \left\| \mathbf{T}_{TSS}\bar{\mathbf{r}}_{SP_j} - {}_T\bar{\mathbf{r}}_{TQ_j} \right\|^2, \quad (2)$$

where P_j and Q_j are a pair of correspondences, $j \in [1, ..., N]$ and N represents the number of pairs of correspondence.

Another type of algorithm is the point-to-plane ICP algorithm, where the residual is defined as the projection of Euclidean distance onto the surface normal at the corresponding target point:

$$E\left(\mathbf{T}_{TS}, \mathcal{P}, \mathcal{Q}\right) = \sum_{j=1}^{N} \left(\left(\mathbf{T}_{TSS}\bar{\mathbf{r}}_{SP_j} - {}_T\bar{\mathbf{r}}_{TQ_j} \right) \cdot {}_T\bar{\mathbf{n}}_{Q_j} \right)^2. \quad (3)$$

To minimize the error function, either a non-linear least square method such as the Gauss-Newton method and Levenberg-Marquardt algorithm [13] or Iteratively Reweighted Least-Squares (IRLS) with robust kernels can be applied [6]. To do that, we first denote a 6 dimensional coordinate $\mathbf{u}$ in the Lie algebra $\mathfrak{se}(3)$ to represent the state vector:

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_\theta^\top & \mathbf{u}_t^\top \end{bmatrix}^\top, \quad (4)$$

where $\mathbf{u}_\theta$ is the rotation component such that $\mathbf{u}_\theta = [\mathrm{Log}(\mathbf{R}_{TS})]^\vee$ and $\mathbf{u}_t = {}_T\mathbf{r}_{TS}$ is the translation component. Then we denote the pseudo-exponential map of $\mathbf{u}$ as in [8] such that

$$\mathbf{T}_{TS} = \text{pseudo-exp}(\mathbf{u}) = \begin{bmatrix} e^{\hat{\mathbf{u}}_\theta} & \mathbf{u}_t \\ \mathbf{0} & 1 \end{bmatrix}. \quad (5)$$

Note $\mathbf{R}_{TS} = e^{\hat{\mathbf{u}}_\theta}$ is the exact exponential map, whereas the $\mathbf{u}_t$ is left intact during the mapping. By doing so it defines a valid retraction on $SE(3)$ and also leads to a more efficient computation of the Jacobians. Thus, the minimization problem described by Equation 1 becomes

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} E(\mathbf{u}, \mathcal{P}, \mathcal{Q}). \quad (6)$$

In the following subsections, we will derive the relation between Doppler velocity and the state vector $\mathbf{u}$, which will be used to formulate new residual terms as well as in the outlier rejection in the DICP algorithm.

C. Doppler Velocity

It is assumed that the LiDAR[1] is mounted on a rigid body on the vehicle and that the offset ${}_V\mathbf{r}_{VL}$ and rotation $\mathbf{R}_{VL}$ (the LiDAR sensor extrinsic calibration) is known by calibration beforehand.

We also denote the vehicle's velocity expressed in the vehicle frame as ${}_V\mathbf{v}_V \in \mathbb{R}^3$ and the angular velocity to be ${}_V\boldsymbol{\omega}_{IV} \in \mathbb{R}^3$. The velocity of the LiDAR expressed in the vehicle frame is defined as ${}_V\mathbf{v}_L$ and the angular velocity to be ${}_V\boldsymbol{\omega}_{IL}$. From the aforementioned assumption that the LiDAR is rigidly mounted on the vehicle, the LiDAR velocity in the vehicle frame is given by

$${}_V\mathbf{v}_L = {}_V\mathbf{v}_V + {}_V\boldsymbol{\omega}_{IV} \times {}_V\mathbf{r}_{VL}. \quad (7)$$

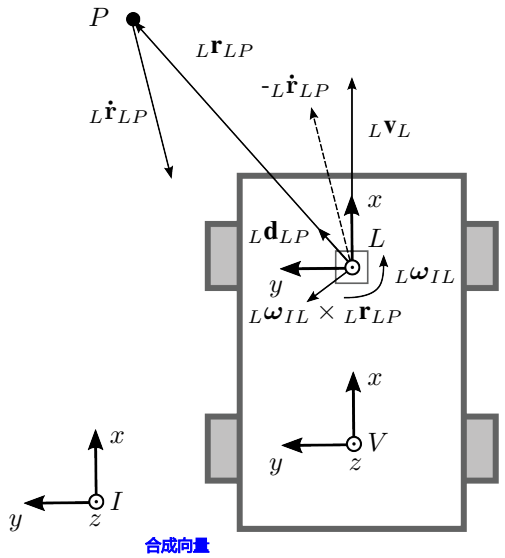

Fig. 2: An illustration of reference frames, position and velocity vectors. The velocity of a static point P with respect to the origin of $\mathcal{F}_L$, ${}_L\dot{\mathbf{r}}_{LP}$, is the resultant vector of ${}_L\mathbf{v}_L + {}_L\boldsymbol{\omega}_{IL} \times {}_L\mathbf{r}_{LP}$ (shown as the dashed arrow) with an opposite sign.

As illustrated in Figure 2, suppose a stationary point P exists within the detectable region of the LiDAR sensor, then its relative velocity ${}_L\dot{\mathbf{r}}_{LP}$ expressed in the LiDAR frame $\mathcal{F}_L$ can be expressed as

$${}_L\mathbf{v}_P = {}_L\mathbf{v}_L + {}_L\dot{\mathbf{r}}_{LP} + {}_L\boldsymbol{\omega}_{IL} \times {}_L\mathbf{r}_{LP}, \quad (8)$$

where ${}_L\mathbf{r}_{LP}$ is the range measurement of the LiDAR, ${}_L\mathbf{v}_L$ is the LiDAR velocity expressed in $\mathcal{F}_L$. Since the point P is stationary with respect to the inertial frame, it follows that ${}_L\mathbf{v}_P = \mathbf{0}$. We then have the relative velocity of point P with respect to the LiDAR frame, ${}_L\dot{\mathbf{r}}_{LP}$ expressed as

$${}_L\dot{\mathbf{r}}_{LP} = -{}_L\mathbf{v}_L - {}_L\boldsymbol{\omega}_{IL} \times {}_L\mathbf{r}_{LP}. \quad (9)$$

This derivation is illustrated in Figure 2 as well.

As Doppler sensors only measure the relative radial velocity, i.e. the relative velocity projected along the direction vector between the sensor and the point, the direction vector from the LiDAR to the point ${}_L\mathbf{d}_{LP}$ is expressed as

$${}_L\mathbf{d}_{LP} = \frac{{}_L\mathbf{r}_{LP}}{\| {}_L\mathbf{r}_{LP} \|}. \quad (10)$$

We here define the Doppler velocity as the projection of the relative velocity vector ${}_L\dot{\mathbf{r}}_{LP}$ along ${}_L\mathbf{d}_{LP}$ as ${}_Lv_{LP} \in \mathbb{R}$ and we further expand it using the definition from Equation 9:

[1]In this and the following sections, the name LiDAR is used to refer to a range measurement device that can measure Doppler velocity for each point. Even though we use the term LiDAR to refer to this class of sensors, it should be noted that it is not restricted to laser-only devices and this methodology can be generalized to a broader array of range sensors that are capable of measuring Doppler velocity, such as FMCW radars.

$$
\begin{aligned}
{}_L v_{LP} &= {}_L\mathbf{d}_{LP} \cdot {}_L\dot{\hat{\mathbf{r}}}_{LP} \\
&= -{}_L\mathbf{d}_{LP} \cdot ({}_L\mathbf{v}_L + {}_L\boldsymbol{\omega}_{IL} \times {}_L\mathbf{r}_{LP}).
\end{aligned} \tag{11}
$$

Note that vector ${}_L\mathbf{d}_{LP}$ has the same direction of the reading vector ${}_L\mathbf{r}_{LP}$ and is therefore orthogonal to the resultant vector of the cross product of ${}_L\mathbf{r}_{LP} \times {}_L\boldsymbol{\omega}_{IL}$. Hence, it follows that ${}_L\mathbf{d}_{LP} \cdot ({}_L\mathbf{r}_{LP} \times {}_L\boldsymbol{\omega}_L) = \mathbf{0}$, thus ${}_L v_{LP}$ is given by

$$
{}_L v_{LP} = -{}_L\mathbf{d}_{LP} \cdot {}_L\mathbf{v}_L. \tag{12}
$$

To express ${}_L v_{LP}$ in the vehicle frame, we substitute Equation 12 into Equation 7 and obtain

$$
\begin{aligned}
{}_L v_{LP} &= -{}_L\mathbf{d}_{LP}^\top (\mathbf{R}_{LV} {}_V\mathbf{v}_L) \\
&= -\left({}_L\mathbf{d}_{LP}^\top \mathbf{R}_{VL}^\top\right) {}_V\mathbf{v}_L \\
&= -\left(\mathbf{R}_{VL} {}_L\mathbf{d}_{LP}\right)^\top {}_V\mathbf{v}_L,
\end{aligned} \tag{13}
$$

which can be expressed, with the direction vector ${}_L\mathbf{d}_{LP}$, in the vehicle frame $\mathcal{F}_V$ as

$$
{}_V\mathbf{d}_{LP} = \mathbf{R}_{VL} {}_L\mathbf{d}_{LP}. \tag{14}
$$

Substituting Equations 7 and 14 into Equation 13, we obtain the Doppler velocity component expressed in the vehicle frame

$$
{}_L v_{LP} = -{}_V\mathbf{d}_{LP} \cdot ({}_V\mathbf{v}_V - {}_V\hat{\mathbf{r}}_{VL} {}_V\boldsymbol{\omega}_{IV}). \tag{15}
$$

Notice that this is the general definition where the vehicle frame $\mathcal{F}_V$ does not coincide with the LiDAR frame $\mathcal{F}_L$. If the vehicle frame is chosen to be the same as the LiDAR frame, the angular velocity ${}_V\boldsymbol{\omega}_{IV}$ is no longer measurable as a Doppler velocity component. This is because Doppler measurement devices only measure radial velocity components and cannot directly observe tangential velocity components elicited by rotation.

获得每一个点的多普勒速度，可以理解为点相对于laser的 相对速度

D. Doppler Velocity Residual

To incorporate the Doppler velocity into the formulation of the ICP algorithm, we first approximate the angular velocity ${}_V\boldsymbol{\omega}_{IV}$ within the period of one LiDAR sample Δt as

$$
{}_V\boldsymbol{\omega}_{IV} \approx -\frac{\mathbf{u}_\theta}{\Delta t}. \tag{16}
$$

Similarly, the linear velocity ${}_V\mathbf{v}_V$ is approximated as the amount of translation within the period of one LiDAR sample Δt as

$$
{}_V\mathbf{v}_V \approx -\frac{\mathbf{u}_t}{\Delta t}. \tag{17}
$$

The negative signs in Equations 16 and 17 are because the state vector that we are optimizing corresponds to the transform $\mathbf{T}_{TS}$, whereas the linear and angular velocities are approximated time derivatives of the transform $\mathbf{T}_{ST}$, which represents the vehicle's motion from the source to the target frame.

Furthermore, to simplify the derivation of the Jacobians of the corresponding residual terms, both definitions above use the time derivative of Lie algebra components to approximate the velocities, assuming that the rotation of the transformation between the source and target point cloud is relatively small and the sampling rate is fast enough.

For a given state-vector $\mathbf{u}$, the residual between the measured Doppler velocity and the expected Doppler velocity for the j-th point P_j given a state-vector $\mathbf{u}$ is given by

$$
r_{v_j} = v_{meas_j} - {}_L v_{LP_j}(\mathbf{u}).
$$

这个速度是我可以直接测量的，同时也可以根据R进行微分进行估计

Substituting Equation 16 and 17 into Equation 15, we have the Doppler velocity for P_j expressed as

$$
{}_L v_{LP_j}(\mathbf{u}) = \frac{1}{\Delta t} {}_V\mathbf{d}_{LP_j} \cdot (\mathbf{u}_t - {}_V\hat{\mathbf{r}}_{VL}\mathbf{u}_\theta). \tag{19}
$$

Substituting Equation 19 into Equation 18, the residual of Doppler velocity term becomes:

$$
r_{v_j} = v_{meas_j} - \frac{1}{\Delta t} {}_V\mathbf{d}_{LP_j} \cdot (\mathbf{u}_t - {}_V\hat{\mathbf{r}}_{VL}\mathbf{u}_\theta). \tag{20}
$$

Equation 6 can be solved using a general-purpose non-linear optimization approach or IRLS with robust kernels. In order to do so, the Jacobian of the residual with respect to the state is often required and we define the Doppler Jacobian term for the point P as

$$
\mathbf{J}_{v_j} = \frac{\partial r_{v_j}}{\partial \mathbf{u}} = -\frac{1}{\Delta t} {}_V\mathbf{d}_{LP_j}^\top \frac{\partial (\mathbf{u}_t - {}_V\hat{\mathbf{r}}_{VL}\mathbf{u}_\theta)}{\partial \mathbf{u}}. \tag{21}
$$

Solving Equation 21, we have the solution for the Jacobian of the Doppler velocity residual of the j-th point expressed in frame $\mathcal{F}_V$:

$$
\mathbf{J}_{v_j} = \frac{1}{\Delta t} \begin{bmatrix} ({}_V\mathbf{d}_{LP_j} \times {}_V\mathbf{r}_{VL})^\top & -{}_V\mathbf{d}_{LP_j}^\top \end{bmatrix}. \tag{22}
$$

One important thing to notice is that the Doppler velocity residual term does not depend on the target point cloud, i.e. it is intrinsic to the source point cloud. While one can perform ICP steps without taking into consideration the target point cloud and only use the Doppler ICP residual, the algorithm benefits greatly from having other types of residuals such as point-to-point or point-to-plane in addition to the Doppler velocity residual. Furthermore, since the residual terms are independent of the target point cloud, the Doppler velocity residual measurement terms will effectively guide the optimization of the overall error when combined with other point matching error terms and will greatly improve the performance where there are ambiguous references; those include long planar surfaces such as hallways, tunnels, parking lots, etc.

Also notice that the target point cloud is not required to contain Doppler velocity measurements. In practice, this means that this algorithm can be utilized not only when we are matching two sequential point clouds but can also be used to register a scan to a prior map of the world.

E. Dynamic Point Outlier Rejection

While a perfectly static point cloud registration is not affected by moving objects, any registration where there are moving objects in a scene can introduce errors in the estimated registration. Here we describe the use of the Doppler velocity measurements to reject moving objects that are to be considered outliers.

如果说给定了旋转的状态向量u，之后其实才能去估计这个量。

随着迭代过程，u会逐渐收敛，在这个过程中逐渐进行点的rejection

For a given state-vector as described in Equation 4, it is possible to estimate what the expected Doppler measurement should be and measure its error as defined in Equation 18. Dynamic points from moving objects will be rejected if their Doppler measurements don't agree with the expected Doppler velocity for a given state-vector during the iteration of the algorithm.

Let Δ_v be the threshold that specifies the maximum deviation from the expected Doppler velocity given the state-vector $\mathbf{u}$ and the actual measured Doppler velocity for a given source point P_j, a point can be determined to be an inlier if the following is true:

$$\mid r_{v_j} \mid < \Delta_v. \tag{23}$$

Notice that Δ_v does not necessarily need to be a constant. It could progressively become smaller as the algorithm converges as to not initially reject valid static points due to a state-vector being initially far from the optimal value.

F. Doppler Iterative Closest Point Algorithm

We wish to progressively apply a transformation to the source point cloud $\mathcal{P}$ to minimize an error function with respect to the target point cloud $\mathcal{Q}$, and we choose to solve the error function using the IRLS approach [6]. In such a formulation, the error function is defined as the weighted sum of squared residuals.

Therefore, the Doppler error term $E_v(\mathbf{u}, \mathcal{P})$ can be defined as

$$E_v(\mathbf{u}, \mathcal{P}) = \sum_{j=1}^{N} w_v(r_{v_j}) r_{v_j}^2, \tag{24}$$

where N is the number of correspondences, j is the index of a correspondence pair, and $w_v(\cdot)$ is the weight function for the Doppler velocity residual component. The corresponding Jacobian with respect to the state vector $\mathbf{u}$ is shown in Equation 22.

In this work, we choose the point-to-plane metric to minimize the geometric error between the correspondence pairs, however, other distance metrics are also applicable here. The point-to-plane residual term is defined as

$$r_{p_j} = \left(\mathbf{R}_{TS}(\mathbf{u}_\theta) {}_S\mathbf{r}_{SP_j} + \mathbf{u}_t - {}_T\mathbf{r}_{TQ_j}\right) \cdot {}_T\mathbf{n}_{Q_j}. \tag{25}$$

We make the small-angle approximation, similar to [2], where $\mathbf{R}(\mathbf{u}_\theta) \approx I_3 + \hat{\mathbf{u}}_\theta$. The Jacobian of the point-to-plane residual with respect to the state vector $\mathbf{u}$ for the j-th point is

$$\mathbf{J}_{p_j} = \frac{\partial r_{p_j}}{\partial \mathbf{u}} = \begin{bmatrix} \left({}_S\mathbf{r}_{SP_j} \times {}_T\mathbf{n}_{Q_j}\right)^\top & {}_T\mathbf{n}_{Q_j}^\top \end{bmatrix}. \tag{26}$$

The overall point-to-plane error function is expressed as

$$E_p\left(\mathbf{u}, \mathcal{P}, \mathcal{Q}\right) = \sum_{j=1}^{N} w_p(r_{p_j}) r_{p_j}^2, \tag{27}$$

where the term $w_p(\cdot)$ is the weight function for point-to-plane residual which could be a robust kernel. We refer the reader to [16] for more details on point-to-plane ICP optimization.

The joint optimization of the Doppler velocity objective and the geometric objective can be weighed by a parameter λ_v, which remains constant over all the correspondences and is defined as

$$E(\mathbf{u}, \mathcal{P}, \mathcal{Q}) = \lambda_v E_v(\mathbf{u}, \mathcal{P}) + (1 - \lambda_v) E_p(\mathbf{u}, \mathcal{P}, \mathcal{Q}). \tag{28}$$

The objective function in Equation 28 can be solved using an IRLS approach. The complete DICP algorithm along with dynamic point outlier rejection is summarized in Algorithm 1.

Algorithm 1: Doppler Iterative Closest Point

Input:
$\mathcal{P}$ Source point cloud containing Doppler velocity measurements
$\mathcal{Q}$ Target point cloud (optionally containing normals)
Δ_d Maximum correspondence distance
Δ_v Maximum velocity error
$\mathbf{u}_0$ Initial state vector
Output:
$\mathbf{u}$ Transform that aligns $\mathcal{P}$ with $\mathcal{Q}$

$\mathbf{u} \leftarrow \mathbf{u}_0$
while *not converged* **do**
  $\mathcal{P}' \leftarrow \varnothing$
  $\mathcal{Q}' \leftarrow \varnothing$
  for $j \leftarrow 1$ **to** $\mid \mathcal{P} \mid$ **do**
    $p'_j \leftarrow$ `TransformSourcePoint`$(\mathbf{u}, p_j)$
    $q_j \leftarrow$ `FindClosestTargetPoint`$(\mathcal{Q}, p'_j)$
    if $\mid r_{v_j} \mid < \Delta_v \wedge$ `Dist`$(p'_j, q_j) < \Delta_d$ **then**
      $\mathcal{P}' \leftarrow \mathcal{P}' \cup p_j$
      $\mathcal{Q}' \leftarrow \mathcal{Q}' \cup q_j$
    end
  end
  $\mathbf{u} \leftarrow \arg\min_{\mathbf{u}} E\left(\mathbf{u}, \mathcal{P}', \mathcal{Q}'\right)$
end

IV. Experiments

In this section, we provide an overview of the dataset, discuss the different parameters used in the implementation, and show quantitative and qualitative results on several sequences which include sequential Doppler LiDAR scans from both real and simulated data.

A. Dataset (Sequences)

We evaluate our method on seven different sequences, of which five were collected from a real sensor and two were generated from a simulator. Since we are interested in understanding the effectiveness of Doppler ICP in feature-denied environments, most of the sequences include tunnels, highways, and bridges as shown in Figure 3. As an exception, we also include an urban-driving scenario from San Francisco city which is rich in geometric features to benchmark our method where conventional ICP methods perform well. The key statistics of all sequences are provided in Table I.

TABLE I: Statistics of the dataset.

Sequence	Trajectory Length (m)	Duration (seconds)	# Avg. Points
Baker-Barry Tunnel (Empty)	860.31	83.7	55.7k
Baker-Barry Tunnel (Vehicles)	906.86	65.5	36.5k
Robin Williams Tunnel	688.76	30.0	43.3k
Brisbane Lagoon Freeway	4941.80	176.3	40.0k
San Francisco City	1378.14	450.0	120.1k
CARLA Town04 (Straight Walls)	599.91	46.4	78.8k
CARLA Town05 (Curved Walls)	426.81	76.0	80.4k

Fig. 3: Robin Williams Tunnel in Sausalito, CA (top left); the stretch of US-101 near Brisbane Lagoon (top right); straight parallel walls placed in CARLA Town04 (bottom left); and curved walls placed in CARLA Town05 (bottom right) to simulate a feature-less scenario.

For the real-world data, we use Aeva's Aeries I FMCW LiDAR to obtain the range and Doppler velocity measurements. The LiDAR has a horizontal field-of-view of $120°$, a vertical field-of-view of $30°$, a 300 m maximum operating range, a Doppler velocity measurement precision of 3 cm/s, and a sampling rate of 10 Hz. Novatel ProPak6 GNSS [19] was used to obtain the ground truth odometry for these sequences. To generate the simulated sequences, we simulated an FMCW LiDAR in the CARLA simulator based on the equations presented in Section III-C. We placed large parallel walls on either side of a straight highway and a curved highway, as shown in Figure 3, to evaluate the effectiveness of our algorithm in a perfectly feature-denied environment. The ground truth transform was obtained by querying the CARLA actor's pose. The same scan pattern and sample rate from the Aeries I LiDAR were used in the simulation to resemble the behavior of the real LiDAR.

B. Implementation

We implement the Doppler ICP algorithm by extending the existing point-to-plane implementation in Open3D [29]. We employ robust kernels to minimize the effect of outlier correspondences (shown in Equations 24 and 27) with Tukey loss with $k = 0.5$ for the point-to-plane residual and Tukey loss with $k = 0.2$ for the Doppler residual term. The Doppler residual robust kernel is enabled only after the second iteration of ICP to not reject too many correspondences in the initial few iterations where the error between the predicted and measured Doppler velocity would be high. We set $\lambda_v = 0.01$ in all our experiments, which was determined empirically. The dynamic point outlier rejection threshold from Equation 23 is set to $\Delta_v = 2\mathrm{m\,s}^{-1}$.

We separate our evaluation into groups of *No Seed Estimate* and *With Seed Estimate*. For *No Seed Estimate* experiments, we do not initialize or seed the pose estimate for the registration methods. For *With Seed Estimate* experiments, we use the estimated pose from the previous pair of registered scans in the sequence under a constant-velocity motion-model assumption. This is a more practical use case for ICP. We do not rely on any external sensor data and solely rely on the range and Doppler velocity measurements from the FMCW LiDAR when seeding the pose estimate. Other ways to seed the pose estimate include using the Doppler velocity measurements to estimate the ego-vehicle velocity and consequently estimate the pose in the sample period, or using additional sensors such as an IMU (Inertial Measurement Unit) along with pre-integration as proposed in [14].

C. Results and Discussion

We benchmark our performance (labeled `DICP`) against Open3D's point-to-plane ICP (labeled `ICP-Open3D`) with a robust kernel (Tukey loss, $k = 0.5$) and do not use any Doppler velocity measurements there. All methods are evaluated based on the Relative Pose Error (RPE) metric [22] in translation and rotation between the ground-truth and the estimated relative pose using point cloud registration. We also report the error in the total length of the estimated trajectory and the number of iterations each registration method took to converge.

The metrics are presented in Table II and the estimated trajectories are shown in Figure 4. Figures 1 and 7 show a comparison of the scenes reconstructed using different registration methods. The trajectories are purely composed of sequential pose estimates represented in the inertial frame and no filtering or graph-based optimization techniques are used.

Better Pose Estimation: As shown in Table II and Figure 4, our method outperforms the point-to-plane ICP method in feature-denied environments in terms of RPE and trajectory path error. In `Robin Williams Tunnel`, `Baker-Barry Tunnel (Empty)`, and the CARLA sequences, due to lack of geometric variations along planar surfaces, sequential point cloud scans appear very similar to each other, as if they were captured from the same location. This under-constraints the geometric objective function and leads to the failure of the point-to-plane ICP, as highlighted in the green boxes in Figure 5 and in Figure 7 (a), (b), and (f) as well. In scenarios where the point cloud geometry is dominated by dynamic objects (examples: `Brisbane Lagoon Freeway`, `Baker-Barry Tunnel (Vehicles)`, and `San Francisco City`), point-to-plane ICP registers to the motion of the moving object(s) instead of estimating the ego-vehicle motion correctly despite using robust kernels for outlier rejection, as shown in the red boxes in Figure 5 and white boxes in Figure 7 (c), (d), and (e). Our proposed method

TABLE II: Point cloud registration results.

Sequence	Method	No Seed Estimate				With Seed Estimate			
		RPE Trans (m)	RPE Rot (deg)	Path Error (m)	# Iters (mean)	RPE Trans (m)	RPE Rot (deg)	Path Error (m)	# Iters (mean)
Baker-Barry Tunnel (Empty)	ICP-Open3D	1.0416	0.1180	525.35	30.8	0.7608	0.1131	203.86	15.3
	DICP (Ours)	**0.0694**	**0.1099**	**1.23**	**7.6**	**0.0694**	**0.1091**	**1.24**	**5.4**
Baker-Barry Tunnel (Vehicles)	ICP-Open3D	1.2641	0.1817	656.57	26.1	0.9385	**0.1468**	326.12	14.6
	DICP (Ours)	**0.0807**	**0.1493**	**15.61**	**8.4**	**0.0807**	0.1498	**15.60**	**6.1**
Robin Williams Tunnel	ICP-Open3D	1.8174	0.1955	366.84	44.3	1.9343	0.1533	219.10	20.3
	DICP (Ours)	**0.0752**	**0.1428**	**0.03**	**13.1**	**0.0758**	**0.1415**	**0.05**	**9.8**
Brisbane Lagoon Freeway	ICP-Open3D	2.7743	0.2019	4337.18	37.5	0.3270	0.0915	45.27	10.6
	DICP (Ours)	**0.1132**	**0.0894**	**4.16**	**17.2**	**0.1301**	**0.0869**	**7.20**	**9.7**
San Francisco City	ICP-Open3D	0.1853	0.0510	181.27	11.8	0.0323	**0.0479**	23.78	**5.9**
	DICP (Ours)	**0.0308**	**0.0489**	**10.74**	**7.4**	**0.0317**	0.0482	**7.30**	6.0
CARLA Town04 (Straight Walls)	ICP-Open3D	1.3935	0.0313	520.53	28.5	18.0178	0.3004	6446.52	35.7
	DICP (Ours)	**0.0101**	**0.0108**	**0.40**	**4.2**	**0.0101**	**0.0108**	**0.41**	**3.2**
CARLA Town05 (Curved Walls)	ICP-Open3D	0.2626	0.0449	100.29	15.3	0.2557	0.0461	91.60	13.9
	DICP (Ours)	**0.0117**	**0.0335**	**1.50**	**4.6**	**0.0119**	**0.0340**	**1.51**	**4.3**

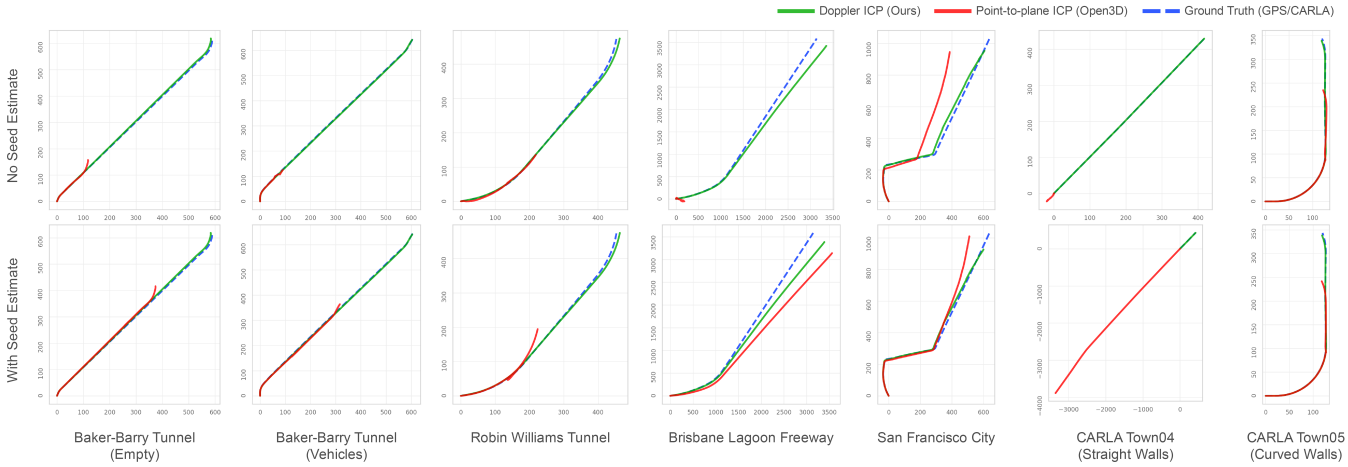

Fig. 4: Trajectories for all sequences from composition of poses estimated using Doppler ICP (green lines) and Open3D point-to-plane ICP (red lines). The ground truth trajectories (based on GPS or obtained from CARLA simulation) are shown in dashed blue lines.

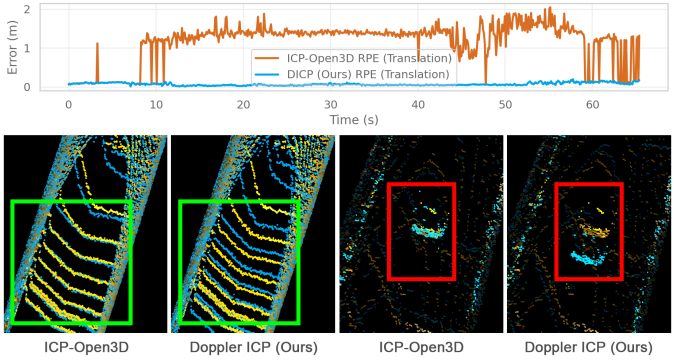

Fig. 5: Plot showing the rise in RPE (translation) for `Baker-Barry Tunnel (Vehicles)` as the ego-vehicle enters it (top). ICP-Open3D registers to the scan pattern (left green-box), thereby estimating no motion; in another instance, it registers incorrectly to the motion of a vehicle in front (left red-box). Our method (DICP) estimates the motion of the ego-vehicle correctly in both failure cases.

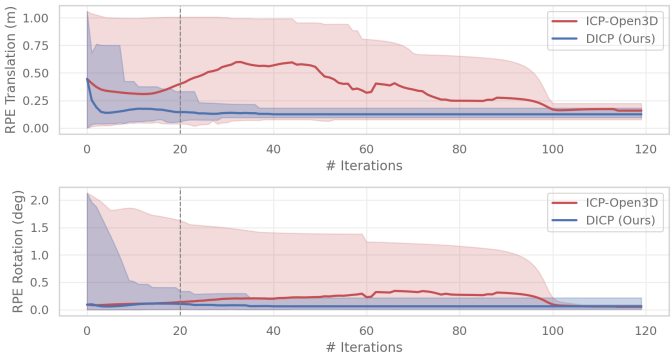

Fig. 6: Plot showing a comparison of mean (solid line) and min/max bounds (faded region) of Relative Pose Error (RPE) averaged at each iteration of the ICP algorithm, across all pairs of scans from the `San Francisco City` sequence. This demonstrates the faster convergence of Doppler ICP.

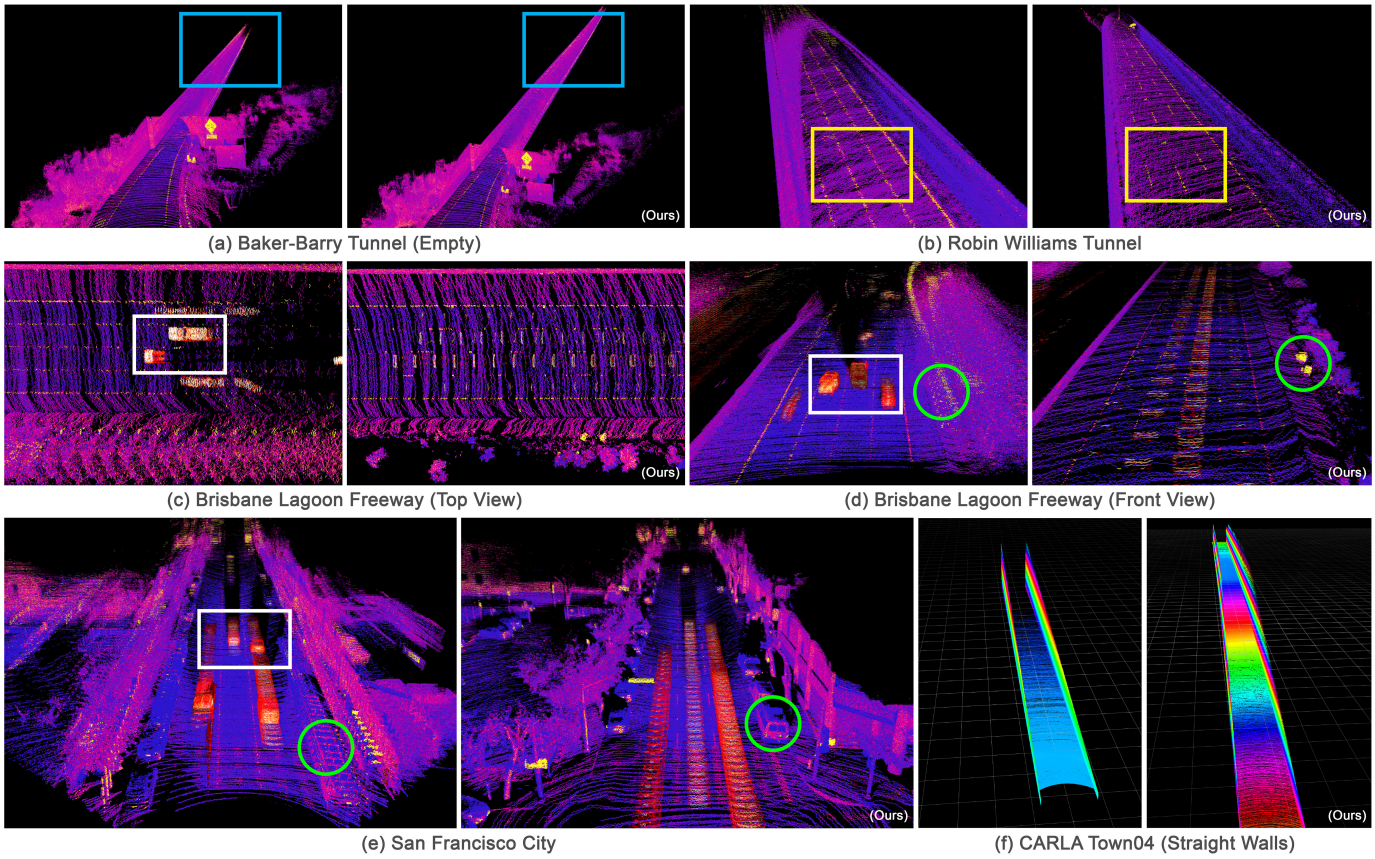

Fig. 7: Reconstruction of different sequences by the composition of pose estimates. In each case, the left figure represents Open3D's point-to-plane ICP results and the right; Doppler ICP (ours). We only use the reflectivity channel from the LiDAR for visualization purposes in (a) to (e) and the height colormap in (f). (a), (b), and (f) demonstrate that point-to-plane ICP fails to reconstruct the entire length of the tunnel(s) correctly due to insufficient geometric constraints. In (a), the blue box highlights DICP's success in reconstructing the entire tunnel. The yellow boxes in (b) show how the lane markers can be clearly differentiated in our method. The white boxes in (c), (d), and (e) show that point-to-plane ICP registers to the moving vehicles in the scene instead of the static points. The green circles here highlight the resultant registration artifacts with point-to-plane ICP.

of pruning outlier correspondences makes it robust to such dynamic points. Although we benchmark only against point-to-plane ICP, the results suggest that other geometric-based ICP variants are likely to fail due to the lack of geometric feature in the data set.

Faster Convergence: We observe that the optimization guided by the Doppler velocity gradients converges faster, by an average factor of around 3.4 in terms of the number of iterations for convergence among the sequences in our *No Seed Estimate* experiments (from Table II). Our method achieves similar performance in terms of RPE for the `San Francisco City` sequence which is rich in geometric features but the faster convergence shows the benefit of using DICP, as shown in Figure 6.

V. Conclusions

In this work, we presented a new algorithm for point cloud registration that leverages the Doppler velocity measurements from a point cloud captured by an FMCW range sensor. We provided a detailed formulation of a new Doppler velocity residual function and its joint optimization along with any of the existing geometric objectives commonly used in the ICP framework. Our experimental evaluation shows this approach significantly improves the convergence rates and registration accuracy by providing additional optimization constraints especially in feature-denied environments with non-distinctive and/or repetitive geometric surfaces where classical ICP variants tend to not converge properly. We also devised a method to reject dynamic points from the optimization process which otherwise would introduce errors in the estimated transform with existing ICP variants.

Despite the encouraging results, there are several avenues for future research to enhance the usage of the method. We believe that the Doppler velocity objective function introduced in this paper, used in conjunction with other filtering or graph-based optimization techniques in a larger SLAM framework, could yield even greater results, as well as fusing the data with other modalities of sensors such as cameras and IMUs. The DICP algorithm may be extended to additionally refine the extrinsic sensor calibration parameters or correct for the distortion due to motion in point cloud geometry.

References

[1] Ioan Andrei Barsan, Shenlong Wang, Andrei Pokrovsky, and Raquel Urtasun. Learning to localize using a lidar intensity map. *arXiv preprint arXiv:2012.10902*, 2020. URL https://arxiv.org/pdf/2012.10902.

[2] Jens Behley and Cyrill Stachniss. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In *Robotics: Science and Systems*, volume 2018, 2018. URL http://www.roboticsproceedings.org/rss14/p16.pdf.

[3] Behnam Behroozpour, Phillip AM Sandborn, Ming C Wu, and Bernhard E Boser. Lidar system architectures and circuits. *IEEE Communications Magazine*, 55 (10):135–142, 2017. URL https://www.tu-chemnitz.de/physik/EXSE/ForPhySe/Behroozpour%20IEEE%20Communications%202017.pdf.

[4] Ben Bellekens, Vincent Spruyt, Rafael Berkvens, and Maarten Weyn. A survey of rigid 3d pointcloud registration algorithms. In *AMBIENT 2014: the Fourth International Conference on Ambient Computing, Applications, Services and Technologies, August 24-28, 2014, Rome, Italy*, pages 8–13, 2014. URL https://repository.uantwerpen.be/docman/irua/1ab789/4a6a3c9a.pdf.

[5] Ben Bellekens, Vincent Spruyt, Rafael Berkvens, Rudi Penne, and Maarten Weyn. A benchmark survey of rigid 3D point cloud registration algorithms. *Int. J. Adv. Intell. Syst*, 8:118–127, 2015. URL https://www.researchgate.net/publication/280040097_A_Benchmark_Survey_of_Rigid_3D_Point_Cloud_Registration_Algorithms.

[6] Per Bergström and Ove Edlund. Robust registration of point sets using iteratively reweighted least squares. *Computational optimization and applications*, 58(3): 543–561, 2014. URL https://link.springer.com/article/10.1007/s10589-014-9643-2.

[7] Paul J Besl and Neil D McKay. Method for registration of 3-D shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. URL http://www.cvl.iis.u-tokyo.ac.jp/class2004/wedenesday/report/besl.pdf.

[8] Jose-Luis Blanco. A tutorial on se (3) transformation parameterizations and on-manifold optimization. *University of Malaga, Tech. Rep*, 3: 6, 2010. URL http://ingmec.ual.es/~jlblanco/papers/jlblanco2010geometry3D_techrep.pdf.

[9] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. URL http://graphics.stanford.edu/courses/cs348a-17-winter/Handouts/chen-medioni-align-rob91.pdf.

[10] Dmitry Chetverikov and Dmitry Stepanov. Robust euclidean alignment of 3d point sets. In *First Hungarian Conference on Computer Graphics and Geometry*, pages 70–75. Citeseer, 2002. URL https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.6.9399&rep=rep1&type=pdf.

[11] Daniele De Martini, Matthew Gadd, and Paul Newman. kRadar++: Coarse-to-Fine FMCW Scanning Radar Localisation. *Sensors*, 20(21):6002, 2020. URL https://www.mdpi.com/1424-8220/20/21/6002/html.

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An Open Urban Driving Simulator, 2017. URL http://proceedings.mlr.press/v78/dosovitskiy17a/dosovitskiy17a.pdf.

[13] Andrew W Fitzgibbon. Robust registration of 2D and 3D point sets. *Image and vision computing*, 21(13-14): 1145–1153, 2003. URL http://luthuli.cs.uiuc.edu/~daf/courses/Opt-2019/Papers/sdarticle.pdf.

[14] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015. doi: 10.15607/RSS.2015.XI.006. URL http://www.roboticsproceedings.org/rss11/p06.pdf.

[15] Mian Guo, Kai Zhong, and Xiaozhi Wang. Doppler velocity-based algorithm for Clustering and Velocity Estimation of moving objects. *arXiv preprint arXiv:2112.12984*, 2021. URL https://arxiv.org/pdf/2112.12984.

[16] Ankur Handa. Simplified Jacobians in 6-DoF Camera Tracking. Technical report, Technical report, University of Cambridge, 2014. URL http://www.doc.ic.ac.uk/~ahanda/simjacob.pdf.

[17] Seungpyo Hong, Heedong Ko, and Jinwook Kim. VICP: Velocity updating iterative closest point algorithm. In *2010 IEEE International Conference on Robotics and Automation*, pages 1893–1898. IEEE, 2010. URL http://vigir.missouri.edu/~gdesouza/Research/Conference_CDs/IEEE_ICRA_2010/data/papers/2156.pdf.

[18] Deok-Hwa Kim and Jong-Hwan Kim. Image-Based ICP algorithm for visual odometry using a RGB-D sensor in a dynamic environment. In *Robot Intelligence Technology and Applications 2012*, pages 423–430. Springer, 2013. URL https://link.springer.com/chapter/10.1007%2F978-3-642-37374-9_41.

[19] *"ProPak6™ Product Sheet"*. Novatel, 11 2015. URL https://portal.hexagon.com/public/Novatel/assets/Documents/Papers/ProPak6-PS-D18297. Rev. 7. Online; accessed Jan-2022.

[20] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *Proceedings of the IEEE international conference on computer vision*, pages 143–152, 2017. URL http://proceedings.mlr.press/v78/dosovitskiy17a/dosovitskiy17a.pdf.

[21] Diego Pierrottet, Farzin Amzajerdian, Larry Petway, Bruce Barnes, George Lockard, and Manuel Rubio. Linear FMCW laser radar for precision range and vector velocity measurements. *MRS Online Proceedings Library (OPL)*, 1076, 2008. URL https://ntrs.nasa.gov/api/citations/20080026181/downloads/20080026181.pdf.

[22] David Prokhorov, Dmitry Zhukov, Olga Barinova,

Konushin Anton, and Anna Vorontsova. Measuring robustness of Visual SLAM. In *2019 16th International Conference on Machine Vision Applications (MVA)*, pages 1–6, 2019. doi: 10.23919/MVA.2019.8758020.

[23] Diego Rodriguez-Losada and Javier Minguez. Improved data association for icp-based scan matching in noisy and dynamic environments. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3161–3166. IEEE, 2007. URL https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.141.5915.

[24] Santiago Royo and Maria Ballesta-Garcia. An overview of lidar imaging systems for autonomous vehicles. *Applied sciences*, 9(19):4093, 2019. URL https://www.mdpi.com/2076-3417/9/19/4093/pdf.

[25] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009. URL https://www.robots.ox.ac.uk/~avsegal/resources/papers/Generalized_ICP.pdf.

[26] Jacopo Serafin and Giorgio Grisetti. Using Augmented Measurements to Improve the Convergence of ICP. *Simulation, Modeling, and Programming for Autonomous Robots Lecture Notes in Computer Science*, page 566–577, 2014. doi: 10.1007/978-3-319-11900-7_48. URL http://jacoposerafin.com/wp-content/uploads/serafin14simpar.pdf.

[27] Heonkyo Sim, The-Duong Do, Seongwook Lee, Yong-Hwa Kim, and Seong-Cheol Kim. Road environment recognition for automotive FMCW radar systems through convolutional neural network. *IEEE Access*, 8: 141648–141656, 2020. URL https://ieeexplore.ieee.org/iel7/6287639/8948470/09153555.pdf.

[28] Damien Vivet, Paul Checchin, and Roland Chapuis. Localization and mapping using only a rotating FMCW radar sensor. *Sensors*, 13(4):4527–4552, 2013. URL https://www.mdpi.com/1424-8220/13/4/4527/pdf.

[29] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A Modern Library for 3D Data Processing. *arXiv:1801.09847*, 2018. URL https://arxiv.org/pdf/1801.09847.