SpaceBeam: LiDAR-Driven One-Shot mmWave Beam Management

Timothy Woodford
University of California San Diego
San Diego, California, USA
twoodfor@ucsd.edu

Xinyu Zhang
University of California San Diego
San Diego, California, USA
xyzhang@ucsd.edu

Eugene Chai
NEC Laboratories America
Princeton, New Jersey, USA
eugene@nec-labs.com

Karthikeyan Sundaresan
NEC Laboratories America
Princeton, New Jersey, USA
karthiks@nec-labs.com

Amir Khojastepour
NEC Laboratories America
Princeton, New Jersey, USA
amir@nec-labs.com

ABSTRACT

mmWave 5G networks promise to enable a new generation of networked applications requiring a combination of high throughput and ultra-low latency. However, in practice, mmWave performance scales poorly for large numbers of users due to the significant overhead required to manage the highly-directional beams. We find that we can substantially reduce or eliminate this overhead by using out-of-band infrared measurements of the surrounding environment generated by a LiDAR sensor. To accomplish this, we develop a ray-tracing system that is robust to noise and other artifacts from the infrared sensor, create a method to estimate the reflection strength from sensor data, and finally apply this information to the multi-user beam selection process. We demonstrate that this approach reduces beam-selection overhead by over 95% in indoor multi-user scenarios, reducing network latency by over 80% and increasing throughput by over 2× in mobile scenarios.

CCS CONCEPTS

• **Hardware** → **Wireless devices**; **Sensor applications and deployments**; • **Human-centered computing** → *Ubiquitous and mobile computing systems and tools*; • **Networks** → *Wireless access points, base stations and infrastructure.*

ACM Reference Format:
Timothy Woodford, Xinyu Zhang, Eugene Chai, Karthikeyan Sundaresan, and Amir Khojastepour. 2021. SpaceBeam: LiDAR-Driven One-Shot mmWave Beam Management. In *The 19th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '21), June 24-July 2, 2021, Virtual, WI, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3458864.3466864

1 INTRODUCTION

Future 5G wireless networks will interconnect a massive, distributed set of edge sensors that seek to sense, understand and digitize our physical world, to realize the promise of a smart society [6, 11]. 5G networks have evolved to be cohesive integrated solutions that closely match innovative applications with customized networking features. And mmWave networks offering several Gbps of throughput at low latency are the key to unlocking these smart capabilities such as AR/VR remote control of robotic platforms, cloud-centric realtime AI for autonomous vehicles and high-resolution video. In particular, multi-user wireless VR/AR environments require high bandwidth, realtime data streams, and are a perfect use-case for multi-user mmWave networks [40].

In practice, mmWave performance scales poorly with increasing device densification envisioned for 5G, primarily because of the overhead of managing "pencil-like" mmWave beams. Candidate beams are probed sequentially during the search process. Hence, the larger the number of beams probed to achieve optimality, the greater the overhead. Furthermore, beam search is sensitive to interference and time coordination between neighboring mmWave devices is necessary to ensure interference-free channel measurements. As the density of mmWave networks increases [53], the untenable coordination overhead across increasing number of devices will limit the scalability of mmWave networks. Machine learning (ML) models have been used to reduce beam search overhead to some success [21, 47]. These blackbox solutions have proven to work well outdoors, *e.g.*, in V2X scenarios with well structured channel [45]. However, indoor environments are much less predictable, with a higher incidence of blockage, multipath, varying reflector types that make training such ML models more difficult. Other beam search approaches [29, 38, 42] reduce this overhead under specific conditions (*i.e.*, quasi-static environments) but do not eliminate it, and thus still encounter scalability challenges in larger networks comprised of many directional links.

These ML approaches, as well as several works on identifying mmWave reflectors [49, 51, 52], all allude to an important facet of mmWave beam search: If one can explicitly model the reflection environment and thus the candidate mmWave paths, beam management can become a *one-shot* process: we can simply pick the optimal beam from the set of known beams and their known propagation characteristics [8, 24, 32]. However, it is difficult to let mmWave radios "see" the environment due to their low spatial resolution [26]. Fortunately, LiDARs and depth cameras, are extremely efficient at this task, and can capture dense spatial details of the environment with higher accuracy than with in-band mmWave measurements. Such depth sensors are increasingly found in edge

MobiSys '21, June 24-July 2, 2021, Virtual, WI, USA

ACM ISBN 978-1-4503-8443-8/21/07.
https://doi.org/10.1145/3458864.3466864

sensors [25, 30, 44] and cellphones [5], and become instrumental for machine perception tasks such as object detection [46], scene understanding [50], *etc.* Inspired by this trend, we pose a fundamental research question: *can we leverage the visual 3D scene models constructed by LiDAR sensors to predict the mmWave channel and guide the beam selection?* Intuitively, we can apply RF ray-tracing over the visual 3D scene, but such a cross-domain modeling entails several non-trivial challenges:

Transformation from 3D maps to RF channel. 3D models built from depth sensors are plagued with noise, and result in erroneous ray tracing outcomes. Depth sensors, including RGB-D camera and high end, long-range LiDAR sensors [30] have systematic error on the order of centimeters (*e.g.*, ±1.1 cm in an Azure Kinect Camera[25], ±10 cm in an Ouster OS1 [30]). While this accuracy is sufficient for their use in autonomous platforms (*e.g.*, self-driving cars), RF modeling requires wavelength-level precision, *i.e.*, mm-level for mmWave channels. As an example, a wall that is flat and smooth in practice might be captured as a bumpy surface in the 3D model. The geometrical angles/lengths of ray traced mmWave signals reflected off such noisy 3D surfaces may deviate significantly from their real-world propagation paths.

Matching mmWave beam SNR from 3D maps. mmWave signal path loss consists of free-space attenuation plus reflection losses. While the former follows the classic Friis model, the latter depends on the "roughness" of the reflecting surface which is not represented in the 3D model. Furthermore, the reflection losses are frequency dependent [22, 34]: the reflection characteristics of mmWave signals do not necessarily match the optical signals. Hence, in order to utilize the 3D maps for accurate ray tracing, one needs to determine the mmWave reflection behavior. Note that this differs from the problem of object detection [46] as even if we can identify the object, we do not necessarily know the specular or diffuse reflection properties of its surface material [36].

1.1 Our Contribution

In this work, we aim to unify the environment sensing capabilities of LiDARs and the high bandwidth of mmWave networks with the question: *How do we transform 3D environment data into RF models to achieve one-shot mmWave beam selection in massive high density networks?*

We utilize LiDAR-equipped RGB-D cameras to construct a 3D model [7, 31] of the physical environment in realtime. This 3D model is then used to ray-trace the propagation paths of mmWave signals in the environment, so that optimal mmWave paths for each Tx/Rx pair can be identified. A *single construction* of the environment from any point-of-view is sufficient to facilitate beam management of mmWave devices throughout the mapped space. We call our solution SpaceBeam. SpaceBeam realizes several immediate benefits to mmWave beam management:

One-Shot Beam Assignment. SpaceBeam enables *one-shot* selection, where we select a single best beam per link using ray-tracing, without sequentially probing available beams. With sufficient edge-compute resources, the optimized ray-tracing algorithm of SpaceBeam can determine optimal beam assignments for any Tx/Rx pair with negligible latency. To further reduce beam assigment delays, SpaceBeam pre-computes beam assignments over a dense 3D grid of possible Tx/Rx positions within the 3D model. A beam assignment request is then fulfilled via a simple look-up table to find the solution of the closest precomputed point.

Scalable Concurrent Beam Management. SpaceBeam beam assignments do not require pairwise beam probing and coordination between mmWave links which may interfere with each other. Hence, beam assignments among nearby Tx/Rx pairs are looked-up concurrently, thereby eliminating the scalabilty bottleneck in large, dense mmWave networks.

While SpaceBeam uses off-the-shelf 3D scanning and reconstruction algorithms to build 3D models of the environment, it innovates solutions to address the following two challenges to enable one-shot mmWave beam selection:

C1: Mapping from 3D to RF. SpaceBeam employs a novel path averaging and pruning step to (a) increase ray-tracing tolerance to errors in Tx/Rx positioning and (b) reduce the impact of 3D scanning noise. It operates by first allowing for a small error tolerance in the angle of the path from the final reflection to the receiver, and then identifying erroneous paths by pruning paths based on the characteristics of their reflection points.

C2: Estimating mmWave path losses. SpaceBeam includes a robust approach to discriminate surface materials in the environment. RGB-D LiDAR cameras use an IR laser illuminator for ranging. In RGB-D cameras, the intensity of the IR transmitter is detectable by the visible light sensor. The specular and scattering properties of the surface scatter the impinging IR beams, resulting in an *intensity pattern* of IR light with varying bright and dark patches. The IR receiver on LiDAR systems [16, 25, 44] enables direct access to these IR intensity values. SpaceBeam uses a heuristic that maps these intensity patterns to estimated mmWave reflection losses of these surfaces with high accuracy.

We implement SpaceBeam using an Azure Kinect LiDAR camera [25] and a commercial 802.11ad radio [2]. Using the techniques that we develop, SpaceBeam can achieve an 88% reduction in latency and 18% increase in available throughput in a network with 4 users in a room over a state-of-the-art multi-user beam training algorithm [19]. Compared to the default 802.11ad interference management methods, SpaceBeam achieves a 66% reduction in latency and 50% increase in throughput. For a larger network with 12 users in a room, SpaceBeam improves throughput by 115% and reduces latency by 86%.

The main contributions of SpaceBeam can be summarized as follows.

(i) We propose a novel framework to transform visual/optical 3D models into mmWave channel profiles. Our framework bridges the gaps between these traditionally orthogonal domains, through a set of mechanisms that contain the geometrical mismatch, and through a cross-domain intensity estimation scheme that masks the discrepancy in reflection properties.

(ii) We apply the SpaceBeam model to capture the invariant environment-dependent structures in the mmWave channel, leading to a one-short beam assignment scheme which can use a pre-computed look-up table to select the best beams for all mmWave links in the environment. We further introduce mechanisms to adapt to environmental dynamics.

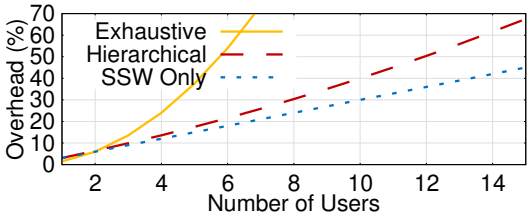

Figure 1: Comparison of exhaustive beam measurements to a hierarchical training method (using BRP) and measurements using only a sector sweep. Overhead is the percentage of channel time dedicated to beam retraining for an indoor environment where users move at a speed of 2 m/s.

(iii) We implement SpaceBeam using commercial mobile LiDAR and mmWave radio platforms, and verify its advantages in comparison with standard protocols and state-of-the-art multi-user mmWave beam assignment solutions.

2 MOTIVATION

2.1 Scalability Limits of mmWave Beam Selection

Current approaches to mmWave beam selection rely almost exclusively on direct measurements of RSS. Due to the directionality of mmWave beams and sparsity of the mmWave channel, these approaches must frequently re-scan beams to avert significant performance drop. The scanning overhead grows as a product of the number of users/links, node moving speed, and number of MIMO arrays per node, and can even overwhelm the useful payload.

In Figure 1, we evaluate the overhead required for beam training with a variable number of mobile users, using a radio with multiple phased array panels for 360° coverage. Using sector sweeps alone requires 3 ms per user to select from among multiple panels. Hierarchical search starts with a sector sweep, then refines its beam selections in a pairwise manner. Exhaustive beam training is clearly infeasible in this situation, but even more-efficient methods clearly struggle as the number of users increases.

2.2 Transforming Models from 3D Optical Space to RF Space

The reliability of 3D maps in mmWave beam selection lies in the ability of SpaceBeam to transform 3D environment models into mmWave channel models that reflect the propagation angles/strengths of signal paths.

2.2.1 The Impact of Imperfect 3D Models. The accuracy of ray-tracing mmWave signal paths depends on the quality of the 3D copy of the physical space. Typically, RF path-tracing requires a 3D CAD model as input, consisting of large, exact faces, such that it can locate an exact point where any given reflection or transmission occurs. In practice, given the limited accuracy of LiDAR scanners, a perfect replica of the physical world is not possible, and the deficiencies have a large impact on ray-tracing performance.

3D modeling commonly adopts mesh representation, which approximately constructs 3D shape polygons. Each polygon is defined by its 3D position boundaries and a surface normal. Consider a flat wall as shown in Figure 2(a). The best 3D mesh reconstruction of this wall from LiDAR measurements shows significant noise and artifacts, as seen in Figure 2(b). Ray traced mmWave beams reflecting off such a noisy surface will diverge significantly from their real-world paths, and be inadvertently dropped from consideration. For example, Figure 3 shows that when range error is a mere 1 cm or greater, fewer than 50% of the candidate mmWave paths that are discovered by ray tracing. Even very recent 3D reconstruction frameworks such as Kimera [35] generate meshes with cm-level or greater mean-squared errors.

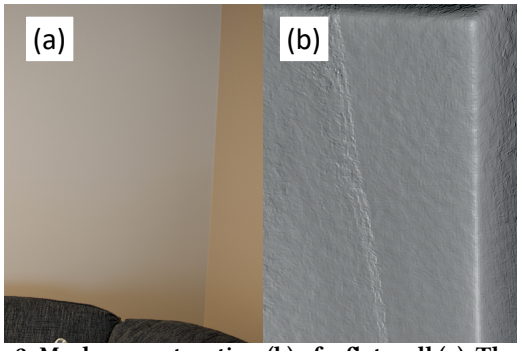

Figure 2: Mesh reconstruction (b) of a flat wall (a). The mesh reconstruction includes an artifact due to systemic error (the diagonal line) and smaller random errors which appear as a texture on a flat surface.

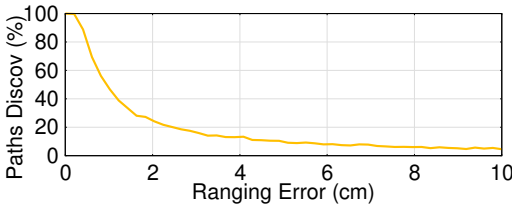

Figure 3: The percentage of paths discovered using a standard ray-tracing algorithm as a function of the ranging error of the sensor.

2.2.2 Estimating mmWave Beam SNR. Additionally, different objects in the scene have different mmWave reflective properties. Direct estimation of reflection coefficients using 3D mapping data is challenging because the visible and near-infrared sensing data cannot fully represent the mmWave interaction properties of an object. For example, dry/wet concrete [22] and multi-pane/single-pane windows [43], though hardly distinguishable by LiDAR, may reflect/attenuate the mmWave signals differently.

As an example, in §7.2 we show that by identifying reflection coefficients of materials, SpaceBeam can select mmWave beams with higher SNR on average than if we had no information on material reflectivity.

3 SYSTEM OVERVIEW

SpaceBeam is a LiDAR-driven mmWave beam selection solution that operates in three phases.

Building 3D Maps for RF Modeling: SpaceBeam first generates a 3D map of the surrounding physical environment and captures the characteristics of the physical materials to better inform the ray-tracer in the later step. SpaceBeam scans the environment

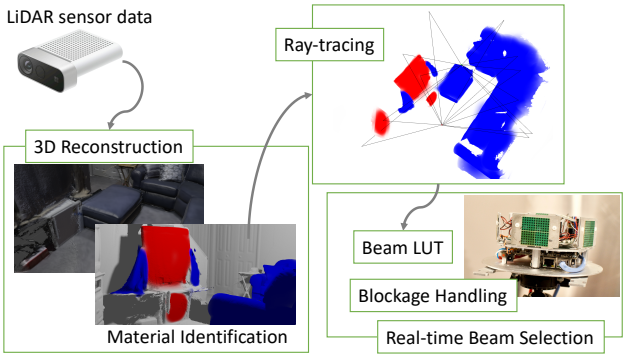

Figure 4: SpaceBeam consists of three components: 3D reconstruction, robust RF ray-tracing, and real-time beam selection. The radio shown in the real-time beam selection block was used in our testbed.

with a LiDAR-equipped RGB-D camera [25]. Since commercial LiDAR sensors may have a ranging error variance of up to 3 cm [44], we use a robust reconstruction method [7, 31] incorporating Truncated Signed Distance Function (TSDF) mesh reconstruction. Centimeter-level accuracy is typical for many commercially available LiDAR sensors, including those manufactured by Velodyne [44], Microsoft [25], and Intel [16]. Concurrently, SpaceBeam uses a novel method to estimate the reflection characteristics of mmWave signals by measuring environmental materials' near-IR scattering characteristics with the LiDAR camera.

RF Ray Tracing: The 3D map contains reconstruction noise due to the limited accuracy (±1cm) of the RGB-D camera [25]. SpaceBeam uses a customized ray-tracing algorithm that works effectively on the noisy 3D map generated in the first phase, we develop an algorithm that can identify real RF paths in a 3D mesh, and reject false reflection paths caused by reconstruction noise.

Real-Time Beam Selection: Finally, SpaceBeam uses the outcomes of ray-tracing to select the best beams for each Tx/Rx location. Given a static (or quasi-static) environment, SpaceBeam can precompute beam assignments for each location in a 3D grid of points that spans the entire 3D map. These grid points are spaced 28cm apart, which we show well balances the trade-off between computation overhead and beam accuracy (§6.1). Most environments include some amount of mobility. In particular, people moving in the environment tend to create significant blockages. We handle these environment dynamics in real time by detecting unexpectedly low SNR and switching to an alternate precomputed beam. For this reason, *LiDAR sensors are not required during the ordinary operation of the system.* Any localization method with cm-level accuracy may be used, such as a VR headset's position-tracking system. When significant changes in the quasi-static environment occur, we can recompute the complete lookup table for the environment in under 15 minutes.

4 3D RECONSTRUCTION OF THE ENVIRONMENT

Broadly, any 3D reconstruction method proceeds by first aligning the sequence of ranging data based on an estimation of the sensor *pose* at each sampling time stamp. Once an optimal alignment is calculated, the separate measurement samples can be *integrated* into a single 3D mesh model. The 3D sensor used in SpaceBeam is a hand-held RGBD (RGB + depth) camera with LiDAR ranging capabilities, which is becoming popular in consumer grade mobile devices. To reconstruct the 3D model, we use an algorithm [7, 31] that leverages the RGB data to improve the sample alignment. The process first aligns adjacent images to form *fragments*, and then reconstructs the global map by merging these fragments. To join the depth images into a single, global image, we first construct a voxel cloud of size 5 mm. Then we use the KinectFusion method to find the location of the surfaces based on the volumetric truncated signed distance function (TSDF), while reducing the effect of depth noise in individual images [27]. The process results in a relatively uniform mesh, where the spacing between vertices is about 5 mm, and the sizes of the triangles are approximately uniform.

Once the locations of surfaces are obtained, we must determine the propagation properties of the surface materials, *i.e.*, the reflection and transmission coefficients. Although inferring exact reflection coefficients from optical sensor data is difficult, we observe that even coarse reflection estimation can help to guide the mmWave beam selection. In fact, it is well known that the mmWave channel is sparse in both indoor and outdoor environment–between a pair of transmitter and receiver, only the LoS and 3-5 dominant NLoS reflection paths determine the angle and strength of signals [33, 39, 41]. Therefore, for beam selection, the primary purpose of material identification is to discriminate objects with significant reflection loss, which will diminish the NLoS path strength. The secondary purpose is to rank the remaining objects and find the strongest available reflectors.

4.1 A Primer on RF Reflection Loss

Many prior studies have shown that common building materials and indoor objects have found widely-varying reflection loss [22, 24, 36, 43] at mmWave frequencies. Therefore, given a 3D mesh that embodies the locations and orientations of environmental surfaces, SpaceBeam must determine the reflection loss of a signal that reflects from each surface.

To understand how SpaceBeam resolves the issue, we first review the physical interactions that determine the reflection properties of a surface. When a radio signal reaches a perfectly flat surface, the signal power partly penetrates the surface, and partly reflects along a specular direction. The relation between the transmitted and reflected power is given by the Fresnel coefficients, which depend on the complex permittivity of the surface material. In reality, most surfaces are not perfectly flat, and the signal will experience *diffuse scattering* which further reduces its strength. For example, a rough plaster surface can have more than 20 dB greater reflection loss than a smooth plaster surface. Such loss due to surface roughness can be modeled as [22]

$$R/R_0 = \exp\left[-8\left(\frac{\pi \Delta h \cos\theta_i}{\lambda_0}\right)^2\right], \quad (1)$$

where Δh is the standard deviation of the surface height, and R_0 is the reflection coefficient calculated using the Fresnel equations. Therefore, for a given surface, the diffuse scattering of IR is greater than or equal to the diffuse scattering of mmWave signals.

4.2 Sensing Surface Roughness using IR Measurements

Clearly, it is not trivial to calculate the Fresnel coefficients, since the relevant electrical characteristics vary with frequency. It is possible to coarsely infer the values by using an image classification algorithm, but this requires intensive training over a large database of object types, together with environment-specific tuning.

To overcome the challenge, SpaceBeam instead models the frequency independent surface scattering properties based on the LiDAR camera data. Unfortunately, the depth resolution of such sensors is insufficient to reveal the sub-mm level surface variation. Instead, we can indirectly infer surface roughness by examining the surface backscatter characteristics. This is accomplished using IR intensity data captured by typical LiDAR devices, including the CMOS sensors found in the Azure Kinect, iPhone Pro, as well as more traditional mechanical LiDARs (*e.g.*, Velodyne HDL-32E). Intuitively, the ratio between the total reflected IR signal power to the backscattered signal power will allow us to infer the surface height variation, which determines the specular reflection loss due to diffuse scattering.

Since the IR camera is measuring returns from an onboard IR laser, a specular reflection exists at the point on the surface where the normal of the surface points towards the camera. We compare the IR luminosity at this point to the luminosity at nearby points on the surface to find the ratio between the specular reflection point, which includes both forward- and back-scatter, and surround points, which only include backscattered light. We extract a 100×100 pixel square centered around this point, and then apply a Gaussian blur to this image to reduce the effect of macro-scale surface roughness and other small details on the object. We further calculate the specularity ratio by finding the maximum intensity within the square (the specular reflection point), and then dividing by the average intensity across the entire image. To further reduce the effects of noise, we average this ratio over 10 consecutive frames, or less than half a second of time.

As a feasibility verification, we use our mmWave radios to measure the reflection loss of a number of representative materials, while using the LiDAR camera to desirve the IR specularity ratios. Table 1 summarizes the results. Clearly, surfaces with higher IR specularity tend to have lower mmWave reflection losses. However, we found that there is no exact mapping between mmWave roughness and IR roughness, because IR backscatter may also capture the surface variation too minor to have an effect at mmWave frequencies. Instead, we approximate the surface roughness effect by *quantizing* it into three groups: highly specular, non-specular, and partially specular, corresponding to objects with IR specularity > 20 dB, < 4 dB, and in between, respectively. Although this method is coarse, and only provides a lower bound on reflection loss, we will show later that it can already substantially improve beam selection performance (§ 7.2).

5 ROBUST RAYTRACING WITH NOISY 3D MODELS

To enable accurate mmWave ray-tracing, we need to account for the mismatch between the dense, noisy mesh generated by LiDAR sensing, and standard RF ray-tracing algorithms which expect smooth

Table 1: Reflection loss values for different objects. Where available, our measurements are compared to prior measurement studies (in parenthesis). IR Specularity is our estimate of the ratio of the total reflected power to the backscattered power at 850 nm.

Material Type	Reflection Loss	IR Spec
Refrigerator	<1 dB	38.3-40.5 dB
Car Body	<1 dB	33.3-34.4 dB
Window	4 dB (2-8 dB[22, 43])	25-26 dB
Drywall	14 dB (9-15[24, 43])	4.6-6 dB
Cinderblock	22 dB (7-11 dB[24])	3.8-4.3 dB
Couch (Fabric)	30-40 dB	3.4-3.6 dB
Stucco Wall	30 dB (25-35 dB[22])	3.3 - 3.9 dB

noiseless surfaces. A straightforward solution would be to create a mesh reconstruction that is as close to the expected CAD-like format as possible. To this end, we can generate a point cloud from our 3D model, then repeatedly run the RANSAC algorithm [12] to find subsets of points that form planes. For each subset, we first create a plane by finding a normal vector and location that best fits the subset. We then divide this plane into 10 cm squares, and remove the squares that are not supported by any points. By removing unsupported squares, we avoid unintentionally blocking open doorways and similar areas that lie behind part of a planar surface. This technique can mitigate the mesh noise. Accordingly, we find that the accuracy of standard RF path tracing improves significantly in an environment consisting primarily of large, flat surfaces. Unfortunately, this method is not applicable in a more crowded environment with many non-flat objects. Our experiments show that the proportion of missing paths nearly doubles in a room that contains furniture and other objects (Section 5.4).

To overcome the challenge, we instead modify the ray tracing process itself to be robust to significant noise in the 3D model. To capture the missed paths, we introduce a small tolerance in reflection angle based on the measured error characteristics of our 3D reconstruction system. On the other hand, our 3D model will include many surfaces too small to produce a reflection strong enough for use in communications, and the model may include small reconstruction artifacts. We must find a way to reject the weak and spurious reflection paths from these surfaces. We develop a pair of simple tests that, when used together, detect and remove the vast majority of such paths. We now elaborate on these mechanisms.

5.1 Approximate Path Generation

Classical RF modeling methods such as Shooting and Bouncing Rays (SBR) [23] use ray tracing to find the primary propagation paths in an environment. In general, SBR works by initializing a set of rays with a uniform spacing of $\Delta\theta_{ray}$ in the elevation and azimuth directions. For each ray, it then finds the first location where the ray intersects another object in the environment. If this first intersection is located at the receiver, then the SBR algorithm terminates for that ray. Otherwise, it creates a new ray in the direction of the specular reflection $\hat{r}_s = 2(\hat{r}_i \cdot \hat{n})\hat{n} - \hat{r}_i$, where $\hat{r}_i$ is the unit vector indicating the direction of the incident wave. It may also create a refracted wave that passes through the surface. For each new ray, it proceeds to recursively find a path to the receiver, unless a predetermined limit on the number of reflections and refractions is reached.

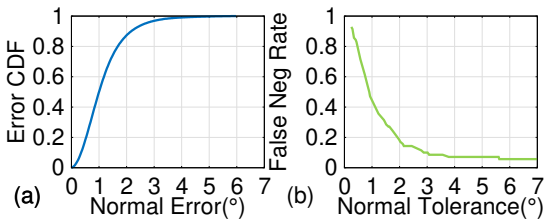

Figure 5: (a) The variation in normal vectors measured using our calibration procedure, and (b) the probability of missing a path as a function of the normal tolerance $\Delta\theta_{\hat{n}}$.

The immediate problem that we face with the conventional RF raytracers is that they assume for each propagation path m, there exists some set of rays $\{\vec{r}_n\}_m$, that lead from the transmitter to the *exact location* of the receiver, and that each of the rays point in the direction of the specular reflection. In a dense triangle mesh model of the environment where the normal vector of each face may contain a substantial amount of noise, such a set of rays frequently does not exist. To accommodate mesh noise, we introduce an error tolerance of $\Delta\theta_{\hat{n}}$ for the angle of the normal vector used to calculate the direction of the specular reflection. If the final ray to the receiver has an angle that falls within $\frac{1}{2}\Delta\theta_{\hat{n}}$, the ray is counted as having arrived at the receiver.

5.1.1 Selecting the Error Tolerance. A key question is how to select the tolerance parameter $\Delta\theta_{\hat{n}}$. An excessively large value leads to the detection of many false paths, requiring increasing processing complexity to correct. On the other hand, a value that is too small may lead to miss detection of valid RF paths.

Therefore, we want to select the minimum $\Delta\theta_{\hat{n}}$ such that we miss a path due to normal noise in fewer than 1% of all cases. In conventional mesh reconstruction algorithms, errors in positions of vertices in the mesh are a result of errors in both range estimation and pose estimation of the 3D sensor. Due to the large number of factors involved in 3D mesh reconstruction, it is generally not possible to find mesh error probabilities analytically. Instead, prior work determined mesh error characteristics using a simple calibration procedure involving measuring a flat surface from multiple distances and running the reconstruction algorithm on each measurement [18, 28]. Using this procedure, we find the differences between the normals of individual triangles and the reference normal value, and use this distribution to estimate the prevalence of missed paths at a given $\Delta\theta_{\hat{n}}$ value. Since mesh error depends on the system configuration, and not on the environment, this calibration only needs to be completed once per combination of sensor, measurement platform, and reconstruction method.

Figure 5 shows the distribution of errors in the normal vector, alongside the probability of miss-detecting an RF path. The calibration scenario finds that 99% of the normal errors fall below 4°. At 4°, the rate of missed reflected paths has converged to a small value of 4%. The remaining paths errors have other causes, *e.g.* mmWave reflectors hidden behind objects that are transparent at mmWave but not infrared.

5.2 Path Clustering

The aforementioned method finds missing paths neglected by exact path discovery methods, but it also finds many additional reflection points near the exact reflection point, since the increased error tolerance in the reflection angle will allow a larger number of the initial rays to reach the receiver.

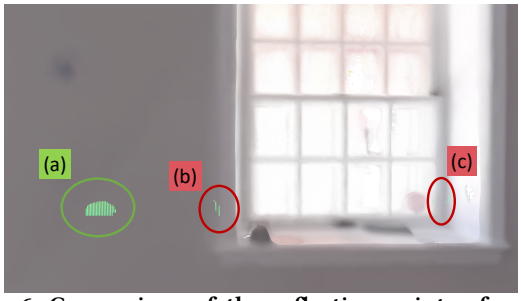

Figure 6: Comparison of the reflection points of a spurious path to the reflection points of a valid path. Reflection points (a) correspond with a valid path with reflects off of a flat wall. Reflection points (b) and (c) reflect off of small portions of a window frame.

To consolidate these paths into a single path, we first group the paths into clusters. We use hierarchical agglomerative clustering with a single-linkage criteria to identify the clusters in $O(n^2)$ time, where n is the total number of paths [37]. The single-linkage criteria successively merges any paths within a given distance of any existing path in the cluster. Since the set of mmWave paths is sparse, clusters are nearly always far apart and the single-linkage threshold does not need to be determined precisely. We then take the mean of each reflection point in the path cluster to merge each cluster into a single path.

5.3 Spurious Path Pruning

Whereas the aforementioned mechanism dealt with many near-duplicate reflection points corresponding to a single path, as in Figure 6(a), in this section we consider *spurious paths* reflecting off very small surfaces such as the edge of a table or LCD panel, or reflecting off small artifacts caused by 3D reconstruction errors. Two examples of spurious paths are shown in Figure 6(b,c). The reflected mmWave power should be small as it is proportional to the surface area [17]. The prior results of ray-tracing find paths off both large surfaces and very small surfaces, so we introduce an additional step to differentiate between these two types of reflections.

Since a detailed mesh reconstruction consists of many small triangles, even for a large, flat surface, it is usually not possible to directly measure the size of the face causing the reflection. Instead, to eliminate these spurious paths, we leverage two geometric intuitions about the nature of the ray-tracing results. First, our ray-tracing model assumes all interactions occur in the far field, such that there is a coherent combination of signals from all the antenna elements across the phased array at each reflection point. If this assumption is not satisfied, the reflection may not be strong enough to use for communications. To verify that the signals coherently combine at each reflecting point, we select six points around the edges of the Tx/Rx phased arrays (which are 4×4 cm in our case), and run ray tracing using these pairs of Tx/Rx locations. We use the same clustering method discussed above to associate the paths discovered across these multiple raytracing rounds. Then, if a cluster is present in all of the Tx/Rx location pairs, we accept it as a valid path. Otherwise, we reject it as spurious.

Table 2: Comparison of spurious path rejection methods. False Pos Rate is the fraction of spurious paths accepted by the classifier and False Neg Rate is the fraction of good paths rejected by it.

Method	False Pos Rate	False Neg Rate
Tx/Rx Rand	23%	1%
Face Count	9%	1%
Combined	9%	1%

Table 3: Comparison of false negative rates using different RF ray tracing techniques in three different rooms.

Raytracing Method	Simple	Crowded	Garage
Wireless InSite	56%	62%	80%
Plane Fitting	10%	20%	68%
RT Approximation	5%	7%	11%
RT Approx + Pruning	5%	7%	11%

Second, recall that increasing the angle tolerance $\Delta\theta_{\hat{n}}$ will lead to a roughly circular disc-shaped set of reflection points on a large, flat surface. Spurious paths will be confined to either a single point or a line of points, as shown in Figure 6. To distinguish between small and large reflection surfaces, we want to determine the total area of the reflection surface. Directly determining the total reflection surface across all ray traced paths using a method such as finding the convex hull is inefficient for large numbers of paths. Instead, we observe that the size of the faces in our mesh will be roughly uniform as a result of the reconstruction process described in § 3. Thus, we can use the number of faces detected as reflectors as a proxy for the area of the reflection surface.

To this end, we need to find a threshold for the minimum number of faces $N_{f,min}$ needed for claiming a a sufficiently large reflecting face. First, we note that the any flat reflecting surface should have an area at least as large as the phased array (*e.g.*, 16cm^2 in our implementation). To calculate the probability distribution of the number of reflection points corresponding to a single path, we begin by finding an approximation for the change in incidence angle as a function of the distance from the ideal reflection point. For a small distance Δx, the incidence angle will change by $\Delta\theta_i = \frac{180}{\pi}\left(\tan^2\theta_i + 1\right)^{-1}$. For any mesh triangle a distance r from the ideal reflecting point on a flat surface, the probability of the normal vector falling within the range of a specular reflection is

$$P_{match} = P\left[\theta_{err} < \Delta\theta_{\hat{n}} - \frac{360}{\pi}\frac{r}{1+\tan^2(\theta_i)}\right]. \quad (2)$$

Using the empirical normal error CDF found using our calibration procedure in Sec 5.1.1, we compute that there is a 99% chance that a flat reflector with an area of 16cm^2 will have at least 30 triangles detected as reflectors during ray tracing, given that vertices are all spaced approximately 5 mm apart. Any cluster of reflections with fewer than 30 triangles is highly unlikely to be a flat surface of sufficient size to produce a strong reflection.

Table 4: Comparison of false positive rate (percentage of discovered paths that are invalid) using different RF ray tracing techniques in three different rooms.

Raytracing Method	Simple	Crowded	Garage
Wireless InSite	0%	0%	25%
Plane Fitting	1.5%	3%	3%
RT Approximation	45%	16%	12%
RT Approx + Pruning	12%	4%	7%

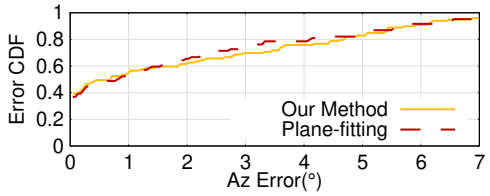

Figure 7: Comparison of azimuth angle error under different ray tracing methodologies. Plane-fitting denotes standard ray-tracing on a mesh model constructed using the modified plane-fitting method.

5.4 Benchmarks

We develop a set of ground-truth paths by finding peaks in the Angle of Departure (AoD) spectrum measured using our mmWave radios over 44 transmit/receiver location pairs for each location (§ 7.1). We use this ground truth data to evaluate the performance of our ray-tracing solution. Table 4 shows the percentage of first-order paths that we discover which are invalid, while Table 3 shows the percentage of known paths which are missed by ray-tracing. After the pruning process, the results from our process significantly outperforms methods based on the unmodified RF ray tracing process. We include evaluations from a garage, in addition to two standard indoor rooms, to show that our system works in a variety of environments. Figure 7 shows that fewer than 10% of all estimated paths have an estimation error greater than 6°. Since our mmWave phased arrays have a 12°half-power beam width, errors of less than 6° will generally not have a significant effect on the system performance. Thus, we experience angle error significant enough to reduce available SNR in fewer than 9% of the paths we detect.

6 REAL-TIME BEAM SELECTION

We now apply the beam power estimation techniques to mmWave network configuration. There is already substantial previous work that considers base station assignment, beam selection, and scheduling [13, 14, 19]. Here, we are primarily interested in the unique challenges presented when we use ray-tracing prediction, rather than run-time network measurement, to handle this configuration. The first is achieving real-time beam selection with ray-tracing, which itself may take significant computation time. To ensure that the beams can be updated in time, we employ a lightweight lookup table scheme to cache the path parameters available at each location. Second, our 3D model is quasi-static, and receives only infrequent updates due to the computational overhead in model construction. This would not affect *reflector* localization and identification, since most dynamic objects indoor, such as people and furniture, are not strong mmWave reflectors. However, *blockages* such as people

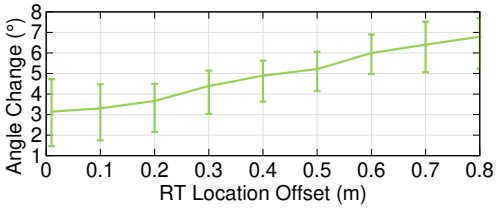

Figure 8: Error in angle of departure for valid paths as a function of the offset of the position used in ray-tracing. The confidence bounds are the 25th and 75th percentiles.

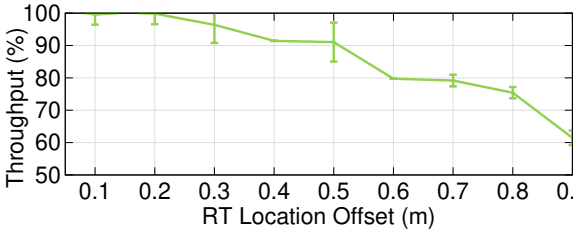

Figure 9: Throughput reduction due to offset between locations used during ray tracing and actual radio locations. Throughput is expressed as a percentage of the throughput obtained when using the exact location.

are often highly mobile, and run-time LiDAR sensing of blockages would require that each node have its own LiDAR sensor to reconstruct the scene in real time. In addition, our system cannot predict reflection or blockage losses with complete accuracy in all situations. To address these problems, we make use of a *fast failover* mechanism to quickly handle unexpectedly weak or blocked paths.

6.1 Ray-tracing Precomputation

Although our method nearly eliminates the communication overhead in traditional beam scanning, the computational complexity associated with ray-tracing is substantially greater. In mobile scenarios, the additional latency introduced between a change of node location and update of beam selection for each user can lead to highly sub-optimal network configurations. To avoid this problem, we precompute a look-up table to store the available paths between any pair of transmitter/receiver locations, rather than computing these values in real time. This solution works for even low-profile embedded devices, and may easily be executed in MAC-layer firmware.

The look-up table requires that we partition the 3D space into a discrete number of spots. At runtime, the transmitter and receiver can simply associate themselves to the nearest spots and retrieve the cached path information. A finer grained partition may ensure higher precision, but result in a larger table. To understand the relevant real-time constraints, we first conduct a set of experiments to determine the effect of an offset between the actual current location and the location used to find a ray-tracing solution. As described in § 7.1, we collect real mmWave measurements and use a LiDAR device to measure the precise location of the radio to within ±2 cm. Figure 8 shows increase in angle prediction error as a function of the distance between the actual position of the radio and the location used for ray-tracing, calculated using the same methodology used to evaluate our ray-tracing performance in § 5.4. Since our half-power beam width is 12°, when the angle errors greater than 6° occur, beginning around 50 cm location offset, the communications performance begins to degrade. Figure 9 shows this effect in a network consisting of two users and two access points, using the same methodology used in § 7.6. In this test case, the system performance begins to drop once the users are more than 20 cm from their ray-tracing locations. Based on this data, we conclude that any given ray-tracing solution may be used with a sphere of radius 20 cm, centered around the radio location used during ray-tracing. To ensure that the user is never more than 20 cm from a precomputed solution, we create a grid of points 28 cm apart. A lookup table containing the precomputed beams and their signal strengths for any point in the entire volume of either room would be less than 50 kilobytes in memory. Since this data only needs to be loaded once per room, the overhead of retrieving and storing this data would be negligible.

6.2 Fast Failover

The second problem we must handle lies in the blocked or otherwise missing paths. Our reflection strength measurements show that the strongest reflectors in indoor environments are generally quasi-stationary objects such as walls, windows, and large metal objects. Smaller, more mobile objects only produce specular reflections at a limited range of locations. However, many mobile objects, especially people, are strong blockages, and may cause substantial attenuation (as demonstrated in § 7.4).

To handle new blockages that occur when a blocker moves into the currently-active beam, we sense drops in SNR of packets that we receive on the connection. Since a small drop in SNR precedes a full blockage, we can select a new beam before the old beam disappears [42]. When we switch to a new beam, either to enable multi-user multiplexing, or to handle a new blockage, we also need to check for dynamic blockages along the new selected beam path before we use it. To check for blockage along a new path, we leverage limited, low-overhead measurements using the Beam Refinement Protocol (BRP) built-in existing mmWave devices such as 802.11ad. BRP is intended to scan a smaller number of sectors to improve the beam selection after a broad beam sweep is completed and a rough beam alignment is available. The device which initiates BRP may specify the exact beams to check. As such, it requires significantly less time to complete than a standard sector sweep. To minimize the overhead of this process in the case where blockage is discovered, we scan the top-3 available paths, ranked by the expected signal strength calculated as specified in § 4.2. Any path where the measured signal strength is significantly lower than the expected signal strength is considered blocked.

7 EVALUATION

7.1 Experimental Setup

To demonstrate the effectiveness of SpaceBeam, we collected 3D modelling data and RF propagation data in multiple real environments. To create the 3D models, we used a Microsoft Azure Kinect Time of Flight camera [25]. We created an RGBD sequence by moving the camera around the environment, and used a robust multi-step reconstruction algorithm [7, 31] to build a 3D mesh model with a 5 mm resolution.

Figure 10: Two rooms used to evaluate our system. Empty Room (a) is 4m in length, 2.7m in width, and 2.7m in height. Crowded Room (b) is 4.4m in length, 3.2m in width, and 2.5m in height, excluding the stairs at the rear.

To collect RF data, we employ a commercial 802.11ad radio using an ARM-based single-board Linux computer as the host. The radio consists of 24 phased array panels, divided into 6 groups. 5 of these groups are mounted vertically around the sides of the device at 72° angles. The 6th group are mounted horizontally facing above, and are not used in our experiments. Together, the horizontal phased arrays achieve 360° field-of-view around the azimuth of the device and ±60° coverage in elevation. We create a codebook for these phased arrays with a 5° spacing between beams, where each beam has a 6° half-power beam width. Each phased array scans 36 beams, for a total of 180 beams across all active phased arrays.

At each measurement location, we collected per-beam per-packet RSS and SNR measurements for each transmitter/receiver pair. Our commercial 802.11ad device does not support cooperative TDMA scheduling between APs, but we can use these measurements to calculate the SNR and interference to noise ratio (INR) for any set of beam configurations and TDMA schedules. We compare SpaceBeam's performance with two baseline beam selection/scheduling schemes. *(i)* The *default 802.11ad* protocol, where each user greedily selects the AP and beam configuration with the highest RSS. To mitigate interference, the protocol creates a *PCP/AP cluster*, which allows neighboring APs and clients to detect and share when a network node overhears signals from a neighboring node. APs within the cluster then use the Extended Schedule mechanism to place users in non-interfering slots [1]. *(i)* The recently proposed *many-to-many beam alignment* protocol [19], which manages interference via AP and beam selection, as well as interference-aware TDMA for multi-AP mmWave networks.

Our 802.11ad device requires 1 ms to complete a sector sweep (SSW) for each group of phased arrays. To complete a full omni-directional scan, we must perform the sector sweep using at least 3 groups of transmit phased arrays, resulting in 3 ms of channel overhead. We must also use at least 3 groups of phased arrays at the receiver to achieve omnidirectional coverage at the receiver. Since the sector sweep phase does not transfer full SNR measurements, each client must also complete a beam refinement phase (BRP) to share per-sector SNR values with the APs. This process takes an additional 0.7 ms.

We conducted experiments in three different indoor environments. The first, shown in Figure 10(a), is an empty room that leads to a short hallway. A window set is recessed 22 cm into the wall on the side opposite the hallway. The second environment, shown in Figure 10(b), is a crowded room containing a couch, a television, a large metal reflector, a small refrigerator, and stairs. The third (Figure 11) is a garage, which is used to benchmark SpaceBeam's performance across environment types.

Figure 11: Garage used to validate that SpaceBeam works in a significantly different indoor environment than the one it was developed in.

7.2 Reflection Loss Estimation

We first verify the effectiveness of the IR-based reflection loss estimation technique in guiding beam selection. We consider a common scenario where the single best available NLoS transmit beam needs to be selected between two nodes in the garage environment. We exhaustively measure all possible transmit beam configurations as a baseline. We conduct experiments in the garage environment and compare our approach with two alternative estimation methods: *(i)* using ray-tracing as in § 5 to discover potential paths, then directly measuring the reflection strength in the real environment, *(ii)* free-space path loss only, assuming a fixed 5 dB reflection loss. Figure 12 compares the resulting SNR to the SNR of the globally optimal NLoS beam selected using exhaustive measurements. The measurement method achieves the global optimum in most cases, but includes errors due to improper raytracing, such as angle errors and undetected paths. The garage environment is more challenging than the previous rooms because of the large number of objects with complex geometries, the degradation in detected NLoS paths due to these errors is relatively small, but ray-tracing performs optimally in the majority of the scenarios we tested. Therefore, the improper ray-tracing, on its own, does not have a significant impact on the beams selected by SpaceBeam. Our IR-based estimation technique selects beams that are 4 dB weaker on average than the exhaustive measurements. Using only free-space path loss to calculate signal strength yields beams that are another 5 dB weaker than our approach. Therefore, even the coarse IR-based reflection estimation technique (§4.2) has significant benefits, although there is still room for improvement compared with the exhaustive measurement.

7.3 Beam Selection Mechanisms

To demonstrate the effectiveness of SpaceBeam in beam selection, we use it to find beam configurations in the crowded room scenario with two users and two APs, in comparison with 4 baselines: *(i)* Max-SSW, *i.e.*, the 802.11ad sector sweep feedback mechanism, which only feeds back index of the strongest beam. *(ii)* The Many to Many

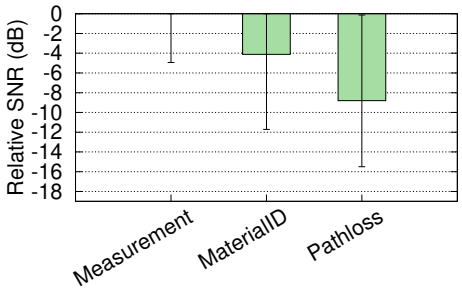

Figure 12: Quality of the selected NLoS path based on different reflection loss estimation techniques. *Measurement* directly measures reflection loss of all paths, *MaterialID* is our method for material identification, and *Pathloss* ignores reflection loss. All SNRs are relative to the SNR of the best NLoS path.

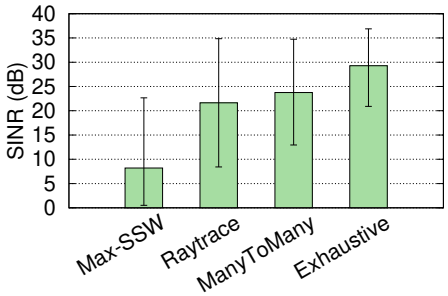

Figure 13: SINR obtained using different beam selection methods. The error bars denote the minimum and maximum SINR values across all trials.

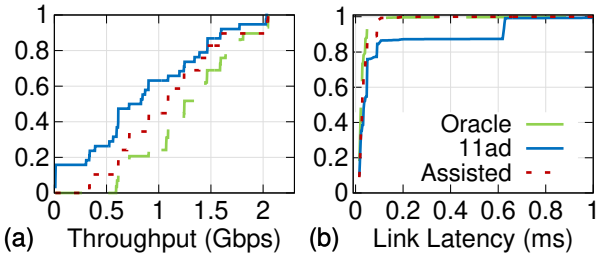

Figure 14: Single-user network performance evaluation. Part (a) shows the throughput, and part (b) shows link-layer latency, both after accounting for control overhead and packet loss. We compare a hypothetical oracle with perfect channel knowledge to our model-assisted solution and the default 802.11ad protocol.

method [19], which configures both transmit and receive beam patterns using on transmit sector sweeps, *i.e.* without taking any measurements with receive beamforming. *(iii)* A Raytrace method that selects the best beam using only estimates derived from our IR-based sensing, without any mmWave sensing. *(iv)* The Exhaustive method denotes complete measurements of all transmit and receive beam patterns, coupled with an exhaustive search for the best SINR.

We place the user nodes in a grid pattern with approximately 30 cm edges on the far half of the empty room, and APs on the opposite side of the room. In the crowded room, we placed the APs on the near side of the room, as shown from Fig. 10(b), and placed the user nodes at a height of 1 m in a 20 cm grid in the remainder of the open floor space in the room. We collected additional measurements where the user nodes were placed at heights of 0.8 m and 0.3 m in 8 total locations near the center of the room. In total, we collected measurements from four AP locations and 52 UE locations.

Fig. 13 shows the resulting SINR values. Notice that even with a limited number of nodes, the default sector sweep method produces very low SINR values. In these cases, other multiplexing methods must be used to accommodate multiple users. The median SINR obtained with our techniques is only about 2 dB lower than the measurements obtained with the many-to-many beam alignment method. The many to many method performs slightly worse than the exhaustive search method because it assumes an nearly-uniform omnidirectional receive pattern, whereas our commerical phased array has significant variation in its receive pattern. The median SINR with the exhaustive measurement is 8 dB better than our method.

7.4 Environment Dynamics

To demonstrate how well SpaceBeam responds to environment dynamics, we conduct two experiments with significant environment dynamics.

In the first experiment, we introduce a mobile blockage (*i.e.*, an absorber board) into the crowded room scenario. We set up a single transmitter, and move a receiver and the absorber by 10 cm at a time to collect RF traces at a total of 25 locations, such that different paths are blocked by the absorber, which necessitate reselection of beams. We measure the SNR, throughput, and latency achieved using our method, in comparison to the standard 802.11ad protocol with a beacon interval of 100 ms, and to an oracle that continuously determines the best beam with no overhead.

The resulting throughput and latency are shown in Figure 14. Although SpaceBeam does not explicitly include the moving blockage in its 3D model, it runs fast beam search when an SNR drop is detected, and can still avoid the outages witnessed by 802.11ad. In addition, there is only a marginal throughput drop compared to the oracle case. SpaceBeam achieves this with minimal overhead, since it tracks the location of the current beam using ray tracing, rather than taking direct measurements.

Second, we incrementally increase the amount of changes in the crowded room (Figure 10(b)) as follows to emulate real-world environmental dynamics: *(i)* keeping the environment the same as it was during the initial LiDAR scan; *(ii)* adding a LoS blockage; *(iii)* rearranging various smaller movable objects (wood furniture, a package of water bottles, boxes, *etc.*); and *(iv)* moving the main metal reflector from the original model. In each condition, we move the user along a 1 m path at 0.5 m/s. The in Figure 15 show that, until step (iv), SpaceBeam's network performance remain almost unaffected, because the blockage detection mechanism simply re-routes our connection to an alternate reflector. When moving the large reflector, SpaceBeam suffers from a throughput reduction of 45% compared to the oracle case. However, such changes are rare (*e.g.*, moving large metal furniture in home environment), and SpaceBeam can afford infrequently rerunning the 3D reconstruction to restore its performance.

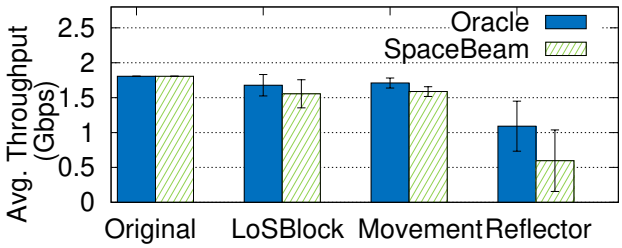

Figure 15: Single-user throughput under different types of environment changes: LoS blockage, movement of wood furniture/small objects, and movement of a metal reflector.

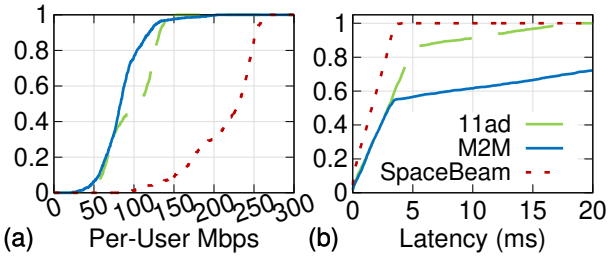

Figure 16: Per-user link-layer (a) throughput and (b) latency CDFs for a network with two base stations and 14 users. The CDFs for other user counts are similar. 11ad denotes 802.11ad interference mitigation, and M2M denotes the protocol in [19].

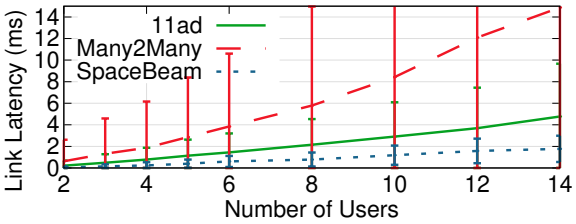

Figure 17: Link-layer latency as a function of the number of users in a two-AP environment.

7.5 Computation Time

While SpaceBeam assumes a quasi-static environment, it needs to update the environment model when the environment changes significantly. To facilitate real-time beam assignment, we need to precompute the available beams and the expected total path loss, for each pair of user location and AP location. Averaging over many ray-tracing runs on an Intel i7-8700 CPU, the entire ray-tracing process takes 1.4 s on average for each user/AP location pair. In total, this process takes 22.5 minutes for the crowded room and 18.6 minutes for the empty room, for two APs. This latency should be affordable for typical environments where the set of large reflectors rarely changes.

7.6 System-Level Evaluation in a Multi-User Network

Next, we experiment with a multi-user network in the empty and crowded room scenarios. We re-use the same measurements that we used in § 7.3, and conduct a trace-driven simulation using measured network data as described in § 7.1. We assume that the users are moving at 2 m/s, such that we must re-train our beam selections at least once every 200 ms (as demonstrated by our previous location offset experiment in § 6.1). For each number of users in our plots, we conduct 50 tests using randomly-selected sets of users drawn from these data sets.

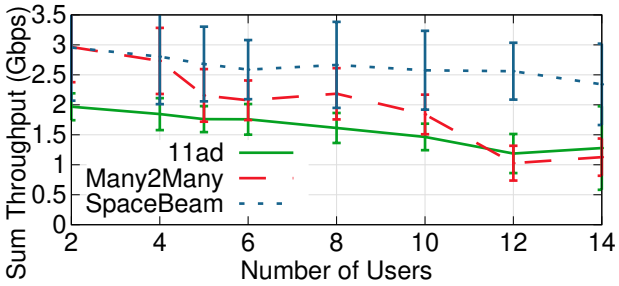

Figure 18: Total link-layer throughput as a function of number of users in a two-AP environment.

Figure 16 shows the detailed link statistics over the trials where 14 users are selected. Specifically, we note that SpaceBeam provides a worst-case latency significantly lower than alternative methods, making it especially suitable for latency-sensitive applications

Figure 17 shows the link-layer latency as a function of the number of concurrent users. The latency is affected by both the measurement overhead required for beam training and latency introduced while waiting for a TDMA slot. We operate with 500 μs TDMA time slots, which is within the allowed size of scheduled transmission periods in 802.11ad [1], and is low enough to allow for latency-sensitive communications. Many-to-many beam alignment halts the network during the training process, leading to a long tail of high latency for packets waiting to be transmitted during this period. It has higher control overhead than the default 802.11ad protocol because it must scan all 180 beams and exchange per-beam SNR information between each user node and the APs. This process requires a total of 5.1 ms per node On the other hand, TDMA delay contributes significantly to the latency when we using 802.11ad, since there are very few opportunities for concurrently transmitting multiple data streams. 802.11ad does not attempt to discover NLoS paths, so end users in our implementation only activate and train the set of phased arrays with the highest SNR from the AP, further reducing the required training overhead. By contrast, SpaceBeam's only management comes from AP beacons, which allow for initial user association, and the blockage-management measurements described in § 6.2.

Fig. 18 shows the total throughput available in the network as a function of the number of users. In this case, network throughput is limited by the number of available APs, and per-user throughput will decrease significantly as the number of users sharing the base station increases. Both the SpaceBeam and 802.11ad protocols have slight declines in total throughput delivered to users as a result of management overhead.

8 DISCUSSION

Our work is a first effort to achieve real-time beam selection using detailed LiDAR data without any kind of environment-specific tuning or training. There are a number of possible paths to improve on this work and adapt it to other use cases, especially for highly-dynamic environments such as outdoor V2X. The clearest

opportunity for improvement is in improving our estimation of the precise reflection characteristics of the environment. Our evaluation in Fig. 12 shows that our estimation of material characteristics is the primary limiting factor in the performance of our system, not the ray-tracer. We may improve this performance in a quasi-static environment by measuring these characteristics using mmWave radios or radars, then mapping these sparse measurements to objects in the dense mesh. More dynamic environments could require more sophisticated deep learning-based semantic segmentation schemes to achieve this kind of learning.

Real-time usage. There are many potential use cases, such as V2X, where LiDAR-driven beam management would be useful, but where advance mapping may not be possible. To achieve this, we need to reconstruct and ray-trace the environment in real time. We were unable to accomplish this in our testbed due to the limitations of our single depth camera, which has a very limited field of view and is therefore unable to sample the full environment in real time. Given multiple depth cameras at known locations on a network node covering all angles around the node, we could continuously generate a model of the surrounding objects. Since the relative locations of all of the point clouds generated in this process are known, point cloud alignment steps could be bypassed, and only the TSDF mesh generation step would remain. This step may be accomplished in only a few milliseconds using GPU acceleration [10].

The second challenge would be to complete the ray-tracing path discovery step in real time. Our custom ray-tracer requires an average of 1.4 s using six parallel processes using Intel's optimized ray-intersection kernels [3]. A straightforward approach to reducing this processing time is to rely on GPU processing. New GPUs with dedicated ray-tracing cores can process up to 10 million rays per second. Using GPU acceleration, we should be able to complete the ray-tracing step in under 100 ms. There are also likely ways to improve the ray-tracing algorithm we presented here to reduce the number of rays that must be processed, which could further improve processing time.

9 RELATED WORK

Efficient beam selection represents the most critical problem for robust mmWave networking, and a variety of solution frameworks have been explored in the past a few years. The first are variants of hierarchical beam search, which measures wider or quasi-omnidirectional beams first, and then uses these initial results to reduce the search space during the beam refinement phase. The fundamental problem with these techniques is that they do not consider the environmental invariant which impacts the channel in structured ways. As such, they still induce substantial trial-and-error overhead in mobile scenarios. A second class of techniques attempts to leverage temporal correlation to speed up the beam selection process [45], assuming the best beam changes slowly over time. This works well for purely line-of-sight (LoS) scenarios, but struggle in non-line of sight (NLoS) scenarios, since mmWave reflectors are sparse and tend to change rapidly over time. The third class of techniques is closest to our approach, and leverages spatial information to guide mmWave network configuration [49, 51, 52]. For example, EMi [49] explicitly senses the locations of reflectors to guide the deployment of mmWave base stations. The main weakness of such solutions is that they still use mmWave sensing. Since the mmWave channel is sparse, these approaches require extensive prior measurements across many locations, and generally assume a two-dimensional environment, i.e., the beams and radios all stay in the same horizontal plane.

Prior work considered using various out-of-band measurements for mmWave configuration, using sub-6 GHz measurements [29], motion sensors [48], LiDAR point clouds [9], and mmWave radar [38]. However, most of these works focus on rapidly identifying LoS paths based on node locations. In particular, the previous LiDAR-based approach [9] requires extensive training data, and performs best in identifying LoS blockage and beam angles. Alternatively, mmWave radar [4, 15] can help estimate the channel covariance at the communications band, to find the available LoS and NLoS paths. However, these methods require specialized radar hardware, potentially at both the transmitter and receiver, and a separate open spectrum band nearby for radar sensing. By contrast, our method requires only one device with a common LiDAR sensor to scan a quasi-static environment, and does not need extra RF spectrum resources.

Other work showed that a manually-constructed 3D model can be used for beam selection [8, 32]. This requires manual construction of a 3D model, and manual assignment of complex permittivity and roughness values. The closest work to ours [20] models a laser-scanned environment using a point cloud, rather than a mesh. It operates by summing received power contributions from all point in the point cloud using a single-lobe scattering model, rather than finding discrete paths, and can find the angular power spectrum and delay profile. This approach requires environment-specific tuning using multiple mmWave measurements to optimize its propagation parameters, and does not model differences in reflection materials. In addition, because it uses a point cloud rather than a mesh, it does not model blockage.

10 CONCLUSION

In this work, we have introduced SpaceBeam, the first system to use real LiDAR measurements for beam assignment in multipath environments. We demonstrated key methods to allow for ray-tracing on imperfect LiDAR data and out-of-band estimation of reflection coefficients. We then showed that our method more than doubles the link-layer throughput available in an indoor scenario with 12 users, while reducing average latency by almost 90%. We believe SpaceBeam materializes a new principle of using cross-domain sensing to capture the invariant wireless channel features that are determined by the ambient environment.

ACKNOWLEDGMENTS

We appreciate the insightful comments and feedback from the anonymous reviewers and shepherd. The work reported in this paper is supported in part by the NSF under Grants CNS-1901048, CNS-1925767, and CNS-1952942.

REFERENCES

[1] 2012. IEEE Std 802.11ad. *IEEE Std 802.11ad-2012 (Amendment to IEEE Std 802.11-2012, as amended by IEEE Std 802.11ae-2012 and IEEE Std 802.11aa-2012)* (2012). https://doi.org/10.1109/IEEESTD.2012.6392842

[2] 2020. Airfide. http://airfidenet.com/

[3] Attila T. Áfra, Ingo Wald, Carsten Benthin, and Sven Woop. 2016. Embree Ray Tracing Kernels: Overview and New Features. In *Proceedings of ACM SIGGRAPH*.

[4] A. Anum, N. GonzalezPrelcic, and A. Ghosh. 2020. Passive Radar at the Roadside Unit to Configure Millimeter Wave Vehicle-to-Infrastructure Links. *IEEE Transactions on Vehicular Technology* (2020). https://doi.org/10.1109/TVT.2020.3027636

[5] Apple. 2020. iPhone 11 Pro Max.

[6] Jiangfeng Cheng, Weihai Chen, Fei Tao, and Chun-Liang Lin. 2018. Industrial IoT in 5G environment towards smart manufacturing. *Journal of Industrial Information Integration* (2018).

[7] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. 2015. Robust Reconstruction of Indoor Scenes. In *IEEE CVPR*.

[8] V. Degli-Esposti, F. Fuschini, E. M. Vitucci, M. Barbiroli, M. Zoli, L. Tian, X. Yin, D. A. Dupleich, R. Müller, C. Schneider, and R. S. Thomä. 2014. Ray-Tracing-Based mm-Wave Beamforming Assessment. *IEEE Access* (2014).

[9] M. Dias, A. Klautau, N. González-Prelcic, and R. W. Heath. 2019. Position and LIDAR-Aided mmWave Beam Selection using Deep Learning. In *IEEE SPAWC*.

[10] Wei Dong, Jaesik Park, Yi Yang, and Michael Kaess. 2019. GPU Accelerated Robust Scene Reconstruction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[11] A. El Saddik. 2018. Digital Twins: The Convergence of Multimedia Technologies. *IEEE MultiMedia* (2018).

[12] Martin A. Fischler and Robert C. Bolles. 1981. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* (1981).

[13] Yasaman Ghasempour, Muhammad K. Haider, Carlos Cordeiro, Dimitrios Koutsonikolas, and Edward Knightly. 2018. Multi-Stream Beam-Training for mmWave MIMO Networks. In *Proceedings of ACM MobiCom*.

[14] Y. Ghasempour, M. K. Haider, and E. W. Knightly. 2018. Decoupling Beam Steering and User Selection for MU-MIMO 60-GHz WLANs. *IEEE/ACM Trans. on Networking* 5 (2018).

[15] N. González-Prelcic, R. Méndez-Rial, and R. W. Heath. 2016. Radar aided beam alignment in MmWave V2I communications supporting antenna diversity. In *2016 ITA Workshop*.

[16] Intel. 2020. RealSense L-515 LiDAR Camera.

[17] ITU. 2012. ITU P.1410-5.

[18] S. Jiang, N. Y. Chang, C. Wu, C. Wu, and K. Song. 2014. Error analysis and experiments of 3D reconstruction using a RGB-D sensor. In *IEEE CASE*.

[19] Suraj Jog, Jiaming Wang, Junfeng Guan, Thomas Moon, Haitham Hassanieh, and Romit Roy Choudhury. 2019. Many-to-Many Beam Alignment in Millimeter Wave Networks. In *Proceedings of USENIX NSDI 19*. Boston, MA.

[20] J. Järveläinen and K. Haneda. 2014. Sixty gigahertz indoor radio wave propagation prediction method based on full scattering model. *Radio Science* (2014).

[21] Aldebaro Klautau, Nuria González-Prelcic, and Robert W. Heath Jr. 2019. LIDAR Data for Deep Learning-Based mmWave Beam-Selection. *IEEE Wireless Commun. Lett.* 8, 3 (June 2019), 909–912. https://doi.org/10.1109/LWC.2019.2899571 arXiv: 1908.07488.

[22] B. Langen, G. Lober, and W. Herzig. 1994. Reflection and transmission behaviour of building materials at 60 GHz. In *IEEE PIMRC*.

[23] H. Ling, R. Chou, and S. Lee. 1989. Shooting and bouncing rays: calculating the RCS of an arbitrarily shaped cavity. *IEEE Transactions on Antennas and Propagation* (1989).

[24] Jonathan Lu, Daniel Steinbach, Patrick Cabrol, Phil Pietraski, and Ravikumar V. Pragada. 2014. Propagation characterization of an office building in the 60 GHz band.

[25] Microsoft. 2020. Azure Kinect.

[26] Jeffrey Nanzer. 2014. *Microwave and Millimeter-Wave Remote Sensing for Security Applications.* Artech House.

[27] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. 2011. KinectFusion: Real-time dense surface mapping and tracking. In *IEEE ISMAR*.

[28] C. V. Nguyen, S. Izadi, and D. Lovell. 2012. Modeling Kinect Sensor Noise for Improved 3D Reconstruction and Tracking. https://doi.org/10.1109/3DIMPVT.2012.84

[29] Thomas Nitsche, Adriana B. Flores, Edward W. Knightly, and Joerg Widmer. 2015. Steering with Eyes Closed: mm-Wave Beam Steering without In-Band Measurement. In *IEEE Conference on Computer Communications (INFOCOM)*.

[30] Ouster. 2020. OS1 Mid-range LiDAR sensor.

[31] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Colored Point Cloud Registration Revisited. In *ICCV*.

[32] J. Pascual-García, J. Molina-García-Pardo, M. Martínez-Inglés, J. Rodríguez, and N. Saurín-Serrano. 2016. On the Importance of Diffuse Scattering Model Parameterization in Indoor Wireless Channels at mm-Wave Frequencies. *IEEE Access* (2016).

[33] T. S. Rappaport, F. Gutierrez, E. Ben-Dor, J. N. Murdock, Y. Qiao, and J. I. Tamir. 2013. Broadband Millimeter-Wave Propagation Measurements and Models Using Adaptive-Beam Antennas for Outdoor Urban Cellular Communications. *IEEE Transactions on Antennas and Propagation* 61, 4 (2013).

[34] Ch. Rayssi, S. El.Kossi, J. Dhahri, and K. Khirouni. [n.d.]. Frequency and temperature-dependence of dielectric permittivity and electric modulus studies of the solid solution. *RSC Adv.* ([n. d.]).

[35] Antoni Rosinol, Marcus Abate, Yun Chang, and Luca Carlone. 2020. Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. https://github.com/MIT-SPARK/Kimera

[36] K. Sato, T. Manabe, T. Ihara, H. Saito, S. Ito, T. Tanaka, K. Sugai, N. Ohmi, Y. Murakami, M. Shibayama, Y. Konishi, and T. Kimura. 1997. Measurements of reflection and transmission characteristics of interior structures of office building in the 60-GHz band. *IEEE Transactions on Antennas and Propagation* (1997).

[37] R. Sibson. 1973. SLINK: An optimally efficient algorithm for the single-link cluster method. *Comput. J.* (1973).

[38] Ljiljana Simić, Julian Arnold, Marina Petrova, and Petri Mähänen. 2016. RadMAC: Radar-Enabled Link Obstruction Avoidance for Agile Mm-Wave Beamsteering. In *HowWireless '16*. ACM.

[39] Peter Smulders. 2002. Exploiting the 60 GHz band for local wireless multimedia access: Prospects and future directions. *IEEE communications magazine* 40, 1 (2002), 140–147.

[40] Jakob Struye, Filip Lemic, and Jeroen Famaey. 2020. Towards Ultra-Low-Latency mmWave Wi-Fi for Multi - User Interactive Virtual Reality. In *IEEE Global Communications Conference*.

[41] Sanjib Sur, Vignesh Venkateswaran, Xinyu Zhang, and Parmesh Ramanathan. 2015. 60 GHz indoor networking through flexible beams: A link-level profiling. In *ACM SIGMETRICS*. 71–84.

[42] Sanjib Sur, Xinyu Zhang, Parmesh Ramanathan, and Ranveer Chandra. 2016. BeamSpy: Enabling Robust 60 GHz Links Under Blockage. In *Proceedings of USENIX NDSI*.

[43] Telecom Infra Project. 2019. Analysis of 28GHz and 60GHz Channel Measurements in an Indoor Environment.

[44] Velodyne. 2016. Puck LITE Surround LiDAR.

[45] Song Wang, Jingqi Huang, and Xinyu Zhang. 2020. Demystifying Millimeter-Wave V2X: Towards Robust and Efficient Directional Connectivity under High Mobility. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking (MobiCom)*. Article 51.

[46] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger. 2019. Pseudo-LiDAR From Visual Depth Estimation: Bridging the Gap in 3D Object Detection for Autonomous Driving. In *CVPR*.

[47] Y. Wang, A. Klautau, M. Ribero, A. C. K. Soong, and R. W. Heath. 2019. MmWave Vehicular Beam Selection With Situational Awareness Using Machine Learning. *IEEE Access* (2019).

[48] Teng Wei and Xinyu Zhang. 2017. Pose Information Assisted 60 GHz Networks: Towards Seamless Coverage and Mobility Support. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking (MobiCom '17)*. ACM, New York, NY, USA, 42–55. https://doi.org/10.1145/3117811.3117832 event-place: Snowbird, Utah, USA.

[49] Teng Wei, Anfu Zhou, and Xinyu Zhang. 2017. Facilitating Robust 60 GHz Network Deployment By Sensing Ambient Reflectors. In *NSDI 17*. USENIX.

[50] B. Wu, A. Wan, X. Yue, and K. Keutzer. 2018. SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud. In *2018 IEEE ICRA*.

[51] Anfu Zhou, Shaoqing Xu, Song Wang, Jingqi Huang, Shaoyuan Yang, Teng Wei, Xinyu Zhang, and Huadong Ma. 2020. Robotic Millimeter-Wave Wireless Networks. *IEEE/ACM Transactions on Networking* 28, 4 (2020).

[52] Anfu Zhou, Xinyu Zhang, and Huadong Ma. 2017. Beam-forecast: Facilitating mobile 60 GHz networks via model-driven beam steering. In *IEEE Conference on Computer Communications (INFOCOM)*.

[53] Yibo Zhu, Zengbin Zhang, Zhinus Marzi, Chris Nelson, Upamanyu Madhow, Ben Y. Zhao, and Haitao Zheng. 2014. Demystifying 60GHz Outdoor Picocells. In *ACM Mobicom*.