Can 5G mmWave support Multi-User AR?

Moinak Ghoshal[1], Pranab Dash[2], Zhaoning Kong[2], Qiang Xu[2], Y. Charlie Hu[2], Dimitrios Koutsonikolas[1], and Yuanjie Li[3]

[1] Northeastern University, USA {ghoshal.m,d.koutsonikolas}@northeastern.edu
[2] Purdue University, USA {dashp,kong102,xu1201,ychu}@purdue.edu
[3] Tsinghua University liyuanjie08@gmail.com

Abstract. Augmented Reality (AR) has been widely hailed as a representative of ultra-high bandwidth and ultra-low latency apps that will be enabled by 5G networks. While single-user AR can perform AR tasks locally on the mobile device, multi-user AR apps, which allow multiple users to interact within the same physical space, critically rely on the cellular network to support user interactions. However, a recent study showed that multi-user AR apps can experience very high end-to-end latency when running over LTE, rendering user interaction practically infeasible. In this paper, we study whether 5G mmWave, which promises significant bandwidth and latency improvements over LTE, can support multi-user AR by conducting an in-depth measurement study of the same popular multi-user AR app over both LTE and 5G mmWave.
Our measurement and analysis show that: (1) The E2E AR latency over LTE is significantly lower compared to the values reported in the previous study. However, it still remains too high for practical user interaction. (2) 5G mmWave brings no benefits to multi-user AR apps. (3) While 5G mmWave reduces the latency of the uplink visual data transmission, there are other components of the AR app that are independent of the network technology and account for a significant fraction of the E2E latency. (4) The app drains 66% more network energy, which translates to 28% higher total energy over 5G mmWave compared to over LTE.

1 Introduction

Augmented Reality (AR) promises unprecedented interactive and immersive experiences to users by augmenting physical objects in the real world with computer-generated perceptual information. As such, a complete AR app often needs to perform several challenging tasks to understand and interact with the physical environment, such as pose estimation or object detection [1].

While single-user AR can potentially perform AR tasks locally on the mobile device [9], *multi-user AR apps*, also known as networked AR apps, which allow multiple users to interact within the same physical space, critically rely on the cellular network and often a cloud server to support user interactions. Further, to provide high-quality, interactive experience, such networked AR apps need to perform the needed AR tasks (e.g. pose estimation and synchronization to the same physical environment) at very low latency, which places high uplink bandwidth demand on the wireless network. It is because of this stringent network requirement that networked AR has been widely viewed as one of the "killer" apps for 5G [10, 29, 38], e.g., in the AT&T and Microsoft alliance as well as the Verizon and AWS alliance to showcase 5G edge computing solutions [11, 39].

Previously, Apicharttrisorn *et al.* conducted an in-depth measurement study [8] of a popular two-user app that performs the most basic multi-user interaction, i.e., displaying an object, to study whether LTE can support the needed QoE of multi-user AR. That study showed that the latency from the moment a user (host) places a virtual object in the physical environment to the moment a second user (resolver) sees that object in their screen is 12.5 s in the median case and can be as high as 26 s over LTE, which renders the most basic user interaction in multi-user AR apps practically infeasible.

5G mmWave is being rapidly deployed by all major mobile operators promising ultra-high bandwidth and lower latency compared to 4G LTE. As an example, Table 1 shows the uplink and downlink TCP throughput (measured with iperf3), the end-to-end (E2E) round trip latency (measured with ping), and the RAN latency (approximated as the round trip latency to the first hop router) between a mobile device and a Google Cloud server. We observe that 5G mmWave offers 16x higher downlink throughput and 3.4x higher uplink throughput compared to LTE while it reduces the RAN (E2E) latency by 56% (42%).

Table 1: Throughput and Latency comparison over 5G mmWave and LTE.

	Throughput (Mbps)		**Latency (ms)**	
	Downlink	Uplink	RAN	E2E
5G	1715±57	152±6	14±2	25±4
LTE	110±17	44±8	32±5	43±4

Driven by these initial observations, in this paper we revisit the previous feasibility study of multi-user AR over cellular networks by conducting an in-depth measurement study of the same popular multi-user AR app side-by-side over both LTE and 5G mmWave. Our dataset is publicly available [2]. Our study tries to answer two key questions: (1) Can 5G mmWave provide much better support for multi-user interactions in AR compared to LTE to the extent that real-time multi-user interaction becomes feasible? (2) Does multi-user AR drain significantly more energy under 5G compared to under LTE?

The main findings of our study are as follows: (1) The E2E latency over LTE is significantly lower (by 6.6 s) compared to the values reported in [8], however, it remains too high for real-time multi-user AR apps. (2) 5G mmWave does not reduce the E2E latency of the AR app compared to LTE in spite of its much higher bandwidth and lower RTT. (3) While 5G mmWave yields a small reduction to the latency of the uplink visual data transmission, there are other components of the AR app that contribute significantly to the E2E latency regardless of the underlying cellular technology. In addition, we discovered a new latency component between the cloud and the resolver, which was not reported in [8], and is often a major contributor to the E2E latency. (4) The app drains on average 66% more network energy over 5G mmWave compared to over LTE. Since the network energy accounts for about 32% of the total energy, such high network energy difference translates into smaller (but still significant) difference in the total app energy drain, by 23% on the host and 43% on the resolver.

2 Background and Related Work

2.1 Multi-user AR

Current mobile AR systems like Google ARCore [4], Apple ARKit [3], and Microsoft Hololens [6] use SLAM to construct a 3D coordinate structure of the physical world and get an estimation of the user's location and orientation (pose). The users first need to share their coordinates to create a common and consistent real-world coordinate system. Once a virtual object is placed on the screen, SLAM is run to get an estimation of the device's current pose and the real-world coordinate features, and objects in the user's field of view are rendered on the screen. Popular multi-user AR apps on the market, enabled by Google ARCore, Apple ARKit, or Microsoft Hololens offload most of the computations to cloud servers to reduce the workload on the phones. In the following, we briefly describe the workflow of such applications, shown in Fig. 1.

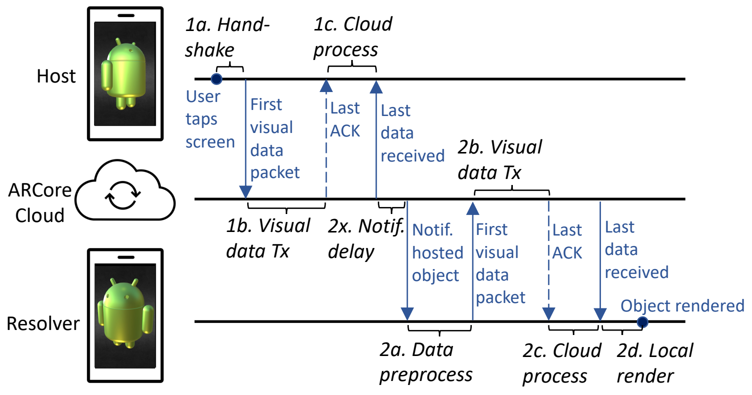

Fig. 1: Cloud-based multi-user AR.

The host initiates a connection with a cloud based Firebase [5] database by creating a room ID (R). The resolver uses the same room ID and waits for incoming connections from the host via the cloud. After an object is placed on the host's screen, the following events take place.

1. Hosting device:(a) Device Handshakes. The host places an object and two connections to Google Cloud are instantiated for object positioning. **(b) Visual Data Transmission.** The host sends the real world visual information about the overlaid virtual object to the cloud. **(c) Cloud Processing.** The cloud processes the host's visual data. It sends back the SLAM-computed world frame to the host and notifies the resolver to start the resolution process.

2. Resolving device:(a) Cloud Connection Initiation. The Firebase notifies the resolver to start a connection with the Google Cloud instance. The resolver scans the world frames through camera and pre-processes the data. **(b) Data Transmission.** On getting notified by the cloud, the resolver uploads its visual data to the cloud. **(c) Cloud Processing.** The cloud, on receiving the resolver's frames, tries to match them against the host's SLAM-computed data, estimates the pose of the resolver in the world frame and send its back to the resolver. If the cloud processing fails (e.g., because the environment lacks visual features, such as high contrast edges, colors, etc.), the cloud asks the resolver to upload new visual data. Hence, this process might involve multiple rounds of communication and cloud processing. In the following, we include only the first round of communication in the data transmission delay (2b), while any additional rounds of communication are included in the cloud processing delay (2c). **(d) Object Rendering.** The resolver uses the data from the cloud to estimate the virtual object's pose and display it on its screen.

We note that there may be an additional delay before the notification of the hosted object is received by the resolver, denoted as **2x: Notification delay** in Fig.1. This delay was not reported in [8], but it is often a major contributor to the E2E latency in our experiments.

2.2 Related Work

Multi-user AR. Unlike single-user AR (e.g., [31, 12, 9, 22]), there have been very few works on multi-user AR. A few works [44, 30, 32] focus on application layer sharing while our work focuses on the impact of the cellular network in multi-user AR performance. In contrast to [8], which studies multi-user AR performance over LTE, our work is the first to our best knowledge to study the performance and energy consumption of multi-user AR over 5G mmWave. A few recent works study edge-assisted [34] or P2P-based [33] multi-user AR. In contrast, our work focuses on cloud-assisted multi-user AR, which is the default approach in most popular AR apps on the market.
5G mmWave performance. A few recent studies focus on early-stage 5G mmWave performance and its impact on downlink-oriented mobile apps (web browsing and video streaming) [24–26]. To our best knowledge, there is no other work studying the impact of 5G mmWave on multi-user AR, which has very different application and communication features compared to web browsing or video streaming.

3 Methodology

Multi-user AR Application. Google's Cloud Anchor API [4] forms the foundation for most of the cloud-based, multi-user AR Android apps today. We used Google's popular multi-user application, Cloud Anchor, which was also used in [8]. The application lets a user place a virtual object on a real-world surface while another user can view it.
Devices. We used two Google Pixel 5 phones for our experiments. For the measurements involving the LTE network, we disabled the 5G radio through the phone's settings.
5G Carrier and Location. We conducted uplink throughput measurements in three different cities over two different cellular operators (Table 2). Based on

Table 2: 5G mmWave Uplink Throughput for different operators and cities.

Operator and City	**Throughput (Mbps)**
Verizon, Boston	152±6
Verizon, Chicago	47±15
Verizon, Indianapolis	43±5
AT&T, Indianapolis	150±50

these measurements, we selected Boston and Verizon for our experiments in this work, as that was the combination that provided the highest throughput. We used Verizon's NSA-based 5G service that provides mmWave coverage over the 28/39 GHz frequency bands (n260/261).

Experimental Methodology. We conducted our experiments near the downtown of Boston, at two different locations. At each location, we stood 80 ft away from the base station (BS); we confirmed via SpeedTest measurements that this distance yielded the maximum possible uplink throughput. The experiments at each location spanned a 1-week period. All measurements were done at day time, from 9 am to 5 pm. For 5G mmWave, we consider two cases – when the users face towards the BS and when they face away from the BS; in the later case, their bodies block the Line of Sight (LOS) between the BS and the UE.

Measurement Tools. To extract the end to end latency of the AR app, we modified the app to log the Unix timestamps and captured packets with timestamps via tcpdump. We also extracted low-level, signalling messages using MobileInsight [20].

4 Performance of Multi-User AR

We begin our study by comparing the E2E latency of the AR app over LTE and 5G mmWave in §4.1 and then study the individual app components in §4.2-4.5. Finally, in §4.6, we study the impact of two optimizations, which were shown in [8] to improve the latency over LTE networks. Fig. 2 plots the E2E latency as well as the latency of the individual components over 20 runs.

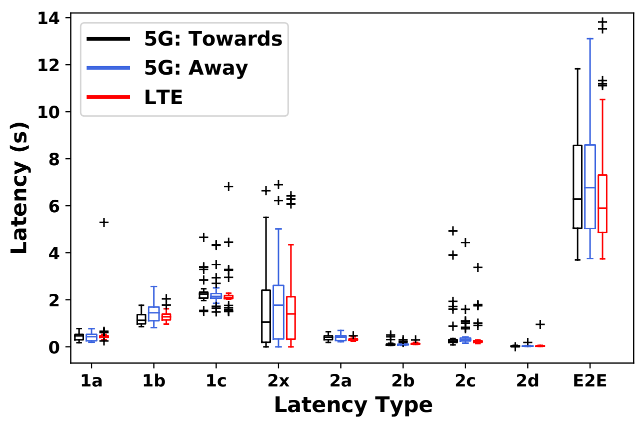

Fig. 2: Multi-user AR latency breakdown.

4.1 End to end performance

From Fig. 2, we make the following observations: (i) The E2E latency over LTE is significantly lower compared to the numbers reported in [8]. The median (maximum) latency is 5.9 s (14 s) vs. 12.5 s (27 s) in [8]). We conjecture that this reduction may be due to different levels of congestion in the LTE network and the cloud and/or technological advances in the LTE network and the UEs. However, the latency remains unacceptably high and severely impacts the user QoE. (ii) 5G mmWave, when the user faces the BS, *reduces the worst-case E2E latency* by more than 2 s. However, surprisingly, *the median latency and the 75-th percentile over 5G mmWave are higher than over LTE by 0.4 s and 1.4 s, respectively*, despite the much higher speeds and lower RTTs brought by 5G mmWave compared to LTE (Table 1). (iii) Self-blockage has minimal impact on the performance of the multi-user AR app, increasing the median E2E latency by about 0.5 s and the 75-th percentile by about 0.2 s. (v) Similar to the results in [8], the key contributors of the E2E latency are on the hosting side for both 5G mmWave and LTE. Together, the handshakes (1a), the visual data transmission (1b), and the cloud processing (1c) account for about 60% of the E2E latency over both 5G mMWave and 64% over LTE. In contrast, the resolving side components (2a - 2d) contribute together only 12% of the median E2E latency.

Overall, 5G mmWave brings practically no improvements to the performance of multi-user AR apps. In the following, we take a closer look at the latency of the individual app components and try to uncover the root causes of this surprising result and the factors that prevent 5G mmWave to unleash its potential.

4.2 Latency 2x: Resolver notification

In our experiments, we often observed a substantial delay between the last data packet sent by the cloud to the host and the moment the notification from the cloud of a new hosted object is received by the resolver. In Fig. 2, we observe that this delay, which we call 2x and was not reported in [8], varies from under 100 ms to as high as 7 s, and can be a significant contributor to the E2E latency over both LTE and 5G mmWave, accounting for about 16%, 26% and 23% of the E2E latency in the median case for 5G-towards (1.05 s), 5G-away (1.77 s) and LTE (1.4 s), respectively.

To understand the root cause of this delay, we set up a proxy between the cloud server and the resolver UE. The proxy is a server on Google Cloud, and is connected to the resolver UE through an L2TP tunnel. We synchronized the proxy and the two UEs using NTP and used tcpdump to capture and analyze packet traces on both sides. By comparing timestamps, we further broke down the 2x latency into two parts: between the last data packet sent by the cloud to the host and the moment the notification from the cloud is received by the proxy (2x_1) and between the moment the notification is received by the proxy and the moment the notification is finally received by the resolver (2x_2).

We found that 2x_1 is always short (about 100 ms), suggesting that the load on the server has minimal impact on the total 2x latency. Hence, the main contributor to the 2x latency is 2x_2 (varying from a few about 100 ms to more

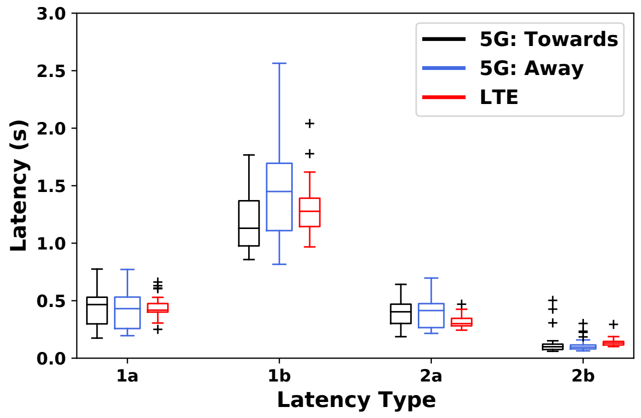

Fig. 3: Closer look at latencies 1a, 1b, 2a, 2b.

than 6 s) and the the root cause of the high 2x latency lies somewhere on the path from the proxy to the UE. We further found that every time the 2x latency was higher than a few 100s of ms, there was a TCP retransmission of the notification packet from the server. In contrast, no retransmission was observed for the cases when 2x_2 was comparable to 2x_1. Since the TCP retransmission packet was always received by the proxy within 100 ms and retransmissions over the wireless link (at the MAC and RLC layers) are unlikely to cause a delay of several seconds, we conjecture that the root cause of the high 2x latency lies in the cellular packet core network and the various middlebox (NATs, firewalls) policies implemented by the operator, which have been shown to often have a significant impact on E2E TCP performance [40].

4.3 Latency 1a and 2a: Connection handshakes

In [8], it was shown that TCP connection handshakes between the app and the cloud take 3 s on average on the hosting side (1a), contributing significantly to the E2E latency, while the handshakes and data pre-processing on the resolving side (2a) finish in less than 1 s. In contrast, Fig. 2 shows that the 1a latency in our experiments is significantly reduced over both 5G mmWave and LTE and is similar to the 2a latency (below 1 s), with the exception of 1 run over LTE that experienced a 2a latency higher than 5.5 s.

One would expect the 1a and 2a latencies to be lower over 5G mmWave compared to over LTE, as 5G mmWave has lower RTTs (Table 1). However, a closer look at these latencies (Fig. 3) shows that this is not the case. While the minimum values of 1a and 2a are indeed lower over 5G mmWave, the 75-th percentiles and maximum values are higher. Analyzing the root cause of this result is difficult, as each of these delays consists of multiple components (e.g., 1a involves tapping the screen, an optional DNS transaction, a TCP handshake with the cloud, and a TLS handshake) and the delay of each component might affect other delays. For example, we found that when the TCP handshake is preceded by a DNS transaction, the time to complete the TCP handshake is

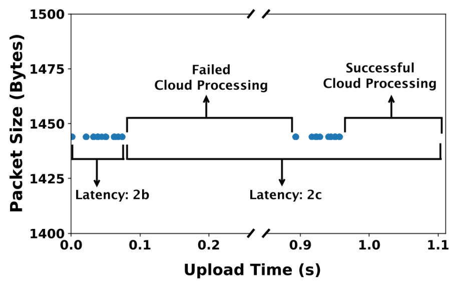

Fig. 4: Visual data re-uploaded by resolver as cloud processing failed in the first attempt.

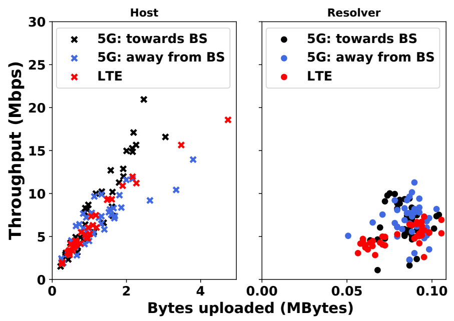

Fig. 5: Throughput as a function of uploaded bytes.

half over 5G mmWave compared to over LTE (∼20 ms vs. ∼40 ms), whereas in the absence of a DNS transaction it is higher (∼220 ms vs. ∼170 ms), possibly due to different promotion delays (e.g., the delay for the radio to switch from the idle state to to the connected state) in 5G and LTE. Since the contribution of both 1a and 2a to the E2E latency is minimal over both cellular networks, we do not further study these latencies.

4.4 Latency 1c and 2c: Cloud processing

Fig. 2 shows that the latency of the cloud processing of the host data (1c) is 2.2 s in the median case over both 5G mmWave and LTE, although there are a few outliers as high as 7.5 s, which we attribute to temporary server overloads. This value is significantly lower than the value reported in [8] (5 s), as the cloud technology has evolved over the past 2 years, but remains high, contributing about 30-35% to the E2E latency. In contrast, the cloud processing latency on the resolving side (2c) is in general negligible (200-400 ms), similar to what was reported in [8]. However, there are outlier values that can be as high as 5.5 s. We found that these outliers are due to multiple rounds of communication and processing when the cloud processing fails and requests the resolver to upload new visual data, as we explained in Fig. 1. One such example is shown in Fig. 4. After the first chunk of visual data upload (0-0.07 s), the cloud processing fails, and the resolver uploads another chunk of visual data (0.89-0.95 s), which the cloud processes successfully. We also observe that the cloud processing delay is much longer in the case of a failure (the first cloud processing delay in Fig. 4 takes 0.82 s while the second one takes only 0.16 s).

4.5 Latency 1b and 2b: Uplink data communication

In [8], the average 1b latency was found to be 10 s over a public LTE network. In contrast, our measurements in Fig. 2 show that the 1b latency over LTE is much lower, with a median value of 1.27 s and a maximum value of 2.04 s, which explain the large drop in the LTE E2E latency compared to the values reported in [8]. Hence, unlike the results reported in [8], 1b is no longer the primary contributor to the E2E latency over LTE, although its contribution still remains significant. On the other hand, the contribution of the 2b latency on the resolving side remains negligible.

Fig. 2 shows that 5G mmWave reduces both 1b and 2b latency when the user faces towards the BS, as expected due to its higher bandwidth and lower RTTs. However, the improvement is very small – 0.14 s and 0.27 s, in the median case, for 1b and 2b, respectively. Self-blockage increases the 1b latency to 1.44 s (vs. 1.13 s in the absence of blockage) in the median case and to 2.6 s (vs. 1.77 s) in the worst case, but no impact on the resolving side.

To understand why 5G mmWave has minimal impact on the uplink transmission latency in spite of the much higher uplink bandwidth compared to LTE, we show in Fig. 5 scatterplots of the uplink throughput vs. the uploaded bytes on the hosting and resolving side for each of the 40 runs. We observe that the data transfer size on both sides is very small – up to 5 MB on the hosting side and up to 0.11 MB on the resolving side, similar to the numbers reported in [8]. The small data transfer sizes explain the small improvement in the uplink data communication latency brought by 5G mmWave compared to LTE. The data uploads always finish before TCP exits the slow start phase, preventing it from taking advantage of the much higher available bandwidth offered by 5G mmWave. This is clearly shown in Fig. 5, where we observe very low throughput values (at most 22 Mbps on the host side and 12 Mbps on the resolver side), which are similar over 5G mmWave and LTE, especially for data transfer sizes up to 2 MB.

Fig. 5 further shows that facing away from the BS results in a small throughput reduction compared to facing towards the BS and to LTE for transfer sizes larger than 2 MB. This result appears to contradict recent studies reporting that the user orientation has a significant impact on 5G mmWave performance. The reason for the small observed throughput (and latency) degradation in our experiments is again the very small data transfer sizes, which prevent TCP from exiting slow start. Fig. 6, which plots the CDF of the modulation and coding scheme (MCS) values collected by MobileInsight every 5 ms, further corroborates this claim. The MCS values are much lower when the user faces away from the BS compared to when the user faces towards the BS, confirming that self-blockage deteriorates significantly the link quality. However, this large degradation of the link quality is perceived to a much lesser degree by the multi-user AR app, due to the very low application layer throughput.

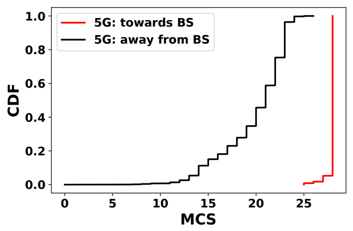

Fig. 6: 5G mmWave MCS.

4.6 AR design optimizations

In this section, we study two optimizations that were proposed in [8] to reduce the uplink data transmission latency (1b and 2b) over LTE networks.

Packet size adaptation. In scenarios when the RAN is congested, large IP packet sizes can experience heavier segmentation at the Radio Link Control

(RLC) layer, which can increase the per-packet RLC latency and subsequently, the TCP RTT, and adversely impact the growth of the TCP window during an AR upload burst. While smaller IP packets can help address this issue, they increase the network overhead. In [8], it was shown that a TCP Maximum Segment Size (MSS) of 650 bytes can increase throughput by 62% and reduce the RAN latency by 37% compared to the default MSS.

We experimented with the same three MSS values used in [8]: 400 bytes, 650 bytes, and default (1356 bytes). We conducted 10 runs with each MSS over 5G mmWave with the user facing towards the BS. The results are shown in Fig. 7.

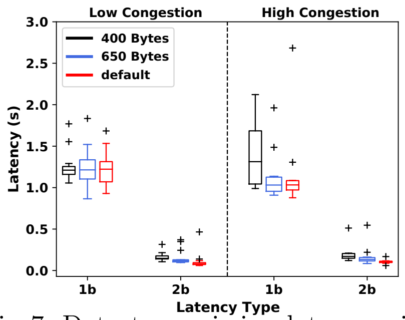

Fig. 7: Data transmission latency with varying MSS.

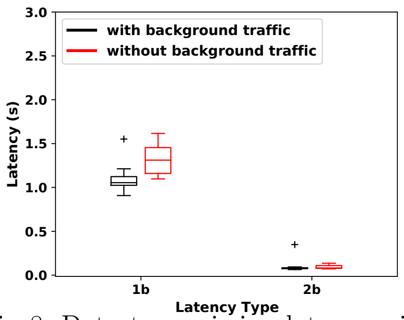

Fig. 8: Data transmission latency with background ICMP traffic.

We observe that changing the MSS has a minimal impact over of 5G mmWave networks. On the hosting side, all three MSS values result in roughly the same median latency. On the resolving side, the default MSS results in a slightly lower median latency than the other two values but in a slightly higher worst case latency. We conjecture that the minimal impact of the MSS on the 1b and 2b latency is due to the fact that 5G mmWave deployments are still at their infancy and they are unlikely to be congested, unlike LTE networks. To test the impact of this optimization in a congested network, we repeated the experiment with a third phone sending backlogged UDP traffic to a cloud server. Fig. 7 shows that in a congested network, the smallest MSS (400 bytes) results in a much higher latency, especially in the case of 1b, similar to what was observed in [8]. However, the other two MSS values still yield similar latency.

Small background traffic. Every time the application starts sending a new uplink data burst, the UE has to request resources from the BS. The BS is initially unaware of the uplink sending buffer, and may allocate a small uplink grant for the UE, causing RLC segmentation and increasing the per-packet RAN latency. In order to make the BS aware of the UE's uplink buffer during an AR session, the authors in [8] proposed to generate small amounts of background uplink traffic, using ICMP packets during an AR session. To examine the effectiveness of this optimization over 5G mmWave, we conducted 10 runs with and without ICMP traffic with the user facing towards the BS. Fig. 8 shows that this optimization is effective over 5G mmWave, bringing a significant reduction to the 1b latency on the hosting side. The median latency reduces from 1.31 s (without ICMP traffic) to 1.05 s (with ICMP traffic). The 2b latency is also reduced but the reduction is much smaller compared to the 1b latency due to the much smaller transfer sizes. However, since the contribution of the 1b and 2b latencies to the E2E latency is small (§4.5), this optimization has a small impact on the E2E latency.

Summary. Overall, in spite of the much higher bandwidth and lower RTT over LTE, 5G mmWave brings no benefits to multi-user AR apps. Although 5G mmWave brings a small reduction to the visual data transmission latency and certain optimizations that were proposed over LTE networks are equally effective over 5G networks and can further reduce this latency, there are other major contributors to the E2E AR latency (resolver notification, cloud processing), which are independent of the underlying cellular network technology. As a result, the E2E latency over 5G mmWave remains too high to enable practical user interaction in multi-user AR apps.

5 Energy Consumption

In this section, we compare the power draw and energy drain of both the host and the resolver in running the AR app over LTE and 5G mmWave.

5.1 Methodology

The AR app uses five power-hungry phone components: CPU, GPU, Camera, Display, and the cellular NIC. We use the utilization-based power models [13] for mobile devices, which have been widely used [36, 43, 42, 16, 14, 18, 27, 19, 41, 37] to model the instantaneous power draw of the CPU and the GPU. In a nutshell, such a model derives the correlation between the utilization of a phone component in each of its power states, and the resulting power draw using carefully designed micro-benchmarks. To use such a model, the CPU and GPU usage are logged during app execution using Linux event trace [21] and afterwards fed into the power model as input to estimate the per-component power draw during the app execution. For the OLED display, we used the piece-wise OLED model recently proposed in [15], which decomposes the RGB color space into $16 \times 16 \times 16$ subgrids and derives accurate pixel power model for each subgrid using liner regression to achieve low OLED power prediction error of no more than 4.6% on four recent generations of phones. We developed a program by modifying the Android "screenrecord" program to record the screen during the app execution and applied the OLED display power model from [15] for the Pixel 5 phone used in our study to estimate the OLED display power. Finally, we model the camera power draw as a constant [13]. Since the LTE/5G NIC power drain is known to be sensitive to external conditions such as the signal strength [17], practical power models based on regression on observed throughput [19, 26] are coarse-grained, and the app does not use other power-hungry phone components such as the hardware decoder, we instead directly measure the instantaneous device power using the built-in power sensor via the Linux power supply class [7] and subtract from it the power drawn by CPU, GPU, Camera, and OLED display to derive the power draw by the cellular NIC.

We adopted the above power modeling methodology for three reasons. (1) We needed to measure component-wise power draw to compute the network power draw by the AR app from the total power. SnapDragon Profiler, Monsoon Power Monitor [23], or BattOR [35] can only provide the total power consumption, as they do not provide power counters for the individual components. (2) Using power modeling eases experimentation, as attaching a Monsoon power monitor to a phone would require dismantling and instrumenting the device, which would be difficult for field experiments. (3) The power models themselves are benchmarked against the Monsoon power monitor, making them reasonably accurate. For example, we used the OLED display model from [15], which has error less than 5% for Pixel 5 on average, and the CPU and GPU models use the Linux event trace to estimate the power consumption with very high resolution.

5.2 Results

Fig. 9 shows the average (over 5 runs) energy drain by the host and the resolver during the AR app execution and the breakdown into 7 app phases, over 5G mmWave (facing towards the BS) and LTE. We make the following observations.

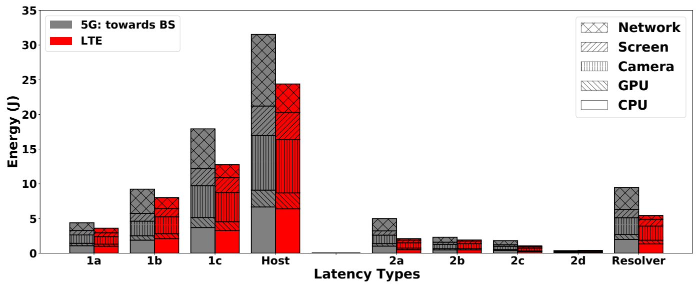

Fig. 9: Energy Breakdown

Host vs. Resolver. Overall the app drains much more total energy on the host than on the resolver, by 70% (31 J vs. 9 J) and 80% (24 J vs. 5 J), over 5G and LTE, respectively. This happens as the 3 phases 1a, 1b, and 1c on the host drain significantly higher energy than the later 4 phases (2a – 2d) on the resolver.

Total energy comparison. On the host, the app drains 23% more total energy when running over 5G compared to running over LTE. The energy breakdown by phone components shows that the major contributor to the difference is the network energy, which over 5G is 60% higher than over LTE. Similarly, on the resolver, the app drains 43% more total energy when running over 5G compared to running over LTE, although the absolute difference is much smaller compared to on the host side. The major difference again comes from the network energy drain, which over 5G is significantly higher than over LTE, by 83%.

Per-app phase comparison. To understand where the energy difference happens in the different phases of the app, we look at the energy breakdown by app phases. Fig. 9 shows that on the host side higher energy drain over 5G compared to over LTE happens in all app phases, by 18%, 14% and 29% in phases 1a, 1b, and 1c, respectively. Hence, the energy difference within each phase mainly comes from network energy, which is 41%, 56% and 68% higher under phases 1a, 1b, and 1c, respectively, over 5G than over LTE. Similarly, on the resolver side, the major contribution to the higher energy drain over 5G compared to over LTE also comes from all app phases, by 42%, 18% and 43% in phases 2a, 2b, and 2c, respectively. Again, the energy difference within each phase mainly comes from network energy, which is 88%, 77%, and 77% higher under phases 2a, 2b, and 2c, respectively, over 5G than over LTE.

Power comparison. Fig. 10 shows a timeline of the instantaneous device and network power consumption for one representative run over 5G mmWave and LTE. In each timeline, we use different colors to denote the different app phases.

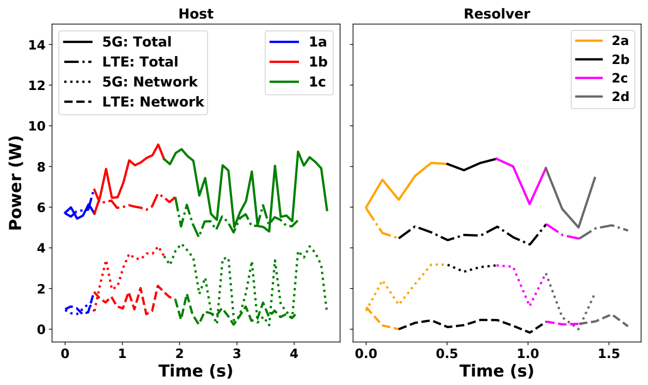

Fig. 10: Power Timeline.

We make the following observations: (1) The total instantaneous device power consumption is higher during phase 1b on the host side over both 5G mmWave and LTE and during phase 2b on the resolver side over 5G mmWave – the two phases where we primarily have network activity (the device power over LTE on the resolver side is largely constant over all four phases, due to low network activity and the low power draw of the LTE NIC). (2) The network power draw is non-zero in phases 1a, 1c, 2a, 2c, 2d, in spite of minimal network activity during those phases due to the well-known tail power state in cellular networks. (3) The network power draw fluctuates largely over time, especially over 5G mmWave. In contrast, the power for all other device components remains largely constant over time. For example, for CPU, even though the average utilization in phase 1a (578%) is higher than in phase 1c (535%), the average power difference is just 108 mW. This is because the app utilizes almost exclusively the LITTLE cores, and the difference in utilization does not get translated to significant power difference. Hence, the fluctuations of the network power contribute to the large fluctuations of the device power consumption observed in Fig. 10.

In summary, our detailed energy drain analysis shows that, as expected [26], the app drains significantly higher network energy under 5G compared to under LTE. Since the network energy accounts for about 32% of the total energy, such significant network energy difference translates into smaller difference in the total app energy drain, by 23% on the host and 43% on the resolver.

6 Conclusion

In this paper, we studied whether 5G mmWave can support multi-user AR by conducting an in-depth measurement study of a popular multi-user AR app over both LTE and 5G mmWave. Our measurements showed that, in spite of the much higher bandwidth and lower RTT, 5G mmWave results in only a small reduction to the visual data transmission latency due to the small data burst sizes, which do not allow TCP to exit slow start and take full advantage of the higher bandwidth. A potential approach to addressing this issue is to leverage TCP splitting [28], and maintain a persistent TCP connection with a very large window between an edge server and the cloud, while the UE establishes a TCP connection to the edge server. We also found that an optimization that was proposed over LTE networks can also be effective over 5G networks and can further reduce the data transmission latency. However, other major contributors to the E2E AR latency keep it in the order of several seconds, rendering user interaction practically infeasible. Since some of these factors (e.g, cloud processing) are independent of the underlying cellular network, one may have to consider more drastic changes to the design of multi-user AR apps, e.g., moving the cloud services to the edge [34] – a rapidly increasing trend among both content providers and cellular operators [11, 39] – or shifting from a client-server to a P2P paradigm [33]. Additionally, there is a need for cellular operators to revisit the middlebox policies in their packet core networks, which can also have an adverse impact on multi-user AR performance. Finally, our energy analysis showed that the app drains 66% more network energy over 5G mmWave compared to over LTE, which translates into 23% and 43% higher total energy on the host and the resolver, respectively, showing that 5G mmWave networks are not currently optimized to efficiently support this type of apps.

Acknowledgement We thank our shepherd Arani Bhattacharya and the anonymous reviewers for their helpful comments. This work was supported in part by NSF grant 2112778-CNS.

References

1. Fundamental concepts of ARCore. https://developers.google.com/ar/discover/concepts (2021)
2. Dataset: Can 5G mmWave support Multi-User AR? https://github.com/NUWiNS/pam2022-5G-mmwave-multi-user-ar-data (2022)
3. Apple ARKit: Creating a Multiuser AR Experience (Online), https://developer.apple.com/documentation/arkit/creating_a_multiuser_ar_experience
4. Google Cloud Anchor (Online), https://developers.google.com/ar/develop/java/cloud-anchors/overview-android
5. Google Firebase (Online), https://firebase.google.com/
6. Microsoft Hololens 2 (Online), https://www.microsoft.com/en-us/hololens
7. Android kernel's linux power supply class, https://android.googlesource.com/kernel/common/+/refs/heads/android-4.14-p/Documentation/power/power_supply_class.txt

8. Apicharttrisorn, K., Balasubramanian, B., Chen, J., Sivaraj, R., Tsai, Y.Z., Jana, R., Krishnamurthy, S., Tran, T., Zhou, Y.: Characterization of Multi-User Augmented Reality over Cellular Networks. In: Proc. of IEEE SECON (2020)
9. Apicharttrisorn, K., Ran, X., Chen, J., Krishnamurthy, S.V., Roy-Chowdhury, A.K.: Frugal Following: Power Thrifty Object Detection and Tracking for Mobile Augmented Reality. In: Proc. of ACM SenSys (2019)
10. Augmented and Virtual Reality: the First Wave of 5G Killer Apps: Qualcomm – ABI Research, https://gsacom.com/paper/augmented-virtual-reality-first-wave-5g-killer-apps-qualcomm-abi-research/
11. AT&T integrates 5G with Microsoft Azure to enable next-generation solutions on the edge, https://www.business.att.com/learn/top-voices/at-t-integrates-5g-with-microsoft-azure-to-enable-next-generatio.html
12. Chen, K., Li, T., Kim, H.S., Culler, D.E., Katz, R.H.: MARVEL: Enabling Mobile Augmented Reality with Low Energy and Low Latency. In: Proc. of ACM SenSys (2018)
13. Chen, X., Ding, N., Jindal, A., Hu, Y.C., Gupta, M., Vannithamby, R.: Smartphone energy drain in the wild: Analysis and implications. ACM SIGMETRICS Performance Evaluation Review **43**(1), 151–164 (2015)
14. Chen, X., Meng, J., Hu, Y.C., Gupta, M., Hasholzner, R., Ekambaram, V.N., Singh, A., Srikanteswara, S.: A fine-grained event-based modem power model for enabling in-depth modem energy drain analysis. Proceedings of the ACM on Measurement and Analysis of Computing Systems **1**(2), 1–28 (2017)
15. Dash, P., Hu, Y.C.: How much battery does dark mode save? An Accurate OLED Display Power Profiler for Modern Smartphones. In: Proc. of ACM MobiSys (2021)
16. Ding, N., Hu, Y.C.: GfxDoctor: A holistic graphics energy profiler for mobile devices. In: Proc. of ACM EuroSys (2017)
17. Ding, N., Wagner, D., Chen, X., Pathak, A., Hu, Y.C., Rice, A.: Characterizing and modeling the impact of wireless signal strength on smartphone battery drain. In: Proc. of ACM SIGMETRICS (2013)
18. Dong, M., Zhong, L.: Self-constructive high-rate system energy modeling for battery-powered mobile systems. In: Proc. of ACM MobiSys (2011)
19. Huang, J., Qian, F., Gerber, A., Mao, Z.M., Sen, S., Spatscheck, O.: A Close Examination of Performance and Power Characteristics of 4G LTE Networks. In: Proc. of ACM Mobisys (2012)
20. Li, Y., Peng, C., Yuan, Z., Li, J., Deng, H., Wang, T.: MobileInsight: Extracting and Analyzing Cellular Network Information on Smartphones. In: Proc. of ACM MobiCom (2016)
21. Linux event trace, https://www.kernel.org/doc/html/v4.18/trace/events.html
22. Liu, L., Li, H., Gruteser, M.: Edge Assisted Real-time Object Detection for Mobile Augmented Reality. In: Proc. of ACM MobiCom (2019)
23. Monsoon power monitor, https://www.msoon.com/online-store
24. Narayanan, A., Ramadan, E., Carpenter, J., Liu, Q., Liu, Y., Qian, F., Zhang, Z.L.: A First Look at Commercial 5G Performance on Smartphones. In: Proc. of ACM WWW (2020)
25. Narayanan, A., Ramadan, E., Mehta, R., Hu, X., Liu, Q., Fezeu, R.A.K., Dayalan, U.K., Verma, S., Ji, P., Li, T., Qian, F., Zhang, Z.L.: Lumos5G: Mapping and Predicting Commercial MmWave 5G Throughput. In: Proc. of ACM IMC (2020)
26. Narayanan*, A., Zhang*, X., Zhu, R., Hassan, A., Jin, S., Zhu, X., Rybkin, D., Zhang, D., Yang, M., Mao, Z.M., Qian, F., Zhang, Z.L.: A Variegated Look at 5G in the Wild: Performance, Power, and QoE Implications. In: Proc. of ACM SIGCOMM (2021)

27. Pathak, A., Hu, Y.C., Zhang, M., Bahl, P., Wang, Y.M.: Fine-grained power modeling for smartphones using system call tracing. In: Proc. of ACM EuroSys (2011)
28. Pathak, A., Wang, Y.A., Huang, C., Greenberg, A., Hu, Y.C., Kern, R., li, J., Ross, K.: Measuring and Evaluating TCP Splitting for Cloud Services. In: Proc. of PAM (2010)
29. "Pokémon Go" maker Niantic wants to turn AR into 5G's first killer app, https://www.fastcompany.com/90545662/pokemon-go-maker-niantic-wants-to-jumpstart-5g-augmented-reality
30. Qiu, H., Ahmad, F., Bai, F., Gruteser, M., Govindan, R.: AVR: Augmented Vehicular Reality. In: Proc. of ACM MobiSys (2018)
31. Ran, X., Chen, H., Zhu, X., Liu, Z., Chen, J.: DeepDecision: A Mobile Deep Learning Framework for Edge Video Analytics. In: Proc. of IEEE INFOCOM (2018)
32. Ran, X., Slocum, C., Gorlatova, M., Chen, J.: ShareAR: Communication-Efficient Multi-User Mobile Augmented Reality. In: Proceedings of ACM HotNets (2019)
33. Ran, X., Slocum, C., Tsai, Y.Z., Apicharttrisorn, K., Gorlatova, M., Chen, J.: Multi-User Augmented Reality with Communication Efficient and Spatially Consistent Virtual Objects. In: Proc. of ACM CoNEXT (2020)
34. Ren, P., Qiao, X., Huang, Y., Liu, L., Pu, C., Dustdar, S., Chen, J.L.: Edge AR X5: An Edge-Assisted Multi-User Collaborative Framework for Mobile Web Augmented Reality in 5G and Beyond. IEEE Transactions on Cloud Computing (2020)
35. Schulman, A., Schmid, T., Dutta, P., Spring, N.: Phone power monitoring with BattOr. In: Proc. of ACM MobiCom (2011)
36. Shye, A., Scholbrock, B., Memik, G.: Into the wild: studying real user activity patterns to guide power optimizations for mobile architectures. In: Proc. of IEEE/ACM MICRO (2009)
37. Sun, L., Sheshadri, R.K., Zheng, W., Koutsonikolas, D.: Modeling WiFi active power/energy consumption in smartphones. In: Proc. of IEEE ICDCS (2014)
38. Telcos seek killer app to recoup billions spent on 5G, https://www.bloomberg.com/news/articles/2021-08-10/telcos-seek-killer-app-to-recoup-billions-spent-on-5g-networks
39. Verizon teams with NFL, AWS to showcase 5G edge, https://www.fiercewireless.com/operators/verizon-teams-nfl-aws-to-showcase-5g-edgehttps://www.business.att.com/learn/top-voices/at-t-integrates-5g-with-microsoft-azure-to-enable-next-generatio.html
40. Wang, Z., Qian, Z., Xu, Q., Mao, Z.M., Zhang, M.: An Untold Story of Middleboxes in Cellular Networks. In: Proc. of ACM SIGCOMM (2011)
41. Xu, F., Liu, Y., Li, Q., Zhang, Y.: V-edge: Fast Self-constructive Power Modeling of Smartphones Based on Battery Voltage Dynamics. In: Proc. of USENIX NSDI (2013)
42. Yue, C., Sen, S., Wang, B., Qin, Y., Qian, F.: Energy considerations for ABR video streaming to smartphones: measurements, models and insights. In: Proc. of ACM Multimedia Systems (2020)
43. Zhang, L., Tiwana, B., Qian, Z., Wang, Z., Dick, R.P., Mao, Z.M., Yang, L.: Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In: Proc. of IEEE/ACM/IFIP CODES+ISSS (2010)
44. Zhang, W., Han, B., Hui, P., Gopalakrishnan, V., Zavesky, E., Qian, F.: CARS: Collaborative Augmented Reality for Socialization. In: Proc. of ACM HotMobile (2018)