Robot Navigation in Radio Beam Space: Leveraging Robotic Intelligence for Seamless mmWave Network Coverage

Anfu Zhou[+], Shaoqing Xu[+], Song Wang[*], Jingqi Huang[*],
Shaoyuan Yang[+], Teng Wei[◇], Xinyu Zhang[*], Huadong Ma[+],
[+]Beijing University of Posts and Telecommunications {zhouanfu, donggua, yangsy, mhd}@bupt.edu.cn
[*] University of California San Diego {sowang, jih032, xyzhang}@ucsd.edu
[◇] University of Wisconsin-Madison twei7@wisc.edu

ABSTRACT

The emerging millimeter-wave (mmWave) networking technology promises to unleash a new wave of multi-Gbps wireless applications. However, due to high directionality of the mmWave radios, maintaining stable link connection remains an open problem. Users' slight orientation change, coupled with motion and blockage, can easily disconnect the link. In this paper, we propose miDroid, a robotic mmWave relay that optimizes network coverage through wireless sensing and autonomous motion/rotation planning. The robot relay automatically constructs the geometry/reflectivity of the environment, by estimating the geometries of all signal paths. It then navigates itself along an optimal moving trajectory, and ensures continuous connectivity for the client despite environment/human dynamics. We have prototyped miDroid on a programmable robot carrying a commodity 60 GHz radio. Our field trials demonstrate that miDroid can achieve nearly full coverage in dynamic environment, even with constrained speed and mobility region.

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous and mobile computing design and evaluation methods**; • **Networks** → *Mobile networks*;

KEYWORDS

mmWave Netw., Mobile Relay, User Tracking, Path Planning

1 INTRODUCTION

The past few years have witnessed major evolution of ultra-high-speed millimeter-wave (mmWave) networks. While originally designed for quasi-stationary use cases such as cellular backhaul, datacenters and wireless display[7], mmWave networks are anticipated to become mainstream in 5G, and enable more versatile applications, including WiFi-like access, untethered virtual reality, mobile offloading, *etc.* [20]. However, the inherent shortcomings of mmWave links, *i.e.*, limited coverage and stability, remain a major barrier in practice. Although a mmWave radio can use phased-array antennas to generate highly-directional, electronically-steerable beams, the joint coverage of all its beams can only form a limited Field of View (FoV) (typically below 170°[26]) just like camera lens. As a result, it is non-trivial to maintain stable connectivity even at a room-level. Link outage occurs when a user exits the access point's FoV due to motion, hand/body rotation, or blockage by other surrounding people.

To overcome the challenge, one natural method is to deploy relays with complementary FoVs to approximate pervasive coverage [1, 26]. However, our field tests reveal that the effectiveness of mmWave relaying highly depends on the environment and device locations, and a static relay still suffers from many blind zones/angles (Sec. 7.2). Although deploying multiple APs may help, it has been shown that even 3 APs can only cover 90% of a simple $7 \times 8m^2$ area without any human activities nearby[26]. Worse still, the blind zones/angles vary dynamically as more people walk around, thus demanding more APs to remove the blockage/shadowing effects. The resulting backhaul or cabling cost thus becomes formidable, even for simple multi-room office or home environment.

In this paper, we propose miDroid, which explores robotic intelligence to realize continuous coverage for mobile mmWave users. miDroid is inspired by the vision that robots will eventually become part of the home and enterprise environments to help automate our daily lives and improve productivity. In fact, cleaning robots and telepresence robots have already been widely adopted, promising a multi-billion market [15]. miDroid piggybacks a mmWave radio on a robot and transforms it into a mobile relay (Fig. 1). A naive way of leveraging this robotic relay is to always navigate it to "follow" the client. But this is infeasible in practice due to the robot's *limited moving speed/agility* and *constrained mobility region*, and when multiple clients coexist. For instance, one may want to set the robot's mobility region as a narrow strip beside the walls, so that the relay will not interfere with human daily activities, as shown in Fig. 1. The key insights behind miDroid are: *(i)* The mmWave channel is intrinsically sparse [23, 27]. Coverage issue is often due to specular reflections not being redirected to the proper

Mobihoc '19, July 2–5, 2019, Catania, Italy

ACM ISBN 978-1-4503-6764-6/19/07…$15.00
https://doi.org/10.1145/3323679.3326514

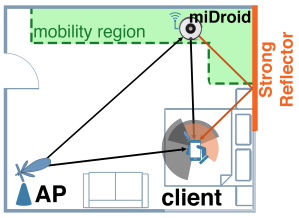

Figure 1: Usage scenario of miDroid.

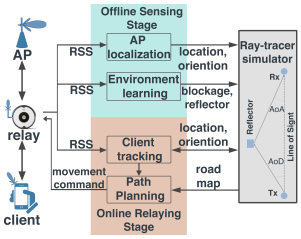

Figure 2: miDroid system overview.

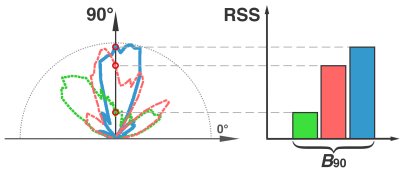

Figure 3: Beam cross section.

angle. Even minor movement of the robot relay can save the client out of the shadowed region. Therefore, it can improve the coverage substantially even under speed/space constraints. *(ii)* The client's orientation may change rapidly, and it is infeasible for the robot to keep itself within the client's FoV. However, if the mmWave robot can sense ambient reflectors (*e.g.*, walls and furniture), it can judiciously navigate itself to a position/angle that may lead to a diverse set of reflection paths, thus probabilistically maximizing the *likelihood* of covering the client at any time. These insights set miDroid apart from the legacy WiFi or cellular relay networks [2, 4, 12, 30] whose designs are oblivious of node orientation and environmental structures.

Realizing the principle of autonomous mmWave relaying requires a revisit of two classical problems in robotics: environment mapping and path planning. Unlike the conventional robotic navigation, however, the unique challenge is that both problems need to be studied within the radio "environment" from the mmWave radios' eyes (comprised of the invisible signal paths and beams), rather than the physical environment. *First of all*, miDroid needs to predict the best feasible relay position based on prior knowledge of ambient reflectors, which in turn requires a reconstruction of the environment (*e.g.*, reflectivity and geometrical layout). At run time, the relay can keep tracking the client's position, and plan its moving trajectory to maximize the likelihood of full coverage for the client. Reconstructing the environment requires disentangling each reflection path and characterizing each reflecting point. Conventional ways of resolving signal paths [27], *i.e.*, discriminating their angle-of-arrival (AoA) and angle-of-departure (AoD), require phase-coherence across packets, which is infeasible on commercial-off-the-shelf (COTS) mmWave communication devices [19]. In miDroid, we propose a beam spatial correlation maximization method, which combines the autonomous motion control and self-localization capability of the robot, with simple RSS measurement from the mmWave radio, to extract the path geometries. Then we build a geometrical model to transform the path information into an environment outline. Meanwhile, we leverage the diversity of phased-array beam patterns, to enable a single relay to track the client's position even under blockage.

Second, the path planning for mmWave relay is intrinsically dynamic, and differs fundamentally from conventional path planning algorithms in robotics and robot-driven omni-directional radio networks (see [3, 30] and references therein). Legacy path planning derives the shortest/optimal path to *one or more target positions*. But the mmWave relay needs to maintain high link performance for the client *along the path*. This implies the relay needs to predict the client's performance along each point of the path, which is non-trivial as the performance depends on the environment, as well as the relative position/orientation between the relay, client and AP. Moreover, even for an oracle with perfect prediction, computing the optimal trajectory across the points entails exponential complexity. In miDroid, we introduce a statistical method to predict the relay performance; We then simplify the relay path planning as a tree searching problem, based on an undiscovered *locality* property of the relay's spatial performance distribution.

miDroid, to our knowledge, represents the first system that harnesses robotic intelligence to facilitate seamless mmWave networking. Its contributions can be summarized as:

(i) We design novel algorithms that combine the motion/orientation control capability of the robot with the RSS measurement capability of COTS mmWave radios, to accurately recover the geometries of signal paths, and subsequently create an outline of the environment from the mmWave radios' eyes.

(ii) We design an adaptive path planning algorithm that navigates the robot relay in real-time, and statistically maximizes network performance under environment dynamics and the client's self-blockage.

(iii) We implement miDroid on a programmable robot, integrated with COTS 802.11ad radios. Our experiments in multi-room environments verify that miDroid can maintain nearly full coverage for an office environment, even when the robot is constrained to a small area and low speed. In comparison, existing solutions need around 4-5 APs to achieve similar performance.

2 MIDROID OPERATIONS

miDroid's operations follow two stages, as illustrated in Fig. 2. When entering a new environment, miDroid needs to initialize itself through an *offline sensing stage*. Here the robotic relay roams around and locates the AP relative to a prescribed starting point, by simply sampling the AP-to-relay RSS. Then the relay, assisted by the AP, reconstructs the environment, *i.e.*, inferring location, shape and reflectivity of major reflectors and obstacles. Both the AP location and environment information is fed into a ray-tracing engine, which models the signal's propagation and interaction with environment, thus enabling miDroid to *predict the channel quality of arbitrarily located AP-client or AP-relay-client links,*

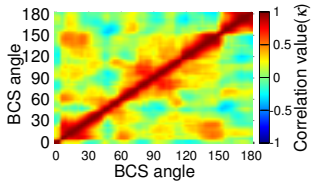

Figure 4: BCS correlation.

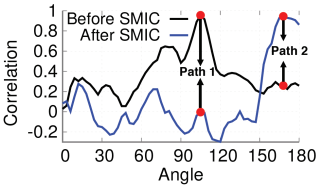

Figure 5: Effect of SMIC.

without requiring war-driving site-survey..

Afterwards, during the *online relaying stage*, the relay keeps tracking the client, based on prior knowledge of the environment, and the RSS of the relay-client link (for LOS and/or NLOS signals). Further, the relay employs an *adaptive path planning* scheme to *compute the optimal moving trajectory*, and roams itself accordingly.

Note that the signaling protocols for establishing two-hop relay links have already been standardized in 802.11ad [7]. However, problems such as whether to use relay and when to switch remain open. miDroid enables such decision making, as it can predict the individual link qualities (AP-client vs. AP-relay-client) and pro-actively switch to the relay mode when needed.

3 THE OFFLINE SENSING STAGE

Modern consumer robots, equipped with optical rangers, motion sensors, and SLAM algorithms, can already achieve self-localization and floor map reconstruction [28]. But they are unaware of mmWave signals' propagation/reflection/penetration in the environment, which determines the relay performance. miDroid augments the robot's physical sensing with mmWave wireless sensing. The core mechanism behind is called beam-cross-section correlation maximization (BCS-CM). BCS-CM extracts spatial information embedded from a series of AP beacons, and then recovers the AoA/AoD of mmWave signal paths. Such signal path geometry, combined with the robot's view, creates a radio environment map which is the key input into miDroid's path planning algorithm.

3.1 Path Geometry Estimation

To estimate the path AoA/AoD, one naive solution is to treat the mmWave radios as directional laser scanners, and steer over all angles to locate the reflecting point, just like optical LIDAR. However, practical mmWave phased-array beam patterns have imperfect directionality and limited spatial resolution. The beamwidth can be very wide (more than 30° for most beams even for a 32-element phased-array[26]), and multiple strong side-lobes often coexist, which hampers accurate path estimation [22, 26, 27]. BCS-CM solves the issue by harnessing redundant spatial information embedded in a series of beams, instead of relying on a single ideal laser-like beam. It can recover the AoA/AoD of multiple co-existing paths through a successive multi-path interference cancellation (SMIC) scheme. We now proceed to the details.

Beam Cross Section. The 802.11ad standard mandates that an AP (Tx) broadcasts a series of beacons in the beginning of each beacon interval to facilitate network discovery. The beacons are directional and bear unique beam patterns. The Rx is typically tuned to quasi-omni mode when trying to capture the beacons [26]. We define the *beam cross section* (BCS) as the set of RSS measured at a certain angle for all beacons. Fig. 3 illustrates the BCS of 3 beam patterns cut at 90° (relative to the antenna plane at 0°), measured on our 32-elment 802.11ad radio (Sec. 6). Formally, suppose we discretize the 2D horizontal plane (perpendicular to the phased-array panel) into multiple angles then a BCS at angle i is denoted as B_i and defined as follows,

$$B_i = \{RSS_{1,i}, .., RSS_{j,i}, ..RSS_{N,i}\} \quad (1)$$

where $RSS_{j,i}$ is the signal strength of j_{th} beacon at the i_{th} angle, and N is the number of beacons.

BCS Correlation Maximization (BCS-CM). Suppose the AoD of a Tx-Rx signal path is the i_{th} direction, then ideally a quasi-omni Rx will measure the vector B_i during Tx's beacon sweeping. These ideal B_i values can be derived during the phased-array's factory calibration, or from a one-time measurement of all its beam patterns (Sec. 6). In practice, BCS deforms due to signal attenuation and multipath reflections, represented by an unknown channel gain h_i. The actual measured BCS for angle i should follow [19, 26]: $B_i^M = h_i B_i + n$ where n is the noise. Note that frequency selective fading may vary h_i across frequency bins in a wideband channel. But to simplify exposition, we can consider a typical frequency bin that represents the average across all bins.

To identify the direction i from the B_i^M measurements, BCS-CM uses RSS trend in the BCS rather than absolute RSS values. Intuitively, BCSs are distinguishable due to the beam pattern heterogeneity across beacons. In Fig. 4, we compute and plot $\kappa(B_i, B_j)$, the correlation coefficient [14] of two BCS vectors (B_i and B_j) to measure their linear dependence, from which we observe that the correlation between two BCSs of different angles is much smaller than that of the same angle, *i.e.*, $\kappa(B_i, B_j) \ll \kappa(B_i, B_i)$. The result implies that BCS correlation is highly resilient to channel distortion. Therefore, we can compute correlation between measured B_i^M and each of the ideal B_i values, and pick the direction i^* corresponding to the maximum correlation as the AoD, *i.e.*,

$$i^* = \arg\max_i \kappa(B_M, B_i) \quad (2)$$

The AoA is derived using the same method, with a Tx-Rx role reversion. We omit the details due to space constraint.

Successive multi-path interference cancellation (SMIC). In practice, multiple paths may coexist between Tx and Rx, and the measured BCS may contain one or more B_i with diverse channel gains, expressed as:

$$B^M = \sum_{i \in \Phi} h_i B_i + n \quad (3)$$

In this case, directly applying BCS-CM will result in erroneous AoD estimation. We illustrate this point using an example BCS measurement that comprises two paths: path 1 with AoD 105° and channel gain 0.8, and path 2 with AoD 168° and channel gain 0.3. Fig. 5 plots the correlation between measured BCS and possible B_i values. While BCS-CM can accurately identify path 1 (correlation peak), it cannot

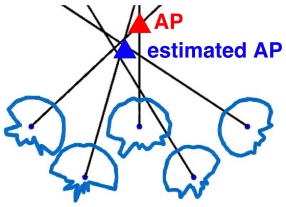

Figure 6: AP localization

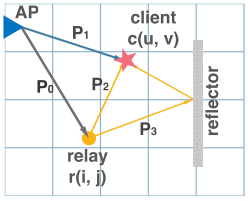

Figure 7: Cell decomp.

Algorithm 1: Path Geometry Estimation

Input: all B_i for each i_{th} direction, measured B_M
Output: the set of AoD angles Φ
Initialization: $\Phi = \varnothing$, BCS residue $B_r = B_M$
while *Eq.* (6) *doest not hold* **do**
  /*paths remian to be detected*/
  identify next path with angle i^* using Eq. (2);
  compute channel gain h_i^* using Eq. (5);
  update B_r using Eq. (4);
  $\Phi = \Phi \cup i^*$;
end

find path 2 whose correlation is immersed in noise.

To recover weaker paths, BCS-CM employs a SMIC design, inspired by successive interference cancellation [25] in communication theory. SMIC first detects the strongest path through BCS-CM. It then subtracts that path's BCS from the measured BCS and reruns BCS-CM from the residue. The process iterates until all the paths are identified. Algorithm 1 summarizes this procedure.

Realizing SMIC entails two challenges: *(i)* After detecting a path with AoD along i_{th} angle, how should SMIC obtain the h_i to be canceled? *(ii)* When to stop SMIC's iterative process, given that the number of paths is unknown? We solve the problems by further exploiting the information embedded in BCS correlation. *First*, suppose we just detected a path with i_{th} angle using Eq. (2), the problem is to estimate its channel gain $h^{'}$ as accurately as possible, so we can get a purified residue B_r after canceling $h_i B_i$ from B^M, *i.e.*,

$$B_r = B^M - h^{'} B_i = (h_i - h^{'})B_i + \sum_{j \in \Phi \& j \neq i} h_j B_j + n \quad (4)$$

From above, we can deduce that the absolute correlation between B_r and B_i will be minimized when $h^{'} = h_i$. Thus, we can estimate channel gain h_i^* as follows,

$$h_i^* = \arg\min_{h'} |\kappa(B_r, B_i)| \quad (5)$$

where we limit h' within $(0, 1)$. After deciding on h_i^*, we will get an accurate residue B_r and keep the iterative SMIC going. Continuing with the example in Fig. 5, path 2's correlation peak appears after path 1 is canceled. *Second*, SMIC should stop when no path exists in B_r, when correlation approaches noise floor $\kappa(B_r, n)$. Formally, the stopping rule is:

$$\frac{\max_{\forall i} |\kappa(B_r, B_i)|}{mean(\sum_{\forall i} |\kappa(B_r, B_i)|)} \leq \sigma \quad (6)$$

Here the smaller the threshold σ is, the more paths would be detected, but possibly with false positives. In practice, we find $\sigma = 2$ suffices. This may miss some weaker paths, but does not affect miDroid's environment sensing, which needs stronger paths to provide more reliable information.

3.2 Environment Reconstruction

Locating the AP. miDroid achieves *fully automatic AP location and orientation sensing, by exploiting mobility and self-localization ability of the robot relay.* Specifically, the robot needs to roam randomly around the AP, and select L locations with highest link throughput (most likely due to existence of a LoS paths). At each location $l \in [1, L]$, the robot derives the AoA of the strongest path, θ_l, using BCS-CM. Meanwhile, the robot relay knows its own position x_l. Given x_l and θ_l, miDroid runs a statistical method to estimate the *AP location*, illustrated in Fig. 6: corresponding to each x_l, the angle θ_l indicates the possible direction of the AP relative to the robot. In total there exist $L(L-1)/2$ intersection points among all L possible directions. We use the center of all intersection points, x_{AP}, as the final estimation. Similarly, we derive *AP orientation* by examining all possible AP orientation $0 \leq \alpha \leq 360$. Specifically, since the radios' beam patterns are known *a priori*, we can predict a BCS received at each relay position given x_{AP} and an α. Then we try each α and find the one with the prediction closest to the actual BCS measured by the robot. To minimize errors, we use the average orientation estimation over all L robot locations.

Locating reflectors and constructing environment map. miDroid reconstructs an outline of the environment by combining robotic motion and mmWave sensing. By sweeping through the space, the robot first generates an environment floor map. Meanwhile, given the AoA and AoD of all NLoS signal paths (obtained through BCS-CM), the robot can locate the points that reflect ("bend") the mmWave signal paths, using classical geometrical models [27]. miDroid then corrects the location error by matching each reflecting point to the closest point on the floor map. Furthermore, it estimates the reflection loss of each point, by subtracting the measured channel gain along the NLoS path from the ideal Friis LoS path loss. Similarly, it can estimate the penetration loss when the robot roams behind an obstacle. Following these steps, miDroid creates a map of the environment from the mmWave radios' "eyes". Note that for multi-room environment, to facilitate environment learning, the AP may need to be temporarily moved to multiple locations (but no backhaul is needed) so that its signals can hit all reflectors.

Using the learned information to drive ray-tracing. The ultimate link quality between the AP, relay and client depends on the radio beam patterns, signal propagation loss, and environmental distortion (reflection and penetration loss). We model such joint effects through a ray-tracing approach, which uses environment information (*i.e.*, AP location, reflector layout/reflectivity, blockage layout/penetration), and the measured beam patterns of each mmWave radio, as input; and outputs the RSS accordingly. Our ray-tracer follows the classical process in [27], whose accuracy has been extensively validated.

4 CLIENT TRACKING

Client tracking is much like AP-localization. To track the

client's location, the relay first derives the measured BCS between itself and the client. It then employs the ray-tracer to compute the BCS of each possible candidate location, and picks the one that best matches the measured BCS (*i.e.*, with the maximum correlation). Here the time consuming ray-tracing process (*i.e.*, predicted BCS between any pair of relay and client) is done *offline* and cached as a look-up table. Note that human dynamics can also be processed offline by enumerating BCSs under all possible combinations of path blockage and client orientation. So at run time, miDroid only needs to run a simple correlation-match computation to enable real-time client tracking. However, *location ambiguity* may arise, *i.e.*, more than one candidate locations may have similar path geometries (*e.g.*, when multiple paths arrive in parallel at different locations). miDroid resolves the ambiguity by exploiting the continuity of client's movement. Among all candidate locations that match the measured BCS, it picks the one closest to the latest client location sample. We emphasize that *miDroid's client tracking works regardless of client orientation or blockage*. This is because its BCS correlation match takes into account all possible LoS and NLoS paths between the relay and client. Even if certain paths are blocked by other users or the client herself, other paths may still survive and lead to the highest correlation between measured and ray-tracing predicted BCS.

5 ADAPTIVE PATH PLANNING

To execute the path planning, we design a statistical performance indicator called *E-index*, to quantitatively characterize how good a relay position & orientation is, based on the learned environment information. For a given client position, the relay's E-index distribution over space is called a *roadmap*. Then we model the relay path planning as a tree searching problem on the roadmap. We further exploit a 'locality' property of relay performance to reduce the model complexity, and design a practical adaptive path planning scheme for mobile client (*i.e.*, dynamic roadmaps). We now describe each design component in detail.

5.1 E-index to Model Relay Performance

A good relay position should enhance the client's resilience against rotation or blockage. Intuitively, a better relay position should manifest two characteristics: *(i)* The relay→client paths' AoAs should be sparsely distributed across all angles, and should be complementary to the AP→client paths', so that the *likelihood* of all paths being simultaneously blocked is low. *(ii)* Both the AP→relay and relay→client links should have strong RSS to ensure high throughput across the two hops.

The *E-index* metric incorporates the two characteristics, drawing on inspiration from the entropy concept in information theory [13]. Consider an arbitrary relay location and orientation which, together with the AP, creates P signal paths to a client, each path with RSS β_p and AoA ϕ_p ($\forall p \in [1, P]$). We denote the corresponding spatial channel profile as $\Psi = \{(\beta_1, \phi_1), (\beta_2, \phi_2), ..., (\beta_P, \phi_P)\}$.

To examine the AoA's spatial distribution, we split the horizontal space into S angular zones, A_s, $\forall s \in [1, S]$. For each zone, we define its *zone RSS* as the sum signal strength of all paths falling in the zone, *i.e.*,

$$RSS_s = \sum_{\forall p, \phi_p \in A_s} \beta_p \tag{7}$$

Then we normalize the zone RSS by dividing each RSS_s by the total RSS, *i.e.*, $sumRSS \triangleq \sum_{s=1}^{S} RSS_s$, as follows,

$$\hat{RSS}_s = RSS_s / sumRSS \tag{8}$$

Treating $\hat{RSS}_s, \forall s \leq S$ as a sequence of numbers that form a probability density function, we characterize how evenly the zone RSS are distributed across all angles, following the entropy definition, *i.e.*,

$$eRSS = \sum_{s=1}^{S} \hat{RSS}_s \times \log(1/\hat{RSS}_s) \tag{9}$$

Then the *E-index* η is defined as follows,

$$\eta = sumRSS \times eRSS \tag{10}$$

where *eRSS* reflects AoA sparsity to keep resilience to rotation/blockage, and *sumRSS* reflects the need for strong RSS.

Computing the spatial distribution of E-index. Note that the E-index η depends on the positions of the client and relay, and also the surrounding environment. To formalize the relationship, we first *discretize the space* into $M \times N$ cells, as shown in Fig. 7. We define C_0, C_1 as the set of cells that relays or clients can move freely, respectively. Note that C_0 can be any constrained region defined by end-users (*e.g.*, limited to within 0.5m against walls as in our experiments in Sec. 7.2), so that the relay will *not* interfere with human daily activities. Given any pair of relay and client, located in cell (i, j) and (u, v) respectively, we can derive the signal path geometries and RSS for the client following Sec 3. We then compute the E-index for the pair, $\eta(i, j, u, v)$, using Eq. (10). For a given client position (u, v), we denote the spatial distribution of E-index across all possible relay positions (i, j) as a *roadmap*.

Determining the optimal relay position. For a client position $c(u, v) \in C_1$, the optimal relay position $r(i^*, j^*)$ can be derived as

$$(i^*, j^*) = \arg\max_{\forall r(i,j) \in C_0} \eta(i, j, u, v) \tag{11}$$

Fig. 8 plots the roadmaps as a client moves across 10 locations in an example scenario. The optimal relay positions within each roadmap are highlighted. We can observe the unique challenges in navigating the robotic mmWave relay. In particular, the *legacy path planning algorithms* (*e.g.*, A^*, D^*, *etc.* [3]) navigate a robot along the shortest feasible path. But they cannot guarantee network performance points along the path, which is critical for mmWave connectivity. Moreover, the *roadmap keeps changing* after each movement of the client, which renders legacy path planning infeasible.

5.2 Tree Model over Dynamic Roadmap

To handle dynamic roadmaps and ensure intermediate relay performance, we build a tree searching model, as illustrated in Fig. 9. The root is the initial relay position; the children nodes are its reachable neighboring cells, and the

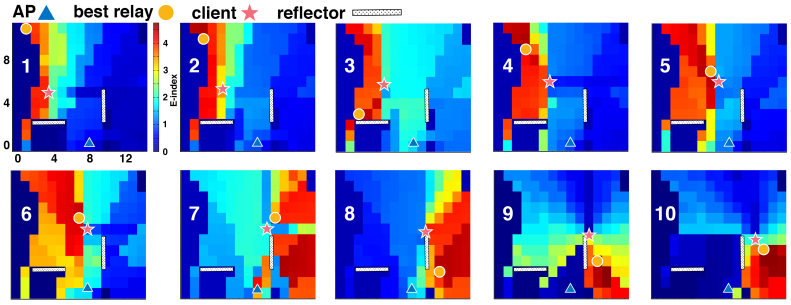

Figure 8: Relay locality: bright(red) color represents high E-index

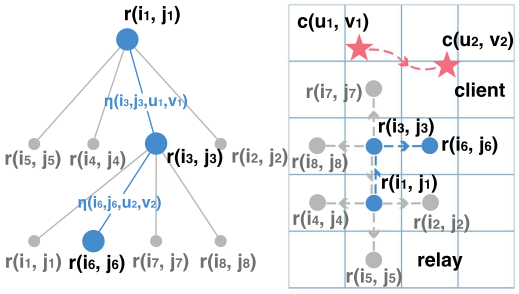

Figure 9: Tree model

Algorithm 2: Adaptive Path Planning Algorithm

Input: current client location c_0 and velocity v_0
Output: movement command to the robot
while *True* **do**
  Predict K client locaions $\{c_1, ...c_K\}$ using Eq. (12);
  Derive roadmaps $\{f(1), ...f(K)\}$;
  Derive $\lambda(k)$, $\mu(k)$ of each $f(k)$'s good relay zone;
  if *Eq.* (13) *holds* **then**
    /*the optimal relay is nearby*/
    Search K depth over the tree model;
    Derive the optimal K-step relay path, $p*$;
    Command the relay move one step along $p*$;
  end
  else
    /*the optimal relay is far away*/
    Compute the optimal relay with max. E-index;
    Derive the optimal path using Dijkstra algo.;
    Command the relay moving along the path;
  end
  update c_0 and v_0 at the new relay position;
end

edge weight is the E-index of the corresponding child. In this way, we essentially model roadmap dynamics (*i.e.*, varying E-index) through different E-index weight at different tree depth. For instance, if the relay plans to move from the root $r(i_1, j_1)$ to $r(i_3, j_3)$ at depth-1, the edge weight is $\eta(i_3, j_3, u_1, v_1)$ given the current client location $c(u_1, v_1)$. Suppose the client also moves to a new location $c(u_2, v_2)$ as the relay moves to $r(i_3, j_3)$, then the edge weights to depth-2 nodes will be the E-index when the relay moves to $r(i_3, j_3)$'s neighbors, *e.g.*, $\eta(i_6, j_6, u_2, v_2)$, if the relay plans to move to $r(i_6, j_6)$ for the next step.

From an oracle's perspective, the optimal relay path can be derived using a brute-force tree searching, assuming the client's trajectory is known *a priori*. However, two challenges arise in practice: *(i)* Predicting client movement is a non-trivial and error-prone task. *(ii)* The search complexity is $O(P^d)$, where P is the cardinality of children (*i.e.*, 4 if we limit the robot to straight-line movement), and d is tree depths (hundreds to thousands scale). Clearly, the computational cost and resulting delay is unaffordable.

5.3 Adaptive Tree Searching

miDroid curtails the path planning complexity through an adaptive tree searching (ATS) algorithm, which harnesses a "locality" property of E-index. As illustrated in Fig. 8, positions with high E-index tend to form a cluster (henceforth referred to as *good-zone*). Relay performance is consistently high inside the zone. More importantly, the good-zone moves forward gradually along with client movement, until an abrupt environmental change (*e.g.*, the client passes a major obstacle, such as a large TV screen, shown in subplot 7). Afterwards, a new good-zone forms and the phenomenon repeats (*i.e.*, subplot 7, 8, 9 and 10). In fact, the locality property roots in the spatial channel sparsity inherent to mmWave channels. Specifically, for a given client position, a few major signal paths dominate the performance, whereas the path strengths and incident angles gradually evolve with user mobility, unless sudden environmental change blocks the paths and/or creates new ones. We can also observe that a good-zone commonly includes many cells, which hints that miDroid robot may find near-optimal location to move to *even if it is constrained to a certain mobility region.*

The *locality of E-index implies that short-depth tree searching suffices, based on short-term prediction of user movement.* Accordingly, ATS operates as follows (Algorithm. 2). *(i)* It estimates K time slots of client user's future movement using first-order prediction, *i.e.*, For each kth slot, the predicted client location is:

$$c_k = c_0 + v_0 * T_k \quad \forall 1 \le k \le K \tag{12}$$

where c_0 and v_0 are the client's current location and velocity, and T_k is the time from now to k. Here we set $T_k = k * BI$, where BI is the beacon interval (typically below 100 ms) during which client localization executes. *(ii)* For each slot, ATS computes the relay's good-zones, and checks whether the zones experience sudden shift. We detect a shift by examining how far the good-zones of consecutive roadmaps are separated from each other. Specifically, for two consecutive roadmaps denoted with $f(k)$ and $f(k+1)$, ATS derives the good-zone centers ($\lambda(k)$ and $\lambda(k+1)$) and radius ($\mu(k)$ and $\mu(k+1)$), using the classical clustering algorithm [6]. It then examines whether the L-1 distance of the centers is larger than the average radius, *i.e.*,

$$|\lambda(k+1) - \lambda(k)| \le \frac{\mu(k+1) + \mu(k)}{2}, \quad \forall k < K \tag{13}$$

If Eq. (13) holds, the relay will search the tree for K steps, and derive an optimal moving trajectory. To minimize the effect of wrong prediction, the relay moves only one step along the trajectory, then it repeats the K-step-prediction &

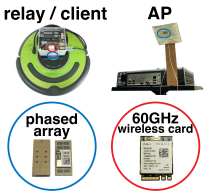

Figure 10: miDroid prototype.

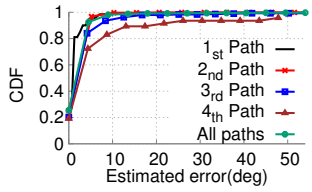

Figure 11: path estimation error.

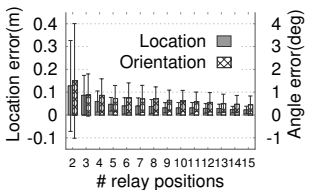

Figure 12: AP estimation error.

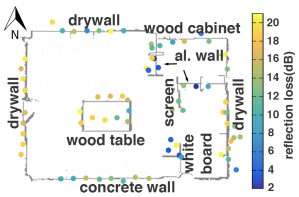

Figure 13: Environment learned by miDroid.

one-step-movement procedure. If Eq. (13) does not hold, then zone shift happens. The relay will be aware that the optimal relay position is far away. Then it computes the optimal relay with the maximum E-index according to Eq. (11), and moves towards it along a path computed from the Dijkstra algorithm. Though Dijkstra algorithm cannot guarantee path optimality, in this case the priority is to reach the new good-zone as soon as possible. Also, note that user movement prediction at sub-second level is of high reliability [26]. Our evaluation result (Sec. 7) will further show that 3 look-ahead steps ($K = 3$) already lead to high relay performance.

6 IMPLEMENTATION

Hardware components. Our miDroid system prototype consists of an AP, relay and client, as depicted in Fig. 10. The AP is a small PC mounted on top of a tripod of 1.8m-height, and the relay and client are Intel NUC mini-PCs (around 4 inches on all dimensions), mounted on Create 2 programmable robots [8]. These devices are all equipped with the Qualcomm 60 GHz network interface card [18], comprised of QCA6310 RF front-end, QCA6320 MAC/baseband, and a 32-element phased array (supporting 36 beams). The NUC controls the robot through maneuvering commands (*e.g.*, translation/rotation directions, speed and duration). The commands for the relay robot follow miDroid's path planning algorithm, whereas the client robot is programmed to move freely like a vanilla user. miDroid can either use the decode-and-forward [18] or amplify-and-forward relaying [1], both already demonstrated in practical mmWave system design[1, 21]. In this work, we emulate the amplify-and-forward mode by ignoring the time-division-multiple-access overhead across the two hops.

Real-time implementation of miDroid modules. We implement miDroid's major design components (*i.e.*, environment reconstruction, client tracking, path planning algorithms, and also the ray-tracer) in Matlab, which run on the relay's NUC. To facilitate *real-time path planning*, we offload the computation of offline sensing (Sec.3) and E-index computation (Sec. 5.1) on a PC server, and feed the results to the AP and relay in the form of static data structures and lookup tables. For instance, after environment learning, miDroid computes the E-index $\eta(i, j, u, v)$ for all pairs of relay $r(i, j)$ and client $c(u, v)$ offline, which avoids the notoriously time-consuming ray-tracing. we emphasize that this offline processing only needs to be done once for each environment. Our experiments reveal that, the offline processing takes less than one hour when running on an ordinary laptop PC, for a sophisticated multi-cubicle office environment (Fig. 24). Then at running stage, only efficient table-look-up and basic arithmetic operations are needed. The overall *latency* is less than 10ms for miDroid making each movement decision in our current implementation, and can be further reduced if miDroid is incorporated and runs inside low-layer NIC driver module. A similar offline caching mechanism is implemented for client localization (see Sec. 4). Recall that the client localization is based on the beacon RSS measurements at 802.11ad's BI intervals (100 ms), which is fast enough for real-time decision making.

7 EXPERIMENTAL EVALUATION

7.1 Micro Benchmarks

7.1.1 *Environment Learning.*

Signal path estimation. We first verify the effectiveness of BCS-CM and SMIC in isolating multiple NLoS/LoS signal paths and estimating their geometries. We set up 100 Tx-Rx pairs randomly located in a typical office (*i.e.*, $8 \times 10m^2$ office with multiple cells, separated by concrete walls in Fig.13) and get the ground-truth from measurements using a software radio with a 3° horn antenna. Fig. 11 plots the CDF of AoA estimation error (we omit AoD as the results are similar). We have two major observations: *(i)* miDroid extracts up to 4 paths before its SMIC iteration terminates. This echoes previous measurement insights that only $3 \sim 5$ major paths exist in typical indoor environment due to channel sparsity [23, 27]. *(ii)* miDroid can accurately estimate signal paths, with a small mean error 1.64° across all paths. A further examination reveals that the error tends to be higher for weaker paths due to residual inter-path interference. However, even for the 4_{th} paths immersed under the previous three, the average error is only 7.31°.

AP localization. To verify miDroid's AP localization method, we place the AP to 10 random locations with arbitrary orientations. Fig. 12 shows the mean error and std. As the robot roams to more points to collect more AoA/AoD samples, the AP sensing errors keep decreasing. Even with 5 sampling points, miDroid can already estimate the AP's location (orientation) with a small mean error of 5 cm (1°). Converging to this small error takes only around 1 minute.

Reflector reconstruction. We let miDroid relay scan the environment following Sec. 3, and depict the estimated reflecting points along with the robot-generated floor map in Fig. 13. We note that: *(i)* The positions of reflecting points closely match the ground truth. The mean error is only 0.17m over all estimated points. *(ii)* The estimated reflection loss

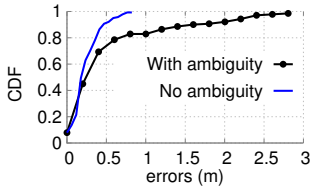

Figure 14: CDF of client tracking errors.

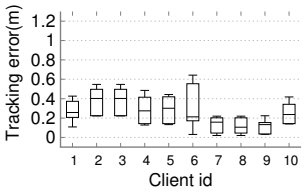

Figure 15: Impact of rotation on client tracking.

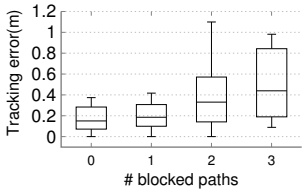

Figure 16: Impact of blockage on client tracking.

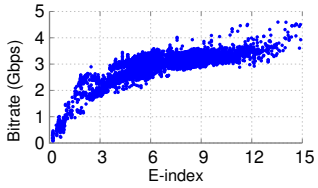

Figure 17: Mapping between E-index and bitrate.

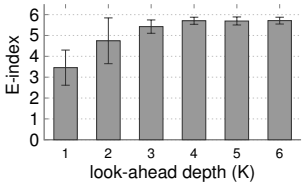

Figure 18: Impact of look-ahead steps.

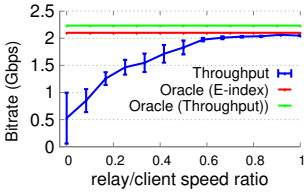

Figure 19: Effect of relay moving speed.

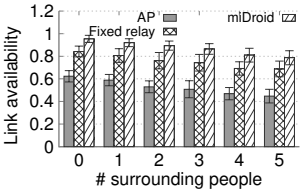

Figure 20: Link availability.

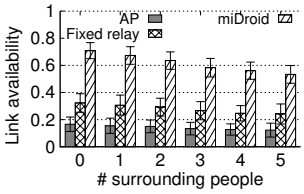

Figure 21: link availability in the complicated lab env.

(represented with different colors) well matches the material properties reported in [27].

7.1.2 *Client tracking.*

Accuracy of client tracking. We now let a client user move continuously along a randomly-generated trajectory, and the relay navigates itself according to the path planning algorithm (Sec. 5). We repeat the experiment across 10 trajectories, and pick 10 equally-spaced positions along each trajectory to verify the client tracking accuracy. From the results in Fig. 14, we observe that: *(i)* The location ambiguity significantly undermines localization accuracy. While 70% points have an error less than 0.5m, there exist points with up to 3m error. *(ii)* Fortunately, miDroid's ambiguity-removing mechanism effectively removes outliers and brings the average error down to 0.23m.

Robustness to client rotation and blockage. Using the same setup as above, we let the client rotate in steps of 30° at each location. The box plot in Fig. 15 shows the client tracking error, at 10%, 25%, 50%, and 90%. We see that the 90-th errors are below 0.62m and median below 0.4m, regardless of the client's position and orientation. We further select 10 client locations that have 4 paths between the relay, and intentionally block 1 to 3 paths. Fig.16 plots the localization errors. Though the average location error keeps increasing, it remains less than 0.5m even when 3 paths are blocked. The experiments validate that *miDroid's client tracking ability is resilient to user rotation dynamics, and does not require the existence of a LoS path as in traditional phased-array localization schemes [29].*

7.1.3 *Path planning.*

How well does E-index model spatial performance? Recall that E-index is a probabilistic metric, intended to reflect potential link performance. We now evaluate it against the ground-truth link bitrate. Specifically, we select 100 relay-client pairs with random locations. For each pair, we first rotate the client to 20 random directions, and then block the client 20 times with different path combinations. For each run, we measure the link RSS and map it to 802.11ad bitrate as in [7]. Then we compute the average bitrate over all rotation and blockage settings for each client, and plot it against the client's E-index. The results (Fig. 17) show that: *(i)* The RSS fluctuates even for a fixed E-index, which roots from the probabilistic entropy-like definition of E-index. *(ii)* A clear positive logarithmic-correlation holds between E-index and RSS, as expected from the log term in Eq. (9). Therefore, *E-index can indeed quantitatively represent relay performance.*

Relay path optimality. We run miDroid's path planning algorithm over the aforementioned 10 client trajectories. For each trajectory, we examine the relay performance by comparing the mean E-index (*i.e.*, over all points along the trajectory) under different look-ahead depths. Fig. 18 shows that: *(i)* E-index increases from 3.4 to 5.6 when look-ahead depth K increases from 1 to 6, because look-ahead offers more information especially about the abrupt environmental changes, so that miDroid can prepare in advance to avoid link outage. *(ii)* More importantly, the gain quickly saturates as look-ahead depth increases to 3. The finding further corroborates the locality effect of relay performance (Sec. 5): the optimal relay position (corresponding to the next client position) is often close by. So, even with a small look-ahead depth, *miDroid can achieve high performance while avoiding the exponential computational complexity, which paves the way for real-time relay path planning.*

Impact of moving speed. We repeat the above 10-trajectory experiment, but vary the robot's speed from 0.1 to 1 (relative to the client's speed 0.8m/s). For comparison, we also compute the performance of Oracle(E-index) and Oracle(throughput), which assumes an infinite-speed relay that can immediately move to the position with the maximum E-index and throughput, respectively. Fig. 19 shows that: although a faster relay may reach the optimal position sooner, the benefit converges at a speed of 0.6 owing to the locality property. Therefore, *the robot relay does not need to follow the client's speed. It can keep near/in the optimal relay zone even when it moves much more slowly.*

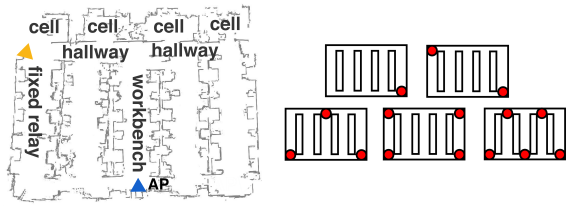

Figure 22: A larger lab and multi-AP placement.

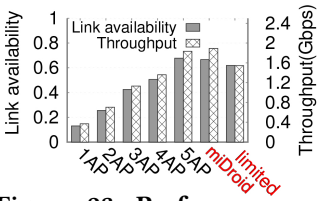

Figure 23: Performance of miDroid vs. multi-AP

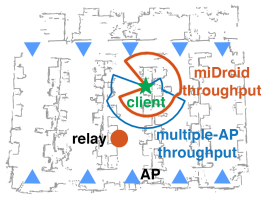

Figure 24: Multi-AP unable to eliminate blind angles.

7.2 System Level Evaluation

Now we evaluate miDroid's performance end to end.

Link availability. We now conduct a system-level evaluation of miDroid in the typical indoor scenario of Fig. 13, with one user holding the client device, while up to 5 other people walking around causing dynamic blockage. We compare miDroid with two baselines: AP-only, and fixed-relay (placed opposite to the AP to provide the most complementary coverage). We first evaluate link availability over 10 random client trajectories. *Link availability* is defined as the percentage of time when throughput exceeding a threshold T (here defined as half of the maximum throughput of our 802.11ad radios). The results (Fig. 20) show that miDroid achieves 87.2% availability on average across all the setup, in comparison to 52.8% and 75.5% for AP-only and fixed-relay, respectively. In particular, miDroid's link availability is around 95.4% when no other people exist, achieving a nearly seamless coverage for the client. We emphasize that the key merit of miDroid lies in the ability of eliminating *worst case performance* (link outages and disconnections), which is the bottleneck to user QoS. Such cases mostly happen when the client moves into a blind spot for the AP or fixed-relay. By investigating the trace corresponding to Fig. 20, we find the results that validate our intuition: for the 3 client positions with lowest performance, miDroid can increase link availability from 35% to 95% with the client alone, and to 88% even with two other active people.

We also repeat the link availability test in a larger and more complicated 14×16 m^2 lab space (Fig. 22), with 4 half-open cells and 6 arrays of workbenches that can easily block the mmWave links. We can observe from the results in Fig. 21 that *even in a challenging environment, miDroid demonstrates superior performance. It improves link availability by more than 2×, compared with the other two solutions.*

Performance comparison with multiple fixed APs. We compare miDroid against a multi-AP solution. In particular, we increasingly deploy from 1 AP to 5 APs in the lab space, which are placed with complementary FoVs (Fig. 22). Measurement results (Fig. 23) show that: miDroid's link availability and throughput is comparable to 5-AP. Even when the robot moves in a highly constrained region (within 0.5m against walls), miDroid's performance (labeled "limited") is still better than 4-AP, and close to 5-AP.

We also showcase an extreme example in Fig. 24, where 10 APs are deployed, and a client is located near the corner of a workbench. The polar graphs show the circular throughput for multi-AP (blue curve) and miDroid (orange curve), respectively. We can observe that even 10 APs cannot provide a stable link due to the existence of blind angles, whereas miDroid achieves a near-omnidirectional coverage.

Discussion on system extension. Note that miDroid relay does not stay close with a client, but elaborately navigates itself to relaying positions that can best harness reflection paths to maximize the E-index for the client. Such methodology enables a natural extension to support multiple clients, by setting other optimization objectives, *e.g.*, maximizing the sum of all users' E-index, or maximizing the minimum E-index across all users. How to choose an appropriate objective and balance trade-off between throughput and fairness shall be an interesting problem. In addition, the current miDroid involves only one AP associated with the robotic relay, and we focus on solving challenges of client tracking and relay path planning. For an extended scenario involving multiple APs and even multiple relays, new challenges will arise including AP switching and movement coordination among relays. We leave such exploration for future work.

8 RELATED WORK

Mobile relaying for mmWave networks. Lack of coverage and stability is a major issue for mmWave networks. Substantial research [5, 19, 22–24, 32, 33] focused on efficient beam steering methods to rapidly recover from mobility/blockage disruptions. Yet even perfect beam steering cannot overcome the blind zones due to limited FoV [26].

Recently, mobile relay/robotics systems [2, 4, 12, 30] started gaining traction due to wide availability of drones and robots. However, these systems operate with low-frequency omni-directional radios, and hence they do not bear the challenges unique to mmWave networks. The benefits of mmWave relaying have been verified theoretically [9, 10], and through preliminary measurements [21]. *Static* mmWave relays or APs have been exploited [1, 26] to overcome the limited FoV. We have provided experimental evidence to show the limitations of such systems, and the advantages of an autonomous mobile relay.

Path planning, as one fundamental problem in robotics, has been extensively investigated [3, 30]. The classical A^* algorithm and its adaptations including D^* and D^*-lite navigate a robot to static/mobile destination while avoiding obstacles. Whereas miDroid shares similar spirit, the objective of robotic mmWave relay differs in that the planned path needs to guarantee the overall statistical link performance comprised of all *signal propagation paths*, instead of simply being the optimal geometrical path towards a number of points of interest. Moreover, mmWave's strong depen-

dence on environment and thus the dynamic roadmap make previous path planning mechanisms infeasible. *Essentially, miDroid solves a set of unique problems in the intersection between mmWave networking and robotics.*

Environment learning and localization. To sense surrounding reflectors, existing systems either move a pair of mmWave Tx/Rx radios following a deliberately designed trajectory to scan a specific reflector [34, 35], or place radios at specific site-survey points [27]. In particular, E-Mi [27] uses a multi-path resolution framework to estimate the AoA/AoD of signal paths. But it relies on both RSS and phase measurements on a custom-built software radio, due to lack of coherent phase information on COTS devices. In addition, it entails non-trivial human intervention to execute the site measurement. In contrast, miDroid integrate a robot's own sensing capabilities with mmWave sensing mechanisms, and uses RSS measurement alone on low-profile COTS devices to realize autonomous environment mapping.

Recent work also proposed device tracking methods to speed up mmWave beam alignment [11, 16]. miDroid's AP/client location sensing module differs in that it leverages precise motion control and diverse beam patterns (*i.e.*, BCS) of the robotic mmWave relay. miDroid accurately senses both location and orientation using COTS mmWave radios which can only measure RSS. It does not need phase measurement, and works even when the LoS path is blocked. In addition, mmWave localization also differs from many counterparts operating at low-frequency (see [17, 29, 31] and references therein), due to new radio characteristics.

9 CONCLUSION

This paper envisions the proliferation of consumer-grade robots, and proposes a robotic mmWave system, miDroid, to achieve seamless mmWave coverage. Through miDroid, we have demonstrated the feasibility of leveraging the autonomous motion of a robot to reverse engineer the geometrical properties of wireless channel. We have also explored new robotic path planning solutions that navigate the robot within radio environment, instead of physical environment alone, to facilitate network performance. We believe miDroid hints on a new direction that embraces robotic intelligence into wireless network design and operations.

Acknowledgment The work was supported by National Natural Science Foundation of China (61772084, 61720106007, 61832010, 61532012) and the 111 Project (B18008).

REFERENCES

[1] Abari, O., Bharadia, D., Duffield, A., and Katabi, D. Enabling high-quality untethered virtual reality. In *USENIX NSDI* (2017).

[2] Dhekne, A., Gowda, M., and Choudhury, R. R. Extending cell tower coverage through drones. In *ACM HotMobile* (2017).

[3] Goerzen, C., and etc. A survey of motion planning algorithms from the perspective of autonomous uav guidance. *J. Intell. Robotics Syst. 57*, 1-4 (Jan. 2010), 65–100.

[4] Gowda, M., and etc. If wifi aps could move: A measurement study. *IEEE Transactions on Mobile Computing* (2018).

[5] Haider, M. K., and Knightly, E. W. Mobility resilience and overhead constrained adaptation in directional 60 ghz wlans: Protocol design and system implementation. In *ACM MobiHoc* (2016).

[6] Han, J., Kamber, M., and Pei, J. *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann Publishers, CA, USA, 2011.

[7] IEEE Standards Association. IEEE Standards 802.11ad-2012: Enhancements for Very High Throughput in the 60 GHz Band, 2012.

[8] iRobot Inc. iRotot Create 2 Programmable Robot. http://www.irobot.com/About-iRobot/STEM/Create-2.aspx, 2018.

[9] Kong, L., and etc. Autonomous relay for millimeter-wave wireless communications. *IEEE JSAC 35*, 9 (Sept 2017), 2127–2136.

[10] Kwon, S., and Widmer, J. Relay selection for mmwave communications. In *IEEE PIMRC* (Oct 2017).

[11] Loch, A., and etc. Zero overhead device tracking in 60 ghz wireless networks using multi-lobe beam patterns. In *ACM CoNEXT* (2017).

[12] Ma, Y., Selby, N., and Adib, F. Drone relays for battery-free networks. In *SIGCOMM* (2017), ACM.

[13] MacKay, D. J. C. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA, 2002.

[14] MATLAB. corrcoef. https://www.mathworks.com/help/matlab/ref/corrcoef.html, 2018.

[15] MIT-tr. Robots are becoming ready to work among us. http://www.technologyreview.com/view/522646/robotsare- becoming-ready-to-work-among-us, 2018.

[16] Palacios, J., and etc. Jade: Zero-knowledge device localization and environment mapping for millimeter wave systems. In *IEEE INFOCOM* (2017).

[17] Qian, K., Wu, C., Zhang, Y., Zhang, G., Yang, Z., and Liu, Y. Widar2.0: Passive human tracking with a single wi-fi link. In *ACM MobiSys* (2018).

[18] Qualcomm Inc. Qualcomm VIVE 802.11ad. https://www.qualcomm.com/products/vive/11ad, 2015.

[19] Rasekh, M. E., Marzi, Z., Zhu, Y., Madhow, U., and Zheng, H. Noncoherent mmwave path tracking. In *ACM HotMobile* (2017).

[20] Rob Sun. IEEE 802.11 TGay use cases 11-15/0625r7. https://mentor.ieee.org/802.11/dcn/15/11-15-0625-07-00ay-ieee-802-11-tgay-usage-scenarios.pptx, 2017.

[21] Saha, S. K., and etc. Improving connectivity, coverage, and capacity in 60 ghz indoor wlans using relays. In *MobiCom S3 workshop* (2015).

[22] Sur, S., Pefkianakis, I., Zhang, X., and Kim, K.-H. Wifi-assisted 60 ghz wireless networks. In *MobiCom* (New York, NY, USA, 2017), ACM.

[23] Sur, S., Zhang, X., Ramanathan, P., and Chandra, R. Beamspy: Enabling robust 60 ghz links under blockage. In *USENIX NSDI* (2016).

[24] T, N., and etc. Steering with Eyes Closed: mm-Wave Beam Steering without In-Band Measurement. In *IEEE INFOCOM* (2015).

[25] Tse, D., and Viswanath, P. *Fundamentals of Wireless Communication*. Cambridge University Press, New York, NY, USA, 2005.

[26] Wei, T., and etc. Pose information assisted 60 ghz networks: Towards seamless coverage and mobility support. In *ACM MobiCom* (2017).

[27] Wei, T., Zhou, A., and Zhang, X. Facilitating robust 60 ghz network deployment by sensing ambient reflectors. In *USENIX NSDI* (2017).

[28] Xiaomi. Mi Robot Vacuum Cleaner. https://www.amazon.com/Xiaomi-Cleaner-Guidance-Powerful-Planning/dp/B01MU4WAUI, 2018.

[29] Xiong, J., and Jamieson, K. Arraytrack: A fine-grained indoor location system. In *USENIX NSDI* (2013).

[30] Yan, Y., and Mostofi, Y. Efficient clustering and path planning strategies for robotic data collection using space-filling curves. *IEEE Trans. Control of Network Systems 4*, 4 (2017), 838–849.

[31] Yang, Z., Zhou, Z., and Liu, Y. From RSSI to CSI: indoor localization via channel response. *ACM Comput. Surv. 46*, 2 (2013), 25:1–25:32.

[32] Zhou, A., Wu, L., Xu, S., Ma, H., We, T., and Zhang, X. Following the shadow: Agile 3-d beam-steering for 60 ghz wireless networks. In *IEEE INFOCOM* (2018).

[33] Zhou, A., Zhang, X., and Ma, H. Beam-forecast: Facilitating mobile 60 ghz networks via model-driven beam steering. In *IEEE INFOCOM 2017* (May 2017).

[34] Zhu, Y., Yao, Y., Zhao, B. Y., and Zheng, H. Object recognition and navigation using a single networking device. In *ACM MobiSys* (2017).

[35] Zhu, Y., Zhu, Y., Zhao, B. Y., and Zheng, H. Reusing 60ghz radios for mobile radar imaging. In *ACM MobiCom* (2015).