Characterizing Perspective Error in Voxel-Based LIDAR Scan Matching

Jason H. Rife and Matthew McDermott, *Tufts University*

BIOGRAPHY

Jason Rife is a Professor and Chair of the Department of Mechanical Engineering at Tufts University in Medford, Massachusetts. He directs the Automated Systems and Robotics Laboratory (ASAR), which applies theory and experiment to characterize integrity of autonomous vehicle systems. He received his B.S. in Mechanical and Aerospace Engineering from Cornell University and his M.S. and Ph.D. degrees in Mechanical Engineering from Stanford University.

Matthew McDermott is a student in the Mechanical Engineering Ph.D. program at Tufts University in Medford, MA. He works in the Automated Systems and Robotics Laboratory (ASAR) with Dr. Jason Rife. He received his B.S. and M.S. degrees in Mechanical Engineering at Tufts University.

ABSTRACT

This paper quantifies a significant error source that limits the accuracy of LIDAR scan matching. LIDAR scan matching, which is used in dead reckoning (aka *LIDAR odometry*) and in mapping, computes the rotation and translation that best align a pair of point clouds. Perspective errors occur when a scene is viewed from different angles, with different surfaces becoming visible or occluded from each point of view. Specifically, this paper models perspective errors for two objects representative of the urban landscapes in which LIDAR frequently operates: a cylindrical column and a dual-wall corner. For each object, we provide an analytical model of the perspective error for voxel-based LIDAR scan matching. We then analyze how perspective errors accumulate as a LIDAR-equipped vehicle moves past these objects.

1 INTRODUCTION

LIDAR systems are data-rich perception sensors, which cast optical beams in known directions and compute the time of flight for the reflected signal. Time of flight can be converted to distance, so each return corresponds to a vector (range and direction) describing a point on some surface in the surrounding environment (or *scene*). By scanning over a set of directions, the LIDAR creates an image, called a *point cloud*, which represents the scene's geometry. It is often useful to align (or *register* or *scan match*) more than one point cloud. Applications of this scan matching process include creating a map, inferring relative motion over time, or doing both via Simultaneous Localization and Mapping (SLAM), as described in Thrun (2005).

Perspective errors, sometimes referred to as self-occlusions (Xu *et al*., 2022), arise in LIDAR scan matching when comparing two views, each capturing the same three-dimensional surface from a different location. When moving past a static three-dimensional object, previously hidden patches of the object may become hidden or visible due to the object's shape. Though related to shadowing, which occurs when one object occludes a second (Hassani & Joerger 2021, McDermott & Rife 2022a), perspective errors refer specifically to an object occluding itself. While shadowing and perspective errors have a common root cause, different strategies may be employed to mitigate each, as long shadows cut through many voxels, whereas perspective errors are grid-resolution problems confined within one voxel.

Existing scan matching algorithms can generally be classified into one of three categories: feature-based methods, voxel-based methods, and direct machine-learning (ML) methods. Feature-based methods seek to correspond recognizable features from a pair of point clouds, either individual points (as in the well-known Iterative Closest Point or ICP algorithm) or extracted features like planes (Besl & McKay, 1992; Shan & Englot, 2018; Zhang & Singh, 2014). Hybrid feature-based methods also exist, which combine ML-based strategies with feature extraction, such as the recent work by Plachetka *et al*. (2021), which applies ML to create bounding boxes for objects identified in a scene. Voxel-based methods, such as the Normal Distribution Transform

(NDT) and the Iterative Closest Ellipsoidal Transform (ICET) algorithms, divide the point cloud into volume elements (or *voxels*), each containing a distribution of points; correspondence is performed purely geometrically by attempting to align point distributions within each voxel (Biber & Straßer, 2003; Stoyanov *et al.*, 2012; McDermott & Rife, 2022b; McDermott & Rife, 2022c). Direct ML algorithms, like LO-Net (Li *et al.*, 2019), do not explicitly form correspondences or compute geometric transformations to relate them; instead, they train neural networks to align point clouds directly.

For safety-of-life navigation systems, voxel-based algorithms are advantageous. Unlike feature-based algorithms, where faulty data association creates errors that are challenging to model as noted by Hassani *et al.* (2018), voxel-based methods do not suffer from data-association faults, at least not in static scenes. Unlike machine-learning algorithms, where processing methods are opaque and where unpredictable faults are possible in corner cases (Willers *et al.,* 2020), voxel-based methods are transparent and interpretable, with each step closely related to geometric analysis (McDermott & Rife, 2022b).

Because of the advantages of voxel-based LIDAR scan matching for safety-of-life applications, we will focus on analyzing the impact of perspective errors on voxel-based methods. The main contribution of the paper is to quantify perspective errors for voxel-based scan matching when viewing two representative types of object: a cylindrical column and a dual-wall corner. Our approach will begin with a closer look at the conditions that create perspective errors. We will then derive analytical equations to model perspective errors for cylinders and corners. Subsequently, we interpret these analytical models using simulations, with the goal of informing mitigation strategies to minimize the impact of perspective errors in future LIDAR applications.

2 VISUALIZING PERSPECTIVE ERROR

When using LIDAR to image objects with curved or complex surfaces, the shape of the resulting point cloud depends on the LIDAR's location, as shown in Figure 1. The figure illustrates the top-down view of a cylindrical column. In Figure 1a, the column is viewed from two LIDAR locations (crosses) that are separated laterally. The point clouds generated from each location share a common overlapping region (green arc), but each point cloud also includes samples from a non-overlapping region (red arcs). The "missing data" from each point cloud leads to a difference in the mean location of the samples in each case (with the distinct point-cloud mean locations each identified by a star). Algorithms like NDT (Biber & Straßer, 2003) and ICET (McDermott & Rife, 2022b) attempt to align the point-cloud means between two scans. Assuming the column is an immobile object, then aligning the two means (stars) results in the false inference that the two viewpoints (crosses) are closer together than their true distance.

Perspective errors are generally less severe for radial motion than for lateral motion, as shown in Figure 1b. Although the symbols in Figure 1b match those of the rest of the figure, the LIDAR locations are different. In Figure 1b, the two LIDAR locations are aligned on the same spoke radiating from the cylindrical column's centerline. Again, there is an overlapping set of samples (on the green arc) and a non-overlapping set of samples (red arc), but in this case, the more distant LIDAR scan (blue) visualizes the entire region seen by the nearer scan (brown). Importantly the data missing from the nearer scan (red arcs) is symmetric, so the point-cloud means (stars) are offset only in radius and not in the circumferential coordinate.

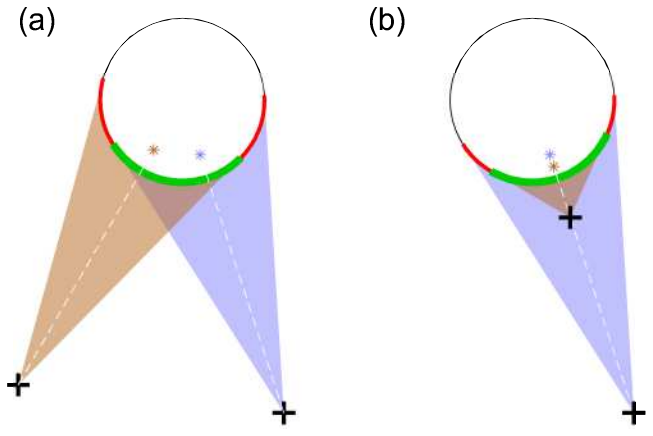

FIGURE 1
Changes in perspective for (a) lateral and (b) radial movement of the LIDAR between two locations (cross markers)

These examples demonstrate that even a smoothly curved, convex surface (like a cylindrical column) can occlude the LIDAR's view and corrupt the resulting point cloud with missing data. Even more dramatic occlusions occur for more complex objects, such as those involving sharp corners or outcroppings. In this paper, we do not consider signal-miss effects due to reflective surfaces (Xu *et al*., 2022).

Interestingly, perspective errors are driven not only by missing-data regions (red arcs from Figure 1), but also by a secondary effect: the density of points within the cloud. As an example of this, consider the flat wall that is visualized from above in Figure 2a and Figure 2b. For a flat wall of finite length, there is no missing data, because the LIDAR sees the entire face of the wall from any viewing location (unless it moves behind the wall). However, even though the full surface of the wall is visible, there may still be minor variations in the mean location of the point-cloud as caused by changes in the density of the LIDAR samples. Consider the difference in the distribution of samples moving from a viewing location on the left of the wall (Figure 2a) to a more central viewing axis (Figure 2b). When viewing from the left, the point density is much higher on the left than the right. When viewing from the middle, the point density is highest in the middle. Where the point density is highest, the interval between samples is shortest (as shown in Figure 2c). In this particular case, the high density of samples on the left side of the wall in Figure 2a will shift the point-cloud mean slightly to the left as compared to Figure 2b.

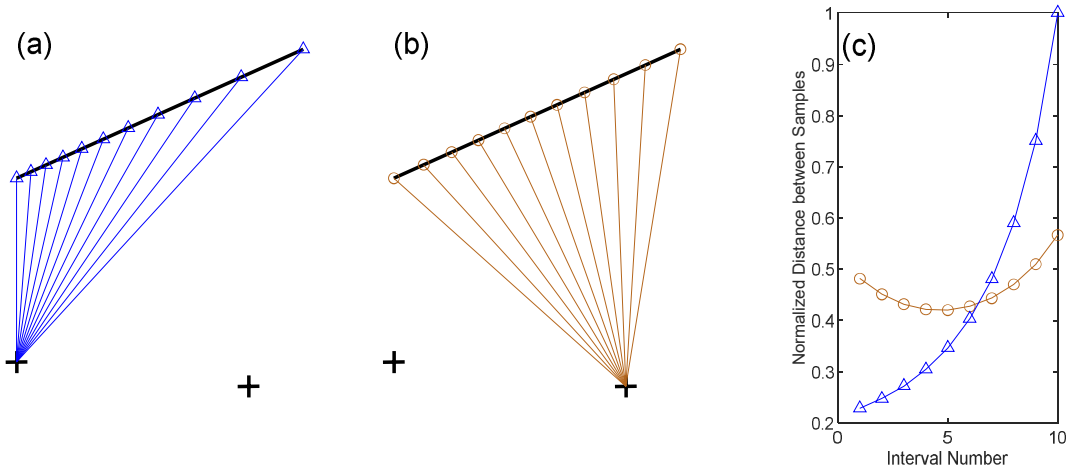

FIGURE 2
Point density along a planar surface is a function of LIDAR location: A short wall is shown visualized from two locations, indicated with cross markers. When the LIDAR is located at the left cross, points are sampled as shown in (a), and when the LIDAR is located on the right cross, points are sampled as shown in (b). Distances between sequential points as a function of point index (counting from left to right) are plotted in (c).

For typical urban scenes, perspective effects are only significant for a small number of objects (or voxels) viewed in the scene. Given the number of voxels in the scene, systematic perspective errors on a few voxels might not have a significant impact if the perspective errors were aligned in random directions; however, as we will show in the next section, perspective effects result in significant and persistent errors that tend to align with the direction of motion. Thus, they may accumulate and be significant even if they impact a relatively small fraction of voxels.

As a final note, it is significant to note that in this paper we do not consider limits on the LIDAR field of view. As a consequence, our models of perspective error consider only LIDAR translation. In the absence of a field of view restriction, LIDAR rotation does not cause perspective error. Consider the point clouds in Figure 1 and Figure 2; these point clouds would have essentially the same shape, regardless of the yaw orientation of the LIDAR, at least assuming a rotating LIDAR that scans in all azimuth directions. The situation would be somewhat different when driving the rotating LIDAR unit past a tall building or under a highway sign or bridge, because parts of the scene would pass above the upper limit of the rotating LIDAR's field of view, noting that the elevation angle limit for a typical scanning LIDAR is about 15° (Velodyne, 2019). Studying the impact of field of view on perspective error, as might be significant for airborne applications, is left as a topic for future work.

3 ANALYTICAL MODELS FOR POINT-CLOUD MEAN

This section develops analytical models to quantify perspective errors associated with two types of objects, cylindrical columns and dual-wall corners. These objects are representative of features common in urban environments. Because we focus on voxel-based algorithms that perform scan matching by attempting to align the point distributions within each voxel, we assess the systematic bias as equal to the shift in the point cloud's mean location. We assume dense, effectively continuous, sampling and assume the point cloud covers the entire surface of the object that is visible from the LIDAR's instantaneous location. As a further simplification, we assume the object is located entirely inside a single voxel. This is not an unreasonable assumption, as scan-matching algorithms often use voxels with a span of a meter or more.

3.1 Perspective Error for a Cylinder

A top-view schematic for analyzing LIDAR scans of a cylindrical column is shown in Figure 3. In the schematic, the LIDAR is located at point L (cross marker) and the centerline of the cylinder passes through point O. The sample point *S* lies on the surface of the cylinder, which has a radius *R*. The point cloud is the locus of all visible points *S* along the cylinder's surface (green highlighted arc in figure).

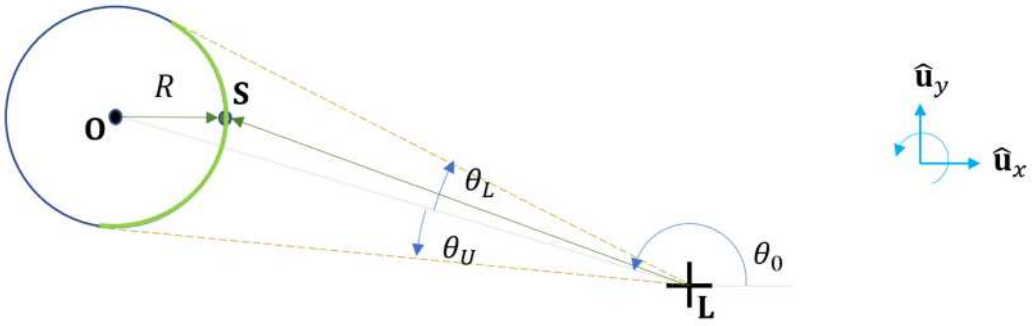

FIGURE 3
Top view of the cylindrical column

Our approach to compute the mean location of the point cloud will be to integrate the coordinates *x* and *y* along the appropriate cylindrical arc (between the brown-dashed lines in the figure). The *x* and *y* Cartesian coordinate pair are defined in by the unit vectors $\hat{\mathbf{u}}_x$ and $\hat{\mathbf{u}}_y$ in the diagram. Although the cylinder and point cloud are defined in 3D, we can compute the mean location for the point cloud using a 2D analysis, because the cylinder has a constant cross-section out of the plane. (Assuming points are evenly distributed in the vertical direction, the vertical component of the point-cloud mean lies at half the height of the cylinder.) Our primary goal is to compute the change in the observed point-cloud mean when viewing the cylinder from two different points L.

In computing the shift in the mean location of the point cloud, we consider not only missing data (as shown in Figure 1), but also point-density effects (as shown in Figure 2). To that end, we model the point cloud density ρ as uniform in the angular coordinate θ about the Lidar location L. The decision to model the density as a function θ (rather than, say, of the Cartesian coordinates *x* and *y*) automatically accounts for point-density variations along object surfaces as shown in Figure 2.

The number of points *N* sampled across the cylinder arc can be obtained by integrating the density. Defining the point L as the origin of a cylindrical coordinate system, the point O lies at (r_0, θ_0). The upper and lower bounds (brown dashed lines in Figure 3) are displaced from θ_0 by counterclockwise-positive angles θ_U and θ_L, noting $\theta_L = -\theta_U$ due the symmetry of the cylinder. Without loss of generality, we can redefine the circumferential angle relative to θ_0 (i.e., set $\theta_0 = 0$), and so

$$N = \int_{\theta_L}^{\theta_U} \rho(\theta) d\theta. \tag{1}$$

Noting the density is uniform, or constant, in the angular coordinate, we can factor out density and evaluate the integral to give

$$N = \rho(\theta_U - \theta_L). \tag{2}$$

The mean location over all the points in the point cloud, sampled at angles θ_i, is

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_i \mathbf{x}_S(\theta_i). \tag{3}$$

If the points are sampled with sufficient density, we can replace the summation with an integral:

$$\bar{\mathbf{x}} = \frac{\int_{\theta_L}^{\theta_U} \mathbf{x}_S(\theta)\rho(\theta)d\theta}{N}. \tag{4}$$

Substituting (2) into (4) and noting that the density can again be factored out of the integral, we can simplify to obtain

$$\bar{\mathbf{x}} = \frac{\int_{\theta_L}^{\theta_U} \mathbf{x}_S(\theta)d\theta}{\theta_U - \theta_L}. \tag{5}$$

The next step is to evaluate the integral in the numerator. Equation (5) uses a 2D vector notation, which will be important for the dual-wall corner; however, we can use a simpler scalar analysis for the cylinder, noting the geometry is symmetric about the axis $\overline{\mathrm{LO}}$. To locate the point-cloud mean for the cylinder, we need only solve for the component in the direction of $\overline{\mathrm{LO}}$ (the x-direction for $\theta_0 = 0$). Using the coordinate system aligned with $\overline{\mathrm{LO}}$, we thus have $\bar{\mathbf{x}} = [\bar{x} \quad 0]^T$. Substituting this expression and invoking symmetry, we convert the general vector equation (5) into a scalar equation for the cylinder:

$$\bar{x} = \frac{1}{\theta_U} \int_0^{\theta_U} \quad x_s(\theta)d\theta. \tag{6}$$

Using geometry and noting that edge $\overline{\mathrm{LO}}$ and $\overline{\mathrm{OS}}$ have lengths r_0 and R, respectively, we can obtain

$$(1 + \tan^2(\theta))\, x_s^2 - 2r_0 x_s + (r_0^2 - R^2) = 0. \tag{7}$$

Solving for x and invoking standard trigonometric identities, this expression can be solved for x_s to give

$$x_s(\theta) = r_0 \cos^2(\theta) - \cos(\theta)\sqrt{R^2 - \sin^2(\theta)\, r_0^2}. \tag{8}$$

Only the negative root of the quadratic equation is considered here, because the positive root corresponds to the far (hidden) side of the cylinder. The analytical solution for (6), eliminating x_s using (8), is

$$\bar{x} = \frac{r_0}{2} + \frac{1}{4\theta_U}\left(r_0 \sin(2\theta_U) - \sin(\theta_U)\sqrt{2r_0^2\cos(2\theta_U) - 2r_0^2 + 4R^2}\right) - \frac{R^2}{2\theta_U r_0}\tan^{-1}\left(\frac{\sqrt{2}\sin(\theta_U)}{\sqrt{\cos(2\theta_U) - 1 + 2R^2/r_0^2}}\right). \tag{9}$$

Drawing a right triangle connecting $\overline{\mathrm{LO}}$ to the tangent point from L, it is possible to show that the limit angle $\theta_U = \mathrm{asin}(R/r_0)$. Using the geometry of that right triangle, R and r_0 can largely be eliminated from equation (9) to give:

$$\bar{x} = \frac{r_0}{2}\left(1 + \frac{\sin(\theta_U)}{\theta_U}\left(\cos(\theta_U) - \frac{\pi}{2}\sin(\theta_U)\right)\right). \tag{10}$$

This value describes the distance from point L to the mean along the direction of $\overline{\mathrm{LO}}$. Subtracting (10) from the length of $\overline{\mathrm{LO}}$, which is r_0, and normalizing by the cylinder radius R, we can plot the point-cloud mean location relative to O, as shown in Figure 4. When the distance from the LIDAR to the cylinder center is at its minimum (e.g., when the LIDAR touches the cylinder, with $r_0 = R$), the LIDAR only sees one point on the cylinder, so the point-cloud mean is located one radial unit from the center (as shown on the far left of Figure 4). As the LIDAR moves increasingly away from the cylinder (moving right on the horizontal axis of Figure 4), the distance from the cylinder center to $\bar{x}$ shrinks, with the distance eventually reaching an asymptote (equal $\frac{\pi}{4}$ or approximately 0.79 radial units) as the LIDAR distance grows toward infinity. It is worth noting, however, that due to the finite sample density of real-world scanning lidar units, experimental performance at longer distances will not precisely match theoretical performance, as we have assumed dense sampling at all distances.

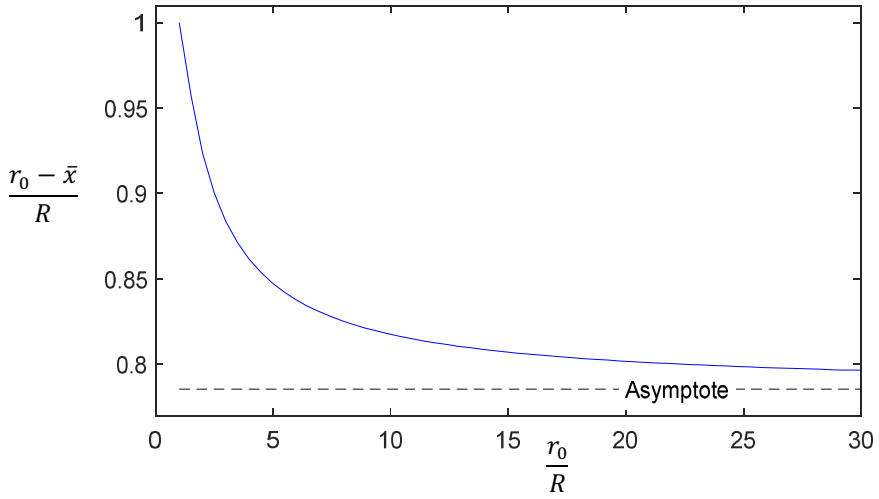

FIGURE 4
Distance from point O to point-cloud mean in the direction of $\overline{LO}$

If two LIDAR positions are considered, then we can compute the distance between the point-cloud mean vectors in each case. If a scan-matching algorithm aligns the two mean values, then this would be the systematic position-estimation bias for the voxel containing the cylinder. In differencing the point-cloud mean vectors, we must account for the different pointing direction of the vector (or equivalently, the distinct θ_0 value) for each location. Using the index *j* to distinguish the two LIDAR locations, we can write the vector from the point-cloud mean to the cylinder enter as:

$$\mathbf{x}_O - {}^{(j)}\mathbf{x}_s = \left({}^{(j)}r_0 - {}^{(j)}\bar{x}\right)\begin{bmatrix}\cos {}^{(j)}\theta_0 & \sin {}^{(j)}\theta_0\end{bmatrix}^T. \tag{11}$$

Using the cylinder center as a common reference, we can difference (11) for the two viewpoints *j* to get:

$${}^{(2)}\mathbf{x}_s - {}^{(1)}\mathbf{x}_s = \left({}^{(1)}r_0 - {}^{(1)}\bar{x}\right)\begin{bmatrix}\cos {}^{(1)}\theta_0 & \sin {}^{(1)}\theta_0\end{bmatrix}^T - \left({}^{(2)}r_0 - {}^{(2)}\bar{x}\right)\begin{bmatrix}\cos {}^{(2)}\theta_0 & \sin {}^{(2)}\theta_0\end{bmatrix}^T. \tag{12}$$

The false movement inferred will be opposite the change in the point-cloud mean location, so the perspective error for the inferred LIDAR translation is $\boldsymbol{\epsilon} = -\left({}^{(2)}\mathbf{x}_s - {}^{(1)}\mathbf{x}_s\right)$.

3.2 Perspective Error for a Dual-Wall Corner

For comparison with the smooth cylindrical column, we also consider a second object with a sharp bend: the dual-wall corner shown from above in Figure 5. The corner consists of two thin walls of length *R* that meet at vertex O. The LIDAR location is again labeled L. Each wall is oriented at a specified angle as viewed from above. Relative to $\overline{\text{LO}}$, the first wall is oriented at an angle ψ_1 and the second at ψ_2. Both angles are positive counterclockwise (so ψ_1 is negative as drawn).

The LIDAR detects points along the dual-wall corner, through a range of angles between a lower and an upper bound (θ_L and θ_U, respectively). The point cloud is generated from the entire visible surface, along both wall segments, each labeled with a subscript *i*. To determine the mean location for the point cloud, the first step is to write the location of an arbitrary point *S* on one of the segments. We use a coordinate system where the corner's vertex O lies on the x-axis ($\theta_0 = 0$ for integration), at a distance of r_0 from L. In this configuration the locus of points S on one wall segment can be described by the following vector, where *R* is the segment length, identical for both segments, and $f_i \in [0,1]$ is a fractional distance of S along segment *i*.

$$\mathbf{x}_s = \mathbf{x}_\text{O} + \begin{bmatrix} r_0 + f_i R\cos(\psi_i) \\ f_i R \sin(\psi_i) \end{bmatrix} \tag{13}$$

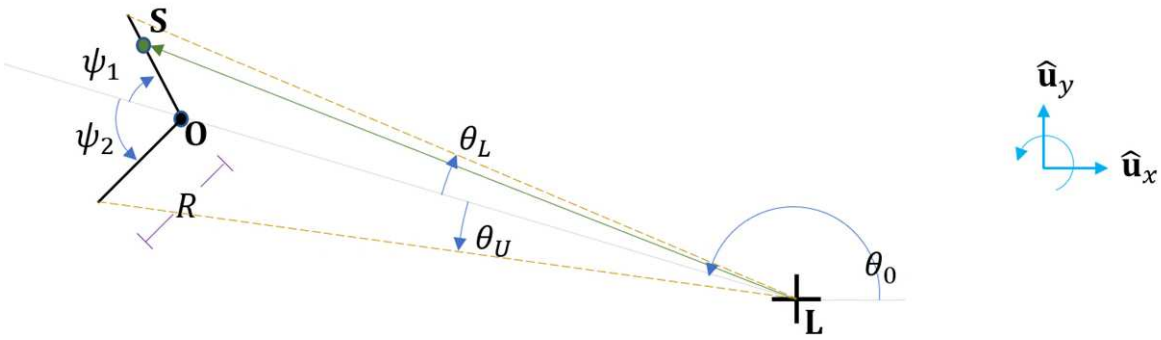

FIGURE 5
Top view of the dual-wall corner

The point-cloud mean $\bar{\mathbf{x}}$ can be computed from (5), where $\mathbf{x}_s$ is defined by (13). Before performing the integral, however, it is first necessary to rewrite (13) in terms of the angle θ. The angle θ describes the direction of the vector $\mathbf{x}_s$, so

$$\tan(\theta) = \frac{f_i R \sin(\psi_i)}{r_0 + f_i R \cos(\psi_i)}. \tag{14}$$

Solving (14) for the fractional length f_i and substituting the result into (13) gives

$$\mathbf{x}_s = \mathbf{x}_O + r_0 \frac{\tan(\theta_i)}{\sin(\psi_i) - \cos(\psi_i)\tan(\theta_i)} \mathbf{u}_i. \tag{15}$$

Here we define the vector $\mathbf{x}_O$ to describe the position of O relative to L and the unit vector $\mathbf{u}_i$ to describe the direction from O to S (along the segment). The integral of $\mathbf{x}_s$ along wall i depends only on the factor multiplying $\mathbf{u}_i$. For wall i, define this integral, over the set of angles $\theta_i \in \Theta_i$, to be

$$g_i = r_0 \int_{\Theta_i} \frac{\tan(\theta_i)}{\sin(\psi_i) - \cos(\psi_i)\tan(\theta_i)} d\theta. \tag{16}$$

We can now use (5) to obtain the mean for the point cloud associated with the visible surface of the corner, by substituting (15) and (16) into (5). Considering both wall segments and modeling the walls as thin, so there is no additional contribution from the wall ends, the integral (5) can be written as follows:

$$\bar{\mathbf{x}} = \frac{g_1 + g_2}{\theta_U - \theta_L} + \mathbf{x}_O. \tag{17}$$

In order to compare the shift in mean as viewed from two LIDAR locations, we need to introduce an index j to distinguish between the two LIDAR viewing locations. Introducing the location index via a leading superscript j, we can rewrite (17) as:

$${}^{(j)}\bar{\mathbf{x}} = \frac{{}^{(j)}g_1\,{}^{(j)}\mathbf{u}_1 + {}^{(j)}g_2\,{}^{(j)}\mathbf{u}_2}{{}^{(j)}\theta_U - {}^{(j)}\theta_L} + \mathbf{x}_O. \tag{18}$$

The change in the location of the point-cloud mean as viewed from the two lidar positions can be expressed by differencing (18) for each of the two viewing locations j. In computing this difference, it is important, to express the wall-tangent vectors ${}^{(j)}\mathbf{u}_i$ in a common coordinate system. To this end, we rotate from $\overline{LO}$-aligned coordinates back to the unit vectors $\hat{\mathbf{u}}_x$ and $\hat{\mathbf{u}}_y$ as shown in Figure 5. Using the latter set of basis vectors, we have

$${}^{(j)}\mathbf{u}_i = \begin{bmatrix} \cos\left({}^{(j)}\psi_i + {}^{(j)}\theta_0\right) \\ \sin\left({}^{(j)}\psi_i + {}^{(j)}\theta_0\right) \end{bmatrix} \tag{19}$$

With this definition, we can express the change in the point-cloud mean location as

$$ {}^{(2)}\mathbf{x}_s - {}^{(1)}\mathbf{x}_s = \frac{{}^{(2)}g_1\,{}^{(2)}\mathbf{u}_1 + {}^{(2)}g_2\,{}^{(2)}\mathbf{u}_2}{{}^{(2)}\theta_U - {}^{(2)}\theta_L} - \frac{{}^{(1)}g_1\,{}^{(1)}\mathbf{u}_1 + {}^{(1)}g_2\,{}^{(1)}\mathbf{u}_2}{{}^{(2)}\theta_U - {}^{(2)}\theta_L}. \tag{20} $$

Again, assuming the corner feature is fixed in space, the perspective error $\boldsymbol{\epsilon}$ will be opposite the apparent object motion, with $\boldsymbol{\epsilon} = -\left({}^{(2)}\mathbf{x}_s - {}^{(1)}\mathbf{x}_s\right)$.

In order to compute (20), we need to evaluate integral (16) for ${}^{(j)}g_i$. Before jumping to evaluating the integral with arbitrary units, let us first evaluate (16) on the range that includes point O, with ${}^{(j)}\Theta_i = \left\{\theta \in [0, {}^{(j)}\theta_i']\right\}$. For that case (16) gives the following result ${}^{(j)}g_i = {}^{(j)}h_i\left({}^{(j)}\theta_i'\right)$, with:

$$ {}^{(j)}h_i\left({}^{(j)}\theta_i'\right) = \begin{cases} -{}^{(j)}r_0\left(\left|{}^{(j)}\theta_i'\right|\cos\left({}^{(j)}\psi_i\right) + \sin\left(\left|{}^{(j)}\psi_i\right|\right)\ln\left(\frac{\sin\left({}^{(j)}\psi_i - {}^{(j)}\theta_i'\right)}{\sin\left({}^{(j)}\psi_i\right)}\right)\right) & {}^{(j)}\psi_i \neq 0 \\ 0 & {}^{(j)}\psi_i = 0 \end{cases}. \tag{21} $$

The absolute value notation in (21) is redundant when ${}^{(j)}\psi_i$ and ${}^{(j)}\theta_i'$ are positive; for the case when these variables (which always have matched sign) are negative, then introducing the absolute values signs gives the correct integral over the range of angles ${}^{(j)}\Theta_i = \left\{\theta \in [{}^{(j)}\theta_i', 0]\right\}$. With the absolute value signs, (21) describes the desired integral over a single wall. If only one of the two walls is visible, then ${}^{(j)}g_i = {}^{(j)}h_i\left({}^{(j)}\theta_i'\right)$ for that wall, where ${}^{(j)}\theta_i'$ is obtained from (14), by setting ${}^{(j)}f_i = 1$. We will refer to this angle associated with the wall endpoint as ${}^{(j)}\theta_{we,i}$:

$$ {}^{(j)}\theta_{we,i} = \operatorname{atan}\left(\frac{R\sin\left({}^{(j)}\psi_i\right)}{{}^{(j)}r_0 + R\cos\left({}^{(j)}\psi_i\right)}\right). \tag{22} $$

Additional consideration must be given to cases when both walls are visible or when one of the walls is partly visible behind the other. To cover these cases, additional logic must be introduced, as summarized by The logic in the table describes three possible outcomes. In the first case, if the two angles ${}^{(j)}\psi_1$ and ${}^{(j)}\psi_2$ do not have the same sign, then both walls are fully visible, and ${}^{(j)}g_i = {}^{(j)}h_i\left({}^{(j)}\theta_i'\right)$ for each wall. In the last two cases, the walls are on the same side, which causes the rear wall (index 2) to be fully or partially blocked from view by the nearer wall (index 1). In both cases the nearer wall analysis is straightforward, with ${}^{(j)}g_1 = \square(j)\square\, h1j\theta 1'$. In the case of full blockage, the second wall does not contribute to the integral: $\square(j)g2$=0. In the case of a partial blockage, the integral must be computed over just the visible section of the wall, with ${}^{(j)}g_2 = (j)\square h_2\left({}^{(j)}\theta_{we,2}\right)\square - {}^{(j)}h_2\left({}^{(j)}\theta_{we,1}\right)$.

4 NUMERICAL RESULTS

To study the perspective-error models derived in the previous section, it is helpful to evaluate the equations for a representative scenario. Let's consider the case of a LIDAR mounted on a mobile platform (e.g. a vehicle or robot) that moves in a straight line past a fixed object.

Table 1. In order to construct the table, we defined the wall angles ${}^{(j)}\psi_i$ as confined to the range ${}^{(j)}\psi_i \in (-pi, pi]$, where the zero angle is aligned with line segment $\overline{\mathrm{LO}}$ between the Lidar and the vertex O. Also, we assume the wall indices are ordered such that the wall nearer to the viewer (on either side) is indexed as 1 and the farther wall is indexed as 2. That is:

$$ \left|{}^{(j)}\psi_1\right| > \left|{}^{(j)}\psi_2\right|. \tag{23} $$

The logic in the table describes three possible outcomes. In the first case, if the two angles ${}^{(j)}\psi_1$ and ${}^{(j)}\psi_2$ do not have the same sign, then both walls are fully visible, and ${}^{(j)}g_i = {}^{(j)}h_i\left({}^{(j)}\theta_i'\right)$ for each wall. In the last two cases, the walls are on the same side, which causes the rear wall (index 2) to be fully or partially blocked from view by the nearer wall (index 1). In both cases the nearer wall analysis is straightforward, with ${}^{(j)}g_1 = {}^{(j)}h_1\left({}^{(j)}\theta_1'\right)$. In the case of full blockage, the second wall does not contribute to the integral: ${}^{(j)}g_2 = 0$. In the case of a partial blockage, the integral must be computed over just the visible section of the wall, with ${}^{(j)}g_2 = {}^{(j)}h_2\left({}^{(j)}\theta_{we,2}\right) - {}^{(j)}h_2\left({}^{(j)}\theta_{we,1}\right)$.

5 NUMERICAL RESULTS

To study the perspective-error models derived in the previous section, it is helpful to evaluate the equations for a representative scenario. Let's consider the case of a LIDAR mounted on a mobile platform (e.g. a vehicle or robot) that moves in a straight line past a fixed object.

TABLE 1
Defining integral for double-wall corner, for the case where the wall indexed 1 is closer than the wall indexed 2

Case	Condition	Integral for this condition
No blockage	$\text{sign}({}^{(j)}\psi_1) \neq \text{sign}({}^{(j)}\psi_2)$	${}^{(j)}g_1 = {}^{(j)}h_1\left({}^{(j)}\theta_{we,1}\right)$ ${}^{(j)}g_2 = {}^{(j)}h_2\left({}^{(j)}\theta_{we,2}\right)$ ${}^{(j)}\theta_U - {}^{(j)}\theta_L = \left\lvert {}^{(j)}\theta_{we,1} - {}^{(j)}\theta_{we,2}\right\rvert$
Blockage	$\text{sign}({}^{(j)}\psi_1) = \text{sign}({}^{(j)}\psi_2)$	
↳ Full blockage	Blockage and $\left\lvert {}^{(j)}\theta_{we,1}\right\rvert \geq \left\lvert {}^{(j)}\theta_{we,2}\right\rvert$	${}^{(j)}g_1 = {}^{(j)}h_1\left({}^{(j)}\theta_{we,1}\right)$ ${}^{(j)}g_2 = 0$ ${}^{(j)}\theta_U - {}^{(j)}\theta_L = \left\lvert {}^{(j)}\theta_{we,1}\right\rvert$
↳ Partial blockage	Blockage and $\left\lvert {}^{(j)}\theta_{we,1}\right\rvert < \left\lvert {}^{(j)}\theta_{we,2}\right\rvert$	${}^{(j)}g_1 = {}^{(j)}h_1\left({}^{(j)}\theta_{we,1}\right)$ ${}^{(j)}g_2 = {}^{(j)}h_2\left({}^{(j)}\theta_{we,2}\right) - {}^{(j)}h_2\left({}^{(j)}\theta_{we,1}\right)$ ${}^{(j)}\theta_U - {}^{(j)}\theta_L = \left\lvert {}^{(j)}\theta_{we,2}\right\rvert$

First, let's consider motion past the cylindrical column. We define the along-track coordinate to be zero when the LIDAR is closest to the object. Also, we assume that the LIDAR first detects the object when it is at an along-track distance of -15R (noting that our simulations express all distances in a nondimensional form, normalized by the column radius R). The systematic perspective error $\boldsymbol{\epsilon}$, associated with apparent motion of the point cloud generated from the cylinder, was computed as the negative of (12) and plotted in Figure 6. Errors were computed as changes in the perceived cylinder location, relative to its initial location at time zero. The starting point is on the left side of plots shown in Figure 6. When the LIDAR is closest to the object (at an along-track distance of $x/R = 0$), the distance from the cylinder center in the cross-track direction is set to one of three values: $y/R = \{1,2,4\}$. The first case, with $y = R$, is the limiting case where the LIDAR just touches the surface of the cylinder. In the subsequent two cases, the cylinder moves progressively farther off the LIDAR's track.

The figure plots along-track error ϵ_x and cross-track error ϵ_y as a function of the LIDAR's along-track position x. A circular cross-section is shown in the lower plot, as a reminder that the plot describes a cylinder shifted to the left (positive y direction) of the LIDAR. In all cases, the along-track error ϵ_x becomes increasingly negative as the LIDAR moves from left to right. The reason for this is that the point cloud rotates around the cylinder from its left (more negative) side to its right (more positive)

side. Because the point-cloud position moves to the right (positive), the vehicle measures less than the true distance of travel (a negative error in the along-track direction) if the cylindrical column is assumed stationary. The accumulated error ϵ_x/R is approximately $\pi/2$ (which follows since the point-cloud center moves from approximately $-\pi/4$ initially to $+\pi/4$ in the end, as expected for the limit case shown in Figure 4).

In contrast with the along-track error ϵ_x, whose magnitude grows in time, the cross-track error ϵ_y grows to a maximum at the moment that the object is viewed laterally (at $x/R = 0$). As the LIDAR travels past the object, the error fades back to zero. This trend makes sense. Initially, when viewed from the side, the cross-track position of the point cloud is near the cross-track position of the cylinder's center. As the LIDAR moves past the object, the point cloud shifts around the circular cross-section until it is centered on the side closest to the LIDAR at $x/R = 0$. This represents a negative apparent motion for the point cloud, so the vehicle appears to move positive ($\epsilon_y > 0$). Later, as the LIDAR moves increasing to the right, the point cloud continues to rotate around the circular cross section, toward the column's right side. This represents a positive change in the point cloud's *y* coordinate, back toward its original value (with the lateral error ϵ_y also returning to zero as the LIDAR moves off the right edge of Figure 6).

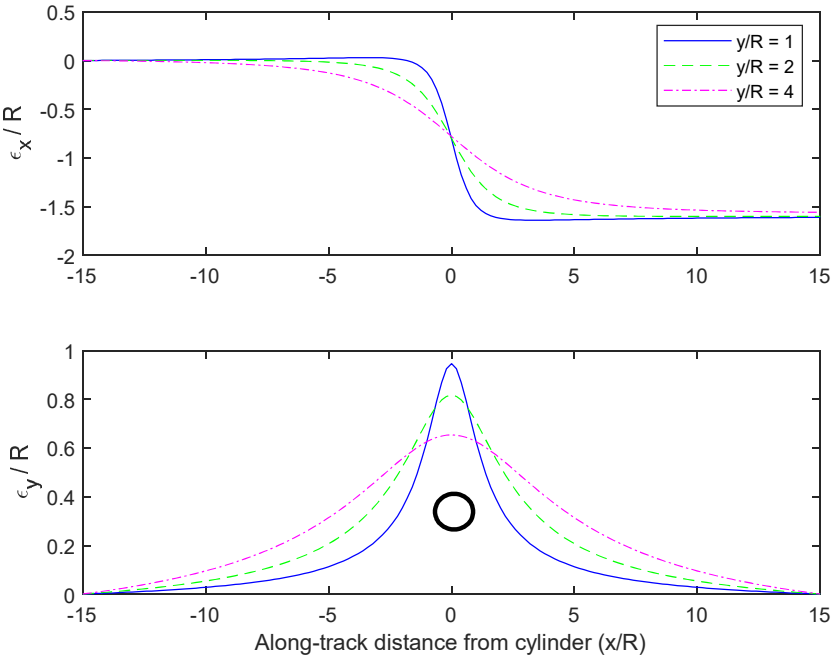

FIGURE 6
Perspective-error that results when a LIDAR moves on a straight trajectory past a cylindrical column

As the LIDAR trajectory is shifted increasingly far from the object (going from $y = R$ to $y = 4R$), the changes are more gradual. For instance, greater lateral spacing *y* causes the along-track error ϵ_x to accumulate sooner and the slope to be more gradual (but still resulting in roughly the same level of total error ϵ_x, as seen on the right side of Figure 6). As for the cross-track error, the error curve is wider (with error triggering sooner) and shorter (lower maximum lateral error) for higher values of *y*.

The trends observed when passing the dual-wall corner in the convex configuration are similar to the trends for the column (see left column of Figure 7, marked with "v-shaped" corner). In this configuration, the wall segments are each canted 45 degrees back from the LIDAR's line of travel. Only the outer faces of the wall segments are seen as the LIDAR passes. The along-track error ϵ_x increases in magnitude monotonically, as the point-cloud mean moves from the negative wall to the positive wall. The cross-track error ϵ_y increases and decreases, with a peak lateral error when the LIDAR is closest to the vertex (at $x/R = 0$). The only detail that is notably different between the cylinder and the convex dual-wall corner is the peak error-magnitude, noting that the objects have cross-sections of similar size (cylindrical column of radius *R* and dual-wall corner where each edge

is length R). For the convex corner in the left column of Figure 7, the largest along-track error is approximately $R/\sqrt{2}$ in magnitude (the distance between the midpoints of the two walls); the largest cross-track error is a very small fraction of R (less than $0.1R$), since both walls are entirely visible at $x/R = 0$. As the location of the walls shift farther laterally (vertex y/R going from 1 to 4), the trends again become more gradual.

The error profile looks somewhat different for other orientations of the dual-wall corner. Whereas the convex corner was a reasonable approximation of the cylinder, the L-shaped corner (middle column of Figure 7) and the concave corner (right column of Figure 7) exhibit curious properties related to occlusions of one wall by the other. All visible wall segments are considered in our analysis here; however, we acknowledge that the more distant wall segment might be excluded in a practical implementation, due to shadow-mitigation (McDermott & Rife 2022a).

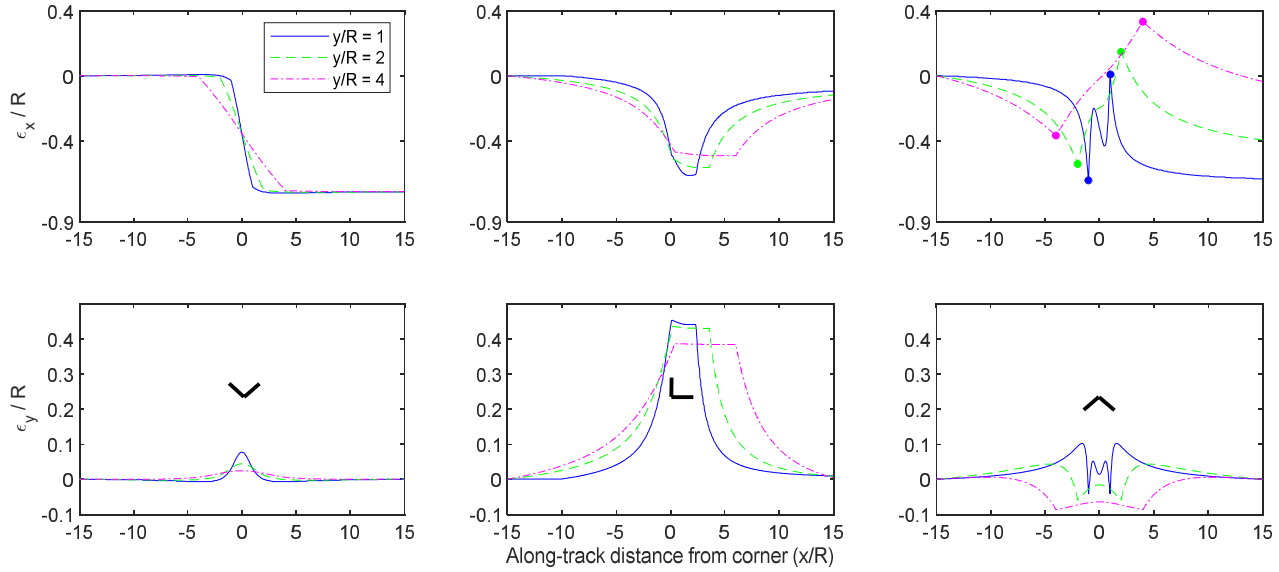

FIGURE 7
Perspective-error relative to starting position (at x/R = -15), as LIDAR moves in a straight line past a corner. Each column above shows a corner with a different orientation (as depicted in the lower row). In the left column, the corner opens away from the LIDAR (convex when viewed from x=0*); in the middle column, the corner opens to the positive x and y directions (flat when viewed from* x=0*); in the right column, the corner opens toward the LIDAR (concave when viewed from* x=0*).*

Consider the L-shaped corner (middle column of Figure 7). In this case, the corner has one wall aligned with the y-direction and the other aligned with the x-direction, with the vertex on the side closest to the LIDAR as it passes. Initially, the LIDAR primarily sees only the outside y-aligned wall. This wall disappears when the LIDAR reaches $x/R = 0$, at which point only the x-aligned wall is visible. Eventually the inside of the y-aligned wall becomes visible again and by the time the LIDAR reaches its final location ($x/R = 15$), this y-aligned wall dominates the point cloud. Since the LIDAR starts and finishes primarily seeing opposite sides of the y-aligned wall, which has no thickness, the net ϵ_x is nearly zero. As for ϵ_y, this error climbs to as high as $0.45R$ (for vertex at $y/R = 1$) when the LIDAR passes the corner. This makes sense since the LIDAR sees only the x-aligned wall at this point, which is closer than the y-aligned wall by a distance of $0.5R$.

Now consider the concave corner (left column of Figure 7). In this case, the corner opens toward the LIDAR, such that both walls are canted 45 degrees toward the LIDAR's track. As the LIDAR translates, it nearly always sees at least a fraction of both walls. Due to a combination of point density effects (see Figure 2) and switching between viewing the inner and outer faces of each wall, the point-cloud mean moves back and forth as the LIDAR progresses. As a result, the error ϵ_x and ϵ_y increases and decreases several times. Though a substantial ϵ_x error accumulates, the ϵ_y error generally remains in a small band near zero (smaller in magnitude than about $0.1R$). Interestingly, the sign of the net ϵ_x error is positive in the region where only the interior faces of the convex corner are visible. (Exterior faces are not visible between the dot markers, shown in the upper right plot of Figure 7). The positive net ϵ_x indicates overprediction of distance traveled. Thus, when only interior walls

are visible, the point cloud generally seems to move backward, which causes a perspective error that increases the LIDAR's apparent travel distance.

It is important to note that the dual-wall corner was modeled as a free-standing object, visible from all sides. If the concave corner were embedded in a larger wall (e.g. an inverted v-shape notched into a flat wall), then only the interior faces of the corner would be visible, much like the case with $y/R = 4$. By extension, the perspective error for the notched wall would also result in an overestimate of distance traveled. As a generalization, we can say that perspective errors on convex objects tend to cause underestimation of distance traveled, whereas perspective errors on concave objects tend to cause overestimation.

6 MITIGATION

Given that perspective errors can significantly bias LIDAR scan matching in some voxels, it is worthwhile mitigating these errors, to increase accuracy. This section considers several approaches for mitigating perspective error.

6.1 Voxel Dimension

Perhaps the most straightforward approach for mitigating perspective errors is to use smaller voxels. The impact of smaller voxels on perspective errors is illustrated in Figure 8. The figure shows four gridding arrangements superimposed on a cylindrical column. The analysis in the prior section applies when the grid cell is large enough to fully contain the cylinder; for that case, the point-cloud mean can fall on any point in an annulus (see Figure 8a). The annulus is bounded on the outside by the column outer boundary and on the inside by a ring at $\frac{\pi}{4}R$ (which is the asymptote of the point-cloud mean shown in Figure 4). Within the annulus, the specific location of the point-cloud mean depends on the location of the LIDAR. In the case when the LIDAR circles halfway around the column, near to the column surface, the point-cloud mean will shift by the column diameter (by a distance of $2R$). In the pathological worst case, the voxel boundary (dashed line in graphic) is closely matched to the size cylindrical column (Figure 8a), resulting in a perspective error that is equal in size to the voxel.

If the voxel edge length is cut in half, from $2R$ to R (as in Figure 8b), then the analysis of the prior sections no longer describes the point-cloud mean. However, we can still limit the point-cloud mean to the red area shown in Figure 8b by noting that the convex hull of the surface can be as large as a quarter circle and that the mean of points on that surface must lie within the convex hull (Boyd *et al*., 2004). The shortest dimension of the convex hull lies along the axis extending from the center of the circular cross section. It is straightforward to show that the depth of the convex hull in this radial direction is $\Delta_r = R(1 - \cos\phi_h)$ where ϕ_h is half the angle between the radial spokes extending from the circle center to the sharp endpoints of the convex hull. The width of the convex hull in the perpendicular, or circumferential, direction is $\Delta_\phi = 2R\sin\phi_h$. In the worst-case alignment of the cylinder relative to voxel boundaries (shown), the circular arc connects the voxel corners (as shown in Figure 8b), such that $\Delta_\phi = \sqrt{2}R$ (or $1.41R$) and $\Delta_\phi = 0.29R$. In short, halving voxel width reduces the worst error by $\sqrt{2}$.

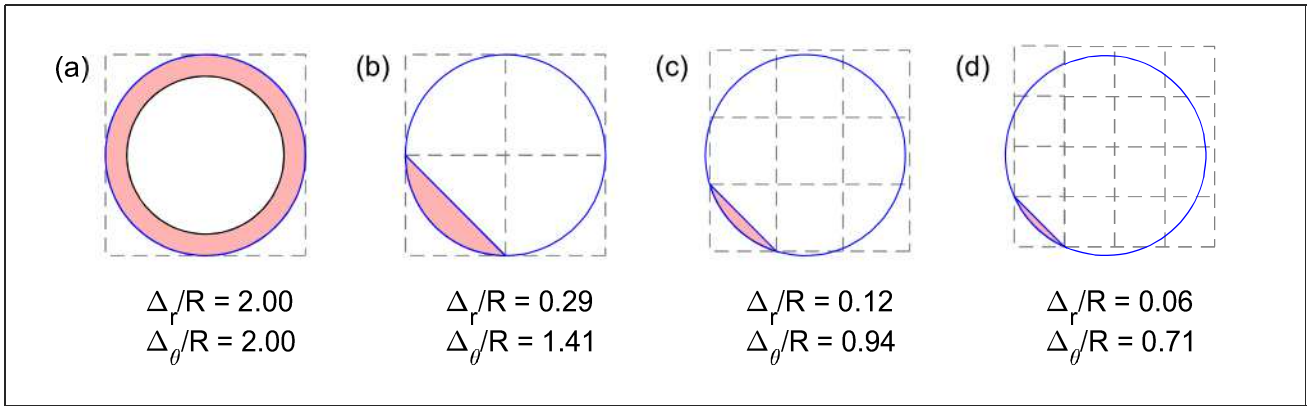

FIGURE 8
Effect of voxel size on worst-case perspective error.

This trend continues as the voxels grow even smaller. When there are three voxels across the diameter of the cylinder (edge width of $\frac{2}{3}R$) the worst case error is $\Delta_\phi = \frac{2\sqrt{2}}{3}R = 0.94R$. This case is shown in Figure 8c. When there are four voxels across the diameter of the cylinder (edge width of $\frac{1}{2}R$) the worst case error is $\Delta_\phi = \frac{\sqrt{2}}{2}R = 0.71R$. This case is shown in Figure 8d.

A secondary benefit of using smaller voxels is that fewer voxels align in the worst-case configuration as the voxels get smaller. In Figure 8b, it is possible for all four voxels to align in the worst case configuration. In Figure 8c, it is possible for only one of nine voxels shown to align in the worst-case configuration. In Figure 8d, it is possible for only one of sixteen voxels shown to align in the worst-case configuration. In short, smaller voxel size always decreases the magnitude of the worst-case error and, at least in the case shown, also decreases the number of voxels that experience the worst-case error. Although higher voxel count can mitigate perspective error, the extra voxels increase computational complexity; as such, adaptive resolution methods may offer benefits in balancing accuracy and computational cost (Eckart, 2018).

6.2 Exclusion of Lateral Features

Another possible mitigation is to avoid using nearby voxels orthogonal to the LIDAR velocity, at least if they contain a compact feature. This heuristic approach is driven by the observation that perspective errors accumulate most rapidly when a feature is located more-or-less orthogonal to the direction of motion (near x/R=0 as shown in Figure 6 and in Figure 7). As the figures show, perspective error is most pronounced when the feature is near to the LIDAR (when y/R is low).

In large part, the reason why perspective error accumulates rapidly is that the viewing angle, from a moving observer to a point O, changes quickly when the point is positioned orthogonal to the motion direction at a relatively short distance. We can see this by defining two position vectors, ${}^{(1)}\mathbf{x}_o$ and ${}^{(2)}\mathbf{x}_o$, which describe the vector to a point O from two viewing locations, labeled 1 and 2. The change in viewing angle ϕ is the angle between these vectors, which we can compute with the dot product:

$$\phi = \operatorname{acos}\left(\frac{{}^{(2)}\mathbf{x}_o \cdot {}^{(1)}\mathbf{x}_o}{\left\|{}^{(2)}\mathbf{x}_o\right\|\left\|{}^{(1)}\mathbf{x}_o\right\|}\right). \tag{24}$$

Let us assume that the motion carries the viewer a distance d is the direction aligned with the x-axis. Then the two vectors are related by ${}^{(2)}\mathbf{x}_o = \hat{\mathbf{u}}_x d + {}^{(1)}\mathbf{x}_o$. We can now use (24) to compute the viewing angle ϕ for points located at any position (x,y) relative to the midpoint of the two viewing locations. A contour plot of the viewing angle for objects at various locations is plotted in Figure 9, where the horizontal (x-axis) and vertical (y-axis) correspond to the position of the point O, and where the isocontours describe the angle shift (in degrees) for a small horizontal movement d. Note that the figure describes the location of point O in distances normalized by d, such that the figure axes are nondimensional.

The isocontours show that the largest changes in viewing angle (in degrees) are along the vertical axis through $x/R = 0$ (orthogonal to the motion) and nearer to the observer. The trends of large changes in viewing angle in Figure 9 are consistent with the trends of high perspective error observed in Figure 6 and Figure 7, supporting the correlation between the two.

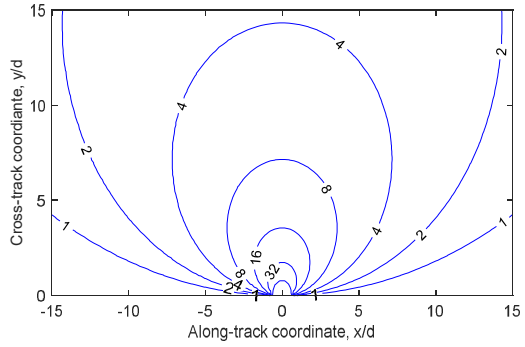

FIGURE 9
Change in viewing angle (deg) as a function of object location, for a movement in the x*-direction of distance* d

An important caveat is that perspective-errors only accumulate rapidly for surfaces that contain significant curvature within a cell. Changes in viewing angle allow the viewer to "see around the bend" for compact objects like cylinders and dual-wall corners, with motion revealing a new region of the object's surface. For flat planes, the full surface remains in view as long as the viewer stays in front of the plane; the result is that perspective errors can result only from density changes (see Figure 2), a relatively minor effect. Even for mildly curved surfaces, the error perpendicular to the surface is small (noting, for example, that the low values of the perpendicular error Δ_R/R shown for short circular arcs, as shown in Figure 8c and Figure 8d). By contrast, for corners and objects with large curvature within a voxel, the change in viewing angle reveals new portions of the object surface as the LIDAR moves.

The distinction between compact and extended surfaces is a major feature of the ICET algorithm, developed by McDermott (2022b). It may be useful to leverage this algorithm's ability to distinguish compact and extended features in order to develop a future strategy for mitigating perspective errors.

6.3 Monitoring for Perspective Change

Another possible mitigation for perspective errors is to introduce a monitor to check for significant changes to the distribution of LIDAR points within a given voxel. If the shape of the distribution changes, that might be a good indicator that a new region of a surface has appeared in view, a condition which is necessary for large perspective errors. The design of such a monitor is outside the scope of this paper.

6.4 Object Reconstruction

The basic tenet of scan matching is that commonality should be identified between two LIDAR scans, so that the scans can be aligned. In concept, it may be possible to extract even more information from a scan by classifying objects. If an object can be identified as the same feature in two scans, even if it is viewed from a radically different angle, then that feature could be used as a landmark for localization. There are two challenging problems here: reliably identifying landmarks and estimating their associated boundaries to register different views of the landmark (Zhou & Tuzel 2018; Li & Wang 2020; Xu *et al*. 2020). These problems are being studied in the machine-learning community, but developing a rigorous assurance case for ML-based algorithms remains a hurdle for safety-of-life applications.

CONCLUSION

The main goal of this paper was to characterize LIDAR perspective errors, a significant source of systematic error in LIDAR scan matching and odometry. In particular, we introduce an analytic model of perspective errors for voxel-based scan matching algorithms, like NDT and ICET, which attempt to match the mean of the point distribution within a voxel. We showed that the systematic effect of the perspective error can accumulate to a significant value, as large as the diameter of a cylindrical column, for instance. For convex objects (like columns, poles or outer corners) the perspective error underpredicts the LIDAR's forward motion. For concave objects (like inner corners), overprediction of forward motion is possible. Both convex and concave objects may result in the false inference of lateral motion, but these effects are transient and disappear as the LIDAR continues on its path. Concepts were discussed for mitigation of perspective errors, including the use of smaller voxels, lateral-feature exclusion, monitoring and object reconstruction.

ACKNOWLEDGMENTS

The authors wish to acknowledge and thank the U.S. Department of Transportation Joint Program Office (ITS JPO) and the Office of the Assistant Secretary for Research and Technology (OST-R) for sponsorship of this work. We also gratefully acknowledge NSF grant CNS-1836942, which supported specific aspects of this research. Opinions discussed here are those of the authors and do not necessarily represent those of the DOT, NSF, or other affiliated agencies.

REFERENCES

Besl, P. J., & McKay, N. D. (1992). Method for registration of 3-D shapes. In *Sensor fusion IV: control paradigms and data structures* (Vol. 1611, pp. 586-606). SPIE. https://doi.org/10.1117/12.57955

Biber, P., & Straßer, W. (2003). The normal distributions transform: A new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)* (Vol. 3, pp. 2743-2748). IEEE. https://doi.org/10.1109/IROS.2003.1249285

Boyd, S., Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Eckart, B., Kim, K., & Kautz, J. (2018). Hgmr: Hierarchical gaussian mixtures for adaptive 3d registration. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 705-721). https://doi.org/10.1007/978-3-030-01267-0_43

Hassani, A., Joerger, M., Arana, G. D., & Spenko, M. (2018). Lidar data association risk reduction, using tight integration with INS. In *Proceedings of the 31st International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2018)* (pp. 2467-2483). https://doi.org/10.33012/2018.15976

Hassani, A., & Joerger, M. (2021). A New Point-Cloud-Based LiDAR/IMU Localization Method with Uncertainty Evaluation. In *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021)* (pp. 636-651). https://doi.org/10.33012/2021.17905

Li, Q., Chen, S., Wang, C., Li, X., Wen, C., Cheng, M., & Li, J. (2019). Lo-net: Deep real-time lidar odometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8473-8482). https://doi.org/10.1109/CVPR.2019.00867

Li, Z., & Wang, N. (2020). DMLO: Deep matching lidar odometry. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS) (pp. 6010-6017). IEEE. https://doi.org/10.1109/IROS45743.2020.9341206

McDermott, M., & Rife, J. (2022a). Mitigating Shadows in Lidar Scan Matching using Spherical Voxels. *IEEE Robotics and Automation Letters*, 7(4), 12363-12370. https://doi.org/10.1109/LRA.2022.3216987

McDermott, M., & Rife, J. (2022b). Enhanced Laser-Scan Matching with Online Error Estimation for Highway and Tunnel Driving. In *Proceedings of the 2022 International Technical Meeting of The Institute of Navigation* (pp. 643-654). https://doi.org/10.33012/2022.18249

McDermott, M., & Rife, J. (2022c). ICET Online Accuracy Characterization for Geometric Based Laser Scan Matching of 3D Point Clouds. Submitted to *NAVIGATION*.

Plachetka, C., Fricke, J., Klingner, M., & Fingscheidt, T. (2021). DNN-Based Recognition of Pole-Like Objects in LiDAR Point Clouds. In *2021 IEEE International Intelligent Transportation Systems Conference* (ITSC) (pp. 2889-2896). IEEE. https://doi.org/10.1109/ITSC48978.2021.9564759

Shan, T., & Englot, B. (2018). Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 4758-4765). IEEE. https://doi.org/10.1109/IROS.2018.8594299

Stoyanov, T., Magnusson, M., Andreasson, H., & Lilienthal, A. J. (2012). Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations. *The International Journal of Robotics Research*, *31*(12), 1377-1393. https://doi.org/10.1177/0278364912460895

Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics*. MIT Press.

Velodyne LiDAR, Inc., (2019). *VLP-16 User's Manual*. https://velodynelidar.com/wp-content/uploads/2019/12/63-9243-Rev-E-VLP-16-User-Manual.pdf

Willers, O., Sudholt, S., Raafatnia, S., & Abrecht, S. (2020). Safety concerns and mitigation approaches regarding the use of deep learning in safety-critical perception tasks. In *International Conference on Computer Safety, Reliability, and Security* (pp. 336-350). Springer, Cham. https://doi.org/10.1007/978-3-030-55583-2_25

Xu, C., Wu, B., Wang, Z., Zhan, W., Vajda, P., Keutzer, K., & Tomizuka, M. (2020). Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. In *European Conference on Computer Vision* (pp. 1-19). Springer, Cham. https://doi.org/10.1007/978-3-030-58604-1_1

Xu, Q., Zhong, Y., & Neumann, U. (2022). Behind the curtain: Learning occluded shapes for 3D object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 3, pp. 2893-2901).

Zhang, J., & Singh, S. (2014). LOAM: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems* (Vol. 2, No. 9, pp. 1-9).

Zhou, Y., & Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3D object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4490-4499). https://doi.org/10.1109/CVPR.2018.00472